

TCP/IP チュートリアル および技術解説書

TCP/IP プロトコル・スイートの
ネットワーキングの基本について

QoS やセキュリティなどの
拡張概念

最新の TCP/IP プロト
コル



Adolfo Rodriguez
John Gatrell
John Karas
Roland Peschke



International Technical Support Organization

TCP/IP チュートリアルおよび技術解説書

ご注意！

本書および本書で紹介する製品をご使用になる前に、947 ページの付録 B『特記事項』に記載されている情報をお読みください。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

<http://www.ibm.com/jp/manuals/main/mail.html>

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは <http://www.ibm.com/jp/manuals/> の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

原典：	GG24-3376-06 International Technical Support Organization TCP/IP Tutorial and Technical Overview
発行：	日本アイ・ビー・エム株式会社
担当：	ナショナル・ランゲージ・サポート

第 2 刷 2003. 5

目次

まえがき	xix
このレッドブックを作成したチーム	xix

第 1 部 中核となる TCP/IP プロトコル 1

第 1 章 アーキテクチャー、ヒストリー、標準、および傾向 3

1.1 TCP/IP アーキテクチャー・モデル	3
1.1.1 インターネットワーキング	3
1.1.2 TCP/IP プロトコル層	5
1.1.3 TCP/IP アプリケーション	8
1.1.4 ブリッジ、ルーター、およびゲートウェイ	10
1.2 インターネットのルーツ	12
1.2.1 ARPANET	14
1.2.2 NSFNET	15
1.2.3 インターネットの営利使用	17
1.2.4 Internet2	18
1.2.5 Open Systems Interconnection (OSI) 参照モデル	20
1.3 TCP/IP 標準	21
1.3.1 Request For Comments (RFC)	23
1.3.2 インターネット標準	25
1.4 インターネットの将来	27
1.4.1 マルチメディア・アプリケーション	27
1.4.2 営利的な使用	27
1.4.3 ワイヤレス・インターネット	28

第 2 章 ネットワーク・インターフェース 29

2.1 イーサネットおよび IEEE 802.x ローカル・エリア・ネットワーク (LAN)	29
2.2 Fiber Distributed Data Interface (FDDI)	32
2.3 Serial Line IP (SLIP)	34
2.4 Point-to-Point Protocol (PPP)	35
2.4.1 Point-to-Point カプセル化	36
2.5 サービス総合ディジタル網 (ISDN)	37
2.6 X.25	38
2.7 フレーム・リレー	40
2.7.1 フレーム・フォーマット	41
2.7.2 相互接続の問題	42
2.7.3 データ・リンク層パラメーターのネゴシエーション	42
2.7.4 フレーム・リレーにおける IP	43
2.8 SONET および SDH 回路における PPP	44

2.8.1 物理層	44
2.9 Multi-Path Channel+ (MPC+)	45
2.10 非同期伝送モード (ATM)	45
2.10.1 アドレス解決 (ATMARP および InATMARP)	46
2.10.2 Classical IP over ATM	49
2.10.3 ATM LAN エミュレーション	56
2.10.4 Classical IP over ATM 対 LAN エミュレーション	59
2.11 Multiprotocol over ATM (MPOA)	60
2.11.1 MPOA の利点	61
2.11.2 MPOA 論理コンポーネント	61
2.11.3 MPOA 機能コンポーネント	62
2.11.4 MPOA 操作	65
2.12 リファレンス	66
第3章 インターネットワーキング・プロトコル	67
3.1 インターネット・プロトコル (IP)	67
3.1.1 IP アドレッシング	68
3.1.2 IP サブネット	71
3.1.3 IP ルーティング	77
3.1.4 配送方式 - ユニキャスト、ブロードキャスト、マルチキャスト、 およびエニーキャスト	83
3.1.5 IP アドレス枯渇問題	85
3.1.6 イントラネット - 専用 IP アドレス	88
3.1.7 クラスレス・ドメイン間ルーティング (CIDR)	89
3.1.8 IP データグラム	92
3.2 Internet Control Message Protocol (ICMP)	105
3.2.1 ICMP メッセージ	106
3.2.2 ICMP アプリケーション	116
3.3 Internet Group Management Protocol (IGMP)	117
3.4 Address Resolution Protocol (ARP)	118
3.4.1 ARP の概要	118
3.4.2 ARP の詳細概念	119
3.4.3 ARP とサブネット	122
3.4.4 プロキシ ARP または透過的なサブネット化	122
3.5 Reverse Address Resolution Protocol (RARP)	124
3.5.1 RARP の概念	124
3.6 Bootstrap protocol (BOOTP)	125
3.6.1 BOOTP 転送	129
3.6.2 BOOTP の考慮事項	130
3.7 Dynamic Host Configuration Protocol (DHCP)	131
3.7.1 DHCP メッセージ・フォーマット	132
3.7.2 DHCP メッセージ・タイプ	134
3.7.3 新規ネットワーク・アドレスの割り振り	135

3.7.4	DHCP リース更新プロセス	137
3.7.5	以前の割り振られたネットワーク・アドレスの再利用	139
3.7.6	構成パラメーター・リポジトリ	140
3.7.7	DHCP の考慮事項	140
3.7.8	BOOTP と DHCP のインターオペラビリティ	141
第 4 章	ルーティング・プロトコル	143
4.1	自律システム (Autonomous System)	144
4.2	IP ルーティングと IP ルーティング・アルゴリズムのタイプ	146
4.2.1	静的ルーティング	147
4.2.2	距離ベクトル・ルーティング (Distance vector routing)	148
4.2.3	リンク状態ルーティング	149
4.2.4	ハイブリッド・ルーティング	150
4.3	Routing Information Protocol (RIP)	151
4.3.1	RIP のパケット・タイプ	151
4.3.2	RIP のパケット・フォーマット	152
4.3.3	RIP の操作モード	152
4.3.4	距離ベクトルの計算	153
4.3.5	収束と無限カウント	154
4.3.6	RIP の制約事項	159
4.4	Routing Information Protocol バージョン 2 (RIP-2)	159
4.4.1	RIP-2 のパケット・フォーマット	160
4.4.2	RIP-2 の制約事項	162
4.5	IPv6 用 RIPng	162
4.5.1	RIPng と RIP-2 の違い	162
4.5.2	RIPng のパケット・フォーマット	163
4.6	Open Shortest Path First (OSPF)	165
4.6.1	OSPF 用語	166
4.6.2	OSPF のパケット・タイプ	173
4.6.3	近隣者通信	174
4.6.4	OSPF 近隣状態マシン	176
4.6.5	OSPF の仮想リンクと中継エリア	177
4.6.6	OSPF の経路再配布	178
4.6.7	OSPF のスタブ・エリア	180
4.6.8	OSPF の経路要約	181
4.7	Enhanced Interior Gateway Routing Protocol (EIGRP)	182
4.7.1	EIGRP のフィーチャー	183
4.7.2	用語	183
4.7.3	近隣者の発見とリカバリー	185
4.7.4	DUAL アルゴリズム	186
4.7.5	EIGRP のパケット・タイプ	188
4.8	Exterior Gateway Protocol (EGP)	189
4.9	Border Gateway Protocol (BGP)	189

4.9.1 BGP の概念と用語	190
4.9.2 IBGP と EBGP の通信	192
4.9.3 プロトコルの説明	194
4.9.4 パスの選択	197
4.9.5 BGP 同期	200
4.9.6 BGP 集約	202
4.9.7 BGP 同盟	203
4.9.8 BGP 経路リフレクター	205
4.10 ルーティング・プロトコルの選択	206
4.11 ルーターが実行する追加機能	207
4.12 UNIX ベース・プロセスでのルーティング・プロセス	208
第 5 章 トランスポート層プロトコル	209
5.1 ポートとソケット	209
5.1.1 ポート	209
5.1.2 ソケット	211
5.2 User Datagram Protocol (UDP)	212
5.2.1 UDP データグラム・フォーマット	213
5.2.2 UDP アプリケーション・プログラミング・インターフェース	214
5.3 Transmission Control Protocol (TCP)	215
5.3.1 TCP の概念	216
5.3.2 TCP アプリケーション・プログラミング・インターフェース	229
5.3.3 TCP 輻輳制御アルゴリズム	230
第 6 章 IP マルチキャスト	237
6.1 マルチキャスト・アドレッシング	237
6.1.1 単一の物理ネットワーク上のマルチキャスト	238
6.1.2 ネットワーク・セグメント間のマルチキャスト	239
6.2 Internet Group Management Protocol (IGMP)	241
6.2.1 IGMP メッセージ	241
6.2.2 IGMP での操作	243
6.3 マルチキャスト送達ツリー	244
6.4 マルチキャスト転送アルゴリズム	246
6.4.1 逆パス転送アルゴリズム	246
6.4.2 中心基点ツリー・アルゴリズム	247
6.4.3 マルチキャスト・ルーティング・プロトコル	248
6.5 Distance Vector Multicast Routing Protocol (DVMRP)	248
6.5.1 プロトコルの概要	248
6.5.2 マルチキャスト送達ツリーの構築と保守	250
6.5.3 DVMRP でのトンネル	252
6.6 Multicast OSPF (MOSPF)	253
6.6.1 プロトコルの概要	253
6.6.2 MOSPF と複数の OSPF エリア	254

6.6.3	MOSPF と複数の自律システム	255
6.6.4	MOSPF のインターオペラビリティ	256
6.7	Protocol Independent Multicast (PIM)	256
6.7.1	PIM 高密度モード	256
6.7.2	PIM 希薄モード	258
6.8	マルチキャスト・ドメインの相互接続	261
6.8.1	Multicast Source Discovery Protocol (MSDP)	261
6.8.2	Border Gateway Multicast Protocol	264
6.9	マルチキャスト・バックボーン	265
6.9.1	MBONE でのルーティング	265
6.9.2	マルチキャスト・アプリケーション	268

第 2 部 TCP/IP アプリケーション・プロトコル 269

第 7 章	アプリケーション構造およびプログラミング・インターフェース	271
7.1	アプリケーションの特性	271
7.1.1	クライアント / サーバー・モデル	271
7.2	アプリケーション・プログラミング・インターフェース (API)	272
7.2.1	ソケット API	273
7.2.2	リモート・プロシージャ・コール (RPC)	278
7.2.3	Windows Sockets Version 2 (Winsock V2.0)	282
7.2.4	SNMP 分散プログラミング・インターフェース (SNMP DPI)	284
7.2.5	FTP API	287
7.2.6	CICS ソケット・インターフェース	288
7.2.7	IMS ソケット・インターフェース	288
7.2.8	Sockets Extended	289
7.2.9	REXX ソケット	289
第 8 章	ディレクトリーおよびネーミングのプロトコル	291
8.1	Domain Name System (DNS)	291
8.1.1	階層ネーム・スペース	292
8.1.2	完全修飾ドメイン・ネーム (FQDN)	293
8.1.3	一般ドメイン	294
8.1.4	国別ドメイン	294
8.1.5	ドメイン・ネームから IP アドレスへのマッピング	295
8.1.6	IP アドレスからドメイン・ネームへのマッピング ポインター照会	295
8.1.7	分散ネーム・スペース	295
8.1.8	ドメイン・ネーム解決	297
8.1.9	Domain Name System リソース・レコード	301
8.1.10	Domain Name System のメッセージ	303
8.1.11	単純シナリオ	308
8.1.12	拡張シナリオ	310

8.1.13	トランスポート	311
8.1.14	DNS アプリケーション	312
8.1.15	参照	313
8.2	Dynamic Domain Name System	313
8.2.1	UPDATE DNS メッセージ・フォーマット	315
8.2.2	IBM での DDNS のインプリメンテーション	317
8.2.3	プロキシ A レコード更新 (ProxyArec)	327
8.3	Network Information System (NIS)	329
8.4	Lightweight Directory Access Protocol (LDAP)	330
8.4.1	LDAP - X.500 への軽快なアクセス	331
8.4.2	LDAP ディレクトリー・サーバー	333
8.4.3	LDAP アーキテクチャーの概要	335
8.4.4	LDAP のモデル	336
8.4.5	LDAP のセキュリティー	343
8.4.6	LDAP URL	346
8.4.7	LDAP と DCE	346
8.4.8	directory-enabled networks initiative (DEN)	349
8.4.9	Web-Based Enterprise Management (WBEM)	349
8.4.10	参照	349
第 9 章	リモート実行と分散コンピューティング	351
9.1	TELNET	351
9.1.1	TELNET 操作	352
9.1.2	端末エミュレーション (Telnet 3270)	359
9.1.3	TN3270 拡張 (TN3270E)	360
9.1.4	参照	362
9.2	リモート実行コマンド・プロトコル (REXEC と RSH)	363
9.2.1	操作の原則	363
9.3	分散コンピューティング環境 (DCE) の概要	364
9.3.1	DCE ディレクトリー・サービス	365
9.3.2	DCE セキュリティー・サービス	368
9.3.3	DCE スレッド	374
9.3.4	DCE リモート・プロシージャ・コール	375
9.3.5	Distributed Time Service	376
9.3.6	分散ファイル・サービス (DFS)	378
9.3.7	参照	381
第 10 章	ファイル関連のプロトコル	383
10.1	ファイル転送プロトコル (FTP)	383
10.1.1	FTP の概要	384
10.1.2	FTP での操作	384
10.1.3	応答コード	387
10.1.4	FTP のシナリオ	388

10.1.5	FTP サンプル・セッション	390
10.1.6	アノニマス FTP	390
10.1.7	FTP を使用したリモート・ジョブ入力	390
10.2	Trivial File Transfer Protocol (TFTP)	391
10.2.1	TFTP の使用法	391
10.2.2	プロトコルの説明	392
10.2.3	TFTP マルチキャスト・オプション	394
10.2.4	セキュリティーの問題	394
10.3	Network File System (NFS)	394
10.3.1	NFS の概念	395
10.3.2	NFS バージョン 4	401
10.3.3	WebNFS	401
10.3.4	参照	402
10.4	Andrew File System (AFS)	402
第 11 章	メール・アプリケーション	405
11.1	Simple Mail Transfer Protocol (SMTP)	405
11.1.1	SMTP の動作	407
11.1.2	SMTP と Domain Name System	414
11.1.3	参照	416
11.2	Multipurpose Internet Mail Extensions (MIME)	417
11.2.1	MIME の動作	421
11.2.2	Content-Type フィールド	422
11.2.3	Content-Transfer-Encoding フィールド	429
11.2.4	メッセージ・ヘッダーでの非 ASCII 文字の使用	435
11.2.5	参照	437
11.3	Post Office Protocol (POP)	437
11.3.1	POP3 のコマンドと応答	438
11.3.2	参照	439
11.4	Internet Message Access Protocol バージョン 4 (IMAP4)	439
11.4.1	IMAP4 の基礎をなす電子メール・モデル	440
11.4.2	IMAP4 のコマンドと応答	441
11.4.3	メッセージ番号	441
11.4.4	IMAP4 の状態	443
11.4.5	クライアント・コマンド	444
11.4.6	参照	446
第 12 章	ワールド・ワイド・ウェブ (WWW)	447
12.1	Web ブラウザー	447
12.2	Web サーバー	449
12.3	Hypertext Transfer Protocol (HTTP)	450
12.3.1	HTTP の概要	450
12.3.2	HTTP の操作	451

12.4	コンテンツ	460
12.4.1	静的コンテンツ	461
12.4.2	クライアント・サイドの動的コンテンツ	462
12.4.3	サーバー・サイドの動的コンテンツ	462
12.4.4	オブジェクト	464
12.4.5	IBM Web アプリケーション・サーバーでのコンテンツの開発	467
12.5	参照	468
第 13 章	マルチメディア・プロトコル	469
13.1	リアルタイム・プロトコル: RTP および RTCP	469
13.1.1	Real-Time Transport Protocol (RTP)	470
13.1.2	Real-Time Control Protocol	475
13.1.3	RTCP パケット形式	477
13.1.4	RTP 変換プログラムおよびミキサー	479
13.1.5	リアルタイム・アプリケーション	482
13.2	IP テレフォニー	483
13.2.1	概要	484
13.2.2	IP テレフォニー・プロトコル・スタック	485
13.2.3	ITU-T 勧告 H.323	486
13.2.4	Session Initiation Protocol (SIP)	490
13.2.5	Media Gateway Control Protocol (MGCP)	493
13.2.6	Media Gateway Controller (Megaco)	494
13.2.7	シグナル・プロトコル機能の比較	495
13.2.8	音声によるエンコードおよび圧縮	496
第 14 章	Wireless Application Protocol (WAP)	499
14.1	WAP 環境	499
14.2	WAP 仕様の主要要素	500
14.2.1	WAP プログラミング・モデルの概要	501
14.2.2	WAP ネットワーク構成	503
14.3	WirelessMarkup Language (WML) および WMLScript	506
14.3.1	WML	506
14.3.2	WMLScript	509
14.4	プッシュ・アーキテクチャー	509
14.4.1	プッシュ・フレームワーク	510
14.4.2	プッシュ・プロキシ・ゲートウェイ (PPG)	511
14.4.3	Push access control プロトコル (PAP)	512
14.4.4	サービス指示	513
14.4.5	over-the-air プロトコル (OTA) のプッシュ	514
14.4.6	クライアント・サイド・インフラストラクチャー	514
14.4.7	セキュリティ	515
14.5	WAP プロトコル・スタックの概要	516
14.5.1	ワイヤレス・アプリケーション環境 (WAE)	516

14.5.2	Wireless Telephony Application (WTA)	518
14.5.3	Wireless Session プロトコル (WSP)	518
14.5.4	WirelessTransaction プロトコル (WTP)	531
14.5.5	ワイヤレス・トランスポート層セキュリティー (WTLS)	534
14.5.6	WirelessDatagram プロトコル (WDP)	539
14.6	プロトコルの要約	541
第 15 章	ネットワーク管理	545
15.1	Simple Network Management Protocol および MIB の概要	545
15.2	管理情報の構造と識別 (SMI)	546
15.3	Management Information Base (MIB)	548
15.3.1	IBM に特定の MIB 部分	551
15.4	Simple Network Management Protocol (SNMP)	552
15.5	Simple Network Management Protocol バージョン 2 (SNMPv2)	556
15.5.1	SNMPv2 エンティティ	557
15.5.2	SNMPv2 パーティ	557
15.5.3	GetBulkRequest	558
15.5.4	InformRequest	560
15.6	SNMPv2 の MIB	560
15.7	新規の管理モデル	561
15.8	Simple Network Management Protocol バージョン 3 (SNMPv3)	563
15.8.1	単一認証およびプライバシー・プロトコル	564
15.9	参照資料	565
第 16 章	ユーティリティー	567
16.1	リモート印刷 (LPR および LPD)	567
16.2	X Window システム	567
16.2.1	機能上の概念	568
16.2.2	プロトコル	573
16.3	Network News Transfer Protocol (NNTP)	574
16.4	Finger プロトコル	574
16.5	Netstat	575

第 3 部 拡張概念および新規テクノロジー 577

第 17 章	IP Version 6	579
17.1	IPv6 の概要	580
17.2	IPv6 ヘッダー・フォーマット	581
17.2.1	パケット・サイズ	585
17.2.2	拡張ヘッダー	585
17.2.3	IPv6 アドレッシング	592
17.2.4	トラフィック・クラス	599
17.2.5	フロー・ラベル	599

17.3 Internet Control Message Protocol Version 6 (ICMPv6)	599
17.3.1 近隣ディスカバリー	601
17.3.2 ステートレス・アドレス自動構成	613
17.3.3 マルチキャスト・リスナー・ディスカバリー (MLD)	614
17.4 IPv6 における DNS	617
17.4.1 IPv6 リソース・レコードのフォーマット	618
17.5 IPv6 での DHCP	621
17.5.1 DHCPv6 と DHCPv4 との間の相違点	622
17.5.2 DHCPv6 メッセージ	622
17.6 IPv6 でのモバイル・サポート	624
17.7 インターネットでの移行 - IPv4 から IPv6 へのマイグレーション ..	624
17.7.1 二重 IP スタックの実装 - IPv6/IPv4 ノード	625
17.7.2 トンネリング	626
17.7.3 ヘッダーの変換	633
17.7.4 インターオペラビリティの要約	633
17.8 IPv6 への推進	634
17.9 リファレンス	635
 第 18 章 Multiprotocol Label Switching (MPLS)	637
18.1 MPLS の概要	637
18.1.1 従来型のルーティング・モデル	637
18.1.2 MPLS 転送モデル	637
18.1.3 その他の利点	638
18.2 MPLS ネットワークのコンポーネント	640
18.2.1 用語	640
18.2.2 ラベル・スワッピング	643
18.2.3 ラベル交換パス (LSP)	645
18.2.4 ラベル・スタックおよびラベル階層	645
18.2.5 BGP 環境での MPLS スタック	647
18.3 ラベル配布プロトコル	649
18.3.1 ラベル分散プロトコルのタイプ	649
18.3.2 ラベル配布メソッド	650
18.4 ストリーム・マージ	650
18.4.1 フレーム・ベースの環境でのマージ	650
18.4.2 ATM 環境でのマージ	651
18.5 マルチプロトコル・ラムダ・スイッチング	652
 第 19 章 モバイル IP	655
19.1 モバイル IP の概要	655
19.2 モバイル IP の操作	656
19.3 モビリティ・エージェント公示の拡張子	658
19.4 モバイル IP の登録プロセス	660
19.5 トンネリング	664

19.6	ブロードキャスト・データグラム	664
19.7	移動の検出	665
19.7.1	ホームに戻る	665
19.8	ARP に関する考慮事項	665
19.9	モバイル IP セキュリティーに関する考慮事項	666
第 20 章	TCP/IP と他のプロトコルの統合	667
20.1	Enterprise Extender	667
20.1.1	パフォーマンスとリカバリー	668
20.2	データ・リンク・スイッチ	668
20.2.1	概要	668
20.2.2	機能の説明	669
20.3	マルチプロトコル・トランスポート・ネットワーク (MPTN)	671
20.3.1	混合プロトコル・ネットワーキングの要件	671
20.3.2	MPTN のアーキテクチャー	672
20.3.3	MPTN の方法論	672
20.3.4	MPTN の主要コンポーネント	673
20.4	TCP/IP での NetBIOS	675
20.4.1	NetBIOS ネーム・サーバー (NBNS) の インプリメンテーション	678
第 21 章	TCP/IP セキュリティー	681
21.1	機密漏れとソリューション	681
21.1.1	セキュリティに対する一般的な攻撃	681
21.1.2	ネットワーク・セキュリティ問題のソリューション	682
21.1.3	セキュリティ・ソリューションの実装	684
21.1.4	ネットワーク・セキュリティ・ポリシー	685
21.2	暗号化についての簡単な概要	686
21.2.1	用語	686
21.2.2	対称または秘密鍵アルゴリズム	688
21.2.3	非対称または公開鍵アルゴリズム	690
21.2.4	ハッシュ機能	695
21.2.5	デジタル証明書と認証局	701
21.2.6	乱数生成プログラム	702
21.2.7	暗号化の輸出 / 輸入制限	703
21.3	ファイアウォール	704
21.3.1	ファイアウォールの概念	705
21.3.2	ファイアウォール・システムのコンポーネント	706
21.3.3	パケット・フィルタ操作ルーター	707
21.3.4	アプリケーション・レベル・ゲートウェイ (プロキシ)	709
21.3.5	回線レベル・ゲートウェイ	714
21.3.6	ファイアウォールのタイプ	715
21.4	ネットワーク・アドレス変換 (NAT)	720

21.4.1 NAT の概念	721
21.4.2 変換メカニズム	722
21.4.3 NAT の制限	724
21.5 IP Security Architecture (IPsec)	725
21.5.1 概念	726
21.5.2 認証ヘッダー (AH)	729
21.5.3 暗号ペイロード (ESP)	735
21.5.4 IPsec プロトコルの組み合わせ	743
21.5.5 Internet Key Exchange (IKE) プロトコル	749
21.6 SOCKS	769
21.6.1 SOCKS バージョン 5 (SOCKSv5)	771
21.7 Secure Shell (I)	776
21.7.1 SSH の概要	777
21.8 Secure Sockets Layer (SSL)	777
21.8.1 SSL の概要	777
21.8.2 SSL プロトコル	780
21.9 Transport Layer Security (TLS)	786
21.10 Secure Multipurpose Internet Mail Extension (S-MIME)	786
21.11 仮想プライベート・ネットワーク (VPN) の概要	786
21.11.1 VPN の紹介および VPN の利点	787
21.12 Kerberos 認証および許可システム	788
21.12.1 前提事項	789
21.12.2 命名	789
21.12.3 Kerberos 認証プロセス	790
21.12.4 Kerberos データベース管理	795
21.12.5 Kerberos 許可モデル	795
21.12.6 Kerberos バージョン 5 の機能強化	796
21.13 リモート・アクセス認証プロトコル	796
21.14 Layer 2 Tunneling Protocol (L2TP)	799
21.14.1 用語	800
21.14.2 プロトコルの概要	801
21.14.3 L2TP セキュリティー問題	803
21.15 Secure Electronic Transactions (SET)	804
21.15.1 SET の役割	804
21.15.2 SET 取引	805
21.15.3 SET 証明書の構造	808
21.16 参照資料	809
第 22 章 サービス品質	811
22.1 QoS が必要な理由	811
22.2 統合サービス	812
22.2.1 サービス・クラス	816
22.2.2 Resource Reservation Protocol (RSVP)	820

22.2.3 統合サービスの見通し	833
22.3 DiffServ	834
22.3.1 DiffServ アーキテクチャー	836
22.3.2 DiffServ ネットワーク上での統合サービス (Intserv)	845
22.3.3 LDAP を使用した DS の構成と管理	848
22.3.4 DiffServ と IPSec の使用	849
22.4 参照情報	850
第 23 章 可用性、拡張容易性、および負荷平準化	853
23.1 可用性	853
23.2 拡張容易性	855
23.3 負荷平準化	855
23.4 この章で使用されている用語	856
23.4.1 シスプレックス	856
23.4.2 Workload Manager (WLM)	857
23.4.3 仮想 IP アドレス (VIPA)	857
23.4.4 動的 XCF	858
23.4.5 シスプレックス中の動的 IP アドレッシング	859
23.4.6 DVIPA アドレスの引き継ぎ / 取り消し	860
23.5 使用可能なソリューションの紹介	863
23.6 Network Dispatcher	864
23.6.1 Network Dispatcher のコンポーネント	864
23.6.2 負荷平準化と重み	868
23.6.3 高可用性	869
23.6.4 サーバーの類縁性	871
23.6.5 規則ベースの平衡化	871
23.6.6 広域 Network Dispatcher	872
23.6.7 ISS と Dispatcher の結合	873
23.6.8 Advisor およびカスタム・アドバイザー	874
23.6.9 SNMP サポート	874
23.6.10 場所共有オプション	874
23.6.11 ISP の構成	875
23.6.12 OS/390 の並列シスプレックスのサポート	876
23.7 Cisco LocalDirector	878
23.7.1 概説	878
23.7.2 接続とデータグラムのフロー	879
23.8 IBM Sysplex Distributor	880
23.8.1 Sysplex Distributor のエレメント	880
23.8.2 Sysplex Distributor の初期設定および引き継ぎ / 取り消し	881
23.8.3 Sysplex Distributor の負荷平準化の規則	882
23.8.4 接続要求の処理	882
23.8.5 接続確立後のデータ・パス	883
23.8.6 引き継ぎ / 取り消し	883

23.8.7 可用性、拡張容易性、および負荷平準化の達成	885
23.9 Cisco MultiNode Load Balancing (MNLB)	886
23.9.1 MultiNode Load Balancing 機能の概要	887
23.9.2 接続とその後のデータ・フローの確立	888
23.9.3 クライアント / サーバー接続の再始動	891
23.9.4 可用性、拡張容易性、および負荷平準化の達成	891
23.10 IBM Sysplex Distributor および Cisco MNLB	893
23.10.1 何を意味するか	894
23.10.2 IBM Sysplex Distributor と Service Manager の概要	895
23.10.3 Cisco Forwarding Agent: 概要と機能	896
23.10.4 Cisco Workload Agent	896
23.10.5 接続確立プロセス	896
23.10.6 スタック、サーバー、または LPAR の障害	899
23.10.7 Sysplex Distributor の障害	900
23.10.8 パケットのルーティング	900
23.10.9 MNLB コンポーネントの追加タスク	901
23.11 OS/390 DNS/WLM	903
23.11.1 シスプレックス環境中の DNS	903
23.11.2 DNS/WLM とリモート・ネーム・サーバー	907
23.12 Virtual Router Redundancy Protocol (VRRP)	909
23.12.1 概要	909
23.12.2 VRRP の定義	910
23.12.3 VRRP の概要	911
23.12.4 構成の例	912
23.12.5 VRRP パケットの形式	913
23.13 ラウンドロビン DNS	916
23.14 負荷平準化に関する代替ソリューション	917
23.14.1 ネットワーク・アドレス変換	917
23.14.2 カプセル化	920
付録 A プラットフォームの実装	921
A.1 IBM Communications Server for OS/390 V2R10	921
A.1.1 サポートされている接続プロトコルと装置	921
A.1.2 サポートされているサポート・アプリケーション	924
A.1.3 Enterprise Extender	925
A.1.4 仮想 IP アドレッシング (VIPA)	925
A.1.5 Sysplex Distributor	926
A.1.6 サービス品質 (QoS)	928
A.2 IBM OS/400 V5R1	929
A.2.1 GUI 構成サポート	930
A.2.2 TCP/IP Connectivity Utilities for IBM iSeries	930
A.2.3 動的 IP ルーティング (RIP および RIP2)	931
A.2.4 拡張機能	931

A.2.5 Proxy Address Resolution Protocol (Proxy ARP)	933
A.2.6 Point-to-Point Protocol (PPP)	933
A.2.7 セキュリティー機能	933
A.2.8 仮想 IP アドレッシング (VIPA)	935
A.2.9 アプリケーション・プログラミング・インターフェース (API) ..	936
A.2.10 サポートされているアプリケーション	938
A.3 Linux	942
A.3.1 Linux ファイアウォール	943
A.4 ネットワーク・コンピューター	943
付録 B 特記事項	947
付録 C 関連資料	951
C.1 IBM のレッドブック	951
C.2 IBM レッドブック・コレクション	952
C.3 その他のリソース	952
C.4 参照 Web サイト	954
省略語と頭字語	955
索引	965

まえがき

TCP/IP プロトコル・スイートは、今日のネットワークで結ばれた世界におけるコンピューター通信のデファクト・スタンダードになっています。特定のネットワーク標準のユビキタス・インプリメンテーションによって使用可能になるアプリケーションに非常に依存した状態が生じることになりました。今日では、TCP/IP プロトコルおよび Internet は、エンターテインメントや情報のためだけでなく、取引を実行し、商品を売買し、顧客にサービスを提供することによるビジネスの運営にも使用されています。TCP/IP を活用する一連のアプリケーションは引き続き増加しており、それによってインフラストラクチャー面でさらなるサポートの必要性が高まりつつあります。

この TCP/IP チュートリアルおよび技術解説書では、TCP/IP プロトコル・スイートを詳細に考察します。第 1 部では、TCP/IP について紹介し、このプロトコルに不可欠な基礎概念についての基本的な理解を得られるようにします。続いて第 2 部では、新生のワイヤレスおよびマルチメディア・アプリケーションを含め、今日の最も一般的な TCP/IP アプリケーション・プロトコルについて概説します。

最後に第 3 部では、IPv6、セキュリティー、Quality of Service、IP モビリティ、および MPLS を含めた、ネットワークにおける拡張概念と最近のインフラストラクチャーの傾向について説明します。TCP/IP が現在直面している課題と、それを克服するために開発中のテクノロジーについても説明します。

このレッドブックを作成したチーム

このレッドブックは、International Technical Support Organization Raleigh Center で働く世界各地の専門家のチームによって作成されました。

Adolfo Rodriguez は、International Technical Support Organization の Raleigh Center の顧問 I/T スペシャリストです。彼は TCP/IP のあらゆる分野について、数多く執筆し、世界中の IBM クラスで教えています。ITSO に加わる前、Adolfo は RTP, NC で Communications Server for OS/390 の設計と開発に携わりました。Duke University, Durham, NC で、数学の学士号と情報科学の理学士および理学修士号を取得しています。現在でも Duke University でネットワーク・システムを専攻しながら、情報科学の博士号の取得を目指しています。

John Gatrell は、英国の IBM で勤務しています。彼は通信カスタマー・サポートにおける 15 年の経験と、プログラミングにおけるさらに 7 年の経験を有しています。Oxford University で物理学の名誉文学士号を取得しています。専門分野には、UNIX と通信が含まれます。

John Karas は、米国の IBM Global Services のネットワーク設計者です。彼はデータ・ネットワーキングの分野における 14 年の経験を有しています。Pace University で、通信学の理学修士号を取得しています。専門分野には、IP ルーティング・アルゴリズム、複合ネットワーク設計、キャパシティー・プランニング、およびアプリケーション・パフォーマンス検査が含まれます。OSPF および BGP ネットワークのサポート、および SAP 環境でのパフォーマンス・モニターについて数多く執筆しています。

Roland Peschke は、OS/390 TCP/IP および SNA 環境のコンサルティングおよび教育サービスを必要とする IBM のカスタマーのために働く、顧問 IT ネットワーキング・スペシャリストです。彼のこの分野での経験の広さは、30 年を超える期間、IBM Germany および ITSO Raleigh で勤務していたことに由来します。何冊かの SNA および TCP/IP のレッドブックの作成に精力的に携わりました。

このプロジェクトに対する計り知れないご協力に対し、以下の方々に感謝いたします。

International Technical Support Organization (ITSO) Raleigh Center

Gail Christensen、Margaret Ticknor、Jeanne Tucker、David Watts、Juan Rodriguez、Byron Braswell、Thomas Barlen、Linda Robinson

International Technical Support Organization (ITSO) Austin Center

Wade Wallace および Matthew Parente

IBM Communication Server for OS/390 Development

Jeff Haggar、Bebe Isrel、Dinakaran Joseph

Cisco Systems, Inc.

Rick Williams および Edward Mazurek

BOPS, Inc.

Ricardo Rodriguez

North Carolina State University

Karina Rodriguez

第 1 版の著者

Peter Frick、Gerard Bourbigot、Frank Vandewiele

第 2 版の著者

Peter Frick、Lesia Cox、Ricardo Haragutchi

第 3 版の著者

Philippe Beaupied および Frederic Debulois

第 4 版の著者

Philippe Beaupied および Francis Li

第 5 版の著者

Eamon Murphy、Matthias Enders、Steve Hayes

第 6 版の著者

Martin Murhammer、Orcun Atakan、Stefan Bretz、Larry Pugh、Kazunari Suzuki、David Wood

第 1 部 中核となる TCP/IP プロトコル

第 1 章アーキテクチャー、ヒストリー、標準、および傾向

今日では、インターネットとワールド・ワイド・ウェブ (WWW) は、世界中の大勢の人々にとってなじみ深い言葉です。多くの人々が、電子メールや Web アクセスなどの、インターネットを介して使用可能なアプリケーションに依存しています。また、ビジネス・アプリケーションの大衆化に伴って、インターネットへの期待はさらに高まっています。Transmission Control Protocol/Internet Protocol (TCP/IP) プロトコル・スイートは、インターネットおよび世界的なネットワークのためのエンジンです。その単純さと能力によって、これは今日の世界で選び抜かれた唯一のネットワーク・プロトコルとなっています。この章では、この TCP/IP プロトコル・スイートについて概説します。ここではインターネットがどのように組織されたか、また将来どのように発展する可能性があるかについて説明します。

1.1 TCP/IP アーキテクチャー・モデル

TCP/IP プロトコル・スイートという名前は、その最も重要な 2 つのプロトコルである Transmission Control Protocol (TCP) と Internet Protocol (IP) に由来します。これにはインターネット・プロトコル・スイートという、あまり使用されない名前もあり、公式のインターネット標準文書の中ではこの句が使用されています。本書では、このプロトコル・スイート全体を指す場合に、より一般的な略語の TCP/IP を使用しています。

1.1.1 インターネットワーキング

TCP/IP の主な設計目標は、異種混合の物理ネットワーク間に世界規模の通信サービスを提供する、インターネットワークまたはインターネットと呼ばれるネットワークの相互接続を構築することでした。このインターネットワークの明確な利点は、大きな地域区分で分断されている場合もある、異なるネットワーク上にあるホスト間で通信が行えるようになることです。

インターネットワーク (internetwork) およびインターネット (internet) という言葉は、*相互接続ネットワーク (interconnected network)* という句を簡略したものです。しかし、大文字の「I」を用いて Internet と表記されている場合、これは世界的な一連の相互接続ネットワークを指します。(ただしこの邦文資料では大文字小文字を問わず、「インターネット」と表記しています。)したがって、Internet はインターネットですが、その逆は成立しません。インターネットはインターネット接続と呼ばれることもあります。

インターネットは、以下のネットワークのグループから構成されます。

- バックボーン：主に他のネットワークを相互接続するために存在する大規模なネットワーク。現在のところ、バックボーンには、米国の NSFNET、ヨーロッパの EBONE、および大規模な商用バックボーンがあります。

- 大学などの地域的なネットワーク接続。
- 加入者にバックボーンへのアクセスを提供する商用ネットワーク。および
営利団体によって内部使用のために所有され、インターネットにも接続する
ネットワーク。
- 大学のキャンパス内のネットワークなどのローカル・ネットワーク。

多くの場合、ネットワークのサイズは、そのネットワークに所属できるユーザーの数、そのネットワークを拡張できる最大の地理的距離、あるいは特定の環境へのそのネットワークの適用度によって制限されます。たとえば、イーサネット・ネットワークは、本質的に地理的なサイズの点で制限されています。したがって、多数のネットワークをいくつかの階層的かつ組織的な方法で相互接続できれば、このインターネットワークに属するどの2つのホスト間でも通信できるようになります。図1は2つのインターネットの例を表しています。それぞれは複数の物理ネットワークで構成されています。

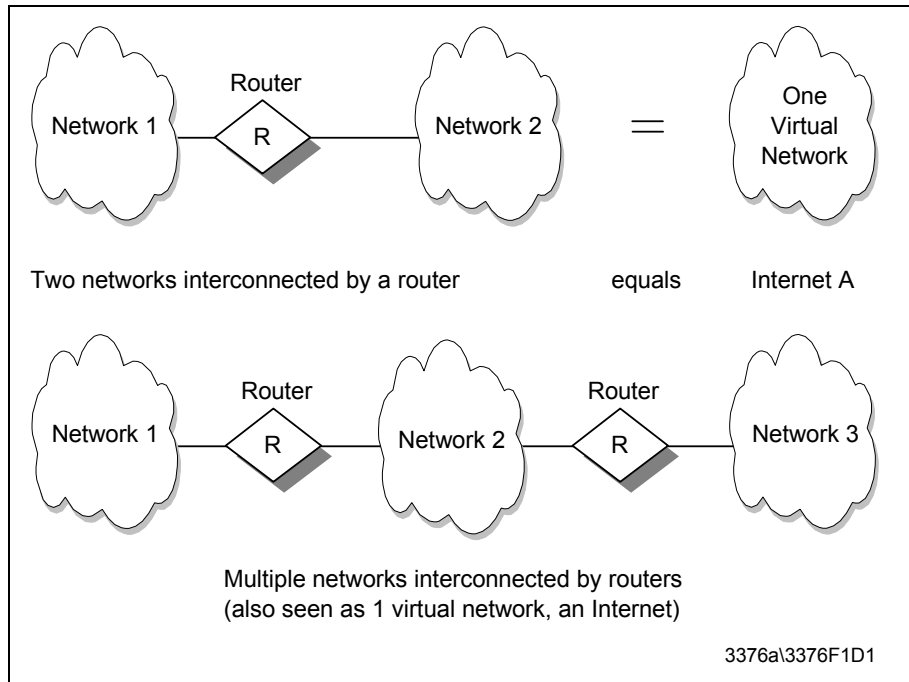


図1 インターネットの例 - それぞれが1つの論理ネットワークと見なされる、2つの相互接続されたネットワークのセット

TCP/IP インターネットワーキングのもう1つの重要な局面は、各タイプのネットワークによって提供される通信メカニズムの、標準化された抽象概念の形成です。各物理ネットワークは、その固有のテクノロジーに依存する通

信インターフェースを、基本通信機能（プリミティブ）として提供するプログラミング・インターフェースの形で持っています。TCP/IP は物理ネットワークのプログラミング・インターフェースとユーザー・アプリケーションとの間で実行する通信サービスを提供します。これにより、下層の物理ネットワークとは独立した、これらのアプリケーションのための共通インターフェースが使用できるようになります。したがって、物理ネットワークのアーキテクチャーはユーザーやアプリケーションの開発者からは隠されます。アプリケーションは、標準化された通信抽象概念に合わせてコーディングするだけで、どんなタイプの物理ネットワークやオペレーティング・プラットフォームの下でも機能させることができます。

図 1 から分かるように、2 つのネットワークを相互接続するには、両方のネットワークに接続して一方のネットワークから他方のネットワークにデータ・パケットを転送することのできるコンピューターが必要です。このようなマシンはルーターと呼ばれます。ルーティング機能は TCP/IP プロトコル・スイートのインターネット・プロトコルの一部なので (5 ページの 1.1.2 『TCP/IP プロトコル層』を参照)、IP ルーターという言葉が使用されることもあります。

インターネットワーク内でホストを識別できるようにするために、各ホストには IP アドレスと呼ばれるアドレスが割り当てられます。ホストがルーターなどの複数のネットワーク・アダプター（インターフェース）を持つとき、各インターフェースは固有の IP アドレスを持ちます。IP アドレスは、以下の 2 つの部分から構成されます。

```
IP address = <network number><host number>
```

IP アドレスの *network number* の部分は、インターネット内のネットワークを識別します。これは中央機関によって割り当てられ、インターネット上で固有です。IP アドレスの *host number* の部分を割り当てる機関は、ネットワーク番号によって識別されたネットワークを制御している組織に所属します。アドレッシング方式については、68 ページの 3.1.1 『IP アドレッシング』で詳しく説明します。

1.1.2 TCP/IP プロトコル層

大半のネットワーキング・ソフトウェアと同様、TCP/IP は層としてモデル化されています。この階層図から、プロトコル・スイート内の層の積み重ねを指す、*プロトコル・スタック*という言葉が生まれました。これは、他の Systems Network Architecture (SNA) や Open System Interconnection (OSI) モデルなどに対する、TCP/IP プロトコル・スイートの位置付け（ただし、機能性の比較ではない）に使用することができます。それぞれのプロトコル・スイートによって使用されている階層モデルには基本的な相違があるので、ここから機能性を比較するのは簡単ではありません。

通信ソフトウェアを複数の層に分割することによって、プロトコル・スタックでは、作業の分割、インプリメンテーションとコードの検査の簡易化、および代替層インプリメンテーションの開発が可能になります。層は簡易なインターフェースを介して上下の層と通信します。その際、層はそのすぐ上の層にサービスを提供し、そのすぐ下の層によって提供されるサービスを利用します。たとえば、IP 層は、信頼性のある送達や重複抑止の保証なしに、データのあるホストから他のホストへ転送する能力を提供します。TCP などのトランスポート・プロトコルは、このサービスを利用して、アプリケーションに信頼性のある順序正しいデータ・ストリーム送達を提供します。図 2 は、TCP/IP プロトコルがどのように 4 つの層にモデル化されているかを表しています。

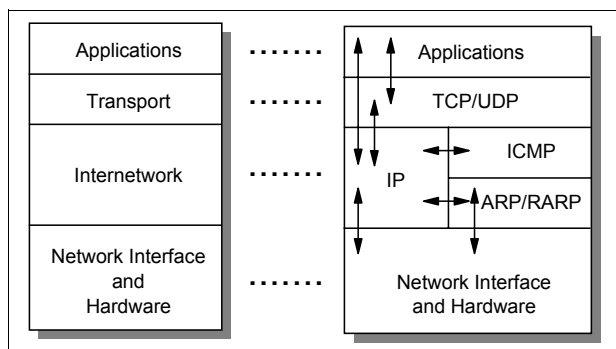


図 2 TCP/IP プロトコル・スタック - 各層は機能のパッケージを表しています

これらの層には、以下が含まれます。

アプリケーション層

アプリケーション層は、通信に TCP/IP を使用するプログラムによって提供されます。アプリケーションとは、通常は異なるホスト上にある他のプロセスと協働するユーザー・プロセスです (単一ホスト内のアプリケーション通信にも利点があります)。アプリケーションの例には、Telnet やファイル転送プロトコル (FTP) があります。アプリケーション層とトランスポート層の間のインターフェースは、ポート番号とソケット (209 ページの 5.1 『ポートとソケット』で詳述) によって定義されます。

トランスポート層

トランスポート層は、アプリケーションからそのリモートのアプリケーションヘータを送達することによって、終端間データ転送を提供します。複数のアプリケーションを同時にサポートすることができます。最も使用されるトランスポート層

プロトコルは、接続指向の信頼性のあるデータ送達、重複データ抑止、輻輳（ふくそう）制御、およびフロー制御を提供する Transmission Control Protocol (TCP) です。これについては、215 ページの 5.3 『Transmission Control Protocol (TCP)』でより詳しく説明します。

もう 1 つのトランスポート層プロトコルは、User Datagram Protocol (UDP、212 ページの 5.2 『User Datagram Protocol (UDP)』で説明) です。これは、コネクションレスの信頼性のないベスト・エフォート・サービスを提供します。そのため、UDP をトランスポート・プロトコルとして使用しているアプリケーションは、必要であれば、その独自の終端間保全性、フロー制御、および輻輳制御を提供する必要があります。UDP は通常、高速なトランスポート・メカニズムを必要とし、一部のデータの逸失を容認できるアプリケーションによって使用されます。

インターネットワーク層

インターネット層またはネットワーク層とも呼ばれるインターネットワーク層は、インターネットの「仮想ネットワーク」イメージを提供します（この層は、この下の物理ネットワーク・アーキテクチャーから上のレベルをシールドします）。インターネット・プロトコル (IP) はこの層の中で最も重要なプロトコルです。これは、下の層から信頼性を前提としないコネクションレス・プロトコルです。IP は信頼性、フロー制御、またはエラー・リカバリーを提供しません。これらの機能はこれより上のレベルで提供される必要があります。

IP は伝送されたメッセージをその宛先へ送達することを試みるルーティング機能を提供します。IP については、67 ページの 3.1 『インターネット・プロトコル (IP)』でより詳しく説明します。IP ネットワーク内のメッセージ単位は、*IP データグラム*と呼ばれます。これは TCP/IP ネットワーク間で伝送される情報の基本単位です。その他のインターネットワーク層プロトコルは、IP、ICMP、IGMP、ARP、および RARP です。

ネットワーク・インターフェース層

リンク層またはデータ・リンク層とも呼ばれる

ネットワーク・インターフェース層は、実際のネットワーク・ハードウェアへのインターフェースです。このインターフェースは、信頼性のある送達を提供することもしないことも可能であり、パケット指向にもストリーム指向にもできます。実際、TCP/IP はこの層のプロトコルは指定していませんが、使用可能なネットワーク・インターフェースのほぼすべてを使用することができ、このことは IP 層の柔軟性を例証しています。IEEE 802.2, X.25 (これはそれ自体に信頼性がある)、ATM、FDDI、および SNA もこの例です。29 ページの第 2 章『ネットワーク・インターフェース』では、いくつかの物理ネットワークとインターフェースについて説明しています。

TCP/IP 仕様自体は、ネットワーク層プロトコルの説明または標準化はしていません。これはインターネットネットワーク層からのこれらのプロトコルへのアクセス方法を標準化しているだけです。

図 3 にはより詳細な階層モデルがあります。

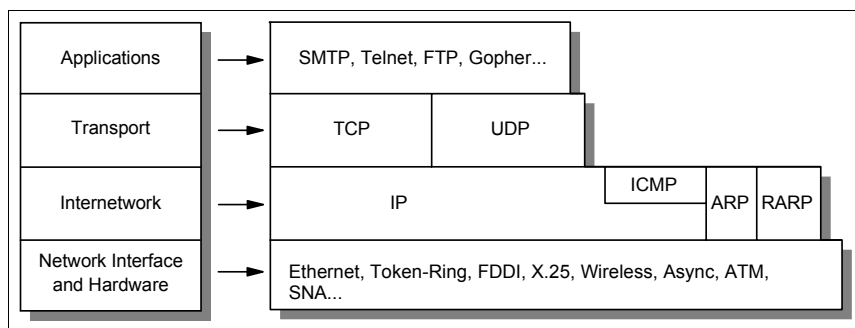


図 3 詳細なアーキテクチャー・モデル

1.1.3 TCP/IP アプリケーション

TCP/IP プロトコル・スタック内の最上位プロトコルは、アプリケーション・プロトコルです。これらは他のインターネット・ホスト上のアプリケーションと通信する、TCP/IP プロトコル・スイートへのユーザー可視のインターフェースです。

すべてのアプリケーション・プロトコルは以下のようないくつかの特性を共通に持っています。

- これらは、ユーザー作成のアプリケーション、または標準化されて TCP/IP 製品に同梱されているアプリケーションのいずれかです。実際に TCP/IP プロトコル・スイートには、次のようなアプリケーション・プロトコルが組み込まれています。

- リモート・インターネット・ホストへの対話式端末アクセスのための TELNET。
- 高速なディスク間ファイル転送のための FTP (ファイル転送プロトコル)。
- インターネット・メーリング・システムとしての SMTP (Simple Mail Transfer Protocol)。

これらは最も広くインプリメントされているアプリケーション・プロトコルの一部で、他のものも多数存在します。特定の各 TCP/IP インプリメンテーションには、多かれ少なかれアプリケーション・プロトコルのセットが組み込まれています。

- これらはトランスポート・メカニズムとして UDP または TCP のいずれかを使用します。UDP は信頼性がなくフロー制御を提供しないので、この場合は、アプリケーションがその独自のエラー・リカバリー、フロー制御、および輻輳制御機能を提供する必要があることを覚えておいてください。TCP は信頼性のあるストリームの、接続指向で、輻輳を処理やすく、フロー制御対応プロトコルなので、アプリケーションは多くの場合 TCP の上に構築するほうが簡単です。以上のことから、大半のアプリケーション・プロトコルは TCP を使用しますが、プロトコル・オーバーヘッドを削減することによってより優れたパフォーマンスを獲得するために、UDP 上に構築されるアプリケーションもあります。
- 大半のアプリケーションは、クライアント / サーバー・モデルの対話を使用します。

1.1.3.1 クライアント / サーバー・モデル

TCP はピアツーピアのコネクション型のプロトコルです。マスター / スレーブ関係はありません。ただし、アプリケーションは一般に、通信にはクライアント / サーバー・モデルを使用します。

サーバーはインターネット・ユーザーにサービスを提供するアプリケーションであり、クライアントはサービスの要求側です。アプリケーションはサーバー部分とクライアント部分の両方で構成され、それらは同一のシステム上で実行させることも、異なるシステム上で実行させることもできます。ユーザーは通常はアプリケーションのクライアントの部分起動します。これは特定のサービスの要求を作成し、それをアプリケーションのサーバーの部分に、TCP/IP をトランスポート手段として使って送信します。

サーバーとは、要求を受信し、必要なサービスを実行し、その結果を応答で
送り返すプログラムです。サーバーは通常、複数の要求と複数の要求元クラ
イアントを同時に処理することができます。

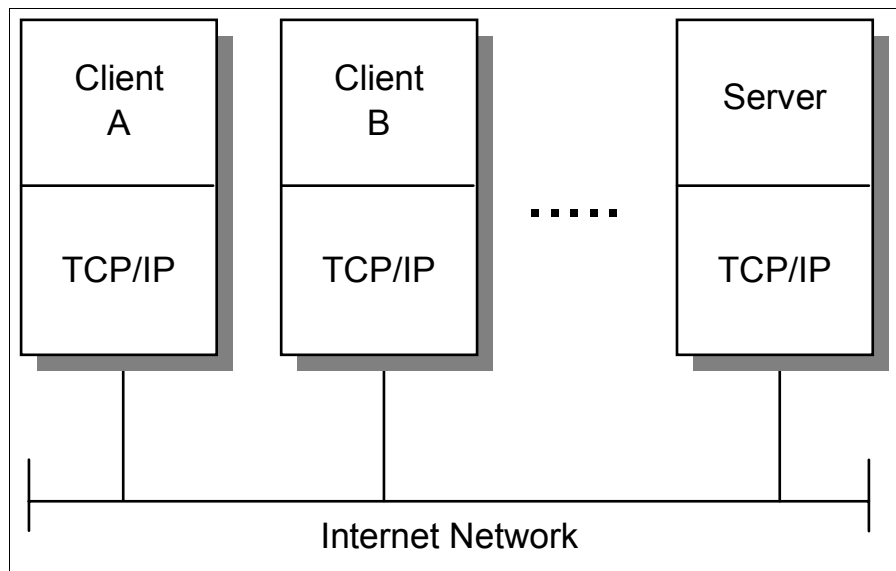


図4 アプリケーションのクライアント/サーバー・モデル

大半のサーバーはウェルノウン・ポートで要求を待つため、クライアントはどのポートに（続いてどのアプリケーションに）要求を送信すべきかが分かっています。クライアントは通常、一時ポートと呼ばれる任意のポートを使用して通信します。ウェルノウン・ポートを使用しないサーバーと通信しようとするクライアントは、その要求をどのポートに宛てるべきかを知るための別のメカニズムを持っていなければなりません。このメカニズムは、ウェルノウン・ポートを使用するポート・マップなどの登録サービスを採用していることがあります。

TCP/IP アプリケーション・プロトコルの詳細については、269 ページの第 2 部『TCP/IP アプリケーション・プロトコル』を参照してください。

1.1.4 ブリッジ、ルーター、およびゲートウェイ

他のネットワークにアクセスする方法は多数あります。インターネットワークでは、これはルーターによって行われます。このセクションでは、リモート・ネットワーク・アクセスを可能にするルーター、ブリッジ、およびゲートウェイの相違について説明します。

ブリッジ ネットワーク・インターフェース層レベルで LAN セグメントを相互接続し、それらの間でフレームを転送します。ブリッジは、MAC リレーの機能を実行し、高位層プロトコル (論理リンク・プロトコルを含む) から独立しています。これは必要に応じて MAC 層プロトコル変換を行います。

ブリッジは、IP に対して透過的であると言われます。つまり、IP ホストは、ブリッジで接続されたネットワーク上の他のホストに IP データグラムを送信するとき、データグラムを直接そのホストに送信し、そのデータグラムは送信側 IP ホストに気付かれずにブリッジを「横断」します。

ルーター ネットワークをネットワーク層レベルで相互接続し、それらの間でパケットをルーティングします。ルーターは、サポートしているネットワーク・プロトコルに関連したアドレッシング構造を理解し、パケットを転送するかどうか、またはそれをどのように転送するかを決定しなければなりません。ルーターは最善の伝送パスと最適なパケット・サイズを選択することができます。基本的なルーティング機能は、TCP/IP プロトコル・スタックの IP 層にインプリメントされているため、理論上、または今日の大半の TCP/IP インプリメンテーションでも、複数のインターフェースを介して TCP/IP を実行するホストまたはワークステーションは、IP データグラムを転送することができます。ただし、専用ルーターは、IP によってインプリメントされている最小限の機能よりもずっと複雑なルーティングを行います。

IP はこの基本ルーティング機能を提供しているので、「IP ルーター」という用語がよく使用されます。ルーターに対応する他の古い用語は、「IP ゲートウェイ」、「インターネット・ゲートウェイ」、および「ゲートウェイ」です。ゲートウェイという用語は、現在では普通ネットワーク層より上の層の接続に対して用いられます。

ルーターは、IP に対して可視的であると言われます。つまり、ホストは、ルーターで接続されたネットワーク上の他のホストに IP データグラムを送信する場合、データグラムをルーターに送信して、ルーターがそれをターゲット・ホストに転送できるようにします。

ゲートウェイ

ネットワークをブリッジおよびルーターよりも上の層で相互接続します。ゲートウェイは通常、1つのネットワークから他のネットワークへのアドレス・マッピングをサポートし、さらに終端間アプリケーション接続をサポートする環境間でのデータの変換を行うこともできます。一般にゲートウェイは、2つのネットワークの相互接続を、いずれかのネットワークでサポートされているアプ

リケーション・プロトコルのサブセットまでに制限します。たとえば、TCP/IP を実行する VM ホストは、SMTP/RSCS メール・ゲートウェイ^注として使用できます。

注

「ゲートウェイ」という語は、この意味で使用されている場合は「IP ゲートウェイ」と同義ではありません。

ゲートウェイは、IP に対して不透明であると言われます。つまり、ホストはゲートウェイを介して IP データグラムを送信することはできず、ゲートウェイにこれを送信できるだけです。データグラムによって伝送される上位プロトコルの情報は、ゲートウェイによって、もう一方の側のゲートウェイで使用されているネットワーク・アーキテクチャーが何であれそれを使用して渡されます。

ルーターおよびゲートウェイと密接に関連しているのは、ファイアウォールまたはファイアウォール・ゲートウェイの概念です。これは、セキュリティのためにインターネットや一部の信用されていないネットワークから、組織によって管理されているネットワークまたはネットワークのグループへのアクセスを制限するために使用されます。ファイアウォールの詳細については、704 ページの 21.3 『ファイアウォール』を参照してください。

1.2 インターネットのルーツ

ネットワークは、最重要というわけではないにしても、今日の情報システムの基礎となっています。これは、企業、政府、および研究グループにおける情報共用のためのバックボーンです。この情報はいろいろな形を取る可能性があります。これはメモや文書、他のコンピューターに処理されるデータ、同僚に送信されたファイル、そしてマルチメディア・データ・ストリームかもしれません。

ネットワーク設計が、コンピューター研究と技量あるインプリメンターの「最新技術的」課題であった 60 年代後半と 70 年代には、多くのネットワークが実装されました。この結果、パケット交換テクノロジー、衝突検出ローカル・エリア・ネットワーク、階層ネットワークなどの数々のネットワークング・モデルや、他の多くの優れた通信テクノロジーが生まれました。

このすべての偉大な技術知識により、あらゆるグループのユーザーがその特定のニーズに適した物理ネットワークとアーキテクチャー・モデルを見いだせるようになりました。これは、ビット対ビットのパリティ機能以外エラー・リカバリーを持たない安価な非同期回線から、公用パケット交換ネットワークや専用 SNA ネットワークなどの信頼性のあるプロトコルを持つ完全

機能の広域ネットワーク（公用または専用）、さらには高速で距離制限があるローカル・エリア・ネットワークにまで及びます。

このような異種混合プロトコル・スイートの発達のマイナス面は、あるユーザーのグループがその情報システムを、異なるネットワーク・テクノロジーおよび異なるネットワーク・プロトコルをインプリメントしている他のユーザーのグループにまで拡張したい場合に、多少骨が折れるということです。この場合、たとえ両者が2つの環境を物理的に相互接続するためのいくつかのネットワーク・テクノロジーについて合意できたとしても、そのアプリケーション（メール・システムなど）は、アプリケーション・プロトコルとインターフェースの違いのために互いに通信できない場合があるからです。

この状態は70年代初頭に、*Defense Advanced Research Projects Agency* (DARPA) から資金を受けた米国の研究者グループによって認識されました。彼らの研究はインターネットワーキングまたはネットワークの相互接続について扱ったものでした。ITU-T（以前はCCITT）およびISOなどの他の公的組織も、この分野に関与するようになりました。その主な目的は、一連のプロトコルを、明確な組として詳細に定義することによって、アプリケーションが、下層のネットワーク・テクノロジーや実行しているオペレーティング・システムに関係なく、他のアプリケーションと通信できるようにすることでした。

これらの研究者の公的な組織はARPANETネットワーク・ワーキング・グループで、この最後の総会は1971年の10月に開かれました。初期の*Network Control Program* (NCP) のホスト対ホストのプロトコルから、1978年前後に現在の形を取るようになったTCP/IPプロトコル・スイートに至るまでの、インターネットワーキング・プロトコル・スイートについてこの研究はDARPAが継続して行いました。当時DARPAは、その無線ネットワークおよびサテライト・チャネルを介したパケット交換の開拓でよく知られていました。インターネットの最初の実際のインプリメンテーションは、DARPAがその調査ネットワーク(ARPANET)のマシンを、新しいTCP/IPプロトコルを使用するように切り替え始めた1980年ころに見受けられるようになりました。1983年にその移行は完了し、DARPAはそのARPANETに接続することを望むすべてのコンピューターがTCP/IPを使用することを要求しました。

DARPAはさらに、Bolt, Beranek, and Newman (BBN) と契約し、VAX上のBerkeley UNIX用のTCP/IPプロトコルのインプリメンテーションを開発し、カリフォルニア大学バークレー校に出資してそのUNIXオペレーティング・システムを使い、そのコードを無料で配布しました。TCP/IPプロトコル・セットを組み込んだ*Berkeley Software Distribution* (BSD)の最初のリリースは、1983年に入手可能になりました(4.2BSD)。その時以来、TCP/IPは大学や研究センターの間で急速に広まり、すべてのUNIX接続用の標準通信サブシステムになりました。2番目のリリース(4.3BSD)は1986年に配布され、1988年(4.3BSD Tahoe)と1990年(4.3BSD Reno)に更新されていま

す。1993 年には 4.4BSD がリリースされました。出資契約のため、4.4BSD はカリフォルニア大学バークレー校の Computer Systems Research Group による BSD の、最後のリリースになりました。

TCP/IP インターネットワーキングが急速に広がるに伴って、新しい広域ネットワークが米国で作成され、ARPANET に接続されました。さらに必ずしも TCP/IP プロトコルをベースとするわけではない世界の他の国々の他のネットワークが、相互接続されたこのネットワークのセットに次々に追加されました。こうしてできたのがインターネットです。続くいくつかのセクションでは、この発展において主要な役割を演じた種々のネットワークについて、そのいくつか例を紹介します。

1.2.1 ARPANET

「パケット・ネットワークの祖父」と呼ばれることもある ARPANET は、パケット交換テクノロジーの研究施設に合わせ、米国国防総省の受託企業がリソースを共用できるようにするために、60 年代後半に DARPA (当時は ARPA と呼ばれた) によって構築されました。このネットワークは研究センター、いくつかの軍事基地、および政府所在地を相互接続しました。まもなくこれは、電子メールおよびその他のサービスによって、共同研究の研究者たちがよく使用するようになりました。1975 年の終わりまでには、これは Defense Communications Agency (DCA) によって運用される調査ユーティリティーへと発展し、1983 年には、軍事サイトの相互接続用の MILNET と、研究サイトの相互接続用の ARPANET に分割されました。これが、大文字 I の Internet の始まりです。

1974 年には、ARPANET は米国本土と西ヨーロッパに散在していたパケット交換ノード (PSN) 相互接続は、56 Kbps 専用回線をベースにしていました。パケット交換ノードとは、1822 (これについて記述するレポートの番号に由来) として知られるプロトコルを実行する、パケット交換タスク専用のミニコンピュータです。各 PSN は、他の PSN への少なくとも 2 つの接続 (回路が故障した場合に代替ルーティングが行えるようにするため) と、ユーザー・コンピュータ (ホスト) 接続のための最大で 22 のポートを持っていました。これら 1822 システムは、宛先ノードへの信頼性のある、フロー制御付きパケット送達を提供しました。元の NCP プロトコルが多少単純なプロトコルであった理由はここにあります。これは、下層のネットワーク・ハードウェアの信頼性を前提とせず、1822 以外のネットワークでも使用できる TCP/IP プロトコルによって置き換えられました。この 1822 プロトコルは業界標準にはならなかったので、DARPA は後にこの 1822 パケット交換テクノロジーを CCITT X.25 標準に置き換えることに決定しました。

データ・トラフィックはすぐにネットワークを構成していた 56 Kbps 回線の容量を超え、これは必要なスループットをサポートできなくなりました。今日では ARPANET は、接続されたインターネットの研究サイドのバックボーン

ンとしてのその役割を、新しいテクノロジー (この章の後述の NSFNET を参照) によって取って代わられましたが、MILNET は引き続き軍事サイドのバックボーンです。

1.2.2 NSFNET

NSFNET、つまり National Science Foundation (NSF) Network は、米国内の 3 つのレベルからなるインターネットワークで、以下によって構成されています。

- バックボーン：別個に管理および操作されるミッドレベルのネットワークと、NSF の出資するスーパーコンピューター・センターを接続するネットワーク。このバックボーンは EBONE (European IP backbone network) などの他のネットワークへの大陸横断リンクもあります。
- ミッドレベル・ネットワーク：3 種類 (地域、訓練ベース、およびスーパーコンピューター連合のネットワーク)
- キャンパス・ネットワーク：学術研究用または商用。ミッドレベル・ネットワークに接続される。

NSF は増大するそのクライアントの要求を満たすため、数年の間にそのバックボーンを以下のようにアップグレードしました。

最初のバックボーン 研究者および科学者たちが NSF スーパーコンピューターにアクセスするための通信ネットワークとして NSF により当初確立された最初の NSFNET は、56 Kbps の専用回線で相互接続された 6 つの DEC LSI/11 マイクロコンピューターをパケット交換として使用しました。NSFNET バックボーンと ARPANET の間の主要な相互接続はカーネギー・メロンにあり、これがこれらの各ネットワークに接続されているユーザー間でのデータグラムのルーティングを許可していました。

2 番目のバックボーン 最初のバックボーンは何か月かで過負荷になり、1987 年には新しいバックボーンの必要が生じました (当時の見積成長率は、年 100 % でした)。NSF と、ミシガンの 8 つの州立大学のコンピューター・ネットワーク連合である MERIT, Inc. は、より優れた伝送および交換能力を備えた新しいより高速なバックボーンを開発および管理することに合意しました。これを管理するため、彼らは Information Center と Technical Support Group から成る *Information Services (IS)* を定義しました。Information Center は、情報配布、情報リソース管理、および電子通信を担当します。Technical Support Group は、その分野を直接サポートします。

その目的は、一式の訓練サービスによってサポートされるネットワーク内のどの地点からでもアクセスでき、使用と管理が簡単なインターフェースを備えた統合情報システムを提供することです。

Merit と NSF は、IBM および MCI と提携してこのプロジェクトを企画しました。IBM はソフトウェア、パケット交換、およびネットワーク管理装置を提供し、MCI は長距離伝送設備を提供しました。1988 年に導入された新しいネットワークは当初 448 Kbps 専用回線を使用して、IBM の提供する 13 の *Nodal Switching System* (NSS) を相互接続していました。各 NSS は、2 つの IBM トークンリング・ネットワークを介して (冗長性のため)、疎結合された 9 つの IBM RISC システム (IBM 版の 4.3BSD UNIX を実行する) で構成されていました。以下を提供するため、13 のそれぞれの場所に、IBM 提供の Integrated Digital Network Exchange (IDNX) が 1 つずつインストールされました。

- 動的代替ルーティング
- 動的帯域幅割り振り

3 番目のバックボーン 1989 年、NSFNET バックボーン回線トポロジーはトラフィック測定の後には再構成され、専用回線の速度は主に光ファイバーを使用する T1 (1.544 Mbps) にまで増加しました。

改善されたパケット交換と伝送の能力の必要性は絶えず間なく増大するため、3 つの NSS がバックボーンに追加され、リンク速度がアップグレードされました。NSFNET バックボーンの T1 から T3 (45Mbps) への移行は、1992 年後半に完了しました。その後のギガビット・レベルへの移行はすでに開始し、現在も継続中です。

1995 年 4 月、米国政府は NSFNET への出資を停止しました。これは、一つに、ネットワークの商業的な使用の増大に対する反応でもありました。ほぼ同時に、NSFNET は米国内のメイン・バックボーン・トラフィックを徐々に移行させて商用のネットワーク・サービス・プロバイダーにし、NSFNET は再び研究者コミュニティ向けのネットワークとなりました。メイン・バックボーン・ネットワークは、現在は vBNS (very high speed Backbone Network Service) として知られる MCI と協働して実行されています。

NSFNET はインターネットの発展において主要な役割を演じました。ただし他の多くのネットワークも、その役目を果たし、現在でもインターネットの一部を構成しています。

1.2.3 インターネットの営利使用

近年インターネットは、サイズと規模の面でこれまでだれも予測し得なかったほど急速に成長しています。この成長には、いくつかの主要な要因が影響しています。最も意義深く画期的な出来事には、1991 年の Gopher の無料配布と、同じく 1991 年のハイパーテキストの仕様の最初の登録、および 1993 年の最初のグラフィックス・ベース・ブラウザである Mosaic のリリースがあります。今日ではインターネットに接続されているホストの大多数は営利的な性質のものです。これは、学術および研究機関の間のオープンな通信を強化するという、インターネットの当初の目的との潜在のおよび現実的な矛盾点です。ただしインターネットの営利使用の今後の増大は不可避なため、この発展の経緯についての説明は参考になるでしょう。

考慮すべき一つの重要な第一歩は、*Acceptable Use Policy* (AUP) の始まりです。このポリシーの最初のは 1992 年に導入され、NSFNET の使用に適用されました。この AUP の中心にあるのは、「開かれた研究と教育をサポートする」という公約です。「認められない使用」の項で、「営利活動目的の使用」は、一般原則あるいは特別許可使用の範囲内でない限り禁止されています。しかし、この明確な制約方針があるにもかかわらず、1995 年にその当初の目的に立ち返るまで、NSFNET はますます多くの営利的な性質の活動を含めた広範な活動に使用されるようになりました。

一般に AUP は、商業的な使用により適した形へと進化してきましたが、AUP の提供は現在インターネット・サービス・プロバイダーの間ではごく普通です。AUP なしでサービスを提供しているネットワークもあります。

次に、インターネットの商業的使用の導入に最も積極的であったインターネット・サービス・プロバイダーに焦点を当ててみましょう。言及に値する 2 つのインターネット・サービス・プロバイダーは、ビジネスと個人の両方にインターネット・アクセスを提供するために 80 年代後半に始まった PSINet と UUNET です。カリフォルニアを本拠地とする CERFnet は、AUP なしでサービスを提供していました。その後まもなく、1 つのネットワークのメンバーのトラフィックが、制約を受けずに他のメンバーのネットワーク上をフローできるようにするという合意に基づいて、PSINet、UUNET、および CERFnet を相互接続する Commercial Internet Exchange (CIX) と呼ばれる組織が編成されました。1997 年 7 月より、CIX はメンバーのインターネットを接続して、世界各地の 146 を超えるメンバーにまで拡大しました。CIX が編成されたのとはほぼ同じ頃、IBM、MCI、および Merit, Inc. によって、NSFNET の T1 (後の T3) バックボーン接続を操作するための、Advance Network and

Services (ANS) という非営利企業が設立されました。このグループはインターネットへの営利的参加の推進に積極的でした。

ANS は、商業的なカスタマーと研究のおよび教育的なドメインとの間にリンケージを提供する、ANS CO+RE と呼ばれる営利指向の子会社を設立しました。ANS CO+RE は、NSFNET へのアクセスを提供すると同時に、CIX にもリンクされています。ANS は 1995 年に America Online に買収されました。

NSFNET がその以前の学術研究的役割に戻った 1995 年には、インターネットのアーキテクチャーは、米国内に単一の支配的なバックボーンを持つものから、複数の商業的に操作されるバックボーンを持つものへと変わりました。異なるバックボーン同士がデータを交換できるようにするため、NSF はバックボーン・サービス・プロバイダー間のデータ交換点となる、4 つの Network Access Points (NAP) をセットアップしました。

交換のもう 1 つのタイプは、Metropolitan Area Ethernet (MAE) です。いくつかの MAE が、同じくその独自のバックボーン・ネットワークを持つ Metropolitan Fiber Systems (MFS) によってセットアップされました。NAP および MAE は、公衆交換点 (IXP) とも呼ばれます。基本的にインターネット・サービス・プロバイダー (ISP) は、パフォーマンスとバックアップのために複数の IXP に接続します。現在の IXP のリストについては、以下にある「Exchange Point」を参照してください。

<http://www.ep.net>

米国の CIX と同様に、ヨーロッパのインターネット・プロバイダーも、技術面および管理面の調整を行うための RIPE (Réseaux IP Européens) 組織を編成しました。RIPE は、ヨーロッパのユーザーに統一的な IP サービスを提供するために 1989 年に設立されました。今日、最大のインターネット・バックボーンは、OC48 (2.4 Gbps) または OC192 (96 Gbps) で実行します。

1.2.4 Internet2

インターネットの成功、およびその後の NSFNET の頻繁な輻輳とその商業的な転換は、以前にインターネットの独占的な使用を楽しんでいた研究者コミュニティの間に不満を招きました。そのため、大学コミュニティは、政府および産業パートナーと共に、また Next Generation Internet (NGI) イニシアチブの出資構成に奨励されて、Internet2 プロジェクトを設立しました。

NGI イニシアチブは、拡張ネットワーク・テクノロジーを開発し、拡張ネットワーク・テクノロジーを必要とするアプリケーションを画期的に導入し、高速のテストベッドでそのテクノロジー機能を実証している連合の研究プログラムです。

1.2.4.1 使命

Internet2 の使命は、拡張された、ネットワーク・ベースのアプリケーションおよびネットワーク・サービスの開発、操作、およびテクノロジー推移を促進および調整し、研究および高等教育における米国の指導力を高め、インターネットでの新しいサービスとアプリケーションの可用性を増進することです。

Internet2 の目標は、以下のとおりです。

- ・ 共同研究したり実験を行ったりする研究者の能力を画期的に強化できる新しいアプリケーションを示す。
- ・ 拡張コミュニケーション・インフラストラクチャーによって作られる *仮想接近性* を利用することによって、教育やその他のサービス（たとえば、健康管理や環境モニターなど）の拡張送達を示す。
- ・ ミドルウェアおよび拡張ツールを提供することによって、拡張アプリケーションの開発および導入をサポートする。
- ・ 研究および教育コミュニティのアプリケーション要件に応じたさまざまな Quality of Service (QoS) をサポートできる、適正なコストのコミュニケーション・インフラストラクチャーの開発、配置、および操作を促進する。
- ・ 次世代コミュニケーション・テクノロジーの実験を推進する。
- ・ 参加機関の間での合意作業基準および共通手順の採用を調整し、終端間のサービス品質とインターオペラビリティを保証する。
- ・ 政府および民間セクターの組織の提携を促す。
- ・ Internet2 からその他の Internet へのテクノロジーの転移を奨励する。
- ・ 高等教育および一般のインターネット・コミュニティに対する、新しいインフラストラクチャー、サービス、およびアプリケーションの影響を研究する。

1.2.4.2 Internet2 の参加者

Internet2 には、米国内の 180 の大学が参加しています。提携組織はこのプロジェクトに価値ある入力データを提供します。Internet2 プロジェクトのすべての参加者は、University Corporation for Advanced Internet Development (UCAID) のメンバーです。

ほとんどの点において、Internet2 の提携および出資調整は、研究機関と政府の以前の合併ネットワーク事業 (NSFnet プロジェクトは、その非常に成功した例) と類似しています。米国政府は、NGI イニシアチブと関連プログラムを通して Internet2 に参加します。

Internet2 は、ブロードバンドのネットワーク・アプリケーションの要件を満たすために必要な拡張ネットワーク・サービスを作成する、企業のリーダーとも連携しています。産業パートナーは、主にキャンパス・ベースの地域大学チームと協働して、このプロジェクトによって開発されたアプリケーションをインプリメントするために必要なサービスおよび製品を提供します。Internet2 に現在参加している主な企業には、Alcatel、Cisco Systems、IBM、Nortel Networks、Sprint、および Sun Microsystems があります。Internet2 の追加サポートは、研究および教育のネットワーキングについて研究している非営利組織との協働によってもたらされます。このプロジェクトに携わっている提携組織には、MCNC、Merit、National Institutes of Health (NIH)、および State University System of Florida があります。

Internet2 の詳細については、以下にあるそれらの Web ページを参照してください。

<http://www.internet2.edu>

1.2.5 Open Systems Interconnection (OSI) 参照モデル

OSI (Open Systems Interconnect) 参照モデル (ISO 7498) は、物理トランスポートを下層に、またアプリケーション・プロトコルを上層に持つ、7 層のデータ通信モデルを定義します。図 5 に示すこのモデルは、ネットワーク・プロトコル・スタックがどのように操作するかを理解するための基礎として、またネットワーク・スタック・インプリメンテーションを比較するための参照ツールとして広く受け入れられています。

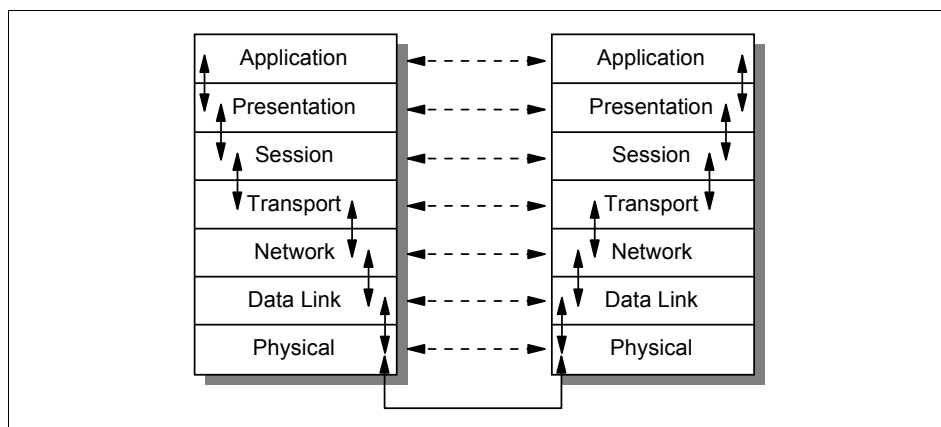


図 5 OSI 参照モデル

各層は、上の層に対して一連の機能を提供すると同時に、下の層が提供する機能に依存します。論理的な観点からすれば、メッセージは重なった層から

層へ縦方向にしか渡っていきませんが、各層は、他のノード上のその対等の層と直接通信します。

7つの層は以下のとおりです。

アプリケーション

端末エミュレーションおよびファイル転送などのネットワーク・アプリケーション

プレゼンテーション

データおよび暗号化のフォーマット

セッション セッションの確立および保守

トランスポート 信頼性ありおよび信頼性なしの終端間送達の提供

ネットワーク ルーティングを含むパケット送達

データ・リンク 情報単位とエラー検査のフレーム

物理 物理ハードウェア上のビットの送信

TCP/IP とは対照的に、OSI のアプローチは白紙の状態から始まり、その独自のモデルに固く準拠しながら、インプリメンテーションを必要としない正式な委員会の過程を経て標準を定義しています。インターネット・プロトコルでは、あまり正式ではない技術アプローチが使用されています。これはだれもが RFC を提案してそれにコメントでき、インプリメンテーションでの実現可能性の証明が求められます。OSI プロトコルの発展は緩やかであり、全プロトコル・スタックの実行はリソース集約的であるため、特にデスクトップや小さなコンピューターの市場では広範にはデプロイされていません。その反面、TCP/IP とインターネットの発展は急速で、デプロイメントは急速に進みました。

1.3 TCP/IP 標準

TCP/IP はその特有のオープンネスと絶え間のない更新のために、開発者にもユーザーにも人気があります。オープンな通信ネットワークとしてのインターネットにも同じことが当てはまります。他方でこのオープンネスは何らかの方法で制御されなければ、容易に両刃の剣へと変化する可能性があります。インターネットのための指示や規定を出す総合的な管理機関は存在しません – 制御は主に相互の協力に基づく – が、Internet Society (ISOC) はインターネット・コミュニティのための標準化機関として機能しています。これは、Internet Architecture Board (IAB) によって編成および管理されています。

IAB 自体は、Internet Engineering Task Force (IETF) に依頼して新しい標準を打ち出し、Internet Assigned Numbers Authority (IANA) に依頼して複数のプ

ロトコル間で共用されている値を調整します。新しい標準文書の検討と公開は、RFC 編集者が担当します。

IETF 自体は、Internet Engineering Steering Group (IESG) の下に置かれており、さらに「分野 (Areas)」と「ワーキング・グループ (Working Groups)」という組織に編成されていて、そこで新しい仕様についての審議と新しい標準の提案が行われます。

RFC 2026 – Internet Standards Process - Revision 3 で説明されているインターネット標準プロセスは、TCP/IP プロトコル・スイートの一部かどうかに関係なく、インターネットで使用するすべてのプロトコル、プロシージャ、および規則に関係しています。

インターネット標準化プロセスの主な目標は、以下のとおりです。

- 技術面での卓越
- 事前のインプリメンテーションとテスト
- 明瞭、簡潔、かつ理解しやすい文書
- オープンネスとフェアネス
- 適時性

標準化のプロセスを以下に要約します。

- 新しい仕様が標準として承認されるためには、アプリケーションがその仕様を IESG に送信し、そこでその仕様が技術的な利点と実現可能性について審議および検討されて、さらにインターネット・ドラフト文書として公開される必要があります。これには 2 週間～ 6 か月かかります。
- IESG は肯定的な結論に達すると、インターネット・コミュニティ全体によるその仕様の検討を許可する最後通告を出します。
- IESG による最終承認の後、インターネット・ドラフトが、標準化過程への組み込みと Request for Comment としての公開のために、IAB のもう 1 つの関連組織である Internet Engineering Taskforce (IETF) に対して推薦されます (23 ページの 1.3.1 『Request For Comments (RFC)』を参照)。
- RFC として公開されると、その寄稿は 25 ページの 1.3.2 『インターネット標準』で説明されている状態に進みます。これは時を経て改訂されたり、より優れたソリューションが見つかった時に廃止されることもあります。
- 提出から 6 か月過ぎても IESG が新しい仕様を承認しない場合、または提出から 6 か月の間文書が未変更のままである場合、これはインターネット・ドラフト要覧から除去されます。

1.3.1 Request For Comments (RFC)

インターネット・プロトコル・スイートは、*Request For Comments* (RFC) の機構を通して現在も進化しています。研究者たちによって新しいプロトコル (ほとんどがアプリケーション・プロトコル) が設計されてインプリメントされ、インターネット・ドラフト (ID) の形でインターネット・コミュニティから注目されることになります。¹ ID の最大のソースは、IAB のサブディレクトリーである Internet Engineering Task Force (IETF) です。ただし、ID として提案されたメモは、誰もが RFC 編集者に提出できます。RFC が受け入れられるために、RFC/ID 作成者が従うべき一連の規則があります。この規則自体は、RFC (RFC 2223) に記述されており、これには RFC に対する提案を提出する方法も示されています。

RFC が公開されると、すべての改訂と置換が新しい RFC として公開されます。既存の RFC を改訂または置換する RFC は、その RFC を「更新する」または「廃止する」ものです。既存の RFC は、新しい RFC によって「更新される」または「廃止される」ものです。たとえば、BOOTP プロトコルについて記述している RFC 1542 は、RFC 1532 の改訂で、RFC 951 の修正である「第 2 版」です。したがって、RFC 1542 には、「Obsoletes RFC 1532; Updates RFC 951」というラベルが付きまます。結果として、複数の現行バージョンが存在することはないので、2 人の人がバージョンの異なる RFC を参照しているかどうかについての混乱はありません。

RFC には、*情報文書* として記述されているものと、インターネット・プロトコルとして記述されているものがあります。プロトコル・スイートについて記述する RFC のリストは、Internet Architecture Board (IAB) が保守しています。これらにはおののちに、*状態* と *状況* が割り当てられています。

インターネット・プロトコルは、以下のいずれかの状態になります。

標準

IAB はこれをインターネットの公式プロトコルとして設定しています。これらは、以下の 2 つのグループに分けられています。

1. インターネット全体に適用される、IP プロトコル以上のプロトコル。
2. ネットワーク固有プロトコル。通常は特定のタイプのネットワーク上での IP 処理方法の仕様書。

ドラフト標準

IAB は、このプロトコルを、標準プロトコルになり得るものとして積極的に考慮します。かなりの数の幅広いテストとコメントが求められます。コメントとテスト結果は IAB に提出される必要があります。ドラフ

¹ これらのプロトコルの一部、特に 4 月 1 日の日付のものは、どう見ても非実用的です。たとえば、RFC 1149 (1990 年 4 月 1 日の日付) は、伝書鳩による IP データグラムの伝送について説明し、RFC 1437 (1993 年 4 月 1 日付け) は、電子メールによる人間の伝送について説明しています。

ト・プロトコルは、標準になる前に、変更が加えられる可能性があります。

提案標準

これらは、将来の標準化に向けて IAB によって考慮される可能性のあるプロトコルの提案です。複数のグループによるインプリメンテーションとテストが求められます。このプロトコルも改訂される可能性があります。

実験

実験のプロトコルは、実験に参加し、そのプロトコルの使用をプロトコルの開発者との間で調整していない限り、システムがこれをインプリメントすることはできません。

情報

他の標準機構またはベンダーによって開発されたプロトコル、またはその他の理由で IAB の権限外になるプロトコルは、インターネット・コミュニティの便宜のために、情報のプロトコルとして、RFC として公開されることがあります。場合によっては、インターネット上での使用にこれらのプロトコルが IAB によって推奨されることもあります。

記録

これらは、以後の開発によって置き換えられたか利点がないために、インターネットの標準になる可能性がほとんどないプロトコルです。

プロトコルの状態は、以下のいずれかになります。

必須

システムはこの必須プロトコルをインプリメントする必要があります。

推奨

システムはこの推奨プロトコルをインプリメントする必要があります。

任意選択

システムはこの任意選択のプロトコルをインプリメントすることも、しないこともできます。一般にこのようなことを行う場合は、正確に行う必要があります。

限定使用

このプロトコルは限定された状況で使用されます。この理由は、その実験状態、特殊な性質、制限された機能、または記録状態などによります。

非推奨

このプロトコルは、一般的な使用には推奨されていません。この理由は、その制限された機能、特殊な性質、または実験か記録状態などによります。

1.3.2 インターネット標準

提案標準、ドラフト標準、および標準のプロトコルは、インターネットの標準化過程上にあるものとして説明されます。プロトコルが標準状態に達すると、それには標準番号 (STD) が割り当てられます。STD 番号の目的は、どの RFC がインターネット標準について記述しているのかを明確に示すことです。標準の仕様が複数の文書に及ぶものである場合、STD 番号は複数の RFC を指します。番号が特定の文書を指す RFC とは異なり、STD 番号は標準が更新されても変化しません。ただし、更新はすべて RFC を介して行われ、RFC 番号が固有であるため、STD 番号はバージョン番号を持ちません。したがって、標準のどのバージョンについて言及しているのかを明確に指定するには、標準番号とそれに含まれるすべての RFC を記述しなければなりません。たとえば、Domain Name System (DNS) は STD 13 で、RFC 1034 と 1035 で説明されています。標準を参照する場合は、「STD-13/RFC1034/RFC1035」といった形式を使用する必要があります。

標準化過程にあるいくつかの RFC には、状態カテゴリーに、有用な十分な情報が必ずしも含まれていません。したがって、これは STD 1 または別個の RFC で指定される適用度ステートメントにより、特にルーティング・プロトコルなどを補足するものです。

RFC および STD 番号はすべての TCP/IP プロトコル・インプリメンテーションの基礎を成すため、本書ではこれらがたびたび参照されます。

以下は、特に重要なインターネット標準です。

- STD 1 - Internet Official Protocol Standards

この標準は、各インターネットのプロトコルまたは標準の状態と状況を定め、各状態または状況に属する意味を定義しています。これは、IAB によっておおむね四半期ごとに発行されます。この標準は RFC 2800 にあります (本書執筆時)。

- STD 2 – Assigned Internet Numbers

この標準は、インターネット・プロトコル・スイートの中で現在割り当てられている番号とその他のプロトコル・パラメーターをリストしています。これは、Internet Assigned Numbers Authority (IANA) によって発行されます。現在の版は、RFC1700 です (本書執筆時)。

- STD 3 – Host Requirements

この標準は、インターネット・ホスト・ソフトウェアの要件を (多くの場合、関連する RFC の参照によって) 定義しています。この標準は、RFC1122 – Requirements for Internet hosts – communications、RFC1123 – Requirements for Internet hosts – application and support、および RFC 2181 – Clarifications to the DNS Specification の 3 つの部分に分かれています。

- STD 4 – Router Requirements

この標準は、IPv4 インターネット・ゲートウェイ (ルーター) ソフトウェアの要件を定義しています。これは、RFC 1812 – Requirements for IPv4 Routers で定義されています。

1.3.2.1 For Your Information (FYI)

インターネット・ユーザーの幅広い関心を引くことを意図して作成された RFC は、*For Your Information* (FYI) 文書として分類されます。これらには紹介情報や他の役立つ情報がよく記載されています。STD 番号と同様、FYI 番号は改訂版の RFC が発行されても変化しません。STD と異なる点は、FYI は単一の RFC 文書に対応するということです。たとえば、FYI 4 - FYI on Questions and Answers - Answers to Commonly asked "New Internet User" Questions は、現在その第 5 版にあたります。RFC 番号は 1177、1206、1325、および 1594、そして 2664 です。

1.3.2.2 RFC の入手

RFC および ID 文書はオンラインで公開されており、その入手源として最も適しているのは、以下に示す IETF の Web サイトです。

<http://www.ietf.org>

RFC 2800 – Internet Official Protocol Standards には、現在の Internet Standards の完全なリストがあります。

1.4 インターネットの将来

インターネットの将来について予言しようとするのは、簡単なことではありません。現在インターネットが仕事、家庭、および学校における毎日の生活の中でこれほどの位置を占めるとは、5年前でさえ、ほとんど誰も想像していなかったことでしょう。ただし、いくつかのことについてはかなりの確信ができます。

1.4.1 マルチメディア・アプリケーション

帯域幅要件は壮絶な速度で増大し続けるでしょう。インターネット・ユーザーの数が急速に増加しているだけでなく、使用されるアプリケーションもより高機能になっており、そのためにより多くの帯域幅を消費します。インターネットに置かれているこのような大きな帯域幅の需要を満たすため、Dense Wave Division Multiplexing (DWDM) などの新しいテクノロジーが出現しています。

この増大する需要の多くは、マルチメディア・アプリケーションの使用の増加に起因しています。その1つの例が、Voice over IP テクノロジーの使用の増加です。このテクノロジーが成熟するにつれ、インターネット上の音声とデータの間で帯域幅が共用されることはほぼ確実になっています。これに伴って、電話会社に関するいくつかの興味深い問題が持ち上がります。ユーザーが支出する、ノースカロライナ州のラーリーとペルーのリマを結ぶインターネット接続の費用は、ラーリー内での接続の費用と同じになります。もちろん従来の電話接続ではそうはいきません。確実なこととして、電話での通話は画像付き会議になり、音声電話もテレビ電話になるでしょう。

今日では、FM 品質を備えたインターネットを介してほぼ世界中のラジオ局を視聴できます。私たちは、世界中のテレビ・チャンネルが、ムービーその他のあらゆるビデオ信号を、あらゆる場所の消費者に送達するための媒体として、インターネットを使用しつつあることを見ることができます。しかし、インターネットのインフラストラクチャーがこのような大きな帯域幅の需要に合わせなければならないため、これには相当な費用がかかります。

1.4.2 営利的な使用

インターネットの商業的な使用は、爆発的に増加しています。今日、マーケティング、セールス、顧客サービスまたは従業員のアクセスのいずれにおいても、大規模なビジネスのほとんどすべてはインターネットに依存しています。この傾向は続いていくものと予測されます。電子ストアは、従来の店に足を運ぶ時間のない顧客に便利さを提供することによって、引き続き繁栄していくことでしょう。

企業は世界中の支部と通信するための手段として、ますますインターネットに頼るようになります。Virtual Private Networks (VPNs) の普及により、企業はインターネットを使用して幅広い領域で安全にその内部業務を実行することができます。従業員は、バーチャル・オフィス環境を生み出すホーム・オフィスから作業することができます。バーチャル会議は、おそらく一般的なこととなるでしょう。

1.4.3 ワイヤレス・インターネット

しかし、インターネットの使用における最も広範な発展は、おそらくワイヤレス・アプリケーションの発展でしょう。最近では、ワイヤレスおよびパーベイシブ・コンピューティングの使用可能性が大きな焦点となってきました。このように注目を集めた理由は、主にワイヤレス接続の簡便性にありました。たとえば、自由にローミングできると定義されているモバイル・ワークステーションを、物理的に接続することは実際的ではありません。そのようなワークステーションを一部の物理地域に束縛するなら、その目的を無にすることになります。その他のケースとして、有線接続は単純に実現不能という場合もあります。この例には、マチュピチュの廃墟やシスティン大聖堂内のオフィスが含まれます。この場合も、固定ワークステーションは、ワイヤレスでなければ利用できないネットワーク・アクセスから益を得ることができます。

Bluetooth、IEEE 802.11、Wireless Application Protocol (WAP) などのプロトコルは、ワイヤレス・インターネットへの道を切り開いています。このアクセスでの個人の利益は非常に大きい一方、さらに魅力的なのは、このテクノロジーによって簡易化されるビジネス・アプリケーションです。工場から病院までのすべてのビジネスが、その個々のサービスを拡張できる可能性があります。ワイヤレス装置は、ドライバーの個人的な楽しみのためだけでなく、なじみの自動車整備士にメンテナンス情報を送るための、車両の標準装置になるでしょう。その応用は無限です。

第 2 章 ネットワーク・インターフェース

この章では、TCP/IP トラフィックがさまざまな種類の物理ネットワーク上でフローできるようにするプロトコルおよびインターフェースについて概説します。インターネットワーク・プロトコル・スイートである TCP/IP は、多数の物理ネットワーク上で操作することができます。このプロトコルのうち、最も一般的で広く使用されているものは、もちろんイーサネットです。IP を固有にサポートするための規定を設けているいくつかのネットワーク・プロトコルは、明らかにこのレッドブックの対象範囲外になります。ただし、ここでは、TCP/IP と共に最も一般に使用されている種々のネットワークのいくつかについて要約します。

2.1 イーサネットおよび IEEE 802.x ローカル・エリア・ネットワーク (LAN)

イーサネット同軸ケーブル上では、以下の 2 つのフレーム・フォーマットを使用することができます。

1. Xerox Corporation、Intel Corporation、および Digital Equipment Corporation によって 1978 年に発行された、通常イーサネット (または DIX イーサネット) と呼ばれる標準。
2. より最近定義された標準である、国際的な IEEE 802.3 標準。

詳細については、図 6 を参照してください。

2 つの標準の間の違いは、ある 1 つのヘッダー・フィールドの使用法にあります。このヘッダー・フィールドには、イーサネットの場合はプロトコル・タイプ番号が、また IEEE 802.3 の場合はフレーム内のデータの長さが含まれます。

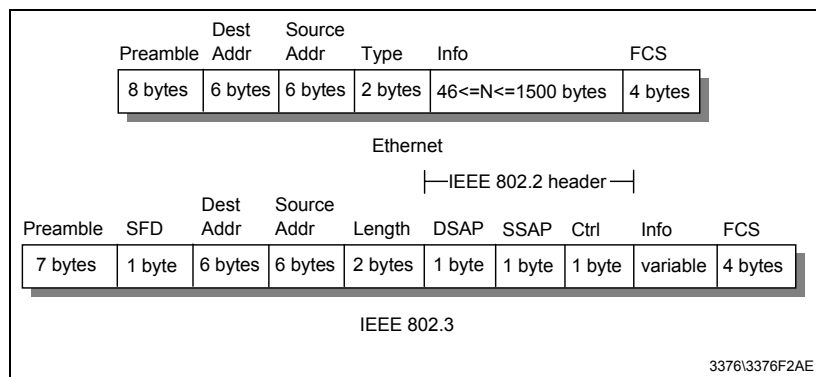


図 6 ARP - イーサネットおよび IEEE 802.3 のフレーム・フォーマット

- イーサネットのタイプ・フィールドは、同軸ケーブル上で実行する種々のプロトコルを区別するために使用され、同一の物理ケーブル上でのそれらの共存を可能にします。
- イーサネット・フレームの最大長は 1526 バイトです。これは、データ・フィールド長が最大 1500 バイトであることを意味します。802.3 データ・フィールドの長さも、10 Mbps ネットワークでは 1500 バイトまでに制限されますが、他の伝送速度の場合は異なります。
- 802.3 MAC フレームでは、データ・フィールドの長さは 802.3 ヘッダー内で示されます。これが伝送するプロトコルのタイプは、802.2 ヘッダー (上位プロトコル・レベル。図 6 を参照) 内で示されます。しかし、実際には、両方のフレーム・フォーマットが同じ物理同軸上に共存可能です。これは、イーサネット・フレームの 1500 より大きなプロトコル・タイプ番号 (タイプ・フィールド) を使用して行われます。ただし、これらのフォーマットそれぞれの処理に、異なるデバイス・ドライバーが必要です。

したがって、実際的な理由から、イーサネットの物理層と IEEE 802.3 の物理層には互換性があります。ただし、イーサネットのデータ・リンク層と IEEE 802.3/802.2 のデータ・リンク層には非互換です。

IEEE 802.3 の上の 802.2 論理リンク制御 (LLC) 層は、リンク・サービス・アクセス・ポイント (LSAP) として知られる概念を使用します。これは 3 バイトのヘッダーを使用し、その DSAP と SSAP とは、それぞれ宛先 (Destination) サービス・アクセス・ポイントと送信元 (Source) サービス・アクセス・ポイントを表します。これらのフィールドの番号は、IEEE 委員会によって割り当てられています (図 7 を参照)。

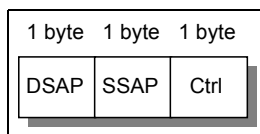


図 7 ARP - IEEE 802.2 LSAP ヘッダー

IEEE 802 を下位プロトコル層として使用するアプリケーションの数の増加のため、Sub-Network Access Protocol (SNAP) の形式での拡張が IEEE 802.2 プロトコルに加えられました (図 8 を参照)。これは図 7 の LSAP ヘッダーに対する拡張であり、この使用は図 7 の LSAP フレームの SSAP と DSAP の両フィールド内の値 170 によって示されます。

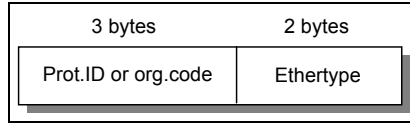


図 8 ARP - IEEE 802.2 SNAP ヘッダー

TCP/IP の発展の中で、これらのネットワークでの IP および ARP フレームのカプセル化について説明する 3 つの標準が確立されました。

1. 1984 年に紹介された、RFC 894 – Standard for the Transmission of IP Datagrams over Ethernet Networks は、イーサネット・タイプのネットワークの使用だけを指定しています。タイプ・フィールドに割り当てられた値は、以下のとおりです。
 - a. IP データグラム用に 2048 (16 進 0800)
 - b. ARP データグラム用に 2054 (16 進 0806)
2. 1985 年に紹介された、RFC 948 – Two Methods for the Transmission of IP Datagrams over IEEE 802.3 Networks は、以下の 2 つの可能性を指定しています。
 - a. イーサネット互換方式：フレームは、実際の IEEE 802.3 ネットワーク上でイーサネット・ネットワーク上と同じ方法で、つまり、IEEE 802.3 のデータ長フィールドをイーサネットのタイプ・フィールドとして使用して (したがって、IEEE 802.3 の規則に違反しますが、イーサネット・ネットワークとは互換的に) 送信されます。
 - b. IEEE 802.2/802.3 LLC type 1 フォーマット：IP と共に 802.2 LSAP ヘッダーを使用し、SSAP および DSAP フィールドに値 6 を使用します。

この RFC は、IEEE 802.2/802.3 方式が好ましい方式であり、IEEE 802.3 ネットワークにおける将来のすべての IP インプリメンテーションが 2 番目の方式を使用することになっていることを明確に示しています。

3. 1987 年に紹介された、RFC 1010 – Assigned Numbers (現在は 1994 年付けの RFC 1700 によって廃止されている) は、IEEE 802.2 の発展とより多くのインターネット・プロトコル番号の必要のため、1986 年 8 月の TCP ベンダー・ワークショップで交換された実体験を基に新しいアプローチが開発されたことを示しています。これは、この RFC のほぼ完全に見落とされる部分の中で、すべての IEEE 802.3、802.4、および 802.5 のインプリメンテーションが、DSAP および SSAP フィールドが 170 (SNAP の使用を示す) に設定され、SNAP が以下のように割り当てられた、IEEE 802.2 LLC の Sub-Network Access Protocol (SNAP) 形式を使用する必要があると述べています。
 - a. 0 (ゼロ) を組織コードとして。

b. EtherType フィールド：

1. IP データグラム用に 2048 (16 進 0800)
2. ARP データグラム用に 2054 (16 進 0806)
3. RARP データグラム用に 32821 (16 進 8035)

これらは、イーサネットのタイプ・フィールドで使用されるのと同じ値です。

4. 1988 年には、RFC 1042 – Standard for the Transmission of IP Datagrams over IEEE 802 Networks が紹介されました。この新しいアプローチ（インプリメンテーションに非常に重要）は、無関係な RFC の小さな注記の中にあってほとんど気付かれずにいたために、非常に紛らわしいものとなり、結局 1988 年 2 月に、その独自の RFC (RFC 948 を廃止する RFC 1042) の中で繰り返されることになりました。

関連する IBM TCP/IP 製品は、DIX イーサネット・ネットワーク用の RFC 894 と IEEE 802.3 ネットワーク用の RFC 1700 をインプリメントしています。

ただし、現状では、古い LSAP 方式 (RFC 948 または 1042) を使用する TCP/IP インプリメンテーションがまだ存在しています。このようなインプリメンテーションは、より最新のインプリメンテーション (IBM のものなど) とは通信しません。

また、最後の方式は、IEEE 802.3 ネットワークだけでなく、IBM トークンリング LAN などの IEEE 802.4 および 802.5 ネットワークもカバーする点にご注意ください。

2.2 Fiber Distributed Data Interface (FDDI)

FDDI 仕様は、ISO/OSI モデルで定義されている、物理層とデータ・リンク層のメディア・アクセス制御副層を提供する、100 Mbps 光ファイバー LAN 用の標準ファミリーを定義しています。

IP-FDDI はドラフト標準プロトコルです。その状態は「任意選択 (elective)」になっています。これは、FDDI フレーム内の IP データグラムと ARP 要求および応答のカプセル化を定義しています。図 9 は、関連したプロトコル層を示しています。

これは、RFC 1188 – A Proposed Standard for the Transmission of IP Datagrams over FDDI Networks for single MAC stations で定義されています。二重 MAC 端末での操作については、今後の RFC で説明されます。

RFC 1188 には、すべてのフレームが、802.2 ヘッダーの DSAP および SSAP フィールドが SNAP 用に割り当てられたグローバル SAP 値 (10 進 170) に設定された、標準の IEEE 802.2 LLC Type 1 Unnumbered Information フォーマットで伝送されることが記されています。SNAP ヘッダー内の 24 ビットの組織コードはゼロに設定されており、残りの 16 ビットは Assigned Numbers (RFC 1700 を参照) の EtherType で、それは以下のとおりです。

- 2048 (IP)
- 2054 (ARP)

32 ビットの IP アドレスから 48 ビットの FDDI アドレスへのマッピングは、ARP ダイナミック・ディスカバリー・プロシーチャーを介して行われます。ブロードキャスト IP アドレス (<host address> がすべてゼロに設定されている) は、ブロードキャスト FDDI アドレス (すべてゼロ) にマップされます。

IP データグラムは、ビッグ・エンディアンまたはネットワーク・バイト順序と呼ばれる通常の TCP/IP 伝送順序を使用して、一続きの 8 ビット・バイトとして伝送されます。

FDDI MAC 仕様 (ISO 9314-2 - ISO, Fiber Distributed Data Interface - Media Access Control) は、すべてのフレーム・フィールドに対して 4500 バイトの最大フレーム・サイズを定義しています。LLC/SNAP ヘッダーを考慮に入れてから、MAC ヘッダーおよびフレーム状況フィールドに対する将来の拡張を可能にするため、FDDI ネットワークの MTU は 4352 バイトに設定されています。

FDDI アーキテクチャーの詳細については、*LAN Concepts and Products* (SG24-4753) を参照してください。

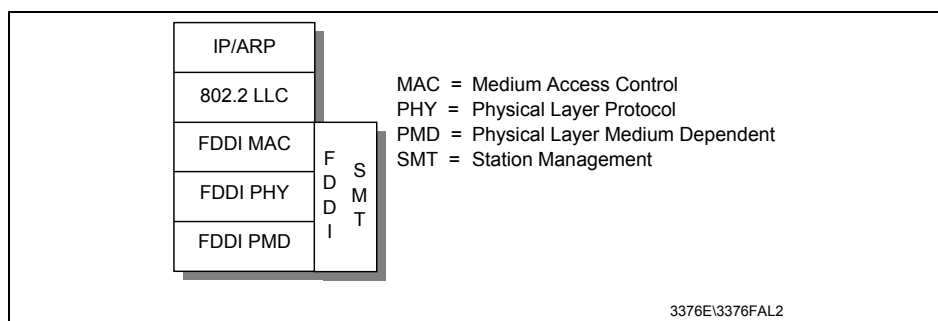


図 9 FDDI における IP および ARP

2.3 Serial Line IP (SLIP)

TCP/IP プロトコル・ファミリーは、IEEE 802.3 および 802.5 LAN、X.25 回線、サテライト・リンク、およびシリアル回線といった多様なネットワーク・メディア上で実行します。IP パケットのカプセル化の標準はこれらのネットワークの多くに対して定義されてきましたが、シリアル回線用の標準はありません。SLIP は現在、デファクト・スタンダードであり、TCP/IP を実行する Point-to-Point シリアル接続に一般に使用されています。SLIP はインターネット標準ではありませんが、RFC 1055 で文書化されています。

SLIP は、かなり昔に設計された単なる非常に単純なプロトコルで、ただのパケット・フレーム・プロトコルです。これは、シリアル回線上の IP パケットを構成する一連の文字を定義しているだけです。これは、以下のいずれをも提供しません。

- アドレッシング：SLIP リンクの両端のコンピューターは、ルーティングのために互いの IP アドレスを認識している必要があります。SLIP はカプセル化プロトコルのみを定義し、いかなる形式のハンドシェークまたはリンク制御も定義していません。リンクは、IP アドレスの指定も含めて、手動で接続および構成されます。
- パケット・タイプの識別：SLIP は単一のリンク上で複数のプロトコルをサポートできません。したがって、SLIP 接続上では 1 つのプロトコルしか実行できません。
- エラー検出 / 修正：SLIP はいかなる形式のフレーム・エラー検出も行いません。上位レベルのプロトコルは、ノイズが発生する回線上のエラーによって生じる破損パケットを検出する必要があります。(IP ヘッダーと UDP/TCP チェックサムが十分でなければなりません。) 変更されたパケットの再伝送には時間がかかるため、SLIP がある種の単純なエラー修正メカニズムを独自に提供することができれば、これは効率的になります。
- 圧縮：SLIP は、頻繁に使用される IP ヘッダー・フィールドを圧縮するためのメカニズムを提供していません。低速シリアル・リンク上の多くのアプリケーションは、TELNET のような単一ユーザー対話式 TCP トラフィックになる傾向があります。これにはよく、パケット・サイズが小さくなり、データグラム間でそれほど変わらない TCP および IP ヘッダー内のオーバーヘッドが相対的に大きくなることが伴いますが、これは対話式の応答時間に著しい悪影響を及ぼす可能性があります。

ただし、多くの SLIP インプリメンテーションは現在 *Van Jacobson Header Compression* を使用しています。これは、リンクの各終端の TCP 接続のセットの状態を記録し、セッションの連続する IP データグラム間で多くのフィールドが未変更か少量ずつ増分している通常の場合に、全ヘッダーをエンコードされた更新で置き換えることによって、IP および

TCP ヘッダーの結合サイズを 40 バイトから 8 バイトに削減するために使用されます。この圧縮は、RFC 1144 で説明しています。

SLIP プロトコルは、基本的に Point-to-Point Protocol (PPP) によって置き換えられています。35 ページの 2.4『Point-to-Point Protocol (PPP)』を参照してください。

2.4 Point-to-Point Protocol (PPP)

PPP は、STD 番号 51 を持つネットワーク固有の標準プロトコルです。その状態は「任意選択 (elective)」になっています。これは、RFC 1661 および RFC 1662 で説明しています。

さまざまな種類の Point-to-Point リンク上での PPP の操作を指定する、多数の *提案標準プロトコル* があります。それぞれの状態は「任意選択 (elective)」です。STD 1 – Internet Official Protocol Standards にある、標準化過程にある PPP 関連の RFC のリストを参照することをお勧めします。

非同期および同期回線の形式の Point-to-Point 回路は、長い間データ通信の要でした。TCP/IP の世界では、デファクト・スタンダードである SLIP プロトコルがこの分野で見事に役目を果たしてきており、現在でもダイヤルアップ TCP/IP 接続に広く使用されています。ただし、SLIP にはいくつかの欠点があり、Point-to-Point Protocol はそれに対する処置を講じています。

PPP には、以下の 3 つの主要なコンポーネントがあります。

1. シリアル・リンク上でのデータグラムのカプセル化のためのメソッド。
2. データ・リンク接続の確立、構成、および検査のための *Link Control Protocol (LCP)*。
3. 種々のネットワーク層プロトコルの確立および構成のための *Network Control Protocol (NCP)* のファミリー。PPP は、複数のネットワーク層プロトコルが同時に使用できるように設計されてます。

リンクがネットワーク層プロトコルによって使用可能であると見なされるまでに、特定の順序のイベントが発生する必要があります。LCP は、接続の確立、構成、保守、および終了のための方式を提供しています。LCP は以下のフェーズを通過します。

1. リンク確立と構成ネゴシエーション：このフェーズでは、リンク制御パケットが交換され、リンク構成オプションがネゴシエーションされます。オプションが合意されると、リンクが開かれますが、必ずしもネットワーク層プロトコルを開始するための準備が完了というわけではありません。

2. リンク品質の判別：このフェーズはオプションです。PPP は品質を判別するためのポリシーを指定していませんが、エコー要求および応答などの低レベル・ツールは提供しています。
3. 認証：このフェーズはオプションです。リンクの各終端は、フェーズ 1 で合意された認証方式を使用して、自身をリモート・エンドに認証します。
4. ネットワーク層プロトコル構成のネゴシエーション：LCP が前のフェーズを終了したら、適切な NCP によってネットワーク層プロトコルを別個に構成することができます。
5. リンクの終了：LCP は、いつでもリンクを終了することができます。これは、通常は人間のユーザーの要求時に行われますが、物理イベントが原因で発生することもあります。

2.4.1 Point-to-Point カプセル化

図 10 では、PPP カプセル化の要約を示しています。

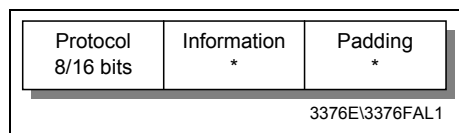


図 10 PPP カプセル化フレーム

- プロトコル・フィールド：プロトコル・フィールドは 1 または 2 オクテットで、その値はパケットの情報フィールド内のカプセル化されたデータグラムを識別します。プロトコル・フィールドの最新の値は、RFC 1700 – Assigned Numbers で指定されています。
- 情報フィールド：情報フィールドは 0 オクテット以上です。情報フィールドには、プロトコル・フィールドで指定されたプロトコルに対応するデータグラムが含まれます。埋め込みを含み、プロトコル・フィールドは含まない情報フィールドの最大長は最大受信単位 (MRU) と呼ばれ、これはデフォルトで 1500 オクテットです。MRU には、ネゴシエーションによって他の値を使用することができます。
- 埋め込み：伝送の際、情報フィールドは MRU 以内の任意の数のオクテットで埋め込まれることがあります。実際の情報と埋め込みオクテットの区別は、各プロトコルが担当します。

IP Control Protocol (IPCP) は IP 用の NCP で、Point-to-Point リンクの両終端での IP プロトコルの構成、使用可能化、および使用不能化を担当します。IPCP のオプションのネゴシエーション順序は LCP の場合と同じで、コードが再利用できる可能性があります。

IPCP と共に使用される 1 つの重要なオプションは、Van Jacobson Header Compression です。これは、リンクの各終端の TCP 接続のセットの状態を記録し、セッションの連続する IP データグラムの中で多くのフィールドが未変更か少量ずつ増分しているような通常の場合に、全ヘッダーをエンコードされた更新で置き換えることにより、IP および TCP ヘッダーの結合サイズを 40 バイトからおよそ 4 バイトに削減するために使用されます。この圧縮は、RFC 1144 で説明しています。

2.5 サービス総合ディジタル網 (ISDN)

このセクションでは、ISDN Point-to-Point リンク上での PPP カプセル化の使用法について説明します。ISDN における PPP については、RFC 1618 で文書化されています。その状態は「任意選択 (elective)」になっています。ISDN B チャネルは定義上では Point-to-Point 回路であるため、PPP はこれらのリンク上での使用によく適しています。

ISDN 基本速度インターフェース (BRI) は通常、それぞれが 64 kbps の容量を持つ 2 つの B チャネルと、制御情報用の 16 kbps の D チャネルをサポートします。B チャネルは、音声またはデータ、あるいは単に混合のデータに対して使用することができます。

ISDN 1 次速度インターフェース (PRI) は、多数の同時 B チャネル・リンク (通常 30) と 1 つの 64 Kbps D チャネルをサポートできます。PPP の LCP および NCP メカニズムは、手動構成を削減または除去し、多様なインプリメンテーション間の通信の容易性を促進するために、この状況では特に役立ちます。ISDN D チャネルも、適切にフレーム化されている場合の PPP パケットの送信に使用できますが、これは帯域幅において制限され、しばしば通信リンクをローカル・スイッチまでに制限します。

PPP は ISDN チャネルをビットまたはオクテット指向の同期リンクとして取り扱います。このリンクは全二重でなければなりませんが、専用か回線交換のいずれかにすることができます。PPP は、物理層にオクテット・インターフェースを提供します。サブオクテットを供給または受信するための措置はありません。PPP は、ある特定の ISDN チャネル・インターフェースの速度を除き、伝送速度に関していかなる制限も課しません。PPP は制御信号の使用を必要としません。使用可能な場合、この信号を使用すれば、機能性とパフォーマンスが向上する可能性があります。

種々のエンコードとスクランブルの定義は、使用中の DTE/DCE 機器が担当します。PPP はビット・ストリームの基礎表記とは無関係に動作しますが、伝送のための標準がないことは、データ・リンク標準がないのと同じくらい確実に相互運用性を妨害します。D チャネル・インターフェースには NRZ エンコードが必要です。したがって、B チャネル・インターフェースにおいて

も NRZ を使用することをお勧めします。そうすれば、B チャンネルと D チャンネルの間でフレームを容易に交換することができます。ただし、エンコードの構成が可能な場合は、クリアな B チャンネル上で必要なときに最小の 1 の密集状態を保証するため、NRZI を代わりに使用することをお勧めします。複数のエンコードと相互操作するインプリメンテーションは、これらのエンコードを自動的に検出するように選択できます。自動エンコード検出は、大規模な事前構成を避けるために 1 次速度インターフェース (PRI) には特に重要です。

V.120¹ に準拠する端末アダプターは、ワークステーションに対する単純なインターフェースとして使用できます。端末アダプターは、非同期 - 同期の変換を提供します。複数の B チャンネルを並列に使用することが可能です。V.120 は、オクテット詰め - ビット詰めの変換を提供していないため、V.120 はビット同期リンクと相互操作できません。実際には HDLC、LAPB、LAPD、および LAPF は通常は区別可能ですが、複数のフレーム化方式を同じ ISDN チャンネル上で並行して使用するべきではありません。PPP が代替のフレーム化技法を認識したり、特定の構成なしにフレーム化技法を切り替えたりする必要はありません。これまでの経緯は、LLC 情報エレメントが終端から終端まで確実に伝送されないことを示しています。したがって、フレーム化またはエンコードの判別で、LLC-IE の伝送に依存すべきではありません。PPP に関連した LLC-IE 値は割り当てられていません。受信される他の値はどれも PPP リンクには無効であり、PPP サービスでは無視することができます。LCP の推奨の同期構成オプションは ISDN リンクに適用されます。標準の LCP 同期構成のデフォルトは ISDN リンクに適用されます。このリンクに接続される典型的なネットワークは、1500 または 2048 バイト以上のいずれかの MRU サイズを持つ可能性があります。フラグメント化を避けるため、ネットワーク層の最大伝送単位 (MTU) は、2048 以上のピア MRU が特にネゴシエーションされていない限り、1500 を超えるべきではありません。

2.6 X.25

このトピックでは、IP over X.25 ネットワークのカプセル化について、ISO/IEC および CCITT 標準に従って説明します。IP over X.25 ネットワークについては、RFC 877 を廃止する RFC 1356 で文書化されています。RFC 1356 は、状態が「任意選択 (Elective)」であるドラフト標準です。X.25 での IP カプセル化の実質的な変更点は、IP データグラムの MTU サイズの増加、X.25 の最大データ・パケット・サイズ、バーチャル・サーキット管理、および、マルチプロトコル・ルーターとブリッジとの間での、IP 以外の X.25 のプロトコルの相互操作可能なカプセル化です。

伝送のためにデータグラムがネットワーク・インターフェースに着信すると、オンデマンドで 1 つ以上の X.25 バーチャル・サーキットが開きます。プロト

¹ CCITT Recommendations I.465 and V.120, "Data Terminal Equipment Communications over the Telephone Network with Provision for Statistical Multiplexing", CCITT Blue Book, Volume VIII, Fascicle VIII.1, 1988.

コル・データ単位 (PDU) が、X.25 完全パケット・シーケンスとして送信されます。つまり、PDU は X.25 データ・パケット境界上で始まり、1 つの X.25 データ・パケットより長い PDU のフラグメント化には M ビット (追加データ) が使用されます。IP カプセル化では、PDU は IP データグラムです。呼び出し利用者データ (CUD) フィールド内の最初のオクテット (呼び出し要求パケット内の最初のデータ・オクテット) は、ISO/IEC TR 9577 の後続プロトコル ID (SPI) に応じて、プロトコルの逆多重化に使用されます。このフィールドには、X.25 バーチャル・サーキット上でカプセル化されているネットワーク層プロトコルを識別する、1 オクテットのネットワーク層プロトコル ID (NLPID) が入ります。インターネット・コミュニティに対し、NLPID は以下の 4 つの関連値を持ちます。

1. 値、16 進 CC (2 進 11001100、10 進 204) は、IP です。
2. 値、16 進 81 (2 進 10000001、10 進 129) は、ISO/IEC 8473 (CLNP) を識別します。
3. 値、16 進 82 (2 進 10000010、10 進 130) は、特に ISO/IEC 9542 (ES-IS) に使用されます。CLNP を伝送するために開かれた回路がすでにある場合は、ES-IS を伝送するために 2 つ目の回路を開く必要はありません。
4. 値、16 進 80 (2 進 10000000、10 進 128) は、単一のネットワーク層プロトコルをさらにカプセル化して識別するための IEEE Subnetwork Access Protocol (SNAP) の使用を識別します。SNAP でカプセル化されたプロトコルは、呼び出し要求 CUD フィールドの 16 進 80 オクテットの直後に 5 オクテットの SNAP ヘッダーが組み込まれていることによって識別されます。SNAP ヘッダーは後続の X.25 データ・パケットには組み込まれません。このエンコードを使用して開かれたバーチャル・サーキット上で運ばれ得る、SNAP でカプセル化されたプロトコルは 1 つだけです。

値、16 進 00 は、同じ回路上の複数のネットワーク層プロトコルを多重化するために使用されるヌル・カプセル化を識別します。RFC 1700 には、インターネットの X.25 カプセル化識別に使用される、もう 1 つの非 CCITT および非 ISO/IEC 値、すなわち Blacker (Blacker front end) X.25 用の 16 進 C5 (2 進 11000101、10 進 197) が含まれています。この値は引き続き使用される可能性があります、それは、この値をサポートする送信および受信 X.25 インターフェースの以前の事前構成によってのみです。RFC 1700 にリストされている ISO-IP 用の 16 進 CD (2 進 11001101、10 進 205) も Blacker によって使用され、送信および受信 X.25 インターフェースの以前の事前構成によってのみ使用可能です。

各システムは、処理できるプロトコルの呼び出しのみを受信しなければなりません。すべてのインターネット・システムは、IP データグラムの CC カプセル化を受信できなければなりません。16 進 CC (IP 用) 以外の NLPID をサポートするシステムは、その使用をピア・アドレスごとの基準で構成する必要があります。16 進 00 の NLPID エンコードによって識別されるヌル・カプ

セル化は、1つの回路上で複数のネットワーク層プロトコルを多重化するために使用されます。ヌル・カプセル化が使用されている場合、回路上で送信される各 X.25 完全パケット・シーケンスは、その特定の完全パケット・シーケンスだけに含まれるネットワーク層プロトコル・データ単位を識別する、1 オクテットの NLPID で始まります。さらに、SNAP の NLPID (16 進 80) が使用されている場合は、NLPID オクテットの直後に 5 オクテットの SNAP ヘッダーが続き、その直後にカプセル化された PDU が続きます。カプセル化されたネットワーク層プロトコルは、同じ回路上の、ある完全パケット・シーケンスとその次の完全パケット・シーケンスとで異なる可能性があります。

システムが限定された数のプロトコルだけをサポートしている場合は、単一ネットワーク層プロトコル回路を使用する方が帯域幅の点でより効率的です。またそうすれば、各システムはその通信パートナーによってサポートされているプロトコルを正確に判別することができます。他の利点には、各プロトコルと種々のサービスの品質を詳述する X.25 アカウンティングや、さまざまなプロトコル用のフロー制御ウィンドウを使用できることが含まれます。多重化のためのヌル・カプセル化は、システムが何らかの理由（たとえば、インプリメンテーションの制約やネットワーク・コストの考慮事項など）で、限定された数のバーチャル・サーキットしか同時にオープンできない場合に役立ちます。これは、妥当な範囲を超えた数のバーチャル・サーキットの使用を避けるために、マルチプロトコル・ルーターによって使用される可能性が最も高い方式です。X.25 での IEEE 802.1d ブリッジの実行が必要な場合は、ヌル・カプセル化を使用する必要があります。

IP データグラムは、デフォルトでは、CC CUD で開かれたバーチャル・サーキット上でカプセル化されます。インプリメンテーションは、さらに以下の実行可能な最大 3 つの IP のカプセル化をサポートできます。

- IP データグラムは、ヌル・カプセル化を使用する回路上で、多重化されたデータ・パケットの中に含めることができます。このようなデータ・パケットは、16 進 CC の NLPID によって識別されます。
- IP を、回路上の SNAP カプセル化内にカプセル化できます。このカプセル化は、5 オクテットの SNAP ヘッダーである 16 進 00-00-00 の組織固有 ID (OUI) と、16 進 08-00 のプロトコル ID (PID) が組み込まれていることによって識別されます。
- ヌル・カプセル化を使用する回路上の、多重化されたデータ・パケット内の IP の SNAP カプセル化内に IP を含めることができます。

2.7 フレーム・リレー

フレーム・リレー・ネットワークは、同じフレーム・リレー・ネットワークに接続する端末間の接続の基礎となるいくつかのバーチャル・サーキットを提供します。結果として生じる相互接続された装置の集合は、専用フレーム・

リレー・グループを形成します。これは、バーチャル・サーキットのすべての網と完全に相互接続されている場合もあれば、一部だけが相互接続されている場合もあります。どちらの場合にも、各バーチャル・サーキットは各フレーム・リレー・インターフェースで、データ・リンク接続 ID (DLCI) によって固有に識別されます。たいていの場合、DLCI は各フレーム・リレー・インターフェースで厳密なローカルでの意味を持ちます。フレーム・リレーについては、RFC 2427 で文書化されています。

2.7.1 フレーム・フォーマット

すべてのプロトコルは、そのパケットを Q.922 Annex A フレーム {1} の中にカプセル化しなければなりません。さらに、フレームにはプロトコル・データ単位 (PDU) 内で伝送されるプロトコルを識別するために必要な情報が含まれており、それによって受信側は着信パケットを適切に処理できます (図 11 を参照)。フォーマットは以下のようになります。

- ・制御フィールドは Q.922 制御フィールドです。UI (0x03) 値は、使用しないようにネゴシエーションされていない限り、使用されます。XID (0xAF または 0xBF) の使用は、許可されています。
- ・埋め込みフィールドは、フレームの (カプセル化ヘッダーを超えた) データ部分を 2 オクテット境界に位置合わせするために使用されます。存在する場合、埋め込みは 1 オクテットであり、値は 0 でなければなりません。
- ・ネットワーク・レベル・プロトコル ID (NLPID) フィールドは、ISO と ITU によって管理されます。これには、IP、CLNP、および IEEE Subnetwork Access Protocol (SNAP) を含む多くのさまざまなプロトコルの値が入ります。このフィールドは、どんなカプセル化やプロトコルが後に続くのかを受信側に示します。このフィールドの値は ISO/IEC TR 9577 {2} で定義されています。NLPID 値 0x00 は、ISO/IEC TR 9577 内でヌル・ネットワーク層または非アクティブ・セットとして定義されています。これは埋め込みフィールドと区別できず、このカプセル化体系のコンテキスト内では意味を持たないため、NLPID 値 0x00 はフレーム・リレーのカプセル化においては無効です。

フレーム・リレーには、共通にインプリメントされる最小または最大フレーム・サイズは存在しません。ただし、ネットワークは最低でも 262 オクテットの最大限度をサポートしている必要があります。通常、最大限度は 1600 オクテット以上ですが、各フレーム・リレー提供者はそのネットワークに適した値を指定します。フレーム・リレー・データ端末装置 (DTE) では、最大許容フレーム・サイズが構成可能でなければなりません。

Flag (7E Hexadecimal)
Q.922 Address*
Control (UI = 0x03)
Pad (When Required) (0x00)
NLPID
Data
Frame Check Sequence
(Two Octets)
Flag (7E Hexadecimal)

図 11 フレーム・リレーの packets フォーマット

2.7.2 相互接続の問題

フレーム・リレー・ネットワーク内を通過するデータ・パケットには、ルート・パケットとブリッジ・パケットという、2つの基本的なタイプが存在します。これらのパケットは明確なフォーマットを持ち、宛先がフレームの内容を正しく解釈するために使用できる標識が含まれている必要があります。この標識は、NLPID および SNAP ヘッダー情報内に組み込まれています。

2.7.3 データ・リンク層パラメーターのネゴシエーション

フレーム・リレー端末は、Q.922 {1} の Appendix III で指定されている交換 ID (XID) をサポートすることを選択できます。この XID 交換を使用すれば、フレーム・リレー回路の初期化時に、パラメーター (最大フレーム・サイズ、再送タイマー、および未解決情報 (I) フレームの最大数) のネゴシエーションが可能になります。

この交換が使用されない場合、これらの値はデータ・リンク接続 (DLC) エンドポイントの相互同意によって静的に構成されるか、または Q.922 {1} の Section 5.9 で指定されているデフォルトの値になります。

フレーム・リレー端末が、永続バーチャル・サーキット (PVC) 上のプロトコル・アドレスを動的に解決できる場合があります。これは、SNAP エンコードのフレーム・リレー・パケット内でカプセル化された、標準の Address Resolution Protocol (ARP) を使用して達成できます。

フレーム・リレー環境でのブロードキャストのエミュレーションの効率が悪いいため、新しいアドレス解決のバリエーションが開発されました。これは Inverse ARP と呼ばれ、ハードウェア・アドレスが既知の場合にプロトコル・アドレスを解決するための方式について記述しています。フレーム・リレー・ネットワークにおいて、既知のハードウェア・アドレスとは DLCI です。この仕様をインプリメントするための Inverse ARP のサポートは必須ではありませんが、フレーム・リレー・インターフェースの自動構成に役立つことが証明されています。

端末は同じ IP サブネット内の 1 つ以上の IP アドレスを、フレーム・リレー・インターフェース上の特定の DLCI にマップできなければなりません。この必要は、同じ DLC 上の帯域幅を共有しながらも、それぞれに固有な IP アドレスが割り当てられた多数のダイヤルイン・クライアントの ARP プロキシとしてサーバーが機能しなければならない、リモート・アクセスなどのアプリケーションで生じます。このようなアプリケーションの動的な性質は、DLC の状況には影響しないアドレス・アソシエーション変更を頻繁に引き起こします。

ARP を利用する他のインターフェースと同様、端末は、DLC に到着する非送信請求の (無償の) ARP 要求を処理することによって、IP アドレスと DLCI の間のアソシエーションを知ることができます。一方の端末がフレーム・リレー DLC の他方の終端にいるそのピア端末に、IP アドレスとその PVC の間の新しいアソシエーションについて通知することを望む場合は、送信元 IP アドレスが宛先 IP アドレスと等しく、その両方が DLC 上で使用されている新しい IP アドレスに設定された非送信請求 ARP 要求を送信する必要があります。こうすることによって、端末は特定の DLCI 上の新しいクライアント接続を「公表する」ことができます。受信側端末は新しいアソシエーションを保管し、必要であればインターフェース上の他の DLCI から古い既存のアソシエーションを除去する必要があります。

2.7.4 フレーム・リレーにおける IP

フレーム・リレー・ネットワークで送信されるインターネット・プロトコル (IP) データグラムは、前述のカプセル化に準拠します。このコンテキストの範囲内では、IP は 2 つの異なる方法 (IP を示す NLPID 値、または SNAP を示す NLPID 値) でカプセル化できます。

これらのカプセル化は両方とも、特定の定義の下でサポートされますが、IP データのカプセル化には、適切なメカニズムである一方の方式だけを選択す

の方が有利です。したがって、IP データは、IP パケットを示す 0xcc の NLPID 値を使用してカプセル化する必要があります。このオプションは、SNAP ヘッダーなしの 48 少ないビットを伝送し、X.25 ネットワークでの IP のカプセル化と整合しているので、より効率的です。

2.8 SONET および SDH 回路における PPP

ここでは、RFC 1619 で文書化されている、Synchronous Optical Network (SONET) および Synchronous Digital Hierarchy (SDH) リンク上での PPP カプセル化の使用について説明します。SONET および SDH は定義によると Point-to-Point 回路であるため、PPP はこれらのリンク上での使用によく適しています。SONET は、一群の標準の速度とフォーマットを定義する、オクテット同期の多重方式です。このような名前でありながら、これは光リンクに限定されません。電気仕様が、単一モード・ファイバー、マルチモード・ファイバー、および CATV 75 オーム同軸ケーブルに対して定義されています。伝送速度は、T3/E3 ビットの同期信号の伝送に使用できる 51.840 Mbps の整数倍です。許可されている倍数は、現在以下のように指定されています。

表 1 SONET 速度階層

種類	長さ	意味
0	-	オプション・リストの終了
1	-	ノーオペレーション
2	4	最大セグメント・サイズ
3	3	ウィンドウ・スケール
4	2	SACK 使用可能
5	X	SACK
8	10	タイム・スタンプ

CCITT Synchronous Digital Hierarchy は、155.52 Mbps で始まる SONET の伝送速度のサブセットを表 1 のように定義しています。

2.8.1 物理層

PPP は、物理層にオクテット・インターフェースを提供します。サブオクテットを供給または受信するための措置はありません。SONET および SDH リンクは、定義では全二重です。オクテット・ストリームは、SONET/SDH の Synchronous Payload Envelope (SPE) にマップされ、オクテット境界が SPE のオクテット境界に位置合わせされます。SPE への挿入時にスクランブルは不要です。パス信号ラベルは、SPE の内容を示すこと目的としています。

実験値 207 (16 進 CF) は PPP を示すために使用されます。マルチフレーム標識は現在は使用されていないため、ゼロでなければなりません。

SONET/SDH における PPP の基本速度は、155.52 Mbps での STS-3c/STM-1 の基本速度です。使用可能な情報帯域幅は 149.76 Mbps で、これは STS-3c/STM-1 SPE からセクション、行、およびパスのオーバーヘッドが取り除かれたものです。これは、ATM および FDDI に使用されるのと同じ上層マッピングです。これより低いデータ通信速度は、SONET/SDH の Virtual Tributary (VT) メカニズムを使用する必要があります。これは、T3/E3 速度までの既存の信号を非同期で SPE にマップするか、ビット同期およびバイト同期マッピングのために使用可能なクロックを使用します。これより高いデータ信号速度は、機器が使用可能になるときに、SONET STS シリーズではなく SDH STM シリーズに準拠している必要があります。STM シリーズは電力 3 ではなく 4 で進行し、採用されるステップも少数ですが、これによって多重化と統合が単純化される可能性があります。

2.9 Multi-Path Channel+ (MPC+)

MPC サポートは、複数の読み取りおよび書き込みサブチャネルが、ホストとチャネル接続装置の間の単一の伝送グループとして扱われることを許可するプロトコル層です。MPC サポートのレベルの 1 つで、MPC+ と呼ばれる High Performance Data Transfer (HPDT) は、非 HPDT MPC 接続よりも能率的なデータの伝送を提供します。Multi-Path Channel+ (MPC+) 接続では、複数の書き込み方向および読み取り方向サブチャネルを使用する、単一の伝送グループ (TG) を定義できます。各サブチャネルは一方方向にしか操作しないので、他のチャネル間接続で発生する半二重ターンアラウンド・タイムは削減されます。

読み取りおよび書き込みパスが少なくとも 1 つずつ正常に割り振られれば、MPC+ チャネル接続はアクティブになります。MPC+ グループ内の追加のパス (定義済みだがオンラインになっていないもの) は、MVS VARY 装置の ONLINE コマンドを使用して、後から動的にアクティブ・グループに追加できます。

たとえば、チャネル上の追加トラフィックのための容量を増やす必要がある場合は、追加のパスをアクティブ・グループへ中断なしに追加することができます。同様に、パスがなくなってきたときには、MVS VARY 装置の OFFLINE コマンドを使用してアクティブ・グループから削除できます。

2.10 非同期伝送モード (ATM)

ATM ベース・ネットワークに対するローカルおよびワイド・エリア・アプリケーションの双方への関心は高まっています。物理 ATM ネットワークの構築

に使用可能ないくつかの製品はすでに存在しています。ATM アーキテクチャーは新しく、したがって標準の LAN アーキテクチャーとは異なります。そのため、従来の LAN 製品を ATM 環境で機能させるための変更が必要です。TCP/IP の場合、必要とされる主な変更点は、ネットワーク・インターフェース (ATM のサポートを提供する) にあります。

すでに利用可能なアプローチがいくつか存在しますが、TCP/IP トラフィックの移送にはそのうちの 2 つが重要です。これらは 49 ページの 2.10.2 『Classical IP over ATM』および 56 ページの 2.10.3 『ATM LAN エミュレーション』で説明しています。これらは 59 ページの 2.10.4 『Classical IP over ATM 対 LAN エミュレーション』でも比較されています。

2.10.1 アドレス解決 (ATMARP および InATMARP)

ATM 論理 IP サブネット内のアドレス解決は、RFC 826 に基づく ATM Address Resolution Protocol (ATMARP) と、RFC 1293 (これは RFC 2390 で更新されている) に基づく Inverse ATM Address Resolution Protocol (InATMARP) によって行われます。ATMARP は、ARP プロトコルと同じプロトコルに、ユニキャスト・サーバーの ATM 環境で ARP をサポートするために必要な拡張を組み合わせたものです。InATMARP は、元の InARP プロトコルと同じプロトコルですが、ATM ネットワークに適用されます。これらのプロトコルの使用法は、永続仮想接続 (PVC) または交換仮想接続 (SVC) のどちらを使用するかによって異なります。

ATMARP および InATMARP は両方とも、状態が「任意選択 (Elective)」の提案標準である RFC 2225 (RFC 1577 を更新する) で定義されています。

ATMARP および InATMARP 要求 / 応答のカプセル化については、49 ページの 2.10.2 『Classical IP over ATM』で説明しています。

2.10.1.1 InATMARP

ARP プロトコルは、既知の IP アドレスについてのホストのハードウェア・アドレスを解決するために使用されます。InATMARP プロトコルは、既知のハードウェア・アドレスについてのホストの IP アドレスを解決するために使用されます。交換環境では、まず他の端末と通信するために、永続仮想接続 (PVC) または交換仮想接続 (SVC) の仮想接続 (VC) を確立します。したがって、パートナーの正確なハードウェア・アドレスは管理によって分かりますが、IP アドレスは不明です。InATMARP は動的なアドレス解決を提供します。InARP は標準 ARP と同じフレーム・フォーマットを使用しますが、以下の 2 つの新しい操作コードを定義します。

- InARP request=8
- InARP reply=9

詳細については、119 ページの 3.4.2.1『ARP パケットの生成』を参照してください。

基本の InATMARP は基本的に ARP と同様に操作しますが、例外として、InATMARP は要求をブロードキャストしません。これは、宛先端末のハードウェア・アドレスはすでに分かっているからです。要求側端末は、要求を、その送信元ハードウェアおよび IP アドレスと既知の宛先ハードウェア・アドレスを挿入することによって、要求を形式化するだけです。そして、宛先プロトコル・アドレス・フィールドにゼロを埋めこみ、それを宛先端末に直接送信します。受信側端末は、各 InATMARP 要求ごとに、要求の送信元アドレスを応答の宛先アドレスとして使用して応答を形式化します。両方の側で ARP テーブルを更新します。RFC 1700 によると、ATM のハードウェア・タイプ値は 10 進 19 で、EtherType フィールドは ARP を示す 0x806 に設定されています。

2.10.1.2 PVC 環境におけるアドレス解決

PVC 環境では、各端末は InATMARP プロトコルを使用して、接続されている他のすべての端末の IP アドレスを判別します。解決は、LLC/SNAP カプセル化用に構成されている PVC に対して実行されます。エージング・プロセスの一環として ARP テーブル・エントリーを再検証する責任は、PVC をサポートする各 IP 端末にあります。

2.10.1.3 SVC 環境におけるアドレス解決

SVC は、ATM の非ブロードキャスト環境での ATMARP のサポートを必要とします。この必要を満たすには、1 つの ATMARP サーバーが Logical IP Subnetwork (LIS) (52 ページの 2.10.2.1『Logical IP Subnetwork (LIS)』を参照) 内になければなりません。このサーバーは、LIS 内のすべての IP メンバーの ATMARP 要求を解決する権限責任を持ちます。ATM 用語の説明は、49 ページの 2.10.2『Classical IP over ATM』を参照してください。

サーバー自体は能動的に接続を確立しません。ATMARP 登録手続きの開始は、LIS 内のクライアントに依存しています。個々のクライアントは Point-to-Point VC を使用して ATMARP サーバーに接続します。サーバーは、LLC/SNAP カプセル化を指定する新規 VC の ATM 呼び出し / 接続の完了時に、InATMARP 要求を送信してクライアントの IP アドレスを判別します。クライアントからの InATMARP 応答には、ATMARP サーバーがその ATMARP テーブル・キャッシュを構築するために必要とする情報が含まれます。このテーブルは、以下のもので構成されます。

- IP アドレス
- ATM アドレス
- タイム・スタンプ
- 関連した VC

この情報は、これが受け取る ATMARP 要求に対する応答を生成するために使用されます。

注

ATMARP サーバー・メカニズムでは、各クライアントが ATMARP サーバーの ATM アドレスで管理的に構成されることが必要です。

ARP テーブルの追加 / 更新アルゴリズム

以下の点を考慮してください。

- ATMARP サーバーが InATMARP 応答で新しい IP アドレスを受け取ると、その IP アドレスが ATMARP テーブルに追加されます。
- InATMARP の IP アドレスがテーブル・エントリーの IP アドレスと重複して、InATMARP の ATM アドレスがテーブル・エントリーの ATM アドレスと一致せず、そのテーブル・エントリーに関連する開かれた VC が存在する場合、その InATMARP 情報は廃棄され、テーブルに変更は加えられません。
- 送信元 IP および ATM アドレスが ATMARP テーブル内にすでに存在するアソシエーションと一致し、ATM アドレスが VC に関連したアドレスと一致する ATMARP 要求を、サーバーが VC を介して受け取ると、サーバーは送信元の ATMARP テーブル・エントリーのタイムアウトを更新します。たとえば、クライアントがその ATMARP エントリーの登録に使用したのと同じ VC を介してサーバーに ATMARP 要求を送信している場合、サーバーはそのクライアントがまだ「活動状態にある」ということに気付いて、そのクライアントの ATMARP テーブル・エントリーのタイムアウトを更新します。
- サーバーは、VC を介して ARP_REQUEST を受け取ると、送信元情報を検査します。ATMARP 要求の受信に使用された VC に関連した IP アドレスがなく、送信元 IP アドレスが他のいかなる接続にも関連付けられていない場合、サーバーはこの端末をその ATMARP テーブルに追加します。前述のとおり、ATMARP サーバーで登録を行うことはクライアントの責任なので、これは正規の方法ではありません。

ATMARP テーブル・エージング

ATMARP テーブル・エントリーは以下の期間有効です。

- クライアントでは、最大で 15 分間
- サーバーでは、最小で 20 分間

ATMARP テーブル・エントリーのエージングの前に、ATMARP サーバーは、そのエントリーと関連する開かれたすべての VC に対する InARP_REQUEST を生成し、以下の規則に従って何を実行するかを判別します。

- InARP_REPLY が受信される場合は、そのテーブル・エントリーは更新され、削除はされません。
- テーブル・エントリーに関連する開かれた VC が存在しない場合、エントリーは削除されます。

したがって、クライアントがサーバーに対して開かれた VC を保持していない場合、クライアントは少なくとも 20 分間に 1 回、サーバーとのその ATMARP 情報の最新表示を行わなければなりません。これは、サーバーに対して VC を開き、初期 InATMARF パケットを交換することによって行われます。

クライアントは、以下に従ってテーブルの更新を処理します。

- ATMARP テーブル・エントリーがエージングを行うとき、ATMARF クライアントはそのテーブル・エントリーを無効にします。
- 無効にされたエントリーに関連する開かれた VC が存在しない場合、そのエントリーは削除されます。
- 無効にされたエントリーと開かれた VC の場合、ATMARF クライアントは、その VC に関する非アドレス解決トラフィックを送送する前にエントリーを再検証します。以下の 2 つの可能性があります。
 - PVC の場合、クライアントは、InARP_REQUEST を伝送し、InARP_REPLY を受信してからエントリーを更新することによって、エントリーを検証します。
 - SVC の場合、クライアントは、ARP_REQUEST を ATMARP サーバーに伝送し、ARP_REPLY を受信してからエントリーを更新することによってエントリーを検証します。
- 無効にされた ATMARP テーブル・エントリーに関連する VC が閉じられている場合、そのテーブル・エントリーは除去されます。

前述のとおり、SVC を使用する各 ATM IP クライアントは、特定の LIS に対するその ATMARP サーバーの ATM アドレスを認識していなければなりません。このアドレスは、カスタマイズ時に各クライアントで指定される必要があります。現時点で定義されているウェルノウン ATMARP サーバー・アドレスはありません。

2.10.2 Classical IP over ATM

Classical IP over ATM (非同期伝送モード) のインプリメンテーションの定義は、RFC 2225 に記述されています。RFC 2400 (STD 1) によれば、これは状態が「任意選択 (Elective)」である提案標準です。この RFC は、「ワイヤー」、IP 終了端末 (メンバー) に接続するローカル LAN セグメント、および古典的な LAN ベース・パラダイム (枠組み) で操作するルーターの直接の置き換え

としての ATM の適用のみを考慮しています。MAC レベル・ブリッジングおよび LAN エミュレーションによって引き起こされる問題は扱われていません。

ATM Forum の ATM マイグレーションの提供方式については、56 ページの 2.10.3『ATM LAN エミュレーション』を参照してください。

ATM の初期配置は、以下に対して LAN セグメント置換を提供します。

- ・イーサネット、トークンリング、または FDDI ネットワーク
- ・既存の (非 ATM) LAN 間のローカル・エリア・バックボーン
- ・IP ルーター間のフレーム・リレー PVC の専用回線

この RFC は、ARP プロトコル (RFC 826) に対する、ATM 上で機能するための拡張についても言及しています。これについては、46 ページの 2.10.1『アドレス解決 (ATMARP および InATMARP)』で別個に説明します。

以下は、いくつかの初歩的な ATM 基本です。

セル

すべての情報 (音声、イメージ、ビデオなど) は、セルと呼ばれる非常に短い (48 データ・バイト + 5 バイトのヘッダー) ブロックによって、ネットワークを経由して移送されます。

ルーティング

情報フローは、ネットワークを通る一続きのポインターとしてセットアップされたパス (仮想チャネルと呼ばれる) に沿います。セル・ヘッダーには、セルと、それが宛先とする正しいパスとをリンクする ID が含まれます。

特定の仮想チャネル上のセルは、常にネットワーク上の同じパスをたどり、受信されたのと同じ順序で宛先に送達されます。

ハードウェア・ベース交換接続 ATM は、交換接続を実行する各ノードで、単純なハードウェア・ベースの論理エレメントを採用できるように設計されています。1 Gbps のリンクでは、0.43 マイクロ秒ごとに新しいセルが到着し、セルが伝送されます。到着するパケットをどう処理するかを決定するためのたくさんの時間はありません。

仮想接続 (VC)

ATM は、仮想接続交換環境を提供します。VC のセットアップは、永続仮想接続 (PVC) または動的な交換仮想接続 (SVC) のいずれに基づ

いて行うことができます。SVC の呼び出し管理は、Q.93B プロトコルのインプリメンテーションを介して実行されます。

エンド・ユーザー・インターフェース

高位層プロトコルが ATM ネットワーク上で通信するための唯一の方法は、ATM アダプテーション層 (AAL) を経由するものです。この層の機能は、プロトコル・データ単位 (PDU) の ATM セルの情報フィールドへのマッピング (およびその逆) を実行することです。AAL1、AAL2、AAL3/4、および AAL5 の定義済みの 4 つの異なる AAL タイプが存在します。これらの AAL は、高位層プロトコルにさまざまなサービスを提供します。以下は、TCP/IP に使用される AAL5 の特性です。

- メッセージ・モードおよびストリーム・モード
- 確実送達
- 非確実送達 (TCP/IP で使用される)
- データのブロック化およびセグメント化
- マルチポイント操作

AAL5 は、メディア・アクセス制御 (MAC) 層で LAN と同じ機能を提供します。AAL タイプは、セルのセットアップ・メカニズムを通して VC エンドポイントに認識され、ATM セル・ヘッダーでは伝送されません。PVC の場合、AAL タイプは、接続 (回路) のセットアップ時にエンドポイントで管理的に構成されます。SVC の場合、AAL タイプは、呼び出しセットアップ確立の一環として Q.93B を介する VC パスに沿って伝達され、エンドポイントは信号化された情報を構成に使用します。ATM スイッチは、通常は VC の AAL タイプを考慮しません。AAL5 フォーマットは、ユーザー・データの最大サイズが 64KB - 1 バイトであるパケット・フォーマットを指定します。高位層プロトコルが AAL 層と (AAL サービス・アクセス・ポイント - SAP で) インターフェースをとるために使用する必要があるブリミティブは、厳密に定義されています。高

位層プロトコルがデータを送信すると、そのデータがまずアダプテーション層、次に ATM 層によって処理され、その後、物理層が引き継いでデータを ATM ネットワークに送信します。セルはネットワークによって移送され、相手側の物理層でまず受信され、それから ATM 層、その次に受信 AAL によって処理されます。このすべてが完了すると、情報（データ）は受信高位層プロトコルに渡されます。ATM ネットワークによって実行されてきたすべての機能は、一方の側から他方の側への情報の非確実移送（一部破損の可能性あり）です。従来のデータ処理の観点で見ると、ATM ネットワークが実行してきた事柄はすべて、物理リンク接続の別の種類の物理接続での置き換えです。すべての高位層ネットワーク機能は今後さらに実行する必要があります（たとえば、IEEE 802.2）。

アドレッシング

ATM フォーラム・エンドポイント・アドレスは、20 バイトの OSI NSAP ベース・アドレス（私設ネットワーク・アドレスに使用され、3 つのフォーマットが可能）としてエンコードされているか、E.164 Public UNI アドレス（公衆 ATM ネットワークに使用される電話番号スタイルのアドレス）であるかのいずれかです。²

ブロードキャスト、マルチキャスト

現在のところ、LAN に類似したブロードキャスト機能は提供されていません。しかし、使用可能なマルチキャスト機能は存在します。マルチキャストに相当する ATM 用語は、*Point-to-Multipoint 接続* です。

2.10.2.1 Logical IP Subnetwork (LIS)

LIS という用語は、論理 IP 構造を ATM ネットワークにマップするために導入されました。LIS のシナリオでは、個々の管理エンティティはそれぞれ、閉ざされた論理 IP サブネットワーク（同じ IP ネットワーク / サブネット番号とアドレス・マスク）内に、そのホストとルーターを構成します。各 LIS は、同じ ATM ネットワーク上の他の LIS とは独立して操作および通信します。ATM ネットワークに接続するホストは、同じ LIS 内の他のホストと直接通信します。このことは、LIS のすべてのメンバーが ATM を介して同じ LIS 内の

² ATM Forum は、産業およびエンド・ユーザー・コミュニティ内での ATM の推進を目的とした、世界的な組織です。このメンバーシップには、通信およびコンピューター産業のすべてのセクターを代表する 500 を超える企業のほか、いくつかの政府機関、研究組織、およびユーザーが含まれます。

他のすべてのメンバーと通信できることを意味します。(VC トポロジは完全メッシュです。) 論理 LIS の外部のホストに対する通信は、IP ルーターを介して行われます。このルーターは、1 つ以上の LIS のメンバーとして構成された、ATM ネットワークに接続する ATM エンドポイントです。この構成では、いくつかの別個の LIS が同じ ATM ネットワークを経て動作することになる場合があります。たとえ、ATM ネットワーク上の 2 つの IP メンバー間に直接の VC を開ける可能性があっても、異なる IP サブネットのホストは中間 IP ルーターを介して通信する必要があります。

2.10.2.2 マルチプロトコルのカプセル化

物理ネットワーク上で並行して 2 つ以上のネットワーク・プロトコルのタイプ (IP、IPX など) を使用する場合は、異なるプロトコルを多重化する方法が必要になります。これは、ATM の場合、VC ベースの多重化または LLC カプセル化のいずれかによって実行できます。VC ベースの多重化を選択する場合は、2 つのホスト間に、異なるプロトコルごとの VC がなければなりません。LLC カプセル化は LLC 層で多重化機能を提供し、したがって 1 つの VC のみを必要とします。TCP/IP は、RFC 2225 および 1483 によれば、2 番目の方式を使用します。この理由は、イーサネット、トークンリング、および FDDI などの他のすべての LAN タイプに対して、この種類の多重化が RFC 1042 ですでに定義されているためです。この定義では、IP は単に LAN の置き換えとして ATM を使用しています。アイソクロナス・トラフィックの移送など、ATM が提供する必要がある他のすべての利点を使用されているわけではありません。IP インプリメンテーションを向上させるという使命を持ち、将来の標準に対するインターネット・コミュニティの関心を表明するために、ATM フォーラムとの仲立ちをする IETF 作業チームが存在します。

厳密に言えば、TCP/IP PDU は、IEEE 802.1a SubNetwork Attachment Point (SNAP) ヘッダーの前の IEEE 802.2 LLC ヘッダーの中にカプセル化され、AAL5 CPCS-PDU (Common Part Convergence Sublayer) のペイロード・フィールドに入れられて伝送されます。AAL5 CPCS-PDU フォーマットは、図 12 に表示しています。

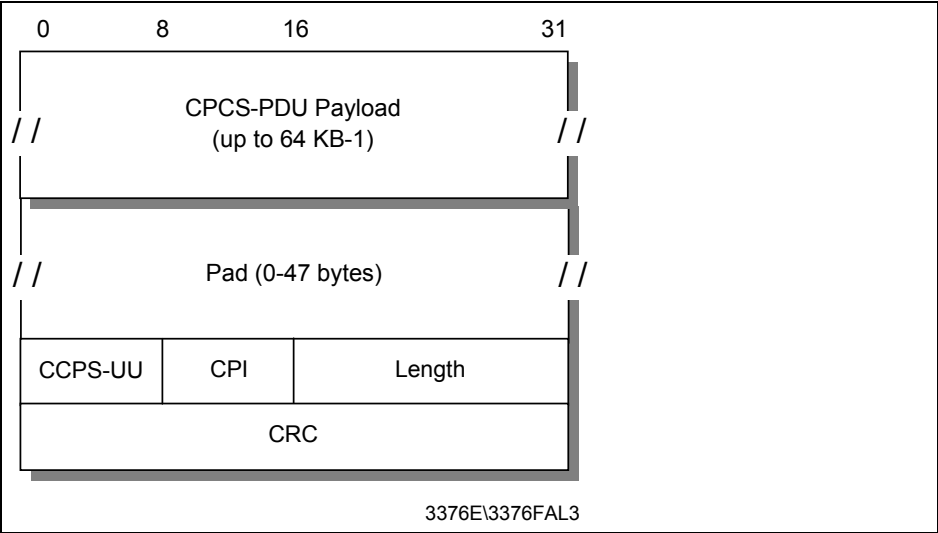


図 12 AAL5 CPCS-PDU フォーマット

CPCS-PDU ペイロード CPCS-PDU ペイロードは、図 13 に表示しています。

埋め込み 埋め込みフィールドは、ATM セルに正確に合うように CDCS-PDU を埋め込みます。

CPCS-UU CPCS-UU (ユーザー間識別) フィールドは、CPCS ユーザー間情報を透過的に転送するために使用されます。このフィールドは、カプセル化のための機能を持たず、どんな値にも設定できます。

CPI Common Part Indicator (CPI) フィールドは、CPCS-PDU トレーラーを 64 ビットで位置合わせします。

長さ 長さフィールドは、ペイロード・フィールドの長さ (バイト単位) を示します。最大値は、64KB - 1 である 65535 です。

CRC CRC フィールドは、CRC フィールド自身を除いた CPCS-PDU 全体を保護します。

図 13 は、経路指定された IP PDU のペイロード・フォーマットを示しています。

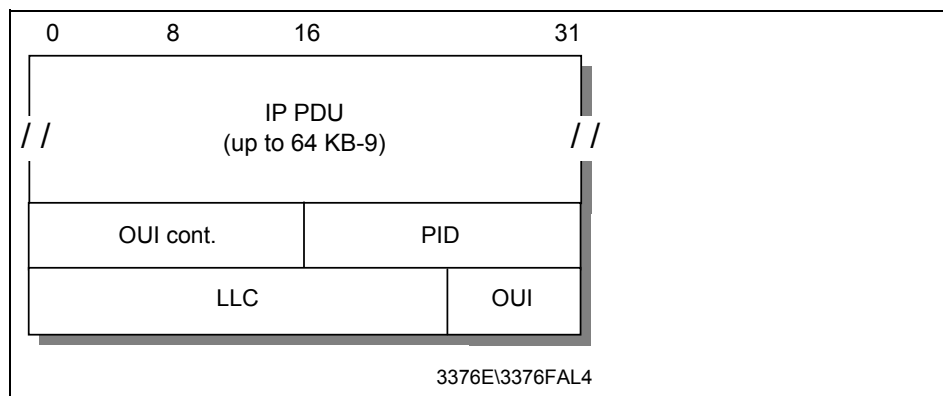


図 13 IP PDU の CPCS-PDU ペイロード・フォーマット

IP PDU IP ヘッダーで始まる通常の IP データグラム。

LLC DSAP-SSAP-Ctrl のフォーマットを持つ 3 バイトの LLC ヘッダー。IP データの場合、これは、SNAP ヘッダーの存在を示す 0xAA-AA-03 に設定されます。Ctrl フィールドの値は常に 0x03 で、Unnumbered Information Command PDU を指定します。

- OUI** 3 バイトの組織固有 ID (OUI) は、続く 2 バイトのプロトコル ID (PID) の意味を管理する組織を識別します。PID に EtherType を指定するには、OUI が 0x00-00-00 に設定されていなければなりません。
- PID** Protocol Identifier (PID) フィールドは、続く PDU のプロトコル・タイプを指定します。IP データグラムの場合、割り当て済みの EtherType または PID は、0x08-00 です。

ATM ネットワーク内の IP メンバーのデフォルトの MTU サイズは、RFC 1626 で説明され、9180 バイトに定義されています。LLC/SNAP ヘッダーは 8 バイトです。したがって、デフォルトの ATM AAL5 PDU サイズは 9188 になります。指定可能な値は 0 ~ 65535 の値にできます。MTU サイズは変更できますが、その場合は LIS のすべてのメンバーを変更して、同じ値を持つようにする必要があります。RFC 1755 は、すべてのインプリメンテーションが 64KB まで (64KB を含む) の MTU サイズをサポートすることを推奨しています。

ATM ネットワーク内のアドレス解決は ARP プロトコルの拡張として定義されており、これについては 46 ページの 2.10.1 『アドレス解決 (ATMARP および InATMARP)』で説明しています。

IP ブロードキャストまたはマルチキャスト・アドレスから、ATM ブロードキャストまたはマルチキャスト・アドレスへの使用可能なマッピングはありません。ただし、RFC 1122 で説明されている 4 つの標準 IP ブロードキャスト・アドレス形式のいずれかを指定した、IP データグラムの伝送または受信に関する制限はありません。メンバーは、その LIS に対する IP ブロードキャストまたは IP サブネット・ブロードキャストを受信したら、その端末にアドレッシングされているかのようにそのパケットを処理しなければなりません。

2.10.3 ATM LAN エミュレーション

ネイティブ ATM ネットワークにマイグレーション・パスを提供するためのもう 1 つのアプローチは、ATM LAN エミュレーションです。ATM LAN エミュレーションは、ATM フォーラム作業チームによってまだ準備中です。IETF アプローチについては、49 ページの 2.10.2 『Classical IP over ATM』を参照してください。ATM での仮想 LAN を扱った有効な ATM フォーラム・インプリメンテーション合意はありませんが、ATM フォーラムに提出されたさまざまな提案についてのいくつかの基本合意は存在します。以下の説明は、IBM の提案に基づいています。

ATM LAN エミュレーションの概念は、ワークステーション・アプリケーション・ソフトウェアがトークンリングなどの実際の共有メディア LAN のメンバーであると「考える」ようなシステムを構成することです。この方式は、既存の LAN ソフトウェアの再利用を最大化し、ATM へのマイグレーションの

コストを大幅に削減します。たとえば、PC LAN 環境では、LAN エミュレーション層は NDIS/ODI タイプ・インターフェースの下にインプリメントすることができます。このようなインプリメンテーションでは、IP、IPX、NetBIOS、および SNA などのすべての高位層プロトコルを ATM ネットワーク上で変更なく実行することができます。

トークンリングおよびイーサネットのインプリメンテーションについては、図 14 を参照してください。

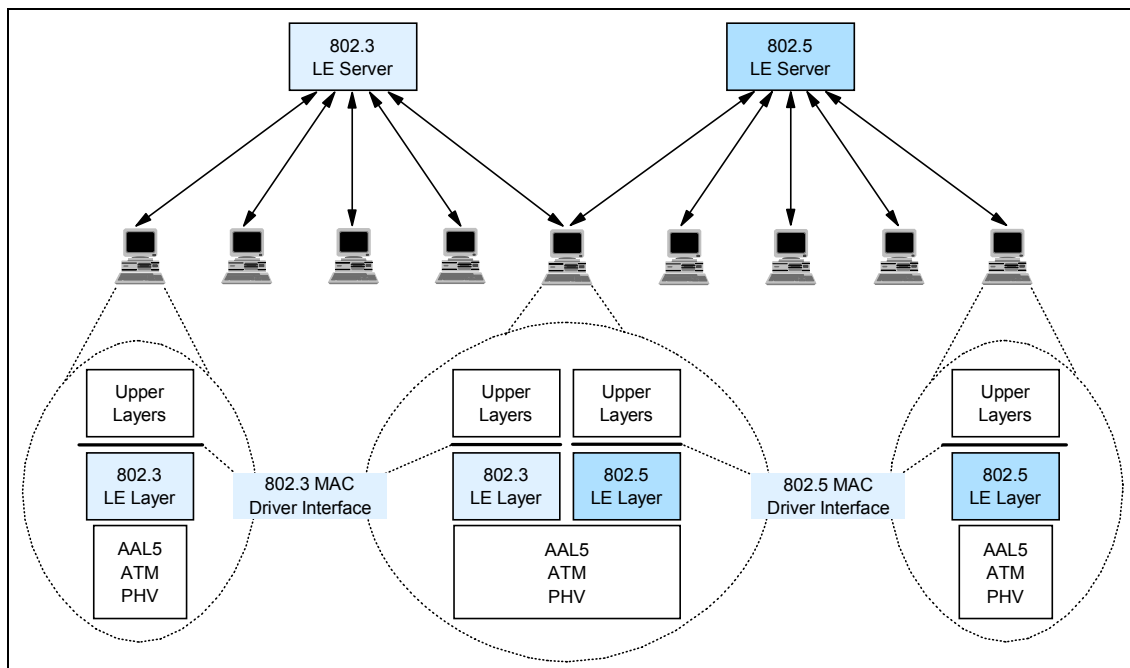


図 14 イーサネットおよびトークンリングの LAN エミュレーション

2.10.3.1 LAN エミュレーション層 (ワークステーション・ソフトウェア)

LE 機能を実行する各ワークステーションは、LE サービスを提供するソフトウェアを持っている必要があります。このソフトウェアは LAN エミュレーション層 (LE 層) と呼ばれます。これは既存のプロトコル・サポート (IP、IPX、IEEE 802.2 LLC、NetBIOS など) へのインターフェースを提供し、実際の共有メディア LAN の機能をエミュレートします。つまり、ATM サービスを使用するために、既存の LAN アプリケーション・ソフトウェアに変更を加える必要はありません。LE 層は、ハードウェア ATM アダプターを通して ATM ネットワークに接続します。

LE 層の 1 次機能は、(高位層から到着する) カプセル化された LAN フレームを、直接に (直接 VC で)、あるいは LE サーバーを通してその宛先へ転送することです。これは、ATM が提供する AAL5 サービスを使用して行われます。

各 LE 層は 1 つ以上の LAN アドレスと ATM アドレスを持ちます。

各ワークステーション内には、サポートされる異なる LAN または LAN のタイプごとに、LE 層の別個のインスタンス (論理コピーまたは LE クライアント) が必要です。たとえば、トークンリングとイーサネットの両方の LAN タイプをエミュレートする場合は、2 つの LE 層が必要です。実際には、これらはおそらく同じコードの同じコピー内の異なるスレッドに過ぎませんが、論理的には別個の LE 層です。1 つのワークステーションを、エミュレートされた異なる 2 つのトークンリング LAN の一部にする必要がある場合にも、別個の LE 層が使用されます。別個のそれぞれの LE 層は、異なる MAC アドレスを必要としますが、同じ物理 ATM 接続 (アダプター) を共有できます。

2.10.3.2 LAN エミュレーション・サーバー

LE サーバーの基本機能は、ディレクトリー、マルチキャスト、およびアドレス解決サービスをワークステーションの LE 層に提供することです。これはまた、必要に応じて、コネクションレス・データ転送サービスをワークステーションの LE 層に提供します。

エミュレートされたそれぞれの LAN は、LE サーバーを持っていない限りなりません。(マルチスレッド化を介して) 同じハードウェアとコードを共有する複数の LE サーバーを持つことは可能ですが、LE サーバーは論理的には別個のエンティティです。LE 層に関する限り、エミュレートされたトークンリング LAN が、イーサネット LAN をエミュレートするメンバーを持つことはできません。したがって、LE サーバーのインスタンスは、LAN エミュレーションの単一のタイプ専用です。LE サーバーは物理的には ATM ネットワークの内部にあたり、外部装置で提供されていたりする可能性があります。論理的には必ず、ATM が提供するサービスを使用してそのジョブを実行する外部機能です。

2.10.3.3 デフォルトの VC

デフォルトの VC は、ワークステーションの LE 層と LE サーバーとの間の接続です。この接続は、永続または交換になります。

すべての LE 制御メッセージは、デフォルトの VC 上の LE 層と LE サーバーとの間で伝送されます。カプセル化されたデータ・フレームも、デフォルトの VC 上で送信できます。

LE 機能が実行されるためには、LE サーバーとデフォルトの VC の存在が必要です。

2.10.3.4 直接 VC

直接 VC は、終端システム内の LE 層の間の接続です。これは必ず交換であり、オンデマンドでセットアップされます。ATM ネットワークが交換接続をサポートしていない場合は、直接 VC を持つことはできず、データはすべてデフォルトの VC 上の LE サーバーを通して送信される必要があります。何らかの理由で使用可能な直接 VC が存在しない場合、データ転送は LE サーバーを通して行われる必要があります。(その他に方法はありません。)

直接 VC は、LE 層による要求時にセットアップされます。(ATM には第三者呼び出しのセットアップ機能は存在しないため、サーバーがこれらをセットアップすることはできません。)宛先 LE 層の ATM アドレスは、LE サーバーによって要求 LE 層に提供されます。直接 VC は、パートナー LE 層のいずれかが(データがなくなったために)接続を終了する決定をするまでそのまま残ります。

2.10.3.5 初期化

初期化の際、LE 層(ワークステーション)は、デフォルトの VC と LE サーバーを設定します。これはまた、後から直接 VC をセットアップする場合に必要なその独自の ATM アドレスを発見します。

2.10.3.6 登録

このフェーズで、LE 層(ワークステーション)は、その MAC アドレスと LE サーバーを登録します。フィルター要件(オプション)などの他のものを指定することもできます。

2.10.3.7 管理および解決

これは、ATM 終端端末が他の終端端末(LE 層)との直接 VC をセットアップするために使用する方式です。この機能には、宛先端末の ATM アドレスを知り、MAC アドレスを ATM アドレスにマップし、マッピングをテーブルに保管し、テーブルを管理するためのメカニズムが含まれます。

サーバーの場合、この機能は、終端端末による直接 VC の使用をサポートするための手段を提供します。これには、終端システムの MAC アドレスをその ATM アドレスにマップし、その情報を保管して、それを要求終端端末に提供するためのメカニズムが含まれます。

この構造は、全 LAN 機能を保持し、最高位層の LAN プロトコルをサポートすることができます。データ転送でのサーバーに対する依存度は、大半のバルク・データの移送に交換 VC を使用することによって最小化されます。

2.10.4 Classical IP over ATM 対 LAN エミュレーション

ATM にマイグレーションするためのより容易な方法を提供するためのこの 2 つのアプローチは、異なった目標を念頭に作成されました。

Classical IP over ATM は、カプセル化とアドレス解決の方式を定義しています。定義は IP についてのみ行われ、他のプロトコルとの使用についての定義は行われていません。そのため、他のプロトコル・スタック (IPX や SNA など) を必要とするアプリケーションがある場合、IP over ATM は完全なソリューションを提供しません。逆に、TCP または UDP ベース・アプリケーションのみを持っている場合は、これはより優れたソリューションになる可能性があります。なぜなら、ATM アーキテクチャーに対する IP プロトコルのこの特化した適応は、よりグローバルなソリューションよりも生じるオーバーヘッドは少なく、したがってより効率的になり得るからです。このインプリメンテーションのもう 1 つの利点は、ラージ MTU サイズなどのいくつかの ATM 固有の機能の使用です。

ATM Forum のアプローチの主な目標は、第 3 層以上のプロトコルを ATM ネットワーク上で変更なく実行することです。これは、たとえば TCP/IP、IPX、NetBIOS、および SNA などの既存のプロトコルとそのアプリケーションが、変更なく高速な ATM ネットワークの利点を利用できることを意味します。すべてのプロトコルのマッピングがすでに定義されています。LAN エミュレーション層は、古典的な LAN のすべてのサービスを提供します。したがって上の層は ATM の存在を知りません。これは利点であり、欠点でもあります。なぜなら、下層のネットワークについての知識は、より効果的なインプリメンテーションを提供するために使用できるからです。

近い将来、特定の要件に応じて両方のアプローチが使用されるようになるでしょう。やがて、ATM へのアプリケーションのマッピングが完全に定義されてインプリメントされれば、専用 ATM インプリメンテーションの体系が使用されるようになる可能性があります。

2.11 Multiprotocol over ATM (MPOA)

MPOA の目的は以下のとおりです。

- ATM ネットワークでの終端間の第 3 層インターネットワーキング接続を提供します。これは以下のいずれかの方法で接続されたホストのためのものです。
 - ATM ネットワークへ直接に
 - レガシー LAN 上の ATM ネットワークへ間接的に
- レガシーおよび ATM 接続装置間でのインターネットワーク層 (たとえば、IP サブネットなど) の分散をサポートします。第 3 層ネットワークへの移植でのルーターの制限を取り除き、プロトコル・ベース仮想 LAN (VLAN) の構築を可能にします。
- 分散ルーティング・コンポーネント間のインターオペラビリティを保証しながら、インプリメンテーションの柔軟性を許容します。

- ・インターネットワーク層プロトコルが ATM ネットワークのサービスを使用する方法を示します。

名前は Multiprotocol over ATM ですが、現在のところ MPOA 副作業チームで行われている実際の研究は、すべて IP に焦点が当てられています。

2.11.1 MPOA の利点

MPOA は、LAN エミュレーションから、インターネットワーキング層プロトコルによる ATM の直接利用への遷移を示しています。その利点は、以下のとおりです。

- ・プロトコルは、ATM を単なる別のリンク以上のものと見なします。そのため、ATM のファシリティーの利用が可能になります。
- ・レガシー LAN フレーム構造のオーバーヘッドのための必要が取り除かれます。

MPOA ソリューションには、Classical IP (RFC 1577) と LAN Emulation の両ソリューションに勝る以下のような利点があります。

- ・サブネット境界を横断できる終端システム間の直接接続が可能なことによる、より短い待ち時間。これは、異なる仮想 LAN 上にある終端システム間の通信のために ATM ルーター間を何度もホップする必要を最小化することによって達成されます。
- ・処理機能をネットワークの端に分散させたことによる、第 3 層のより高い総転送能力。
- ・特定のフローを特定の QOS 特性にマップすることができます。
- ・第 3 層サブネットを物理ネットワーク上に分散させることができます。

2.11.2 MPOA 論理コンポーネント

MPOA ソリューションは、いくつかの論理コンポーネントとそれらのコンポーネント間の情報フローで構成されます。論理コンポーネントには、以下の 2 つの種類があります。

MPOA サーバー MPOA サーバーは、自身がサービスを提供する IASG の MAC およびインターネットワーキング層トポロジーについての全知識を保守します。これを達成するため、これらは、これら自身の間と MPOA クライアントとで情報を交換します。

MPOA クライアント

MPOA クライアントは、マッピング (パケット接頭部から ATM 情報への) のローカル・キャッシュを保守します。こ

これらのキャッシュは、必要に応じて適切な MPOA サーバーからの情報を要求することによって移植されます。

MPOA クライアントに関連した第 3 層アドレスは、クライアント自身の第 3 層アドレスか、クライアントを経由して到達可能な第 3 層アドレスのいずれかを示します。(クライアントは終端装置またはルーターを持ちます。)

MPOA クライアントは、その MPOA サーバーに接続してクライアントの ATM アドレスとクライアントを介して到達可能な第 3 層アドレスを登録します。

2.11.3 MPOA 機能コンポーネント

図 15 は、以下の層の間で分割されている、論理コンポーネントと物理コンポーネントの間のマッピングを示しています。

- MPOA 機能グループ層
- LAN エミュレーション層
- 物理層

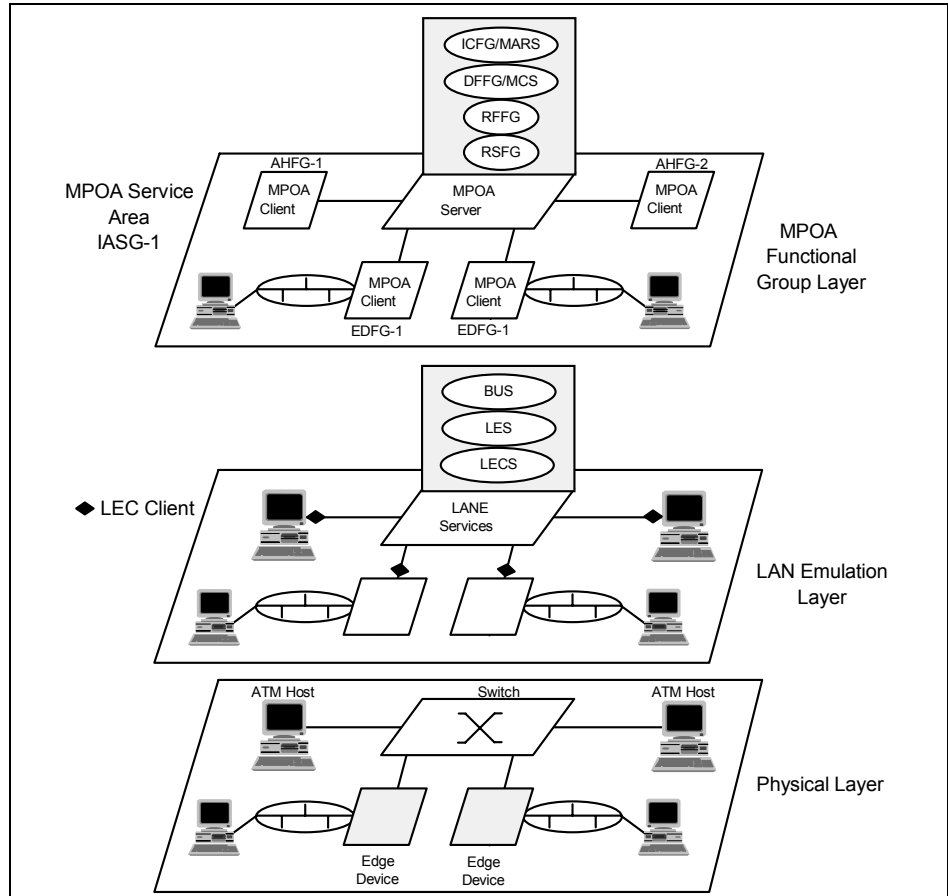


図 15 MPOA 機能コンポーネント

MPOA ソリューションは、以下を含む多様な機能グループにインプリメントされます。

- インターネットワーク・アドレス・サブグループ (IASG): インターネットワーク層アドレス (たとえば、IPv4 サブネット) の範囲。したがって、ホストが 2 つのインターネットワーク層プロトコルを操作する場合、それは少なくとも 2 つの IASG のメンバーになります。
- 終端装置機能グループ (EDFG): EDFG は、レガシー・サブネットワークと ATM との間にインターネットワーキング・レベルの接続を提供する装置によって実行される機能のグループです。
 - EDFG は第 3 層パケット転送をインプリメントしますが、ルーティング・プロトコル (RSFG で実行される) は実行しません。

- シンプルとスマートの2つのタイプの EDFG が可能です。
 - スマート EDFG は、インターネットワーキング・アドレスの解決を要求します (つまり、宛先のエントリーを持たない場合、これは照会 ARP タイプ・フレームを送信します。)
 - シンプル EDFG は、エントリーが存在しない場合、デフォルトのクラスを介してデフォルトの宛先にフレームを送信します。
- 共存プロキシ LEC 機能は必須です。
- ATM 接続ホスト機能グループ (AHFG): AHFG は、MPOA ネットワークに参加している ATM 接続ホストによって実行される機能のグループです。
共存プロキシ LEC 機能はオプションです。
IASG 内では、レガシー装置または LAN エミュレーション装置のいずれかへの移送メカニズムとして LAN エミュレーションが使用され、これには LEC へのアクセスが必要です。AHFG が LANE またはレガシー装置と通信しない場合は、共存 LEC は不要です。
- IASG 調整機能グループ (ICFG): ICFG は、1 つ以上の EDFG および ATM の両方または一方の装置上の、複数のレガシー LAN ポート間での単一の IASG の分散を調整するために実行される機能のグループです。ICFG は、第 3 層アドレスの照会に応答できるよう、機能コンポーネントの位置をトラッキングします。
- デフォルト転送機能機能グループ (DFFG): 直接のクライアント間接続がないため、DFFG は IASG の内部または外部のいずれかに宛てられたトラフィック用のデフォルトの転送を提供します。
 - IASG 内にインターネットワーキング層マルチキャスト転送を提供します。つまり、DFFG は MPOA ベース MARS インプリメンテーションでマルチキャスト・サーバー (MCS) として動作します。
 - AHFG に (言い換えると、LANE クライアントを持たない AHFG に) プロキシ LAN エミュレーション機能を提供し、AHFG がレガシー接続システムとトラフィックを送受信できるようにします。
- 経路サーバー機能グループ (RSFG): RSFG は、MPOA ネットワークでインターネットワーキング・レベルの機能を実行します。これには以下が含まれます。
 - 従来型のインターネットワーキング・ルーティング・プロトコル (たとえば OSPF、RIP、および BGP) の実行
 - IASG 間のアドレス解決の提供、要求の処理、および応答の作成
- リモート転送機能機能グループ (RFFG): RFFG は、送信元から宛先へ (IASG または MPOA クライアントのいずれかが存在する場合がある) のト

ラフィックの転送に関連して実行される機能のグループです。RFFG は、典型的な IPv4 サブネットのデフォルト・ルーター 機能と同義です。

注

これらの機能グループの 1 つ以上が同じ物理エンティティ内に共存することがあります。MPOA は、これらのグループの任意の物理位置を許可しています。

2.11.4 MPOA 操作

MPOA システムは、望ましい動作を提示するために情報を交換する機能グループのセットとして操作します。MPOA システムの概要を示すため、コンポーネントの動作を *重要なイベント* の順に説明します。

構成	すべての機能グループが管理情報の適切な設定を持つことを保証します。
登録とディスカバリー	その存在と接続装置の ID を互いに通知し合う機能グループと、ICFG にレガシー装置を通知する EDFG が含まれます。
宛先解決	宛先インターネットワーク層アドレスに与えられた経路記述を、おそらく他の情報 (たとえば QOS) と共に判別するアクション。これは、これが横断 (IASG 境界に関して) を実行することを許可している MPOA システムの一部です。
データ転送	インターネットワーク層データを、ある MPOA クライアントから別の MPOA クライアントに送ること。
IASG 内調整	IASG が複数の物理インターフェース間に拡張できるようにする機能。
ルーティング・プロトコル・サポート	MPOA システムが従来のインターネットワークと対話できるようにします。
スパンニング・ツリー・サポート	MPOA システムが既存の拡張 LAN と対話できるようにします。
複製サポート	容量や回復力の理由でのキー・コンポーネントの複製のために提供します。

2.12 リファレンス

以下の RFC は、本章で示された接続プロトコルとアーキテクチャーに関する詳細情報を提供しています。

- RFC 826 – Ethernet Address Resolution Protocol
- RFC894 – Standard for the Transmission of IP Datagrams over EthernetNetworks
- RFC1042 – Standard for the Transmission of IP Datagrams over IEEE802 Networks
- RFC1055 – Nonstandard for Transmission of IP Datagrams over SerialLines: SLIP
- RFC1144 – Compressing TCP/IP Headers for Low-Speed Serial Links
- RFC1188 – Proposed Standard for the Transmission of IP Datagramsover FDDI Networks
- RFC1356 – Multiprotocol Interconnect on X.25 and ISDN in the PacketMode
- RFC1483 – Multiprotocol Encapsulation over ATM Adaptation Layer5
- RFC1618 – PPP over ISDN
- RFC1619 – PPP over SONET/SDH
- RFC1661 – The Point-to-Point Protocol (PPP)
- RFC1662 – PPP in HDLC-Like Framing
- RFC1700 – Assigned Numbers
- RFC1755 – ATM Signaling Support for IP over ATM
- RFC 1795 – Data Link Switching: Switch-to-Switch Protocol AIW DLSw RIG: DLSw Closed Pages, DLSw Standard Version 1
- RFC 2225 – Classical IP and ARP over ATM
- RFC 2390 – Inverse Address Resolution Protocol
- RFC 2400 – Internet Official Protocol Standards
- RFC 2427 – Multiprotocol Interconnect over Frame Relay

第 3 章インターネットワーキング・プロトコル

この章では、TCP/IP インターネットワーク層に関連した最も重要で一般的なプロトコルについて概説します。これには、以下が含まれます。

- インターネット・プロトコル (IP)
- Internet Control Message Protocol (ICMP)
- Address Resolution Protocol (ARP)
- Reverse Address Resolution Protocol (RARP)
- Bootstrap Protocol (BOOTP)
- Dynamic Host Configuration Protocol (DHCP)

これらのプロトコルは、データグラム・アドレッシング、ルーティングと送達、および動的アドレス構成を実行し、インターネットワーク層アドレスとネットワーク・インターフェース層アドレスとの間の解決を行います。

3.1 インターネット・プロトコル (IP)

IP は、STD 番号 5 の標準プロトコルです。この標準には、ICMP (105 ページの 3.2 『Internet Control Message Protocol (ICMP)』を参照) および IGMP (241 ページの 6.2 『Internet Group Management Protocol (IGMP)』を参照) も含まれます。IP の状態は「必須 (Required)」です。

現行の IP 仕様は、RFC 791、950、919、および 922 にあり、更新が RFC 2474 にあります。

IP は、仮想ネットワーク・ビューを作成することで、下層の物理ネットワークを隠すプロトコルです。これは、信頼性のないベストエフォートのコネクションレス・パケット送達プロトコルです。ベストエフォートは、IP によって送信されたパケットが破損したり、順序を誤って到着したり、さらには重複したりする可能性を意味することにご注意ください。IP は、高位層プロトコルがこのような異変に対処するものと見なします。

コネクションレス・ネットワーク・プロトコルを使用する理由の 1 つは、階層接続指向ネットワークを使用していた特定のコンピューティング・セクターに対する依存度を最小限にすることでした。米国国防総省は、国の一部が破壊されても作動可能なネットワークの配備を計画しました。インターネットについてはこれが実現したことが証明されてきました。

3.1.1 IP アドレッシング

IP アドレスは、32 ビット符号なし 2 進値で表記されます。これは、通常、小数点付き 10 進数形式で表現されます。(たとえば、9.167.5.8 は有効な IP アドレスです。) 数値形式は IP ソフトウェアによって使用されます。IP アドレスと読みやすいシンボル名 (たとえば、myhost.ibm.com) の間のマッピングは、291 ページの 8.1 『Domain Name System (DNS)』で説明されている *Domain Name System (DNS)* によって行われます。

3.1.1.1 IP アドレス

IP アドレッシング標準は、RFC 1166 – Internet Numbers で説明されています。インターネット上のホストを識別するため、各ホストにはアドレス、*IP アドレス*、または場合によっては *インターネット・アドレス* が割り当てられます。ホストが複数のネットワークに接続している場合、それは *マルチホーム* と呼ばれ、各ネットワーク・インターフェースごとに 1 つの IP アドレスを持ちます。IP アドレスは、以下のような番号のペアで構成されています。

IP address = <network number><host number>

IP アドレスの *network number* の部分は、以下の 3 つのいずれかの地域インターネット・レジストリー (RIR) によって管理されています。

- American Registry for Internet Numbers (ARIN): このレジストリーは、北米、南米、カリブ海、およびサハラ以南のアフリカの、インターネット・プロトコル (IP) 番号の管理および登録を担当します。
- Reseaux IP Europeens (RIPE): このレジストリーは、欧州、中東、およびアフリカの一部の、インターネット・プロトコル (IP) 番号の管理および登録を担当します。
- Asia Pacific Network Information Centre (APNIC): このレジストリーは、アジア太平洋地域内のインターネット・プロトコル (IP) 番号の管理および登録を担当します。

IP アドレスは、*小数点付き 10 進数形式* (10 進表記の、ドットで連結された 4 つの 8 ビット値) で表記された 32 ビット数値です。(たとえば、128.2.7.9 は、ネットワーク番号が 128.2 で、ホスト番号が 7.9 である IP アドレスです。) IP アドレスのそのネットワークとホストの部分での分割に使用される規則については、以下で説明します。

IP アドレス 128.2.7.9 の 2 進形式は、以下のとおりです。

10000000 00000010 00000111 00001001

IP アドレスは、Internet (またはより一般的にインターネット) 上のホストを一意的に識別するために、IP プロトコルによって使用されます。厳密には、IP アドレスは、IP データグラムを送受信する能力を持つインターフェースを識別します。1 つのシステムは複数のこのようなインターフェースを持つこと

ができます。ただし、この単純化された定義が許容されるためには、ホストとルーターの両方が、少なくとも 1 つの IP アドレスを持っていなければなりません。IP データグラム (ホスト間で交換される基本データ・パケット) は、ホストに接続された物理ネットワークによって伝送されます。各 IP データグラムには、送信元 IP アドレスと宛先 IP アドレスが含まれます。特定の IP 宛先にデータグラムを送信する場合は、宛先 IP アドレスが物理アドレスに変換またはマップされなければなりません。これには、ネットワーク上で宛先の物理ネットワーク・アドレスを見つけるための伝送が必要になることがあります。(たとえば、LAN 上では、IP アドレスを物理 MAC アドレスに変換するために、118 ページの 3.4 『Address Resolution Protocol (ARP)』で説明されている Address Resolution Protocol が使用されます。)

3.1.1.2 クラス・ベース IP アドレス

IP アドレスの最初のビットは、アドレスの残りの部分のネットワークおよびホスト部分への分割方法を指定します。ネットワーク番号の代わりに、ネットワーク・アドレスや *netID* という用語が使用されることがありますが、RFC 1166 で使用されている正式な用語は、ネットワーク番号です。同様に、ホスト番号の代わりに、ホスト・アドレスや *hostID* という用語が使用されることがあります。

IP アドレスには、5 つのクラスがあります。これらは図 16 に示しています。

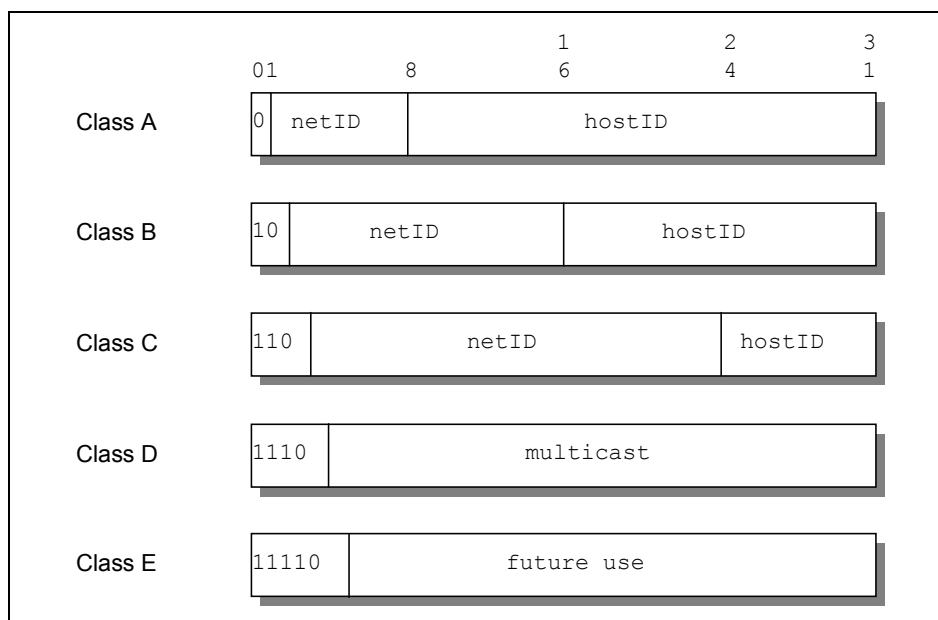


図 16 IP - IP アドレスの割り当て済みクラス

- ・クラス A アドレス：このアドレスは、IP アドレスの <network> 部分に 7 ビット、<host> 部分に 24 ビットを使用します。これは、 2^7-2 (126) 個のネットワークと、それぞれに $2^{24}-2$ (16777214) 個のホスト (合計で 20 億個を超えるアドレス) が可能です。
- ・クラス B アドレス：このアドレスは、IP アドレスの <network> 部分に 14 ビット、<host> 部分に 16 ビットを使用します。これは、 $2^{14}-2$ (16382) 個のネットワークと、それぞれに $2^{16}-2$ (65534) 個のホスト (合計で 10 億個を超えるアドレス) が可能です。
- ・クラス C アドレス：このアドレスは、IP アドレスの <network> 部分に 21 ビット、<host> 部分に 8 ビットを使用します。これは、 $2^{21}-2$ (2097150) 個のネットワークと、それぞれに 2^8-2 (254) 個のホスト (合計で 5 億個を超えるアドレス) が可能です。
- ・クラス D アドレス：このアドレスは、マルチキャスト (ブロードキャストの一種。ただし限定エリア内のもので、同じクラス D アドレスを使用するホストだけを対象とする) 用に予約済みです。
- ・クラス E アドレス：このアドレスは将来の利用のために予約済みです。

クラス A アドレスは、非常に多数のホストを擁するネットワークに適しています。クラス C アドレスは、少数のホストを擁するネットワークに適しています。これは、中程度のサイズのネットワーク (254 を超えるホストを擁するものか、254 を超えるホストが見込まれるもの) は、クラス B アドレスを使用する必要があることを意味します。ただし、小～中程度のサイズのネットワークの数は非常に急速に増大しています。この増大が衰えずに続けば、使用可能なクラス B ネットワーク・アドレスのすべてが、1990 年代半ばまでには使用されてしまう恐れがありました。これは、IP アドレス消費問題 (85 ページの 3.1.5 『IP アドレス枯渇問題』を参照) と呼ばれています。

IP アドレスが 2 つの部分に分割されていることから、全 IP アドレスの選択のための責任も分割されています。アドレスのネットワーク番号部分は、RIR によって割り当てられます。ホスト番号部分は、ネットワークを制御する機関によって割り当てられます。次のセクションで示すように、ホスト番号はさらに細かく分けることができます。この分割はネットワークを管理する機関によって制御されます。これは RIR によっては制御されません。

3.1.1.3 予約済み IP アドレス

すべてのビットが 0 またはすべてのビットが 1 の値を持つ IP アドレスのコンポーネントは、以下のような特別な意味を持ちます。

- ・すべてのビットが 0: ホスト番号部分のすべてのビットがゼロであるアドレスは、このホスト (<host address>=0 の IP アドレス) として解釈されます。ネットワーク番号部分のビットがすべてゼロの場合は、このネット

ワーク (<network address>=0 の IP アドレス) になります。ホストはネットワークを経由して通信しようとするとき、ネットワーク IP アドレスが分からなければ、<network address>=0 のパケットを送信することができます。ネットワーク上の他のホストは、このアドレスがこのネットワークを表していると解釈します。これらのホストの応答には完全修飾ネットワーク・アドレスが含まれ、送信側はこれを将来の利用のために記録します。

- すべてのビットが 1: すべてのビットが 1 であるアドレスは、すべてのネットワークまたはすべてのホストとして解釈されます。たとえば、以下はネットワーク 128.2 (クラス B アドレス) 上のすべてのホストを意味します。

128.2.255.255

これには、有効な <network address> とブロードキャスト <host address> の両方が含まれるので、直接ブロードキャスト・アドレスと呼ばれます。

- ループバック: クラス A ネットワーク 127.0.0.0 は、ループバック・ネットワークとして定義されています。このネットワークのアドレスは、ローカル・システム内のデータを処理するインターフェースに割り当てられています。これらループバック・インターフェースは、物理ネットワークにはアクセスしません。

3.1.2 IP サブネット

インターネットの爆発的な成長により、割り当て IP アドレスの原則は柔軟性を失い、ローカル・ネットワーク構成を容易に変更できなくなりました。この変更とは、以下の場合に実行されることがあります。

- ある位置に新しいタイプの物理ネットワークがインストールされる。
- ホストの数の増大により、ローカル・ネットワークを複数の別個のネットワークに分割する必要がある。
- 距離の拡大により、1つのネットワークを、ゲートウェイを間に持つ複数のより小さなネットワークに分割する必要がある。

追加の IP ネットワーク・アドレスを要求する必要を避けるため、IP サブネット化の概念が導入されました。サブネットの割り当ては、ローカルに行われます。ネットワーク全体は、外界からは引き続き 1つの IP ネットワークに見えます。

IP アドレスのホスト番号部分は、第 2 のネットワーク番号とホスト番号とに分かれています。この第 2 のネットワークは、サブネットワークまたはサブネットと呼ばれます。現在主なネットワークはいくつかのサブネットから構成されています。IP アドレスは以下のようなものとして解釈されます。

<network number><subnet number><host number>

サブネット番号とホスト番号の組み合わせは、しばしばローカル・アドレスまたは IP アドレスのローカル部分と呼ばれます。サブネット化は、リモート・ネットワークに透過的な形でインプリメントされます。サブネットを持つネットワーク内のホストは、サブネット化構造を認識します。しかし、異なるネットワーク内のホストは認識しません。このリモート・ホストは依然として IP アドレスのローカル部分をホスト番号と見なします。

IP アドレスのローカル部分のサブネット番号とホスト番号への分割は、ローカル管理者によって選択されます。サブネットの形成には、ローカル部分の任意のビットを使用できます。分割は 32 ビットのサブネット・マスクを使用して行われます。サブネット・マスク内の値がゼロ・ビットのビットは、ホスト番号に割り当てられた位置を表します。値が 1 のビットは、サブネット番号に割り当てられた位置を表します。サブネット・マスク内の元のネットワーク番号に属するビット位置は 1 に設定されますが、使用はされません (一部のプラットフォームの構成ではこの値が実際に 1 ではなくゼロで指定されていましたが、いずれにしても使用されません)。IP アドレスと同様、サブネット・マスクは通常、小数点付き 10 進数形式で表現されます。

すべてゼロのビットおよびすべて 1 のビットの特別な扱いは、サブネット化されていない IP アドレスの両方の部分に適用されるのと同様に、サブネット化された IP アドレスの 3 つの各部分にも適用されます (70 ページの 3.1.1.3 『予約済み IP アドレス』を参照)。たとえば、クラス B ネットワークをサブネット化する場合は、以下のいずれかの構造を使用できます。

- 最初のオクテットがサブネット番号で、2 番目のオクテットがホスト番号。この場合は、それぞれが最大で 2^8-2 (254) 個のホストを持つ、 2^8-2 (254) 個のサブネットが可能です。ここでは、すべて 1 およびすべてゼロの場合を考慮に入れて、可能数から 2 を減算していること思い出してください。サブネット・マスクは 255.255.255.0 です。
- 最初の 12 ビットがサブネット番号に使用され、最後の 4 ビットがホスト番号に使用される。この場合は、 $2^{12}-2$ (4094) 個のサブネットが可能です。1 つのサブネットのホスト数は 2^4-2 (14) 個だけです。サブネット・マスクは 255.255.255.240 です。

この例では、これ以外にも、アドレスのサブネット部分とホスト部分の割り当ての方法があり得ます。この構造を定義する場合は、事前にサブネットおよびホストの数と将来の要件について考慮する必要があります。最後の例において、サブネット化されたクラス B ネットワークは 16 ビットを持ち、これがサブネット番号フィールドとホスト番号フィールドに分けられます。ネットワーク管理者は、それぞれが少数のホストを擁する多数のサブネットか、それぞれが多数のホストを擁する少数のサブネットを定義することになります。

ローカル・アドレスのサブネット部分を割り当てるとき、その目的は、いくつかのビットをサブネット番号に割り当て、残りをローカル・アドレスに割り当てることです。したがって、サブネット番号には、ローカル・アドレス部分の先頭の連続するビットのブロックを使用するのが普通です。こうすると、アドレスは判読しやすくなります。(サブネットが 8 または 16 ビットを使用する場合は特にそう言えます。) このアプローチでは、上記のサブネット・マスクはどちらも「受け入れ可能」マスクになります。255.255.252.252 および 255.255.255.15 などのマスクは「受け入れ不可」です。実際、大半の TCP/IP インプリメンテーションは連続的でないサブネット・マスクをサポートしません。この使用は一般的に奨励されていません。

3.1.2.1 サブネット化のタイプ

サブネット化には、2 つのタイプ (静的および可変長) があります。可変長サブネット化は、静的よりも柔軟です。ネイティブ IP ルーティングと RIP バージョン 1 は静的サブネット化のみをサポートしています。しかし、RIP バージョン 2 は可変長サブネット化をサポートしています (143 ページの第 4 章『ルーティング・プロトコル』を参照)。

静的サブネット化

静的サブネット化は、同じネットワークから取得されるすべてのサブネットが同じサブネット・マスクを使用することを意味します。これはインプリメントが単純で保守も簡単ですが、小さなネットワークではアドレス・スペースを浪費する可能性があります。サブネット・マスク 255.255.255.0 を使用する 4 つのホストからなるネットワークについて考慮しましょう。この割り振りでは、250 の IP アドレスが浪費されています。すべてのホストとルーターは静的サブネット化をサポートしている必要があります。

可変長サブネット化

可変長サブネット化が使用されている場合は、同じネットワーク内に割り振られたサブネットがそれぞれ異なるサブネット・マスクを使用できます。わずかなホストを持つ小さなサブネットはこの必要を満たすマスクを使用できます。多数のホストを持つサブネットには異なるサブネット・マスクが必要です。個々のサブネットの必要に応じてサブネット・マスクを割り当てられることは、ネットワーク・アドレスの節約に寄与します。可変長サブネット化では、各サブネットに必要な数のホストをサポートするだけの十分なアドレスが含まれるように、ネットワークが分割されます。

既存のサブネットは、サブネット・マスクのサブネット部分にもう 1 つのビットを追加することによって、2 つの部分に分割することができます。ネットワーク内の他のサブネットはこの変更の影響を受けません。

静的サブネット化と可変長サブネット化の混合

すべての IP 装置が可変長サブネット化をサポートしているとは限りません。最初は、静的サブネット化だけをサポートするホストの存在が可変長サブ

ネット化の使用を妨げるように見えるかもしれませんが、しかし、そのとおりではありません。異なるマスクをホストから隠すために、サブネットを相互接続するルーターが使用されています。ホストは引き続き基本の IP ルーティングを使用します。このようにして、サブネット化の複雑さは専用ルーターにまかせます。

3.1.2.2 静的サブネット化の例

図 17 に表示されたクラス A ネットワークについて考えましょう。

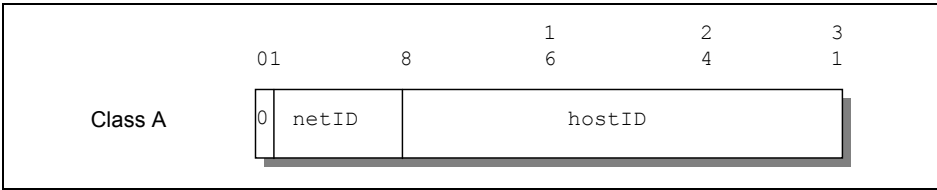


図 17 IP - サブネットのないクラス A アドレス

これは以下の IP アドレスを使用しています。

```
00001001 01000011 00100110 00000001    a 32-bit address
      9      67      38      1      decimal notation (9.67.38.1)

9.67.38.1                                is an IP address (class A) having

9                                           as the <network address>
67.38.1                                   as the <host address>
```

ネットワーク管理者は、8 番目から 25 番目までのビットを、サブネット・アドレスを示すものとして選択するとします。その場合、26 番目から 31 番目までのビットは実際のホスト・アドレスを示します。図 18 は、元のクラス A アドレスから派生した、サブネット化されたアドレスを示しています。

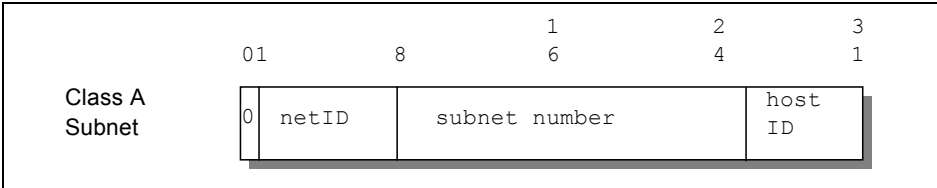


図 18 IP - サブネット・マスクとサブネット・アドレスを持つクラス A アドレス

元のホスト・アドレス・フィールドのどのビットがサブネット番号を示すのかを識別するために、サブネット・マスクと呼ばれるビット・マスクが使用されます。上記の例では、サブネット・マスクは 255.255.255.192 (または、

ビット表記では 11111111 11111111 11111111 11000000) です。規則上、マスクには <network address> も含まれるという点にご注意ください。

「すべてのビットが 0」および「すべてのビットが 1」という制約により、これは $2^{18}-2$ (1 ~ 262143) 個の有効サブネットを定義します。この分割によって、それぞれが最大で 2^6-2 (62) 個のホストを擁する、262142 個のサブネットができます。

サブネット番号に設定された値は、有効でないビットが 0 に設定されたフル・オクテットの値を取ります。たとえば、このサブネット・マスク内の 16 進値 01 は、8 ビット値 01000000 を取ります。これは、サブネット値 64 を示します。

255.255.255.192 をサンプルのクラス A アドレス 9.67.38.1 に適用すると、以下の情報が提供されます。

```
00001001 01000011 00100110 00000001 = 9.67.38.1 (class A address)
11111111 11111111 11111111 11----- 255.255.255.192 (subnet mask)
===== logical_AND
00001001 01000011 00100110 00----- = 9.67.38.0 (subnet base address)
```

このとき、以下のホスト・アドレスはそのまま残されます。

```
----- ----- ----- --000001 = 1 (host address)
```

IP では、上記の logical_AND 演算によって同じ結果が生成されるすべてのホスト・アドレスが、同じローカル・ネットワーク上にあるものと見なされます。これは、サブネット環境での IP データグラムのルーティングに関して重要な点です (77 ページの 3.1.3 『IP ルーティング』を参照)。

実際のサブネット番号は以下のとおりです。

```
----- 01000011 00100110 00----- = 68760 (subnet number)
```

このサブネット番号は相対番号です。つまり、これは、指定されたサブネット・マスクを持つネットワーク 9 の 68760 番目のサブネットです。この番号は、このホストが割り当てられている実際の IP アドレス (9.67.38.1) との類似性はありません。これには、IP ルーティングにおける意味はありません。

元の <host address> の <subnet><host> への分割は、ネットワーク管理者によって選択されます。<subnet> フィールドのすべてゼロの値とすべて 1 の値は予約済みです。

注：

使用可能な IP アドレスの範囲は急速に減っているのです、現在では多くのルーターが <subnet> フィールドでのすべてゼロおよびすべて 1 の使用をサポートしています。これは定義されている標準とは異なります。

3.1.2.3 可変長サブネット化の例

クラス C ネットワーク 165.214.32.0 が割り当てられた会社について考慮しましょう。この会社は、それぞれが以下の数のホストを擁する 5 つの別個のネットワークにこのアドレス範囲を分割する必要があります。

- サブネット #1: 50 個のホスト
- サブネット #2: 50 個のホスト
- サブネット #3: 50 個のホスト
- サブネット #4: 30 個のホスト
- サブネット #5: 30 個のホスト

これは、静的サブネット化では実行できません。この例の場合、静的サブネット化を行うと、ネットワークはそれぞれが 64 個のホストを擁する 4 つのサブネットか、それぞれが 32 個のホストを擁する 8 つのサブネットに分割されます。このサブネット割り振りでは、前述の要件は満たされません。

ネットワークを 5 つのサブネットに分割するには、複数のマスクを定義する必要があります。このネットワークは、255.255.255.192 のマスクを使用して、それぞれが 64 個のホストを擁する 4 つのサブネットに分割できます。さらにその 4 番目のサブネットを、255.255.255.224 のマスクを使用することによって、それぞれが 32 個のホストを擁する 2 つのサブネットに分割できます。そうすると、それぞれが 64 個のホストを擁する 3 つのサブネットと、それぞれが 32 個のホストを擁する 2 つのサブネットができます。これは、前述の要件を満たします。

3.1.2.4 サブネット・マスクの判別

通常、ホストはサブネット・マスクを構成ファイルに保管します。しかし、それができないこともあります（たとえば、ディスクレス・ワークステーションの場合）。ICMP プロトコルには、2 つのメッセージ（アドレス・マスク要求とアドレス・マスク応答）が含まれます。これらによって、ホストは正しいサブネット・マスクをサーバーから取得することができます（115 ページの 3.2.1.10 『アドレス・マスク要求 (17) およびアドレス・マスク応答 (18)』を参照）。

3.1.2.5 アドレッシング・ルーターとマルチホーム・ホスト

ホストが複数のネットワークまたはサブネットに対する物理接続を持つ場合、それはマルチホームであると言えます。ルーターの目的はネットワークまた

はサブネットを結合することなので、デフォルトでは、すべてのルーターがマルチホームです。マルチホーム・ホストは、各ネットワーク・アダプターに関連した異なる IP アドレスを持ちます。各アダプターは、異なるサブネットまたはネットワークに接続します。

3.1.3 IP ルーティング

IP 層の重要な機能は、*IP ルーティング*です。これは、ルーターが異なる物理ネットワークを相互接続するための基本メカニズムを備えています。装置は、正規のホストとルーターの両方として同時に機能することができます。

このタイプのルーターは、部分的なルーティング情報を持つルーターと呼ばれます。このルーターは、以下の 4 種類の宛先についての情報のみを持ちます。

- ルーターが接続している物理ネットワークのいずれかに直接接続されているホスト。
- ルーターに明示的に定義されているホストまたはネットワーク。
- ルーターがその ICMP リダイレクト・メッセージを受け取ったホストまたはネットワーク。
- 他のすべての宛先のデフォルト。

完全な機能のルーターをインプリメントするには、追加のプロトコルが必要です。このタイプのルーターは、環境内の他のルーターと情報を交換することができるので、大半のネットワークにおいて不可欠です。このルーターによって使用されるプロトコルについては、143 ページの第 4 章『ルーティング・プロトコル』で検討します。

IP ルーティングには、2 つのタイプ (直接と間接) があります。

3.1.3.1 直接ルーティング

宛先ホストが送信元ホストと同じ物理ネットワークに接続されている場合は、IP データグラムを直接交換することができます。これは、IP データグラムを物理ネットワーク・フレーム内にカプセル化することによって行われます。これは直接送達といい、直接ルーティングと呼ばれます。

3.1.3.2 間接ルーティング

間接ルーティングは、宛先ホストが、送信元ホストに直接接続されたネットワークに接続していないときに行われます。宛先ホストに到達するには、1 つ以上の IP ゲートウェイを経由しなければなりません。(TCP/IP 用語では、ゲートウェイとルーターという用語が交換可能な語として使用されていることにご注意ください。これはルーターの任務を実行するシステムを指します。)IP ルーティング・アルゴリズムにおいて、最初のゲートウェイ (最初の

ホップ) のアドレスは間接経路と呼ばれます。最初のゲートウェイのアドレスは、送信元ホストが宛先ホストにパケットを送信するために必要とする唯一の情報です。

場合によって、同じ物理ネットワーク上に複数のサブネットが定義されることがあります。送信元と宛先のホストが同じ物理ネットワークに接続しながら、異なるサブネットに定義されている場合、この装置ペアの間の通信には間接ルーティングが使用されます。サブネット間でトラフィックを転送するには、ルーターが必要です。

図 19 は、直接および間接経路の例を示しています。

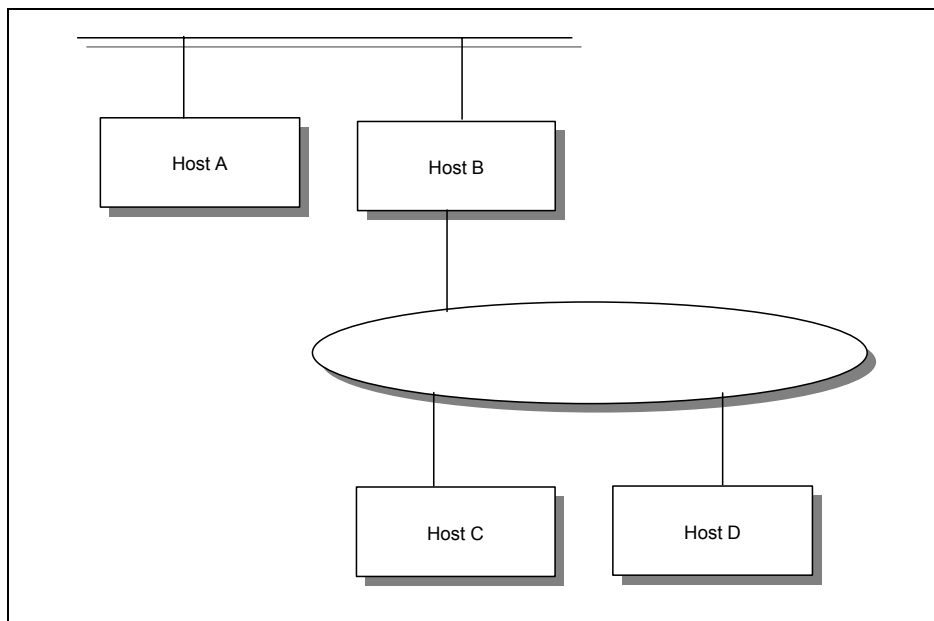


図 19 IP - 直接および間接経路 - ホスト C は、ホスト B および D への直接経路と、ゲートウェイ B を介したホスト A への間接経路を持ちます。

3.1.3.3 IP ルーティング・テーブル

直接経路の判別は、ローカル・インターフェースのリストに基づきます。これは初期化時の IP ルーティング・プロセスで自動的に作成されます。さらに、ネットワークおよび関連するゲートウェイ (間接経路) のリストを構成することもできます。このリストは IP ルーティングの簡易化のために使用されます。各ホストは、以下の間のマッピングのセットを保持します。

- 宛先 IP ネットワーク・アドレス
- 次のゲートウェイへの経路

この情報は、IP ルーティング・テーブルと呼ばれるテーブルに保管されます。このテーブルには、以下の 3 つのタイプのマッピングが含まれます。

- 1. ローカルに接続されたネットワークを示す直接経路
- 2. 1 つ以上のゲートウェイを介して到達できるネットワークを示す間接経路
- 3. 宛先 IP ネットワークが上記のタイプ 1 および 2 のマッピングの中にない場合に使用される (直接または間接) 経路を含むデフォルトの経路。

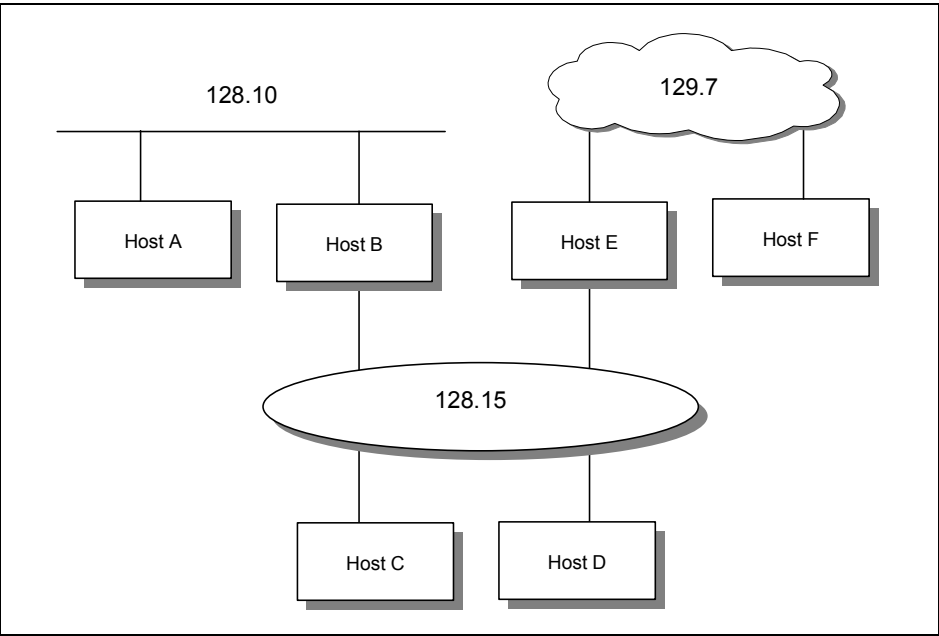


図 20 IP - ルーティング・テーブル・シナリオ

図 20 は、ネットワークの一例です。ホスト D のルーティング・テーブルには、以下の (シンボリック) エントリーが含まれているとします。

destination	router	interface
129.7.0.0	E	lan0
128.15.0.0	D	lan0
128.10.0.0	B	lan0
default	B	lan0
127.0.0.1	loopback	lo

D は、ネットワーク 128.15.0.0 に直接接続されているので、このネットワークへの直接経路を保持しています。ただし、ネットワーク 129.7.0.0 および 128.10.0.0 に接続するには、E および B をそれぞれ経由する間接経路を持つ

ていなければなりません。なぜなら、これらのネットワークは直接には接続されていないからです。

ホスト F のルーティング・テーブルには、以下の (シンボリック) エントリーが含まれているとします。

destination	router	interface
129.7.0.0	F	wan0
default	E	wan0
127.0.0.1	loopback	lo

129.7.0.0 ネットワーク上にない各ホストへ到達するにはホスト E を経由する必要があるので、単純に F は E を経由するデフォルト経路を保持しています。

3.1.3.4 IP ルーティング・アルゴリズム

IP は、図 21 のような固有なアルゴリズムを使用してデータグラムをルーティングします。

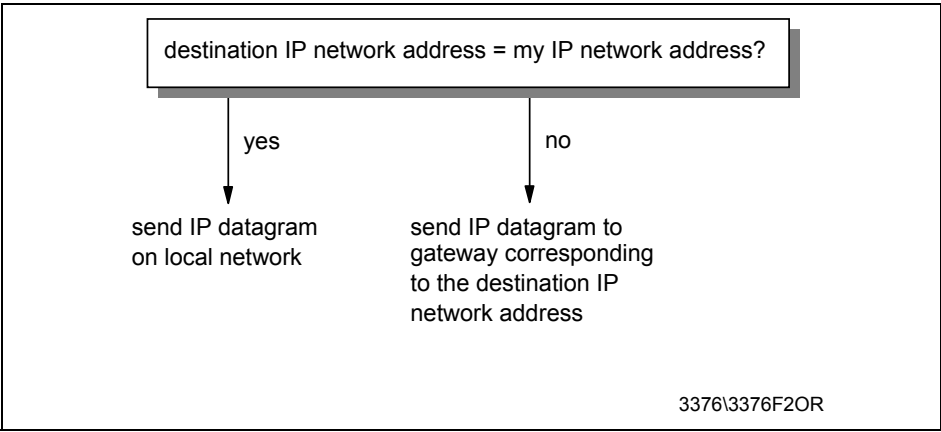


図 21 IP - サブネットがない場合のルーティング

サブネットを識別する場合、IP ルーティング・アルゴリズムは図 22 のように更新されます。

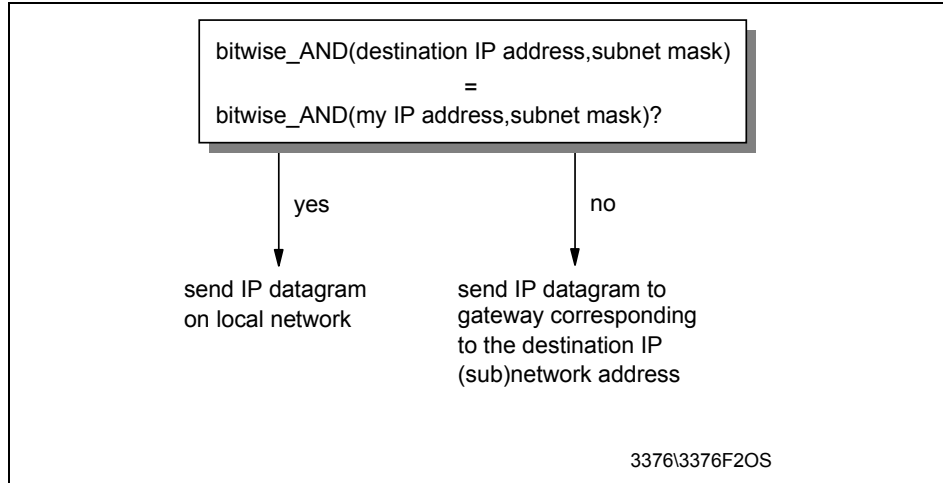


図 22 IP - サブネットがある場合のルーティング

この変更は、以下を示しています。

- このアルゴリズムは、汎用の IP アルゴリズムに対する変更を表しています。したがって、この方法で操作できるようになるには、特定のゲートウェイにこの新しいアルゴリズムが含まれていなければなりません。一部のインプリメンテーションはまだ汎用のアルゴリズムを使用している可能性があり、サブネット化された他のネットワーク内のホストとは通信できませんが、サブネット化されたネットワーク内では機能しません。
- すべてのホスト（ルーターだけでなく）で IP ルーティングが使用されている場合、サブネット内のすべてのホストが以下を持っている必要があります。
 - サブネット化をサポートする IP ルーティング・アルゴリズム。
 - 同じサブネット・マスク（サブネット内にサブネットが作成されていない場合）。
- 任意のホストの IP インプリメンテーションがサブネット化をサポートしていない場合、そのホストはその独自のサブネット内のどのホストとも通信できますが、同じネットワーク内の別のサブネット上にあるマシンとはどれとも通信できません。これは、ホストが 1 つの IP ネットワークしか認識しておらず、そのルーティングで、ローカル・サブネット上のホストに直接送信される IP データグラムと、ルーターを介して異なるサブネットへ送信されなければならないデータグラムが区別できないためです。

1 つ以上のホストがサブネット化をサポートしていない場合、同じゴールに到達するための代替の方法はプロキシ ARP の形式になります。この場合、

シングルホーム・ホストの IP ルーティング・アルゴリズムの変更は不要です。ネットワーク内のサブネット間のルーターには変更が必要です (122 ページの 3.4.4 『プロキシ ARP または透過的なサブネット化』を参照)。

図 23 は、全 IP アルゴリズムを表しています。

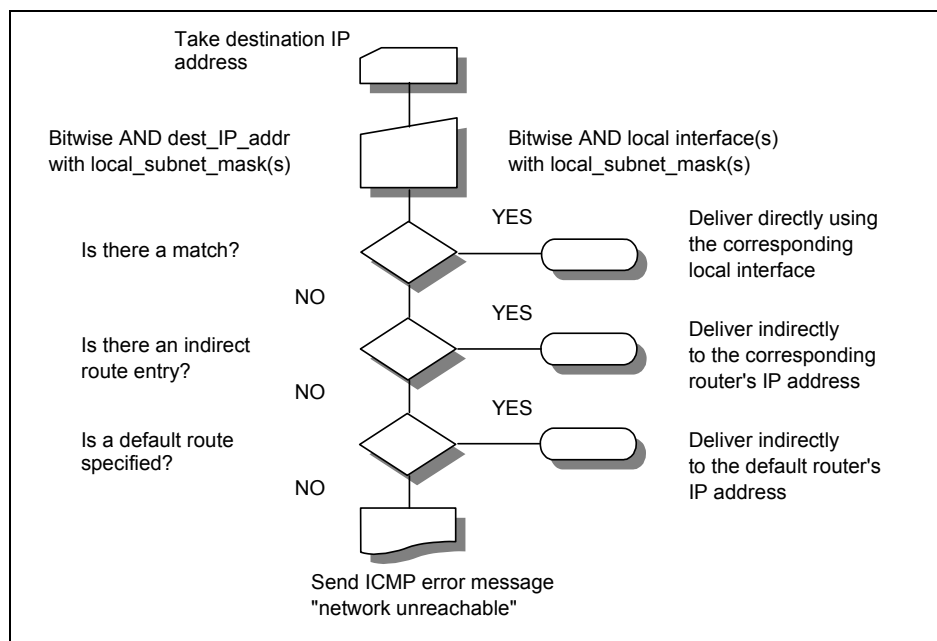


図 23 IP - ルーティング・アルゴリズム (サブネットがある場合)

注

1. これは反復プロセスです。これは、宛先ホストを除いた、データグラムを処理するすべてのホストに適用されます。
2. ルーティング・テーブルとルーティング・アルゴリズムは、IP ネットワーク内のどのホストに対してもローカルです。IP データグラムを転送するために、ルーターは通常ネットワーク内の他のルーターとそのルーティング・テーブル情報を交換します。これは、特殊なルーティング・プロトコルを使用して行われます (143 ページの第 4 章 『ルーティング・プロトコル』を参照)。

3.1.4 配送方式 - ユニキャスト、ブロードキャスト、マルチキャスト、およびエニーキャスト

IP アドレスの大半は単一の宛先を指し、これはユニキャスト・アドレスと呼ばれます。ユニキャスト接続は、単一の送信元と単一の宛先の間の 1 対 1 の関係を指定します。この他に、複数の宛先のアドレッシングに使用される 3 つの特別なタイプの IP アドレス (ブロードキャスト・アドレス、マルチキャスト・アドレス、およびエニーキャスト・アドレス) が存在します。これらの操作は、図 24 に示しています。

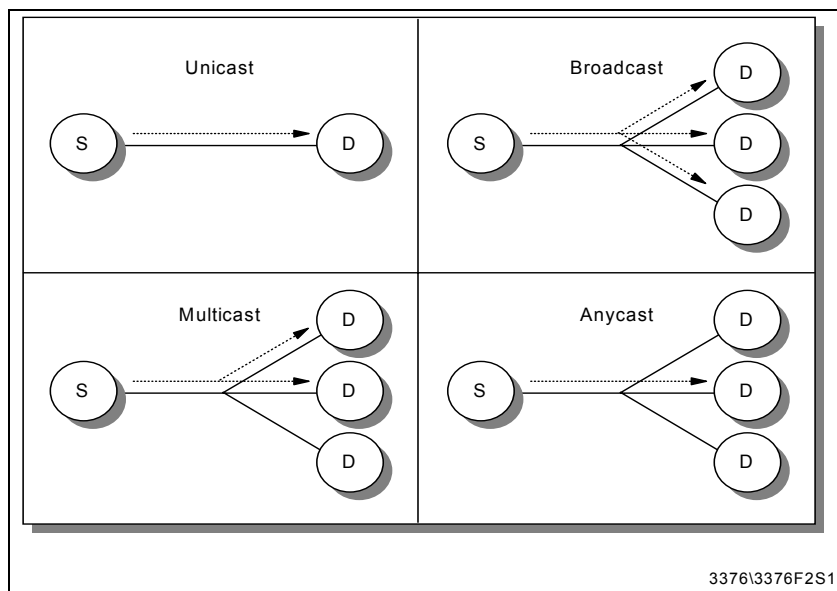


図 24 IP - パケット送達モード

コネクションレス・プロトコルは、ユニキャスト、ブロードキャスト、マルチキャスト、またはエニーキャスト・メッセージを送信できます。接続指向プロトコルは、ユニキャスト・アドレスのみを使用できます (特定のホストのペアの間に接続がなければなりません)。

3.1.4.1 ブロードキャスト

ブロードキャスト・アドレスは送信元アドレスとしては無効です。これらは宛先アドレスを指定していなければなりません。ブロードキャスト・アドレスには、以下のようなさまざまなタイプがあります。

- 限定ブロードキャスト・アドレス：これは、アドレス 255.255.255.255 (IP アドレスのすべての部分ですべてのビットが 1) を使用します。これは、ローカル・サブネット上のすべてのホストを指します。このことはすべて

のホストに認識されます。ホストは IP 構成情報を必要としません。ルーターはこのパケットを転送しません。

この規則に対する 1 つの例外は、*BOOTP 転送* と呼ばれます。BOOTP プロトコルは、限定ブロードキャスト・アドレスを使用して、ディスクレス・ワークステーションがブート・サーバーとコンタクトできるようにします。BOOTP 転送は、一部のルーターで使用可能な構成オプションです。この機能がない場合は、各サブネット上に BOOTP サーバーが必要です (125 ページの 3.6 『Bootstrap protocol (BOOTP)』を参照)。

- ネットワーク宛ブロードキャスト・アドレス：これはサブネット化されていない環境で使用されます。ネットワーク番号は有効なネットワーク番号で、ホスト番号はすべて 1 (たとえば、128.2.255.255) です。このアドレスは、指定されたネットワーク上のすべてのホストを指します。ルーターはこれらのブロードキャスト・メッセージを転送する必要があります。これは、サブネット化されていないネットワーク上での ARP 要求 (118 ページの 3.4 『Address Resolution Protocol (ARP)』を参照) で使用されます。
- サブネット宛ブロードキャスト・アドレス：ネットワーク番号が有効なネットワーク番号で、サブネット番号が有効なサブネット番号で、ホスト番号がすべて 1 の場合、アドレスは指定されたサブネット上のすべてのホストを指します。送信側のサブネットと宛先サブネットとが異なるサブネット・マスクを持っている可能性があるため、送信側は何とかして宛先で使用されているサブネット・マスクを突き止める必要があります。実際のブロードキャストは、そのサブネットへのデータグラムを受信するルーターによって実行されます。
- すべてのサブネット宛てのブロードキャスト・アドレス：ネットワーク番号が有効なネットワーク番号で、ネットワークはサブネット化され、ローカル部分がすべて 1 (たとえば、128.2.255.255) の場合、そのアドレスは指定されたネットワーク内のすべてのサブネット上のすべてのホストを指します。基本的にルーターはすべてのサブネットに対してブロードキャストを伝搬できますが、その実行は必須ではありません。そして実際に実行しません。このようなブロードキャストが望ましい状況はほとんどありません。構成を誤れば、問題が発生する可能性があります。サブネット化されたクラス A ネットワーク内の、誤って構成されたホスト 9.180.214.114 について考えてください。装置が、9.180.214.255 の代わりにアドレス 9.255.255.255 をローカル・ブロードキャスト・アドレスとして用いて構成されている場合、このネットワーク内のすべてのルーターはすべてのクライアントに要求を転送します。

すべてのサブネット宛ブロードキャスト・アドレスを信頼する場合、ルーターは*逆パス転送*と呼ばれるアルゴリズムを使用して、ブロードキャスト・メッセージの無制限な増大を防ぎます。このアルゴリズムの詳細については、RFC 922 を参照してください。

3.1.4.2 マルチキャスト

IP データグラムがサブネットに対するブロードキャストである場合、これはサブネット上のすべてのホストによって受信されます。各ホストは、パケットを処理して宛先プロトコルがアクティブかどうかを判断します。これがアクティブでないと、この IP データグラムは廃棄されます。マルチキャストでは宛先グループが選択されるのでこのオーバーヘッドは防止されます。

各グループはクラス D の IP アドレスで表されます。各マルチキャスト・アドレスごとに、0 個以上のホストのセットが、そのアドレスにアドレッシングされたパケットを受信します。このホストのセットは、ホスト・グループと呼ばれます。マルチキャスト・アドレスに送信されたパケットは、対応するホスト・グループのメンバーにのみ転送されます。マルチキャストは、1 対多の接続を可能にします (237 ページの第 6 章『IP マルチキャスト』を参照)。

3.1.4.3 エニーキャスト

同じ IP サービスが複数のホストによって提供されることがあります。たとえば、ユーザーが FTP を介してファイルをダウンロードするとき、そのファイルが複数の FTP サーバーで入手可能なことがあります。同じサービスをインプリメントしているホストは、そのサービスを必要とする他のホストにエニーキャスト・アドレスを提供します。接続は、応答するエニーキャスト・アドレス・グループ内の最初のホストに対して作成されます。このプロセスは、受信側への最善の接続を持つホストによるサービスの提供を保証するために使用されます。

エニーキャスト・サービスは、IPV6 (592 ページの 17.2.3『IPv6 アドレッシング』を参照) に組み込まれています。

3.1.5 IP アドレス枯渇問題

ここ数年、インターネット上のネットワークの数は 1 年ごとにおよそ 2 倍になっています。しかし、クラス A、B、および C ネットワークの使用量は、大きく異なります。1980 年代後半に割り当てられた新しいネットワークのほとんどすべてはクラス B であり、1990 年の時点では、その傾向が続けば、1994 年中には最後のクラス B ネットワーク番号が割り当てられてしまうように思われました。これに対し、クラス C ネットワークはほとんど使用されていませんでした。

この傾向の理由は、クラス B ネットワークは 65534 個までのホストを収容するのにに対し、最大のホストが 254 個のクラス C ネットワークは、小さな初期ネットワークの成長の可能性でさえ極端に制限していたため、ほとんどの潜在ユーザーは、クラス B ネットワークを自分たちの予期するニーズを満たす大きさと見なしたことにあります。さらに、割り当てられていたクラス B ネットワークのほとんどは、小さなネットワークでした。65,534 個ものホスト・アドレスを必要とするネットワークは比較的少数ですが、254 個のホス

トが適当な限度になるネットワークはほとんどありません。要約すると、IP アドレスのクラス A、クラス B、およびクラス C の分割は論理的で使いやす (これらはバイト単位で行われるため) もではありませんが、今となってみれば、クラス C ネットワークは小さすぎて大半の組織では役立てられず、クラス B ネットワークは大きすぎて巨大な組織でなければこれに高密度で移植することはできないので、これらが最も実用的であるとは言えません。

1996 年 5 月には、すべてのクラス A アドレスと、クラス B アドレスの 61.95 % およびクラス C IP ネットワーク・アドレスの 36.44 % が割り振られるか割り当てられるかされました。このコンテキストでの「割り当て済み」および「割り振り済み」という用語には、以下のような意味があります。

- 割り当て済み：使用されているネットワーク番号の数。クラス C の数字は少々不正確です。なぜならこの数字には、RIPE に割り振られ、その後に割り当てられたにもかかわらずまだ「割り振り済み」として記録されているヨーロッパの多数のクラス C ネットワークが含まれていないからです。
- 割り振り済み：これには、すべての割り当て済みのネットワークに加え、IANA によって予約されている (たとえば、63 クラス A ネットワークはすべて IANA によって予約されている) か、IANA によって領域レジストリーに割り振られていて後にこれらのレジストリーによって割り当てられるネットワークが含まれます。

これらの番号を見るもう 1 つの方法は、使用されているアドレス・スペースの割合を調べることです。テーブル内の数字は、たとえば、クラス A アドレス・スペースが残りを含合わせたものと同じ大きさであるとか、単一のクラス A ネットワークが理論上は 66,000 個のクラス C ネットワークと同じ数のホストを持てるということを表すものではありません。

1990 年以来、割り当て済みのクラス B ネットワークの数は割り当て済みネットワークの合計数よりかなり低い率で増加しており、予想されていたクラス B ネットワーク番号の枯渇はまだ起きていません。この理由は 1990 年後半に既存のアドレス・スペースの節約のため (特にクラス B アドレス・スペースの枯渇を防ぐため) に、ネットワーク番号の割り振りのポリシーが変更されたためです。この新しいポリシーは以下のように要約することができます。

- クラス A アドレス・スペースの上半分 (ネットワーク番号 64 ~ 127) は、新しい番号付け方式への遷移に使用できるように無期限に予約されています。
- クラス B ネットワークは、必要であることを明確に示せる組織にのみ割り当てられます。同じことはクラス A アドレスにももちろん当てはまります。クラス B ネットワークの要件は、要求元の組織が以下を有していることです。
 - その組織ネットワーク内の 32 個を超えるサブネットについて記述したサブネット化計画

- 4096 を超えるホスト

クラス A ネットワークの要件は、個々の場合に応じて調整されます。

- クラス B ネットワークの要件を満たさない組織には、クラス C ネットワーク番号の連続した番号のブロックが割り当てられます。
- クラス C アドレス・スペースの下半分 (ネットワーク番号 192.0.0 ~ 207.255.255) は、8 つのブロックに分割され、それは以下のような地方分権に割り振られています。

192.0.0 - 193.255.255多地域

194.0.0 - 195.255.255ヨーロッパ

196.0.0 - 197.255.255その他

198.0.0 - 199.255.255北米

200.0.0 - 201.255.255中南米

202.0.0 - 203.255.255環太平洋地域

204.0.0 - 205.255.255その他

206.0.0 - 207.255.255その他

「その他」として定義された範囲は、地域境界の制約外の、柔軟性が求められる場所です。「多地域」として定義された範囲には、この新しい方式が採用される前に割り当てられたクラス C ネットワークが含まれます。192 ネットワークは InterNIC によって割り当てられ、193 ネットワークは以前にヨーロッパの RIPE に割り振られました。

クラス C アドレス・スペースの上半分 (208.0.0 ~ 223.255.255) は、未割り当ておよび未割り振りのままです。

- 組織が 1 つの範囲のクラス C ネットワーク番号を持つところでは、提供されている範囲は**ビット単位で連続した 1 つの範囲のネットワーク番号**として割り当てられ、この範囲内のネットワークの数は 2 の累乗になります。つまり、この範囲内のすべての IP アドレスが共通の接頭部を持ち、その接頭部を持つアドレスはすべてこの範囲内にあることになります。たとえば、1500 の IP アドレスを必要とするヨーロッパの組織は、ヨーロッパのネットワーク用に予約されている番号スペース (194.0.0 ~ 195.255.255) から 8 つのクラス C ネットワーク番号 (2048 の IP アドレス) が割り当てられ、これらのネットワーク番号の最初のものは 8 で割り切れます。この規則を満たすアドレスの範囲は 194.32.136 ~ 194.32.143 で、この場合、範囲は 21 ビットの接頭部 194.32.136 または B '110000100010000010001' を持つすべての IP アドレスで構成されます。

連続して割り当てられるネットワーク番号の最大数は、18 ビットの接頭部に対応する 64 です。4096 より多く 16,384 より少ないアドレスを必要とする組織は、クラス B または一続きのクラス C アドレスのいずれかを要求

できます。通常、割り当てられるクラス C ネットワークの数は、2 年間の見通しを基本に、必要な数の IP アドレスを組織に提供するために必要な最少数です。しかし、場合によっては、組織は、複数のネットワークが別個に取り扱われるよう要求できます。たとえば、600 個のホストを擁する組織は、通常 4 つのクラス C ネットワークが割り当てられます。しかし、これらのホストが、1 つの LAN が 50 ～ 70 個のホストを持つ 10 個のトークンリング LAN の間で分散されている場合、組織は 10 ビットのローカル・アドレス範囲内に 10 のサブネットを見つけなければならないので、このような割り振りは深刻な問題の原因になります。これは、少なくともいくつかの LAN が、LAN 1 つあたり 62 個のホストしか許可しないサブネット・マスク 255.255.255.192 を持っていることを意味します。この規則の意図は、組織を小さなネットワークの複雑なサブネット化の中に強制して入れないようにすることであり、組織は各 LAN について 1 つずつ、10 個の異なるクラス C 番号を要求する必要があります。

現在の規則は、RFC 1466 を更新する RFC 2050 – Internet Registry IP Allocation Guidelines にあります。クラス C ネットワーク番号の割り振りのための規則の理由は、続くいくつかのセクションで明らかにします。この方法でのクラス C ネットワーク番号の使用はクラス B アドレス・スペースの枯渇を防いできましたが、これは、IP に重要な全体的なアドレス・スペース制約に対する永続的なソリューションではありません。長期的なソリューションについては、579 ページの第 17 章『IP Version 6』で説明します。

3.1.6 イントラネット - 専用 IP アドレス

IP アドレス・スペースを確保するためのもう 1 つのアプローチは、RFC 1918 – Address Allocation for Private Internets で説明されています。この RFC は、IP アドレスが全世界的に固有でなければならないという規則を緩和しています。これは、グローバル・アドレス・スペースの一部を、インターネットへの接続を必要としないネットワークでの使用のために予約しています。本来これらのネットワークは、単一の組織によって管理されます。アドレスの以下の 3 つの範囲は、この目的のために予約されています。

- 10.0.0.0: 単一のクラス A ネットワーク
- 172.16.0.0 ～ 172.31.0.0: 16 の連続するクラス B ネットワーク
- 192.168.0.0 ～ 192.168.255.0: 256 の連続するクラス C ネットワーク

どの組織であってもこの範囲内のどのアドレスをも使用できます。ただし、これらのアドレスは全世界的に固有なものではないので、外部ルーターには定義されません。専用アドレスを使用しないネットワーク内のルーター、特にインターネット・サービス・プロバイダーによって操作されるものは、これらのアドレスに関連したすべてのルーティング情報を暗に破棄することが求められます。専用アドレスを使用する組織内のルーターは、すべての参照を内部リンクへの専用アドレスまでに制限することが求められます。これら

は、専用アドレスへの経路を外部に公示することも、外部ルーターへの専用アドレスを含む IP データグラムを転送することはありません。

専用 IP アドレスしか持たないホストは、インターネットへの直接 IP 層接続を持ちません。外部インターネット・ホストへの接続は、すべてアプリケーション・ゲートウェイ (709 ページの 21.3.4 『アプリケーション・レベル・ゲートウェイ (プロキシ)』を参照)、SOCKS (769 ページの 21.6 『SOCKS』を参照)、またはネットワーク・アドレス変換 (NAT) (720 ページの 21.4 『ネットワーク・アドレス変換 (NAT)』を参照) によって提供されなければなりません。

3.1.7 クラスレス・ドメイン間ルーティング (CIDR)

標準の IP ルーティングは、クラス A、B、および C ネットワーク・アドレスのみを認識します。これらの各ネットワーク内では、サブネット化を使用してより優れた細分度を提供することはできますが、複数のクラス C ネットワークが関係していることを示す方法は存在しません。この結果は、ルーティング・テーブル爆発問題と呼ばれ、3000 個のホストのクラス B ネットワークが各バックボーン・ルーターで 1 つのルーティング・テーブル・エントリーを必要とします。一続きのクラス C ネットワークとしてアドレッシングされている場合は、同じ環境が 16 のエントリーを必要とします。

この問題に対するソリューションは、クラスレス・ドメイン間ルーティング (CIDR) と呼ばれます。CIDR は、RFC 1518 ~ 1520 で説明されています。CIDR は、ネットワーク番号のクラスに従ってはルーティングしません (そのために、クラスレスと呼ばれます)。これは、IP アドレスの上位ビットにのみ基づきます。これらのビットは、IP 接頭部と呼ばれます。

各 CIDR ルーティング・テーブル・エントリーには、32 ビットの IP アドレスと 32 ビットのネットワーク・マスク (これらは 2 つで、IP 接頭部の長さとして示します) が含まれます。これは、タプル <IP_address network_mask> として表記されます。たとえば、8 つのクラス C アドレスのブロックを 1 つの単一のルーティング・テーブル・エントリーでアドレッシングするには、<192.32.136.0 255.255.248.0> という表記で十分です。バックボーンの観点から見ると、これは 192.32.136.0 ~ 192.32.143.0 のクラス C ネットワーク範囲を 1 つの単一ネットワークとして参照しています。これについては、図 25 で図示しています。

```

11000000 00100000 10001000 00000000 = 192.32.136.0 (class C
address)
11111111 11111111 11111--- ----- 255.255.248.0 (network mask)
===== logical_AND
11000000 00100000 10001--- ----- = 192.32.136 (IP prefix)

11000000 00100000 10001111 00000000 = 192.32.143.0 (class C
address)
11111111 11111111 11111--- ----- 255.255.248.0 (network mask)
===== logical_AND
11000000 00100000 10001--- ----- = 192.32.136 (same IP prefix)

```

図 25 クラスレス・ドメイン間 IP スーパーネット化の例

複数のネットワークを単一のエントリー内に結合するプロセスは、スーパーネット化と呼ばれます。ルーティングは、IP アドレスの自然なネットワーク・マスクより短いネットワーク・マスクに基づきます。これは、サブネット・マスクが自然なネットワーク・マスクよりも長いサブネット化（71 ページの 3.1.2 『IP サブネット』を参照）と対照を成します。

現在の IP アドレス割り振りポリシーと、これらのポリシーの基となった前提事項は、RFC 1518 – An Architecture for IP Address Allocation with CIDR で説明されています。これらは以下のように要約できます。

- IP アドレス割り当ては、組織のトポロジではなく、ネットワークの物理トポロジを反映します。組織と管理の境界がネットワークのトポロジと一致しない場合、それらは IP アドレスの割り当てに使用すべきではありません。
- 通常、ネットワーク・トポロジは、大陸と国の境界に厳密に従います。したがって、IP アドレスはこの基礎に基づいて割り当てる必要があります。
- ルーティング・ドメイン間で大量のトラフィックを伝送する比較的小さなネットワークのセットが存在するようになります。このネットワークは、国境を超える非階層的な方法で相互接続されます。このネットワークは、*伝送ルーティング・ドメイン (TRD)* と呼ばれます。各 TRD は固有な IP 接頭部を持ちます。適切な階層が存在しない場合、TRD は階層的な方法では編成されません。しかし、TRD が完全に大陸境界内にある場合、その IP 接頭部は大陸の IP 接頭部の拡張でなければなりません。
- 他の組織との間にその 2 つ組織専用の接続を持つ、多数の組織が存在するようになります。この接続は、他のドメインに宛てられたトラフィック

(通過トラフィック) は伝送しません。このような専用接続は、ルーティング・トポロジーに重大な影響を及ぼさないので、無視できます。

- ルーティング・ドメインの大多数は、シングルホームです。つまり、これらは単一の TRD に接続されます。これらには、TRD の IP 接頭部で始まるアドレスが割り当てられる必要があります。したがって、TRD に接続されたすべてのシングルホーム・ドメインのアドレスはすべて、その TRD の外にあるすべてのドメインの単一のルーティング・テーブル・エントリ内に集約することができます。

注：

これは、組織がそのインターネット・サービス・プロバイダーを変更する場合、その IP アドレスをすべて変更しなければならないことを暗示しています。これは通例の慣行ではありませんが、CIDR のインプリメンテーションの広がりによって、より一般的になりつつあります。

- マルチホーム・ドメインに使用できるいくつかのアドレス割り当て方式があります。これには、以下が含まれます。
 - ドメインに対する単一の IP 接頭部の使用。外部ルーターは、部分的または完全に通常の階層の外にある組織のエントリを持っていない必要ありません。ドメインがマルチホームでも、接続されているすべての TRD 自体が位相的に近隣にある場合は、接続されている TRD すべてに共通なビットをドメインの IP 接頭部に含めるのが妥当です。たとえば、すべての TRD が完全に米国内にあるとすれば、北米のドメインのみを暗黙指定する IP 接頭部が適切です。
 - ドメイン内のホストが最も適切な TRD の IP 接頭部を含んだ IP アドレスを持つような、接続された各 TRD ごとに 1 つの IP 接頭部の使用。組織はルーティング・ドメインのセットのように見えます。
 - 接続された TRD の 1 つからの IP 接頭部の割り当て。この TRD は、ドメインのデフォルト TRD になりますが、他のドメインは、明示的に代替の TRD の 1 つによってルーティングすることができます。
 - TRD 接続を持つマルチホーム・ドメインのセットを指す IP 接頭部の使用。たとえば、ネットワーク A に接続されたシングルホーム・ドメインを指す IP 接頭部、ネットワーク B に接続されたシングルホーム・ドメインを指す IP 接頭部、およびネットワーク A と B に接続されたデュアルホーム・ドメインを指す IP 接頭部があります。

これらそれぞれは、さまざまな利点、欠点および副次作用を持ちます。たとえば、最初のアプローチでは、2 番目のアプローチより送信ホストに近い宛先ドメインにインバウンド・トラフィックが入る結果になる傾向があ

ります。したがって、ネットワーク・コストの大部分は、受信側の組織が負います。

マルチホーム・ドメインは、性質の点でそれぞれ大きく異なります。そのため、上記のいかなる方式も、すべてのドメインに適するものではありません。最善な単一のポリシーは存在しません。RFC 1518 は、これらの選択のための規則は指定していません。

3.1.7.1 CIDR インプリメンテーション

インターネットにおける CIDR のインプリメンテーションは、基本的に Border Gateway Protocol バージョン 4 (189 ページの 4.9 『Border Gateway Protocol (BGP)』を参照) に基づきます。RFC 1520 – Exchanging Routing Information Across Provider Boundaries in the CIDR Environment で説明されているインプリメンテーション戦略には、バックボーン・ルーターで始まるルーティング階層を通して段階分けされたプロセスが含まれます。ネットワーク・サービス・プロバイダーは、以下の 4 つのタイプに分割されています。

- タイプ 1: デフォルトのドメイン間ルーティングを採用できないプロバイダー。
- タイプ 2: デフォルトのドメイン間ルーティングを使用するが、割り当てられている IP ネットワーク番号のかかなりの割合について明示的な経路を必要とするプロバイダー。
- タイプ 3: デフォルトのドメイン間ルーティングを使用し、それに少数の明示経路を補足するプロバイダー。
- タイプ 4: デフォルトの経路のみを使用してドメイン間ルーティングを実行するプロバイダー。

CIDR インプリメンテーションはタイプ 1 のネットワーク・プロバイダーで始まり、後からタイプ 2、そして最終的にはタイプ 3 のプロバイダーが加わりました。

3.1.8 IP データグラム

IP ネットワークにおける伝送単位は、IP データグラムと呼ばれます。これには、IP ヘッダーと上位レベル・プロトコルに関連したデータが含まれます。詳細については、図 26 を参照してください。

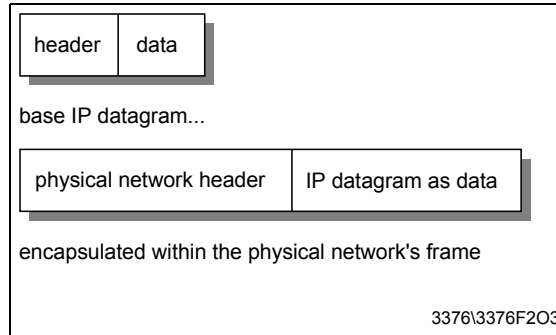


図 26 IP - 基本 IP データグラムのフォーマット

IP はデータグラムのフラグメント化と再組み立てを提供することができます。IP データグラムの最大長は、65,535 オクテットです。すべての IP ホストは、フラグメントなしの 576 オクテットのデータグラムをサポートしている必要があります。

データグラムのフラグメントはそれぞれヘッダーを持っています。ヘッダーは、元のデータグラムからコピーされます。フラグメントは、その宛先の移送されているとき、通常の IP データグラムとして扱われます。ただし、フラグメントの 1 つが破損すると、データグラム全体が破損したと見なされます。IP は認識メカニズムを備えていないので、残りのフラグメントは宛先ホストによって廃棄されます。

3.1.8.1 IP データグラム・フォーマット

IP データグラム・ヘッダーの最小長は 20 オクテットです。

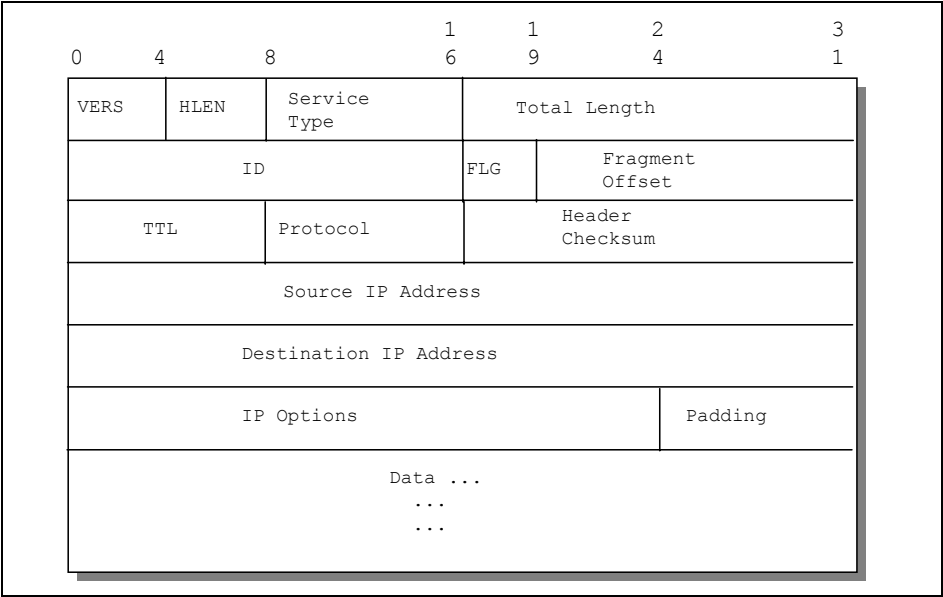


図 27 IP - IP データグラム・ヘッダーのフォーマット

- **VERS:** IP プロトコルのバージョンが含まれたフィールド。現行バージョンは 4 です。5 は試験的なバージョンです。6 は IPv6 (581 ページの 17.2 『IPv6 ヘッダー・フォーマット』を参照) 用のバージョンです。
- **HLEN:** 32 ビット単位でカウントされた IP ヘッダーの長さ。これにはデータ・フィールドは含まれません。
- **サービス・タイプ:** サービス・タイプは、この IP データグラムに要求されているサービスの品質を示すものです。このフィールドには以下の情報が含まれます。

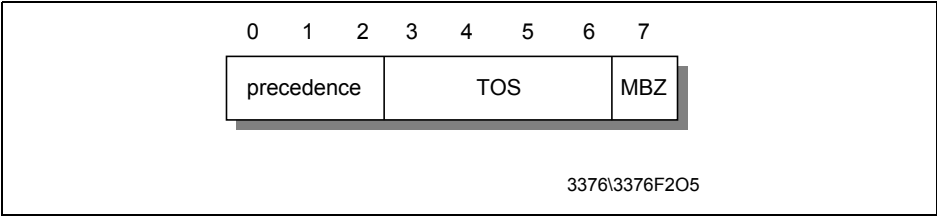


図 28 IP - サービス・タイプ

- 優先順位：このフィールドはデータグラムの性質と優先順位を指定します。

- 000: ルーチン
- 001: 優先
- 010: 緊急
- 011: 一時的
- 100: 一時的なオーバーライド
- 101: 重大
- 110: インターネットワーク制御
- 111: ネットワーク制御

- TOS: サービスのタイプの値を指定します。

- 1000: 遅延の最小化
- 0100: スループットの最大化
- 0010: 信頼性の最大化
- 0001: 金銭コストの最小化
- 0000: 通常のサービス

サービスのタイプの詳細な説明は、RFC 1349 (811 ページの 22.1 『QoS が必要な理由』を参照) にあります。

- MBZ: 将来の利用のために予約済み。

- 全長：データグラム、ヘッダー、およびデータの長さの合計。
- 識別：フラグメント化されたデータグラムの再組み立てを助けるために、送信側によって割り当てられた固有の番号。データグラムの各フラグメントは、同じ識別番号を持ちます。
- フラグ：このフィールドには制御フラグが含まれます。

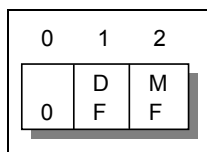


図 29 IP - フラグ

- 0: 予約済み。ゼロでなければなりません。
- DF (Do not Fragment): 0 はフラグメント化を許可することを意味し、1 はフラグメント化を許可しないことを意味します。

- MF (More Fragments): 0 はこれがデータグラムの最後のフラグメントであることを意味し、1 は追加のフラグメントが後に続くことを意味します。
- フラグメント・オフセット: これは、全データグラムの再組み立てを助けるために使用されます。このフィールドの値には、前のフラグメントに含まれている 64 ビット・セグメントの数 (ヘッダーのバイトはカウントされません) が含まれます。これが最初 (または唯一) のフラグメントである場合、このフィールドにはゼロが含まれます。
- 存続時間 (TTL): このフィールドは、データグラムが行き来できる時間 (秒単位) を指定します。理論上、このデータグラムを処理する各ルーターは、その処理時間をこのフィールドから減算することになっています。実際、ルーターは 1 秒にも満たない秒数でデータグラムを処理します。そのため、ルーターはこのフィールドの値から 1 を減算します。TTL は、時刻メトリックではなくホップ・カウント・メトリックになります。値が 0 に到達すると、このデータグラムは閉じたループ内を行き来して、破棄されたと見なされます。初期値はデータグラムを作成する上位レベル・プロトコルによって設定されます。
- プロトコル番号: このフィールドは、IP がこのデータグラム内のデータを送達すべき上位レベル・プロトコルを示します。これには、以下が含まれます。
 - 0: 予約済み
 - 1: Internet Control Message Protocol (ICMP)
 - 2: Internet Group Management Protocol (IGMP)
 - 3: Gateway-to-Gateway Protocol (GGP)
 - 4: IP (IP encapsulation)
 - 5: Stream
 - 6: Transmission Control Protocol (TCP)
 - 8: Exterior Gateway Protocol (EGP)
 - 9: Private Interior Routing Protocol
 - 17: User Datagram Protocol (UDP)
 - 41: IP Version 6 (IPv6)
 - 50: Encap Security Payload for IPv6 (ESP)
 - 51: Authentication Header for IPv6 (AH)
 - 89: Open Shortest Path First

完全なリストは、STD 2 – Assigned Internet Numbers に詳細に記されています。

- ヘッダー・チェックサム: このフィールドはヘッダー内に含まれた情報のチェックサムです。ヘッダー・チェックサムが内容と一致しない場合、そのデータグラムは破棄されます。

- 送信元 IP アドレス：このデータグラムを送信したホストの 32 ビット IP アドレス。
- 宛先 IP アドレス：このデータグラムの宛先ホストの 32 ビット IP アドレス。
- オプション：IP インプリメンテーションは、データグラム内にオプションを生成できなくても構いません。しかし、すべての IP インプリメンテーションは、オプションを含んだデータグラムを処理できなければなりません。オプション・フィールドは可変長です (0 個以上のオプションを含めることができます)。2 つのオプション・フォーマットが存在します。それぞれのフォーマットは、最初のおクテットにあるオプション番号の値によって決まります。
 - タイプ・オクテットのみ

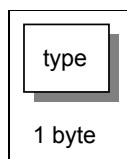


図 30 IP - タイプ・バイト

- タイプ・オクテット、長さオクテット、および 1 つ以上のオプション・データ・オクテット

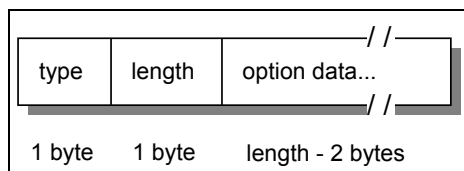


図 31 IP - タイプ・バイト、長さバイト、および 1 つ以上のオプション・データ・バイト

タイプ・バイトは、どちらの場合にも同じ以下のような構造を持ちます。

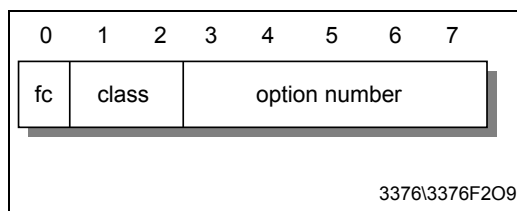


図 32 IP - タイプ・バイトの構造

- fc (Flag copy): このフィールドは、データグラムがフラグメント化されるときにオプション・フィールドがコピーされるか (1)、されないか (0) を示します。
- クラス: オプション・クラスは、以下の 2 ビット符号なし整数です。
 - 0: 制御
 - 1: 予約済み
 - 2: デバッグと測定
 - 3: 予約済み
- オプション番号: オプション番号は以下の 5 ビット符号なし整数です。
 - 0: オプション・リストの終了。このクラスは 0 で、fc ビットがゼロに設定され、長さバイトやデータは持ちません。つまり、オプション・リストは X'00' バイトで終わっています。これは、IP ヘッダー長 (これは 4 バイトの倍数です) が実際のオプションの長さとも一致しない場合に必要です。
 - 1: ノーオペレーション。このクラスは 0 で、fc ビットは設定されておらず、長さバイトやデータは存在しません。つまり、X'01' バイトが NOP です。これはデータグラム内のフィールドの位置合わせに使用できます。
 - 2: セキュリティー。このクラスは 0 で、fc ビットが設定され、値 11 を持つ長さバイトと 8 バイトのデータが存在します。これは、米国国防総省の要件で必要とされているセキュリティー情報に使用できます。
 - 3: 非厳密ソース・ルーティング。このクラスは 0 で、fc ビットが設定され、可変長データ・フィールドが存在します。このオプションについては、以下でより詳しく説明します。
 - 4: インターネット・タイム・スタンプ。このクラスは 2 で、fc ビットは設定されておらず、可変長データ・フィールドが存在します。全長は、40 バイトにまでできます。このオプションについては、以下でより詳しく説明します。
 - 7: 経路記録。このクラスは 0 で、fc ビットは設定されておらず、可変長データ・フィールドが存在します。このオプションについては、以下でより詳しく説明します。
 - 8: ストリーム ID。このクラスは 0 で、fc ビットが設定され、値 4 を持つ長さバイトと 1 データ・バイトが存在します。これは SATNET システムで使用されます。

- ・ 9: 厳密ソース・ルーティング。このクラスは 0 で、fc ビットが設定され、可変長データ・フィールドが存在します。このオプションについては、以下でより詳しく説明します。
- 長さ: このフィールドは、タイプおよび長さフィールドを含めたオプションの長さ (オクテット単位) をカウントします。
- オプション・データ: 特定のオプションに関連したデータが含まれます。
- ・ 埋め込み: オプションが使用されている場合、データグラムはすべてゼロのオクテットで次の 32 ビットの境界まで埋め込まれます。
- ・ データ: データグラムの含まれるデータ。これは、プロトコル・フィールドで指定されている上位レベル・プロトコルに渡されます。

3.1.8.2 フラグメント化

IP データグラムは、あるホストから別のホストへ移るとき、異なる物理ネットワークを通過することがあります。物理ネットワークはそれぞれ最小フレーム・サイズを持っています。これは **最大伝送単位 (MTU)** と呼ばれます。これは 1 つの物理フレームに入れることのできるデータグラムの長さを制限するものです。

IP は MTU を超えるデータグラムをフラグメント化するためのプロセスをインプリメントしています。このプロセスでは、最大サイズ以内の一連のデータグラムが作成されます。受信ホストは元のデータグラムを再組み立てします。IP では、各リンクが 68 オクテットの最小 MTU をサポートしている必要があります。これは、最大 IP ヘッダー長 (60 オクテット) と、最終ではないフラグメント内のデータの最小可能長 (8 オクテット) の合計です。いずれかのネットワークがこれより小さい値を提供している場合は、ネットワーク・インターフェース層にフラグメント化と再組み立てをインプリメントする必要があります。これは、IP に対して透過的でなければなりません。IP インプリメンテーションは、フラグメント化されていない 576 バイトより大きいデータグラムを処理する必要はありません。実際には、大半のインプリメンテーションはこれより大きな値に順応します。

フラグメント化されていないデータグラムは、すべてゼロのフラグメント化情報フィールドを持ちます。つまり、追加フラグメントのフラグ・ビットがゼロで、フラグメント・オフセットがゼロです。データグラムをフラグメント化するために、以下のステップが実行されます。

- ・ DF フラグ・ビットが検査され、フラグメント化が許可されているかが確認されます。このビットが設定されている場合、データグラムは破棄され、発信元に ICMP エラーが戻ります。

- MTU 値に基づいて、データ・フィールドが複数の部分に分割されます。新しく作成されたデータ部分は、最終データ部分以外はすべて、長さが 8 オクテットの倍数でなければなりません。
- 各データ部分が IP データグラム内に置かれます。これらのデータグラムのヘッダーは、元のヘッダーに以下の小さな変更が加えられたものになります。
 - 最終以外のすべてのフラグメント内に追加フラグメント・フラグ・ビットが設定されている。
 - それぞれのフラグメント・オフセット・フィールドが、フラグメント化されていない元のデータグラムの開始に対する相対位置として、このデータが元のデータグラム内で占めていた位置に設定されている。オフセットは、8 オクテット単位で測定されます。
 - 元のデータグラムにオプションが含まれていた場合は、オプション・タイプ・バイトの上位ビットによって、この情報がすべてのフラグメント・データグラムにコピーされるか、それとも最初のデータグラムだけにされるかが決まります。たとえば、ソース経路オプションはすべてのフラグメントにコピーされます。
 - 新しいデータグラムのヘッダー長フィールドが設定されている。
 - 新しいデータグラムの全長フィールドが設定されている。
 - ヘッダー・チェックサム・フィールドが再計算されている。
- ここで、フラグメント化されたこれらの各データグラムは、通常の IP データグラムとして転送されます。IP は、各フラグメントを独立して取り扱います。フラグメントは、さまざまなルーターを経由して目標の宛先に行くことができます。これらは、より小さな MTU を指定しているネットワークを通過する場合、さらにフラグメント化される可能性があります。

宛先ホストで、データが元のデータグラムへと再組み立てされます。送信ホストによって設定された識別フィールドが、データグラム内の送信元および宛先 IP アドレスと共に使用されます。このフィールドは、フラグメント化によって変更されません。

最初のフラグメントが到着すると、受信ホストはフラグメントを再組み立てするためにストレージ・バッファを割り振ります。またホストはタイマーを開始します。データグラムの後のフラグメントが到着すると、データがバッファ・ストレージ内へ、フラグメント・オフセット・フィールドで示された位置にコピーされます。すべてのフラグメントが到着すると、フラグメント化されていない元の完全なデータグラムが復元されます。フラグメント化されていないデータグラムに対するのと同様の処理が続行します。

タイマーが過ぎ、フラグメントが未解決のまま残ると、データグラムは破棄されます。このタイマーの初期値は、IP データグラム存続時間 (TTL) 値と呼ばれます。これは、インプリメンテーションに依存します。一部のインプリメンテーションでは、この構成が可能です。一部の IP ホストでは、`netstat` コマンドを使用してフラグメント化の詳細をリストすることができます。この例は、TCP/IP for OS/2 の `netstat -i` コマンドです。

3.1.8.3 IP データグラム・ルーティング・オプション

IP データグラム・オプション・フィールドは、IP データグラムの発信元が明示的にルーティング情報を指定するための 2 つの方式を示します。これはまた、IP データグラムがその移動経路を決定するための方式も示します。

非厳密ソース・ルーティング

Loose Source and Record Route (LSRR) オプションとも呼ばれる非厳密ソース・ルーティング・オプションは、IP データグラムの送信元が明示的なルーティング情報を供給するための手段を提供します。この情報は、宛先へのデータグラムの転送時にルーターによって使用されます。またこれは、経路の記録にも使用されます。

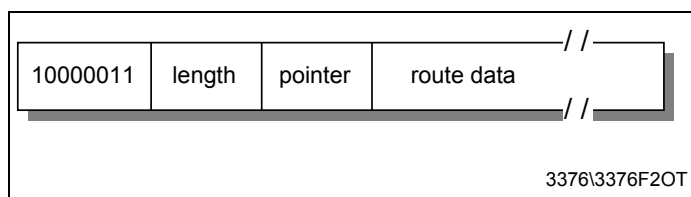


図 33 IP - 非厳密ソース・ルーティング・オプション

このヘッダーには、以下のフィールドが含まれます。

- 10000011(10 進 131): これは、非厳密発信元ルーティングを示すオプション・タイプ・オクテットの値です。
- 長さ: タイプおよび長さフィールドを含めた、このオプション・フィールドの長さが含まれます。
- ポインター: 処理される次の IP アドレスにあるオプション・データを指します。これはオプションの開始に比例してカウントされるため、この最小値は 4 です。ポインターがオプションの長さより大きい場合は、ソース経路の終了に達しており、それ以後のルーティングは (このオプションのないデータグラムの場合と同様に) 宛先 IP アドレスに基づくこととなります。
- 経路データ: このフィールドには、一連の 32 ビット IP アドレスが含まれます。

データグラムがその宛先に到着し、ソース経路が空でない (pointer < length の) とき、受信ホストは以下を行います。

- 経路データ・フィールド内の次の IP アドレス (ポインター・フィールドで示されたもの) を取り、それをデータグラムの宛先 IP アドレス・フィールド内に挿入します。
- ローカル IP アドレスを、送信元リスト内の、ポインター・フィールドで指示された位置に挿入します。この IP アドレスは、データグラムが転送されるネットワークと一致するローカル IP アドレスです。(ルーターは複数の物理ネットワークに接続されているため、複数の IP アドレスを持ちます。)
- ポインターを 4 ずつ増分します。
- データグラムを新しい宛先 IP アドレスに伝送します。

この手順では、必ず、戻り経路が経路データ内に (逆順で) 記録され、最終受信側はこのデータを使用して非厳密ソース経路を逆向きで構成します。転送ルーターが経路内の次のアドレスへ到達するために任意の経路と任意の数の中間ルーターを使用できるので、これは *非厳密* ソース経路です。

注：

発信ホストは最初の中間ルーターの IP アドレスを宛先アドレス・フィールドに入れ、ターゲット宛先を含めたパス内の残りのルーターの IP アドレスはソース経路オプションに入れられます。宛先に到着するときこのデータグラム内に記録されている経路には、データグラムを転送した各ルーターの IP アドレスが含まれます。各ルーターはソース経路内の一箇所の転送を行います。また、ルーターはアウトバウンド・インターフェースの IP アドレスを記録しますが、ソース経路には元々インバウンド・インターフェースが含まれているため、普通は異なる IP アドレスが使用されます。

厳密ソース・ルーティング

Strict Source and Record Route (SSRR) オプションとも呼ばれる厳密ソース・ルーティング・オプションは、非厳密ソース・ルーティング・オプションと同じ原理を使用します。ただし、中間ルーターが、直接接続されたネットワークを介して、データグラムを発信元経路内の次の IP アドレスに送信しなければならないという点は除きます。これは中間ルーターを使用できません。これが実行できないと、ICMP 宛先到達不能エラー・メッセージが発行されます。図 34 は、SSRR オプションの概要を示しています。

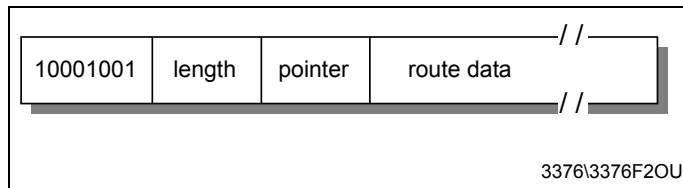


図 34 IP - 厳密ソース・ルーティング・オプション

- 1001001 (10 進 137): 厳密ソース・ルーティングを示すオプション・タイプ・バイトの値。
- 長さ：これについては、非厳密ソース・ルーティングのセクションで説明しています。
- ポインター：これについては、非厳密ソース・ルーティングのセクションで説明しています。
- 経路データ：一連の 32 ビット IP アドレス。

経路記録

このオプションは、IP データグラムがトラバースした経路を記録するための手段を提供します。これは、ソース・ルーティング・オプションと似た働きをします。ただし、このオプションでは、空のルーティング・データ・フィールドはできません。このフィールドは、データグラムがネットワークを通過するときに埋められます。このルーティング情報に対して十分なスペースが送信元ホストによって供給される必要があります。データグラムがその宛先に到達する前にデータ・フィールドが埋め込まれた場合、そのデータグラムはその後の経路の記録は行われずに転送されます。図 35 は、経路記録オプションの概要を示しています。

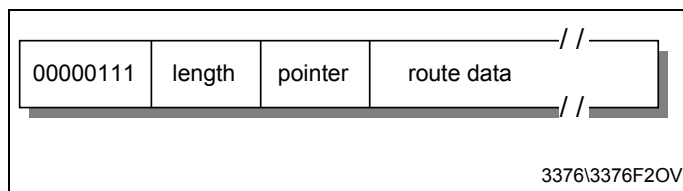


図 35 IP - 経路記録オプション

- 0000111 (10 進 7): 経路記録を示すオプション・タイプ・バイトの値。
- 長さ：これについては、非厳密ソース・ルーティングのセクションで説明しています。

- ・ ポインター：これについては、非厳密ソース・ルーティングのセクションで説明しています。
- ・ 経路データ：一連の 32 ビット IP アドレス。

3.1.8.4 インターネット・タイム・スタンプ

タイム・スタンプは、宛先への経路を通るいくつか（またはすべて）のルーターに、オプション・データへタイム・スタンプを入れることを強制するオプションです。タイム・スタンプは秒単位で測定され、デバッグの目的に使用することができます。これらは、以下の 2 つの理由で、パフォーマンスの測定には使用できません。

- ・ 大半の IP データグラムは 1 秒足らずで転送されるため、タイム・スタンプは正確ではありません。
- ・ IP ルーターには同期化されたクロックを持つ必要が課せられていないため、正確でない可能性があります。

図 36 は、インターネット・タイム・スタンプ・オプションの概要を示しています。

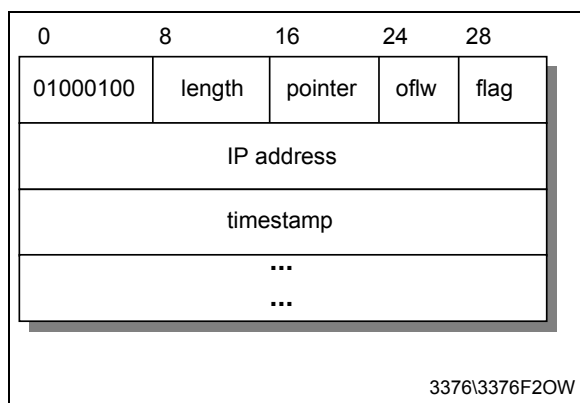


図 36 IP - インターネット・タイム・スタンプ・オプション

- ・ 01000100 (10 進 68): これは、インターネット・タイム・スタンプ・オプションを示すオプション・タイプの値です。
- ・ 長さ：タイプおよび長さフィールドを含めた、このオプションの全長が含まれます。
- ・ ポインター：処理される次のタイプ・スタンプ（最初の空きタイム・スタンプ）を指します。

- oflw (オーバーフロー): このフィールドには、データ・フィールド内のスペースの不足のためにタイム・スタンプを刻印できない装置の数が含まれます。
- フラグ: これは、以下のようなタイム・スタンプの刻印方法を示す 4 ビット値です。
 - 0: タイム・スタンプのみ。連続する 32 ビット・ワードで保管される。
 - 1: 各タイム・スタンプの前に刻印装置の IP アドレスが付く。
 - 2: IP アドレス・フィールドが事前指定され、IP 装置はリスト内にその独自のアドレスを見つけたときにだけ刻印を行う。
- タイム・スタンプ: 真夜中 UT (GMT) よりミリ秒単位で記録された 32 ビット・タイム・スタンプ。

発信ホストは、すべてのタイム・スタンプを保持する十分なデータ域を指定してこのオプションを構成する必要があります。タイム・スタンプ域が満杯になると、それ以上のタイム・スタンプは追加されません。

3.2 Internet Control Message Protocol (ICMP)

ICMP は、STD 番号 5 の標準プロトコルです。この標準には、IP (67 ページの 3.1 『インターネット・プロトコル (IP)』を参照) と IGMP (241 ページの 6.2 『Internet Group Management Protocol (IGMP)』を参照) も組み込まれています。この状態は「必須 (Required)」です。これは RFC 792 で説明され、その更新が RFC 950 にあります。

パス MTU ディスカバリーは、状態が「任意選択 (Elective)」であるドラフト標準プロトコルです。これは、RFC 1191 で説明されています。

ICMP ルーター・ディスカバリーは、提案標準のプロトコルで、状態は「任意選択 (Elective)」です。これは、RFC 1256 で説明されています。

ルーターまたは宛先ホストは、データグラム処理でのエラーについて送信元ホストに通知する必要がある場合、Internet Control Message Protocol (ICMP) を使用します。ICMP には以下の特徴があります。

- ICMP は、ICMP が上位レベル・プロトコルであるかのように IP を使用します (つまり、ICMP メッセージは IP データグラム内にカプセル化されます)。ただし、ICMP は IP の主要な部分であり、すべての IP モジュールによってインプリメントされている必要があります。
- ICMP は、IP を信頼できるものにするためではなく、エラーをレポートするために使用されます。その脱落に関するレポートがないままデータグラムが未送達になっていることもあります。信頼性は IP サービスを使用し

て上位レベル・プロトコルによってインプリメントされている必要があります。

- ICMP は、ICMP メッセージのエラーのレポートには使用できません。これによって無限の反復を避けられます。ICMP 応答は ICMP 照会メッセージ (ICMP タイプ 0、8、9、10、および 13 ~ 18) に対する応答として送信されます。
- フラグメント化されたデータグラムの場合は、最初のフラグメントのエラーに関する ICMP メッセージだけが送信されます。つまり、ICMP メッセージがゼロ以外のフラグメント・オフセット・フィールドを持つ IP データグラムについて言及していることはありません。
- ICMP メッセージは、ブロードキャストまたはマルチキャスト宛先アドレスを持つデータグラムに対する応答として送信されることはありません。
- ICMP メッセージは、固有なホストを示す送信元 IP アドレスを持たないデータグラムに対する応答として送信されることはありません。つまり、送信元アドレスは、ゼロ、ループバック・アドレス、ブロードキャスト・アドレス、マルチキャスト・アドレスのいずれになることもありません。
- RFC 792 には、IP データグラム処理エラーをレポートするために ICMP メッセージを生成することができると記述されています。しかし、これは必須ではありません。実際には、エラーに対する ICMP メッセージはほとんどいつもルーターが生成します。宛先ホストの場合、ICMP メッセージの生成はインプリメンテーションに依存します。

3.2.1 ICMP メッセージ

ICMP メッセージは、STD 5 に属する必須の RFC 792 および RFC 950 で説明されています。

ICMP メッセージは、IP データグラムに入れて送信されます。IP ヘッダーのプロトコル番号は 1 (ICMP) で、サービスのタイプはゼロ (ルーチン) です。IP データ・フィールドには、図 37 に表示されている ICMP メッセージが含まれます。

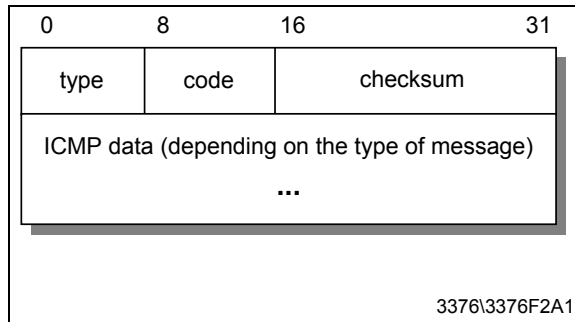


図 37 ICMP- メッセージ・フォーマット

メッセージには以下のコンポーネントが含まれます。

- タイプ：以下のようなメッセージのタイプを指定します。
 - 0: エコー応答
 - 3: 宛先到達不能
 - 4: ソース消失
 - 5: リダイレクト
 - 8: エコー
 - 9: ルーター公示
 - 10: ルーター送信請求
 - 11: 時間超過
 - 12: パラメーター問題
 - 13: タイム・スタンプ要求
 - 14: タイム・スタンプ応答
 - 15: 情報要求 (廃止)
 - 16: 情報応答 (廃止)
 - 17: アドレス・マスク要求
 - 18: アドレス・マスク応答
 - 30: 経路追跡
 - 31: データグラム変換エラー
 - 32: モバイル・ホスト・リダイレクト
 - 33: IPv6 Where-Are-You
 - 34: IPv6 I-Am-Here
 - 35: モバイル登録要求
 - 36: モバイル登録応答
 - 37: ドメイン・ネーム要求
 - 38: ドメイン・ネーム応答
 - 39: SKIP
 - 40: Photuris

- **コード**: この ICMP メッセージによってレポートされているデータグラムのエラー・コードが含まれます。この解釈は、メッセージのタイプに依存します。
- **チェックサム**: ICMP タイプ・フィールドで始まる ICMP メッセージのチェックサムが含まれます。チェックサムが内容と一致しない場合、そのデータグラムは破棄されます。
- **データ**: この ICMP メッセージの情報が含まれます。通常これには、ICMP メッセージが生成される理由となった、元の IP メッセージの部分が含まれます。

各 ICMP メッセージについて別個に説明します。

3.2.1.1 エコー (8) およびエコー応答 (0)

エコーは、ネットワーク上で他のホストがアクティブかどうかを調べるために使用されます。これは Ping コマンド (116 ページの 3.2.2.1 『Ping』を参照) によって使用されます。送信側は識別、シーケンス番号、およびデータ・フィールドを初期設定します。その後でデータグラムが宛先ホストに送信されます。受信側はタイプをエコー応答に変更してから、データグラムを受信側に戻します。詳細については、図 38 を参照してください。

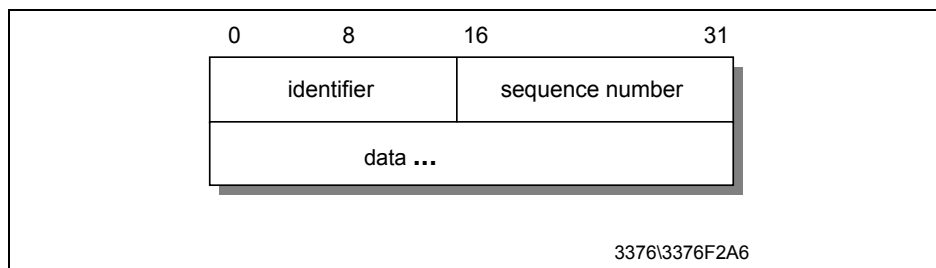


図 38 エコーおよびエコー応答

3.2.1.2 宛先到達不能 (3)

中間ルーターからこのメッセージを受け取る場合、それは、ルーターが宛先 IP アドレスを到達不能と見なしていることを意味します。

宛先ホストからこのメッセージを受け取る場合、それは、元のデータグラムのプロトコル番号フィールドで指定されているプロトコルがアクティブでないか、指定されているポートが非アクティブであることを意味します。(ポートに関する追加情報については、212 ページの 5.2 『User Datagram Protocol (UDP)』を参照してください。) 詳細については、図 39 を参照してください。

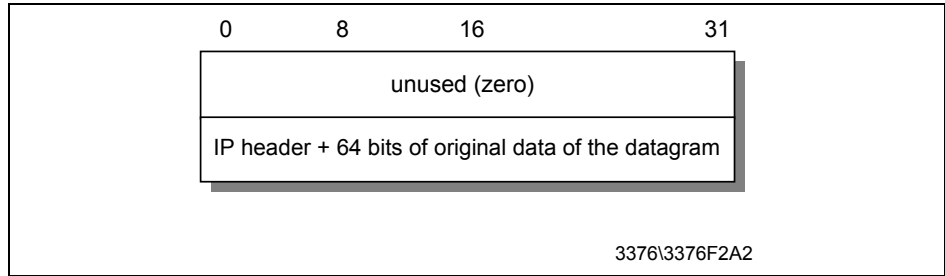


図 39 ICMP - 宛先到達不能

ICMP ヘッダー・コード・フィールドには、以下のいずれかの値が含まれます。

- 0: ネットワークに到達不能
- 1: ホストに到達不能
- 2: プロトコルに到達不能
- 3: ポートに到達不能
- 4: フラグメント化が必要だが Do Not Fragment ビットが設定された
- 5: ソース経路が故障した
- 6: 宛先ネットワークが不明
- 7: 宛先ホストが不明
- 8: 送信元ホストが分離されている (廃止)
- 9: 宛先ネットワークが管理上禁止されている
- 10: 宛先ホストが管理上禁止されている
- 11: このサービスのタイプではネットワークに到達不能
- 12: このサービスのタイプではホストに到達不能
- 13: フィルター操作によって通信が管理上禁止されている
- 14: ホスト優先順位違反
- 15: 事実上の優先順位カットオフ

ルーターがパス MTU ディスカバリー・プロトコルをインプリメントしている場合、宛先到達不能メッセージのフォーマットはコード 4 用に変更されます。これには、データグラムを受け入れなかったリンクの MTU が含まれます。詳細については、図 40 を参照してください。

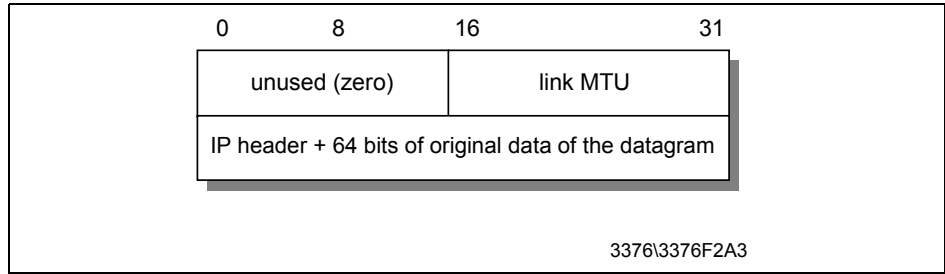


図 40 ICMP - リンクの MTU が含まれた「フラグメント化が必要」

3.2.1.3 ソース消失 (4)

中間ルーターからこのメッセージを受け取る場合、それは、ルーターが、データグラムをキューに入れるために必要なバッファー・スペースを持っていなかったことを意味します。

宛先ホストからこのメッセージを受け取る場合、それは、着信データグラムの到着が速すぎて処理できないことを意味します。

ICMP ヘッダー・コード・フィールドは常にゼロです。

詳細については、図 41 を参照してください。

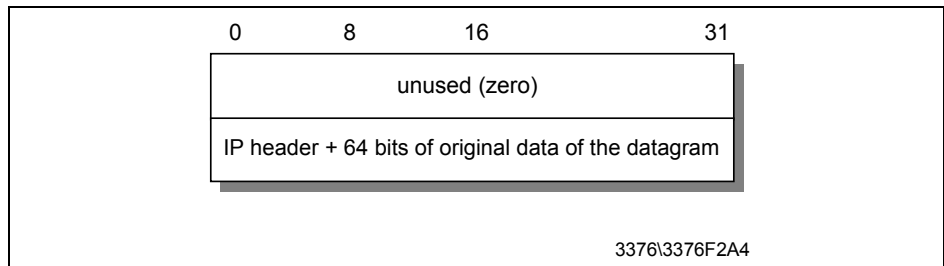


図 41 ICMP - ソース消失

3.2.1.4 リダイレクト (5)

中間ルーターからこのメッセージを受け取るならば、それは、ホストがネットワークに対する将来のデータグラムを、ICMP メッセージで指定されている IP アドレスを持つルーターに送信すべきことを意味します。この推奨ルーターは、データグラムを送信したホストおよび IP データグラムを戻したルーターと常に同じサブネット上になります。ルーターはそのネクスト・ホップ宛先にデータグラムを転送します。IP データグラムにソース経路が含まれている場合、このメッセージは送信されません。

ICMP ヘッダー・コード・フィールドには、以下のいずれかの値が含まれます。

- 0: ネットワーク・リダイレクト
- 1: ホスト・リダイレクト
- 2: このサービスのタイプの場合のネットワーク・リダイレクト
- 3: このサービスのタイプの場合のホスト・リダイレクト

詳細については、図 42 を参照してください。

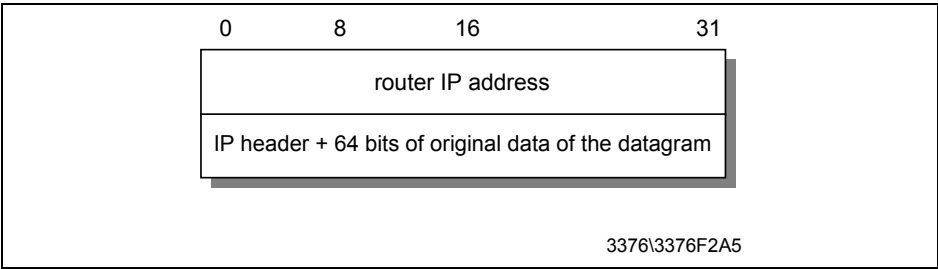


図 42 ICMP - リダイレクト

3.2.1.5 ルーター公示 (9) およびルーター送信請求 (10)

ICMP メッセージ 9 および 10 はオプションです。これらは、選択である RFC 1256 で説明されています。詳細については、図 43 および 図 44 を参照してください。

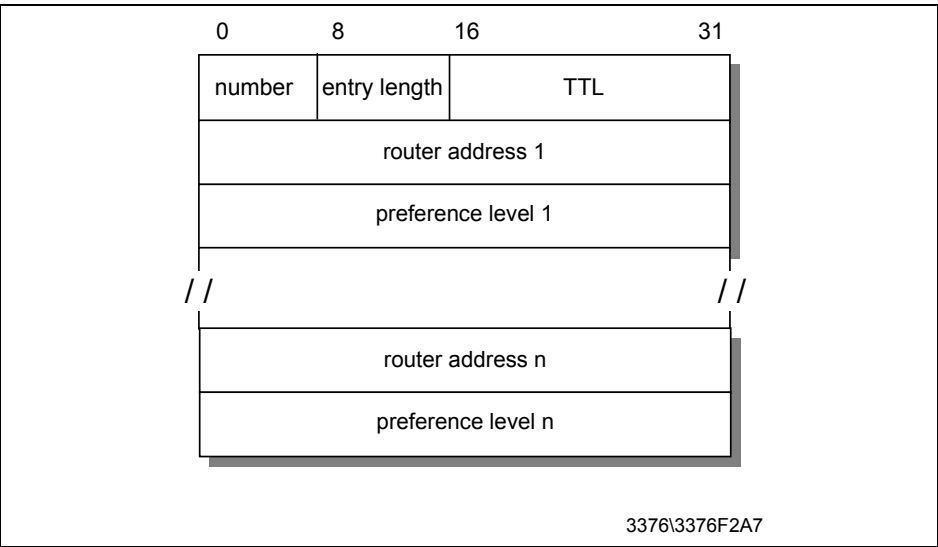


図 43 ICMP - ルーター公示

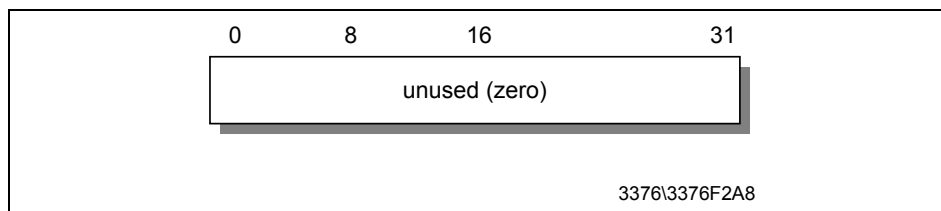


図 44 ICMP - ルーター送信請求

- 番号：メッセージ内のエントリーの番号。
- エントリー長：32 ビット単位のエントリーの長さ。これは 2 (IP アドレス用の 32 ビットとプリファレンス値用の 32 ビット) です。
- TTL: エントリーが有効と見なされる秒数。
- ルーター・アドレス：送信側の IP アドレスの 1 つ。
- プリファレンス・レベル：デフォルト・ルーターの選択時にこのアドレスに割り当てられたプリファレンスを示す、符号付き 32 ビットのレベル。サブネット上の各ルーターには、その独自のプリファレンス・レベルを公示する責務があります。より大きな値はより高いプリファレンスを暗示し、より小さな値はより低いプリファレンスを暗示します。デフォルトはゼロで、これは可能範囲の中間に位置します。値 X'80000000' (-231) は、そのルーターをデフォルトのルーターとして使用すべきでないことを示します。

これらのどちらのメッセージの場合にも、ICMP ヘッダー・コード・フィールドはゼロです。

これらの 2 つのメッセージは、ホストまたはルーターがルーター・ディスカバリー・プロトコルをサポートしている場合に使用されます。ルーターは、構成で指定されたサブネット上で定期的にその IP アドレスを公示します。公示は、すべてのシステムのマルチキャスト・アドレス (224.0.0.1) か、限定ブロードキャスト・アドレス (255.255.255.255) で行われます。デフォルト動作では、10 分ごとに公示が送信され、TTL 値は 1800 (30 分) です。ルーターは、自分が受け取る請求メッセージに対する応答も行います。ルーターは請求ホストに直接応答するか、短いランダムな間隔を置いてマルチキャストで応答することができます。

ホストは請求メッセージを送信できます。請求メッセージは、すべてのルーターのマルチキャスト・アドレス (224.0.0.2) か、限定ブロードキャスト・アドレス (255.255.255.255) に送信されます。通常は、3 つの請求メッセージが 3 秒おきに送信されます。ホストはこの代わりに定期的な公示を待つこともで

きます。ホストは、より高いプリファレンス値を持つ公示を受信するたびに、そのデフォルト・ルーターを変更します。さらにホストは、新しいエントリの TTL タイマーを、公示の中の値と一致するように変更します。ホストは、その現行のデフォルト・ルーターに対する新しい公示を受信すると、TTL 値を新しい公示の中の値にリセットします。

このプロセスは、ルーターが自分自身を使用不能として宣言するためのメカニズムにもなります。その場合ルーターは TTL 値がゼロの公示を送信します。

3.2.1.6 時間超過 (11)

中間ルーターからこのメッセージを受け取る場合、それは、IP データグラムの存続時間フィールドの有効期限が切れたことを意味します。

宛先ホストからこのメッセージを受け取る場合、それは、ホストがデータグラムのフラグメントを待っている間に、IP フラグメントの再組み立て存続時間タイマーの有効期限が切れたことを意味します。ICMP ヘッダー・コード・フィールドには、以下のいずれかの値が含まれる可能性があります。

- 0: 伝送 TTL 超過
- 1: 再組み立て TTL 超過

詳細については、図 45 を参照してください。

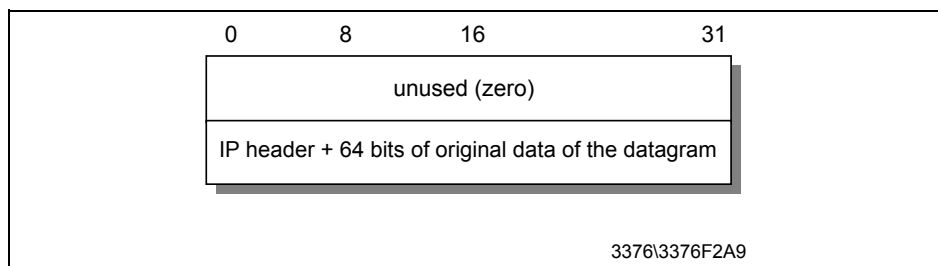


図 45 ICMP - 時間超過

3.2.1.7 パラメーター問題 (12)

このメッセージは、IP ヘッダー・パラメーターの処理中に問題が発生したことを示します。ポインター・フィールドは、元の IP データグラム内の、問題の発生したオクテットを示します。ICMP ヘッダー・コード・フィールドには、以下のいずれかの値が含まれる可能性があります。

- 0: 詳細不明エラー
- 1: 必須オプションが欠落している

詳細については、図 46 を参照してください。

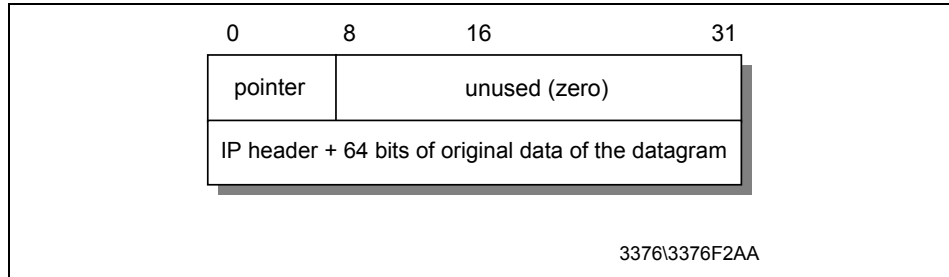


図 46 ICMP - パラメーター問題

3.2.1.8 タイム・スタンプ要求 (13) およびタイム・スタンプ応答 (14)

これらの 2 つのメッセージは、デバッグとパフォーマンス測定のためのものです。これらはクロック同期には使用されません。

送信側は識別のシーケンス番号 (これは複数のタイム・スタンプ要求を送信する場合に使用されます) を初期設定し、発信タイム・スタンプを設定して、データグラムを受信側に送信します。受信ホストは受信および伝送タイム・スタンプを記入し、タイプをタイム・スタンプ応答に変更して、それを元の送信側に戻します。受信時刻と伝送時刻の間に知覚できる時差がある場合、データグラムは 2 つのタイム・スタンプを持ちます。実際、大半のインプリメンテーションは、1 回の操作で 2 つ (受信と応答) を実行します。これは、2 つのタイム・スタンプを同じ値に設定します。タイム・スタンプは、真夜中 UT (GMT) より経過したミリ秒数です。

詳細については、図 47 を参照してください。

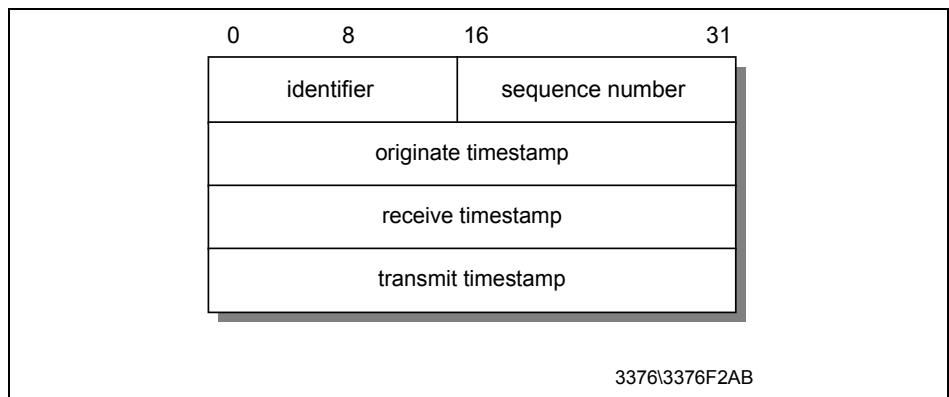


図 47 ICMP - タイム・スタンプ要求およびタイム・スタンプ応答

3.2.1.9 情報要求 (15) および情報応答 (16)

情報要求は、ホストが、接続されているネットワークの IP アドレスを取得するために発行します。送信側は IP ヘッダー内の宛先 IP アドレスをゼロ (このネットワークを意味する) に設定して要求を記入し、IP アドレスを他のホストに割り当てることを許可されているサーバーからの応答を待ちます。ICMP ヘッダー・コード・フィールドはゼロです。応答には、IP ヘッダーの発信元と宛先の両方のフィールド内に IP ネットワーク・アドレスが含まれます。このメカニズムは現在では廃止されています (124 ページの 3.5 『Reverse Address Resolution Protocol (RARP)』も参照してください)。

詳細については、図 48 を参照してください。

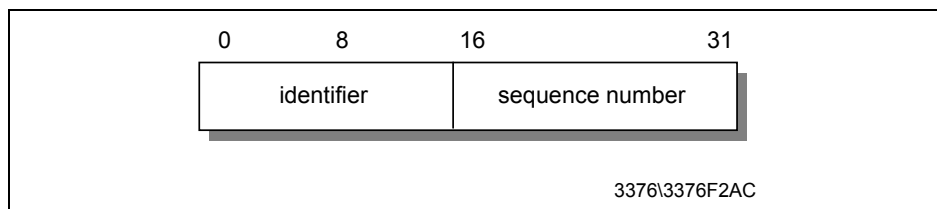


図 48 ICMP - 情報要求および情報応答

3.2.1.10 アドレス・マスク要求 (17) およびアドレス・マスク応答 (18)

アドレス・マスク要求は、ホストが、接続されているネットワーク上で使用されているサブネット・マスクを判別するために使用します。大半のホストは、そのサブネット・マスクと共に構成されています。しかし、ディスクレス・ワークステーションなどの一部のホストは、サーバーからこの情報を取得しなければなりません。ホストは RARP (124 ページの 3.5 『Reverse Address Resolution Protocol (RARP)』を参照) を使用してその IP アドレスを取得します。ホストは、サブネット・マスクを取得するために、アドレス・マスク要求をブロードキャストします。アドレス・マスク応答を送信するように構成されているネットワーク上の任意のホストは、サブネット・マスクを記入し、パケットをアドレス・マスク応答に変更して、それを送信側に戻します。ICMP ヘッダー・コード・フィールドはゼロです。

詳細については、図 49 を参照してください。

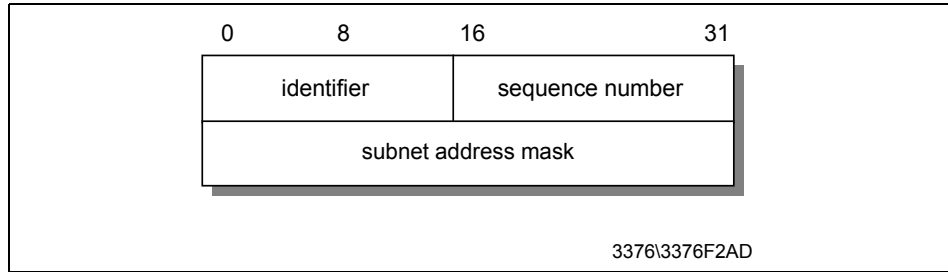


図 49 ICMP - アドレス・マスク要求および応答

3.2.2 ICMP アプリケーション

ICMP に基づく 2 つの単純で広く使用されているアプリケーションとして、Ping および Traceroute があります。Ping は ICMP のエコーおよびエコー応答メッセージを使用してホストが到達可能かどうかを判別します。Traceroute は、IP データグラムが宛先への途中で有効期限切れになるよう、低い TTL 値を付けてこれを送信します。これは、結果としての ICMP 時間超過メッセージを使用して、インターネット内のどこでデータグラムの有効期限が切れたかを判別し、ホストへの経路の全体像を把握します。これらのアプリケーションについては、続くいくつかのセクションで説明します。

3.2.2.1 Ping

Ping は最も単純な TCP/IP アプリケーションです。これは IP データグラムを指定された宛先ホストに送信し、応答を受け取るまでの往復伝送時間を測定します。名詞および動詞として使用される *Ping* という用語は、水中の物体を見つけるためのソナー操作から取られています。これは、*Packet InterNet Groper* の省略でもあります。

通常、正常にホストを PING できれば、Telnet や FTP などの他のアプリケーションはそのホストに到達できます。ただしこれは、アプリケーション・プロトコルまたはポート番号あるいはその両方によってネットワークへのアクセスを制御するインターネットのセキュリティー方策、特にファイアウォール (704 ページの 21.3 『ファイアウォール』を参照) の出現により、必ずしもそうではなくなっています。それでも、ホストの到達可能性の最初のテストはやはりこれに PING を試行することです。

種々の PING のインプリメンテーションで使用される構文は、プラットフォームによって異なります。以下の構文は、OS/2 インプリメンテーション用です。

```
ping [-switches] host [size [packets]]
```

- switches: さまざまな PING オプションを使用可能にするスイッチです。
- host: 宛先。シンボル名か IP アドレスのいずれか。
- size: パケットのデータ部分のサイズ
- packets: 送信するパケットの数

PING は ICMP エコーおよびエコー応答メッセージ (108 ページの 3.2.1.1 『エコー (8) およびエコー応答 (0)』を参照) を使用します。ICMP はすべての TCP/IP インプリメンテーションで必須なので、ホストは別個のサーバーで PING 要求に応答する必要はありません。

PING は、IP のインストールの検査に役立ちます。以下のさまざまなコマンドは、それぞれ IP インストールの異なる部分の操作を必要とします。

- ping loopback: 基本 TCP/IP ソフトウェアの操作を検査します。
- ping my-IP-address: 物理ネットワーク装置をアドレッシングできるかどうかを検査します。
- ping a-remote-IP-address: ネットワークにアクセスできるかどうかを検査します。
- ping a-remote-host-name: ネーム・サーバー (またはインストールによっては、ネーム・スペース・リゾルバー) の操作を検査します

PING はすべての IBM TCP/IP 製品にインプリメントされています。

3.2.2.2 Traceroute

Traceroute プログラムは、ネットワーク上で IP データグラムがたどる経路を判別するために使用されます。

Traceroute は ICMP および UDP に基づいています。これは IP データグラムを TTL 1 で宛先ホストに送信します。最初のルーターは TTL を 0 に減分し、データグラムを破棄して、送信元に ICMP 時間超過メッセージを戻します。こうして、パスの中の最初のルーターが識別されます。このプロセスは、順次により大きな TTL 値で繰り返され、宛先ホストへのパスの中の正確な一連のルーターを識別します。

Traceroute は UDP データグラムを宛先ホストに送信します。このデータグラムは、標準範囲外のポート番号を参照します。送信元は、ICMP ポート到達不能メッセージを受信したら、宛先ホストに到達していると判別します。

Traceroute はすべての IBM TCP/IP 製品にインプリメントされています。

3.3 Internet Group Management Protocol (IGMP)

IGMP は、STD 番号 5 の標準プロトコルです。この標準には、IP (67 ページの 3.1 『インターネット・プロトコル (IP)』を参照) と ICMP (105 ページの

3.2『Internet Control Message Protocol (ICMP)』を参照) も組み込まれています。その状態は「推奨 (Recommended)」です。これは、RFC 1112 で説明され、RFC 2236 にはその更新点があります。

ICMP と同様、Internet Group Management Protocol (IGMP) もまた IP の主要な部分です。これによって、ホストは IP マルチキャストに参加できます。さらに、IGMP によってルーターは、ローカル・サブネット上に特定のマルチキャストに関与するホストがないかを検査することができます。

IGMP の詳細な説明については、241 ページの 6.2『Internet Group Management Protocol (IGMP)』を参照してください。

3.4 Address Resolution Protocol (ARP)

ARP プロトコルは、ネットワークに固有な標準プロトコルです。アドレス解決プロトコルは、上位レベル・プロトコル・アドレス (IP アドレス) を物理ネットワーク・アドレスに変換することを担当します。これは、RFC 826 で説明されています。

3.4.1 ARP の概要

単一の物理ネットワークにおいて、ネットワーク上の個々のホストは、その物理ハードウェア・アドレスによって認識されます。上位レベル・プロトコルは、シンボリック・アドレス (この場合は IP アドレス) の形式で宛先ホストをアドレッシングします。このプロトコルが宛先 IP アドレス *w.x.y.z* にデータグラムを送信しようとしても、デバイス・ドライバはこのアドレスを認識しません。

そのために、IP アドレスを宛先ホストの物理アドレスに変換するモジュール (ARP) が備えられています。これは索引テーブル (しばしば *ARP キャッシュ* と呼ばれる) を使用してこの変換を実行します。

ARP キャッシュの中にアドレスが見つからないと、ネットワーク上で *ARP 要求* と呼ばれる特別な形式のブロードキャストが送信されます。ネットワーク上のマシンの 1 つが要求の中に自分の IP アドレスを認識すると、そのマシンは *ARP 応答* を要求ホストに送信して戻します。応答には、ホストの物理ハードウェア・アドレスとソース経路情報 (パケットがそのパス上でブリッジを横断した場合) が含まれます。このアドレスとソース経路情報は両方とも要求ホストの ARP キャッシュに保管されます。これ以後は、この宛先 IP アドレスへのその後のすべてのデータグラムを、デバイス・ドライバがネットワーク上でデータグラムを送信するために使用する物理アドレスに変換できます。

この規則の例外は非同期転送モード (ATM) テクノロジーになります。前述のとおりこの非同期転送モード (ATM) テクノロジーでは ARP を物理層にインプリメントできません。したがって、すべてのホストは、初期化の後、IP アドレスをハードウェア・アドレスに解決できるようになるために ARP サーバーに登録する必要があります (45 ページの 2.10 『非同期伝送モード (ATM)』も参照してください)。

ARP は、ハードウェア・ブロードキャストをサポートするネットワーク上で使用されるよう設計されています。つまり、たとえば X.25 ネットワーク上では ARP は機能しません。

3.4.2 ARP の詳細概念

ARP は、IEEE 802 ネットワークとそれより古い DIX イーサネット・ネットワーク上で、IP アドレスを物理ハードウェア・アドレスにマップするために使用されます (29 ページの 2.1 『イーサネットおよび IEEE 802.x ローカル・エリア・ネットワーク (LAN)』を参照)。そのため、これは、そのネットワーク用のデバイス・ドライバに密接に関連します。実際、RFC 826 の ARP 仕様は、その機能だけを説明し、そのインプリメンテーションについては説明していません。インプリメンテーションはかなりの程度ネットワーク・タイプ用のデバイス・ドライバに依存し、これらは通常アダプター・マイクロコードでまとめてコーディングされます。

3.4.2.1 ARP パケットの生成

アプリケーションが特定の IP 宛先アドレスにデータを送信しようとする、IP ルーティング・メカニズムはパケットの次のホップの IP アドレス (これは宛先ホストそのものかルーターになります) と、これが送信されるべきハードウェア装置を判別します。IEEE 802.3/4/5 ネットワークの場合、< プロトコル・タイプ、ターゲット・プロトコル・アドレス > を物理アドレスにマップするために ARP モジュールを参照する必要があります。

ARP モジュールはこの ARP キャッシュ内でアドレスの検索を試行します。一致するペアを見つけると、これは対応する 48 ビットの物理アドレスを呼び出し元 (デバイス・ドライバ) に戻し、その後その呼び出し元がパケットを伝送します。これは、そのテーブル内にペアを見つけないと、パケットを破棄 (上位レベル・プロトコルが再送するという前提で) し、ARP 要求のネットワーク・ブロードキャストを生成します。詳細については、図 50 を参照してください。

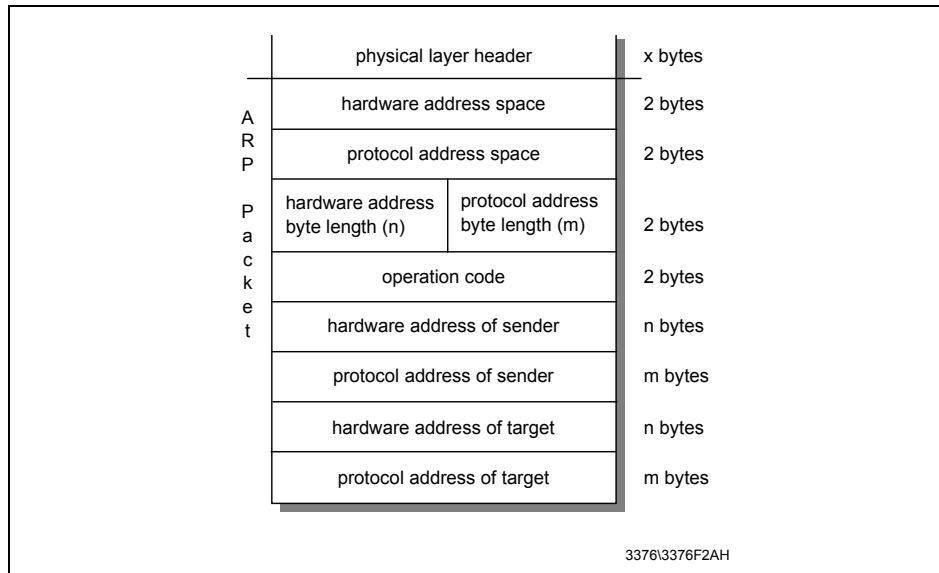


図 50 ARP - 要求 / 応答パケット

- ハードウェア・アドレス・スペース：ハードウェアのタイプを指定します。この例としては、イーサネットや Packet Radio Net があります。
- プロトコル・アドレス・スペース：プロトコルのタイプを指定します。IEEE 802 ヘッダー (IP または ARP) のイーサ・タイプ フィールドと同じです。
- ハードウェア・アドレス長：このパケットの中のハードウェア・アドレスの長さ (バイト単位) を指定します。IEEE 802.3 および IEEE 802.5 の場合、これは 6 です。
- プロトコル・アドレス長：このパケットの中のプロトコル・アドレスの長さ (バイト単位) を指定します。IP の場合、これは 4 です。
- 命令コード：これが ARP 要求 (1) または応答 (2) のどちらなのかを指定します。
- 送信元 / 宛先ハードウェア・アドレス：物理ネットワーク・ハードウェア・アドレスが入っています。IEEE 802.3 の場合、これは 48 ビットのアドレスです。
- 送信元 / 宛先プロトコル・アドレス：プロトコル・アドレスが入っています。TCP/IP の場合、これは 32 ビットの IP アドレスです。

ARP 要求パケットの場合は、宛先ハードウェア・アドレスがパケット内の唯一の未定義フィールドになります。

3.4.2.2 ARP パケットの受信

ホストが ARP パケット (ブロードキャスト要求か Point-to-Point 応答のいずれか) を受信すると、パケットは受信デバイス・ドライバーによって ARP モジュールに渡され、そこで図 51 のように処理されます。

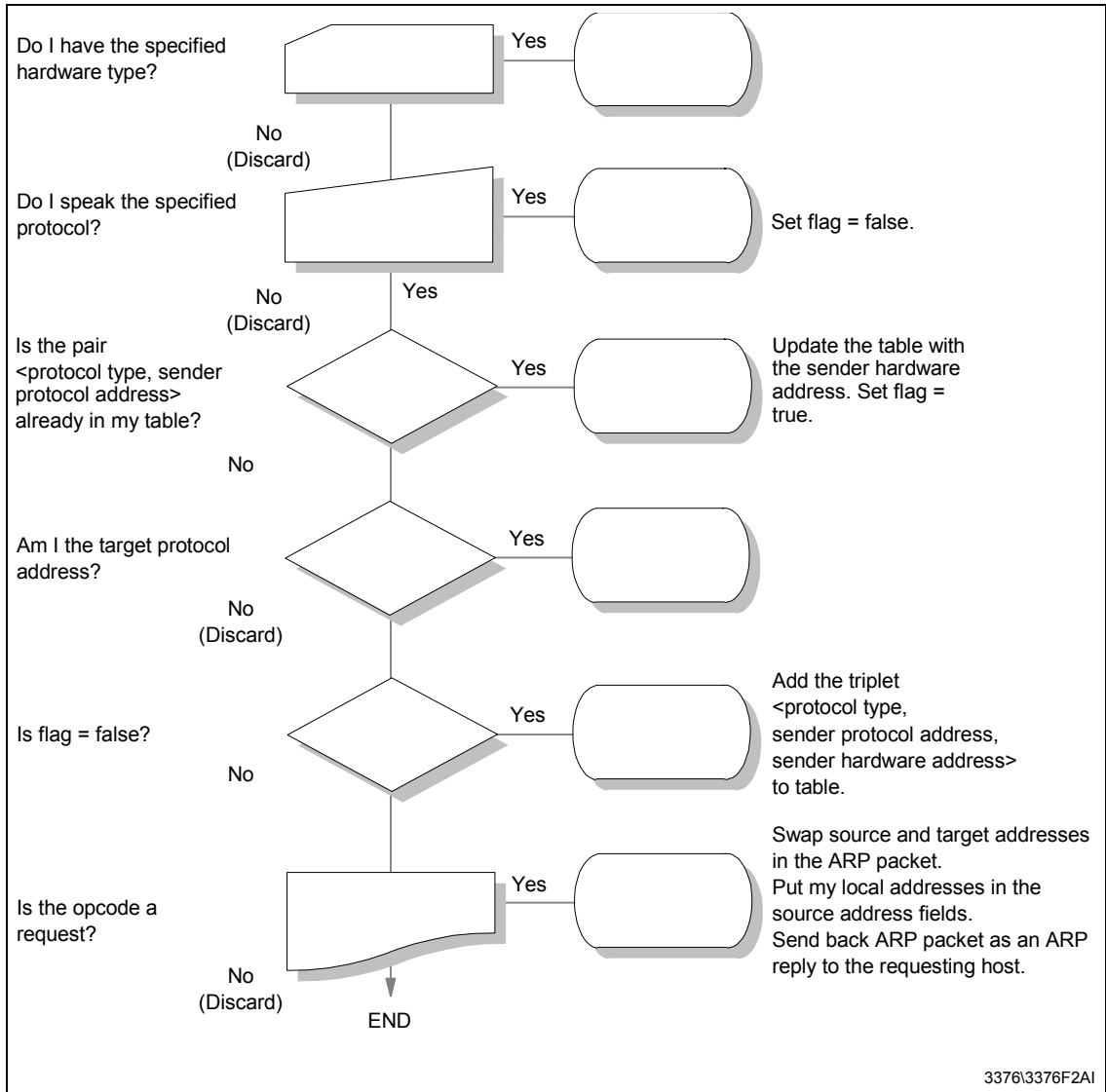


図 51 ARP - パケットの受信

要求ホストはこの ARP 応答を受信し、それを同じアルゴリズムに従って処理します。その結果、望ましいホストに対する < プロトコル・タイプ、プロトコル・アドレス、ハードウェア・アドレス > の 3 つがその検索テーブル (ARP キャッシュ) に追加されます。次に上位レベル・プロトコルがそのホストにパケットを送信しようとするときは、ARP モジュールが宛先ハードウェア・アドレスを検索し、そのホストにパケットが送信されます。

元の ARP 要求はネットワーク上のブロードキャストだったため、そのネットワーク上のすべてのホストがそのテーブル内の送信側ハードウェア・アドレスを更新する (ただし、すでにテーブル内に存在していた場合のみ) という点にご注意ください。

3.4.3 ARP とサブネット

サブネットが存在する場合も ARP プロトコルは変わりません。各 IP データグラムは最初に IP ルーティング・アルゴリズムを通過することを思い出してください。このアルゴリズムは、パケットの送信を行うべきハードウェア・デバイス・ドライバを選択します。その後で初めて、そのデバイス・ドライバと関連した ARP モジュールが参照されます。

3.4.4 プロキシ ARP または透過的なサブネット化

プロキシ ARP は、RFC 1027 – Using ARP to Implement Transparent Subnet Gateways で説明されています。これは実際には RFC 925 – Multi-LAN Address Resolution で提案されている方式の一部です。これは、ローカル・サブネットを構成するためのもう 1 つの方式であり、この方式では IP ルーティング・アルゴリズムの変更は不要ですが、サブネットを相互接続するルーターに変更が必要です。

3.4.4.1 プロキシ ARP の概念

複数のサブネットに分割され、ルーターによって相互接続された 1 つの IP ネットワークについて考慮しましょう。ここでは「古い」IP ルーティング・アルゴリズムを使用しており、このことはホストが複数の物理ネットワークの存在を認識しないことを意味します。以下では、同じ IP ネットワーク内の異なる物理ネットワーク上にあるホスト A および B と、その 2 つのサブネットワーク間のルーター R について考慮します。

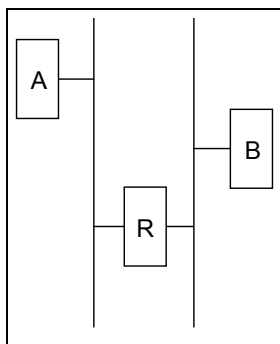


図 52 ARP - ルーターによって相互接続されたホスト

ホスト A は、ホスト B に IP データグラムを送信するとき、まず ARP プロトコルの使用を通してホスト B の物理ネットワーク・アドレスを判別する必要があります。

ホスト A は物理ネットワークの区別ができないので、この IP ルーティング・アルゴリズムはホスト B がローカル物理ネットワーク上にあると見なし、ブロードキャスト ARP 要求を送信します。ホスト B はこのブロードキャストを受信しませんが、ルーター R が受信します。ルーター R はサブネットを認識する、つまり、サブネット版の IP ルーティング・アルゴリズムを実行し、(宛先プロトコル・アドレス・フィールドの) ARP 要求の宛先が別の物理ネットワーク上にあることを認識することができます。その別のネットワークへの次のホップが異なる物理装置を経由することを、ルーター R のルーティング・テーブルが示している場合、これは、ホスト B のネットワーク・アドレスはルーター R 自身のホスト・アドレスであると述べて、自身がホスト B であるかのように ARP に応答します。

ホスト A はこの ARP 応答を受信してそれをそのキャッシュに入れ、ホスト B に対する将来の IP パケットはルーター R に送信します。ルーターはそのパケットを正しいサブネットに転送します。

結果は以下のような透過的サブネット化になります。

- 通常のホスト (A および B など) はサブネット化について認識しないため、「古い」IP ルーティング・アルゴリズムを使用します。
- サブネット間のルーターは以下を行う必要があります。
 - サブネット IP ルーティング・アルゴリズムを使用する。
 - 他のホストに代わって応答することのできる、変更された ARP モジュールを使用する。

詳細については、図 53 を参照してください。

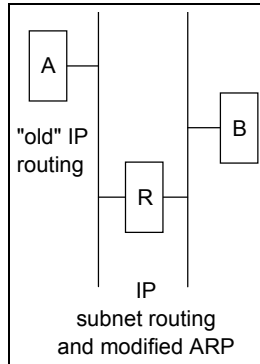


図 53 ARP - Proxy-ARP ルーター

3.5 Reverse Address Resolution Protocol (RARP)

RARP プロトコルは、ネットワークに固有な標準プロトコルです。これは、RFC 903 で説明されています。

ディスクレス・ワークステーションなどの一部のネットワーク・ホストは、ブートされるときにその独自の IP アドレスを認識しません。その独自の IP アドレスを判別するために、これらは ARP に似たメカニズムを使用しますが、ここでは、ホストのハードウェア・アドレスは既知のパラメーターで、IP アドレスは照会されるパラメーターです。これは、ハードウェア・アドレスからプロトコル・アドレスへのマッピングのデータベースを事前構成する必要のあるネットワーク上に、RARP サーバーが存在していなければならないという点で、ARP と根本的に異なっています。

3.5.1 RARP の概念

逆のアドレス解決が ARP アドレス解決を同じ方法で実行されます。ARP と同じパケット・フォーマット (図 50 を参照) が使用されます。

例外は命令コード・フィールドで、以下の値を取ります。

- 3: RARP 要求の場合
- 4: RARP 応答の場合

もちろん、フレームの物理ヘッダーは、この場合 ARP (0806 hex) や IP (0800 hex) の代わりに RARP を上位レベル・プロトコル (8035 hex) としてイーサ・タイプ フィールド内に示します。

RARP の概念自体から以下のようないくつかの違いが生じます。

- ARP だけが、すべてのホストがその独自のハードウェア・アドレスとプロトコル・アドレスの間のマッピングを認識していると見なします。RARP では、ネットワーク上の 1 つ以上のサーバー・ホストがハードウェア・アドレスとプロトコル・アドレスの間のマッピングのデータベースを保守し、これらがクライアント・ホストからの要求に応答できるようにでなければなりません。
- このデータベースが取ることのできるサイズのために、サーバー機能の部分は、通常はアダプターのマイクロコードの外にインプリメントされ、オプションで小さなキャッシュがマイクロコードの中に入れられます。そしてマイクロコード部分は RARP フレームの受信と伝送だけを担当し、RARP マッピング自体はホスト・マシンの正規プロセスとして実行しているサーバー・ソフトウェアによって保守されます。
- さらに、このデータベースの性質のため、データベースを手動で作成および更新するためのいくつかのソフトウェアが必要です。
- ネットワーク上に複数の RARP サーバーが存在する場合、RARP 要求側はそのブロードキャスト RARP 要求で受信した最初の RARP 応答だけを使用し、その他は破棄します。

3.6 Bootstrap protocol (BOOTP)

Bootstrap Protocol (BOOTP) によって、クライアント・ワークステーションは最小の IP スタックで初期化を行い、その IP アドレス、ゲートウェイ・アドレス、およびネーム・サーバーのアドレスを BOOTP サーバーに要求することができます。ネットワークで BOOTP を使用する場合、通常サーバーとクライアントは同じ物理 LAN セグメント上になります。BOOTP は、BOOTP 転送機能を持つルーターがある場合に、ブリッジ・セグメント間 (ソース・ルーティング・ブリッジが使用されているとき)、あるいはサブネット間でのみ使用することができます。

BOOTP はドラフト標準プロトコルです。その状態は「推奨 (Recommended)」です。BOOTP 仕様は、RFC 951 – Bootstrap Protocol にあります。

BOOTP には更新も存在し、RFC 951 を更新する RFC 1542 – Clarifications and Extensions for the Bootstrap Protocol と RFC 2132 – DHCP Options and BOOTP Vendor Extensions で説明されており、その一部は DHCP (131 ページの 3.7 『Dynamic Host Configuration Protocol (DHCP)』を参照) とのインターオペラビリティに関連しています。BOOTP に対する更新は、状態がそれぞれ「任意選択 (Elective)」と「推奨 (Recommended)」であるドラフト標準です。

BOOTP プロトコルは元々、ディスクレス・ホストをワークステーション、ルーター、端末集線装置などとしてネットワーク上でリモートにブートすることを可能にするメカニズムとして開発されました。これによって、構成情報を含まない最小の IP プロトコル・スタックは、必要なブート・コードのダウンロード・プロセスを開始するのに十分な情報を取得できます。BOOTP はダウンロード方法を定義していませんが、このプロセスは一般に RFC 906 – Bootstrap Loading Using TFTP で説明されている TFTP (391 ページの 10.2 『Trivial File Transfer Protocol (TFTP)』も参照) を使用します。BOOTP は、現在でも上記の目的でディスクレス・ホストに広く使用されていますが、手動で構成されていないクライアントに構成情報を送達するメカニズムとしてのみ使用されることもよくあります。

BOOTP プロセスには、以下のステップが含まれます。

1. クライアントがその独自のハードウェア・アドレスを判別します。これは普通はハードウェア上の ROM 内になります。
2. BOOTP クライアントがそのハードウェア・アドレスを UDP データグラムに入れてサーバーに送信します。このデータグラムの全内容は、図 54 に表示されています。クライアントは、その IP アドレスとサーバーのアドレスの両方または一方を知っているならそれらを使用するべきですが、BOOTP クライアントは通常の IP 構成データをまったく持ちません。クライアントは、その独自の IP アドレスを知らない場合は 0.0.0.0 を使用し、サーバーの IP アドレスを知らない場合は限定ブロードキャスト・アドレス (255.255.255.255) を使用します。UDP ポート番号は 67 です。
3. サーバーがデータグラムを受信し、クライアントの IP アドレスが含まれたその構成ファイル内でクライアントのハードウェア・アドレスを検索します。サーバーは UDP データグラム内の残りのフィールドを埋め、UDP ポート 68 を使用してそれをクライアントに戻します。これを行うためには、以下の 3 つの方式のいずれかが使用されます。
 - a. クライアントがその独自の IP アドレスを知っている (これが BOOTP 要求の中に含まれていた) 場合、サーバーはこのアドレスに直接データグラムを戻します。サーバーのプロトコル・スタック内の ARP キャッシュが、その IP アドレスとマッチングするハードウェア・アドレスを知らない可能性もあります。その判別には通常どおり ARP が使用されます。
 - b. クライアントがその独自の IP アドレスを知らない (BOOTP 要求でこれが 0.0.0.0 だった) 場合、サーバーはその独自の ARP キャッシュにだけ注目する必要があります。
 - c. クライアントがその独自の IP アドレスを知らないため、サーバーの ARP はクライアントのハードウェア・アドレスの検索に使用できず、そのためにサーバーの ARP は ARP 要求に応答できません。これは

「ニワトリが先か卵が先か」の問題と呼ばれます。このソリューションとしては、以下の 2 つが考えられます。

1. サーバーに、ARP そのものを使用せずにその独自の ARP キャッシュを直接更新するためのメカニズムがある場合は、それを行って直接データグラムを送信します。
 2. サーバーは、その独自の ARP キャッシュを更新できない場合、ブロードキャスト応答を送信する必要があります。
4. BOOTP クライアントは、応答を受信すると、その独自の IP アドレスを記録して (これが ARP 要求に応答できるようにし)、ブートストラップ・プロセスを開始します。

図 54 は、BOOTP メッセージ・フォーマットの概要を示しています。

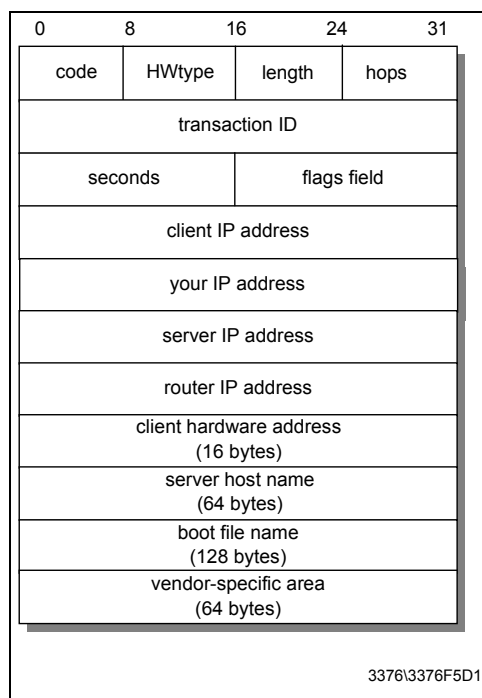


図 54 BOOTP メッセージ・フォーマット

- ・コード：要求か応答かを示します。
 - 1: 要求
 - 2: 応答

- HWtype: ハードウェアのタイプ。たとえば、以下のようなものがあります。

- 1: イーサネット
- 6: IEEE 802 ネットワーク :

STD 2 – Assigned Internet Numbers の完全なリストを参照してください。

- 長さ: バイト単位のハードウェア・アドレスの長さ。たとえば、イーサネットとトークンリングは両方とも 6 を使用します。

- ホップ: クライアントはこれを 0 に設定します。

これは、要求を他のサーバーにリレーするルーターによって増分され、ループの識別に使用されます。RFC 951 は、値 3 がループを表すことを示しています。

- トランザクション ID: このブート要求をこれが生成する応答と突き合わせるために使用される乱数。

- 秒数: クライアントによって設定されます。これは、クライアントがそのブート・プロセスを開始してからの経過時間 (秒単位) です。

- フラグ・フィールド: フラグ・フィールドの最上位ビットは、ブロードキャスト・フラグとして使用されます。他のビットはすべてゼロに設定されていなければなりません。これらは将来の利用のために予約されています。通常、BOOTP サーバーはユニキャスト・デリバリーを使用して、BOOTREPLY メッセージを直接クライアントに送達しようとします。IP ヘッダー内の宛先アドレスは、BOOTP IP アドレスに設定され、MAC アドレスは BOOTP クライアント・ハードウェア・アドレス に設定されます。ホストがその IP アドレスを知るまではユニキャスト IP データグラムを受信できない場合、このブロードキャスト・ビットは BOOTREPLY を IP および MAC ブロードキャストとして送信する必要のあるサーバーを示すように設定されていなければなりません。そうでない場合、このビットはゼロに設定されていなければなりません。

- クライアント IP アドレス: クライアントによって、その既知の IP アドレスか 0.0.0.0 のいずれかに設定されます。

- IP アドレス: クライアント IP アドレス・フィールドが 0.0.0.0 だった場合に、サーバーによって設定されます。

- サーバー IP アドレス: サーバーによって設定されます。

- ルーター IP アドレス: これは、クライアントに使用される汎用 IP ルーターではなく、BOOTP リレー・エージェントのアドレスです。これは、BOOTP 転送が使用されている場合に転送エージェントによって設定されます (129 ページの 3.6.1『BOOTP 転送』を参照)。

- ・クライアント・ハードウェア・アドレス：クライアントによって設定され、どの登録済みクライアントがブートしているかを識別するためにサーバーによって使用されます。
- ・サーバー・ホスト名：X'00' で終わるオプションのサーバー・ホスト名。
- ・ブート・ファイル名：クライアントはこれをヌルのまま残すか、使用するブート・ファイルのタイプを示すルーターなどの総称名を指定します。サーバーはクライアントに適したブート・ファイルの完全修飾ファイル名を戻します。値は X'00' で終わります。
- ・ベンダー固有エリア：オプションのベンダーに固有なエリア。クライアントが常に最初の 4 つのバイトを「magic cookie」で埋めるようにすることをお勧めします。ベンダーに固有な magic cookie を使用しない場合、クライアントは終了タグ (255) が後に続く 99.130.83.99 を使用して、残りのバイトをゼロに設定しなければなりません。ベンダー固有エリアには、*BOOTP* ベンダー拡張を含めることもできます。これらは、ブート時に、クライアントにその IP アドレスと共に渡すことのできるオプションです。たとえば、クライアントは、デフォルト・ルーターのアドレス、ドメイン・ネーム・サーバーとサブネット・マスクのアドレスを受信することもできます。いくつかの DHCP に固有なオプションを除き、BOOTP は DHCP と同じオプションを共用します。詳細については、RFC 2132 を参照してください。

BOOTP クライアントは応答の処理を済ませると、ブート・ファイルの転送と全ブート・プロセスの実行に移ります。TFTP を用いたこの実行方法の仕様については、RFC 906 を参照してください。ディスクレス・ホストの場合、全ブート・プロセスでは通常、ROM からロードされ、BOOTP および TFTP によって使用される最小の IP プロトコル・スタックが、ブート・ファイルの一部として送信され、クライアント用の正しいカスタマイズを含んだ通常の IP プロトコル・スタックで置き換えられます。

3.6.1 BOOTP 転送

BOOTP クライアントは BOOTP 要求に対して限定ブロードキャスト・アドレスを使用しますが、その場合は BOOTP サーバーがクライアントと同じサブネット上になければなりません。BOOTP 転送は、ルーターがサブネット間に BOOTP 要求を転送するためのメカニズムです。これは、大半のルーターで使用可能な構成オプションです。BOOTP 要求を転送するために構成されたルーターは、*BOOTP* リレー・エージェントと呼ばれます。

ルーターは通常、BOOTP クライアントによって使用される 0.0.0.0 などの正しくないソース・アドレスの含まれたデータグラムは破棄します。また、ルーターは通常、限定ブロードキャスト宛先アドレスを含むデータグラムも破棄します。しかし、BOOTP リレー・エージェントは、BOOTP クライアントからのこのようなデータグラムをポート 67 で受信します。

BOOTPREQUEST の受信で BOOTP リレー・エージェントによって実行されるプロセスは、以下のとおりです。

1. BOOTP リレー・エージェントは、BOOTPREQUEST を受信すると、要求を転送するかどうかを決定するために、まずホップ・フィールドを検査してすでに完了済みのホップの数を調べます。許容されるホップの数のしきい値は、通常構成可能です。
2. リレー・エージェントは、要求をリレーすることを決定すると、ルーター IP アドレス・フィールドの内容を検査します。このフィールドがゼロの場合、これは、このフィールドを BOOTPREQUEST が受信されたインターフェースの IP アドレスで埋めます。このフィールドにすでに他のリレー・エージェントの IP アドレスが入っている場合、それには手はつけられません。
3. ホップ・フィールドの値が増分されます。
4. 次に、リレー・エージェントは、BOOTPREQUEST を 1 つ以上の BOOTP サーバーに転送します。BOOTP サーバーのアドレスはリレー・エージェントで事前に構成されます。一部のインプリメンテーションはブロードキャスト転送を使用しますが、BOOTPREQUEST は通常はユニキャスト・フレームとして転送されます。
5. BOOTP サーバーは、非ゼロのルーター IP アドレス・フィールドを持つ BOOTPREQUEST を受信すると、ポート 67 でそのフィールド内のアドレスにある BOOTP リレー・エージェントに IP ユニキャスト BOOTREPLY を送信します。
6. BOOTP リレー・エージェントが BOOTREPLY を受信すると、メッセージ内の HWtype、長さ、およびクライアント・ハードウェア・アドレス・フィールドが、クライアントに応答を戻すための十分なリンク層情報を供給します。リレー・エージェントはブロードキャスト・フラグを検査します。このフラグが設定されている場合、エージェントは BOOTPREPLY をブロードキャストとしてクライアントに転送します。ブロードキャスト・フラグが設定されていない場合、リレー・エージェントは、IP アドレスで指定されているアドレスに応答をユニキャストとして送信します。

ルーターが BOOTP リレー・エージェントとして構成されている場合、BOOTP 転送タスクは、通常ルーターによって伝送されるサブネット間のデータグラムの交換のタスクとはかなり異なります。BOOTP メッセージの転送は、最終宛先として BOOTP メッセージを受信してから、他の宛先に転送する新しい BOOTP メッセージ生成することと考えられます。

3.6.2 BOOTP の考慮事項

BOOTP を使用すると、複数のクライアントの集中構成が行えます。しかし、そのためには、BOOTP サーバーに接続する可能性のあるすべてのクライアン

ト（たとえそのクライアントがほとんどアクティブにならないとしても）に対する、事前割り振り済み IP アドレスの入った静的テーブルが保持されていなければなりません。このことは、必要な IP アドレスの数に関する保証がないことを意味します。BOOTP を使用する環境では、クライアントは有効な MAC アドレスを持つ場合にだけ、サーバーによって IP アドレスを割り振られるので、ある程度のセキュリティーが存在します。

3.7 Dynamic Host Configuration Protocol (DHCP)

DHCP はドラフト標準プロトコルです。その状態は「任意選択 (elective)」になっています。現行の DHCP 仕様は、RFC 2131 – Dynamic Host Configuration Protocol および RFC 2132 – DHCP Options and BOOTP Vendor Extensions にあります。

Dynamic Host Configuration Protocol (DHCP) は、TCP/IP ネットワーク上のホストに構成情報を渡すためのフレームワークを提供しています。DHCP は、BOOTP プロトコルに基づき、それに加えて再利用可能ネットワーク・アドレスの自動割り振り機能と追加の構成オプションを持ちます。BOOTP については、125 ページの 3.6『Bootstrap protocol (BOOTP)』を参照してください。DHCP メッセージは、BOOTP サーバーのウェルノウン・ポートである UDP ポート 67 と、BOOTP クライアントのウェルノウン・ポートである UDP ポート 68 を使用します。DHCP 参加者は、BOOTP 参加者と相互操作することができます。詳細については、141 ページの 3.7.8『BOOTP と DHCP のインターオペラビリティ』を参照してください。

DHCP は以下の 2 つのコンポーネントで構成されます。

1. ホストに固有な構成パラメーターを DHCP サーバーからホストに送達するプロトコル。
2. 一時または永続ネットワーク・アドレスをホストに割り振るためのメカニズム。

IP は、プロトコル・インプリメンテーション・ソフトウェア内の多数のパラメーターの設定を必要とします。IP は多数の異なる種類のネットワーク・ハードウェア上で使用できるため、これらのパラメーターの値は、推測したり、正確なデフォルトを持つと見なしたりすることはできません。すでに使用されているネットワーク・アドレスを発見するために、ポーリング / 防御メカニズムに基づく分散アドレス割り振り方式を使用しても、ホストが常にそのネットワーク・アドレスを防御できるとは限らないので、固有なネットワーク・アドレスは保証されません。

DHCP は以下の 3 つの IP アドレス割り振りメカニズムをサポートしています。

1. 自動割り振り

DHCP は永続 IP アドレスをホストに割り当てます。

2. 動的割り振り

DHCP は限定された期間 IP アドレスを割り当てます。このようなネットワーク・アドレスは、リースと呼ばれます。これは、割り当てられていたホストに必要とされなくなったアドレスの自動再利用を可能にする唯一のメカニズムです。

3. 手動割り振り

ホストのアドレスはネットワーク管理者によって割り当てられます。

3.7.1 DHCP メッセージ・フォーマット

DHCP メッセージのフォーマットは、図 55 に表示しています。

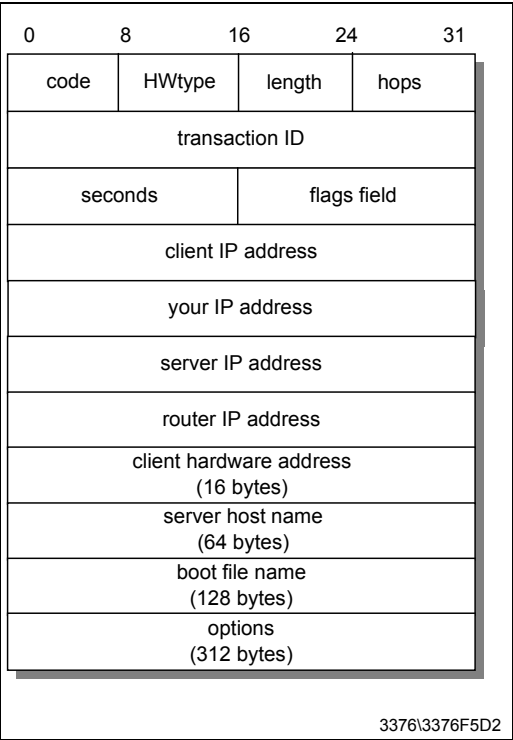


図 55 DHCP メッセージ・フォーマット

- ・コード：要求か応答かを示します。

- 1 要求
- 2 応答
- HWtype: ハードウェアのタイプ。たとえば、以下のようなものがあります。
 - 1 イーサネット
 - 6 IEEE 802 ネットワーク

STD 2 – Assigned Internet Numbers の完全なリストを参照してください。

- 長さ: バイト単位のハードウェア・アドレスの長さ。たとえば、イーサネットとトークンリングは両方とも 6 を使用します。
- ホップ: クライアントはこれを 0 に設定します。これは、要求を他のサーバーにリレーするルーターによって増分され、ループの識別に使用されます。RFC 951 は、値 3 がループを表すことを示しています。
- トランザクション ID: このブート要求をこれが生成する応答と突き合わせるために使用される乱数。
- 秒数: クライアントによって設定されます。これは、クライアントがそのブート・プロセスを開始してからの経過時間 (秒単位) です。
- フラグ・フィールド: フラグ・フィールドの最上位ビットは、ブロードキャスト・フラグとして使用されます。他のすべてのビットはゼロに設定されていなければならない、これらは将来の利用のために予約されています。通常、DHCP サーバーはユニキャスト・デリバリーを使用して、DHCP メッセージを直接クライアントに送達しようとします。IP ヘッダー内の宛先アドレスは、DHCP IP アドレスに設定され、MAC アドレスは DHCP クライアント・ハードウェア・アドレスに設定されます。ホストがその IP アドレスを知るまではユニキャスト IP データグラムを受信できない場合、このブロードキャスト・ビットは、DHCP 応答を IP および MAC ブロードキャストとして送信する必要のあるサーバーを示すように設定されていなければならない。そうでない場合、このビットはゼロに設定されていなければならない。
- クライアント IP アドレス: クライアントによって設定されます。その既知の IP アドレス、または 0.0.0.0 のいずれかです。
- IP アドレス: クライアント IP アドレス・フィールドが 0.0.0.0 だった場合に、サーバーによって設定されます。
- サーバー IP アドレス: サーバーによって設定されます。
- ルーター IP アドレス: これは、クライアントに使用される汎用 IP ルーターではなく、BOOTP リレー・エージェントのアドレスです。これは、BOOTP 転送が使用されている場合に転送エージェントによって設定されます (129 ページの 3.6.1 『BOOTP 転送』を参照)。

- クライアント・ハードウェア・アドレス：クライアントによって設定されます。DHCP は、クライアントの識別に使用されるクライアント ID オプションを定義しています。このオプションが使用されていない場合、クライアントはその MAC アドレスによって識別されます。
- サーバー・ホスト名：X'00' で終わるオプションのサーバー・ホスト名。
- ブート・ファイル名：クライアントはこれをヌルのまま残すか、使用するブート・ファイルのタイプを示すルーターなどの総称名を指定します。DHCPDISCOVER 要求では、これはヌルに設定されます。サーバーは完全修飾パス名を DHCPOFFER 要求に入れて戻します。値は X'00' で終わります。
- オプション：DHCP メッセージのオプション・フィールドの最初の 4 つのバイトには、magic cookie (99.130.83.99) が含まれます。オプション・フィールドの残りの部分は、オプションと呼ばれるタグ付きパラメーターで構成されます。詳細については、RFC 2132 を参照してください。

3.7.2 DHCP メッセージ・タイプ

DHCP メッセージは、以下のいずれかのカテゴリに該当します。

- DHCPDISCOVER: クライアントによる、使用可能な DHCP サーバーを見付けるためのブロードキャスト。
- DHCPOFFER: DHCPDISCOVER に対するサーバーからの応答、および IP アドレスと他のパラメーターのオファリング。
- DHCPREQUEST: クライアントからサーバーへのメッセージ。これは以下のいずれかを行います。
 - いずれかのサーバーから提案されたパラメーターを要求し、他のすべての提案は拒否します。
 - システムまたはネットワーク変更（たとえば、リブート）の後に、以前の割り振られたアドレスを検査します。
 - 特定のアドレスのリースの延長を要求します。
- DHCPACK: IP アドレスを含むパラメーターの入ったサーバーからクライアントへの確認通知。
- DHCPNACK: サーバーからクライアントへの否定通知。クライアントのリースの有効期限が切れているか、要求された IP アドレスが正しくないことを示します。
- DHCPDECLINE: 提案されたアドレスがすでに使用中であることを示す、クライアントからサーバーへのメッセージ。
- DHCPRELEASE: 残りのリースをキャンセルしてネットワーク・アドレスを解放する、クライアントからサーバーへのメッセージ。

- DHCPINFORM: DHCP サーバーに対して追加の構成パラメーターを要求する、すでに IP アドレス (たとえば、手動で構成されたもの) を持っているクライアントからのメッセージ。

3.7.3 新規ネットワーク・アドレスの割り振り

このセクションでは、クライアントがそのネットワーク・アドレスを知らない場合の、クライアント / サーバー対話について説明します。DHCP サーバーは、ネットワーク・アドレスのブロックを持ち、それから新規アドレスの要求を満たせることを想定します。各サーバーはまた、永続ローカル・ストレージ内に、割り振り済みのアドレスとリースのデータベースを保持しています。

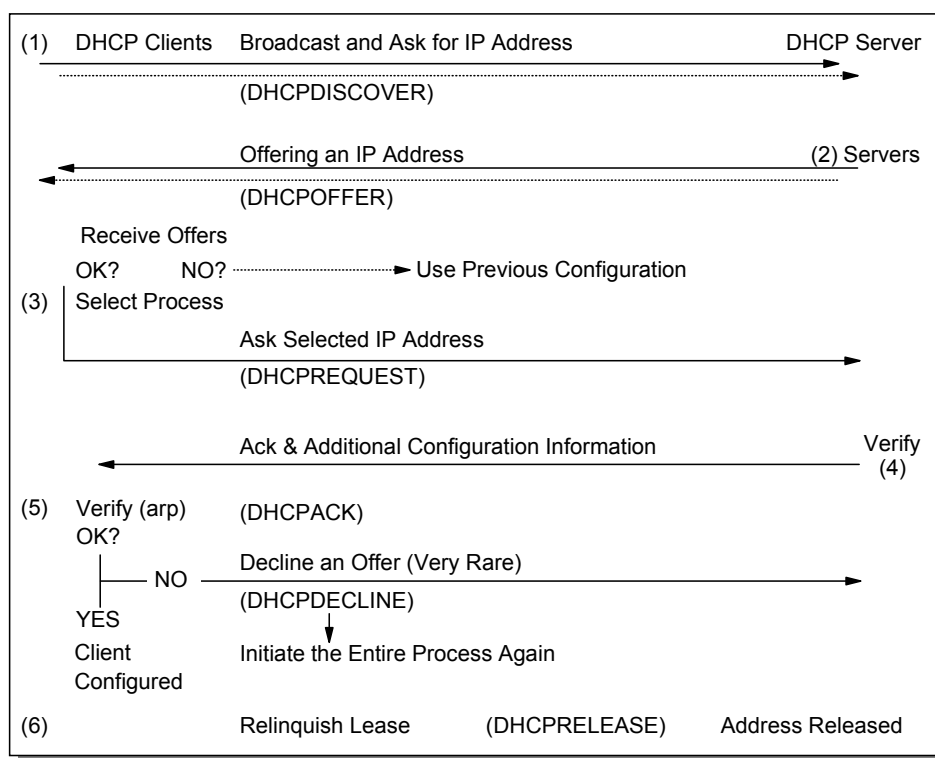


図 56 DHCP クライアントと DHCP サーバーの対話

以下は、図 56 に示されている DHCP クライアント / サーバー対話ステップの説明です。

1. クライアントはそのローカル物理サブネット上で DHCPDISCOVER メッセージをブロードキャストします。この時点で、クライアントは INIT 状態にあります。DHCPDISCOVER メッセージには、ネットワーク・アドレス

提案やリース期間などのいくつかのオプションが含まれる可能性があります。

2. 各サーバーは、使用可能ネットワーク・アドレス (IP アドレス) と他の構成オプションの含まれた DHCPPOFFER メッセージで応答することができます。サーバーは、そのアドレスをクライアントに提案済みのものとして記録することにより、最初のクライアントがその構成を完了する前にその次の DHCPDISCOVER メッセージを受信した場合に、同じアドレスが他のクライアントに提案されるのを防ぐことができます。
3. クライアントは、1 つ以上のサーバーから 1 つ以上の DHCPPOFFER メッセージを受け取ります。クライアントは、提案された構成パラメーターに基づくメッセージを選択し、これが選択したメッセージを示すサーバー ID オプションと、選択された提案内の IP アドレスから取られた要求された IP アドレス・オプションを含む DHCPREQUEST メッセージをブロードキャストします。

提案が受信されない場合、クライアントに以前のネットワーク・アドレスの知識があり、そのリースがまだ有効であれば、リースの有効期限が切れるまでクライアントはそのアドレスを再利用することができます。

4. サーバーは DHCPREQUEST ブロードキャストをクライアントから受信します。DHCPREQUEST メッセージで選択されていないサーバーは、クライアントがそのサーバーの提案を拒否したということの通知としてメッセージを使用します。DHCPREQUEST メッセージで選択されたサーバーは、永続ストレージへのクライアントのバインディングをコミットし、要求クライアント用の構成パラメーターを含む DHCPACK メッセージで応答します。クライアント・ハードウェアと割り当てられたネットワーク・アドレスの組み合わせは、クライアントのリースの固有 ID を構成し、クライアントとサーバーの両方によって任意の DHCP メッセージ内で参照されるリースの識別に使用されます。DHCPACK メッセージ内の IP アドレス・フィールドは、選択されたネットワーク・アドレスで埋め込まれます。
5. クライアントは構成パラメーターを含んだ DHCPACK メッセージを受信します。クライアントは、たとえば、割り振り済みのネットワーク・アドレスに対して ARP を用いるなどしてパラメーターに対する最終検査を実行し、リースの期間と DHCPACK メッセージで指定されているリース識別 cookie を記録します。この時点で、クライアントは構成済みです。

クライアントが DHCPACK メッセージ内のパラメーターについての問題 (たとえば、アドレスがネットワーク上ですでに使用されているなどの) を検出した場合、そのクライアントは DHCPDECLINE メッセージをサーバーに送信して構成プロセスを再始動します。ループの場合の過剰ネットワーク・トラフィックを防ぐため、クライアントは構成プロセスを再始動する前に少なくとも 10 秒待たなければなりません。DHCPDECLINE を受信したら、サーバーは提案されているアドレスを使用不可としてマークす

る (そしておそらく構成問題が存在することをシステム管理者に通知する) 必要があります。

クライアントは、DHCPNAK メッセージを受信する場合、構成プロセスを再始動します。

6. クライアントはサーバーに DHCPRELEASE メッセージを送信することによって、ネットワーク・アドレス上のリースを解放することを選択することができます。クライアントは、そのネットワーク・アドレスとハードウェア・アドレスを含めることによって解放するリースを識別します。

注：

DHCP サーバーから DHCP クライアントへの応答は、TCP/IP スタックが完全に構成される前にクライアントがユニキャスト・メッセージを受信できるかどうか (これはインプリメンテーションによって異なります) に応じて、ブロードキャストかユニキャストになります。(133 ページのフラグ・フィールドを参照してください。)

3.7.4 DHCP リース更新プロセス

このセクションでは、すでに構成済みの DHCP サーバーとクライアントの間の対話と、リースの満了と更新を保証するプロセスについて説明します。

1. サーバーは、DHCPACK を IP アドレスおよび構成パラメーターと共にクライアントに送信するときに、そのアドレスのリース時間の開始の記録も行います。このリース時間は、2 つのタイマー値 T1 および T2 と共に、DHCPACK メッセージ内のオプションの 1 つとしてクライアントに渡されます。クライアントは、リース時間が存続している間、指定されたアドレスを使用する正当な権利を有します。受信した構成を適用したら、さらにクライアントはタイマー T1 および T2 を始動します。この時点で、クライアントは BOUND 状態にあります。T1 および T2 の時間は、サーバーで構成可能なオプションですが、T1 は T2 より小さく、T2 はリース時間より小さくしなければなりません。RFC 2132 によれば、T1 はデフォルトで (0.5 * リース時間) であり、T2 はデフォルトで (0.875 * リース時間) です。
2. タイマー T1 の有効期限が切れると、クライアントはそのアドレスを提案したサーバーに DHCPREQUEST (ユニキャスト) を送信し、指定された構成のリースを延長するよう求めます。この時点でクライアントは RENEWING 状態に入ります。通常サーバーは新しいリース時間を指示する DHCPACK メッセージで応答し、タイマー T1 と T2 がクライアントで適宜リセットされます。サーバーはまた、そのリース時間の記録をリセットします。通常の場合、アクティブ・クライアントはこの方法で無期限にリースを更新し続け、リースの有効期限が切れることはありません。

3. タイマー T2 の有効期限が切れるまでに DHCPACK が受信されないと、クライアントは REBINDING 状態に入ります。この時点で、これはこのリースを延長するための DHCPREQUEST メッセージをブロードキャストします。この要求は、ネットワーク上の任意の DHCP サーバーからの DHCPACK メッセージで確認することができます。
4. クライアントは、そのリースの有効期限が切れた後に DHCPACK メッセージを受信しない場合、その現行の TCP/IP 構成の使用を停止しなければなりません。この後クライアントは、DHCPDISCOVER ブロードキャストを発行して任意の有効アドレスの取得を試行し、INIT 状態に戻ることができます。

図 57 は、DHCP プロセスとこのプロセス中のクライアントの状態の変化を示しています。

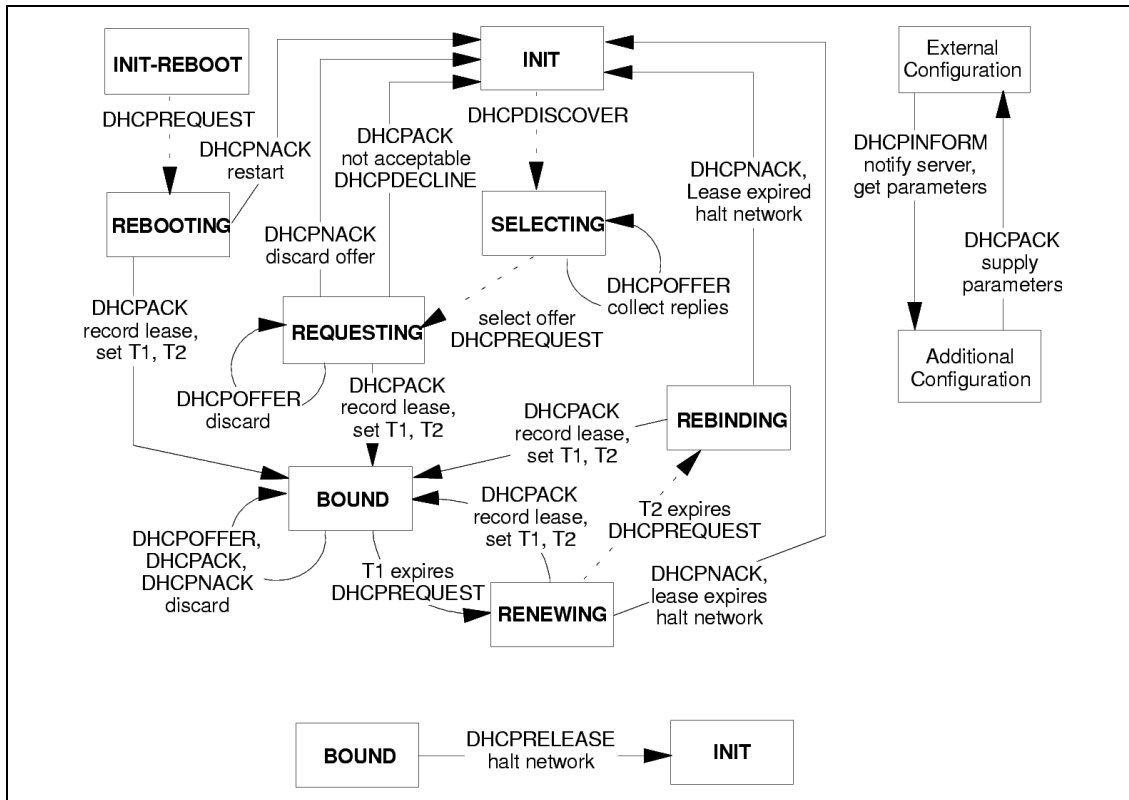


図 57 DHCP クライアントの状態と DHCP プロセス

3.7.5 以前の割り振られたネットワーク・アドレスの再利用

クライアントが以前に割り振られたネットワーク・アドレスを覚えていてそれを再利用する場合は、以下のステップを実行します。

1. クライアントはそのローカル・サブネット上で DHCPREQUEST メッセージをブロードキャストします。DHCPREQUEST メッセージにはクライアントのネットワーク・アドレスが含まれます。
2. クライアントの構成パラメーターの知識を持つサーバーが、DHCPACK メッセージでクライアントに応答し (ただし、リースがまだ現行である場合)、同時にリースが更新されます。

クライアントのリースの有効期限が切れている場合、クライアントの知識を持つサーバーは DHCPNACK で応答します。

3. クライアントは構成パラメーターを含んだ DHCPACK メッセージを受信します。クライアントは、パラメーターに対する最終検査を実行し、リースの期間と DHCPACK メッセージで指定されているリース識別 cookie を記録します。この時点で、クライアントは構成され、その T1 および T2 タイマーがリセットされます。

クライアントが DHCPACK メッセージ内のパラメーターについての問題を検出した場合、そのクライアントは DHCPDECLINE メッセージをサーバーに送信し、新規ネットワーク・アドレスを要求して構成プロセスを再始動します。クライアントは、DHCPNAK メッセージを受信する場合、記憶しているそのネットワーク・アドレスを使用することはできません。代わりに、135 ページの 3.7.3『新規ネットワーク・アドレスの割り振り』の説明に従って構成プロセスを再開することによって、新規アドレスを要求しなければなりません。

注

ローカル・ネットワーク構成は、ホストまたはユーザーが知ることなく変更される可能性があります。たとえば、システム・リブート時やローカル・ネットワークからの切断の後など、ローカル・ネットワーク・パラメーターが変更されたときはその都度、ホストが DHCP を使用してその IP アドレスとネットワーク・パラメーターを再取得あるいは検査する必要があります。クライアントが複数の IP インターフェースを持つ場合は、そのおのおのを別個に DHCP で構成する必要があります。

詳細については、上記の RFC を参照してください。

3.7.6 構成パラメーター・リポジトリ

DHCP は、ネットワーク・クライアント用のネットワーク・パラメーターの永続ストレージを提供しています。DHCP サーバーは、各クライアント用のキー値エントリー、たとえば、IP サブネット番号とサブネット内の固有 ID (通常はハードウェア・アドレス) などのいくつかの固有 ID であるキーを保管し、この値にはこの特定のクライアントに最後に割り振られた構成パラメーターが含まれます。

この効果として、アドレスのプールが過剰にサブスクライブされておらず、以前のアドレスがすでに他のクライアントに割り振られていない限り、DHCP クライアントはサーバーによって常に同じ IP アドレスに割り振られる傾向を持つようになります。

3.7.7 DHCP の考慮事項

DHCP の IP アドレスと構成パラメーターの動的割り振りは、ネットワーク管理者の膨大な手動構成作業を軽減します。装置をネットワークからネットワークへと移動したり、現行ネットワーク用の有効な構成パラメーターを自動的に取得したりするための機能は、モバイル・ユーザーにも大きな益となる可能性があります。また、IP アドレスはクライアントが実際にアクティブなときにだけ割り振られるので、適当な短さのリース時間の使用と、モバイル・クライアントには複数のアドレスを割り振る必要がないという事実により、組織で使用されるアドレスの合計数を削減することが可能です。しかし、DHCP がインプリメントされているときは、以下の点について考慮する必要があります。

- DHCP は本来保護されない UDP 上に構築されます。通常の操作では、許可されていないクライアントはネットワークに接続して有効な IP アドレスと構成を取得することができません。これを防ぐため、特定の MAC アドレスに IP アドレスを事前割り振りする (BOOTP と同様に) こともできますが、その場合は管理ワークロードが増え、アドレスのリサイクルの利点が失われます。許可されていない DHCP サーバーがセットアップされ、クライアントに偽の、もしかすると破壊的な情報が送信される可能性もあります。
- 自動または動的アドレス割り振りが使用されている DHCP 環境では、任意の特定の時点でのクライアントの IP アドレスを判別することは通常はできません。この場合、静的 DNS サーバーが使用されているなら、その DNS サーバーにはクライアントの IP アドレス・マッピングへの有効なホスト名が含まれていない可能性があります。DNS 内にクライアント・エントリーを持つことがネットワークにとって重要であれば、DHCP を使用して IP アドレスを手動でこのクライアントに割り当ててから、DNS 内のクライアント・マッピングを適宜管理することができます。

3.7.8 BOOTP と DHCP のインターオペラビリティ

DHCP メッセージのフォーマットは BOOTP メッセージのフォーマットに基づいています。そのため、BOOTP および DHCP クライアントは特定の状況下で相互作用することができます。すべての DHCP メッセージは DHCP メッセージ・タイプ (51) オプションを含んでいます。このオプションを含んでいないメッセージは、BOOTP クライアントからのものと見なされます。

DHCP サーバーでの BOOTP クライアントのサポートは、必要に応じてシステム管理者によって構成される必要があります。DHCP サーバーは BOOTPREQUEST メッセージに対し、DHCP OFFER ではなく BOOTPREPLY で応答します。この方法で構成されていない DHCP サーバーはどれも、送信されてくるすべての BOOTPREQUEST フレームを破棄します。DHCP サーバーは (その未割り振りアドレスのプールから) 静的アドレスまたは動的アドレスを BOOTP クライアントに提案することがあります (ただし、すべての BOOTP インプリメンテーションが自動アドレスを認識するとは限りません)。BOOTP クライアントに自動アドレスが提案されている場合、クライアントは DHCP のリース・メカニズムを認識しないので、そのアドレスは無限のリース時間を持っている必要があります。

DHCP メッセージは、BOOTP リレー・エージェントとして構成されたルーターによって転送されます。

第 4 章ルーティング・プロトコル

IP プロトコルによって提供されている基本機能の 1 つは、異なる物理ネットワーク間に接続を形成する機能です。この機能を実行するシステムは *IP ルーター* と呼ばれます。このタイプの装置は、複数の物理ネットワークに接続し、ネットワークの間でデータグラムを転送します。

データをリモート宛先に送信するときに、ホストはデータグラムをローカル・ルーターに渡します。ルーターはデータグラムを最終宛先まで転送します。これらのデータグラムは、宛先の LAN セグメントに接続したルーターに到達するまで、ルーターからルーターへと移動します。終端間パス上の各ルーターは、宛先へ到達するために使用するネクスト・ホップ装置を選択します。ネクスト・ホップとは、宛先に到達するためのパス上の次の装置を表します。これは、この中間システムに接続された物理ネットワーク上にあります。この物理ネットワークは、システムが最初にデータグラムを受信した物理ネットワークとは異なるので、中間ホストは IP データグラムをあるネットワークから他のネットワークへ転送した（つまり、ルーティングした）ということになります。

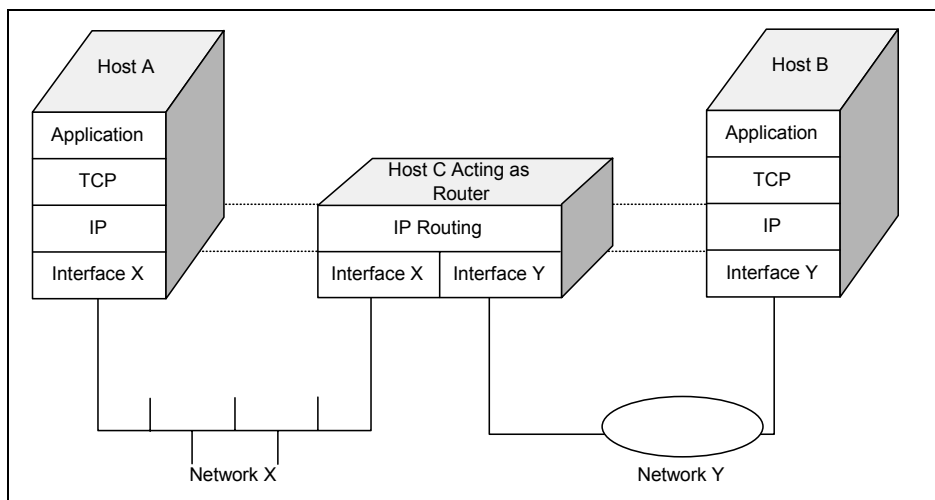


図 58 IP ルーティング操作

図 58 は、ホスト C がネットワーク X とネットワーク Y の間でパケットを転送している環境を表しています。

ネットワーク・セグメント間でのパケットの転送には、各装置の IP ルーティング・テーブルが使用されます。基本テーブルには、ルーターにローカルに接続されたネットワークに関する情報が含まれます。装置の構成を拡張して、

これにリモート・ネットワークの詳細情報が含まれるようにすることもできます。この情報は、環境全体のより完全なビューを提示します。

堅固なルーティング・プロトコルは、IP ルーティング・テーブル内に情報を動的に作成して管理するための機能を提供しています。ネットワーク・トポロジーが変更されると、ルーティング・テーブルは手操作による介入をほとんどあるいはまったくなしに更新されます。この章では、いくつかの IP ルーティング・プロトコルと、その各プロトコルがどのようにこの情報を管理するのかについて説明します。

注

OSI プロトコル・スタックの階層化モデル内の各プロトコルの位置は、本書の他のセクションで示しています。ルーティング機能は、インターネットワーク層の一部として組み込まれています。ただし、ルーティング・プロトコルの主要な機能は、他のルーターとルーティング情報を交換することです。この点では、アプリケーション・プロトコルのほうがルーティング・プロトコルによく似た働きをします。したがって、この章ではプロトコル・スタック全体の中でのこれらのプロトコルの位置は示しません。

注

初期の IP ルーティング資料では、IP ルーターが *IP ゲートウェイ* と呼ばれていることがよくあります。

4.1 自律システム (Autonomous System)

ルーティング・プロトコルの機能と目的について理解する上で、自律システム (AS) の定義を欠かすことはできません。AS は、より大きな IP ネットワークの論理部分として定義されます。AS は通常 1 つの組織内のインターネットワークから成ります。これは単一の管理機関によって管理されます。図 59 で表されているように、AS は同じ組織によって管理されている他の自律システムに接続することができます。あるいは、他の公用または私設ネットワークに接続することもできます。

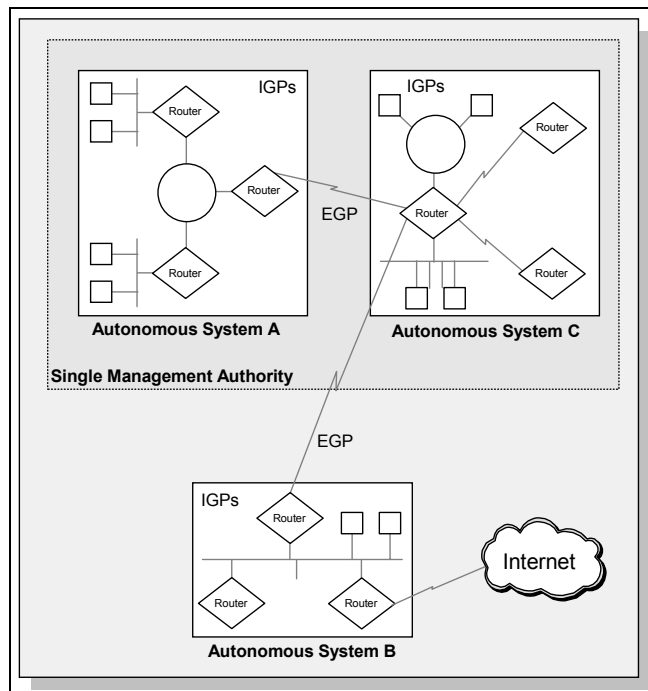


図 59 自律システム

あるルーティング・プロトコルは、AS 内のルーティング・パスを決定するために使用されます。またあるものは、自律システムのセットを相互接続するために使用されます。

- Interior Gateway Protocol (IGP): Interior Gateway Protocol によって、ルーターは AS 内で情報を交換することができます。このプロトコルの例としては、Open Short Path First (OSPF) と Routing Information Protocol (RIP) があります。
- Exterior Gateway Protocol (EGP): Exterior Gateway Protocol は、自律システム間での要約情報の交換を可能ができます。このタイプのルーティング・プロトコルの例としては、Border Gateway Protocol (BGP) があります。

図 59 は、内部プロトコルと外部プロトコルの間の相互作用を表しています。これには、各 AS 内でルーティング情報を保守するために使用される内部プロトコルが示されています。この図にはまた、自律システム間でルーティング情報を保守する外部プロトコルも示されています。

AS 内では、複数の内部ルーティング・プロセスが使用されることがあります。その場合、他の自律システムには、その AS が単一の一貫した内部ルーティング計画を持つものとして映らなければなりません。AS は一貫性のある内部宛先のビューを提供する必要があります。

4.2 IP ルーティングと IP ルーティング・アルゴリズムのタイプ

装置での IP ルーティング・テーブルの作成と保守には、ルーティング・アルゴリズムが使用されます。ルーティング・テーブルの作成に使用される基本方式は以下の 2 つです。

- 静的ルーティング：静的ルーティングでは、ネットワーク上のパスを表す事前にプログラミングされた定義が使用されます。
- 動的ルーティング：動的ルーティング・アルゴリズムによって、ルーターは自動的にネットワーク上のパスを発見してその認識を保守します。この自動ディスカバリーは、いくつかの現在使用可能な動的ルーティング・プロトコルを使用することができます。これらのプロトコルの間の違いは、これらが宛先ネットワークへの新しい経路を発見して計算する方法にあります。それらは以下の 3 つの大きなカテゴリーに分類することができます。
 - Distance vector protocol
 - Link state protocol
 - Hybrid protocol

このセクションの残りの部分では、各アルゴリズムの操作について詳細に説明します。

さまざまなプロトコルが存在することには、以下のようないくつかの理由があります。

- ネットワーク内のルーティングとネットワーク間のルーティングとは、セキュリティ、安定度、およびスケーラビリティに関する要件が基本的に異なります。これらの要件を満たすために、異なるルーティング・プロトコルが開発されました。
- 確立されているプロトコルで観察された欠陥を解消するために、新しいプロトコルが開発されました。
- サイズの異なるネットワークは、異なるルーティング・アルゴリズムを使用することができます。小～中程度のサイズのネットワークは、多くの場合、環境の単純さを反映したルーティング・プロトコルを使用します。しかし、これらのプロトコルは大規模な相互接続ネットワークをサポートするための規模を持ちません。これらの環境をサポートするには、もっと複雑なルーティング・アルゴリズムが必要です。

4.2.1 静的ルーティング

静的ルーティングはネットワーク管理者によって手動で実行されます。ネットワーク上の経路の発見と伝搬は、管理者が担当します。この定義は環境内のすべてのルーティング装置で手動でプログラミングされます。

装置が構成されると、それは単純に事前に決められたポートへパケットを転送します。ネットワークの現行トポロジーに関連したルーター間の通信はありません。

冗長度の低い小さなネットワークでは、このプロセスは比較的管理が簡単です。ただし、このアプローチには IP ルーティング・テーブルの保守に関連した以下のようないくつかの欠点があります。

- 静的経路は、大規模なネットワーク環境ではかなりの調整と保守を必要とします。
- 静的経路は、ネットワークの現行の操作状態に動的に適応できません。宛先サブネットワークが到達不能になっても、そのネットワークを指す静的経路はルーティング・テーブル内に残ります。トラフィックは引き続きその宛先まで転送されます。ネットワーク管理者が静的経路を変更して新しいトポロジーを反映しない限り、トラフィックは代替パスが存在したとしてもそれを使用できません。

通常静的経路は単純ネットワーク・トポロジーでのみ使用されます。しかし、静的ルーティングが望ましい可能性のある状況はこれ以外にもあります。たとえば、静的ルーティングは以下の場合に使用できます。

- デフォルト経路を手動で定義する。ルーティング・テーブルに宛先へのより具体的な経路が含まれていない場合は、この経路がトラフィックの転送に使用されます。
- ネットワーク内で自動的に公示されない経路の定義。
- 使用状況や回線料金のため、容量の小さい WAN 接続を通してルーティング公示トラフィックを送信するのが望ましくない場合。
- 複雑なルーティング・ポリシーが必要な場合。たとえば、特定のホストに宛てられたトラフィックが指定されているネットワーク・パスをトラバースするよう保証するには、静的経路を使用できます。
- より安全なネットワーク環境を提供する。管理者は環境内に定義されたすべてのサブネットワークを認識します。これらのサブネットワーク間で許可される通信はすべて管理者が明確に認可します。
- リソースの使用効率を良くする。この方式のルーティング・ネットワーク管理では、近隣装置間の経路を公示するためのネットワーク帯域幅は不要です。さらに、ネットワーク・パスの計算にプロセッサ・メモリーと CPU サイクルがそれほど使用されません。

4.2.2 距離ベクトル・ルーティング (Distance vector routing)

距離ベクトル・アルゴリズムは、動的ルーティング・プロトコルの一例です。このアルゴリズムを使用する場合、ネットワーク内の各装置は自動的に IP ルーティング・テーブルを作成して保守できます。

距離ベクトル・ルーティングの背後にある原理は単純です。インターネットワーク内の各ルーターは、すべての既知宛先へのそこからの距離またはコストを保持します。この値は、パスの全体的な望ましさを表します。大きい値が関連付けられたパスよりも、小さいコスト値が関連付けられたパスのほうが使用するのに魅力あるパスとなります。最小コストで表されたパスは、宛先へ到達するための優先パスになります。

この情報は距離ベクトル・テーブル内に保持されます。このテーブルは各近隣ルーターに定期的に公示されます。各ルーターはこの公示を処理ネットワーク上の最善のパスを判別します。

距離ベクトル・アルゴリズムの主な利点は、基本的にインプリメントとデバッグが簡単なことです。これは冗長度の制限された小さなネットワークでは非常に役立ちます。しかし、このタイプのプロトコルには、以下のようないくつかの欠点があります。

- ・ 障害状態時に、ネットワーク内の各装置が正確なルーティング・テーブルを生成する時間の長さは、*収束時間*と呼ばれます。距離ベクトル・アルゴリズムを使用する巨大で複雑なインターネットワークでは、この時間は超過する可能性があります。ルーティング・テーブルの収束中は、ネットワークは矛盾したルーティング動作をしやすくなります。このことはルーティング・ループや他のタイプの不安定なパケット転送を引き起こす可能性があります。
- ・ 収束時間を削減するため、単一の経路に含まれるホップの最大数に制限が設けられることがあります。距離ベクトル・ネットワークではこの制限を超える有効パスは使用できません。
- ・ 距離ベクトル・ルーティング・テーブルは、近隣装置に定期的に伝送されます。これは、テーブルの内容が変更されていなくても送信されます。このために、能力が限られた環境ではかなりの期間使用率の高まった状態が続く可能性があります。

収束や不安定性の露出を減らすために、基本の距離ベクトル・アルゴリズムの拡張が開発されてきました。この拡張については、154 ページの 4.3.5『収束と無限カウント』で説明します。

RIP と BGP は、よく知られた距離ベクトル・ルーティング・プロトコルの 2 つの例です。

4.2.3 リンク状態ルーティング

近年のネットワークのサイズと複雑さの増大のため、より堅固なルーティング・アルゴリズムの開発が必要になりました。このアルゴリズムは、距離ベクトル・プロトコルで観察された欠点に対処しています。

このアルゴリズムは *リンク状態* の原理を使用してネットワーク・トポロジーを決定します。リンク状態とは、ルーターのインターフェース（たとえば、IP アドレス、サブネット・マスク、ネットワークのタイプ）と、近隣ルーターとのその関係の説明です。このリンク状態の集合は、リンク状態データベースを構成します。

リンク状態アルゴリズムが、ネットワーク・トポロジーを決定するために使用するプロセスは簡潔です。

- 各ルーターは、直接接続されたネットワーク上の他のすべてのルーティング装置を識別します。
- 各ルーターは、すべての直接接続されたネットワークのリンクと各リンクに関連したコストのリストを公示します。これは、ネットワーク内の他のルーターとのリンク状態公示 (LSA) の交換を通して実行されます。
- 各ルーターはこれらの公示を使用して、現行ネットワーク・トポロジーについての詳細を示すデータベースを作成します。各ルーターのトポロジー・データベースは同一です。
- 各ルーターはトポロジー・データベース内の情報を使用して、各宛先ネットワークへの最も望ましい経路を計算します。この情報が IP ルーティング・テーブルの更新に使用されます。

4.2.3.1 Shortest-Path First (SPF) アルゴリズム

SPF アルゴリズムは、トポロジー・データベース内の情報の処理に使用されます。これは、ネットワークのツリー表記を提供します。SPF アルゴリズムを実行している装置がツリーのルートになります。このアルゴリズムの出力は、各宛先ネットワークへの最短パスのリストになります。図 60 は、ルーター A で実行された最短パス・アルゴリズムの例です。

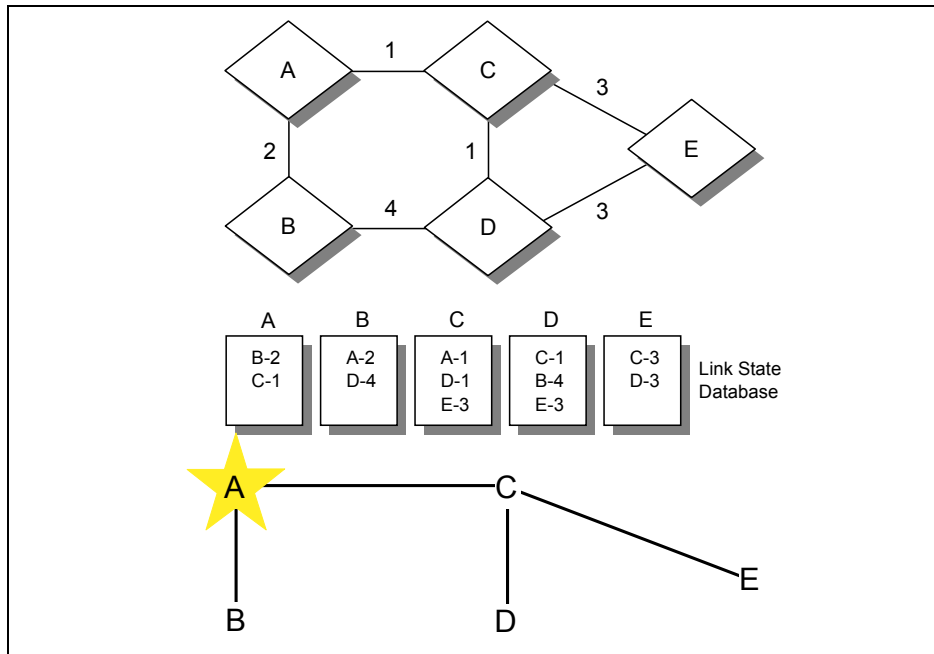


図 60 Shortest-Path First (SPF) の例

各ルーターは同じ LSA のセットを処理しているため、各ルーターは同一のリンク状態データベースを作成しています。ただし、各装置はネットワーク・トポロジ内の異なる場所を使用しているため、SPF アルゴリズムを適用すると各ルーターごとに異なるツリーが生成されます。

OSPF プロトコルは、リンク状態ルーティング・プロトコルのよく知られた例です。

4.2.4 ハイブリッド・ルーティング

ルーティング・プロトコルの最後のカテゴリーは、ハイブリッド・プロトコルです。このプロトコルは、距離ベクトルとリンク状態ルーティング双方の肯定的な属性を結合することを試みたものです。距離ベクトルと同様に、ハイブリッド・プロトコルはメトリックを使用して経路にプリファレンスを割り当てます。ただし、このメトリックは従来の距離ベクトル・プロトコルよりも正確です。リンク状態アルゴリズムと同様に、ハイブリッド・プロトコルのルーティング更新は、定期的ではなくイベント・ドリブンです。ハイブリッド・プロトコルを使用するネットワークは、距離ベクトル・プロトコルを使用するネットワークよりも短時間に収束する傾向があります。さらに、このプロトコルでは、リンク状態の更新と距離ベクトルの公示のオーバーヘッドが削減される可能性があります。

オープンなハイブリッド・プロトコルも存在しますが、このカテゴリーはほとんどすべて所有権付きの EIGRP アルゴリズムに関連したものです。EIGRP は、Cisco Systems, Inc. によって開発されました。

4.3 Routing Information Protocol (RIP)

RIP は、小さな自律システム内で使用するために設計された、内部ゲートウェイ・プロトコルの一例です。RIP は Xerox XNS ルーティング・プロトコルに基づいています。RIP の初期のインプリメンテーションは、コードがバークレー・ソフトウェア・ディストリビューション (BSD) の UNIX ベース・オペレーティング・システムに統合されていたので、簡単に受け入れられました。RIP は距離ベクトル・プロトコルです。

1988 年の中ごろ、IETF は RIP システムの標準操作について説明する RFC 1058 を発行しました。しかし、この RFC が発行されたのは、多数の RIP インプリメンテーションが完成した後でした。このため、一部の RIP システムは、距離ベクトル・アルゴリズムの拡張（たとえば、poison reverse やトリガー更新）をすべてはサポートしていません。

4.3.1 RIP のパケット・タイプ

RIP プロトコルは 2 つのパケット・タイプを規定しています。RIP プロトコルを実行している任意の装置は、以下のパケットを送信できます。

- 要求パケット：要求パケットは近隣 RIP 装置に照会してその距離ベクトル・テーブルを取得します。この要求には、近隣者がテーブルの特定のサブセットを戻すべきか、あるいはテーブルの全内容を戻すべきかが示されます。
- 応答パケット：応答パケットは、装置がそのローカル距離ベクトル・テーブルに含まれている情報を公示するために送信します。テーブルは以下のような状況で送信されます。
 - テーブルは 30 秒ごとに自動的に送信されます。
 - テーブルは他の RIP ノードによって生成された要求パケットに対する応答として送信されます。
 - トリガー更新がサポートされている場合は、ローカル距離ベクトル・テーブルが変更されたときにテーブルが送信されます。トリガー更新については、159 ページの 4.3.5.3 『トリガー更新』で説明します。

装置が応答パケットを受信すると、その更新に含まれる情報がローカル距離ベクトル・テーブルと比較されます。更新に宛先へのよりコストの低い経路が含まれている場合、テーブルはその新しいパスを反映するように更新されます。

4.3.2 RIP のパケット・フォーマット

RIP は既知のネットワーク宛先への距離に関する情報を共有するための特定のパケット・フォーマットを使用します。RIP パケットは、UDP データグラムを使用して伝送されます。RIP は UDP ポート 520 を使用してデータグラムを送受信します。

RIP データグラムの最大サイズは 512 オクテットです。このサイズより大きい更新は、複数のデータグラムに入れて公示する必要があります。LAN 環境では、RIP データグラムは MAC 全端末ブロードキャスト・アドレスと IP ネットワーク・ブロードキャスト・アドレスをして送信されます。Point-to-Point または非ブロードキャスト環境では、データグラムは限定的に宛先装置へアドレッシングされます。

図 61 は、RIP パケット・フォーマットを表しています。

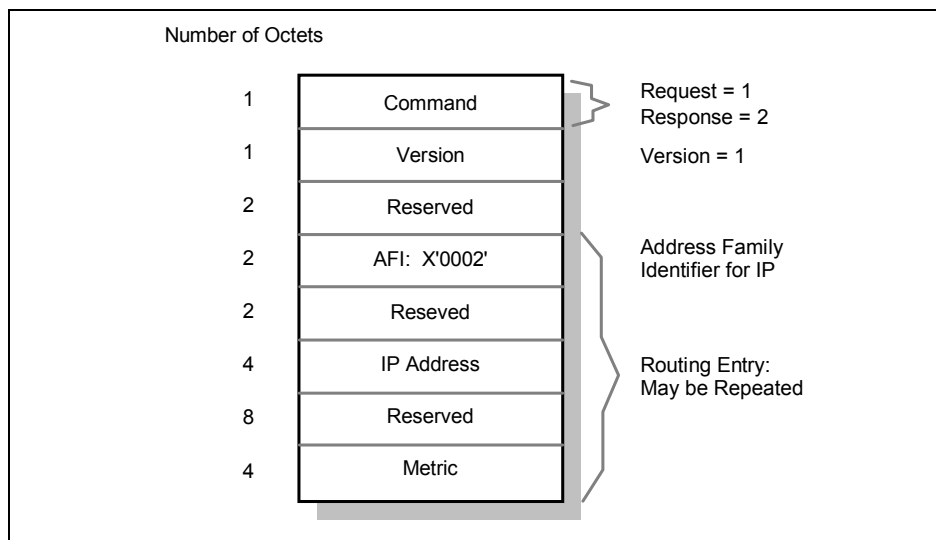


図 61 RIP パケット・フォーマット

512 バイトのパケット・サイズでは、単一の RIP 公示に最大で 25 のルーティング・エントリーを含めることができます。

4.3.3 RIP の操作モード

RIP ホストは、以下の 2 つの操作モードを持ちます。

- ・アクティブ・モード: アクティブ・モードで操作している装置は、その距離ベクトル・テーブルを公示し、また近隣 RIP ホストからルーティング更

新を受信します。ルーティング装置は普通アクティブ・モードで操作するよう構成されます。

- パッシブ (またはサイレント) モード: このモードで操作している装置は、近隣 RIP 装置からのルーティング更新の受信だけを行います。これはその距離ベクトル・テーブルの公示は行いません。終端端末は普通パッシブ・モードで操作するように構成されます。

4.3.4 距離ベクトルの計算

距離ベクトル・テーブルには、各宛先ネットワークの説明があります。このテーブル内のエントリーには以下の情報が含まれています。

- テーブル内のこのエントリーが説明する宛先ネットワーク (ベクトル)。
- この宛先に到達するための最も望ましいパスに関連したコスト (距離)。これは宛先への複数のパスを区別する機能を備えています。この文脈での距離およびコストという語は誤解されやすいかもしれません。これらは、物理的な距離や金銭コストとは直接関係ありません。
- 宛先ネットワークに到達するために使用されるネクスト・ホップ装置の IP アドレス。

装置はルーティング・テーブル公示を受信するたびに、それを処理してもっとコストの低いパスを介して到達できる宛先がないかを判別します。これは RIP 距離ベクトル・アルゴリズムを使用して行われます。このアルゴリズムは以下のように要約することができます。

- ルーターの初期設定で、各装置には直接接続される各ネットワークとその構成済みコストをリストする距離ベクトル・テーブルが含まれます。基本的に、各ネットワークにはコスト 1 が割り当てられます。これはネットワーク上の単一のホップを表します。経路内のホップの合計数は、その経路の合計コストと等しくなります。ただし、コストは使用率、速度、または信頼性などの他の測量を表すように変更することもできます。
- 各ルーターは定期的 (基本的に 30 秒ごと) にその距離ベクトル・テーブルをその近隣の各装置へ伝送します。ルーターはトポロジが変更したときにテーブルを伝送することもあります。
- 各ルーターはこの情報を使用してそのローカル距離ベクトル・テーブルを更新します。
 - 各宛先への合計コストは、近隣者の距離ベクトル・テーブル内で報告されているコストをその近隣者へのリンクのコストに加算することによって計算されます。最もコストの低いパスが距離ベクトル・テーブルに保管されます。

- すべての更新は距離ベクトル・テーブル内の以前の情報を自動的に取り替えます。これによって、RIP はルーティング・テーブル内の経路の整合性を維持することができます。
- IP ルーティング・テーブルが各宛先への最低コスト・パスを反映するように更新されます。

図 62 は、単純なインターネットワーク内の 3 つのルーターの距離ベクトル・テーブルを表しています。

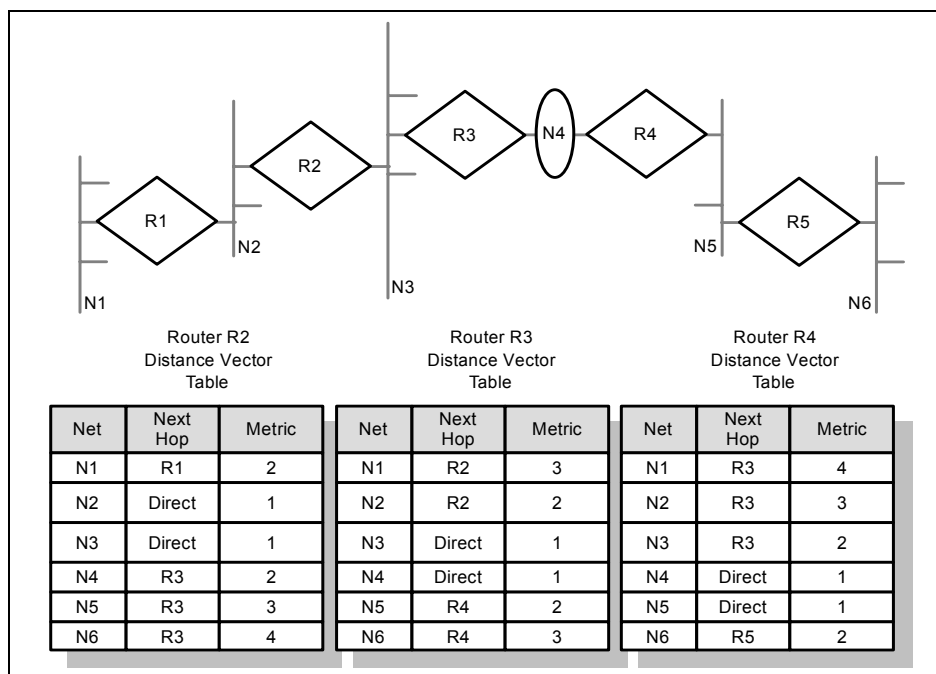


図 62 サンプル距離ベクトル・ルーティング・テーブル

4.3.5 収束と無限カウント

十分な時間があれば、このアルゴリズムは各装置の距離ベクトル・テーブルを正確に計算します。しかし、この収束時間中に誤った経路がネットワークに伝搬することがあります。この問題は図 63 に示しています。

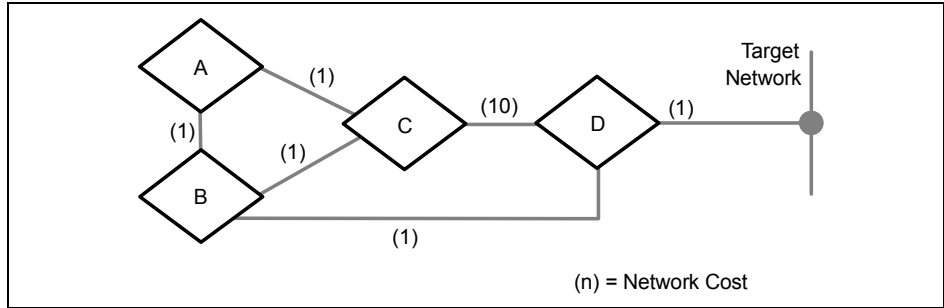


図 63 無限カウントのサンプル・ネットワーク

このネットワークには、相互接続された 4 つのルーターが含まれています。ルーター C およびルーター D に接続するリンク（このコストは 10 です）を除き、各リンクのコストは 1 です。コストがこのような定義されているので、ルーター C およびルーター D に接続するリンクでのパケットの転送は望ましくありません。

ネットワークが収束すると、各装置はすべてのネットワークを説明するルーティング情報を持ちます。たとえば、宛先ネットワークに到達するため、ルーターは以下の情報を持ちます。

- 宛先ネットワークへのルーター D: 直接接続されたネットワーク。メトリック 1。
- 宛先ネットワークへのルーター B: ネクスト・ホップはルーター D。メトリックは 2。
- 宛先ネットワークへのルーター C: ネクスト・ホップはルーター B。メトリックは 3。
- 宛先ネットワークへのルーター A: ネクスト・ホップはルーター B。メトリックは 3。

ルーター B とルーター D を接続するリンクが故障した障害状態について考えてください。ネットワークが収束すると、すべての経路が宛先ネットワークに到達するためにルーター C とルーター D を接続するリンクを使用します。しかし、この収束時間はかなりのものになる可能性があります。図 64 は、収束期間中に宛先ネットワークへの経路がどのように更新されているかを示しています。分かりやすくするため、この図では、すべてのルーターが同時に更新を送信することを想定しています。

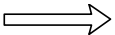
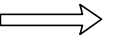
Time												
D: Direct	1	Direct	1	Direct	1	Direct	1		Direct	1	Direct	1
B: Unreachable	C	4	C	5	C	6			C	11	C	12
C: B	3	A	4	A	5	A	6		A	11	D	11
A: B	3	C	4	C	5	C	6		C	11	C	12

図 64 ネットワーク収束順序

再収束は、B がルーター D への経路が使用不能であることに気付いた時点で始まります。リンクはタイムアウトになっているので、ルーター B は即時に障害経路を取り除くことができます。しかし、他のルーターがこの障害経路に対するその参照を取り除くまでには、かなりの時間が経過します。このことは、図 64 に表されている更新の順序で説明されています。

1. 障害状態が発生する前に、ルーター A およびルーター C はルーター B を介する宛先ネットワークへの経路を持っています。
2. ルーター D とルーター B を接続するリンクが故障したときに、障害状態が発生します。ルーター B は、宛先ネットワークへのその優先パスが無効であることを認識します。
3. ルーター A とルーター C は、引き続きルーター B を介する経路を反映する更新を送信します。ルーター D とルーター B を接続するリンクが故障しているため、この経路は実際には無効です。
4. ルーター B は、ルーター A とルーター C から更新を受け取ります。ルーター B は、今やルーター A またはルーター C のいずれかを介して、宛先ネットワークにトラフィックをルーティングしなければならないと思い込みます。実際にはこれは有効な経路ではありません。なぜなら、ルーター A とルーター C の中の経路はルーター B を経由した以前の経路の名残だからです。
5. ルーター B から送信されたルーティング公示を使用して、ルーター A とルーター C は、ルーター B を経由する経路が故障していることを判別することができます。ただし、ルーター A とルーター C は、この時お互いを經由する優先経路が存在すると思い込みます。

ルーター A とルーター C が長期間相互の欺き合いを行っている間、ネットワーク収束は続行します。各装置は、相手装置を經由して宛先ネットワークに到達できると主張します。宛先ネットワークに到達するためのパスには、ルーティング・ループが含まれるようになります。

距離ベクトル・テーブル内のコストが増分する方法は、*無限カウント*と呼ばれます。コストは理論上無限に増分し続けます。この影響を最小化するため、いつでもネットワークが使用不能になったら、ルーティング更新を通してメトリックが増分するのをできるだけ早く止める必要があります。RIP 環境では、コストは最大値の 16 に達するまで増分し続けます。この限度は RFC で定義されています。

このメトリック限度の副次作用として、パケットがトラバースできる、送信元ネットワークから宛先ネットワークまでのホップの数も制限されます。RIP 環境では、15 ホップを超えるパスはどれも無効と見なされます。ルーティング・アルゴリズムはこのパスを破棄します。

無限カウント問題を最小化することのできる、基本距離ベクトル・アルゴリズムに対する以下のような 2 つの拡張があります。

- split horizon with poison reverse
- トリガー更新

これらの拡張は最大メトリック限度には影響しません。

4.3.5.1 Split horizon

無限カウントによって引き起こされる過剰な収束時間は、Split horizon の使用によって削減することができます。この規則により、情報の受信に使用されたインターフェースについてのルーティング情報がルーターから流出することが防止されます。

基本 split horizon 規則は RFC 1058 ではサポートされていません。その代わり、この標準は split horizon with poison reverse アルゴリズムを規定しています。背景を知って完全を期すため、ここでは基本規則について説明します。拡張アルゴリズムについては、次のセクションで検討します。

split horizon を導入すると、図 64 のルーティング更新順序は変更されます。新しい順序は図 65 に示されています。この表は、split horizon 規則を使用すると収束はかなり速く行われることを示しています。

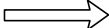
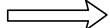
Time							
D: Direct	1	Direct	1	Direct	1	Direct	1
B: Unreachable		Unreachable		Unreachable		C	12
C: B	3	A	4	D	11	D	11
A: B	3	C	4	Unreachable		C	12
Note: Faster Routing Table Convergence							

図 65 split horizon を使用したネットワーク収束

この規則の制約事項は、各ノードが到達不能宛先への経路を距離ベクトル・テーブルから除去する前に、その経路がタイムアウトになるのを待たなければならないことです。RIP 環境では、このタイムアウトは最低でも初期障害から 3 分後です。この時間の間、この装置は到達不能宛先に関する誤った情報を他のノードに提供し続けます。これはルーティング・ループと他のルーティング障害を伝搬します。

4.3.5.2 split horizon with poison reverse

poison reverse は、標準の split horizon インプリメンテーションに対する拡張です。これは RFC 1058 でサポートされています。poison reverse では、各ルーティング更新ごとに既知のすべてのネットワークが公示されます。しかし、特定のインターフェースを通して認識されるネットワークは、そのインターフェースに送信されるルーティング通知で到達不能として公示されません。

これは、複雑な冗長度の高い環境での収束時間を大幅に改良します。poison reverse では、ネットワークが到達不能であることをルーティング更新が示すと、経路はそのルーティング・テーブルから即時に除去されます。これは、誤ったループ経路がネットワークに伝搬する前に、それを切断します。このアプローチは、経路がタイムアウトを通して除去される基本の split horizon とは異なります。

poison reverse は、冗長度のないネットワーク（単一パス・ネットワーク）では無益です。

poison reverse の欠点の 1 つは、近隣者間で交換されるルーティング通知のサイズが著しく増大する可能性があることです。これは、各通知に距離ベクトル・テーブル内のすべての経路が含まれるためです。通常これはローカル・エリア・ネットワークでは問題になりませんが、容量の小さい WAN 接続ではこのために使用率の高い状態が続く可能性があります。

4.3.5.3 トリガー更新

split horizon with poison reverse と同様、トリガー更新をインプリメントするアルゴリズムは、ネットワーク収束時間を削減するために設計されています。トリガー更新では、ルーターは経路のコストを変更するたびに、変更された距離ベクトル・テーブルを即時に近隣装置へ送信します。このメカニズムは、正規の定期的な間隔におけるよりも早急にトポロジー変更通知が伝搬されるようにします。

トリガー更新は RFC 1058 でサポートされています。

4.3.6 RIP の制約事項

RIP 環境では、以下のようないくつかの制約事項が観察されています。

- パス・コスト限度：無限カウント問題の解決のために、ネットワーク・パスの最大コストが設けられています。これは最大ネットワーク直径に上限を置いています。15 ホップより大きなパスが必要なネットワークは、代替のルーティング・プロトコルを使用する必要があります。
- ネットワーク集中のテーブル更新：距離ベクトル・テーブルの定期的なブロードキャストのために、ネットワーク・リソースの使用率が増加する可能性があります。これは容量の限定されたセグメントで問題になる可能性があります。
- 比較的遅い収束：他の距離ベクトルと同様に RIP は収束が比較的低速です。アルゴリズムはタイマーに頼ってルーティング・テーブルの公示を開始します。
- 可変長サブネット・マスキングがサポートされていない：RIP 環境での経路公示にはサブネット・マスキング情報は含まれません。そのため、RIP ネットワークは可変長サブネット・マスクを配備することができません。

4.4 Routing Information Protocol バージョン 2 (RIP-2)

IETF は RIP の 2 つのバージョンを認識しています。

- RIP バージョン 1 (RIP-1): このプロトコルは RFC 1058 で説明されています。
- RIP バージョン 2 (RIP-2): RIP-2 も AS 内で使用するために設計された距離ベクトル・プロトコルです。これは、RIP-1 に観察された制約事項を解消するために開発されました。RIP-2 は RFC 1723 で説明されています。この標準は 1994 年後半に公開されました。

実際には RIP という語は RIP-1 を指します。TCP/IP 資料内で RIP という語が出てきたときはいつでも、特に注記がない限り RIP バージョン 1 を指していると思えるのが無難です。この資料でも同じ慣例を用いています。ただし、

2つのバージョンを比較する場合は、混乱を避けるために RIP-1 という語を使用しています。

RIP-2 は RIP-1 と似ています。これは小さなネットワークでの RIP-1 の機能を拡張するために開発されました。RIP-2 には、RIP-1 にはなかったさらに以下のような利点があります。

- CIDR および VLSM のサポート：RIP-2 はサブネット化（つまり、CIDR）および可変長サブネット・マスクをサポートしています。この新しい標準が開発された主な理由は、これをサポートすることにあります。この拡張により、この標準は RIP-1 ではサポートされていなかったある程度複雑なアドレッシングにも順応しています。
- マルチキャストのサポート：RIP-2 はルーティング通知の単純なブロードキャストよりもマルチキャストをサポートしています。これにより、RIP-2 メッセージを受信しないホストの処理ロードが削減されます。RIP-1 環境とのインターオペラビリティを保証するため、このオプションは各ネットワーク・インターフェースごとに構成されます。
- 認証のサポート：RIP-2 は経路公示を伝送する任意のノードの認証をサポートしています。これにより、不正な送信元によるルーティング・テーブルの破壊が防止されます。
- RIP-1 のサポート：RIP-2 は RIP-1 との完全な共通操作性を持ちます。そのため、この 2 つの標準の間には後方互換性があります。

RIP-1 のセクションでも指摘したとおり、RIP-1 標準の目立った欠陥の 1 つはメトリック・フィールドのインプリメンテーションです。RIP-1 はメトリックを 0 と 16 の間の値として規定しています。RIP-1 ネットワークとの互換性を保証するため、RIP-2 はこの定義を保持しています。どちらの標準においても、ホップ・カウントが 15 より大きいネットワーク・パスは到達不能と解釈されます。

4.4.1 RIP-2 のパケット・フォーマット

元の RIP-1 仕様は、将来の拡張をサポートするよう設計されていました。RIP-2 標準はこのフィーチャーを利用することができました。RIP-2 の開発者は、RIP-1 パケットにすでにバージョン・フィールドが含まれていることと、50 % のオクテットが未使用であることに注目しました。

図 66 は RIP-2 パケットの内容を表しています。表示されているパケットには認証情報が含まれています。更新の中の最初のエントリーには、ルーティング・エントリーか認証エントリーのいずれかが含まれます。最初のエントリーが認証エントリーである場合、メッセージにはさらに 24 のルーティング・エントリーを含めることができます。認証情報がない場合は、25 のルーティング・エントリーを含めることができます。

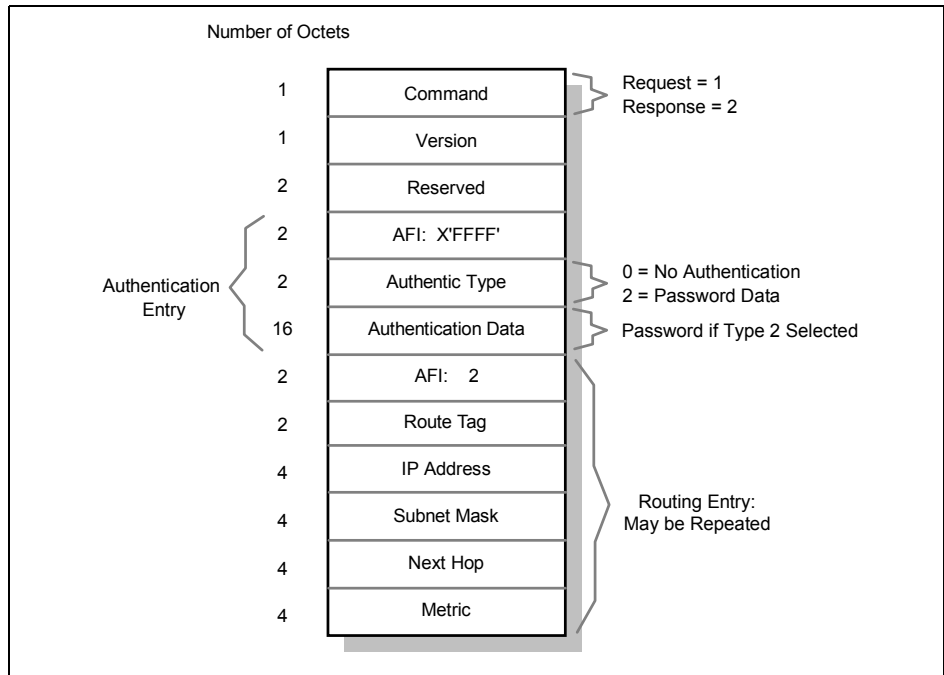


図 66 RIP-2 のパケット・フォーマット

RIP-2 メッセージのコマンド・フィールド、IP アドレス・フィールド、およびメトリック・フィールドの使用法は RIP-1 メッセージでの使用法と同じです。これ以外で、RIP-2 パケットにインプリメントされた変更点は以下のとおりです。

- **バージョン:** このフィールドに含まれる値は 2 です。これは RIP-1 ルーターに、以前使用されていなかったフィールドに含まれる値はどれも無視するように指示します。
- **AFI (アドレス・ファミリー):** 値 x'0002' は、ネットワーク・アドレス・フィールドに含まれるアドレスが IP アドレスであることを示します。値 x'FFFF' は、認証エントリーを示します。
- **認証タイプ:** このフィールドは、認証エントリーの残りの 16 バイトを定義します。値 0 は、認証なしを示します。値 2 は、認証データ・フィールドにパスワード・データが含まれることを示します。
- **認証データ:** このフィールドには 16 バイトのパスワードが含まれます。
- **経路タグ:** このフィールドは、内部経路と外部経路を区別するためのものです。内部経路は同じネットワークまたは AS 内で RIP-2 を介して認識されます。

- ・サブネット・マスク：このフィールドには参照されているネットワークのサブネット・マスクが含まれます。
- ・ネクスト・ホップ：このフィールドには、ルーターが参照されているネットワークにデータグラムを送信するときに使用すべきネクスト・ホップに関する勧告が含まれます。

4.4.2 RIP-2 の制約事項

RIP-2 は、RIP-1 で観察された多くの制約事項を解消するために開発されました。ただし、RIP-1 ネットワークに存在していたパス・コスト限度と遅い収束は、RIP-2 環境でも同じく問題となります。

これらの問題の他に、RIP-2 認証プロセスの制約事項が存在します。RIP-2 標準は認証パスワードを暗号化しません。これはクリア・テキストで伝送されます。このため、ネットワークは、その環境への直接物理アクセスを持つ人物からの攻撃を受けやすくなっています。

4.5 IPv6 用 RIPng

RIPng は、IPv6 ベース・ネットワーク内のルーターが経路の計算に使用された情報を交換できるようにするために開発されました。これは RFC 2080 に文書化されています。579 ページの第 17 章『IP Version 6』には、IPv6 に関する追加情報があります。

RIP ファミリーの他のプロトコルと同様、RIPng は小さな自立走行式システム内で使用するように設計された距離ベクトル・プロトコルです。RIPng は RIP-2 で使用されているのと同じアルゴリズム、タイマー、および論理を使用します。

RIPng には、多くの他の距離ベクトル・プロトコルに存在するのと同じ制限があります。パス・コストの制限と収束時間は、RIPng ネットワークでもやはり問題となります。

4.5.1 RIPng と RIP-2 の違い

RIP-2 と RIPng の間には以下の 2 つの重要な相違があります。

- ・認証のサポート：RIP-2 標準には、経路情報を伝送するノードの認証のサポートが含まれます。RIPng には、ネイティブ認証サポートは含まれません。むしろ、RIPng は IPv6 に備わっているセキュリティー・フィーチャーを使用します。このセキュリティー・フィーチャーは、認証のほか、各 RIPng パケットを暗号化する機能を提供しています。これにより、ルーティング情報を受信する一連の装置を制御することが可能です。

IPv6 セキュリティー・フィーチャーを使用した結果の 1 つとして、RIPng パケットでは AFI フィールドが除去されています。公示の中の認証エントリーとルーティング・エントリーを区別する必要はもはやありません。

- IPv6 アドレス・フォーマットのサポート：RIPng パケットに含まれるフィールドは、もっと長い IPv6 アドレス・フォーマットをサポートするように更新されています。

4.5.2 RIPng のパケット・フォーマット

RIPng パケットは、UDP データグラムを使用して伝送されます。RIPng は UDP ポート番号 521 を使用してデータグラムを送受信します。

RIPng パケットのフォーマットは RIP-2 のフォーマットと類似しています。具体的には、どちらのパケットにも 4 オクテットのコマンド・ヘッダーとそれに続く一連の 20 オクテットの経路エントリーが含まれます。図 67 は、RIPng パケット・フォーマットを表しています。

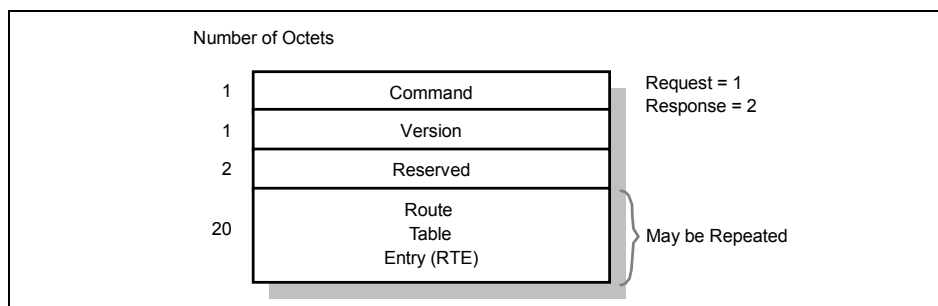


図 67 RIPng のパケット・フォーマット

コマンド・フィールドとバージョン・フィールドの使用法は、RIP-2 パケットでの使用法と同じです。しかし、ルーティング情報を含むフィールドは、16 オクテットの IPv6 アドレスに対応するように更新されています。これらのフィールドは、RIP-1 または RIP-2 パケット内の対応するフィールドとは違った使われ方をします。図 68 は、経路テーブル・エントリー (RTE) のフォーマットを表しています。

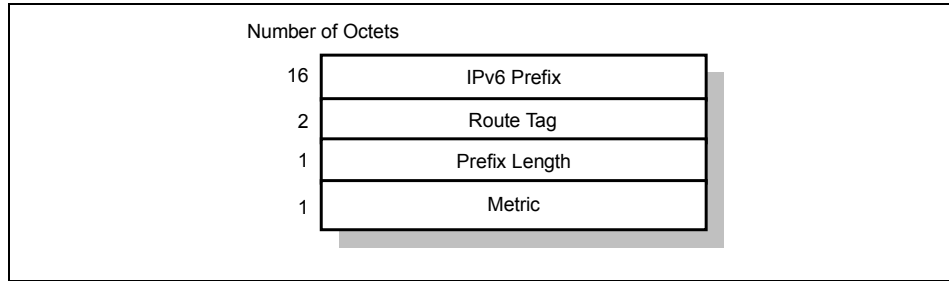


図 68 経路テーブル・エントリー (RTE)

RIPng では、IP 接頭部と接頭部の長さの組み合わせが、公示されている経路を示します。メトリックは 1 オクテット・フィールドでやはりエンコードされます。RIPng は最大で 16 のホップ・カウントを使用するので、この長さは十分です。

RIPng と RIP-2 の間のもう 1 つの相違は、ネクスト・ホップの決定に使用されるプロセスです。RIP-2 では、各経路テーブル・エントリーにネクスト・ホップ・フィールドが含まれます。RIPng では、各 RTE にこの情報を含めると、公示のサイズが 2 倍になる可能性があります。したがって、RIPng ではネクスト・ホップは特殊なタイプの RTE に含められます。指定されたネクスト・ホップは、公示の中の後続の各ルーティング・テーブル・エントリーに適用されます。ネクスト・ホップの指定に使用される RTE のフォーマットは、図 69 に示されています。

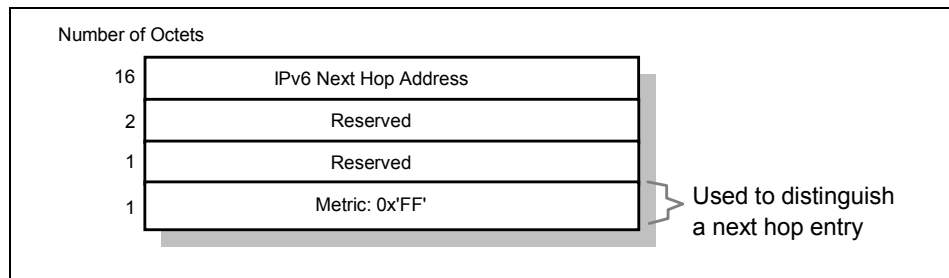


図 69 ネクスト・ホップ経路テーブル・エントリー (RTE)

ネクスト・ホップ RTE は、メトリック・フィールド内の値 0xFF' で識別されます。この予約済み値はメトリックの有効範囲外になります。

RTE とネクスト・ホップ RTE の使用法は、図 70 に示されています。

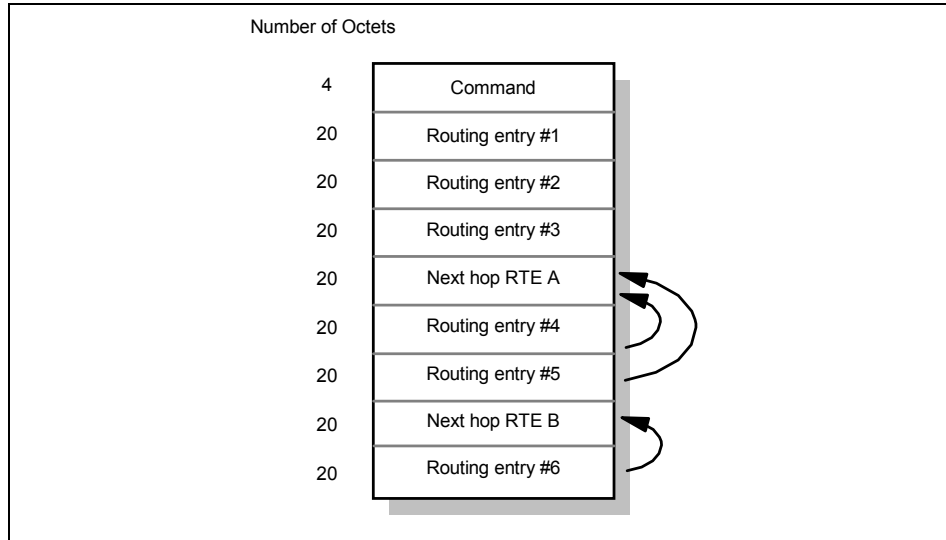


図 70 RIPng RTE の使用法

この例で、最初の 3 つのルーティング・エントリには対応するネクスト・ホップ RTE がありません。これらのエントリで指定されたアドレス接頭部は、公示ルーターを通してルーティングされます。ルーティング・エントリ 4 および 5 に含まれた接頭部は、ネクスト・ホップ RTE A で指定されたネクスト・ホップ・アドレスを経てルーティングします。ルーティング・エントリ 6 に含まれた接頭部は、ネクスト・ホップ RTE B で指定されたネクスト・ホップ・アドレスを経てルーティングします。

4.6 Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) プロトコルは、内部ゲートウェイ・プロトコルのもう 1 つの例です。これは、RIP の制限を解消するためにルーティング代替として著作権なしで開発されました。初期開発は 1988 年に始まり 1991 年に完了しました。この後もこのプロトコルの更新が公開され続けています。この標準の現行バージョンは、RFC 2328 に文書化されています。

OSPF は距離ベクトル・プロトコルにはないいくつかのフィーチャーを備えています。これらのフィーチャーのサポートにより、OSPF は大規模なネットワーク環境において広く配備されたルーティング・プロトコルになりました。実際、RFC 1812 – Requirements for IPv4 Routers では、OSPF が唯一の必須動的ルーティング・プロトコルとしてリストされています。以下のフィーチャーは、OSPF 標準の引き継ぎの支持に貢献しています。

- 等しいコストのロード・バランシング：複数のパスの同時使用により、ネットワーク・リソースの使用効率が良くなります。
- ネットワークの論理区画：これにより、障害時の障害情報の伝搬が軽減されます。さらにこれはルーティング通知の集約を可能にし、それによって不必要なサブネット情報の公示は制限されます。
- 認証のサポート：OSPF は経路公示を送送する任意のノードの認証をサポートしています。これにより、不正な送信元によるルーティング・テーブルの破壊が防止されます。
- より高速な収束時間：OSPF は、ルーティング変更の瞬間伝搬を提供しています。これにより、ネットワーク・トポロジーの更新に必要な収束時間は短縮されています。
- CIDR および VLSM のサポート：これにより、ネットワーク管理者は IP アドレス・リソースを効率的に割り当てることができます。

OSPF はリンク状態プロトコルです。他のリンク状態プロトコルと同様に、各 OSPF ルーターは SPF アルゴリズム (149 ページの 4.2.3.1 『Shortest-Path First (SPF) アルゴリズム』を参照) を実行して、リンク状態データベースに保管されている情報を処理します。このアルゴリズムは、各宛先ネットワークへの優先経路を詳細に示す最短パス・ツリーを生成します。

4.6.1 OSPF 用語

OSPF はプロトコルの操作の説明に特有の用語を使用しています。

4.6.1.1 OSPF エリア

OSPF ネットワークは、*エリア*の集合に分割されています。エリアはネットワークとルーターの論理グループから成ります。このエリアは地理または管理の境界と一致します。各エリアには 32 ビットの *エリア ID* が割り当てられます。

ネットワークを再分割することには、以下のような利点があります。

- エリア内では、すべてのルーターがそのエリア内のルーティング装置とリンクについて説明する同一のトポロジー・データベースを持ちます。これらのルーターはエリア外のトポロジーの知識は持ちません。これらは、これらの外部宛先への経路だけを認識します。このため、各ルーターによって保守されるトポロジー・データベースのサイズは小さくなります。
- エリアにより、リンク状態更新の数の爆発的な増加の可能性が制限されます。大半の LSA はエリア内にのみ分散されます。
- エリアにより、トポロジー・データベースの保守に必要な CPU 処理が少なくなります。SPF アルゴリズムは、エリア内の変更の管理に限定されます。

バックボーン・エリアおよびエリア 0

すべての OSPF ネットワークには、少なくとも 1 つのエリアが含まれます。このエリアは、エリア 0 またはバックボーン・エリアと呼ばれます。追加のエリアは、ネットワーク・トポロジーかその他の設計要件に基づいて作成できます。

複数のエリアを含むネットワークでは、バックボーンが他のすべてのエリアに物理的に接続します。OSPF では、すべてのエリアがルーティング情報をバックボーンに直接通知することになります。この情報はその後バックボーンが他のエリアに通知します。

図 71 は、バックボーン・エリアと 4 つの追加エリアを持つネットワークを表しています。

4.6.1.2 エリア内、エリア・ボーダー、および AS 境界ルーター

OSPF ネットワークのルーターには 3 つの種類があります。図 71 はこれらの装置の相互作用を表しています。

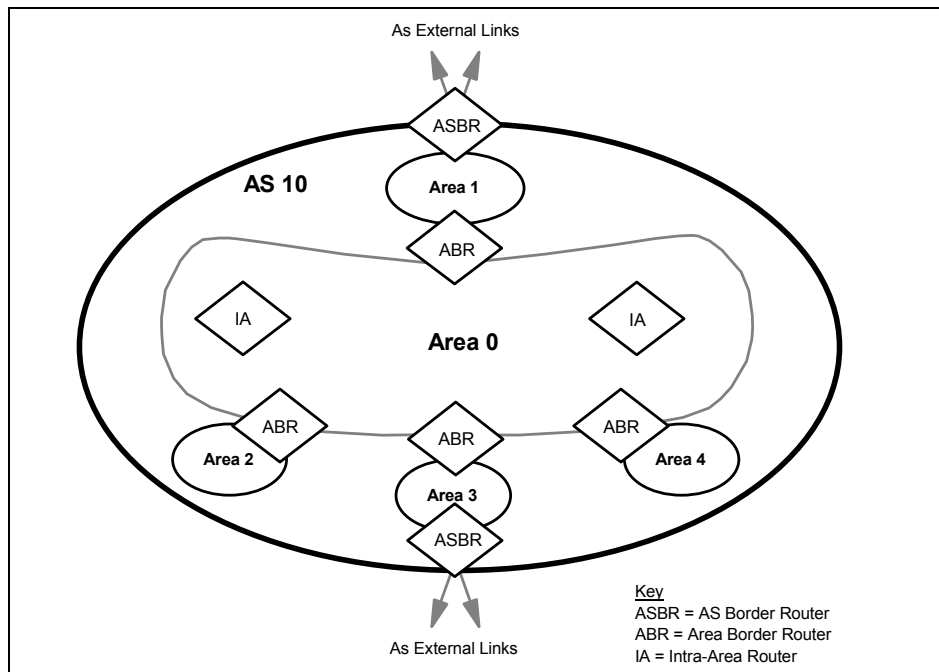


図 71 OSPF のルーターのタイプ

- 内部エリア・ルーター：このクラスのルーターは論理的に完全に OSPF エリア内部に置かれます。内部エリア・ルーターはそのローカル・エリアのトポロジー・データベースを保守します。
- エリア・ボーダー・ルーター (ABR): このクラスのルーターは論理的に複数のエリアに接続します。1 つのエリアはバックボーン・エリアでなければなりません。ABR はエリアの相互接続に使用されます。これは、接続されている各エリアごとに別個のトポロジー・データベースを保守します。ABR はまた、各エリアごとに SPF アルゴリズムの別個のインスタンスを実行します。
- AS 境界ルーター (ASBR): このクラスのルーターは、OSPF インターネットワークの周辺に置かれます。これは、OSPF ネットワークと他のルーティング環境の間で到達可能性を交換するゲートウェイとして機能します。ASBR は、AS への AS 外部リンク公示の通知を担当します。外部リンク公示については、178 ページの 4.6.6 『OSPF の経路再配布』でさらに詳しく説明しています。

各ルーターには 32 ビットのルーター ID (RID) が割り当てられます。RID は装置を一意的に識別します。よく知られたあるインプリメンテーションでは、ルーターに構成された最も番号の小さい IP アドレスから RID が割り当てられます。

4.6.1.3 物理ネットワークのタイプ

OSPF はネットワーク・セグメントを 3 つのタイプに分類しています。これらのネットワークに接続された OSPF 装置間で発生する通信の頻度とタイプは、そのネットワークのタイプによって左右されます。これには以下のようなものがあります。

- Point-to-Point: Point-to-Point ネットワークは 2 つのルーターを直接リンクします。
- マルチアクセス：マルチアクセス・ネットワークは複数のルーターの接続をサポートします。これはさらに細かく以下の 2 のタイプに分かれています。
 - ブロードキャスト・ネットワークには、接続されているすべてのルーターにパケットを同時に送信する機能があります。この機能は、すべての装置に認識されるアドレスを使用します。イーサネットおよびトークンリング LAN は、OSPF ブロードキャスト・マルチアクセス・ネットワークの例です。
 - 非ブロードキャスト・ネットワークには、ブロードキャスト機能はありません。各パケットはネットワーク内のすべてのルーターに明確にアドレッシングされる必要があります。X.25 およびフレーム・リレー・

ネットワークは、OSPF 非ブロードキャスト・マルチアクセス・ネットワークの例です。

- Point-to-Multipoint: Point-to-Multipoint ネットワークは、マルチアクセスの非ブロードキャスト・ネットワークの特殊な例です。Point-to-Multipoint ネットワークでは、装置が他のすべての装置それぞれに直接接続する必要があります。これは部分メッシュ環境と呼ばれます。

4.6.1.4 近隣ルーターと隣接関係

共通ネットワーク・セグメントを共有する複数のルーターは、そのセグメント上に近隣関係を確立しています。複数のルーターが近隣の関係になるには、以下の点を満たしている必要があります。

- エリア ID：それらのルーターが同じ OSPF エリアに属している必要があります。
- 認証：認証が定義されている場合は、それらのルーターが同じパスワードを指定している必要があります。
- Hello と非活動の間隔：ルーターは Hello プロトコルで使用されているのと同じタイマー間隔を指定していなければなりません。このプロトコルについては、173 ページの 4.6.2『OSPF のパケット・タイプ』で詳細に説明します。
- スタブ・エリア・フラグ：ルーターはエリアがスタブ・エリアとして構成されることに同意している必要があります。スタブ・エリアについては、180 ページの 4.6.7『OSPF のスタブ・エリア』で詳細に説明します。

2 つのルーターが近隣の関係になったら、その装置の間に隣接関係を確立することができます。近隣ルーターがそのトポロジー・データベースと同期している場合、それらは隣接していると見なされます。これはリンク状態情報の交換を通して行われます。

4.6.1.5 指定ルーターとバックアップ指定ルーター

近隣者間でリンク状態情報を交換すると、かなりの量のネットワーク・トラフィックが発生する可能性があります。データベースを同期してリンク状態情報を公示するのに必要な合計帯域幅を縮小するためにも、ルーターは必ずしもすべての近隣装置と隣接関係を確立するわけではありません。

- マルチアクセス・ネットワーク：隣接関係は個々のルーターと（バックアップ）指定ルーターの間に形成されます。
- Point-to-Point ネットワーク：隣接関係は両方の装置の間に形成されます。

各マルチアクセス・ネットワークは、指定ルーター (DR) とバックアップ指定ルーター (BDR) を選出します。DR はネットワーク・セグメント上で以下の 2 つのキー機能を実行します。

- マルチアクセス・ネットワーク上のすべてのルーターと隣接関係を築きます。このとき DR は LSA を転送するためのフォーカル・ポイントになります。
- マルチアクセス・ネットワークに接続された各ルーターをリストするネットワーク・リンク公示を生成します。171 ページの 4.6.1.7 『リンク状態公示とフラッドイング (はんらん)』には、ネットワーク・リンク公示に関する追加情報があります。

BDR は指定ルーターと同じ隣接関係を築きます。DR が故障している場合は、これが DR の機能を引き継ぎます。

各ルーターには、これが DR または BDR として選択される可能性を示す 8 ビットの優先度が割り当てられます。ルーター優先度 0 は、そのルーターが選択に適さないことを示します。この優先度はルーターの各インターフェースごとに構成されます。

図 72 は、近隣者間の関係を表しています。DR または BDR として選択されていないルーターの間には、隣接関係は築かれませんが、

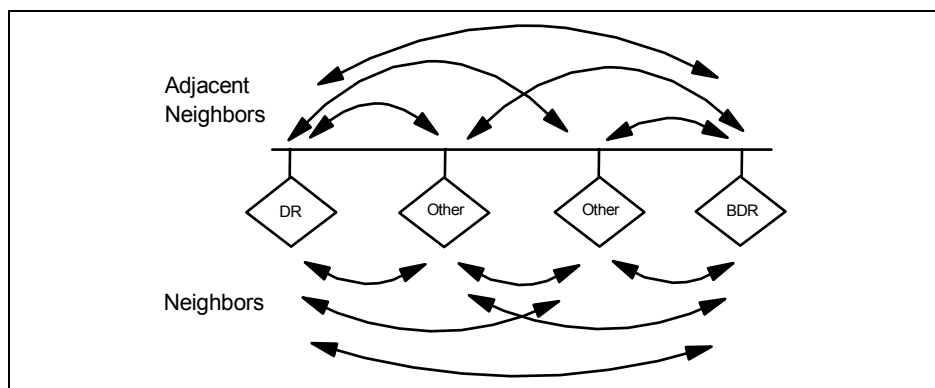


図 72 隣接関係と近隣者間の関係

4.6.1.6 リンク状態データベース

リンク状態データベースはトポロジー・データベースとも呼ばれます。これには OSPF ネットワークと任意の外部接続について説明するリンク状態公示のセットが含まれます。エリア内の各ルーターは、同一のリンク状態データベースのコピーを保守します。

注：

RFC 2328 はトポロジー・データベースを指してリンク状態データベースという語を使用しています。後者はデータベースの内容を説明するのに都合がよく、前者はデータベースの目的をよく表しています。この理由で、本書ではこれまでトポロジー・データベースという語を使用していました。しかし、この OSPF セクションでは今後これをリンク状態データベースと呼びます。

4.6.1.7 リンク状態公示とフラッドイング (はんらん)

LSA の内容は個々のネットワーク・コンポーネント (ルーター、セグメント、または外部宛先) について説明しています。LSA は隣接する OSPF ルーターの間で交換されます。これは各装置上のリンク状態データベースを同期するために行われます。

ルーターが LSA を生成または変更するときは、その変更をネットワークに伝達する必要があります。ルーターは各隣接装置にその LSA を転送することによって、このプロセスを開始します。LSA を受信すると、これらの近隣者はそのリンク状態データベースに情報を保管し、さらにその近隣者にこの LSA を伝達します。この蓄積交換活動は、すべての装置が更新を受け取るまで続きます。このプロセスは信頼性のあるフラッドイングと呼ばれます。このフラッドイングによって、過剰な量の LSA トラフィックでネットワークに過度の負担をかけずに、変更が効率的に伝送されるようにするために、以下の 2 つのステップが取られます。

- 各ルーターは、情報をその近隣装置へ伝搬するまでの期間 LSA を保管します。この時間内に LSA の新しいコピーが到着した場合、ルーターは保管されているバージョンを置き換えます。しかし、新しいコピーが古いものであれば、それは破棄されます。
- 信頼性を保証するため、各リンク状態公示の受信が通知される必要があります。複数の確認通知をグループにして単一の確認通知パケットに入れることができます。確認通知が受信されない場合、元のリンク状態更新パケットが再送されます。

リンク状態公示には、以下のような 5 つのタイプの情報が含まれます。これらの情報を合わせると、OSPF ネットワーク全体と任意の外部環境について説明するために求められる必須情報になります。

- ルーター LSA: このタイプの公示は、エリア内のルーターのインターフェース (リンク) の状態を説明します。これはすべての OSPF ルーターによって生成されます。この公示はエリアにフラッドイングされます。

- ネットワーク LSA: このタイプの公示には、マルチアクセス・ネットワークに接続されたルーターがリストされます。これはマルチアクセス・セグメント上の DR によって生成されます。この公示はエリアにフラッディングされます。
 - 要約 LSA (タイプ 3 およびタイプ 4): このタイプの公示は ABR によって生成されます。要約リンク公示には、以下のような 2 つのタイプがあります。
 - タイプ 3 の要約 LSA は、OSPF ネットワーク内の他のエリアにある宛先 (エリアを越えた宛先) への経路について説明します。
 - タイプ 4 の要約 LSA は、ASBR への経路について説明します。
- 要約 LSA は、エリア間で到達可能性情報を交換するために使用されます。通常、情報はバックボーン・エリアに通知されます。この情報はその後バックボーンが他のエリアに送ります。
- AS 外部 LSA: このタイプの公示は、OSPF ネットワークの外部の宛先への経路について説明します。これは ASBR によって生成されます。この公示は OSPF ネットワーク内のすべてのエリアにフラッディングされます。

図 73 は、リンク状態公示のさまざまなタイプを表しています。

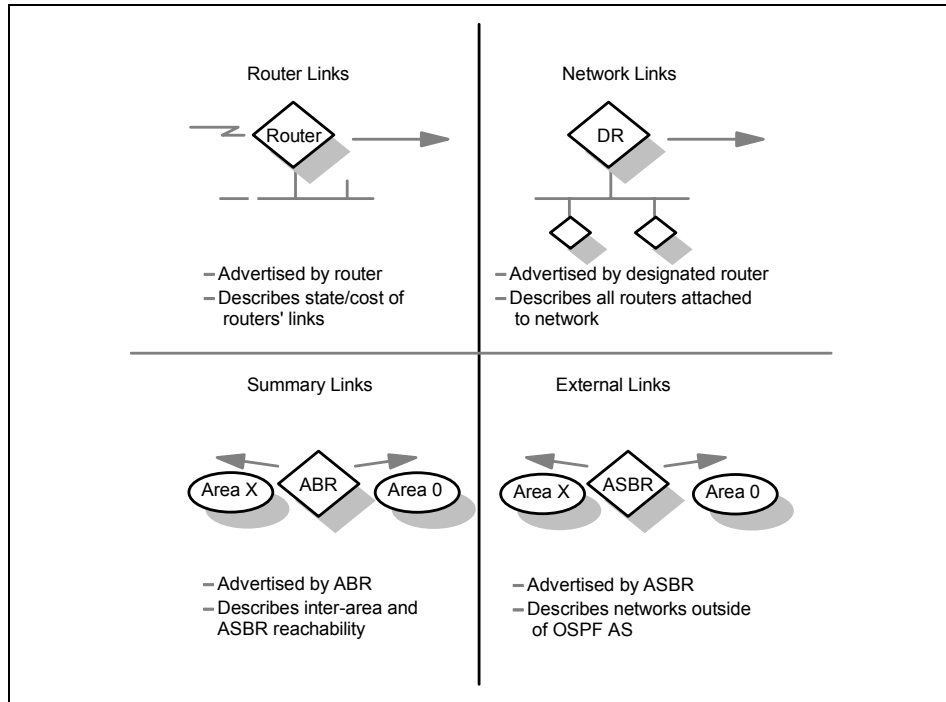


図 73 OSPF リンク状態公示

4.6.2 OSPF のパケット・タイプ

OSPF パケットは IP データグラムに入れて伝送されます。これは TCP または UDP パケット内にはカプセル化されません。IP ヘッダーはプロトコル ID 89 を使用します。OSPF パケットは IP ToS が「0」、IP 優先順位が「インターネットワーク制御」で送信されます。これは、パケットの優先処理を行うために使用されます。ToS および IP 優先順位の詳細については、812 ページの 22.2『統合サービス』にあります。

OSPF はできる限りマルチキャスト機能を使用して近隣装置と通信します。ブロードキャストおよび Point-to-Point 環境では、パケットは予約済みのマルチキャスト・アドレス 224.0.0.5 に送信されます。RFC 2328 はこれを AllSPFRouters アドレスと呼んでいます。非ブロードキャスト環境では、パケットは近隣者の特定の IP アドレスにアドレッシングされます。

すべての OSPF パケットは図 74 に示されている共通ヘッダーを共有します。このヘッダーにはエリア ID、RID、チェックサム、および認証情報を含む一般情報が含まれます。

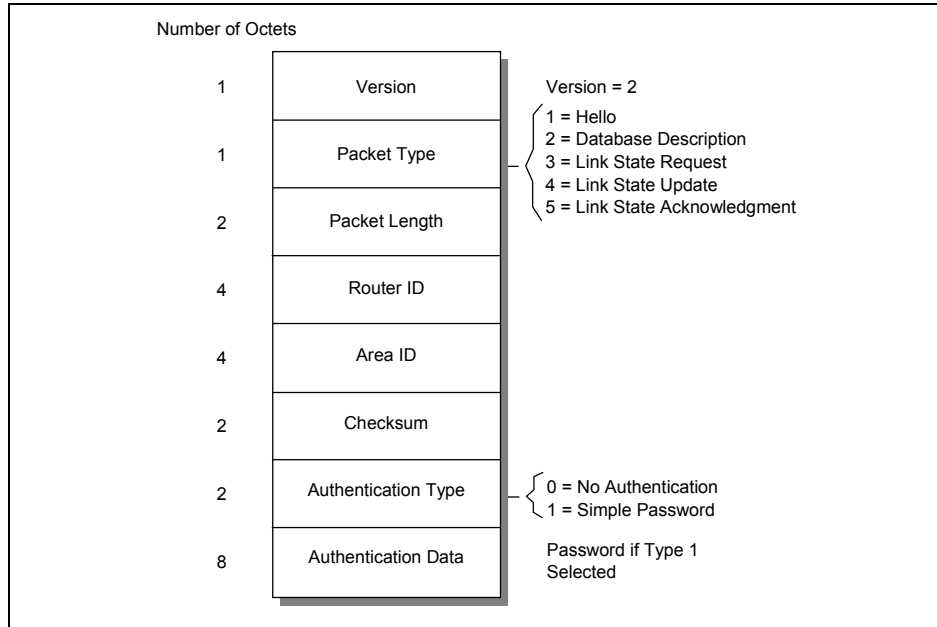


図 74 OSPF 共通ヘッダー

このタイプのフィールドは、OSPF パケットを以下の 5 つのいずれかのタイプとして識別します。

- Hello: このパケット・タイプは、近隣関係を発見して保守するために使用されます。
- データベース記述: このパケット・タイプは、ルーターのリンク状態データベースに含まれる LSA のセットについて説明します。
- リンク状態要求: このパケット・タイプは、より最新の LSA のインスタンスを近隣者に要求するために使用されます。
- リンク状態更新: このパケット・タイプは、より最新の LSA のインスタンスを近隣者に提供するために使用されます。
- リンク状態確認通知: このパケット・タイプは、新しく受信した LSA の受信を通知するために使用されます。

これらのパケットの使用法は、次のセクションで示します。

4.6.3 近隣者通信

OSPF はネットワーク上のパスの最適セットの判別を担当します。そのために、各ルーターはネットワーク内の他のルーターと LSA を交換します。

OSPF プロトコルは、この情報交換を行うための以下のようないくつかのアクティビティーを定義しています。

- 近隣者の発見
- 指定ルーターの選出
- 隣接関係の確立とデータベースの同期

5 つの OSPF パケット・タイプは、この情報交換をサポートするために使用されます。

4.6.3.1 近隣者の発見 - OSPF Hello プロトコル

Hello プロトコルは近隣ルーターとの関係を発見して保守します。Hello パケットは各ルーター・インターフェースに定期的送信されます。このパケットには、その Hello パケットがそのインターフェースを介してすでに受信されている他のルーターの RID が含まれます。

装置が他のルーターによって生成された Hello パケット内に自分の RID を見付けると、その 2 つの装置は近隣関係を確立します。

Hello パケットにはさらにルーター優先度、DR ID、および BDR ID が含まれます。これらのパラメーターはマルチアクセス・ネットワーク上の DR を選出するために使用されます。

4.6.3.2 指定ルーターの選出

すべてのマルチアクセス・ネットワークが DR を持っている必要があります。BDR を選択することもできます。バックアップは DR が故障した場合にルーティング機能に大きな損失が生じないようにします。

DR および BDR は、Hello パケットに含まれている情報を使用して選択されます。セグメント上で最も高い OSPF ルーター優先度を持つルーターがそのセグメント上の DR になります。同じプロセスが BDR の選択のために繰り返されます。順位がタイの場合は、最も大きい RID を持つルーターが選択されます。DR として宣言されたルーターは BDR になる資格はありません。

DR および BDR は選択されるとマルチアクセス・セグメント上のすべてのルーターとの隣接関係の確立に取りかかります。

4.6.3.3 隣接関係の確立とデータベースの同期

近隣ルーター同士がそのリンク状態データベースを同期している場合、それらは隣接していると見なされます。ルーターはすべての近隣装置とは隣接関係を確立しません。マルチアクセス・ネットワークでは、隣接関係は DR および BDR との間にだけ築かれます。これは以下の 2 ステップのプロセスになります。

ステップ1: データベース交換プロセス

データベース同期の最初のフェーズは、データベース交換プロセスです。これは 2 つの近隣者が隣接関係を確立しようとするときすぐ始まります。このプ

ロセスはデータベース記述パケットの交換から成り立っています。パケットにはローカル・データベースに保管されている LSA のリストが含まれています。

データベース交換プロセスの間、ルーターはマスター / スレーブ関係を築きます。マスターは最初に伝送を行います。各パケットはシーケンス番号によって識別されます。このシーケンス番号を使用して、スレーブはマスターからの各データベース記述パケットの受け取りを通知します。スレーブはまたその確認通知に自分自身のリンク状態ヘッダーのセットを含めます。

ステップ 2: データベース・ロード

データベース交換プロセス中、各ルーターは、より最新のそのインスタンス (すべての公示にタイム・スタンプが押されている) が近隣者に保有されているリンク状態ヘッダーを認識します。プロセスが完了すると、各ルーターはより最新の情報を近隣者に要求します。この要求は、リンク状態要求パケットを使用して行われます。

ルーターはリンク状態要求を受信したら、要求されている LSA を提供するリンク状態更新パケットのセットで応答しなければなりません。受信側は伝送された各 LSA の受け取りを通知します。このプロセスは、トポロジー変更のネットワークへの伝送のために使用される、信頼性のあるフラッディングの手続きと類似しています。

すべての LSA には、公示の開始からの時間 (秒単位) を示す経過時間フィールドが含まれます。経過時間は LSA がトポロジー・データベースにインストールされたあと増え続けます。これはフラッディング・プロセスの各ホップ中も増加します。最大経過時間に達すると、その LSA はルーティング情報の判別に使用されなくなり、リンク状態データベースから破棄されます。この経過時間はその他の点が同一な公示の 2 つのコピーを区別するためにも使用されます。

4.6.4 OSPF 近隣状態マシン

OSPF 仕様は近隣状態のセットと、近隣者がある状態から他の状態に移移する原因となる可能性のあるイベントを定義しています。状態マシンは以下のような遷移について説明するために使用されます。

- Down: これは初期状態です。これは、セグメント上の装置から新着情報を受け取っていないことを示します。
- Attempt: この状態は非ブロードキャスト・ネットワークで使用されます。これは、近隣者が非アクティブのようであることを示します。接続の再開が引き続き試みられています。

- Init: 近隣者との通信が開始していますが、双方向通信は確立していません。具体的には、近隣者から Hello パケットを受け取りましたが、近隣者の Hello パケットにそのローカル・ルーターがリストされていません。
- 2-way: 2 つのルーターの間に双方向通信が確立しています。隣接関係を築くことができます。近隣者は指定ルーターとして選出することができます。
- ExStart: 近隣者同士は隣接関係を築き始めています。
- Exchange: 2 つの装置はそのデータベース・トポロジを交換しています。
- Loading: 2 つの近隣者はそのトポロジ・データベースを同期しています。
- Full: 2 つの近隣者は完全に隣接関係にあり、そのデータベースは同期されています。

近隣者の OSPF 状態はネットワーク・イベントによって変化します。たとえば、ルーターが近隣装置から Hello パケットを受け取ると、OSPF 近隣状態は Down から Init に変化します。双方向通信が確立されると、近隣状態は Init から 2-Way に変化します。RFC 2328 には状態変更を引き起こすイベントの完全な説明があります。

4.6.5 OSPF の仮想リンクと中継エリア

仮想リンクは、ネットワークが標準の OSPF ネットワーク・トポロジをサポートしていない場合に使用されます。このトポロジは各追加 OSPF エリアに直接接続するバックボーン・エリアを定義します。仮想リンクは以下の 2 つの状況に対応します。

- これは、バックボーン・エリアが隣接していない場合にそれを論理的に接続することができます。
- これは、直接接続が存在しない場合にエリアをバックボーンに接続することができます。

仮想リンクは共通の非ブロードキャスト・エリアを共用する 2 つの ABR 間に確立されます。このリンクは Point-to-Point リンクとして扱われます。共通エリアは中継エリアと呼ばれます。図 75 は、エリアをバックボーンに接続するために、仮想リンクと中継エリアが使用されているときのそれらの間の相互作用を表しています。

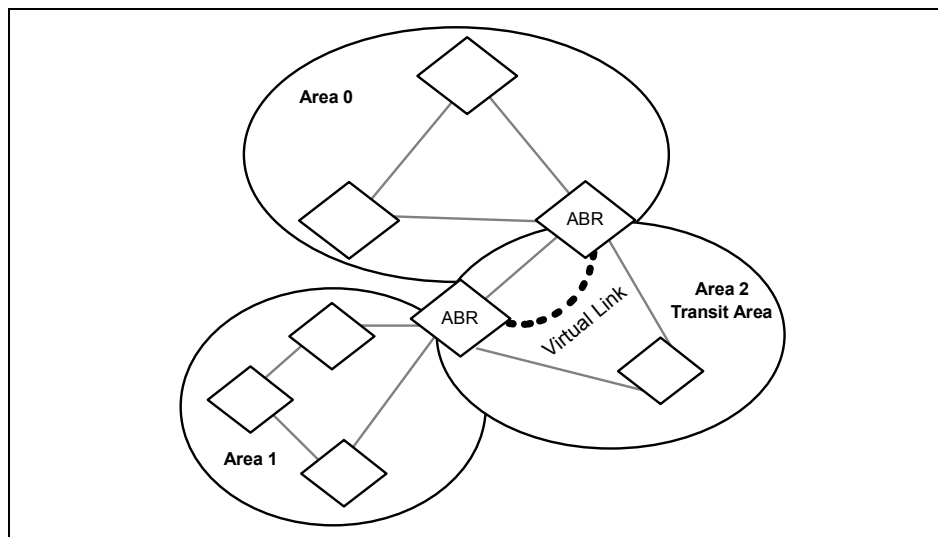


図 75 OSPF の仮想リンクと中継エリア

このデータグラムは、エリア 1 がバックボーンへの直接接続を持っていないことを示しています。この接続を供給するためには、エリア 2 を中継エリアとして使用することができます。仮想リンクはエリア 2 にある 2 つの ABR 間に確立されています。この仮想リンクの確立により、バックボーン・エリアは論理的に拡張されてエリア 1 に接続するようになっています。

仮想リンクは、ルーティング情報を伝送するためだけに使用されます。これはリモート・エリアとバックボーンの間での正規のトラフィックは伝送しません。このトラフィックと仮想リンク・トラフィックは、中継エリア内の標準のエリア内ルーティングを使用してルーティングされます。

4.6.6 OSPF の経路再配布

経路再配布とは、外部経路を OSPF ネットワークに紹介するプロセスです。この経路は、静的経路か、あるいは他のルーティング・プロトコルを介して認識される経路のいずれかにすることができます。これらは ASBR によって OSPF ネットワークに公示されます。これらの経路は OSPF の外部経路になります。ASBR は、OSPF AS 外部 LSA を OSPF ネットワーク全体にフラッディングすることによってこれらの経路を公示します。

経路が記述する終端間パスは以下の 2 つの部分から成り立ちます。

- ・ 外部部分：これはパスのうちの OSPF ネットワークの外部にあたる部分です。この経路が OSPF 内に配布されると、ASBR が初期コストを割り当て

ます。このコストは、パスの外部部分のトラバースに関連した**外部コスト**を表します。

- 内部部分：これはパスのうちの OSPF ネットワークの内部にあたる部分です。ネットワークのこの部分のコストは、標準の OSPF アルゴリズムを使用して計算されます。

OSPF は外部経路の 2 つのタイプを区別しています。これらは、経路のコストの計算方法が異なります。ASBR は経路を以下のように再配布するよう構成されています。

- 外部タイプ 1: 経路の合計コストは、外部コストと内部 OSPF コストの合計になります。
- 外部タイプ 2: 経路の合計コストは常に外部コストになります。この場合 ASBR に到達するために必要な内部 OSPF のコストは無視されます。

図 76 は、OSPF の外部経路のタイプの例を表しています。

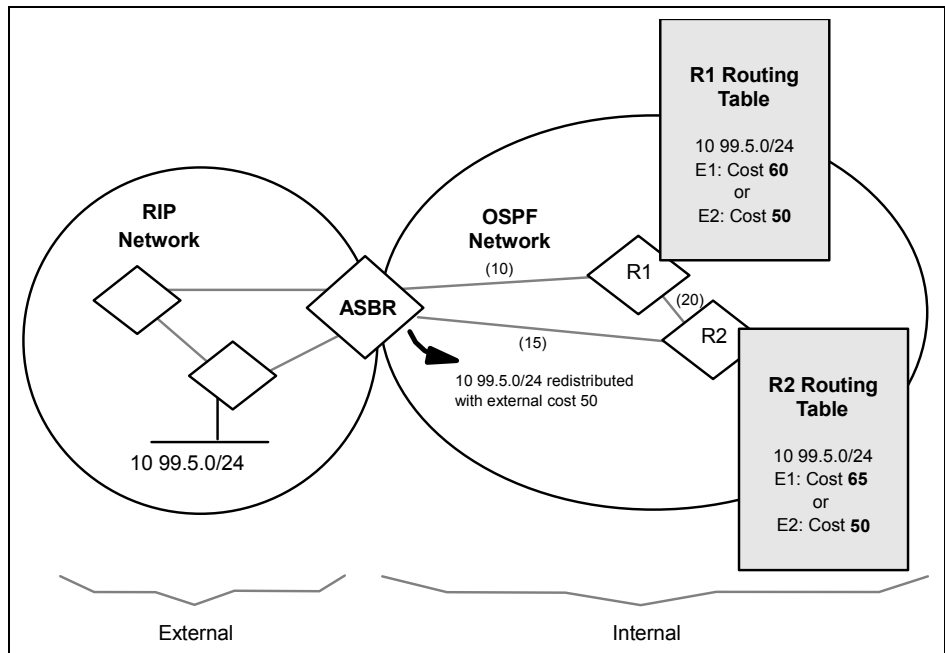


図 76 OSPF の経路再配布

この例では、ASBR は 10.99.5.0/24 経路を OSPF ネットワークに再配布しています。このサブネットは RIP ネットワーク内にあります。この経路は、外部コスト 50 の OSPF 内に通知されています。これは、パスのうちの RIP ネットワークをトラバースする部分のコストを表しています。

- ASBR が経路を E1 経路として再配布した場合、R1 にはコストが 60 (50 + 10) であるこのサブネットへの外部経路が含まれるようになります。R2 はコストが 65 (50 + 15) の外部経路を持つようになります。
- ASBR が経路を E2 経路として再配布した場合は、R1 と R2 の両方にコストが 50 であるこのサブネットへの外部経路が含まれるようになります。OSPF ネットワーク内のセグメントのトラバースに関連したコストは、宛先に到達するための合計コストには含められません。

4.6.7 OSPF のスタブ・エリア

OSPF では、特定のエリアをスタブ・エリアとして定義することができます。スタブ・エリアは、スタブ・エリアに接続する ABR が、そのエリアへの AS 外部 LSA のフラッディングを禁止する場合に作成されます。これは、スタブ・エリア・ルーター内に保持されるリンク状態データベースのサイズを縮小するために行われます。外部ネットワークへの特定の経路は存在しないので、これらの宛先へのルーティングは、ABR によって生成されるデフォルトの経路に基づきます。スタブ・エリア内に保持されるリンク状態データベースには、デフォルトの経路と OSPF 環境内からの経路 (たとえば、エリア内およびエリア間経路) だけが含まれます。

スタブ・エリアは外部 LSA を許可しないので、スタブ・エリアに ASBR を含めることはできません。スタブ・エリア内から外部経路を生成することはできません。

スタブ・エリアは、エリアとバックボーンを接続する単一の出口点が存在する場合に配備できます。複数の出口点が存在するエリアをスタブ・エリアにすることもできます。しかし、そのエリアを出るパケットが最適なパスをたどるという保証はありません。これは、各 ABR がデフォルトの経路を生成するためです。トラフィックを特定のデフォルト経路と関連付ける機能はありません。

このエリア内のルーターはすべてスタブ・ルーターとして構成される必要があります。この構成は Hello パケットの交換を通して検査されます。

4.6.7.1 Not-So-Stubby エリア

スタブ・エリア概念の拡張は、*Not-So-Stubby エリア (NSSA)* です。この代替案は RFC 1587 に文書化されています。NSSA を保守する ABR が、外部経路を NSSA にフラッディングしないという点で、NSSA はスタブ・エリアと類似しています。NSSA にフラッディングする唯一の経路は、デフォルトの経路と OSPF 環境内からのその他の経路 (たとえば、エリア内およびエリア間) です。

しかし、スタブ・エリアとは異なり ASBR は NSSA 内に置くことができます。この ASBR は外部経路を生成できます。したがって、NSSA 内に保持さ

れるリンク状態データベースには、デフォルトの経路、OSPF 環境内からの経路 (たとえば、エリア内およびエリア間経路)、およびエリア内の ASBR によって生成された外部経路が含まれます。

NSSA を保守する ABR は NSSA 内からの外部経路を残りの OSPF ネットワークにフラッドします。

4.6.8 OSPF の経路要約

経路要約とは、連続する複数のルーティング・エントリーを単一の公示の中に統合するプロセスです。これにより、リンク状態データベースと IP ルーティング・テーブルのサイズは縮小されます。OSPF ネットワークでは、要約はボーダー・ルーターで実行されます。要約には以下のような 2 つのタイプがあります。

- エリア間経路要約：エリア間要約は、エリアの ABR によって実行されます。これはエリア内を起点とする経路公示の要約に使用されます。要約された経路はバックボーンに通知されます。バックボーンは集約された経路を受信し、その要約を他のエリアに通知します。
- 外部経路要約：このタイプの要約は、特に OSPF 内に送信される外部経路に適用されます。これはその経路を OSPF ネットワーク内に配布する ASBR によって実行されます。

図 77 は、OSPF の経路要約の例を表しています。

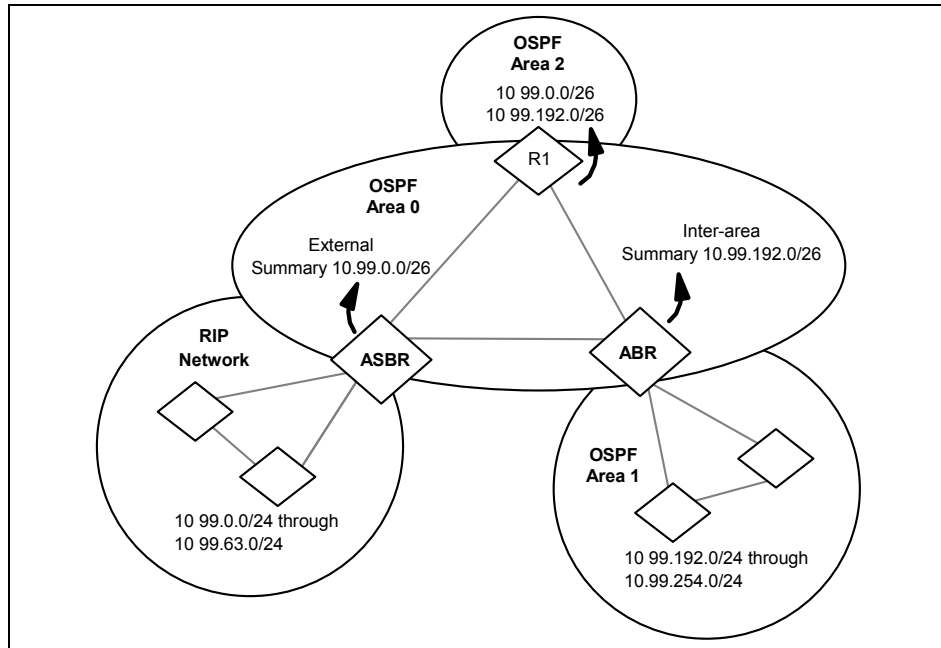


図 77 OSPF の経路要約

この図で、ASBR は RIP 環境内にある 64 のネットワークに対する単一の要約経路を公示しています。この単一の要約経路は OSPF ネットワーク全体にフラッドイングされます。さらに、ABR はエリア 1 にある 64 のサブネットワークに対する単一の要約経路を生成しています。この要約経路はエリア 0 とエリア 2 にフラッドイングされます。ASBR の構成によっては、エリア間要約経路が RIP ネットワーク内に再配布されることもあります。

4.7 Enhanced Interior Gateway Routing Protocol (EIGRP)

Enhanced Interior Gateway Routing Protocol (EIGRP) は、ハイブリッド・ルーティング・プロトコルとして分類されています。距離ベクトル・アルゴリズムと類似して、EIGRP はメトリックを使用してネットワーク・パスを決定します。しかし、リンク状態プロトコルと同様に、EIGRP 環境でのトポロジー更新はイベント・ドリブンです。

この名前が示すとおり、EIGRP は AS 内で使用するために設計された内部ゲートウェイ・プロトコルです。正しく設計された環境では、基本の距離ベクトル・アルゴリズムよりも EIGRP のほうがスケーラビリティが高く収束が高速になる可能性があります。さらに EIGRP は複雑な冗長度の高いネットワークをサポートするのに適しています。

EIGRP は Cisco Systems, Inc. によって開発された所有権付きのプロトコルです。本書の執筆時点では、これは IETF の標準プロトコルではありません。

4.7.1 EIGRP のフィーチャー

EIGRP にはいくつかの利点があります。これらの利点の一部は、距離ベクトルまたはリンク状態アルゴリズムにも存在します。

- より高速な収束：EIGRP は優先パスが故障した場合に使用することのできる代替経路のリストを保持しています。パスが故障すると、新しい経路が即時に IP ルーティング・テーブルへインストールされます。経路の再計算は実行されません。
- 部分的なルーティング更新：EIGRP が近隣ルーターを発見すると、各装置はそのルーティング・テーブル全体を交換します。初期情報交換が終わると、ルーティング・テーブルの変更点だけが伝搬されます。ルーティング・テーブル全体の定期的な再ブロードキャストはありません。
- 低帯域幅使用率：通常のネットワーク操作の際は、Hello パケットだけが継続的なネットワークに伝送されます。
- CIDR と VLSM: EIGRP はサブネット化と可変長サブネット・マスクをサポートしています。これにより、ネットワーク管理者は IP アドレス・リソースを効率的に割り当てることができます。
- 経路要約：EIGRP はルーティング通知を要約する機能をサポートしています。これにより、不必要なサブネット情報の公示は制限されます。
- マルチプロトコル：EIGRP は AppleTalk、IPX、および IP ネットワークにネットワーク層ルーティングを提供することができます。
- 等しくないコストのロード・バランシング：EIGRP は、コストが等しくない宛先への複数のパスの同時使用をサポートしています。各経路は IP ルーティング・テーブルにインストールされます。EIGRP はまた、複数のパス上のトラフィックのロード・バランシングを適切な判断で行います。

4.7.2 用語

EIGRP はプロトコルの操作の説明に特有の用語を使用しています。

- 継承者：継承者とは、特定の宛先へのパケット転送に現行で使用されている近隣ルーターです。この装置は宛先への最低コスト・パスが含み、ルーティング・ループに参加していないことが保証されています。図 78 に示されている宛先ネットワークに到達しようとする場合、ルーター B はルーター A に対する現行の継承者になります。
- 継承適任者：継承適任者は現行の継承者ルーターが故障したときに転送の責任を負います。継承適任者のセットは、経路の再計算を必要としたりルーティング・ループを取り入れたりせずに継承者になれる装置を表します。

宛先への継承適任者のセットは、近隣ルーターによって公示された最低コスト・パスの全リストを検討することによって判別されます。このリストの中で、現行のルーティング・テーブル・メトリックより小さい公示済みメトリックを持つ近隣者が継承適任者と見なされます。

図 78 は、継承適任者の関係の例です。

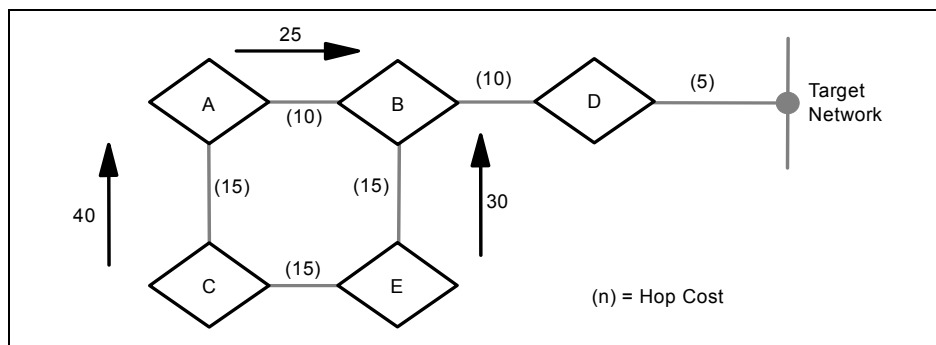


図 78 EIGRP 継承適任者

この図には、宛先ネットワークに到達するためのコストが表示されています。たとえば、ルーター C から宛先ネットワークへのコストは 40 (15 + 10 + 10 + 5) です。ルーター E から宛先ネットワークへのコストは 30 (15 + 10 + 5) です。

ルーター E は、現行のルーター C のルーティング・メトリック (40) より小さいコスト (30) を公示しています。したがって、ルーター C はルーター E を宛先ネットワークに到達するための継承適任者として認識します。逆は真でない点に注意してください。ルーター C によって公示されているコスト (40) は現行のルーター E の経路 (30) よりも大です。したがって、ルーター E はルーター C を宛先ネットワークへの継承適任者として認識しません。

- 近隣テーブル：EIGRP は隣接関係にある各近隣者の状態をトラッキングするためのテーブルを保守します。このテーブルには近隣者に到達するために使用されるアドレスとインターフェースが含まれます。またこれには近隣者からのパケットに含まれていた最後のシーケンス番号も含まれます。これによって、EIGRP の信頼性のあるトランスポート・メカニズムは順番の狂ったパケットを検出することができます。
- トポロジー・テーブル：EIGRP はトポロジー・テーブルを使用して IP ルーティング・テーブル内に経路をインストールします。トポロジー・テーブルには、近隣ルーターによって現行で公示されているすべての宛先ネットワークがリストされます。このテーブルには、各宛先への距離とベ

クトルのセットを構築するのに必要なすべての情報が含まれます。この情報には以下が含まれます。

- この宛先に到達するために使用されるセグメント上の最小使用可能帯域幅。
- 合計遅延、信頼性、およびパスのロード。
- パスで使用する最小 MTU。
- 実現可能なパスの距離。これは、宛先ネットワークへのパスの最良メトリックを表します。これにはパスを公示している近隣者に到達するために使用されるメトリックも含まれています。
- 報告されているパスの距離。これは、アップストリームの近隣者によって公示されている宛先ネットワークへのパスの合計メトリックを表します。
- 経路の原始。EIGRP は外部経路をマークします。これによって、ルーティング・パターンをカスタマイズするポリシー制御をインプリメントすることができます。

トポロジー・テーブル内のエントリーの状態は、以下の 2 つのいずれかになります。

- パッシブ状態：ルーターはこのエントリーの経路の再計算を実行していません。
 - アクティブ状態：ルーターはこのエントリーの経路の再計算を実行しています。経路の継承適任者が存在する場合は、エントリーがこの状態に入ることはありません。これによって、プロセッサに負荷のかかる経路の再計算が防止されています。
- 信頼性のあるトランスポート・プロトコル：EIGRP は、近隣者への規制されたパケットの送達を保証することができます。ただし、すべてのタイプのパケットを信頼性のある方法で伝送する必要があるわけではありません。たとえば、マルチキャストをサポートする環境では、個々の確認 Hello パケットを各近隣者に送信する必要はありません。操作を効率的にするため、信頼性は必要な場合にだけ提供されます。これにより、多様な速度の接続を含むネットワークの収束時間は改善されます。

4.7.3 近隣者の発見とリカバリー

EIGRP は、直接接続されたネットワーク上の他のルーターについて動的に学習することができます。これは OSPF 環境で近隣者の発見に使用される Hello プロトコルと類似しています。

EIGRP ネットワーク内の装置は Hello パケットを交換して各近隣者が作動可能であることを検査します。OSPF と同様、パケットの交換に用いられる頻度はネットワーク・タイプに基づきます。高帯域幅リンク（たとえば、LAN セグメント）では、パケットは 5 秒間隔で交換されます。その他の場合、低帯域幅接続では Hello パケットは 60 秒ごとに交換されます。

OSPF と同様、EIGRP は保持タイマーを使用して非アクティブ近隣者を除去します。このタイマーは、装置が近隣者から Hello パケットを受信しなくてもその近隣者をアクティブと見なし続ける時間を示します。

4.7.4 DUAL アルゴリズム

典型的な距離ベクトル・プロトコルは、定期的な更新を使用して宛先への最善のパスを計算します。これは距離、ネクスト・ホップ、およびローカル・インターフェース・コストを使用してパスを決定します。この情報は一度処理されると、破棄されます。EIGRP はトポロジーの収束のために定期的な更新には頼りません。その代わり、これはその近隣者それぞれの公示を含んだトポロジー・テーブルを構築します。距離ベクトル・プロトコルとは違い、このデータは破棄されません。

EIGRP は、トポロジー・テーブル内の情報を処理して、各宛先ネットワークへの最善のパスを決定します。EIGRP は、DUAL (Diffusing Update ALgorithm) と呼ばれるアルゴリズムをインプリメントしています。このアルゴリズムには、以下のようないくつかの利点があります。

- DUAL アルゴリズムは、経路計算と収束期間におけるループのない操作を保証します。
- DUAL アルゴリズムでは、すべてのルーターを同時に同期することができます。これは、ルーティング更新の伝搬によって装置が異なる速度で収束する RIP 環境とは異なります。
- DUAL アルゴリズムでは、トポロジー変更に関係しないルーターは経路の再計算をしないで済みます。

DUAL アルゴリズムは、宛先に対する継承適任者のセットを見つけるために使用することができます。ネットワークで障害状態が発生すると、即時に代替経路が IP ルーティング・テーブルへ追加されます。これによって、代替パスを判別するための不必要な計算は防止されます。継承適任者が分かっている場合は、経路の再計算が行われます。この動作は、図 79 と図 80 に示されています。

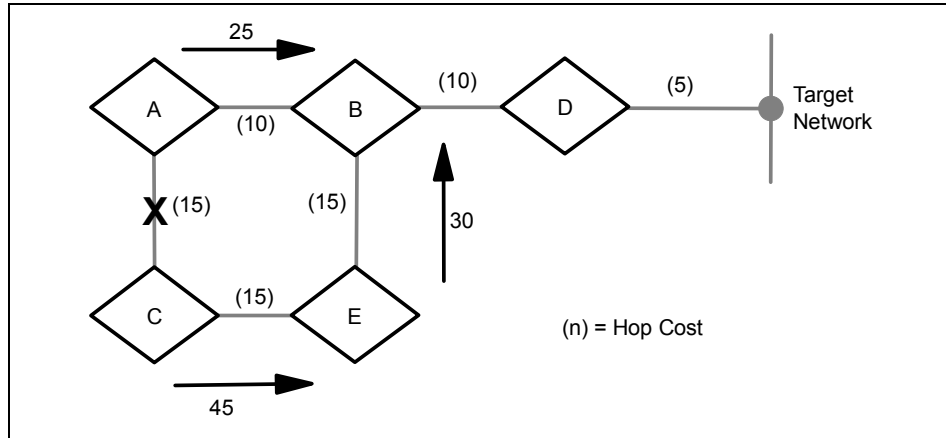


図 79 継承適任者の使用

この例で、ルーター C はルーター E を宛先ネットワークに到達するための継承適任者として使用しています。ルーター A とルーター C の間の接続が故障すると、ルーター C は即時にルーター E を通してトラフィックを転送します。新しい経路が IP ルーティング・テーブル内で更新されます。

4.7.4.1 経路の再計算

経路の再計算は、宛先への既知の継承適任者が存在しない場合に行われます。このプロセスは、宛先への継承適任者を知る近隣者がいないか判別するため、ルーターがマルチキャストの照会パケットを送信するときに始まります。近隣者は継承適任者を持っていればこれに応答します。近隣者が継承適任者を持っていない場合、その近隣者は、自分も経路の再計算を実行中であることを示す照会を戻す可能性があります。

図 80 は、継承適任者を判別するための照会の例を表しています。この例で、ルーター E は、宛先ネットワークへの継承適任者を保有していません。ルーター E とルーター B を接続するリンクが故障したら、ルーター E は新しいパスを決定しなければなりません。ルーター E はその近隣者それぞれにマルチキャストの照会を送信します。ルーター C は継承適任者を保有しているのでルーター E に応答します。ルーター E は、コストが 55 の新しいパスで、その IP ルーティング・テーブルを更新します。

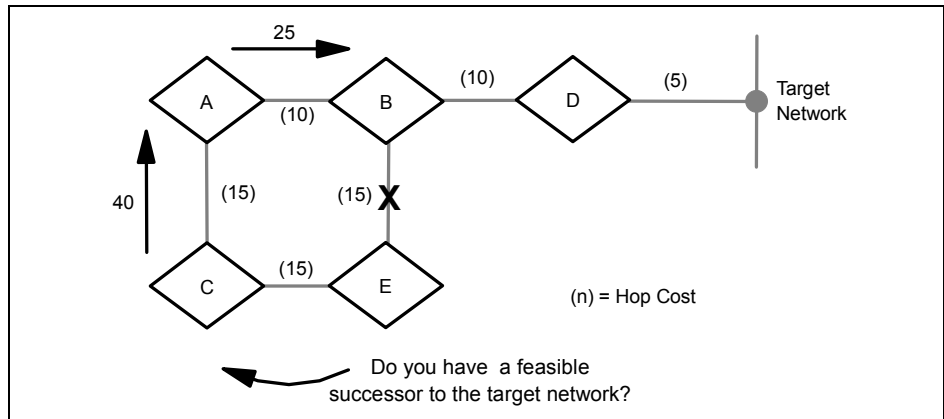


図 80 継承適任者の照会

近隣者へのリンクが故障すると、その近隣者を唯一の継承適任者として使用していたすべての経路に経路再計算が必要になります。

4.7.4.2 EIGRP のメトリック

EIGRP は数式を使用してパスに関連したメトリックを判別します。デフォルトでは、この数式は宛先に到達するために使用されるセグメントの最小帯域幅を参照します。これはまたパスの遅延を合計します。メトリックを判別するためのデフォルトの数式は以下のとおりです。

$$\left[\left(\frac{10^7}{\text{minbandwidth}} \right) + \text{sumofdelays} \right] \times 256$$

EIGRP は、メトリック計算への他の測定の組み込みをサポートしています。

4.7.5 EIGRP のパケット・タイプ

EIGRP は以下の 5 つのタイプのパケットを使用して近隣関係を確認し、ルーティング情報を公示します。

- Hello/ 確認通知：このパケットは近隣者の発見に使用されます。これは各ネットワーク・セグメントで公示されるマルチキャストです。Hello パケットに対してはユニキャスト応答が戻されます。

データを含まない Hello パケットは確認通知と見なされます。

- 更新：このパケットは各宛先の到達可能性情報を伝達するために使用されます。新しい近隣者が発見されると、各近隣者がそのトポロジー・テーブルを構築できるようにするために、ユニキャストの更新パケットが交換されます。他のタイプの公示（たとえば、メトリック変更）では、マルチ

キャスト・パケットが使用されます。更新パケットは常に信頼性のある方法で伝送されます。

- 照会と応答：これらのパケットは宛先がアクティブ状態に入るときに交換されます。マルチキャストの照会パケットは、いずれかの近隣者に宛先への継承適任者が含まれていないかを判別するために送信されます。ユニキャストの応答パケットは、継承適任者が識別されているので近隣者がアクティブ状態に入る必要がないことを示すために送信されます。照会および応答パケットは信頼性のある方法で伝送されます。
- 要求：このパケットは近隣者から特定の情報を取得するために使用されます。このパケットは経路サーバー・アプリケーションで使用されます。

4.8 Exterior Gateway Protocol (EGP)

EGP は、歴史的功績としての外部ゲートウェイ・プロトコルです。これは、自律システム (AS) 間の通信のために開発された最初のプロトコルの 1 つでした。これは RFC 904 で説明されています。

EGP は、ネットワークが単一のバックボーンを含み、どの 2 つの自律システムの間にも単一のパスが存在することを想定しています。この制限のため、EGP は現在あまり使用されていません。実際、EGP は BGP によって置き換えられています。

EGP は Hello/I-hear-you メッセージ交換を使用した定期的なポーリングに基づきます。これらは近隣者の到達可能性をモニターして更新応答を要求するために使用されます。

AS に接続しているゲートウェイは、ローカル AS 内の到達可能な宛先ネットワークだけを公示することを許可されます。これは AS の外にあるその EGP 近隣者に関する到達可能性情報は公示しません。

4.9 Border Gateway Protocol (BGP)

Border Gateway Protocol (BGP) は外部ゲートウェイ・プロトコルです。これは、もともと自律システム間でのループのないルーティング情報交換方式を、提供するために開発されました。以来 BGP は進化し、ルーティング情報の集約および要約をサポートするようになっています。

BGP は、RFC 1771 で説明されている IETF のドラフト標準プロトコルです。この RFC で説明されているバージョンは、BGP バージョン 4 です。標準慣例に従い、本書では BGP バージョン 4 を参照するときに BGP という用語を使用しています。

4.9.1 BGP の概念と用語

BGP はプロトコルの操作の説明に特有の用語を使用しています。この用語は図 81 を使用して説明します。

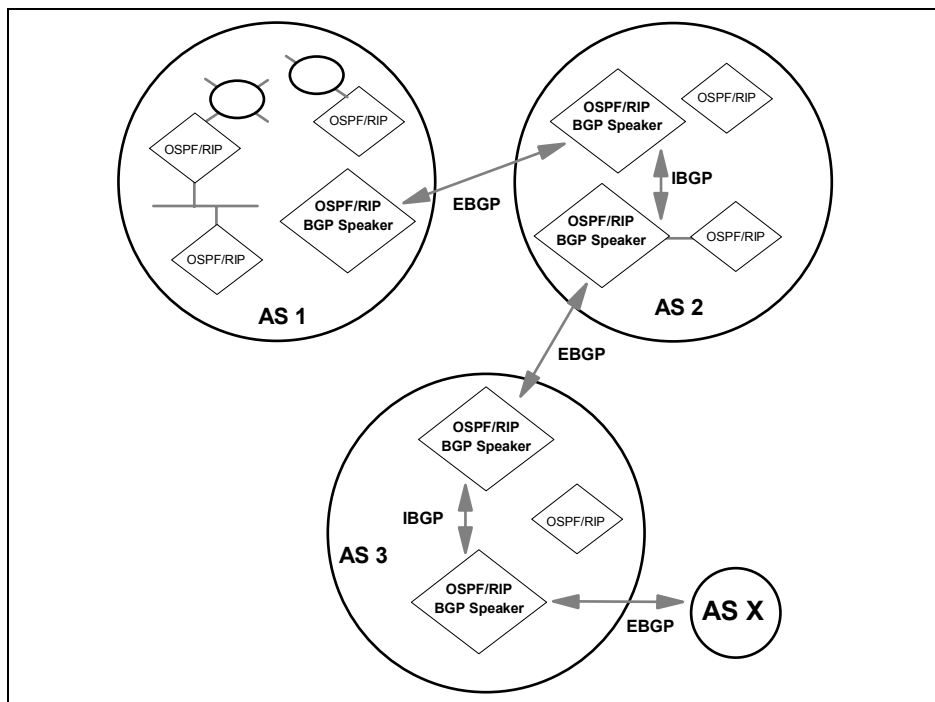


図 81 BGP ネットワークのコンポーネント

- BGP スピーカー：BGP をサポートするよう構成されているルーター。
- BGP 近隣者 (ピア)：ルーティング情報を交換する BGP スピーカーのペア。BGP 近隣者には以下のような 2 つのタイプがあります。
 - 内部 (IBGP) 近隣者：同じ AS 内の BGP スピーカーのペア。
 - 外部 (EBGP) 近隣者：それぞれ異なる AS 内にある BGP 近隣者のペア。これら近隣者同士は基本的に、直接接続されたネットワークを共有します。
- BGP セッション：2 つの BGP 近隣者を接続する TCP セッション。このセッションは、ルーティング情報を交換するために使用されます。近隣者はキープアライブ・メッセージを送信することによってセッションの状態をモニターします。¹

¹ このキープアライブ・メッセージは、アプリケーション層にインプリメントされます。これは、多くの TCP インプリメンテーションで使用可能なキープアライブ・メッセージとは無関係です。

- **トラフィック・タイプ**: BGP は以下のような 2 つのタイプのトラフィックを定義しています。
 - **ローカル**: AS に対してローカルなトラフィックは、その AS 内に起点または終点を持ちます。送信元か宛先のいずれか IP アドレスがその AS 内にあります。
 - **中継**: ローカル・トラフィックでないトラフィックはいずれも中継トラフィックです。BGP の目標の 1 つは、中継トラフィックの量を最小化することです。
- **AS タイプ**: BGP は以下のような 3 つのタイプ自律システムを定義しています。
 - **スタブ**: スタブ AS は他の 1 つの AS への単一の接続を持ちます。スタブ AS はローカル・トラフィックだけを伝送します。
 - **マルチホーム**: マルチホーム AS は複数の自律システムへの接続を持ちます。しかし、マルチホーム AS は中継トラフィックが転送しないように構成されています。
 - **中継**: 中継 AS は複数の自律システムへの接続を持ち、ローカルと中継の両方のトラフィックを伝送します。この AS は転送される中継トラフィックのタイプにポリシー制限を課すことがあります。

図 81 の AS 2 内の BGP 装置の構成によって、この自律システムはマルチホーム AS と中継 AS のいずれにもなります。

- **AS 番号**: AS を一意的に識別する 16 ビットの番号。
- **AS パス**: ネットワーク上の経路について説明する AS 番号のリスト。BGP 近隣者はそのピアにパスを伝達します。
- **ルーティング・ポリシー**: ネットワーク上のデータ・パケットのフローを制限する規則のセット。BGP プロトコルではルーティング・ポリシーは定義されていません。むしろこれらは BGP 装置の構成に使用されます。たとえば、BGP 装置は以下のように構成することができます。
 - マルチホーム AS が中継 AS として動作することを拒否できる。このことは AS 内に含まれるネットワークだけを公示することによって実現できます。
 - マルチホーム AS が限定された隣接自律システムのセットに対してのみ、中継 AS ルーティングを実行できる。これは EBGP ピアに送信されるルーティング公示を調整することによってこのことを行います。
 - AS がトラフィック最適化して、特定のカテゴリのトラフィックに対して特定の AS パスを使用できる。
- **ネットワーク層到達可能性情報 (NLRI)**: NLRI は、BGP が経路を公示するために使用します。これはタプル <length,prefix> で表されたネットワーク

のセットで構成されます。たとえば、タプル <14,220.24.106.0> は CIDR 経路 220.24.106.0/14 を表します。

- 経路とパス：経路は宛先を、宛先へのパスについて説明する属性の集合と関連付けます。宛先は NRLI フォーマットで指定されます。パスはパス属性の集合として報告されます。この情報は、UPDATE メッセージに入れて公示されます。UPDATE メッセージについての追加情報は、194 ページの 4.9.3『プロトコルの説明』にあります。

4.9.2 IBGP と EBGP の通信

BGP は AS 内で働く IGP を置き換えません。その代わりに、これは IGP と協調して自律システム間の通信を確立します。AS 内の BGP はローカル IGP 経路の公示に使用されます。これらの経路は、他の自律システム内の BGP ピアに公示されます。図 82 は、BGP ピア間で行われる通信を表しています。この例は 4 つの自律システムを表しています。AS 2、AS 3、および AS 4 はそれぞれ AS 1 への EBGP 接続を持ちます。AS 1 内の BGP 装置間には完全メッシュの IBGP セッションが存在します。

ネットワーク 10.0.0.0/8 は AS 3 内にあります。このネットワーク存在は BGP を使用して残りの環境に公示されます。

- AS 3 内の R4 は、その EBGP 接続を使用してそのネットワークを AS 1 に通知します。
- AS 1 内の R1 は、その IBGP 接続を使用してそのネットワークを R2 と R3 に通知します。
- AS 1 内の R2 は、その EBGP セッションを使用してそのネットワークを AS 2 に通知します。AS 1 内の R3 は、その EBGP セッション 5 を使用してそのネットワークを AS 4 に通知します。

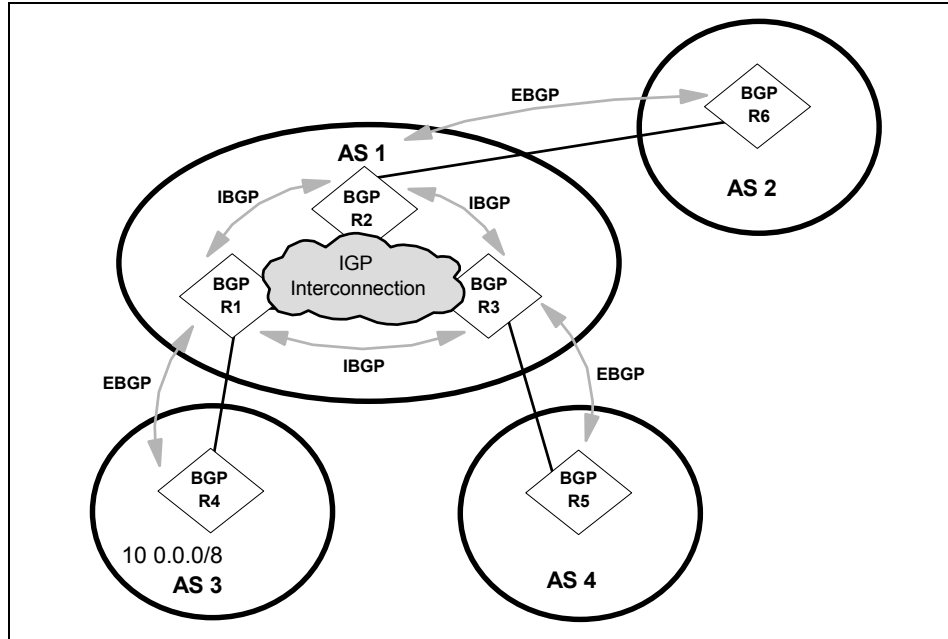


図 82 EBGP と IBGP の通信

図 82 は、さらに以下のようないくつかの操作問題を表しています。

- BGP と IGP の役割：この図は、自律システム間の情報は BGP だけが伝送するのに対し、AS 内の情報の伝送には BGP と IGP の両方が使用されていることを示しています。
- ピア間の TCP セッションの確立：BGP セッションを確立する前に、装置は以下のピアに到達するためのルーティング情報が提供されていることを検査します。
 - EBGP ピア：EBGP ピア同士は基本的に、直接接続されたネットワークを共有します。これらのピア間で BGP パケットを交換するのに必要なルーティング情報はそれほど大きくありません。
 - IBGP ピア：IBGP ピアは AS 内のどこにでも存在する可能性があります。これらを直接接続する必要はありません。BGP はピアを見つけるために IGP に頼ります。IBGP ピア間のパケット転送には、IGP で認識された経路が使用されます。
- AS 内の完全メッシュの BGP セッション：IBGP スピーカーは、同一の AS 内のピア間には完全メッシュの BGP セッションが確立していると想定します。図 82 では、AS 1 内の 3 つのすべての BGP ピアが BGP セッションで相互接続されています。

BGP スピーカーが IBGP ピアから経路更新を受け取ると、この受信スピーカーは EBGP を使用してその更新を外部ピアに伝搬します。受信スピーカーは完全メッシュの IBGP セッションが確立されていることを想定しているので、他の IBGP ピアにはこの更新を伝搬しません。

たとえば、図 82 の R1 と R3 の間に IBGP セッションがなかったと考えてください。R1 は 10.0.0.0/8 に関する更新を AS 3 から受け取ります。R1 はこの更新をその BGP ピア、すなわち R2 に転送します。R2 はこの IBGP 更新を受け取り、その EBGP ピア、すなわち R6 にこれを転送します。R3 には更新は送信されません。R3 がこの情報を受信する必要がある場合は、R1 と R3 が BGP ピアになるように構成する必要があります。

4.9.3 プロトコルの説明

BGP はピア間に信頼性のある TCP 接続を確立します。セッションは TCP ポート 179 を使用して確立されます。BGP はトランスポート接続がフラグメント化、再送、確認通知、および順序付けを行えることを想定しています。

2 つのスピーカーは最初に BGP セッションを築くとき、そのルーティング・テーブル全体を交換します。このルーティング情報には、各宛先に到達するために使用される完全な AS パスが含まれます。この情報は、ルーティング・ループと RIP ネットワークで観察されている無限カウント動作を防止します。一度テーブル全体が交換されたら、このテーブルに対する変更はインクリメンタル更新として伝達されます。

4.9.3.1 BGP のパケット・タイプ

BGP パケットはすべて標準ヘッダーを含んでいます。このヘッダーは BGP パケット・タイプを指定します。有効な BGP パケット・タイプは以下のとおりです。

- オープン (OPEN²): このメッセージ・タイプは、2 つのピア・ノード間に BGP セッションを確立するために使用されます。
- 更新 (UPDATE): このメッセージ・タイプは、BGP ピア間でルーティング情報を転送するために使用されます。
- 通知 (NOTIFICATION): このメッセージは、エラー状態が検出されたときに送信されます。
- キープアライブ (KEEPALIVE): このメッセージは、ピアが到達可能か判別するために使用されます。

図 83 は、2 つの自立走行式システム間でのこれらのメッセージ・タイプのフローを表しています。

² RFC 1771 は BGP メッセージの名前に英大文字を使用しています。このセクションでも同じ規則を使用しています。

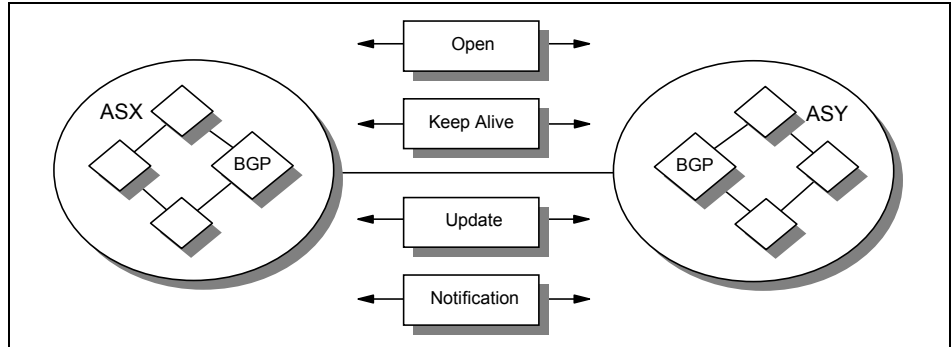


図 83 BGP スピーカー間での BGP メッセージのフロー

4.9.3.2 BGP 接続のオープンと確認

2 つのピア・ノード間に TCP セッションが確立すると、各ルーターは近隣者にオープン・メッセージを送信します。オープン・メッセージには以下が含まれます。

- 発信ルーターの AS 番号と BGP ルーター ID。
- 保持タイマーの推奨値。このタイマーの機能については、次のセクションで説明します。
- オプション・パラメーター。この情報は、ピアを認証するために使用されます。

オープン・メッセージは BGP ピアの ID の認証のサポートを含みます。しかし、BGP 標準は特定の認証メカニズムを規定していません。したがって、BGP ピアはサポートされている任意の認証方式を選択できることになっています。

オープン・メッセージの受け取りはキープアライブ・メッセージで通知されます。ピア・ルーター同士が BGP 接続を確立したら、これらは追加情報を交換できます。

4.9.3.3 BGP 接続の保守

BGP はピアが到達可能か判別するためにトランスポート・ベースのキープアライブは使用しません。その代わりに、BGP メッセージがピア間で定期的に交換されます。保持タイマーで指定された期間、ピアからメッセージを 1 つも受信しないと、発信ルーターはエラーが発生したと見なします。これが起こると、エラー通知がピアに送信され、その接続はクローズされます。

RFC 1771 では、90 秒の保持タイマーと 30 秒のキープアライブ・タイマーが推奨されています。

4.9.3.4 到達可能性情報の送信

到達可能性情報は、更新メッセージに入ってピア間で交換されます。BGP は、BGP ルーティング・テーブル全体の定期的な更新は要求していません。したがって、各 BGP スピーカーが各ピアによって使用されている現行の BGP ルーティング・テーブルのコピーを保持する必要があります。この情報は、接続の続いている間保持されます。近隣者同士が完全なルーティング情報の初期交換を実行した後は、その情報に対するインクリメンタル更新だけが交換されます。

更新メッセージは、実現可能な経路を公示したり実現不可能な経路を撤回したりするために使用されます。メッセージは、実現可能な経路の公示と、実現不可能な複数の経路のサービスからの撤回を同時に行う場合もあります。図 84 は以下の更新メッセージのフォーマットを表しています。

- ネットワーク層到達可能性情報 (NLRI) (Network Layer Reachability Information)
- パス属性 (Path Attributes) (パス属性については、197 ページの 4.9.4.1 『パス属性』で説明します)
- 撤回経路 (Withdrawn Routes)

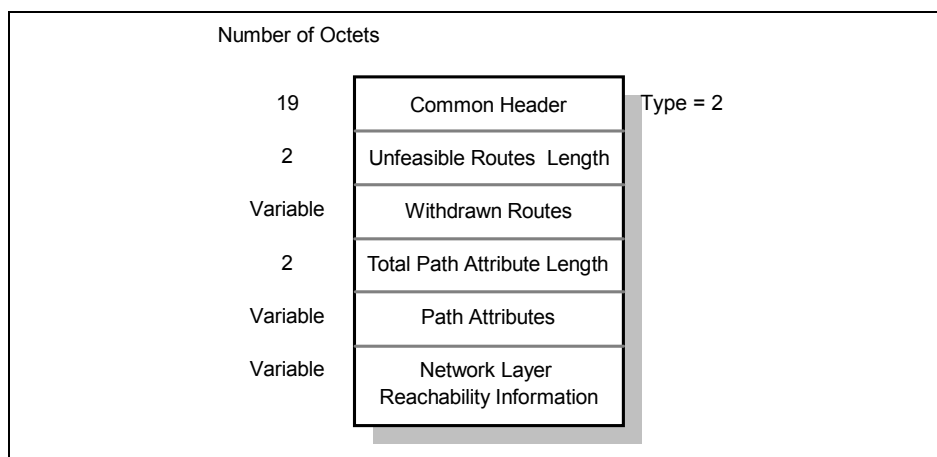


図 84 BGP の更新メッセージ

経路の説明にはいくつかのパス属性が使用される可能性があります。

撤回経路

実現不可能経路の長さ (Unfeasible Routes Length) フィールドは、撤回経路フィールドの全長を示します。

撤回経路フィールドには、実現不可能あるいはもはやサービスされていない IP アドレス接頭部のリストが示されます。これらのアドレスは BGP ルーティ

ング・テーブルから撤回される必要があります。撤回経路は、NLRI と同じタプル・フォーマットで表記されます。

4.9.3.5 エラー状態の通知

BGP 装置はピアへの接続に影響するエラー状態を監視することができます。この状態が検出されると、近隣者に通知メッセージが送信されます。このメッセージが送信されると、その BGP トランスポート接続はクローズされます。これは、この BGP 接続のすべてのリソースが割り振り解除されることを意味します。リモート・ピアに関連したルーティング・テーブル・エントリは、無効としてマークされます。最後に、これらの経路が無効であることが他のピアに通知されます。

通知メッセージには、エラー・コードとエラー・サブコードが含まれます。

BGP によって規定されているエラー・コードには、以下が含まれます。

- ・メッセージ・ヘッダー・エラー
- ・オープン・メッセージ・エラー
- ・更新メッセージ・エラー
- ・保持タイマーの満了
- ・有限状態マシン・エラー
- ・停止

エラー・サブコードは、特定のエラーをさらに限定します。各エラー・コードには、複数のサブコードが関連付けられている可能性があります。

4.9.4 パスの選択

BGP は距離ベクトル・プロトコルです。従来の距離ベクトル・プロトコルでは、単一のメトリック（たとえば、ホップ・カウント）がパスと関連付けられます。最善パスは、各実現可能経路のメトリックを比較することによって取得されます。しかし、AS 間ルーティングではこのプロセスは複雑になります。外部パスの評価に使用できる一般に合意されたメトリックはありません。各 AS は、その独自のパス評価基準セットを持ちます。

4.9.4.1 パス属性

パス属性は、パスの説明と評価に使用されます。ピアは他のルーティング情報を共にパス属性を交換します。装置は経路を公示するとき、経路をピアに公示する前にパス属性を追加または変更することができます。この属性の組み合わせが最善パスの選択に使用されます。

各パス属性は、以下の 4 つの別個のカテゴリのいずれかに入ります。

- ウェルノウン必須：この属性はすべての BGP インプリメンテーションに認識されなければなりません。これはすべての更新メッセージに入れて送信する必要があります。
- ウェルノウン任意：この属性はすべての BGP インプリメンテーションに認識されなければなりません。しかし、これはすべての更新メッセージに入れて送信する必要はありません。
- オプション通過型：すべての BGP インプリメンテーションがこのタイプの属性を認識する必要はありません。認識されていないオプションの通過型の属性を含むパスは、受信されるとそのまま他の BGP ピアに転送されます。
- オプション非通過型：すべての BGP インプリメンテーションがこのタイプの属性を認識する必要はありません。この属性は無視でき、他の BGP ピアに渡されなくても構いません。

BGP は公示経路を定義するための以下の 7 つの属性タイプを定義しています。

- 起点：この属性はパス情報の起点を定義します。有効な選択肢は IGP (AS 内部)、EGP、または INCOMPLETE です。これはウェルノウン必須属性です。
- AS_PATH: この属性は公示ネットワークに到達するためにトラバースする必要のある一連の自律システムを定義します。各 BGP 装置は、ルーティング情報を EBGp ピアに送信する前に、自分の AS 番号を AS パス・シーケンスの前に付加します。193 ページの図 82 に示されたサンプル・ネットワークで、R4 は AS_PATH が 3 のネットワーク 10.0.0.0 を公示しているとします。更新が AS 1 をトラバースすると、R2 は自分の AS 番号をこれの前に付加します。ルーティング更新が R6 に到達するとき、ネットワーク 10.0.0.0 の AS_PATH 属性は <1 3> になります。これはウェルノウン必須属性です。
- NEXT_HOP: この属性は宛先に到達するために使用されるネクスト・ホップの IP アドレスを定義します。これはウェルノウン必須属性です。

EBGP 接続を介して受信される更新の場合、ネクスト・ホップは普通リモート AS 内の EBGp 近隣者の IP アドレスになります。BGP は、このネクスト・ホップがそのまま各 IBGP 近隣者に渡されることを規定しています。そのため、各 IBGP 近隣者はリモート AS 内の近隣者に到達するための経路を保有していなければなりません。図 85 はこの相互作用を表しています。

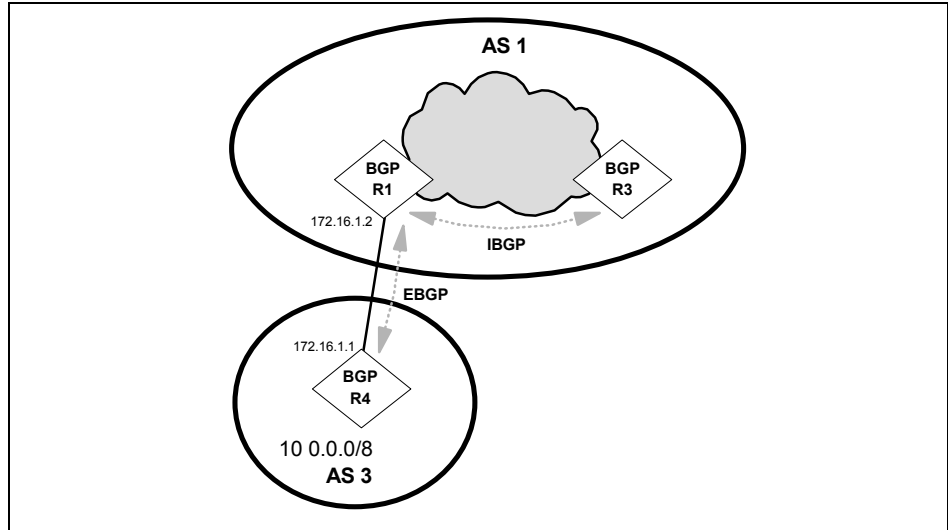


図 85 NEXT_HOP 属性

この例で、ネットワーク 10.0.0.0/8 のルーティング更新が AS 3 から送信されると、R1 は NEXT_HOP 属性が 172.16.1.1 に設定された更新を受け取ります。この更新が R3 に転送される時も、ネクスト・ホップ・アドレスは 172.16.1.1 のままです。R3 はこのアドレスに到達するための適切なルーティング情報を保有している必要があります。そうでない場合、ネクスト・ホップがアクセス不能な場合には、R3 は AS 3 宛てのパケットを除去します。

- MULTI_EXIT_DISC (複数出口識別、MED): この属性は、近隣 AS への複数の出口点を識別するために使用されます。EBGP ピアからこの情報が受信されると、これは各 IBGP ピアに伝搬されます。この属性は他の自立走行式システム内のピアには伝搬されません。他のすべての属性が等しい場合は、MED 値の最も小さい出口点が優先されます。これはオプション非通過型属性です。
- LOCAL_PREF (ローカル・プリファレンス): この属性は、BGP スピーカーが AS 内の他のスピーカーに、公示経路に対する発信スピーカーのプリファレンス度を通知するために使用します。MED とは異なり、この属性は AS 内でのみ使用されます。ローカル・プリファレンスの値は、AS 外部には配布されません。他のすべての属性が等しい場合は、プリファレンス度が最も高い経路が優先されます。これはウェルノウン任意属性です。
- ATOMIC_AGGREGATE: この属性は、BGP ピアが、複数の一致しない経路で示された、同じ経路への複数の公示 (つまり、重複経路) を受け取った場合に使用されます。一方の経路は小さな宛先のセット (より限定的な

接頭部)について説明し、他方の経路は大きな宛先のセット(あまり限定的でない接頭部)について説明しています。この属性は、BGP スピーカーがピアに、より限定的な経路を選択しないであまり限定的でない経路のほうを選択したことを通知するために使用します。これはウェルノウン任意属性です。

この属性を含む経路は、実際には AS_PATH にリストされていない自立走行式システムをトラバースする可能性があります。

- AGGREGATOR: この属性では、集約経路を構成した最後の AS 番号と、集約経路を構成した BGP スピーカーの IP アドレスがそれに続いて示されます。経路集約の詳細については、202 ページの 4.9.6 『BGP 集約』にあります。これはオプション通過型属性です。

4.9.4.2 決定プロセス

最善パスを選択するためのプロセスでは、各経路を説明するパス属性が使用されます。属性は分析され、*プリファレンス度*が割り当てられます。指定の宛先に対して複数のパスが存在する可能性があるので、経路選択プロセスでは各実現可能経路のプリファレンス度が判別されます。プリファレンス度の最も高いパスは最善パスとして選択されます。これが各 BGP 近隣者に公示されるパスになります。

このプロセス中には、経路集約を実行することもできます。

宛先への複数のパスが存在する場合、BGP は個々のそれぞれのパスをトラッキングします。これによって、基本パスが故障した場合の代替パスへの収束はより高速になります。

4.9.5 BGP 同期

図 86 は、中継サービスを提供する AS の例を表しています。この例で、AS 1 は AS 3 と AS 4 の間でトラフィックをトランスポートするために使用されています。AS 1 内で、R2 は BGP 用に構成されていません。しかし、R2 は R1 と R3 の間の通信に使用されています。これら 2 つの BGP ノード間のトラフィックは、物理的に R2 をトラバースします。

前述のルーティング更新フローで、10.0.0.0/8 ネットワークは R4 と R1 の間の EBGP 接続を使用して公示されているとします。R1 はその既存の IBGP 接続を使用してネットワーク公示を R3 に渡します。R2 は BGP 用に構成されていないので、AS 3 内のいずれのネットワークも認識しません。問題は、R3 が AS 3 内の装置と通信する必要がある場合に生じます。R3 はトラフィックを R2 に渡します。しかし R2 は AS 3 ネットワークへの経路を保有していないので、このトラフィックは除去されます。

R3 が 10.0.0.0/8 ネットワークを AS 4 に公示すると、問題は続きます。AS 4 が AS 3 内の装置と通信する必要がある場合、パケットは R5 から R3 に転送されます。R3 はパケットを R2 に転送し、そこでこれは破棄されます。

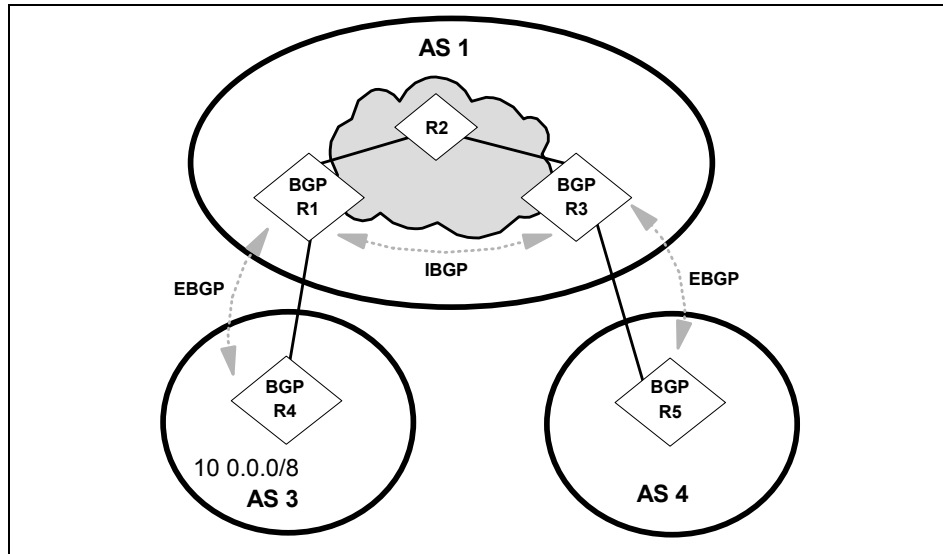


図 86 BGP 同期

この状態は、BGP の同期規則で取り上げられています。この規則によれば、中継 AS はその AS 内のすべてのルーターがその経路について学習するまでは経路の公示を行いません。この例でいうと、R3 は、R2 が正しいルーティング・テーブルを構築するまでは AS 3 内のネットワークの存在を公示しません。同期規則のインプリメント方式には、以下の 3 つがあります。

- 中継 AS 内のすべての装置を BGP 対応にする。このソリューションでは、R2 が R1 および R3 の両方との IBGP セッションを持つことになります。R2 が 10.0.0.0/8 ネットワークを認識すると、それと同時に R3 にこれが公示されます。この時、R3 は AS 4 内のそのピアに経路を通知します。
- 中継エリア内で使用されている IGP に経路を再配布する。このソリューションでは、R1 が 10.0.0.0/8 ネットワークを AS 1 内の IGP に再配布します。R3 は、BGP と IGP の 2 つのルーティング・プロトコルを介してネットワークを認識します。R3 が IGP を介してネットワークを認識したら、必ずその AS 内の他のルーターもその経路を認識していることになります。この時、R3 は AS 4 内のそのピアに経路を通知します。
- AS を越える中継トラフィックをカプセル化する。このソリューションでは、中継トラフィックが、出口ゲートウェイにアドレッシングされた IP データグラム内にカプセル化されます。この場合、IGP が外部ルーティン

グ情報を伝送する必要はないので、BGP と IGP の間で同期は必要ありません。R3 は即時に経路を AS 4 内のそのピアに通知することができます。

4.9.6 BGP 集約

BGP バージョン 4 に導入された主な改善点は、CIDR のサポートと経路集約です。これらのフィーチャーにより、BGP ピアは複数の連続するルーティング・エントリーを単一の公示の中に統合することができます。これにより、大規模なインターネットワーク環境への BGP のスケーラビリティは大きく向上しています。これらのフィーチャーは、図 87 で示しています。

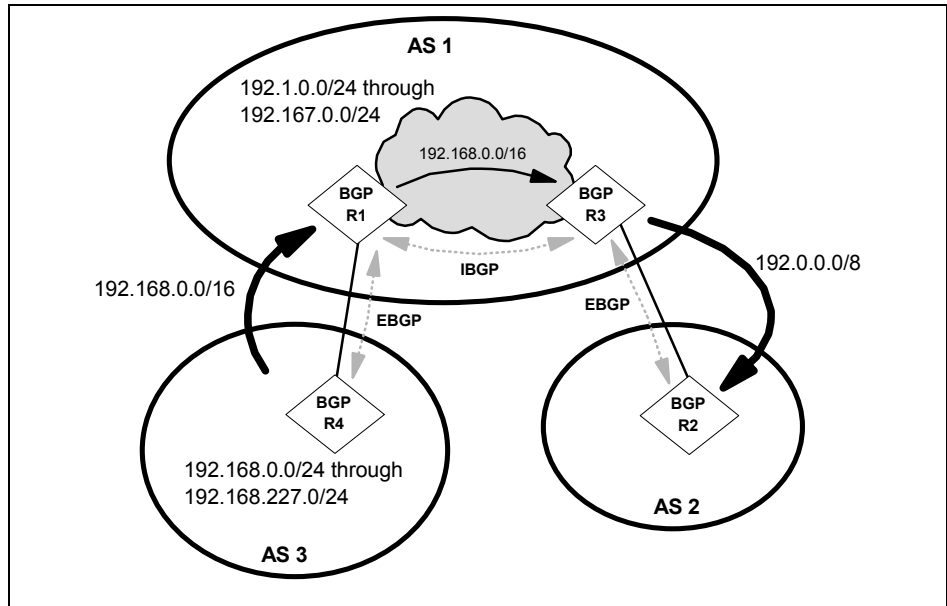


図 87 BGP 経路集約

この図は、BGP を介して相互接続された 3 つの自律システムを表しています。この例では、ネットワーク 192.168.0.0 から 182.168.227.0 まだが AS 3 内にあります。ルーティング通知のサイズを小さくするため、R4 はこれらの個々のネットワークを単一の経路エントリーに集約してから AS 1 に公示します。単一のエントリー 192.168.0.0/16 は、正しくないクラス C ネットワークですが、有効な CIDR スーパーネットを表します。

BGP 集合経路は、その AS_PATH パス属性内に追加情報を含みます。より限定的な経路のセットから集約エントリーが生成されるときは、より限定的なその経路の AS_PATH 属性が結合されます。たとえば図 87 では、集約経路 192.0.0.0/8 は AS 1 から AS 2 に通知されます。この集約は、AS 1 および AS 3 内に配置されているより限定的な経路のセットを表します。この集約経路が

AS 2 に送信されるとき、AS_PATH パス属性には <1 3> が含まれます。これはルーティング情報ループを防ぐために行われます。AS 1 が AS_PATH 属性が <1> の集約を生成すれば、ループが発生する可能性があります。AS 2 が AS 3 への直接接続を持つ場合、AS 1 から公示されるあまり限定的でない AS_PATH を持つ経路はループを生成する可能性があります。これは、AS 2 がこの集約に AS 3 内にあるネットワークが含まれていることを認識していないためです。

4.9.7 BGP 同盟

BGP では、単一の AS 内のすべてのスピーカーが完全メッシュの IBGP 接続のセットを持っている必要があります。このことは多数の IBGP ピアを含むネットワークでのスケーリング問題になる可能性があります。この問題に対処するには、BGP 同盟を使用します。

BGP 同盟は、同盟の外のピアには単一の AS に見える自律システムのセットを作成します。これによって完全メッシュの必要はなくなり、管理の複雑さは軽減されます。

図 88 は BGP 同盟の操作を表しています。このサンプル・ネットワークで、AS 1 には 8 つの BGP スピーカーが含まれています。標準の BGP ネットワークの場合、このスピーカーを完全メッシュにするには 28 の IBGP セッションが必要です。

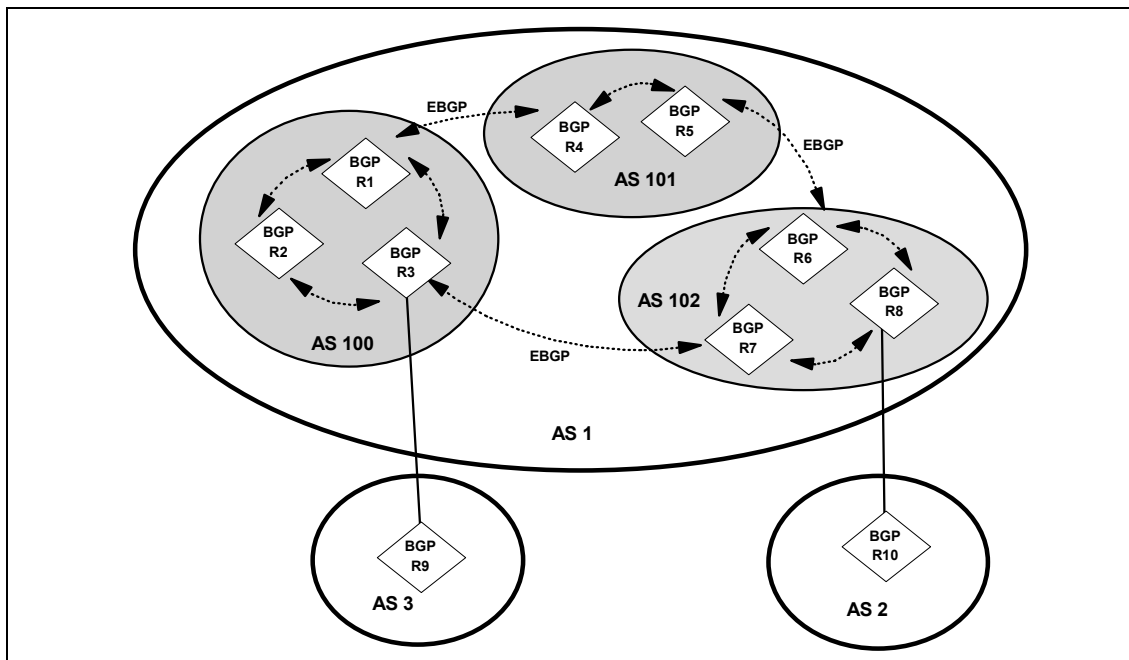


図 88 BGP 同盟

同盟では、AS は一連のドメインに分かれています。この例では、AS 1 に 3 つのドメインが含まれています。ドメイン内の装置は、完全メッシュの IBGP 接続のセットを持ちます。各ドメインはまた同盟内の他のドメインへの EBGP 接続も持ちます。このネットワーク例では、R1、R2、および R3 は完全メッシュの IBGP セッションを持ちます。R1 は R4 への同盟内の EBGP セッションを持ちます。R3 は R9 への同盟外の EBGP セッションを持ちます。

同盟内の各ルーターには、同盟 ID が割り当てられます。同盟のメンバーは、同盟外の装置とのすべての通信でこの ID を使用します。この例では、各ルーターに AS 1 の同盟 ID が割り当てられています。AS 1 から AS 2 または AS 3 へのすべての通信は、同盟 ID AS 1 から発信しているように見えます。

同盟内のドメイン間の通信は EBGP で行われますが、ドメインはあたかも IBGP を介して接続されているかのようにルーティング更新を交換します。具体的には NEXT_HOP、MULTI_EXIT_DESC、および LOCAL_PREF 属性に含まれている情報がドメインの間で保持されます。同盟は他の自立走行式システムには単一の AS に見えます。

BGP 同盟は、RFC 3065 で説明されています。本書の執筆時点では、これは提案標準です。それでも、BGP 同盟はインターネットにおいて広く配備されています。多くのベンダーがこのフィーチャーをサポートしています。

4.9.8 BGP 経路リフレクター

経路リフレクターは、AS 内のピア間の完全メッシュの IBGP セッションの要件に対処するためのもう 1 つのソリューションです。前述のとおり、BGP スピーカーが IBGP ピアから更新を受け取ると、この受信スピーカーはその更新を EBGP ピアにだけ伝搬します。この受信スピーカーは他の IBGP ピアには更新を転送しません。経路リフレクターはこの制限を緩和します。BGP スピーカーは、IBGP で認識した経路を特定の IBGP ピアに公示することを許可されます。

図 89 は、経路リフレクターを利用している環境を表しています。R1 は R2 および R3 の経路リフレクターとして構成されています。R2 および R3 は R1 の経路リフレクター・クライアントになります。R2 と R3 の間には IBGP セッションは定義されていません。R3 が AS 3 から EBGP 更新を受け取ると、それは IBGP を使用して R1 に渡されます。R1 はリフレクターとして構成されているので、R1 はその IBGP 更新を R2 に転送します。

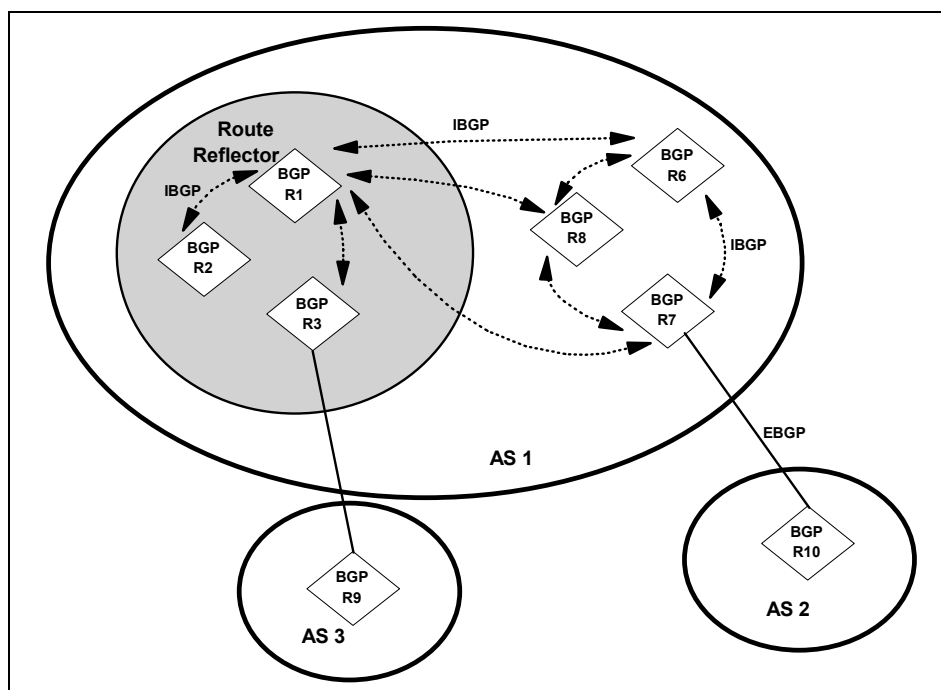


図 89 BGP 経路リフレクター

図 89 も、AS 内の経路リフレクターと従来の BGP スピーカーの間の相互作用を表しています。この図で、R1、R2、および R3 は経路リフレクター・ドメイン内にあります。R6、R7、および R8 は、完全メッシュの IBGP ピア接続を含む従来の BGP スピーカーです。さらに、これらの各スピーカーは経路リフレクターのピアです。この構成では、AS 1 内で完全な IBGP 通信を行うことができます。

図 89 には表示されていませんが、AS には複数の経路リフレクターを含めることができます。その場合、各リフレクターは他のリフレクターを従来の IBGP ピアとして扱います。

経路リフレクターは、RFC 2796 で説明されています。本書の執筆時点では、これは提案標準です。

4.10 ルーティング・プロトコルの選択

ルーティング・プロトコルの選択は、ネットワーク管理者の主要な決定です。これは、ネットワーク・パフォーマンス全体に大きく影響します。選択はネットワークの複雑さ、サイズ、および管理ポリシーに基づきます。あるタイプのネットワーク用に選択されたプロトコルは、他のタイプのネットワークには適さない可能性があります。固有なそれぞれの環境を、以下のようないくつかの基本的な設計要件に照らして評価する必要があります。

- 大規模な環境へのスケーラビリティ：ネットワークが大きくなる可能性がある場合は、この要件が重要になります。大規模な冗長度の高いネットワークのサポートが必要な場合は、リンク状態またはハイブリッド・アルゴリズムを考慮すべきです。距離ベクトル・アルゴリズムでは、このような環境へと拡張することはできません。
- 障害時の安定度：距離ベクトル・アルゴリズムは障害期間中のネットワークの安定性をもたらす可能性があります。無限カウント問題 (154 ページの 4.3.5 『収束と無限カウント』) は、ルーティング・ループまたは他の最適でないルーティング・パスを発生させる可能性があります。リンク状態またはハイブリッド・アルゴリズムではこの問題が発生する可能性は小さくなります。
- 収束の速度：トリガー更新は、障害が検出されたときに即時に収束を開始するための機能を提供します。3 つのすべてのタイプのプロトコルがこのフィーチャーをサポートしています。収束に寄与する要因の 1 つは、障害の検出に必要な時間です。OSPF および EIGRP ネットワークでは、収束が始まる前に必ず一連の Hello パケットが失われます。RIP 環境では、収束が開始される前に必ず後続の経路公示が失われます。これらの検出時間は、通信の復元に必要な時間を増加させます。

- **メトリック**：メトリックにより、ネットワーク上の適切なルーティング・パスを調整することができます。リンク状態アルゴリズムは経路の計算時に帯域幅を考慮します。EIGRP ではこれが改善され、経路計算にネットワーク遅延が含まれます。
- **VLSM のサポート**：IP アドレス範囲を使用する可能性がある場合は、この要件が重要になります。アドレスの供給が限られた環境では、ネットワーク管理者はネットワークを賢くオーバーレイするアドレッシング方式を開発する必要があります。VLSM はこの計画の主要コンポーネントになります。専用アドレス範囲を使用することによって、この問題に対処することもできます。
- **ベンダー・インターオペラビリティ**：ネットワークに配置される装置のタイプによっては、この要件が重要になります。ネットワークに複数のベンダーの装置が含まれている場合は、標準のルーティング・プロトコルを使用すべきです。IETF は、本書で説明されている距離ベクトルおよびリンク状態アルゴリズムの操作ポリシーを指定しています。これらのアルゴリズムをインプリメントすれば、標準でないプロトコルで発生するインターオペラビリティ問題は防止されます。
- **インプリメンテーションの容易さ**：距離ベクトル・プロトコルは、構成と保守の最も簡単なルーティング・プロトコルです。この理由から、このプロトコルには、インプリメント可能な環境が最も多く存在します。これらの環境で問題解決を実行するには、訓練はそれほど必要ありません。

小さく変化のない環境では、静的経路もまたインプリメントが簡単です。これらの定義は、サイトがネットワークに追加、あるいはそこから除去されるときにだけ変わります。

環境に適したルーティング・プロトコルを判別する際、管理者はこれらの各要件の重要性を見積もる必要があります。

4.11 ルーターが実行する追加機能

ルーターが実行する主要な機能は、IP ルーティング・テーブルの管理とデータの転送に関連しています。しかし、ルーターはネットワーク問題の可能性を他の装置に警告する情報を提供できなければなりません。この情報は、105 ページの 3.2 『Internet Control Message Protocol (ICMP)』で説明されている ICMP プロトコルを介して提供されます。この情報には以下が含まれます。

- **ICMP 宛先到達不能**：IP パケットで指定されている宛先アドレスが、不明な IP ネットワークを指しています。
- **ICMP リダイレクト**：宛先へのパス上のより適切なルーターへのトラフィックのリダイレクト転送。

- ICMP 始点抑制：宛先へのパス上の装置で輻輳問題（たとえば、使用可能バッファ・スペースに対する過剰な着信データグラム）が発生しています。
- ICMP 時間超過：IP データグラムの存続時間フィールドがゼロに達しています。パケットを最終宛先に送達することができません。

さらに、各 IP ルーターは以下の基本 ICMP 操作およびメッセージをサポートしている必要があります。

- パラメーター問題：IP ヘッダーに問題が見付かった場合は、パケットの送信元にこのメッセージが戻されます。このメッセージは、問題のタイプと場所を示します。エラーのあるパケットはルーターが破棄します。
- アドレス・マスク要求 / 応答：ルーターは ICMP アドレス・マスク要求メッセージを受信して、ICMP アドレス・マスク応答メッセージで応答するためのサポートをインプリメントしている必要があります。
- タイム・スタンプ：ルーターは受信したすべてのタイム・スタンプ・メッセージに対してタイム・スタンプ応答を戻す必要があります。これは、遅延の差が最も小さくなるように設計されている必要があります。ルーターのクロックを同期するには、UDP Time Server Protocol か Network Time Protocol (NTP) を使用することができます。
- エコー要求 / 応答：ルーターは、ルーターに送信された要求を受信して対応する応答を送信する ICMP エコー・サーバー機能をインプリメントしている必要があります。ルーターは IP ブロードキャストまたは IP マルチキャスト・アドレスにアドレッシングされた ICMP エコー要求を無視する選択をすることもできます。

4.12 UNIX ベース・プロセスでのルーティング・プロセス

この章では、標準の IP ルーターで使用可能なプロトコルに焦点を当てています。しかし、これらのプロトコルのいくつかは、UNIX ベース・システムでも使用可能です。これらのプロトコルは、普通以下の 2 つのプロセスのいずれかを使用してインプリメントされます。

- routed (ルート D と読みます)：これは内部ルーティングのための基本ルーティング・プロセスです。これは大半の TCP/IP インプリメンテーションで提供されています。これは RIP プロトコルです。
- gated (ゲート D と読みます)：これは内部および外部ルーティングを可能にするより洗練されたプロセスです。これは OSPF、RIP-2、および BGP-4 を含むいくつかのプロトコルをインプリメントすることができます。

第 5 章 トランスポート層プロトコル

この章では、TCP/IP トランスポート層の最も重要で一般的なプロトコルについて概説します。これには、以下が含まれます。

- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)

トランスポート層プロトコルは、Internet Protocol (IP) によって提供される機能の上に構築され、IP ホストで実行するアプリケーションにデータを送達します。これは、ポート (209 ページの 5.1『ポートとソケット』で説明) を使用することによって行われます。トランスポート・プロトコルは、輻輳制御、信頼性のあるデータ送達、重複データ抑止、および TCP によって行われるようなフロー制御などの追加機能を提供することもできます。

5.1 ポートとソケット

このセクションでは、ポートとソケットの概念を紹介します。これらは、特定のホストのどのローカル・プロセスが実際にどのプロセスと、どのリモート・ホストで、どのプロトコルを使用して通信しているのかを判別するために必要です。これが分かりにくく思われる場合は、次のように考えてください。

- アプリケーション・プロセスにはプロセス ID 番号 (プロセス ID) が割り当てられます。これは、そのプロセスの開始ごとに異なる可能性があります。
- プロセス ID はオペレーティング・システム・プラットフォームごとに異なるため、これらは一様ではありません。
- サーバー・プロセスは一度に複数のクライアントに対する複数の接続を持つことができるので、単純な接続 ID は固有にはなりません。

ポートとソケットの概念を用いれば、これらを使用する接続とプログラムとホストを特定のプロセス ID の別なく一様に、また一意的に識別することができます。

5.1.1 ポート

他のプロセスと通信しようとする各プロセスは、1 つ以上のポートで TCP/IP プロトコル・スイートに対して自分自身を示します。ポートは、host-to-host プロトコルが着信メッセージをどの上位レベル・プロトコルまたはアプリケーション・プログラム (プロセス) に送達する必要があるかを判別するために使用する 16 ビットの番号です。ポートには以下のような 2 つのタイプがあります。

- ウェルノウン：ウェルノウン・ポートは標準サーバーに属します。たとえば、Telnet はポート 23 を使用します。ウェルノウン・ポート番号の範囲は 1 と 1023 の間 (1992 年より前は、256 と 1023 の間の範囲は UNIX 固有サーバー用に使用されていた) になります。ウェルノウン・ポート番号は基本的に奇数です。これは、ポート概念を使用した初期のシステムが、二重動作のためにポートの奇数 / 偶数ペアを必要としたためです。大半のサーバーは単一ポートだけを必要とします。例外は、67 と 68 の 2 つの使用する BOOTP サーバー (125 ページの 3.6 『Bootstrap protocol (BOOTP)』を参照) と、20 と 21 の 2 つを使用する FTP サーバー (383 ページの 10.1 『ファイル転送プロトコル (FTP)』を参照) です。

ウェルノウン・ポートは、Internet Assigned Number Authority (IANA) によって制御されて割り当てられ、大半のシステムでは、特権ユーザーによって実行されるシステム・プロセスまたはプログラムでのみ使用できます。ウェルノウン・ポートの目的は、クライアントが構成情報がなくてもサーバーを見つけられるようにすることです。ウェルノウン・ポート番号は、STD 2 – Assigned Internet Numbers で定義されています。

- 一時：クライアントはサーバーとの通信を開始し、使用しているポート番号はそのサーバーに送信される UDP データグラムに含まれるので、ウェルノウン・ポート番号を知る必要はありません。各クライアント・プロセスには、必要な場合、それが実行しているホストによってポート番号が割り当てられます。一時ポート番号は 1023 より大きい (普通は 1024 から 65535 までの範囲の) 値になります。<transport protocol, IP address, port number> が固有である限り、クライアントは割り当てられるどんな値でも使用できます。

一時ポートは IANA によって制御されず、大半のシステム上の普通のユーザー開発プログラムで使うことができます。

2 つの異なるアプリケーションが 1 つのホスト上で同じポート番号を使用することによって生じる混乱を防止するには、これらのアプリケーションが TCP/IP に使用可能ポートを要求するように作成します。このポート番号は動的に割り当てられるので、アプリケーションを起動するたびに異なる可能性があります。

UDP、TCP、および ISO TP-4 はすべて同じポート原則を使用します。可能な限り UDP、TCP、および ISO TP-4 の上の同じサービスには同じポート番号が使用されます。

注：

通常、サーバーは TCP か UDP のいずれかを使用しますが、例外はあります。たとえば、ドメイン・ネーム・サーバー (291 ページの 8.1『Domain Name System (DNS)』を参照) は、UDP ポート 53 と TCP ポート 53 の両方を使用します。

5.1.2 ソケット

ソケット・インターフェースは、通信プロトコルに対するいくつかのアプリケーション・プログラミング・インターフェース (API) の 1 つです。汎用の通信プログラミング・インターフェースとして設計された API は初め 4.2 BSD によって紹介されました。標準化はされていませんが、これは業界のデファクト・スタンダードになっています。

4.2 BSD は Internet と UNIX という 2 つの異なる通信ドメインを認めていました。4.3 BSD では Xerox Network System (XNS) プロトコルが追加されており、4.4 BSD では ISO OSI プロトコルをサポートするための拡張インターフェースが追加される予定です。

以下の用語を考慮しましょう。

- ソケットは、プロセスがオペレーティング・システムにネットワーク・サービスを要求するために使用する、特殊なタイプのファイル・ハンドルです。
- ソケット・アドレスは以下の 3 つの組です。
<protocol, local-address, local-process>
たとえば、TCP/IP スイートでは次のようになります。
<tcp, 193.44.234.3, 12345>
- 会話とは、2 つのプロセス間の通信リンクです。
- アソシエーションは、接続を構成する 2 つのプロセスを完全に指定する以下の 5 つから成るタプルです。
<protocol, local-address, local-process, foreign-address, foreign-process>
TCP/IP スイートでの有効なアソシエーションの例は以下のとおりです。
<tcp, 193.44.234.3, 1500, 193.44.234.5, 21>
- ハーフ・アソシエーションは以下のいずれかになります。
<protocol, local-address, local-process>
または

<protocol, foreign-address, foreign-process>

これらは接続を半分ずつを指定します。

- ハーフ・アソシエーションは、ソケットまたはトランスポート・アドレスとも呼ばれます。つまりソケットとは、ネットワークで命名してアドレッシングできる通信用のエンドポイントです。

2つのプロセスは TCP ソケット を介して通信します。ソケット・モデルは、プロセスに他のプロセスへの全二重バイト・ストリーム接続を提供します。アプリケーションはこのストリームの管理に携わる必要はありません。その機能は TCP によって提供されます。

TCP は UDP を同じポート原則を使用して多重化を行います。UDP と同様、TCP はウェルノウン・ポートと一時ポートを使用します。TCP 接続のそれぞれの側はタプル <TCP, IP address, port number> で識別できるソケットを持ちます。2つのプロセスが TCP を介して通信している場合、これらは関連する2つのソケット、つまり <TCP, local IP address, local port, remote IP address, remote port> の組み合わせによって一意的に識別できる論理接続を持ちます。サーバー・プロセスは、単一ポートを通して複数の会話を管理することができます。ソケット API の詳細については、273 ページの 7.2.1 『ソケット API』を参照してください。

5.2 User Datagram Protocol (UDP)

UDP は STD 番号 6 の標準プロトコルです。UDP は、RFC 768 – User Datagram Protocol で説明されています。この状態は「推奨 (Recommended)」ですが、実際には、ルーティング専用に使われていない TCP/IP インプリメンテーションはすべて UDP を含みます。

UDP は基本的には IP に対するアプリケーション・インターフェースです。これは IP に信頼性、フロー制御、またはエラー・リカバリーは追加しません。これは単純に図 90 のようにポートを使用してデータグラムを送信する、データグラムの送受信のためのマルチプレクサー/デマルチプレクサーとして働きます。ポートの詳細な説明については、209 ページの 5.1 『ポートとソケット』を参照してください。

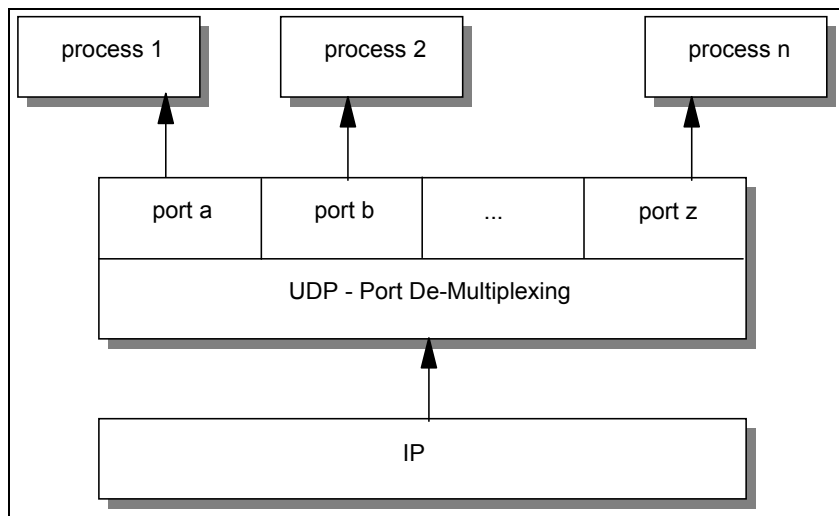


図 90 UDP - ポートに基づく多重化

UDP は、アプリケーションが他のアプリケーションにデータグラムを送信するためのメカニズムを提供しています。UDP 層は非常に薄いものと見なすことができ、したがってオーバーヘッドは大きくありませんが、エラー・リカバリーなどの責任はアプリケーションが負う必要があります。

データグラムは通常システム全体ではなく特定のプロセスに送信されるので、ホストにデータグラムを送信するアプリケーションは IP アドレスより限定的な宛先を示す必要があります。UDP はポートを使用することによって、これを行います。ポートの概念は、209 ページの 5.1 『ポートとソケット』で説明しています。

5.2.1 UDP データグラム・フォーマット

各 UDP データグラムは単一の IP データグラムに入って送信されます。IP データグラムは伝送時にフラグメント化されることがありますが、受信 IP インプリメンテーションはそれを UDP 層に提供する前に再組み立てします。すべての IP インプリメンテーションは 576 バイトのデータグラムを受け入れる必要があります。最大サイズの IP ヘッダーの 60 バイトを考慮に入れると、このことは、すべてのインプリメンテーションが 516 バイトの UDP データグラムを受け入れられることを意味します。多くのインプリメンテーションはもっと大きなデータグラムを受け入れますが、それは保証されていません。UDP データグラムは、図 91 に示されている 16 バイトのヘッダーを持ちます。

Source Port	Destination Port
Length	Checksum
Data...	

図 91 UDP - データグラム・フォーマット

- ・送信元ポート：送信プロセスのポートを示します。これは、応答の宛先にすべきポートです。
- ・宛先ポート：宛先ホスト上の宛先プロセスのポートを示します。
- ・長さ：ヘッダーを含む、このユーザー・データグラムの長さ（バイト単位）。
- ・チェックサム：偽 IP ヘッダー、UDP ヘッダー、および UDP データの 1 の補数合計である、オプションの 16 ビットの 1 の補数。偽 IP ヘッダーには、送信元と宛先の IP アドレス、プロトコル、および UDP 長が含まれます。

Source IP address		
Destination IP address		
Zero	Protocol	UDP Length

図 92 UDP - 偽 IP ヘッダー

偽 IP ヘッダーにより、チェックサムは事実上元の（フラグメント化されていない）IP データグラムを含むまでに拡張されます。

5.2.2 UDP アプリケーション・プログラミング・インターフェース

UDP が提案するアプリケーション・インターフェースは、RFC 768 で説明されています。これは以下を規定しています。

- ・新しい受信ポートの作成。
- ・データ・バイト、および送信元ポートと送信元 IP アドレスの指示を戻す受信操作。
- ・データ、送信元、および宛先ポートとアドレスをパラメーターとして持つ送信操作。

このインターフェースのインプリメント方法は、各ベンダーの裁量に任されています。

UDP と IP は保証付きの送達、フロー制御、またはエラー・リカバリーを提供しておらず、したがって、これらはアプリケーションによって提供されなければならないという点に注意してください。

UDP を使用する標準アプリケーションには、以下が含まれます。

- Trivial File Transfer Protocol (TFTP)
- Domain Name System (DNS) ネーム・サーバー
- Network File System (NFS) で使用されるリモート・プロシーチャー・コール (RPC)
- Simple Network Management Protocol (SNMP)
- Lightweight Directory Access Protocol (LDAP)

5.3 Transmission Control Protocol (TCP)

TCP は STD 番号 7 の標準プロトコルです。TCP は、RFC 793 – Transmission Control Protocol で説明されています。この状態は「推奨 (Recommended)」ですが、実際には、ルーティング専用に使われていない TCP/IP インプリメンテーションはすべて TCP を含みます。

TCP は UDP と比べてかなり多くのアプリケーション用機構、すなわちエラー・リカバリー、フロー制御、および信頼性を提供しています。コネクションレスの UDP とは異なり、TCP は *接続指向* のプロトコルです。Telnet および FTP などのユーザー・アプリケーション・プロトコルの大半は TCP を使用します。

図 93 に示されているように、2 つのプロセスは TCP 接続を介して互いと通信 (InterProcess 通信 - IPC) します。ポートとソケットの詳細については、209 ページの 5.1 『ポートとソケット』を参照してください。

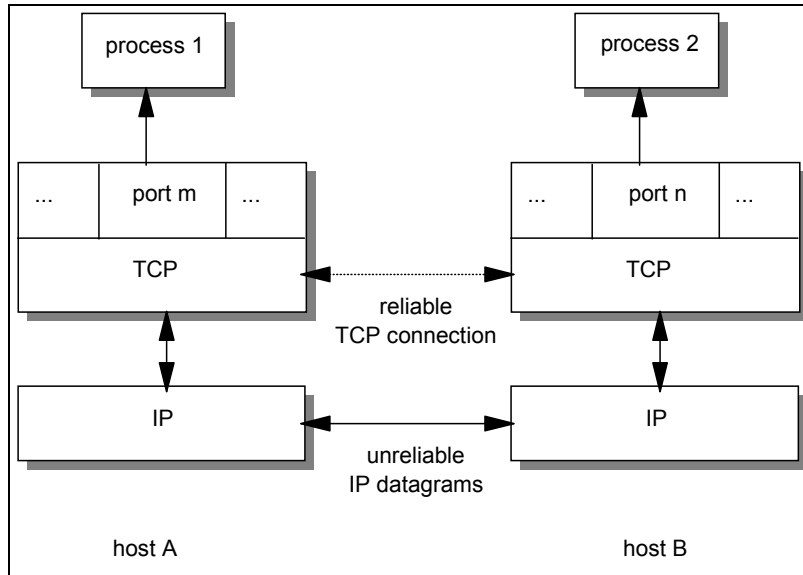


図 93 TCP - プロセス間の通信 - (プロセス 1 と 2 は、IP データグラムが媒介する TCP 接続を介して通信する)

5.3.1 TCP の概念

前述のとおり、TCP の主要な目的は、プロセスのペアの間に信頼性のある論理回路または接続サービスを提供することです。低レベル・プロトコル (IP など) からの信頼性を受け継がないので、TCP は自分でこれを保証しなければなりません。

TCP の特徴は、これを使用するアプリケーションに対して提供される以下の機能です。

- ・ ストリーム・データ転送 : アプリケーションの視点からすると、TCP は連続するバイトのストリームをネットワークに転送します。アプリケーションは、データを基本ブロックまたはデータグラムに分割するという手間は必要ありません。TCP は TCP セグメント内のバイトをグループ化することによってこれを行い、それが宛先への伝送のために IP に渡されます。また、TCP 自身がデータの分割方法を決定し、自分の都合の良いときにデータを転送することができます。

アプリケーションは、TCP に渡されたすべてのデータが宛先へ実際に伝送済みであることの確認が必要な場合があります。そのために、プッシュ機能が定義されています。これは、まだストレージ内にあるすべての残りの TCP セグメントを宛先ホストにプッシュします。通常の接続クローズ機能もデータを宛先にプッシュします。

- 信頼性：CP は伝送される各バイトにシーケンス番号を割り当て、受信 TCP からの肯定確認通知 (ACK) を待ちます。タイムアウト間隔内に ACK が受信されないと、そのデータは再送されます。データはブロック (TCP セグメント) で伝送されるので、セグメント内の最初のデータ・バイトのシーケンス番号だけが宛先ホストに送信されます。

受信 TCP はこのシーケンス番号を使用して、セグメントの順番が狂って到着したときにそれを再配置したり、重複セグメントを除去したりすることができます。

- フロー制御：受信 TCP はまた、ACK を送信側に返送するとき、最後に受信した TCP セグメント以外にこれがオーバーランやその内部バッファのオーバーフローを引き起こさずに受信することのできるバイト数を送信側に示します。これは、これが問題なく受信することのできる最大のシーケンス番号の形式で ACK に入って送信されます。このメカニズムはウィンドウ・メカニズムとも呼ばれ、これについては後にこの章でより詳しく説明します。
- 多重化：UDP の場合と同様、ポートの使用を通して行われます。
- 論理接続：前述の信頼性とフロー制御メカニズムのため、TCP は各データ・ストリームの状況情報を初期化して保守する必要があります。ソケット、シーケンス番号、およびウィンドウ・サイズを含むこの状態の組み合わせは、論理接続と呼ばれます。各接続は、送信および受信プロセスが使用するソケットのペアによって一意的に識別されます。
- 全二重：TCP は同時データ・ストリームを両方向で提供します。

5.3.1.1 ウィンドウ原則

単純なトランスポート・プロトコルは、「パケットを送信したら、次のパケットを送信する前に受信側からの確認通知を待ち、特定の時間内に ACK を受信しなかったらそのパケットを再送する」という原則を使用している可能性があります。詳細については、図 94 を参照してください。

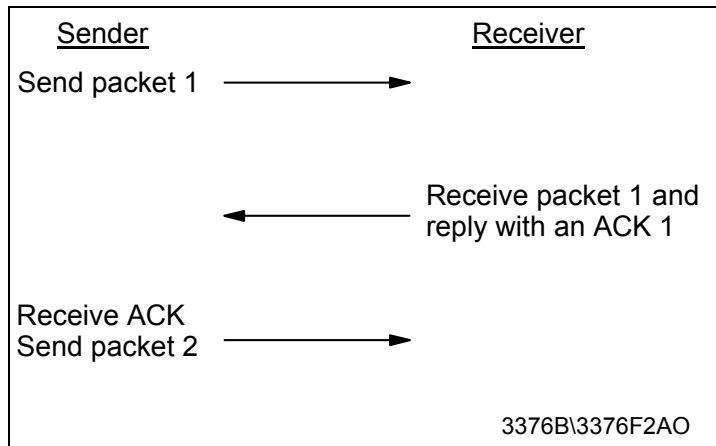


図 94 TCP - ウィンドウ原則

このメカニズムは信頼性を保証し、使用可能なネットワーク帯域幅の一部だけを使用します。

ここで、送信側が伝送するそのパケットを図 95 のようにグループ化し、以下の規則を使用するプロトコルについて考えてください。

- 送信側は ACK を受け取らずにウィンドウ内のすべてのパケットを送信することができるが、そのたびにタイムアウト・タイマーを開始しなければならない。
- 送信側は、最後に正常に受信したパケットのシーケンス番号を示して、受信した各パケットの受け取りを通知しなければならない。
- 送信側は、各 ACK の受信ごとにウィンドウをスライドさせる。

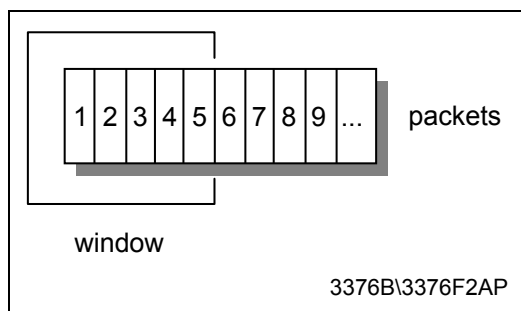


図 95 TCP - メッセージ・パケット

この例では、受信側はパケット 1 から 5 までを確認通知を待たずに伝送できます。

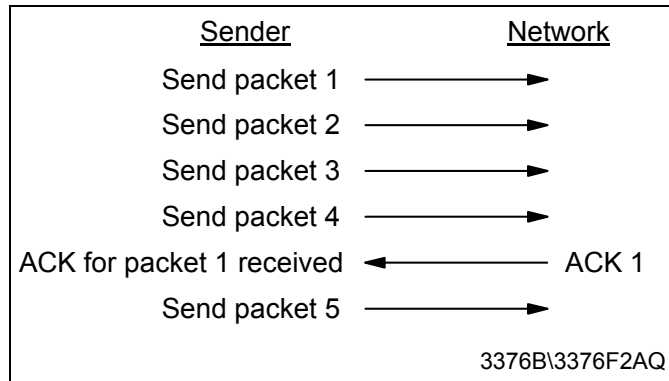


図 96 TCP - ウィンドウ原則

送信側は ACK 1 (パケット 1 の確認通知) の受信と同時に、そのウィンドウを右に 1 パケット分スライドさせることができます。

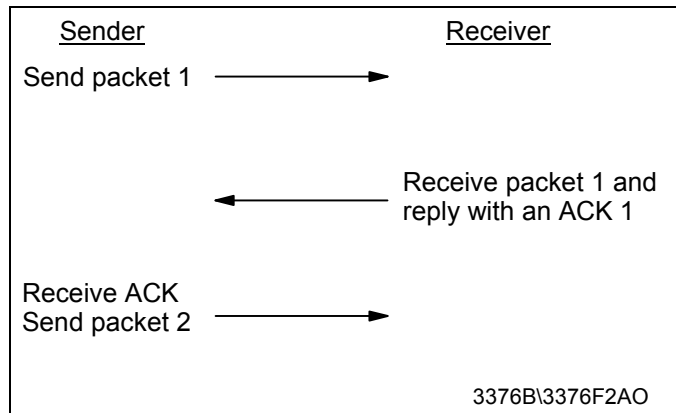


図 97 TCP - メッセージ・パケット

この時点で、送信側はパケット 6 を伝送することができます。

以下のようないくつかの特殊な場合について考えましょう。

- ・パケット 2 が逸失した：送信側は ACK 2 を受信しないので、このウィンドウは 1 の位置に残ります (図 97 を参照)。実際、受信側はパケット 2 を受け取らなかったため、これはパケット 3、4、および 5 の受け取りを ACK 1 で通知します。なぜなら、パケット 1 が順序正しく受信された最後のパケットだったからです。送信側では、ようやくパケット 2 のタイムアウト

が発生し、これが再送されます。このパケットが受信側に受信されると、1 から 5 までのすべてのパケットが正常に受信されたことになるので ACK 5 が生成され、この ACK 5 の受信と同時に送信側のウィンドウは 4 つスライドするという点に注意してください。

- パケット 2 が到着したのに確認通知が逸失した：送信側は ACK 2 は受信しませんが、ACK 3 は受信します。ACK 3 は、3 までのすべてのパケット（パケット 2 を含む）の確認通知なので、この後送信側はそのウィンドウをパケット 4 までスライドさせることができます。

このウィンドウ・メカニズムは以下を保証します。

- 信頼性のある伝送。
- ネットワーク帯域幅の効率的な使用（スループットの向上）。
- フロー制御。受信側は、その空きバッファが使用可能であることと通信のウィンドウ・サイズを理解して、パケットに対する確認通知での応答を遅らせることができるため。

5.3.1.2 TCP に適用されているウィンドウ原則

TCP では上記のウィンドウ原則が使用されていますが、以下のようないくつかの相違点があります。

- TCP はバイト・ストリーム接続を提供しているので、ストリーム内の各バイトにはシーケンス番号が割り当てられています。TCP はこの連続するバイト・ストリームを TCP セグメントに分割して伝送します。ウィンドウ原則はバイト・レベルで使用される、つまり、送信されるセグメントと受信される ACK はバイトのシーケンス番号を伝送し、ウィンドウ・サイズはパケットの数ではなくバイトの数として表されます。
- ウィンドウ・サイズは、接続が確立したときに受信側によって判別され、データ転送中に変化します。各 ACK メッセージには、その特定時に受信側に取り扱い可能なウィンドウ・サイズが入ります。

送信側のデータ・ストリームはこの場合以下のようになります。

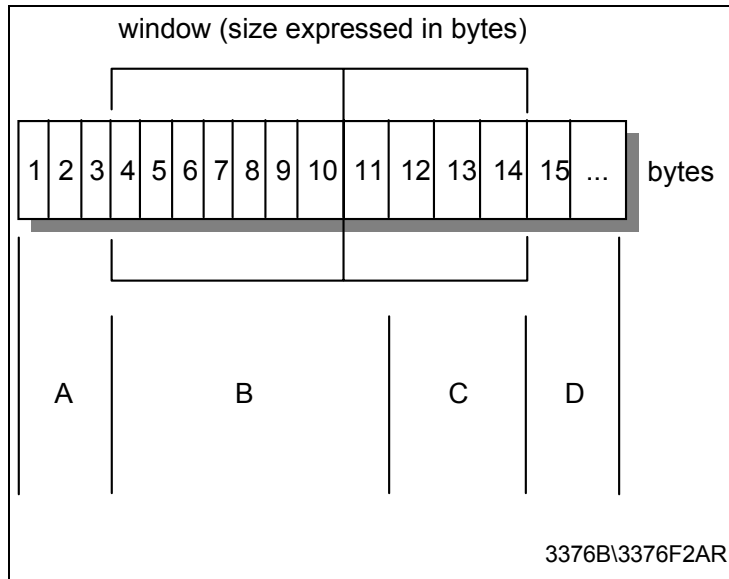


図 98 TCP - TCP に適用されているウィンドウ原則

- A: 伝送され、受信が確認通知されているバイト。
- B: 送信されているが、まだ受信が確認通知されていないバイト。
- C: 確認通知を待たずに送信できるバイト。
- D: まだ送信できないバイト。

TCP はバイトをブロック化してセグメントにし、TCP セグメントはセグメント内の最初のバイトのシーケンス番号だけを伝送するという点を思い出してください。

5.3.1.3 TCP セグメント・フォーマット

図 99 は、TCP セグメント・フォーマットを表しています。

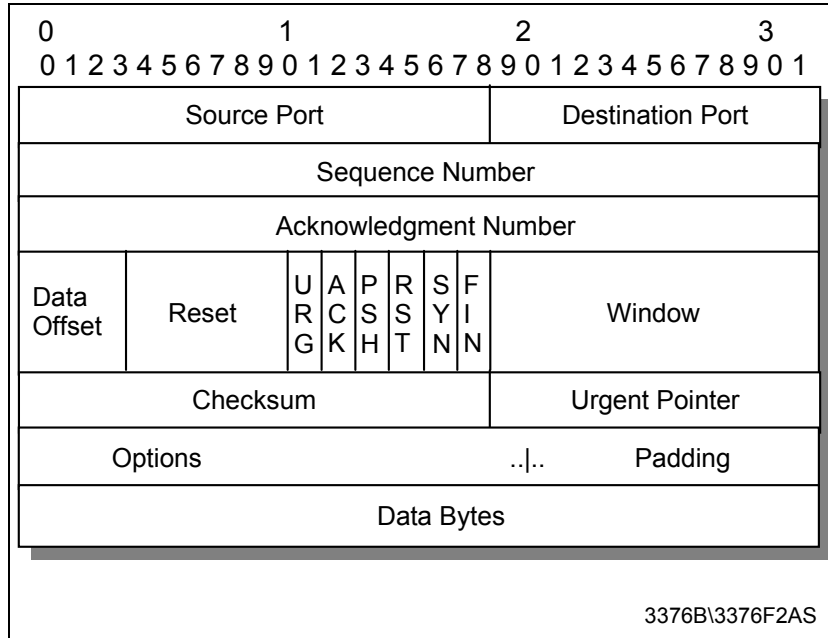


図 99 TCP - セグメント・フォーマット

- 送信元ポート：受信側が応答に使用する、16 ビットの送信元のポート番号。
- 宛先ポート：16 ビットの宛先のポート番号。
- シーケンス番号：このセグメント内の最初のデータ・バイトのシーケンス番号。SYN 制御ビットが設定されている場合、シーケンス番号は初期シーケンス番号 (n) で、最初のデータ・バイトは n+1 です。
- 確認通知番号：このフィールドには、ACK 制御ビットが設定されている場合に、受信側が受信する予定である次のシーケンス番号の値が入ります。
- データ・オフセット：TCP ヘッダー内の 32 ビット・ワードの数。これはデータが開始する場所を示します。
- 予約済み：将来の利用のために予約されている 6 ビット。これは必ず 0 です。
- URG: このセグメント内の緊急ポインター・フィールドが有効であることを示します。
- ACK: このセグメント内の確認通知フィールドが有効であることを示します。

- PSH: プッシュ機能。
- RST: 接続をリセットします。
- SYN: シーケンス番号を同期します。
- FIN: 送信側からのデータはこれ以上なし。
- ウィンドウ: ACK セグメントで使用されます。これは、受信側 (= このセグメントの送信側) に受信する用意ができている、確認通知番号フィールドで示されたデータ・バイトから始まるデータ・バイトの数を示します。
- チェックサム: 偽 ヘッダー、TCP ヘッダー、および TCP データ内のすべての 16 ビット・ワードの 1 の補数合計である、16 ビットの 1 の補数。チェックサムの計算の際、チェックサム・フィールド自体はゼロと見なされます。

偽ヘッダーは、UDP がチェックサムの計算に使用するものと同じです。これはチェックサムの計算にだけ使用され、図 100 のようなフォーマットを持つ偽 IP ヘッダーです。

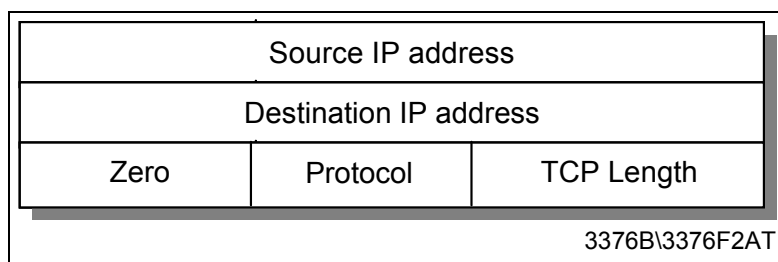


図 100 TCP - 偽 IP ヘッダー

- 緊急ポインター: 緊急データの後に続く最初のデータ・オクテットを指します。URG 制御ビットが設定されている場合のみ有効です。
- オプション: IP データグラム・オプションと同様で、オプションは以下のいずれかになります。
 - オプション番号を含む単一バイト。
 - 以下のフォーマットの可変長オプション。

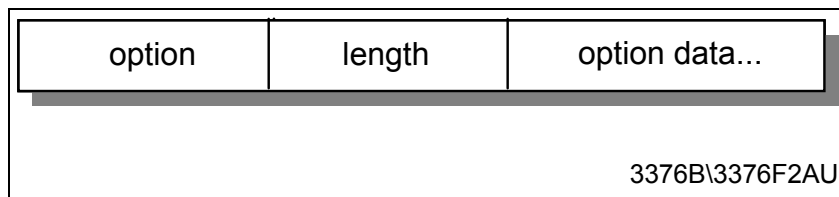


図 101 TCP - IP データグラム・オプション - 可変長オプション

現在のところ以下の 7 つのオプションが定義されています。

表 2 TCP - IP データグラム・オプション

種類	長さ	意味
0	-	オプション・リストの終了
1	-	ノーオペレーション
2	4	最大セグメント・サイズ
3	3	ウィンドウ・スケール
4	2	SACK 使用可能
5	X	SACK
8	10	タイム・スタンプ

- 最大セグメント・サイズ・オプション：このオプションは接続の確立に際しのみ使用され (SYN 制御ビットが設定されている場合)、データを受信する側が処理できる最大セグメント長を示すために送信します。このオプションが使用されていない場合、セグメント・サイズは自由です。詳細については、図 102 を参照してください。

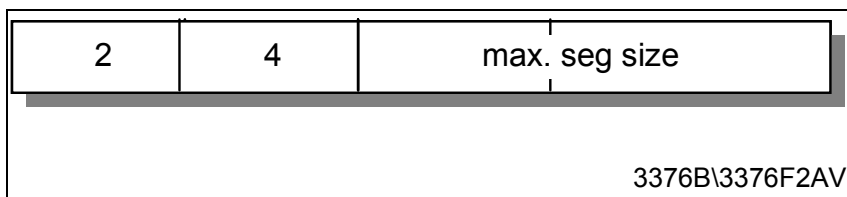


図 102 TCP - 最大セグメント・サイズ・オプション

- ウィンドウ・スケール・オプション：このオプションは必須ではありません。両方向でウィンドウ・スケーリングを使用可能にするには、両方の側がウィンドウ・スケーリング・オプションをその SYN セグメントに入れて送信する必要があります。ウィンドウ・スケールは TCP ウィンドウの定義を 32 ビットにまで拡張します。これは、標準の 16 ビット・ウィンドウ・サイズを超える SYN セグメント内でスケール因数を使用することによって、32 ビット・ウィンドウ・サイズを定義します。受信側は 16 ビット・ウィンドウ・サイズとスケール要因を使用して 32 ビット・ウィンドウ・サイズを再ビルドします。このオプションはハンドシェイク時に判別されます。接続が確立した後にこれを変更することはできません。詳細については、図 103 を参照してください。

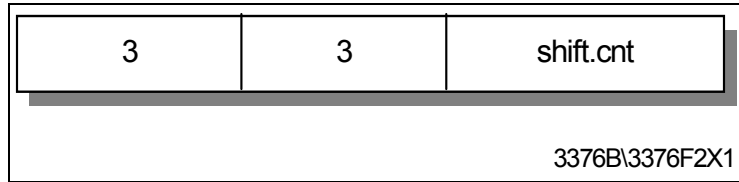


図 103 TCP - ウィンドウ・スケール・オプション

- SACK 許可オプション：このオプションはその TCP 接続で選択確認通知が使用されている場合に設定されます。詳細については、図 104 を参照してください。

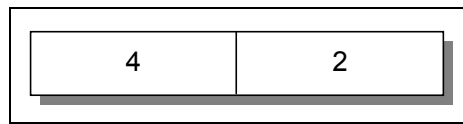


図 104 TCP - SACK 許可オプション

- SACK オプション：選択確認通知 (SACK) は、受信側が正常に受信されたすべてのセグメントについて送信側に通知することを許可します。したがって、送信側は実際に逸失したセグメントだけを送信します。最後の SACK 以来逸失したセグメントの数が多過ぎると、SACK オプションは大きくなり過ぎます。このため、SACK オプションが報告できるブロックの数は 4 つに限定されています。これを削減するため、SACK オプションは最新の受信データに使用すべきです。詳細については、図 105 を参照してください。

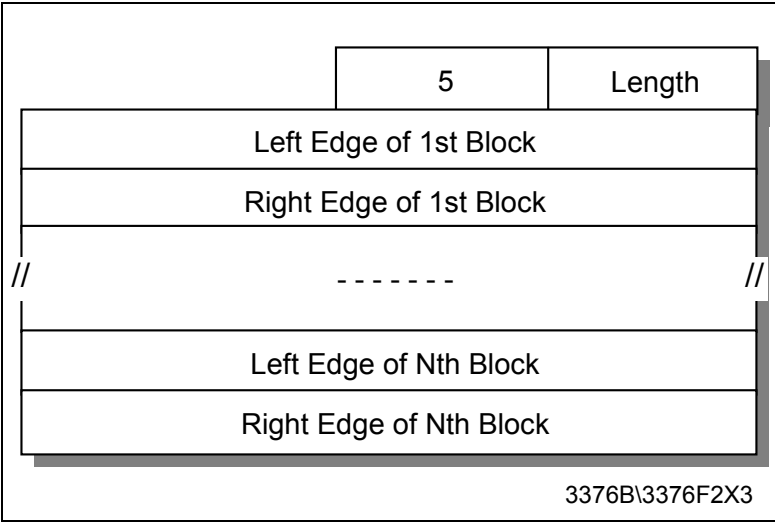


図 105 TCP - SACK オプション

- タイム・スタンプ・オプション：タイム・スタンプ・オプションは、このオプションを送信する TCP のタイム・スタンプ・クロックの現行値を示すタイム・スタンプ値を送信します。タイム・スタンプ・エコー値は、TCP ヘッダー内で ACK ビットが設定されている場合にのみ使用できます。詳細については、図 106 を参照してください。

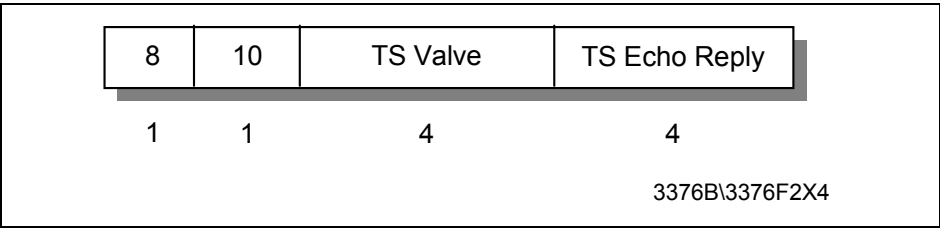


図 106 TCP - タイム・スタンプ・オプション

- ・ 埋め込み：TCP ヘッダーを、32 ビットの倍数である全長まで埋め込むために、すべてゼロのビットが使用されます。

5.3.1.4 確認通知と再送

TCP は可変長のセグメントでデータを送信します。シーケンス番号はバイト・カウントに基づきます。確認通知は、送信側が受信する予定である次のバイトのシーケンス番号を示します。

セグメントが逸失または破損したことを考えてください。この場合、受信側はその後正常に受信するすべてのセグメントに、未着パケットの最初のバイトを指し示す確認通知で応答します。送信側はウィンドウ内のすべてのバイトを送信すると伝送を止めます。最終的には、タイムアウトが発生して未着パケットは再送されます。

図 107 は、1500 バイトのウィンドウ・サイズと 500 バイトのセグメントが使用されている例を表しています。

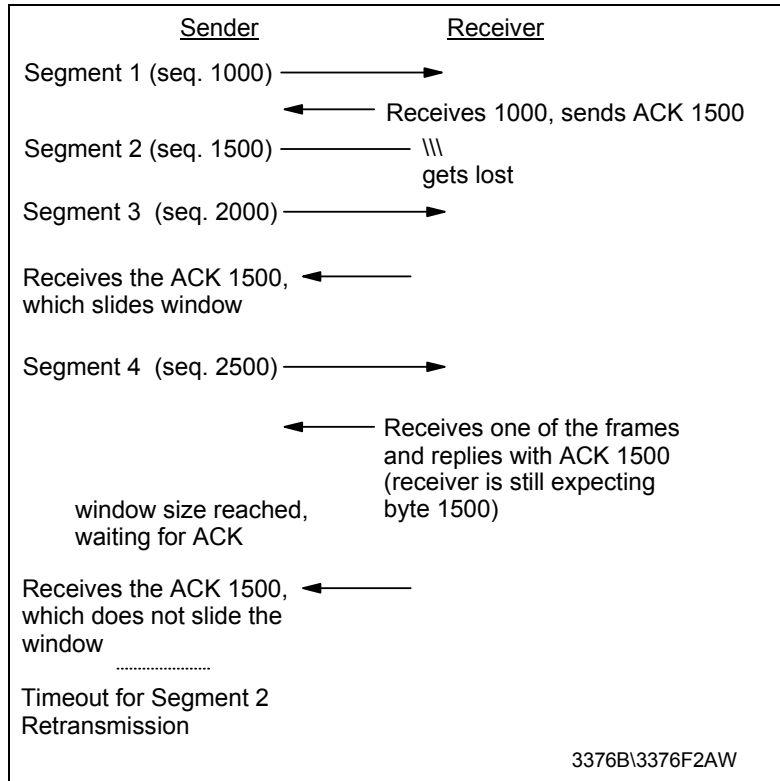


図 107 TCP - 確認通知と再送のプロセス

この時、送信側はセグメント 2 が逸失または破損していることは分かっていますが、セグメント 3 と 4 については何も分からないために、問題が発生します。送信側は少なくともセグメント 2 は再送しなければなりません、さらにセグメント 3 と 4 も再送できます (これらはウィンドウ内にあるため)。以下の可能性があります。

1. セグメント 3 は受信済みですが、セグメント 4 については分からない。これは受信されたのに ACK がまだ届いていないのかもしれませんが、逸失しているのかもしれませんが。

2. セグメント 3 が逸失し、セグメント 4 の受信で ACK 1500 を受け取った。

各 TCP インプリメンテーションは、インプリメント方針に応じて自由にタイムアウトに反応します。これはセグメント 2 だけを再送する可能性があります。2 つ目のケースでは、セグメント 3 がタイムアウトになるまで再び待つことになります。この場合、ウィンドウ・メカニズムのスループットの利点はすべて失われます。TCP はあるいは、現行ウィンドウ内のすべてのセグメントを即時に再送する可能性もあります。

どう選択しても、最大のスループットは失われます。これは、受信された実際のフレームを示す 2 番目の確認通知のシーケンス番号が ACK に含まれていないためです。

可変タイムアウト間隔

各 TCP は、セグメントの往復時間に使用するタイムアウト値を調整するためのアルゴリズムをインプリメントしている必要があります。このために、TCP はセグメントが送信された時間と、ACK が受信される時間を記録します。これらの往復時間のいくつかから、次に送信されるセグメントのタイムアウト値として使用される加重平均が計算されます。

IP ネットワーク内の遅延は、中間の低速ネットワークの負荷や中間 IP ゲートウェイの浸透などの複数の要因によってさまざまに異なるので、これは重要なフィーチャーです。

5.3.1.5 TCP 接続の確立

データが転送できるようになるには、2 つのプロセス間に接続が確立されなければなりません。一方のプロセス (通常のサーバー) は *受動オープン・コール* を、もう一方は *能動オープン・コール* を発行します。受動オープン・コールは、他のプロセスが能動オープンでこれに接続することを試みるまで休止したままになります。

ネットワーク上では、以下の 3 つの TCP セグメントが交換されます。

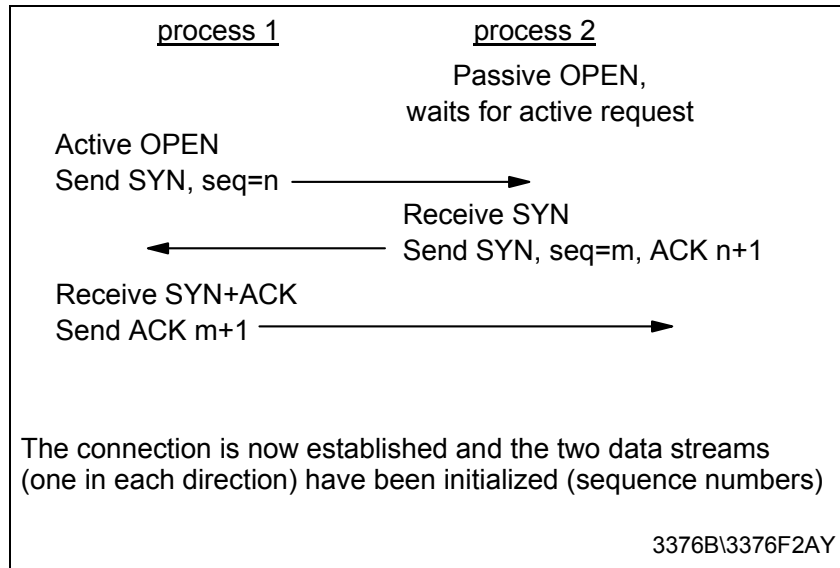


図 108 TCP - 接続の確立

このプロセス全体は、3 ウェイ・ハンドシェークと呼ばれます。交換される TCP セグメントには、後のデータ転送で使用する、両方の側の初期シーケンス番号が含まれることに注意してください。

接続のクローズは、FIN ビット (データはもうない) が設定された TCP セグメントを送信することによって暗黙的に行われます。接続は全二重 (つまり、各方向に 1 つずつの、2 つの独立したデータ・ストリームが存在する) なので、FIN セグメントは 1 つの方向のデータ転送だけをクローズします。もう一方のプロセスはこの後まだ伝送する必要がある残りのデータを送信し、同じく FIN ビットの設定された TCP セグメントで終了します。両方向のデータ・ストリームがクローズされると、接続 (と両方の側の状況情報) は削除されます。

5.3.2 TCP アプリケーション・プログラミング・インターフェース

TCP アプリケーション・プログラミング・インターフェースは完全には定義されていません。これが提供すべきいくつかの基本機能だけが RFC 793 – Transmission Control Protocol で説明されています。TCP/IP プロトコル・スイート内の大半の RFC の場合と同様に、かなりの程度の自由が開発者にゆだねられており、効率の向上 (スループットの増大) につながる最適な (オペレーティング・システムに依存した) インプリメンテーションを実現できます。

この RFC では、以下の関数呼び出しが説明されています。

- ・ オープン：接続を確立するために、以下のようないくつかのパラメーターを取ります。
 - 能動 / 受動
 - 外部ソケット
 - ローカル・ポート番号
 - タイムアウト値 (オプション)
- これは、他のすべての関数の中でこの特定の接続を参照するために使用されるローカル接続名を戻します。
- ・ 送信：参照されているユーザー・バッファ内のデータを接続を介して送信します。オプションで URGENT フラグまたは PUSH フラグを設定することができます。
- ・ 受信：着信 TCP データをユーザー・バッファにコピーします。
- ・ クローズ：接続をクローズします。残りのすべてのデータと FIN フラグの設定された TCP セグメントをプッシュします。
- ・ 状況：以下のような情報を戻すインプリメンテーションに依存した呼び出し。
 - ローカルおよび外部ソケット
 - 送信および受信ウィンドウ・サイズ
 - 接続状態
 - ローカル接続名
- ・ 打ち切り：保留にされているすべての送信および受信操作を打ち切り、外部 TCP に「リセット」を送信します。

詳細は、RFC 793 – Transmission Control Protocol にあります。

5.3.3 TCP 輻輳制御アルゴリズム

TCP と UDP の間の 1 つの大きな違いは、輻輳制御アルゴリズムです。TCP 輻輳アルゴリズムは、送信側がネットワーク (たとえば、低速の WAN リンク) の容量をオーバーランするのを防止します。TCP は送信側の速度をネットワーク容量に適合させて、潜在的な輻輳状態を回避することができます。TCP と UDP の間の違いを理解するのに、基本的な TCP の輻輳制御アルゴリズムを理解していることは非常に役立ちます。

長年に渡って、いくつかの輻輳制御拡張が TCP に追加および提案されてきました。これは現在でも活発な進展中の研究分野ですが、近年の TCP のインプリメンテーションには、以下の 4 つの関連し合うアルゴリズムが基本のインターネット標準として含まれています。

- ・ スロー・スタート

- 輻輳回避
- 高速再送
- 高速リカバリー

5.3.3.1 スロー・スタート

古い TCP のインプリメンテーションが接続を開始するとき、送信側は受信側に公示されたウィンドウ・サイズ以内の複数のセグメントをネットワークに送り出します。2 つのホストが同じ LAN 上にある場合これは OK ですが、送信側と受信側の間にルーターと低速のリンクがある場合は問題が発生する可能性があります。一部の中間ルーターはこれを処理することができず、パケットは除去され、その結果、再送が行われてパフォーマンスは低下します。

これを防止するためのアルゴリズムはスロー・スタートと呼ばれます。これは、ネットワークに新しいパケットが送り出されるべき速度が、もう一方の終端によって確認通知が戻される速度と同じであることを監視することによって働きます。スロー・スタートは、送信側の TCP にもう 1 つのウィンドウ、つまり `cwnd` と呼ばれる輻輳ウィンドウを追加します。他のネットワーク上のホストとの新しい接続が確立されると、この輻輳ウィンドウが 1 つのセグメント（たとえば、もう一方の終端によって通知されたセグメント・サイズや、通常 536 か 512 であるデフォルト）に初期設定されます。ACK が受信されるたびに、輻輳ウィンドウはセグメント 1 つ分ずつ拡大します。送信側は、輻輳ウィンドウまたは公示ウィンドウより小さな値を伝送することができます。輻輳ウィンドウは送信側によって課されたフロー制御で、公示ウィンドウは受信側によって課されたフロー制御です。前者は認知されたネットワーク輻輳の送信側の評価に基づき、後者はこの接続の受信側の使用可能バッファ・スペース量に関連します。

送信側は 1 つのセグメントを伝送してその ACK を待つことによって開始します。この ACK が受信され、輻輳ウィンドウが 1 から 2 に増分すると、2 つのセグメントが送信できるようになります。これらの 2 つのそれぞれのセグメントが確認通知されると、輻輳ウィンドウは 4 にまで拡大します。これは指数関数的に拡大していきます。ただし、受信側は通常、受信する 2 つのセグメントごとに 1 つの ACK を送信してその ACK を遅らせる可能性があるため、これは正確には指数関数ではありません。

ある時点で、IP ネットワーク（たとえば、低速の WAN リンク）の容量に達し、中間ルーターがパケットを破棄し始める可能性があります。これはこの輻輳ウィンドウが大きくなり過ぎていることを送信側に示します。作動中のスロー・スタートの概要については、図 109 を参照してください。

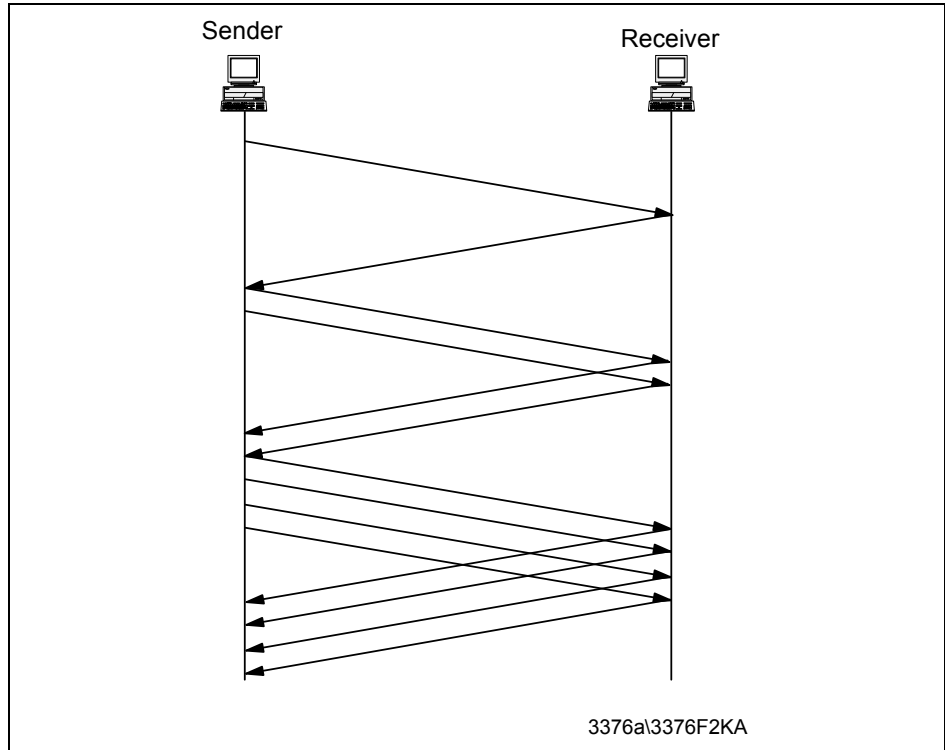


図 109 作動中の TCP スロー・スタート

5.3.3.2 輻輳回避

このアルゴリズムの前提事項は、損傷によって引き起こされるパケットの逸失が非常に小さいもの (1 % 未満) であることです。したがって、パケットの逸失は、送信元と宛先の間のネットワーク内のどこかの輻輳を示唆します。パケットの逸失は、以下の 2 つの事柄を示します。

1. タイムアウトが発生した。
2. 重複 ACK を受信した。

輻輳制御とスロー・スタートは、異なる目的を持つ独立したアルゴリズムです。しかし輻輳が発生したときは、TCP はそのネットワークへのパケットの伝送速度を低下させ、スロー・スタートを呼び出して再び事を進める必要があります。実際に、これらは共にインプリメントされます。

輻輳回避とスロー・スタートでは、各接続ごとに以下の 2 つの変数が保持されている必要があります。

- 輻輳ウィンドウ、cwnd

- ・ スロー・スタートのしきい値サイズ、ssthresh

結合されたアルゴリズムは以下のように働きます。

1. 特定の接続の初期設定では、cwnd が 1 つのセグメントに、ssthresh が 65535 バイトに設定されます。
2. TCP 出力ルーチンは、cwnd または受信側の公示ウィンドウより小さい値以外は送信しません。
3. 輻輳 (タイムアウトまたは重複 ACK) が発生すると、現行ウィンドウ・サイズの半分が ssthresh に保管されます。さらに、タイムアウトによって輻輳が示されている場合は、cwnd が 1 つのセグメントに設定されます。
4. もう一方の終端が新しいデータの受信を通知すると、cwnd は拡大します。ただし、この拡大方法は TCP がスロー・スタートと輻輳回避のどちらを実行しているかによって異なります。cwnd が ssthresh 以下であれば TCP はスロー・スタートであり、そうでなければ TCP は輻輳回避を実行しています。

フロー・スタートは TCP が輻輳の発生時にいた場所に達する途中まで続き (なぜならこれはステップ 2 で問題の原因になったウィンドウ・サイズの半分を記録しています)、その後は輻輳回避が引き継ぎます。フロー・スタート cwnd を 1 つのセグメントで開始させ、ACK が受信されるたびに 1 セグメントずつ増分させます。前述のとおり、これはウィンドウを指数関数的に開いていきます。つまり、セグメントを 1 つ、その次に 2 つ、その次に 4 つというように送信します。

輻輳回避は、ACK が受信されるたびに cwnd が $\text{segsz} \times \text{segsz} / \text{cwnd}$ ずつ増分するよう指示します (segsz はセグメント・サイズで、cwnd はバイト単位で保持されます)。スロー・スタートの指数関数的増加と比べると、この cwnd の増加は線形的です。cwnd の拡大は、最大でも各往復ごとに 1 セグメント (その往復回にいくつの ACK を受信したかには関係ない) ですが、スロー・スタートは 1 往復回に受信した ACK の数だけ cwnd を拡大します。多くのインプリメンテーションは、輻輳回避中に誤ってセグメント・サイズの小さな断片 (通常は 8 で割ったセグメント・サイズ) を追加します。これは誤りであり、将来のリリースでエミュレートされるべきではありません。作動中の TCP のスロー・スタートと輻輳回避の例については、図 110 を参照してください。

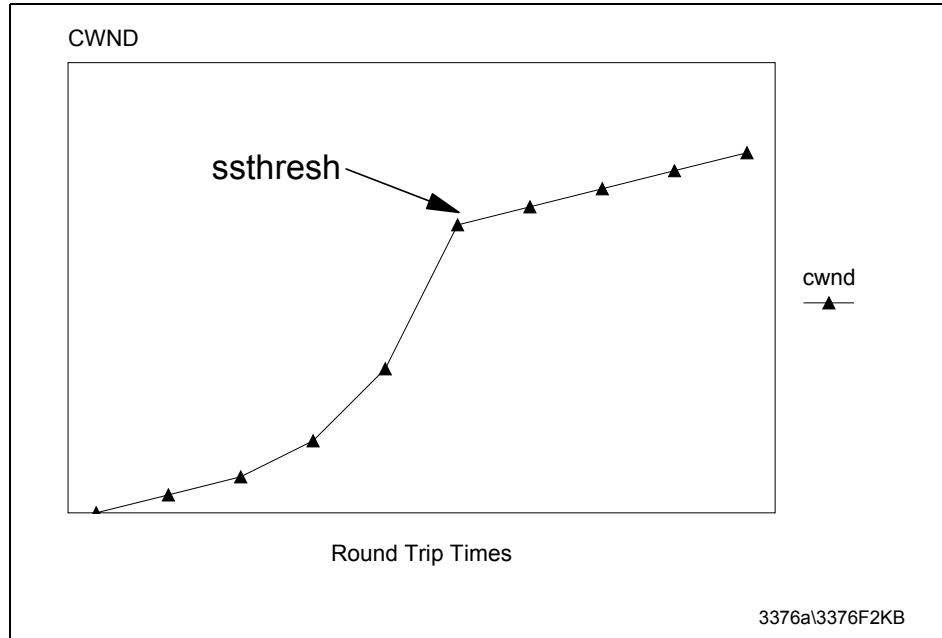


図 110 作動中の TCP のスロー・スタートと輻輳回避の動作

5.3.3.3 高速再送

高速再送は、TCP にタイムアウトを待たせずに逸失セグメントを再送させます。

輻輳回避アルゴリズムに対する修正は 1990 年に提案されました。この変更の説明の前に、TCP は順番の狂ったセグメントを受信するときに、即時確認通知 (重複 ACK) を生成する可能性があることを銘記しておいてください。この重複 ACK は遅れるべきではありません。この重複 ACK の目的は、もう一方の終端に順番の狂ったセグメントを受信したことを知らせ、予想されるシーケンス番号をこれに示すことです。

TCP は、重複 ACK が逸失セグメントによって発生したのか、それとも単なるセグメントの再配列によって発生したのかが分からないので、少数の重複 ACK が受信されるのを待ちます。これがただのセグメントの再配列であれば、再配列されたセグメントが処理される前に重複 ACK が 1 つか 2 つだけ生成され、その後で新しい ACK が生成されるはずですが、3 つ以上の重複 ACK が連続して受信された場合、それはそのセグメントが逸失していることを強く示すものとなります。TCP はその後、再送タイマーが満了するのを待たずに、未着セグメントと思われるものの再送を実行します。作動中の TCP 高速再送の概要については、図 111 を参照してください。

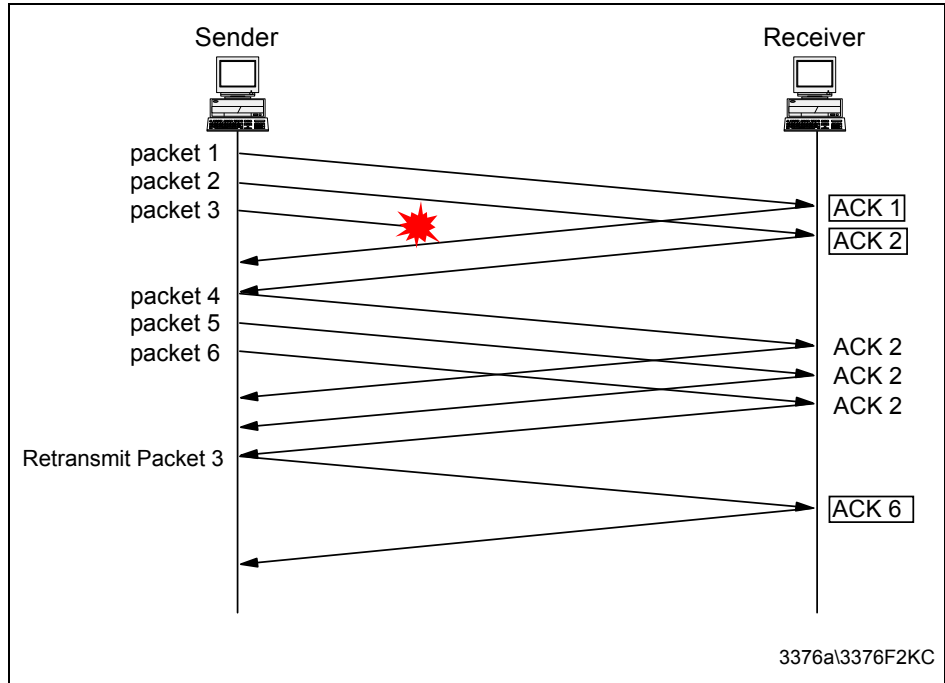


図 111 作動中の TCP 高速再送

5.3.3.4 高速リカバリー

高速再送が未着セグメントと思われるものを送信した後は、スロー・スタートではなく輻輳回避が実行されます。これは高速リカバリー・アルゴリズムです。緩やかな輻輳のもとでの高スループットを可能にする、特に大きなウィンドウのための改良です。

この場合にスロー・スタートを実行しない理由は、重複 ACK の受信が、単なるパケット以上のものが逸失していることを TCP に指し示しているためです。受信側は他のセグメントが受信されるときにだけ重複 ACK を生成できるので、そのセグメントはネットワークからなくなって受信側のバッファに入っています。つまり、まだ 2 つの終端間でフローしているデータが存在するので、TCP はスロー・スタートを開始することによって突然にこのフローを削減しようとはしません。高速再送と高速リカバリーのアルゴリズムは通常以下のように一緒にインプリメントされます。

1. 連続内の 3 番目の重複 ACK を受信すると、`ssthresh` を現行の輻輳ウィンドウである `cwnd` の半分に、ただし少なくとも 2 セグメントに設定します。逸失セグメントを再送します。`cwnd` を、`ssthresh` にセグメント・サイズの 3 倍を加算したものに設定します。こうすると、輻輳ウィンドウ

は、ネットワークを出てもう一方の終端にキャッシュされたセグメントの数だけ増大します (3)。

2. 他の重複 ACK が到着するたびに、そのセグメント・サイズ分だけ cwnd を増分します。そうすると、ネットワークからなくなった追加セグメントのために輻輳ウィンドウが増大します。cwnd の新しい値で許可されているなら、パケットを伝送します
3. 新しいデータの受信を通知する次の ACK が到着したら、cwnd を ssthresh (ステップ 1 で設定された値) に設定します。この ACK は、ステップ 1 で再送の受信確認通知であるはずですが、さらに、この ACK は逸失パケットと最初の重複 ACK の受信の間に送信されたすべての中間セグメントの受信を通知するものであるはずですが、TCP は、パケットが逸失したときのその速度の 1/2 に低下しているので、このステップは輻輳回避です。

第 6 章 IP マルチキャスト

初期の IP ネットワークでは、パケットの送信先は単一の装置（ユニキャスト）かまたはすべての装置（ブロードキャスト）でした。装置のグループを送信先にした単一の送信は不可能でした。しかし、ここ何年かの間に新しい一連のアプリケーションが現れてきています。これらのアプリケーションでは、マルチキャスト送信を使って、装置のグループ間の通信を実際に可能にしています。データは単一のマルチキャスト IP アドレスに送信され、その送信を受け取る必要がある装置がデータを受信します。

この章では、IP マルチキャスト、Internet Group Management Protocol (IGMP)、およびマルチキャスト・ルーティング・プロトコルの間の相互協調処理について解説します。

6.1 マルチキャスト・アドレッシング

マルチキャスト装置は、クラス D の IP アドレスを使用して通信を行います。この場合のアドレスは、224.0.0.0 ~ 239.255.255.255 の範囲に含まれます。マルチキャスト・アドレスごとに 0 個以上のホストの集合があり、そのホストはそのアドレスに送信されるパケットを受信します。装置のこの集合をホスト・グループと呼びます。特定のグループにパケットを送信するホストは、グループのメンバーである必要はありません。そのホストはグループの現行メンバーを知る必要もありません。ホスト・グループには、以下の 2 つのタイプがあります。

- ・ 永続：このタイプのグループに含まれる複数のアプリケーションは、IANA によって永続的に割り当てられた 1 つの IP アドレスを持っています。このタイプのホスト・グループ内のメンバーシップは永続的ではありません。つまりホストは必要に応じてグループに参加したり離脱することができます。永続グループはメンバーがいなくても継続して存在します。永続ホスト・グループに割り当てられた IP アドレスのリストが RFC 1700 に記載されています。そこでの予約済みアドレスは次のとおりです。

- 224.0.0.0: 予約済み基底アドレス
- 224.0.0.1: サブネット上のすべてのシステム
- 224.0.0.2: サブネット上のすべてのルーター
- 224.0.0.9: すべての RIP2 ルーター

その他に OSPF (165 ページの 4.6 『Open Shortest Path First (OSPF)』を参照) のために予約済みの次のアドレスが示されています。

- 224.0.0.5: すべての OSPF ルーター
- 224.0.0.6: OSPF が指定されているルーター

アプリケーションは、永続ホスト・グループに割り当てられている IP アドレスを、DNS でドメイン mcast.net を使って取得できます (291 ページの 8.1 『Domain Name System (DNS)』を参照)。アプリケーションは、ドメイン 224.in-addr.arpa でポインター照会を使って、アドレスから永続グループを判別することができます (295 ページの 8.1.6 『IP アドレスからドメイン・ネームへのマッピング - ポインター照会』を照会)。

- ・一時：永続ではないグループが一時グループになります。このグループは必要に応じた動的な割り当てに使用することができます。一時グループはメンバーの数がゼロになるとなくなります。

6.1.1 単一の物理ネットワーク上のマルチキャスト

このプロセスは簡単です。送信プロセスで宛先の IP マルチキャスト・アドレスが指定されます。デバイス・ドライバーがこの IP アドレスを対応するイーサネット・アドレスに変換し、パケットを宛先に送信します。宛先のプロセスはそのネットワーク・デバイス・ドライバーに、指定したマルチキャスト・アドレスが宛先になっているデータグラムを受信したいということを通知します。デバイス・ドライバーはそのアドレスあてのパケットを受信できるようにします。

標準の IP ユニキャスト・トラフィック転送の場合とは対照的に、IP マルチキャスト宛先アドレスとデータ・リンク・アドレスの間のマッピングは、ARP によって行われるわけではありません。代わりに静的マッピングが定義されています。イーサネット・ネットワークでは、データ・リンク・アドレスの高位のオクテットが 0x'01' の場合に、マルチキャストがサポートされます。IANA は、0x'01005E000000' ~ 0x'01005E7FFFFFFF' の範囲をマルチキャスト用に予約済みです。この範囲では 23 ビットが使用可能です。32 ビットのマルチキャスト IP アドレスからイーサネット・アドレスへのマッピングは、クラス D のアドレスの下位 23 ビットを IANA 予約済みアドレス・ブロックの下位 23 ビットにすることによって行われます。図 112 は、マルチキャスト IP アドレスから対応するイーサネット・アドレスへのマッピングを示したものです。

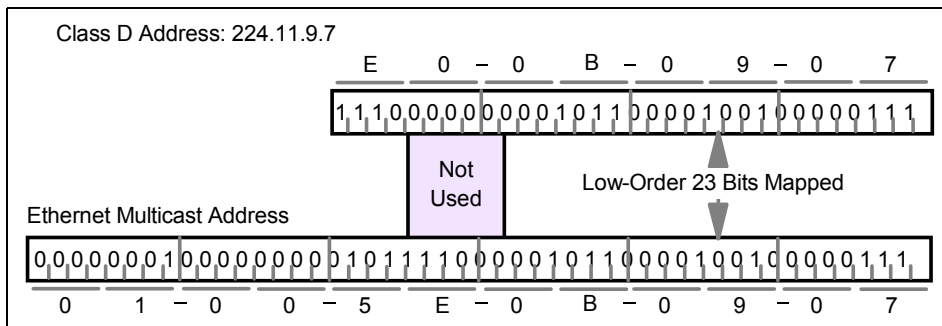


図 112 クラス D の IP アドレスからイーサネット・アドレスへのマッピング

IP マルチキャスト・グループの高位 5 ビットは無視されるため、同一イーサネット・アドレスに 32 の異なるマルチキャスト・グループがマップされます。このようにマッピングが非固有であるため、デバイス・ドライバによるフィルタ操作が必要です。これは、パケットを IP 層に渡す前に IP ヘッダーの中の宛先アドレスを検査することによって行われます。これによって、受信プロセスが自分宛てではないデータグラムを受信することがなくなります。フィルタ操作が必要になる場合の理由としてさらに次の 2 つがあります。

- LAN アダプターの中には、並行して処理できるマルチキャスト・アドレスが、ある有限数に限定されているものがあります。それらは、この限界を超えるとすべてのマルチキャスト・パケットを受信します。
- LAN アダプターによっては、フィルターがマルチキャスト・アドレス全体ではなくハッシュ・テーブル値を使用するものがあります。同じハッシュ値を持つ 2 つのアドレスが同時に使用されると、フィルターは余分なパケットを渡す場合があります。

ソフトウェアのこのようなフィルター操作が必要ですが、マルチキャスト送信では特定のセッションに無関係のホストのオーバーヘッドは少なくなります。具体的には、ホスト・グループに参加していないホストはマルチキャスト・アドレスを受信しません。その状態では、マルチキャスト・パケットは下位層のネットワーク・インターフェース・ハードウェアによってフィルターに掛けられます。

6.1.2 ネットワーク・セグメント間のマルチキャスト

マルチキャスト・トラフィックは単一の物理ネットワークに限定されるものではありません。ただし、ネットワーク間でマルチキャストを行うときには危険が付き物です。環境に複数のルーターが含まれている場合は、マルチキャスト・パケットがネットワーク上をループし続けなくにする特定の予防策が必要です。簡単にマルチキャスト・ルーティング・ループになって

します。これに対処できるように、ルーティング・ループと余分な送信を同時に回避しながらパケットを送達するためのマルチキャスト・ルーティング・プロトコルが開発されました。

複数のネットワーク間でデータをマルチキャストする場合の要件として、次の 2 つがあります。

- マルチキャストに参加しているものの判別：マルチキャスト・データグラムを特定のネットワークに転送する必要があるかどうかを判断するメカニズム。このメカニズムは RFC 2236 Internet Group Management Protocol (IGMP) で定義されています。
- マルチキャストの有効範囲の判別：送信の有効範囲を判断するメカニズム。マルチキャスト・アドレスは、ユニキャスト・アドレスと違ってインターネット全体にまで広がっている場合があります。

マルチキャスト・データグラム内の TTL フィールドを使って送信の有効範囲を判別することができます。マルチキャスト・データグラムには、他の場合のデータグラムのように、それぞれに存続時間 (TTL) フィールドがあります。このフィールドに含まれている値は各ホップで 1 つずつ減分されます。ホストやマルチキャスト・ルーターがデータグラムを受信したときのパケット処理は、TTL 値と宛先 IP アドレスの両方によって次のように異なります。

- TTL = 0: TTL 値がゼロの状態を受信されたマルチキャスト・データグラムの宛先は、送信元ホストに制限されます。
- TTL = 1: TTL 値が 1 のマルチキャスト・データグラムは、サブネット上の、グループのメンバーのすべてのホストに到達します。マルチキャスト・ルーターはその値を減分してゼロにします。ただしユニキャスト・データグラムと違って、ICMP Time Exceeded エラー・メッセージは送信元ホストに戻されません。データグラムの期限切れはマルチキャスト環境では普通に起こることだからです。
- TTL = 2 (以上): この TTL 値のマルチキャスト・データグラムは、サブネット上の、グループのメンバーのすべてのホストに到達します。マルチキャスト・ルーターによって行われるアクションは、グループの特定のアドレスによって次のように異なります。
 - 224.0.0.0 - 224.0.0.255: この範囲のアドレスは単一ホップのマルチキャスト・アプリケーションのためのものです。マルチキャスト・ルーターは、宛先アドレスがこの範囲に入っているデータグラムは転送しません。

マルチキャスト・ルーターはこのアドレス範囲内にはデータグラムを転送しませんが、ホストはグループのメンバーシップをこの範囲内にレポートしなければなりません。このレポートによって、レ

ポートしているホストがそのグループのメンバーであることがサブネット上の他のホストに通知されます。

- その他：クラス D の他の有効な宛先アドレスを持つデータグラムは、マルチキャスト・ルーターによって正常なものとして転送されます。TTL 値は各ホップで 1 つずつ減分されます。

このためホストに**拡張リング検索**をインプリメントすると、特定のマルチキャスト・アドレスを受信しようとしている最近隣サーバーを見つけることができます。ホストは TTL 値が 1 (同一サブネット) のデータグラムを送信して応答を待ちます。応答を受信しなかったときは、ホストは TTL 値が 2 のデータグラムを再送信します。ホストは応答を受信しなかった場合は、最近隣サーバーが見つかるまで自動的に TTL 値を 1 つずつ増分し続けます。

6.2 Internet Group Management Protocol (IGMP)

Internet Group Management Protocol は、ホストがマルチキャスト・ホスト・グループに参加するときや離脱するときにホストによって使用されます。グループ・メンバーシップ情報は、特定のホストと最近隣マルチキャスト・ルーターとの間で交換されます。

IGMP は ICMP を拡張したものと見なすことができます。これはプロトコル・スタックでは同じ位置です。IGMP バージョン 2 が RFC 2236 に記載されています。

IGMP の機能は IPv6 に直接組み込まれています (599 ページの 17.3 『Internet Control Message Protocol Version 6 (ICMPv6)』を参照)。すべての IPv6 ホストでマルチキャストをサポートする必要があります。IPv4 ではマルチキャストと IGMP のサポートはオプションです。

6.2.1 IGMP メッセージ

IGMP メッセージは IP データグラム内にカプセル化されます。IGMP パケットを示すために IP ヘッダーにはプロトコル番号として 2 が含まれています。IP データ・フィールドには図 113 に示すように 8 オクテットの IGMP メッセージが含まれています。

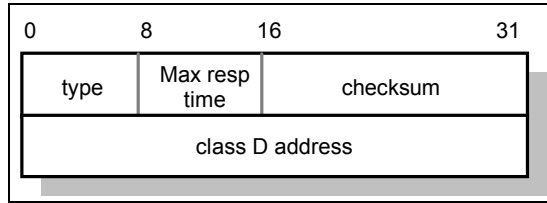


図 113 IGMP メッセージ・フォーマット

IGMP メッセージ内の各フィールドには次の情報が含まれています。

- タイプ：このフィールドで IGMP パケットのタイプを次のように指定します。
 - 0x'11': メンバーシップ照会パケットを指定します。これはマルチキャスト・ルーターによって送信されます。メンバーシップ照会メッセージには次の 2 つのサブタイプがあります。
 - 一般照会：接続されたネットワーク上のメンバーが参加しているグループを知るために使用されます。
 - グループ固有照会：接続されたネットワーク上のメンバーが特定のグループに参加しているかどうかを知るために使用されます。
 - 0x'16': メンバーシップ・レポート・パケットを指定します。これはマルチキャスト・ホストによって送信され、特定のマルチキャスト・ホスト・グループへの参加をシグナル通知します。
 - 0x'17': リーブ・グループ・パケットを指定します。これはマルチキャスト・ホストによって送信されます。
- 最大応答時間：このフィールドはメンバーシップ照会メッセージで使用されます。これはホストが対応レポートを送信するまで待つことができる最大許容時間を指定します。この設定値を変えてルーターのリーブ待ち時間を調整することができます。これは最後のホストがグループを離脱したときの時刻から、メンバーが存在しなくなったことがルーティング・プロトコルに通知される時刻までの時間になります。
- チェックサム：このフィールドには 16 ビットのチェックサムが含まれます。
- クラス D アドレス：このフィールドには有効なマルチキャスト・グループ・アドレスが含まれます。これはレポート・パケットで使用されます。

IGMP バージョン 3 の開発が始まりました。このバージョンでは、受信側がマルチキャスト・グループ内の特定の送信元の集合にサブスクライブできるようになります。

6.2.2 IGMP での操作

ホストとマルチキャスト・ルーターの両方が IGMP の機能に関与します。

6.2.2.1 ホストでの操作

ホストは、マルチキャスト・データグラムを受信するためには、ホスト・グループに参加していなければなりません。マルチホームのホストは、それが接続されている 1 つ以上のインターフェースでグループに参加することができます。ホストが複数のインターフェースで同じグループに参加すると、ホストが受信するマルチキャスト・メッセージが異なる場合があります。たとえば 244.0.0.1 は、サブネット上のすべてのホストのグループです。あるサブネットを介してこのグループで受信されるメッセージは、別のサブネット上で受信されるメッセージとは必ず異なります。

単一ホスト上の複数のプロセスが同一グループのメッセージを受信する場合があります。この状況になると、ホストは一度グループに参加します。ホストはそのグループに関係がある各プロセスを内部で追跡します。

ホストがグループに参加するときは、ホストは接続されているインターフェースを介して IGMP メンバーシップ・レポート・パケットを送信します。このレポートの宛先は、参加するマルチキャスト・グループです。ホストは全ホストのグループ (244.0.0.1) に参加する必要はありません。このグループのメンバーシップは自動的なものです。

6.2.2.2 マルチキャスト・ルーターでの操作

ホストがグループに参加しようとする、サブネット上のマルチキャスト・ルーターがメンバーシップ・レポート・パケットを受信し、ルーターのローカル・グループ・データベース内に項目を作成します。そのルーターが直接接続されているネットワークのグループ・メンバーシップは、このデータベースで追跡されます。このデータベース内の各項目のフォーマットは [グループ、接続ネットワーク] です。これは、接続ネットワークにそのグループに属している IP ホストが少なくとも 1 つあることを表しています。マルチキャスト・ルーターはマルチキャスト・アドレスをすべて受信してこれらのレポートを検出します。

ローカル・グループ・データベース内の情報を使ってマルチキャスト・データグラムの転送が行われます。ルーターがデータグラムを受信すると、そのデータグラムはグループに属しているホストが含まれる各インターフェースに対して転送されます。

マルチキャスト・ルーターはグループ・メンバーシップを検証するために、定期的に IGMP 照会メッセージを全ホストのマルチキャスト・アドレスに送信します。まだグループのメンバーである必要があるホストはそれぞれ応答を送信します。RFC 2236 では、この検証は 125 秒ごとに行われなければな

らないと指定されています。照会メッセージへの応答は、サブネット上のトラフィックのバーストを回避するために、ランダム遅延を使って送信されます。ルーターは各グループ内のホストの数を追跡しないため、別の装置がメンバーシップを請求していること検知したホストは、保留中のメンバーシップ応答をいずれも取り消します。指定された時間間隔内にどのホストもメンバーシップを請求しなかった場合、マルチキャスト・ルーターはそのネットワーク上のいずれのホストもそのグループのメンバーではないと見なします。

6.3 マルチキャスト送達ツリー

IGMP では受信側のホストとそれらのローカル・マルチキャスト・ルーターの間で起こる通信だけを指定しています。マルチキャスト・ルーター間のパケットのルーティングは、別のルーティング・プロトコルによって管理されます。それらのプロトコルについては 246 ページの 6.4 『マルチキャスト転送アルゴリズム』で説明しています。図 114 は、マルチキャスト・ルーティング・プロトコルと IGMP がマルチキャスト送達ツリーの別々の部分で作用することを示したものです。

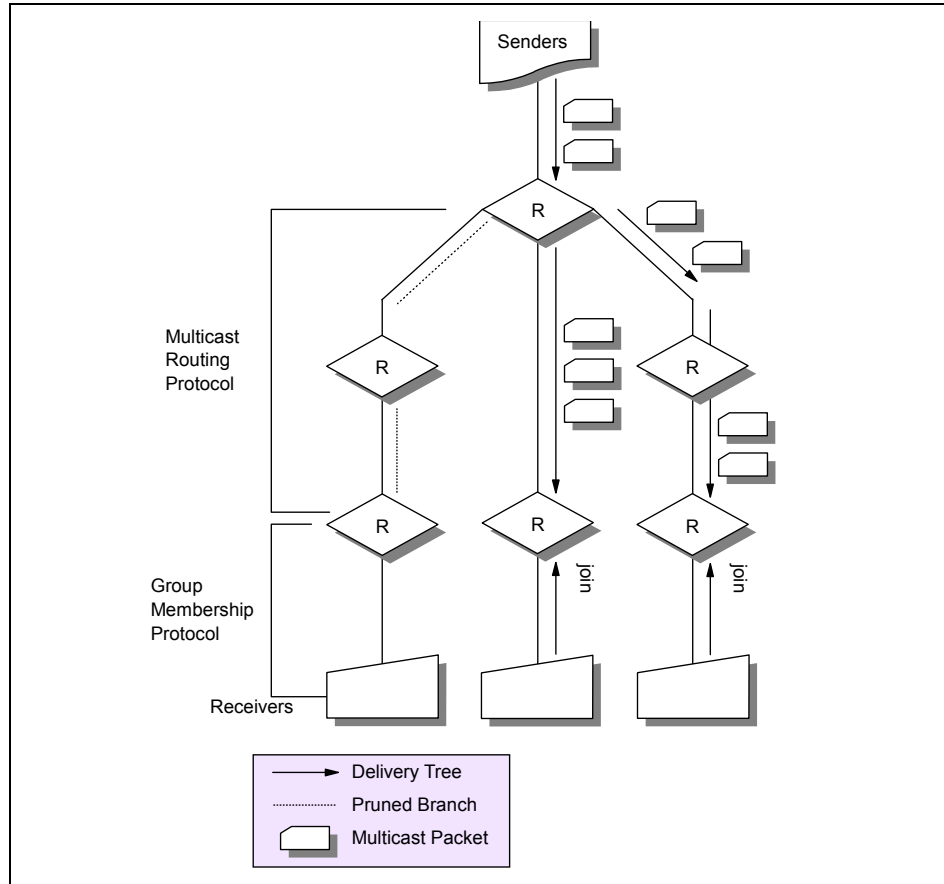


図 114 マルチキャスト送達ツリー

この図はマルチキャスト送信側と受信側群との間で形成されるツリーを表しています。マルチキャスト・ルーターに接続されたホストがこの特定のグループに参加していない場合、それらのホストがつながっている分岐にはこのグループ向けのマルチキャスト・パケットは送達されません。その分岐は送達ツリーから枝取りされます。この処置によってツリーのサイズが、すべてのグループ・メンバーに到達するために必要な最小数の分岐まで縮小します。グループに新しいメンバーが参加すると、ツリーに新しい部分が動的に追加される場合があります。これによって送達ツリーに新しい部分が枝付けされます。

6.4 マルチキャスト転送アルゴリズム

ネットワークでのパスを設定するときにマルチキャスト・アルゴリズムが使用されます。マルチキャスト・トラフィックはそのパスによってすべてのグループ・メンバーに実際に到達できるようになります。アルゴリズムはそれぞれ次の一連の要件を満たさなければなりません。

- ・ アルゴリズムでグループ・メンバーだけへのデータの経路を定める。
- ・ アルゴリズムで送信元から宛先までのパスを最適化する。
- ・ アルゴリズムでループがない経路を維持する。
- ・ アルゴリズムでグループ・メンバーシップの作成と維持に使用される拡張が容易なシグナル機能を提供する。
- ・ アルゴリズムでリンクのサブネット上にトラフィックを集中させてはならない。

マルチキャスト・ルーティング・プロトコルで使用するためのアルゴリズムがいくつか開発されました。それらのアルゴリズムでは、ここで述べている設計要件を満たす成功レベルはさまざまです。続くいくつかのセクションでは、2つのアルゴリズムを検討します。

6.4.1 逆パス転送アルゴリズム

逆パス転送 (RPF) アルゴリズムでは、マルチキャスト送達ツリーを使って送信元のデータグラムをマルチキャスト・グループ内の各メンバーに転送します。図 115 に示すように、パケットは送達ツリー内の必要な分岐でのみ複製されます。

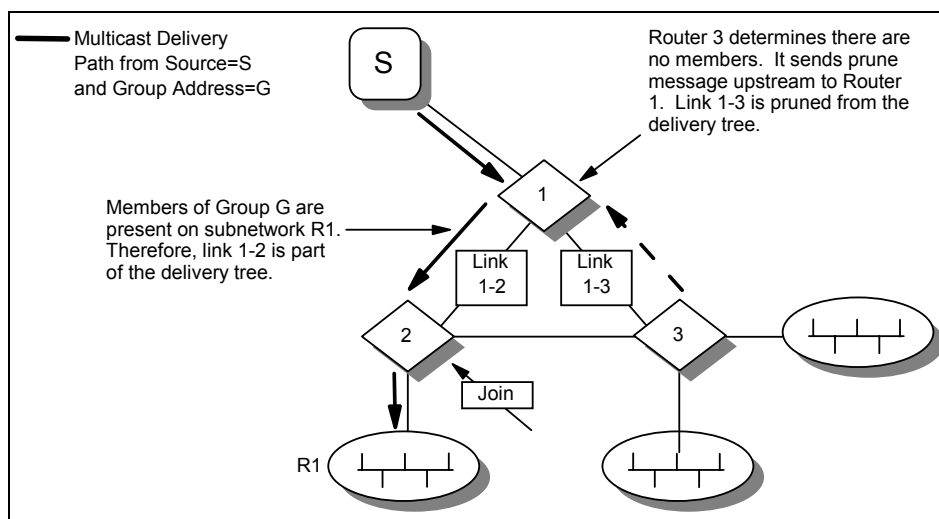


図 115 逆パス転送 (RPF)

個々のグループのメンバーシップを追跡するために、ツリーは動的に計算され更新されます。

アルゴリズムは各送信元に到達するために使用される逆パス・テーブルを維持しています。このテーブルでは、すでに判明している送信元ネットワークのそれぞれが、送信元に到達するときに使用される優先インターフェースにマップされています。データを転送するとき、データグラムがデータグラムを送信元へ送り返すときに使用されるインターフェースを介して到着した場合は、そのデータグラムはそれぞれの適切なダウンストリーム・インターフェースを介して転送されます。そうでない場合の2次最適パスを介して到着したデータグラムは廃棄されます。このプロセスを使って、ネットワーク・ループが原因で起こる重複パケットがフィルターに掛けられます。

RPF を使用すると次の利点があります。

- RPF ではマルチキャスト・データの最も高速な送達が保証されます。この構成では、トラフィックは送信元から各宛先まで最短のパスを通ります。
- 送信元ノードごとに別々のツリーが計算されます。パケット送達は複数のネットワーク・リンク上に分散されます。これはネットワーク・リソースをより有効に使用することになります。

6.4.2 中心基点ツリー・アルゴリズム

中心基点ツリー (CBT) アルゴリズムでは、別のメソッドでマルチキャスト・グループのメンバー間の最適パスを決定します。このアルゴリズムには次のステップがあります。

- ネットワーク内の中心点が選ばれます。この固定点は、マルチキャスト・グループの中心を表します。
- 受信側は、それぞれ参加要求を中心点に向けて送信します。これはそのグループの IGMP メンバーシップ・レポートを使って行われます。
- この要求は、マルチキャストの受信側と中心点の中間にあるすべての装置によって処理されます。要求を受信したルーターがすでにそのツリーのメンバーである場合は、そのルーターはそのグループに属するものとしてさらにもう1つのインターフェースを区分します。要求が最初の参加要求だった場合、ルーターはその要求を送信元に向けてさらに1ステップ転送します。

この手順によってマルチキャスト・グループごとに送達ツリーが構築されます。すべての送信元について同一のツリーになります。各ルーターがグループ全体の単一のツリーを維持します。これは RPF アルゴリズムで使用されるプロセスと大きく異なる点です。RPF アルゴリズムではマルチキャスト・グループ内で送信元ごとにツリーを構築します。

送信元がグループのメンバーである必要性はないため、送信元のマルチキャスト・パケットはツリーに属しているルーターに到達するまで中心点に向かって転送されます。この段階はパケットは中心基点ツリーのマルチキャスト処理を使って転送されます。

中心基点ツリー・アルゴリズムの欠点として、送信元や受信側によっては 2 次最適なパスが構築される場合があるということが挙げられます。

6.4.3 マルチキャスト・ルーティング・プロトコル

マルチキャスト・ルーティング・プロトコルが多数開発されましたが、それらは次のアルゴリズムを使用しています。

- Distance Vector Multicast Routing Protocol (DVMRP)
- Multicast OSPF (MOSPF)
- Protocol Independent Multicast (PIM)

この章の残りでは、これらのプロトコルの詳細について説明します。

6.5 Distance Vector Multicast Routing Protocol (DVMRP)

DVMRP はマルチキャスト・ルーティング・プロトコルとして確立されています。これは元は RFC 1075 で定義されたものです。この標準が最初にインプリメントされたのは、多くの UNIX システム上で使用できる *mrouted* プロセスとしてでした。その後 RPF をサポートするように拡張されました。DVMRP は内部ゲートウェイ・プロトコルです。これは、送信元ごととグループごとのマルチキャスト送達ツリーを AS 内に構築するときに使用されます。

DVMRP ではユニキャスト・データグラムの経路を定めません。マルチキャストおよびユニキャストの両方のデータグラムを処理するルーターは、2 つのルーティング・プロセスを別々に持つように構成されなければなりません。別々のプロセスが使用されるため、マルチキャストおよびユニキャストのトラフィックはネットワークの同じパスを通らなくなります。

6.5.1 プロトコルの概要

DVMRP は次のようにブロードキャストと枝取りのマルチキャスト・ルーティング・プロトコルといえます。

- DVMRP ではルーティングの交換を基に送信元ごとのブロードキャスト・ツリーを構築します。
- DVMRP では送信元ごとのブロードキャスト・ツリーを動的に枝取りしてマルチキャスト送達ツリーを作成します。DVMRP ではマルチキャスト・トラフィックに使用される一連のダウンストリーム・インターフェースを決定するために RPF アルゴリズムを使用します。

6.5.1.1 近隣ディスカバリー

DVMRP の各ルーターは、ローカル・インターフェースごとに近隣プローブ・メッセージを定期的に送信することによって、それぞれの近隣を動的に検出します。これらのメッセージは全 DVMRP ルーター・マルチキャスト・アドレス (224.0.0.4) に対して送信されます。各メッセージには受信済みの近隣プローブ・メッセージの送信元の近隣 DVMRP ルーターのリストが含まれています。これにより DVMRP ルーターはメッセージがそれぞれの近隣によって受信されていたことを確認することができます。

ルーターが近隣リストの中に自分のアドレスが含まれているプローブ・メッセージを受信すると、そのペアのルーターが両方向の近隣隣接を確立します。

6.5.1.2 ルーティング・テーブルの作成

DVMRP では RPF アルゴリズムで使用される一連の逆パスを計算します。すべての DVMRP ルーターで送信元に接続するためのパスを整合したものにするために、各近隣ルーター間でルーティング・テーブルが交換されます。DVMRP では独自のユニキャスト・ルーティング・プロトコルをインプリメントしています。このルーティング・プロトコルは RIP に似ています。

そのアルゴリズムはホップ・カウントに基づいています。DVMRP ではすべてのインターフェースについてメトリックを構成する必要があります。各ルーターは各インターフェースのネットワーク番号、マスク、およびメトリックを通知します。またルーターは近隣ルーターから受信した経路も通知します。経路が受信されると、他の距離ベクトル・プロトコルの場合のように、通知されたメトリックにそのインターフェースのメトリックが追加されます。メトリックがこのように調整され、それをもとに送信元への最適なアップストリーム・パスが決定されます。

DVMRP には RIP と大きく異なる点が 1 つあります。RIP では特定のユニキャスト宛先へ転送するルーティングとデータグラムを管理します。DVMRP では特定のマルチキャスト・データグラムの送信元へのリターン・パスを管理します。

6.5.1.3 従属ダウンストリーム・ルーター

ルーティング情報を交換すると、送信元ネットワークへのパスを整合したものにする他に、付加的な利点もあります。DVMRP ではこのメカニズムを使って、アップストリーム・ルーターに特定のダウンストリーム・ルーターがマルチキャスト・トラフィックの転送を要求していることを通知しています。

DVMRP では poison reverse 手法 (158 ページの 4.3.5.2 『split horizon with poison reverse』を参照) を使ってこれを行います。ダウンストリーム・ルーターが特定の送信元へのネクスト・ホップとしてアップストリーム・ルーターを選択すると、ダウンストリーム・ルーターのルーティングが更新され

て、送信元ネットワークに対するメトリックとして無限大が指定されます。アップストリーム・ルーターはこの通知を受信すると、そのダウストリーム・ルーターをそのときの送信元用の**従属ダウストリーム・ルーター**のリストに追加します。この手法によってマルチキャスト送達ツリーの枝取りに必要な情報を得ることができます。

6.5.1.4 指定転送装置

マルチアクセス・ネットワークに 2 つ以上のマルチキャスト・ルーターが接続されていると、そのネットワークに対してパケットが重複して転送される場合があります。DVMRP では送信元ごとに指定転送装置を選ぶことによってこれが起こらないようにしています。

各ルーターは、それぞれのルーティング・テーブルを交換したときに、同位のルーターでの送信元ネットワークに到達するために必要なメトリックが分かれます。最小のメトリックを持つルーターが共用ネットワークへのデータの転送を行います。複数のルーターが同じメトリックを持っている場合は、IP アドレスが最小のルーターがネットワークへの指定転送装置になります。

6.5.2 マルチキャスト送達ツリーの構築と保守

前述のように RPF アルゴリズムはマルチキャスト・データグラムを転送するときに使用されます。データグラムが送信元への最適パスになっているインターフェースで受信されると、ルーターはそのデータグラムを一連のダウストリーム・インターフェースへ転送します。この一連のダウストリーム・インターフェースには、マルチキャスト送達ツリーに組み込まれている各ダウストリーム・インターフェースが含まれています。

6.5.2.1 マルチキャスト送達ツリーの構築

マルチキャスト・ルーターは、ダウストリーム従属ルーターと、特定のマルチキャスト・グループのメンバーになっているホストとの 2 タイプの装置にデータグラムを転送します。マルチキャスト・ルーターに特定のインターフェースを介して従属するダウストリーム近隣がない場合、そのネットワークはリーフ・ネットワークです。送達ツリーはこのようなタイプが異なる宛先の詳細を含むルーティング情報を使って構築されます。

リーフ・ネットワークの追加

ダウストリーム・インターフェースがリーフ・ネットワークに接続されている場合、パケットは特定のマルチキャスト・グループのメンバーになっているホストがあるときだけ転送されます。ルーターはこの情報を IGMP ローカル・グループ・データベースから取得します。そのデータベースにグループ・アドレスがリストされていてそのルーターが送信元の指定転送装置である場合、そのインターフェースがマルチキャスト送達ツリーに組み込まれます。グループ・メンバーが存在しない場合はそのインターフェースは除外されます。

非リーフ・ネットワークの追加

最初はすべての非リーフ・ネットワークがマルチキャスト送達ツリーに組み込まれます。これによって各ダウンストリーム・ルーターは各グループのトラフィック転送に関与できます。

6.5.2.2 マルチキャスト送達ツリーの枝取り

リーフ・ネットワークに接続されているルーターは、特定のマルチキャスト・グループに参加しているアクティブなメンバーがなくなるとインターフェースを除去します。これが行われると、マルチキャスト・パケットはそのインターフェースを介して転送されなくなります。

ルーターが特定のグループのための自分のダウンストリーム・インターフェースをすべて除去できる場合、そのルーターはその特定の送信元とグループのペアでのトラフィックがなくなっただけを自分のアップストリーム近隣へ通知します。この通知は、枝取りメッセージをアップストリーム近隣へ送信することによって行われます。アップストリーム近隣が、あるインターフェースの従属ダウンストリーム・ルーターのそれぞれから枝取りメッセージを受信すると、アップストリーム・ルーターはそのインターフェースをマルチキャスト送達ツリーから除去することができます。

アップストリーム・ルーターは、自分のすべてのインターフェースをツリーから枝取りできる場合は、自分のアップストリーム・ルーターに対して枝取りメッセージを送信します。不要な分岐がすべて送達ツリーから除去されるまでこれが続けて行われます。

枝取り情報の保守

古くなった枝取り情報を除去するために、それぞれの枝取りメッセージには枝取り存続時間タイマーが含まれています。これは枝取りの状態が実際に継続できる時間の長さを示しています。タイマーの期限が切れたときにそのインターフェースが枝取りされているままだった場合、そのインターフェースはマルチキャスト送達ツリーに再接続されます。このために不要なマルチキャスト・データグラムがダウンストリーム装置へ送達されるようになると、枝取りが再度開始されます。

6.5.2.3 枝取りされたネットワークの枝付け

IP マルチキャストでは動的グループ・メンバーシップをサポートしているため、ホストはいつでもマルチキャスト・グループに参加できます。これが行われると、DVMRP ルーターは枝付けメッセージを使って、そのネットワークをマルチキャスト送達ツリーに再接続します。枝付けメッセージは、その前に枝取りされているグループの IGMP メンバーシップ・レポートの受信の結果として送信されます。枝取りされていた送信元ネットワークごとに該当するアップストリーム近隣に対して、別々の枝付けメッセージが送信されます。

枝付けメッセージの受信の応答は、枝付け ACK メッセージで行われます。送信側はこれによって、枝付けパケットの紛失か、装置が非アクティブだったのかを区別することができます。枝付けタイムアウト時間内に応答を受信できなかった場合は、要求が再送信されます。枝付け ACK メッセージの目的は枝付けメッセージの受信を確認することです。これは要求の結果として何かのアクションが行われたことを意味するものではありません。したがって受信ルーターがいずれの要求メッセージに対しても何のアクションも起こさなかった場合でも、それらすべての枝付け要求メッセージに対して応答が行われます。

6.5.3 DVMRP でのトンネル

IP ルーターの中には固有マルチキャスト・ルーティングをサポートするように構成できないものがあります。DVMRP には非マルチキャスト・ルーターを含むネットワークでの IP マルチキャスト・データグラムのトンネリングを行える機能があります。データグラムはユニキャスト IP パケットにカプセル化されてネットワークを転送されます。この動作を図 116 に示します。

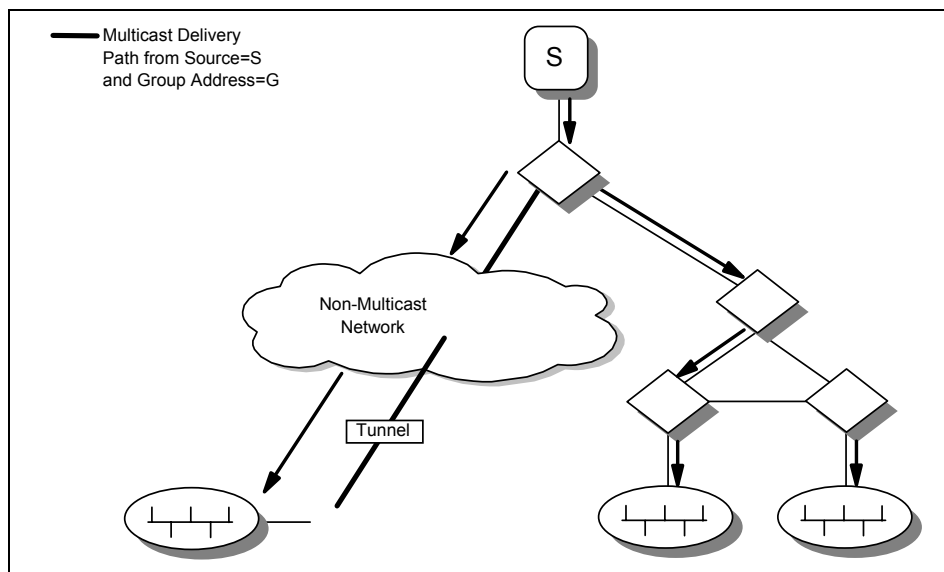


図 116 DVMRP でのトンネル

パケットがトンネルのリモート・エンドで受信されると、そのパケットはカプセル化解除され、標準 DVMRP マルチキャスト操作でサブネットワークを介して転送されます。

6.6 Multicast OSPF (MOSPF)

MOSPF は OSPF バージョン 2 (165 ページの 4.6 『Open Shortest Path First (OSPF)』を参照) のマルチキャスト拡張です。これは RFC 1584 で定義されています。MOSPF は DVMRP と違って、単独のマルチキャスト・ルーティング・プロトコルではありません。これはユニキャスト IP ルーティングにすでに OSPF を使用しているネットワークで使用されます。マルチキャスト拡張では既存の OSPF トポロジー・データベースを利用して、送信元がルートの最短パス送達ツリーを作成します。

MOSPF では送信元と宛先の両方のアドレスを使ってマルチキャスト・データグラムを転送します。これは宛先アドレスだけを使用する標準 OSPF アルゴリズムと大きく異なる点です。

6.6.1 プロトコルの概要

MOSPF の概要については、続くいくつかのセクションで説明します。

6.6.1.1 グループ・メンバーシップ LSA

すべてのグループ・メンバーのロケーションがその環境の他のメンバーにも伝えられなければなりません。これによって各メンバーにマルチキャスト・データグラムが転送されるようになります。OSPF では新規タイプのリンク状態通知 (グループ・メンバーシップ LSA) を追加して各グループ・メンバーのロケーションを追跡します。この LSA は OSPF リンク状態データベース内に保管されます。このデータベースには AS のトポロジーが保管されます。

6.6.1.2 指定ルーター

ネットワーク・セグメントごとに 1 つの MOSPF ルーターが指定ルーター (DR) として選択されます。このルーターは IGMP ホスト・メンバーシップ照会の生成を定期的に行うものです。このルーターは IGMP メンバーシップ・レポートを受信することもあります。ルーターは自分が DR ではないネットワークで受信したレポートは無視します。これによって、多くても 1 つのルーターのローカル・グループ・データベース内に各ネットワーク・セグメントが現れるようになります。またこれによって、データグラムがローカル・グループ・メンバーに送達されるときに複製されないようにもしています。

すべてのルーターは、そのルーターのローカル・グループ・データベース内に少なくとも 1 つの項目を持っているマルチキャスト・グループごとに、グループ・メンバーシップ LSA のフラッディングを行います。この LSA のフラッディングは全 OSPF エリアに対して行われます。

6.6.1.3 最短パス送達ツリー

マルチキャスト・データグラムを転送するときに使用されるパスの計算は、そのデータグラムの送信元をルートとする最短パス送達ツリーを構築するこ

とによって行われます (149 ページの 4.2.3.1 『Shortest-Path First (SPF) アルゴリズム』を参照)。このツリーはリンク状態データベース内に含まれている情報を基に構築されます。最短パス送達ツリー内で、対応するグループ・メンバーシップ LSA を持っていない分岐は枝取りされます。これらの分岐には特定のグループのマルチキャスト・メンバーは含まれていません。

最短パス送達ツリーは最初のデータグラムが受信されたときに最初に構築されます。その結果はそれ以後の同じ送信元と宛先を持つデータグラムで利用されるようにキャッシュに入れられます。ツリーはリンク状態に変化が起きたときやキャッシュ情報がタイムアウトになったときに再計算されます。

MOSPF ネットワークでは、特定のマルチキャスト・データグラムについて、すべてのルーターが同一の最短パス送達ツリーを計算します。データグラムの送信元と任意の特定の宛先の間で単一のパスが存在しています。このため、OSPF でのユニキャスト・トラフィックの処理と違って、MOSPF では同等コスト・マルチパスは用意されないことになります。

6.6.2 MOSPF と複数の OSPF エリア

OSPF では AS を複数のエリアに分割することができます。これにはトラフィック管理上の利点がある一方、各ルーター内で維持されるトポロジー情報が制限されます。ルーターではローカル・エリア内のネットワーク・トポロジーが分かっているだけです。それらの環境での最短パスのツリーを構築する場合、リンク状態データベースに含まれている情報だけでは各送信元と宛先の間で完全なパスを作成できません。このため、最適ではないパスが選択される場合があることになります。

OSPF エリア内では、エリア・ボーダー・ルーター (ABR) がルーティング情報とデータ・トラフィックをエリア間で転送します。MOSPF 環境でのこれに相当する機能は、*エリア間マルチキャスト転送装置*によって行われます。この装置は、グループ・メンバーシップ情報とマルチキャスト・データグラムをエリア間で転送します。OSPF ABR は、MOSPF エリア間マルチキャスト転送装置としても機能します。

グループ・メンバーシップ LSA のフラッディングはエリア内だけで行われるため、メンバーシップ情報をエリア間で伝達するプロセスが必要です。これを行うために、エリア間マルチキャスト転送装置のそれぞれが、接続されているエリアのグループ・メンバーシップの要件を要約し、その情報を OSPF バックボーンへ転送します。この通知はグループ・メンバーシップ LSA で形成され、そこには非バックボーン・エリア内のメンバーを含む各グループがリストされています。この通知は標準 OSPF エリアで生成される要約 LSA と同じ機能を果たします。

ただし標準 OSPF ネットワークでの経路の要約化と違って、MOSPF でのマルチキャスト・グループ・メンバーシップの要約化は均整がとれていません。非バックボーン・エリアのメンバーシップ情報がバックボーンに向けて要約されます。しかしこの情報は他の非バックボーン・エリアには再通知されません。

マルチキャスト・データ・トラフィックをエリア間で転送する場合は、ワイルドカード・マルチキャスト・レシーバーが使用されます。これは宛先にかかわらずすべてのマルチキャスト・トラフィックの転送先になるルーターです。非バックボーン・エリアでは、エリア間マルチキャスト転送装置はすべてワイルドカード・マルチキャスト・レシーバーになります。そのため非バックボーン・エリア内から発信されるマルチキャスト・トラフィックはすべてエリア間マルチキャスト転送装置へ転送されるようになります。このルーターはマルチキャスト・データグラムをバックボーン・エリアに送信します。バックボーンではすべてのグループ・メンバーシップ情報の詳細が分かっているため、次にデータグラムは他のエリア内の該当するグループ・メンバーに転送されます。

6.6.3 MOSPF と複数の自律システム

別の AS 内に少なくとも 1 つのマルチキャスト装置が置かれている場合は、エリア間マルチキャスト・ルーティングに類似した状態になります。両方の場合とも、送信元から宛先への完全なパスを示す最短パス・ツリーを構築することはできません。

この環境では MOSPF ドメイン内の ASBR が AS 間マルチキャスト転送装置として構成されます。このルーターは AS 間マルチキャスト・ルーティング・プロトコルを使用するように構成される場合もあります。MOSPF 標準では AS 間プロトコルの操作を規定していませんが、そのプロトコルが RPF の原理を使ってデータグラムを転送することを想定しています。具体的には MOSPF では、送信元がドメイン外のマルチキャスト・データグラムが、送信元への最適経路を (OSPF へ) 通知しているポイントからドメインに入ることを想定しています。MOSPF ではこの情報を使ってそのドメインでのデータグラムのパスを計算します。

MOSPF では AS 間マルチキャスト転送装置をワイルドカード・マルチキャスト・レシーバーとして指定します。これによって、エリア間通信の場合に、枝取りされたすべての最短パス送達ツリー上にレシーバーが残ることになります。それらのレシーバーは宛先にかかわらず、すべてのマルチキャスト・データグラムを受信します。この装置には AS 範囲外のすべてのグループ・メンバーシップの詳細情報があるため、データグラムを他の自律システム内のグループ・メンバーに転送することができます。

6.6.4 MOSPF のインターオペラビリティ

MOSPF ネットワークをサポートするように構成されたルーターは、非マルチキャスト OSPF ルーターと混在することができます。ユニキャスト・データ・トラフィックを転送するときに、両方のタイプのルーターが相互に協調して処理を行います。ただし IP マルチキャスト・トラフィックの転送は MOSPF ドメインに限定されています。DVMRP と違って、MOSPF では非マルチキャスト・ルーターでのマルチキャスト・トラフィックのトンネリングの機能はありません。

6.7 Protocol Independent Multicast (PIM)

MOSPF にまつわる複雑さが PIM の開発と展開につながりました。PIM はもう一つのマルチキャスト・ルーティング・プロトコルです。MOSPF と違って PIM は、基礎的ななどのユニキャスト・ルーティング・プロトコルから独立しています。これは既存のすべてのユニキャスト・ルーティング・プロトコルと相互に協調して動作します。

PIM では次の 2 つのモードまたは操作を定義しています。

- 高密度モード (PIM-DM)
- 希薄モード (PIM-SM) (RFC 2362 で規定)

高密度モードと希薄モードはエリア内のグループ・メンバーの密度を指しています。ランダム・サンプリングで、サンプル内でグループ・メンバーを少なくとも 1 つ検出する確率が高い場合にそのグループは高密度であると見なされます。これはサンプル・サイズが小さい場合でも、それが合理的であれば適用されます。サンプル内でグループ・メンバーを検出する確率が低い場合に、そのグループは希薄であると見なされます。

PIM には希薄モードと高密度モードを切り替える機能があります。同じグループ内で両方のモードを使用することもできるようになっています。

6.7.1 PIM 高密度モード

PIM-DM プロトコルには 246 ページの 6.4.1 『逆パス転送アルゴリズム』で説明した RPF プロセスがインプリメントされています。具体的には PIM-DM 装置は、パケットを受信するときに着信インターフェースを既存のユニキャスト・ルーティング・テーブルを使って妥当性の検査を行います。着信インターフェースが送信元への最適パスになっていたときは、ルーターはマルチキャスト・パケットのフラッドिंगを行います。そのパケットはマルチキャスト送達ツリーから枝取りされていないすべてのインターフェースへ送出されます。

DVMRP とは異なり、PIM-DM ではマルチキャストの特定の経路の計算を試みません。むしろこれは、ユニキャスト・ルーティング・テーブル内の経路が対称性があると想定します。

DVMRP での操作のように、PIM-DM 装置は最初はすべてのダウンストリーム・インターフェースがマルチキャスト・トラフィックを受信する必要があると想定しています。ルーターはネットワークのすべてのエリアに対してデータグラムのフラッディングを行います。特定のマルチキャスト・グループの受信側がないエリアがあった場合は、PIM-DM ではそれに応じて送達ツリーからそれらの分岐を枝取りします。この反応枝取りが行われるのは、PIM-DM ではユニキャスト・ルーティング・テーブルからダウンストリーム受信側情報を取得しないためです。図 117 は PIM-DM での枝取りの例を示したものです。

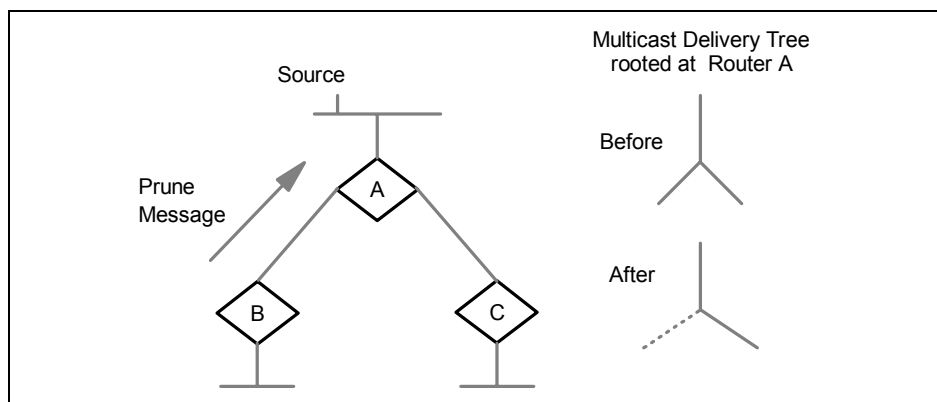


図 117 PIM-DM でのフラッディングと枝取りの操作

PIM-DM のインプリメントは比較的簡単です。唯一の前提事項としてルーターは枝取り要求のリストを保存することが必要です。

6.7.1.1 PIM-DM の利点

PIM-DM ではフラッディングと枝取りの方法が使われているため、このプロトコルはドメイン内の大多数のホストがマルチキャスト・データを受信する必要がある環境で使用すべきです。これらの環境の大多数のネットワークでは、送達ツリーからの枝取りは行われません。フラッディングに関連したオーバーヘッドは最小限度のものになります。この構成は次の場合にも適用できます。

- ・ 送信側と受信側が相互に近接している。
- ・ 送信側が少なく受信側が多い。
- ・ マルチキャスト・トラフィックの量が多い。
- ・ マルチキャスト・トラフィックのストリームが一定である。

DVMRP と違って、PIM-DM では非マルチキャストが可能なネットワークで、マルチキャスト・トラフィックを送信するためのトンネルをサポートしていません。そのためネットワーク管理者は、エンドツーエンドのパスに接続された各装置をマルチキャストが可能になるようにしなければなりません。

6.7.2 PIM 希薄モード

PIM-SM プロトコルでは中心基点ツリー・アルゴリズムの変形を使用します。PIM-SM ネットワークでは集結点 (RP) がそのアルゴリズムで説明した中心点に類似したものになります。具体的には、RP はネットワーク内でマルチキャスト送信側がマルチキャスト受信側に接続されるロケーションです。受信側は RP をルートとするツリーに参加します。送信側は自分が存在していることを RP に登録します。送信側からのトラフィックは最初は RP を経由して流れて各受信側に到達します。

PIM-SM の利点は、DVMRP や PIM-DM のネットワークと違って、ダウンストリーム装置がデータを受信すると特別に要求しない限り、マルチキャスト・データがネットワーク・セグメントでブロックされることです。これによって、ネットワークを横断するトラフィックの総量が大幅に減少する可能性があります。また、受信側がないロケーションでは枝取り情報を維持しないということにもなります。この情報は、マルチキャスト送達ツリーに接続された装置だけで維持されます。このような利点があるため、PIM-SM はインターネットで使用されるマルチキャスト・ルーティング・プロトコルとして現在最も普及しています。

6.7.2.1 PIM-SM でのマルチキャスト送達ツリーの構築

PIM-SM での RP との基本的な相互作用は以下のようになります。

1. マルチキャスト・ルーターが参加メッセージをグループ固有の RP に定期的に送信します。RP へのパス上にある各ルーターが参加要求を作成して RP に送信します。これによって RP をルートとしたグループ固有のマルチキャスト送達ツリーが構築されます。参加要求は受信側から RP に向かう逆パスを通るため、このツリーは、他のマルチキャスト・プロトコルの場合と同様に、実際は逆パス・ツリーです。この作用を図 118 に示します。

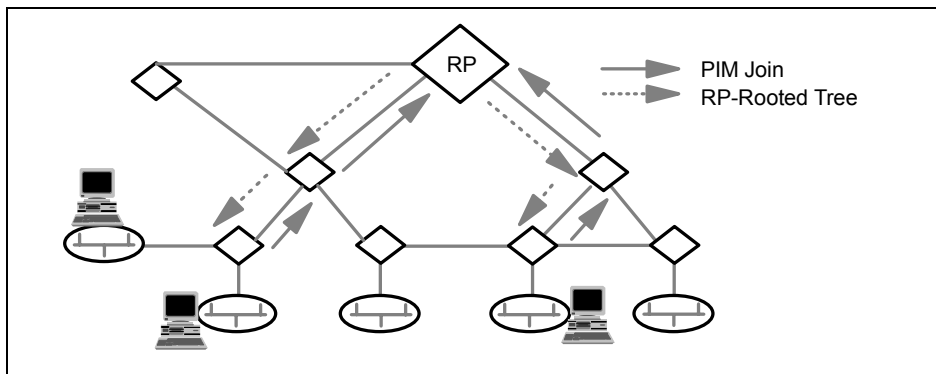


図 118 RP をルートとした送達ツリーの作成

2. 送信元に接続されているマルチキャスト・ルーターは最初に登録メッセージ内の各マルチキャスト・パケットをカプセル化します。これらのメッセージが RP に送信されます。RP はそれらのユニキャスト・メッセージをカプセル化解除して、データ・パケットを一連のダウンストリーム受信側に転送します。この作用を図 119 に示します。

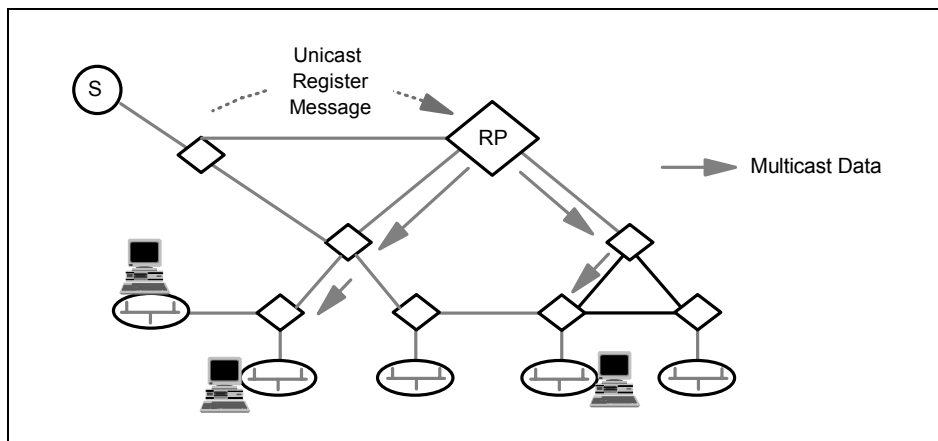


図 119 送信元の登録

3. RP 基点送達ツリーは一部の受信側にとっては 2 次最適な経路になる場合があります。これらの接続を最適化するために、ルーターは送信元基点マルチキャスト送達ツリーを作成することができます。この作用を図 120 に示します。

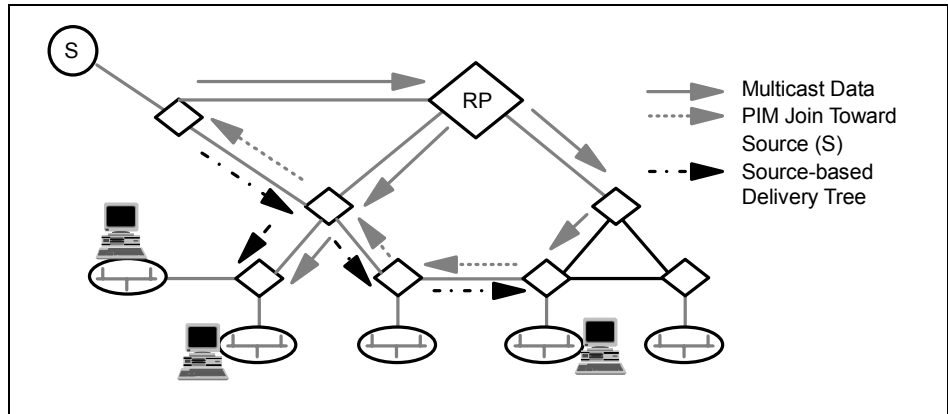


図 120 送信元基点送達ツリーの確立

4. ルーターが送信元基点送達ツリーと RP 基点送達ツリーの両方からマルチキャスト・パケットを受信すると、PIM 枝取りメッセージが RP に向けて送信されてツリーのその分岐が枝取りされます。その完了後は、送信元からのマルチキャスト・データは送信元基点送達ツリーだけを通ります。この作用を図 121 に示します。

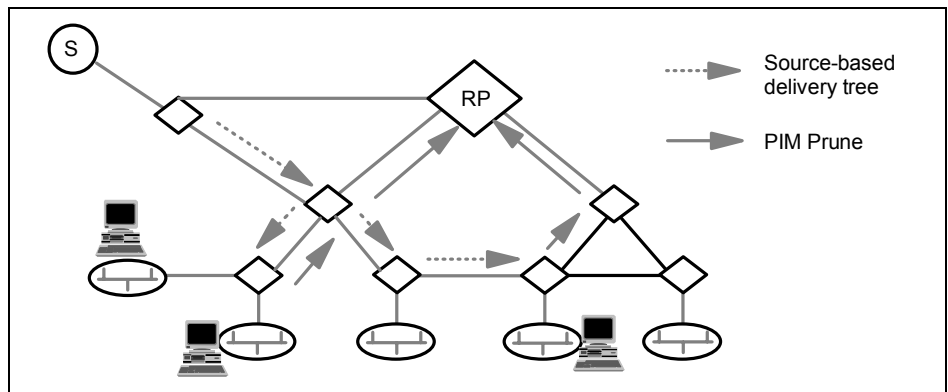


図 121 RP 基点送達ツリーの除去

送信元基点ツリーへの切り替えは、送信元から有効数のパケットを受信した後に行われます。このポリシーをインプリメントするためには、ルーターが RP 基点送達ツリーを介して受信したパケットの数量をモニターします。このデータの程度が構成済みのしきい値を超えると、装置は切り替えを開始します。

6.7.2.2 RP の選択

RP の選択は標準 PIM-SM 操作の一部として行われます。RP は特定の各マルチキャスト・グループにマップされます。このマッピングを行うには、PIM-SM をサポートするように構成されているルーターが、その環境内の他のすべてのルーターに対して候補 RP のリストを配布します。マッピングを行う必要があるときは、各ルーターがマルチキャスト・グループ・アドレスをハッシュして RP を表す IP アドレスにします。

6.7.2.3 PIM-SM の利点

PIM-SM はマルチキャスト・データ・ストリームが多数含まれる環境に合うように最適化されています。各ストリームは、比較的少数の LAN セグメントに流れなければなりません。これらのグループで、PIM-DM と DVMRP に関連したフラッドイングと枝取りを行えば、ネットワーク帯域幅を非効率的に使用することになります。PIM-SM は次の場合にも適用できます。

- グループ内に受信側が少ない。
- 送信側と受信側が WAN リンクによって分離されている。
- マルチキャスト・トラフィックのストリームが断続的である。

PIM-DM のように、PIM-SM でもユニキャスト・ルーティング・プロトコルによって取得した経路は、マルチキャスト・ルーティングをサポートしているものと想定しています。そのためネットワーク管理者は、エンドツーエンドのパスに接続された各装置をマルチキャストが可能になるようにしなければなりません。

6.8 マルチキャスト・ドメインの相互接続

初期のマルチキャストの開発では、フラット・ネットワークのトポロジーに焦点を合わせていました。これはインターネットにおいて展開されている階層トポロジーと異なるものです。マルチキャスト開発者は、インターネットのようにドメイン間ルーティングを行う必要性をすぐに実感しました。この欠陥に対処するために新しいクラスのプロトコルが提案されてきました。マルチキャスト・ドメインを相互接続する方法として現在 2 つあります。

6.8.1 Multicast Source Discovery Protocol (MSDP)

MSDP は複数の PIM-SM ドメインを論理的に接続するためのプロトコルです。これは他のドメイン内のアクティブなマルチキャスト送信元を検出するために使用されます。別の複数の自律システム内の複数の RP が MSDP を使って通信し、それらのマルチキャスト送信元についての情報を交換します。各ドメインではその独自の RP が継続して使用されます。ドメイン内通信では他ドメインの RP に直接従属することはありません。

通常は AS ごとに MSDP スピーカーの RP が 1 つ含まれます。他の自律システムではこの RP との MSDP ピア・セッションを作成します。それらのセッションを使って特定の各マルチキャスト・グループ内の送信元スピーカーのリストが交換されます。

MSDP はマルチキャスト・データの送達に直接関係するわけではありません。この目的は他のドメイン内の送信元を発見することです。あるドメイン内の装置が別のドメイン内の送信元からのデータを受信する必要がある場合は、そのデータは PIM-SM での標準操作を使って送達されます。

MSDP ピアは通常は複数のホップごとに分けられます。この遠隔通信をサポートするために使用される装置はすべてマルチキャストが可能なものでなければなりません。

MSDP はドメイン間通信のための Multiprotocol Extensions to BGP (MBGP) に大きく関係します。MBGP の概要は、263 ページの 6.8.1.3 『BGP-4 のマルチプロトコル拡張』で説明しています。

6.8.1.1 MSDP での操作

MSDP では、あるドメイン内の送信元と別のドメイン内の受信側の間のマルチキャスト通信を、以下のステップで確立します。

- 標準 PIM-SM 操作に従って、送信元の DR が PIM 登録メッセージを RP に送信します。そのデータ・パケットは RP によってカプセル化解除され、共用ツリーに沿って同じドメイン内の受信側に転送されます。
- パケットは送信元アクティブ (SA) メッセージの中にも再びカプセル化されます。このメッセージはすべての MSDP ピアに送信されます。SA メッセージは送信元とグループを識別するものです。この操作は送信元がアクティブになったときに行われます。

ドメインの RP は MSDP ピアから SA メッセージを受信すると、SA メッセージで示されているグループに関係するメンバーが自分の範囲にあるかどうかを判別します。関係するメンバーがあったときは、RP は送信元に向けて参加を発信します。パスが確立されて RP がデータを転送すると、受信側は送信元への直接の最短パス・ツリーに切り替えることができます。これは PIM-SM の標準のきまりに基づいて行われます。

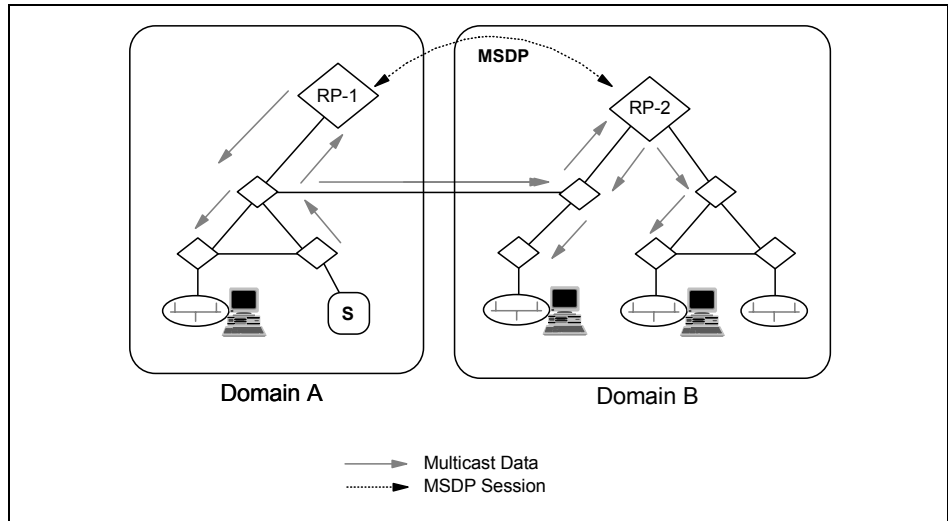


図 122 MSDP での操作

各 MSDP ピアが SA メッセージを受信して転送します。各ピアは MBGP ルーティング・テーブルを調べて、起点 RP へ向けてのネクスト・ホップ・ピアを決定します。ルーターは RPF ピア以外のすべての MSDP ピアに対して SA メッセージを転送します。ルーターがネクスト・ホップ・ピアではないピアから SA メッセージを受信すると、そのメッセージは廃棄されます。これによってネットワーク全体に通知のフラッディングが行われます。これは、246 ページの 6.4.1 『逆パス転送アルゴリズム』で説明した RPF でのプロセスに似ています。図 122 で MSDP での操作の概要を確認してください。

6.8.1.2 MSDP での制限

現在 MSDP は IETF 標準ではありませんが、多数のネットワーク環境に展開されています。MSDP に関連してフラッディングと枝取りの定期的なメッセージが必要なため、このプロトコルはインターネットで必要になるようなアドレスまでは適用されません。MSDP は Border Gateway Multicast Protocol (BGMP) に置き換わるだろうと予想されています。このプロトコルは 264 ページの 6.8.2 『Border Gateway Multicast Protocol』で説明しています。

6.8.1.3 BGP-4 のマルチプロトコル拡張

ドメインを相互接続するときにマルチキャスト・トラフィックをサポートしていない装置を含むパスがユニキャスト・ルーティングで選択される可能性があります。この場合はマルチキャスト参加メッセージが目的の宛先に到達しないことになります。

MSDP ではこの問題を解決するために MBGP プロトコルを使用します。このプロトコルは BGP で異なるプロトコル用に別のルーティング・テーブルを維持できるように一連の拡張が行われたものです。したがって、MBGP はユニキャストとマルチキャストの両方のトラフィックの経路を作成することができます。マルチキャスト経路は環境内のマルチキャストをサポートしていない部分の周囲を横断することができます。これによって、リンクをマルチキャスト・トラフィックに使用できるようになります。代わりに、トラフィックの各タイプをサポートするために使用されるリソースが限定されることがあります。

マルチキャスト経路に関連した情報を PIM が使用して配布ツリーが構築されます。フィルター操作と設定に関する標準サービスが MBGP で使用可能です。MBGP は RFC 2283 で定義されています。

6.8.2 Border Gateway Multicast Protocol

BGMP は共用ドメイン・ツリーを構築するマルチキャスト・ルーティング・プロトコルです。BGMP では PIM-SM のように送達ツリーのグローバル・ルートを選択します。ただし BGMP の場合のルートは単一のルーターではなくドメインになります。これによって、そのドメインへのいずれかのパスが使用可能なときは常にそのドメインへの接続が維持されるようになります。

ユニキャスト環境での EGP と IGP の間の連携のように、BGMP はドメイン間マルチキャスト・プロトコルとして使用されます。内部では任意のマルチキャスト IGP を使用できます。

BGMP は各ドメイン内のボーダー・ルーター間で使用されます。このプロトコルでは RP は使用しません。参加メッセージを使ってドメイン間のツリーが構成されます。ボーダー・ルーターは、ホストがいつドメイン間マルチキャスト・グループへの参加に関係するかをマルチキャスト IGP で確認しています。それが確認できると、ボーダー・ルーターはルート・ドメインへ参加メッセージを送信します。ピア装置がこの要求をルートに向けて転送します。マルチキャスト送達で使用する共用ツリーがこのようにして形成されます。

BGMP 仕様では特定のドメインにマルチキャスト・アドレスを割り振ることを要求しています。そのようにするためにこの仕様では Multicast Address-Set Claim (MASC) プロトコルの使用を推奨しています。MASC は別のプロトコルであり、これを使用するとドメインは一時的にある範囲のアドレスの対象であることを示すことができます。BGMP 仕様では MASC の使用を必須とはしていません。MASC は RFC 2909 に記載されています。

6.9 マルチキャスト・バックボーン

インターネット・マルチキャスト・バックボーン (MBONE) は 1992 年 3 月に確立されました。最初はマルチキャスト・プロトコルの実地の実験を行うために配置されました。最初の利用は IETF の会合の音声のマルチキャストでした。そのときはバックボーンに 20 サイトが接続されていました。その 2 年後には 15 か国の 500 以上の参加者に対して音声と映像の同時送信が行われました。それ以来 MBONE は NASA スペース・シャトルの活動、ロック・コンサート、および多数の技術会議のブロードキャストに使用されてきました。MBONE の商用および私用での利用は増え続けています。

マルチキャスト・バックボーンは、多くの物理インターネット・インフラストラクチャーを使った仮想オーバーレイ・ネットワークとして始まりました。その時点では、標準的なルーティング装置ではマルチキャスト・ルーティングはサポートされていませんでした。MBONE が最初にその存在を示したのは mrouted ルーティング・プロセスを構成された UNIX システムでした。現在でも MBONE は運用可能ですが、今では多くのインターネット・ルーターに初めからマルチキャスト接続が組み込まれています。MBONE をインターネット・インフラストラクチャーに直接組み込む活動が続けられています。

6.9.1 MBONE でのルーティング

マルチキャスト・トラフィックは、すべてのインターネット・ロケーションに向かって流れるわけではありません。そのようになるまでは、MBONE はマルチキャスト・ネットワークの島の集合で形成されると思われます。これらの島々は仮想トンネルを介して相互接続されます。トンネルは、マルチキャスト・トラフィックをサポートしていないエリアをブリッジするものです。

マルチキャスト・パケットを別のマルチキャストの島に送信する必要があるルーターは、そのパケットをユニキャスト・パケット内にカプセル化します。カプセル化されたそのパケットは標準的なインターネット・ルーターを介して伝送されます。ユニキャスト・パケット内に含まれている宛先アドレスはトンネルのエンドポイントです。トンネルのリモート・エンドにあるルーターはカプセル化ヘッダーを除去して、マルチキャスト・パケットを受信装置に転送します。

MBONE トンネルの構成の概要を図 123 に示します。

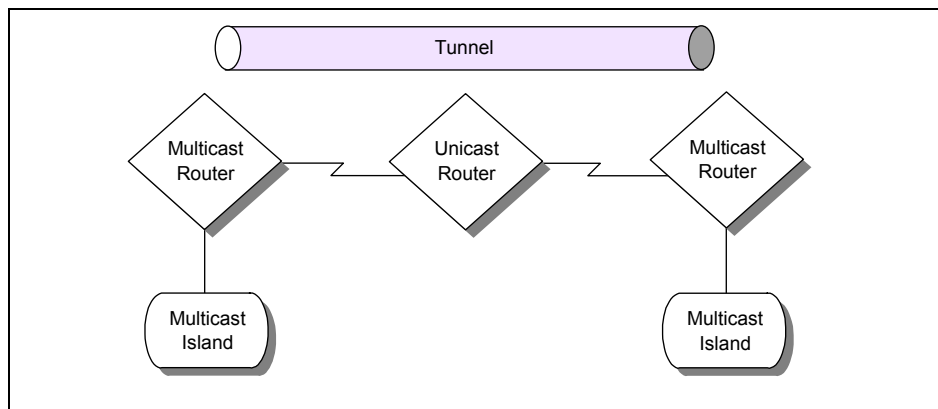


図 123 MBONE トンネル

MBONE トンネルには関連したメトリックとしきい値の各パラメーターがあります。メトリック・パラメーターはマルチキャスト・ルーティング・アルゴリズムでコストとして使用されます。ルーティング・アルゴリズムではこの値を使用してネットワークでの最適パスを選択します。図 124 は MBONE トンネルを介して相互接続されたマルチキャスト・サイトが 4 つある環境を示したものです。各トンネルには、ネットワーク上でトラフィック転送をスキューさせるための異なるメトリック値が割り当てられています。

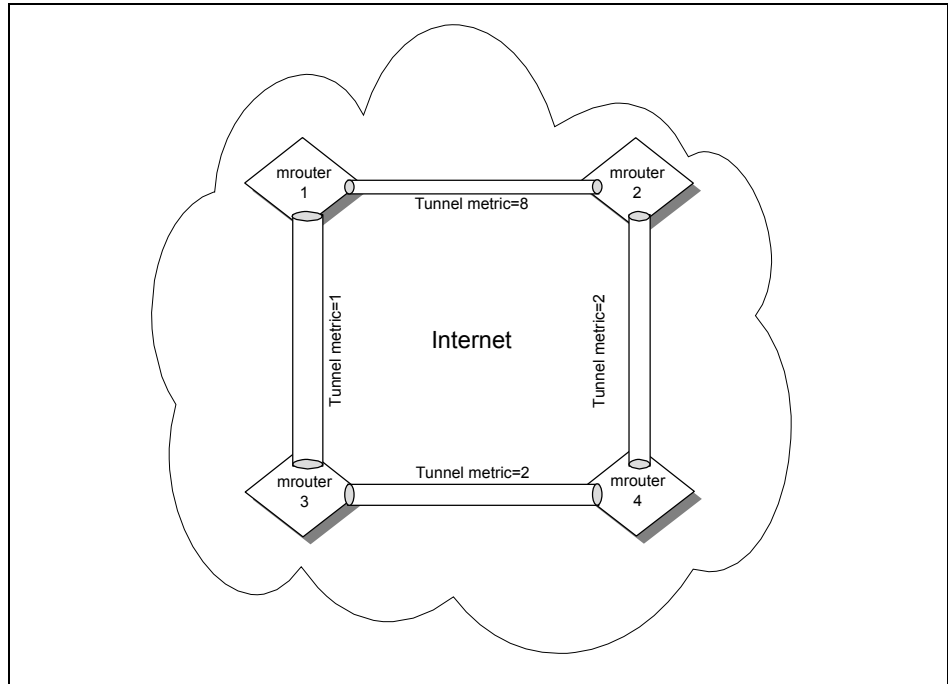


図 124 MBONE トンネルのメトリック

ルーター 1 からルーター 2 へ送信されるマルチキャスト・パケットでは、ルーター 1 とルーター 2 を直接接続しているトンネルは使用されません。ルーター 3 とルーター 4 を使用する代替パスのコストは 5 ($1 + 2 + 2$) です。このパスの方がルーター 1 とルーター 2 の間の直接パス (このパスのコストは 8) よりも適切です。

しきい値パラメーターでマルチキャスト・パケットの配布が制限されます。このパラメーターは、確立されているトンネルに転送されるマルチキャスト・パケットの最小 TTL を指定しています。TTL は各マルチキャスト・ルーターで 1 つずつ減分されます。

将来はほとんどのインターネット・ルーターが IP マルチキャストを直接サポートするようになると思われます。そのときはマルチキャスト・トンネルの必要性はなくなるでしょう。現在の MBONE のインプリメンテーションは一時的なソリューションにすぎません。すべてのインターネット・ルーターでマルチキャストが完全にサポートされると、これは旧式のものになるでしょう。

6.9.2 マルチキャスト・アプリケーション

最初のマルチキャスト・アプリケーションは音声会議機能を持つものでした。これらのアプリケーションの使用可能度と機能性は向上してきています。最近マルチキャスト・システムの開発が加速されています。次のことをサポートする新規アプリケーションや改良アプリケーションが出てきています。

- マルチメディア会議：これらのツールは MBONE 上で数年間使用されてきました。これらは音声のみまたは音声と映像の多対多の通信をサポートしています。ホワイトボード・アプリケーションと併用すると、これらを使用した会議でのコラボレーションを最小の帯域幅で高めることができます。
- データ配布：これらのツールには多数の受信側にデータを同時に送達できる機能があります。たとえば更新したデータ・ファイルを中央のサイトから地方の各オフィスへ効率的にプッシュすることができます。
- ゲームおよびシミュレーション：これらのアプリケーションは容易に使用できるようになっています。しかしマルチキャスト・サービスを組み込むと、これらのアプリケーションを多数のユーザーにまで広げることができます。マルチキャスト・グループをゲームやシミュレーションでの別々のセクションにすることができます。ユーザーは、あるセクションから次のセクションに移るときは、それまでのマルチキャスト・グループから離れて別のマルチキャスト・グループに参加します。
- リアルタイム・データのマルチキャスト：これらのアプリケーションは多数のユーザーにリアルタイム・データを配布するものです。たとえば株式相場情報を多数のワークステーションに表示することができます。マルチキャスト・グループを使用すれば、特定の装置に受信される情報を調整することができます。

これらのアプリケーションの多くは、通常の TCP トランスポートのサポートではなく UDP を使用しています。TCP では、マルチメディア・データのリアルタイム・ブロードキャストに合うような最適な信頼性とフロー制御メカニズムになっていません。わずかな割合のパケットが紛失する可能性があっても、その方が TCP に起因する伝送遅延よりむしろしばしば優先されています。

多くのアプリケーションでは、UDP に加えて Real-Time Transport Protocol (470 ページの 13.1.1 『Real-Time Transport Protocol (RTP)』を参照) を使用しています。このプロトコルを使用すると、インターネット上でマルチメディア・データ・ストリームを、さらに遅延が発生しないように連続して伝送できるメカニズムを使用できます。

第 2 部 TCP/IP アプリケーション・プロトコル

第 7 章アプリケーション構造およびプログラミング・インターフェース

最も上位のプロトコルをアプリケーション・プロトコルと呼びます。これらは他のホスト上のアプリケーションと通信を行う、TCP/IP プロトコル・スイートへのユーザー可視のインターフェースです。

7.1 アプリケーションの特性

上位のプロトコルにはすべて次のような共通の特性があります。

- それらはユーザー作成のアプリケーションか、または標準化されて TCP/IP 製品に同梱されたアプリケーションである。実際に TCP/IP プロトコル・スイートには次のようなアプリケーション・プロトコルが組み込まれています。
 - 対話式端末によるリモート・ホストへのアクセスのための TELNET。
 - 高速なディスク間ファイル転送のための FTP (ファイル転送プロトコル)。
 - インターネットのメーリング・システムとしての SMTP (Simple Mail Transfer Protocol)。

これらは最も広くインプリメントされているアプリケーション・プロトコルですが、この他にも多数あります。特定の TCP/IP インプリメンテーションにはそれぞれ、多少限定されたアプリケーション・プロトコル群が組み込まれるものと思われます。

- それらはトランスポートのメカニズムとして UDP または TCP を使用している。前述のように UDP (212 ページの 5.2 『User Datagram Protocol (UDP)』を参照) は信頼性が低くてフロー制御機能がないため、UDP を使用する場合はアプリケーション自体でエラー・リカバリーとフロー制御のルーチンを持たなければなりません。多くの場合、高信頼性で接続型のプロトコルの TCP (215 ページの 5.3 『Transmission Control Protocol (TCP)』を参照) 上でアプリケーションを構築する方が容易です。アプリケーション・プロトコルの大半は TCP を使用すると思われますが、プロトコルのオーバーヘッドを減らしてパフォーマンスを上げるために UDP 上に構築されるアプリケーションもあります。
- それらの大半はクライアント / サーバー・モデルの対話を使用している。

7.1.1 クライアント / サーバー・モデル

TCP はピアツーピアの接続型のプロトコルです。ここではマスター / スレーブの関係はありません。しかしアプリケーションでは通信にクライアント / サーバー・モデルを使用します (図 125 を参照)。

サーバーはユーザーにサービスを提供するアプリケーションです。一方クライアントはサービスの要求側です。アプリケーションはサーバー部分とクライアント部分の両方で構成され、これらは同一のシステム上で実行する場合も、異なるシステム上で実行する場合もあります。

ユーザーは通常はアプリケーションのクライアント部分を起動します。これで特定のサービスの要求を作成し、それをそのアプリケーションのサーバー部分へ送信します。このときトランスポート手段として TCP/IP が使用されます。

サーバーは要求を受け取り、要求されたサービスを実行し、そしてその結果を応答で送り返すプログラムです。サーバーは通常は複数の要求（複数のクライアント）を同時に扱うことができます。

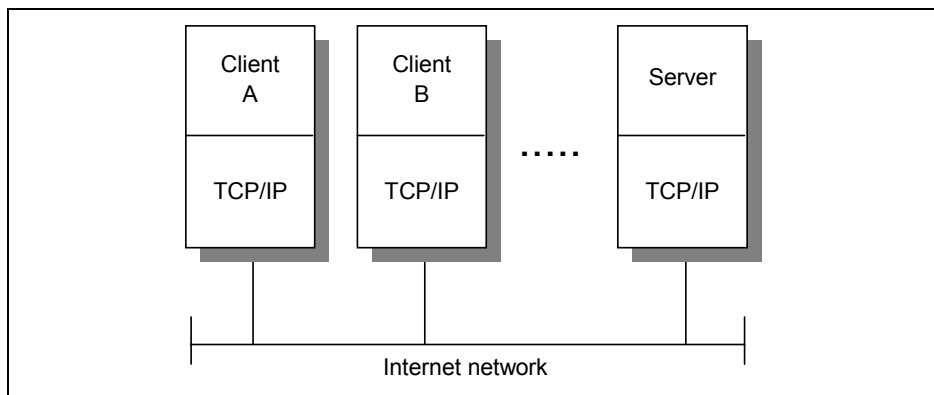


図 125 アプリケーションのクライアント/サーバー・モデル

サーバーの中にはウェルノウン・ポート (209 ページの 5.1 『ポートとソケット』を参照) で要求を待つものがあり、この場合クライアントは自分の要求をどの IP ソケットへ送信すればよいのか分かっています。クライアントは通信に任意のポートを使用します。ウェルノウン・ポートを使用しないサーバーと通信しようとするクライアントは、要求の宛先のポートを知るための別のメカニズムが必要です。このメカニズムとしてポート・マップなどの登録サービスを使用できる場合があります、これでウェルノウン・ポートを使用します。

7.2 アプリケーション・プログラミング・インターフェース (API)

開発者は、アプリケーション・プログラミング・インターフェースを使用すると、TCP/IP サービスを利用できるアプリケーションを作成することができます。続くいくつかのセクションでは、TCP/IP アプリケーションのための最も一般的な API の概要を説明します。

7.2.1 ソケット API

ソケット・インターフェースは、通信プロトコルとのアプリケーション・プログラミング・インターフェース (API) がいくつかある中の 1 つです。汎用通信プログラミング・インターフェースになるように設計されており、最初は 4.2BSD UNIX ベースのシステムに採用されました。標準化はされていませんが、これは業界のデファクト・スタンダードになっています。

ソケット・インターフェースはアプリケーションで使用するサービスごとに、ストリーム・ソケット (コネクション型)、データグラム・ソケット (コネクションレス)、ロー・ソケット (下位層プロトコルへの直接アクセス) の各サービスに分けられます。ポートとソケットの詳細については、209 ページの 5.1 『ポートとソケット』を参照してください。

Microsoft や他のベンダーによって開発された Winsock インターフェースでは、BSD ソケット・インターフェースの変種が提供されており、Windows オペレーティング・システム上の TCP/IP アプリケーションをサポートしています。最新バージョンの Winsock 2.0 ではさらに汎用化されたインターフェースになっており、アプリケーションは TCP/IP に限らず使用可能な任意のトランスポート層プロトコルおよび下層のネットワーク・サービスとの通信を行うことができます。Winsock の詳細については、282 ページの 7.2.3 『Windows Sockets Version 2 (Winsock V2.0)』を参照してください。

7.2.1.1 基本的なソケット呼び出し

基本的なソケット・インターフェース呼び出しのいくつかを以下に示します。次のセクションでは、これらのソケット・インターフェース呼び出しを使用したシナリオを例として取り上げます。

- ソケットを初期化する。

フォーマット：

```
int sockfd = socket(int family, int type, int protocol)
```

各部の説明：

- family はアドレス・ファミリーを表します。使用できる値は AF_UNIX、AF_INET、AF_NS、AF_OS2、AF_IUCV などです。ソケットで使われるアドレッシングのメソッドをこれで指定することが目的です。
- type は使用されるソケット・インターフェースのタイプを表します。使用できる値は SOCK_STREAM、SOCK_DGRAM、SOCK_RAW、SOCK_SEQPACKET などです。
- protocol は UDP、TCP、IP、ICMP のいずれかにします。
- sockfd は socket 呼び出しで戻される (ファイル記述子に似た) 整数です。これは以後の socket 呼び出しにパラメーターとして渡されます。

- ソケットをポート・アドレスへバインド (登録) する。

フォーマット:

```
int bind(int sockfd, struct sockaddr *localaddr, int addrlen)
```

ここで localaddr は bind 呼び出しで戻されるローカル・アドレスです。

なお bind 呼び出し後は、次の 5 つのタプルの集合のうちの最初の 3 つのパラメーターの値が分かります。

```
{protocol, local-address, local-process, foreign-address, foreign-process}
```

- 着信接続を listen する。接続を受け付ける状態になっていることを示します。

フォーマット:

```
int listen(int sockfd, int queue-size)
```

ここで queue-size は、ローカル・プロセスが accept 呼び出しをまだ発行していない間にシステムがキューに入れられる接続要求の数を示します。

- 接続を受諾する。

フォーマット:

```
int accept(int sockfd, struct sockaddr *foreign-address, int addrlen)
```

ここで foreign-address は、accept 呼び出しで戻される外部 (クライアント) プロセスのアドレスです。

この accept 呼び出しは、クライアント・プロセスではなくサーバー・プロセスによって発行されることに注意してください。このソケット接続のキューに待機中の接続要求がある場合は、そのキューにある 1 番目の要求が受諾され、sockfd と同じプロパティを持つ別のソケットが作成されます。そうでない場合は、接続要求が到着するまで呼び出し側プロセスは accept によってブロックされます。

- サーバーに対して接続する。

フォーマット:

```
int connect(int sockfd, struct sockaddr *foreign-address, int addrlen)
```

ここで foreign-address は、connect 呼び出しで戻される外部 (サーバー) プロセスのアドレスです。

この呼び出しは、サーバー・プロセスではなくクライアント・プロセスによって発行されることに注意してください。

- データを送受信する。

read()、readv(sockfd, char *buffer int addrlen)、recv()、readfrom()、send(sockfd, msg, len, flags)、write() の各呼び出しを使って、確立されたソケット・アソシエーション (接続) でデータの送受信を行うことができます。

これらの呼び出しは、標準的なファイル入出力システムの読み取りと書き込みの呼び出しに似ていることに注意してください。

- ソケットをクローズする。

フォーマット：

```
int close(int sockfd)
```

詳細については、951 ページの付録 C『関連資料』に記載の資料を参照してください。

7.2.1.2 シナリオの例

コネクション型プロトコルの場合のソケットのシステム呼び出しの例を次に示します。

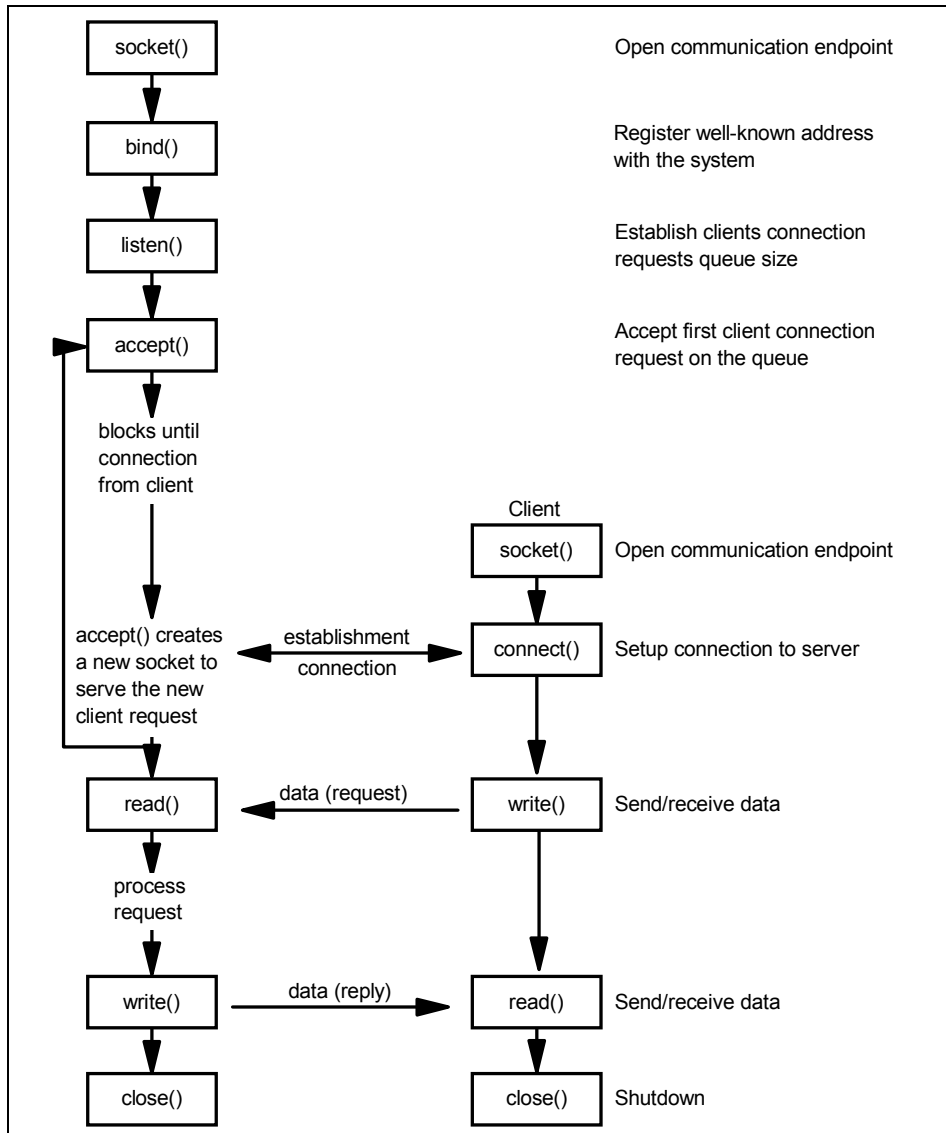


図 126 コネクション型プロトコルの場合のソケットのシステム呼び出し

コネクションレスの場合のシナリオはこれより単純になります。その場合は listen、accept、connect の各段階が省略されます。

次に示すのはコネクション型の場合とコネクションレス方式の場合のシステム呼び出しを比較したものです。

表3 ソケットのシステム呼び出し

タイプ	確立	送信	受信
コネクション型サーバー	bind、listen、 または accept	write または sendto	read または recvfrom
コネクション型クライアント	connect	write または sendto	read または recvfrom
コネクションレス・サーバー	bind	sendto	recvfrom
コネクションレス・クライアント	bind	sendto	recvfrom

ソケット・インターフェースは使用されるサービスごとに異なります。ストリーム、データグラム、およびローの各ソケットごとにアプリケーションで利用できるサービスが定義されており、それらのサービスはそれぞれ異なります。

- ・ストリーム・ソケット・インターフェース (SOCK_STREAM):
高信頼性コネクション型サービス（たとえば TCP 上のサービス）を定義しています。データはエラーや重複なしに送信され、送信されたときと同じ順序で受信されます。データのオーバーランが起これないようにフロー制御が組み込まれています。交換データ上には境界が設定されていないため、これはバイトのストリームと見なすことができます。ストリーム・ソケットを使用するアプリケーションの例として、ファイル転送プログラム (FTP) があります。
- ・データグラム・ソケット・インターフェース (SOCK_DGRAM):
コネクションレス・サービス（たとえば UDP 上のサービス）を定義しています。データグラムは独立パケットとして送信されます。このサービスではデータの紛失や重複が起きたり、データグラムが順序どおりに到着しないことがあり、これらについては保証はありません。パケットの分解と再組み立ては行われません。データグラム・ソケットを使用するアプリケーションの例として、Network File System (NFS) があります。
- ・ロー・ソケット・インターフェース (SOCK_RAW):
これを使用して IP や ICMP などの下位層プロトコルに直接アクセスすることができます。このインターフェースは、新しいプロトコルのインプリメンテーションをテストするときによく使われます。ロー・ソケットを使用するアプリケーションの例として、ping コマンドがあります。

7.2.2 リモート・プロシージャ・コール (RPC)

リモート・プロシージャ・コールは Sun Microsystems によって開発された標準で、UNIX ベースのシステムの多くのベンダーで使用されています。

RPC は分散アプリケーションの開発に使用できるアプリケーション・プログラミング・インターフェース (API) です。これを使用すると、プログラムはリモート・システムで実行されるサブルーチンを呼び出すことができます。呼び出し側プログラム (いわゆるクライアント) はサーバー・プロセスへ呼び出しメッセージを送信し、応答メッセージを待ちます。呼び出しメッセージにはプロシージャのパラメーターが組み込まれており、応答メッセージにはプロシージャの結果が含まれています。RPC では、データを異なるシステム間でのポータブルな形式にエンコードするための標準的な方法「外部データ表現」(XDR) も使用します。

7.2.2.1 RPC の概念

RPC の概念はプロシージャ呼び出しを発行するアプリケーション・プログラムの概念に非常によく似ています。

- ・ 呼び出し側プロセスは呼び出しメッセージを送信してその応答を待ちます。
- ・ サーバー・サイドではプロセスは休止しており、呼び出しメッセージが到着するのを待っています。サーバー・プロセスは、ある呼び出しメッセージが到着すると、プロシージャのパラメーターを抽出して結果を計算し、それを応答メッセージで送り返します。

図 127 に RPC の概念モデルを示します。

これは 1 つのモデルとして想定したものであり、Sun RPC プロトコルでは並行性モデルについての制約事項はありません。このモデルでは、呼び出し側の実行は応答メッセージを受信するまでブロックされます。他のモデルも考えられます。たとえば呼び出し側が応答を待ちながら処理を続けたり、サーバーが着呼ごとに別々のタスクをディスパッチして、他のメッセージを受信できるようにフリーの状態にしておくなどです。

リモート・プロシージャ・コールは、ローカルのプロシージャ呼び出しと次の点が異なります。

- ・ サーバーは、呼び出し側プログラムのアドレス・スペースにアクセスしないため、グローバル変数を使用する。
- ・ 伝送回数によってパフォーマンスが影響を受けることがある。
- ・ ユーザー認証が必要な場合がある。
- ・ サーバーのロケーションが分かっている必要はない。

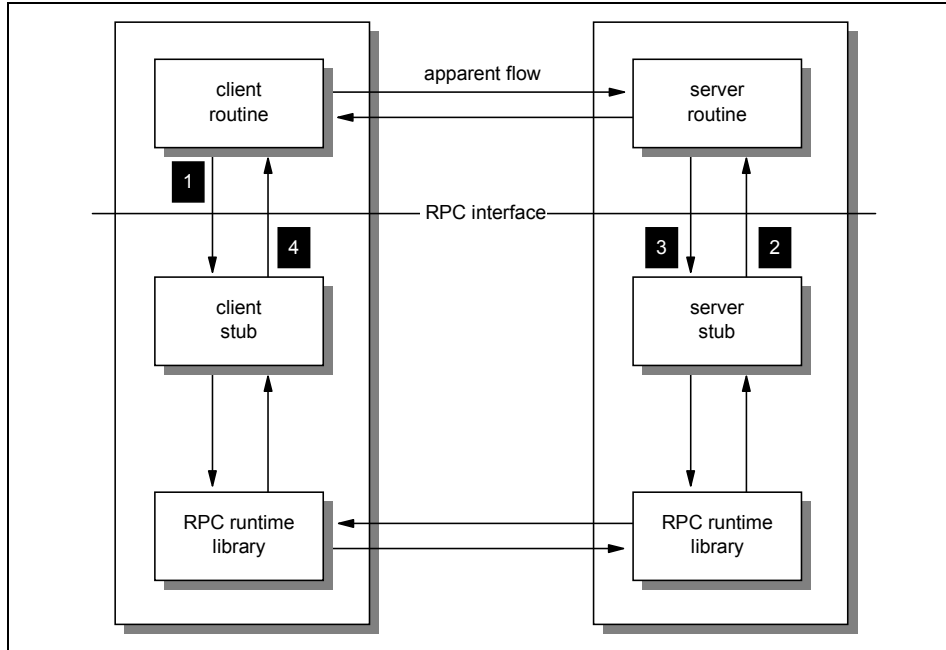


図 127 RPC - リモート・プロシージャールール・モデル

トランスポート

RPC プロトコルは任意のトランスポート・プロトコル上にインプリメントすることができます。TCP/IP の場合はトランスポート手段として TCP か UDP を使用します。トランスポートのタイプは RPCGEN コマンドのパラメーターになっています。UDP を使用する場合、前述したようにこれは信頼性が低いため、信頼性の確保 (RPC ライブラリー・ルーチンに通常インプリメントされているタイムアウトと再送を使用) は、呼び出し側プログラム自体で決まることになります。なお、TCP を使用する場合でも、サーバーのクラッシュなどの例外状態に対処するために、呼び出し側プログラムにはやはりタイムアウト・ルーチンが必要です。

呼び出しメッセージと応答メッセージのデータは XDR 標準にフォーマットが設定されます。

RPC 呼び出しメッセージ

RPC 呼び出しメッセージは、以下のフィールドで構成されています。

- ・プログラムおよびプロシージャールの番号

各呼び出しメッセージには、実行するプロシージャーを一意的に識別するための次の 3 つのフィールド (符号なし整数) が含まれています。

- リモート・プログラム番号
- リモート・プログラム・バージョン番号
- リモート・プロシージャ番号

リモート・プログラム番号で、たとえばファイル・システムなど、プロシージャの機能グループを識別します。プロシージャの機能グループには、たとえば読み取りや書き込みなど、個々のプロシージャが含まれています。これらの個々のプロシージャは、リモート・プログラム内の固有のプロシージャ番号で識別されます。リモート・プログラムは進化するため、リリースが異なるごとにバージョン番号が割り当てられます。

各リモート・プログラムはインターネット・ポートに接続されます。このポートの番号は、予約済みのウェルノウン・サービスのポート番号を除いて、自由に選択することができます。当然のことですが、呼び出し側はこのリモート・プログラムで使用するポート番号を、分かっている必要があります。

割り当て済みのプログラム番号は次のとおりです。

00000000 - 1FFFFFFFSun が定義

20000000 - 3FFFFFFFユーザーが定義

40000000 - 5FFFFFFF一時的 (一時的番号)

60000000 - FFFFFFFF予約済み

• 認証フィールド

サービスでの呼び出し側の認証のために、**クリデンシャル**と**ベリファイヤー**の2つのフィールドがあります。この情報をユーザー認証に使用するかどうかはサーバーで決めます。またインプリメンテーションごとに、サポートされているさまざまな認証プロトコルを自由に選択することができます。次の認証プロトコルがあります。

- Null 認証。
- UNIX 認証。リモート・プロシージャの呼び出し側は、自分は UNIX システム上で確認されているものとして示すことができます。
- DES 認証。ユーザー ID に加えてタイム・スタンプ・フィールドがサーバーに送信されます。タイム・スタンプは現在時刻ですが、これは呼び出し側マシンとサーバー・マシンだけが分かっている鍵を使用して、暗号化されます (DES の **秘密鍵** と **公開鍵** の概念に基づいています)。

• プロシージャ・パラメーター

リモート・プロシージャに渡されるデータ (パラメーター)。

RPC 応答メッセージ

応答は行われた処理によって異なり、次のものがあります。

- SUCCESS: プロシージャの結果がクライアントに送り返されます。
- RPC_MISMATCH: サーバーは呼び出し側とは別のバージョンの RPC を実行中です。
- AUTH_ERROR: 呼び出し側認証に失敗しました。
- PROG_MISMATCH: プログラムを使用できない場合、または要求されたバージョンが存在していない場合、またはプロシージャーを使用できない場合。

呼び出しメッセージと応答メッセージの詳細な説明は、RFC 1057 – RPC: Remote Procedure Call Protocol Specification Version 2 を参照してください。ここには XDR 言語でのメッセージのタイプ定義 (typedef) も含まれています。

ポート・マップまたはポートマッパー

ポート・マップまたはポートマッパーは、プログラム番号とそのバージョン番号をプログラムで使用するインターネット・ポート番号にマップするサーバー・アプリケーションです。ポート・マップには予約済み (ウェルノウン・サービス) ポート番号 111 が割り当てられています。

ポート・マップは、自分が実行されているホスト上の RPC プログラムだけについて認識します。ポート・マップに RPC プログラムについて認識させるためには、各 RPC プログラムが始動時に自分自身をローカル・ポートマッパーに登録する必要があります。

RPC クライアント (呼び出し側) は、必要なサーバー・プログラムで使われるポートについて、リモート・ホスト上のポート・マップ・サービスに問い合わせを行う必要があります。

呼び出し側アプリケーションは通常、宛先ホスト上のポート・マップに問い合わせで特定のリモート・プログラムの正しいポート番号を取得してから、その特定のポートに対して呼び出しメッセージを送信することになります。呼び出し側がプロシージャー・データもポート・マップへ送信し、そのリモート・ポート・マップが直接プロシージャーを起動するという別の場合もあります。

詳細については、図 128 を参照してください。

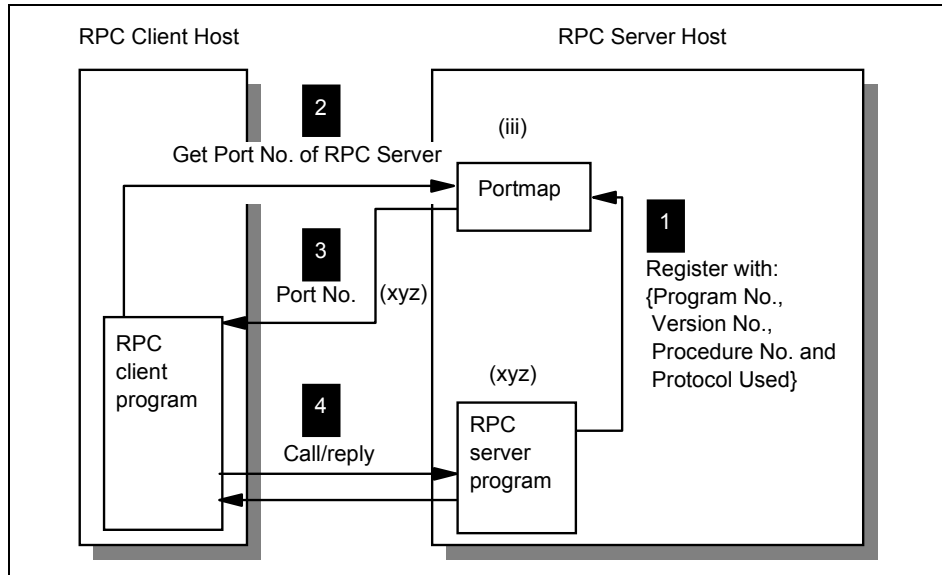


図 128 RPC - ポート・マップ

RPCGEN

RPCGEN は RPC プロトコルをインプリメントするための C コードを生成するツールです。RPCGEN に対する入力、C に似た言語 (RPC 言語と呼ばれる) で作成されたファイルです。使用する入力ファイルの名前を proto.x とすると、RPCGEN によって次の出力ファイルが生成されます。

- proto.h というヘッダー・ファイル。これには定数とマクロの共通定義が含まれています。
- クライアント・スタブのソース・ファイル protoc.c。
- サーバー・スタブのソース・ファイル protos.c。
- XDR ルーチンのソース・ファイル protox.c。

7.2.3 Windows Sockets Version 2 (Winsock V2.0)

Winsock V2.0 はネットワーク・プログラミング・インターフェースです。これは基本的には Berkeley Sockets を Microsoft Windows オペレーティング・システムに適応させてさらに機能を拡張したものです。前のバージョンの Winsock API である Windows Sockets V1.1 は広くインプリメントされています。そのため Winsock V2.0 は Winsock V1.1 との上位互換性を保っていますが、その他さらに多くの機能を備えています。Winsock V2.0 の中で最も重要な性質の 1 つは、プロトコルに依存しないトランスポート・インターフェースを備えており、これによってさまざまなネットワーキング機能をサポートしていることです。さらに Winsock V2.0 では複数のプロトコル・スタックの

共存もサポートしています。Winsock V2.0 の新しい機能は次のように要約することができます。

- マルチプロトコルのサポート
- プロトコルに依存しないネーム解決
- サービス品質
- プロトコルに依存しないマルチキャストとマルチポイント
- 入出力とイベント・オブジェクトのオーバーラップ
- ソケット共用
- 階層化サービス・プロバイダー

Winsock V1.1 アーキテクチャーでは、システム上で同時に使用できるのは WINSOCK.DLL か WSOCK32.DLL のいずれか 1 つのダイナミック・リンク・ライブラリー (DLL) です。この DLL は、下位のトランスポート・プロトコルとの通信の手段である Winsock API を備えています。この方法では、さまざまなタイプの Winsock インプリメンテーションを、Winsock V1.1 と共に使用することが制限されます。複数のネットワーク・インターフェースを持つシステムの場合、これは障害になる可能性があります。Winsock V2.0 ではこの問題がかなり解決されています。Winsock V2.0 アーキテクチャーでは複数のプロトコル・スタック、インターフェース、およびサービス・プロバイダーの同時サポートが可能です。

詳細については、図 129 を参照してください。

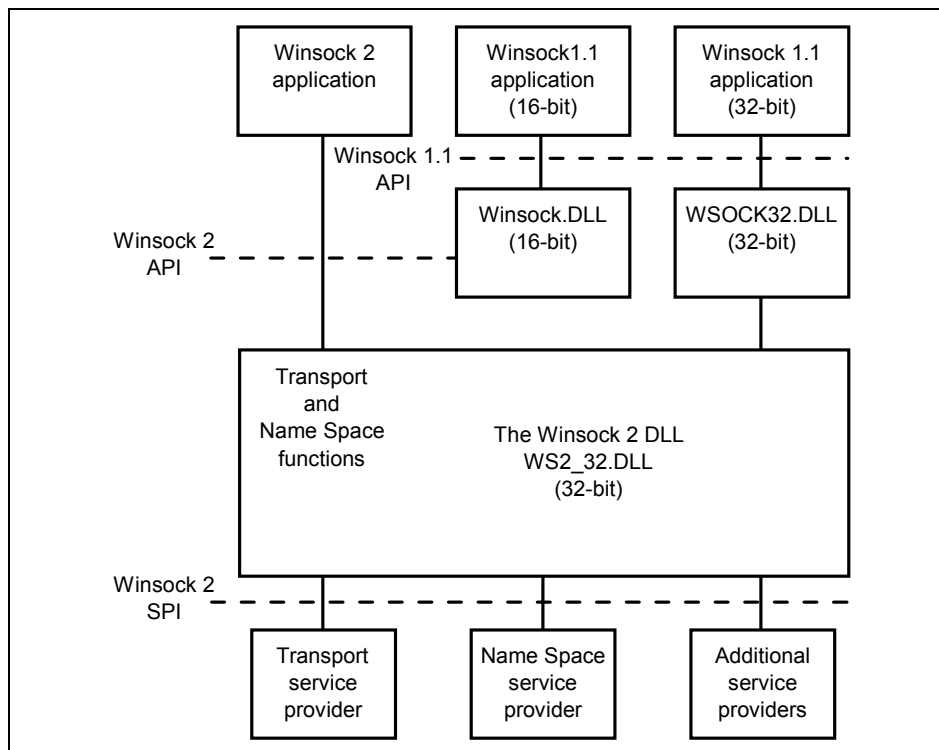


図 129 Winsock V2.0 のアーキテクチャー

7.2.4 SNMP 分散プログラミング・インターフェース (SNMP DPI)

SNMP では変数の収集の操作が可能なプロトコルを定義しています。変数のこのセット (MIB) と変数のコア・セットは以前に定義されています。ただし MIB の設計はこのコア・セットを拡張できるようになっています。しかし残念なことに従来の SNMP エージェントのインプリメンテーションでは、エンド・ユーザーが新しい変数を使えるようにできる手段を備えていません。SNMP DPI はこの問題に軽快なメカニズムで対処したものです。これを使用するとエンド・ユーザーは、SNMP エージェントを再コンパイルする必要なしに、ローカル MIB 内の管理変数を動的に追加、削除、および置き換えることができます。これは SNMP DPI を介してエージェントと通信を行う、いわゆるサブエージェントを作成することによって実現します。これは RFC 1592 に記載されています。

SNMP DPI を使用すると、プロセスが MIB 変数の存在を SNMP エージェントに登録することができます。SNMP エージェントは変数の要求を受け取ると、その照会をサブエージェントとして機能するプロセスに渡します。次にこのサブエージェントは特定の応答を SNMP エージェントへ戻します。最後

に SNMP エージェントは SNMP 応答パケットをパックして、要求を開始したリモート・ネットワーク管理ステーションへその応答を送り返します。SNMP エージェントが他のプロセスを呼び出して応答を得ていることは、どのリモート・ネットワーク管理ステーションからもまったく分かりません。

SNMP エージェントとそのクライアント (サブエージェント) の間の通信は、ストリーム接続で行われます。これは通常は TCP 接続ですが、他のストリーム指向のトランスポート・メカニズムも使用できます。(たとえば VM SNMP エージェントでは IUCV での DPI 接続が可能です。)

SNMP エージェント DPI で次のことを行うことができます。

- MIB 内のサブツリーを作成して削除する。
- サブエージェントのための登録要求パケットを作成して SNMP エージェントに通知する。
- サブエージェントのための応答パケットを作成して SNMP エージェントの要求に応答する。
- 「トラップ」要求パケットを作成する。

図 130 は SNMP エージェントとサブエージェントの間のフローを示したものです。

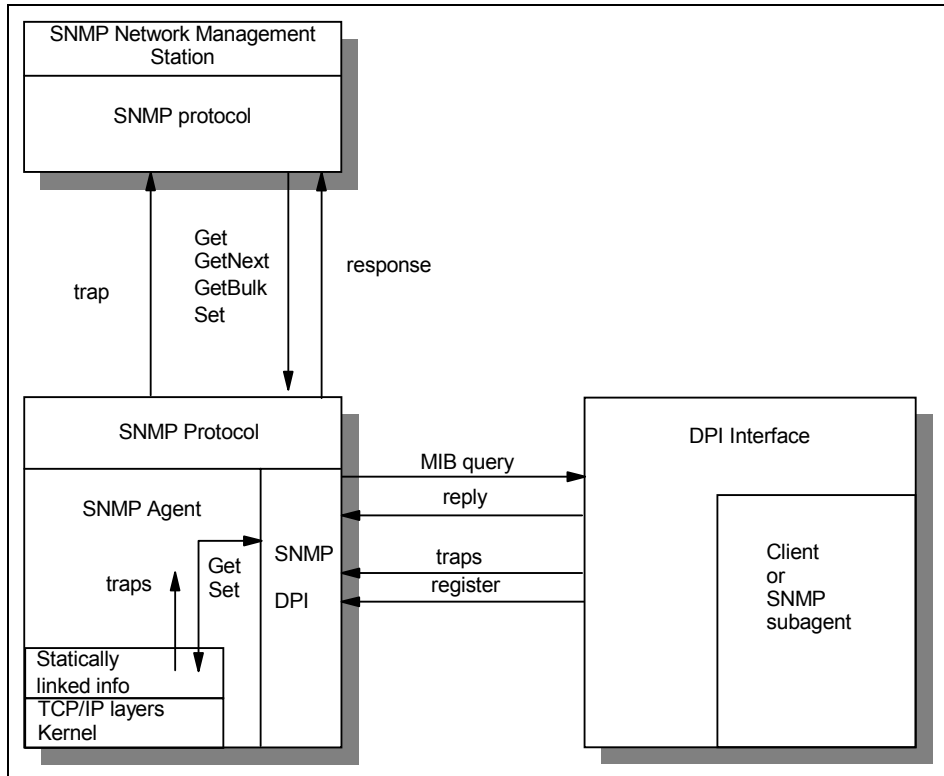


図 130 SNMPDPI の概要

- SNMP エージェントは SNMP プロトコルを介して SNMP マネージャーと通信を行います。
- SNMP エージェントは静的にリンクされた何らかのインストールメンテーション (おそらく MIB II 用のもの) と通信を行います。次にそのインストールメンテーションがインプリメンテーション依存の方法で TCP/IP 層とカーネル (オペレーティング・システム) と通信を行います。
- 別のプロセスとして (おそらく別のマシン上で) 実行されている SNMP サブエージェントが、エージェントとの接続をセットアップします。サブエージェントは SNMP エージェントとの通信を UDP または TCP ソケットで行うか、あるいは他のメカニズムで行うかを選択します。
- 接続が確立されると、サブエージェントは SNMP エージェントに 1 つ以上の MIB サブツリーを登録するために、DPI OPEN と 1 つ以上の REGISTER の各要求を発行します。
- SNMP エージェントは DPI OPEN と REGISTER の各要求に対して RESPONSE パケットで応答して、正常に行えたか失敗したかを示します。

- SNMP エージェントは SNMP パケットをデコードします。サブエージェントによって登録された、サブツリー内のオブジェクトの Get または GetNext 要求がそのパケットに含まれていた場合は、それに対応する DPI パケットをサブエージェントに送信します。要求が GetBulk だった場合は、エージェントはそれを複数の DPI GETNEXT パケットに変換して、それらをサブエージェントに送信します。ただしサブエージェントは、GETBULK をそのサブエージェントに渡すように (REGISTER パケットの中で) 要求することができます。要求が Set だった場合は、エージェントは 2 段階のコミット方式を使って、SET/COMMIT、SET/UNDO、または SET/COMMIT/UNDO DPI の連続したパケットをサブエージェントに送信します。
- SNMP サブエージェントは RESPONSE パケットで応答を送り返します。
- 次に SNMP エージェントはその応答を SNMP パケットにエンコードして、それを要求側の SNMP マネージャーに送り返します。
- サブエージェントが重要な状態変化をレポートする必要がある場合は、サブエージェントは DPI TRAP パケットを SNMP エージェントに送信します。SNMP エージェントはそれを SNMP トラップ・パケットにエンコードしたものをマネージャーに送信します。
- サブエージェントは、操作を停止する必要がある場合は、DPI UNREGISTER と DPI CLOSE の各パケットをエージェントに送信します。エージェントは UNREGISTER 要求に対する応答を送信します。
- CLOSE に対する RESPONSE はありません。この場合エージェントは DPI 接続をクローズするだけです。CLOSE には、クローズされる DPI 接続のために行われたすべての登録に対する UNREGISTER が暗黙に含まれています。
- エージェントは (より優先順位が高い登録に対応するかまたはその他の理由の場合に) DPI UNREGISTER をサブエージェントに送信することができます。サブエージェントはそれに対して DPI RESPONSE パケットで応答します。
- エージェントは (理由が何であれ) DPI 接続を終了させることを示すために、DPI CLOSE を送信することもできます。
- サブエージェントは ARE_YOU_THERE を送信して、接続がまだオープンであることを確認することができます。その場合はエージェントはエラーなしという RESPONSE を送信しますが、そうでない場合はエラーありという RESPONSE を送信します。

7.2.5 FTP API

ファイル転送プロトコル (FTP) API は、OS/2 用 TCP/IP の一部として提供されています。これを使用して、アプリケーションにファイル転送用のクライアント・インターフェースを持たせることができます。アプリケーションがこのインターフェースで作成されていれば、それらの複数のアプリケーションが複数の FTP サーバーと同時に通信を行うことができます。この場合、最

大 256 までの同時接続が可能で、FTP サーバーのペア間の第三者プロキシー転送が可能です。FTP サーバーのペア間を任意の順序で連続して第三者転送することができます。このようなアプリケーションの例として FTPPM があります。

FTP API はアプリケーションの現行接続先のサーバーを追跡します。FTP サービスに対して新しい要求があると、API はそのサーバーに接続済みであるかどうかを検査し、そうでない場合は接続を確立します。サーバーが接続を最後の使用後に切っていた場合は、API は接続を再度確立します。

注：

FTP API は再入可能ではありません。マルチスレッド・プログラムではこの API へのアクセスは逐次化しなければなりません。逐次化をしないと、たとえばプログラムのスレッドが 2 つ並行して実行されていて、各スレッドがそれぞれサーバーに接続していると、そのプログラムは失敗する場合があります。

7.2.6 CICS ソケット・インターフェース

顧客情報管理システム (CICS) は高性能トランザクション処理システムです。これは IBM によって開発されたもので、MVS/ESA、MVS、VSE、OS/400、OS/2、AIX/6000 での製品にインプリメントされています。

CICS は市場で現在最も広く使用されているオンライン・トランザクション処理 (OLTP) システムです。CICS は、(SNA を使った) データ通信および (VSAM や IMS または DB2 を使った) データベース用のアプリケーション・トランザクション・プログラム向けに、CICS コマンド・レベル API を豊富に備えています。

異種ネットワーク・プロトコル間のインターオペラビリティが必要なため、CICS データ通信インターフェースを拡張して SNA に加えて TCP/IP のサポートを組み込むことが必要であるという要求があります。CICS 用 IBM ソケット・インターフェースは、その要求に応えるための最初のステップです。

7.2.7 IMS ソケット・インターフェース

IMS ソケット・インターフェースは、OS/390 V2R5 用 TCP/IP にインプリメントされています。

IMS から TCP/IP へのソケット・インターフェースを使用すると、他の TCP/IP ホスト内の対等プログラムと会話できる IMS メッセージ処理プログラムを開発することができます。このアプリケーションは、クライアント・アプリケーションまたはサーバー・アプリケーションにすることができます。

IMS から TCP/IP へのソケット・インターフェースには、データグラム (コネクションレス) とストリーム (コネクション型) の各ソケットを使用できるように、IBM C/370、アセンブラー言語、COBOL、PL/I の各言語用のソケット・インターフェースが組み込まれています。また、ASCII-EBCDIC 変換ルーチン、TCP/IP 通信に対する従来の IMS 呼び出しを使用できる ASSIST モジュール、およびリスナー機能も備えています。listener 機能では、受信した接続要求を受諾し、その要求にサービスするために適切な IMS トランザクションを開始します。

7.2.8 Sockets Extended

ここに記載している Sockets Extended の情報は MVS でのインプリメンテーションだけに関係します。

Sockets Extended は、アセンブラー言語、COBOL、PL/I で作成できるアプリケーション・プログラム・インターフェースです。これを使って TCP/IP ネットワーク内の他のホストとのピアツーピアの会話を行うことができます。この API を使用して TSO、バッチ、CICS、IMS 用のアプリケーションを開発できます。それらのアプリケーションは、アプリケーションの要件によっては、再入可能でマルチスレッドのものを設計することができます。一般にサーバー・アプリケーションはマルチスレッド化しますが、クライアント・アプリケーションはマルチスレッド化しない場合があります。

7.2.9 REXX ソケット

REXX ソケットを使用して、TCP/IP ネットワーク上で通信を行う REXX アプリケーションを開発することができます。ソケットの初期化、ソケットを介したデータの交換、管理アクティビティーの実行、およびソケットのクローズのための呼び出しが用意されています。

REXX ソケット API は、MVS および OS/2 用 TCP/IP にインプリメントされています。

第 8 章ディレクトリーおよびネーミングの Protokol

TCP/IP Protokol・スイートにはアプリケーションが多数含まれています。それらは一般にネットワーク・ユーティリティーの形式になっています。それらはネットワークを使用する企業にとって当然重要なものですが、そのもの自体が企業がネットワークに投資する第一の理由ではありません。ネットワークは、たとえば建物や都市あるいは世界中の多数のサーバーに分散している企業のビジネス・アプリケーション、データ、およびリソースに、ローカルでリモートの場合もあるユーザーがアクセスできるようにするものです。それらのサーバーは多数の異なるベンダーのハードウェア上で、そしていくつかの異なるオペレーティング・システム上で実行されていると考えられます。この章では、分散ネットワークにあるリソースとアプリケーションにアクセスするためのメソッドを取り上げます。

8.1 Domain Name System (DNS)

Domain Name System は STD 番号 13 の標準 Protokol です。その状態は「推奨 (Recommended)」です。これは RFC 1034 と RFC 1035 に記述されています。このセクションでは、Domain Name System のインプリメンテーションとネーム・サーバーのインプリメンテーションについて説明します。

初期のインターネットの構成では、ユーザーは数字の IP アドレスだけを使用しなければならませんでした。これはすぐに、ホストのシンボル名を使用できるように進化しました。たとえば `TELNET 128.12.7.14` と入力する代わりに、`TELNET eduv9` と入力できるようになりました。この `eduv9` はある方法で IP アドレス 128.12.7.14 に変換されるのです。これによって、IP アドレスと高水準 (直感的に理解しやすい) マシン名の間のマッピングを、整合させて集中的に維持しなければならないという問題が出てきたのです。

ホスト名からアドレスのマッピングは、最初は Network Information Center (NIC) によって単一のファイル (HOSTS.TXT) で維持され、このファイルは FTP を使ってすべてのホストが取り出すことができました。これはフラット・ネーム・スペースと呼ばれます。

ホストの数が爆発的に増加したためこのメカニズムはあまりに扱いにくいものになり (わずか 1 つのホストをインターネットに対して追加するために必要な作業を考えれば分かります)、*Domain Name System* という新しい概念に置き換わりました。ホストは Domain Name System の代わりにあるいはそれに加えて、ローカルなフラット・ネーム・スペース (HOSTS.LOCAL ファイル) を使い続けることができますが、小規模なネットワークを除いて Domain Name System は実質的に必須のものです。Domain Name System では、ホスト上で実行するプログラムは、高水準シンボル名から他のホストの

IP アドレスへのマッピングを行うことができます。この場合、すべてのホストがホスト名の詳細なデータベースを持つ必要はありません。

8.1.1 階層ネーム・スペース

大規模な組織の社内の構成を考えてみます。代表取締役会長がすべてのことをできるわけではないため、おそらく組織は部門に分割されます。各部門は一定の限度内で自治権を持っています。具体的には、部門を預かる取締役は代表取締役の許可を得なくても直接決定できる権限を持っています。

ドメイン・ネームはこれと似たやり方で形成されており、多くの場合それを割り当てる際の権限の階層委任を反映しています。たとえば次の名前を考えてみます。

`small.itso.raleigh.ibm.com`

ここで `itso.raleigh.ibm.com` は最も下位レベルのドメイン・ネームで `raleigh.ibm.com` のサブドメイン、さらに `raleigh.ibm.com` は `ibm.com` のサブドメイン、そして `ibm.com` は `com` のサブドメインです。このネーミングの概念は階層ツリーで表すこともできます (図 131 を参照)。

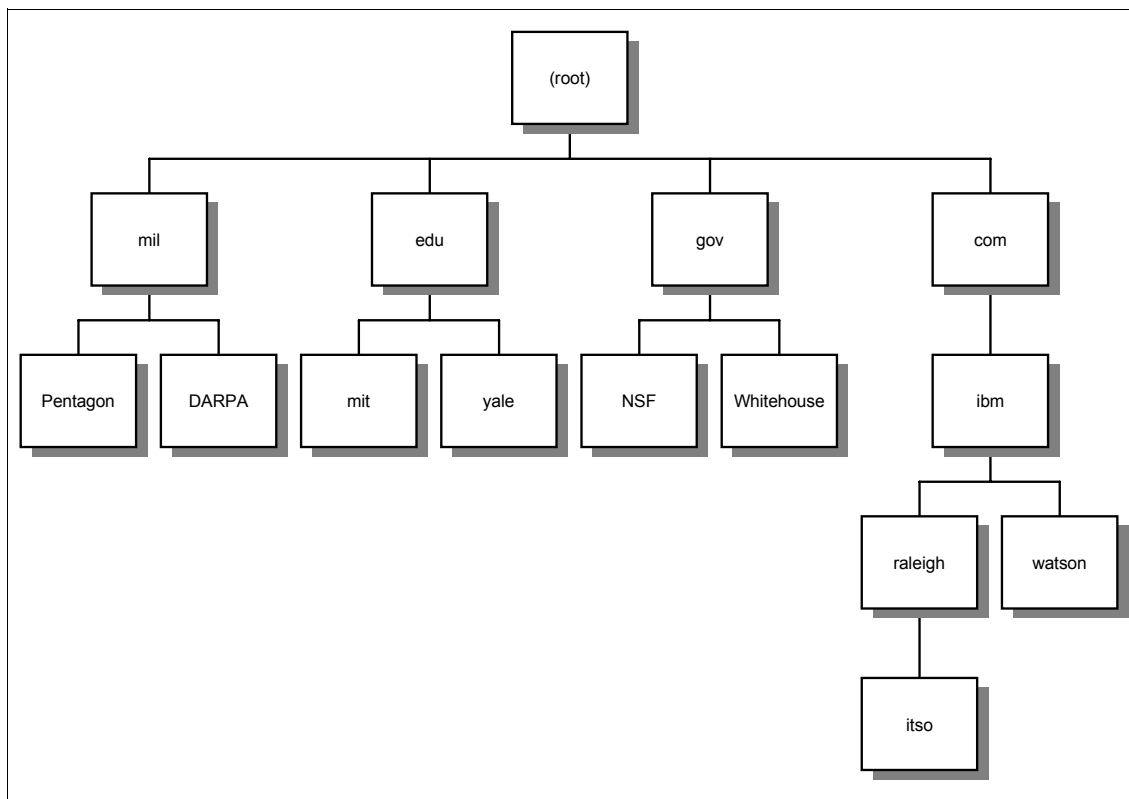


図 131 DNS - 階層ネーム・スペース

詳細な構成については、続くいくつかのセクションで説明します。

8.1.2 完全修飾ドメイン・ネーム (FQDN)

Domain Name System を使用するときは、ドメイン階層 (たとえば `ral.ibm.com` ドメイン) の一部だけを扱うのが普通です。Domain Name System では、そのときの環境で必要な入力が最小になる単純なメソッドを使用します。ドメイン・ネームがドットで終わっている場合 (たとえば `wtscpok.itsc.pok.ibm.com.`)、そのドメイン・ネームは完全と見なされます。これを**完全修飾ドメイン・ネーム (FQDN)** または**絶対ドメイン・ネーム** と呼びます。一方、ドメイン・ネームがドットで終わっていない場合 (たとえば `wtscpok.itsc`)、これは不完全であり DNS リゾルバー (下記を参照) がこれを完成することになります。たとえばドメイン・ネームに `.pok.ibm.com` などの接尾部を付加します。これを行うための規則はインプリメンテーションによって異なり、ローカルで設定することができます。

8.1.3 一般ドメイン

トップレベルの 3 文字の名前を、一般ドメインまたは組織ドメインと呼びます。表 4 は、現在のインターネット・ドメイン・ネーム・スペースのトップレベル・ドメインの一部を示したものです。

表 4 DNS - トップレベル・インターネット・ドメインの一部

ドメイン・ネーム	意味
com	商業組織
edu	学術機関
gov	政府機関
int	国際組織
mil	米国軍事機関
net	主要ネットワーク・サポート・センター
org	非営利組織
国別コード	国別ドメインのための ISO の 2 文字 ID

インターネットは米国で始まったため、階層ネーム・スペースの組織としては最初は階層の最上位に米国の組織があるだけでした。そして現在もおおむねネーム・スペースの一般ドメインの部分には米国の組織が組み込まれています。ただし .gov および .mil ドメインだけは米国向けに限定されています。

本書を執筆している時点では、米国商務省電気通信情報局は .us ドメインを別の組織に任せる意向です。この結果 Internet Assigned Numbers Authority (IANA) の位置付けを変えることが決定されています。これにより米国政府による IANA の支援と運営はなくなる予定です。代わりにドメイン・レジストリーによって、国際的な理事会を持つ新しい非営利組織が設立される予定です。その一方で、新しいトップレベル・ドメインの登録をすでに開始している他の組織もいくつかあります。現在の情報については、次の IANA Web サイトを参照してください。

<http://www.iana.org>

8.1.4 国別ドメイン

ISO 3166 国際 2 文字国別コード (アラブ首長国連邦を表す ae からジンバブエ共和国を表す zw まで) のそれぞれで指定されるトップレベル・ドメインもあります。これらは**国別**ドメインまたは**地理的**ドメインと呼ばれます。多くの国ではそれぞれ、一般トップレベル・ドメインに匹敵する独自の 2 次レベル・ドメインをその下に持っています。たとえば英国の場合、一般ドメインの .com と .edu に相当するドメインは .co.uk と .ac.uk (ac は academic の省

略語)です。トップレベル・ドメインに .us がありますが、これは州ごとの地理的な組織として使用されます(たとえば .ny.us は New York 州を指します)。ドメイン .us についての詳細な記述は RFC 1480 を参照してください。

8.1.5 ドメイン・ネームから IP アドレスへのマッピング

名前からアドレスへのマッピングには、ネーム・サーバーと呼ばれる独立した連携システムが必要です。ネーム・サーバーは名前からアドレスへのマッピングのデータベースのマスターまたはコピーを保持しているか、それらを保持していない場合は保持しているサーバーを指定するサーバー・プログラムです。そしてネーム・リゾルバーと呼ばれるクライアント・ソフトウェアからの要求に応答します。

概念的には、すべてのインターネット・ドメイン・サーバーは、298 ページの図 132 に示すネーミング階層に対応するツリー構造に配置されます。各リーフは単一のサブドメインの名前を処理するネーム・サーバーを表します。概念のツリーでのリンクは物理接続を示すものではありません。それらは特定のサーバーが接続できる他のネーム・サーバーを示しています。

8.1.6 IP アドレスからドメイン・ネームへのマッピング - ポインター照会

Domain Name System にはシンボル名から IP アドレスへのマッピング、およびその逆のマッピングの機能があります。データベースでシンボル名を使って IP アドレスを検索することは(階層構造であるため)原則的には簡単なことですが、その逆のプロセスは階層を使用できません。そのため逆マッピングのための別のネーム・スペースがあります。これはドメイン in-addr.arpa に置かれています(インターネットは最初は ARPAnet であったため arpa が使用されています)。

IP アドレスは通常はドット 10 進フォーマットで書かれ、それぞれの階層ごとにドメインが 1 層あります。しかしドメイン・ネームでは名前の最下位部分が最初にあり、ドット 10 進フォーマットでは最上位バイトが最初に来るため、ドット 10 進アドレスは逆の順序で表されます。たとえば Domain Name System で IP アドレス 129.34.139.30 に対応するドメインは、30.139.34.129.in-addr.arpa になります。したがって IP アドレスが指定されれば、Domain Name System を使用してそれに合致するホスト名を検索することができます。IP アドレスに関連してホスト名を検索するためのドメイン・ネーム照会を、*ポインター照会* と呼びます。

8.1.7 分散ネーム・スペース

Domain Name System では分散ネーム・スペースの概念を使用します。シンボル名は**権限**のゾーン(一般的にはゾーンと呼ぶ)にグループ化されます。これらの各ゾーン内で 1 つ以上のホストが、シンボル名と IP アドレスのデータベースを維持するタスク、およびシンボル名と IP アドレスの間の変換が必

要なクライアントに対してサーバー機能を提供するタスクを持っています。次にこれらのローカル・ネーム・サーバーは、(それらが接続されているインターネットネットワークを介して)ドメインの階層ツリーに論理的に相互接続されます。各ゾーンには階層ツリーの一部つまりサブツリーが含まれており、ゾーン内の名前は他のゾーン内の名前とは独立して管理されています。ゾーンの権限はネーム・サーバーに与えられています。

通常はゾーンに対する権限を持っているネーム・サーバーはそのゾーンのドメイン・ネームを持ちますが、これは必須ではありません。ドメインに含まれるサブツリーが別のゾーンに入っている場合、上位のドメインの権限を持つネーム・サーバーは、そのサブドメインの権限を持つネーム・サーバーに権限を委任します。ネーム・サーバーは自分自身に対しても権限を委任することができます。この場合ドメイン・ネーム・スペースは複数ゾーンに分割されたままドメイン・ネーム・ツリーを移動しますが、2つのゾーンに対する権限は同一のサーバーが保持しています。ドメイン・ネーム・スペースを複数ゾーンに分割するときは、Domain Name System 内に保管されているリソース・レコードを使って行われます。

これについては、トップレベルがルートになっているドメインの場合は例外です。権限を委任される上位のシステムないため、完全修飾ドメイン・ネームの照会がすべて1つだけのシステムに集中しないようにするためです。そのためトップレベル・ゾーンに対する権限は、一連のルート・ネーム・サーバーによって共有されています。¹ 例として tt.ibm.com. の照会を考えてみます。ここではネーム・サーバーのキャッシュにはまだ答えが入っていないものとします。この照会は .com ルート・ネーム・サーバーに対して行われますが、そのネーム・サーバーはこの照会を ibm.com の NS レコードを持つサーバーに転送します。必要な答えがキャッシュに入っているネーム・サーバーがこの段階で見つかる可能性があります。しかしこの照会はさらに tt.ibm.com のネーム・サーバーに委任されることがあります。

この方式の結果として次のことが挙げられます。

- データベース用の中央サーバーを持つ場合に比べて、このデータベースの維持に必要な作業の負荷は、ネーム・スペースのすべてのホストで軽減されます。
- ホストのシンボル名の作成と変更の権限、およびそれらのデータベースの維持の役割は、それらのホスト名を含む(ネーム・スペース内の)そのゾーンを所有する組織に委任されます。
- ユーザーから見れば、単一のデータベースがこれらのアドレス解決を処理しているように見えます。データベースが分散していることをユーザーが知っている場合でも、通常はユーザーはそのデータベースに関心を向ける必要はありません。

¹ 本書執筆の時点ではルート・サーバーは13ありました。現在のリストは匿名FTPのftp.rs.internic.netのファイルnetinfo/root-servers.txtに記述されています。

注：

ネーム・スペース内のドメインが IP アドレッシング方式でのネットワークとサブネットにしばしば論理的方法で対応付けられますが、そのことは Domain Name System の要件ではありません。たとえば 2 つのサブネットの間にあるルーターを考えてみてください。このルーターには IP アドレスが 2 つありますが (1 つは各ネットワーク・アダプターのためのもの)、通常はこのルーターにシンボル名が 2 つあるわけではありません。

8.1.8 ドメイン・ネーム解決

ドメイン・ネーム解決のプロセスは次のステップに要約することができます。

1. ユーザー・プログラムが `gethostbyname()` システム呼び出しなどの要求を発行します。(この特定の呼び出しはホスト名を渡してホストの IP アドレスを求めるときに使用されます。)
2. リゾルバーがネーム・サーバーに対して照会を示します。(フル・リゾルバーには最初に調べるためのローカルのネーム・キャッシュがありますが、スタブ・リゾルバーにはありません。)
3. ネーム・サーバーが、自分のローカルの信頼できるデータベースまたはキャッシュ内に答えがあるかどうかを調べます。答えがあった場合はそれをクライアントに返します。そうでない場合は、そのネーム・サーバーは他の有効なネーム・サーバーに照会します。この場合 DNS ツリーのルートまたはできるだけツリーの上位から開始します。
4. 最終的には、対応する IP アドレス (または照会によってはホスト名)、またはその照会に応答できなかった場合は、エラーがユーザー・プログラムに示されます。照会を処理するために確認されたすべてのネーム・サーバーのリストは、通常はユーザー・プログラムには示されません。

照会 / 応答メッセージは UDP または TCP によって移送されます。

ドメイン・ネーム解決はクライアント / サーバー・プロセスです。クライアント機能 (リゾルバー または ネーム・リゾルバー と呼ばれる) はユーザーにとって透過的であり、アプリケーションによって呼び出されて高水準シンボル名を実際の IP アドレスにまたはその逆に解決します。ネーム・サーバー (ドメイン・ネーム・サーバー と呼ばれる) は、高水準マシン名と IP アドレスの間の変換を行うサーバー・アプリケーションです。

8.1.8.1 ドメイン・ネームのフル・リゾルバー

図 132 はフル・リゾルバーと呼ばれるプログラムを示したものです。これはユーザー・プログラムとは別のもので、すべての照会を処理のためにネーム・サーバーに転送します。応答は以後の利用のためにネーム・サーバーによってキャッシュに入れられます。

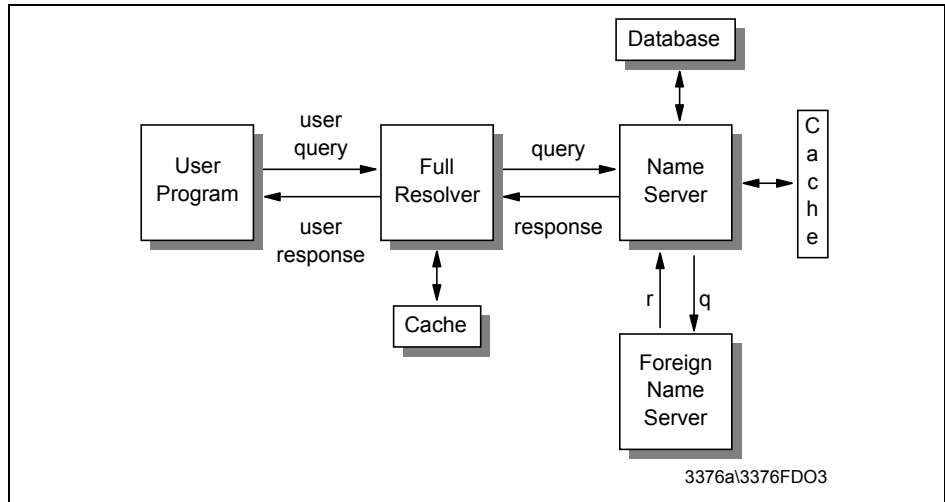


図 132 DNS - ドメイン・ネーム解決のためのフル・リゾルバーの使用

8.1.8.2 ドメイン・ネームのスタブ・リゾルバー

図 133 はスタブ・リゾルバーを示したものです。これはユーザー・プログラムとリンクしたルーチンで、照会を処理のためにネーム・サーバーに転送します。応答は通常はリゾルバーではなくネーム・サーバーによってキャッシュに入れられますが、これはインプリメンテーションによって異なります。UNIX の場合、ホスト名を IP アドレスにまたはその逆に変換するためのスタブ・リゾルバーは、`gethostbyname()` と `gethostbyaddr()` の 2 つのライブラリー・ルーチンにインプリメントされています。他のプラットフォームでも同じまたは同等のルーチンがあります。フル・リゾルバーよりもスタブ・リゾルバーの方が大幅に一般的です。

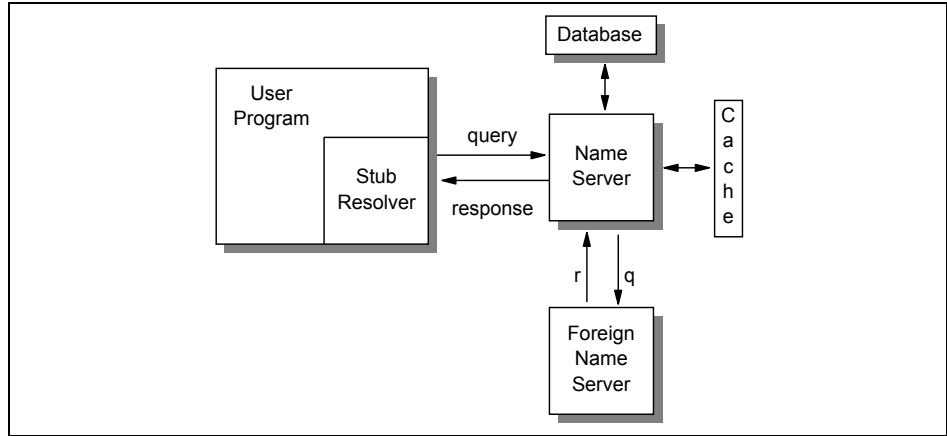


図 133 DNS - ドメイン・ネーム解決のためのスタブ・リゾルバーの使用

8.1.8.3 ドメイン・ネーム・リゾルバーの操作

ドメイン・ネーム照会は、**再帰的**または**反復**(**非再帰的**とも呼ばれる)の2つのタイプのいずれかになります。ドメイン・ネーム照会の中のフラグ・ビットでクライアントが再帰的照会を必要としているかどうか指定され、応答の中のフラグ・ビットでサーバーが再帰的照会をサポートしているかどうか指定されます。再帰的照会と反復照会の違いは、サーバーがそれ自体では完全な答えを提供できない要求を受け取ったときに現れます。再帰的照会では、サーバーが照会自体を発行して要求されている情報を決定し、完全な答えをクライアントに戻さなければなりません。反復照会ではネーム・サーバーは自分が持っている使用可能な情報と、クライアントが照会を完了させるために確認する必要がある追加のサーバーのリストも戻さなければなりません。

ドメイン・ネーム応答は、**信頼できる**と**信頼できない**の2つのタイプのいずれかになります。応答がどちらのタイプかは、応答の中のフラグ・ビットで示されます。ネーム・サーバーは、自分が権限を持っているゾーン内のドメインについての照会を受け取ると、信頼できる応答のフラグをセットし、要求されたすべての情報を応答で戻します。ネーム・サーバーが自分が権限を持っていないドメインについての照会を受け取ったときのアクションは、照会の中にある再帰要求フラグの設定によって異なります。

- 再帰要求フラグがセットされていてサーバーが再帰的照会をサポートしている場合は、サーバーはその照会を別のネーム・サーバーへ送信します。このときのネーム・サーバーは照会の中で指定されたドメインの権限を持っているものか、またはいずれかのルート・ネーム・サーバーのどちらかになります。2番目のサーバーが信頼できる応答を戻さなかった場合

(たとえばそのサーバーが権限を別のサーバーに委任したとき)は、このプロセスが繰り返されます。

サーバー(またはフル・リゾルバー・プログラム)は応答を受け取ると、繰り返し照会のパフォーマンスを上げるためにそれをキャッシュに入れます。キャッシュ項目は、応答に含まれる 32 ビットの *存続時間 (TTL)* フィールドで発信元が指定した最大時間長の間だけ保管されます。172,800 秒(2 日間)が一般的な TTL 値です。

- 再帰要求フラグがセットされていないかサーバーが再帰的照会をサポートしていない場合、サーバーは自分のキャッシュに入っている情報、および信頼できる情報を得るために確認する必要がある追加のネーム・サーバーのリストも戻します。

8.1.8.4 ドメイン・ネーム・サーバーの操作

各ネーム・サーバーは 0 個以上のゾーンに対して *権限* を持っています。ネーム・サーバーには次に示す 3 つのタイプがあります。

プライマリー プライマリー・ネーム・サーバーはディスクからゾーンの情報をロードし、そのゾーンの権限を持ちます。

セカンダリー セカンダリー・ネーム・サーバーはゾーンに対する権限を持ちますが、そのゾーンの情報は *ゾーン転送* と呼ぶプロセスを使ってプライマリー・サーバーから取得します。セカンダリー・ネーム・サーバーは同期を維持するために定期的(通常は 3 時間ごと)にプライマリー・サーバーに照会し、プライマリー・サーバーが更新されていたときはゾーン転送を再実行します。ネーム・サーバーは複数のドメインに対してプライマリーまたはセカンダリー・ネーム・サーバーとして、つまりいくつかのドメインに対してはプライマリーとしてその他のドメインに対してはセカンダリーとして動作することができます。プライマリーまたはセカンダリー・ネーム・サーバーは、キャッシングのみのネーム・サーバーのすべての機能を実行します。

キャッシングのみ

どのゾーンに対しても権限を持っていないネーム・サーバーを、*キャッシングのみのネーム・サーバー* と呼びます。キャッシングのみのネーム・サーバーは、自分の情報のすべてを必要に応じてプライマリーまたはセカンダリー・ネーム・サーバーから取得します。最初に情報を取得できるネーム・サーバーを指定するために、このサーバーには少なくとも 1 つの NS レコードが必要です。

ドメインがルートに登録されていて別の権限のゾーンが設定されているときは、次の規則が適用されます。

- そのドメインはルート管理者に登録されていなければなりません。
- そのドメイン特定の管理者がいなければなりません。
- そのゾーンに対して権限を持つネーム・サーバーが少なくとも 2 つなければなりません。そしてそれらのサーバーは、Single Point of Failure がないようにするために、ドメインの外部と内部からアクセスできなければなりません。

権限を委任するネーム・サーバーにもこれらの規則を適用することをお勧めします。委任する側のネーム・サーバーはその権限下のネーム・サーバーの動作の責任を負っているためです。

8.1.9 Domain Name System リソース・レコード

Domain Name System の分散データベースはリソース・レコード(RR)で構成され、このリソース・レコードは種類が異なるネットワークに応じたクラスに分けることができます。ここではインターネット・クラスのレコードだけを取り上げます。リソース・レコードはドメイン・ネームとネットワーク・オブジェクトの間のマッピングを提供するものです。最も一般的なネットワーク・オブジェクトはインターネット・ホストのアドレスですが、Domain Name System は広範囲のさまざまなオブジェクトに適応できるように設計されています。

ゾーンはリソース・レコードのグループから成っており、Start of Authority (SOA) レコードから始まります。SOA レコードはゾーンのドメイン・ネームを識別するものです。そのゾーンのプライマリー・ネーム・サーバーのためのネーム・サーバー (NS) レコードがあります。また、そのゾーンのセカンダリー・ネーム・サーバーのための NS レコードがある場合もあります。NS レコードは、どのネーム・サーバーが「信頼できる」か、つまりどのネーム・サーバーがゾーンを担当しているかを識別するために使用されます。次にその他のすべてのリソース・レコードがあり、これらで名前が IP アドレスに、または別名が名前にマップされている場合があります。

リソース・レコードの一般的なフォーマットは次のようになっています。

表 5 DNS リソース・レコードの一般的なフォーマット

名前	TTL	クラス	タイプ	Rdata

各部の説明：

- 名前：定義されているドメイン・ネーム。Domain Name System でのドメイン・ネーム構成の規則は非常に概括的です。ただし、ドメイン・ネームの構文に推奨されていることは、ドメイン・ネームを誤って解釈することにより、DNS リゾルバー（つまりほとんどすべての TCP/IP アプリケーション）を使用するアプリケーションの可能性を最小限にすることです。

この推奨構文に従ったドメイン・ネームは、英数字やハイフンから成る一連のラベルで構成されることになります。各ラベルでの文字の長さは 1 ～ 63 で、英字で始まります。ラベルの各ペアは人間が判読できる形式でドット (ピリオド) で区切られますが、この形式で DNS メッセージの中で使用されるわけではありません。ドメイン・ネームは大文字小文字の区別はありません。

- 存続時間：このリソース・レコードがネーム・サーバーのキャッシュ内で有効な存続時間 (TTL) (秒単位)。これは DNS に符号なし 32 ビット値で保管されます。IP アドレスを指定するレコードの場合の一般的な値は 86400 (1 日間) です。
- クラス：プロトコル・ファミリーを識別します。唯一共通に使用される値は IN (インターネット・システム) です。
- タイプ：このリソース・レコードでのリソースのタイプを指定します。
さまざまなタイプが RFC 1034、1035、1706 に詳細に記述されています。タイプにはそれぞれ名前と値があります。一般に使用されるタイプとして次のものがあります。

表 6 リソース・レコードのタイプ

タイプ	値	意味
A	1	ホスト・アドレス。
CNAME	5	正規名。ホストの別名を指定します。
HINFO	13	ホストで使用する CPU と OS。この場合はコメント・フィールドのみです。
MX	15	ドメインのメール変換。メールボックス名をホスト名にマップします。414 ページの 11.1.2 『SMTP と Domain Name System』を参照。
NS	2	ドメインの信頼できるネーム・サーバー。
PTR	12	ドメイン・ネーム・スペースの別の部分へのポインター。
SOA	6	権限のゾーンの開始。
WKS	11	ウェルノウン・サービス。このホスト上で常にアクティブであることが必要ないいくつかのサービス (たとえば SMTP) を指定します。

- Rdata: この値は、たとえば次のようにタイプによって異なります。

A 32 ビット IP アドレス (クラスが IN の場合)

CNAME ドメイン・ネーム

MX	ドメイン・ネームに続く 16 ビットの設定値 (希望する下位値)
NS	ホスト名
PTR	ドメイン・ネーム

リソース・レコードのタイプの追加情報は、313 ページの 8.2 『Dynamic Domain Name System』を参照してください。

8.1.10 Domain Name System のメッセージ

Domain Name System プロトコルでは、すべてのメッセージに単一のフォーマットを使用します。このフォーマットを図 134 に示します。このフレームはリゾルバーによってネーム・サーバーに送信されます。照会の作成にはヘッダーと質問セクションのみを使用します。照会の応答または転送 (あるいはその両方) には同じフレームを使用しますが、さらに他のセクション (応答 / 権限 / 追加セクション) も埋められます。

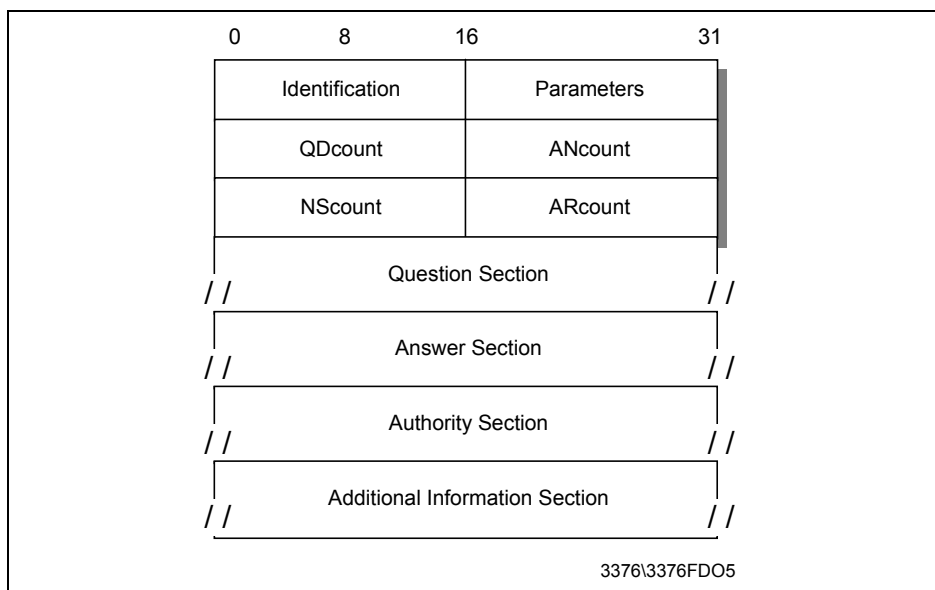


図 134 DNS メッセージ・フォーマット

8.1.10.1 ヘッダー・フォーマット

ヘッダー・セクションは常に必要であり、12 バイトの固定長です。この他のセクションは可変長です。

ID プログラムごとに割り当てられている 16 ビット ID。この ID はネーム・サーバーからの対応する応答の中にコピーされ、同時に複数の照会が未解決のときに応答を区別するために使用されます。

パラメータ次のフォーマットによる 16 ビット値です。

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Q R	Op code				AA	TC	R D	RA	Zero			Rcode			

各部の説明：

QR 照会 (0) または応答 (1) を識別するフラグ。

Op code 照会の種類を指定するための 4 ビットのフィールドで、次のように指定します。

- 0 標準照会 (QUERY)
- 1 逆照会 (IQUERY)
- 2 サーバー状況要求 (STATUS)

他の値は将来の使用のために予約済みです。

AA 信頼できる応答のフラグ。応答でこのフラグがセットされている場合は、応答したネーム・サーバーは照会で送られたドメイン・ネームに対して権限を持っていることを示しています。

TC 切り捨てフラグ。メッセージが物理チャネルで許可されている長さを超えていた場合にセットされます。

RD 再帰要求フラグ。このビットで再帰的解決を要求していることをネーム・サーバーに通知します。このビットは応答にコピーされます。

RA 再帰使用可能フラグ。ネーム・サーバーが再帰的解決をサポートしているかどうかを示します。

Zero 将来の使用のために予約済みの 3 ビット。ゼロでなければなりません。

Rcode4 ビットの応答コード。使用される値は次のとおりです。

- 0 エラーなし。
- 1 フォーマット・エラー。サーバーはメッセージを解釈できませんでした。
- 2 サーバー障害。サーバーに問題があったためメッセージは処理されませんでした。
- 3 名前エラー。照会の中のドメイン・ネームは存在しません。この値は応答の中に AA ビットがセットされている場合のみ有効です。

- 4 非インプリメント。要求された照会のタイプはネーム・サーバーにインプリメントされていません。
- 5 拒否。サーバーはポリシー上の理由で応答を拒否します。他の値は将来の使用のために予約済みです。

QDcount	符号なし 16 ビット整数で、質問セクションにある項目の数を指定します。
ANcount	符号なし 16 ビット整数で、応答セクションにある RR の数を指定します。
NScount	符号なし 16 ビット整数で、権限セクションにあるネーム・サーバー RR の数を指定します。
ARcount	符号なし 16 ビット整数で、追加レコード・セクションにある RR の数を指定します。

8.1.10.2 質問セクション

次のセクションにはネーム・サーバーへの照会が含まれます。これには QDcount (通常は 1) 項目が含まれ、各項目は図 135 に示すフォーマットになります。

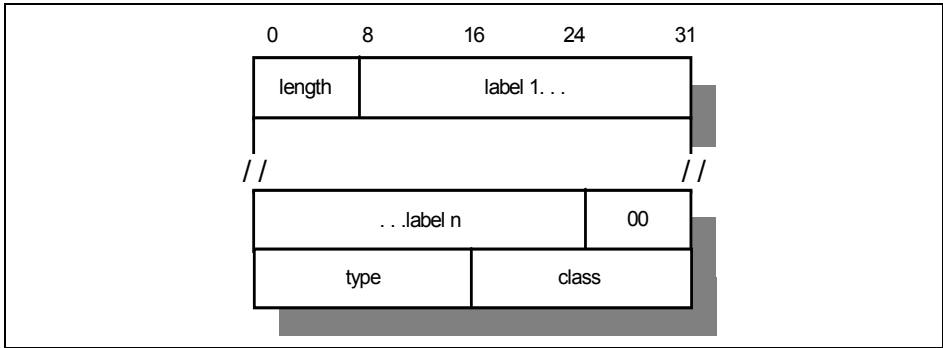


図 135 DNS - 質問フォーマット - 各フィールドはすべてバイト整合されています。「タイプ」フィールドの 4 バイト境界での位置合わせは例を目的としたものであり、このフォーマットで必須のものではありません。

各部の説明：

- 長さ** 単一バイトで次のラベルの長さを指定します。
- ラベル** ドメイン・ネーム文字の 1 要素 (たとえば ral.ibm.com の ibm)。照会の対象のドメイン・ネームは、これらの可変長ラベルが連続したものとして保管されます。このとき、それぞれのラベルの前に 1 バイト長が置かれます。

00 X'00' はドメイン・ネームの終わりを示し、ルート・ドメインのヌル・ラベルを表します。

タイプ 2 バイトで照会のタイプを指定します。リソース・レコードの「タイプ」フィールドのいずれかの値になります。

クラス 2 バイトで照会のクラスを指定します。インターネットの照会の場合はこれは IN になります。

たとえば、ドメイン・ネーム `raleigh.ibm.com` は、次の各フィールドのようにエンコードされることになります。

```
X'07'
"raleigh"
X'03'
"ibm"
X'03'
"com"
X'00'
```

したがって、`raleigh.ibm.com` のための質問セクションの項目には、21 バイト必要になります。このうち 17 バイトが、ドメイン・ネームを保管するために必要なもので、この他 `Qtype` と `Qclass` の各フィールドにそれぞれ 2 バイトが必要です。

8.1.10.3 応答、権限、および追加リソース・セクション

これらの 3 つのセクションにはリソース・レコードの変数番号が含まれます。この番号は対応するヘッダーのフィールドで指定されます。リソース・レコードのフォーマットを図 136 に示します。

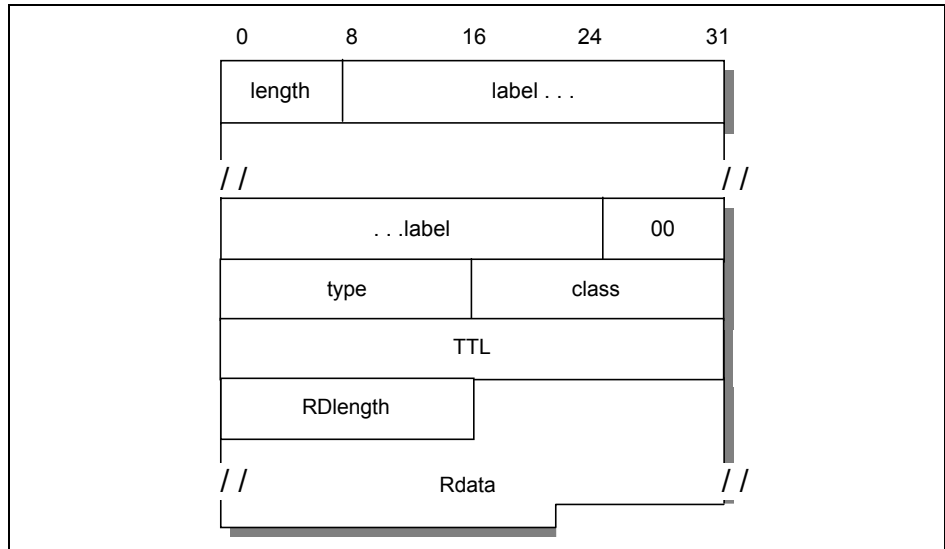


図 136 DNS - 応答レコード項目フォーマット - 各フィールドはすべてバイト整合されています。「タイプ」フィールドの 4 バイト境界での位置合わせは例を目的としたものであり、このフォーマットで必須のものではありません。

ここで TTL フィールドより前の各フィールドは質問の項目の場合と同じ意味です。それ以外のものの意味は次のとおりです。

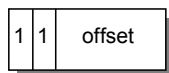
TTL レコードの存続時間を示す 32 ビットの値 (秒単位)。レコードを有効と見なせる時間の長さがこれによって定義されます。

RDlength Rdata フィールドの長さを示す 16 ビット。

Rdata 可変長ストリング。このストリングの解釈は「タイプ」フィールドによって異なります。

8.1.10.4 メッセージ圧縮

メッセージ・サイズを小さくするために、圧縮方式を使って各 RR 内のドメイン・ネームの反復を除去します。重複するドメイン・ネームやラベルのリストは、それぞれ直前の出現を指すポインターに置き換えられます。このポインターは次のような 2 バイト・フィールドの形式になっています。



- 最初の 2 ビットでポインターを通常のラベルと区別します。通常のラベルは 63 バイト長にその前の長さを示すバイトを加えたバイト長に制限されています (値 <64 バイトを持ちます)。

- ・ オフセット・フィールドでメッセージの開始からのオフセットを指定します。ゼロ・オフセットはヘッダーの中の ID フィールドの最初のバイトを指定します。
- ・ メッセージの応答、権限、または追加セクションの Rdata フィールドで圧縮が使用される場合は、その前にある RDlength フィールドには圧縮処理後の実際の長さが含まれます。

メッセージ・フォーマットの追加情報は、313 ページの 8.2 『Dynamic Domain Name System』を参照してください。

8.1.11 単純シナリオ

2 つの物理ネットワークから成るスタンドアロン・ネットワーク（外部との接続なし）を考えてみます。一方はインターネット・ネットワーク・アドレス 129.112 を持っており、もう一方はネットワーク・アドレス 194.33.7 を持ち、IP ゲートウェイ (VM2) で相互接続されています。詳細については、図 137 を参照してください。

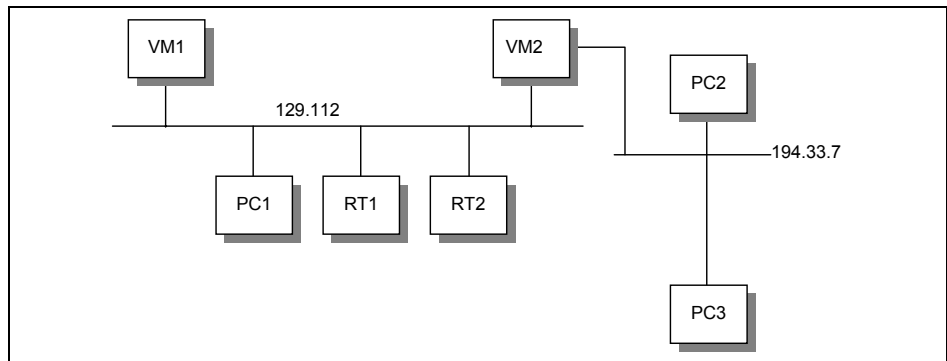


図 137 DNS - 単純構成 - IP ゲートウェイを介して接続されている 2 つのネットワーク

ネーム・サーバー機能を VM1 に割り当てます。ドメイン階層ツリーは、物理構成から完全に独立した論理ツリーを形成していることを思い出してください。この単純なシナリオではドメイン・ツリーにあるのは 1 レベルだけです。この構成にドメイン・ネーム test.example を付けます。

ネーム・サーバーのゾーン・データは図 138 に示すようになります。

```

;note: an SOA record has no TTL field
;
$origin test.example.                                     ;note 1
@           IN SOA VM1.test.example. ADM.VM1.test.example.
              (870611          ;serial number for data
              1800             ;secondary refreshes every 30 mn
              300              ;secondary reties every 5 mn
              604800           ;data expire after 1 week
              86400)           ;minimum TTL for data is 1 week
;
@           99999 IN NS  VM1.test.example.                ;note 2
;
VM1          99999 IN A   129.112.1.1                      ;note 3
            99999 IN WKS 129.112.1.1 TCP (SMTP            ;note 4
                                FTP
                                TELNET
                                NAMESRV)
;
RT1          99999 IN A   129.112.1.2
            IN HINFO IBM RT/PC-AIX                        ;note 5
RT2          99999 IN A   129.112.1.3
            IN HINFO IBM RT/PC-AIX
PC1          99999 IN A   129.112.1.11
PC2          99999 IN A   194.33.7.2
PC3          99999 IN A   194.33.7.3
;
;VM2 is an IP gateway and has 2 different IP addresses
;
VM2          99999 IN A   129.112.1.4
            99999 IN A   194.33.7.1
            99999 IN WKS 129.112.1.4 TCP (SMTP FTP)
            IN HINFO IBM-3090-VM/CMS
;
4.1.112.129.in-addr.arpa. IN PTR VM2                    ;note 6
;
;Some mailboxes
;
central 10    IN MX  VM2.test.example.                    ;note 7 and 8
;
;a second definition for the same mailbox, in case VM2 is down
;
central 20    IN MX  VM1.test.example.
waste  10     IN MX  VM2.test.example.

```

図 138 DNS - ネーム・サーバーのゾーン・データ

図 138 の注：

- 1 \$origin ステートメントで @ 変数をゾーン・ネーム (test.example.) に設定します。ピリオドで終わらないドメイン・ネームには接尾部にゾーン・ネームが付きます。完全修飾ドメイン・ネーム (ピリオドで終わるもの) はゾーン・ネームの影響を受けません。
- 2 このゾーンのネーム・サーバーを定義します。
- 3 このゾーンのネーム・サーバーのインターネット・アドレスを定義します。
- 4 このホストのウェルノウン・サービスを指定します。これらは常に使用可能である必要があります。
- 5 ホストに関する情報を指定します。
- 6 逆マッピング照会に使用します (295 ページの 8.1.6 『IP アドレスからドメイン・ネームへのマッピング - ポインター照会』を参照)。
- 7 メールの宛先が user@central.test.example になります。
- 8 これらの定義の使用については、414 ページの 11.1.2 『SMTP と Domain Name System』を参照してください。

8.1.12 拡張シナリオ

ここで第 3 のネットワーク (129.113) に接続を行う場合を考えてみます。このネットワークにはそのゾーンの権限を持つ既存のネーム・サーバーがあります (図 139 を参照)。

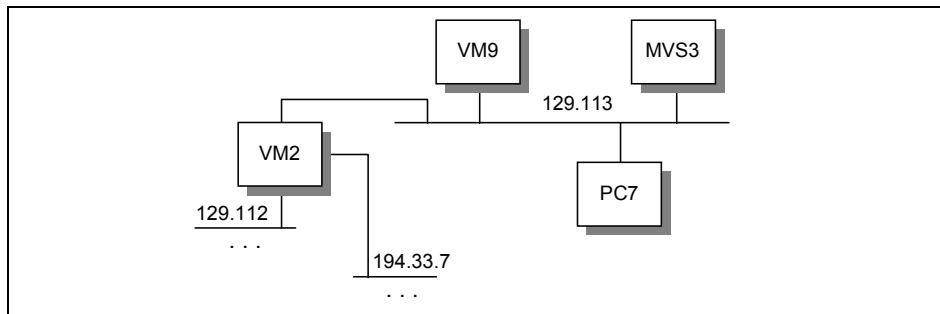


図 139 DNS - 拡張構成 - 既存の構成に第 3 のネットワークを接続する

別のネットワークのドメイン・ネームは tt.ibm.com で、そのネーム・サーバーは VM9 に置かれているものとします。ここで必要になるのは、そのネー

ム・サーバーのアドレスをこれまでのネーム・サーバーのデータベース (named.ca 初期キャッシュ・ファイル) に追加し、このネーム・サーバーで別のネットワークを参照することだけです。それを行うために必要なのは次の 2 行だけです。

```
tt.ibm.com.          99999  IN  NS      VM9.tt.ibm.com.
VM9.tt.ibm.com.      99999  IN  A       129.13.1.9
```

これは VM9 が新しいネットワークの権限を持っており、そのネットワークに対する照会はずべてそのネーム・サーバーに送信されることを示しているだけです。

8.1.13 トランスポート

Domain Name System のメッセージは、データグラム (UDP) として、またはストリーム接続 (TCP) を介して伝送されます。

- UDP の使用：サーバー・ポート 53 (10 進数)。

UDP によって送られるメッセージは 512 バイトに制限されます。これより長いメッセージは切り捨てられて、ヘッダー内の TC ビットがセットされます。UDP フレームは紛失する可能性があるため、運用では再送が必要です。

- TCP の使用：サーバー・ポート 53 (10 進数)。

この場合は、メッセージの前にメッセージの合計フレーム長を示す 2 バイトのフィールドが置かれます。

- STD 3 – Host Requirements として次のことが必要です。

- 非ゾーン転送の照会を送信する Domain Name System のリゾルバーまたはサーバーは、最初に UDP 照会を送信しなければなりません。応答での応答セクションが切り捨てられていた場合で要求側が TCP をサポートしている場合は、要求側は TCP を使って照会を再度試みなければなりません。UDP 照会の方がオーバーヘッドが大幅に少なく、負荷が大きいサーバーでは UDP の使用が必須であるため、照会の場合は TCP よりも UDP が使用されます。メッセージの切り捨ては、Domain Name System データベースの現在の内容だとすると、データグラムの中に通常は 15 応答レコードが入るため、まれにしか問題になりません。しかし Domain Name System に新しいレコード・タイプが追加されると、この状況は変わる可能性があります。
- ゾーン転送アクティビティーの場合は、UDP データグラムでの 512 バイトという制限は常に不十分なため、TCP を使用しなければなりません。
- ネーム・サーバーは両方のタイプのトランスポートをサポートしなければなりません。

8.1.13.1 動的 DNS (DDNS)

Dynamic Domain Name System (DDNS) は Domain Name System を拡張して定義したプロトコルで、要求を受信した DNS サーバーが、DNS データベースの中の項目を動的に追加、更新、および削除できるようにしたものです。DDNS の機能は既存の DNS サーバーのスーパーセットであるため、DDNS サーバーは静的ドメインと動的ドメインの両方に同時にサービスを行うことができます。これはマイグレーションと DNS 全体の設計にとって歓迎すべきフィーチャーです。

DDNS は現在、非セキュアおよびセキュア的に使用することができ、それぞれ RFC 2136 および RFC 3007 で定義されています。セキュア・バージョンの DDNS では、どのホストでも自分の DNS レコードを更新できるようにはしないで、公開鍵セキュリティとデジタル署名を使って DDNS ホストからの更新要求を認証します。たとえば、IBM はセキュア DDNS を Windows NT だけでなく、OS/2 Warp Server、AIX、OS/390、および AS/400 の各プラットフォームに完全にインプリメント済みです。

クライアント認証がない場合、無防備なホストのアドレス項目を別のホストが自分のアドレス項目に再マップして、その無防備なホストの名前を偽名として使用することが考えられます。再マップが行われると、ログオン・パスワードやそのホスト宛てのメールなどの重要なデータが、偽名を使用しているホストの方に不適切に送信されることになります。

DDNS が DHCP と連携して DNS 逆マッピング項目のシームレスな更新を行う方法についての詳細については、313 ページの 8.2 『Dynamic Domain Name System』を参照してください。また IPv6 での DNS の詳細については、617 ページの 17.4 『IPv6 における DNS』を参照してください。

8.1.14 DNS アプリケーション

ネーム・サーバーに照会するための次の 3 つの共通ユーティリティは、多くの DNS インプリメンテーションが備えています。

host ホスト名に関連付けられている IP アドレスや IP アドレスに関連付けられているホスト名を取得します。

nslookup ネットワーク・ノードについての情報の探索、ネーム・サーバー・データベースの内容の調査、およびネーム・サーバーのアクセス可能度の確立を行うことができます。

dig ネーム・サーバーを試してみることで、大量のドメイン・ネーム情報の収集、およびドメイン・ネームの単純な照会の実行が可能です。DIG は Domain Internet Groper のことです。

8.1.15 参照

Domain Name System 標準およびシステムで保持される情報は、次の RFC で定義されています。

- RFC 1032 – Domain Administrator's Guide
- RFC 1033 – Domain Administrator Operations Guide
- RFC 1034 – Domain Names – Concepts and Facilities
- RFC 1035 – Domain Names – Implementation and Specification
- RFC 1101 – DNS Encoding of Networks Names and Other Types
- RFC 1183 – New DNS RR Definitions
- RFC 1480 – The US Domain
- RFC 1591 – Domain Name System Structure and Delegation
- RFC 1706 – DNS NSAP Resource Records
- RFC 3007 – Secure Domain Name System (DNS) Dynamic Update
- RFC 1995 – Incremental Zone Transfer in DNS
- RFC 1996 – A Mechanism for Prompt Notification of Zone Transfer
- RFC 2535 – Domain Name System Security Extensions
- RFC 2136 – Dynamic Updates in the Domain Name System
- RFC 3007 – Secure Domain Name System (DNS) Dynamic Updates

8.2 Dynamic Domain Name System

291 ページの 8.1 『Domain Name System (DNS)』で述べた Domain Name System は、セキュリティの観点からは推奨しない静的インプリメンテーションです。DHCP を利用してもホスト名などの分かりやすいラベルを使って特定のホストを探し出せることができるようにするには、DNS に対して次の拡張が必要です。

- クライアントが DHCP サーバーからアドレスを取得したら、ドメイン・ネーム・サーバー内にある、ホスト名からアドレスへのクライアント用のマッピング項目を更新するメソッド。
- クライアントが自分のアドレスを取得したら、アドレスからホスト名への逆マッピングが行われるメソッド。
- DNS に対する更新が、管理者が介入する必要なしに即時に有効になること。
- 許可されていないホストがネットワークにアクセスできないようにするために、そして無防備なホストのアドレス項目を偽者が既存のホスト名を使って自分のアドレス項目に再マップできないように、DNS に対する更新が認証されること。
- 項目はクライアントによって動的に更新されるため、プライマリーおよびセカンダリー DNS サーバーが変更内容を迅速に転送して受信するメソッド。

簡単に言えば、セキュアな Dynamic Domain Name System (DDNS) が必要です。

DDNS にはいくつかの RFC が関係しています。次のものがあります。

- RFC 2535 – Domain Name System Security Extensions

この RFC に記述されている拡張機能は、ディジタル署名と認証済み公開鍵を DNS のセキュアなゾーンに保管するものです。KEY と SIG (署名) のリソース・レコード・タイプが採り入れられています。次の 3 つの異なるサービスがあります。

- 鍵の配布
- データ・オリジン認証 (ディジタル署名を DNS 内のリソース・レコードと関連付けて行う)
- トランザクションおよび要求の認証

この RFC では任意のタイプの鍵およびセキュリティーのアルゴリズムを使用できますが、デフォルトとして MD5/RSA アルゴリズムを想定しています (686 ページの 21.2 『暗号化についての簡単な概要』を参照)。

- RFC 2136 – Dynamic Updates in the Domain Name System

この RFC では UPDATE DNS メッセージ・フォーマットを採り入れ、これを DNS 内のリソース・レコードの追加や削除に使用しています。DNS が再ロードされなくても更新が有効になる機能についても記述されています。

- RFC 1995 – Incremental Zone Transfer in DNS

この RFC では IXFR DNS メッセージ・タイプを採り入れ、これを使ってプライマリーおよびセカンダリー DNS サーバー間で DNS ゾーン・データの増分転送ができるようにしています。つまりゾーン・データ対して更新が行われたときは、ゾーン・データのコピーを保持している他の DNS サーバーに対して、(AXFR DNS メッセージ・タイプの場合のように) DNS データベース全体をコピーするのではなく、変更内容だけがコピーされます。

- RFC 1996 – Prompt Notification of Zone Transfer

この RFC では NOTIFY DNS メッセージ・タイプを採用し、マスター・サーバーがこれを使って、スレーブ・サーバーに対して、更新が行われたことおよびスレーブ・サーバーは新しいデータを発見するために照会を開始する必要があることを通知します。

- RFC 3007 – Secure DNS Dynamic Updates

この RFC では RFC 2535 と RFC 2136 の両方の良いところを拡張したもので、DNS のセキュアで動的な更新について詳細に記述しています。

上記の RFC はすべて提案標準のプロトコルであり、状態は「任意選択 (Elective)」です。

8.2.1 UPDATE DNS メッセージ・フォーマット

DNS メッセージ・フォーマット (303 ページの図 134 に図示) は、静的 DNS データベースの照会のために設計されたものです。RFC 2136 では更新のために変更した DNS メッセージを定義しています。これを図 140 に示します。

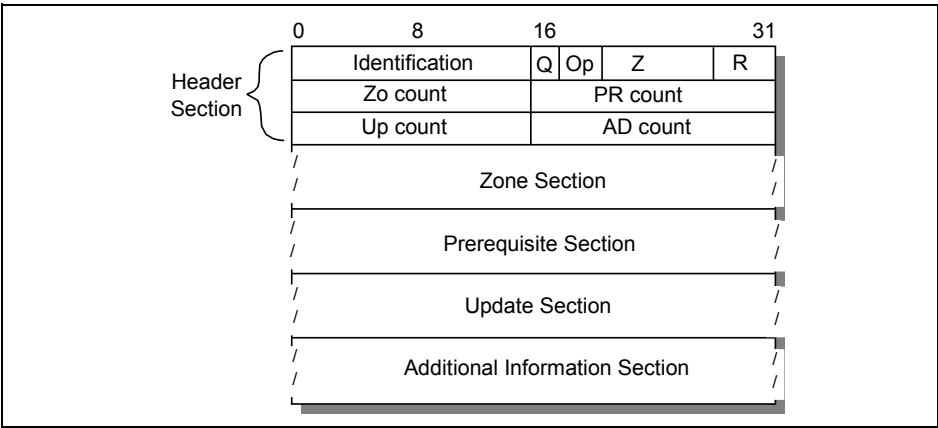


図 140 DDNS UPDATE メッセージ・フォーマット

ヘッダー・セクションは常に必要であり、12 バイトの固定長です。その他のセクションは可変長です。次のもので構成されています。

識別

プログラムごとに割り当てられている 16 ビット ID。この ID はネーム・サーバーからの対応する応答の中にコピーされ、同時に複数の照会や更新が未解決のときに応答を区別するために使用されます。

Q

更新要求 (0) または応答 (1) を識別するフラグ。

Op

Opcode - 値 5 が UPDATE メッセージを表します。

z

0 に設定される 7 ビット・フィールドで、このフィールドは将来の使用のために予約済みです。

R

応答コード (更新要求の場合は未定義)。使用される値は次のとおりです。

0 エラーなし。

- 1 フォーマット・エラー。サーバーはメッセージを解釈できませんでした。
- 2 サーバー障害。サーバーに問題があったためメッセージは処理されませんでした。
- 3 名前エラー。指定された名前は存在しません。
- 4 非インプリメント。Opcode で指定されたメッセージのタイプは、このサーバーではサポートされていません。
- 5 拒否。セキュリティまたはポリシーの理由で、サーバーは要求された UPDATE の実行を拒否しています。
- 6 名前エラー。存在しないはずの名前が存在しています。
- 7 RRset エラー。存在しないはずのソース・レコード・セットが存在しています。
- 8 RRset エラー。指定されたリソース・レコード・セットは存在しません。
- 9 ゾーン権限エラー。指定されたゾーンに対してサーバーの権限がありません。
- 10 ゾーン・エラー。前提条件または更新のセクションで指定された名前は、指定されたゾーンにありません。

ZOcount

PRcount

UPcount

ADcount

ゾーン・セクション

ゾーン・セクションにある RR の数。

前提条件セクションにある RR の数。

更新セクションにある RR の数。

追加情報セクションにある RR の数。

このセクションは、更新されるレコードのゾーンを示すために使用します。更新されるレコードはすべて同じゾーンに属していなければならないためにゾーン・セクションの項目は単一になっており、これでゾーン名、ゾーン・タイプ (SOA でなければな

りません)、およびゾーン・クラスを指定します。

前提条件セクション

このセクションには、存在しなければならないあるいは存在してはならない RR または RRset が含まれますが、そのいずれになるかは更新のタイプによって決まります。

更新セクション

このセクションには、ゾーンに追加されるまたはゾーンから削除される RR または RRset (あるいはその両方) が含まれます。

追加情報セクション

このセクションを使って、更新操作に関する追加 RR をプロセスに渡すことができます。

UPDATE メッセージ・フォーマットの詳細については、RFC 2136 を参照してください。

注：

IBM での DDNS のインプリメンテーションでは別の UPDATE メッセージ・フォーマットを使用しています。326 ページの 8.2.2.6 『IBM DDNS UPDATE メッセージ・フォーマット』を参照してください。

8.2.2 IBM での DDNS のインプリメンテーション

IBM の Dynamic Domain Name System (DDNS) は、Berkeley Internet Name Domain (BIND) レベル 4.9.3 (セキュリティ機能は組み込まれていません) を Internet Software Consortium がインプリメントしたフリーウェアに基づいており、またそのスーパーセットです。BIND Version 9.1 については、324 ページの 8.2.2.4 『BIND バージョン』を参照してください。IBM のインプリメンテーションでは静的と動的の両方の DNS ドメインをサポートしています。動的ドメインでは、許可されているクライアントだけが自分自身のデータを更新することができます。クライアント認証には RSA 公開鍵デジタル署名テクノロジーが使用されています。

管理者は IBM DHCP サーバーを使って IP アドレスなどのホスト構成パラメーターを (そのサーバーで) 構成して、IP ホストの構成を自動化します。DDNS には、動的 IP クライアントのための、ホスト名から IP アドレス (および IP アドレスからホスト名への) への動的マッピングの機能があります。DDNS を使用する場合、クライアントは DHCP サーバーから自分の新しいアドレスを受信すると、DNS サーバー上にある自分の A レコードを新しいアド

レスを使って自動的に更新します。IBM DHCP サーバーではクライアントに代わって PTR レコードを自動的に更新します。

8.2.2.1 DDNS のメカニズム

DDNS クライアント・プログラムである NSUPDATE を使って、DDNS サーバーにある情報の更新が行われます。動的更新は次のいずれかによって行われます。

ネットワーク・クライアント	DHCP および DDNS クライアント・ソフトウェアがあるホストです。自分の A および TXT レコードを現行 IP アドレス情報を使って更新することができます。
DHCP サーバー	割り振られたばかりのアドレスに応じて、現行ホスト名情報を使って PTR レコードを更新します。特定の環境では、A レコードを、それ自体では更新できないか、または更新しないクライアントに代わって更新します (327 ページの 8.2.3 『プロキシ A レコード更新 (ProxyArec)』を参照)。
DDNS システム管理者	システム管理者は DDNS データベースにあるすべてのタイプのリソース・レコードを手動で変更することができます。この変更はコマンド行から NSUPDATE コマンドを使って行うことができます。すべての変更が即時に有効になります。

図 141 は、DDNS クライアント、DDNS サーバー、および DHCP サーバーの間の相互作用を示したものです。

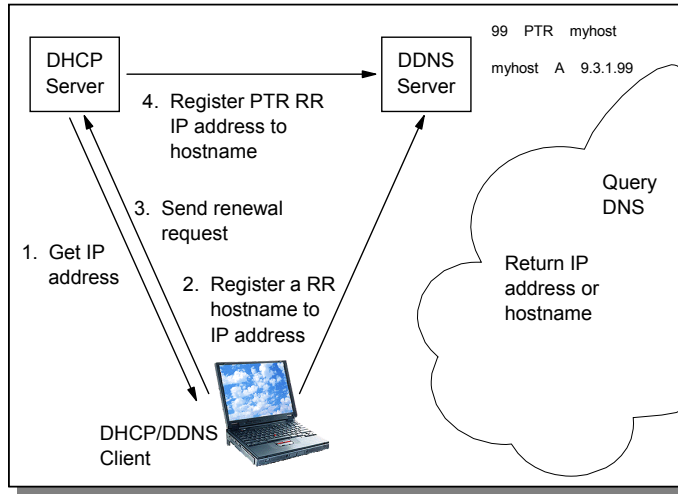


図 141 DDNS と DHCP クライアントの相互作用

注：

図 141 では説明を分かりやすくするために、DHCP サーバーと DDNS サーバーは別々の装置として示しています。実際はこれらの 2 つのサーバーは同じマシン上にあるのが一般的です。

各部の説明：

1. DHCP によって DHCP サーバーで通常の交換が行われて構成データが取得されます (135 ページの 3.7.3 『新規ネットワーク・アドレスの割り振り』を参照)。
2. 次に DDNS クライアントは、自分のホスト名に関連したリソース・レコードの更新要求を、自分が取得した新しい IP アドレスを組み込んで DDNS サーバーに送信します。このクライアントは自分の公開鍵の送信、およびすべてのリソース・レコードへのデジタル署名も行います。そのクライアントがそのホスト名を使用する権利を本当に与えられていること、およびそのクライアントのレコードに含まれる情報が有効であることの検証が、その鍵と署名の組み合わせによって行われます。

クライアントの更新が正常に終了すると、サーバーはそのデータベースに対する変更をコミットします。これでこのクライアントのホスト名を宛先にできるようになりました。

3. DHCP クライアントが、リース更新要求メッセージを DHCP サーバーに送信します。このメッセージにはクライアントが使用しているホスト名が含まれています。DHCP サーバーは DDNS サーバーのクライアントの

PTR レコード (IP アドレスからホスト名へのマッピング) を更新します。これで、クライアントの IP アドレスでの DDNS サーバーへの逆照会で、そのクライアントのホスト名が戻されるようになりました。

8.2.2.2 DDNS のセキュリティ

ネットワークで DHCP と DDNS サーバーを使用すると、クライアントがネットワークで機能する有効な構成を自動的に取得することができ、そしてその構成が存在していることを DDNS サーバーに自動的に登録することができます。しかし DDNS クライアントの認証を行わないと、許可されていないホストが無防備なクライアントのホスト名をコピーして、そのホスト名を偽名として使用する可能性があることとなります。その場合この許可されていないホストは自分自身の IP アドレスを使って DDNS サーバーを更新して、その無防備なホスト宛てのデータ (たとえばログオン・パスワード) をインターセプトできることとなります。

IBM の DDNS サーバーとそのクライアントのコードには、DNS レコードのオーナーを認証して DNS データベースの更新を保護するために、RSA デジタル署名テクノロジーがインプリメントされています。RSA デジタル署名テクノロジーの詳細については、686 ページの 21.2 『暗号化についての簡単な概要』を参照してください。DDNS では動的ドメインでの更新の保護に、動的セキュア・モードと動的事前セキュア・モードの 2 つのモードをサポートしています。両方のモードとも、データベースにある A レコードを無許可の更新から保護します。

動的セキュア・モード

DDNS クライアントは、自分のホスト名を初めて登録するときは、RSA 鍵ペアを生成します。公開鍵が DDNS サーバーに送られ、その特定のホスト名用のものとして登録されます。

DDNS クライアントは RSA 鍵ペアを (暗号化された秘密鍵とともに) クライアント鍵ファイルに保管します。DDNS クライアントが DDNS サーバー上のリソース・レコードを更新するときは、秘密鍵を使ってデジタル署名が生成され、更新データとともにサーバーに送信されます。DDNS サーバーは更新要求を受信すると、(KEY リソース・レコードにある) クライアントの公開鍵を使って次のことを行います。

- 更新要求のオーナーの認証を行い、対応する秘密鍵を使って更新要求が署名されていることを検証します。
- データが署名されてから変更されていないことを検証します (データがネーム・サーバーに届くまでに、インターセプトされて変更されていないことをそれで検証します)。

このようにして、正しいデジタル署名を生成するために必要な正しい秘密鍵を所有しているオリジナル・レコードのオーナーだけが、既存の KEY リ

ソース・レコードによって保護されているリソース・レコードを更新することができます。

さらに DDNS 管理者は、それぞれの動的ドメインごとにゾーン鍵を作成して使用することができます。ゾーン鍵は管理者の特定ドメイン用 RSA 鍵ペアになるものです。管理者はこの鍵を使用して、ドメイン内の任意のホストのレコードを、そのレコードがどこで作成されたかにかかわらず、作成、変更、または削除することができます。秘密鍵は管理者が更新要求を実行するとき署名を生成する際に使用され、サーバーはその署名を調べて更新要求が管理者からのものであることを検査します。公開鍵はドメインのデータ・ファイルにドメインの KEY リソース・レコードとして登録されます。秘密鍵と公開鍵は管理者の鍵ファイルとしてサーバー上に保管されます。

動的セキュア・モードを使用するとドメイン内でのホストの偽名の使用は起こらなくなりますが、ドメインで機能する有効な構成を新しいホスト名の DDNS クライアントが取得することは防ぐことはできません。動的セキュア・ドメインでは管理者が行う部分の構成を最小限にする必要があります。

動的事前セキュア・モード

事前セキュア動的ドメインでは、DDNS クライアントは自分の名前のレコードを作成する前に、DDNS 管理者によって事前に許可を得ていなければなりません。DDNS 管理者はホストを事前に登録して、クライアントごとに RSA 鍵を生成しなければなりません。つまり事前セキュア・モードでは、クライアントからの更新を受け入れる前に、そのクライアント用の KEY リソース・レコードがドメインであらかじめ定義されていなければならないことになります。DDNS 管理者は各ホストがリソース・レコードを作成する前に、対応する正しい鍵情報を各ホストに配布する必要もあります。

事前セキュア・ドメインでは、最もセキュアな DNS ネーム・スペースを使用できます。不明または非認証のホストによって、A および PTR レコードが更新されるのを防ぐことに加え、無効な更新の試みを DDNS サーバーでログに記録することができます。それによって、管理者は問題のクライアントの MAC アドレスを識別して適切なアクションを起こすことができます。

事前セキュア・ドメインを確立することは、DDNS 管理者にとって明らかに作業負荷が大きくなりますが、モバイル・クライアントのサポートやアドレス・スペースの保存など、動的ドメインの他の利点ももたらされます。

8.2.2.3 DDNS リソース・レコード

DNS サーバー・データベースを構成する情報はリソース・レコード (RR) の形式で表されます。RR のフォーマットは、301 ページの 8.1.9 『Domain Name System リソース・レコード』に示しています。DDNS に採用されたセキュリティ・メカニズムに伴って、KEY と SIG という 2 つの新しいタイプのリソース・レコードが導入されています。

1. KEY リソース・レコード (タイプ 25)

このレコードは DDNS データベースにある名前の公開暗号鍵のためのものです。このレコードはゾーン、ホスト、またはユーザー用の鍵になります。KEY RR は SIG RR によって認証されます。KEY RR には暗号鍵の公開指数およびモジュラスが含まれます。

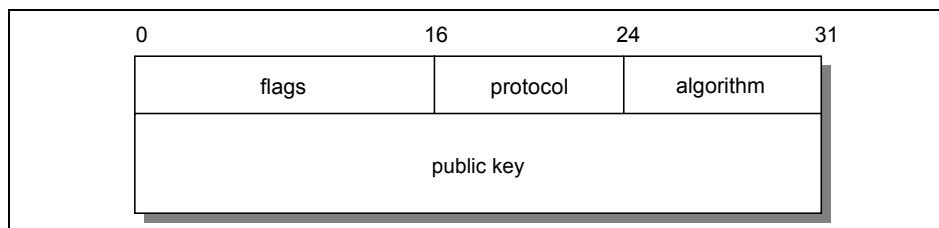


図 142 KEY リソース・レコード Rdata フォーマット

すべてのリソース・レコードのフォーマットは、303 ページの図 134 に示すフォーマットです。図 142 は KEY リソース・レコードの Rdata セクションの中で使用されるフィールドを示したものです。各フィールドの意味は次のとおりです。

- フラグ** この KEY RR が適用されるリソース・レコードのタイプをこのフィールドで示します。
- プロトコル** この KEY RR による認証で (DDNS に加えて) 保護されるプロトコルをこのフィールドで示します。
- アルゴリズム** この鍵で使用する暗号化アルゴリズムをこのフィールドで示します。IBM Dynamic IP の場合はこのフィールドの値は 1 で、RSA/MD5 アルゴリズムが使用されることになります。
- 公開鍵** 認証に使用される実際の公開鍵。このフィールドは公開指数の長さフィールド、公開鍵指数の部分、および公開鍵モジュラスの部分で構造化されています。

KEY リソース・レコードの例を次に示します。

```
client1 IN KEY 0x0000 0 1 AqO3P+UqipNXsuijeL3yyfJLw9PagI+NZg9oXrgYI1cSKOAo
+WwPOxpEqUsj0hFsKNo4V0q6LH1LK17XcytwAI01 ;Cr=auth
```

2. SIG リソース・レコード (タイプ 24)

このレコードは、DDNS データベース内のリソース・レコードを認証するデジタル署名のためのものです。SIG RR のデジタル署名そのものの中に、これによって署名されているリソース・レコードのタイプ、この署名が有効な時間、RR が署名された時刻、署名されている RR のオリジナル存続時間 (TTL) の値が含まれています。

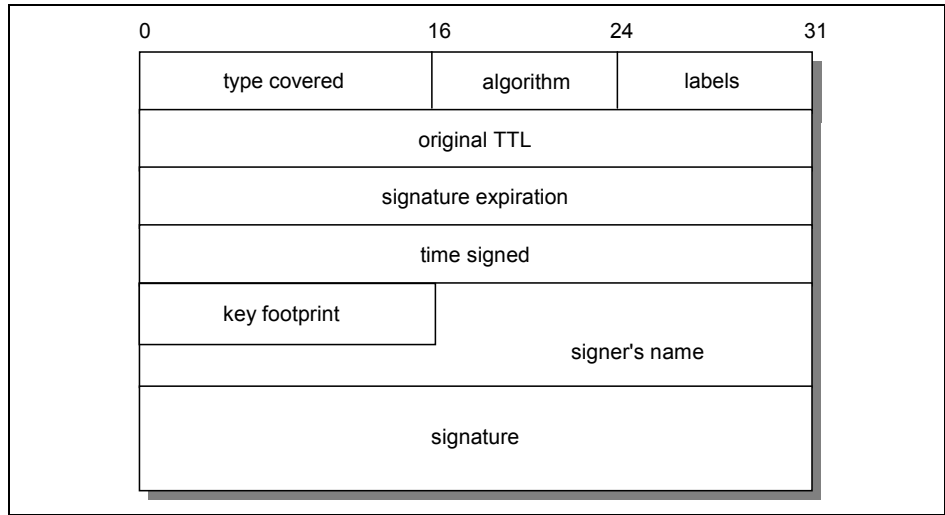


図 143 SIG リソース・レコード Rdata フォーマット

図 143 は SIG リソース・レコードの Rdata セクションの中で使用されるフィールドを示したものです。各フィールドの意味は次のとおりです。

- 対象タイプ** この署名が対象とする RR のタイプをこのフィールドで示します。
- アルゴリズム** この鍵で使用する暗号化アルゴリズムをこのフィールドで示します。IBM Dynamic IP の場合はこのフィールドの値は 1 で、RSA/MD5 アルゴリズムが使用されることになります。
- ラベル** このフィールドは SIG オーナー名のフィールドにあるラベル (ドットで区切られたホスト名およびドメイン・ネームのストリング) の数を示します。
- オリジナル TTL** リソース・レコードの存続時間値がキャッシングを行うネーム・サーバーによって減らされないように、署名されているリソース・レコードのオリジナル存続時間が組み込まれます。この値は署名によって保護され、SIG レコード自体の TTL とは異なります。
- 署名有効期限** この署名が有効な時間。この値は GMT (グリニッジ標準時) 1970 年 1 月 1 日からの秒数で表します (うるう秒は無視)。
- 署名時刻** この署名が実際に行われた時刻で、上記と同じフォーマットで表します。

鍵フットプリント	該当する暗号化アルゴリズムに応じて署名をデコードする方法がこのフィールドで決まります。
署名者名	この SIG RR を生成している署名者の完全修飾ドメイン・ネーム。
署名	対象タイプ・フィールドで示されているタイプの RR を認証する実際のデジタル署名。

SIG リソース・レコードの例を次に示します。

```
4660    IN    SIG    KEY 1 4 4660 820470267 817356268 0x8d00
                                client1.test.itsc.austin.ibm.com
                                ecK2LlzhTyVnNrI24/Viit141reduDy7TU8dxSCoGoc9zc4IIGey4E4uVPu
                                d4fjessH8XS+H2UVjLXhr66y6Gg==      ;Cr=auth
```

DDNS サーバー内のデータ・トラフィックとメモリー所要量をできるだけ小さくするために、公開暗号鍵とデジタル署名はハッシュ機能を使ってストリングに変換され、その結果それらはいわゆる base64 形式で表されます。

注：

KEY および SIG リソース・レコードは常に単一行で使います。上記の例は説明のみを目的としたものです。

8.2.2.4 BIND バージョン

IBM では、z/OS での BIND 4.9.3 および BIND 9.1 の両方の拡張バージョンを提供しています。

BIND 4.9.3 (拡張版) はシスプレックスのロード・バランシング、および IBM のバージョンの DDNS にも必要な場合があります。

BIND 9.1 は、他のベンダーの BIND 8 および BIND 9 のシステムと互換性を持たせることができます。BIND 9.1 は RFC 2136 に基づく DDNS 機能、つまり DNSSEC と TSIG をサポートしています。

BIND バージョンの説明が次のサイトにあります。

<http://www.isc.org/products/BIND>

8.2.2.5 現在の標準に関する IBM のインプリメンテーション

IBM は Dynamic DNS を 1995 年にインプリメントしました。このインプリメンテーションは、当時 IETF で検討されていたインターネット・ドラフトに基づいて IBM の研究所が行ったものです。そこでの意見は、そのプロトコルは非常に短期間のうちに標準になるだろうというものでした。IETF は有線通信のプロトコルだけを標準化するという点に注目することは重要です。鍵管理ポリシーと DHCP および DDNS の統合は IBM の設計によるものです。この

ように、統合された製品は純粋なテクノロジー (RFC の提案標準など) から分化したものです。

IETF は IBM の研究所が対象としたドラフトを標準化しませんでした。その標準は発展して現在の RFC 3007 – Secure Dynamic Updates になっています。そこでの DNS のリソース・レコードは IBM のインプリメンテーションとほとんど同じものです。しかしワイヤー上のプロトコルが異なります。また、そこでは動的更新をセキュア DNS 解決 (RFC 2065) に組み込む際の問題も扱い、モード A とモード B の 2 つのモードの操作を記述しています。モード A ではクライアントが署名した RR が DNS データの一部として保管されますが、モード B ではサーバーがすべてのレコードに署名します。IBM インプリメンテーションでの DDNS は実際はモード A に非常に近いものです。

TSIG セキュリティーは RFC 2845 に記述されています。

TSIG ではネーム・サーバーとクライアントの間の通信を保護するために、公開鍵の暗号化ではなく共有秘密鍵を使用します。これは、クライアントとサーバーが信頼関係を確立できる状況での、セキュアな解決のための RFC 2065 と動的更新のための RFC 2137 の両方をさらに単純にした代替を求めています。

IBM インプリメンテーションでの Dynamic IP が RFC とどのように違うかを、その違いの理由とともに以下に要約します。

- IBM DDNS と RFC 2137

IBM インプリメンテーションの論理展開は、RFC 2137 でのメカニズムの方向へ向かうことになるでしょう。そのために IBM が機能を追加することにはなりません、IBM の DDNS クライアントとサーバーは RFC 2137 に準拠した他のベンダーのソフトウェアと互換性を維持できることになります。このことは、他のベンダーがその機能を持つクライアントやサーバーを開発しているとすれば重要なことです。現在のところ知る限りでは RFC 2137 に準拠した DDNS クライアントやサーバーを持っている他のベンダーはありません。したがって、現時点でこの変更を行うことは IBM にとって実際の利点はないと思われます。

- IBM DDNS と RFC 2136\

RFC 2136 はリソース・レコードの認証や有効期限のために設計されたものではありません。また DNS レコードを更新するクライアントのために設計されたものでもありません。その主な設計ポイントは、管理者が DNS をリサイクルする必要なしに、特定のレコードを更新できるようにすることでした。したがって、これには IBM の統合製品にとっては必須である次の機能がありません。

- 本当の意味のセキュリティーのメカニズムがない。システム内のレコードに対する更新要求を起こせる (ゾーンごとの) IP アドレスのリストを

構成することは可能です。このメカニズムはセキュアでないだけでなく (更新機能があれば他のどのレコードでも更新できる)、スケーラブルでもありません。

- 有効期限のメカニズムがない。したがって、アドレスが割り振られていた DHCP クライアントがネットワークから離脱した場合、DHCP サーバーのその IP アドレスのリースはある時間がたてば、有効期限が切れてそのアドレスは再使用されることとなりますが、DNS の項目を手動で除去しなければならないこととなります。
- 不適切な更新要求を検出したり、リプレイ・アタックから防御するためのメカニズムがない。(更新要求へのタイム・スタンプの関連付けが行われない。)

- IBM DDNS と T-Sig

T-Sig はクライアントからの更新の場合の認証メカニズムを規定するという点ではゴールに到達しています。しかし有効期限メカニズムを規定していません。したがって、自分の A-RR を作成したクライアントがある時間の間ネットワークから離脱すると、DHCP サーバーによって作成された PTR レコードはその DHCP サーバーによって削除されますが、A-RR (ホスト名から IP アドレスへのマッピング) は DNS 内に残ることになります。これは旧ホスト名に宛てられたデータはその IP アドレスの新しい所有者に転送されることになるので、セキュリティの意味合いがあります。

T-Sig にはスケーラビリティの問題もあります。つまり、共有秘密鍵は理論的にはクライアントとネーム・サーバーの各ペア間で固有であり、それぞれ事前に定義する必要があります。

T-Sig ではタイム・スタンプが更新要求に関連付けられますが、そのタイム・スタンプは更新されたデータと一緒に保管されないため、不適切な更新要求やリプレイ・アタックを実際どのようにして検出したり防御できるのか疑問です。

8.2.2.6 IBM DDNS UPDATE メッセージ・フォーマット

IBM インプリメンテーションでの DDNS では、図 144 に示す UPDATE メッセージ・フォーマットを使用します。

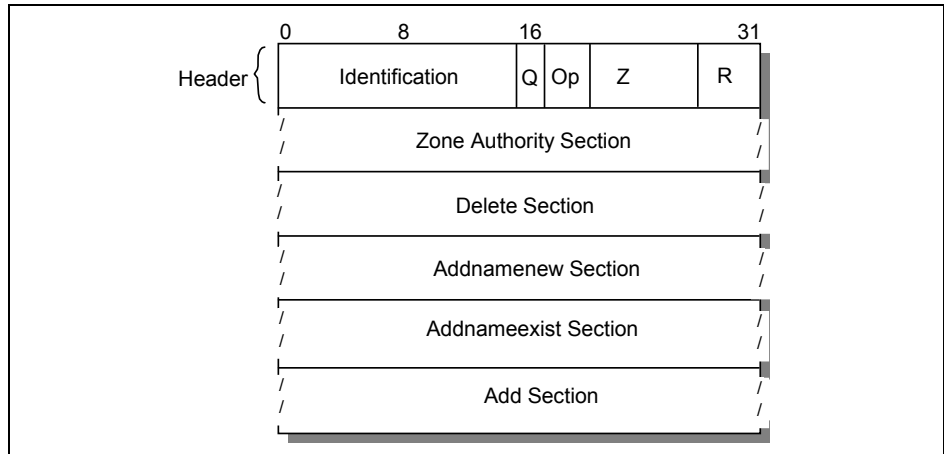


図 144 IBM DDNS UPDATE メッセージ・フォーマット

IBM のメッセージには 32 ビットのヘッダーがありますが、これは RFC 2136 でのフォーマットのヘッダーの最初の 32 ビットと同じものです (図 140 を参照)。なお、このヘッダーにはカウント (count) のセクションはありません。応答はヘッダーのみで構成されます。

UPDATE メッセージの本体には次のセクションが 1 つ以上含まれます。各セクションは、1 バイトのタイプのコード、その後に 1 バイトのカウントのフィールド (RR 単位)、その後に RR そのものという構成になっています。

ZONEAUTHORITY これには更新されるゾーンの SOA RR が含まれます。ZONEAUTHORITY セクションは 1 つだけであり、これは常にメッセージの本体の最初のセクションです。

DELETE 削除する RR を指定するときに使用します。

ADDNAMENEW 新しい名前の RR を追加するときに使用します。

ADDNAMEEXIST 既存の名前の RR を更新するときに使用します。

ADD 新規または既存の名前のレコードを追加するときに使用します。

ZONEAUTHORITY セクションを除いて、メッセージの本体の各セクションは複数あっても、どの順序になってもかまいません。

8.2.3 プロキシ A レコード更新 (ProxyArec)

DDNS 環境のクライアント・ワークステーションでは、DDNS クライアントのコードが DDNS サーバーにある自分の A レコードを更新することが必要

です。DHCP サーバーは、クライアントに代わって DDNS サーバーにある PTR レコードを更新します。

代替の方法として DHCP サーバーが、DDNS サーバーにある PTR レコードおよび A レコードの両方を更新する方法があります。このプロセスは「プロキシ A レコード更新」(ProxyArec) と呼ばれ、クライアントが DHCP をサポートしているが DDNS クライアント・コードを置いていない状態のときに使用することができます。ProxyArec はインターネット・ドラフトで DHCP と DNS 間の相互作用として検討されましたが、そのドラフトは有効期限が切れました。

ProxyArec を行うときは、DDNS サーバーは更新が行われるクライアントに対する RSA 認証検査を行うことはできません。つまりクライアントと DDNS サーバーの間の相互作用はありません。

ProxyArec は次の状態のときに使用することができます。

- ホストがモバイルではない、つまり少なくともあるドメインから別のドメインへ移動しないこと。
- モバイル・ホストに対応する必要がある場合は、ProxyArec モバイル・ホストの移動範囲にある DHCP サーバーは、すべて同じ RSA 公開 / 秘密鍵ペアを使用する構成にしなければなりません。そうしない場合デジタル署名での検査が失敗することになります。
- クライアント・ホストの DHCP プログラムは、DHCP オプションの 12 と 15 または 81 を使って、自分の完全修飾ドメイン・ネーム (FQDN) を DHCP サーバーに送信できること。
- ホスト名を盗まれてもセキュリティ上心配ないこと。

ProxyArec で使用するための新しい DHCP オプション 81 がインターネット・ドラフトで指定されています。オプション 81 は次のフォーマットです。

Code	Length	Flags	Rcode1	Rcode2	FQDN
81	n	00	00	00	

図 145 DHCP オプション 81 ProxyArec フォーマット

各部の説明：

フラグ

次の値があります。

- 0 クライアントは DDNS サーバーにある A レコードを更新する責任を保持する。

- 1 クライアントは DHCP サーバーに A レコードを更新させる。
- 3 DHCP サーバーは自分が A レコード更新の責任を持っていることをクライアントに通知する (クライアントが更新の責任は自らが持つことを望んでいても)。

デフォルト値は 1 です。

Rcode1 および Rcode2

DHCP サーバーが使用し、DHCP クライアントに DDNS 更新の応答コードを通知します。

FQDN

クライアントの完全修飾ドメイン・ネーム。

DHCP クライアントは、オプション 12 または 15 (あるいはその両方) を使って DHCP サーバーにホスト名を渡していれば、必ずしも ProxyArec を使用するためにオプション 81 をサポートする必要はありません。

8.2.3.1 DDNS と ProxyArec の併用

DDNS クライアントと DHCP だけをサポートしている、クライアントの両方がある混合環境で、DHCP サーバーが ProxyArec 用に構成されている場合に、DDNS クライアントに自分自身の (セキュアな) 更新を実行させるときは、DDNS クライアントをオプション 81 をフラグ 0 で使用するように手動で構成する必要があります。そうしないと DHCP がすべての A レコード更新を実行してしまいます。

8.2.3.2 IBM インプリメンテーションでの ProxyArec

IBM では ProxyArec を拡張し、あるレベルのセキュリティーを採り入れたものをインプリメントしました。DHCP サーバーは「ProxyArec 保護」モードになるように構成することができます。このモードにすると DHCP サーバーは、自分が最初に A レコードを作成したクライアントの MAC アドレスのレコードを維持します。以後 A レコードに対して更新の要求があると、DHCP サーバーは更新を行う前にクライアントの MAC アドレスを検査します。これによりホスト名の偽名の使用に対して一定のセキュリティーを確保できます。

8.3 Network Information System (NIS)

Network Information System (NIS) はインターネット標準ではありません。これは Sun Microsystems, Inc. によって開発されました。これは当初は Yellow Pages と呼ばれていました。

NIS は AIX または UNIX ベースの環境内のシステム情報の共用を可能にする分散データベース・システムです。たとえば共用できるシステム情報として、

/etc/passwd、/etc/group、および /etc/hosts の各ファイルがあります。NIS には次の利点があります。

- 多数のシステム間で一貫性のあるユーザー ID とグループ ID ネーム・スペースを使用できる。
- ユーザーが自分のユーザー ID、グループ ID、および NFS ファイル・システムの所有権を管理するときの時間と手間が省ける。
- システム管理者がユーザー ID、グループ ID、および NFS ファイル・システムの所有権を管理するときの時間と手間が省ける。

NIS は SUN-RPC 上に構築されます。その場合クライアント/サーバー・モデルが使用されます。NIS ドメインは次のものから成るシステムが集合したものです。

NIS マスター・サーバー

マップ(つまりデータベース)を維持し、これにはパスワードやホスト名などのシステム情報が含まれています。

NIS スレーブ・サーバー

NIS マスター・サーバーの処理の負荷を軽減するとき、または NIS マスター・サーバーが使用できないときに定義します。

NIS クライアント NIS サーバーからサービスを受けるその他のシステム。

NIS クライアントは NIS マップを維持しません。そのため NIS クライアントはシステム情報を NIS サーバーに照会します。NIS マップのどのような変更でも (RPC 経由で) NIS マスター・サーバーだけに対して行います。それを受けてマスター・サーバーは、変更内容を NIS スレーブ・サーバーに伝搬させます。

NIS マップのパフォーマンスと可用性はネットワークの速度で決まるので注意が必要です。NIS を使用する場合はそれらのゴールに合わせて、スレーブ・サーバーの数を調整する必要があります。

8.4 Lightweight Directory Access Protocol (LDAP)

DCE アーキテクチャーには独自のディレクトリー・サービスが含まれています。DCE は広くインプリメントされていますが、すべての企業にとって実際のソリューションというわけではありません。DCE は「すべてか無かの」インプリメンテーションです。つまり、企業が DCE の洗練されたセキュリティとディレクトリー・サービスから最大の効果を得るには、すべてのワークステーションに DCE クライアントをインストールし、すべてのアプリケーションおよびデータ・サーバーを DCE サーバーにし、DCE インフラス

トラクチャーとして必要なセキュリティー・サーバーとディレクトリー・サーバーもさらに設置しなければなりません。しかし、どのような種類でもネットワークを持っている企業はすべて、DCE が提供するサービスのうちの少なくともいくつかのタイプを必要としています。

現在はネットワーク・ベースのアプリケーションが多数ありますが、それらは独自のディレクトリー（データベース）を使用しており、そこにはさまざまなユーザー、アプリケーション、ファイル、およびネットワークからアクセス可能なその他のリソースについての情報が含まれています。たとえば、ファイル・サーバー用にすべてのユーザーとリソースについての情報を含むディレクトリーが 1 つあり、E メール・システム用にすべてのユーザーと部門についての情報を含むディレクトリーが、別にあるという企業があるかもしれません。それらの 2 つのディレクトリーにある情報の大半は共通のものですが、その 2 つは全体としては相互に互換性はないはずです。さまざまなネットワークとアプリケーションの数が増加したため、情報のための特化したディレクトリーの数も増えました。その結果、共有できないそして維持するにも困難な情報の島ができてしまいます。この情報をすべて一貫性のある制御されたやり方で維持してアクセスすることができれば、それは分散環境を統合したシームレスなシステムへ統合するためのフォーカル・ポイントになります。

Lightweight Directory Access Protocol (LDAP) はこれらのニーズに合うように発展してきたオープン・インダストリー・スタンダードです。LDAP ではディレクトリーにある情報にアクセスして更新するための標準メソッドを定義しています。LDAP はインターネットのディレクトリー・アクセス方式として広く受け入れられており、企業内イントラネットでも戦略上重要なものになりつつあります。これをサポートするソフトウェア・ベンダーの数は増えており、これを採り入れるアプリケーションの数も増えています。

LDAP の詳細については、レッドブック *Understanding LDAP* (SG24-4986) に書かれています。

8.4.1 LDAP - X.500 への軽快なアクセス

OSI のディレクトリー標準である X.500 では、ディレクトリー・クライアントとディレクトリー・サーバーの間の通信に Directory Access Protocol (DAP) を使うことを規定しています。しかしアプリケーション層プロトコルである DAP では OSI プロトコル・スタック全体が動作することが必要です。OSI プロトコル・スタックをサポートしようとすると、多くの小規模な環境で使用するものより多くのリソースが必要になります。そのため、X.500 ディレクトリー・サーバーへのインターフェースとして、よりリソース集約的ではない、つまり軽快なプロトコルを使用するものが求められました。

LDAP は DAP に替わる軽快なものとして開発されました。LDAP で必要なのは OSI プロトコル・スタックよりも軽量で一般的な TCP/IP プロトコル・スタックです。また LDAP では X.500 のいくつかの操作を単純化し、非公式な機能のいくつかを省略しています。LDAP の先行版として 2 つの RFC が IETF から発行されました。RFC 1202 – Directory Assistance Service と RFC 1249 – DIXIE Protocol Specification です。これらは両方とも広報 RFC であり、標準として提唱されたものではありませんでした。この Directory Assistance Service (DAS) では、ディレクトリー・クライアントが OSI に適合したホスト上のプロキシに対して通信を行えるメソッドを定義しています。このプロキシはクライアントに代わって X.500 要求を発行します。DIXIE は DAS に似ていますが DAP の変換をより直接的に行っています。最初のバージョンの LDAP は RFC 1487 – X.500 Lightweight Access で定義されましたが、これは RFC 1777 – Lightweight Directory Access Protocol に置き換えられました。LDAP では DAS と DIXIE のアイデアとプロトコルがさらに改善されています。インプリメンテーションにより中立になり、クライアントの複雑さを軽減して、ディレクトリーが可能なアプリケーションの展開をやすくしています。DIXIE と LDAP に関する作業のほとんどはミシガン大学で行われました。ミシガン大学は LDAP のリファレンス・インプリメンテーションを提供しており、また LDAP 関連の Web ページとメーリング・リストを維持しています (349 ページの 8.4.10 『参照』を参照)。

RFC 1777 は LDAP プロトコルそのものを定義しています。RFC 1777 と次の RFC の全体で LDAP バージョン 2 が定義されています。

- RFC 1778 – The String Representation of Standard Attribute Syntaxes
- RFC 1779 – A String Representation of Distinguished Names
- RFC 1959 – An LDAP URL Format
- RFC 1960 – A String Representation of LDAP Search Filters

LDAP バージョン 2 は IETF 標準化プロセスの中の "draft standard" の段階まで進んでいます。多くのベンダーが LDAP バージョン 2 をサポートする製品をインプリメント済みです。いくつかのベンダーは LDAP バージョン 3 のすべてまたは一部をサポートする製品もインプリメント中です。

LDAP バージョン 3 は RFC 2251 – Lightweight Directory Access Protocol (v3) で定義されており、これは提案標準です。LDAP バージョン 3 のための新規または更新された関連 RFC は次のとおりです。

- RFC 2252 – Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions
- RFC 2253 – Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names
- RFC 2254 – The String Representation of LDAP Search Filters

- RFC 2255 – The LDAP URL Format
- RFC 2256 – A Summary of the X.500 (96) User Schema for use with LDAPv3

LDAP ではディレクトリー・クライアントとサーバーの間の通信プロトコルを定義していますが、クライアントのためのプログラミング・インターフェースは定義していません。RFC 1823 – LDAP Application Program Interface には、LDAP バージョン 2 を使用しているディレクトリーにアクセスするための C 言語 API が定義されています。これは単なる広報 RFC ですが、デファクト・スタンダードになっています。標準化されたプロトコルおよび異なるプラットフォームでの共通 API の可用性が、LDAP が広く受け入れられている大きな理由です。本書を執筆の時点では、RFC 1823 は LDAP バージョン 3 をサポートできるように更新中の段階ですが、新しい RFC 番号はまだ割り当てられていません。進行中のこの作業は、インターネット・ドラフト "The C LDAP Application Program Interface" の対象になっています (349 ページの 8.4.10 『参照』を参照)。

8.4.2 LDAP ディレクトリー・サーバー

LDAP は通信プロトコルを定義したものです。つまりクライアントが X.500 のようなディレクトリーにあるデータにアクセスするときに使用するメッセージのトランスポートとフォーマットを定義しています。LDAP ではディレクトリー・サービスそのものは定義していません。アプリケーション・クライアント・プログラムは LDAP API を呼び出すことによって LDAP メッセージを開始します。しかし X.500 ディレクトリー・サーバーは LDAP メッセージを理解できません。実際 LDAP クライアントと X.500 サーバーは使用する通信プロトコルが異なるのです (TCP/IP と OSI)。LDAP クライアントは実際はゲートウェイ・プロセス (プロキシーまたはフロントエンドとも呼ばれる) と通信を行い、このゲートウェイ・プロセスが要求を X.500 ディレクトリー・サーバーに転送します (図 146 を参照)。このゲートウェイは LDAP サーバーと呼ばれます。これによって LDAP クライアントからの要求が達成されます。X.500 サーバーのクライアントになることによってこれを実行します。LDAP サーバーは TCP/IP と OSI の両方を使って通信を行わなければなりません。

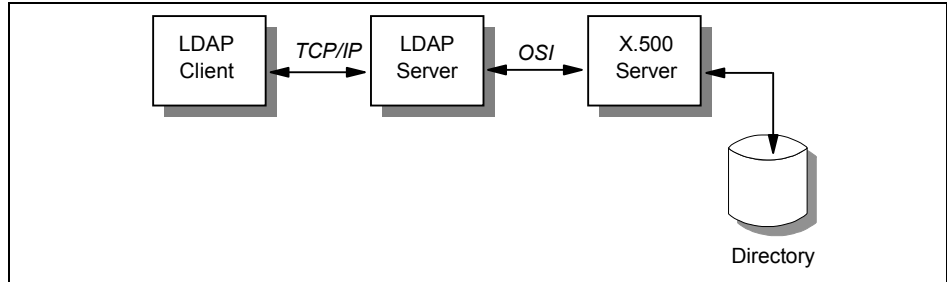


図 146 X.500 ディレクトリー・サーバーへのゲートウェイとして機能する LDAP サーバー

LDAP の使用が増えてその利点が明らかになると、X.500 サーバーやそれをサポートする環境を持っていない人たちは、LDAP クライアントでアクセスできるディレクトリーの構築を望みました。このためには LDAP サーバーが X.500 サーバーへのゲートウェイとしてだけ機能するのではなく、それ自体がディレクトリーを保管してアクセスできることが必要です (図 147 を参照)。こうすると OSI プロトコル・スタックはもう必要になりますが、当然ながら LDAP サーバーはディレクトリー項目を保管して検索しなければならないためかなり複雑になります。このような LDAP サーバーは X.500 ディレクトリー・サーバーに依存しないために、しばしばスタンドアロン LDAP サーバーと呼ばれます。LDAP は X.500 のすべての機能はサポートしていないため、スタンドアロン LDAP サーバーは LDAP で必要な機能だけをサポートすれば済みます。

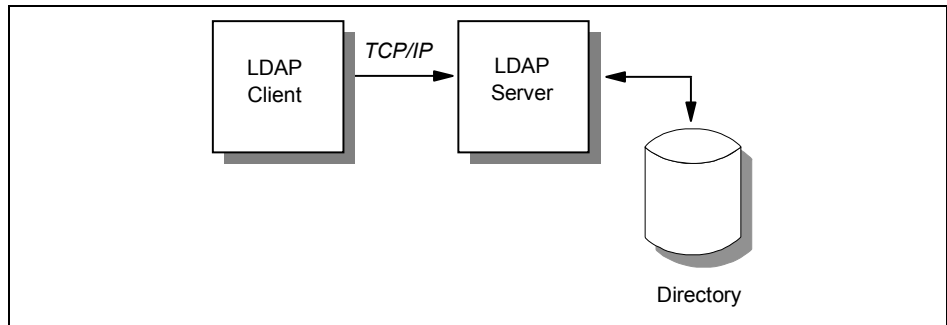


図 147 スタンドアロン LDAP サーバー

X.500 サーバーへのゲートウェイとしてだけ機能するのではなく、X.500 モデルをサポートしているローカル・ディレクトリーにアクセスできる LDAP サーバーの概念が、RFC 2251 – LDAP Version 3 で紹介されています。クライアントから見れば、LDAP プロトコルをインプリメントしているサーバーは、そのサーバーが実際にディレクトリーをインプリメントしようとして、X.500 サーバーへのゲートウェイであろうと、LDAP ディレクトリー・サー

バーです。アクセスされるディレクトリーは、そのディレクトリーがスタンダードな LDAP サーバーにインプリメントされていようと、X.500 サーバーにインプリメントされていようと、LDAP ディレクトリーと呼ぶことができます。

8.4.3 LDAP アーキテクチャーの概要

LDAP では LDAP クライアントと LDAP サーバーの間に交換されるメッセージの内容を定義しています。メッセージは、クライアントが要求する操作（検索、変更、削除など）、サーバーからの応答、およびメッセージで伝達されるデータのフォーマットを指定します。LDAP メッセージはコネクション型プロトコルである TCP/IP で送られるため、クライアントとサーバーの間のセッションの確立と切断の操作もあります。

LDAP クライアントと LDAP サーバーの間の相互作用は一般に次のように行われます。

- クライアントが LDAP サーバーとのセッションを確立します。これはサーバーへのバインディングと呼ばれます。クライアントが、ホスト名か IP アドレス、および LDAP サーバーが待機している TCP/IP ポート番号を指定します。クライアントはサーバーで正しく認証されるようにユーザー名とパスワードを指定するか、またはデフォルトのアクセス権を使って匿名セッションを確立することができます。また、クライアントとサーバーは、データの暗号化などのより強力なセキュリティ・メソッドを使うセッションを確立することもできます (343 ページの 8.4.5 『LDAP のセキュリティ』を参照)。
- 次にクライアントはディレクトリー・データに対する操作を行います。LDAP には読み取りと更新の両方の機能があります。これによってディレクトリー情報について照会だけでなく管理も行うことができます。LDAP ではユーザー指定の任意の基準に合うデータをディレクトリーから検索することもサポートしています。検索は LDAP の中で最も一般的な操作です。ユーザーはディレクトリーの中のどの部分を検索するかということと、何の情報を戻すかということを指定できます。どのディレクトリー・データを検索と突き合わせるかを、検索フィルターでブール条件を使って指定します。
- クライアントは要求をし終わると、サーバーとのセッションをクローズします。これはアンバインドとも呼ばれます。

LDAP の元々の狙いは X.500 ディレクトリーにアクセスするときの DAP に替わる軽快なものということだったため、LDAP サーバーは X.500 モデルに倣っています。ディレクトリーでは項目と呼ばれるデータ構造を保管して編成します。ディレクトリー項目には通常は、個人、プリンター、サーバーなどのオブジェクトが記述されます。各項目には識別名 (DN) と呼ばれる名前があり、各項目はこれで一意的に識別されます。DN は相対識別名 (RDN) と呼

ばれる部分が連続したものから成っており、これは UNIX や OS/2 などの多くのオペレーティング・システムでファイル名がディレクトリー名のパスから成っているのによく似ています。各項目はそれらの識別名に基づいて階層ツリーのような構造に配置することができます。ディレクトリー項目のこのツリーはディレクトリー情報ツリー (DIT) と呼ばれます。

LDAP はディレクトリー項目のアクセスおよび変更の操作を次のように定義しています。

- ユーザー指定の基準に合う項目の検索
- 項目の追加
- 項目の削除
- 項目の変更
- 項目の識別名または相対識別名の変更 (move)
- 項目の比較

8.4.4 LDAP のモデル

LDAP はそのベースになっている次の 4 つのモデルで考えると理解しやすくなります。

情報 LDAP ディレクトリーに保管されている情報の構造を解説します。

ネーミング LDAP ディレクトリーにある情報がどのように編成されて識別されるかを解説します。

機能 LDAP ディレクトリーに保管されている情報に対して行える操作を解説します。

セキュリティ LDAP ディレクトリーにある情報が無許可アクセスからどのように保護されるかを解説します。

続くいくつかのセクションでは、最初の 3 つの LDAP モデルを説明します。LDAP セキュリティは 343 ページの 8.4.5 『LDAP のセキュリティ』で取り上げています。

8.4.4.1 情報モデル

ディレクトリーに保管される情報の基本単位は項目です。これは個人、サーバー、組織など実世界でのインタレストのオブジェクトを表します。各項目にはその項目を説明する 1 つ以上の属性が含まれます。各属性はタイプと 1 つ以上の値を持っています。たとえば個人のディレクトリー項目には `telephoneNumber` という属性があることが考えられます。`telephoneNumber` 属性の構文として、電話番号は数値のストリングでなければならない、これに

はスペースとハイフンを含められるということが指定されます。この属性の値は 919-555-1212 などの個人の電話番号になります。(個人が複数の電話番号を持っている場合は、この属性は複数の値を持つことになります。)

属性の構文は、属性の値としてどういうデータを保管できるかということとを定義していることに加え、検索やその他の操作のときにそれらの値がどのように振る舞うかも定義しています。たとえば属性 `telephoneNumber` の構文では次のことが指定されます。

- 辞書編集法での順序付け。
- 比較のときは、大文字小文字、スペース、およびダッシュは無視される。
- 値は文字ストリングでなければならない。

たとえば 512-838-6008、512838-6008、および 5128386008 の各電話番号は、この定義に従えば同じものとして見なされます。LDAP で定義されている構文のいくつかを次の表に示します。

表 7 一般的な LDAP 属性の構文

構文	説明
<code>bin</code>	バイナリー情報。
<code>ces</code>	大文字小文字を区別するストリングで、ディレクトリー・ストリングとも呼ばれる。比較のときは大文字小文字の区別は有効です。
<code>tel</code>	電話番号。番号はテキストとして扱われますが、空白とダッシュはすべて無視されます。
<code>dn</code>	識別名。
<code>Generalized Time</code>	印刷可能ストリングで表される年、月、日、および時刻。
<code>Postal Address</code>	郵便宛先住所。複数の行は "\$" 文字で区切る。

表 8 は一般的な属性をいくつか示したものです。一部の属性には別名があり、完全属性名を使用できるところであればどこでも使用することができます。

表 8 一般的な LDAP 属性

属性, 別名	構文	説明	例
<code>commonName, cn</code>	<code>cis</code>	項目の共通名	John Smith
<code>surname, sn</code>	<code>cis</code>	個人の姓 (ラストネーム)	Smith
<code>telephoneNumber</code>	<code>tel</code>	電話番号	512-836-6008

オブジェクト・クラスは特定のオブジェクトの記述とは対照的に、オブジェクトの一般的な記述です (テンプレートと呼ばれることもあります)。たとえばオブジェクト・クラス person には surname 属性がありますが、John Smith を表すオブジェクトには値 Smith を持つ surname 属性があります。ディレクトリー・サーバーが保管できるオブジェクト・クラスとそのオブジェクト・クラスに含まれる属性はスキーマで記述します。スキーマでは、どのオブジェクト・クラスをディレクトリーのどこに置いてよいか、そのオブジェクト・クラスにどの属性がなければならないか、どの属性がオプションかということと、各属性の構文を定義します。たとえば、スキーマで person オブジェクト・クラスを定義します。この person のスキーマでは、person には文字ストリングの surname 属性が必須であり、person 項目はオプションで telephoneNumber 属性を持つことができ、この属性はスペースとハイフンを伴う数値のストリングであると指定するなどが考えられます。

項目が保管される前に、項目に必要な属性がすべて存在していることがスキーマ検査で確認されます。スキーマではオブジェクトの継承とサブクラス化、およびオブジェクトを DIT 構造 (階層) のどこに配置するかということも定義します。表 9 は一般的なスキーマ (オブジェクト・クラスとそこでの必須属性) の一部を示したものです。多くの場合、項目を複数のオブジェクト・クラスで構成します。

表 9 オブジェクト・クラスと必須属性

オブジェクト・クラス	説明	必須属性
InetOrgPerson	個人用の項目を定義する。	commonName (cn) surname (sn) objectClass
organizationalUnit	組織単位用の項目を定義する。	ou objectClass
organization	組織用の項目を定義する。	o objectClass

サーバーごとに独自のスキーマを定義することができますが、相互運用性のためには多くの共通スキーマが標準化されることが望まれます。特定のサーバーでまたは組織内で新しいスキーマが必要になることがあります。LDAP バージョン 3 では、サーバーがサーバー自体についての情報を、そのサーバーが使用しているスキーマも含めて戻せなければなりません。したがって、プログラムはサーバーに照会してスキーマの内容を判断することができます。

8.4.4.2 ネーミング・モデル

LDAP ネーミング・モデルでは項目がどのように識別されて編成されるかを定義します。項目はディレクトリー情報ツリー (DIT) と呼ばれるツリーのような構造に編成されます。項目はそれらの識別名 (DN) に基づいて DIT 内に配置されます。DN は単一の項目を明確に識別する固有の名前です。DN は相対識

別名 (RDN) が連続したもので構成されています。DN の中の各 RDN は DIT のルートからディレクトリー項目に達する DIT の分岐に対応しています。

各 RDN はディレクトリー項目の属性から派生したものです。単純で一般的なケースとして、RDN の形式は <属性名>=<値> になります。DN は複数の RDN がコンマで区切られて連続したもので構成されます。

DIT の例を図 148 に示します。この例は非常に単純なものです、いくつかの基本的な概念はこれを使って説明することができます。各ボックスはディレクトリー項目を表します。ルート・ディレクトリー項目は概念上のもので実際には存在しません。属性は各項目の内部に示されています。ここで示している属性は完全ではありません。たとえば UK (c=UK) という国の項目には実際には呼称記述の属性があり、その値は United Kingdom のはずです。

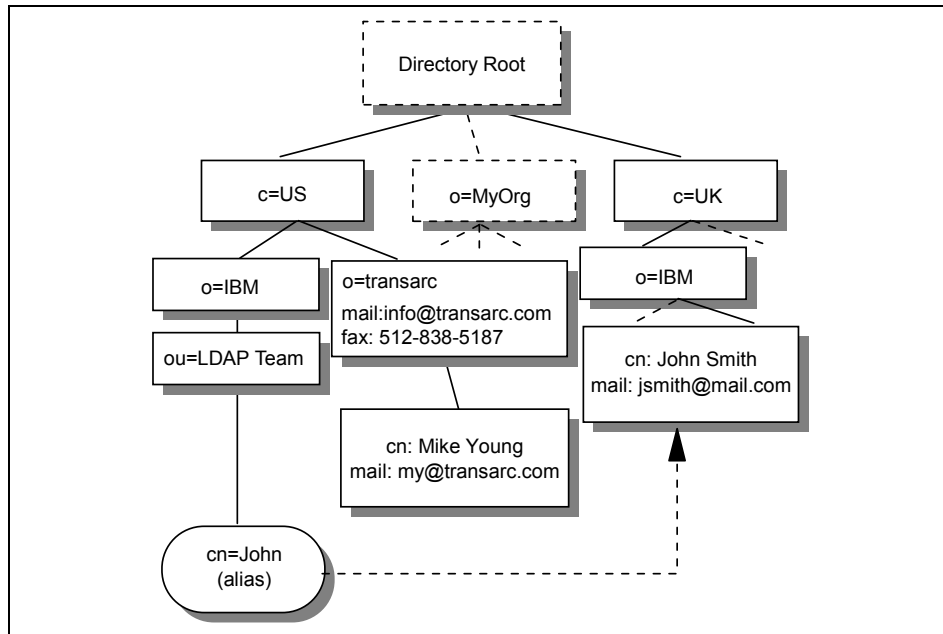


図 148 ディレクトリー情報ツリー (DIT) の例

DIT の中の項目の位置を決めるには、通常は地理または組織のスキーマに従います。たとえば、国を表す項目は DIT の最上位になります。国の下には国の組織、州や都道府県などがきます。このレベルの下の子項目は、それらの組織内の人たちやその組織をさらに細分化したものを表すものになります。DIT の最下位層の項目は、人、プリンター、アプリケーション・サーバーなどのオブジェクトを表します。DIT の縦の長さや広がりには制限されていないため、アプリケーションの要件に合わせて設計することができます。

項目は DIT でのその位置に従って名前が付けられます。図 148 の右下にあるディレクトリー項目の DN は、cn=John Smith,o=IBM,c=DE になります。

注：

ファイル・システムのディレクトリーの名前が通常はルートからリーフへと読まれる順になるのとは反対に、DN の場合はリーフからルートへと読まれる順になります。DCE ディレクトリー項目の名前もルートから始まります。

DN は RDN が連続したもので構成されています。各 RDN はその項目の属性 (複数の場合あり) から構成されます。たとえば DN cn=John Smith,o=IBM,c=DE は、RDN cn=John Smith をその上位の項目の DN o=IBM,c=DE に付加することによって組み立てられます。

DIT をツリーのようなと説明しましたが、これには、それはツリーではないという意味が暗に含まれています。それは別名があるためです。別名があるためにツリー構造にとらわれなくて済みます。このことは、項目が複数の組織に属していたり、一般に使用される DN が複雑すぎる場合に役に立ちます。項目が DIT の中で移動したが今までどおりアクセスして作業を続けたい場合にも別名が普通に使用されます。図 148 の場合、cn=John,ou=LDAP Team,o=IBM,c=US は cn=John Smith,o=IBM,c=UK の別名です。

LDAP バージョン 3 での DN は、LDAP V2 の場合よりさらに限定されています。たとえば LDAP V2 の場合は、セミコロンは RDN を区切るときにも使用できました。LDAP V3 では従来の構文は受け入れられませんが、新規構文に準拠していない DN は生成できません。

LDAP ディレクトリーは分散できるため、個々の LDAP サーバーは DIT 全体を保管しなくて済みます。サーバーでは特定の部門の項目を保管し、その部門の上位の項目は保管しなくてもかまいません。たとえば、あるサーバーが IBM の ITSO 部門の項目を保管します。この場合、そのサーバーで保管される DIT での最上位ノードは、ou=ITSO,o=IBM,c=US になります。そのサーバーでは、c=US や o=IBM,c=US の項目は保管しないことになります。サーバーによって保管される最上位項目を接尾部と呼びます。サーバーで保管される各項目の終わりは、この接尾部になります。

単一のサーバーで複数の接尾部をサポートすることができます。たとえば、ITSO 部門についての情報を保管しているサーバーが、さらに Transarc の営業部門についての情報を保管することもできます。この場合、そのサーバーにある接尾部は、ou=ITSO,o=IBM,c=US と ou=sales,o=Transarc,c=US になります。サーバーは DIT 全体を保管しなくてもかまいませんが、各サーバーは DIT 全体を含む分散ディレクトリーを形成するために、何かの方法で相互にリンクできる必要があります。これは照会を使って行われます。照会は、要

求情報が保管されている別の LDAP サーバーにある項目を指すポインターとして作用します。参照は objectClass 参照の項目です。これには属性参照があり、この値は別の LDAP サーバーにある参照される項目の LDAP URL です。詳細については、346 ページの 8.4.6 『LDAP URL』を参照してください。

参照は LDAP バージョン 3 の新しい機能です。参照によって DIT を分割して複数のサーバーに分散することができます。DIT の各部分を複製することもできます。これによってパフォーマンスと可用性を向上させることができます。

注：

アプリケーションが LDAP を使ってサーバーのディレクトリー情報を要求してもそのサーバーにその情報の参照しかないときは、その情報の LDAP URL がそのクライアントに渡されるため、さらに別のサーバーに接触して情報を取得するのはそのクライアントの責任になります。これは DCE と X.500 の両方の標準メカニズムと異なります。それらのメカニズムではディレクトリー・サーバーは、要求情報をローカルに保持していない場合は、常に別のサーバーから情報を取得してそれをクライアントに戻します。

8.4.4.3 機能モデル

LDAP はディレクトリー項目のアクセスおよび変更の操作を定義しています。LDAP 操作は次の 3 つのカテゴリに分けることができます。

照会	ディレクトリーの情報の検索に使用される検索および比較の各操作を含んでいます。
更新	ディレクトリーに保管されている情報の更新に使用される、追加、削除、変更、および RDN 変更の各操作を含んでいます。これらの操作は通常は管理者によって行われます。
認証LDAP	サーバーとの接続と切断、アクセス権の確立、および情報の保護に使用される、バインド、アンバインド、および中止の各操作を含んでいます。詳細については、343 ページの 8.4.5 『LDAP のセキュリティー』を参照してください。

検索操作

最も一般的な操作が検索です。この操作は非常に柔軟性があり、また最も複雑なオプションのいくつかを持つものでもあります。クライアントはこの検索操作を使用して、ユーザー指定の基準に一致する情報を DIT のある部分から検索するよう LDAP サーバーに要求し、その結果を読み取ってリストすることができます。

検索は非常に広範囲にわたるもの、または非常に特定化したものにできます。検索操作では、DIT 内の開始点、DIT 内の検索の深さ、一致と見なすために項目に必要な各属性、一致した項目について戻す各属性を指定することができます。

たとえば、検索のいくつかの例を簡単に示すと、次のようなものがあります。

- `cn=John Smith,o=IBM,c=DE` の郵便宛先住所を検索する。
- 項目 `ou=ITSO,o=IBM,c=US` の子になっているすべての項目を検索する。
- ある組織で、姓に "miller" という文字を含んでいて FAX 番号も持っている人の E メール・アドレスと電話番号を検索する。

検索を実行するには、次のパラメーターを指定する必要があります。

基本	検索の開始点 (基本オブジェクトと呼ばれる) を定義する DN。基本オブジェクトは DIT 内のノードです。
有効範囲	DIT 内の基本オブジェクトからの検索の深さを指定します。これには次の 3 つの選択項目があります。 baseObject 基本オブジェクトだけを検査対象にします。 singleLevel 基本オブジェクトの直接の子だけを検査対象にします。基本オブジェクト自体は検査対象になりません。 wholeSubtree 基本オブジェクトおよびその下位のすべてを検査対象にします。
検索フィルター	項目が検索結果として戻されるためには一致しなければならない基準を指定します。検索フィルターは属性値を断定してブール演算子と組み合わせたものです。属性値を断定することにより、属性の値の「等価」、「より小か等しい」などを検査します。
戻す属性	検索基準に一致した項目から検索する属性を指定します。1 つの項目に属性が多数ある場合がありますが、ユーザーはこれによって関心がある属性だけを知ることができます。一般にユーザーは属性の値に関心があります。ただし、属性の値ではなく属性タイプだけを戻すようにすることもできます。
別名参照解除	別名を参照解除するかどうかを指定します。つまり、別名項目で指される実際に関係があるオブジェクトが検査されます。項目を参照解除しない場合に、別名項目自体を検査することができます。

制限

検索では非常に広範囲にわたって大規模なサブツリーがいくつも検査され、その結果多数の項目が戻されることがあります。ユーザー (つまりサーバー) は時間とサイズの限度を指定して、不合理な検索によって過多のリソースが消費されないようにすることができます。サイズの限度で、検索から戻される項目の数を制限します。時間の限度で、検索の合計時間を制限します。

8.4.5 LDAP のセキュリティー

セキュリティーは、コンピューターのネットワーク化領域では非常に重要なことですが、これは LDAP でも同じです。非セキュアなネットワーク上でデータを送信するときは、移送の間は機密情報を内部または外部で保護することが必要になる場合があります。また、その情報を要求している人、およびそれを送信している人を知る必要もあります。このことは、その情報がディレクトリーの更新操作の場合は特に重要です。セキュリティーという用語には、次の 4 つの性質が含まれていると考えることができます。

認証	対向の通信相手 (マシンまたは個人) が実際に申請どおりの人 / 物であることの確認。
保全性	到着した情報が送信されたものと実際に同一であることの確認。
機密性	データ暗号化による、受信対象になっていない人たちへの情報の露出の保護。
許可	通信相手が要求していることが実際に許可されていることの確認。これは通常ユーザー認証後に検査されます。LDAP バージョン 3 では、現時点ではこれはプロトコル仕様の一部になっていません。そのためこれはインプリメンテーション (またはベンダー) 固有のものになります。許可は、読み取り、書き込み、または削除などのアクセス制御を、ユーザー ID または共通名に応じて、アクセスされるリソースに割り当てることで行います。LDAP でのアクセス制御を提唱しているインターネット・ドラフトがあります (349 ページの 8.4.10 『参照』を参照)。

続くいくつかのセクションでは、最初の 3 つの性質である認証、保全性、および機密性に焦点を合わせます (許可は LDAP バージョン 3 標準に含まれていないため)。この目的のために使用できるメカニズムはいくつかありますが、ここでは最も重要なものを取り上げます。次のものがあります。

- 無認証
- 基本認証

- Simple Authentication and Security Layer (SASL)

LDAP で使用されるセキュリティ・メカニズムは、クライアントとサーバー間の接続が確立されたときにネゴシエーションされます。

8.4.5.1 無認証

無認証を使用するのは、データ・セキュリティが問題にならず、特別なアクセス制御許可が必要ないときだけにします。これはたとえば、ディレクトリーが誰からでも参照できるアドレス帳である場合に該当します。無認証は、バインド API 呼び出しの際にパスワードと DN のフィールドが空のままになっていることを前提にしています。それによって LDAP サーバーは自動的に匿名ユーザー・セッションを想定し、その種のアクセスに定義されている適切なアクセス制御を認可します (344 ページの 8.4.5.3 『Simple Authentication and Security Layer (SASL)』で説明している SASL 匿名ユーザーと混同しないようにしてください)。

8.4.5.2 基本認証

基本認証は HTTP など Web 関連の他のいくつかのプロトコルでも使用されます。LDAP で基本認証を使用するときは、クライアントはサーバーに対して自分自身を DN とパスワードで示します。これらはネットワーク上を暗号化しないで送信されます (インプリメンテーションによっては base64 エンコード方式を使用しているものもあります)。サーバーは、クライアントから送信された DN とパスワードがディレクトリーに保管されているその DN のパスワードと一致すると、そのクライアントを認証できたと見なします。base64 エンコード方式は、Multipurpose Internet Mail Extensions (MIME) で定義されています (417 ページの 11.2 『Multipurpose Internet Mail Extensions (MIME)』を参照してください)。base64 は比較的単純な暗号化であるため、ネットワーク上でデータを一度取り込むと解読するのは難しいことはありません。

8.4.5.3 Simple Authentication and Security Layer (SASL)

SASL はコネクション型プロトコルに付加的な認証メカニズムを追加するためのフレームワークです。LDAP バージョン 2 の認証の欠点を解消するために、LDAP バージョン 3 ではこれが追加されました。SASL は最初は IMAP プロトコルにより強力な認証を追加するために工夫されたものでした。それ以降 SASL はプロトコルと認証システムの間に介在させるための、より汎用的なシステムに発展してきました。これは RFC 2222 – Simple Authentication and Security Layer (SASL) で定義されており、インターネット提案標準です。

SASL では LDAP や IMAP などの接続プロトコルはプロファイルによって表されます。各プロファイルはプロトコルの拡張機能と見なされ、これによりプロトコルと SASL が調和して動作できるようになっています。SASL のプロファイルの詳細なリストは、Information Sciences Institute (ISI) から入手で

きます (349 ページの 8.4.10 『参照』を参照)。その中には IMAP4、SMTP、POP3、および LDAP が含まれています。SASL を使用する各プロトコルは、認証メカニズムを識別できて認証交換を実行できるように、コマンドを使って拡張する必要があります。オプションとして、認証後にデータを暗号化して機密性を確保するようにセキュリティ層をネゴシエーションできます。LDAP バージョン 3 にはそのコマンド (`ldap_sasl_bind()`) が組み込まれています。使用されるセキュリティ・メソッドに影響を及ぼすキー・パラメーターには、次のものがあります。

dn バインドする項目の識別名です。これは通常のユーザー ID とパスワードの認証の場合のユーザー ID と考えることができます。

メカニズム 使用するセキュリティ・メソッドの名前です。有効なセキュリティ・メカニズムは、現時点では Kerberos バージョン 4、S/Key、GSSAPI、CRAM-MD5、および EXTERNAL です。ANONYMOUS メカニズムもあり、これを使用すると匿名ユーザーの認証が可能です。LDAP で最も一般的に使用されるメカニズムは SSL (またはその後継の TLS) ですが、これは EXTERNAL メカニズムとして指定できます。

クレデンシャル

これには DN を識別する任意のデータが含まれます。パラメーターのフォーマットと内容は、選択したメカニズムによって異なります。たとえば ANONYMOUS メカニズムの場合、これはユーザーを識別する任意のストリングまたは E メール・アドレスになります。

SASL には、使用する特定のセキュリティ・メカニズムを、関係する通信当事者が決定できる高水準フレームワークがあります。SASL セキュリティ・メカニズムでのクライアントとサーバーの間のネゴシエーションは、暗号化されずに行われます。クライアントとサーバーが共通のメカニズムに一度同意すると、その接続は認証識別の変更に対してセキュアなものになります。しかしアタッカーがメカニズム・ネゴシエーションの盗聴を試みる可能性があるため、通信当事者は最低限のセキュアなメカニズムを使用することになります。これが起こらないようにするためには、クライアントとサーバーをセキュリティの最小メカニズムを使用するように構成します (クライアントとサーバーがそのような構成オプションをサポートしている場合)。前述のように SSL とその後継の TLS は、LDAP の場合の SASL で一般的に使用されるメカニズムです。これらのプロトコルの詳細については、777 ページの 21.8 『Secure Sockets Layer (SSL)』を参照してください。

LDAP バージョン 2 ではデータ暗号化メソッドが指定されていなかったために、たとえば Netscape や IBM のように、独自の SSL 呼び出しを LDAP API に追加したベンダーもあります。そのような方法は、さまざまなベンダーの

インプリメンテーションの間で API 呼び出しに互換性がない可能性があるという欠点があります。LDAP バージョン 3 で指定されている SASL を使用すると、ベンダーの製品は可能な範囲のメカニズム (おそらく SSL のみ) のサブセットのみをサポートすることが考えられますが、互換性は確保できるはずです。

8.4.6 LDAP URL

LDAP はインターネットでの重要なプロトコルになったため、LDAP リソースの URL フォーマットが定義されました。RFC 2255 – The LDAP URL Format には LDAP URL のフォーマットが記述されています。LDAP サーバーが SSL を使って通信を行う場合、LDAP URL は `ldap://` または `ldaps://` で始まります。LDAP URL で単に LDAP サーバーを指定したり、複雑なディレクトリー検索を指定することができます。

例を使って LDAP URL のフォーマットを説明します。次の例はホスト `saturn.itso.austin.ibm.com` (デフォルト・ポート 389 を使用) 上の LDAP サーバーを指しています。

```
ldap://saturn.itso.austin.ibm.com/
```

次の例は、ホスト `saturn.itso.austin.ibm.com` 上の LDAP サーバーから DN `o=Transarc,c=US` のすべての属性を検索するものです。なお、例としてここでは、ポート 389 が明示的に指定されています。389 はデフォルト・ポートであるため、URL で必ずしもこれを指定する必要はありません。

```
ldap://saturn.itso.austin.ibm.com:389/o=Transarc,c=US
```

次の例は、DN `cn=JohnSmith,ou=Austin,o=IBM,c=US` のすべての属性を検索するものです。いくつかの文字は、プログラムによっては除去されたり区切り文字として扱われることがあるため、それらの文字は URL ではアンセーフと考える必要があります。スペース、コンマ、大括弧などのアンセーフ文字はそれらの 16 進値で表し、その前に % 記号を付けます。

```
ldap://saturn.itso.austin.ibm.com/cn=John%20Smith,ou=Austin,o=IBM,c=US
```

この例では `%20` はスペースです。アンセーフ文字および URL の全般の詳細については、RFC 1738 – Uniform Resource Locators (URL) に記載されています。

8.4.7 LDAP と DCE

DCE には独自のセル・ディレクトリー・サービス (CDS) があります (366 ページの 9.3.1.1 『セル・ディレクトリー・サービス』を参照)。アプリケーションが自分の DCE セルの外部のリソースにアクセスすることがない場合に必要なのは CDS のみです。しかしアプリケーションが他の DCE セルにあるリソースと通信を行う必要がある場合は、グローバル・ディレクトリー・

エージェント (GDA) が必要です。GDA は DCE セルの名前を登録できるグローバル (つまり CDS 以外の) ディレクトリーにアクセスします。このグローバル・ディレクトリー (GDS) は、Domain Name System (DNS) ディレクトリーまたは X.500 ディレクトリーです。GDA はリモート・セル内の CDS サーバーのアドレスを検索します。次にそのリモート CDS に接触してそのセル内の DCE リソースを検出します。GDA を使用すると、ある組織を複数の DCE セルとまとめてリンクすることができます。この場合イントラネット上のプライベート・ディレクトリーか、またはインターネット上のパブリック・ディレクトリーを使用します。

インターネットでの LDAP の存在が顕著なため、オープン・グループ (The Open Group) では LDAP と DCE テクノロジーの統合を調査研究するための LDAP プロジェクトを 2 つ抱えています。

8.4.7.1 GDA の LDAP インターフェース

LDAP を DCE に統合する 1 つの方法は、DCE セルを LDAP ディレクトリーに登録できるようにすることです。リモート・セルに接続する必要があるセル内 GDA は、LDAP ディレクトリーにアクセスできるように構成されます (図 149 を参照)。

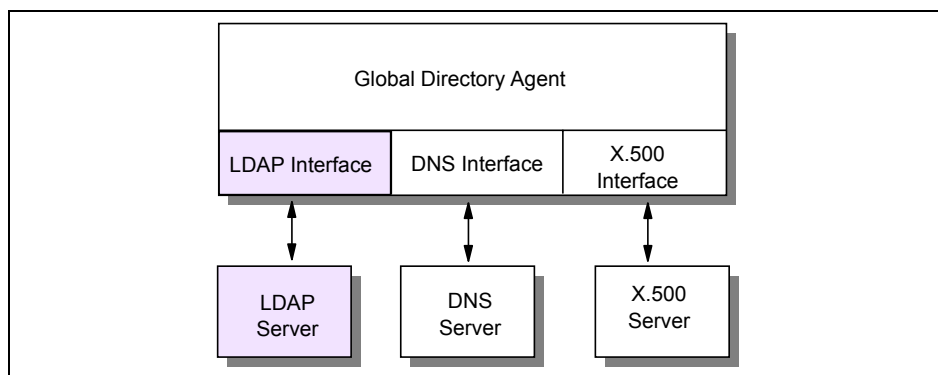


図 149 GDA の LDAP インターフェース

DCE は最初はセル名として X.500 と DNS の名前構文だけをサポートしていました。LDAP と X.500 での名前は両方とも同じ階層ネーミング・モデルに従っていますが、それらの構文は多少異なっています。X.500 の名前は逆順に書かれていて、相対識別名を区切るためにコンマ (,) ではなくスラッシュ (/) を使用します。GDA を LDAP を使用するように構成すると、GDA は X.500 フォーマットのセル名を LDAP フォーマットに変換して LDAP ディレクトリーの中を検索します。

8.4.7.2 CDS の LDAP インターフェース

DCE にはディレクトリー・サービスとのプログラミング・インターフェースとして、Name Service Interface (NSI) と X/Open Directory Service (XDS) の 2 つがあります。XDS は X.500 互換のインターフェースで、GDS での情報へのアクセスに使用されますが、CDS での情報へのアクセスにも使用されます。しかし DCE アプリケーションでは NSI の方がかなり一般的に使用されています。NSI API の機能は、RPC を使用する DCE クライアントとサーバー・プログラムでの使用に特に合わせたものです。NSI を使用するとサーバーは、自分のアドレスとサポートしている RPC インターフェースのタイプを登録することができます。このアドレス / インターフェース情報は RPC バインディングと呼ばれ、サーバーに接触する必要があるクライアントにとって必要なものです。クライアントは NSI を使用して、CDS から RPC バインディング情報を検索することができます。

NSI は、RPC バインディングが保管されるディレクトリーから独立したものになるように、設計されたものです。しかし、現在のところサポートされているディレクトリーは CDS のみです。NSI は、LDAP ディレクトリーの RPC バインディングの、追加と検索もサポートできるように拡張される予定です。これを使用するとサーバーは、自分の RPC バインディング情報を CDS または LDAP ディレクトリーに公示することができます。アプリケーション・プログラムは LDAP ディレクトリーを使用する場合は、NSI または LDAP API を使用することができます (図 150 を参照)。

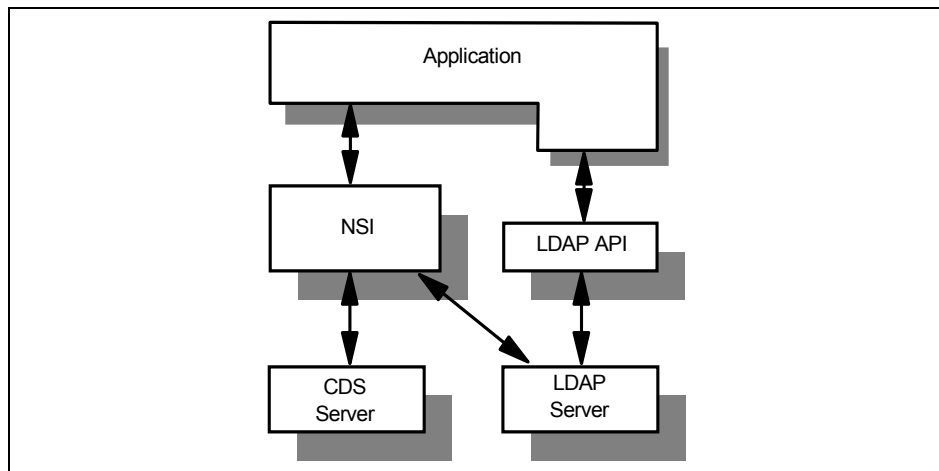


図 150 NSI の LDAP インターフェース

RPC バインディング情報のための LDAP スキーマは IETF において開発のドラフト段階です。

8.4.8 directory-enabled networks initiative (DEN)

1997 年 9 月、Cisco Systems Inc. と Microsoft Corp. は、共同作業の結果として、いわゆる directory-enabled networks initiative (DEN) を発表しました。IBM など多数の企業が、このイニシアチブをサポートしたり、特別ワーキング・グループ (ADWG) に積極的に参加しています。DEN は、人、ネットワーク、装置、アプリケーションについての情報を保管する、統合されたディレクトリーの情報モデルの仕様を表すものです。DEN スキーマではオブジェクト・クラスとそれに関係するオブジェクトの属性を定義しています。その意味で DEN は、複数のベンダーの製品がトポロジと構成関連データを保管して検索できるインテリジェント・ネットワークを構築するときに重要になります。DEN は比較的新しい仕様であるため、これをサポートする製品は最初のドラフト (1997 年末に発行) から約 1 ～ 2 年の間は現れないと思われます。

DEN 仕様で特に興味を引くのは、DEN 情報をアクセスするためのコア・プロトコルとして LDAP バージョン 3 を定義していることです。これにより LDAP が使用可能なクライアントまたはネットワーク装置 (あるいはその両方) で情報にアクセスできることになります。

DEN では Common information Model (CIM) を使用します。CIM は SNMP MIB や DMTF MIF などの異なる管理モデルを統合する方法を詳細に表現したものです。

DEN イニシアチブの詳細については、設立メンバーによる次の Web サイトで参照できます。

<http://www.dmtf.org/spec/denh.html>

<http://www.dmtf.org/spec/cims.html>

8.4.9 Web-Based Enterprise Management (WBEM)

WBEM は企業のための管理ツールの統合セットを打ち出すために策定された一連の標準です。XML と CIM を使って、ネットワーク装置、デスクトップ・システム、通信システム、アプリケーション・システムのすべての管理を、Web ブラウザーから行うことが可能になります。詳細については、以下を参照してください。

<http://www.dmtf.org/spec/wbem.html>

8.4.10 参照

LDAP の詳細については、次の資料を参照してください。

- RFC
 - RFC 1487 – X.500 Lightweight Directory Access Protocol
 - RFC 1777 – Lightweight Directory Access Protocol

RFC 1778 – The String Representation of Standard Attribute Syntaxes
RFC 1779 – A String Representation of Distinguished Names
RFC 1823 – The LDAP Application Program Interface
RFC 1959 – An LDAP URL Format
RFC 1960 – A String Representation of LDAP Search Filters
RFC 2222 – Simple Authentication and Security Layer (SASL)
RFC 2251 – Lightweight Directory Access Protocol (v3)
RFC 2252 – Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions
RFC 2253 – Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names
RFC 2254 – The String Representation of LDAP Search Filters
RFC 2255 – The LDAP URL Format
RFC 2256 – A Summary of the X.500(96) User Schema for use with LDAPv3
RFC 2820 – Access Control Requirements for LDAP

- インターネット・ドラフト

The C LDAP Application Program Interface : 次のサイトで参照できます。

<http://search.ietf.org/internet-drafts/draft-ietf-ldapext-ldap-c-api-05.txt>

Access Control Model for LDAP : 次のサイトで参照できます。

<http://search.ietf.org/internet-drafts/draft-ietf-ldapext-acl-model-07.txt>

- その他のリソース

SASL のメカニズム :

<ftp://ftp.isi.edu/in-notes/iana/assignments/sasl-mechanisms>

ミシガン大学は LDAP の開発に大きな貢献をしてきており、ミシガン大学の次のサイトは信頼できる中立の広範囲の情報源と考えられます。

<http://www.umich.edu/~dirsvcs/ldap>

LDAP のオープン・ソースのインプリメンテーションが次のサイトにあります。

<http://www.OpenLDAP.org>

第 9 章 リモート実行と分散コンピューティング

ネットワーク・コンピューターで使用する最も基本的なメカニズムの 1 つとして、リモート・システム上で実行する機能があります。つまりユーザーがリモート・マシン上のアプリケーションを呼び出す場合です。このリモート実行機能を可能にするアプリケーション・プロトコルは多数ありますが、最も知られているのは TELNET プロトコルです。この章ではこれらのプロトコルのいくつかを解説します。さらに分散コンピューティングの概念を解説します。

9.1 TELNET

TELNET は STD 番号 8 の標準プロトコルです。その状態は「推奨 (Recommended)」です。これは RFC 854 – TELNET Protocol Specifications と RFC 855 – TELNET Option Specifications に記述されています。

TELNET プロトコルには標準化されたインターフェースがあり、ホスト上のプログラム (TELNET クライアント) はこのインターフェースを介して、別のホストのリソース (TELNET サーバー) にアクセスすることができます。このときクライアントはあたかもそのサーバーに接続されたローカル端末のようになります。詳細については、図 151 を参照してください。

たとえば LAN 上のワークステーションのユーザーは、その LAN に接続されているホストに、そのワークステーションがあたかもそのホストに直接接続されている端末装置であるかのように接続することができます。もちろん TELNET は LAN 間だけでなく WAN 間でも使用できます。

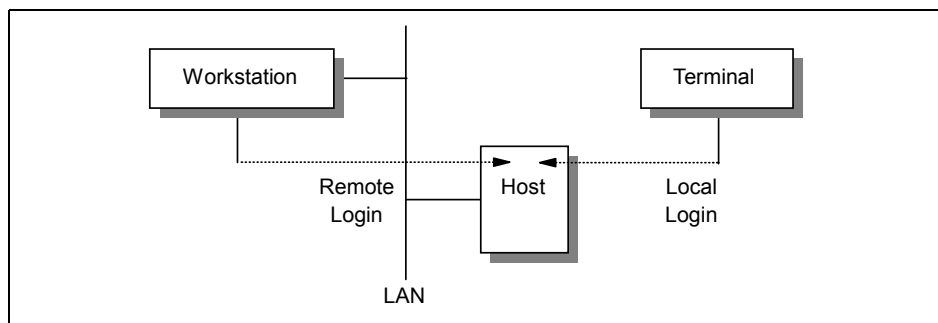


図 151 TELNET - TELNET を使用するリモート・ログイン

ほとんどの TELNET インプリメンテーションにはグラフィックス機能がありません。

9.1.1 TELNET 操作

TELNET プロトコルは次の 3 つの考えに基づいています。

1. ネットワーク仮想端末装置 (NVT) の概念。NVT は仮想上の装置ですが、現実には広範囲に実在する端末装置と共通の基本構造を持っています。各ホストは自分の端末特性を NVT の端末特性にマップし、他のホストもすべて同じことをするものと想定しています。
2. 端末装置とプロセスの対称視点。
3. 端末装置オプションのネゴシエーション。多くのホストは NVT で使用できるサービスの範囲を超えた別のサービスを使用しようとするため、ネゴシエーションされたオプションの基準が TELNET プロトコルで使用されます。さまざまなオプションをネゴシエーションすることができます。サーバーとクライアントは一連の規則を使用して、そこでの TELNET 接続の操作上の特性を "DO, DONT, WILL, WONT" メカニズム (この章で後述) で確立します。

2 つのホストは相互の協定の確認から始めます。最初のこのネゴシエーションが完了すると、これらは NVT にインプリメントされている最小限のレベルで動作できるようになります。この最小限の協定に達すると、使用している現実のハードウェアの機能をより正確に反映するために、追加オプションをネゴシエーションして NVT の機能を拡張することができます。TELNET では対称モデルが使用されるため (図 152 を参照)、使用する追加オプションの提案は、ホストとクライアントの両方が行うことができます。

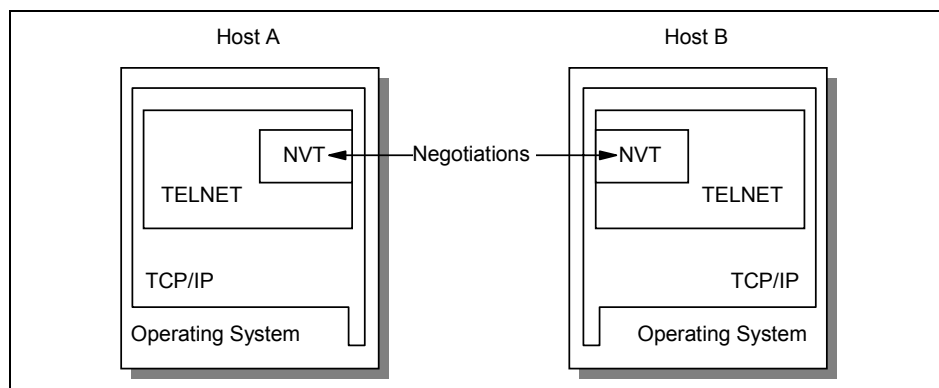


図 152 TELNET - 対称 TELNET モデル

9.1.1.1 ネットワーク仮想端末装置

NVT にはプリンター (つまりモニター) とキーボードがあります。キーボードは発信データを作成し、このデータは TELNET 接続上に送信されます。プ

リントーは着信データを受信します。NVT の基本特性は、相互に合意したオプションで変更されない限り、次のようになります。

- データ表記は 7 ビット ASCII で、これが 8 ビットでのバイト単位で伝送される。
- NVT は半二重装置で、ライン・バッファ・モードで動作する。
- NVT にはローカル・エコー機能がある。

これらはすべて 2 つのホストでネゴシエーションすることができます。たとえば、ネットワーク負荷が小さくなりパフォーマンスが良いためにローカル・エコーが使用されますが、リモート・エコー (図 153 を参照) を使用するためのオプションもあります (ただし、この使用が必須のホストはありません)。

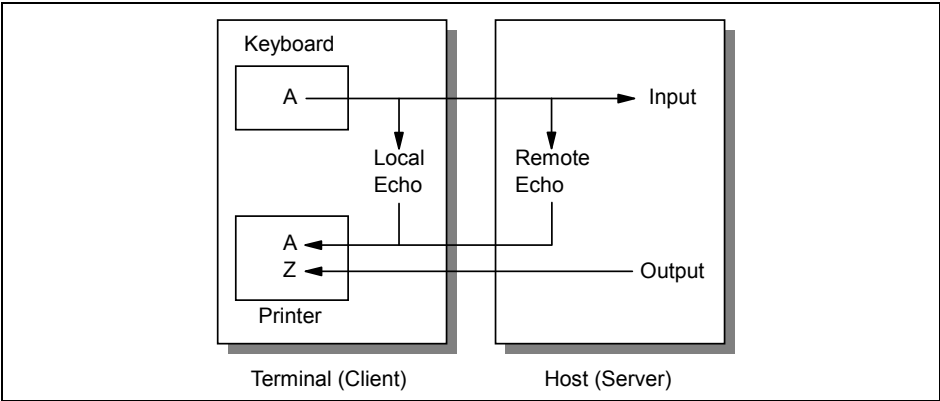


図 153 TELNET - エコー・オプション

NVT プリンターのキャリッジ幅とページ長さは指定されていません。このプリンターは印刷可能な ASCII 文字 (ASCII コード 32 ~ 126) を扱い、ASCII 制御文字のいくつかの次を処理できます。

表 10 TELNET ネゴシエーション

コマンド	ASCII	アクション
ヌル (NUL)	0	ノーオペレーション。
改行 (LF)	10	次の印刷行に移動する。このとき同じ水平位置を保持する。
復帰 (CR)	13	左マージンまで移動する。
ベル (BEL)	7	音によるまたは可視のシグナルを作成する。

コマンド	ASCII	アクション
バックスペース (BS)	8	印刷ヘッドを左マージンに向かって 1 文字位置だけ移動する。
水平タブ (HT)	9	印刷ヘッドを次の水平タブ停止位置まで移動する。
垂直タブ (VT)	11	次の垂直タブ停止位置まで移動する。
書式送り (FF)	12	次のページの先頭まで移動する。このとき同じ水平位置を保持する。

9.1.1.2 TELNET のオプション

TELNET オプションを広範囲でまとめたものがあります。それぞれの標準化の状態と状況は、STD 1 – Official Internet Protocol Standards を参照してください。本書を執筆の時点では、次のオプションが定義されていました。

表 11 Telnet のオプション

番号	名前	状態	RFC	STD
0	Binary transmission (バイナリー伝送)	Standard	856	27
1	Echo (エコー)	Standard	857	28
3	Suppress Go Ahead (ゴアヘッド抑制)	Standard	858	29
5	Status (状況)	Standard	859	30
6	Timing mark (タイミング・マーク)	Standard	860	31
255	Extended options list (拡張オプション・リスト)	Standard	861	32
34	Linemode (ライン・モード)	Draft	1184	
2	Reconnection (再接続)	Proposed		
4	Approximate. message size negotiation (メッセージ概略サイズ・ネゴシエーション)	Proposed		
7	Remote controlled trans and echo (リモート制御での伝送とエコー)	Proposed	726	
8	Output line width (行幅の出力)	Proposed		
9	Output page size (ページ・サイズの出力)	Proposed		

番号	名前	状態	RFC	STD
10	Output carriage-return disposition (復帰 (CR) 後処理の出力)	Proposed	652	
11	Output horizontal tabstops (水平タブ停止位置の出力)	Proposed	653	
12	Output horizontal tab disposition (水平タブ後処理の出力)	Proposed	654	
13	Output formfeed disposition (用紙送り後処理の出力)	Proposed	655	
14	Output vertical tabstops (垂直タブ停止位置の出力)	Proposed	656	
15	Output vertical tab disposition (垂直タブ後処理の出力)	Proposed	657	
16	Output linefeed disposition (改行後処理の出力)	Proposed	658	
17	Extended ASCII (拡張 ASCII)	Proposed	698	
18	Logout (ログアウト)	Proposed	727	
19	Byte macro (バイト・マクロ)	Proposed	735	
20	Data entry terminal (データ入力端末)	Proposed	1043	
21	SUPDUP	Proposed	736	
22	SUPDUP output (SUPDUP 出力)	Proposed	749	
23	Send location (ロケーション送信)	Proposed	779	
24	Terminal type (端末タイプ)	Proposed	1091	
25	End of record (レコード終わり)	Proposed	885	
26	TACACS user identification (TACACS ユーザー識別)	Proposed	927	
27	Output marking (マーキングの出力)	Proposed	933	
28	Terminal location number (端末ロケー ション番号)	Proposed	946	
29	Telnet 3270 regime	Proposed	1041	
30	X.3 PAD	Proposed	1053	

番号	名前	状態	RFC	STD
31	Negotiate window size (ウィンドウ・サイズのネゴシエーション)	Proposed	1073	
32	Terminal speed (端末速度)	Proposed	1079	
33	Remote flow control (リモート・フロー制御)	Proposed	1372	
35	X Display location (X モニター・ロケーション)	Proposed	1096	
39	Telnet environment option (Telnet 環境オプション)	Proposed	1572	
40	TN3270 enhancements (TN3270 拡張)	Proposed	1647	
37	Telnet authentication option (Telnet 認証オプション)	Experimental	1416	
41	Telnet xauth	Experimental		
42	Telnet charset	Experimental	2066	
43	Telnet remote serial port (Telnet リモート・シリアル・ポート)	Experimental		
44	Telnet com port control (Telnet COM ポート制御)	Experimental	2217	

Standard になっているオプションはすべて「推奨 (Recommended)」の状態であり、他のものは「任意選択 (Elective)」の状態になっています。Historic バージョンの TELNET Environment Option (Telnet 環境オプション) がありますが、これは推奨されていません。これは TELNET オプション 36 として RFC 1408 で定義されていたものです。

フルスクリーン機能

クライアントとサーバーのフルスクリーン機能に互換性があれば、フルスクリーン TELNET が可能です。たとえば VM と MVS は、TN3270 機能を持ったサーバーとして使用することができます。この機能を使用するためには、TELNET クライアントは TN3270 をサポートしていなければなりません。

9.1.1.3 TELNET のコマンド構造

クライアントとサーバー間の通信は内部コマンドで処理され、このコマンドはユーザーからはアクセスできません。内部 TELNET コマンドはすべて 2 または 3 バイトのシーケンスから成り、そのシーケンスはコマンド・タイプによって異なります。

Interpret As Command (IAC) 文字の後にコマンド・コードが続きます。このコマンドがオプションのネゴシエーションを処理する場合は、このコマンドに3番目のバイトが続き、関係するオプションのコードがそこで示されます。詳細については、図 154 を参照してください。

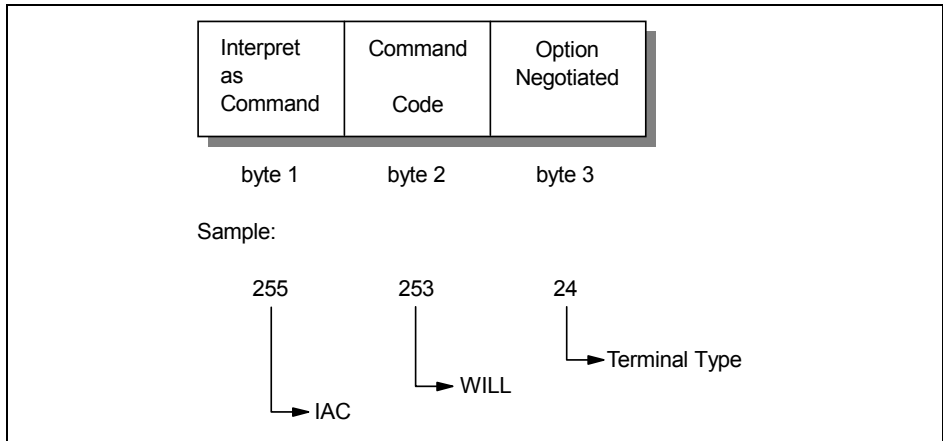


図 154 TELNET - 内部 TELNET コマンド (端末タイプについてのネゴシエーションを提案)

使用できるコマンド・コードのいくつかを次に示します。

表 12 コマンド・コード

コマンド	コード	コメント
SE	240	サブネゴシエーション・パラメーターの終わり。
NOP	241	ノーオペレーション。
Data Mark	242	同期のデータ・ストリーム部分。これには常に TCP 緊急通知が伴います。
Break	243	NVT 文字の BRK。
Go Ahead	249	GA シグナル。
SB	250	直後に続くコードで示されるオプションのサブネゴシエーションの開始。
WILL	251	直後に続くコードで示されるオプションを使用する意があること、またはそのオプションを使用することの確認を示します。
WONT	252	オプションの使用または継続使用の拒否を示します。

コマンド	コード	コメント
DO	253	直後に続くコードで示されるオプションを相手を使用することを要求、またはそのオプションを相手を使用することを予期していることを確認します。
DONT	254	直後に続くコードで示されるオプションを相手を使用するのを停止することを要求、またはそのオプションを相手を使用することはもう予期していないことを確認します。
IAC	255	コマンドとして解釈する (Interpret As Command)。後続のものは TELNET コマンドであり、データではないことを示します。

9.1.1.4 オプションのネゴシエーション

TELNET では内部コマンドを使用して各ホストでのオプションをネゴシエーションすることができます。ネゴシエーションの開始ベースは NVT の機能です。接続される各ホストは最低限これに合意しなければなりません。すべてのオプションのネゴシエーションを、4 つのコマンド・コード WILL、WONT、DO、および DONT を使って行うことができます。さらにいくつかのオプションにはサブオプションがあります。通信の双方は、オプションに合意したら、SB および SE コマンドを使ってサブオプションを管理することができます。次の表はオプションのネゴシエーションの方法の例を単純化して示したものです。

表 13 オプションのネゴシエーション

送信	応答	意味
DO transmit binary	WILL transmit binary	
DO window size	WILL window size	ウィンドウ・サイズをネゴシエーションしましょう。
SB Window size 0 80 0 24 SE		ウィンドウ・サイズを指定します。
DO terminal type	WILL terminal type	端末タイプをネゴシエーションしましょう。
SB terminal type SE		そちらの端末特性をこちらに送信してください。
	SB terminal type IBM=3278-2 SE	こちらの端末は 3278-2 です。
DO echo	WONT echo	

端末タイプは STD 2 – Assigned Numbers で定義されています。

9.1.1.5 TELNET 基本コマンド

TELNET プロトコルの本来のゴールは、ネットワーク上のホストの標準インターフェースを規定することです。TELNET プロトコルでは接続を開始するためのいくつかの機能について標準表記が定義されています。

IP	InterruptProcess (プロセス割り込み)
AO	Abort Output (出力打ち切り)
AYT	AreYou There (呼びかけ「もしもし」)
EC	Erase Character (文字消去)
EL	Erase Line (行消去)
SYNCH	Synchronize (同期化)

9.1.2 端末エミュレーション (Telnet 3270)

Telnet を使って SNA ホストに TCP/IP 接続を行うことができます。ただし 3270 Telnet エミュレーション (TN3270) を行うには Telnet 3270 を使用します。従来の Telnet と 3270 端末エミュレーションの間には次のような違いがあるため、TN3270 用に特別な Telnet オプションを追加して定義することが必要になります。

- 3270 端末エミュレーションでは、ライン・モードではなくブロック・モードを使用する。
- 3270 端末エミュレーションでは、ASCII 文字セットではなく EBCDIC 文字セットを使用する。
- 3270 端末エミュレーションでは、ATTN や SYSREQ などの特殊なキー機能を使用する。

Telnet での TN3270 接続は、次の 3 つの別々の Telnet オプションのネゴシエーションによって行われます。

1. Terminal type (端末タイプ)
2. Binary transmission (バイナリー伝送)
3. End of record (レコード終わり)

TN3270 サーバーは、最初のクライアント / サーバー・セッションのネゴシエーションのときは、これらの特性をサポートしていなければなりません。TN3270 ネゴシエーションのときの、バイナリー伝送およびレコード終わりオプションの送信の順序は任意です。TN3270 では TN3270 ネゴシエーション時にこれら以外のオプションは使用しないことに注意してください。つまり通常の Telnet オプションを使用します。TN3270 接続が確立されると、その他のオプションを使用することができます。使用できるオプションとして、TELNET-REGIME、SUPPRESS-GO-AHEAD、ECHO、および TIMING-MARK があります。

端末タイプ・オプションは、ホストの端末タイプを指定する IBM 3278-3 などのストリングです。3278 の後の -3 は、画面サイズとして標準サイズの 24x80 ではなく代替画面サイズを使用していることを示しています。バイナリー伝送 Telnet オプションは、その接続が初期 NVT モード以外になることを示します。クライアントまたはサーバーが NVT モードに切り替える場合は、バイナリー・オプションを使用不可にするコマンドを送信する必要があります。3270 データ・ストリームはコマンドと関連データから成っています。コマンドに関連したデータの長さはさまざまであるため、コマンドとその関連データはすべて IAC EOR シーケンスで分離する必要があります。EOR Telnet オプションは、この目的でネゴシエーション時に使用されます。

TN3270 接続のもう一つの重要な問題は、ATTN および SYSREQ 機能を正しく扱うことです。3270 ATTN キーは SNA 環境で現行プロセスに割り込むときに使用します。3270 SYSREQ キーは SNA 環境で接続を終了しないでセッションを終了するときに使用します。ただし SYSREQ および ATTN コマンドは、Telnet 接続で TN3270 サーバーに直接送信することはできません。TN3270 サーバーのほとんどのインプリメンテーションでは、BREAK コマンドを SNA ネットワークでのホストへの ATTN 要求に変換します。このためクライアント・サイドでは、キーやキーの組み合わせが BREAK にマップされます。SYSREQ キーの場合は、Telnet の Interrupt Process コマンドを送信するか、または SYSREQ コマンドを TN3270 データ・ストリームに組み込んで送信します。クライアント・サイドでは同様にキーやキーの組み合わせが SYSREQ 用にマップされます。

従来の TN3270 では扱えない機能がいくつかあります。その問題の中からいくつかを挙げると次のようになります。

1. TN3270 は 328x タイプのプリンターをサポートしていない。
2. TN3270 は SNA BIND 情報を扱えない。
3. SNA 肯定 / 否定応答プロセスのサポートがない。
4. 3270 の ATTN および SYSREQ キーが、すべてのインプリメンテーションでサポートされているわけではない。
5. TN3270 では Telnet セッションを SNA 装置名にマップできない。

9.1.3 TN3270 拡張 (TN3270E)

3270 構造化フィールドを使って非 3270 データを 3270 データの中に含めることができます。したがって、グラフィックス、IPDS プリンターのデータ・ストリームやその他のものを送信することができます。構造化フィールドは、構造化フィールド・コマンドとそのコマンドに続く 1 つ以上のブロックから成っています。ただし、すべての TN3270 クライアントがすべてのタイプのデータをサポートできるわけではありません。クライアントがこれらの機能のいずれかをサポートできるようになるには、サポートするデータ・タイプ

の範囲を Telnet 接続が確立されたときに決定する必要があります。このプロセスを TN3270 に追加する必要があります。従来の TN3270 の欠点を解消するために、TN3270 拡張属性が定義されています。TN3270 拡張 (TN3270E) の詳細については、RFC 2355 を参照してください。

TN3270E の拡張属性を使用するためには、クライアントとサーバーの両方が TN3270E をサポートしていなければなりません。いずれかのサイドが TN3270E をサポートしていない場合は、従来の TN3270 が使用されることになります。両方のサイドが TN3270E を使用することに合意すると、TN3270E オプションのサブセットのネゴシエーションが開始されます。その場合のオプションは装置タイプおよびサポートする一連の 3270 機能で、次のものがあります。

- ・ プリンター・データ・ストリーム・タイプ
- ・ 装置状況情報
- ・ サーバーからクライアントへの BIND 情報のパス
- ・ 肯定 / 否定応答交換

9.1.3.1 装置タイプのネゴシエーション

装置タイプ名は NVT ASCII スtringですべて英大文字です。TN3270E サーバーがクライアントに対して DEVICE-TYPE SEND コマンドを発行するときは、サーバーは装置タイプ、装置名、またはリソース名を示し、その後、DEVICE-TYPE REQUEST コマンドを続けます。装置タイプには次のものがあります。

表 14 TN3270E 装置タイプ - 端末装置

端末装置	端末装置 -E	画面サイズ
IBM-3278-2	IBM-3278-2-E	24 行 x 80 桁表示
IBM-3278-3	IBM-3278-3-E	32 行 x 80 桁表示
IBM-3278-4	IBM-3278-4-E	43 行 x 80 桁表示
IBM-3278-5	IBM-3278-5-E	27 行 x 132 桁表示
IBM-DYNAMIC	該当なし	該当なし

表 15 TN3270E 装置タイプ - プリンター

プリンター
IBM-3287-1

3278 と 3287 が一般的に使用されている装置であるため、装置タイプを 3278 と 3287 端末装置およびプリンターの各タイプに限定して、ネゴシエーションを単純化しています。このことは、他のタイプの装置を使用できないという

ことではありません。使用される 3270 装置の一般特性を、単に装置タイプのネゴシエーションで判別しているに過ぎません。クライアントがサポートしている 3270 データ・ストリームのさらに拡張された機能は、区分読み取り照会と照会応答の組み合わせで判別されます。

サフィックスの -E は、区画、グラフィックス、拡張カラー、および代替文字セットなどの拡張属性の使用を示しています。クライアントとサーバーが拡張属性の使用に合意し、-E サフィックスの装置や IBM-DYNAMIC 装置、プリンターなどについてネゴシエーションを終えたら、両方のサイドで 3270 構造化フィールドを扱えなければなりません。構造化フィールドを使用すると、3270 Telnet クライアントは、そのクライアントが使用できるホスト・アプリケーションに対しても、特定の 3270 データ・ストリームを送出することができます。

TN3270E クライアントの立場では、ネットワーク上で使用可能な装置名を知ることは常に可能または容易であるとは限りません。TN3270E サーバーはクライアントに対して適切な装置を割り当てる必要があります。これは TN3270E サーバーで定義されている装置プールを使って行われます。その装置プールには、端末装置やプリンターなどの基本的な SNA ネットワーク装置が含まれています。つまり TN3270E インプリメンテーションでは、TN3270 セッションを特定の SNA 論理装置 (LU) 名にマップすることにより、TN3270 セッションを有効な SNA 装置に変えています。装置プールは SNA ネットワーク装置を定義するだけでなく、TN3270E セッションのためのその他のいくつかの重要な機能にも使用されます。その中には次のものがあります。

- 特定の端末に対して 1 つ以上のプリンターを割り当てることができます。
- 特定の組織に対して装置のグループを割り当てることができます。
- ホスト上の特定のタイプのアプリケーションに対してだけアクセス権を持つプールを定義することができます。

TN3270E クライアントは CONNECT または ASSOCIATE コマンドを発行して、セッションを特定のタイプのリソースに接続または関連付けることができます。ただしそのリソースは、サーバー上での定義およびネゴシエーション時に判別された装置タイプと矛盾してはなりません。

9.1.4 参照

Telnet の詳細については、次の RFC を参照してください。

- RFC 854 – TELNET Protocol Specifications
- RFC855 – TELNET Option Specifications
- RFC 2355 – TN3270 Enhancements

9.2 リモート実行コマンド・プロトコル (REXEC と RSH)

Remote EXEcution Command Daemon (REXECD) は、TCP/IP ネットワーク上のリモート・ホストからサブミットされたジョブの実行を可能にするサーバーです。クライアントは REXEC または Remote Shell Protocol (RSH) コマンドを使ってジョブをそのサーバーに転送します。標準出力またはエラー出力が、表示またはさらに別の処理のためにクライアントに送信されて戻されます。

9.2.1 操作の原則

REXECD はサーバー (あるいはデーモン) です。これは外部のホストから発行されたコマンドを扱い、ジョブ実行のためのオーダーをスレーブ仮想計算機に転送します。ユーザー ID とパスワードが入力されている場合、このデーモンは自動ログインと自動ユーザー認証を実行します。

REXEC コマンドは、ユーザー ID、パスワード、ホスト・アドレス、およびリモート・ホスト上で始動させるプロセスの、それぞれの定義に使用されます。一方 RSH の場合はユーザー名とパスワードを送信する必要はありません。この場合はその代わりにホスト・アクセス・ファイルが使われます。サーバーとクライアントの両方が TCP/IP ネットワーク上でリンクされています。REXEC では TCP ポート 512 が、RSH では TCP ポート 514 が使用されます。詳細については、図 155 を参照してください。

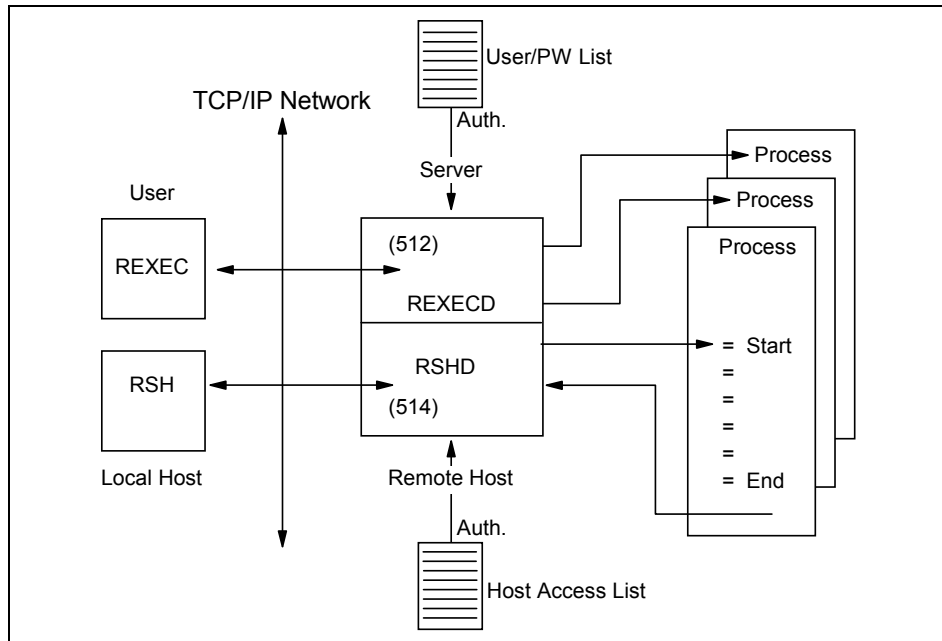


図 155 REXEC - REXECD の原則

9.3 分散コンピューティング環境 (DCE) の概要

分散コンピューティング環境 (DCE) はアーキテクチャーであり、マルチプラットフォーム・マルチベンダー環境で分散アプリケーションの開発と管理をサポートするために使用するオープン・スタンダード・サービスと関連 API の集合です。

DCE は OSF (Open Software Foundation - 現在はオープン・グループ (Open Group)) での作業の成果であり、多数のハードウェア・ベンダー、ソフトウェア・ベンダー、カスタマー、コンサルティング会社の共同作業によるものです。OSF は、ベンダーに中立なテクノロジーと業界標準の、調査研究、開発、および配布をサポートすることを目的に、1988 年に設立されました。そこで開発された標準の 1 つが DCE でした。DCE バージョン 1.0 は 1992 年 1 月にリリースされました。

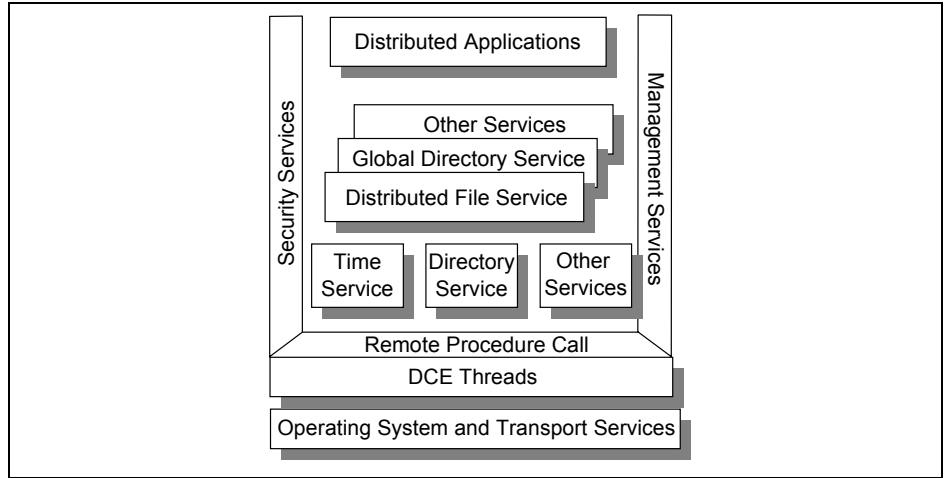


図 156 DCE アーキテクチャー上のコンポーネント

図 156 に示すように、DCE には次の主要サービスが組み込まれています。

- ディレクトリー・サービス
- セキュリティー・サービス
- Distributed Time Service (DTS)
- 分散ファイル・サービス
- スレッド
- リモート・プロシージャ・コール

上記のサービスはすべてアプリケーション・プログラム・インターフェース (API) を備えており、プログラマーはこれを使ってその機能を使用することができます。これらのサービスの詳細については、次のページから説明しています。

DCE アーキテクチャーではトランスポート・サービスに TCP/IP を使うように特に要求していませんが、オープンでマルチベンダーという DCE の設計目標に適合するプロトコルは現在他にはほとんどありません。実際のところ、すべてではないにしても、圧倒的多数の DCE インプリメンテーションが TCP/IP ネットワークを基にしています。

9.3.1 DCE ディレクトリー・サービス

大規模で複雑なネットワーク環境で作業をするときは、そのネットワークの参加者とリソースのロケーション、名前、サービス (およびその他の多数の詳細内容) が分かるようにしておくことは重要なことです。また、その情報に容

易にアクセスできることも重要です。これらを可能にするには、情報が論理的に中央になるロケーションに保管され、その情報にアクセスするための標準インターフェースがあることが必要です。DCE セル・ディレクトリー・サービスは、まさにこれになります。

DCE ディレクトリー・サービスには次の主要コンポーネントがあります。

- ・セル・ディレクトリー・サービス (CDS)
- ・グローバル・ディレクトリー・サービス (GDS)
- ・グローバル・ディレクトリー・エージェント (GDA)
- ・アプリケーション・プログラム・インターフェース (API)

9.3.1.1 セル・ディレクトリー・サービス

セル・ディレクトリー・サービスは、密接に連携しているホストのグループ内のリソースについての情報のデータベースを管理します。そしてこのグループをセルと呼びます。DCE セルは非常にスケーラブルであり、これに何千もの項目を含むことができます。かなり大規模な企業でも通常は単一セル内に編成され、これで何か国もカバーできます。ディレクトリー・サービスのデータベースには名前の一連の階層が含まれており、この名前はセル内のマシン、アプリケーション、ユーザー、およびリソースの論理ビューを表しています。これらの名前は通常、ディレクトリー単位でのディレクトリー項目になります、名前のこの一連の階層は、しばしばネーム・スペースとも呼ばれます。すべてのセルには、セル・ディレクトリー・サービスに構成された DCE サーバー (ディレクトリー・サーバー) が少なくとも 1 つ必要です。

CDS には非常に重要な特性が 2 つあります。分散できるということと、複製できるということです。分散という意味は、データベース全体をセル内の 1 台の物理マシン上に置く必要がないということです。データベースを複数のセクション (レプリカと呼ぶ) に論理的に分割し、各レプリカを別々のマシンに置くことができます。そのレプリカの最初のインスタンスがマスター・レプリカで、読み取り / 書き込みアクセスが可能です。セル・ディレクトリーを複数のマスター・レプリカに分割できるため、リソースの管理責任をセル内の別々の部分に分散させるオプションが可能です。このことは、セルがたとえば数か国をカバーしている場合は特に重要です。

マスター・レプリカは、それぞれ複製することができます。つまり、マスター・レプリカを別のマシン上にコピーすることができます (このマシンもディレクトリー・サーバーになります)。これは読み取り専用レプリカと呼ばれます。読み取り専用レプリカを使用すると、ホスト・マシンは最近隣の使用可能なレプリカを検索できるため、回復力とパフォーマンスの両方が向上します。

レプリカはクリアリングハウスに保管されます。クリアリングハウスは特定のサーバーにあるディレクトリー・レプリカの集合です。すべてのディレクトリー・レプリカは、クリアリングハウスの一部でなければなりません (同じクリアリングハウスである必要はありません)。

セル・ディレクトリー・サービス (CDS) は DCE セキュリティー・サービスを使用します。CDS は初期化時に DCE セキュリティー・サービスに対して自分自身を証明しなければなりません。これによって、不正 CDS が既存のセルに参加するのを防御しています。

図 157 は CDS ネーム・スペースのディレクトリー構造を示したものです。これで分かるように、ネーム・スペースは階層の方法で編成されます。

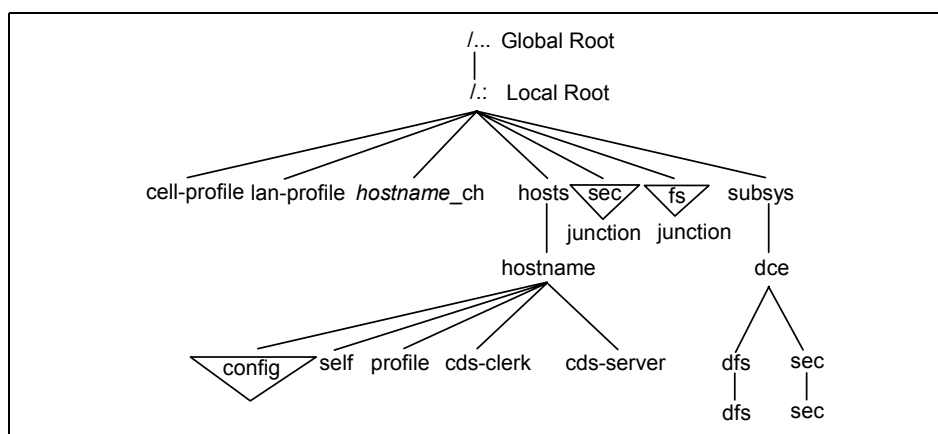


図 157 DCE - CDS ネーム・スペースのディレクトリー構造

DCE でのすべての名前が DCE ディレクトリー・サービスで直接保管されるとは限りません。セキュリティ・サービス (sec) や分散ファイル・システム (fs) などのいくつかのサービスで管理されるリソース項目は、接点と呼ばれる特殊な CDS 項目を使ってネーム・スペースに結合されます。接点項目にはバインディング情報が含まれており、クライアントはこの情報によってディレクトリー・サービスの範囲外のディレクトリー・サーバーに接続することができます。

セキュリティ・ネーム・スペースは、DCE セキュリティー・コンポーネントのレジストリー・サービスによって管理され、DFS ネーム・スペースは DFS のファイルセット・ロケーション・データベース (FLDB) サービスによって管理されます。

9.3.1.2 グローバル・ディレクトリー・サービスおよびエージェント

セル・ディレクトリー・サービスは、リソースがセル内のどこにあるかを認識しておく責任があります。しかしマルチセル・ネットワークでは、各セルはグローバル・ディレクトリー・ネーム・スペースと呼ばれる大規模な階層ネーム・スペースの一部になっています。グローバル・ディレクトリー・サービス (GDS) を使用すると、外部のセルにあるリソースのロケーションを解決することができます。これは会社がその会社の各セルを相互に接続したい場合や、インターネットに接続したい場合に該当します。

別のセルにあるリソースを検索するには、2 つのセルの間に通信パスが存在している必要があります。この通信パスは、現時点では次の 2 つのタイプのいずれかになります。

- CCITT X.500
- インターネット・ドメイン・ネーム・サービス (DNS)

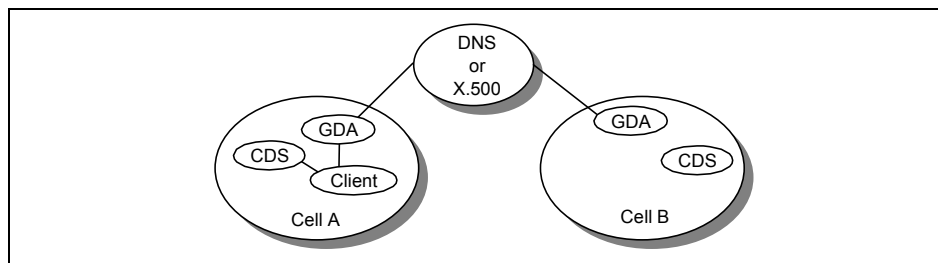


図 158 DCE - グローバル・ディレクトリー・エージェント

セル間通信を行えるようにするには、もう一つのコンポーネントであるグローバル・ディレクトリー・エージェントが必要です。グローバル・ディレクトリー・エージェント (GDA) は、ローカル・セルとグローバル・ディレクトリー・サービスの間に介在するものです。図 158 では、リソースのロケーションが CDS には不明の場合は、CDS はクライアントに対して GDA に援助を求めるよう通知します。GDA ではそれが接続されているグローバル・ネーム・スペースが分かっているため、通信を行う相手の外部セル・ディレクトリー・サーバーの名前を GDS (DNS または X.500) に照会します。外部セル・ディレクトリー・サーバーと直接通信を行えるようになると、要求されているリソースのネットワーク名を検索することができます。グローバル・ディレクトリー・エージェントは、DNS または X.500 の環境での通信をサポートするコンポーネントです。

9.3.2 DCE セキュリティー・サービス

ネットワーク環境ではセキュリティーは常に重要なことです。大規模分散環境では、すべての参加者が、処理することを許可されたデータだけにアクセ

スする有効なユーザーであることの確認は、さらに厳密になります。基本的に重要なのは認証と許可の2つです。認証は、ユーザーまたはサービスの識別を行うまたは確認するプロセスです。許可は、アクセスが試みられるときにユーザーの権限のレベルを検査するプロセスです。たとえば、読み取り専用アクセスの権限が付与されているユーザーが変更を行おうとすると、その更新の試みは失敗します。

DCE セキュリティー・サービスは、分散環境内のリソースへのセキュアな通信と制御されたアクセスを確保するものです。これはマサチューセッツ工科大学の Athena プロジェクトが基になっており、このプロジェクトで Kerberos が開発されました。Kerberos はユーザーやサービスの妥当性を検査する認証サービスです。現在の DCE セキュリティー・サービス (DCE 1.2.2) は Kerberos バージョン 5 がベースになっています。

DCE セキュリティー・サービスはユーザーとサービスの妥当性を検証できなければならないため、その情報を保持するためのデータベースも持たなければなりません。実際は次のようになっています。DCE セキュリティー・サービスは、プリンシパル、アカウント、グループ、組織、ポリシー、プロパティ、および属性のデータベースを維持しています。このデータベースはレジストリーと呼ばれます。図 159 はレジストリー・ツリーを絵で表したものです。レジストリーは、保管されるサーバーは異なりますが、実際はセル・ディレクトリー・ネーム・スペースの一部です。

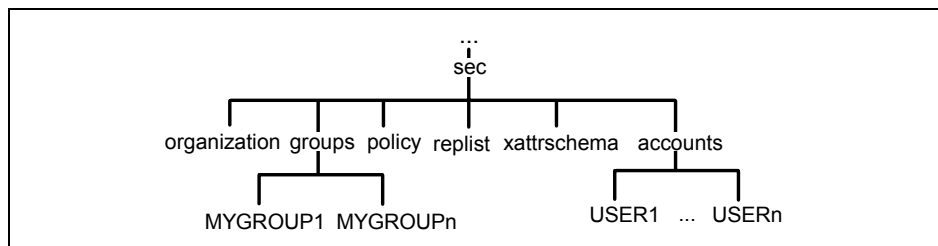


図 159 DCE - レジストリーのディレクトリー構造

DCE セキュリティー・サービスは次の複数のコンポーネントから成っています。

認証サービス

プリンシパルが正しく識別されていることを検査するプロセスを扱います。これにはセキュアな通信を可能にするチケット許可サービスも含まれています。

特権サービス

ユーザーの特権属性を示し、それを DCE サーバーに転送できるようにします。

レジストリー・サービス レジストリー・データベースを維持します。このデータベースには、アカウント、グループ、プリンシパル、組織、およびポリシーが含まれています。

アクセス制御リスト・ファシリティ プリンシパルのアクセス要求をリソースのアクセス制御と突き合わせるときに使用されるメカニズムです。

ログイン・ファシリティ ユーザーがログインしてセキュリティ環境とクリデンシャルを初期化するとき使用される環境です。

これらのサービスによって、ユーザー認証、セキュアな通信、リソースへの許可されたアクセス、およびセキュリティの適切な実施が可能になります。

DCE セキュリティ・サービスはセル・ディレクトリー・サービスと通信を行い、そのセルに含まれる他のシステムに対して自分の存在を公示します。DCE セキュリティ・サービスは Distributed Time Service も使用して、自分の多数のプロセスで使用するタイム・スタンプを取得します。

9.3.2.1 認証サービス

認証サービスの役割は、プリンシパルが自らを明確に識別して DCE ネットワークに参加できるようにすることです。DCE 環境ではユーザーとサーバーの両方が自分自身を証明します。これはユーザーだけが認証される他の大半のクライアント/サーバー・システムでのセキュリティとは異なります。認証には 2 つの異なるステップがあります。最初のログオン時に DCE 内で Kerberos 第三者プロトコルが使用され、DSS ネットワークへの参加を要求しているクライアントの識別の検証が行われます。このプロセスによってクライアントはクリデンシャルを取得することになります。クリデンシャルは、ユーザーがリソースへのアクセスを試みるときに、DCE サーバーとのセキュアなセッションのセットアップの基礎になるものです。

DCE バージョン 1.1 では事前認証の概念が採り入れられました。事前認証の概念は Kerberos 認証プロトコルにはありません。事前認証は、システムをハッキングするために、有効なユーザー ID を推測しようとする不正クライアントからセキュリティ・サーバーを保護するものです。DCE 1.1 には事前認証のためのプロトコルとして次の 3 つがあります。

事前認証なし これはバージョン 1.1 より前の DCE クライアントをサポートするためのものです。

タイム・スタンプ 第三者プロトコルを使用できない DCE バージョン 1.1 クライアントで使用されるものです。暗号化されたタイム・スタンプがセキュリティ・サーバーに送信さ

れます。タイム・スタンプが暗号化解除されてその時刻が 5 分以内のものだった場合に、そのユーザーは事前認証済みと見なされます。このオプションは、セル管理者および非対話のプリンシパルに対して指定する必要があります。

第三者

DCE バージョン 1.1 (および以降) のクライアントでデフォルトとして使用されます。これはタイム・スタンプ・プロトコルに似ていますが、このプロトコルではクライアントについての追加情報もさまざまな鍵で暗号化されます。

第三者事前認証プロトコルを使用したログインおよび認証プロセスを図 160 に示します。

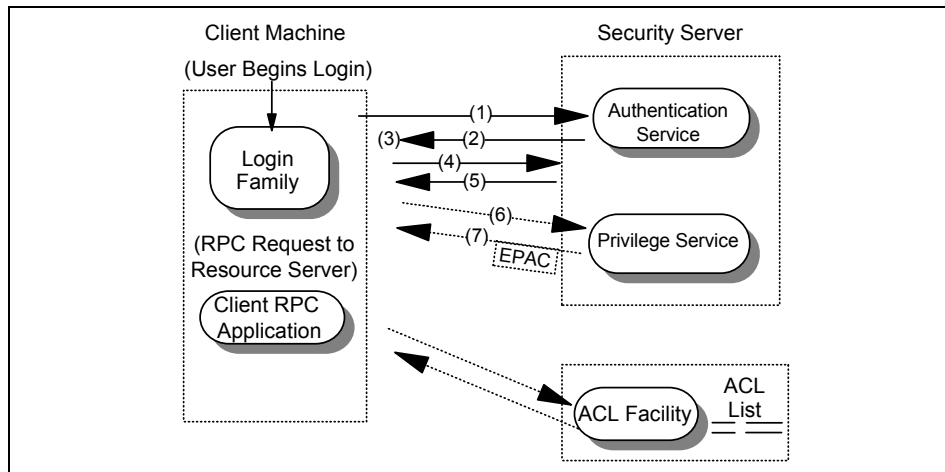


図 160 DCE - 第三者プロトコルを使用した認証とログイン

このプロセスの詳細は次のとおりです。

1. ユーザーがセルへのログの要求を発行します。しかしユーザーはまず認証されなければなりません。クライアントが会話鍵をランダムに 2 つ作成します。このうちの 1 つはマシン・セッション鍵を基にしています。次にクライアント上のログイン・ファシリティーがこれらの鍵と指定されたパスワードを使って、セキュリティ・サーバーへの認証チケット (チケット許可チケット (TGT)) の要求を暗号化します。
2. セキュリティ・サーバー上の認証サービス (AS) がその要求を受信します。AS はレジストリーからマシン・セッション鍵を検索して、その要求を暗号化解除します。(セキュリティ・サーバーは、マシン・セッション鍵を認識することによって、自分がそのセルで有効であることを証明し

ていることに注意してください。偽のサーバーの場合はマシン・セッション鍵を認識できません。) 暗号化解除が成功してタイム・スタンプが 5 分以内のものだったときは、AS は指定されたクライアント会話鍵の 1 つを使って TGT を暗号化します。これが正常に終了すると、その暗号化された TGT がクライアントに戻されます。

3. クライアントは TGT エンベロープを受信し、自分が作成したクライアント会話鍵の 1 つを使ってそれを暗号化解除します。AS のための会話鍵も組み込まれています。TGT 自体は AS の会話鍵を使って暗号化されていることに注意してください。この有効な TGT が、その時点でユーザーが認証されたという証明です。
4. ここでユーザーは特権サービス (PS) が発行する自分の許可証明書が必要です。この許可証明書は拡張特権属性証明書 (EPAC) と呼ばれます。このためには特権チケット許可チケット (PTGT) 要求を構成して、それを PS で検索しなければなりません。クライアントは PS と通信を行うために、PS と接触するという要求を AS に送信します。この要求は AS の会話鍵を使って暗号化されます。
5. AS がその要求を受信します。AS は、クライアントが PS に接触するとき使用する会話鍵を PS の秘密鍵を使って生成します。これはクライアントに戻されますが、AS 会話鍵を使って再び暗号化されます。クライアントがそのエンベロープを受信して (会話鍵を使って) 暗号化解除し、PS の会話鍵を見つけます。これでクライアントは特権サービス・チケットを PS に送信できるようになりました。
6. PS はその要求を受信し、その秘密鍵を使って正常に暗号化解除します。このことは、そのサービス・チケットが正当なものであることを証明しています。これはまた、関係する AS も正当であることを暗黙に含んでいます。このことから PS はクライアントと AS が有効であることを認識します。PS は EPAC を構成します。これには、ユーザーの標準および拡張レジストリー属性が、グループ・メンバーシップも含めてリストされています。PS はさらに会話鍵を作成し、EPAC と他の情報を、暗号化された PTGT エンベロープに入れてクライアントに送信します。
7. クライアントは PS 会話鍵を使って PTGT エンベロープを暗号化解除します。クライアントは、会話鍵情報と暗号化された PTGT (AS 秘密鍵を使って暗号化されているためクライアントは暗号化解除できない) も持っています。
8. これからクライアントがアプリケーション・サーバーに接触することになります。そのためにクライアントは、PTGT を AS に送信してアプリケーション・サーバー用のサービス・チケットを要求します。AS は PTGT を受信して暗号化解除し、EPAC 情報を取得します。AS はアプリケーション・サーバーの秘密鍵を使って EPAC 情報を暗号化し、アプリケーション・サーバーの会話鍵も提供します。この情報は AS の会話鍵 (クライアントは認識している) を使って暗号化され、クライアントに戻されます。

9. クライアントはエンベロープを暗号化解除し、アプリケーション・サーバーの秘密会話鍵を見つけます。これでクライアントは、その鍵を使ってアプリケーション・サーバーに接触することができます。アプリケーション・サーバーは、クライアントからの要求を正しく暗号化解除できたことにより、そのクライアントが認証済みであることを判別することができます。さらにクライアントへの応答により、クライアントは自分が接触したアプリケーション・サーバーが間違いなく本当のアプリケーション・サーバーだったことを認識します。次にこの2つは相互認証済みセッションを確立します。

ログオン時に秘密鍵が広く使用されるほか、第三者認証でタイム・スタンプを使って、会話を侵入者や盗聴から保護するようにします。記録とプレーバックなどの偽名使用の手法は、タイム・スタンプによって役に立たないものになります。また、ログオン時に入力された実際のユーザー・パスワード自体がサーバーに流出することはありません。代わりにユーザー・パスワードは、初期ログオン・メッセージの暗号鍵として使用されます。このログオン・メッセージは、セキュリティー・サーバーが、レジストリー・データベースに保管されているパスワードの、自分のところにあるコピーを使って暗号化解除します。

たとえば無効パスワードの入力など、何らかの理由でセキュリティー・サーバーがクライアントを認証できない場合は、エラーが戻されてそのログオンは終了します。一方、正常に認証されているクライアントとの交換が完了すると、セキュリティー・サーバーはクリデンシャルを戻します。このクリデンシャルは、クライアントがリソースおよびディレクトリー・サーバーなど他の DCE サーバーとのセッションの確立に使用します。クリデンシャルには、次に示す特権チケット許可チケット (PTGT) および拡張特権属性証明書 (EPAC) の形式で情報が含まれています。

EPAC セキュリティー・サーバーによって提供される検証済みのリストで、これにはクライアントの名前、そのクライアントが属しているグループ、およびその認証済みクライアントの拡張レジストリー属性 (定義済みでそのアカウントに関連付けられているものがある場合) が含まれています。クライアントは、接続するいずれのサーバーに対しても自分の EPAC (第三者認証時に取得済み) を示し、そのリソースにアクセスできるようにしなければなりません。

PTGT PTGT は特権チケット許可チケットのことです。これには EPAC が含まれており、EPAC にはユーザーについてのすべての関係情報 (UUID、グループ・メンバーシップ、ERA など) が入っています。PTGT は、DCE クライアントがリソースにアクセスする必要があるときに、DCE クライアントから DCE サーバーに実際に渡されるものです。

公開鍵のサポート

最新バージョンの DCE (DCE バージョン 1.2.2) では、公開鍵のテクノロジー (RSA やスマート・カードなどのテクノロジー) を使ってログインをサポートするという意見が採り入れられています。このテクノロジーを使うと、ユーザー (または他の DCE オブジェクト) の長期の鍵 (またはパスワード) をセキュリティ・サーバーに保管する必要がなくなるため、セキュリティ・サーバーの暗号漏えいのときのセキュリティを向上させることができます。管理者は、一部のプリンシパルは以前の DCE 1.2 メカニズムを使用できるようにし、その他のプリンシパルは公開鍵メカニズムへのアクセス権を持つように指定することができます。DCE 1.2.2 は、以前のリリースの DCE との完全なインターオペラビリティを維持しています。公開鍵ユーザーはログイン時にクリデンシャルを受け取り、これによって現行の (Kerberos ベースの) DCE 認証メカニズムを使用することができます。新規の事前認証プロトコルが使用されています。ログイン・クライアントは、示されたユーザーが公開鍵を使用可能かどうかを、クリデンシャルを要求する前に判別する必要はありません。

9.3.3 DCE スレッド

従来の (C や COBOL などの言語で書かれた) アプリケーションにはプログラミング・コードの行が多数あり、そのコードは通常は順次 to 実行されます。常にプログラムの中のどこかが実行されています。これを単一スレッドと定義することができます。スレッドとは、プロセス内の単一の実行の流れのことです。プログラムをいくつかの領域を並行して実行できるように構造化すると、アプリケーションのパフォーマンスが良くなる場合がしばしばあります。これをマルチスレッド化と呼びます。複数のスレッドを実行できる機能は、オペレーティング・システムにも依存します。

クライアント / サーバー・モデルに基づいている分散コンピューティング環境では、複数のスレッドを使用すると、同時に多数のプロシージャーを実行することができます。特定の応答を待機しているスレッドがブロックされている間 (たとえばネットワークからの応答待ち)、別のスレッドでの作業が続け行われます。サーバーはプロシージャー呼び出しの処理を並行して実行することができます。あるサーバーのスレッドが入出力操作の終了を待機している間、別のサーバーのスレッドが別の要求に対する処理を続行できます。

これが適切に機能するためには、オペレーティング・システムにスレッドのサポートが組み込まれている必要があります。複数のスレッドがオペレーティング・システム内ではなくアプリケーション・ソフトウェア・レベルにインプリメントされていると、マルチスレッド・アプリケーションのパフォーマンスは遅くなる場合があります。

DCE スレッド API は、(Windows 3.x など複数のスレッドをサポートしていないオペレーティング・システムでの) ユーザー・レベル、またはカーネル・

スレッド (AIX および OS/2 など) になります。これらは POSIX 1003.4a ドラフト 4 規格に基づいています。OS/2 にも複数のスレッドの機能があるため、プログラマーは DCE スレッドか OS/2 スレッドを使用できます。

DCE スレッドは特殊なプログラミング構成を使って、OS/2 スレッドにマップすることができます。ただしさまざまなプラットフォーム上で実行できるポータブルなアプリケーションを作成するためには、DCE スレッドのみを使用する必要があります。多くの場合、そのマッピングによってパフォーマンスに多少の差が生じます。

9.3.4 DCE リモート・プロシージャ・コール

DCE リモート・プロシージャ・コール (RPC) アーキテクチャーは、DCE 環境でのクライアントとサーバー間の通信の基礎になるものです。

注：

DCE RPC は概念的には ONC RPC (278 ページの 7.2.2 『リモート・プロシージャ・コール (RPC)』を参照) と似ていますが、ワイヤー上で使用されるプロトコルは互換性はありません。

RPC を使用すると、アプリケーション・プログラムのコードを複数システムに分散することができます。その場合の複数のシステムはネットワーク内ならどこにあってもかまいません。

DCE RPC を使って書かれたアプリケーションにはクライアント部分とサーバー部分があり、クライアント部分は通常は RPC 要求を発行し、サーバー部分は RPC 要求を受信してそれを処理し、その結果をクライアントに戻します。RPC には次の 3 つの主要コンポーネントがあります。

- Interface Definition Language (IDL) およびその関連コンパイラ。仕様ファイルから、ヘッダー・ファイル、クライアント・スタブ、およびサーバー・スタブを生成します。これを使用すると、アプリケーションはローカルのプロシージャ呼び出しを発行するときと同じようにリモート・プロシージャ・コールを発行することができます。
- ネットワーク・データ表記。入出力パラメーターなどのデータのパスのフォーマットをこれで定義します。これによって、ビット配列およびプラットフォーム固有データの表記が、ターゲット・システムに到着後に適切に変換できるようになります。RPC 呼び出しのためのデータの作成のこのプロセスを、マーシャルと言います。
- ランタイム・ライブラリー。これによってクライアント・ノードとサーバー・ノード間のネットワーク通信の詳細からアプリケーションがシールドされます。

アプリケーション・プログラマーは RPC 呼び出しを行うときに複数のスレッドの使用を選択できます。これは RPC が同期式であるためです。つまり RPC 呼び出しが行われると、その呼び出しを発行したスレッドは、応答を受信するまでそれ以降の処理がブロックされます。

リモート・プロシージャ・コールを使って、セル・ディレクトリー・サービス (CDS) やセキュリティ・サービスなどの他の DCE 機能を使用するアプリケーションを構築することができます。CDS を使って、サーバーを検出したり、クライアントのアクセスのためにサーバーのアドレスを公示することができます。セキュリティ・サービスを使って、さまざまなレベルのデータ保全性、Commercial Data Masking Facility (CDMF) や Data Encryption Standard (DES) を使った暗号化、および許可など他の機能を可能にする、認証済み RPC を作成することが考えられます。

9.3.5 Distributed Time Service

異なるホスト上のクロックを同期させておくことは、ハードウェアのクロックが通常は同一速度で動作しないため、困難な作業です。このことは、実行中に起こるイベントの順序に依存する分散アプリケーションにとっては問題になります。たとえばプログラマーがワークステーションで、あるコードをコンパイルしていて、いくつかのファイルがサーバーにも置かれている場合を考えてみます。ワークステーションとサーバーの時刻が同期していないと、データがサーバー上にある既存のものより古いと、コンパイラはファイルを処理できない可能性があります。実際にはそのファイルは新しいものなのですが、ワークステーションのクロックが遅れている場合です。その結果、コンパイルされたコードはソース・コードの最新の変更を反映していないことになります。この問題は、サーバーが複数の時間帯に分散している大規模セルではさらに顕著になります。

DCE の Distributed Time Service (DTS) は、分散環境での異なるホストのクロックを同期させるための標準ソフトウェア・メカニズムを備えています。また、ホストの時刻を絶対時刻に近似させておく手段も備えています。DTS はオプションです。これは DCE セルに必須のコア・サービスではありません。ただし DTS をインプリメントしない場合は、管理者が他の手段を使ってセル内のすべてのシステムでクロックを同期させておく必要があります。

Distributed Time Service (DTS) にはいくつかのコンポーネントが含まれています。次のもので構成されています。

- ローカル・タイム・サーバー
- グローバル・タイム・サーバー
- クーリエおよびバックアップ・クーリエ・タイム・サーバー

9.3.5.1 ローカル・タイム・サーバー

ローカル・タイム・サーバーの役割は、LAN 上の時刻クランクからの時刻の照会に応答することです。複数のローカル・タイム・サーバーが相互に照会して LAN 上での同期を維持することもあります。時刻クランクは、(minservers 属性で指定されている) 必要な数のローカル・タイム・サーバーに接触できなかった場合は、CDS 検索を使ってグローバル・タイム・サーバーに接触しなければなりません。

LAN ごとにローカル・タイム・サーバーが少なくとも 3 つあることが推奨されています。これによって LAN 上の時刻が同期化されるようになります。セル内の複数 LAN 間の同期化のタスクは、グローバルおよびクーリエ・タイム・サーバーによって行われます。

9.3.5.2 グローバル・タイム・サーバー

グローバル・タイム・サーバー (GTS) は、セル・ディレクトリー・サービスのネーム・スペースに自分を公示するため、すべてのシステムが容易に検出することができます。GTS は、ローカル・タイム・サーバーと同じようにローカル LAN に参加しますが、さらに追加の役割を担っています。これはその時刻を、別の LAN に置かれているクーリエ・タイム・サーバーにも示します。

9.3.5.3 クーリエの役割

ローカルおよびグローバル・タイム・サーバーはクーリエの役割を持つこともできます。つまりこれらのサーバーは、クーリエ、バックアップ・クーリエ、または非クーリエになることができます。クーリエは、他のローカル・タイム・サーバーと似たような振る舞いで、時刻同期化処理に参加します。ただしクーリエは自分自身のクロックは調べません。クーリエは、別の LAN またはセルの別の部分に置かれているグローバル・タイム・サーバーの時刻を要求します。時刻はネットワークの別の部分からインポートされるため、セルのすべての部分の多数のリモート LAN セグメントが、非常に近似した同期化された時刻値を持てるようになります。

バックアップ・クーリエの役割は、その LAN のプライマリー・クーリエが使用できないときにサポートすることです。バックアップ・クーリエはネゴシエーションして新しいクーリエを選び、グローバル・タイム・サーバーとの適切な時刻の同期化を維持します。なお、クーリエ・タイム・サーバーが定義されていない場合でも、ローカル・タイム・サーバーとクランクは、ローカル・セグメントの最小数のサーバーに接触できないときは、グローバル・タイム・サーバーに接触を試みます。

タイム・サーバーのデフォルトは非クーリエ役割です。十分な数のローカル・タイム・サーバーが接触可能であれば、それらのサーバーはグローバル・タイム・サーバーに接触しません。

大規模または分散ネットワークでは、ローカル・タイム・サーバー、グローバル・タイム・サーバー、およびクォーリエ・タイム・サーバーは、時刻同期化機能のプロセスを自動的に正確に実行します。

9.3.6 分散ファイル・サービス (DFS)

分散ファイル・サービスは実際には DCE のコア・コンポーネントではありませんが、他の DCE サービスと統合された、また他の DCE サービスを使用するアプリケーションです。DFS にはグローバル・ファイル共有の機能があります。アクセスするファイルが相互接続 DCE セル内のどこに置かれていても、そのファイルへのアクセスはユーザーにとって透過的になります。ユーザーにとっては、そのファイルがあたかもローカル・ドライブに置かれているように見えます。DFS サーバーおよびクライアントは、異なるオペレーティング・システムが動作している異機種混合のコンピュータでもかまいません。

DFS の原点は、カーネギー・メロン大学の Andrew file system (AFS) の、Transarc Corporation でのインプリメンテーションです (402 ページの 10.4 『Andrew File System (AFS)』を参照)。DFS は、ファイル・システム・セマンティクスとして POSIX 1003.1 に、アクセス制御セキュリティとして POSIX 1003.6 に、それぞれ準拠しています。DFS は他のすべての DCE サービス上に構築されて組み込まれています。これは分散ファイル・システムで明らかになった次のニーズに応えるために開発されたものです。

- ロケーションの透過性
- 一様なネーミング
- 高パフォーマンス
- セキュリティー
- 高可用性
- ファイルの整合性制御
- NFS インターオペラビリティ

DFS はクライアント / サーバー・モデルに従っており、DFS 管理可能ドメインを規定することにより DCE セルの概念を拡張しています。DFS 管理可能ドメインは、DFS サーバーおよびクライアントのシステムの、DCE セル内での管理上独立した集合です。

セル内に多数の DFS ファイル・サーバーを置くことができます。各 DFS ファイル・サーバーは、ファイルを DFS クライアントで使用できるようにするファイル・エクスポート・サービスを実行します。ファイル・エクスポートはプロトコル・エクスポートとも呼ばれます。DFS クライアントはキャッシュ・マネージャーを実行します。このキャッシュ・マネージャーは DFS サーバーにファイルを要求するアプリケーション間に介在するものです。キャッシュ・マネージャーは、ファイル要求をファイル・サーバー・システム上のファイル・エクスポートに対する RPC に変換し、サーバー・ア

クセスを最小限にするためにファイル・データをディスクまたはメモリーに保管します (キャッシュに入れる)。これによりクライアントは常にファイルの最新のコピーを持てることにもなります。

DFS ファイル・サーバーは、タイプが異なる次の 2 つのファイル・システムにサービスを提供します。

- ローカル・ファイル・システム (LFS)。これはエピソード・ファイル・システムとも呼ばれます。
- UNIX ファイル・システム (UFS) などの他のいくつかのファイル・システム。

DFS の全機能は LFS でのみ使用可能で、次の特徴があります。

- ハイパフォーマンス。
- 障害からの高速リカバリーのための、ログ・ベースのファイルセット高速再始動。
- 複製、自動更新、および障害ファイル・サーバーの自動バイパスによる高可用性。
- DCE セキュリティー・サービスとの統合により ACL 許可制御を行う強力なセキュリティ。

9.3.6.1 ファイルのネーミング

DFS ではその自己管理ネーム・スペースの接点として、セル・ディレクトリー・サービス (CDS) 名 `./fs` を使用します (367 ページの図 157 を参照)。セルの DFS オブジェクト (ファイルとディレクトリー) を基に各セルの `./fs` がルートになって、ファイル・システムのツリーが構築されます。ユーザーはネットワーク上のどこからでも、ディレクトリーとファイルにアクセスすることができます。このときに使用するファイル名やディレクトリー名は、ファイルやディレクトリーが物理的にどこに置かれていても同じです。これは DCE リソースはすべてグローバル・ネーム・スペースの一部になっているためです。

DFS ファイルのネーミングの例として、セル内の特定のファイルにアクセスする場合にユーザーが使用する名前は次のようになります。

```
././fs/usr/woodd/games/tictactoe.exe
```

セル外のファイルの場合は GDS (X.500) フォーマットを使って、次のような名前を使用します。

```
./.../C=US/O=IBM/OU=ITSC/fs/usr/woodd/games/tictactoe.exe
```

または DNS フォーマットで次のようにします。

```
./.../itsc.ibm.com/usr/woodd/games/tictactoe.exe
```

9.3.6.2 DFS のパフォーマンス

パフォーマンスは DFS の主要なゴールの 1 つですが、DFS では次の機能を組み込むことによりこれを達成しています。

キャッシュ・マネージャー

サーバーに要求したファイルはクライアントのキャッシュに保管されるため、クライアントはユーザーがファイルを必要とするたびにデータの要求をネットワークを介して送信する必要はありません。これによってサーバーのファイル・システムでの負荷が減少し、ネットワーク・トラフィックが最小になるため、パフォーマンスが向上します。

マルチスレッド・サーバー

DFS サーバーでは DCE の複数のスレッドのサポートを使って、複数クライアントからの複数のファイル要求を効率的に処理します。

RPC パイプ

RPC パイプ機能を広範囲で使って大量のデータ移送を効率的に行っています。

複製

複製をサポートしていることによりファイルの要求を複数サーバーに広げることができるため、効率的なロード・バランシングが可能です。

9.3.6.3 ファイルの整合性

クライアント・サイドのメモリーにキャッシュされているコピーされたファイルを使用することは、その元のファイルが別のロケーションにある複数のクライアントで使用されている場合、問題になる可能性があります。DFS ではトークン・メカニズムを使って複数ユーザーによる並行ファイル・アクセスを同期化し、各ユーザーは常に最新のバージョンのファイル进行处理できるようにしています。このプロセス全体はユーザーには透過的です。

9.3.6.4 可用性

可用性を向上させるために LFS ファイルセットを複数のサーバーに複製することができます。すべてのファイルセットに、単一の読み取り / 書き込みバージョンと複数の読み取り専用レプリカがあります。読み取り / 書き込みバージョンは変更できる唯一のものです。読み取り / 書き込みファイルセットでの変更はすべて、複製されたファイルセットに反映されます。複製されたファイルセットがあるサーバー・システムが破損した場合でも、作業は割り込みを受けません。このときクライアントは自動的に別のレプリカに切り替わられます。

9.3.6.5 DFS のセキュリティー

DFS は DCE のセキュリティーであるユーザー識別の認証、ユーザー特権の検査、および許可制御を使用しています。DFS では DCE のセキュリティーである ACL メカニズムを使って、オペレーティング・システムによる通常のもの (たとえば UNIX の読み取り、書き込み、および実行の許可) より柔軟で強力なアクセス制御を提供しています。

9.3.6.6 DFS/NFS インターオペラビリティ

DFS ファイルは NFS にエクスポートできるため、NFS クライアントは非認証ユーザーでもそのファイルにアクセスすることができます。このためには NFS/DFS 認証のためのゲートウェイ機能が必要ですが、そのゲートウェイ機能はすべてのインプリメンテーションで使用可能になっているわけではありません。

9.3.7 参照

DCE についての追加情報は、次の IBM Redbooks のいずれかを参照してください。

Understanding OSF DCE 1.1 for AIX and OS/2 (SG24-4616)

DCE Cell Design Considerations (SG24-4746)

Administering DCE and DFS 2.1 for AIX (and OS/2Clients) (SG24-4714)

Security on the Web Using DCE Technology (SG24-4949)

現行リリースの DCE (バージョン 1.2.2) についての情報は、オープン・グループの次の Web サイトで参照してください。

<http://www.opengroup.org/dce>

第 10 章ファイル関連のプロトコル

TCP/IP プロトコル・スイートにはファイルの操作のための多数のプロトコルがあります。リモート・ファイルにアクセスするためのメカニズムとして一般に 2 つあります。最も単純なメカニズムは特定のファイルをローカル・マシンに転送する方法です。この場合は同一ファイルの複数のコピーが存在するようになります。ファイル転送プロトコル (FTP) と Trivial File Transfer Protocol (TFTP) は、両方ともこのメカニズムでファイル共用を行っています。

ファイルにアクセスするためのもう一つの方法は、ファイル・システムを使用する方法です。この場合、ローカル・ホストの操作マシンは、リモート・マシン上のファイルにアクセスするための必要な機能を備えています。ローカル・マシン上のユーザーやアプリケーションは、そのファイルが実際はリモート・マシン上に置かれていることに気付きません。ユーザーやアプリケーションはそのファイルがあたかもローカル・マシン上にあるように、ファイル・システムを介してそのファイルの読み取りと書き込みを行うだけです。この場合はファイルのコピーが 1 つだけ存在し、その整合更新はファイル・システムが行います。Network File System (NFS) と Andrew File System (AFS) が備えている機能はこのタイプです。

10.1 ファイル転送プロトコル (FTP)

FTP は STD 番号 9 の標準プロトコルです。その状態は「推奨 (Recommended)」です。このプロトコルは RFC 959 – ファイル転送プロトコル (FTP) に記述されており、RFC 2228 – FTP Security Extensions で更新されています。

最も頻繁に行われる操作の 1 つとして、あるマシンから別のマシンへのファイルのコピーがあります。クライアントとサーバーの間のデータ転送は、いずれの方向でも行うことができます。クライアントはファイルをサーバー・マシンに送信することができます。また、そのサーバーのファイルを要求することもできます。

リモート・ファイルにアクセスする場合、ユーザーはユーザー自身の識別をサーバーに示さなければなりません。それを受けてサーバーは、ファイル転送を行えるようにする前にクライアントの認証を行います。

FTP ユーザーから見ると、このリンクはコネクション型です。つまりファイル転送を確立するためには、両方のホストで TCP/IP を起動して実行させておく必要があります。

10.1.1 FTP の概要

FTP ではトランスポート・プロトコルとして TCP を使って高信頼性エンドツーエンド接続を行います。FTP サーバーはポート 20 と 21 の接続を待機します。2 つの接続が使用されますが、1 番目のポートはログインのためのもので、その後は TELNET プロトコルに使用されます。2 番目のポートはデータ転送の管理のためのものです。リモート・ホストにログインするためがあるため、ユーザーはファイルとディレクトリーにアクセスするためのユーザー名とパスワードを持っていなければなりません。接続を開始するユーザーはクライアント機能とされ、サーバー機能はリモート・ホストによって提供されます。

FTP アプリケーションはリンクの両サイドとも、プロトコル・インタープリター (PI)、データ転送プロセス (DTP)、およびユーザー・インターフェースで構成されます (図 161 を参照)。

ユーザー・インターフェースはプロトコル・インタープリターと通信を行い、プロトコル・インタープリターには制御接続の機能があります。このプロトコル・インタープリターは、リモート・システムに対して必要な制御コマンドの通信を行います。

リンクの対向サイドのプロトコル・インタープリターでは、その機能として TELNET プロトコルに応答し、データ接続を開始します。ファイル転送の間のデータ管理は DTP によって行われます。ユーザーの要求が完了すると、サーバーの PI はデータ接続をクローズします。

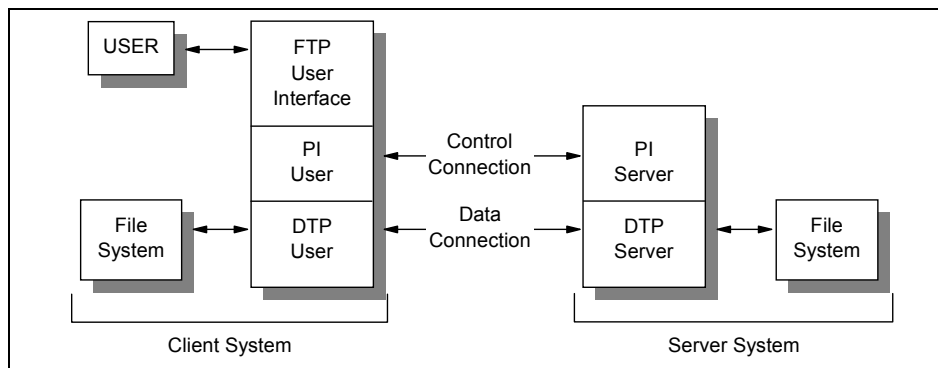


図 161 FTP - FTP の動作原理

10.1.2 FTP での操作

FTP を使用する場合、ユーザーは次の操作の一部またはすべてを実行することになります。

- ・ リモート・ホストへ接続する。
- ・ ディレクトリーを選択する。
- ・ 転送可能なファイルをリストする。
- ・ 転送モードを定義する。
- ・ リモート・ホストとの間でファイルをコピーする。
- ・ リモート・ホストとの接続を切断する。

10.1.2.1 リモート・ホストへの接続

ファイル転送を実行する場合、ユーザーはリモート・ホストへのログインから始めます。これはセキュリティの処理の基本的なメソッドです。ユーザーはアノニマス FTP (390 ページの 10.1.6 『アノニマス FTP』を参照) を使用する場合を除いて、リモート・ホストのためのユーザー ID とパスワードを持っていなければなりません。

使用されるコマンドとして次の 4 つがあります。

Open リモート・ホストを選択してログイン・セッションを開始する。

User リモート・ユーザー ID を識別する。

Pass ユーザーを認証する。

Site そのホスト固有のサービスを提供するために必要な情報を外部ホストに対して送信する。

10.1.2.2 ディレクトリーの選択

制御リンクが確立されたら、ユーザーは `cd` (change directory) サブコマンドを使って、対象とするリモート・ディレクトリーを選択できます。もちろん、ユーザーがアクセスできるディレクトリーは、リモート・ユーザー ID に適切な許可が与えられているディレクトリーだけです。ユーザーは `lcd` (local change directory) コマンドを使ってローカル・ディレクトリーを選択することができます。これらのコマンドの構文は、使用するオペレーティング・システムによって異なります。

10.1.2.3 転送可能なファイルのリスト作成

この操作は `dir` または `ls` サブコマンドを使って行います。

10.1.2.4 転送モードの指定

異システム間のデータ転送では、多くの場合、転送プロセスの一部としてデータの変換が必要になります。ユーザーはデータ処理に関して次の 2 点を決める必要があります。

- ・ ビットをある場所から別の場所へ移動させる方法
- ・ システムのアーキテクチャーに合わせた異なるデータ表記

これらは次の 2 つのサブコマンドを使って制御されます。

Mode	ファイルをバイト・ストリーム・フォーマットのレコード構造を持っているものとして扱うかどうかを指定する。
Block	ファイルの論理レコード境界が保持される。
Stream	ファイルはバイト・ストリームとして扱われる。これはデフォルトになっており、かなり効率的な転送が可能です。レコード・ベースのファイル・システムを処理する場合は期待した結果にならないことがあります。
Type	データで使用する文字セットを指定する。
ASCII	両方のホストが ASCII ベースであることを示す。または一方が ASCII ベースで他方が EBCDIC ベースの場合、ASCII-EBCDIC 変換を行うことを示します。
EBCDIC	両方のホストが EBCDIC データ表記を使用することを示す。
Image	データを 8 ビットのバイトにパックされた連続ビットとして扱うことを示す。

これらのサブコマンドはシステム間であり得るすべての相違点をカバーしているわけではないため、インプリメンテーションに依存したコマンドを発行する場合は、SITE サブコマンドを使用することができます。

10.1.2.5 ファイルの転送

FTP のクライアントとサーバー間でファイルをコピーする場合は、次のコマンドが使用されます。

Get	リモート・ホストのファイルをローカル・ホストにコピーする。
Mget	リモート・ホストの複数のファイルをローカル・ホストにコピーする。
Put	ローカル・ホストのファイルをリモート・ホストにコピーする。
Mput	ローカル・ホストの複数のファイルをリモート・ホストにコピーする。

10.1.2.6 受動モード (パッシブ・モード) の使用

受動モード (パッシブ・モード) ではデータ転送接続の確立の方向が逆になります。リモート・ホスト上の FTP サーバーがポートを選択し、FTP クライアント・プログラムに対して、リモート・ホスト上のサーバーにクライアントが接続するときに使用するポートを通知します。受動モードでは接続のため一時ポートを FTP サーバーが作成できるため、危険なサービスであるリスナーを固定ポートで行う必要がなくなります。また、制御接続とデータ接続

の両方が同一サイド (クライアント・サイド) から開始されるため、ファイアウォールのためのフィルター規則の構成が容易になります。そのためこのモデルは、ファイアウォール・フレンドリー・モードとも呼ばれます。FTP プロキシ・サーバーおよび受動モードの詳細については、712 ページの 21.3.4.2 『1 つの例: FTP プロキシ・サーバー』を参照してください。

10.1.2.7 プロキシ転送の使用

プロキシ転送を使用すると、接続が遅いクライアントが 2 つのリモート・サーバー間の第三者転送を使用することができます。あるサーバーに接続されているクライアントが、`proxy open` コマンドを発行することによって、そのサーバーを使用している別のサーバーとの FTP 接続を開きます。たとえば、クライアント A がサーバー B からファイルをダウンロードする必要があるがその接続が遅い場合です。この場合、クライアント A はまず最初にサーバー C に接続し、次に `proxy open server B` コマンドを発行してサーバー B にログインすることができます。クライアント A はサーバー C に対して、サーバー B のファイルを転送するための `proxy get filename` を送信します。

10.1.2.8 転送セッションの終了

FTP セッションを終了する場合は、次のコマンドが使用されます。

- Quit** リモート・ホストとの接続を切断して FTP を終了する。インプリメンテーションによっては BYE サブコマンドを使用しているものもあります。
- Close** リモート・ホストとの接続を切断するが、FTP クライアントは実行させておく。`open` コマンドを発行して、新しいホストを接続対象にすることができます。

10.1.3 応答コード

これらの操作を管理するために、クライアントとサーバーは TELNET の規則に基づいて対話を行います。クライアントがコマンドを発行し、サーバーは応答コードで応答します。この応答にはユーザーのためのコメントも含まれていますが、クライアント・プログラムで使用されるのはコードだけです。

応答コードの長さは 3 桁で、最初の桁が最も重要な桁です。

表 16 FTP 応答コード - 2 桁目と 3 桁目は応答の詳細を示す

応答コード	説明
1xx	仮の肯定応答
2xx	完了の肯定応答
3xx	中間の肯定応答
4xx	一時的な完了否定応答

応答コード	説明
5xx	永続的な完了否定応答

10.1.3.1 例

次の例は、各ユーザー・コマンドに対して、FTP サーバーは 3 桁の応答コードで始まるメッセージで応答します。

```

FTP foreignhost
220 service ready
USERNAME cms01
331 user name okay
PASSWORD xyxyx
230 user logged in
TYPE Image
200 command okay

```

10.1.4 FTP のシナリオ

ある LAN ユーザーが、ワークステーションのファイルを、VM を実行しているシステムに転送する必要があります。このファイルは、ワークステーションのディスク・ドライブから、CMS ユーザー cms01 が所有しているミニディスク 191 に転送される必要があります。リソース・アクセス管理機能 (RACF) はインストールされていません。インターネット・アドレスに対応するシンボル名は host01.itsc.raleigh.ibm.com です。詳細については、図 162 を参照してください。

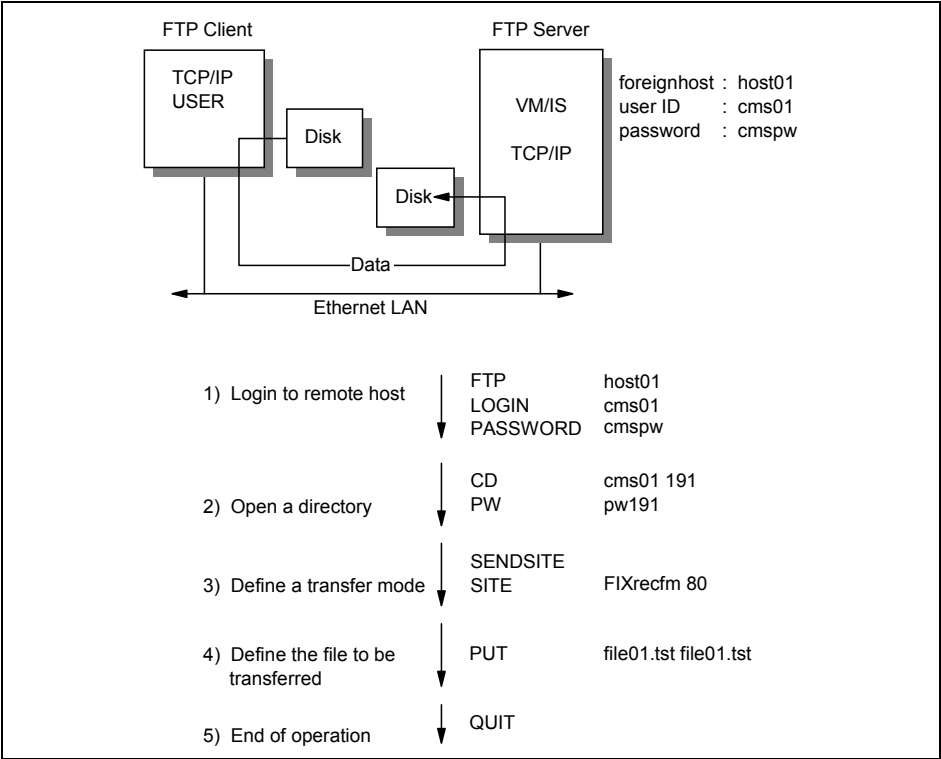


図 162 FTP - FTP のシナリオ

10.1.5 FTP サンプル・セッション

図 163 は FTP クライアント・プログラムから見たときの FTP セッションを示したものです。

```
[C:\SAMPLES]ftp host01.itsc.raleigh.ibm.com
Connected to host01.itsc.raleigh.ibm.com.
220 host01 FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
Name (rs60002): cms01
331 Password required for cms01.
Password: xxxxxx
230 User cms01 logged in.
ftp> put file01.tst file01.tst
200 PORT command successful.
150 Opening data connection for file01.tst (1252 bytes).
226 Transfer complete.
local: file01.tst remote: file01.tst
1285 bytes received in 0.062 seconds (20 Kbytes/s)
ftp> close
221 Goodbye.
ftp> quit
```

図 163 FTP - FTP サンプル・セッション

10.1.6 アノニマス FTP

多くの TCP/IP サイトでアノニマス FTP と呼ばれるものがインプリメントされており、これらのサイトではいくつかのファイル・ディレクトリーに対して公用アクセスが可能になっています。リモート・ユーザーが使用する必要があるのは、ログイン名として *anonymous*、パスワードとして *guest* またはたとえばユーザーのインターネット E メール ID などの他の共通パスワード規則だけです。システムで使用されるパスワード規則は、ログイン・プロセスのときにユーザーに示されます。

10.1.7 FTP を使用したリモート・ジョブ入力

MVS 上の FTP サーバーでは、ジョブ制御言語 (JCL) を内部読み取りプログラムに送信することができます。この機能を使って、TCP/IP での一種のリモート・ジョブ入力 (RJE) をインプリメントすることができます。この場合、送信されるファイルは実際はファイルではなくジョブであることを示すために、*site filetype=jes* サブコマンドが使用されます。MVS 上の FTP サーバーは次にそのジョブを、スプーリングと実行のために、ジョブ入力システム (JES) に転送します。個々のスプール・ファイルは FTP の *get* サブコマンドで受信できます。

10.2 Trivial File Transfer Protocol (TFTP)

TFTP プロトコルは STD 番号 33 の標準プロトコルです。この標準プロトコルの状態は「任意選択 I」で、RFC 1350 – The TFTP Protocol (Revision 2) に記述されています。TFTP のアップデートは RFC 1785、2347、2348、および 2349 に記述されています。

TFTP ファイル転送はディスク対ディスクのデータ転送です。これに対してたとえば VM SENDFILE コマンドは、TCP/IP の領域ではメーリング機能と見なされていますが、この場合は誰かのメールボックス (VM の場合は読み取りプログラム) にデータを送信します。

TFTP はファイルを転送するための非常に単純なプロトコルです。この単純さはこのプロトコルを容易にインプリメントできるように意図的になされたものです。ディスクレス装置の中にはブート時に TFTP を使って必要なファームウェアをダウンロードするものもあります。

TFTP は UDP (User Datagram Protocol) の上位にインプリメントされます。最初に TFTP クライアントがポート 69 を使って読み取り / 書き込み要求を送信し、次にサーバーとクライアントがあとの接続に使用するポートを決定します。TFTP には FTP の機能 (383 ページの 10.1 『ファイル転送プロトコル (FTP)』を参照) の大半がありません。TFTP が行えるのは、サーバーとの間のファイルの読み取り / 書き込みだけです。

注：

TFTP はユーザー認証のためのものは備えていません。この点ではこれは非セキュアなプロトコルです。

10.2.1 TFTP の使用法

コマンド TFTP <hostname> で対話式プロンプトが出されます。ここで次のようなサブコマンドを入力できます。

Connect <host>	宛先ホスト ID を指定する。
Mode <ascii/binary>	転送モードのタイプを指定する。
Get <remote filename> [<local filename>]	ファイルを検索する。
Put <remote filename> [<local filename>]	ファイルを保管する。
Verbose	冗長モード (ファイル転送中に追加情報を表示する) のオン / オフを切り替える。

Quit

TFTP を終了する。

これらのコマンドの詳細なリストは、TFTP インプリメンテーションそれぞれのユーザーズ・ガイドを参照してください。

10.2.2 プロトコルの説明

転送は常にファイルの読み取りまたは書き込みの要求から始まります。サーバーがその要求を認可すると、接続が開いて、ファイルが 512 バイト (固定長) のブロックで送信されます。ファイルのブロックには 1 から始まる連続番号が付けられます。データ・パケットごとに応答パケットで確認されてから、次のデータ・パケットの送信が可能になります。512 バイトより少ないデータ・パケットがきたときにその転送は終了と見なされます。

エラーのほとんどすべての場合に接続が終了してしまいます (信頼性の欠如)。ネットワーク内でパケットの脱落が起こるとタイムアウトが発生し、その後最後のパケット (データまたは応答) の再送が行われます。

RFC 783 に Sorcerer's Apprentice Syndrome と呼ばれる重大なバグがありました。それは、ネットワークの遅延の状況によっては、両サイドで過度の再送が行われる場合があるというものです。これは RFC 1123 に記述されていたものですが、RFC 1350 で訂正されました。TFTP パケットの拡張として OACK パケットが追加されました。これは RFC 2347 に記述されています。詳細については、それぞれの RFC を参照してください。

10.2.2.1 TFTP パケット

次の 6 タイプのパケットがあります。

表 17 TFTP パケット・タイプ

命令コード (Opcode)	操作
1	読み取り要求 (RRQ)
2	書き込み要求 (WRQ)
3	データ (DATA)
4	応答 (ACK)
5	エラー (ERROR)
6	オプション応答 (OACK)

TFTP ヘッダーには、パケットに関連した命令コードが含まれています。詳細については、図 164 を参照してください。

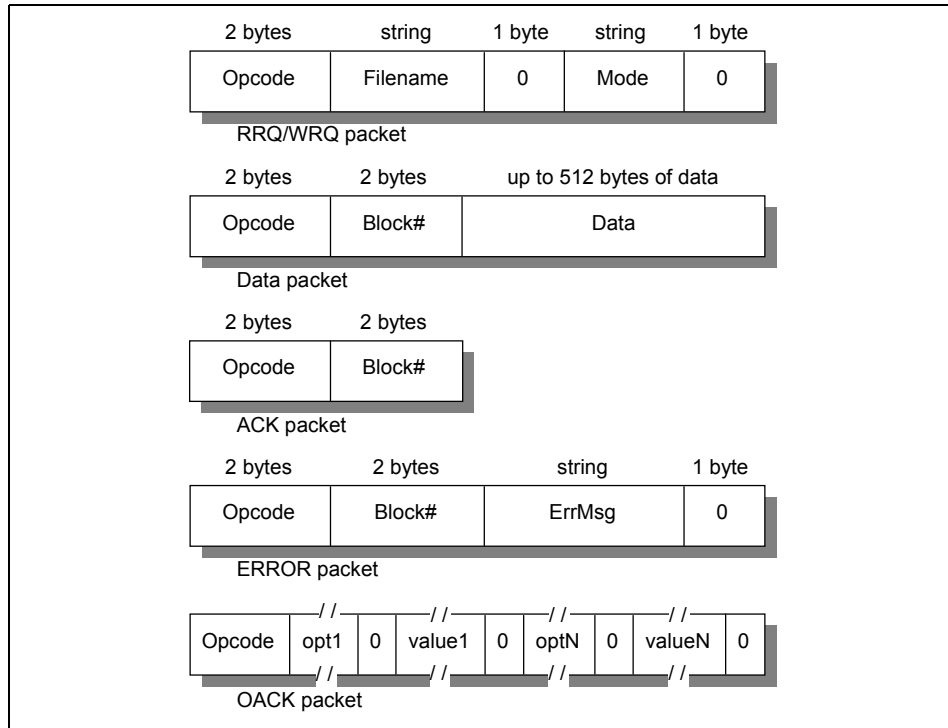


図 164 TFTP- TFTP パケット

10.2.2.2 データ・モード

現在 RFC 1350 で、次の 3 つのモードの転送が定義されています。

NetASCII USA Standard Code for Information Interchange で定義されている US-ASCII を RFC 854 – Telnet Protocol Specification の仕様で変更して、高位ビットを使用するように拡張したもの。(つまりこれは 8 ビットの文字セットです。一方 US-ASCII は 7 ビットです。)

Octet 生の 8 ビットのバイトで、バイナリーとも呼ばれます。

Mail このモードは最初は RFC 783 で定義されていましたが、RFC 1350 によって廃止が宣言されました。これはファイルへの転送ではなくユーザーへのメールの送信を可能にするものでした。

使用するモードは読み取り / 書き込み要求 (RRQ/WRQ) パケット内で示されます。

10.2.3 TFTP マルチキャスト・オプション

このオプションを使用すると複数のクライアントが、マルチキャスト・パケットを使って、サーバーから同時にファイルを取得することができます。たとえば類似している 2 台のマシンがリモートでブートされたとき、それらのマシンは TFTP オプション・セットにマルチキャスト・オプションを追加することにより、同一の構成ファイルを同時に検索することができます。TFTP マルチキャスト・オプションは RFC 2090 に記述されています。

次に示しているのは、マルチキャスト・オプションを組み込んで変更された TFTP 読み取り要求パケットです。

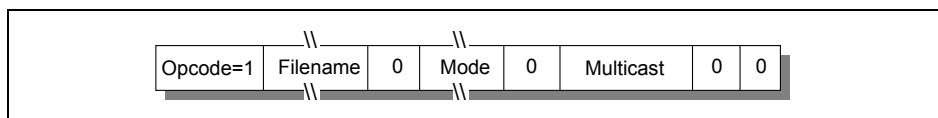


図 165 TFTP - マルチキャスト・オプションを組み込んだ読み取り要求パケット

サーバーはマルチキャストを受け入れる場合は、マルチキャスト・オプションを組み込んだオプション応答 (OACK) パケットをサーバーに対して送信します。このパケットは、マルチキャスト・アドレス、およびクライアントが応答 (ACK) を送信すべきかどうかを指定するフラグから成っています。

10.2.4 セキュリティの問題

TFTP には認証メカニズムがないため、サーバーがホストのファイルを保護する必要があります。TFTP サーバーでは一般に、書き込みアクセスは不可であり、公用ディレクトリーに対する読み取りアクセスだけが可能です。サーバーのインプリメンテーションによっては、ホスト・アクセス・リストを持っているものもあります。

10.3 Network File System (NFS)

SUN Microsystems の Network File System (NFS) プロトコルを使用すると、複数のマシンがネットワーク上の複数のファイル・システムを共用することができます。NFS プロトコルは、マシン、オペレーティング・システム、およびトランスポート・プロトコルに依存しないように設計されています。これはリモート・プロシージャー・コール (278 ページの 7.2.2 『リモート・プロシージャー・コール (RPC)』を参照) の上位のインプリメンテーションにより実現されています。RPC では、外部データ表記規則を使用することによりマシンの独立性を確立しています。

SUN-NFS は提案標準のプロトコルです。その状態は「任意選択 (elective)」になっています。NFS の現在の仕様は、RFC 1813 – NFS: NFS Version 3

Protocol Specification、および RFC 3010 – NFS version 4 Protocol に書かれています。

10.3.1 NFS の概念

NFS では許可ユーザーが、リモート・システム上に置かれているファイルに対して、それらがあたかもローカルのものであるようにアクセスすることができます。この目的のために次の 2 つのプロトコルを使用しています。

1. マウント・プロトコルは、アクセスされるリモート・ホストとファイル・システム、およびそれらをローカル・ファイル階層のどこに配置するかを指定します。
2. NFS プロトコルは、リモート・ファイル・システムとの実際のファイル入出力を行います。

マウントおよび NFS プロトコルはいずれも RPC アプリケーションであり (呼び出し側 / サーバーの概念)、トランスポートは TCP と UDP によって行われます。

10.3.1.1 マウント・プロトコル

マウント・プロトコルは NFS に同梱されている RPC アプリケーションです。そのプログラム番号は 100005 です。マウント・プロトコルのトランスポートは TCP と UDP の両方で行われます。マウントは RPC サーバー・プログラムであり、次に示す合計 6 プロシーチャーをサポートしています。

NULL	無実行。サーバーの応答テストの際に便利です。
MNT	マウント機能。ディレクトリーを指すファイル・ハンドルを戻します。
DUMP	マウント済みのすべてのファイル・システムのリストを戻す。
UMNT	マウント・リスト項目を除去する。
UMNTALL	このクライアントのマウント・リスト項目をすべて除去する。
EXPORT	使用可能なファイル・システムについての情報を戻す。

MOUNT 呼び出しでディレクトリーのファイル・ハンドルが戻されます。このファイル・ハンドルは最大 64 バイトの可変長配列で、続いてクライアントがファイルをアクセスするときに使用します。各ディレクトリーおよびファイルはハンドルを介して参照されるため、ファイル・ハンドルは NFS の重要な部分です。インプリメンテーションによっては、セキュリティ上の理由でハンドルを暗号化するものも出てくるでしょう。(たとえば VM 上の NFS では、この機能を持たせるために、オプションとして VM 暗号化プログラムを使用することができます。)

この RPC アプリケーションのユーザー・インターフェースは MOUNT コマンドになります。ユーザーは MOUNT コマンドを発行して、ユーザーのファイル階層にリモート・ファイル・システムを配置します。

たとえば VM NFS サーバーを考えてみます。これにはサブディレクトリー（階層ファイル・システム）の概念はありません。複数のミニディスクがあるだけです（それぞれが 1 ディレクトリーと見なされます）。次に AIX クライアントを考えてみます。（AIX にはサブディレクトリー・ファイル・システムがあります。）クライアントは次のように MOUNT コマンドを発行することにより、自身のローカル・サブディレクトリー /u/vm/first として、user 191 の VM ミニディスクにアクセスすることができます。

```
MOUNT -o options
      host:user.191,ro,pass=password,record=type,names=action
      /u/vm/first
```

各部の説明：

- options** メッセージ・サイズなどのシステム・オプション。
- host** リモート・ホストの TCP/IP 名。
- user** VM ユーザー ID。
- 191** ミニディスクのアドレス。
- pass=** NFS マシンがミニディスクにアクセスできるようになるリンク・パスワード。
- record=** CMS レコードに対して行われる変換処理を指定します。
 - binary** 無処理。
 - text** EBCDIC (サーバー) と ASCII (クライアント) 間のコード変換。
 - nl** EBCDIC から ASCII への変換で、改行文字は CMS レコード境界として解釈されます。
- names=** ファイル名の扱いを次のように指定します。
 - fold** クライアントによって指定されたファイル名が英大文字に変換されます。
 - mixed** クライアントによって指定されたとおりのファイル名が使用されます。

名前変換オプションが指定されない場合は大文字変換が行われ、さらに CMS では無効なクライアント名が有効な CMS 名に変換されます。

この結果、VM ミニディスクがクライアント・マシンでローカル・サブディレクトリーとして表示されます。

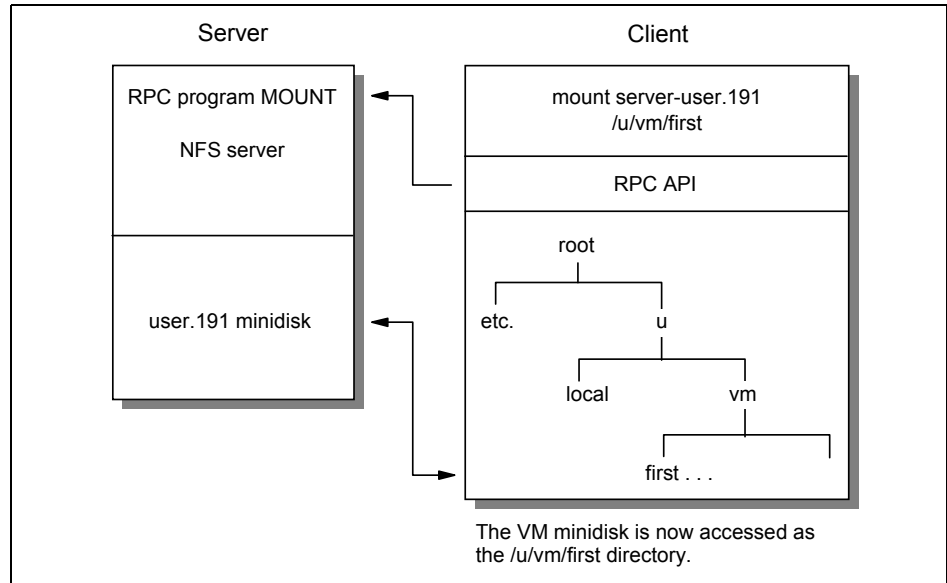


図 166 NFS - クライアントが VM ミニディスク user.191 を自身のローカル・ディレクトリーとしてマウントする

前述のコマンドは次のものでした。

```
MOUNT -o options host:user.191,ro,pass=password,record=type,names=action  
/u/vm/first
```

これには次の 3 つの部分があります。

1. -o options はクライアント部分です。これは NFS クライアントだけが分かっている部分です。つまり、この部分はクライアント・ホストによって異なり、それについてはクライアントの資料に記載されています。
2. host:user.191,ro,....,names=action はサーバー部分です。この構文はサーバーのファイル・システムによって異なります。(当然のことですが、user.191 は MVS NFS サーバーにとっては意味はありません。) NFS サーバーが受け入れるパラメーターについては、NFS サーバーの資料を参照してください。
3. /u/vm/first はクライアント部分でマウント・ポイントと呼ばれ、リモート・ファイル・システムがローカル・ファイル・システムにフックされる場所です。

UMOUNT コマンドは、ローカル・ファイル階層からリモート・ファイル・システムを除去します。上記の例に続いて次のコマンドを使用すると、`/u/vm/first` ディレクトリーが除去されます。

```
UMOUNT /u/vm/first
```

10.3.1.2 NFS プロトコル

NFS は RPC のアプリケーション・プログラムであり、MOUNT コマンドを使ってリモート・ホストが要求された後に、そのリモート・ホストとのファイル入出力機能を提供します。このプログラム番号は 100003 で、IP ポート 2049 を使用する場合があります。これは公式に割り当てられたポートではなく、またいくつかのバージョンの NFS (およびマウント) がすでに存在しているため、ポート番号は変更することができます。マウントおよび NFS プロトコルの両方のポート番号を、ポート・マップ (ポート番号 111) (281 ページの『ポート・マップまたはポートマッパー』を参照) で取得することをお勧めします。NFS プロトコルのトランスポートは TCP と UDP の両方で行われます。

NFS プログラムでは 22 のプロシーチャーをサポートしており、次のように基本入出力操作をすべて備えています。

ACCESS	そのユーザーについて、ファイルの許可の設定に従ってアクセス権限を解決する。
LOOKUP	現行ディレクトリーでファイルを検索し、見つかった場合はそれを指すファイル・ハンドルとそのファイルの属性情報を戻す。
READ/WRITE	ファイルをアクセスするための基本的な読み取り / 書き込みプリミティブ。
RENAME	ファイルをリネームする。
REMOVE	ファイルを削除する。
MKDIR, RMDIR	サブディレクトリーの作成 / 削除。
GET, SET-ATTR	ファイル属性を取得または設定する。

この他の機能も備えています。

これらはローカル・オペレーティング・システムでローカル・ファイルのアクセスに使用されるファイル入出力プリミティブの大部分に相当します。実はローカル・オペレーティング・システムは、リモート・ディレクトリーがマウントされたら、ファイル入出力プリミティブをリモート・ホストに転送するだけです。これによって、ファイルがローカルに置かれているかリモート側に置かれているかにかかわらず、すべてのファイル入出力が同じように見えることとなります。ユーザーは両方の種類のファイルに対して自身の通

常のコマンドやプログラムを操作することができます。つまり、この NFS プロトコルはユーザーには完全に透過的です (図 167 を参照)。

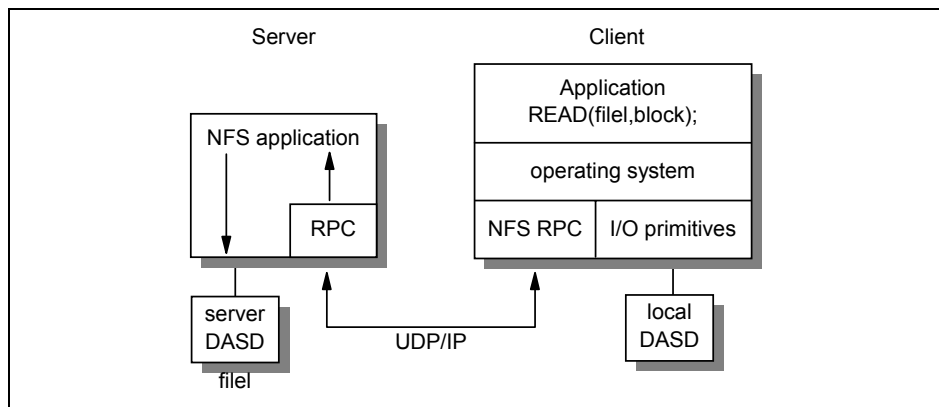


図 167 NFS - オペレーティング・システム・レベルでインターセプトされてユーザーには透過的なファイル入出力

10.3.1.3 ファイル整合性

NFS はステートレス・プロトコルであるため、NFS にマウントされたファイルのファイル整合性を保護する必要があります。多くのインプリメンテーションではこの目的のために、ロック・マネージャー・プロトコルを組み込んでいます。状況によっては複数のプロセスが同時に同じファイルを開く可能性があります。そのファイルが単に読み取りアクセスのために開かれた場合は、すべてのプロセスが最新のデータを取得することになります。そのファイルに書き込みを行うプロセスが同時に複数存在している場合は、書き込み側によって行われた変更が整合されると同時に、読み取り側には最も正確な変更内容が与えられなければなりません。ファイルの一部が各クライアントのローカル・システムのキャッシュに入っていると、それに応じて書き込み側と読み取り側を同期させるのは一層複雑なジョブになります。このような状況が頻繁に起こる可能性はそれほどありませんが、このような問題を考慮しておかなければなりません。

ロック・マネージャー・プロトコル

このプロトコルを使用すると、クライアントおよびサーバー・プロセスは、ファイルに書き込むときに他のプロセスを排除することができます。プロセスがファイルを排他的アクセス用にロックすると、他のプロセスはそのファイルにアクセスできません。プロセスがファイルを共用アクセス用にロックすると、他のプロセスはそのファイルを共用できますが、排他的アクセスのロックを開始することはできません。他のプロセス要求がロック状態と競合した場合、そのプロセスはロックが除去されるまで待機するか、またはエラー・メッセージを戻す場合があります。

10.3.1.4 NFS ファイル・システム

NFS は階層ファイル・システム (ディレクトリー) を前提としています。ファイルは非解釈バイトの非構造化ストリームです。つまりファイルは、レコード・レベルの構造を持たない連続バイト・ストリームとして見ることができます。

これは AIX と PC/DOS で使用されている種類のファイル・システムであり、そのためこれらの環境では、それらの独自のローカル・ファイル・システムに NFS クライアントを拡張して組み込むことは容易です。VM および MVS で使用されるファイル・システムでは、その種の拡張はそれほど容易ではありません。

NFS ではファイル操作はすべて同期しています。したがって、ファイル操作の呼び出しは、サーバーがその操作の処理をすべて完了したときだけ戻ることになります。書き込み要求の場合、サーバーはディスクに物理的にデータを書き込みますが、必要であればディレクトリー構造を更新してからクライアントに応答を戻します。こうすることによってファイル整合性が確保されています。

NFS では、サーバーはステートレスでなければならないということも指定されています。つまり、サーバーは適切に動作するためにそのクライアントについての特別な何かの情報を保持する必要はないということです。サーバーの障害時には、クライアントはサーバーが応答するまで要求を再試行するだけです。この場合マウント操作を繰り返す必要はありません。

10.3.1.5 キャッシュ・ファイル・システム

キャッシュ・ファイル・システム (CacheFS) は、あるファイル・システムを別のファイル・システムのキャッシュに入れるという機能を備えています。CacheFS では、リモート・ディレクトリーをローカル・システムにマウントしてキャッシングを行います。クライアントはマウント済みファイルが必要なときは常に、まず最初にローカル・キャッシュを参照します。要求ファイルがそこにある場合は、そのファイルがローカルにアクセスされます。そうでない場合は、そのファイルは NFS を使ってサーバーから検索されます。ファイルを順次読み取る場合は、今後のアクセスに備えて今後のデータが検索され、キャッシュに入れられます。この手順によってファイル・アクセス速度が増加します。キャッシュをランダム・アクセス・メモリー (RAM) に保管するかディスクに保管するかは、インプリメンテーションによって異なります。データの量がそれほど大きくない場合は、RAM を使用した方がキャッシュへのアクセスは大幅に高速になります。

CacheFS は、キャッシュ・データのデータの正確性の検査、キャッシュの中の不正確なデータの一掃、およびサーバーでの現行データの検索とそれをキャッシュに入れることを定期的に行います。CacheFS は、ファイルが頻繁に変更されない場合に適しています。その他の環境では、CacheFS を使用す

るとサーバーおよびネットワークの負荷が減少し、リンクが遅いクライアントのパフォーマンスが向上します。

10.3.2 NFS バージョン 4

バージョン 4 では NFS の以下の拡張が行われています。

- NFS に伴うマウント・プロトコルの必要性の除去。このために、ルート・ファイル・ハンドル、公用ファイル・ハンドル、および疑似ファイル・システムを使用して、エクスポートされるディレクトリー名におけるギャップを埋めています。
- RPC 層での新規セキュリティ。これは GSS-API に基づいています。
- RFC 2203 で記述されている RPCSEC_GSS のサポート。
- Kerberos 5、LIPKEY、および SPKM-3 セキュリティのサポート。
- 新規の揮発性スタイルのファイル・ハンドル。これはサーバーのインプリメンテーションでファイル・システムの再編成を処理するときに役立ちます。
- パフォーマンスのための複合コマンド (1 操作での検索と読み取り)。
- 指定のクライアントへの、サーバーによる状況とロックの代行。これはキャッシング / パフォーマンス上の理由からです。
- ファイル名での 16/32 ビット文字のサポートによる国際化対応。このために UTF-8 エンコード方式を使用しています。

10.3.3 WebNFS

WebNFS は標準 NFS の拡張バージョンです。WebNFS では、クライアントは広域ネットワーク (WAN) 上のファイルをアクセスすることができます。ローカル・エリア・ネットワーク (LAN) ではトランスポート・プロトコルとして UDP が高速であるため、多くのインプリメンテーションで標準 NFS に UDP が使用されています。しかし広域ネットワークでは TCP 接続の方が大幅に信頼性が高くなります。新しい NFS インプリメンテーションのほとんどは TCP と UDP をサポートしています。WebNFS クライアントは最初に TCP 接続の使用を試みます。それが失敗または拒否された場合、次に UDP 接続を使用します。WebNFS では、ドロップした伝送路からのリカバリーやデータ脱落のリカバリーを容易に行えます。

10.3.3.1 パス名の評価

NFS バージョン 3 は LAN のために設計されています。LOOKUP 要求ごとの時間の合計は大きくありません。このため NFS プロトコルでは一度に 1 つだけの単一パス名要求が許されています。WAN やインターネットを考えると、パス名を要求してそれを評価することは重大な遅延の原因になる可能性があります。このプロセスは非常に不経済です。特にファイルが離れた場所にあ

るいくつかのディレクトリーの場合にそうなります。WebNFS ではパス名全体の検索を処理して、それを評価することができます。

10.3.3.2 NFS URL

この方式では NFS クライアントが、ちょうど他のインターネット・アプリケーションのようにインターネット上にアクセスすることができます。URL 方式は、一般のインターネット方式の構文を基にしています。NFS URL は RFC 2224 に記述されています。NFS URL 方式の一般的な形式は次のようなものです。

```
nfs://<host>:<port><url-path>
```

port はオプションです。これが指定されない場合のデフォルト値は 2049 です。WebNFS サーバーは、公用ファイル・ハンドルと相対のマルチコンポーネント索引を使って、パスを評価します。

10.3.4 参照

次の RFC に NFS についての詳細情報が記述されています。

- RFC 1813 – NFS Version 3 Protocol Specification
- RFC 2054 – WebNFS Client Specification
- RFC 2055 – WebNFS Server Specification
- RFC2203 – RPCSEC_GSS Protocol Specification
- RFC 2224 – NFS URL Scheme
- RFC 3010 – NFS version 4 Protocol

10.4 Andrew File System (AFS)

Andrew File System (AFS) は非 DCE 環境で使用される分散ファイル・システムです。DCE の DFS は AFS を基にしたため、この 2 つのファイル・システムのアーキテクチャーは類似しています。

AFS では情報を複数のサーバーに保管することにより可用性を高めています。またこれは NFS バージョン 3 と違って WAN やインターネット上でも機能するように設計されています。

最新バージョン (AFS 3) には次の属性があります。

単一論理共用ネーム・スペース

すべての AFS ユーザーが同一の様なネーム・スペースを共用します。ファイル名は、ユーザーおよびファイルの両方の物理ロケーションから独立していま

クライアント・キャッシュ

す。クライアントおよびサーバー・マシンのグループはセルと呼ばれます。

データは、ファイル・サーバーに向けられる後続のデータ要求を減らすために、クライアント・マシンのキャッシュに入れます。これによってネットワークとサーバーの負荷を軽減します。サーバーがコールバックによってクライアント・キャッシュをトラッキングし続けるため、サーバーに一定の照会をしてファイルが変更されたかどうかを調べなくてもキャッシュの整合性が保証されます。

RPC

AFS ではネットワーク間の効率的なデータ転送のためにリモート・プロシージャ・コール (RPC) の読み取りと書き込みを使用します。

セキュリティ

Kerberos ベースの認証では、ユーザーはネットワーク・サービスにアクセスする前に自分の身元を証明する必要があります。認証されると、個々のユーザーまたはユーザーのグループがディレクトリー内のファイルに対して操作を実行できるさまざまなレベルの権限が、AFS アクセス制御リスト (ACL) で示されます。

複製

ファイル・システムの信頼性のために複製の手法が使用されます。アプリケーションとデータの複数のコピーを、セル内の複数のファイル・サーバーに複製することができます。クライアントはこの情報にアクセスするときに、選択可能なサーバーから選択します。アクセス頻度の多い情報の複製を複数のサーバーに配置すると、特定のサーバーの負荷を軽減することにもなります。

管理ユーティリティー

バックアップ、再構成、および日常保守はすべてシステム・ダウン時間なしで行われるため、これらの操作の間でもユーザーは引き続きファイルを使用可能です。これはボリューム (関連ファイルのサブセット) のオンライン複製を作成することにより行われます。

AFS コマンドは RPC を基にしています。管理コマンドは、認証済み管理者ならどのクライアント・ワークステーションからでも発行できます。システム・データベースによって、ファイル場所、認証情報、およびアクセス制御リストのトラッキングが行われます。これらのデータベースは複数のサーバーに複製され、情報が変更されるたびに動的に更新されます。

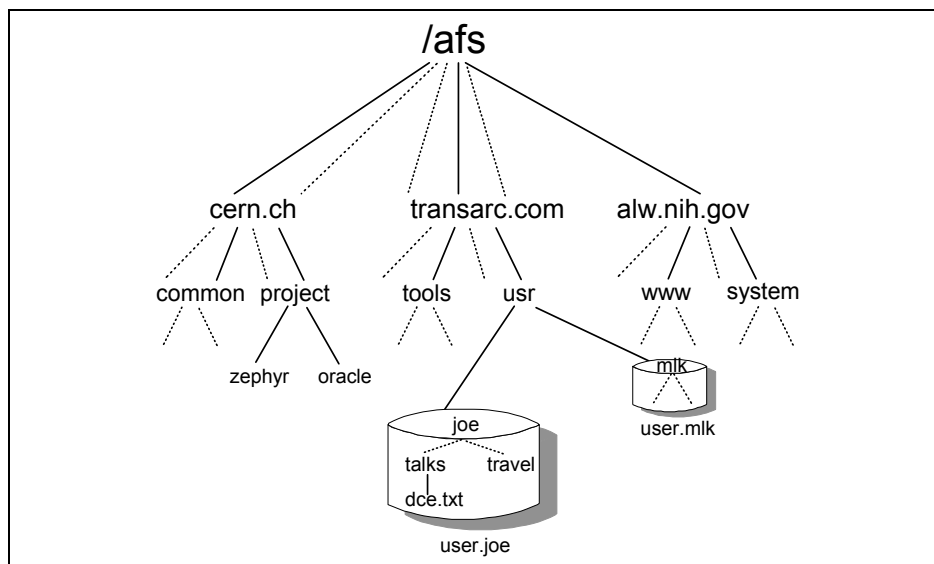


図 168 AFS 3 のネーム・スペース

図 168 は AFS のネーム・スペースの例を示したものです。user.joe はボリュームであり、単一の単位として管理される複数ファイルのグループです。ユーザーはネットワーク内ならどこからでも、次のファイル名を使って Joe の dce.txt ファイルにアクセスすることができます。

/afs/transarc.com/usr/joe/talks/dce.txt

AFS の詳細については、次の Transarc Web サイトで参照することができます。

<http://www.transarc.com/>

第 11 章 メール・アプリケーション

電子メール (E メール) は、おそらく最も広く使用されている TCP/IP アプリケーションです。これは大半の人にとって日常生活で不可欠なものになりました。この章では電子メールを扱う TCP/IP アプリケーション・プロトコルの概要を解説します。

11.1 Simple Mail Transfer Protocol (SMTP)

基本的なインターネット・メール・プロトコルを使って TCP/IP ホスト間でメール (メモ) やメッセージの交換を行うことができます。7 ビット ASCII テキストで表すことができないデータの転送のためにいくつかの機能が追加されています。この種のメールに適用される標準プロトコルは 3 つあります。それぞれは「推奨 (Recommended)」です。なお Simple Mail Transfer Protocol (SMTP) という用語は、多くの場合プロトコルが複合したものを指して使用します。これはそれらのプロトコルが非常に密接に相互に関連しているためですが、厳密に言えば SMTP はその 3 つの中の 1 つにすぎません。3 つのプロトコルの中のどれを指しているかは、通常はコンテキストから明らかです。疑問があるときは STD または RFC 番号を参照して、あいまいさを解消するようにしてください。3 つの標準とは次のものです。

- 2 コンピューター間のメールの交換のための標準 (STD 10/RFC 821)。TCP/IP ホスト間のメールの送信に使用するプロトコルを規定しています。この標準が SMTP そのものです。
- メール・メッセージのフォーマットに関する標準 (STD 11)。これは 2 つの RFC に含まれています。RFC 822 にはメールのヘッダー・フィールドの構文が記述されており、ヘッダー・フィールドとその解釈がセットで定義されています。RFC 1049 にはプレーン・テキスト ASCII 以外の一連の文書タイプをメール本体で使用方法が記述されています (文書自体は 7 ビット ASCII であり、組み込みフォーマット設定情報を含みます。PostScript、Scribe、SGML、TEX、TROFF、および DVI がすべてこの標準で示されています)。

この標準の正式なプロトコル名は MAIL です。

- Domain Name System を使用したメールのルーティングのための標準 (RFC 974 に記述)。この標準の正式なプロトコル名は DNS-MX です。

STD 10/RFC 821 では、SMTP によって送信されるデータは 7 ビット ASCII データで、その高位ビットはゼロにクリアされたものと規定されています。これは英語のテキスト・メッセージの伝送でのインスタンスの大部分には適切ですが、英語以外のテキストやテキスト以外のデータにはこれは適切ではありません。これらの制限を解消する方法として次の 2 つがあります。

- Multipurpose Internet Mail Extensions (MIME)。RFC 2045 ~ 2049 で定義されており、テキストとバイナリー・データを RFC 822 で定義されているメール・エンベロープ内に 7 ビット ASCII としてエンコードするためのメカニズムを規定しています。MIME は、417 ページの 11.2『Multipurpose Internet Mail Extensions (MIME)』で説明しています。
- SMTP サービス拡張機能。RFC 821 で規定されている制限を超えて SMTP の機能を拡張するためのメカニズムを定義しています。SMTP サービス拡張機能が記述されている RFC として、現在次の 3 つがあります。

- 受信側 SMTP が、自身がサポートしているサービス拡張機能を送信側 SMTP に通知するための標準 (RFC 1869)。

RFC 1869 は RFC 821 を変更したもので、SMTP セッションの開始時にクライアント SMTP エージェントが、サーバーにそのサーバーがサポートしているサービス拡張機能のリストを応答するように要求することができます。サーバーの SMTP が RFC 1869 をサポートしていない場合、そのサーバーはエラーで応答します。この場合クライアントはそのセッションを終了するか、または RFC 821 の規則に従ってセッションの開始を試みることができます。サーバーが RFC 1869 をサポートしている場合、サーバーは自身がサポートしているサービス拡張機能のリストも応答します。サービスのレジストリーは IANA によって維持されます。RFC 1869 で定義されている最初のリストには、RFC 1123 – Requirements for Internet Hosts – Application and Support で SMTP サーバーのオプションとしてリストされているコマンドが含まれていません。

他のサービス拡張機能は RFC で通常の方法で定義されています。次の 2 つの RFC は特定の拡張機能を定義しています。

- 8 ビット・テキスト伝送のためのプロトコル (RFC 1652)。SMTP サーバーはこれを使って 8 ビット・バイトから成るデータを受け入れ可能であることを示すことができます。この拡張機能が使用可能であることをクライアントに報告するサーバーは、SMTP メッセージで受信したバイトの高位ビットを変更しないで、そのままにしておかなければなりません (クライアントがそれを要求した場合)。

MIME および SMTP サービス拡張の方法は、競合する標準ではなく補完し合うものです。たとえば RFC 1652 のタイトルは SMTP Service Extension for 8-bit-MIMEtransport となっていますが、これは MIME 標準ではメッセージを 7 ビット・データではなく 8 ビット・データから成るものとして宣言できるためです。そのようなメッセージは RFC 821 に厳密に準拠している SMTP エージェントでは伝送できませんが、クライアントとサーバーの両方が RFC 1869 および 1652 に準拠しているときは伝送することができます。この拡張機能をサポートしていないサーバーに対してクライアント SMTP が 8 ビット・データを送信しようとするときは常に、クライアント SMTP はメッセージの内容を

MIME 標準に準拠した 7 ビット表記にエンコードするか、またはユーザーに永続エラーを戻さなければなりません。

このサービス拡張機能で任意のバイナリー・データを送信できるわけではありません。RFC 821 で SMTP サーバーが受け入れる必要がある行の最大長が 1000 文字に定義されているためです。テキスト以外のデータでは <CRLF> シーケンスなしで 1000 文字を超えて連続する場合がよくあります。

注：

サービス拡張では非 ASCII 文字 (10 進で 127 より大きい値のもの) の使用を特にメッセージ本体に制限しています。それらを RFC 822 メッセージ・ヘッダーで使用することはできません。

- メッセージ・サイズの宣言のためのプロトコル (RFC 1870)。これによりサーバーは自身が受け入れることができる最大サイズのメッセージをクライアントに通知することができます。この拡張機能がない場合、クライアントがメッセージ全体を送信した後で、そのメッセージはサーバーが受け入れ可能な最大サイズ (固定的上限またはサーバーでの使用可能なストレージ・スペースの不足による一時的制限) を超えていたということが、そのクライアントに通知されるだけです。これが起きると、サーバーは失敗メッセージを廃棄します。クライアントとサーバーの両方がメッセージ・サイズ宣言の拡張機能をサポートしている場合は、クライアントは転送するメッセージの見積サイズを宣言することができ、サーバーはそのメッセージが大きすぎるときはエラーを戻します。

これらの SMTP サービス拡張機能はそれぞれドラフト標準のプロトコルで、状態はいずれも「任意選択 (Elective)」です。

11.1.1 SMTP の動作

SMTP (つまり STD 11/RFC 821) はエンドツーエンド送達に基づいており、SMTP クライアントは宛先ホストの SMTP サーバーに接触してメールを直接送信します。このとき SMTP クライアントはメール項目を、それが宛先の SMTP に正常にコピーされるまで伝送し続けます。このことは多くのメーリング・システムで一般的なストア・アンド・フォワードの原則と異なる点です。ストア・アンド・フォワードでは、メール項目は宛先に到達するまで同一ネットワーク内の多数の中間ホストを介して渡されていきます。また送信側からの伝送が正常な場合、メール項目が最初の中間ホップに到達したただけが示されます。

さまざまなインプリメンテーションで、TCP/IP SMTP メーリング・システムとローカルに使用されているメーリング・システムの間でメールを交換する

ことが考えられます。これらのアプリケーションをメール・ゲートウェイまたはメール・ブリッジと呼びます。メール・ゲートウェイを介してメールを送信することは、エンドツーエンド送達の仕様を変えることになります。つまりこの場合の SMTP では TCP/IP ネットワークを越えたところに置かれている実際の宛先ホストまでの送達ではなく、メール・ゲートウェイ・ホストまでの送達だけを保証することになります。メール・ゲートウェイが使用されときの SMTP エンドツーエンド伝送は、ホストからゲートウェイ、ゲートウェイからホスト、またはゲートウェイからゲートウェイになります。ゲートウェイを越えたところでの動作は SMTP では定義されていません。CSNET にメール・ゲートウェイ・サービスの興味深い例があります。科学技術の研究所と会社の研究所を相互接続するための低コストのファシリティーとして始まった CSNET では、サブスクライバーがダイヤルアップ・モデムだけを使ってインターネット上でメールの送信と受信を行えるメール・ゲートウェイ・サービスを運用しています。メール・ゲートウェイでは定期的にサブスクライバーをポーリングし、そこに宛てられたメールを送達して、発信メールをピックアップします。これは直接のエンドツーエンド送達ではありませんが、非常に有用なシステムであることが立証されています。

各メッセージには次のものがあります。

- ヘッダー (つまりエンベロープ)。この構造は RFC 822 で厳密に定義されています。

メール・ヘッダーはヌル行 (<CRLF> シーケンスの前に何も無い行) で終了します。ただしインプリメンテーションによっては (たとえばファイル内に長さがゼロのレコードをサポートしていない VM)、これを別に解釈することがあるため、ターミネーターとしてブランク行を受け入れるものもあります。

- 内容

ヌル (ブランク) 行の後ろはすべてメッセージ本体です。これは ASCII 文字 (10 進で 128 より小さい値の文字) を含む行が連続したものです。

RFC 821 はクライアント / サーバー・プロトコルを定義しています。ここでもクライアント SMTP はセッション (つまり送信 SMTP) を開始するものであり、サーバーはセッション要求に応答するもの (受信 SMTP) です。ただしクライアント SMTP はユーザーのメーリング・プログラムに対してしばしばサーバーとして動作するため、多くの場合クライアントを送信側 SMTP、サーバーを受信側 SMTP と呼ぶ方が簡単になります。

11.1.1.1 メール・ヘッダーのフォーマット

メッセージ・ヘッダーは SMTP だけで扱われるため、ユーザーは通常はメッセージ・ヘッダーを気にする必要はありません。念のために以下に簡単に解説します。

RFC 822 にはメール・ヘッダーの字句解析について詳細に記述されています。そこでの構文は増補バックス正規形式 (BNF) と呼ばれる形式で書かれています。RFC 822 には増補 BNF の説明があり、RFC 822 に関連した多数の RFC がこのフォーマットを使用しています。RFC 822 には、継続行の展開、意味のないスペースの削除、コメントの除去などを行って、メール・ヘッダーを解析して *正規表記* にする方法が説明されています。この構文は強力ですが、解析するには比較的扱いにくい構文です。ここで基本的な説明をしますが、これらは読者が目にする単純なメール・ヘッダーの意味を解釈するには十分なものです。ただしここでの非常に単純化した説明では RFC 822 メール配布機能の詳細な動作の理解は困難です。詳細な説明は RFC 822 を参照してください。

簡単に言うとヘッダーは次の形式の行のリストです。

```
field-name: field-value
```

フィールドは列 1 から始まります。空白文字 (スペースまたはタブ) で始まる行は継続行であり、これは展開されてフィールドごとに正規表記での単一行になります。ASCII の引用符で囲まれたストリングは単一のトークンを示し、この内部ではコロンなどの特殊文字は無効であることを示します。重要なフィールド値 (To や From フィールドのフィールド値など) の多くはメールボックスです。これらの最も一般的な形式は次のようなものです。

```
octopus@garden.under.the.sea
The Octopus <octopus@garden.under.the.sea>
"The Octopus" <octopus@garden.under.the.sea>
```

ストリング The Octopus とありますが、宛先であり、これはメールボックス所有者の名前です。ストリング octopus@garden.under.the.sea はメールボックスの機械可読アドレスです。(不等号括弧がアドレスの区切りに使用されていますが、これはアドレスの一部ではありません。) この形式のアドレッシングは Domain Name System の概念に密接に関連していることが分かります。実際にクライアント SMTP は Domain Name System を使って宛先メールボックスの IP アドレスを判別します。

よく使用されるフィールドを次の表に示します。

表 18 SMTP - メール・ヘッダーのよく使用されるフィールド

キーワード	値
to	メッセージの 1 次宛先。
cc	メッセージの 2 次 (カーボン・コピー) 宛先。
from	差し出し人の識別。

キーワード	値
reply-to	応答の送信先のメールアドレス。このフィールドは発信元で追加されます。
return-path	発信元のアドレスとそこまでの経路。このフィールドはメールを送達する最終トランスポート・システムで追加されます。
Subject	メッセージの要約。通常これはユーザーによって設定されます。

11.1.1.2 メール交換

SMTP の設計は図 169 に示す通信のモデルに基づいています。ユーザーのメール要求の結果、送信側 SMTP が受信側 SMTP と両方向の接続を確立します。受信側 SMTP は最終宛先か中間のもの（メール・ゲートウェイ）になります。送信側 SMTP が受信側 SMTP によって応答されるコマンドを生成します。

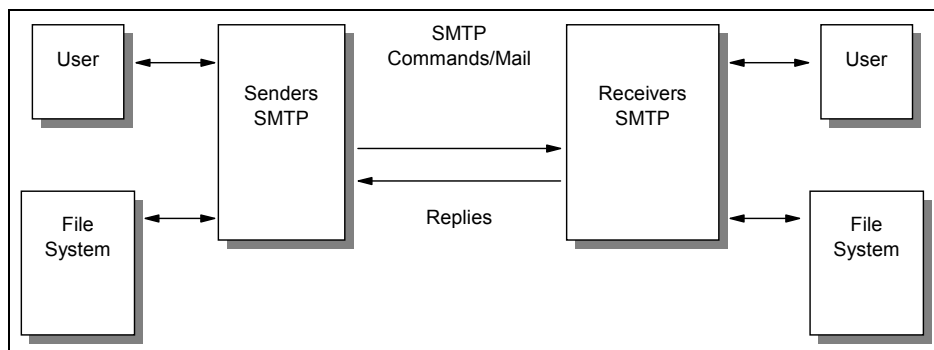


図 169 SMTP - SMTP のモデル

SMTP メール・トランザクション・フロー

メールのコマンドと応答は厳密に定義されていますが、その交換は 412 ページの図 170 で容易に理解できます。交換されるコマンド / 応答 / データはすべてテキスト行であり、<CRLF> で区切られます。応答にはすべてその行の先頭に数字コードがあります。このフローのステップは以下になります。

1. 送信側 SMTP が宛先 SMTP と TCP 接続を確立し、次にサーバーが 220 Service ready (サービス作動可能) メッセージ、または宛先が一時的に処理できないときの 421 Service not available (サービス利用不能) メッセージを待ちます。
2. HELO (HELO は hello の省略形) が送信され、これに対し受信側は自身のドメイン・ネームを送り返すことによって受信側の識別を示します。送信側 SMTP はこれを使って、自身が正しい宛先 SMTP に接触したかどうかを確認することができます。

送信側 SMTP が RFC 1869 で定義されている SMTP サービス拡張機能をサポートしている場合、送信側 SMTP は HELO コマンドの代わりに EHLO コマンドを使用することができます。サービス拡張機能をサポートしていない受信側 SMTP は、500 Syntax error, command unrecognized (構文エラー、コマンド認識不能) メッセージで応答します。それを受けて送信側 SMTP は HELO を使って再試行するか、または送信側が 1 つ以上のサービス拡張機能を使用しないとメッセージを伝送できない場合は QUIT メッセージを送信します。

受信側 SMTP はサービス拡張機能をサポートしている場合は、複数行の 250 OK メッセージで応答します。これには受信側 SMTP がサポートしているサービス拡張機能のリストが含まれています。

3. 次に送信側は受信側に MAIL コマンドを送信することによってメール・トランザクションを開始します。このコマンドにはエラーを報告するときに使用できる逆パスが含まれています。パスは user mailbox@host domain name のペアだけではない場合がありますので注意してください。これに加えてルーティング・ホストのリストが含まれることがあります。メール・ブリッジを通るときや宛先アドレスにルーティング情報を明示するときがその例です。受信側はこのコマンドを受け入れると 250 OK で応答します。
4. 実際のメール交換の 2 番目のステップは、サーバー SMTP にメッセージの宛先を指定することです。宛先が複数の場合があります。これは 1 つ以上の RCPTTO:<forward-path> コマンドを送信することにより行われます。サーバーで宛先が分かる場合はコマンドごとに応答 250 OK を受信しますが、そうでない場合は 550 No such user here (ユーザー不明) を受信します。
5. 送信側は RCPT コマンドをすべて送信すると、DATA コマンドを発行して、この後にメッセージの内容が続くことを受信側に通知します。サーバーは 354 Start mail input (メール入力開始) で応答し、<CRLF>.<CRLF> で終了します。送信側がメッセージ・データを終了するときに使用する終了シーケンスに注意してください。
6. 次にクライアントはデータを 1 行ずつ送信し、5 文字シーケンス <CRLF>.<CRLF> の行で終了します。これに対して受信側は 250 OK で応答するか、または何か失敗したときは該当するエラー・メッセージで応答します。
7. ここで次のアクションとして以下のことが考えられます。
 - 送信するメッセージが送信側にもう残っていない。ユーザーは QUIT コマンドを使ってその接続を終了します。これに対して 221 Service closing transmission channel (サービスは伝送チャネルを終了) 応答が返されます。
 - 送信するメッセージが送信側にもう残っていないが、他のサイドからのメッセージがあればそれを受信できる。ユーザーは TURN コマンドを

発行します。それによって 2 つの SMTP が送信側 / 受信側の役割を切り替え、今度は送信側 (それまでは受信側) が上記のステップ 3 から開始してメッセージを送信します。

- 送信する別のメッセージが送信側にある。この場合は単にステップ 3 に戻って新たに MAIL コマンドを送信します。

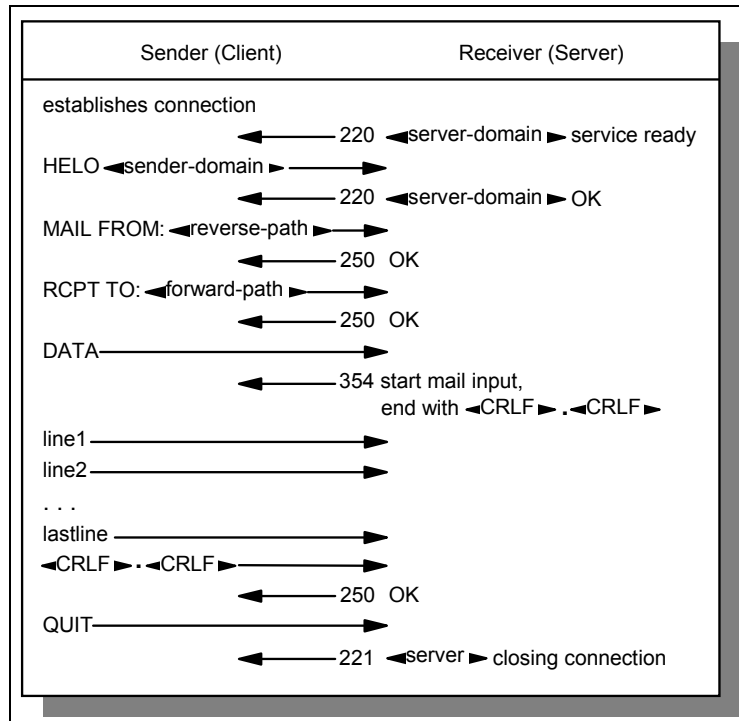


図 170 SMTP - 通常の SMTP データ・フロー - 1 宛先メールアドレスへの 1 メール・メッセージ

SMTP 宛先アドレス (メールボックス・アドレス)

この一般的な形式は local-part@domain-name ですが、次のような形式にすることもあります。

user@host 同一の TCP/IP ネットワーク上の直接の宛先の場合。

user%remote-host@gateway-host

メール・ゲートウェイのゲートウェイ・ホスト経由する、非 SMTP の宛先リモート・ホスト上のユーザーの場合。

@host-a,@host-b:user@host-c

中継されるメッセージの場合。これにはルーティング情報が明示的に含まれます。メッセージはまず最初に host-a に送達され、そのメッセージはここで host-b 宛てに再送 (中継) されます。次に host-b はそのメッセージを実際の

宛先 host-c に転送します。メッセージは中間にある各ホストに保管されるため、この場合はエンドツーエンド送達にはならないことに注意してください。

上記の説明では最も重要なコマンドだけを取り上げました。これらはすべて各 SMTP インプリメンテーションで認識されなければならないコマンドです。その他のコマンドもありますが、それらの大部分はオプションです。つまり RFC 標準では、それらをすべてにインプリメントすることを要求しているわけではありません。ただし、中継、転送、メーリング・リストなど、非常に面白い機能がインプリメントされています。

コマンドの詳細なリストは、RFC 821 Simple Mail Transfer Protocol および RFC 1123 Requirements for Internet Hosts – Application and Support を参照してください。SMTP サービス拡張機能の詳細については、RFC 1869 SMTP Service Extensions、RFC 1652 SMTP Service Extension for 8-bit-MIMEtransport、および RFC 1870 SMTP Service Extension for Message Size Declaration を参照してください。

例

次のシナリオでは、ホスト vm1.stockholm.ibm.com のユーザー abc が、ホスト delta.aus.edu のユーザー xyz、opq、rst にメモを送信します。先頭が R: の行は受信側によって送信される行、先頭が S: の行は送信側によって送信される行です。

```
R: 220 delta.aus.edu Simple Mail Transfer Service Ready
S: HELO stockholm.ibm.com
R: 250 delta.aus.edu
S: MAIL FROM:<abc@stockholm.ibm.com>
R: 250 OK
S: RCPT TO:<xyz@delta.aus.edu>
R: 250 OK
S: RCPT TO:<opq@delta.aus.edu>
R: 550 No such user here
S: RCPT TO:<rst@delta.aus.edu>
R: 250 OK
S: DATA
R: 354 Start mail input, end with <CRLF>.<CRLF>
S: Date: 23 Jan 89 18:05:23
S: From: Alex B. Carver <abc@stockholm.ibm.com>
S: Subject: Important meeting
S: To: <xyz@delta.aus.edu>
S: To: <opq@delta.aus.edu>
S: cc: <rst@delta.aus.edu>
S:
S: Blah blah blah
S: etc.....
S: .
R: 250 OK
S: QUIT
R: 221 delta.aus.edu Service closing transmission channel
```

図 171 SMTP - シナリオの例

メッセージ・ヘッダーは、伝送されるデータの一部であることに注意してください。

11.1.2 SMTP と Domain Name System

ドメインの概念が使用されているネットワークでは、TEST.IBM.COM 宛てに送信するメールは、SMTP が単に TEST.IBM.COM との TCP 接続を開くだけでは送達できません。この場合 SMTP はまずネーム・サーバーに照会して、メッセージの送達先のホスト (ドメイン・ネーム) を検索します。

ネーム・サーバーはメッセージの送達のために MX RR と呼ばれるリソース・レコード (RR) を保管しています。そこではドメイン・ネームが次の 2 つの値にマップされています。

- 優先設定値。同じドメイン・ネームに対して複数の MX リソース・レコードが存在できるため、それらに優先設定 (優先順位) が割り当てられます。

最小の優先設定値のものが最も優先されるレコードになります。これは最も優先されるホストに到達できないときに有用です。この場合、送信 SMTP は次の (次に優先される) ホストへの接触を試みます。

- ホスト名

ネーム・サーバーが MX RR の空のリストで応答することもあります。これはドメイン・ネームはネーム・サーバーの権限内にあるが、ドメイン・ネームに MX が割り当てられていないことを意味します。この場合、送信 SMTP はホスト名そのものを使って接続の確立を試みることができます。

RFC 974 に重要な推奨事項が記載されています。そこでは、送信 SMTP は MX レコードを取得したらそのホストのウェルノウン・サービス (WKS) レコードを照会して、参照したホストの WKS 項目として SMTP があることを検査するように推奨しています。

注

これはプロトコルのオプションにすぎませんが、すでに広くインプリメントされています。

次に MX リソース・レコードの例を示します。

```
fsc5.stn.mlv.fr.      IN      MX 0   fsc5.stn.mlv.fr.
                      IN      MX 2   psfred.stn.mlv.fr.
                      IN      MX 4   mvs.stn.mlv.fr.
                      IN      WKS    152.9.250.150 TCP (SMTP)
```

上記の例では、メールは優先設定により fsc5.stn.mlv.fr 宛てのメールがホストそのものに送達されますが、そのホストが到達不能の場合はそのメールは psfred.stn.mlv.fr または to mvs.stn.mlv.fr (psfred.stn.mlv.fr にも送達不能の場合) にも送達されることになります。

11.1.2.1 サーバー・システム上のメールボックスのアドレッシング

あるユーザーがすべてのメール機能にサーバー・システムを使用する場合、他の SMTP ユーザーに見えるメールボックス・アドレスはそのメール・サーバー・システムだけを指します。たとえば 2 つの OS/2 システムの名前が、

hayes.itso.ral.ibm.com

と

itsol80.itso.ral.ibm.com

で、最初のは UltiMail クライアントとして使用され、2 番目のものが UltiMail サーバーとして使用される場合、メールボックス・アドレスは次のようになります。

```
hayes@itsol180.itso.ral.ibm.com
```

このメールボックス・アドレスは、すべての発信メールの From: ヘッダー・フィールドと、UltiMail サーバー・システムによってリモート・サーバーに対して発行される SMTP コマンドの中に現れます。

ただしユーザーが POP サーバーを使用するときは、アウトバウンド・メールの項目にあるメールボックス・アドレスにはワークステーションのホスト名 (たとえば `steve@hayes.itso.ral.ibm.com`) が含まれます。この場合、送信側はメール・ヘッダーに Reply-To: フィールドを組み込んで、応答をその発信メールボックス宛てに送信してはならないことを示す必要があります。たとえばメール・ヘッダーは次のようになります。

```
Date: Fri, 10 Feb 95 15:38:23
From: steve@hayes.itso.ral.ibm.com
To: "Steve Hayes" <tsgsh@gford1.warwick.uk.ibm.com>
Reply-To: hayes@itsol180.itso.ral.ibm.com
Subject: Test Reply-To: header field
```

受信メール・エージェントは、応答を From: アドレスではなく Reply-To: アドレスに送信するようになっています。

メールの送信のための Domain Name System の使用

Reply-To: ヘッダー・フィールドを使う代わりに Domain Name System を使って、メールを正しいメールボックスに送信する方法があります。ドメイン・ネーム・サーバーの管理者 (ユーザーのワークステーションとネーム・サーバーを含むドメインの権限を持つ) は、メールを適切に送信できるように MX リソース・レコードを Domain Name System に追加することができます (414 ページの 11.1.2 『SMTP と Domain Name System』で説明)。たとえば次の MX レコードは、`hayes.itso.ral.ibm.com` の SMTP サーバーが利用不能のときは、代わりに使用できるメール・サーバーが `itsol180.ral.ibm.com` (9.24.104.180) にあることをクライアント SMTP に示しています。

```
itsol180.itso.ral.ibm.com.  IN      WKS    9.24.104.180 TCP (SMTP)
hayes.itso.ral.ibm.com.   IN      MX     0 hayes.itso.ral.ibm.com.
                           IN      MX     1 itsol180.itso.ral.ibm.com.
```

11.1.3 参照

SMTP、メール、および DNS-MX 標準の詳細については、次の RFC に記述されています。

- RFC 821 – Simple Mail Transfer Protocol

- RFC 822 – Standard for the format of ARPA Internet text messages
- RFC 974 – Mail Routing and the Domain System
- RFC 1049 – A Content Type Header Field for Internet messages
- RFC 1652 – SMTP Service Extension for 8-bit-MIMEtransport

11.2 Multipurpose Internet Mail Extensions (MIME)

MIME はドラフト標準のプロトコルです。その状態は「任意選択 (elective)」になっています。

電子メール (405 ページの 11.1 『Simple Mail Transfer Protocol (SMTP)』で説明) は、おそらく最も広く使用されている TCP/IP アプリケーションです。しかし SMTP (つまり STD 10/RFC 821 に準拠したメーリング・システム) は 7 ビット ASCII テキストで、行の最大長が 1000 文字に制限されており、このままでは以下のような多くの制約があります。

- SMTP では実行可能ファイルや他のバイナリー・オブジェクトを伝送できない。SMTP のメール項目の中には、バイナリー項目をカプセル化するという特別なメソッドが多数あります。たとえば次のようなものがあります。
 - ファイルを純 16 進数としてエンコードする。
 - UUCP メーリング・システムでバイナリー・データをエンコードするために使用される、UNIX の uuencode および uudecode ユーティリティ。これは 7 ビットのトランスポートという同じ制限を解消するためのものです。
 - Andrew Toolkit 表記

これらのいずれもデファクト・スタンダードとは言えません。インターネットにおける UNIX システムの先駆的な役割によって、おそらく uuencode が最も普及しています。

- SMTP では各国語文字を含むテキスト・データを伝送できない。これはそれらの文字が、ASCII を基にしたすべての文字セットで 128 (10 進) 以上の値を持つコード・ポイントで表記されるためです。
- SMTP サーバーは一定のサイズを超えるメール・メッセージをリジェクトすることがある。どのサーバーでも、クライアントからのメール・データの受け入れ可能な最大量について、永続的または一時的な (あるいはその両方の) 制限を、どの時点でも持つ可能性があります。
- ASCII から EBCDIC に変換する、およびその逆にも変換する SMTP ゲートウェイでは、コード・ページのマッピングに一貫性があるものを使用しないため、変換の問題が起こることになる。

- SMTP インプリメンテーションによっては、またインターネットでの他のメール・トランスポート・エージェント (MTA) は、RFC 821 で定義されている SMTP 標準を完全に順守しているわけではない。一般的には次の問題があります。
 - 末尾の空白文字 (タブとスペース) の除去。
 - メッセージ内のすべての行を同じ長さにするための埋め込み。
 - 76 文字を超える行の折り返し。
 - 異なる規則間での改行シーケンスの変更。(たとえば <CR> 文字が <CRLF> シーケンスに変換されることがあります。)
 - 複数スペースへのタブ文字の変換。

MIME はこれらの問題を、既存の RFC 822 標準と高い互換性を保ちながら解決するメカニズムを組み込んだ標準です。メール・メッセージはメール・ゲートウェイを介して頻繁に転送されるため、宛先メールボックスを管理するサーバーと別のネットワークへのゲートウェイとして機能するサーバーを SMTP クライアントが区別するのは不可能です。ゲートウェイを通るメールはさらに複数のゲートウェイによってトンネリングが行われることがあり、それらのゲートウェイの一部またはすべてがメッセージング・プロトコルとして別のものを使用していることがあります。そのため送信 SMTP が宛先メールボックスまでの経路のすべてのステージに共通する最小公分母的な機能を判別することは一般に不可能です。このため MIME では最悪のケースとして、RFC 821 に厳密には準拠していないまたは互換性のない 7 ビット ASCII のトランスポートを想定しています。これは RFC 821 の拡張はまったく定義していませんが、RFC 822 のフレームワークの範囲内の拡張にとどめています。したがって MIME メッセージに対して、RFC 821 に緩やかに準拠しているネットワークや、RFC 821 のメッセージを送送できるネットワークを、いくつでも経路指定できます。

MIME はドラフト標準のプロトコルで、その状態は「任意選択 (Elective)」です。これは次の 5 つのパートで記述されています。

- US ASCII テキストのメール・メッセージ以外のオブジェクトを RFC 822 に準拠しているメッセージの本体に組み込むための各プロトコル。これらは RFC 2045 に記述されています。
- MIME メディア・タイプのシステムの一般的な構造。ここでメディア・タイプの初期設定を定義しています。これは RFC 2046 に記述されています。
- US ASCII 以外のテキストを RFC 822 に準拠しているメール・メッセージのヘッダー・フィールドにエンコードするためのプロトコル。これは RFC 2047 に記述されています。

- MIME 関連のファシリティーのためのさまざまな IANA 登録手順。これは RFC 2048 に記述されています。
- MIME 規格合致基準。これは RFC 2049 に記述されています。

RFC 2045 は X.400 メッセージの非テキスト・データを RFC 822 と互換性がある形式で記述するのに適したメカニズムを示していますが、X.400 メッセージの部分を MIME メッセージの部分にマップする方法に言及していません。X.400 と MIME 間の変換が RFC 1494、2156、および 1496 で定義されており、これらによって RFC 822 と X.400 間の変換のプロトコルがアップデートされています。

MIME 標準は以下の汎用的な優先順位で設計されました。

1. RFC 822 などの既存の標準との互換性。

それまでの標準との互換性が完全ではない領域として次の 2 つがあります。

- a. RFC 1049 (STD 11 の一部) で、メッセージ本体の (ASCII テキスト) データのタイプを示すために使用する Content-Type: フィールドを記述していました。PostScript や SGML ではそれらに応じてユーザーのメール・エージェントがこのフィールドを処理することが可能でした。MIME ではこのフィールドを残していますが、そこに定義される値を変更しています。メール・エージェントがこのフィールドで不明な値を見つけたときの正しい応答は基本的にはそれを無視することであるため、このことは互換性に関する重要な問題にはなりません。
- b. RFC 934 はメッセージ転送のコンテキストにおけるメッセージのカプセル化を検討し、カプセル化の境界としてカプセル化されたメッセージの先頭と終わりを示す行を定義しました。MIME は RFC 934 との広い互換性を保持していますが、RFC 934 ではカプセル化されたメッセージ内の行に使用された引用メカニズムは、MIME では境界として正しく解釈されない可能性があるため組み込まれていません。¹

互換性で最も重要なことは、MIME メッセージの標準形が RFC 821 に準拠したメール・リーダーで読み取れることです。これはもちろん、そのようになっています。具体的には、MIME メッセージ本体のデフォルトのエンコード方式は、RFC 822 と同じようにまったくエンコードを行いません。

2. 既存のものに対する堅固性。上記のように、STD 10/RFC 821 に準拠していない MTA がインターネットに広く多数配置されています。RFC 2045 で規定されているエンコード・メカニズムは、それらに最も共通すること (行を 76 文字の短さに折り畳むことと末尾の空白文字を除去すること) を

¹ この変更の理由は、MIME ではカプセル化のネストを深くすることが可能ですが、テキスト行を元に戻せるように列 76 以前であふれる方法でテキストをエンコードして、非準拠 SMTP エージェントによって元に戻せない行があふれることになるのを回避するためです。RFC 934 の引用メカニズムではカプセル化のレベルごとに行が長くなるため、列 76 を超えることがあります。

常に回避できるように設計されています。そのために末尾の空白文字がない短い行だけを伝送し、メール・セーフな方法での任意のデータのエンコードを可能にしています。

注：

MIME ではメール項目のエンコードを必須にしていません。これはユーザーまたはメール・プログラム（あるいはその両方）の判断に任せられています。SMTP (7 ビット) 間で伝送されるバイナリー・データの場合は常にエンコードが必要ですが、大部分がテキストから成るデータの場合は必ずしも必要ありません。

大部分がテキストのデータの場合に望ましいエンコード・メカニズムは、最低限のレベルとしては ASCII システムでのどの SMTP 準拠エージェントでもメール・セーフなもの、最大限のレベルとしてはすべての既知のゲートウェイと MTA でメール・セーフなものです。MIME が最大限のエンコードを必須としていないのは、メールが MIME 準拠システム以外に伝送されるとそのエンコードのために判読が困難になるためです。

3. 拡張の容易性。RFC 2045 は メール本体のエレメントを 7 つのコンテンツ・タイプにカテゴリー化しており、そこにはサブタイプがあります。そしてコンテンツ・タイプ / サブタイプのペアには、関係するオブジェクトをさらに記述するパラメーターがあります。この RFC ではこれらや他の MIME フィールドについての新しい値を Internet Assigned Numbers Authority (IANA) に登録するためのメカニズムを定義しています。このプロセス自体は RFC 2048 によって更新されています。

MIME のすべての値についての現在のリストは、STD 2 – Assigned Internet Numbers を参照してください。この章の残りの部分では、RFC 2045 に記載されている値とタイプだけを説明します。

このアプローチの 1 つの結果は、RFC 2045 を引用すると、「[MIME で使用される] メカニズムのいくつかは、最初はやや奇妙にあるいは異様にさえ思えるかもしれませんが。具体的には、洗練された方法よりも常に互換性の方が優先されています。』

RFC 822 はメッセージ・ヘッダーの構文を定義して（さらにそこに記述されている一連のヘッダーの追加ができるように考慮して）いますが、メッセージ本体の構成は定義していません。そのため MIME 標準は大部分で RFC 822 と互換性があります。具体的にはメッセージ本体の構造およびその構造の記述に使用される一連のヘッダー・フィールドを定義している RFC 2045 のパートと互換性があります。

MIME は上位プロトコルとして見ることができます。つまり、MIME は完全に STD 10 および STD 11 の範囲内で機能するため、プロトコル・スタックのトランスポート層 (下位層) をまったく必要としません。

11.2.1 MIME の動作

MIME 準拠のメッセージは、次のテキストと同じテキストを持つヘッダー・フィールドを含んでいなければなりません。

```
MIME-Version: 1.0
```

MIME ヘッダー・フィールド名の太文字小文字は RFC 822 ヘッダーの場合と同じように意味を持ちませんが、フィールド値の太文字小文字はフィールド名とコンテキストによって意味を持つ場合があります。下記の MIME フィールドの値は、特に明記しない限り太文字小文字を区別しません。

MIME ヘッダー・フィールドの一般的な構文は RFC 822 のものと同じです。したがって次のフィールドは、括弧付きの句はコメントとして扱われて無視されるため、有効です。

```
MIME-Version: 1.0 (this is a comment)
```

次の 5 つのヘッダー・フィールドが MIME 用に定義されています。

MIME-Version 前記のように、この値は 1.0 でなければなりません。

Content-Type 本体内のオブジェクトを解釈する方法をここに記述します。デフォルト値は text/plain; charset=us-ascii です。これは不定形式 7 ビット ASCII テキスト・データ (RFC 822 の定義によるメッセージ本体) を示します。

Content-Transfer-Encoding オブジェクトをメール・セーフな形式でメッセージに組み込むために本体内のオブジェクトをエンコードした方法をここに記述します。

Content-Description 本体内のオブジェクトについてのプレーン・テキストでの記述。これは読めないオブジェクトのときに役立ちます (たとえば音声データ)。

Content-ID このメッセージのこの部分の内容を指定する世界固有の値。

これらのフィールドの最初の 2 つを、続くいくつかのセクションでさらに詳細に説明します。

11.2.2 Content-Type フィールド

次の形式の Content-Type フィールドにメッセージの本体が記述されます。

```
Content-Type: type/subtype ;parameter=value ;parameter=value
```

使用できるパラメーターは、タイプとサブタイプによって異なります。type/subtype のペアには、パラメーターがないもの、オプションのパラメーターがあるもの、必須パラメーターがあるもの、およびオプションと必須の両方のパラメーターがあるものがあります。subtype パラメーターを省略することはできません。ただしこのフィールド全体の省略は可能であり、その場合のデフォルト値は text/plain です。

標準内容タイプとして次の 7 つがあります。

1. text

次の中から単一のサブタイプを定義します。

- a. plain: 不定形式テキスト。テキストの文字セットは charset パラメーターで指定できます。次の値を使用可能です。
- b. us-ascii: テキストは 0 ~ 127 (10 進) の範囲の ASCII 文字から成ります。これはデフォルトです (RFC 822 との互換性のため)。
- c. iso-8859-x: ここで x は ISO-8859 規格の各パートに応じた 1 ~ 9 の範囲です。テキストは 0 ~ 255 (10 進) の範囲の ISO 文字から成ります。ISO-8859 文字セットはすべて ASCII を基に 128 ~ 255 の範囲に各国語文字などを含めたものです。テキストに 127 より大きい値の文字が含まれていない場合、文字セットは us-ascii として指定すべきであるということに注意してください。そのテキストはその文字セットで十分に表記できるからです。

他の判読可能テキストのフォーマット (ワード・プロセッサのフォーマットなど) を記述するためにさらにサブタイプを追加することができます。この場合のサブタイプには、テキストの意味を判別するための補正ソフトウェアを必要としなければ、アプリケーションがテキストの出現を判別するための書式設定情報が含まれます。

2. multipart

メッセージ本体にはデータ・タイプが独立している複数のオブジェクトが含まれます。本体はカプセル化境界と呼ばれる行でオブジェクトごとに複数の部分に分割されます。境界の内容は Content-Type フィールドのパラメーターで定義されます。たとえば次のようにします。

```
Content-Type: multipart/mixed; boundary="1995021309105517"
```

境界はメッセージのどの部分にも現れてはいけません。境界は大文字小文字の区別があり、メール・ゲートウェイを通っても非常に堅固であることが判明している 75 のセットからの 1 ~ 70 文字から成ります。またこれは

スペースで終わることはできません。(この例では 16 桁の 10 進タイム・スタンプを使用しています。) 各カプセル化境界は、境界値、および接頭部の <CRLF> シーケンスと 2 つのハイフン (RFC 934 との互換性のため) から成ります。最後の部分の終わりを明示する最終境界の接尾部にもハイフンが 2 つ必要です。各部分の中に MIME ヘッダーがあり、これは普通のメール・ヘッダーのようにシーケンス <CRLF><CRLF> で終了しますが、ブランクの場合があります。カプセル化されたメッセージの内容はヘッダー・フィールドで定義します。

次の 4 つのサブタイプが定義されています。

- a. mixed: 独立している別の部分と共に伝送されます。これらは受信側でメール・メッセージに現れる順序で表示されなければなりません。
- b. parallel: これが mixed サブタイプと異なるのは、これは各部分の順序に依存しないということだけです。したがってたとえば受信メール・プログラムは、これらをすべて並列に表示することができます。
- c. alternative: 別の部分は同じ情報の代替バージョンです。これらはオリジナルにより忠実なものが後になるような順序になり、受信側のメール・システムは最良のバージョンをユーザーに表示しなければなりません。
- d. digest: これは multipart/mixed の変形で、デフォルトの type/subtype は text/plain の代わりに message/rfc822 (下記を参照) になります。これは、複数の RFC 822 または MIME メッセージと一緒に伝送される一般的な場合に使用されます。

複合 multipart メッセージの例を図 172 に示します。

```

MIME-Version: 1.0
From: Steve Hayes <steve@hayessj.bedfont.uk.ibm.com>
To: Matthias Enders <enders@itso180.itso.ral.ibm.com>
Subject: Multipart message
Content-type: multipart/mixed; boundary="1995021309105517"

このセクションはブリアンブルと呼ばれます。これはヘッダーの後ろ、最初の境界の前にあります。
multipart メッセージを処理するメール・リーダーは、これを無視しなければなりません。
--1995021309105517

最初の部分。ヘッダーがありません。したがってこれはデフォルトにより、
charset=us-ascii の text/plain です。直前の <CRLF> は、ヌル・ヘッダーを終了する
<CRLF><CRLF> シーケンスの一部です。末尾のものは次の境界の一部です。したがってこの部分は <CRLF>
が 4 つある 5 行のテキストから成っています。
--1995021309105517
Content-type: text/plain; charset=us-ascii
Comments: このヘッダーはデフォルトを明示的に記述しています

これは 1 行のテキストですが、改行で終了しています。

--1995021309105517
Content-Type: multipart/alternative; boundary=
Comments: カプセル化された multipart メッセージ！

このブリアンブルも無視されます。multipart 本体には base64 でエンコードされたイメージと音声が含ま
れています。432 ページの 11.2.3.5 『base64 エンコード方式』を参照してください。multipart の境
界で許されている文字 "_" は base64 エンコード方式にはないため、非常に簡単な境界を使用できるという
特徴があります。
--
Content-type: text/plain

このメッセージには端末装置では表示できないイメージが含まれています。
とてもよいものなので残念です。

--
Content-type: image/jpeg
Content-transfer-encoding: base64
Comments: ユーザーのシステムが MPEG ビデオを表示できれば、この写真が表示されます。本書では
データのごく一部を示しています。
読者は MIME 準拠の眼鏡はお持ちではないでしょうから。

Qk1OAAAAAAAAAAE4EAABAAAAQAEEAPAAAAABAAGAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQAAAAAA
AAAAAAAAAAAAAAAAAAAAAB4VjQSAAAAAAAAAAAGAAAKgAAAJKAACoAAACqAIAAqpIAAMHBwQDJyckA
/9uqAKpJAAD/SQAAAG0AAfVtAACqbQAA/20AAAAkAABVkgAAqiQAAP+SAAAAtgAAVbYAAKq2AAD/
<base64 データがさらに 1365 行続きます >
--
Content-type: video/mpeg
Content-transfer-encoding: base64

AAABswoAeBn//+CEAAABsgAAAOgAAAG4AAAAAAAAAAAAQAAT////wAAAGy//8AAAEbQ/ZlIwwBGWCX
+pqMiJQDjAKyws/1NRrtXcTCLgzVQymqqHaf0sL1sMgMq4SWLCwOTYRdgyAyrhNYsLhhF3DLjAGG
BwdXbV3yMV8/4tzrp3zsAWIGAJg1IBKTeFFI2IsgutIdfuSAAGCTsBVnWdz8afdMMAMgKgMEkPE
<base64 データがさらに 1839 行続きます >
--
ネストされた multipart メッセージはこの前で終わっています。これはエピローグです。
ブリアンブルのようにこれも無視されます。
--1995021309105517--
multipart メイン・メッセージはこの前で終わっています。以上です。では！

```

図 172 MIME - 複合 multipart の例

説明：

message

本体はカプセル化されたメッセージ、またはその一部です。次の 3 つのサブタイプが定義されています。

rfc822

本体自体は RFC 822 メッセージの構文を持つカプセル化されたメッセージです。From:、Subject:、または Date: の少なくとも 1 つがあることが必要です。

注：

RFC 822 はカプセル化されたメッセージのエンベロープの構文を参照しますが、たとえば MIME メッセージをあらかじめ除外することは行いません。

partial

このタイプは IP フラグメント化の場合と同様に、大規模メール項目のフラグメント化を可能にするときに使用されます。SMTP エージェントはメール最大サイズの上限を規制する場合があるため、大規模な項目はフラグメントとして送信する必要があります。message/partial のメール項目の意味は、フラグメント化は受信側にとって透過的であるということです。受信ユーザー・エージェントはフラグメントを再組み立てして、オリジナルと同一のセマンティクスを持つメッセージを新たに作成しなければなりません。Content-Type: フィールドには次の 3 つのパラメーターがあります。

id= メッセージのすべての部分に共通する固有 ID。

number= この部分のシーケンス番号。1 番目の部分に 1 の番号が付けられます。

total= 各部分の合計数。これは最後の部分以外はすべての部分でオプションです。最後の部分はそこでの number および total パラメーターの値が同一であることにより識別されます。

オリジナル・メッセージは常に RFC 822 の規則に従っているメッセージです。1 番目の部分は message/rfc822 メッセージと構文的に同等であり (つまり本体自体にメッセージ・ヘッダーが含まれている)、後続の部分は text/plain メッセージと構文的に同等です。メッセージを再ビルドするときは、RFC 822 ヘッダー・フィールドは囲まれたメッセージからではなく最上位メッセージから取り

出されます。ただしフラグメント化が行われるときに内部メッセージを外部へコピーできないフィールドの場合は例外です（たとえば Content-Type: フィールド）。

注：

message/partial メッセージをさらにフラグメント化することは明示的に許されています。したがってメール・ゲートウェイがメッセージを自由にフラグメントして、すべての部分を確実に伝送できるように小さくすることができます。これを行わない場合は、フラグメント化を行うメール・エージェントは、宛先までの途中で設定されているメール項目の、最大サイズ制限の最小のものが分かっていなければならないことになります。

external-body このタイプには、他の場所のどこかに存在するオブジェクトを指すポインターが含まれます。この構文は message/rfc822 タイプのものです。外部オブジェクトのアクセス方法は最上位メッセージ・ヘッダーで定義します。これには Content-Type: フィールドの access-type: パラメーターおよびそのアクセス・タイプに固有の一連の追加パラメーターを使用します。これはメール・リーダーが、指定されたアクセス・タイプを使って、外部オブジェクトに同期アクセスできるようにするためです。次のアクセス・タイプが定義されています。

ftp	ファイル転送プロトコル (FTP)。受信側は必要なユーザー ID とパスワードを指定する必要があります。セキュリティ上の理由で、これらがメッセージと共に伝送されることはありません。
tftp	Trivial File Transfer Protocol。
anon-ftp	アノニマス FTP。
local-file	データは受信側のローカル・ファイル・システムによって直接アクセス可能なファイルに含まれています。
mail-server	データはメール・サーバーでアクセス可能です。このアクセスは他と違って必然的に非同期になります。

外部オブジェクトを受信すると、そのオブジェクトを message/external-body メッセージの本体内にカプセル化されたメッ

セージ・ヘッダーに付加することにより、必要なメッセージが得られます。このカプセル化されたメッセージ・ヘッダーでは、そのメッセージを解釈する方法を定義しています。(Content-ID: があることが必須であり、また通常は Content-Type: フィールドを持ちます。) カプセル化されたメッセージ本体は使用されない (実質的なメッセージ本体は結局は他の場所のどこかにある) ため、これはファントム本体と呼ばれます。ただしこの例外として、アクセス・タイプが mail-server の場合は、ファントム本体には実質的なメッセージ本体を取り出すために必要なメール・サーバーのコマンドが含まれます。これは、メール・サーバーの構文は幅広く多様であるため、任意のメール・サーバーのコマンドを Content-Type: フィールドのパラメーターとしてエンコードするための構文を分類するよりは、冗長なファントム本体を別の方法で使用する方が簡単になるからです。

3. image

本体にはイメージ・データが含まれ、このデータを表示するにはグラフィカル・ディスプレイまたは他のプリンターなどの装置が必要です。最初に次の 2 つのサブタイプが定義されています。

- a. jpeg: イメージは JPEG フォーマットで JFIF エンコード方式です。
- b. gif: GIF フォーマット。

4. video

本体には動画のイメージ・データ (同期した音声を伴うことがある) が含まれ、このデータを表示するにはインテリジェント端末またはマルチメディア・ワークステーションが必要です。最初に次の 1 サブタイプが定義されています。

- a. mpeg: MPEG フォーマット。

5. audio

本体にはイメージ・データが含まれ、これを表示するにはスピーカーとサウンド・カード (または類似のハードウェア) が必要です。最初に次の 1 サブタイプが定義されています。

- a. basic: 音声エンコード方式のデファクト・スタンダードがないときの最小公分母的なフォーマット。具体的にはこれは、単一チャンネル 8 ビット ISDN mu-law エンコード方式 (サンプル・レート 8kHz) です。

6. application

このタイプは他のカテゴリーに適合しないタイプのためのもので、特にスプレッドシート・データなどユーザーに表示される前にアプリケーション・プログラムで処理されるデータのためのものです。またこれは、メール読み取りプロセスの一部として処理されるアプリケーション・プログラムのためのものでもあります (例として下記の PostScript タイプを参照)。

このタイプの使用は、実行可能メール・メッセージを安全なまたは埋め込みセル環境で実行するようにしているインプリメンテーションでない限り、セキュリティにおいて重大なリスクがあります。

最初に次の 2 つのサブタイプが定義されています。

a. PostScript: Adobe Systems PostScript (Level 1 または Level 2)。

PostScript のセキュリティ上の問題：PostScript はプリンター・データのフォーマットと考えられがちですが、これはプログラム言語であり、PostScript インタープリターを使って application/PostScript タイプを処理することはセキュリティ上の重大な問題があります。PostScript プログラムを自動的にインタープリットするメール・リーダーも、それが受信する実行可能プログラムを自動的に実行するものと原理的には同等です。RFC 2046 には、これに伴う問題が概説されています。

b. octet-stream: このサブタイプは 8 ビット・バイトから成る一般的なバイナリー・データを示します。これはまた、メール・リーダーが不明のタイプまたはサブタイプを見つけたときに想定しなければならないサブタイプでもあります。任意のパラメーターが使用可能ですが、RFC では 2 つのパラメーターに言及しています。受信側にデータの一般的なタイプを通知する type= パラメーターと、バイト・ストリームの中にエンコードされたビット・ストリームを示す padding= です。(padding の値はストリームのバイト境界まで埋め込むために追加された後続ゼロ・ビットの数です。)

推奨インプリメンテーションとしては、そのデータをユーザー・プログラムの入力として使用するか、またはファイルに保管するかのオプションをユーザーに提供するようにします。オプションの“Content-Disposition:” フィールド (RFC 2183 に記述) を使用すると、そのようなファイルの希望名を指定することができます。

セキュリティの問題：RFC では、application/octet-stream の部分を自動的に実行したり、それをメール・ヘッダーで指定されたプログラムの入力として使用するインプリメンテーションに対して、強く勧告しています。そのようなことは受信システムをセキュリティの重大なリスクにさらすことになり、そのシステムが接続されているネットワークの健全性に影響を及ぼすことになります。

上記のサブタイプのいずれにも適合しないデータのタイプももちろん多数あります。メール・プログラム間の連携では、RFC 822 の規則に従って、プライベート値として X- で始まるタイプまたはサブタイプ (あるいはその両方) を使用することができます。その他の値の使用は、最初に Internet Assigned Numbers Authority (IANA) に登録していない限り、許可されません。詳細については、RFC 2048 を参照してください。そこでの趣旨は、追加が必要なタ

イプはあったとしてもわずかだが、サブタイプは多数がそのセットに追加されるだろうというものです。

11.2.3 Content-Transfer-Encoding フィールド

前述のように SMTP エージェントとメール・ゲートウェイでは、安全に伝送できるメール・メッセージの内容は非常に制限されたものになります。前記の MIME タイプでは、メール・メッセージに組み込むことができるさまざまなタイプのオブジェクトを多数示しましたが、それらの多くは上記の制限の中に入りません。そのためそれらのタイプのデータを伝送できるようにエンコードし、受信時にはデコードする必要があります。RFC 2045 ではメール・セーフなエンコード方式として 2 つの形式を定義しています。形式が 1 つではなく 2 つある理由は、メール・セーフと判明している文字の小さなセットがあるとして、テキストの判読の容易さに与える影響を最小限にしてテキスト・データをエンコードできることと、さらに実用上十分コンパクトな 256 バイトのすべての値にランダムに分散している文字から成るバイナリー・データをエンコードできることの両方が可能な形式を案出するのは不可能だからです。

これらの 2 つのエンコード方式は、本体だけに使用されてヘッダーには使用されません。ヘッダーのエンコード方式は、435 ページの 11.2.4 『メッセージ・ヘッダーでの非 ASCII 文字の使用』で説明しています。

Content-Transfer-Encoding: フィールドで、使用されたエンコード方式を定義します。このフィールド名はスマートではありませんが、これは、エンコードはトランスポート・プロセスの機能であり、メールで送られるオブジェクトに組み込まれたプロパティではないということを強調したものです。定義されているエンコード方式は 2 つだけですが、このフィールドに指定できる値は 5 つあります。(通常どおり、これらの値は大文字小文字を区別しません。) 値のうち 3 つは実際にはエンコードが行われていないことを指定しますが、それらはそれぞれ、エンコードが行われなかった別々の理由を暗黙に示します。このことは意味が明確でなくて分かりにくいのですが重要な点です。SMTP 準拠のメール・システムがインターネットに (広く) 普及しているにもかかわらず、MIME のトランスポート・エージェントは SMTP に限定されているわけではありません。そのためメール・エージェントは、SMTP (つまり STD 10/RFC 821) の標準ではメール・セーフではないデータを伝送できることになります。そのようなメール項目が、より制約のあるシステムへのゲートウェイに到達すると、ゲートウェイは指定されたエンコード・メカニズムを使って、その本体を安全に伝送するためにエンコードすべきかどうかを項目ごとに 1 つずつ決定することができます。

エンコード方式として次の 5 つがあります。

- 7-bit (Content-Transfer-Encoding: ヘッダーが省略されている場合のデフォルト)

- 8-bit
- binary
- quoted-printable
- base64

これらについては、続くいくつかのセクションで説明します。

11.2.3.1 7-bit エンコード方式

7-bit エンコード方式は、エンコードが行われていなくて、本体は長さが 1000 文字以下の ASCII テキストの行から成っていることを意味します。そのためこれは、STD 10/RFC 821 に厳密に準拠しているメール・システムなら、メール・セーフであることが分かります。これらは MIME STD 11/RFC 822 より前のメッセージに適用される制約事項であるため、これがデフォルトになっています。

注：

7-bit エンコード方式は、内容が本当にメール・セーフであることを保証するわけでは *ありません*。これには 2 つの理由があります。まず、EBCDIC ネットワークへのゲートウェイではメール・セーフ文字のセットが小さくなるということ、次に、非準拠の SMTP インプリメンテーションが多数あるためです。quoted-printable エンコード方式は、テキスト・データに関するこれらの弱点を解消するように設計されています。

11.2.3.2 8-bit エンコード方式

8-bit エンコード方式は、行は SMTP トランスポートにとって十分短いものですが、非 ASCII 文字（つまり高位ビットが設定されたオクテット）を含んでいる可能性があります。SMTP エージェントが SMTP サービス拡張機能の 8-bit-MIMEtransport (RFC 1652 に記述) をサポートしている場合に 8-bit エンコード方式が可能です。それ以外の場合は SMTP インプリメンテーションで高位ビットがゼロに設定されるため、8-bit エンコード方式が無効になります。

11.2.3.3 binary エンコード方式

binary エンコード方式では、非 ASCII 文字がある場合があります、また行は SMTP トランスポートにとっては長すぎる場合があります。（つまり CRLF シーケンスなしに 999 以上の文字が続いている場合があります。）現在、TCP/IP プロトコル・スタックを基にしたメールによる非エンコード・バイナリー・データのトランスポートのための標準がないため、インターネットまたは TCP/IP ベースの他のネットワーク上で送信される MIME メッセージで binary エンコード方式の使用が有効なのは、external-body 部分のヘッダーに

おいてだけです (前記の message/external-body タイプを参照)。binary エンコード方式は、MIME が、他のメール・トランスポート・メカニズムと共に使用できるか、または長行をサポートする (仮定) SMTP サービス拡張機能と共に使用できれば有効です。

11.2.3.4 quoted-printable エンコード方式

これは実質的な 2 つのエンコード方式の中の 1 つであり、エンコードされた形式でテキスト・ファイルをおおむね判読できます。

- ・ 非メール・セーフ文字はその ASCII 文字の 16 進表記で表されます。
- ・ メッセージの中のすべての行の長さを 76 文字以下に保持するために、可逆 (ソフト) 改行を採り入れています。

quoted-printable エンコード方式では、これらの両方の場合を示す引用文字として等号を使用します。このエンコード方式には 5 つの規則があり、以下にその要約を示します。

1. 改行シーケンス (つまりテキスト・ファイル上の X'0D0A' シーケンス) の部分のものを除いて、どの文字も =XX で表記できる。ここで XX は 2 つの英大文字 16 進数字。他の規則をどれも適用しない場合は、文字は XX として表記されなければならない。
2. X'3D' ("=") を除いて、X'21' ~ X'7E' の範囲のどの文字も ASCII 文字として表記できる。
3. ASCII タブ (X'09') およびスペース (X'20') は、行の最後の文字のときを除いて、ASCII 文字として表記できる。
4. 改行は <CRLF> シーケンス (X'0D0A') で表されなければならない。バイナリー・データをエンコードするときは X'0D0A' は改行ではなく、規則 1 に従って =0D=0A としてコード化されなければならない。
5. エンコードされた行の長さは 76 文字を超えることはできない (<CRLF> を除いて)。行がこれよりも長いときは、列 75 以前にソフト改行が挿入されなければならない。ソフト改行は、シーケンス =<CRLF> (X'3D0D0A')。

この方式は判読性、効率、および堅固性の間の妥協策です。規則 1 および 2 では「エンコードできる」という表現を使っているため、インプリメンテーションでは引用する文字の数について、かなり大きな選択の自由度があります。規則の有効範囲内で可能な引用できる文字がほとんどない場合は、作法に従った ASCII SMTP エージェントを使ってエンコードが行われます。作法に従った EBCDIC ゲートウェイには、引用される文字のリストに次の ASCII 文字のセットを追加すれば十分です。

! " # \$ % & ' () * + , - . / : ;
[\] ^ _ { | } ~

全体の堅固性のためには、すべてのゲートウェイで変わらないことが分かっている 73 文字のセットを除く すべての文字 を引用することをお勧めします。つまり文字と数字 (A-Z、a-z、および 0-9) および次の 11 文字です。

' () + , - . / : = ?

注：

この不変リストにはスペース文字が含まれていません。実際上は、テキスト・ファイルをエンコードするときはスペースだけが引用されるべきです。そうしないと、行の終わりの読みやすさが大きな影響を受けます。

11.2.3.5 base64 エンコード方式

このエンコード方式はテキスト文字を主として含んでいないデータのためのものです。quoted-printable では非テキスト文字それぞれを 3 バイト・シーケンスに置き換えますが、バイナリー・データにとって 3 バイト・シーケンスは総体的には非効率です。base64 エンコード方式では、入力ストリームをビット・ストリームとして扱い、そのビットをより短いバイトに再グループ化し、それらの短いバイトに埋め込みをして 8 ビットにして、次にそれらのバイトをメール・セーフと分かっている文字に変換します。前のセクションの説明のとおり、安全な文字は 73 だけなので、使用可能な最大バイト長は 6 ビットです。この 6 ビットで 64 の固有の文字を表すことができます (これから base64 という名前)。入力と出力は両方ともバイト・ストリームであるため、このエンコードは 24 ビットのグループで行われなければなりません (つまり 3 入力バイトと 4 出力バイト)。このプロセスは次のように表すことができます。

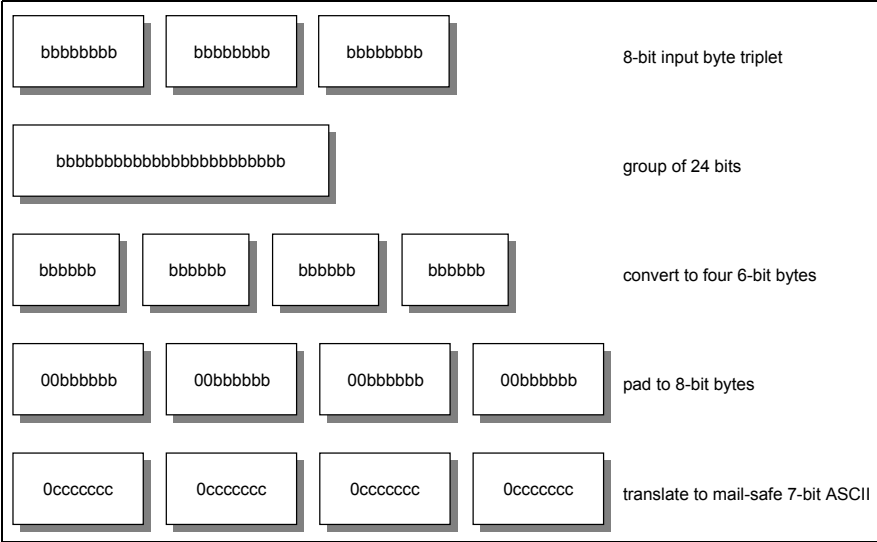


図 173 MIME - base64 エンコード方式 - base64 エンコード方式での 3 入力バイトから 4 出力バイトへの変換方法

使用される変換テーブルは *base64* アルファベットと呼ばれますが、これを次に示します。

表 19 MIME - base64 アルファベット

base64 値	ASCII 文字	base64 値	ASCII 文字	base64 値	ASCII 文字	base64 値	ASCII 文字
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	Y
3	D	19	T	35	j	51	Z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6

base64 値	ASCII 文字	base64 値	ASCII 文字	base64 値	ASCII 文字	base64 値	ASCII 文字
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

埋め込み用に 1 文字 (= 文字) 追加する必要があります。入力 は 24 ビットのグループでエンコードされるバイト・ストリームであるため、その不足分はゼロ、8、または 16 ビットになります。出力のときも同様です。出力が適切な長さのものである場合は、埋め込みは必要ありません。出力が 8 ビット不足している場合、これは 2 つの完全なバイト、1 つの短いバイト、および 1 つの欠落バイトという出力のカルテットに相当します。短いバイトは下位にゼロ・ビットが 2 つ埋め込まれます。欠落バイトは = 文字に置き換えられます。出力が 16 ビット不足している場合、これは 1 つの完全なバイト、1 つの短いバイト、および 2 つの欠落バイトという出力のカルテットに相当します。短いバイトは下位にゼロ・ビットが 6 つ埋め込まれます。2 つの欠落バイトは = 文字に置き換えられます。ゼロ文字 (つまり A) が使用されたとなると、受信エージェントはその入力ストリームをデコードするときに、出力ストリームの最後または最後から 2 つの位置にある末尾の X'00' 文字がデータなのか埋め込みなのか判別できなくなります。埋め込み文字の "=" の数 (0、1、または 2) は、入力ストリームの長さのモジュロ 3 (それぞれ 0、2、または 1) を示します。

11.2.3.6 エンコード方式間の変換

base64 エンコード方式は binary エンコード方式との間であいまいさを伴わない変換を自由に行うことができます。これは両方のエンコード方式がデータをオクテット・ストリームとして扱うためです。このことは quoted-printable から他の 2 つのいずれかに変換する場合にも同じことが言えます (quoted-printable から binary に変換するときのプロセスは中間 binary エンコードを行っているように見ることができます)。このときは、引用文字シーケンスをそれぞれ 8 ビット形式に変換し、ソフト改行を削除し、そしてハード改行を <CRLF> シーケンスに置き換えます。quoted-printable は実際にはレコード・ベースのシステムであるため、この逆のプロセスは厳密には当てはまりません。ハード改行と組み込み =0D=0A シーケンスはセマンティックが異なります。(たとえば、VM などの EBCDIC レコード・ベースのシステム上で quoted-printable をデコードすると、ハード改行はレコード境界にマップされますが、=0D=0A シーケンスは X'0D25' シーケンスにマップされます。)

11.2.3.7 複数のエンコード

MIME ではネストされるエンコードは許されていません。他の Content-Type フィールド (特に multipart および message タイプ) が再帰的に組み込まれた Content-Type では、7-bit、8-bit、binary 以外の Content-Transfer-Encoding を使用することはできません。エンコードはすべて最も内部のレベルで行われなければなりません。この制約の目的は、ユーザー・メール・エージェントの操作を単純化するためです。ネストされたエンコードが許されていないと、メール・エージェントはメッセージの外部の層をデコードする必要なしに、メッセージ全体の構造を常に見ることができます。

ユーザー・メール・エージェントのためのこの単純化は、ゲートウェイにとっての複雑さという代償を伴います。ユーザー・エージェントは 8-bit または binary のエンコード方式を指定できるため、それらのエンコード方式が安全ではないネットワークへのゲートウェイでは、メッセージを次のネットワークに渡す前にそれをエンコードしなければなりません。当然のソリューションとして単にメッセージ本体をエンコードして Content-Transfer-Encoding: フィールドを変更することが考えられますが、これは multipart や message タイプでは許されていません。上記の制約に違反することになるからです。そのためゲートウェイはメッセージのコンポーネントを正しく解析して、必要に応じて最も内部の部分を再エンコードしなければなりません。

さらにタイプ message/partial のメッセージでは、常に 7-bit エンコード方式でなければならないというもう 1 つの制約があります。(8-bit と binary は許されていません。) これは、ゲートウェイがメッセージを再エンコードする必要がある場合はメッセージ全体に対して行う必要がありますが、メッセージの各部分をすべて一緒に使用できない場合があるためです。(ゲートウェイはメッセージ全体を一度に保管できないことや、各部分がそれぞれ異なるゲートウェイを通る経路になる場合もあるため、各部分がシリアルに伝送されることがあります。) そのため message/partial 本体の各部分は、最小公分母的なネットワーク上でメール・セーフでなければなりません。つまり、それらは 7-bit でエンコードされなければなりません。

11.2.4 メッセージ・ヘッダーでの非 ASCII 文字の使用

上記のメカニズムはすべて本体だけについて述べており、ヘッダーについては言及していません。メッセージ・ヘッダーの内容も US-ASCII でコード化されなければなりません。人間が理解できるテキストを含むヘッダー・フィールドの場合、英語以外の言語にとって、これは適切ではありません。各国語文字を組み込むためのメカニズムは、MIME (RFC 2047) の 2 番目の部分で定義されています。このメカニズムは quoted-printable エンコード方式とは異なるものです。これは次の理由でメッセージ本体に使用されると思われます。

- メッセージ・ヘッダーのフォーマットは RFC 822 で厳密に規定されているため、MIME でヘッダー・フィールドに使用されるエンコード方式は本体で使用されるものよりも、より制限された制約の中で機能しなければなりません。
- メッセージ中継プログラムは、しばしばメッセージ・ヘッダーを変更します。たとえば、ヘッダー・フィールドの再配列、一部のフィールドだけの削除、リスト内のメールボックスの再配列、オリジナル・メッセージとは異なる位置でのフィールドのあふれなどです。
- メッセージ処理プログラムの中には RFC 822 の比較的分かりにくい機能のいくつかを適切に処理しないものもあります (< と > などの特殊文字を引用するための \ 文字の使用など)。

MIME がとったアプローチは、RFC 822 では構文上重要ではない正規 ASCII 文字のありそうもないシーケンスを、このプロトコルでの使用のために予約するということです。国別文字が必要なヘッダー・フィールド内の語は、次の形式のエンコードされた語で置き換えられます。

=?charset?encoding?word?=-

各部の説明：

charset	text/plain MIME タイプでの charset パラメーターに使用できる値。つまり、"us-ascii" または "iso-8859-1" ~ "iso-8859-9"。
encoding	B または Q。B はメッセージ本体で使用される base64 エンコード方式と同じです。Q は quoted-printable エンコード方式と似ていますが、X' 20' (ASCII スペース) は _ を使って表します。 ² Q エンコード方式では _ 文字のエンコードが必要であり、改行は許されていません。_、=、およびスペース以外の印刷可能 ASCII 文字は、ヘッダー・フィールドが RFC 822 に従って解析されるときにそれらの文字が構文的に意味がなければ、エンコードされた語の中に引用されないまま残してもかまいません。
	charset および encoding は両方とも大文字小文字を区別しません。
word	スペース以外の ASCII テキスト文字のストリングで、指定のエンコード方式の規則に従います。

エンコードされた語に組み込み空白文字 (スペースまたはタブ) を含むことはできません。また、エンコードされた語の長さは 75 文字までで、76 文字の長さより長い行の上に置くことはできません (<CRLF> は除く)。これらの規

² 下線文字は厳密にはメール・セーフではありませんが、他の文字を使ってスペースを表すと非常に読みにくくなるためにこれが使用されています。

則により、ゲートウェイがエンコードされた語を語の途中で折り返すことはなくなります。エンコードされた語は一般に、ヘッダー・フィールドの中の人間が理解できる部分に使用されます。たとえばメールボックスが次の形式で指定されているとします。

The Octopus <octopus@garden.under.the.sea>

エンコードされた語は The Octopus のセクションに使用できますが、< と > の間のアドレス部には使用できません。RFC 2047 では RFC 822 の構文に関して、エンコードされた語を使用できる場所を細かく指定しています。

11.2.5 参照

MIME の詳細については、以下の RFC に記述されています。

- RFC 2045 – MIME (Multipurpose Internet Mail Extensions) Part One: Format of Internet Message Bodies
- RFC 2046 – MIME (Multipurpose Internet Mail Extensions) Part Two: Media Types
- RFC 2047 – MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text
- RFC 2048 – MIME (Multipurpose Internet Mail Extensions) Part Four: Registration Procedures
- RFC 2049 – MIME (Multipurpose Internet Mail Extensions) Part Five: Conformance Criteria and Examples
- RFC 2156 – MIXER (MIME Internet X.400 Enhanced Relay): Mapping between X.400 and RFC 822/MIME
- RFC 2159 – A MIME Body Part for FAX
- RFC 2183 – Communicating Presentation Information in Internet Messages
- RFC 2231 – MIME Parameter Value and Encoded Word Extensions

11.3 Post Office Protocol (POP)

Post Office Protocol バージョン 3 は STD 番号 53 の標準プロトコルです。その状態は「任意選択 (elective)」になっています。これは RFC 1939 に記述されています。旧 Post Office Protocol バージョン 2 は Historic プロトコルで、状態は「非推奨 (Not Recommended)」です。

Post Office Protocol は、クライアント (送信側 / 受信側) とサーバー (保管) の両方の機能を持つ電子メール・プロトコルです。POP3 は電子メール検索のための基本機能 (ダウンロードと削除) をサポートしています。さらに拡張された機能が IMAP4 (439 ページの 11.4 『Internet Message Access Protocol バージョン 4 (IMAP4)』を参照) でサポートされています。

POP3 クライアントはポート 110 を使ってサーバーとの TCP 接続を確立します。接続が確立されると POP3 サーバーは、クライアントに対してグリーティング・メッセージを送信します。セッションは次に **認証状態** に入ります。セッションがこの状態のとき、クライアントはサーバーに対して自身の識別を送信しなければなりません。サーバーがその ID を正常に検証できると、セッションは **トランザクション状態** に入ります。クライアントはこの状態でメールボックスにアクセスすることができます。クライアントが QUIT コマンドを送信すると、セッションが **更新状態** に入った後にその接続が閉じられます。

11.3.1 POP3 のコマンドと応答

POP3 コマンドは、キーワードと、場合によってはそれに続く 1 つ以上の引き数から成ります。キーワードの長さは 3 または 4 文字で、引き数とは 1 つのスペース文字で区切られます。各引き数の長さは 40 文字以下でなければなりません。

サーバーはクライアントによって発行されたコマンドに対して応答を送信します。この応答は最大 512 文字までで、その応答が肯定か否定かを示す状況標識から始められなければなりません。このときの標識は (+OK) または (-ERR) です。サーバーはこれらの標識を大文字で送信しなければなりません。

POP3 セッションの 3 つの状態を以下に説明します。

- 許可状態

この状態ではクライアントがサーバーに対して識別を送信します。これをインプリメントする方法として次の 2 つがあります (認証の詳細については、RFC 1734 に記述されています)。

- USER および PASS コマンドを使用する。
- APOP コマンドを使用する。

- トランザクション状態

この状態でクライアントは、リスト作成、検索、および削除のコマンドを発行することができます。削除のアクションはこの状態では行われなことに注意してください。クライアントは QUIT コマンドを送信しなければなりません。サーバーはそれを受けて更新状態に進みます。

- 更新状態

この状態では、サーバーがトランザクション状態でクライアントから受信したコマンドに従ってメールボックスを更新して、TCP 接続を終了します。クライアントから quit コマンドを受信する前に何かの理由で接続が切断されていた場合は、サーバーは更新状態に入りません。したがってサーバーは何も更新しないことになります。

11.3.1.1 POP3 の重要なコマンド

POP3 の重要なコマンドをいくつか取り上げて以下に説明します。

USER name	認証の際のユーザー名。
PASS password	認証の際のパスワード。
STAT	メッセージの数とメッセージの合計サイズを取得する。
LIST [msg]	メッセージ番号が指定されると、そのメールのサイズがリストされる (そのメールがある場合)。メッセージ番号が指定されない場合は、すべてのメッセージがメッセージ・サイズとともにリストされる。
RETR msg	メッセージ全体をクライアントに送信する。
DELE msg	指定されたメッセージを削除する。
NOOP	サーバーは肯定応答を送信する以外は何も行わない。
RSET	それまでの削除要求を取り消す (ある場合)。
QUIT	許可状態に入っているときは、単に TCP 接続を終了する。 トランザクション状態に入っているときは、まずメールボックスを更新 (それまでに要求されたメッセージを削除) してから TCP 接続を終了する。

11.3.2 参照

Post Office Protocol の詳細については、次の RFC に記述されています。

- RFC 937 – Post Office Protocol – Version 2
- RFC 1939 – Post Office Protocol – Version 3

11.4 Internet Message Access Protocol バージョン 4 (IMAP4)

Internet Message Access Protocol バージョン 4 は提案標準のプロトコルです。その状態は「任意選択 (elective)」になっています。これは RFC 2060 に記述されています。

IMAP4 はクライアントとサーバーの両方の機能を持つ電子メッセージング・プロトコルです。POP と同様、IMAP4 サーバーは複数ユーザー宛てのメッセージを保管してそれらをクライアントの要求に応じて検索しますが、その際 IMAP4 クライアントには POP クライアントよりも多くの機能があります。IMAP4 ではクライアントは複数のリモート・メールボックスを持つことができ、そこからメッセージを検索したり、いつでも任意のメールボックスを選択することができます。IMAP4 クライアントは、ダウンロードするメッセージの基準を指定して、たとえば遅いリンクでは大規模メッセージは転送しないようにすることができます。また IMAP4 はメッセージをサーバー上に常に

保持しており、クライアントにはコピーを複製します。接続が切断されたクライアントによって行われたトランザクションは、クライアントとサーバーの定期的な再同期によってサーバーのメールボックスに対して実施されます。IMAP4 の機能をよく理解できるように、まず最初に IMAP4 の基礎をなす電子メール・モデルを解説します。これらの詳細については、RFC 1733 - Distributed Electronic Mail Models In IMAP4 に記述されています。

11.4.1 IMAP4 の基礎をなす電子メール・モデル

IMAP4 は 3 つの主要な電子メール・モデルをすべてサポートしています。これらのモデルは、オフライン、オンライン、および切断使用の各モデルです。

オフライン・モデルではクライアントは、サーバーに定期的に接続してメール・メッセージをダウンロードします。サーバー上のメール・メッセージは削除されます。そのためメールはクライアント上でローカルに処理されます。このモデルを使用するメール・プロトコルの例として POP3 があります。詳細については、437 ページの 11.3 『Post Office Protocol (POP)』を参照してください。

オンライン・モデルでは、クライアントはサーバー上で変更を行うことができます。これは多くの場合 NFS などのリモート・ファイル・システムのプロトコルを使って行われます。つまり、メールはサーバー上に保持されていますが、クライアント上のソフトウェアによって処理されます。詳細については、394 ページの 10.3 『Network File System (NFS)』を参照してください。

切断使用モデルは、オフライン・モデルとオンライン・モデルが複合したものです。このモデルでは、クライアントはデータをダウンロードしてそれをローカルで変更し、後でそれをサーバーにアップロードします。このモデルを使用できるメール・プログラムの例として、Lotus Notes メールがあります。

これらの 3 つのモデルにはいくつかの利点と欠点があります。IMAP4 はこれらのモデルをすべてサポートするため、クライアントは別のモデルに切り替えることができます。たとえば接続が遅すぎるときにメールボックスに大規模なメッセージがある場合などです。IMAP4 クライアントはメッセージの中の小さな部分を検索してその部分を変更し、次にサーバーに接続してその部分を再びアップロードすることができます。

IMAP4 は MIME multipart メッセージを認識します。FETCH コマンドを使って、MIME のカプセル化されたメッセージの n^{th} (n 番目) の部分だけをダウンロードすることができます。

11.4.2 IMAP4 のコマンドと応答

POP3 (437 ページの 11.3 『Post Office Protocol (POP)』を参照) と同様、IMAP4 クライアントはポート 143 を使ってサーバーとの TCP 接続を確立します。接続が確立されるとサーバーは、グリーティング・メッセージを送信します。その後クライアントとサーバーは対話式でデータを交換します。クライアントがコマンドを送信するときは常に、サーバーはそのコマンドに対する完了結果応答を送信します。サーバーはその他の理由でもデータを送信できます。コマンドと応答の形式は、すべて行が CRLF で終わります。

クライアント・コマンドはすべて、タグと呼ばれる別々の ID で始まります。コマンドに対するサーバーの応答は、サーバーがコマンドを受け取ったとおりの順序にならない場合があります。タグ ABC005 のコマンドが送信された後にタグ ABC006 の別のコマンドが送信される場合を考えてみます。2 番目のコマンドの処理にそれほど時間がかからない場合は、サーバーはまず ABC006 コマンドに対して、関係のあるタグ (この場合は ABC006) で始まる行で応答します。送信されるすべてのコマンドに、固有のタグを使用しなければなりません。

次の 2 つの場合、クライアント・コマンドは 1 行では完全に送信することはできません。このような場合は、継続応答を受信するか BAD 応答を受信するまで他のコマンドを送信できません。次の場合です。

1. クライアントからサーバーヘリテラル・ストリングを送信するとき、クライアントは {, length, } シーケンスを送信した後に、サーバーが + で始まる行で応答するのを待機する。ストリングの残りの部分はその応答があった後に送信される。
2. 認証ステージ時、クライアントは + 継続行を待機する必要もある。

サーバーは、クライアントによって発行されたコマンドに対してコマンド完了応答を送信します。この応答は関係するコマンドのタグと同じタグとその前に置かれる状況標識で始まります。この状況標識は操作の結果が肯定が否定かを示すものです。標識には、OK、NO、または BAD があります。

データ応答と状況応答はすべて (*) で始まります。これらはタグなし応答と呼ばれます。これらはサーバーによって随時送信されます (コマンドが完了していなくても)。たとえばセッション中に新しいメール・メッセージが到着すると、サーバーは関係のあるフラグを送信してクライアントに通知します。

11.4.3 メッセージ番号

メッセージを識別するために使用されるメソッドは 2 つあります。固有 ID とメッセージ・シーケンス番号です。属性の一部については、続くいくつかのセクションで説明します。詳細については、RFC 2060 を参照してください。

11.4.3.1 「固有 ID (UID)」メッセージ属性

すべてのメッセージには 32 ビットの ID があり、固有 ID 妥当性値と結合すると 64 ビット値になります。メールボックスに新しいメッセージが追加されると、それまで追加されていたメッセージのものより高位の UID がそのメッセージに割り当てられます。固有 ID は連続している必要はありません。固有 ID は他のセッションにも持続します。したがって、クライアントは接続を切断しても、それまでのセッションでの値と同じ値を次のセッションでも使用することができます。

各メールボックスには固有 ID 妥当性値があります。次のセッションで同じ値を使用できない場合、新しい値はそれまでのセッションで使用されていた値よりも大きくなければなりません。たとえば、あるセッションでメールボックスが削除され、次のセッションで新しいメールボックスが同じ名前で作成されると、クライアントには、メールボックスの名前が同じであるため、それが新しいメールボックスであることが分かりません。この場合は固有 ID 妥当性値が変更されなければなりません。固有 ID 妥当性値はメールボックス選択に伴ってクライアントに UIDVALIDITY として送信されます。

11.4.3.2 「メッセージ・シーケンス番号」メッセージ属性

メッセージ・シーケンス番号は、メールボックス内でのメッセージの相対位置を示します。これは昇順になっていなければなりません。メッセージ・シーケンス番号はセッション中や次のセッションで変わることがあります。新規メッセージが追加されると、メッセージ全体（その新規メッセージも含む）の数がそのメッセージに割り当てられます。メッセージの合計数と最新メッセージのメッセージ・シーケンス番号は、常に同じになっていなければなりません。それは、新しいメール・メッセージごとにメッセージ・シーケンス番号を正しく割り当てるためです。メッセージが永続的に除去された場合、メッセージ・シーケンス番号はそのセッションの中で再計算されます。

11.4.3.3 「フラグ」メッセージ属性

フラグはメッセージの現在の状況を示すために使用されます。フラグには、永続 (permanent) とセッションのみ (session-only) の 2 つのタイプがあります。システムのフラグはすべて \ で始まります。詳細については、RFC 2060 を参照してください。

\Seen	メッセージはすでに読まれています。
\Answered	メッセージの返事がすでに送られています。
\Flagged	メッセージは特別に重要としてマークされています。
\Deleted	メッセージは今後の永続除去で削除されます。
\Draft	メッセージはすでに完成しています。

\Recent

新しいメッセージが到着しています。このセッションはその到着後の最初のセッションです。このフラグはクライアントが変更することはできません。

11.4.4 IMAP4 の状態

POP3 と同様 (437 ページの 11.3 『Post Office Protocol (POP)』を参照)、IMAP4 のセッションも異なる状態で動作します。特定の状態で有効なコマンドと、すべての状態で有効なコマンドがあります。クライアントがその状態では適切ではないコマンドを送信すると、サーバーはエラー・メッセージで応答します。エラー・メッセージは一樣ではありません。サーバーは BAD または NO を送信しますが、どちらを送信するかはインプリメンテーションによって異なります。IMAP4 には次の 4 つの状態があります。

未認証状態	この状態ではクライアントがサーバーに対して識別を送信します。
認証済み状態	この状態ではクライアントは先に進むためにメールボックスを選択しなければなりません。
選択済み状態	この状態ではメールボックスの選択が正常に終了しています。
ログアウト状態	この状態では、クライアントの要求により、または他の理由により接続が終了します。

IMAP4 のセッションのフロー・チャートを図 174 に示します。

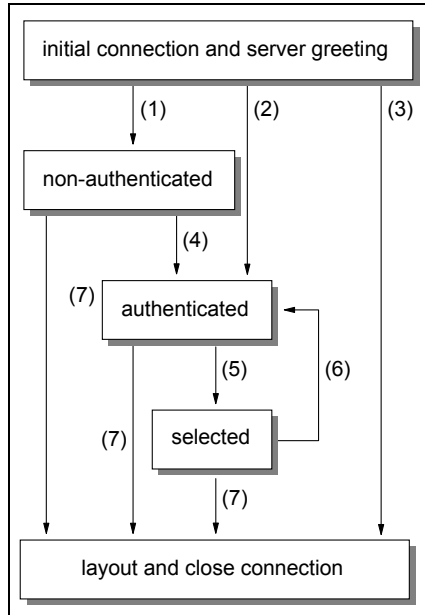


図 174 IMAP4 - フロー・チャート

各部の説明：

- (1) 事前認証なしの接続 (OK グリーティング)
- (2) 事前認証済み接続 (PREAUTH グリーティング)
- (3) 拒否された接続 (BYE グリーティング)
- (4) 正常な LOGIN または AUTHENTICATE コマンド
- (5) 正常な SELECT または EXAMINE コマンド
- (6) CLOSE コマンド、または失敗した SELECT または EXAMINE コマンド
- (7) LOGOUT コマンド、サーバー・シャットダウン、または接続クローズ

11.4.5 クライアント・コマンド

IMAP4 コマンドの大部分は対応する状態で使用しなければなりません。それらの中のいくつかは複数の状態で使用することができます。コマンドとそれが使用される状態を以下に示します。

- 任意の状態で使用

この状態では次のコマンドが有効です。

- CAPABILITY: サーバーがサポートしている機能のリストの要求を送信する。
- NOOP: 無実行。サーバーの非アクティブ・オートログアウト・タイマーのリセットに使用できます。
- LOGOUT: 接続終了の要求を送信する。

- 未認証状態で使用

この状態では、任意の状態でのすべてのコマンドと次のコマンドが有効です。

- AUTHENTICATE: サーバーの引き数を使用する特殊認証メカニズムを要求する。サーバーがそのメカニズムをサポートしていない場合、サーバーはエラー・メッセージを送信します。
- LOGIN: ユーザー名とパスワードをプレーン・テキストで送信する。

- 認証済み状態で使用

この状態では、任意の状態でのすべてのコマンドと次のコマンドが有効です。

- SELECT: メールボックスを選択する。
- EXAMINE: このコマンドもメールボックスを選択するが、このコマンドでのメールボックスへのアクセスは読み取り専用になる。
- CREATE: 指定した名前のメールボックスを作成する。INBOX や既存のメールボックスを作成することはできません。
- DELETE: 指定した名前のメールボックスを永続的に除去する。
- RENAME: メールボックスの名前を変更する。
- SUBSCRIBE: 指定したメールボックスをサブスクリプション・リスト (LSUB コマンドで取得可能) に追加する。
- UNSUBSCRIBE: 指定したメールボックス名をサブスクリプション・リストから除去する。
- LIST: サーバーで使用可能なすべての名前の完全セットの名前のサブセットを要求する。
- LSUB: サブスクリプション・リストの名前のサブセットを要求する。
- STATUS: 指定したメールボックス名の状況を要求する。サーバーは各フラグを検査して各フラグの状況に従って状況を送信します。
- APPEND: 指定したメールボックスにメッセージ・テキストを新規メッセージとして追加する。

- 選択済み状態で使用

この状態では、任意の状態でのすべてのコマンド、認証済み状態でのすべてのコマンド、および次のコマンドが有効です。

- CHECK: 選択したメールボックスの状態の、メモリーおよびディスクでの解決を要求する。
- CLOSE: それまでに削除済みとしてマークされていたすべてのメッセージを現在選択済みのメールボックスから永続的に削除して、選択済み状態から認証済み状態に戻る。
- EXPUNGE: それまでに削除済みとしてマークされていたすべてのメッセージを現在選択済みのメールボックスから永続的に削除する。
- SEARCH: 指定の検索基準に一致するメッセージをメールボックスから検索する。
- FETCH: 選択済みメールボックスでのメッセージに関連したデータを検索する。
- STORE: メッセージを FETCH コマンドで検索されたデータを使って更新する。
- COPY: 指定したメッセージを、指定した宛先メールボックスの最後にコピーする。
- UID: メッセージ・シーケンス番号の代わりに固有 ID を戻す。このコマンドは他のコマンドと併用するものです。

11.4.6 参照

IMAP プロトコルの詳細については、以下の RFC に記述されています。

- RFC 1733 – Distributed Electronic Mail Models in IMAP4
- RFC 2060 – Internet Message Access Protocol – Version 4rev1

第 12 章ワールド・ワイド・ウェブ (WWW)

ここ数年でインターネットの利用がより容易になり広く普及しましたが、この章ではそれに寄与したプロトコルとアプリケーションのいくつかを紹介します。ワールド・ワイド・ウェブ (WWW) のトラフィック (その大部分が Hypertext Transfer Protocol (HTTP) を使用) は、他のアプリケーション・プロトコル (Telnet や FTP など) を実際に大きく上回っており、インターネットの帯域幅の大部分を使用しています。現在のコンピュータのオペレーティング・システムはデフォルトで Web ブラウザー・アプリケーションをサポートしており、Web サーバーをサポートしているものもあります。したがってエンド・ユーザーやビジネスで、ワールド・ワイドにネットワーク化されたコンピュータの多大な能力を探索して活用することが非常に容易になりました。

ワールド・ワイド・ウェブ (WWW) は、最初はスイスにある欧州素粒子物理学研究所 (CERN) の Tim Berners Lee 氏によって、地理的に分散した科学者のグループ間での研究資料の共用と編集を容易にすることを目的として、1989 年に開発されたグローバル・ハイパーテキスト・システムです。

1993 年、Web は急速に成長し始めました。その原動力は主に、X Window ベースのアプリケーションで Mosaic と呼ばれる Web ブラウザー・プログラムを、National Center for Supercomputing Applications (NCSA) が開発したことによるものでした。このアプリケーションは Web への最初のグラフィカル・ユーザー・インターフェースを備えており、ブラウズが便利になりました。現在の Web ブラウザーやサーバーは、ほとんどすべてのプラットフォームで使用できます。Web の普及の急速な発展の原因は、インターネットでのワールド・ワイドのリソースのナビゲートとそれらの検索が柔軟にできることです。

Web サーバーの数も急速に増加しており、NSF バックボーンでのポート 80 (HTTP Web サーバー用のウェルノウン・ポート) 上のトラフィックも驚くべき速度で増加してきました。ただし NSFNET は 1995 年に研究目的の専用ネットワークに転換されたため、バックボーン・トラフィックに関する包括的な統計の入手は現在は容易ではありません。

12.1 Web ブラウザー

一般に Web サーバーへのアクセス機能を備えているアプリケーションをブラウザと呼びます。そのためブラウザの機能と構造はインプリメンテーションによってさまざまです。Web ブラウザーは最低限でも、HTML インタープリターと、HTML Web ページの検索に使用される HTTP クライアントから成っています。多くのブラウザはこの基本要件の他に、FTP、NNTP、E メール (POP および SMTP クライアント) などを、使いやすいグラフィカ

ル・インターフェースでサポートしています。448 ページの図 175 に Web ブラウザーの基本的な構造を示します。

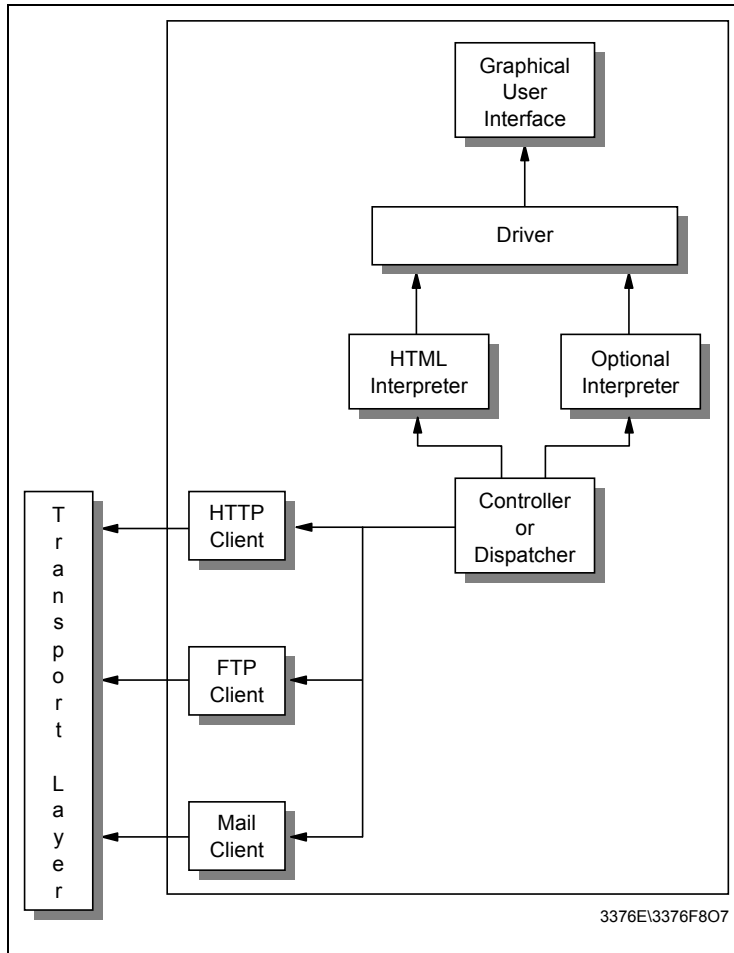


図 175 Web ブラウザーの構造

他の多くのインターネット・ファシリティと同様、Web でもクライアント / サーバー処理モデルを使用します。Web ブラウザーはクライアント・コンポーネントです。Web ブラウザーの例として、Mosaic、Netscape Navigator、Microsoft Internet Explorer があります。Web ブラウザーの役割は、情報のフォーマット設定と表示、ユーザーとの相互作用、および Web ブラウザーが直接はサポートしていないデータ・タイプのための外部ビューアーや Telnet などの外部機能の呼び出しです。Web ブラウザーは GUI ワークステーション環境での「汎用クライアント」になりました。これは、DEC VT100 や IBM 3270 などの普及した端末をエミュレートできる機能が、広範

図のさまざまなコンピューター上の文字ベースのアプリケーションへの接続とアクセスを可能にしているのと同じことです。Web ブラウザーは普及している GUI ワークステーション・プラットフォームのすべてで広範囲に使用可能であり、高価なものではありません。

12.2 Web サーバー

Web サーバーの役割は Web ブラウザーからの情報の要求にサービスを提供することです。情報は、サーバーのローカル・ディスクから検索されたファイル、またはサーバーによって呼び出されたプログラムによって特定のアプリケーション機能が実行されて生成されたものです。

パブリック・ドメイン Web サーバーが多数あり、これらは Windows NT などのパーソナル・コンピューター環境だけでなく UNIX 系の大部分を含むさまざまなプラットフォームで使用できます。有名なパブリック・ドメイン・サーバーとして、CERN、NCSA httpd、および Apache サーバーがあります。

IBM HTTP Server (現行バージョンは V1.3.19) は、Web で最も普及しているサーバーである Apache HTTP Server を基にしています。この HTTP Server は、AIX、Solaris、Windows NT、Windows 2000、HP-UX、および Linux 上で動作します。他のバージョンとして IBM HTTP Server for AS/400 と IBM HTTP Server for OS/390 があり、これらによって IBM オファリングが広範囲のプラットフォームで確保されます。IBM は Apache-powered HTTP Server を拡張しました。たとえば IBM は WebSphere バンドルの一部としてセキュア・トランザクションのために SSL を追加し、フル・サポートを提供しています。IBM HTTP Server のフィーチャーとして次が挙げられます。

- 容易なインストール。
- SSL および TLS セキュア接続のサポート。
- 高速応答キャッシュ・アクセラレーター
- WebSphere バンドルの一部としての IBM サポート。
- ハードウェア暗号のサポート。
- IHS サーバーの管理と構成を支援する管理サーバー。
- すべての WebSphere 製品に共通するナビゲートが容易な設計を使用したヘルプ情報。

詳細については、次のサイトで参照してください。

<http://www.software.ibm.com/webserver>

12.3 Hypertext Transfer Protocol (HTTP)

HTTP 1.1 は Draft standard のプロトコルです。これは RFC 2616 に記述されています。以前の HTTP 1.0 は Informational のプロトコルであり、RFC 1945 に記述されています。

HTTP は、HTML 文書を転送できるように設計されたプロトコルです。HTML はハイパーテキスト文書の作成に使用されるタグ言語です。ハイパーテキスト文書には、強調表示された用語や件名についての追加情報が含まれている他の文書へのリンクが含まれています。そのような文書には、グラフィック・イメージ、音声とビデオ・クリップ、Java アプレット、さらには仮想現実の世界（その種のエレメントのためのスクリプト記述言語である VRML で記述）など、テキストとは別に他のエレメントを含めることができます。HTML の詳細については、461 ページの 12.4.1.1 『ハイパーテキスト・マークアップ言語 (HTML)』を参照してください。

12.3.1 HTTP の概要

HTTP は要求応答アクティビティーがベースになっています。クライアントはブラウザと呼ばれるアプリケーションを実行し、サーバーとの接続を確立して、要求を要求メソッドの形式でサーバーに送信します。サーバーは状況行で応答します。これには、メッセージのプロトコル版番号、成功またはエラーのコード、それに続いてサーバー情報、エンティティー情報、可能な本体コンテンツを含むメッセージが含まれています。

HTTP トランザクションは、次の 4 つのステップに分けられます。

1. ブラウザーが接続を開く。
2. ブラウザーがサーバーに要求を送信する。
3. サーバーがブラウザに応答を送信する。
4. 接続が閉じられる。

インターネット上の HTTP 通信は一般に、TCP 接続上で行われます。デフォルト・ポートは TCP 80 ですが、他のポートも使用できます。このことは、HTTP をインターネットや他のネットワークでの他のプロトコルの上位にインプリメントすることを可能にしています。HTTP の前提となっているのは信頼できるトランスポートであるということだけであり、それが保証されるプロトコルであれば使用することができます。

実験的なアプリケーションは別にして、現在の方法では、クライアントは各要求の前に接続を確立すること、およびサーバーは応答の送信後にその接続を閉じることが必要です。クライアントとサーバーは両方とも、ユーザーのアクション、自動タイムアウト、あるいはプログラム障害のために、どちらかが通常より早く接続を閉じる場合があることを分かっている必要があります。また接続のそのようなクローズは予測可能な望ましい方法で処理しなければ

なりません。いずれの場合も、一方または双方によって接続が閉じられると、そのときの状況にかかわらず現行要求は常に終了します。

簡単に言うと HTTP は、接続を追跡しないため、ステートレス・プロトコルです。たとえば 2 つのグラフィックスを含むページをロードする場合、グラフィックスが可能なブラウザーは TCP 接続を 3 つ開くことになります。1 つはそのページのために、そして 2 つはグラフィックスのためです。ただし大半のブラウザーは、このような複数の接続を同時に処理することができます。

非常に多数の Web ページがそうなのですが、1 ページが多くのエレメントで構成されていると、この動作はかなりのリソース集中になることがあります。HTTP 1.1 (RFC 2616 で定義) では、ページ上のエレメントのタイプあたり 1 TCP 接続が確立されるようにして、そのタイプのすべてのエレメントがそれぞれ同一接続で転送されるようにするというところまでこの問題を軽減しています。これは接続を持続するという点で HTTP 1.0 と異なります。

しかし要求がそれまでの接続時に交換された情報によって決まることになると、その情報はプロトコルの外側で保持する必要があります。そのような持続情報を追跡する 1 つの方法として cookies を使用します。cookie は、HTTP トランザクション時にクライアント Web ブラウザーと Web サーバーの間で交換される一連の情報です。cookie の最大サイズは 4 KB です。これらの情報のすべて (cookie) が単一のファイルに保管され、Web ブラウザーのディレクトリーに置かれます。cookies が使用不可のときは、そのファイルは自動的に削除されます。サーバーは後のどの接続でも cookie を検索して検査することができます。cookies はプライバシーの露出の可能性があると思なされるため、Web ブラウザーではユーザーが特定のサーバーの cookies を受け入れるかどうかを決定できるようにしておく必要があります。cookies は単に HTTP 接続のいくつかの種類の状態を保持することが目的ですが、クライアントとサーバーのセキュアな認証が Secure Sockets Layer (SSL) でサポートされています (777 ページの 21.8 『Secure Sockets Layer (SSL)』を参照)。

12.3.2 HTTP の操作

HTTP の通信はほとんどの場合、ユーザー・エージェントによって開始され、起点サーバー上のリソースの要求が行われます。最も単純な場合の接続は、図 176 に示すように、ユーザー・エージェントと起点サーバー間の単一の接続で確立されます。

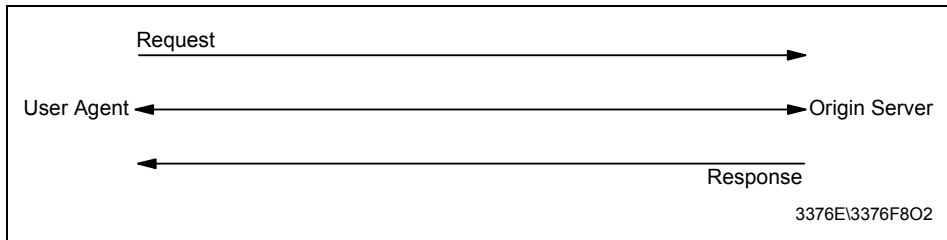


図 176 HTTP - クライアント / サーバーの単一接続

ユーザー・エージェントと起点サーバーの間の直接接続がない場合があります。ユーザー・エージェントと起点サーバーの間に、プロキシ、ゲートウェイ、トンネルなどの媒介が1つ(以上)ある場合です。要求と応答は媒介によって評価され、図 177 に示すように、宛先または別の媒介に要求応答チェーンで転送されます。

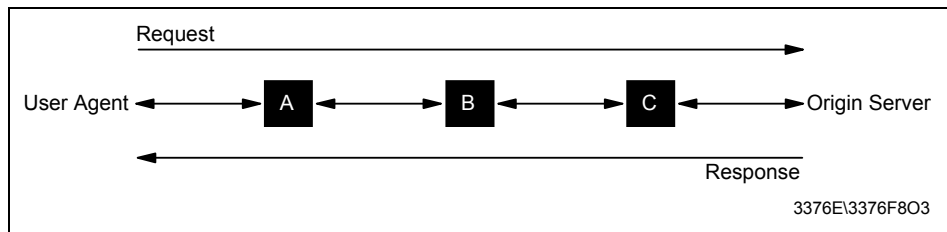


図 177 HTTP - 中間に媒介があるクライアント / サーバー接続

709 ページの 21.3.4 『アプリケーション・レベル・ゲートウェイ (プロキシ)』の説明のように、プロキシはデータの内容を処理でき、したがってデータを変更することができます。プロキシは要求を受信すると、メッセージのすべてまたは一部を書き換えて、そのメッセージを次の宛先に転送します。ゲートウェイはメッセージを受信すると、基礎となっているプロトコルにそのメッセージを適切なフォーマットで送信します。トンネルではメッセージの内容は処理されないため、メッセージは単にそのまま転送されるだけです。

プロキシとゲートウェイは一般に HTTP メッセージのキャッシングを処理することができます。これによってネットワーク上の応答時間と IP トラフィックを大幅に低減できます。トンネルではメッセージの内容を理解できないため、HTTP メッセージのキャッシュ・データを保管できません。上記の図 (図 177) の場合、媒介 (A、B、および C) のいずれかが HTTP メッセージに内部キャッシュを使用していると、起点サーバーの応答が応答チェーンでそれまでにキャッシュに入れられているときは、ユーザー・エージェントはその応答を媒介から取得することができます。図 178 は、A が応答チェー

ンで起点サーバーの以前の応答のコピーをキャッシュに入れている場合を示したものです。この場合、要求に対するサーバーの応答がユーザー・エージェントの内部キャッシュにまだ入っていないときは、ユーザー・エージェントはその応答を A から直接取得することができます。

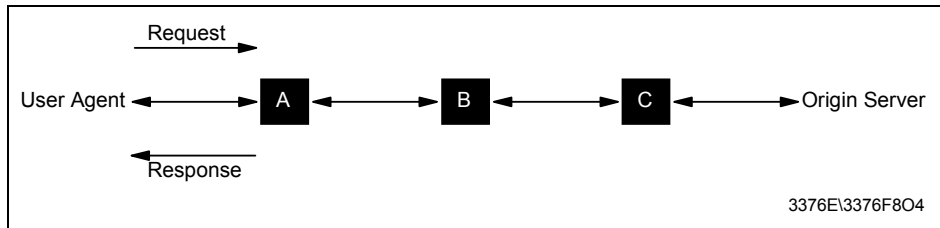


図 178 HTTP - キャッシュされているサーバー応答

キャッシングがすべてのサーバー応答に適用されるわけではありません。キャッシング動作は、キャッシュに入れることができるサーバー応答を決定する特殊な要求で変更することができます。そのためにサーバー応答は、キャッシュ不可、共通、またはプライベート（共通キャッシュにはキャッシュ不可）がマークされます。キャッシング動作とキャッシュ可能応答については、459 ページの 12.3.2.11 『HTTP のキャッシング』で説明しています。

12.3.2.1 プロトコル・パラメーター

HTTP プロトコル・パラメーターのいくつかを以下に説明します。完全なリストおよび詳細な説明は、RFC 2616 を参照してください。

• HTTP バージョン

HTTP では <メジャー>.<マイナー> の番号方式でプロトコルのバージョンを示します。さらに連結される場合はプロトコル・バージョン・ポリーシーに従って行われます。<メジャー> 番号は、メッセージ・フォーマットの変更など、プロトコルに重要な変更があるときに 1 つ増えます。<マイナー> 番号は、メッセージ・フォーマットに影響しない変更のときに 1 つ増えます。

HTTP メッセージのバージョンは、メッセージの 1 行目の HTTP-Version フィールドで送信されます。HTTP-Version フィールドのフォーマットは次のとおりです（増補バックス正規形式は、RFC 2822 を参照してください）。

```
HTTP-Version = "HTTP" "/" 1*DIGIT "." 1*DIGIT
```

• Uniform Resource Identifier (URI)

WWW アドレスおよび Uniform Resource Locator (URL) と Uniform Resource Name (URN) の組み合わせを、一般に Uniform Resource Identifier (URI) と呼びます。URI は実際はサーバー上のソースの場所と名

前を示す文字列です。URI および URL の構文の詳細については、RFC 2616 と RFC 2396 を参照してください。

- HTTP URL

HTTP URL 方式を使って、HTTP プロトコルでネットワーク・リソースの位置を指定することができます。これは URI 汎用構文に基づいており、RFC 2396 に記述されています。URL 方式の汎用構文は次のとおりです。

```
HTTP_URL = "http" "://" host [ ":" port ] [ abs_path ]
```

ポート番号はオプションです。ポート番号が指定されない場合のデフォルト値は 80 です。

12.3.2.2 HTTP メッセージ

HTTP メッセージは、以下のフィールドで構成されています。

- メッセージ・タイプ

HTTP メッセージはクライアント要求かサーバー応答です。次の文字列は HTTP メッセージ・タイプを示しています。

```
HTTP-message = Request | Response
```

- メッセージ・ヘッダー

HTTP メッセージ・ヘッダー・フィールドは、次のいずれかになります。

- 汎用ヘッダー
- 要求ヘッダー
- 応答ヘッダー
- エンティティ・ヘッダー

- メッセージ本体

転送エンコードが設定されていない場合のエンティティ本体を、メッセージ本体と呼びます。メッセージ本体は、関係のある要求または応答のエンティティ本体をそのまま運びます。

- メッセージ長

メッセージ長は、メッセージ本体が組み込まれている場合のメッセージ本体の長さを示します。メッセージ長は、RFC 2616 に詳細が記述されている基準に従って決定されます。

- 汎用ヘッダー・フィールド

汎用ヘッダー・フィールドは要求メッセージと応答メッセージの両方に適用されます。現在定義済みの汎用ヘッダー・フィールドのオプションは、次のとおりです。

- キャッシュ制御

- 接続
- 日付
- プラグマ
- 転送エンコード
- アップグレード
- 経由

12.3.2.3 要求

クライアントからサーバーに対する要求メッセージには、リソースに適用するメソッド、ソースの ID、および使用中のプロトコル版番号が含まれています。要求メッセージ・フィールドは次のようになります。

```
Request = Request-Line
          *( general-header | request-header | entity-header )
          CRLF
          [ message-body ]
```

詳細については、RFC 2616 を参照してください。

12.3.2.4 応答

HTTP サーバーは、クライアント要求を評価した後に応答を戻します。応答メッセージ・フィールドは、次のようになります。

```
Request = Request-Line
          *( general-header | request-header | entity-header )
          CRLF
          [ message-body ]
```

詳細については、RFC 2616 を参照してください。

12.3.2.5 エンティティ

特に指示されない限り、エンティティは、クライアントまたはサーバーによって、要求メッセージまたは応答メッセージで送信されます。エンティティは次のもので構成されます。

- エンティティ・ヘッダー・フィールド
- エンティティ本体

12.3.2.6 持続接続

HTTP 1.1 とそれ以前のバージョンの HTTP の大きな違いは、HTTP 1.1 では持続接続をデフォルトとして使用するということです。従来のバージョンのインプリメンテーションでは、URL ごとに別々の TCP 接続が確立され、クライアントは同一 URL にあるイメージとその関連データについて複数の要求を

行う必要があります。この方法は、ネットワーク上の輻輳とパフォーマンス上の問題を引き起こしていました。持続 HTTP 接続には多くの利点があります。最大の利点は TCP 接続数の低減、次に待ち時間の縮小です。

12.3.2.7 メソッドの定義

現在定義済みのメソッドは、以下のとおりです。

- 安全およびアイデムポtentのメソッド

副次作用を引き起こさないと見なせるメソッドは、安全と表現します。アイデムポtentのメソッドは GET、HEAD、PUT、および DELETE です。

- OPTIONS

クライアントは、サーバーのソースまたは機能に関連したオプションまたは要求を、リソース検索をしないで決定することができます。

- GET

クライアントは、要求 URI によって決定したデータを検索することができます。

- HEAD

クライアントは、エンティティについてのメタ情報を検索することができます。ユーザーはエンティティ本体を転送する必要がありません。

- POST

post 機能はサーバーによって決定されます。

- PUT

このメソッドは post メソッドに似ていますが、1 つの重要な違いは、post 要求内の URI はエンクローズ・エンティティを処理するリソースの識別を示すことです。

- DELETE

このメソッドでは、サーバーは要求 URI によって決定されたソースを削除する必要があります。

- TRACE

この trace メソッドはテストや診断を目的とするものであり、クライアントは他方のサイドでメッセージがどのように検索されたかを調べることができます。

12.3.2.8 状況コードの定義

状況コードは以下のように定義されています。

- 通知 (1xx)

通知状況コードは仮の応答を示します。現在定義済みのコードは、次のとおりです。

- 100 Continue (継続)
- 101 Switching Protocols (プロトコル切り替え中)

• 正常 (2xx)

このクラスのコードは、特定の要求の受信、理解、および受け入れが正常に終了したことを示します。現在定義済みのコードは、次のとおりです。

- 200 OK
- 201 Created (作成済み)
- 202 Accepted (受け入れ済み)
- 203 Non-Authoritative Information (参考情報)
- 204 No Content (コンテンツなし)
- 205 Reset Content (コンテンツ・リセット)
- 206 Partial Content (部分コンテンツ)

• リダイレクト (3xx)

このクラスのコードは、要求を完了するためにはユーザー・エージェントのアクションが必要であることを示します。現在定義済みのコードは、次のとおりです。

- 300 Multiple Choices (多肢選択)
- 301 Moved Permanently (永続的に移動済み)
- 302 Moved Temporarily (一時的に移動済み)
- 303 See Other (他を参照)
- 304 Not Modified (変更なし)
- 305 Use Proxy (プロキシ使用)

• クライアント・エラー (4xx)

このクラスのコードはクライアント・エラーを示します。現在定義済みのコードは、次のとおりです。

- 400 Bad Request (正しくない要求)
- 401 Unauthorized (無許可)
- 402 Payment Required (支払い必須)
- 403 Forbidden (禁止)
- 404 Not Found (発見不能)
- 405 Method Not Allowed (非許可メソッド)
- 406 Not Acceptable (受け入れ不能)
- 407 Proxy Authentication Required (プロキシ認証必須)
- 408 Request Timeout (要求タイムアウト)
- 409 Conflict (矛盾)
- 410 Gone (消滅)
- 411 Length Required (長さ必須)

- 412 Precondition Failed (前提条件失敗)
- 413 Request Entity Too Large (要求エンティティー過大)
- 414 Request-URI Too Long (要求 URI 長超過)
- 415 Unsupported Media Type (非サポート・メディア・タイプ)
- サーバー・エラー (5xx)

このクラスのコードはサーバー・エラーを示します。現在定義済みのコードは、次のとおりです。

- 500 Internal Server Error (サーバー内部エラー)
- 501 Not Implemented (非インプリメント)
- 502 Bad Gateway (正しくないゲートウェイ)
- 503 Service Unavailable (サービス利用不能)
- 504 Gateway Timeout (ゲートウェイ・タイムアウト)
- 505 HTTP Version Not Supported (非サポート HTTP バージョン)

12.3.2.9 アクセス認証

HTTP は認証メカニズムを備えており、サーバーはこれを使って、リソースへのアクセス許可とそれらのリソースを使用できるクライアントを定義することができます。認証メソッドは次のいずれかになります。

• 基本認証方式

基本認証はユーザー ID とパスワードに基づいて行われます。この認証方式ではサーバーは、ユーザー ID とパスワードが検証できた場合だけ接続を許可します。基本認証ではユーザー ID とパスワードの暗号化は行われません。これらは base64 フォーマットでエンコードされます (432 ページの 11.2.3.5 『base64 エンコード方式』を参照)。そのため SSL または TLS の使用を強くお勧めします。

• ダイジェスト認証方式

ダイジェスト認証方式は HTTP の拡張であり、RFC 2617 に記述されています。この認証方式では、ユーザー ID とパスワードのハッシュ値を含むダイジェストがサーバーに送信されます。サーバーは同様のダイジェストを計算し、2 つのダイジェストが等しければ、保護リソースへのアクセスを認可します。ダイジェスト認証が使用可能になっている場合、ネットワーク上で送信されるものは、暗号化された形式のパスワード (該当する鍵があれば暗号化解除できる可能性がある) だけではなく、パスワードの 1 つのハッシュ値 (暗号化解除できない) も送信されることに注意してください。したがってダイジェスト認証のセキュリティは、base-64 でエンコードされたパスワードの場合よりも高水準になります。ただし残念ながら、ダイジェスト認証はまだすべてのブラウザでサポートされているわけではありません。

12.3.2.10 コンテンツ・ネゴシエーション

さまざまなタイプのデータの最適な処理方法を見つけるためには、特定のエンティティー本体の実際の表記についてのネゴシエーションが必要です。

ネゴシエーションには次に示す 3 つのタイプがあります。

- サーバー主導型ネゴシエーション

応答の表記はサーバーに置かれているアルゴリズムに従って決定されます。

- エージェント主導型ネゴシエーション

応答の表記はクライアントに置かれているアルゴリズムに従って決定されます。

- 透過的ネゴシエーション

これはサーバー主導型ネゴシエーションとクライアント主導型ネゴシエーションの組み合わせです。これが可能なのは、使用可能なすべての表記のリストがキャッシュに含まれている場合です。

12.3.2.11 HTTP のキャッシング

HTTP の最も注目すべき機能はキャッシング機能です。HTTP は分散情報ベースのプロトコルであるため、キャッシングによってパフォーマンスを大幅に向上させることができます。HTTP 1.1 プロトコルにはキャッシングを効率的に適切に使用できる機能が多数含まれています。

多くの場合、クライアント要求とサーバー応答は相応の時間キャッシュに入れられて、その後のそれらに対応する要求の処理に使用されます。応答がキャッシュに入っていてそれが正確なものであれば、サーバーに別の応答を要求する必要はありません。この方法によってネットワーク帯域幅の所要量が減少するだけでなく、速度が増加します。そのためのメカニズムとして、サーバーが応答メッセージが有効な最小時間を見積もります。つまり、特定の応答メッセージの有効期限がサーバーによって決定されるということです。そのため、その期限内はサーバーを参照しなくてもメッセージを使用することができます。

その期限を超えているときに、その応答メッセージが必要な場合を考えてみます。有効期限後はメッセージ内のデータは変更されている可能性があります（変更されていない可能性もあります）。データが変更されているかどうかを確認できるように、妥当性検査メカニズムが次のように定義されています。

有効期限メカニズム

データが新しいものかどうかを決めるには、有効期限を判別する必要があります。多くの場合、起点サーバーは特定の応答メッセージの有効期限をそのメッセージの中で明示的に定義します。そしてその有効期限内は、後続

の要求についてはキャッシュからキャッシュ・データが送信されます。

起点サーバーが有効期限を定義していない場合は、有効期限（最終変更日時）を合理的に見積もったり計算するためのメソッドがいくつかあります。その期限はサーバーが発信したものではないため、それらのメソッドは慎重に使用する必要があります。

妥当性検査メカニズム

有効期限を超えると、データが古くなっている可能性があります。応答メッセージの妥当性を確認するには、応答メッセージがまだ使用可能かどうかを、キャッシュが起点サーバー（または新しい応答が入っている中間のキャッシュ）に確認する必要があります。HTTP 1.1 にはそのための条件付きメソッドがあります。

起点サーバーは完全応答を送信するときに、ある種のバリデーターをそのメッセージに付加します。これはその後ユーザー・エージェントやプロキシ・キャッシュによって、キャッシュ・バリデーターとして使用されます。クライアント（ユーザー・エージェントまたはプロキシ・キャッシュ）は、キャッシュ・バリデーターを付加した条件付き要求を生成します。次にサーバーはそのメッセージを評価して、エンティティー本体ではなく特殊コード（通常は 304 (Not Modified (変更なし))) で応答します。そうでない場合、サーバーは完全応答（エンティティー本体を含む）を送信します。この方法によって、バリデーターが一致しないときに余分に往復しないで済み、またバリデーターが一致したときも完全応答を送信しないで済みます。

HTTP のキャッシングの詳細については、RFC 2616 を参照してください。

12.4 コンテンツ

Web サーバーは、静的または動的（呼び出されるプログラムによって生成される）コンテンツのサービスを行うことができます。このセクションでは、コンテンツの提供に使用されたり、Web サーバーと、通常はクライアント（た

たとえば Web ブラウザー)には直接アクセスできないアプリケーション・サーバーとの間の相互作用を容易にするために使用される、一般的なテクノロジーをいくつか解説します。

12.4.1 静的コンテンツ

通常、静的コンテンツつまり静的ページは、あまり変わらない、そしてクライアントの入力に依存しない、URLに関連したデータから成ります。Web ページは度々更新されることがあるため、静的コンテンツがまったく変わらないとは言えません。このコンテンツは通常は HTML や XML などのマークアップ言語の形式になっています。

12.4.1.1 ハイパーテキスト・マークアップ言語 (HTML)

HTML は Web の大きな魅力の 1 つです。HTML には設計済みの一連のタグがあり、すべての Web ブラウザーおよび Web サーバーはそれらのタグを理解できなければなりません。新しい機能が HTML に追加されていますが、従来の Web ブラウザーではそれらをサポートしていないことがあります。これらのタグは装置独立です。パーソナル・コンピューター、AIX または UNIX マシン、またはメインフレームから同一の文書を送信することができます。そしてどのクライアント・マシン上の Web ブラウザーでも HTML タグを理解でき、データ・ストリームをビルドしてそれをターゲット装置に表示することができます。HTML タグは、ヘッダー、段落、本文スタイル、リストなどの Web 文書の基本エレメントを記述するものです。これらはテーブルを作成したり、フォーム、スクリプト、Java アプレットなどの対話式エレメントを組み込むためのより精巧なタグもあります。

文書作成者やプログラマーが HTML をマスターすると、どのマシン上のどのオペレーティング・システムでも、そこに Web ブラウザーが搭載されていれば、そのスキルを適用できます。

HTML はハイパーテキストをサポートしているため、文書作成者は HTML を使って他の HTML 文書へのリンクを組み込むことができます。それらの文書はオリジナルと同じマシン上にあることもあれば、地球の反対側の別のネットワークのマシン上にあることもあります。これが HTML リンクのパワーです。

12.4.1.2 Extensible Markup Language (XML)

Extensible Markup Language (XML) では、コンピューターに保管される XML 文書と呼ばれるデータ・オブジェクトのクラスを記述し、そしてそれらのオブジェクトを処理するプログラムの動作を必要部分に記述します。XML は SGML のアプリケーション・プロファイル、あるいは制限付きフォームです。XML のゴールは、現在 HTML で可能なように、Web 上で汎用 SGML が提供され、受信され、そして処理されるようにすることです。XML は、インプリ

メンテーションの容易さと、SGML と HTML の両方とのインターオペラビリティを目的に設計されています。

12.4.2 クライアント・サイドの動的コンテンツ

クライアントの機能の拡張が進み、クライアントのサイドで動的コンテンツを利用するテクノロジーが開発されました。

12.4.2.1 プログラムとアプレット

HTML (Web) ページ内部から開始される Java プログラムを Java アプレットと呼ぶのに対し、ローカル・システム上でコマンド行やその他の方法で実行されるものを Java プログラムと呼びます。アプレットは Web ブラウザーによってサーバーからダウンロードされ、定義によってローカル・システムのリソースの使い方がいくらか限定されます。

Java アプレットは最初はローカル (その Java 仮想マシン (JVM) の外側) のものへのアクセスは何もサポートされておらず、ダウンロード元のサーバーとの通信ができるだけでした。Java 1.1 になってアプレットにセキュリティ・キーと証明書で署名できるようになったため、アプレットの認証が可能になりました。したがってアプレットは、ファイル・システムなどのローカル・リソースへのアクセスが許可され、また他のシステムと通信を行うことができます。

12.4.2.2 JavaScript

JavaScript は HTML を拡張したプログラム言語であり、Netscape によって開発されました。これはオブジェクト・ベースの単純な言語で、Java と互換性があります。JavaScript プログラムはソースが HTML 文書に直接埋め込まれます。このプログラムで、フォーム、ボタン、および本文エレメントの振る舞いを制御することができます。これを使って Web ページの各エレメントでの動的な動作を作成します。またこれを使って、フィールドにエラー検査ルーチンを組み込んだフォームを作成することもできます。

12.4.3 サーバー・サイドの動的コンテンツ

クライアント・サイドの動的コンテンツの生成を補完するもう一方のものとして、当然ですが、サーバー・サイドのコンテンツの生成があります。このテクノロジーを使用すると、サーバー・サイドの必要な機能を実行することによって、サーバーの処理能力とオブジェクトに関する要求の調整機能を活用することができます。

12.4.3.1 コモン・ゲートウェイ・インターフェース (CGI)

コモン・ゲートウェイ・インターフェース (CGI) は、Web サーバーが、ファイルを検索するのではなく、Web サーバー管理者によって指定されたプログラムを実行できる手段です。CGI プログラムを使用すると、Web サーバーは

動的応答 (通常はクライアントの入力に基づく) を生成することができます。普及している多くの Web サーバーが CGI をサポートしており、また CGI とのインターフェースを持つプログラムの開発にはさまざまなプログラム言語を使用できます。ただし CGI プログラムのプラットフォーム間の移植は、Perl を使用しない限り容易ではありません。

12.4.3.2 サーバー固有 API

Web サーバーの中には固有の API を提供しているものもあり、開発者はこれを使って、特定のイベント時に特殊な目的のために呼び出すことができるプログラムを作成できます。通常、これらの API は非常に強力ですが、Web サーバー・プラットフォーム間の移植性はありません。最も普及しているサーバー固有 API は、Netscape Server API (NSAPI) と Microsoft Internet Information Server API (ISAPI) です。

12.4.3.3 サーブレット

Java アプレットは、クライアントやネットワーク上のリソースを使わないようにするために、ダウンロードしてクライアント上で開始するのではなくサーバー上で実行することができます。そのようなプログラムをサーブレットと呼びます。このメソッドではかなり強力なサーバーが必要になりますが、ネットワーク・コンピューターなどメディアがないシステム的环境 (921 ページの付録 A『プラットフォームの実装』を参照) には非常に適しています。このメソッドは通常、プラットフォーム間の移植性が非常に高く、処理オーバーヘッドが少なく済みます。

12.4.3.4 サーバー・サイド・インクルード (SSI)

これは、Java が使用可能な Web サーバー (サーブレット・エンジンを備えている Web サーバー) が、文書がクライアントのブラウザに送信されるたびに、HTML ファイルのセクションをそれに代わる動的な部分に変換するテクノロジーです。この動的な部分は、該当するサーブレットを呼び出して必要なパラメーターを渡します。置き換えはサーバーで行われますが、クライアントには完全に透過的です。このテクノロジーを使用するページには、.html (または .htm) の代わりに .shtml という拡張子が付きます。

12.4.3.5 Java Server Pages (JSP)

これは動的コンテンツがある HTML (または XML などの他のマークアップ言語) ページを生成するときに使いやすいソリューションです。JSP ファイルには、HTML タグ、NCSA タグ (サーバー・サイド・インクルードのインプリメントの最初のメソッドだった特殊タグ)、<SERVLET> タグ、および JSP 構文の組み合わせから成ります。JSP ファイルの拡張子は .jsp です。JSP には多くの利点がありますが、その中の 1 つに、プログラマーが Web ページ内の HTML コーディングとビジネス・ロジックを実際に分離できるという利点があります。JSP を使って、サーブレット、JavaBean (再使用可能な Java オブジェクト)、および Java ベースの Web アプリケーションなど、再使用可

能なコンポーネントにアクセスすることができます。JSP は Web ページ内へのインライン Java コードの組み込みもサポートしています。JSP は通常はサーブレットにコンパイルされて実行されます。

12.4.4 オブジェクト

Web のコンテンツの開発における主な推力の 1 つはオブジェクトの使用です。オブジェクトを使うと一般に、コードの再利用が促進されるため、アプリケーション開発のコストと手間を減らすことができます。また、特定のオブジェクトの状態を変更する操作を使用可能にすれば、異なるプロセス（およびマシン）間での連携と調整が可能です。つまり、オブジェクトという用語は再利用可能なデータ構造と機能のインプリメンテーションを表現する場合に使われることがあります。それらのデータ構造と機能のインスタンス化を表現する場合にも使うことができます。

12.4.4.1 JavaBean

JavaSoft にいる JavaBean の考案者たちによれば、JavaBean はビルダー・ツールを使ってビジュアルに扱うことができる再使用可能なソフトウェア・コンポーネントです。JavaSoft の定義では広範囲のコンポーネントを bean として扱うことができます。

JavaBean はビジュアル・コンポーネントの場合があり、たとえばボタンや入力フィールド、あるいはスプレッドシート・アプリケーション全体があります。また JavaBean はノンビジュアル・コンポーネントの場合もあり、たとえば従業員の給料支払小切手、銀行口座、あるいは信用度格付けコンポーネント全体の処理などのカプセル化ビジネス・タスクやエンティティーがあります。ノンビジュアル bean にもアイコンまたは名前（あるいはその両方）などのビジュアル表記があり、視覚的な操作が可能です。そのビジュアル表記はアプリケーションのユーザーには表示されないことがあります。ノンビジュアル bean は画面に示されるため開発者はそれら进行处理することができます。

JavaBean の取り扱いと再使用が可能なのは、それらが標準化された方法で作成された場合だけです。bean を作成するための JavaBean API が JavaSoft から提供されています。これは Java のソフトウェア・コンポーネント・モデルを定義するアーキテクチャーです。JavaBean アーキテクチャーには次の 4 つの重要な利点が含まれています。

- コンポーネント細分度の広範囲のサポート。さまざまな形状とサイズの bean が可能です。
- 移植性。この API はプラットフォームに依存しません。たとえば Windows のもとで開発されたノンビジュアル・コンポーネントの bean は、Windows、AIX、Solaris、または z/OS、OS/400 のどれで実行しても同じ動作になります。

- 一様で高品質な API。理想としては、Java をサポートするすべてのプラットフォームが JavaBean API 全体をサポートすることになります。
- 単純さ。この API は単純、汎用的、そしてコンパクトであるため習得しやすく、容易に使い始めることができます。

JavaBean API では、bean の特徴的な特性、特に bean の画面上でのルック・アンド・フィールを定義します。

12.4.4.2 CORBA

共通オブジェクト・リクエスト・ブローカー・アーキテクチャー (CORBA) は、コンピューターが他のコンピューターにあるオブジェクトのメソッドをリモートで呼び出すような分散アプリケーションのための標準です。CORBA では、言語、ロケーション、コンピューター・アーキテクチャーにかかわらず、オブジェクトとアプリケーションの相互接続が可能です。

アプリケーションはオブジェクト・リクエスト・ブローカー (ORB) を使うことにより、分散オブジェクトを使用できるようになります。ORB は、リモート・オブジェクト要求 (たとえばメソッド呼び出し) の該当するサーバー・オブジェクトへの転送、要求のディスパッチ、および結果 (たとえばメソッド戻り値) の戻しを透過的に行います。

Internet Inter-ORB Protocol (IIOP) は CORBA 仕様を基にした通信プロトコルで、オブジェクト・マネージメント・グループ (OMG) によって規定されています。オブジェクト・マネージメント・グループの詳細については、次の URL の OMG ホーム・ページを参照してください。

<http://www.omg.org/>

12.4.4.3 Enterprise JavaBean

Enterprise JavaBean (EJB) は他の EJB や他の Java コンポーネントと結合して、分散複数層アプリケーションを作成できる Java コンポーネントです。図 179 は、EJB 環境を示したものです。サーブレットや Java アプリケーションなどの EJB クライアントは、リモート・メソッド呼び出し (RMI) を使って EJB サーバーと対話して、調整を行うオブジェクト (EJB) にアクセスします。実際のオブジェクト・データはいずれかのデータ・ソースに含まれています。

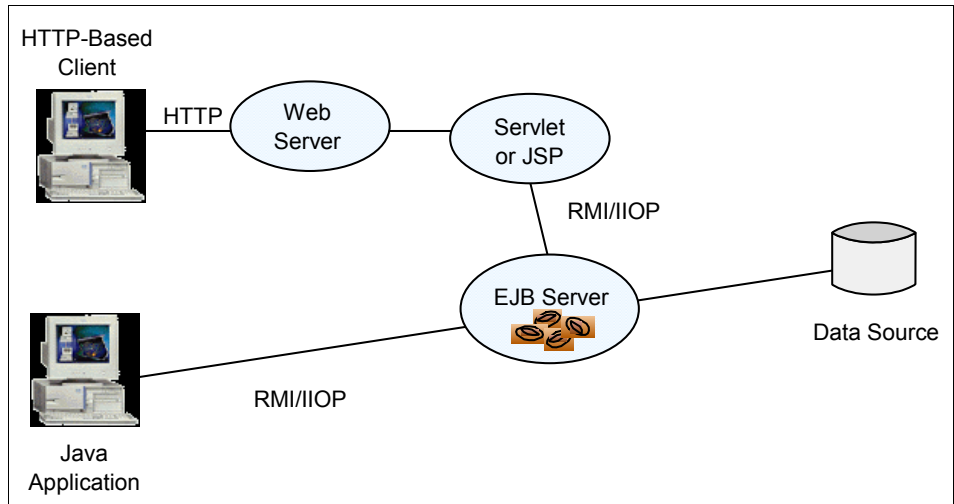


図 179 Enterprise JavaBean 環境

リモート・メソッド呼び出し (RMI)

RMI は、異なるコンピューターに置かれている Java オブジェクト間の通信のための標準プロトコルです。RMI は、クライアントおよびサーバー・アプリケーションが、JVM を実行するクライアントおよびサーバーの分散ネットワーク上でメソッドを呼び出す方法を規定しています。これを使用すると、リモート RMI オブジェクトのメソッドを、ローカル Java オブジェクトの場合のように呼び出すことができます。RMI はプラットフォーム間で共通ですが、言語間の共通性はありません (Java のみ)。

Sun Microsystems は最近 IBM などとともに、より移植性が高いバージョンの RMI を開発しました。そこではオブジェクト・マネージメント・グループ (OMG) の Internet Inter-ORB Protocol が使われています。J2EE を配置して CORBA システムと相互運用できるようにするには IIOP が必要です。

EJB のタイプ

Enterprise JavaBean (EJB) には次の 2 つのタイプがあります。

- Entity EJB: 永続データとそのデータを扱う関連メソッドをカプセル化します。カプセル化された永続データはデータベースやファイル・システムなどのデータ・ソースに保管されます。Entity Bean へのアクセスはほとんどの場合、トランザクショナルな方法で行われます。Entity Bean のインスタンスは固有であり、複数のユーザーがアクセスすることができます。たとえば、銀行口座に関する情報は Entity Bean にカプセル化されます。口座 Entity Bean には、口座 ID、口座タイプ (小切手または預金)、差引

残高変数、およびそれらの変数を扱うメソッドが含まれることが考えられます。

- Session EJB: 特定の EJB クライアントに関連した一時 (非永続) データをカプセル化します。Entity Bean でのデータと違い、Session Bean でのデータは永続データ・ソースに保管されません。またそのデータが失われても害はありません。ただし Session Bean は、基礎を成すデータベースのデータを更新することができます。この場合通常は、Entity Bean をアクセスすることにより行います。Session Bean もトランザクションに加わることができます。

作成時に Session Bean のインスタンスが同じときでも、Session Bean によっては、自身を固有にする半永続データを、そのライフ・サイクルの中の特定の時点で保管することができます。Session Bean は常に単一のクライアントと関連付けられています。したがって並行呼び出しをしようとすると、例外がスローされることになります。

たとえば 2 つの銀行口座間での預金の振替に関連したタスクは、Session Bean でカプセル化することができます。そのような振替 Session Bean は、口座 Entity Bean のインスタンスを (口座 ID を使用して) 2 つ見つけ出し、指定された金額を一方の口座から減算し、次にそれと同じ金額をもう一方の口座に追加します。

12.4.5 IBM Web アプリケーション・サーバーでのコンテンツの開発

強力な Web アプリケーション・サーバーやソリューションを開発し最適化できるように、IBM は特に次のようなさまざまなサーバー・プラットフォーム上の製品を提供しています。

WebSphere Application Server	Java サブレットとオープン・スタンダード (XML、CORBA、JDBC など) に基づいたアプリケーション・サーバー環境。WebSphere Application Server を使って開発が可能であり、またエンタープライズ・トランザクション処理へのオブジェクト・コンポーネントの配布を単純な Web パブリッシュで行うことができます。WebSphere Application Server は、IBM HTTP Server、Apache、Netscape Enterprise Server、および Microsoft Internet Information Server など、既存の Web サーバーの能力を活用します。
WebSphere Edge Server	拡張キャッシング、フィルター操作、サーバー・モニター、プロキシ、スケーラビリティ、および高可用性の機能性を

Web サーバーに提供します。その機能性の中には、Java Server Page (JSP) テクノロジーおよびサーブレットの結果をキャッシュする機能が含まれています。さらに、リバース・プロキシのキャッシング (Web サーバーの応答時間を改善する) とフォワード・プロキシのキャッシング (ネットワーク帯域幅の所要量を低減する) の両方を備えています。

Lotus Domino

ドミノはメッセージングとコラボレーションの Web アプリケーションであり、またワークフロー・サーバー・プラットフォームです。サーバー (Notes、HTTP、NNTP、SMTP、POP、IMAP、LDAP)、クライアントおよびセキュリティー (SSL、階層証明書ベースの認証、マルチレベル・アクセス制御リストでの許可) の各機能、および連携アプリケーションのための一貫性のあるユーザーおよび開発のインターフェースなど、豊富に備えています。

IBM e-business Web サーバーの詳細については、次の URL で参照してください。

<http://www.software.ibm.com/webserver>

12.5 参照

インターネットのプロトコルおよびアプリケーションの詳細については、次の RFC を参照してください。

- RFC 1945 – Hypertext Transfer Protocol – HTTP/1.0
- RFC 2109 – HTTP State Management Mechanism
- RFC 2396 – Uniform Resource Identifiers (URI): Generic Syntax
- RFC 2616 – Hypertext Transfer Protocol – HTTP/1.1
- RFC 2617 – HTTP Authentication: Basic and Digest Access Authentication
- RFC 2822 – Standard for the Format of ARPA Internet Text Messages

第 13 章マルチメディア・プロトコル

デスクトップ・コンピュータで活用できる処理能力が向上したため、広範囲のマルチメディア・アプリケーションが開発されてきました。このようなアプリケーションが出現したため、既存のネットワーク・インフラストラクチャーは強化され、ビデオ・ベースや音声ベースのアプリケーションをエンド・ユーザーに提供できるようになりました。ネットワークは、従来のデータ伝送を支援するためだけに使われるものではなくなりました。

このようなアプリケーションには、両方向のビデオ会議、音声ブロードキャスト、ホワイトボード・コラボレーション、対話式トレーニング、および IP テレフォニーのために、強化された機能が備えられています。このようなアプリケーションを使うことにより、ピアとピアの間あるいはクライアントとサーバー間のネットワークを通じ、ビデオおよび音声ストリームがやり取りされます。

この章では、これらのアプリケーションを支援するために使われる、2 つのピア・プロトコルについて説明します。Real-Time Transport Protocol (RTP) および Real-Time Control Protocol (RTCP) は、マルチメディア・アプリケーションのトラフィック・フローを同期させて制御するときに使われます。章の最後には、IP テレフォニー規格の最新情報の分析を記載しています。このような規格を使うアプリケーションは、サービスを提供するために、RTP および RTCP に大きく依存しています。

13.1 リアルタイム・プロトコル：RTP および RTCP

リアルタイム・トランスポート・プロトコル規格は、Internet Engineering Task Force (IETF) 内の Audio-Video Transport Working Group によって開発されました。この基本規格については、RFC 1889 と RFC 1890 に述べられています。

アプリケーションでリアルタイム・サービスを利用する場合、以下の 2 つのプロトコルを実装する必要があります。

- Real-Time Transport Protocol (RTP) では、リアルタイム・データ・パケットをトランスポートすることができます。
- RTP Control Protocol (RTCP) は、既存の RTP セッションに対して提供されるサービスの品質をモニターします。

これらの規格の最終バージョンは、1996 年 1 月に公開されています。これには、すべてのタイプのリアルタイム・アプリケーションで共通に求められる機能が詳述されています。新しいリアルタイム・アプリケーションを順応させるため、アーキテクチャーは意図的に未完成のままにされています。従来

の Protocol とは異なり、RTP は、必要に応じてヘッダーを変更したり、ヘッダーに追加することで作り替えられます。このため、この Protocol には新しい音声やビデオの規格を取り入れやすくなっています。

13.1.1 Real-Time Transport Protocol (RTP)

RTP には、マルチメディア・データ・ストリームの同期を取る時に必要なトランスポート機能が実装されています。

ビデオと音声コンポーネントの両方を使うアプリケーションを考えてください。RTP を使うことで、それぞれのビデオおよび音声ストリームに関連付けられたパケットをマークできます。これにより、受信側ホストでもストリームを同期化させることができます。図 180 には、マルチメディア伝送での RTP の運用が示されています。音声およびビデオ・データは、送信側から受信側へ伝送される前に、RTP パケットにカプセル化されます。

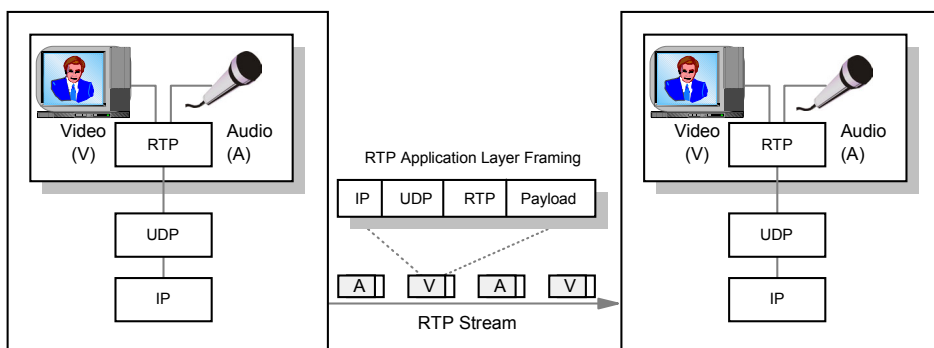


図 180 マルチメディア環境での RTP の運用

マルチメディア・アプリケーションが RTP サービスを利用しない場合、受信側は、対応する音声およびビデオ・パケットを関連付けることができない可能性があります。これは、マルチメディア・セッション時に生み出されるネットワーク・パフォーマンスのレベルによって異なります。環境内の輻輳条件や一時条件が原因で、伝送時にパケットが失われたり、再配列される場合もあります。そのために、時間の長短を問わず、パケットの送達が遅れる可能性があります。このような性質があるため、一般的なマルチメディア・アプリケーションに品質問題を引き起こします。

規格では、RTP は適切な任意のネットワークまたはトランスポート・Protocol で使えることが指定されています。実際には、マルチメディア・アプリケーションは、一般的に、RTP を UDP と共に使います。これにより、アプリケーションは、UDP に備えられている多重なサービスや、チェックサム・サービスを使うようになります。RTP は、マルチキャスト・アプリケーションをサポートするように実装されることもあります。

RTP プロトコル単体には、送達や他のサービス品質機能を保証するメカニズムはありません。この規格では、順不同のパケット送達が妨げられることはありませんし、基礎となるネットワークが信頼できるのでパケットが順番に送達されると見なされるわけでもありません。また、RTP では、ネットワーク輻輳の発生を妨げることもありません。各アプリケーションの設計担当者は、これらのレベルのサービスを受け入れるかどうかを決める必要があります。

13.1.1.1 RTP ヘッダー形式

RTP パケットのヘッダーは、図 181 に示されているような形式になります。

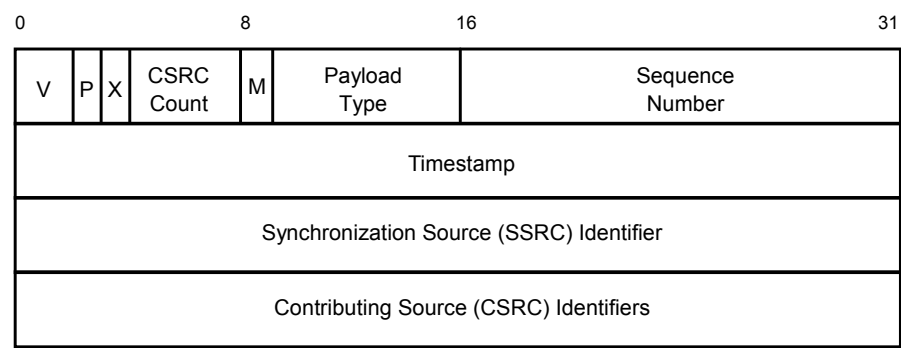


図 181 RTP ヘッダー・フィールド

最初の 12 オクテットは、各 RTP パケットで必須なものです。CSRC ID のリストは、ミキサーによって挿入される場合にだけ示されます (479 ページの 13.1.4 『RTP 変換プログラムおよびミキサー』を参照)。それぞれのフィールドは、以下のように解釈されます。

- V: RTP のバージョンを示します。
- P: 埋め込みビットを含みます。このフィールドを設定する場合、パケットには一連の埋め込みオクテットが含まれますが、それらはペイロードの一部とはなりません。この機能は、特定の暗号化アルゴリズムによって使われます。
- X: 拡張子ビットを含みます。このフィールドを設定する場合、固定ヘッダーの後にヘッダー拡張子が付けられます。
- CSRC カウント: このフィールドには、固定ヘッダーの後に続く付与ソース ID の番号が含まれます。
- M: このフィールドでは、パケット・ストリームで重要なイベント (すなわち、フレーム境界) をマークできます。

- **ペイロード・タイプ**: RTP パケットでのペイロードの形式を指定します。RTP 送信側は、所定の時刻に 1 つの RTP ペイロード・タイプを送出します。ペイロード・タイプ・フィールドの内容の詳細については、472 ページを参照してください。
- **シーケンス番号**: シーケンス番号は、パケット・シーケンスを復元したり、パケットの消失を検出するために、受信側によって使われます。シーケンス番号フィールドの内容の詳細については、474 ページを参照してください。
- **タイム・スタンプ**: タイム・スタンプには、ペイロード・データをサンプル化したときの時刻を表す値が示されます。タイム・スタンプ・フィールドの詳細については、474 ページを参照してください。
- **SSRC ID**: 同期ソースは、RTP ホスト用にランダムに選んだ ID です。同じソースに含まれるパケットには、すべて同じ SSRC ID が付けられています。同じ RTP セッション内の各デバイスには、固有の SSRC ID が必要です。これにより、受信側は、パケットを再生用にグループ化することができます。
- **CSRC ID**: 付与ソース・フィールドには、現在のパケットのペイロード用のソースのリストが含まれます。このフィールドは、ミキサー (479 ページの 13.1.4 『RTP 変換プログラムおよびミキサー』を参照) が別のストリームのパケットを結合する場合に使われます。このフィールドに入る情報を見ることにより、受信側は送信元を識別することができます。

13.1.1.2 プロトコル・サービス

RTP には、リアルタイム・データを伝送するアプリケーション用に、終端間トランスポート・サービスがあります。このサービスには、以下の機能が含まれます。

- ペイロード・タイプの識別
- シーケンス番号付け
- タイム・スタンプ付け

ペイロード・タイプの識別

RTP パケットには、音声またはビデオ・データ・ストリームの一部を含められます。これらのストリームを区別するために、送信側アプリケーションの RTP ヘッダーには、ペイロード・タイプ ID が組み込まれています。この ID は、ペイロードを作成するときに使われる、特定エンコード方式を示しています。受信側アプリケーションはこの ID を使い、適切なデコード・アルゴリ

ズムを決定します。表 20 には、サポートされているペイロード・タイプが示されています。

表 20 RTP ペイロード・タイプ

音声		ビデオ	
ペイロード・タイプ	エンコード名	ペイロード・タイプ	エンコード名
0	PCMU	24	未割り当て
1	1016	25	CelB
2	G726-32	26	JPEG
3	GSM	27	未割り当て
4	G723	28	nv
5	DVI4 (8 KHz)	29	未割り当て
6	DVI4 (16 KHz)	30	未割り当て
7	LPC	31	H261
8	PCMA	32	MPV
9	G722	33	MP2T
10	L16 Stereo	34	H263
11	L16 Mono	35-71	未割り当て
12	QCELP	72-76	予約済み
13	予約済み	77-95	未割り当て
14	MPA	96-127	動的
15	G728	動的	BT656
16	DVI4	動的	H263-1998
17	DVI4	動的	MP1S
18	G729	動的	MP2P
19	予約済み	動的	BMPEG
20-23	未割り当て		
動的	GSM-EFR		
動的	L8		
動的	RED		
動的	VDVI		

このリストは、Internet Assigned Numbers Authority (IANA) によって保守されます。これは、新しい音声およびビデオ形式が開発されたら拡張することができます。RTP は、主にマルチメディア・データをサポートするように設計されたものですが、音声およびビデオ・トラフィックに限定されているわけではありません。連続したデータ・ストリームを生成するすべてのアプリケーションが、RTP サービスを利用できます。

複数のエンコード方式で、1つの動的ペイロード・タイプを指定できます。このようなエンコード方式を動的にデプロイするシステムは、ペイロード・タイプ ID に一致します。

シーケンス番号付け

シーケンス番号は、元のパケット順序を復元するために、受信側 RTP ホストによって使われます。受信側は、このフィールドの情報を使い、パケットの消失を検出できます。

シーケンス番号は、送信した RTP データ・パケットごとに、1つずつ増えていきます。シーケンス番号の初期値は、ランダムに決められます。そのため、暗号化に対するハッキング攻撃はさらに困難になります。ソース・デバイスが RTP パケットを暗号化しない場合でも、ランダムな数値が使われます。このようなパケットは、暗号化サービスを提供する変換プログラムを通過できます。

タイム・スタンプ付け

タイム・スタンプは、別のソースからのパケットを同期化するために RTP で使われます。このタイム・スタンプは、RTP データ・パケットの最初のオクテットを抽出（作成）した時刻を表します。この時刻は、単調かつ直線的に増分するクロックから得られます。タイマーのレゾリューションは、該当アプリケーションで必要な同期の正確さに応じて異なります。

複数の連続した RTP パケットに同じタイム・スタンプを付けることが可能です。たとえば、1つのビデオ・フレームを、複数の RTP パケットで伝送するときに、そのようにすることができます。これらのパケットのペイロードは同じ瞬間に論理的に生成されたため、タイム・スタンプは変わりません。タイム・スタンプの初期値は、ランダムに決められます。図 182 には、ビデオ・アプリケーションでのタイム・スタンプ生成の一例が示されています。

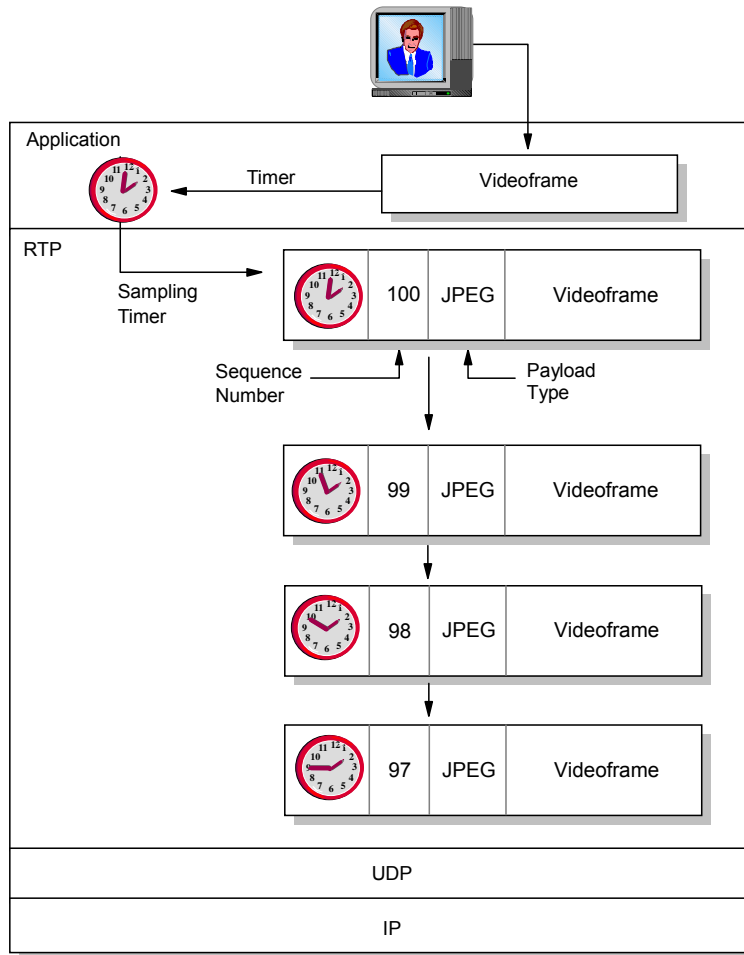


図 182 ビデオ・アプリケーションでの RTP パケット生成

13.1.2 Real-Time Control Protocol

リアルタイム送達を管理するため、多くのアプリケーションでネットワークの現行パフォーマンスについてのフィードバックを必要とします。この機能は、Real-Time Control Protocol (RTCP) によって提供されます。このプロトコルは、1つのセッション内で、制御パケットをすべての参加者へ周期的に伝送することを基本としています。

RTCP の主な機能は、RTP データ配布の品質についてのフィードバックを提供することです。この機能は、他のトランスポート・プロトコルで提供されるフローおよび輻輳制御機能と比較することができます。各受信側によって

提供されるフィードバックは、配布障害を診断するときに使われます。セッション内のすべての参加者にフィードバックを送信することにより、問題を監視しているデバイスは、問題がローカルかリモートかを判別できます。このことにより、セッション内で参加者となっていない管理エンティティー（たとえば、ネットワーク・サービス提供者）は、フィードバック情報を受け取ることも可能です。その後、ネットワーク提供者がサード・パーティー・モニターとして動作し、ネットワーク問題を診断できます。

RTCP では、通信のために UDP 接続を使います。これは、RTP プロトコルで使われる UDP 接続とは別のものです。図 183 には、RTP プロトコルと RTCP プロトコルの相関関係が示されています。

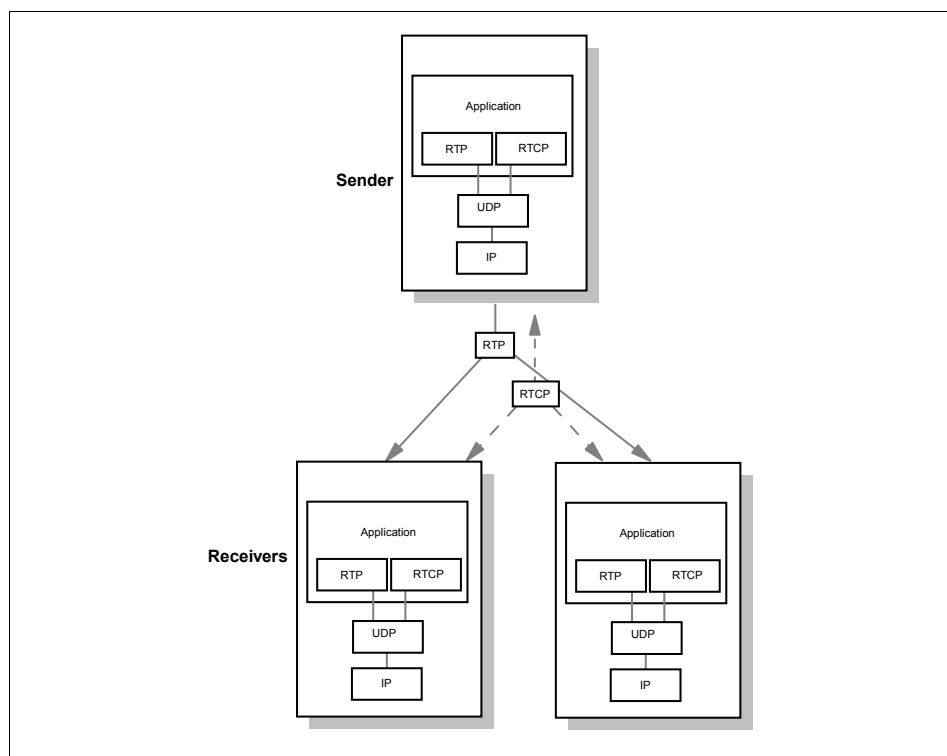


図 183 RTCP および RTP パケット送達

RTCP アーキテクチャーでは、現行パフォーマンスをレポートするときに使われる、5 つのタイプの制御情報が定義されています。

- ・ 送信側レポート：RTCP 送信側レポートは、RTP データ・ストリームのソースによって送信されます。ここには、送信側によって監視されている伝送および受信統計が示されます。このレポートは、すべての RTP セッション参加者によって処理されるマルチキャスト・パケットとして送信さ

れます。このレポートの packets 形式については、477 ページの 13.1.3 『RTCP packets 形式』のセクションで詳しく説明しています。

- 受信側レポート：RTCP 受信側レポートには、アクティブな送信側ではない参加者の受信統計が示されます。送信側レポートは、最後のレポートの発行以降にデバイスがデータ・パケットを送信した場合に発行されます。それ以外の場合は、受信側レポートが発行されます。
- ソース記述レポート：ソース記述レポートは、ローカル機能の情報を提供するために、RTP 送信側によって使われます。現在定義されているソース記述には、以下のものがあります。
 - CNAME: ソースの固有名。
 - NAME: ソースの実名。
 - EMAIL: アプリケーション・ユーザーの E メール・アドレス。
 - PHONE: アプリケーション・ユーザーの電話番号。
 - LOC: アプリケーション・ユーザーの地理上の位置。
 - TOOL: 特定のアプリケーションまたはツール名。
 - NOTE: ソースについてのその他の注意事項。
 - PRIV: 専用拡張子。
- BYE: RTCP bye メッセージは、会議を抜けるときに、ソースによって使われます。また、ミキサーがシャットダウンする場合に使われます。この bye メッセージを使い、セッションに参加しているすべてのソースに指示します。
- APP: APP パケットは、新しいアプリケーションや機能が開発されるときに試験的に使用することを目的としています。テストの後、より広範囲に使えることが認められれば、各 APP パケットを Internet Assigned Numbers Authority に登録することをお勧めします。

13.1.3 RTCP packets 形式

図 184 には、RTCP 送信側レポートの形式が示されています。このレポートは、グループ化されて 3 つのセクションに分けられています。最初のセクションには、ヘッダーが含まれます。このセクションでは、packets・タイプ、長さ、および送信側 ID を指定します。

2 番目のセクションには、送信側情報が含まれます。3 番目のセクションには、受信側レポート・ブロックが含まれます。この packets には、複数の受信側レポート・ブロックを含められます。このようにすることで、送信側は、他の送信側から受信した RTP packets からのフィードバックをレポートすることができます。

送信側レポートの形式は、明快にするために示されているものです。他の 4 つのレポート・タイプの形式は同じようなものです。

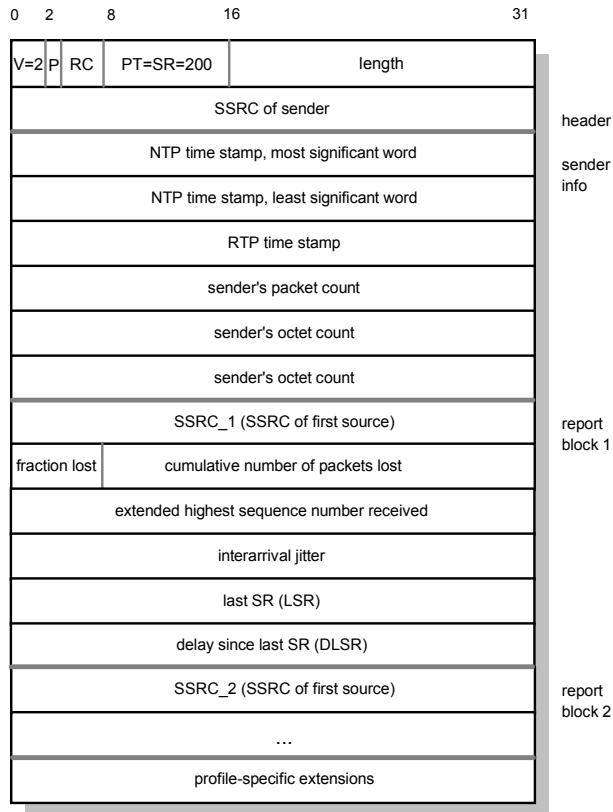


図 184 RTCP パケット形式

パケット内のフィールドは、以下の目的で使われています。

- V: このフィールドはバージョンを示します。
- P: このフィールドは、パケットの最後に別の埋め込みがあるかどうかを示します。
- RC: このフィールドには、このパケットに含まれるレポート・ブロックの数が示されます。
- PT: このフィールドには、レポート・タイプが示されます。
- 長さ: このフィールドには、パケットの長さが示されます。
- 送信側の SSRC: このフィールドには、このパケットを送信するホストの SSRC ID が示されます。

- NTP タイム・スタンプ：このフィールドには、NTP によって報告された絶対時刻が示されます。このプロトコルは、1900 年 1 月 1 日以降の秒数をカウントします。
- RTP タイム・スタンプ：このフィールドには、送信側に対応した RTP パケットからのタイム・スタンプが示されます。
- 送信側のパケット・カウント：このフィールドには、伝送を開始してから送信側によって伝送された RTP データ・パケットの合計数が示されます。
- 送信側のオクテット・カウント：このフィールドには、伝送を開始してから送信側によって伝送されたペイロード・バイトの合計数が示されます。
- SSRC_n: このフィールドには、この送信側がパケットを受信した別の RTP 送信側の SSRC ID が示されます。異なる送信側 SSRC が付けられたレポート・ブロックの数は、最後のレポート以降にこの送信側によって聞き取られたその他のソースの数によって異なります。
- 消失断片：このフィールドには、ソース SSRC_n から前の送信側レポートまたは受信側レポートが送信されてから消失した RTP データ・パケットの断片が示されます。
- 累積消失パケット数：このフィールドには、ソース SSRC_n から消失した RTP フィールドの合計数が示されます。
- 拡張された最高受信シーケンス番号：このフィールドには、ソース SSRC_n から RTP パケットで受信した最高シーケンス番号が示されます。
- 着信間ジッター：このフィールドには、該当するソースからの着信間時間の見積差異が示されます。パケットが定期的に到着する場合、ジッター値はゼロです。パケットの到着が不定期な場合、ジッター値が高くなります。
- 最終 SR タイム・スタンプ (LSR): このフィールドには、ソース SSRC_n からの最終 RTCP 送信側レポート・パケットで受信した 64 ビット NTP タイム・スタンプのうち、中間の 32 ビット分が示されます。
- 最終 SR 以降の遅延 (DLSR): このフィールドには、ソース SSRC_n から最終 SR パケットを受信するときと、1/65536 秒の単位で最新例外レポート・ブロックを送信するときの間の遅延が示されます。

13.1.4 RTP 変換プログラムおよびミキサー

RTP プロトコルでは、RTP パケット・ストリームを変更するときに、変換プログラムとミキサーを使うことができます。これらのデバイスは、マルチメディア・セッションの複数の参加者が別の形式でデータを受信する必要がある場合に使われます。

13.1.4.1 RTP 変換

RTP 変換プログラムは、RTP パケットのデータのタイプを変換するときに使われます。図 185 には、RTP 変換プログラムが使われる環境が示されています。この例では、3 つのビデオ会議ワークステーションが、高速ローカル・エリア・ネットワーク経由で MPEG トラフィックを交換しています。各ワークステーションは、1.5 Mbps のレートで MPEG データを生成しています。低速のシリアル接続で接続された別のワークステーションが、ビデオ会議に参加したい場合があります。この接続の帯域幅では、ビデオ・ストリームをサポートするのに十分ではありません。

この問題の 1 つの解決策は、すべてのワークステーションを特定のビデオ形式に変更し、トラフィックを少なくすることです (たとえば、256 Kbps での H.261 形式)。ただし、データ・ビット・レートを下げると、ビデオの全体の品質も下がります。

別の解決策では、RTP 変換デバイスを使います。この解決策では、LAN 接続ワークステーションからの MPEG データを含むパケットは、シリアル接続されたワークステーションに転送する前に、低速ビット・レートを生成する形式に変換されます。この例では、それぞれ個別の MPEG ビデオ・ストリームが、H.261 ビデオ・ストリームに変換されます。これにより、各 1.5 Mbps ストリームが 256 Kbps ストリームに下がります。その後、シリアル接続を介してそれぞれの H.261 ストリームを伝送できます。3 つの LAN 接続ワークステーションでは、相互に通信するために、引き続き高品質の MPEG 形式を使います。

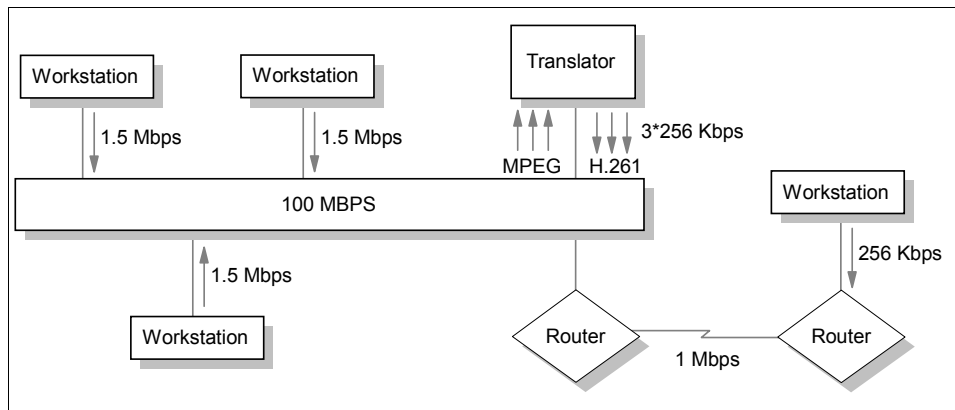


図 185 RTP 変換

変換された RTP パケットは、そのままの同期ソース ID で転送されます。これにより、変換された RTP パケットを受信する側は、元のソースを識別できます。受信側は、元のソースで使われるペイロード・タイプが分からなければ、変換プログラムの存在を検出することができません。

RTP 変換プログラムの一般的な使い方の 1 つにより、アプリケーション・レベルのファイアウォールとのインターオペラビリティが可能になります。この構成では、会議の参加者は、IP マルチキャストを経由して直接に到達することはできません。参加者は、IP マルチキャスト・パケットを通さないファイアウォールの後ろに存在します。変換プログラムは、このような参加者をサポートするときに使うことができます。2 つの変換プログラムがインストールされますが、その 1 つは、ファイアウォールの中か外にインストールされます。もう 1 つの変換プログラムでは、すべてのマルチキャスト・パケットが、1 つのセキュア・トンネルを経由して、ファイアウォールの内側にあるパートナー変換プログラムに到達するように構成されます。この変換プログラムは、情報をマルチキャスト・パケットにして、サイトの内部ネットワーク内に登録されているグループへ転送します。

13.1.4.2 RTP 混合

RTP ミキサーは、複数のデータ・ストリームを結合して 1 つの RTP ストリームにするとときに使われます。これらのデバイスは、同時に話す人が 1 人か 2 人だけである音声伝送アプリケーションをサポートするために使われます。RTP 混合は、ビデオ・アプリケーション環境では使えません。

図 186 には、RTP 混合が使われる環境が示されています。この例では、音声会議の 3 人の参加者が、1 つの 1 Mbps LAN に接続されています。各参加者は、64 Kbps のレートで PCM 音声ストリームを生成します。低速のシリアル接続で接続された 4 番目のワークステーションが、音声会議に参加したい場合があります。この接続の帯域幅では、既存の 3 つの参加者によって生成された結合 192 Kbps をサポートするのに十分ではありません。RTP ミキサーは、3 つの送信側ストリームを 1 つの 64 Kbps ストリームにマージします。これにより、新しい端末が会議に加われるようになります。

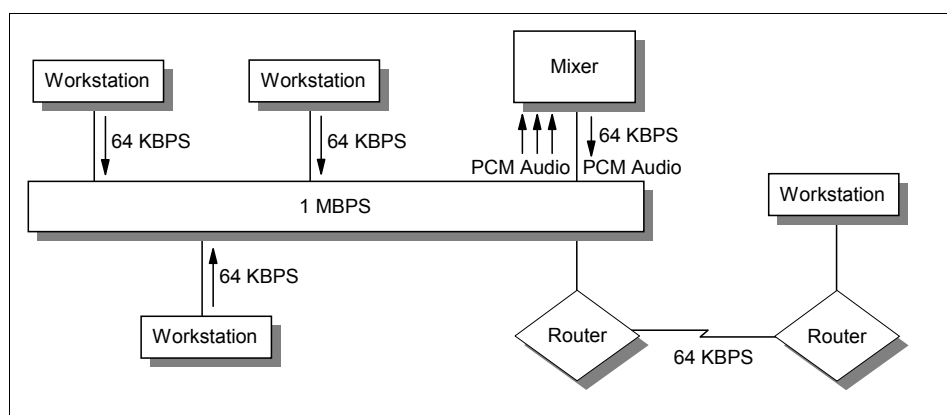


図 186 RTP 混合

図 186 に示されている例では、着信および発信パケットのペイロード・タイプは同じままです。しかし、RTP 混合と RTP 変換を同じ環境に結合することは可能です。このような結合は、図 186 に示されているワークステーションが低速モデムを介して接続されている場合に必要になります。この構成では、発信ストリームのペイロード形式は、低い帯域幅の仕様 (たとえば、8-16 Kbps を生成する GSM 音声) に変更する必要があります。

RTP ミキサーによって複数の入力ストリームが処理される場合、異なるソース・ストリームのタイミングを同期させないことも可能です。その場合、ミキサーは、1 つの出力ストリームに対して新しいタイミングを生成します。出力ストリームの SSRC フィールドには、ミキサーの SSRC ID が含まれます。受信側が音声データの元のソースを識別できるようにするため、ミキサーには、元のソースそれぞれの SSRC ID が含まれます。この情報は、RTP パケット・ヘッダーに含まれる CSRC ID リストにあります。

13.1.5 リアルタイム・アプリケーション

既存のリアルタイム・アプリケーションの多くには、ビデオ会議、IP テレフォニー、またはメディア・ストリーミングのためのサポートが備えられています。268 ページの 6.9.2 『マルチキャスト・アプリケーション』のセクションで示しているように、ほとんどの MBONE ツールは、RTP を使ってリアルタイム・データを送達します。次のリストには、別のリアルタイム・マルチメディア・アプリケーションがいくつか示されています。

- QuickTime: QuickTime は、Apple Computer, Inc. のアプリケーションで、デジタル・ビデオとメディア・ストリーミングの機能があります。ここにはライブ・ストリーミング機能があるので、ファイルをローカル環境へダウンロードする必要はなくなります。高速で高品質なブロードキャスト・パフォーマンスを実現できます。インターネットを介してコンテンツをストリーミングするために、RTP および RTSP 規格がサポートされています。
- RealAudio および RealVideo: RealNetworks, Inc. がデジタル・メディアを送達するために開発したアプリケーションです。高品質な音声およびビデオ・ストリーミングが実現します。これらのアプリケーションでは、RTP と RTSP がサポートされています。
- NetMeeting: NetMeeting は、Microsoft Corp. が開発したアプリケーションで、IP テレフォニー、ホワイトボード、テキスト・チャット、およびアプリケーションとファイルの共用が可能です。このアプリケーションでは、RTP と RTSP がサポートされています。
- CU-seeMe: CU-seeMe は、CUseeMe Networks, Inc が提供する一組のアプリケーションです。ここには、ビデオ、音声、テキスト、およびホワイトボード通信をサポートするインターネット・ビデオ・チャット・ソフトウェアがあります。リアルタイム伝送のために RTP が使われます。

- IP/TV: これは、Cisco Systems, Inc. によって提供されるクライアント / サーバー型のビデオ・アプリケーションです。このソフトウェアでは、複数の高品質ビデオおよび音声規格 (たとえば、MPEG、H.264) がサポートされています。ライブ・ビデオ、スケジュール・ビデオ、およびビデオ・オンデマンドについても、すべて IP/TV に備えられています。
- Real-Time Streaming Protocol (RTSP): RTSP は、連続した音声およびビデオの時刻同期ストリームをいくつか設定して制御します。データのソースには、ライブ・データ・フィードと保管クリップの両方を含められます。RTSP セッション時には、RTSP クライアントは、RTSP 要求を発行するために、サーバーへの多くの信頼できるトランスポート接続をオープンしたりクローズすることができます。そうしない場合でも、UDP など、接続不要のトランスポート・プロトコルを使えます。RTSP で制御されているストリームは RTP を使えますが、RTSP の操作が、このプロトコルに依存しているわけではありません。RTSP は、通常はデータ・ストリームの送達を担当しません。しかし、制御ストリームを使ったメディア・ストリームのインターリービングは可能です。

13.2 IP テレフォニー

IP テレフォニー製品 (IP 経由の音声とも言われる) には、IP ネットワークを介して音声トラフィックをトランスポートする機能があります。この機能は、インターネット、企業のイントラネット、またはローカル・エリア・ネットワークを含む、あらゆる IP ネットワーク・タイプで実装できます。このような環境では、音声シグナルは、デジタル化されて IP パケットに入れられます。その後、このパケットは IP ネットワークを介して伝送されます。

音声トラフィックをデータ・ネットワークにマージするときに必要なシグナル構造を提供する標準プロトコルが、現在開発されています。これらのプロトコルには、ユーザーを突き止め、機能をネゴシエーションし、呼び出しを確立するための機能が備えられています。

音声およびデータ・ネットワークを収束するためのドライバーがいくつか存在します。

- 企業は、大規模なデータ・ネットワークを開発し、マージしたマルチメディア・ネットワークでのコストを節約して出費を抑えることを期待しています。
- 新しいアプリケーションが音声およびデータを統合しています。これらのアプリケーションにより、既存のビジネス・プロセスが改善されます。このようなアプリケーションの例には、コンピューターでサポートされている共同作業機能、Web で利用できるコール・センター、および統合された音声および E-mail が含まれます。

13.2.1 概要

IP テレフォニーを成功させるために、いくつかの重要な問題に取り組む必要があります。いくつかの問題は、IP ネットワークが元々はデータのトランスポート用に設計されているという事実に起因しています。他の問題は、環境の発展性と拡張容易性の可能性に関係するものです。IP テレフォニー・ネットワークは、最低でも以下を実装する必要があります。

- 品質：通信品質は、最低でも、現在 Plain Old Telephone System (POTS) ネットワークで提供されているレベルと同じにしなければなりません。終端間の遅延は、知覚される品質に対して大きな影響があります。この遅延の中には、音声サンプル群を収集してエンコードするためにデバイスを呼び出すときの時間が含まれます。また、それらのエンコードしたサンプルをネットワーク経由で伝送するときに必要な時間も含まれます。遅延が大きすぎると、IP テレフォニー環境にいくつかの問題が生じる可能性があります。
 - 話す人の音声のシグナル反射により、音声エコーが生じます。これは、従来の POTS ネットワークでも生じる場合があります。しかし、IP 経由の音声 (VoIP または IP テレフォニー) ネットワークでの遅延は、一般的に POTS ネットワークでの遅延より長いため、音声エコーは、このようなネットワークで大きな問題になります。エコーは、合計遅延が 50 ミリ秒を超過すると、重大な品質問題になります。この問題によって引き起こされた影響を最低限に抑えるため、エコー取り消し技術を使う必要があります。
 - 遅延が大きくなりすぎると、送話者のオーバーラップが生じる可能性も出てきます。トラフィックの優先順位を決定する機能を使い、この問題が最小限に抑えられるように、ネットワーク遅延を制御する必要があります。
 - 音声の明瞭さも、IP ネットワークの全体のパフォーマンスによって影響を受けます。音声パケットが脱落したり到着が遅れると、知覚される明瞭度に悪影響を与える場合もあります。音声シグナルを IP パケットに変換するときに使われるデジタル化および圧縮方式も、音声の明瞭度に影響を与える可能性があります。
- 使用可能度：VoIP ネットワークの運用の機能性と簡便さは、最低でも、現在 POTS ネットワークで提供されているものと同じレベルにしなければなりません。IP テレフォニー・ネットワークは、サービスで複雑なダイヤル計画が必要な場合、呼び出しの脱落の割合が非常に大きい場合、または呼び出しを完了するためにかなりの時間がかかる場合には、一般的に受け入れられません。
- 拡張容易性：VoIP システムには、既存の公衆交換電話網 (PSTN) よりはるかに低いコストで高品質のサービスを提供する潜在能力があります。これにより、このようなサービスで高い成長率を得られる見込みがあります。

IP テレフォニー・システムは、このような潜在的な成長率をサポートできるように拡張可能でなければなりません。

- ・ インターオペラビリティおよび統合：IP テレフォニー環境では、別のベンダーによる同様の製品を運用する必要があります。これらの製品は、既存の PSTN でも機能しなければなりません。このような異なるネットワークは、これらのサービスのユーザーにとっては、単一の環境として示されている必要があります。

13.2.2 IP テレフォニー・プロトコル・スタック

IP テレフォニー環境は、多数の個別プロトコルで構成されています。これらのプロトコルは、1 つの階層方式で相互に運用され、必要なサービスを提供します。プロトコルの対話は、スタックとして表すことができます。これは、他の多くのシステム間通信を表すときに使うメソッドと似ています。

スタックを表す 1 つの方法は、プロトコルを 2 つの機能セクションに分けることです。これは図 187 で示しています。

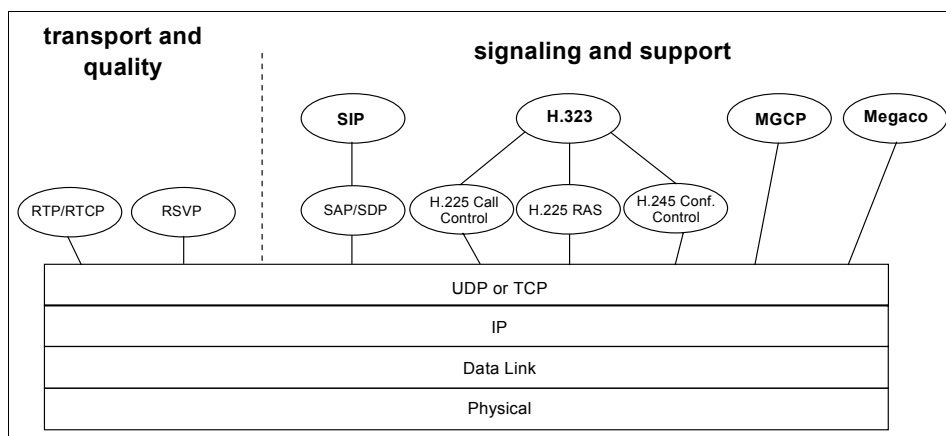


図 187 IP テレフォニー・プロトコル・スタック

プロトコル・スタックの 2 つのセクションには、以下の機能があります。

- ・ トランスポートおよび品質プロトコル：これらのプロトコルは、実際の音声トラフィックをトランスポートするときに使われます。
- ・ シグナルおよびサポート・プロトコル：IP テレフォニー・アクティビティの大部分は、シグナル・プロトコルの開発に費やされています。これらのプロトコルは、ユーザーを突き止め、機能をネゴシエーションし、音声呼び出しを管理します。現在は、以下の 4 つのシグナル・プロトコルがあります。

- International Telecommunications Union Telecommunications Standardization Sector (ITU-T) 勧告 H.323
- Session Initiation Protocol (SIP)
- Media Gateway Control Protocol (MGCP)
- ITU-T 勧告 H.248/Media Gateway Controller (Megaco)

これらのプロトコルは非互換で、しばしば冗長な機能が提供されます。これらが同じ環境で使われる場合、これらのプロトコル間の通信では、コンバーターかゲートウェイが必要になります。

この章の残りでは、現在のシグナル・プロトコルをそれぞれ説明します。

13.2.3 ITU-T 勧告 H.323

H.323 は、元々はパケット交換ネットワークを介したマルチメディア・ビデオ会議用に開発された ITU-T 規格です。この規格は後で拡張されて、IP テレフォニー要件が含まれました。本書の作成時点では、これは最も広範囲にデプロイされている呼び出し処理プロトコルです。

初期バージョンの H.323 は、非常に早い時点で開発されています。ITU-T において、作業は 1995 年 5 月に開始され、1996 年 6 月に完了しました。そのとき以来、3 つの別バージョンが公開されています。

標準の H.323 ネットワークには、以下の 4 つのコンポーネントがあります。

- 端末
- ゲートウェイ
- ゲートキーパー
- マルチポイント制御装置

13.2.3.1 端末

IP テレフォニー・ネットワークでは、端末は LAN クライアントのエンドポイントになります。H.323 端末は、他の H.323 端末、H.323 ゲートウェイ、または H.323 MCU と通信できます。

端末はすべて、音声通信をサポートしていなければなりません。ビデオおよびデータのサポートは、オプションです。H.323 は、別のタイプの端末を同時に運用するために必要な操作を指定しています。H.323 端末はすべて、セクション 489 ページの 13.2.3.6 『H.323 プロトコル・スタック』で説明されているプロトコル群をサポートしていなければなりません。

H.323 端末は、マルチポイント会話をサポートしており、いつでも会議を開始する機能があります。さらに、マルチキャスト機能があるので、複数のユーザーが、混合や交換を集約せずに、1 つの呼び出しに参加することができます。

13.2.3.2 ゲートウェイ

ゲートウェイを使うと、標準の電話で VoIP サービスを利用できます。これにより、H.323 端末と、IP ベースのネットワークか別の H.323 ゲートウェイのいずれかに接続された端末との通信が可能になります。ゲートウェイは、変換プログラムとして機能し、PSTN と IP ネットワークとの間のインターフェースとなります。

ゲートウェイには、異なるネットワークの間で、呼び出しのシグナル・プロトコルと制御プロトコルを割り当てる役割があります。さらに、ネットワーク間でのメディアを割り当てる（つまり、多重化する、レートを突き合わせる、音声をトランスコーディングする）役割もあります。

13.2.3.3 ゲートキーパー

ゲートキーパーは、H.323 環境で最も重要なコンポーネントです。特定ゲートキーパーの制御下にある H.323 端末およびゲートウェイのネットワークは、より大きな IP ネットワーク環境内で統合されたサブネットワークを形成します。ゲートキーパーの機能には、以下のものがあります。

- ディレクトリー・サーバー：この機能は、端末登録時に取得する情報を使い、H.323 別名アドレスを IP（トランスポート）アドレスに変換します。これにより、ユーザーは、システム内の他のユーザーを指す、分かりやすく、変わらない名前を持つことができます。これらの名前は、任意に付けるもので、E メールやボイス・メール・アプリケーションで使われる名前に似たものを付けることができます。
- 監視：ゲートキーパーを使い、呼び出しを確立する許可を付与できます。この機能は、帯域幅の制限を適用して、ネットワーク内での輻輳の可能性を減らすために使うことができます。
- 呼び出しシグナル：ゲートキーパーは、呼び出しのルーティング機能を実行して、補足サービスを提供することが可能です。さらに、たくさんの参加者がいる呼び出しをサポートするマルチポイント・コントローラー機能も提供可能です。
- 呼び出し管理：ゲートキーパーを使い、呼び出しの会計処理と呼び出しの管理を実行できます。

13.2.3.4 マルチポイント制御装置 (MCU)

マルチポイント制御装置 (MCU) とは、ネットワーク内のエンドポイントのことです。これには、3 つ以上の端末やゲートウェイが、1 つのマルチポイント会議に参加する機能が備えられています。MCU では、会議管理、メディア切り替え、およびマルチポイント会議が可能です。

マルチポイント会議には、以下の 3 つのタイプがあります。

- 集中化：すべての端末が、MCU に対する二地点間通信ストリームを持ちます。MCU には、会議を管理する役割があります。音声パケットを受け取り、処理し、他の終端へ送信します。
- 非集中：端末は互いに直接に通信します。MCU は、直接には関係しません。
- 混合マルチポイント：これは、集中および非集中会議の混合を表します。MCU により、各端末は境界を意識せずに会議の操作を行うことができます。

13.2.3.5 H.323 サンプル構成

図 188 には、H.323 ネットワーク構成が示されています。この IP ベースのネットワークは、H.323 から PSTN へのゲートウェイを介して PSTN へ接続されています。

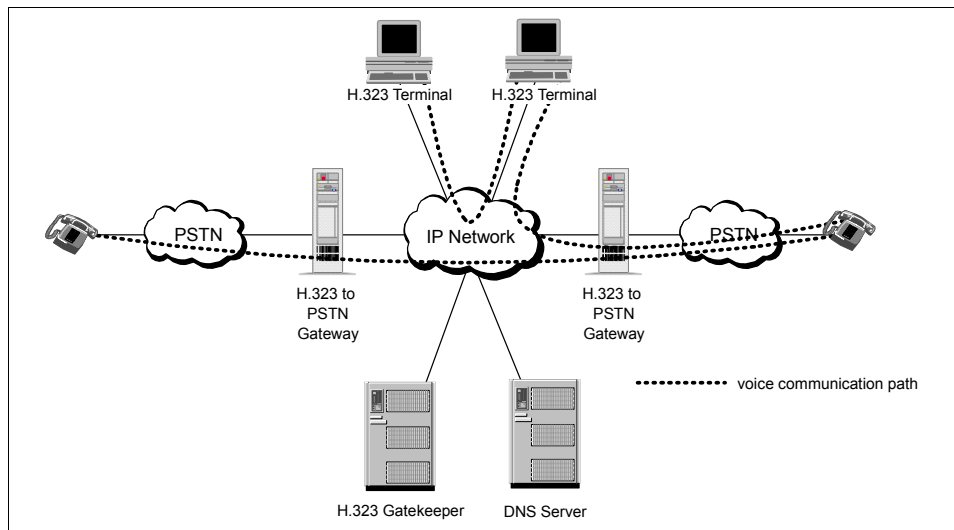


図 188 H.323 ネットワーク・エレメント

この図には、以下の 3 つのタイプの接続が示されています。

- ネイティブ H.323 端末は、IP ネットワーク・インフラストラクチャーを介して通信できます。
- IP ネットワークに接続された H.323 端末は、PSTN で標準の電話と通信できます。この構成では、H.323 ゲートウェイを使うと、IP 接続された H.323 端末と、非 IP ベースのネットワークへ接続された他の音声デバイスとの間でのインターオペラビリティが実現します。

- 2つの標準の PSTN 電話は、IP ネットワーク・インフラストラクチャーを介して通信できます。この構成では、IP ネットワークは、純粋にトランスポート・サービス向けに使われます。

13.2.3.6 H.323 プロトコル・スタック

H.323 仕様は、包括的な勧告です。H.323 のさまざまなコンポーネントは、他の ITU-T 規格によって定義されます。図 189 には、一群の H.323 定義が示されています。

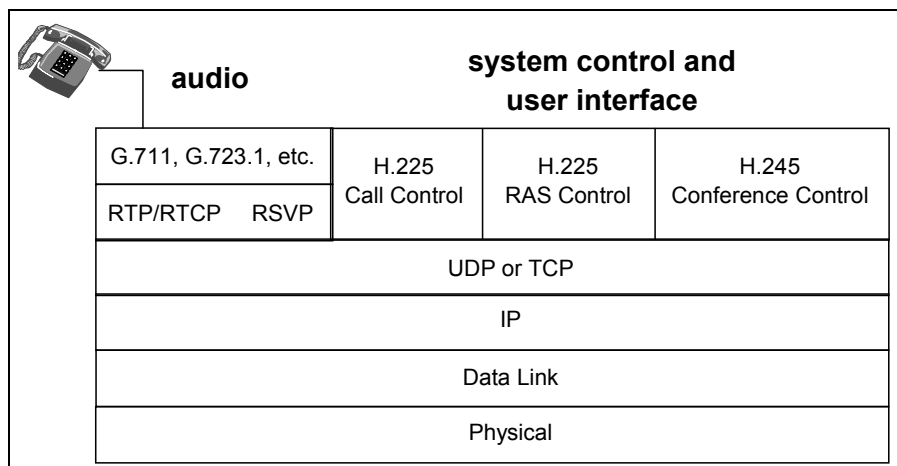


図 189 H.323 プロトコル・スタック

H.323 規格を取り巻くサブプロトコルには、以下のものがあります。

- H.225 呼び出し制御：このプロトコルには、呼び出しシグナルおよび制御機能があります。ゲートキーパーのないネットワークでは、呼び出しシグナル・メッセージは、呼び出し側エンドポイントと呼び出し先エンドポイントとの間で直接に受け渡しされます。ゲートキーパーのあるネットワークでは、これらのメッセージは、まず呼び出し側エンドポイントとゲートキーパーとの間で交換されます。H.225 呼び出し制御メッセージでは、TCP トランスポート・サービスが使われます。
- H.225 登録、承認およびシグナル (RAS) 制御：RAS チャンネルは、エンドポイントとゲートキーパーとの間の通信に使われます。このプロトコルは、ゲートキーパー・ディスカバリー、エンドポイント登録、エンドポイント検出、および承認制御のための標準手順を定義しています。H.225 RAS メッセージでは、UDP トランスポート・サービスが使われます。
- H.245 会議制御：これは、呼び出しの確立フェーズが完了した後に使われるプロトコルを指定します。H.245 は、RTP と RTCP によって使われるメディア・チャンネルをネゴシエーションするときに使われます。このプロ

トコルは、機能の交換、マスターおよびスレーブ割り当ての決定、および会議の制御の標準手順を定義しています。

- G.711、G.723.1: 496 ページのセクション 13.2.8『音声によるエンコードおよび圧縮』には、これらのエンコードおよび圧縮仕様についての詳細が示されています。
- Real-time Transport Protocol (RTP): このプロトコルでは、受信側はパケットの消失を検出できます。さらに、受信側が変数パケットの到着時刻を補正できるようにする、タイミング情報も提示されます (つまり、ジッター)。RTP 機能の詳細は、470 ページの 13.1.1『Real-Time Transport Protocol (RTP)』のセクションにあります。
- Real-time Control Protocol (RTCP): このプロトコルは、RTP に付随するものです。これは、ネットワーク状況や QoS フィードバック情報を入手するときに使われます。RTCP 機能の詳細は、475 ページの 13.1.2『Real-Time Control Protocol』のセクションにあります。
- Resource Reservation Protocol (RSVP): ホストは、RSVP を使い、ネットワークから特定の QoS を要求します。この要求は、セッション・パスに沿った各ノードによって処理されます。デバイスは、適切なリソースを予約し、アプリケーション・データ・ストリームをサポートします。

13.2.4 Session Initiation Protocol (SIP)

SIP は、呼び出しを開始、変更、および終了するときに使われるアプリケーション層シグナル・プロトコルです。これにより、クライアント、VoIP ゲートウェイ、および他の通信システムは通信できるようになります。SIP によって提供されるサービスおよび機能は、H.323 バージョン 2 と機能的に似ています。

SIP は、元々は、ユーザーが MBONE セッションに加わるようにするための 1 つの方法として、1996 年後期に開発されたものです。追加機能の開発が続けられ、プロトコルを説明する RFC は 1998 年に公開されました。基本規格は、RFC 2543 に説明されています。RFC が公開されてすぐに、開発者はこのプロトコルをテレフォニー規格として受け入れ始めました。

標準の SIP ネットワークには、以下の 2 つのコンポーネントがあります。

- ユーザー・エージェント
- ネットワーク・サーバー

13.2.4.1 ユーザー・エージェント

ユーザー・エージェントは、各 SIP 端末にあります。これは、以下の 2 つのコンポーネントがあります。

- User Agent Client (UAC) には、SIP 要求を発行する役割があります。

- User Agent Server (UAS) には、要求を受信して応答する役割があります。

ユーザー・エージェントは H.323 端末に相当します。

13.2.4.2 ネットワーク・サーバー

SIP ネットワーク・サーバーは、拡張された呼び出しの機能をサポートするために使われます。ネットワーク・サーバーには、以下の 3 つのタイプがあります。

- 転送サーバーは、呼び出しの開始時に使われ、呼び出されるデバイスのアドレスを判別します。この情報は、呼び出し側デバイスに戻されます。
- プロキシ・サーバーには、SIP 要求および応答をルーティングするアプリケーション層があります。サーバーが要求を受信すると、その要求は、呼び出し先デバイスの場所についての詳細が示されたネクスト・ホップ・サーバーへ転送されます。
- 登録サーバーは、SIP アドレスと、デバイスの関連した IP アドレスを記録するときに使われます。

SIP ネットワーク・サーバーでは、通常は、異なるサーバー・タイプの組み合わせが実装されます。ネットワーク・サーバーは、H.323 ゲートキーパーと機能的に同等です。

13.2.4.3 SIP アーキテクチャー

SIP 設計では、規格の開発時にはインターネット・モデルが使われていました。そのような SIP のアーキテクチャーは、HTTP のアーキテクチャーと似ています。どちらのシステムも、通信のために要求/応答モデルを使います。呼び出し側デバイスは、要求を呼び出し先デバイスへ送信します。呼び出し先デバイスは、要求を受け入れるか拒否するかを選択し、応答をイニシエーターに戻します。

SIP は、HTTP の他の機能を強化します。SIP のメッセージ形式は、HTTP に基づいています。SIP は、HTTP のほとんどのヘッダー・フィールド、エンコード規則、エラー・コードを再利用しています。これにより、情報を表示するために、読みやすいテキスト・ベースの形式が備えられます。

SIP アドレスは、SIP Uniform Resource Locator (SIP-URL) と言われます。これは、sip:username@hostname.domain のような形式になります。このアドレスは、個々のデバイスまたはデバイス・グループを識別するために使われます。これにより、ユーザーは、標準のブラウザでリンクを選択することによって、呼び出しを開始できるようになります。

13.2.4.4 SIP プロトコル・スタック

SIP 環境では、音声呼び出しは、H.323 でそうであったように、RTP を使って運ばれます。SIP と H.323 との主な違いは、呼び出しのシグナルと制御を行う方法です。SIP プロトコル・スタックについては、図 190 に示されています。

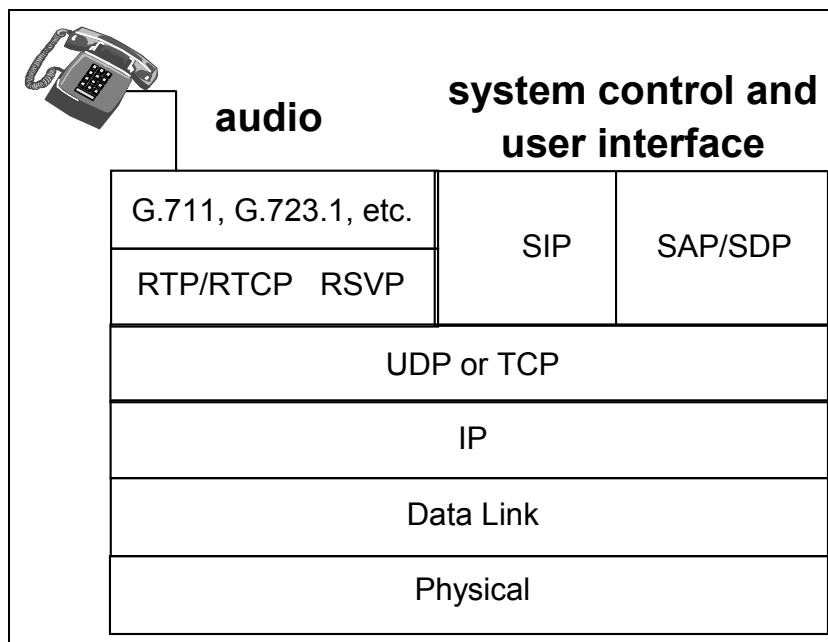


図 190 SIP プロトコル・スタック

SIP は、シグナルおよび制御機能を実現するために、別の規格を利用します。以下が含まれます。

- Session Description Protocol (SDP): SDP は、マルチメディア環境で、セッション・アナウンスとセッション案内を実現するプロトコルです。これにより、セッション・アナウンスの受信側は、セッションに参加できます。このプロトコルの詳細は、RFC 2327 に説明されています。
- Session Announcement Protocol (SAP): SAP は、マルチキャスト会議およびセッションを公示するための別のプロトコルです。これは、長期で広範囲なマルチキャスト・セッションの存在をアナウンスすることを目的としています。

13.2.5 Media Gateway Control Protocol (MGCP)

MGCP は、代替の呼び出しシグナル・プロトコルです。これは元々、H.323 がないことがはっきりとしている状態に取り組むために開発されたものです。これには、電話のゲートウェイをコンポーネント群に分ける機能があります。このようなコンポーネントは、マスター/スレーブの関係を構成します。マスター・コンポーネントは、呼び出しエージェントと呼ばれます。スレーブ・コンポーネントは、ゲートウェイと呼ばれます。

MGCP モデルでは、コアに拡張サービスを実装する知能が含まれているものと想定されています。エンドポイントの電話には、最低限の技術機能しかありません。これは、知能がネットワークのエンドポイントにも備え付けられている SIP アーキテクチャーとは異なる点です。MGCP モデルは、低コストのエンドポイントを介して拡張サービスを送達できるため、キャリアにとっては利点となると見なせます。

MGCP を形成するアーキテクチャーの開発は、1998 年中旬に開始されました。IETF は、MGCP RFC を 1999 年後期に公開しました。基本的な MGCP の機能については、RFC 2705 で説明されています。この規格には、アプリケーション・プログラミング・インターフェース (API) と、対応するプロトコル (MGCP) が備えられています。

標準の MGCP ネットワークには、以下の 2 つのコンポーネントがあります。

- 呼び出しエージェント
- ゲートウェイ

13.2.5.1 呼び出しエージェント

呼び出しエージェントは、ゲートウェイの操作を指示します。呼び出し操作を実行するのに必要な知能は、呼び出しエージェントにあります。また、これらのデバイスには、シグナル機能および呼び出し処理機能を実行する役割もあります。

MGCP ネットワークは、H.323 ネットワークに統合する必要はありません。しかし、MGCP 呼び出しエージェントはシグナル機能を実装するので、H.323 ゲートキーパーとして示すことができます。

13.2.5.2 ゲートウェイ

ゲートウェイには、PSTN で使われる音声シグナルと IP ネットワークで使われるデータ・パケット間で変換をする役割があります。ゲートウェイの例には、以下のものがあります。

- トランク・ゲートウェイには、従来の PSTN 電話ネットワークと VoIP ネットワークとの間のインターフェースがあります。これらのゲートウェイは、通常は多数の回線を管理します。

- ・常駐ゲートウェイには、従来のテレフォニー・エンド・ユーザー（アナログ RJ-11）と VoIP ネットワークとの間のインターフェースがあります。ここには、ケーブル・モデムと xDSL デバイスへのインターフェースが含まれます。

MGCP は、マスター / スレーブ・プロトコルであるため、ゲートウェイは、呼び出しエージェントによって提供されるコマンドを実行することになります。

13.2.6 Media Gateway Controller (Megaco)

IP テレフォニー・ネットワークでの MGCP の使い方と同様で、Megaco は、ゲートウェイと呼び出しエージェントの関係を保ちます。しかし、MGCP と違う点は、Megaco は ATM を含むより広範囲のネットワークをサポートしていることです。これにより、このプロトコルは、さらに広範囲のメディア・ゲートウェイに対応できるようになります。

MGCP 規格が承認された後、別の提案および拡張が IETF へ提出されました。Megaco プロトコルは、このような個別の規格を 1 つのアーキテクチャーに結合するものです。これは、ITU と IETF によるジョイント開発の結果です。

Megaco は、サイズの範囲が 1 つのポートから幾千のポートまでのゲートウェイをサポートするように設計されています。標準の Megaco ネットワークには、以下の 2 つのコンポーネントがあります。

- ・ 終端
- ・ コンテキスト

13.2.6.1 終端

終端は、ゲートウェイに入るまたはそこから出るストリームを表します。一部の終端（一般にはゲートウェイでのポートを表す）は、デバイスをアクティブにすると自動的に作成されます。他の終端は、必要なときに作成されます。これらは、RTP トラフィック・フローや VoIP ストリームのような、ネットワークを介した一時フローを表します。

13.2.6.2 コンテキスト

終端は、コンテキストに置かれます。この状況は、複数の終端ストリームが相互に接続される場合に生じます。1 つの呼び出しの中には、1 つのコンテキストあたり 2 つの終端を含めることができます（1 つは、端末デバイスを表す終端、もう 1 つは、ネットワークへの接続を表す終端）。1 つの会議通話の中には、1 つのコンテキストあたり多数の終端を含めることができ、各終端は、会議の 1 人の参加者を表します。

Megaco では、終端およびコンテキストを扱うために、一連のコマンドを使います。たとえば、ADD コマンドは、コンテキストに終端を追加します。MOVE コマンドは、終端をあるコンテキストから別のコンテキストへ移動します。このようなコマンドは、呼び出しエージェントからゲートウェイに送信されます。

13.2.7 シグナル・プロトコル機能の比較

複数のシグナル規格が開発されるに際し、以下のような経緯がありました。

- ・ プロトコルは、個別の規格組織によって開発されていました。たとえば、H.323 は ITU によって開発されました。従来は、この組織は、回線交換環境の規格を開発していました。SIP 規格は、Internet Engineering Task Force (IETF) によって開発されました。従来は、この組織は、データ・ネットワーク規格を開発していました。
- ・ このプロトコルは別のときに開発されました。H.323 は、1995 年の初期にまでさかのぼります。SIP の開発は、1999 年まで標準化されていませんでした。

各シグナル・プロトコルには、利点と欠点があります。

13.2.7.1 デプロイメント

H.323 は、最も初期の主要な VoIP シグナル・プロトコルでした。長い間、インターオペラビリティの規格として推進されていました。これは最初の主要なプロトコル・オプションであったため、この規格は、マーケットの非常に大きな部分を占めています。従来のテレフォニー・プロバイダーやベンダーは、この概念とアーキテクチャーに精通していたため、元々はこれらのプロバイダーやベンダーに採用されていました。しかし、プロバイダーが拡張サービスの開発を模索していくにつれ、業界は H.323 から SIP へ移行するようになりました。

Megaco 規格には、MGCP 規格で提供された機能のスーパーセットが備えられているため、MGCP のサポートは勢いを失いつつあります。

13.2.7.2 プロトコルの複雑さと拡張容易性

H.323 は、比較的複雑なプロトコルです。多くのサービスでは、この規格を取り巻く多数のサブプロトコル間での対話が必要です。これにより複雑さが増すので、これは IP テレフォニー・アプリケーションの欠点といえます。このような複雑さは、この規格に基づいた新しい機能の開発に要する時間に悪影響を与えます。

SIP は、HTTP の既存の成熟度を強化します。HTTP は、大規模なネットワーク環境をサポートできることが証明されています。HTTP の進化に応じて開発されてきた、このような拡張容易性の強化は、SIP にも組み込まれていま

す。これにより、SIP は非常に大規模な環境へ拡大できる可能性を持っています。

13.2.7.3 インターオペラビリティ

インターオペラビリティには、さまざまなベンダーが規格を実装するときの整合性と、同じ規格でバージョンが異なる場合の整合性も含まれます。

- ベンダー実装の違い：H.323 のサービス定義は非常に詳しく記述されているので、インターオペラビリティの問題はほとんど発生しません。しかし、SIP は非常に柔軟で拡張しやすくなっており、あるベンダーが他のベンダーとは別の方法でプロトコルを拡張しないように、最低限の安全機能が設定されています。
- バージョンの違い：H.323 では、異なるバージョンでの互換性を保証するための手段が講じられています。逆に、より新しいバージョンの SIP では、最低限のサポートを受けていた古い機能が破棄されている可能性があります。これにより、プロトコルの複雑さは少なくなりますが、バージョン間の非互換性の問題が生じてしまいます。以前の SIP バージョンで実装されていた一部の機能は、新しいバージョンではサポートされない可能性があります。

13.2.7.4 実装のしやすさ

H.323 シグナルは、ASN.1 表記でエンコードされています。このエンコード方式では、複雑な構文解析が求められるため、実装およびデバッグの手間は複雑になります。逆に、SIP メッセージはテキスト・ベースです。このタイプのエンコード方式では、実装とデバッグが容易になります。

13.2.7.5 サービス品質のサポート

H.323 と SIP のサービス品質機能は同等です。どちらのプロトコルの呼び出しセットアップ回数も、同じようなものです。さらに、H.323 と SIP のどちらも、リソース予約の直接サポートはありません。ネットワーク・リソースを予約するために、それぞれが RSVP 規格に依存します。

13.2.7.6 受け入れ状況

SIP、MGCP、および Megaco には、IETF の後ろ盾があります。これは、最も認められている標準規格制定機関の 1 つです。

13.2.8 音声によるエンコードおよび圧縮

圧縮されていない音声パケットをネットワークで伝送すると、大量の帯域幅を占有してしまう可能性があります。この問題を軽減させるため、特定のデバイスを使い、伝送する前に音声パケットをエンコードして圧縮します。その同じパケットが伝送先に到着したら、伝送先のデバイスを使い、それらのパケットを解凍してデコードします。

これらのデバイスを使うことにより、音声パケットを送達するのに要する時間全体に遅れが生じます。

- 圧縮を実行するハードウェアでは、特定の音声セグメントを評価するためにデータをバッファに入れるときに、遅れが生じます。
- 圧縮を実行するハードウェアでは、音声を圧縮するための実際の計算を行うと、余分の遅れが生じます。

この遅れの影響を少なくするために、音声の品質と帯域幅の使用量との間のバランスを決定する必要があります。このバランスは、特定の音声アプリケーションとデータ・アプリケーションが混合されたものに固有であり、ある環境とその隣の環境とは異なります。

音声圧縮を実現するために使われている既存の規格は、以下のものを含め、たくさん存在します。

- G.711: この規格は、64 Kbps で流れている圧縮されていない音声ストリームをエンコードします。これは、LAN の帯域幅制限内では十分ですが、ローカル・ネットワーク外での伝送には適していません。この規格では、長距離品質の音声を提供されます。音声を圧縮するために、すべての H.323 端末が G.711 音声規格をサポートしていなければなりません。
- G.723.1: この規格では、出力ビット・レートが 5.3 または 6.4 Kbps の、比較的に高密度の圧縮を実現します。数ある圧縮規格の中で、G.723.1 では、非常に低いビット・レートで長距離品質のパフォーマンスが提供されます。これは、International Multimedia Teleconferencing Consortium (IMTC) VoIP Forum により、IP テレフォニーのデフォルト・エンコーダーとして選定されています。

この規格の実装は、CPU に負荷をかける場合があります。G.723.1 を使った拡張が容易なソリューションを提供する機能にも悪影響を与える可能性があります。

- G.729A: この規格でも、出力ビット・レートが 8 Kbps の、比較的に高密度の圧縮を実現します。この規格では、長距離品質に近いパフォーマンスを実現します。この規格は、IMTC VoIP Forum の代替デフォルト・エンコーダーとして選定されました。

第 14 章 Wireless Application Protocol (WAP)

Wireless Application Protocol (WAP) は、携帯電話、ポケットベル、その他のワイヤレス装置などのデジタル・モバイル装置でインターネット通信および拡張サービスを提供するための、世界的なデファクト・スタンダードです。このプロトコルは、言及されたデジタル装置のユーザーがインターネット、イントラネット、およびエクストラネットのアプリケーションとサービスに安全にアクセスして対話することを可能にする、オープンでグローバルな仕様です。

調査の専門家によれば、ワイヤレス加入者の数は 2001 年の末までに 5 億 3 千万を超えると予想されています。一部の見積もりでは、ワイヤレス加入者の数は 2004 年には 10 億を超えると報告されています。近い将来に販売される電話のほとんどには、以下を含むマルチメディア機能が含まれることになります。

- Eメールの検索
- 企業のデータベースにあるデータへのアクセス
- 株取引の実行
- 勘定書の支払い
- 旅行の予約
- 他の興味深いアプリケーションの実行

これらの予想に基づいて、ワイヤレス・サービス提供者、携帯電話の製造者、およびソフトウェア開発会社は、情報に飢えているユーザーからなる巨大な新しいマーケットを展望しています。

WAP フォーラムは、Ericsson、Motorola、Nokia、および Phone.com によって 1997 年 6 月に設立されました。このフォーラムは、すべてのワイヤレス・ネットワークのためのグローバルなワイヤレス仕様のドラフトを作成しました。このフォーラムは、ワイヤレス・サービス提供者、ハンドセット製造者、インフラストラクチャー提供者、ソフトウェア開発者などのさまざまな産業グループを支援しています。さらにこのフォーラムは、ARIB、CDG、ECMA、ETSI、TIA、および W3C などの標準化団体にも仕様を提供しています。現在では、IBM を含む 400 以上のメンバーが WAP フォーラムに所属しています。

14.1 WAP 環境

ワイヤレス・データ・ネットワークには有線ネットワークと比較して、通信に関する難点が多くあります。たとえば、ワイヤレス・リンクに以下の特徴があるために、ワイヤレス通信環境は制約を受けています。

- 狭い帯域幅

- 長い待ち時間
- 不安定な接続
- 可用性が予測しにくい

そのため、ワイヤレスに適用できるプロトコルはこれらのタイプの問題に対応できるものでなければなりません。さらに、これらのワイヤレス・リンクを使用するマシンは処理能力その他の機能の点で制限されています。たとえば、ワイヤレス装置には Web ブラウザーなどの従来からあるデスクトップ GUI が備わっていません。さらに、携帯用のワイヤレス装置には、以下の基本的な制限も課されます。

- バッテリーの寿命
- 強力でない CPU
- 少ないメモリー
- 電力消費の制限
- 小さいディスプレイ
- 入力装置の相違 (電話キーパッドや音声入力など)
- マウスを使用できない

現在のユーザー・アプリケーションで必要とされるのは、トランザクション処理ではなく主として情報照会です。たとえば、ユーザーは小さな画面上で E メールの内容を読むのではなく、受信した Eメールの概要を表示するだけであったり、株取引を行うのではなく、株価を表示するだけであったりします。しかしそれほど遠くない将来に、銀行取引を行うなどの他の作業が必要になると予想されます。これは信頼性のあるデータ伝送、メッセージ・ブライバシー、整合性、および認証を必要とします。そのため、これらの機能も WAP の仕様の一部として含まれている必要があります。

14.2 WAP 仕様の主な要素

WAP はワイヤレスのインターネット・アクセスをインプリメントするオープン・スタンダード・アーキテクチャーおよびプロトコルのセットを定義します。WAP 仕様の主な要素には、以下のものが含まれます。

- WAP プログラミング・モデルの定義 (501 ページの図 191 を参照)
- Wireless Markup Language (WML) (506 ページの 14.3 『WirelessMarkup Language (WML) および WMLScript』を参照)
- マイクロブラウザーの仕様
- 軽量プロトコル・スタック (516 ページの 14.5 『WAP プロトコル・スタックの概要』を参照)
- ワイヤレス電話のフレームワーク

14.2.1 WAP プログラミング・モデルの概要

図 191 に図示されている WAP アーキテクチャーは、以下について説明しています。

- WAP クライアント (ハンドヘルド・デバイスまたは WAP 端末)
- WAP ゲートウェイ
- Web サーバー

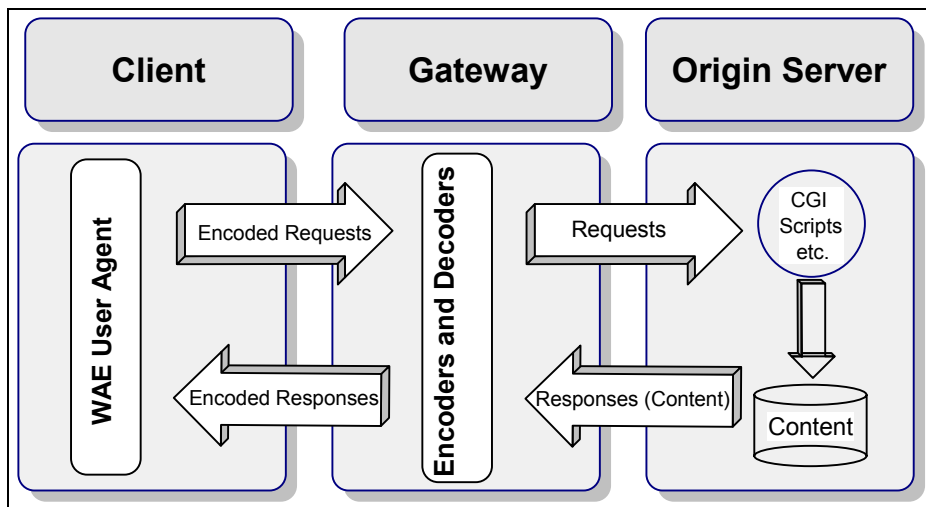


図 191 WAP プログラミング・モデル

このプログラミング・モデルは、Web プログラミング・モデルと似ています。クライアント / サーバー・テクノロジーに基づくよく知られたインターフェースを使用していることは、プログラマーにとって利点となります。

ハンドヘルド・デバイス (WAP 端末) の制約を解決するために、クライアントと Web サーバーとの間にゲートウェイ・プロトコルが存在します。ゲートウェイは従来の Web コンテンツをコンパクト・エンコード形式に変換します。これにより、ワイヤレス・ネットワーク内を移動するパケットのサイズと数が減少します。そのため、このゲートウェイまたはプロキシーを介して 2 つの接続が確立されます。クライアントおよびプロキシーの間 (WSP/WTP) と、プロキシーおよび Web サーバーの間 (TCP/IP HTTP) とでは、異なるプロトコルが使用されます。

14.2.1.1 WAP クライアント

クライアントは、ワイヤレス・アプリケーション環境 (WAE) ユーザー・エージェントとしても知られ、WAP 端末のコンポーネントです。それは WAP 階

層化構造を通過するすべての要求と応答の実行を処理するために、マイクロブラウザおよび WAP プロトコル・スタック (516 ページの図 197 を参照してください) からなります。たとえば、これには以下のものが含まれています。

- セッション確立
- 非接続または接続指向のデータ・トランスポート
- 以下を含む機密保護機能のある環境のセットアップ。
 - 暗号化および認証の適用
 - 発信要求のエンコード
- 帯域幅を最小化するための着信応答のデコード

マイクロブラウザは、ユーザーとインターネットまたはイントラネットから送られてきた WAP 内容とのインターフェースです。マイクロブラウザによって、以下の 2 つの規格がサポートされています。

- エンコードされた Wireless Markup Language (WML)
- コンパイルされた Wireless Markup Language Script (WMLScript)

詳細については、506 ページの 14.3 『Wireless Markup Language (WML) および WMLScript』を参照してください。

14.2.1.2 WAP ゲートウェイ

ゲートウェイは WAP プロキシとして機能します。それはクライアントから要求を受け取り、HTTP メッセージを変換して、それを (URL に基づいて) アドレッシングされた Web サーバーに送信します。Web サーバーが応答を戻したとき、ゲートウェイはそれを再び変換してビットでコーディングされた出力にします。その後それをモバイル・ネットワークに送り、モバイル・ネットワークはそれを WAP クライアントに送ります。このメソッドにより、従来の Web テクノロジーを使用する標準の Web サーバーが、データ内容とアプリケーションのホストとなることができます。

WAP ゲートウェイは WAP アプリケーション・サーバーとしても機能します (506 ページの図 193 を参照してください)。このソリューションには、Web サーバーは必要ありません。これを使用して、エンドツーエンドのセキュリティ構成をサポートすることができます。これは Wireless Telephony Application (WTA) (518 ページの 14.5.2 『Wireless Telephony Application (WTA)』を参照してください) で使用されるものなど、高度なアクセス制御またはサービスの保証を要求するアプリケーションに適しています。

WAP ゲートウェイは、複数の異なる Web サーバーからのデータを集約して、使用された情報を頻繁にキャッシングすることにより、WAP 端末への応答時間を短縮します。それはさらに、応答をクライアントに送信する前に、大きな HTTP 応答を小さな伝送単位に分割します。

WAP ゲートウェイは加入者データベースと接続して、情報を使用することにより、特定のユーザー・グループのために WML ページを動的にカスタマイズします。

WAP ゲートウェイはインターネットと他の非音声モバイル・サービスとの間の遷移を提供します。これらは、短メッセージ・サービス (SMS)、Circuit Switched Data (CSD)、または General Packet Radio Services (GPRS) などです。

14.2.1.3 発信元サーバー

クライアントのマイクロブラウザーは、Wireless Markup Language (WML) ページを要求します。WML ページはインターネットまたはイントラネットで接続された発信元サーバーに保管されます。発信元サーバーは Web サーバーなどです。WML ページも、ゲートウェイ自体にインストールされたアプリケーション・サーバーに保管されます。WML ページは、WML デックからなります。1 つの WML デックは、1 つ以上の WML カードに分割されます。WML カードは、対話の単位として認識されます。ユーザーはサービスを使用して 1 つ以上の WML ページから、カードとカードの間を両方向にナビゲートすることができます。WML は特に WAP 端末用に設計されており、HTML よりも小さなナットのマークアップ・タグを備えています。これは HTTP 1.1 に基づいています。WML デックには、Web ページをコーディングする別の方法である WMLScripts も含まれています。

14.2.2 WAP ネットワーク構成

WAP ネットワークは、次の 2 つの構成を使用して設計することができます。

- 最初の構成は、WAP ゲートウェイを介する Web サーバーへのクライアントのアクセスを記述します。この構成は、HTTP 接続による従来の Web サービスが使用されている場合に使用されます。
- 2 番目の構成は、クライアントが IP ネットワークを横断することなく WAP アプリケーション・サーバーに直接アクセスできるようにします。この場合、WAP ゲートウェイは使用されません。このソリューションが主に使用されるのは、ワイヤレス電話のアプリケーションに関してです。これは、電話帳、天気や交通状態に関するブロードキャスト情報、株価情報などのアプリケーションです。

ハードウェアのコストを節約するために、WAP アプリケーション・サーバーのすべての機能を WAP ゲートウェイにインプリメントすることもできます。

14.2.2.1 WAP ゲートウェイの使用法

図 192 に図示されているこのタイプの構成は、次の 2 つの接続からなります。

- 最初の接続は、クライアントと WAP ゲートウェイの間の接続です。

- 2 番目の接続は、WAP ゲートウェイと Web サーバーの間の接続です。

最初の接続は Wireless Session Protocol (WSP) および Wireless Transaction Protocol (WTP) を使用し、2 番目の接続は TCP 上で実行される従来の Hypertext Transfer Protocol (HTTP) を使用します。プロキシ・ゲートウェイは、これら 2 つの接続をクライアントと Web サーバーとの間の 1 つの論理接続として保守します。

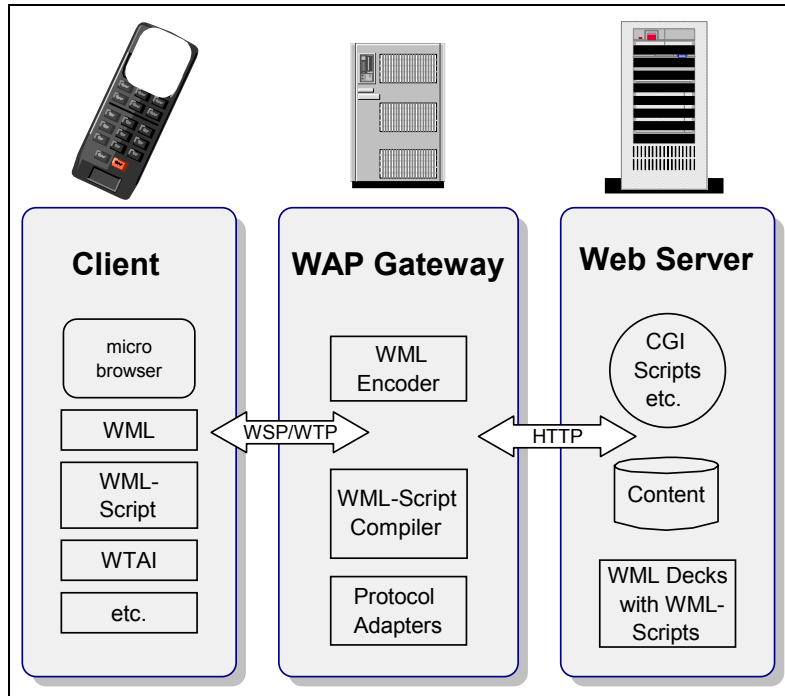


図 192 WAP ゲートウェイの使用方法

クライアント / サーバー間のフロー

クライアント / サーバー間のフローは以下のとおりです。

1. 発信元サーバーに特定のサービスを要求しているクライアントは、WML ユーザー・エージェントを使用してこのサーバーに要求を発信します (517 ページの図 198 を参照してください)。
2. WML ユーザー・エージェントは URL アドレッシング方式操作を使用して、発信元サーバーを検索します。URL を使用して Common Gateway Interface (CGI) などの Web サーバー上のアプリケーションをアドレッシングします。¹

¹ Common Gateway Interface (CGI) スクリプトは本質的に、内容を出力として生成する実行可能エンティティです。

3. クライアントの要求は WSP/WTP を使用してゲートウェイに送信されます。
4. ゲートウェイはビットのエンコードを行い、要求を HTTP メッセージに変換して、それを (URL でアドレッシングされた) Web サーバーに送信します。
5. 発信元サーバーは、単一のデックが発信元サーバーにテキスト形式で保管されている場合、それを戻して要求に応答します (506 ページの 14.3.1 『WML』を参照してください)。
6. デックはゲートウェイに伝送されます。
7. ゲートウェイに移動する間、各デックのテキスト形式は WAP 端末への空中伝送に適した形式に変換されなければなりません。ゲートウェイはこの変換を、テキスト形式をバイナリー形式に変換する WML エンコーダーを使用して行います。
8. ゲートウェイはエンコード済みコンテンツを WAP プロトコル・スタック層を介してワイヤレス・ネットワーク上のクライアントに送信します (516 ページの図 197 を参照してください)。
9. クライアントの WAP 端末では、マイクロブラウザなどのように、データが受信されて表示されたり解釈されます。

ゲートウェイとクライアントとの間のネゴシエーションに基づく追加の最適化プロシージャもあります。クライアントは 1 つ以上の要求を WML ユーザー・エージェントを介して発信することができます。これらの要求は WMLScripts にアドレッシングすることもできます。これは WMLScript ユーザー・エージェントにある発信元サーバーから受け取ることができます。

ゲートウェイでのプロシージャは、WML プロシージャとは異なります。ゲートウェイが WMLScript を受け取ると、

- WMLScript コンパイラーがスクリプトを入力として受け取ります。
- それをコンパイルしてバイト・コードにします。
- それを WAP プロトコル・スタック層を使用して、ワイヤレス・ネットワークを介して WAP 端末に送信します。

14.2.2.2 WAP アプリケーション・サーバーの使用方法

WAP ゲートウェイの使用は必須ではありません。テキスト内容のエンコードおよびコンパイルは、ゲートウェイで行う必要はありません。一部の発信元サーバーには WML エンコーダーおよび WMLScript コンパイラーがインプリメントされていると考えられます。図 193 に示されているように、これを WAP アプリケーション・サーバーで行うこともできます。

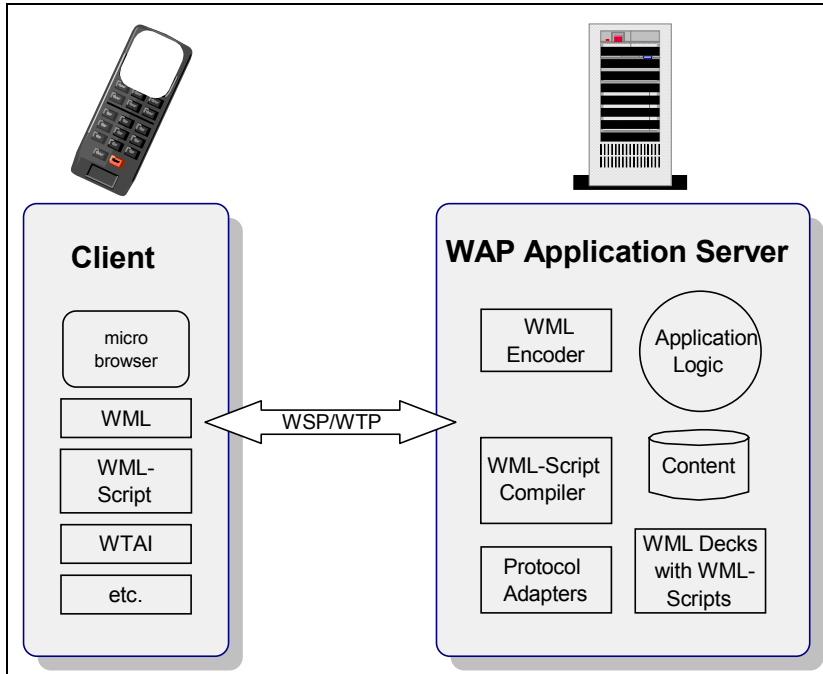


図 193 WAP アプリケーション・サーバーの使用方法

14.3 WirelessMarkup Language (WML) および WMLScript

クライアントは WAP アプリケーション・エンティティー (WAE) 層を介して発信元サーバーと対話します。WAE はユーザー・エージェントおよびサービスからなります (517 ページの図 198 を参照してください)。WML ユーザー・エージェントは、クライアントとサーバーとの間の通信のために、2 つの異なるサービスを提供します。

- WML
- WMLScript

14.3.1 WML

WML はタグ・ベースの文書言語です。それは狭帯域で伝送するために特に設計された電話その他のワイヤレス・モバイル端末などの装置上で表示およびユーザー対話を行うために最適化されています。WML にはデッキおよびカード構造がインプリメントされています。これにより、アプリケーションは複数のカードから形成された文書 (通常はデッキと呼ばれる) を指定することができます。ユーザーは以下の方法でこれらの WML カードと対話します。

- 前後にナビゲートする

- 内容をレビューする
- 要求情報を入力できる
- 選択を行うことができる
- 他のカードに移動する

発信元サーバーに対するサービスを開始する指示がカード内に存在することがあります。WML デックは発信元サーバー上にファイルとして保管されます。それらは要求に応じてサーバーから取り出されます。デックは、発信元サーバー上で実行しているコンテンツ・ジェネレーターによって動的に生成することもできます。デック内の各カードには、特定のユーザー対話のための仕様が含まれています。

WML には、さまざまな機能が備わっています。以下に例を示します。

- テキストおよびイメージのサポート

このサポートには、強調エレメント (太字、イタリック、拡大文字、その他など)、行分割モデル (行折り返し、行折り返し抑止など)、および単純なタブ位置合わせをサポートするタブ列による出力表示が含まれます。

- ユーザー入力のサポート

ユーザー入力を求めるいくつかのエレメントがサポートされています。それらのエレメントは結合して 1 つ以上のカードにすることができます。WML には、テキストおよびパスワード入力のためにテキスト入力制御が含まれています。テキスト入力フィールドにはマスクをして、ユーザーが不正な文字を入力しないように保護することができます。WML には、ユーザーがデータの設定、カード間のナビゲート、またはスクリプトの起動を行うためのオプションのリストを表示する、選択制御などのオプションが備わっています。さらに、タスク開始制御もセットアップできます。

- ナビゲーションおよびヒストリー・スタック

WML では、URL によるいくつかのナビゲーション機構を使用できます。ナビゲーションには、HTML スタイルのハイパーリンク、カード間ナビゲーション・エレメント、およびヒストリー・ナビゲーション・エレメントが含まれます。

- インターナショナル・サポート

WML は ISO/IEC-1046 の汎用文字セットを使用します。現在、この文字セットはユニコード 2.0 と同じです。

- 狭帯域の最適化

狭帯域端末での通信を最適化するさまざまなテクノロジーがあります。たとえば、複数ユーザー対話 (カード) は、1 つのデータ転送 (デック) で実行されます。

- 状態およびコンテキスト管理

WML はフラット・コンテキスト制御を公開しています。これは、それぞれの WML テキスト入力に変数を使用できることを意味しています。変数の状態を使用して、サーバーと通信しなくてもパラメーター化されたカードの内容を変更することができます。

14.3.1.1 WML の例

図 194 に図示されているこの例では、<WML> で開始して </WML> で終了する 1 つのデッキからなる WML が示されています。デッキには、2 つのカードが含まれています。カードは <CARD> で開始して </CARD> で終了します。

最初のカードはウェルカム・パネルを制御します。2 番目のカードは、変数を含む入力機能を提供します。さらに、ユーザーが 2 つの処理オプションから選択することを可能にしています。

図 194 のサンプルは、ユーザーをウェルカム・パネル内をナビゲートします。後続の入力のために、2 つの変数 N および S が提供されています。ユーザーは変数 S に 0 または 1 を入力することができます。

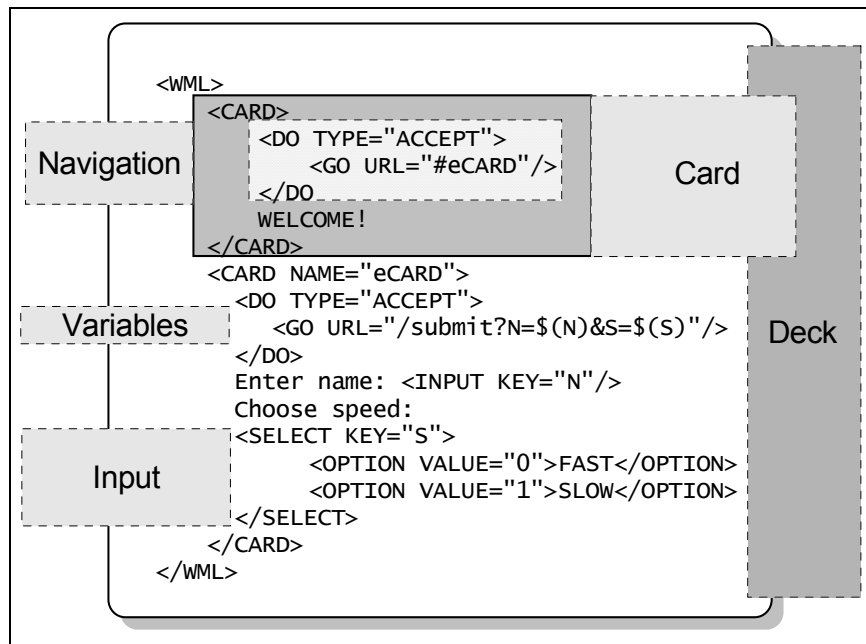


図 194 WML の例

14.3.2 WMLScript

WMLScript は軽快なプロシージャ型言語です。これは WML の標準のブラウザおよび表示機能を強化します。以下に例を示します。

- クライアントに情報を追加します。
- プログラミング・ループを作成することができます。
- 条件を実行します。

WMLScript は、JavaScript WWW スクリプト言語から派生したものです。そのため、プロシージャ論理を WML デックに適用できます。WMLScript により、アプリケーション・プログラマーはさまざまな興味深い機能を使用できます。

- ユーザー入力コンテンツ・サーバーに送信される前にその妥当性を検査する機能
- 装置設備および周辺装置にアクセスする機能
- 発信元サーバーとの間を往復しなくてもユーザーと対話できる機能 (エラー・メッセージの表示など)。

14.4 プッシュ・アーキテクチャー

クライアント / サーバー・モデルの設計によれば、サーバーにサービスを要求できるのはクライアントだけです。サーバーは情報をクライアントに送信して応答します。このタイプの情報交換は、クライアントがサーバーから情報を引き出すために、**プル・テクノロジー**と呼ばれます。

WWW メソッドはプル・テクノロジーの典型的な例です。クライアントは URL を入力して、Web サーバーから情報を検索します。Web サーバーは Web ブラウザー上の Web ページに表示する情報をクライアントに送信します。

従来の Web テクノロジーによれば、ワイヤレス・サービス提供者は要求されない限りクライアントにメッセージを送ることができませんでした。このサービスを可能にするために、**プッシュ・アーキテクチャー**が設計されました。

それとは対照的に、WAP アーキテクチャーはさらに **プッシュ・テクノロジー**を図 195 に示すように定義しています。このテクノロジーもクライアント / サーバー・モデルに基づいていますが、クライアントからサーバーへの明示的な要求はありません。プッシュ・トランザクションはサーバーだけによって開始されます。この方法によって、事前のユーザー・アクションがなくても情報を WAP 端末に送ることができます。

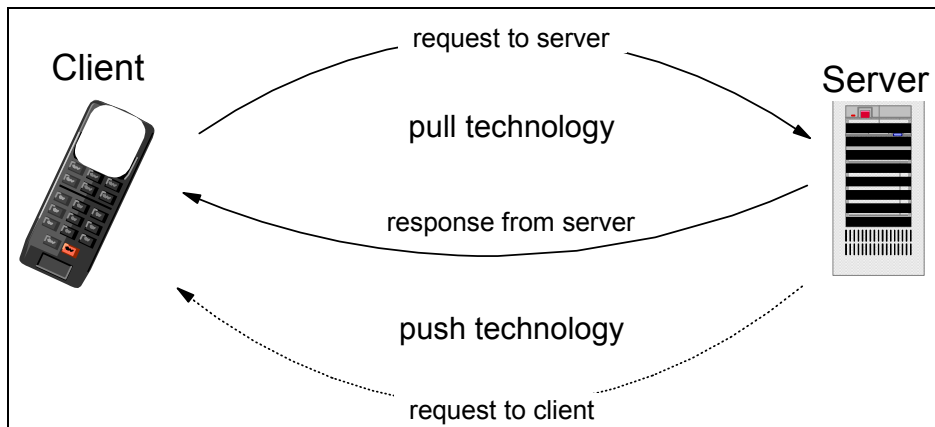


図 195 プルとプッシュのテクノロジーの比較

14.4.1 プッシュ・フレームワーク

TCP/IP 内で、サーバーに対する情報の要求を開始できるのはクライアントだけです。しかし、情報サービス提供者には特別のクライアント要求がなくてもクライアントに連絡する方法も必要です。たとえば、メール・サービス提供者は WAP クライアントに E メールがクライアントのメールボックスに入っていることを通知したり、すべての送信者名を知らせたり、WAP クライアントが予約している最新の株価、ヘッドライン・ニュース、天気や交通の情報などを送りたいと考えます。

サーバーが WAP クライアントとの接続の開始者となるこの種類のトランザクションは、新しい機能です。これは、図 196 で図示されているように、プッシュ・イニシエーターの名前で設計されています。

WAP クライアントは WAP ドメインに存在し、サーバーはインターネット内に存在するために、それらはプッシュ・イニシエーター・プロトコルを共有しません。そのため、ここでも、この機能をサポートするためにゲートウェイ・サポートが必要になります。プッシュを実行するゲートウェイは、プッシュ・プロキシ・ゲートウェイ (PPG) と呼ばれます。

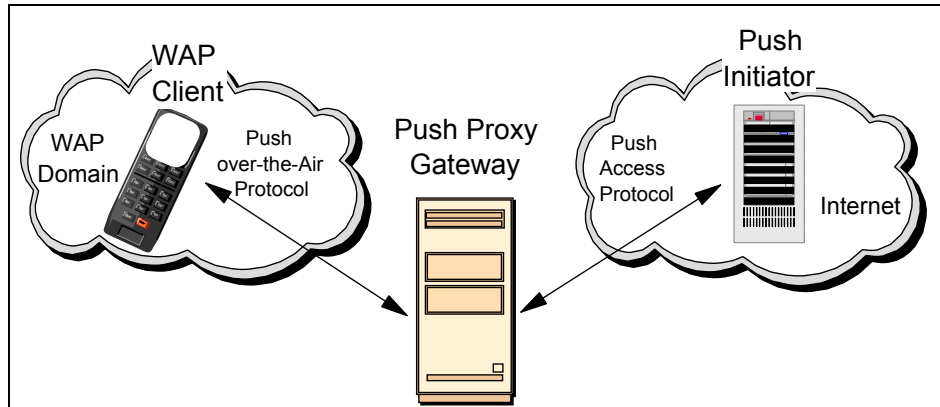


図 196 プッシュ・フレームワーク

プッシュ・イニシエーターが情報を WAP クライアントに送信したい場合、それはプッシュ・プロキシ・ゲートウェイ (PPG) と連絡を取ります。それは従来のインターネット・プロトコルと共に、push access プロトコル (PAP) を使用します。PPG はプッシュされたコンテンツを WAP ドメインに転送し、そのコンテンツはそこでプッシュ over-the-air プロトコル (OTA) を使用して空中に送信されます。

PPG には、プッシュ・イニシエーターにクライアントへのメッセージの送達状態を知らせる機能があります。しかし、WAP クライアントはオンラインではないことがあるので、これには時間がかかります。クライアントがオンラインである場合、それはプッシュされたコンテンツを受け入れるかまたは拒否します。

PAP は、インターネットを介して HTTP によりトンネルを通過してきた可能性がある XML メッセージを使用します。

14.4.2 プッシュ・プロキシ・ゲートウェイ (PPG)

PPG はインターネットからモバイル・ネットワークへのコンテンツ・プッシュのアクセス・ポイントです。これは、この種の操作に必要なすべての事柄を行います。以下に例を示します。

- 認証
- セキュリティー
- クライアント制御

ゲートウェイの所有者として、それは WAP ドメインへのアクセスを許可されるユーザー、およびメッセージのプッシュを許可されるユーザーを決めます。

14.4.2.1 PPG サービスの概要

プッシュ・プロキシ・ゲートウェイのサービスには、以下が含まれます。

- プッシュ・イニシエーターの識別認証、アクセス制御
- 内容制御情報の構文解析およびエラー検出
- クライアント・ディスカバリー・サービス
- アドレス解決
- OTA を改良するための特定コンテンツのバイナリー・エンコードおよびコンパイル
- プロトコル変換

14.4.2.2 インターネット・サイドからのアクセス

PPG は、インターネット側からのコンテンツを受け入れます。それはマルチパーツ / 関連したコンテンツ・タイプを使用して、複数のパーツに分割されます。最初のパーツには、PPG に関する以下のような情報が含まれています。

- 受信側情報
- タイムアウト
- コールバック要求

PPG はこの制御情報の構文解析を認識します。プッシュ・イニシエーターが要求した場合、WAP クライアントへのプッシュ実行依頼の最終状態に到達したとき、それはプッシュしたサーバーを呼び戻して、状態に関して報告します (送達された、キャンセルされた、または満了した)。

14.4.2.3 PPG によるプッシュされるコンテンツの送達

PPG は適正な WAP クライアントを見つけようとします。そして、見つけた場合には push over-the-air プロトコルを使用してコンテンツを送達します。送達の試行は、そのクライアントについて設定されたタイムアウトにより制限されます。このタイムアウトは、プッシュ・イニシエーターまたはモバイル・オペレーターのポリシーによって設定されます。

PPG はアドレッシング別名割り当て体系をインプリメントして、特別なマルチおよびブロードキャスト機能を使用可能にしていることがあります。これには、特別なアドレスをブロードキャスト操作に変換することが必要です。プッシュ・イニシエーターは、特定の WAP 端末用に適切にフォーマットされたコンテンツを作成するクライアント機能および優先度も照会することがあります。

14.4.3 Push access control プロトコル (PAP)

プッシュ・イニシエーターは PAP を使用して、コンテンツを PPG を介して WAP ドメインにプッシュします。PAP は HTTP 1.1 トンネルを介して移送されます。PAP は XML スタイルのエンティティを選びます。

14.4.3.1 PAP 操作

PAP は以下の操作をサポートします。

- プッシュ実行依頼 (PPG のイニシエーター)

プッシュ・メッセージには、PPG の送達指示を含む制御エンティティ、WAP 端末のテキスト形式コンテンツであるコンテンツ・エンティティ、およびオプションで、機能エンティティが含まれています。

メッセージ・タイプ (WML または WMLScript) に応じて、PPP はこのメッセージを over-the-air (OTA) に転送する前に、帯域幅がより最適化されている形式に変換します。さらに、メッセージをクライアントに送信する前に暗号化することもできます。

- 結果の通知 (PPG からイニシエーターへ)

プッシュ・イニシエーターが正常な送達の確認を要求していた場合、通知メッセージがプッシュ・イニシエーターに戻されます。これにより、プッシュ・イニシエーターは WAP クライアントも PPG への正常な送達を確認したことが分かります。

- プッシュ・キャンセル (イニシエーターから PPG へ)

この XML エンティティはプッシュ・イニシエーターから PPG に送られて、直前に実行依頼されたコンテンツをキャンセルするように要求します。PPG はキャンセルが成功したかどうかを知らせる応答をします。

- 状況照会 (イニシエーターから PPG へ)

プッシュ・イニシエーターは WAP クライアントへの 2 段階の処理の際に、送達の状況を要求します。

- 最初の段階：プッシュ・イニシエーター -> PPG
- 2 番目の段階：PPG -> クライアント

- クライアント機能の照会 (イニシエーターから PPG へ)

プッシュ・イニシエーターは PPG から、特定の WAP 端末の機能に関する情報を要求します。PPG は 2 つのパートからなるマルチパート / 関連メッセージを知らせて応答します。

- 最初のパートは、メッセージの実行そのものに関する情報です。
- 2 番目のパートでは、ユーザー・エージェント・プロファイル・グループによって定義された WAP 端末の機能が報告されます。

14.4.4 サービス指示

サービス指示 (SI) コンテンツ・タイプは、非同期の方法 (新規の E メールなど) で通知をエンド・ユーザーに送信することを可能にします。SI には、

サービスを指示する短いメッセージとユニフォーム・リソース・インディケータ (URI) が含まれます。

メッセージが受け取られると、エンド・ユーザーに表示されます。ここでエンド・ユーザーは即時にサービスを開始するか、延期して後に実行するかを選択します。SI が延期されると、クライアントはそれを保管してユーザーが後で SI を処理できるようにします。

14.4.5 over-the-air プロトコル (OTA) のプッシュ

OTA はコンテンツを PPG からワイヤレス・ネットワークを介してクライアントのユーザー・エージェントに伝送することを担当します。

OTA は WSP セッションを使用してコンテンツを送達することができます。しかし、WSP セッションは接続指向モードで作動するため、プッシュされるコンテンツの送達よりも前にクライアントによって確立されていなければなりません。この場合、アクティブな WSP セッションがない場合に、クライアントのセッション開始アプリケーションがセッションを確立しなければなりません。クライアントのこの新しい機能は、OTA サーバーからのセッション要求を受信して WSP セッションをプッシュ目的でセットアップすることにより応答する、サーバーのような働きをします。

クライアントはプッシュ・セッションを開始する前に、OTA サーバーの識別情報をそのようなサーバーのリストに照らして確認することができます。

14.4.6 クライアント・サイド・インフラストラクチャー

接続指向のプッシュには、アクティブな WSP セッションが必要です。

514 ページの 14.4.5 『over-the-air プロトコル (OTA) のプッシュ』で説明されているように、PPG とのプッシュ・セッションをセットアップするためには、クライアントには特別な *セッション開始アプリケーション* (SIA) が必要です。PPG からセッション要求を受け取った後、SIA は PPG とのセッションを確立して、どのクライアント・アプリケーションが新規にオープンしたセッションによって、コンテンツを受け取るかを報告します。適切なアプリケーションが使用可能ではなく、インストールされてもない場合、SIA は PPG からのセッション要求を無視することもできます。

クライアントがプッシュされたコンテンツを受け取る時、ディスパッチャーがプッシュ・メッセージ・ヘッダーを調べてその宛先アプリケーションを判別します。このディスパッチャーには、宛先アプリケーションがインストールされていない場合にコンテンツを拒否する働きがあります。さらに、PPG への操作を確認する働きもあります。

14.4.7 セキュリティー

トラステッド環境で、いくつかの質問が生じることがあります。以下に例を示します。

- プッシュ・イニシエーターをどのようにして認証することができますか。
- セキュリティーおよびトラステッド・モデルでは PPG にどのような役割がありますか。
- プッシュ・イニシエーターおよびプッシュされたコンテンツについて、どのようなアクセス制御ポリシーがありますか。
- 証明書がない場合、クライアントはどのようにそれを認証することができますか。

14.4.7.1 プッシュ・イニシエーターの認証

以下のソリューションのいくつかを使用して、プッシュ・イニシエーターと PPG との間に安全保護環境をインプリメントすることができます。

- セッション・レベル証明書 (TSL および SSL) の使用
プッシュ・コンテンツがプッシュ・イニシエーターと PPG との間でインターネットを横断する場合、TSL または SSL を使用できます。
- オブジェクト・レベル証明書の使用
証明書を使用して、プッシュされたコンテンツをエンドツーエンドに署名または暗号化 (もしくはその両方を) することができます。これにより、クライアント側でコンテンツの認証性に対する信頼レベルを高めることができます。
- HTTP 認証
ユーザー ID およびパスワードによる基本認証を使用できます。さらに、HTTP 認証 (ダイジェストに基づくものなど) をインプリメントすることもできます。
- テクノロジーの組み合わせ
別のアプローチとして、テクノロジーを組み合わせることにより、PPG で TLS/SSL セッションを使用しながら、HTTP 認証を使用してプッシュ・イニシエーターを認証することができます。その後、署名または暗号化 (もしくはその両方) をしたコンテンツをこの認証済みセッションと共に送信できます。

14.4.7.2 クライアント認証

クライアントおよび PPG がトラステッド環境を作成するのであれば、PPG はそのクライアントのためにプッシュ・イニシエーターを認証することができます。つまり、トラストは推移可能なものとなります。クライアント・システム内でトラステッド PPG のリストを保守することにより、クライアント

と PPG との間にトラスト状態を確立することができます。プッシュ・イニシエーターは、発信元サーバーのエンドツーエンド認証のための証明書または公開鍵を所有しています。

14.5 WAP プロトコル・スタックの概要

WAP プロトコル・スタックは、従来のネットワーク層スタック・モデル内で機能を提供するいくつかのプロトコルからなります。これについては、図 197 で概要が示されています。

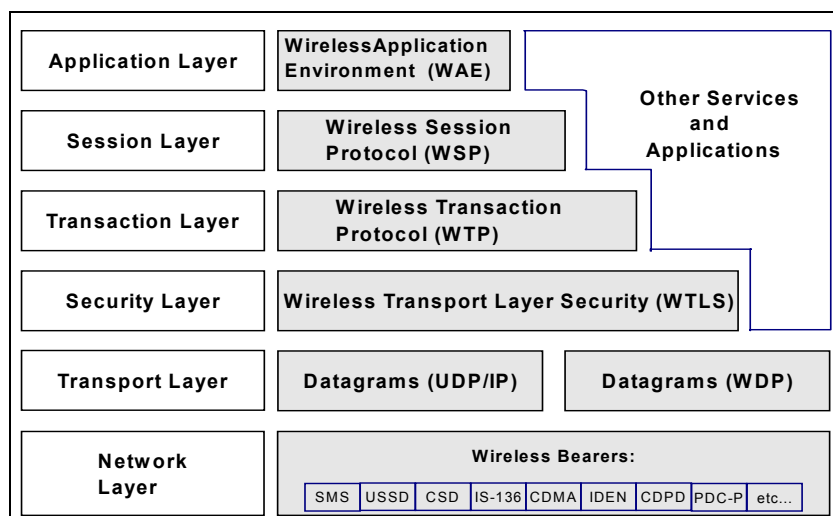


図 197 WAP プロトコル・スタック

14.5.1 ワイヤレス・アプリケーション環境 (WAE)

この層は、類似した WWW 機能および携帯電話テクノロジーからなります。これには WML、WML-Script、および Wireless Telephony Application (WTA) などの機能を含むマイクロブラウザ環境が含まれています。

WTA によって、ユーザーは携帯電話機能、およびカレンダーや電話帳機能など WAE で指定されていない他のユーザー・エージェントと対話することができます。詳細については、518 ページの 14.5.2 『Wireless Telephony Application (WTA)』を参照してください。

WAE には異なるサービスを実行するための異なるユーザー・エージェントが含まれています (517 ページの図 198 を参照してください)。ユーザー・エージェントは、ユーザーおよびワイヤレス装置上で現在選択されているサービスへのインターフェースです。これらのサービスの例としては、WML、

WMLScript、および WTA などがあります。ユーザー・エージェントは URL を使用して、特定のサービス・コンテンツにサーバーをアドレッシングします。ユーザー・エージェントは、サービスの URL を指定した要求を WAP ゲートウェイ / プロキシに送信します。WAP ゲートウェイ / プロキシは IP ホスト・アドレスによって URL を解決します。

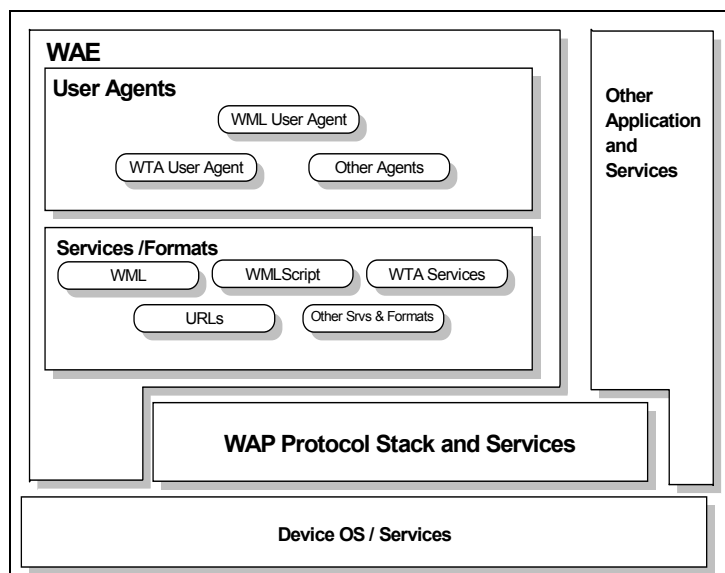


図 198 WAP クライアント・コンポーネント

WAE は相互運用できるデータ交換のためのコンテンツ・フォーマットのセットを定義しています。データ交換の方法は、データおよびターゲットとなる WAE ユーザー・エージェントに依存しています。2 つの主要なフォーマットは、エンコードされた WML および WMLScript バイト・コード・フォーマットです。これらのフォーマットは主に、WAP 端末が処理電力を最小にするため、および帯域幅の使用を小さくするために設計されています。それに加えて、以下のようなデータ型のためのフォーマットもあります。

- イメージ

ピクセルの深さ、カラー・スペース・テーブル、小さなエンコード、低い CPU および RAM エンコードおよび表示要求、および共通ツールの使用可能性、その他の多肢選択をサポートしています。

- 複数パーツ・メッセージ

複数のタイプによるコンテンツを WSP 上で交換するために最適化された、複数パーツ・エンコード体系をサポートしています (518 ページの 14.5.3 『Wireless Session プロトコル (WSP)』を参照してください)。

- ・ ユーザー・エージェント固有のフォーマット

これはユーザー・エージェント間でデータを交換するための 2 つの追加フォーマットをサポートする機能です。それらは、電子ビジネス・カード (vCard 2.1)、および IMC によって指定された電子カレンダーおよびスケジュール交換フォーマット (vCalender 1.0) です。

WAE は、ワイヤレス・テレホン・アプリケーションをサポートする WTA 特定のフォーマットもサポートしています。

14.5.2 Wireless Telephony Application (WTA)

WTA には、テレフォニー・アプリケーションを作成するためのツールが備わっています。それは基本的に、ネットワーク・オペレーター、キャリア、および機器ベンダーのために設計されています。ネットワーク・セキュリティおよび信頼性が主要な考慮事項となります。標準の WML/WMLScript ブラウザーに拡張機能が加えられ、追加の WTA アプリケーション・プログラミング・インターフェース (WTAI) がサポートされています。

WTAI には、呼び出し制御、ネットワーク・テキスト・メッセージング、電話帳インターフェース、およびイベント処理などの機能が備わっています。

WTA は TCP/IP の範囲外であるため、ここでは詳しく扱いません。以下の URL で、WAP の公式資料を参照してください。

www.wapforum.org

14.5.3 Wireless Session プロトコル (WSP)

WSP はクライアント (516 ページの図 197 を参照してください) とサーバーとの間に信頼性のあるセッションを確立します。このサーバーは WAP ゲートウェイ / プロキシであり、確立されたセッションを適切な方法で解放します。WSP セッション確立は、ネゴシエーション機能を使用してプロトコル機能の共通レベルと呼応します。WSP はコンパクト・エンコードを使用してクライアントとサーバーとの間でコンテンツを交換し、さらに通信割り込みの制御も行います。通信割り込みが発生するのは、SMS などのベアラーが変更された場合です (516 ページの図 197 を参照してください)。

WSP は、次の 2 つのプロトコルを定義しています。

1. トランザクション・サービス上での接続モード・セッション・サービス

このモードは、長期接続のために使用されます。セッション状態は維持されます。接続モード・セッションによって送信されたデータには信頼性があります。

2. データグラム・トランスポート・サービス上の非確認、非接続サービス

このサービスは、アプリケーションが信頼性のあるデータ送達を必要としていない場合で、確認の必要がないときに適しています。セッションを実際に確立していなくても、このサービスを使用できます。

WSP は Hypertext Transport プロトコル (HTTP) 1.1 に基づくセマンティクスおよびメカニズムを提供し、ワイヤレス・ネットワークとその WAP 端末のための機能強化を備えています。

14.5.3.1 基本機能

WSP 設計のコアは、HTTP 1.1 のバイナリー形式です。

コンテンツ・ヘッダーを使用して、コンテンツ・タイプ、文字セットエンコード、言語、その他が定義されています。プロトコル・オーバーヘッドを少なくするために、コンパクト・バイナリー・エンコードがウェルノウン・ヘッダーに定義されています。複合データ・オブジェクト内の各コンポーネントにコンテンツ・ヘッダーを提供する短縮データ・フォーマットがサポートされています。これは、HTTP 1.1 によって使用される MIME-multipart/multimixed フォーマットと意味的に等しいバイナリー形式です。

セッション確立プロセスの一部として、セッションの存続期間中一定に保たれる要求および応答ヘッダーは、クライアントおよびサーバー内のサービス利用者間で交換できます。WSP は、クライアントおよびサーバー・ヘッダーおよび要求および応答ヘッダーを、追加したり除去したりすることなく通過します。

WSP セッションのライフ・サイクルは、基礎となる移送プロトコルによって制限されることはありません。最初に確立するオーバーヘッドなしでセッションを中断および再開できるようにする、セッション再確立プロトコルが定義されています。これにより、使用されていない間はセッションを中断して、ネットワーク・リソースを解放したりバッテリー電力を節約することができます。セッションは別のベアラー・ネットワークで再開することもできます。

14.5.3.2 層から層への通信

セッション層内での、層と層およびエンティティとエンティティの間の通信は、サービス・プリミティブによって実現します。それはセッション層と隣接層との間での情報の論理変換および制御を表しています。

サービス・プリミティブは、提供されるサービスとパラメーターに関連したコマンドおよび対応する応答で構成されます。以下に例を示します。

```
X-service.type (parameter)
```

ここで、X はサービスを提供している層を指定します。WSP では S となります。

service.type については、以下の表を参照してください。

表 21 サービス・プリミティブ

タイプ	省略形	説明
要求	req	高水準が低水準からのサービスを要求するときに使用されます。
指示	ind	ピア（呼び出し要求など）またはサービスの提供者（プロトコルにより生成されたイベント）に対して、次の高位層アクティビティを知らせるために使用されます。
応答	res	層は応答を使用して、次の下位層からの指示の受け取りを確認します。
確認	cnf	要求されたサービスを提供する層は確認を使用して、アクティビティが正常に完了したことを報告します。

サービス・プリミティブを使用するとき、追加のパラメーターも指定可能です。それらは特定のタイプのパラメーターを説明しています。以下に例を示します。

- アドレス

セッションを確立するためのクライアントおよびサーバーのアドレスを記述します。

- ヘッダーおよび本体

HTTP エンティティの本体を記述します。ヘッダーは、クライアントからサーバーまたはその反対方向に送信された要求と応答とを識別します。本体には、メッセージの内容が含まれています。

- 機能

これらはサービス機能です。以下に例を示します。

- 最大トランザクション・データ単位
- コード・ページ名のセット
- 未解決の要求の最大数

機能はクライアントとサーバーとの間でネゴシエーションすることができます。

- プッシュ ID

受信メッセージがサービス・インターフェース上で保留されているセッションのプッシュ・トランザクションであることを示します。

- 理由

サービス提供者は理由タイプを使用して、プリミティブの特定の状態の原因を報告します (プロトコル・エラー、モードの切断、セッションの中断、セッションの再開、など)。

サービス・プリミティブのサンプル

サービス・プリミティブの動作は、図 199 で時間順の表によって図示されています。

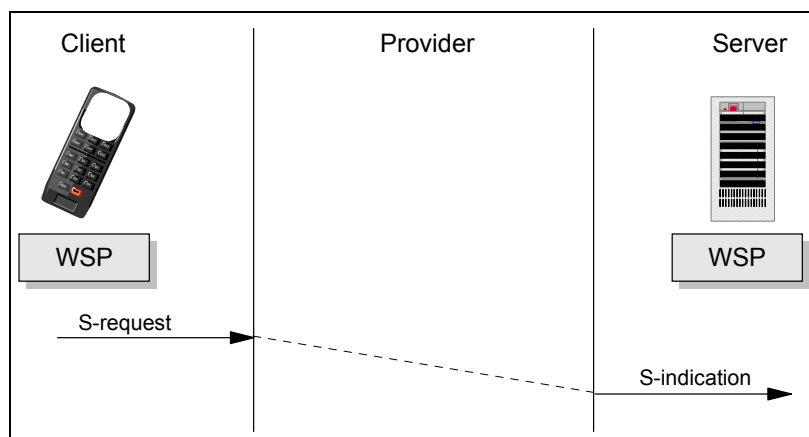


図 199 非確認サービス

図 199 は、単純な非確認サービスを示しています。クライアントは S 要求プリミティブを起動し、その結果、S 指示プリミティブがサーバーに入ります (WSP ピア層)。破線は、一定の期間でのプロバイダーによる伝搬を表しています。

14.5.3.3 セッション・サービスおよび操作

WSP はトランザクション・サービス上で機能するように、および WTP 層を使用しないでデータグラム・サービス上に直接機能するように設計されています。これは、アプリケーションのアーキテクチャーに応じて、WSP が WDP (539 ページの 14.5.6 『WirelessDatagram プロトコル (WDP)』を参照) または WTP (531 ページの 14.5.4 『WirelessTransaction プロトコル (WTP)』を参照) と通信できることを意味しています。522 ページの図 200 は、WAP クライアントから WAP プロキシへの 2 つのセッションを示しています (Web サーバーへのその後の接続は示されていません)。

1 つの WAE アプリケーション (破線を参照してください) は、メッセージをワイヤレス・ネットワークを介して WAP プロキシに送信するときに、WAP クライアント内のセッション層 (WSP)、トランスポート層 (WDP)、およびネットワーク層からのサービスを使用します。WAP プロキシ内で、等しいピア層も受信 WAE に到達するために使用されます。

他の WAE アプリケーション (実線を参照してください) はトランザクション層 (WTP) からの追加サービスを使用します。これは、特定のトランザクションでサービスの追加クラスが必要になるためです (527 ページの『wireless transaction プロトコル (WTP) を使用した WSP 接続モード』を参照してください)。

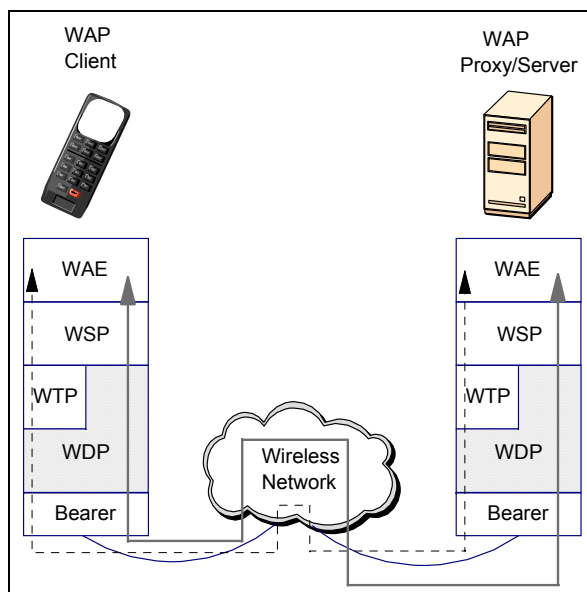


図 200 クライアント/サーバー接続フロー

WSP 接続モード - 概要

接続モードのセッション・サービスは、次の 2 つの標準機能を提供します。

- セッション管理機能

セッション管理機能によって、クライアントは使用する機能およびオプションの合意事項に基づいて、サーバーに接続することができます。セッション確立の際、クライアントはセッションの期間中、属性情報を交換することもできます。セッションを確立できるのはクライアントだけですが、セッションを終了させるのはどちらでも行うことができます。一方の側がセッションを終了させようとする、ピアが起動されます。ユーザー・エージェントにも通知が送られます。

- 例外報告機能

例外報告機能により、サービス提供者はユーザーにセッション中のイベントを通知することができます。

セッション確立中にネゴシエーション機能によって制御される他の機能もあります。それらは、以下のとおりです。

- メソッド呼び出し機能

メソッド呼び出し機能は、クライアントがサーバーに操作を実行して結果をクライアントに戻すように要求することを許可します。これらの操作は、同じ要求 / 応答またはトランザクション・パターンの GET、PUT、その他の HTTP 操作 (RFC 2616 を参照)、またはユーザー定義の操作と比較できます。サービス利用者には、操作の完了、およびそれが成功したか失敗したかが通知されます。

- プッシュ機能

プッシュ機能により、サーバーは非送信請求情報をクライアントに送ることができます。送信される情報は確認されないため、送達には信頼性がありません。

- 確認されるプッシュ機能

確認されるプッシュ機能は、直前の段落で説明した機能と同じです。しかし、この非送信請求情報は、クライアントからサーバーへの受信通知を送ります。

- セッション再開機能

セッション再開機能によって、両方のピアがセッションを中断することができます。現行のセッション状態が保存されます。クライアントがセッションを再開するまで、このセッションを介して通信できなくなります。この機能を使用して、代替ベアラーにセッションを切り替えることもできます。

WSP 接続モード - ネゴシエーション機能

セッション・サービス提供者の操作に関連した情報は、機能によって処理されます。機能ネゴシエーションをピアとピアの間で使用して、サービスの受け入れ可能水準について合意するようにします。

ネゴシエーション・プロセスを開始したピアがイニシエーターとなり、他方のピアはレスポンドーとなります。イニシエーターは機能のセットを提案するだけです。レスポンドーはその提案に合意するか、またはそれを拒否します。

ネゴシエーションは以下のような機能に関して行われます。

- 要求された同じサービスに対する代替アドレスのリスト

- トランザクション・データの最大サイズに関する合意事項
- ヘッダー・コード・ページ
- 未解決のメソッド要求の最大数
- 未解決のプッシュ要求の最大数

WSP 接続モード - 操作

525 ページの図 201 は、クライアントによる正常なセッション確立プロセス、およびそれに続くサーバー内のアクションの起動のサンプルを示しています。いくつかのサービス・プリミティブが、クライアントおよびサーバー・サイドで使用されています。

- サービス・プリミティブ

S-Connect

セッション確立を開始して、正常な実行について通知するために使用されます。クライアント・アドレス (セッションの開始元)、サーバー・アドレス (セッションのターゲット)、クライアントおよびサーバー・ヘッダー (両方のパートナーの要求および応答ヘッダーがセッション全体で使用されることを示す、アプリケーション・レベルのパラメーター)、およびセッションの期間中の要求されてネゴシエーションされた機能などのいくつかのパラメーターが提供されます (たとえば、代替アドレスのリスト、クライアント送信のデータ単位サイズ、サーバー送信のデータ単位サイズ、未解決メソッド要求の最大数、未解決プッシュ要求の最大数、など)。

S-MethodInvoke

操作がサーバーによって実行されるように要求するために使用されます。このサービス・プリミティブを使用するとき、実行後にサーバーからクライアントに結果を戻す *S-MethodResult* も必ず同時に使用します。以下のパラメーターが有効です: クライアントおよびサーバーのトランザクション ID (同じセッションの複数の保留トランザクションを識別するため)、メソッド (どの操作を使用するかを示す (GET や PUT などの HTTP メソッド、または機能ネゴシエーションの際に確立された拡張メソッドの 1 つ)、ユニフォーム・リソース ID の要求 (操作が適用されるエンティティを判別するための URI)、ヘッダーの要求 (HTTP ヘッダーと同じ)、および 本体の要求 (要求に関連したデータを含む)。

S-MethodResult

このサービス・プリミティブは、操作要求に対する応答を戻します。これを起動できるのは、

S-MethodInvoke が先に生じた後だけです。このサービス・プリミティブには、以下のパラメーターを使用できます：クライアントおよびサーバー・トランザクション ID (保留中の複数のトランザクションを区別します)、状況 (HTTP 状況コードに相当するものを使用して (RFC 2613)、クライアントにより S-MethodInvoke で起動された要求済み操作の状態を通知します)、応答ヘッダー (S-MethodInvoke の項目で説明されているように、応答の本体、応答に関連したデータ、または状況がエラーを示している場合は詳細なエラー情報を含みます)、および 受信通知ヘッダー (情報をサーバーに戻すために使用できます)。

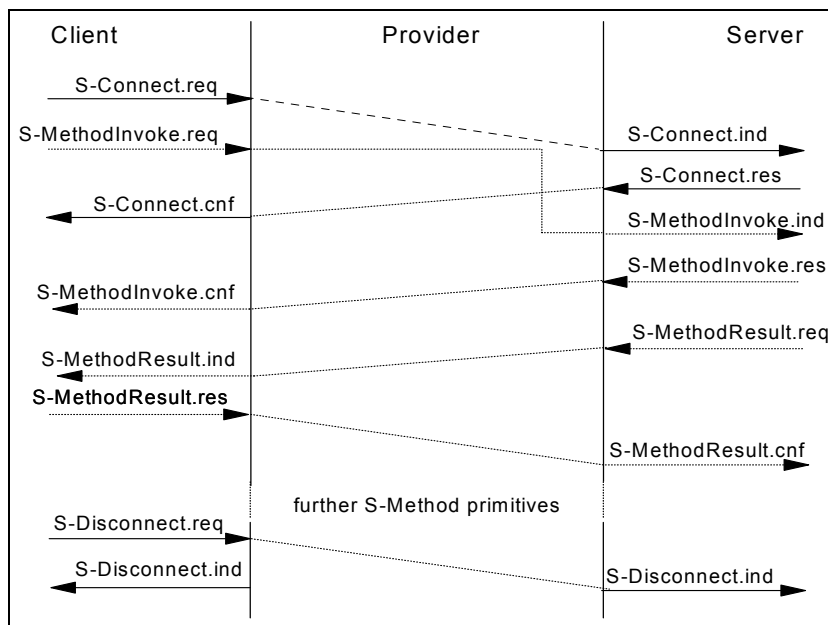


図 201 通常のセッション・フロー：確立、アクション、および切断

・プリミティブの使用方法：

プリミティブは、情報（コマンドまたはデータ）をプロバイダーのネットワークを介してセッション相手側に送信するために使用されます。

- 情報を送信するために、プリミティブ要求タイプ *****.req** が使用されます。
- 送信された情報を通信パートナーが受け取ると、それは同じピア層内のサービスにコマンドが受信されたことを通知し、上の層には上の層のた

めのデータがあることを通知します。この通知は、タイプが指示で拡張子が `***.ind` のプリミティブを介して行われます。

- セッション確立プロセスで、オプション *結果の確認* がネゴシエーションされた場合、タイプが応答で拡張子が `***.res` のプリミティブが直前の要求の発信元に戻されます。
- 応答の受信側は上の層と同じ層のサービスとに、拡張子が `***.cnf` のタイプのプリミティブによって受け取りを通知します。

拡張子の意味は以下のとおりです。

- `***.req` = 要求
 - `***.ind` = 指示
 - `***.res` = 応答
 - `***.cnf` = 確認
- 通常のセッション確立、セッション・アクション、およびセッション終了のフローについての説明。

クライアントのセッション層は、S-Connect.req 要求によって開始します。サーバーのセッション層は、S-Connect.ind 指示によってその上の層（アプリケーション層）に接続要求が受け取られたことを通知します。

サーバーは S-Connect.res 応答プリミティブを使用して、指示プリミティブの受け取りを通知します。クライアントのセッション層は確認プリミティブを使用して、アクティビティ（S-Connect.req）が正常に完了したことを報告します。

セッション開始中に、クライアントのセッション層は S-MethodInvoke.req 要求を送信してサーバーのマシン上で実行するように求めることにより、即時に開始することができます。この要求は、上の層がサーバーの側で操作を実行しなければならないことを示しています。

サーバーは S-MethodInvoke.res 応答をクライアントに送信して、直前の S-MethodInvoke.req 要求が正常に完了したことを確認します。クライアントは S-Method.cnf によって通知を受けます。

サーバーが操作を実行した後、それは結果をクライアントに S-MethodResult.req を介して送信します。クライアントは S-Method.Result.ind を介して上の層（アプリケーション層）に戻されたデータを受け取ったことを知らせます。

セッション相手側はすべての要求 (`***.req`) の確認をネゴシエーションしているので、クライアントは S-MethodResult.req 要求を S-MethodResult.res 応答によって確認します。

それ以降の要求および応答は、このセッションを使用することができます。

この例では、S-Disconnect.req 要求をクライアントを介してサーバーに発行することにより、アクティブ・セッションを終了することができます。サーバーはセッションを終了するための S-Diconnect.ind をセッション層のセッション・サービスに送信します。クライアント・サイドのセッション・サービスは、S-Diconnect.ind を介してセッションを停止するように通知されます。

セッション層は重複している複数のメソッド呼び出しの間で順序付けを行わないために、送信された要求とは異なる順序で指示が送られることもあります。応答および確認、そして対応する S-MethodResult プリミティブについても同様です。順序付けはアプリケーションによって行われる必要があります。

WSP 接続モードのサンプルでは使用されていない他のセッション・サービス・プリミティブもいくつかあります。それらは、以下のとおりです。

S-Suspend	セッションを中断します。
S-Resume	中断されているセッションを再開します。
S-Exception	イベントを報告します。
S-MethodAbort	完了していない操作要求を打ち切ります。
S-Push	非送信請求情報をサーバーから送信します。
S-ConfirmedPush	非送信請求情報をサーバーから送信します。しかし、クライアントがこの情報を確認しなければなりません。
S-PushAbort	確認されたプッシュのプッシュ操作を拒否します。

WSP 非接続

非接続のセッション・サービスは、非確認の機能を提供します。これらを使用して、層と層の間でコンテンツ・エンティティを交換できます。接続モード・サービスと同様に、非接続のサービスは非対称、つまりメッセージの順序付けを行いません。

このサービスは主にプッシュ機能のために使用されます。

以下のサービス・プリミティブが定義されています。

- S-Unit-MethodInvoke
- S-Unit-MethodResult
- S-Unit-Push

wireless transaction プロトコル (WTP) を使用した WSP 接続モード
続くセクションでは、WSP セッションが wireless transaction プロトコル (WTP) サービス上で実行する際の操作について説明します (531 ページの 14.5.4 『WirelessTransaction プロトコル (WTP)』を参照してください)。

WSP セッション内で WTP トランザクションを使用する利点は、TCP と比較できます。WTP は送信側と受信側の間で、信頼性のある送信、選択的に定義された脱落パケットの再伝送、セグメント化、および大型メッセージおよび制御データの再組み立てを提供します。

別の利点は、WTP トランザクション・クラスの使用です。3 つのトランザクション・クラスで、データグラム伝送を定義することができます。それらは、以下のとおりです。

- クラス 0** 結果メッセージのない非確認の起動メッセージ。主に同じソケット・アソシエーションを使用しているセッション内の信頼性の低いプッシュに関して使用されます。
- クラス 1** 結果メッセージのない確認された起動メッセージ。主に WSP が信頼性の高いプッシュ・サービスを実現するために使用されます。
- クラス 2** 1 つの確認済み結果メッセージがある確認済み起動メッセージ。これは WSP がメソッドの起動に使用する基本的な要求 / 応答トランザクション・サービスです。

以下の表では、これら 3 つのトランザクション・クラスがいつ使用されるかを示しています。

表 22 WTP 機能およびトランザクション・クラス

WSP 機能	WTP トランザクション・クラス
セッション管理	クラス 0 およびクラス 2
メソッド呼び出し	クラス 2
セッション再開	クラス 0 およびクラス 2
プッシュ	クラス 0
確認されたプッシュ	クラス 1

セッション・サービスとトランザクション・サービスとの間の協力関係を示すために、いくつかの図が追加されています。これらはセッション・サービスによって使用されるいくつかの異なるトランザクション・サービスのサンプルです。

- トランザクション・クラス 2 を使用する通常のセッション確立
- トランザクション・クラス 0 を使用する通常のセッション終了
- トランザクション・クラス 0 を使用する通常のセッション中断
- トランザクション・クラス 2 を使用する通常のメソッド呼び出し

通常のセッション確立

図 202 は、通常のセッション確立のプロセスを示しています。

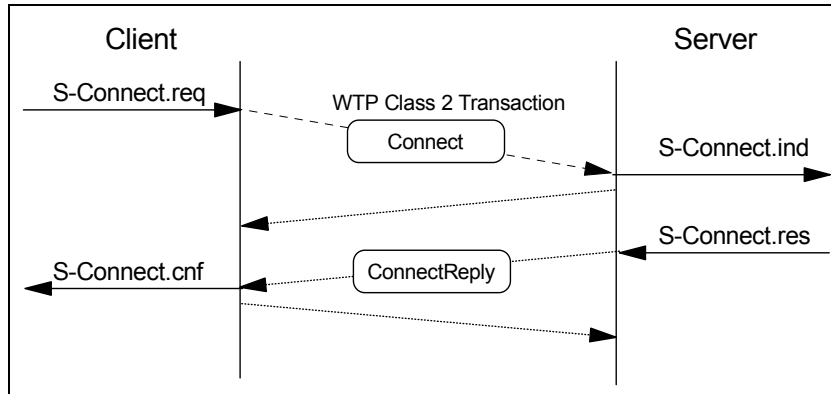


図 202 WSP WTP 通常のセッション確立

通常のセッション終了

図 203 は、通常のセッション終了のプロセスを示しています。

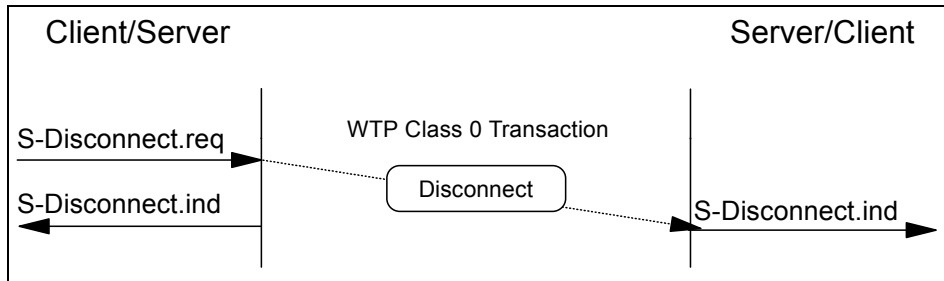


図 203 WSP WTP 通常のセッション終了

通常のセッション中断および再開

図 204 は、通常のセッション中断および再開のプロセスを示しています。

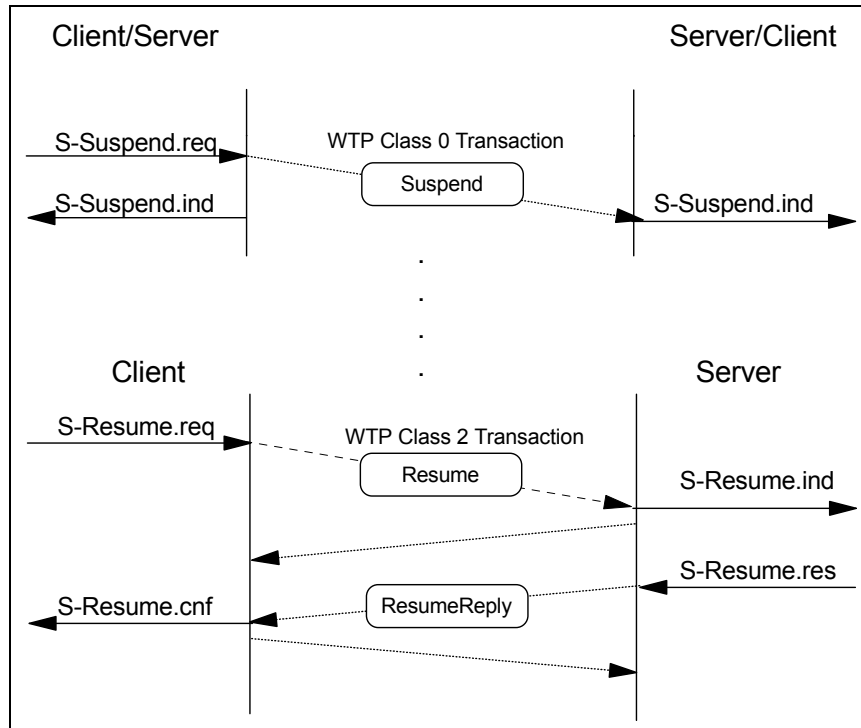


図 204 WSP WPT 通常のセッション中断および再開

通常のメソッド呼び出し

図 205 は、通常のメソッド呼び出しプロセスを示しています。

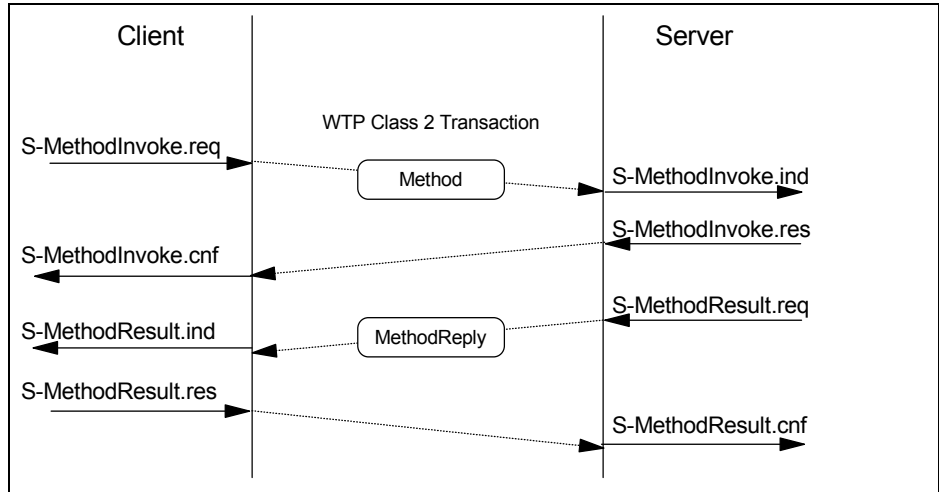


図 205 WSP WTP 通常のメソッド呼び出し

14.5.3.4 イベント処理

セッションはセッションの期間中有効となる固有のセッション ID によって区別されます。各セッションは対等の 4 つの部分からなるアドレスに関連しています。4 つの部分とは、以下のとおりです。

- ・ クライアント・アドレス
- ・ クライアント・ポート
- ・ サーバー・アドレス
- ・ サーバー・ポート

着信トランザクションは、対等の 4 つの部分からなるアドレスに基づいて特定のセッションに割り当てられます。この種類のセッション・アドレッシング体系により、一度に 1 つのセッションが対等な 4 つの部分からなるアドレスの 1 つに結合されます。

14.5.4 WirelessTransaction プロトコル (WTP)

WTP は、ネットワーク上でのリソースが限られており、狭いまたは中間の帯域幅を使用する WAP 端末のために特に設計されています。このテクノロジーにより、帯域幅の使用率が小さくなるので、同じネットワーク上により多くの参加者が存在できるようになります。

WTP は、要求 / 応答の枠組みに基づいて信頼性の低いデータ転送および信頼性の高いデータ転送を提供します。TCP とは異なり、接続のセットアップと切断は行われません。最初のデータが送信される前に SYN および ACK フローが開始される TCP と比較して、WTP はプロトコル交換の最初のパケットにすでに含まれているデータを搬送します。

WTP は TCP のようにストリーム指向ではなく、メッセージ指向で作動します。

14.5.4.1 操作の WTP クラス

操作のクラスには、以下の 3 つがあります。

- クラス 0：結果メッセージのない非確認の起動メッセージ
これは既存の WSP セッションのコンテキスト内に送信できるデータグラムです。
- クラス 1：結果メッセージのない確認済みの起動メッセージ
これは宛先からの応答が期待されていないデータ・プッシュに使用されます。
- クラス 2：1 つの結果メッセージがある確認済みの起動メッセージ
1 つの要求は 1 つの応答を生成します。

14.5.4.2 層から層への通信

サービス・プリミティブを使用して、クライアントの層とサーバーの層との間のトランザクション・トラフィックを制御することができます。WSP に関して説明されたように、これらのサービス・プリミティブには類似の構文があります。

X - generic name. type (parameters)

ここで、X はサービスを提供している層を指定します。トランザクション層では、これは TR となります。プリミティブのタイプとその省略形は WSP に関して説明されたものと同じです (520 ページの表 21 を参照してください)。

上の層に対するサービスのために、3 つのプリミティブがあります。

TR-Invoke	新規のトランザクションを開始します。
TR-Result	前に開始されたトランザクションの結果を送信して戻します。
TR-Abort	既存のトランザクションを打ち切ります。

14.5.4.3 WSP-WTP シーケンス・フローの例

533 ページの図 206 は、プリミティブ・シーケンスのフローを示し、WSP および WTP の要求および応答の関係を示しています。フローはクラス 2 トランザクション、つまり信頼性の高い、確認済みの交換に基づいています。

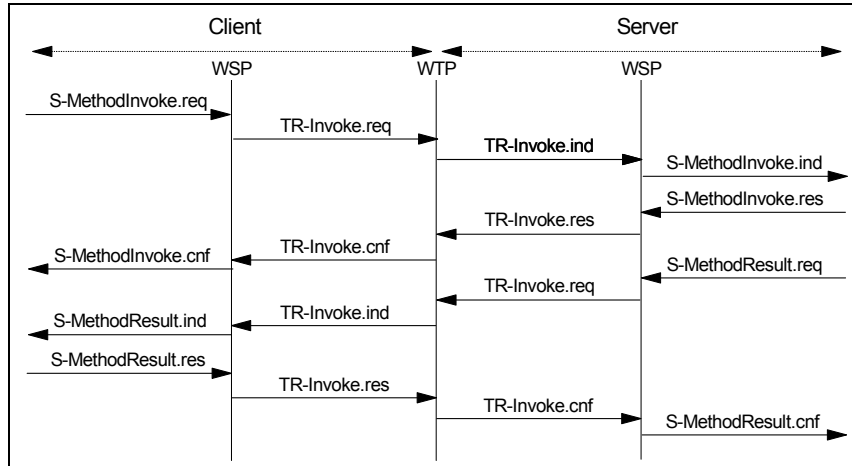


図 206 要求 - 応答のための WSP-WTP プリミティブ・シーケンス

最初のシーケンスはクライアント・アプリケーションによって開始されます (サービスの照会など)。それはセッション内のサーバー・アプリケーションに対する S-MethodInvoke.req 操作によって開始します。

2 番目のシーケンスは、クライアント・スタック内の WSP に起動メッセージ (照会) がサーバーによって受信されたことを確認する、サーバーからの応答です。

3 番目のシーケンスは、サーバーからクライアントに結果の要求 (照会に対する応答) を戻します。

4 番目のシーケンスは、応答が正常に受信されたことを示すサーバーへの受け取り通知を確認します。

トランザクション層のピア WTP のメッセージ層を制御するために、以下の 2 つのタイプのメッセージを使用します。

- ユーザー・データを搬送するデータ・メッセージ
- 制御メッセージ：
 - それらは受け取り通知、エラー報告、その他のために使用されます。
 - ユーザー・データを搬送しない他の制御メッセージ。

14.5.4.4 WTP サポートの要約

WTP サポート：

- 脱落パケットの再送

- 選択的な再送
- セグメント化および再組み立て
- ポート番号アドレッシング (UDP ポート番号が使用されます)
- データ・フロー制御

14.5.5 ワイヤレス・トランスポート層セキュリティ (WTLS)

14.5.5.1 概要

今日 Web 上にある多数のアプリケーションでは、クライアントとアプリケーション・サーバーとの間にセキュアな接続が必要となっています。WTLS は WAP 端末上でセキュア・トランザクションを保証するセキュリティ・プロトコルです。WTLS は業界標準の Transport Layer Security (TLS) プロトコルに基づいています。このプロトコルは、以前は Secure Socket Layer (SSL) として知られていました。

WTLS 狭帯域の通信チャンネルで使用するために最適化されています。これはデータ保全性、プライバシー、認証、およびサービスの拒否による保護を確実にします。TLS を使用する標準のインターネット・セキュリティ技法を採用した Web アプリケーションでは、WAP ゲートウェイは自動的かつ透過的に、そして最小のオーバーヘッドでワイヤレス・セキュリティを管理します。これはエンドツーエンド・セキュリティおよびアプリケーション・レベルのセキュリティを提供します。これには暗号化および暗号化解除のためのセキュリティ機能、強力な認証、保全性、およびキー管理が含まれます。WTLS は暗号アルゴリズムおよびキーの長さに関する各国の規定に準拠して、コンパイルを行います。

WTLS はワイヤレス環境用に特別に調整されたメカニズムを採用しています。たとえば、存続期間の長いセキュア・セッション、ワイヤレス・ネットワーク用に最適化されたハンドシェーク手順、データグラム・ベアラー上で操作するための単純なデータ信頼性などです。

535 ページの図 207 は、WAP アーキテクチャー・モデル内での WTLS の位置、および異なる WTLS プロトコルによる内部 WTLS アーキテクチャーを示しています。

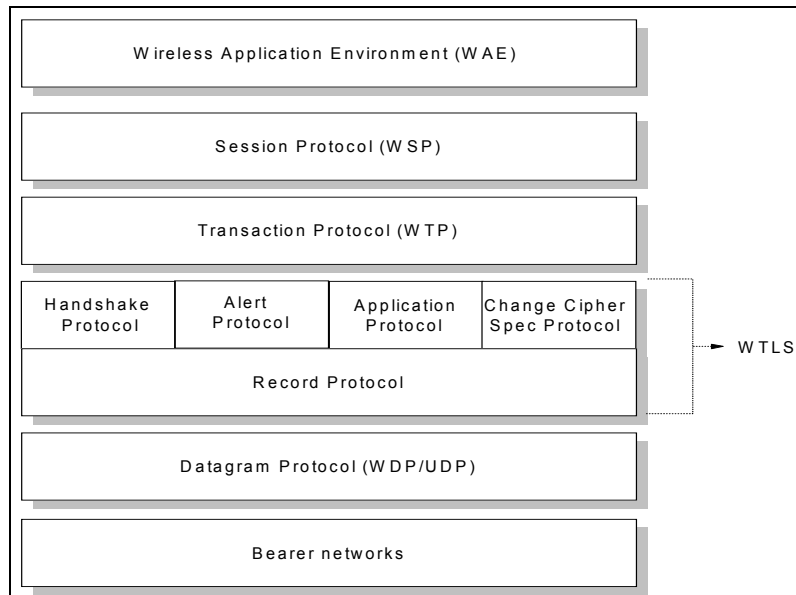


図 207 WAP アーキテクチャー内の WTLS および WTLS コンポーネント

14.5.5.2 WTLS 層の目標

基本的な目標は、以下についてクライアントとサーバーとの間のセキュリティを提供することです。

プライバシー	送信されるデータはクリアなテキストの形式ではないので、他のネットワーク・ユーザーがデータを閲覧することが困難になっています。これはデータ・ストリームを暗号化することによって行われます。
データ 保全性	ネットワーク・ユーザーが送信されたデータを変更した場合、その事実はデータを送信したクライアントまたはサーバーによって検出可能でなければなりません。これはメッセージ・ダイジェストによって行われます。
認証	ネットワーク・パートナーは、当人になりすましている別人とではなく、接続したいパートナーと本当に接続していることを確信できます。これはデジタル証明書によって行われます。
サービスの拒否	正常に検証できなかったデータの拒否および検出。これは WTLS 機能によって行われます。

14.5.5.3 WTLS プロトコルの概要

WTLS は接続指向のプロトコルまたはデータグラム・トランスポートのプロトコル、もしくはその両方によって操作されます。セキュリティ機能は、トランスポート層上のオプションとして実行されます。

WTLS 接続管理

WTLS 接続管理は、クライアントとサーバーとの間のセキュアな通信を提供します。クライアントとサーバーの間のセキュリティ・パラメーターに適合するには、ネゴシエーションによる接続確立プロセス内でいくつかのステップが必要になります。これは Secure Socket Layer (SSL) によるセキュア接続の確立プロセスとほぼ同様です (777 ページの 21.8 『Secure Sockets Layer (SSL)』を参照してください)。

セキュア接続の確立プロセス用のセキュリティ・パラメーターには、暗号アルゴリズム、鍵交換プロシージャ、および認証などが含まれていることがあります。

サービス・プリミティブは、このセキュア接続を開始するためのサポートを提供します。以下のサービス・プリミティブを使用できます。

SEC-Create	以下のオプション・パラメーターを指定してセキュア接続を開始します。 - クライアント証明書 - 鍵交換スイート - 暗号スイート - 圧縮方式 - 鍵のリフレッシュ・ルール - セッション ID
SEC-Exchange	公開鍵による認証またはクライアントとの鍵交換を行います。
SEC-Commit	ハンドシェークが完了して、どちらかのピアが合意した接続状態に切り替えることを要求したときに開始します。
SEC-Terminate	接続を終了します。
SEC-Exception	他方のパートナーに、警告レベルのアラートについて知らせます。
SEC-Create-Request	サーバーがクライアントに新規のハンドシェークを開始する要求を出します。

プロトコルの概要

図 208 は、WAP 層モデル内の WTLS の位置、および WTLS プロトコルを示しています。

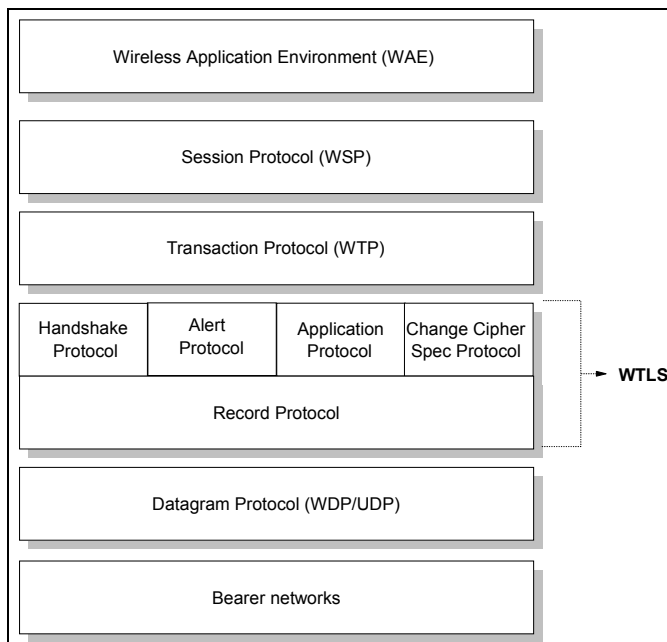


図 208 WAP アーキテクチャー・モデル内の WTLS および WTLS プロトコル

• Record プロトコル

record プロトコルは、上位の層（トランザクションまたはセッション層）および下位の層（トランスポート層）へのインターフェースです。

record プロトコルは、送信するためのメッセージを上位層から受信します。それはオプションでデータを圧縮してから、メッセージ確認コード (MAC) を適用し、メッセージを暗号化して送信します。

受信したデータは暗号化解除されるされ、検査されてから圧縮解除されて、クライアントのより高い層に送られます。

クライアント・インプリメンテーション環境で record プロトコルと密接に協働する 4 つのプロトコルが定義されています。

- Handshake プロトコル

このプロトコルは、ピアがレコード層のセキュリティ・パラメーターに合意することを可能にする 3 つのサブプロトコルからなります。

handshake プロトコルは、クライアントとサーバーとの間のネゴシエーション・プロセスを担当します。以下のパラメーターは、ハンドシェーク中にネゴシエーションされます。

セッション ID	アクティブかつ再開可能なセキュア・セッションを識別します。
プロトコル版番号WTLS	プロトコルの版番号。
ピア証明書	ピアの証明書。
圧縮方式	暗号化の前にデータを圧縮するために使用されるアルゴリズム。
暗号スペック	バルク・データ暗号化アルゴリズム (NULL、RC5、DES、その他、など) および MAC アルゴリズム (SHA-1 など) を指定します。さらに、MAC サイズなどの暗号属性を定義します。
マスター秘密	クライアントとサーバーとの間で共有される 20 バイトの秘密。
シーケンス番号モード	このセキュア接続で使用されるシーケンス番号体系。
鍵のリフレッシュ	いくつかの接続状態値 (暗号鍵、MAC 秘密、および IV) の計算を実行する頻度を定義します。
再開可能	セキュア・セッションを使用して新規のセキュア接続を開始できるかどうかを示すフラグ。

- **Change cipher spec プロトコル**

change cipher spec は、後続のレコードが新規にネゴシエーションされた暗号スペックおよび鍵の下で送信されることを他方のパートナーに通知するために、クライアントまたはサーバーによって送られます。このメッセージはハンドシェイク中のセキュリティー・パラメーターに関して合意した後、しかし終了メッセージの検査が送信される前に送信されます。

- **Alert プロトコル**

アラート・メッセージには、メッセージの重大度およびアラートの記述が含まれています。重大レベルのメッセージは、セキュア接続を終了することがあります。

- **User data プロトコル**

WAP クライアントが送信しようとするペイロードからなります。

暗号計算の詳細については、686 ページの 21.2 『暗号化についての簡単な概要』を参照してください。

14.5.6 WirelessDatagram プロトコル (WDP)

WDP は、WAP アーキテクチャーの上位層 (セキュリティ、トランザクション、およびセッション) に対して一貫性のあるサービスを提供します。これにより、アプリケーションは使用可能な異なるベアラ・サービス上を透過的に実行することができます。これは複数のネットワーク・タイプによってサポートされている異なるベアラ・サービス上で、透過的に通信を行います。WDP は非接続で、信頼性の低いデータグラム・サービスです。これはポート番号アドレッシングをサポートしています。ポート番号は WDP のより高い層レベルを指し示します。それは WTP、WTLS、WSP、またはアプリケーションなどです。

それぞれに特定の機能および特性がある異なるベアラ・サービスをサポートするには、WDP をさまざまなベアラ・サービスの共通層として維持するために適応が必要となります。そのため、WDP は一種の適応層を使用して基礎となるベアラ層と協働します。図 209 は、汎用の WDP 構造を示しています。

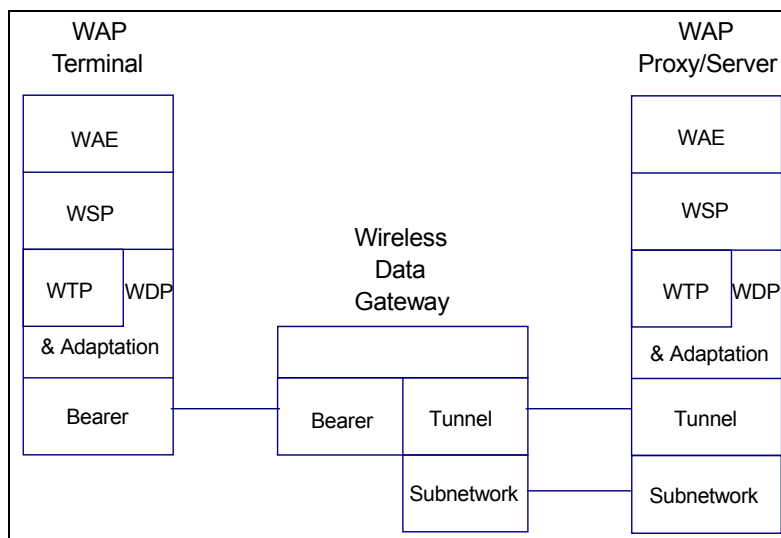


図 209 汎用の WDP アーキテクチャー

WDP メッセージはベアラ・サービスを使用して、WAP 端末によってワイヤレス・データ・ゲートウェイに送信されます。ワイヤレス・データ・ゲートウェイは、WDP パケットを WAP プロキシ/サーバーにトンネリング・プロトコルを経由して渡すこともできます。このプロトコルは、ベアラ・サービスを提供するゲートウェイと WAP プロキシ・サーバーとの間のインターフェースです。たとえば、ベアラ・サービスが GSM SMS である場合、ゲートウェイは GSM SMSC となり、SMSC と他のサーバーとの間のイン

ターフェースとなる特定のトンネリング・プロトコルをサポートすることになります。さらに、2つの通信装置を接続するためにサブネットワークを共通テクノロジーとして使用することも可能です。この接続はたとえば、TCP/IP またはフレーム・リレーに基づく広域ネットワーク、またはイーサネット上で TCP/IP を運用している LAN などを経由したものとなります。WAP プロキシ/サーバーは、アプリケーション・コンテンツを提供すること、またはワイヤレス WTP プロトコルと有線のインターネットとの間のゲートウェイとして機能することができます。

IP ネットワーク層で使用される場合、UDP が WDP の代わりに使用されます。

14.5.6.1 ベアラー層

ベアラー層は、以下のようなベアラー・サービスです。

- IS-136 (GUTS、R-Data、DSD、および Packet Data)
- GSM (SMS、USSD、GPRS、および CSD)
- CDPD
- IDEN
- Flex および ReFLEX
- PHS
- PDC
- CDMA

14.5.6.2 WDP サービス・プリミティブ

サービス・プリミティブを使用して、クライアントの層とサーバーの層との間のトランザクション・トラフィックを制御することができます。これらのサービス・プリミティブには、WSP に関して説明されたものと類似の構文があります。

X - generic name. type (parameters)

ここで、X はサービスを提供している層を指定します。WDP 層では、これは T となります。プリミティブのタイプとその省略形は WSP に関して説明されたものと同じです (520 ページの表 21 を参照してください)。

サービス層には、以下の2つのプリミティブがあります。

T-DUnitdata データをデータグラムとして送信します。

T-DError WDP さらに、WDP プロトコル層が要求された送信を実行できない場合に T-DError プリミティブを受け取ることがあります。

14.5.6.3 IP のための WDP のマッピング

User datagram プロトコル (UDP) は非接続のデータグラム・サービスですが、IP がルーティング・プロトコルとして使用されているワイヤレス・ベアラ・ネットワークに対して WDP プロトコルとして使用されます。UDP はポートに基づくアドレッシングを提供します (宛先およびソース・ポート)。IP はセグメント化および再組み立てを提供します。

以下のベアラ・サービスは、UDP を WDP 定義として採用していることが知られています。

GSM Circuit-Switched Data、GSM GPRS、ANSI-136 R-Data、ANSI-136 Circuit-Switched Data、GPRS-136、CDPD、CDMA Circuit-Switched Data、CDMA Packet Data、PDC Circuit-Switched Data、PDC Packet Data、iDEN Circuit-Switched Data、iDEN Packet Data、PHS Circuit-Switched Data、DECT 接続指向のサービス、および DECT パケット交換サービス。

14.5.6.4 他のベアラ・サービスのための WDP のマッピング

GSM SMS、ANSI-136 GHOST、USSD、その他など、WDP をマップするさまざまなサービスがあります。

14.6 プロトコルの要約

図 210 は、TCP/IP ネットワークに接続した WAP 環境のプロトコル構造を示しています。この図の目的は、異なる環境で使用される異なるプロトコルを示し、WAP プロキシ/ゲートウェイが行う必要のあるプロトコル変換を示すことです。このプロトコル変換に加えて、HTML コードを WML コードに変換した後、ゲートウェイはコンテンツをバイナリ形式にエンコードして帯域幅を最小化しなければなりません。

プロトコル変換が必要になるのは、Web サーバーへの接続時に HTML コードが要求および応答に使用される場合だけです。WML コンテンツまたは WMLScripts が Web サーバーから検索される場合は、プロトコル変換は必要ありません。

WAP クライアントは信頼性の高いトランザクションを使用して、Web サーバーから情報を検索するために WAP プロキシ/ゲートウェイのあるセッションを開始します。クライアントはこの行をクリックすることにより、マイクロブラウザのメニューから必要なサービスを選択します。これにより、Web ターゲット (Web サーバーの IP アドレスおよびポート番号) を識別する URL に関連したトランザクションが開始されます。

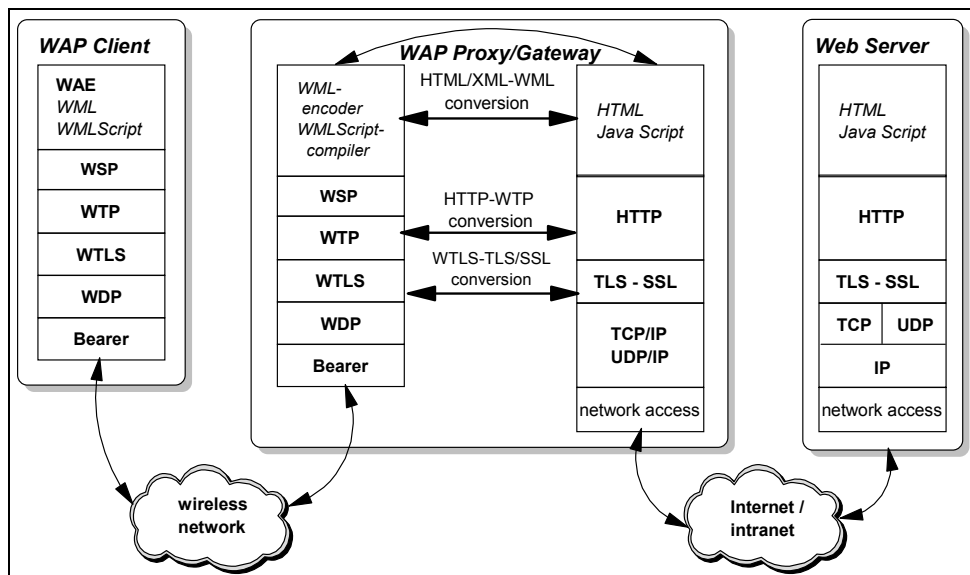


図 210 WAP のプロトコル構造およびインターネット/イントラネット構造

セキュリティがこのセッションでの関心事であれば、WAP クライアントと WAP プロキシ/ゲートウェイとの間にセキュア WTLS 接続がセットアップされます。暗号化、認証、および証明書が、セキュア・セッションのハンドシェイク・プロセス中にネゴシエーションされます。それぞれの環境 (WAP および TCP/IP 環境) には異なるセキュリティ・プロトコルがあるため、WAP クライアントと WAP サーバーとの間にセキュアなエンドツーエンドの接続は存在しないことに注意してください。WAP プロキシ/ゲートウェイと Web サーバーとの間の接続にセキュリティの必要性がある場合、TLS または SSL を使用して 2 つ目のセキュアな接続パスを確立しなければなりません。

WAP プロキシ/ゲートウェイは内部キャッシュ経由でまたはドメイン・ネーム・サービスを使用して URL を解決して実際の IP アドレスにし、HTTP-、TCP-、および IP- ヘッダー構造を作成します。

すべてのヘッダーの作成プロセスが終了した場合、WAP プロキシ/ゲートウェイは Web サーバーとの HTTP/TCP 接続をセットアップして、従来の IP データグラムをインターネット/イントラネットに送信します。インターネットまたはイントラネット経由のパスは、ルーターによって制御されています。

Web サーバーは、HTTP 要求を受け取ります。それは必要な Web ページ、WML デック、または WMLScript を検索してから、HTTP 要求に WAP プロキシ

シー / ゲートウェイのコンテンツを使用して応答します。それは HTTP/TCP 接続の反対側です。

WAP プロキシ / ゲートウェイはコンテンツのエンコード (WLM に基づいている場合)、コンテンツのコンパイル (WMLScript である場合)、または WML へのプロトコル変換 (HTML コードである場合)を行います。その後、エンコードされた応答を適切なセッションおよびトランザクションに関連付けて、WSP/WTP/WDP ヘッダー構造を準備してから、データを WAP クライアントに送信します。

第 15 章 ネットワーク管理

TCP/IP ベースの相互ネットワークの規模および複雑さが増大したため、ネットワーク管理の必要性が非常に重要になっています。

Internet Architecture Board (IAB) は、その推奨事項について詳細に示した RFC 1052 を発行しました。そこでは、次の 2 つの異なるアプローチが採用されています。

- Simple Network Management Protocol (SNMP)。
- ISO Common Management Information Services/Common Management Information Protocol (CMIS/CMIP)。

IAB によれば、短期的には SNMP の使用が勧められています。SNMP は非常に普及したために、IP ベースのネットワークでのレポート管理における業界標準となっています。

15.1 Simple Network Management Protocol および MIB の概要

SNMP ネットワーク管理フレームワークには、以下が含まれます。

- それぞれに SNMP エンティティ (エージェント) がある、多数の管理対象ノード。SNMP エージェントは、マネージャーからの SNMP 要求に応答する管理対象ホストにあるサーバーです。SNMP マネージャーがそのホストを管理できるようにするためには、各 IP ホストに 1 つのエージェントが存在しなければなりません。SNMP エージェントは、それが存在する IP ホスト上の管理情報データベース (MIB) を管理 / サポートします。
- 管理アプリケーション (マネージャー) がある、1 つ以上の SNMP エンティティ。SNMP マネージャーは管理ステーション上で実行するアプリケーションで、通常は SNMP プロトコルを使用して SNMP エージェントから管理データを要求します。
- 各エンティティの管理情報データベース (MIB)。MIB はエージェントに管理されるデータ・オブジェクトを表現したものです。MIB は ASN.1 表記を使用して表現されます。
- エンティティ相互間で管理情報を伝達するプロトコル。エージェントとマネージャーとの間で使用されるプロトコルは SNMP です。エージェントとサブエージェントとの間では、DPI/SMUX/AgentX または任意の所有プロトコルを使用できます。

IAB では、すべての IP および TCP インプリメンテーションをネットワークが SNMP によって管理できることが必要です。つまり、すべてのホスト、

ゲートウェイ、および他の IP を認識する装置には、少なくとも MIB-II をインプリメントする必要があります。

以前のプロトコルである Simple Gateway Monitoring Protocol (SGMP, RFC 1028)、および MIB-I (RFC 1156) を使用することは勧められていませんのでご注意ください。

SNMP はインターネットの標準プロトコルです。その状態は「推奨 (Recommended)」です。その現行の仕様は、RFC 1157 Simple Network Management Protocol (SNMP) にあります。

MIB-II はインターネットの標準プロトコルです。その状態は「推奨 (Recommended)」です。その現行の仕様は、RFC 1213 TCP/IP ペースの相互ネットワークのネットワーク管理の Management Information Base : MIB-II にあります。

詳細については、以下を参照してください。

1. SMI (RFC 1155) - MIB に含まれる管理対象オブジェクトがどのように定義されているかを説明しています。(546 ページの 15.2 『管理情報の構造と識別 (SMI)』で解説します。)
2. MIB-II (RFC 1213) - MIB に含まれる管理対象オブジェクトについて説明しています。(548 ページの 15.3 『Management Information Base (MIB)』で解説します。)
3. SNMP (RFC 1157) - これらのオブジェクトの管理に使用されるプロトコルを定義します。(552 ページの 15.4 『Simple Network Management Protocol (SNMP)』で解説します。)

15.2 管理情報の構造と識別 (SMI)

SMI は、管理対象オブジェクトが記述される方法、および管理プロトコルがこれらのオブジェクトにアクセスする方法についての規則を定義します。管理対象オブジェクトの記述は、データ記述言語である ASN.1 (Abstract Syntax Notation 1, ISO 規格 8824) のサブセットを使用して行われます。オブジェクト・タイプ定義は、次の 5 つのフィールドから成ります。

- Object: オブジェクト記述子と呼ばれるテキスト形式の名前、および以下に定義されている対応するオブジェクト ID。
- Syntax: オブジェクト・タイプの要約構文。これは SimpleSyntax (Integer、Octet String、Object Identifier、Null)、ApplicationSyntax (NetworkAddress、Counter、Gauge、TimeTicks、Opaque)、またはアプリケーション全体に適用される他のタイプから選択できます (詳細については、RFC 1155 を参照してください)。

- Definition: オブジェクト・タイプのセマンティクスに関するテキスト形式の説明。
- Access: read-only、read-write、write-only、または not-accessible のいずれか。
- Status: mandatory、optional、または obsolete のいずれか。

以下に例を示します。

```
OBJECT
    sysDescr { system 1 }
Syntax OCTET STRING
Definition This value should include the full name and version
    identification of the system's hardware type, software
    operating system, and networking software. It is
    mandatory that this contain only printable ASCII
    characters.
Access read-only.
Status mandatory.
```

この例は、Management Information Base (MIB) に含まれているオブジェクトの定義を示しています。この名前は sysDescr であり、システム・グループに属しています (548 ページの 15.3 『Management Information Base (MIB)』を参照してください)。

管理対象オブジェクトは記述する必要があるだけでなく、識別する必要があります。これは ASN.1 Object Identifier を使用して番号のグループを異なる位置に予約することにより、電話番号と同様の方法で行います。TCP/IP ベースのネットワーク管理の場合、割り当てられる番号は 1.3.6.1.2 で、SMI はそれを基礎にして新規のオブジェクトを定義します。

番号 1.3.6.1.2 は、以下の意味を持つ数字のグループを結合することによって取得されます。

- 最初のグループには、ノード・アドミニストレータが定義されます。
 - (1) は ISO です。
 - (2) は CCITT です。
 - (3) は結合された ISO-CCITT です。
- ISO ノード・アドミニストレータに対する 2 番目のグループには、他の組織によって使用されることを示す (3) が定義されています。
- 3 番目のグループには、米国国防総省 (DoD) によって使用されることを示す (6) が定義されています。
- 4 番目のグループでは、DoD がグループの管理方法を指示していないので、インターネット・コミュニティが独自に管理することを示す (1) を想定しています。
- 5 番目のグループは IAB によって以下のとおり承認されています。

- (1) はインターネットで OSI ディレクトリーを使用することを示します
- (2) は管理目的でのオブジェクト識別を示します
- (3) は試験目的でのオブジェクト識別を示します
- (4) は私用でのオブジェクト識別を示します。

上記の例でオブジェクト名の横にある “{ system 1 }” は、オブジェクト ID が 1.3.6.1.2.1.1.1 であることを意味します。これは Management Information Base (MIB) 内の最初のグループ (システム) にある最初のオブジェクトです。

15.3 Management Information Base (MIB)

MIB は TCP/IP プロトコル内の層ごとに管理可能なオブジェクトを定義します。2 つのバージョン、MIB-I と MIB-II があります。MIB-I は RFC 1156 で定義されたもので、現在では推奨されていない状況の歴史的なプロトコルであると分類されています。

MIB-II は RFC 1213 で記述されています。グループ定義は以下のとおりです。

表 23 Management Information Base II (MIB-II) のグループ定義

グループ	オブジェクト	#
システム	基本システム情報	7
インターフェース	ネットワーク接続	23
AT	アドレス変換	3
IP	インターネット・プロトコル	42
ICMP	Internet Control Message Protocol	26
TCP	伝送制御プロトコル	19
UDP	User Datagram Protocol	7
EGP	外部ゲートウェイ・プロトコル	18
SNMP	SNMP アプリケーション・エンティティー	39
凡例 : # = グループ内のオブジェクトの数		

さらに、基礎となるメディアを記述するためのスペースが伝送グループの定義内にあります。

各管理対象ノードは、適切なグループだけをサポートします。たとえば、ゲートウェイが存在しない場合、EGP グループをサポートする必要はありません。グループが適切であれば、そのグループ内のすべてのオブジェクトをサポートする必要があります。

定義済み管理対象オブジェクトのリストは、重要と見なされているエレメントから導出されます。SMI には、MIB の新規バージョンを定義したり、私用または非標準オブジェクトを定義するための拡張性機構が備わっているので、重要なオブジェクトだけを取得するこのアプローチは制限的ではありません。

各グループ内のオブジェクトの例を以下に示します。完全なリストは RFC 1213 で定義されています。さらに、IP、TCP、および UDP の更新情報について、RFC 2011、RFC 2012、および RFC 2013 を参照してください。

- システム・グループ

- sysDescr - システムについての完全な記述 (バージョン、HW、OS)
- sysObjectID - ベンダーのオブジェクト識別
- sysUpTime - 最後の再初期設定から経過した時間
- sysContact - 連絡担当者の名前
- sysServices - 装置が提供するサービス

- インターフェース・グループ

- ifIndex - インターフェース番号
- ifDescr - インターフェース記述
- ifType - インターフェース・タイプ
- ifMtu - 最大の IP データグラムのサイズ
- ifAdminisStatus - インターフェースの状況
- ifLastChange - インターフェースが現在の状況になった時刻
- ifInErrors - エラーを含むインバウンド・パケットの数
- ifOutDiscards - 廃棄されたアウトバウンド・パケットの数

- アドレス変換グループ

- atTable - 使用すべきでない (MIB-I)。
- atEntry - 使用すべきでない (MIB-I)。
- atIfIndex - IfIndex インターフェース番号。
- atPhysAddress - メディア依存の物理アドレス
- atNetAddress - メディア依存の物理アドレスに対応するネットワーク・アドレス

- IP グループ

- ipForwarding - このエンティティが IP ゲートウェイかどうかを示す標識
- ipInHdrErrors - IP ヘッダー内でエラーが生じたために廃棄された入力データグラムの数
- ipInAddrErrors - IP アドレス内でエラーが生じたために廃棄された入力データグラムの数
- ipInUnknownProtos - プロトコルが不明またはサポートされていないために廃棄された入力データグラムの数
- ipReasmOKs - 正常に再構築された IP データグラムの数

- ipRouteDest - 宛先 IP アドレス
- ICMP グループ
 - icmpInMsgs - 受け取った ICMP メッセージの数
 - icmpInDestUnreachs - 受け取った ICMP 宛先到達できないことを示す受信メッセージの数
 - icmpInTimeExcds - 受け取った ICMP 時刻を超過したことを示すメッセージの数
 - icmpInSrcQuenchs - 受け取った ICMP ソース消去メッセージの数
 - icmpOutErrors - ICMP 内で問題が生じたために送信されなかった ICMP メッセージの数
- TCP グループ
 - tcpRtoAlgorithm - 認識されないオクテットを再送するためのタイムアウトを判別するアルゴリズム
 - tcpMaxConn - エンティティーがサポートできる TCP 接続の数の制限
 - tcpActiveOpens - TCP 接続が CLOSED 状態から SYN-SENT 状態に直接変位した回数
 - tcpInSegs - エラーで受け取ったものも含む、受け取りセグメントの数
 - tcpConnRemAddress - この IP 接続のリモート TCP アドレス
 - tcpInErrs - フォーマット・エラーが生じたために廃棄されたセグメントの数
 - tcpOutRsts - 生成されたりセットの数
- UDP グループ
 - udpInDatagrams - UDP ユーザーに送られた UDP データグラムの数
 - udpNoPorts - 宛先ポートにアプリケーションが存在しなかった受け取り UDP データグラムの数
 - udpInErrors - 宛先ポートにアプリケーションが存在しないこと以外の理由で送達できなかった、受け取り UDP データグラムの数
 - udpOutDatagrams - このエンティティーから送信された UDP データグラムの数
- EGP グループ
 - egpInMsgs - エラーなしで受け取った EGP メッセージの数
 - egpInErrors - エラーが生じた EGP メッセージの数
 - egpOutMsgs - ローカルで生成された EGP メッセージの数
 - egpNeighAddr - この項目の EGP ネイバーの IP アドレス
 - egpNeighState - この項目の EGP ネイバーに関連したローカル・システムの EGP 状態

これは完全な MIB 定義ではありませんが、各グループに定義されるオブジェクトの例として示します。これらのモジュールは、現在 IPv4 をサポートしています。

これを例示するため、インターフェース・グループにはノード上のインターフェース接続の数 (ifNumber) およびそれらのインターフェースに関する情報を含む表 (ifTable) の 2 つの最上位オブジェクトが含まれています。その表内の各項目 (ifEntry) には、特定のインターフェースに関するオブジェクトが含まれています。それらの中で、インターフェース・タイプ (ifType) は ASN.1 表記の 1.3.6.1.2.1.2.2.1.3 によって MIB ツリー内で識別されます。そしてトークンリング・アダプターの対応する変数の値は 9 であり、これは iso88025-tokenRing を表します (図 211 を参照)。

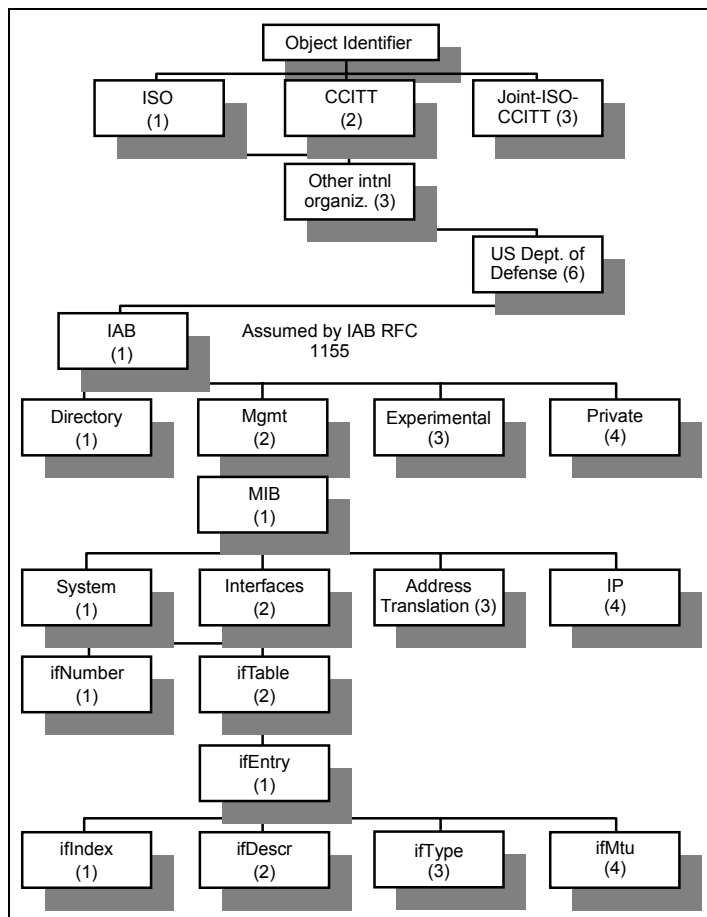


図 211 MIB-II - TCP/IP ベースのネットワークでのオブジェクト ID の割り振り

15.3.1 IBM に特定の MIB 部分

IBM は以下のオブジェクトを MIB-II データベース内に追加しました。

* IBM SNMP エージェント DPI UDP ポート			
DPI_port	1.3.6.1.4.1.2.2.1.1.	number	2
* IBM "PING" 往復時間表			
RTTaddr	1.3.6.1.4.1.2.2.1.3.1.	internet	60
minRTT	1.3.6.1.4.1.2.2.1.3.2.	number	60
maxRTT	1.3.6.1.4.1.2.2.1.3.3.	number	60
aveRTT	1.3.6.1.4.1.2.2.1.3.4.	number	60
RTTtries	1.3.6.1.4.1.2.2.1.3.5.	number	60
RTTresponses	1.3.6.1.4.1.2.2.1.3.6.	number	60

ここで、

- DPI_port は、エージェントとサブエージェントとの間のポート番号を戻します。
- *RTT* によって、SNMP マネージャーはリモート・ホストを PING することができます。RTT は往復時間表 (Round Trip Time table) の略です。
 - RTTaddr: ホスト・アドレス
 - minRTT: 最小往復時間
 - maxRTT: 最大往復時間
 - aveRTT: 平均往復時間
 - RTTtries: これから実行する PING の数
 - RTTresponses: 受け取った応答の数

15.4 Simple Network Management Protocol (SNMP)

SNMP は長年の経験に基づく改良を SGMP に加えて、それが SIM で定義された表記と共に MIB で定義されたオブジェクトで作動するようにしました。

RFC 1157 はネットワーク管理ステーション (NMS) を、ホスト、ゲートウェイ、端末サーバーなどのネットワーク・エレメント (NE) をモニターおよび制御するネットワーク管理アプリケーション (NMA) を実行するものとして定義しています。これらのネットワーク・エレメントは管理エージェント (MA) を使用して、ネットワーク管理ステーションによって要求されたネットワーク管理機能を実行します。Simple Network Management Protocol (SNMP) は、ネットワーク管理ステーションとネットワーク・エレメント内のエージェントとの間で、管理情報を伝達するために使用されます (詳細については、図 212 を参照してください)。

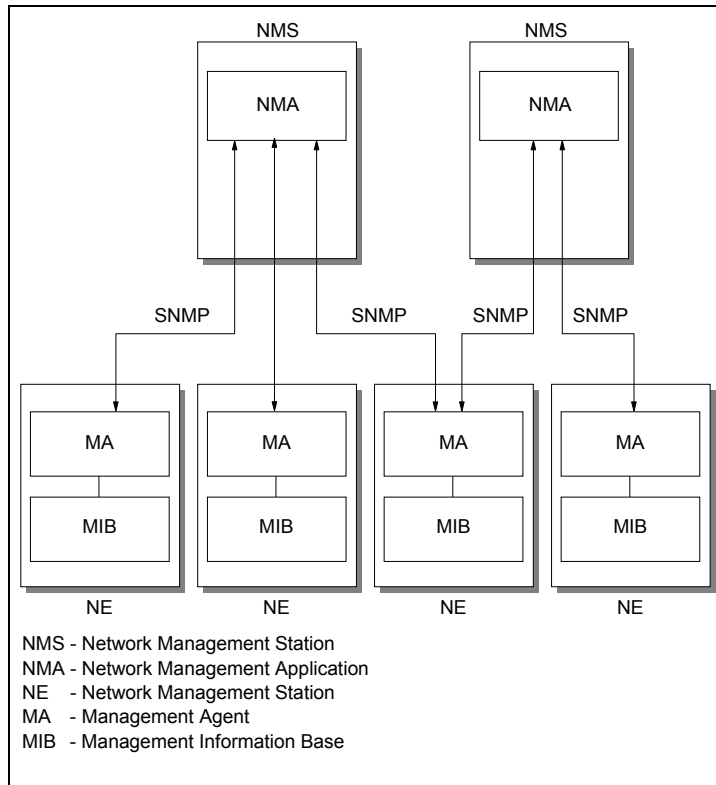


図 212 SNMP - Simple Network Management Protocol のコンポーネント

すべての管理エージェント機能は、重要な管理機能を 2 つに制限してより複雑なプロトコルを回避する変数の交換 (set) または検査 (get) にすぎません。他の方向の NE から NMS では、限られた数の非送信請求メッセージ (トラップ) を使用して非同期イベントについて通知します。同様に、簡潔さを維持するために、情報の交換に必要なのは信頼性のないデータグラム・サービスだけであり、すべてのメッセージは全体的および独立して単一のトランスポート・データグラムによって表されます。さらにこれは、SNMP のメカニズムが多く種類のトランスポート・サービスでの使用に適していることを意味します。RFC 1157 では UDP プロトコルによるメッセージの交換が指定されていますが、多くの種類のトランスポート・プロトコルが使用可能です。

SNMP を使用して互いに通信する管理ステーションとネットワーク・エレメントとに存在しているエンティティは、SNMP アプリケーション・エンティティと呼ばれます。それをインプリメントするピア・プロセスは、プロトコル・エンティティです。SNMP アプリケーション・エンティティの任意のセットを持つ SNMP エージェントは、SNMP コミュニティと呼ば

れます。それぞれの SNMP コミュニティーは、そのコミュニティーに参加しているエージェントに関して固有のオクテットのストリングによって名前が付けられます。

バージョン ID、SNMP コミュニティー名、およびプロトコル・データ単位 (PDU) から成る、SNMP プロトコル内のメッセージ。SNMP のすべてのインプリメンテーションは、以下の 5 つの PDU をサポートしている必要があります。

GetRequest	特定のオブジェクトの値を MIB から検索します。
GetNextRequest	MIB の各部分を移動します。
SetRequest	MIB の特定のオブジェクトの値を変更します。
GetResponse	GetRequest、GetNextRequest、および SetRequest からの応答。
Trap	エージェントの初期化、エージェントの再始動、リンク障害などのイベントをネットワーク管理ステーションに生成するネットワーク・エレメントの機能。RFC 1157 では、coldStart、warmStart、linkDown、linkUp、authenticationFailure、egpNeighborLoss、および enterpriseSpecific の 7 つのトラップ・タイプが定義されています。

これらのメッセージのフォーマットは、図 213 に例示されています。

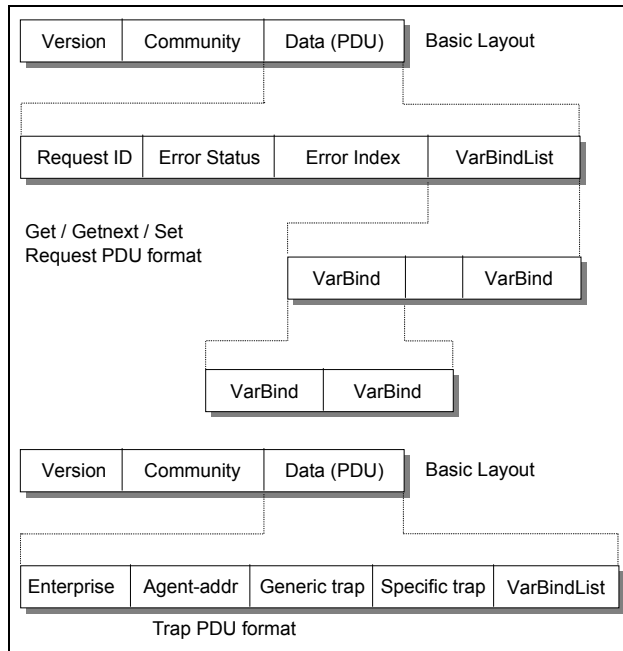


図 213 SNMP メッセージ・フォーマット - request、set、および trap PDU フォーマット

3 つのバージョンの SNMP が使用可能です。それらは通常、SNMPv1、SNMPv2、および SNMPv3 と呼ばれます。SNMP プロトコルが提供するセキュリティ機能は、以下の 2 つのモデルに分類されます。

- コミュニティー・ベースのセキュリティ・モデル。このデータはコミュニティ名と呼ばれるパスワードだけによって保護されています。このレベルのセキュリティは、SNMPv1 および SNMPv2c コミュニティー・ベースのセキュリティ・モデルによって提供されます。
- ユーザー・ベースのセキュリティ・モデル (USM)。これは管理対象情報にアクセスしているユーザーに応じて異なるレベルのセキュリティを提供します。このセキュリティ・レベルをサポートするために、SNMPv3 フレームワークはいくつかのセキュリティ機能を定義しています。それらの機能には、認証とプライバシー用の USM や、ユーザーごとに異なる MIB オブジェクトへのアクセスを制限したり、プライバシーのために認証とデータ暗号化を使用できるようにする、ビュー・ベースのアクセス制御モデル (VACM) などがあります。

15.5 Simple Network Management Protocol バージョン 2 (SNMPv2)

Simple Network Management Protocol バージョン 2 (SNMPv2) のフレームワークは 1993 年 4 月に公開されて、12 の RFC から成っています。その最初のものは RFC 1441 であり、それは紹介です。1993 年 8 月には、12 の RFC すべてが状況が選択的な推奨される規格となりました。

このフレームワークは、以下の規律から成ります。

- 管理情報の識別 (SMI)

MIB モジュールを作成するための OSI ASN.1 サブセットの定義。説明については、RFC 2578 を参照してください。

- テキストの規則

すべての MIB モジュール で使用可能なテキスト規則の初期セットの定義。説明については、RFC 2579 を参照してください。

- プロトコル操作

PDU の送受信に関するプロトコル操作の定義。説明については、RFC 1905 を参照してください。

- トランスポート・マッピング

さまざまなプロトコルの組に使用できるため、SNMPv2 をトランスポート・ドメインの個別セットにマッピングする定義。UDP へのマッピングが推奨されています。RFC はさらに、OSI、DDP、IPX など定義しています。説明については、RFC 1906 を参照してください。

- プロトコル・インスツルメンテーション

SNMPv2 用の MIB の定義。説明については、RFC 1907 を参照してください。

- 管理フレームワーク

SNMPv2 の管理インフラストラクチャー、SNMPv2 のユーザー・ベースのセキュリティ・モデル、およびコミュニティ・ベースの SNMPv2 の定義。説明については、RFCs 1909、1910、および 1901 を参照してください。

- 適合ステートメント

エージェントの表記適合または能力に関する定義。説明については、RFC 2580 を参照してください。

続くいくつかのセクションでは、SNMPv1 から SNMPv2 への主要な差異と改善点について説明しています。

15.5.1 SNMPv2 エンティティ

SNMPv2 エンティティは、SNMPv2 プロトコル操作を使用して SNMPv2 プロトコル・メッセージを生成するか、またはそのメッセージに応答する、もしくはその両方により、ネットワーク管理操作を実行する実際のプロセスです。エンティティの実行可能な操作はすべて、管理的に定義された特定のパーティーに属する実行可能なすべての操作のサブセットに制限することができます (557 ページの 15.5.2 『SNMPv2 パーティ』を参照してください)。SNMPv2 エンティティは、複数の SNMPv2 パーティのメンバーとすることができます。以下のローカル・データベースは、SNMPv2 エンティティによって保守されます。

- SNMPv2 エンティティによって知られているすべてのパーティー用に 1 つのデータベース。それは以下のものとして行うことができます。
 - ローカルに認識されている操作
 - リモートのパーティーまたは装置を持つプロキシ対話によって認識されている操作
 - 他の SNMPv2 エンティティによって認識されている操作
- その SNMPv2 エンティティに知られているすべての管理対象オブジェクトを表す別のデータベース。
- そして、少なくとも 1 つの、既知の SNMPv2 パーティと調和したアクセス権を定義するアクセス制御ポリシーを表すデータベース。

SNMPv2 エンティティは、SNMPv2 エージェントまたはマネージャーとして機能します。

15.5.2 SNMPv2 パーティ

SNMPv2 パーティは、操作がセキュリティーその他の目的のために、特定の SNMPv2 エンティティに関する実行可能なすべての操作の管理的に定義されたサブセットに制限されている、概念上の仮想実行環境です (557 ページの 15.5.1 『SNMPv2 エンティティ』を参照してください)。構造的には、各 SNMPv2 パーティは以下から成ります。

- 1 つの固有なパーティー識別。
- トランスポート・プロトコル・ドメインおよびトランスポート・アドレス情報によって表現される、パーティーが実行する論理ネットワークの位置。
- パーティが発信したすべてのプロトコル・メッセージを発信元および整合性に関して認証するための、1 つの認証プロトコルおよび関連パラメーター。

- ・ パーティーが受け取ったすべてのプロトコル・メッセージを開示されないように保護するための、1つのプライバシー・プロトコルおよび関連パラメーター。

15.5.3 GetBulkRequest

GetBulkRequest は RFC 1905 で定義されているので、プロトコル操作の一部です。GetBulkRequest は、SNMPv2 アプリケーションの要求として生成および伝送されます。GetBulkRequest の目的は、潜在的に大量のデータを転送するように要求することです。それには大きな表を効果的かつ高速に検索すること、および他の事柄が含まれます。大きな MIB オブジェクト・テーブルの場合、GetBulkRequest は GetNextRequest よりも効率的です。GetBulkRequest の構文は以下のとおりです。

```
GetBulkRequest [ non-repeaters = N, max-repetitions = M ]
    ( RequestedObjectName1,
      RequestedObjectName2,
      RequestedObjectName3 )
```

ここで、

RequestedObjectName1, 2, 3

sysUpTime などの MIB オブジェクト ID。オブジェクトは辞書的な番号付きリストで示されます。各オブジェクト ID には、1 つ以上の変数に対するバインディングがあります。たとえば、オブジェクト ID ipNetToMediaPhysAddress には ARP テーブル内の各 IP アドレスに対する変数バインディングがあり、その内容は MAC アドレスに関連付けられています。

N

非リピーター値を指定します。これは、括弧で囲まれたオブジェクト名から最初の N 個のオブジェクトについて、要求で指定したオブジェクトの隣にある変数の内容だけを要求するということです。これは GetNextRequest が提供する機能と同じです。

M

最大反復数値を指定します。これは、要求で指定したオブジェクトの隣にある M 個の変数の内容を、残りの (要求されたオブジェクト数 - N) オブジェクトから要求するということです。GetNextRequest を繰り返すことに似ていますが、1 つだけの要求で伝送されます。

GetBulkRequest を使用すると、次の変数または次の M 個の変数の内容を 1 回の要求で効率的に取得することができます。

SNMPv2 エージェントを実行するホストで、以下の ARP テーブルを想定します。

Interface-Number	Network-Address	Physical-Address	Type
1	10.0.0.51	00:00:10:01:23:45	static
1	9.2.3.4	00:00:10:54:32:10	dynamic
2	10.0.0.15	00:00:10:98:76:54	dynamic

SNMPv2 マネージャーは以下の要求を送信して、sysUpTime および完全な ARP テーブルを検索します。

```
GetBulkRequest [ non-repeaters = 1, max-repetitions = 2 ]
    ( sysUpTime,
      ipNetToMediaPhysAddress,
      ipNetToMediaType )
```

エージェント役割を果たしている SNMPv2 エンティティは、以下の応答 PDU によって応答します。

```
Response ( ( sysUpTime.0 = "123456" ),
  ( ipNetToMediaPhysAddress.1.9.2.3.4 =
    "000010543210" ),
  ( ipNetToMediaType.1.9.2.3.4 = "dynamic" ),
  ( ipNetToMediaPhysAddress.1.10.0.0.51 =
    "000010012345" ),
  ( ipNetToMediaType.1.10.0.0.51 = "static" ) )
```

マネージャー役割を果たしている SNMPv2 エンティティは、以下のよう
に続けます。

```
GetBulkRequest [ non-repeaters = 1, max-repetitions = 2 ]
    ( sysUpTime,
      ipNetToMediaPhysAddress.1.10.0.0.51,
      ipNetToMediaType.1.10.0.0.51 )
```

エージェント役割を果たしている SNMPv2 エンティティは、以下のよう
に応答します。

```
Response ( ( sysUpTime.0 = "123466" ),
  ( ipNetToMediaPhysAddress.2.10.0.0.15 =
    "000010987654" ),
  ( ipNetToMediaType.2.10.0.0.15 =
    "dynamic" ),
  ( ipNetToMediaNetAddress.1.9.2.3.4 =
```

```
        "9.2.3.4" ),  
( ipRoutingDiscards.0 = "2" ))
```

この応答は、テーブルの終了を、マネージャー役割を果たしている SNMPv2 エンティティに知らせます。GetNextRequest では、この同じ情報を検索するために 4 つの要求が必要になります。この例で、GetBulkRequest の最大反復数値を 3 に設定した場合、必要となる要求は 1 つだけです。

15.5.4 InformRequest

InformRequest は、SNMPv2 マネージャー・エンティティ内のアプリケーションが同じく SNMPv2 マネージャー・エンティティ内で実行している他のアプリケーションに、送信元アプリケーションのローカルに存在するパーティーについての MIB ビュー¹ 内の情報を知らせるために生成および送信する要求です。発生元のパーティーにアクセス可能な情報について、他のパーティーのマネージャーに対して表明する手段としてパケットが使用されます (パーティー境界間でのマネージャー間通信)。InformRequest の変数バインディング・リストにある最初の 2 つの変数は、それぞれ sysUpTime.0 および snmpEventID.i² です。他の変数をそれらに続けることが可能です。

15.6 SNMPv2 の MIB

この MIB は SNMPv2 エンティティの動作を説明する管理対象オブジェクトを定義します。

注：

これは MIB-II に代わるものではありません。

内容の概要を知るために役立ついくつかのオブジェクト定義を以下に示します。

```
sysName OBJECT-TYPE  
    SYNTAX      DisplayString (SIZE (0..255))  
    MAX-ACCESS  read-write  
    STATUS      current  
    DESCRIPTION  
        "An administratively-assigned name for this managed node.  
        By convention, this is the node's fully-qualified domain  
        name. If the name is unknown, the value is the zero-length  
        string."  
    ::= { system 5 }
```

¹ MIB ビューは、SMI に応じて定義されたすべてのオブジェクト・タイプのすべてのインスタンスのセットのサブセットです。

² snmpEventID.i は、イベントについての信頼できる識別を示す、SNMPv2 マネージャー間 MIB オブジェクトです。


```
warmStart NOTIFICATION-TYPE
  STATUS current
  DESCRIPTION
    "A warmStart trap signifies that the SNMPv2
    entity, acting in an agent role, is reinitializing
    itself such that its configuration is unaltered."
 ::= { snmpTraps 2 }
```

15.7 新規の管理モデル

SNMPv2 の管理モデルの目的は、管理フレームワークが適用される方法を定義して、さまざまな構成および環境での効果的なネットワーク管理を実現することです。

このモデルでは、SNMPv2 メッセージを交換するピアが明白な識別を使用することになります。そのため、これは元の SNMPv1 のコミュニティー・ベース管理モデルからの離脱を表します。各 SNMPv2 メッセージの送信元および意図された宛先を明白に識別することにより、より便利なアクセス制御モデルをサポートし、将来には非対称（公開鍵）セキュリティー・プロトコルを効果的に使用できるようになるため、この新しいストラテジーは従来のコミュニティー方式を改良したものとなっています。新規のメッセージ・フォーマットについては、図 214 を参照してください。

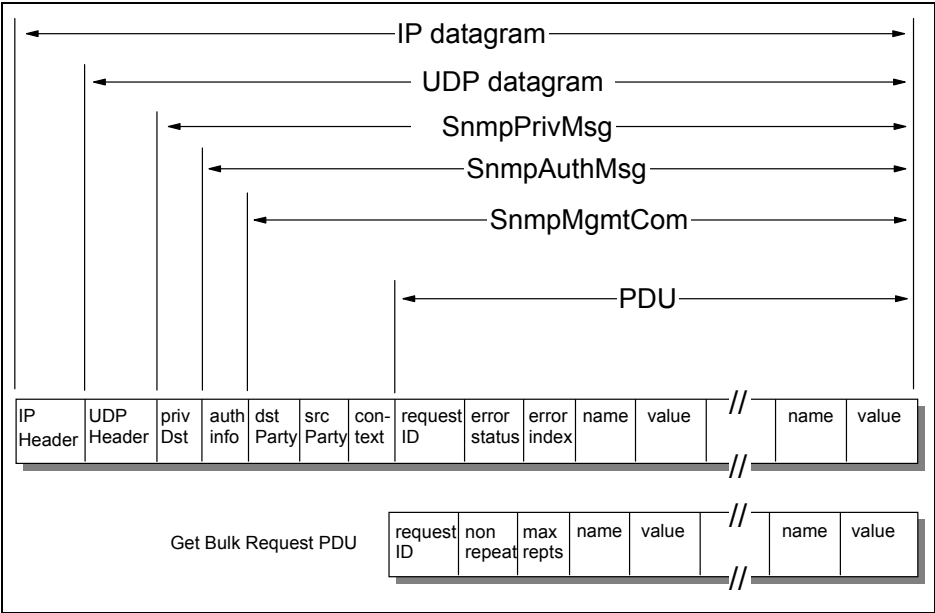


図 214 SNMP バージョン 2 メッセージ・フォーマット

- PDU

以下のプロトコル・データ単位 (PDU) の 1 つを含みます。

- GetNextRequest
- GetRequest
- Inform
- Report
- Response
- SNMPv2-Trap
- SetRequest

前述のように、GetBulkRequest には異なる PDU フォーマットがあります (558 ページの 15.5.3 『GetBulkRequest』を参照してください)。

注：

SNMP トラップは現在、他のすべての要求と同じフォーマットを持っています。

- SnmpMgmtCom (SNMP Management Communication)

ソース・パーティー ID (srcParty)、宛先パーティー ID (dstParty)、および内容を PDU に追加します。この内容は、通信によって参照される管理情報を含む SNMPv2 の内容を指定します。

- SnmpAuthMsg

このフィールドは、そのパーティーが使用する認証プロトコルからの認証情報として使用されます。SnmpAuthMsg は ASN.1 BER³ に基づいてシリアル化されているので、暗号化することができます。

- SnmpPrivMsg SNMP プライベート・メッセージ

SNMPv2 プライベート・メッセージは、SNMPv2 が認証した管理通信であり、(なるべく) 開示されないように保護されています。プライベート宛先 (privDst) が宛先パーティーをアドレス指定するために追加されています。

その後メッセージは、通常の UDP/IP データグラムにカプセル化されて、ネットワークを介して宛先に送られます。

詳細については、上記の RFC を参照してください。

³ ASN.1 BER は、ISO 8825 に従った、OSI Abstract Syntax Notation One の基本エンコード・ルールを表します。

15.8 Simple Network Management Protocol バージョン 3 (SNMPv3)

SNMPv3 は、RFC 2570、RFC 2571、RFC 2572、RFC 2573、および RFC 2574 で説明されています。SNMPv3 は既存の SNMP アーキテクチャーの拡張版です。

SNMPv3 は以下をサポートしています。

- 新規の SNMP メッセージ・フォーマット
- メッセージの認証
- メッセージのセキュリティー
- アクセス制御
- SNMPv2 のサポートの継続

RFC 2574 で説明されているユーザー・ベースのセキュリティー・モデル (USM) は、MD5 およびハッシュ・アルゴリズムを使用して指定します。これにより、データ保全性、セキュリティー、およびプライバシーが提供されます。認証プロトコル HMAC-MD5-96 および HMAC-SHA-96 がサポートされていて、暗号化プロトコル CBC-DES がオプションでサポートされています。

RFC 2575 にあるビュー・ベースのアクセス制御モデル (VACM) は、完全な MIB ツリーのサブセットであるビューを定義する方法を示しています。その後、アクセス制御がこれらのビューで使用可能になります。

SNMP はモジュラー構造なので、個別のモジュールを変更しても他のモジュールは直接の影響を受けません。そのため、SNMPv3 を既存のモデル上に簡単に定義することができます。たとえば、新規の SNMP メッセージ・フォーマットを追加するには、メッセージ処理モデルを更新するだけで十分です。さらに、SNMPv1 および SNMPv2 メッセージも同様にサポートする必要がありますので、新規の SNMPv3 メッセージ・モジュールをメッセージ処理サブシステムに追加することにより、これが実現されます。図 215 はこの構造を例示しています。

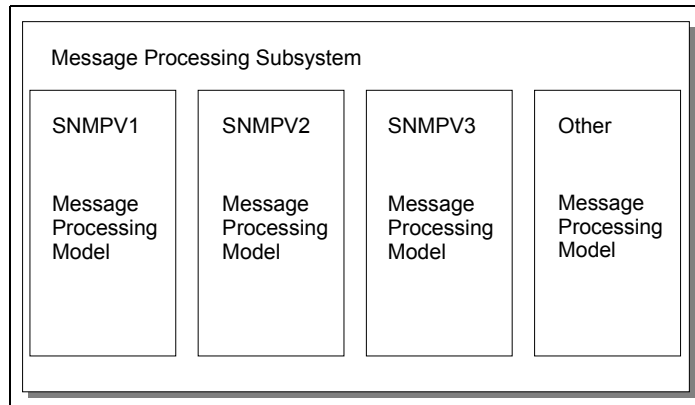


図 215 SNMP - メッセージ処理サブシステム

15.8.1 単一認証およびプライバシー・プロトコル

認証プロトコルは、パーティーによって送信される SNMPv3 管理通信がそのパーティーからのものであることを信頼できる方法で識別するためのメカニズムを提供します。

プライバシー・プロトコルは、パーティーに送信される SNMPv3 管理通信が開示されないように保護するためのメカニズムを提供します。

SNMPv3 セキュリティ・プロトコルは、以下の基本的な危険を回避するための保護を提供します。

- 情報の変更
- なりすまし
- メッセージ・ストリームの変更
- 開示

以下のセキュリティ・サービスは、上記の危険を回避するための保護を提供しています。

- データ完全性

MD5 メッセージ・ダイジェスト・アルゴリズムによって提供されます。128 ビットのダイジェストは、SNMPv3 メッセージの指定した部分に対して計算されて、受信側に送信されるメッセージの一部として含まれます。

- データ起点認証

ダイジェストの前にメッセージの発信元と意図した受信側とで共有している秘密値を、各メッセージにプレフィックスとして付加することにより提供されます。

- メッセージの遅延または再生

各メッセージにタイム・スタンプ値を含めることにより提供されます。

- データ機密性

メッセージの発信元と受信側だけに知られている秘密鍵に従って、メッセージの適切な部分を暗号化するシンメトリック・プライバシー・プロトコルにより提供されます。このプロトコルは、Data Encryption Standard (DES) の一部である、暗号化ブロック・チェーン・モードのシンメトリック暗号化アルゴリズムに関連して使用されます。SNMPv3 メッセージの宛先部分は暗号化されて、受信側に送られるメッセージの一部として含まれます。

15.9 参照資料

以下の RFC は、Simple Network Management Protocol およびシステム内に保持される情報を定義しています。

- RFC1052 – インターネット・ネットワーク管理標準の開発に関する IAB 勧告
- RFC1085 – TCP/IP ペースのインターネットの最上位にある ISO Presentation Service
- RFC1155 – TCP/IP ペースのインターネットの管理情報の構造と識別
- RFC 1157 – A Simple Network Management Protocol (SNMP)
- RFC1213 – TCP/IP ペースのインターネットのネットワーク管理の Management Information Base : MIB-II
- RFC1215 – SNMP で使用するトラップを定義するための規則
- RFC1228 – SNMP-DPI: Simple Network Management Protocol 分散プログラミング・インターフェース
- RFC1239 – 試験的 MIB から標準 MIB への再割り当て
- RFC1351 – SNMP 管理モデル
- RFC1352 – SNMP セキュリティー・プロトコル
- RFC1441 – インターネット標準ネットワーク管理フレームワークのバージョン 2 の概要
- RFC1592 – Simple Network Management Protocol 配布済みプロトコル・インターフェース、バージョン 2.0
- RFC1748 – IEEE 802.5 トークンリング MIB
- RFC1901 – コミュニティー・ベース SNMPv2 の概要
- RFC1904 – Simple Network Management Protocol バージョン 2 (SNMPv2) の適合ステートメント

- RFC1905 –Simple Network Management Protocol バージョン 2 (SNMPv2) のプロトコル操作
- RFC1906 –Simple Network Management Protocol バージョン 2 (SNMPv2) のトランスポート・マッピング
- RFC1907 –Simple Network Management Protocol バージョン 2 (SNMPv2) の Management Information Base
- RFC1909 – SNMPv2 の管理インフラストラクチャー
- RFC1910 – SNMPv2 ユーザー・ベースのセキュリティー・モデル
- RFC2011 – SMIv2 を使用するインターネット・プロトコルの SNMPv2 Management Information Base
- RFC2012 – SMIv2 を使用する伝送制御プロトコルの SNMPv2 Management Information Base
- RFC2013 – SMIv2 を使用する User Datagram Protocol の SNMPv2 Management Information Base
- RFC2570 – インターネット標準ネットワーク管理フレームワークのバージョン 3 の概要
- RFC2571 – SNMP 管理フレームワークを記述するためのアーキテクチャー
- RFC2572 – Simple Network Management Protocol (SNMP) のメッセージ処理およびディスパッチング
- RFC2573 – SNMP アプリケーション
- RFC2574 – SNMPv3 用のユーザー・ベースのセキュリティー・モデル (USM)
- RFC2575 – Simple Network Management Protocol (SNMP) 用のビュー・ベースのアクセス制御モデル (VACM)
- RFC2578 – Simple Network Management Protocol バージョン 2 (SNMPv2) の管理情報の構造
- RFC2579 – Simple Network Management Protocol バージョン 2 (SNMPv2) 用のテキスト変換
- RFC2580 – SMIv2 の適合ステートメント
- RFC2742 – 拡張可能 SNMP エージェントの管理対象オブジェクトの定義

第 16 章ユーティリティー

この章では、各種のアプリケーション・プロトコルおよびユーティリティーについて簡潔に説明します。

16.1 リモート印刷 (LPR および LPD)

ライン・プリンター・リクエストによって、ライン・プリンター・デーモンを実行している他のコンピューター上のプリンターに、あたかも自分のコンピューターであるかのようにアクセスできます。提供されるクライアント (LPR、LPQ、LPRM、LPRMON、または LPRPORTD) によって、ユーザーはファイルを送信したりリモート・プリント・サーバー (LPD) を実行しているホストにプリンター出力をリダイレクトすることができます。これらのクライアントの一部は、ジョブの状況を照会したり、ジョブを削除するためにも使用できます。リモート印刷の詳細については、RFC 1179 を参照してください。

16.2 X Window システム

X Window システム (以後は X と記します) は、最も広く使用されているグラフィカル・ユーザー・インターフェースまたはビットマップ・ウィンドウ表示システムの 1 つです。このシステムは主要なワークステーション・ベンダーすべてによってサポートされています。世界中でますます多くのユーザーがこれを使用しています。X Window システムが提供するものは、未加工の環境だけではありません。それは一意的に統合された商業パッケージのためのプラットフォームも提供します。アプリケーション・ソフトウェアを記述することに加えて、一部の産業グループは所有権を主張できるソフトウェア・パッケージおよびインターフェース用の標準を作成することにより、X Window システムの表示能力を強化しています。その後、これらのパッケージは外観を改良するためにアプリケーションに組み込まれます。

この分野での 2 つの優れた商業パッケージは、オープン・ソフトウェア・ファウンデーションの *MOTIF* および UNIX International の *Open Look* です。X は、Robert Scheifler、Jim Gettys、および MIT の他の研究者たちが、パーソナル・コンピューターとワークステーションの大規模ネットワークのための調査プロジェクトである *Project Athena* の一部として開発したものです。(Project Athena の概要については、*Project Athena: Supporting Distributed Computing at MIT* を参照してください。) この研究の一部として、すべてのシステムに拡張される統合ウィンドウ・システム環境が必要だと判断されました。X は、さまざまな異機種混合のコンピューターおよびネットワークの間で使うことができる、このウィンドウ・システムとして構想されたものです。

Project Athena の進行に伴って、X はポータブルなネットワーク・ベースのウィンドウ・システムとなりました。X に関する初期の作業の多くは、スタンフォード大学で開発された W と呼ばれる既存のウィンドウから派生したものです。実は、X という名前は以前の名前 W をもじったものにすぎません。1988 年に設立された MIT X Consortium は、X Window System を進展させるため、および X Window System インターフェースを標準化するためにコンピュータ産業内で協力を推進するためのものです。

現在の X リリースには、2 つの番号が含まれています。バージョン番号は大きなプロトコルまたは規格の変更を示し、リリース番号は小さな変更を示しています。この文章を執筆している時点での最新バージョンは X11 リリース 6.4 であり、これは X11R6.4 とも呼ばれています。OSF/MOTIF の最新リリースは V2.1.10 (X11R5 に基づく) です。X の大きな改訂の間には互換性はありませんが、大きな改訂カテゴリ内の小さなリリース間には逆方向の互換性があります。

X の目的は、アプリケーションをウィンドウ内で実行するか、または別の仮想端末で実行して最初の画面にはユーザーにそのアプリケーションの存在を知らせるアイコンが表示される (OS/2 プレゼンテーション・マネージャーと同じ機能) ようにしながら、ユーザーが 1 つの画面からすべてのセッションを制御できるようにすることでした。

X Window System は、ローカル・ウィンドウとリモート・ウィンドウの両方を管理する機能を提供します。リモート・ウィンドウは TCP/IP を介して確立され、ローカル・ウィンドウは BSD ソケットを使用して確立されます。X11R6.4 の重要な機能には、インターネットを介してリモート・アプリケーションにアクセスできるということが含まれます。インターネット接続は相対的に低速なので、データを圧縮して、それに応じて送信する必要があります。この目的で、最新のリリースには Low Bandwidth X (LBX) 拡張機能が追加されています。これにより、インターネット・セキュリティに関する別の問題が生じます。X Window の最新のリリースには、この問題を解決するためのセキュリティ機能があります。

16.2.1 機能上の概念

基本的には、次の 2 つの部分が互いに通信しています。

1. アプリケーション。これはユーザーからの入力を受け入れて、コードを実行してから、出力をユーザーに送り戻します。ディスプレイに直接読み書きする代わりに、アプリケーションは Xlib プログラミング・インターフェースを使用して、データをユーザーの端末との間で送受信します。アプリケーションの部分は、X クライアントとも呼ばれます。
2. ユーザーの端末。これはアプリケーションとの間でデータを送受信するディスプレイ管理ソフトウェアを実行し、X サーバーと呼ばれます。

詳細については、図 216 を参照してください。

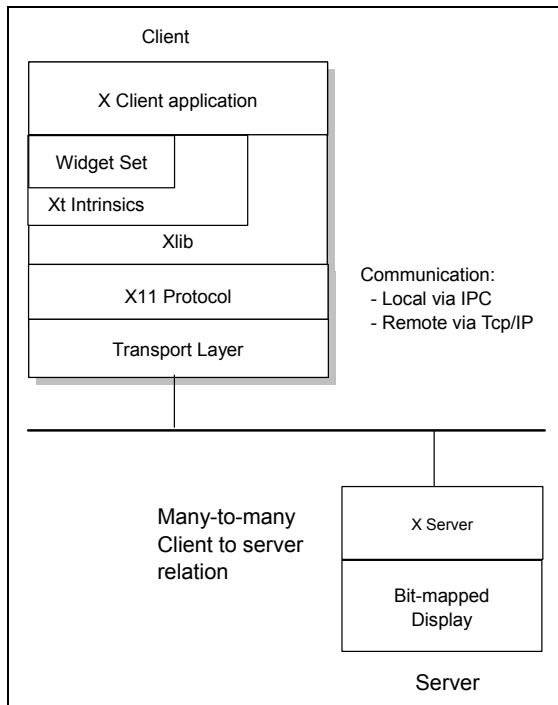


図 216 X Window システムの概念 - クライアントとサーバーが互いに通信する

用語：

X Server

これはユーザーの X クライアント・プログラムの要求に応じて、ユーザーのためにグラフィック端末上に表示を行うサービスを提供する専用プログラムです。これは画面を制御して、1 つ以上の X クライアントのキーボードとマウス（または他の入力装置）を処理します。同様に、ディスプレイへの出力、色のマッピング、フォントのロード、およびキーボード・マッピングも処理します。通常 X サーバー・プログラムは高性能グラフィック PC 上、および X サーバー・プログラムだけを実行するように設計された X 端末上で実行します。

X Font Service プロトコルは、X サーバーがフォント管理のタスクをフォント・サーバーに委任するために使用できます。X11R6.4 X サーバーは、LBX、セキュ

リティー、印刷、および AddGroup 拡張機能をサポートしています。

X クライアント

これは実際のアプリケーションであり、出力を表示するグラフィカル・ユーザー・インターフェースを使用するように設計されています。通常、ユーザーごとのディスプレイごとに、1つのXサーバーによるサービスを求めて多数のXクライアントが競合します。サービスを求める競合は完全に別個のエンティティーであるX Window Managerによって解決されます。Xterm および Xclock は、Xクライアントの2つの例です。

X11R6 のリリース以来、XクライアントはMIT-KERBEROS-5 許可方式を使用しています。接続が確立されると、Xdm は Xlib がクライアントをサーバーに許可するための認証を開始します。

X Window Manager

これはXサーバーが実行するワークステーション上に通常存在するXクライアント・プログラムです。ウィンドウ・マネージャーが存在しなくてもウィンドウを作成できますが、ウィンドウ・マネージャーを使用すると、要求に応じてウィンドウをサイズ調整、移動、または変更することが可能になります。

X プロトコル

これはネットワーク接続内で実行して、クライアントとサーバーとの間で要求および応答を可能にします。これは信頼性のあるバイト・ストリーム接続 (つまり TCP) を使用して、この接続を介してクライアントとサーバーとの間で交換されるメッセージの書式について説明します。

Xlib

基本的なアプリケーション・プログラミング・インターフェースがXlibに含まれています。これはすべてのXクライアントに組み込まれたCプリミティブ・サブルーチンの集合であり、Xプロトコルへの最低レベルのアクセスを提供します。Xlib内のプロシーチャーは、クライアント要求をXプロトコル要求に変換し、Xサーバーからの着信メッセージ (イベント、応答、およびエラー) を解析し、ストレージ管理やオペレーティング・システムに依存しない操作などいくつかの追加のユーティリティーを提供します。アプリケーション・プログラム全体をXlibで記述することも可能です。実際に、既存のXクライアントのほとんどはこの方法で開発されています。Xlibの他の優れた機能としては、装置独立のカラー、国際言語、ローカル・カ

スタム、および文字ストリング・エンコードのサポートがあります。

X Toolkits

低レベル Xlib インターフェースおよび基礎となる X プロトコルの複雑さは、種類が増加している使用可能な X ツールキットによって扱われます。X ツールキットは、ボタン、メニュー、スクロール・バーなどの共通ユーザー・インターフェース・オブジェクト、およびこれらのオブジェクトをディスプレイ上に編成するためのレイアウト・ツールをインプリメントするための高レベル機能を提供するソフトウェア・ライブラリーです。ツールキットのファミリーの基礎は、MIT からの標準 X リリースに備わっています。このライブラリーは X Intrinsics または Xt と呼ばれ、ウィジェットと呼ばれるユーザー・インターフェース・オブジェクトのセットのための構築ブロックを形成します。

ウィジェット (widgets)

X Intrinsics に基づくツールキットでは、ウィジェットと呼ばれる共通インターフェース・メカニズムが使用されます。ウィジェットは本質的に、X Window およびいくつかの追加データ、そしてそのデータを操作するためのプロシージャのセットです。ウィジェットはクライアント・サイドだけからの概念です。X サーバーも X プロトコルもウィジェットを認識できません。サンプルのウィジェット・セットである Xaw は、*Athena Widget Set* としてよりよく知られていますが、X11 ソースと共に MIT によって配布されます。

16.2.1.1 機能性

以下の段落では、X Window システムの機能上の操作について概説します。

- X クライアントと X サーバーは、異なるホスト上に存在することができます。その場合、それらは TCP/IP プロトコルを使用してネットワーク上で通信します。それらは同じマシン上に存在することも可能で、その場合 IPC (プロセス間通信) を使用して通信します (ソケットを介して) 。
- 端末ごとに 1 つだけの X サーバーがあります。複数の X クライアント・アプリケーションが、この 1 つの X サーバーと通信することができます。X サーバーの役割は、アプリケーション・ウィンドウを表示してユーザー入力を適切な X クライアント・アプリケーションに送ることです。
- 作成されたウィンドウを保守することは X クライアントの役割です。ディスプレイ上の表示が他のクライアントによって変更されると、X サーバーからのイベントによって通知されます。しかし、ウィンドウを描画または

再描画するときに、ウィンドウのどの部分が表示されているかを考慮する必要はありません。

- X サーバーはスタックを保守することにより、ウィンドウの可視性を保持します。スタックには、親ウィンドウの「第 1 世代の」子がすべて含まれています。子ウィンドウ自体が 1 つ以上の子ウィンドウを持つことによって親ウィンドウになることもあります。それらはサブスタックに保持されます。1 次スタックは、ルートのすぐ下に存在するすべてのウィンドウを保持するスタックです。図 217 に図が示されています。サブウィンドウが完全に表示されるのは、それらの親が該当するスタックの再上部にあり、ディスプレイにマップされているときだけです。
- X サーバーそのものには管理機能がありません。それはスタックに応じてウィンドウのクリッピングを行うだけです。すべてのクライアントは、そのウィンドウを管理します。すべてのクライアントの最上位ウィンドウを操作する、ウィンドウ・マネージャーがあります。ウィンドウ・マネージャーは X サーバーの一部ですが、それ自体がクライアントとなります。ウィンドウ・マネージャーは、スクリーン上に表示されているものを変更すると（ウィンドウをサイズ調整するなど）、X サーバーが他のすべてのクライアントにエクスポージャー・イベントを送信するようにします。
- クライアント・アプリケーションは、要求メッセージを X サーバーに送信して、X サーバーは応答メッセージまたはエラー・メッセージで応答します。X サーバーはイベント・メッセージをアプリケーションに送信することもできます。イベント・メッセージは、ウィンドウ（およびその可視性）に対する変更、およびユーザー入力（マウスとキーボード）について通知します。

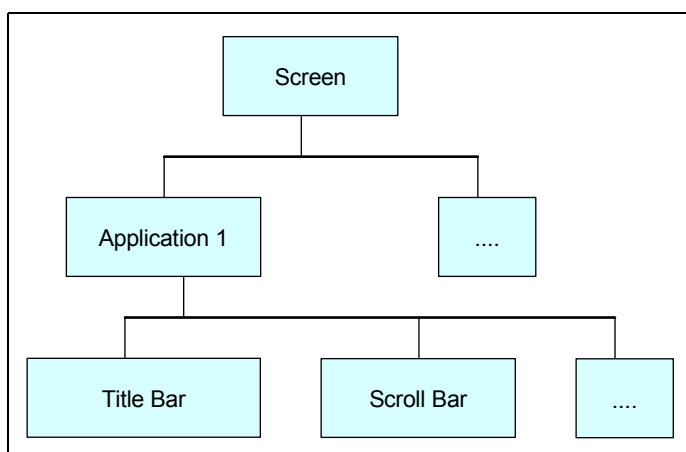


図 217 X Window システムのウィンドウ構造 - 各ウィンドウは他のウィンドウの子となる

このクライアント / サーバーの概念を適用すると、以下の利点があります。

- アプリケーションは端末のハードウェア特性を知っている必要がない。
- アプリケーションは端末と同じコンピューター上に存在する必要がない。
- Xlib に記述されたプログラムは移植可能である。
- 適切な X サーバーを指定することにより、新規の端末タイプを追加できる。
- プログラマーが通信に関して処理する必要がない。ユーザーがリモートの場合もローカルの場合も、プログラマーはグラフィック・アプリケーションを Xlib に記述するだけです。

16.2.2 プロトコル

X Window System プロトコルは、信頼できる任意のバイト・ストリーム・トランスポート機構の最上部にインプリメントすることができます。それは簡単なブロック・プロトコルをストリーム層の最上部に使用します。以下の 4 種類のメッセージが使用されます。

- 要求フォーマット：要求はクライアント・アプリケーションから X サーバーにフローします。

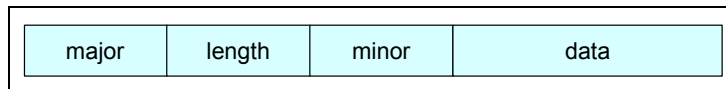


図 218 X Window 要求フォーマット

ここで、

- major および minor はそれぞれが 1 バイトの長さの命令コードです。
- length の長さは 2 バイトです。
- data の長さは要求に応じてゼロバイト以上です。
- 応答フォーマット：32 バイトのブロック。
- エラー・フォーマット：32 バイトのブロック。
- イベント・フォーマット：32 バイトのブロック。

応答、エラー、およびイベント・メッセージは、X サーバーによって X クライアント・アプリケーションに送られます。

ディスプレイは常に 0 から番号が付けられます。TCP 接続では、番号 N のディスプレイはポート 5800+N (hex 5800) およびポート 5900+N と関連付けられます。X サーバーは 58xx ポート上の接続を下位バイトが先頭となるフォーマットのホストに対する接続として、また 59xx ポート上の接続を高位バイトが先頭となるフォーマットのホストに対する接続として処理します。

可能な異なる要求の数は 100 以上あり、それぞれが 1 つの Xlib アプリケーション呼び出しに対応しています。本書はプログラマーの手引きではないので、Xlib 機能については説明しません。RFC 1013 – X Window System プロトコル、バージョン 11 には、X11 プロトコルの 1987 alpha 更新が含まれています。現行リリース X11R6 についての資料は、MIT または商用コンピューター書籍の出版社に問い合わせてください。

16.3 Network News Transfer Protocol (NNTP)

インターネット上で特に人気のあるアプリケーションの 1 つは、Usenet News としても知られている Network News です。Network News Transfer Protocol (NNTP) に基づいて、インターネット上のユーザーは、科学、教育、コンピューター、ビジネス、政治、娯楽、スポーツなどの多数のトピックに関するニュースグループを閲覧したり、それに寄稿することができます。ニュースグループはニュース・サーバー上に保管されます。NNTP はサーバーからサーバーへの通信、およびクライアントからサーバーへの通信の両方に使用されます。

- クライアントは IBM NewsReader/2 などのニュース・エージェント・アプリケーションを使用して、1 つ以上のニュースグループから記事を受信したり、1 つ以上のニュースグループに記事を投稿することができます。NNTP の詳細については、RFC 977 – Network News Transfer Protocol を参照してください。

16.4 Finger プロトコル

Finger は、ドラフト標準のプロトコルです。その状態は「任意選択 (elective)」になっています。finger の現行の仕様は、RFC 1288 – The Finger User Information Protocol に示されています。

finger コマンドは、リモート・ホストのユーザーに関する情報を表示します。finger は UNIX コマンドです。その書式は以下のとおりです。

```
finger user@host
または
finger @host
```

finger コマンドで表示されるユーザーについての情報は、フィンガー・サーバーのインプリメンテーションに依存しています。ユーザーが指定されていない場合、表示される情報は通常、このホストに現在ログオンしているすべてのユーザーのリストとなります。

接続は TCP ポート 79 (10 進数) を介して確立されます。クライアントは <CRLF> で終了する ASCII コマンド・ストリングを送信します。サーバーは

接続をクローズするまで、1 つ以上の ASCII スtringを使用して応答します。

16.5 Netstat

`netstat` コマンドは、ローカル・ホストのネットワーク状況に関して TCP/IP に照会を行うために使用します。このコマンドの正確な構文は、インプリメンテーションにかなり依存しています。詳細については、インプリメンテーションの *ユーザーズ・ガイド* または *コマンド・リファレンス* を参照してください。これはデバッグのために役立つツールです。

一般的に、`netstat` は以下に関する情報を提供します。

- このローカル・ホストに存在するアクティブな TCP 接続
- このローカル・ホスト上のすべての TCP/IP サーバー、およびそれらが使用するソケットの状態
- TCP/IP が使用する装置およびリンク
- このローカル・ホストで使用されている IP ルーティング・テーブル (ゲートウェイ・テーブル)

`netstat` コマンドは、すべての IBM ソフトウェア TCP/IP 製品にインプリメントされています。

第 3 部 拡張概念および新規テクノロジー

第 17 章 IP Version 6

インターネットは急速に成長しています。2000 年 1 月に実施された最新のインターネット・ドメイン調査によれば¹、109,500,000 を超えるホストが存在しています。32 ビットのアドレス・フィールドを持つ IPv4 アドレッシング体系は 4 億以上のアドレスを処理することが可能であり、インターネット・ホストの数の 40 倍を処理する余地があるため、すべてのホストをインターネットにアドレッシングすることは極めて適切であるかのように思われます。残念ながら、以下を含むいくつかの理由により、そうではありません。

- IP アドレスは、別々に管理されるネットワーク部分とローカル部分とに分けられます。ネットワーク内のアドレス・スペースはあまり満たされていないことがあります。特定の管理可能ドメインにアドレス・スペースの一部 (IP アドレスの範囲) を割り振ると、その範囲内のすべてのアドレスは他の場所に割り振ることができなくなります。
- ネットワークのアドレス・スペースはサイズの異なるクラス A、B、および C ネットワークに構造化されています。それぞれの容量については、個別に検討する必要があります。
- IP アドレス・モデルでは、実際にインターネットに接続されていなくても、すべての IP ネットワークに固有のネットワーク番号を割り当てる必要があります。
- TCP/IP の使用が、接続された PC という従来の範囲を超えて新しい領域に拡大しているので、まもなく IP アドレスの需要が急速に増大すると予想されます。たとえば、相互接続されたハンドヘルド・デバイス、電子的 POS 端末、または Web 対応のテレビ受信機 (これらはすべて、現在入手可能です) の使用が広く拡大しているので、IP ホストの数は非常に多くなります。

これらの要素は、アドレス・スペースによる制約が、単純な分析が示唆するよりも大きいということを意味しています。この問題は、*IP アドレスの枯渇*と呼ばれます。この問題を軽減する方法はすでに実施されていますが、現在の IP アドレス・スペースはいつか枯渇してしまいます。Internet Engineering Task Force (IETF) は IP のこ渇が重大な問題となる時期を予測することを緊急の目標とした *Address Lifetime Expectations (ALE)* の作業グループを発足させました。その最終的な予測 (1994 年 12 月に ALE 作業グループのドラフトで報告された) によれば、IP アドレス・スペースは 2005 年から 2011 年までの間のいつか枯渇します。それ以来、CIDR (Classless Inter Domain Routing - 89 ページの 3.1.7 『クラスレス・ドメイン間ルーティング (CIDR)』を参照) の使用および DHCP の使用拡大という点でグループの見解は変化した可能性があります。他方では現在の増大率が予想を超えていると思われる。

¹ Source: Internet Software Consortium (<http://www.isoc.org/>)

アドレスのこぼれに加えて、IPv4 での他の制約によっても、新しい IP プロトコルを定義する必要が生じています。

1. CIDR を使用しても、最初に IP バックボーン・ルーターに含まれていたルーティング・テーブルが増大して管理不能になっています。
2. トラフィック優先順位つまりサービス・クラスは、IPv4 ではあいまいに定義され、ほとんど使用されず、強制力はありませんでしたが、現代のリアルタイム・アプリケーションでは事情が異なっています。

これらの問題点を踏まえて、IETF は IPng (IP next generation) 作業グループを確立して RFC 1752 – 「IP Next Generation Protocol の勧告」を公表しました。その結果、インターネット・プロトコル バージョン 6 (IPv6) が RFC 1883 で出されました。その後、RFC 2460 が出されたため以前のは古くなっています。

17.1 IPv6 の概要

IPv6 には、以下の重要な機能があります。

- 非常に大きなアドレス・スペース。今後の少なくとも 30 年間はこれで十分だと予想されています。
- アドレス・クラスではなくプレフィックスに基づく、グローバルに固有で階層的なアドレッシング。これにより、ルーティング・テーブルが小さく保たれ、バックボーン・ルーティングが効果的に保たれます。
- ネットワーク・インターフェースを自動構成するメカニズム。
- それ自体および他のプロトコルのカプセル化のサポート。
- データのタイプを識別するサービス・クラス。
- 改良されたマルチキャスト・ルーティング・サポート (ブロードキャストよりも優先される)。
- 組み込まれた認証および暗号化。
- IPv4 からマイグレーションするための遷移メソッド。
- IPv4 と共存し、それと通信するための互換性メソッド。

注：

IPv6 では、データグラムの代わりに パケット という用語を使用します。これらの意味は同じですが、フォーマットが異なります。

IPv6 では、ノード という用語を IPv6 を実行しているシステム、つまりホストまたはルーターに対して使用します。IPv6 ホストは、明示的にそれにアドレッシングされていない IPv6 パケットを転送しないノードのことです。ルーターは、それにアドレッシングされていない IP パケットを転送するノードのことです。

17.2 IPv6 ヘッダー・フォーマット

IPv6 パケット・ヘッダーのフォーマットは、IPv4 でそれに相当するものと比較すると単純化されています。IPv6 ヘッダーの長さは (20 バイトから) 増加して 40 バイトになり、図 219 に示すように 2 つの 16 バイト・アドレス (ソースおよび宛先)、およびその前に置かれる 8 バイトの制御情報が含まれます。IPv4 ヘッダー (94 ページの図 27 を参照) には 2 つの 4 バイト・アドレス、およびその前に置かれる 12 バイトの制御情報があり、その後にはオプション・データが続くことがあります。ヘッダー内の制御情報を縮小し、大部分の IP パケットに関してはオプションを除去したのは、ルーター内でパケットごとの処理時間を最適化するためです。ヘッダーから除去された頻繁に使用されないフィールドは、必要な場合に指定できるオプションの拡張ヘッダーに移動されました。

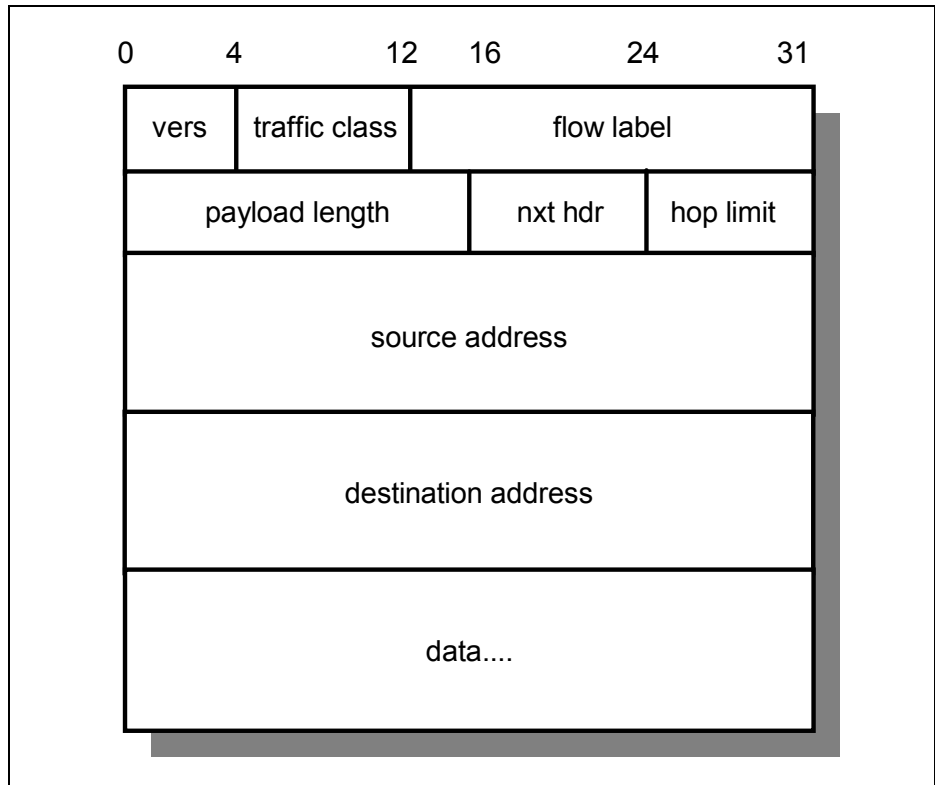


図 219 IPv6 ヘッダー

ここで：

- Vers** 4 ビットのインターネット・プロトコルのバージョン番号：6.
- Traffic class** 8 ビットのトラフィック・クラス値。599 ページの 17.2.4 『トラフィック・クラス』を参照してください。
- Flow label** 20 ビットのフィールド。599 ページの 17.2.5 『フロー・ラベル』を参照してください。
- Payload length** 16 ビットの符号なし整数としてエンコードされたパケットの長さ（このヘッダーは除く）を示すバイト数。長さが 64 KB よりも大きい場合、このフィールドは 0 となってオプション・ヘッダー (Jumbo Payload) に真の値が示されます。

Nextheader

基本 IP ヘッダーの直後に置かれるヘッダーのタイプを示します。これは、IP オプション・ヘッダーまたは上部層プロトコルとなります。使用されるプロトコル番号は、IPv4 で使用されているものと同じです (67 ページの 3.1 『インターネット・プロトコル (IP)』のリストを参照してください)。このフィールドは、拡張ヘッダーの存在を示すためにも使用されます。これにより、オプション情報を IPv6 パケットに付加するためのメカニズムが備わります。IPv4 に関して示された値に加えて、IPv6 パケットでは以下の値も出現します。

- 41 IPv6 ヘッダー
- 45 Interdomain Routing Protocol
- 46 Resource Reservation Protocol
- 58 IPv6 ICMP パケット

以下の値はすべて拡張ヘッダーです。

- 0 ホップごとのオプション・ヘッダー
- 43 IPv6 ルーティング・ヘッダー
- 44 IPv6 フラグメント・ヘッダー
- 50 暗号ペイロード
- 51 IPv6 認証ヘッダー
- 59 No Next Header
- 60 宛先オプション・ヘッダー

さまざまなタイプの拡張ヘッダーについては、585 ページの 17.2.2 『拡張ヘッダー』で説明されています。

Hoplimit

これは IPv4 TTL フィールドですが、秒数ではなくホップ数で指定されるようになりました。変更された理由は、以下の 2 つです。

- IP は通常、データグラムを 1 秒間に 1 ホップ以上の速さで転送し、TTL フィールドは常にホップごとに小さくなっていくので、実用面を考えると秒数ではなくホップ数で測定されることになりました。
- 多くの IP インプリメンテーションでは、経過した時間に基づいて未解決のデータグラムを満了にするということはありません。

ホップ限度が小さくなってゼロになると、パケットは廃棄されます。

- Source address** 128 ビットのアドレス。IPv6 のアドレスについては、592 ページの 17.2.3 『IPv6 アドレッシング』で説明されています。
- Destination address** 128 ビットのアドレス。IPv6 のアドレスについては、592 ページの 17.2.3 『IPv6 アドレッシング』で説明されています。

IPv4 と IPv6 のヘッダー・フォーマットを比較すると、いくつかの IPv4 ヘッダー・フィールドは IPv6 ヘッダーに直接対応するフィールドが存在しないことが分かります。

- Type of Service

IPv6 での Type of Service 問題は、599 ページの 17.2.5 『フロー・ラベル』で説明されている フローの概念を使用して取り組まれています。

- 識別、フラグメント・フラグ、およびフラグメント・オフセット

フラグメント化されたパケットでは、IPv6 ヘッダー内にフラグメント情報を保持する代わりに、拡張ヘッダーを持ちます。これにより、基本 IPv6 ヘッダーのサイズが小さくなります。高水準プロトコル (特に TCP) では、データグラムのフラグメント化を回避する傾向があります (これにより、通常の場合は IPv6 ヘッダーのオーバーヘッドが小さくなります)。以下に示すように、IPv6 がパケットのフラグメント化を行うのはそれがソースにあるときだけで、宛先に移動する間はフラグメント化を行いません。

- ヘッダー・チェックサム

トランスポート・プロトコルがチェックサムをインプリメントするため、および IPv6 が整合性を保証するためにも使用できるオプションの認証ヘッダーを含んでいるために、IPv6 では IP パケットのチェックサム・モニターは行われ *ません*。

TCP も UDP も、使用するチェックサムに疑似 IP ヘッダーを含んでいるので、これらの場合に IPv4 内の IP ヘッダーは 2 回チェックされることになります。

TCP および UDP、そして IPv6 上で実行される他のプロトコルで同じチェックサム・メカニズムを使用するものは、疑似 IPv6 ヘッダーのフォーマットが疑似 IPv4 ヘッダーと明らかに異なるにもかかわらず、疑似 IP ヘッダーの使用を続けます。ICMP、IGMP、および IPv4 で疑似 IP ヘッダーを使用しない他のプロトコルは、チェックサム内で疑似 IPv6 ヘッダーを使用します。

- オプション

IPv6 パケットに関連したすべてのオプション値は拡張ヘッダー内に含まれるので、基本 IP ヘッダーは常に同じサイズとなります。

17.2.1 パケット・サイズ

すべての IPv6 ノードは、パスに沿ったすべてのリンクによってサポートされている最大伝送単位 (MTU) を動的に判別することが (RFC 1191 –「パス MTU の判別」に記述されているとおり) 期待されています。ソース・ノードが送信するパケットは、パス MTU を超過しないものだけです。そのため IPv6 ルーターは複数ホップの経路の途中でパケットをフラグメント化する必要がなくなり、さまざまな物理送信メディアを横断するパスをはるかに効率的に使用することができます。IPv6 では、すべてのリンクが 128 バイト以上の MTU をサポートしている必要があります。

17.2.2 拡張ヘッダー

すべての IPv6 パケットは、基本ヘッダーから始まります。ほとんどの場合、これはパケットを送るために必要な唯一のヘッダーです。しかし場合によっては、パケットと共に追加の情報を宛先または途中の中間システムに伝達する必要があります (これは、従来であれば IPv4 データグラム内の「Options (オプション)」フィールドによって伝達された情報です)。この目的で、拡張ヘッダーが使用されます。

拡張ヘッダーは IPv6 基本パケット・ヘッダーの直後に配置されて、ペイロード長の一部として数えられます。それぞれの拡張ヘッダー (59 は例外) には、ヘッダーの第 1 バイトとして固有の 8 ビットの「Next Header (次のヘッダー)」フィールドがあります。このフィールドは、続くヘッダーのタイプを示しています。この構造により、IPv6 は複数の拡張ヘッダーを 1 つに連結することができます。図 220 は、複数の拡張ヘッダーがあるパケットの例を示しています。

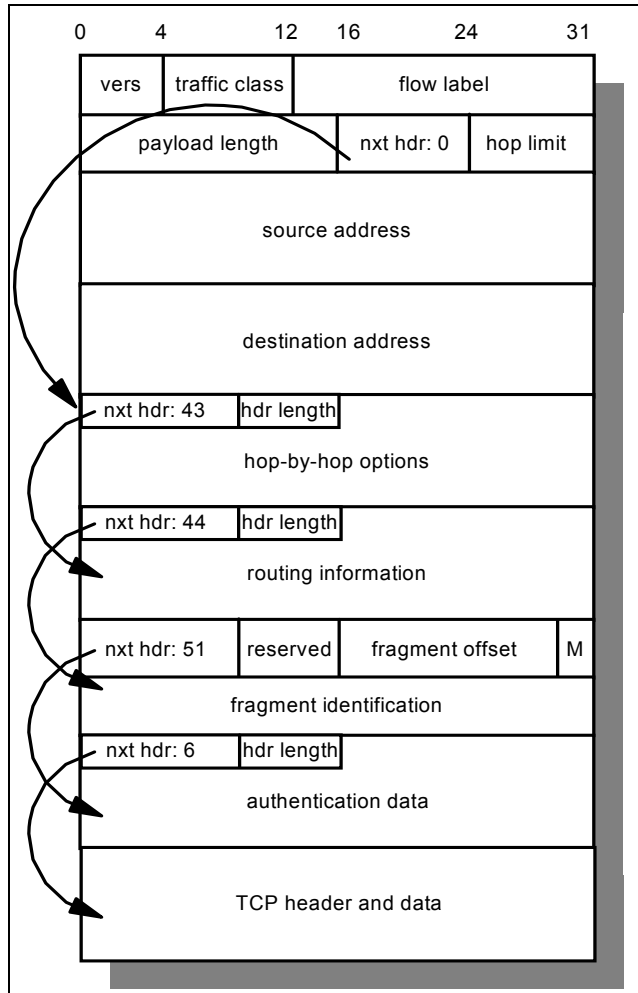


図 220 複数の拡張ヘッダーを含む IPv6 パケット

それぞれのヘッダーの長さはタイプによって異なりますが、常に 8 バイトの倍数です。限られた数の IPv6 拡張ヘッダーがあり、いずれも IPv6 パケット内に 1 回だけ出現します (例外は宛先オプション・ヘッダー - 60 であり、このヘッダーは複数回出現することがあります)。パケットを発信した IPv6 ノードは、パケットを受け取る IPv6 ノードがこれを検証する必要がなくても、拡張ヘッダーを特定の順序 (数値の順序、ただし例外は 60) で配置しなければなりません。この順序は、中間ルーターが効率的に処理を行うために大切なものです。一般的に、ルーターが知る必要があるのはホップごとのオプションとルーティング・ヘッダーだけです。ルーターがここまで読み取った後は、パケット内をさらに読み取る必要はないので、すぐに転送することが

できます。「Next Header (次のヘッダー)」 フィールドに拡張ヘッダーを意味する 1 以外の値が含まれている場合、これは IPv6 ヘッダーが終了して高水準プロトコル・データが開始することを意味しています。

IPv6 では、IPv6 内で IPv6 をカプセル化すること (" トンネリング ") が可能です。これを行うには、「Next Header (次のヘッダー)」 値を 41 (IPv6) に指定します。カプセル化された IPv6 パケットにも、固有の拡張ヘッダーがある場合があります。パケットのサイズはパス MTU と一致するように起点ノードによって算出されるので、IPv6 ルーターは拡張ヘッダーをパケットに追加するのではなく、受け取ったパケットを自らが作成した IPv6 パケット内でカプセル化するようにするべきです (必要であれば、フラグメント化することもできます)。

ホップごとのヘッダー (これが存在する場合には IP ヘッダーの直後に配置します) および場合によっては宛先オプション・ヘッダー (592 ページの 17.2.2.6 『宛先オプション・ヘッダー』を参照してください) が例外となりますが、拡張ヘッダーはパケットのパスにある最後のルーター以外のどのルーターによっても処理されません。

17.2.2.1 ホップ単位ヘッダー

ホップ単位ヘッダーには、パケットが横断するすべてのノードと宛先ノードで検査する必要があるオプションが含まれます。このヘッダーは、IPv6 ヘッダー (ある場合) の直後になければならず、IPv6 基本ヘッダーの「Next Header (次のヘッダー)」 フィールド中の特殊値 0 で識別されます。(この値は、実際のプロトコル番号ではなく、この固有なタイプの拡張ヘッダーを識別するための特殊な用法です。)

ホップ単位ヘッダーには、以下の形式の可変長オプションが含まれます (一般的にタイプ - 長さ - 値 (TLV) 形式といいます)。

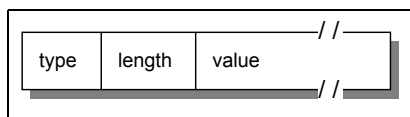


図 221 IPv6 タイプ - 長さ - 値 (TLV) オプション形式

ここで：

Type (タイプ)

オプションのタイプ。オプション・タイプの形式はすべて共通です。

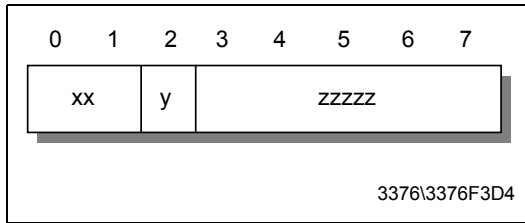


図 222 IPv6 タイプ-長さ-値 (TLV) オプション・タイプ形式

ここで：

xx オプションを認識しない IPv6 ノードがそのオプションを処理する方法を示す、2 ビットの数値。

- 0** オプションをスキップして続行します。
- 1** 通知せずにパケットを廃棄します。
- 2** パケットを廃棄し、ICMP 非認識タイプ・メッセージを送信側に通知します。
- 3** パケットを廃棄し、宛先アドレスがマルチキャスト・アドレスでない場合には ICMP 非認識タイプ・メッセージを送信側に通知します。

y このビットを設定すると、途中でオプションの値を変更できることを示します。このビットを設定すると、「Option Data (オプション・データ)」フィールド全体が、パケットに対して実行される整合性の計算から除外されます。

zzzzz 残りのビットは、以下のオプションを定義します。

- 0** Pad1 (1 バイト埋め込み)
- 1** PadN (N バイト埋め込み)
- 194** Jumbo Payload Length (特大のペイロード長)

Length (長さ)

オプション値フィールドのバイト単位の長さ。

Value (値)

オプションの値。この値はタイプによって異なります。

ホップ単位ヘッダーのオプションのタイプ

個々の拡張ヘッダーの長さは 8 バイトの倍数の整数になっており、これ以降のヘッダー (8 バイト) と桁数がそろえられています。この処理が行われる理由は、単に「整とん」するためではなく、マルチバイト値がメモリー内の本来の境界 (現行のプロセッサの元来のワード・サイズは 32 ビットか 64 ビット) にあるほうが処理が効率的になるからです。

同様の方法で、個々のオプションも位置合わせされ、マルチバイト値が本来の境界上にあります。多くの場合、この処理のためにオプション・ヘッダーは本来必要な長さより長くなりますが、それでもノードでパケットを処理する速度は上がります。この位置合わせを行えるようにするには、ホップ単位ヘッダー中で以下の 2 つの埋め込みオプションを使用します。

Pad1 (1 バイト埋め込み)

X'00' バイトを使用して 1 バイト分埋め込みます。長い埋め込みシーケンスの場合は、PadN オプションを使用します。

PadN (N バイト埋め込み)

TLV 形式 (前述) のオプション。長さのバイトで、必須の最小値の 2 バイトに追加する埋め込みバイト数を指定します。

ホップ単位ヘッダーの 3 番目のオプション・タイプは、*Jumbo Payload Length* (特大のペイロード長) です。このオプションを使用して、65,535 バイト (IPv6 基本ヘッダーの 16 ビットの「Payload Length (ペイロードの長さ)」フィールドで指定できる最大サイズ) より長いペイロード・サイズのパケットを指示します。このオプションを使用する際には、基本ヘッダー中の「Payload Length (ペイロードの長さ)」をゼロに設定しなければなりません。このオプションを使用すると、基本ヘッダー中の合計パケット・サイズは 40 バイト未満になります。詳細については、図 225 を参照してください。

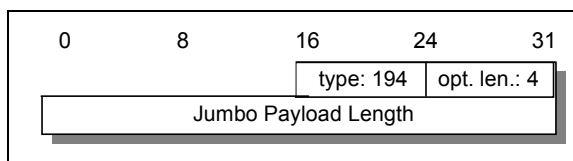


図 223 特大のペイロード長オプション

17.2.2.2 ルーティング・ヘッダー

パケットがネットワーク中を移動する際のパスは、通常はネットワーク自体で決められます。しかしながら、ソース側でパケットが移動する経路を制御する度合いを増やしたい場合が時折あります。たとえば、特定のデータが、速度は遅くなっても通常の経路より安全性の高い経路を移動するようにしたい場合があります。ルーティング・ヘッダー (図 224 を参照) を使用すると、ネットワーク中を移動する際のパスを事前定義できます。ルーティング・ヘッダーは、その前の「Next Header (次のヘッダー)」フィールド中の値を 43 にすると識別されます。このヘッダーの最初のバイトは「Next Header (次のヘッダー)」フィールドで、2 番目のバイトは 1 バイトのルーティング・タイプです。内部定義されているタイプは 0 - Strict/Loose Source Routing (密 / 疎ソース・ルーティング) だけで、これは IPv4 中のソース・ルーティン

グと同様の働きをします (101 ページの 3.1.8.3 『IP データグラム・ルーティング・オプション』を参照)。

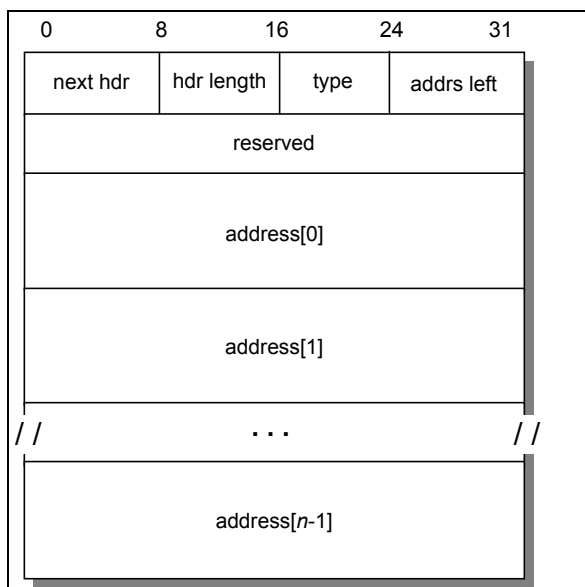


図 224 IPv6 ルーティング・ヘッダー

ここで：

Next hdr (次のヘッダー)

このヘッダーの後のヘッダーのタイプ。

Hdr length (ヘッダー長)

このルーティング・ヘッダーの長さ (先頭 8 バイトを除く)。

Type (タイプ)

ルーティング・ヘッダーのタイプ。現在このフィールドには値 0 だけを指定でき、これは Strict/Loose Source Routing (密 / 疎ソース・ルーティング) を表します。

Segments left (残りのセグメント)

残りの経路セグメントの数。つまり、明示的にリストされている、最終宛先に着信するまでに通過する中間ノードの数。

Address 1..n (アドレス 1..n)

ソース経路を構成する一連の 16 バイトの IPv6 アドレス。

パケットの必須パス上の最初のホップは、そのパケットの基本ヘッダー中の宛先アドレスによって示されます。パケットがこのアドレスに着信すると、

ルーターにより、基本ヘッダー中の宛先アドレスが、ルーター拡張ヘッダー中の次のアドレスに交換されます。さらにルーターにより「Segments left (残りのセグメント)」フィールドの値が 1 ずつ小さくされてから、パケットが転送されます。

17.2.2.3 フラグメント・ヘッダー

585 ページの 17.2.1 『パケット・サイズ』で説明されているように、ソース・ノードではパケットを送信する前にパスの MTU が判別されます。送信対象のパケットが MTU より大きいと、そのパケットは分割され、その個々は 8 バイトの倍数になりフラグメント・ヘッダーが付けられます。フラグメント・ヘッダーは、その前の「Next Header (次のヘッダー)」フィールド中の値を 44 にすると識別され、以下のような形式になります。

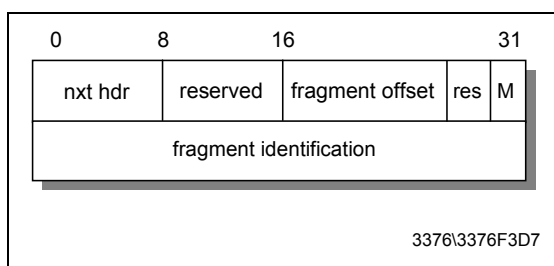


図 225 IPv6 フラグメント・ヘッダー

ここで：

Nxt hdr (次のヘッダー) このヘッダーの後に続くヘッダーのタイプ。

Reserved (予約済み) 8 ビットの予約フィールド。伝送の際にゼロに初期設定され、受信時には無視されます。

Fragment offset (フラグメント・オフセット)

13 ビットの符号なし整数。フラグメント化される前のオリジナル・データの先頭と比較した、後続のデータのオフセット (8 バイト単位) を指定します。

Res (予約済み) 2 ビットの予約フィールド。伝送の際にゼロに初期設定され、受信時には無視されます。

M (続き) 続きがあることを示すフラグ。このフィールドを設定すると、このフラグメントが最後のフラグメントではないことを示します。

Fragment identification (フラグメント識別)

これは、同一のデータグラムのフラグメントの識別に使用する確定 ID です。このフィールドは、

IPv4 の「Identifier (ID)」フィールドに非常に似ていますが、長さが 2 倍あります。

17.2.2.4 認証ヘッダー

認証ヘッダーは、受信したパケットが転送中に変更されなかったことや、本当に請求先の送信側から送信されたものであることを確認するのに使用します。認証ヘッダーは、その前の「Next Header (次のヘッダー)」フィールド中の値を 51 にすると識別されます。認証ヘッダーの形式と、認証の詳細については、729 ページの 21.5.2 『認証ヘッダー (AH)』を参照してください。

17.2.2.5 暗号ペイロード

暗号ペイロード (ESP) は特殊なヘッダーで、パケット中の基本ヘッダーと上位層プロトコルの間のどこにあってもかまいません。ESP ヘッダーの後のデータはすべて暗号化されます。詳細については、735 ページの 21.5.3 『暗号ペイロード (ESP)』を参照してください。

17.2.2.6 宛先オプション・ヘッダー

このヘッダーは、ホップ単位ヘッダーと同じ形式ですが、宛先ノードだけで検査されます。通常、宛先オプションの対象は最終宛先だけで、宛先オプション・ヘッダーは上位層ヘッダーの直前にあります。しかしながら、宛先オプションの対象を中間ノードにすることもできます。この場合、宛先オプションの前にルーティング・ヘッダーがなければなりません。したがって、1 つのパケットに 2 つの宛先オプション・ヘッダーが組み込まれていることもあります。現在このヘッダーのオプション・タイプとして、Pad1 と PadN だけが指定されています (587 ページの 17.2.2.1 『ホップ単位ヘッダー』を参照)。このヘッダーの前にある「Next Header (次のヘッダー)」フィールドの値は 60 です。

17.2.3 IPv6 アドレッシング

IPv6 アドレス・モデルは、RFC 2373 – IP Version 6 Addressing Architecture で指定されています。IPv4 では 32 ビットのアドレスが使用されていますが、IPv6 ではその代わりに 128 ビットのアドレスが使用されています。理論的には、340,282,366,920,938,463,463,374,607,431,768,211,456 個のアドレスを使用できます。現在の IPv4 アドレス・スペースと同じ効率で使用したとしても、地球上の 1 平方メートル当たり 50,000 個のアドレスを使用できます。

IPv6 アドレスは、8 つの 16 進数をコロンで区切った形式で表します。以下に例を示します。

```
FE80:0000:0000:0000:0001:0800:23e7  
:f5db
```


アドレスの表記を短くするには、いずれかのグループの先行ゼロを省略できます。以下に例を示します。

FE80:0:0:0:1:800:23e7:f5db

さらに、すべてゼロのグループが1つあったり、連続したりしている場合は、2つのコロンに置き換えることができます。以下に例を示します。

FE80::1:800:23e7:f5db

注：

二重コロンのショートカットは、IPv6 アドレスの表記の中で1回しか使用できません。連続しないすべてゼロのグループが他にも存在する場合は、1つだけを二重コロンで置き換えることができます。その他は0として表記されなければなりません。

電話番号の国別コードや市外局番の場合と同様に、IPv6 アドレス・スペースは形式設定接頭部を使用して編成されます。アドレス・スペースは形式設定接頭部によりツリー形式で論理的に分類されているので、ネットワーク間の経路を簡単に見つけることができます。現時点で以下の接頭部が割り当て済みです。

表 24 :IPv6 - 形式設定接頭部の割り振り

割り振り	接頭部 (2 進数)	アドレス範囲 の先頭 (16 進数)	マスク長 (ビット)	アドレス・スペース中で占める部分
予約済み	0000 0000	0::/8	8	1/256
NSAP 用に予約済み	0000 001	200::/7	7	1/128
IPX 用に予約済み	0000 010	400::/7	7	1/128
集合可能グローバル・ユニキャスト・アドレス	001	2000::/3	3	1/8
リンク・ローカル・ユニキャスト	1111 1110 10	FE80::/10	10	1/1024
サイト・ローカル・ユニキャスト	1111 1110 11	FEC0::/10	10	1/1024

割り振り	接頭部 (2 進数)	アドレス範囲 の先頭 (16 進数)	マスク長 (ビット)	アドレス・ス ペース中で占 める部分
マルチキャスト	1111 1111	FF00::/8	8	1/256
割り振りの合 計				15%

IPv6 では以下のタイプのアドレスが定義されます。

- ユニキャスト・アドレス

ユニキャスト・アドレスとは、1 つのインターフェースに割り当てられる ID のことです。このアドレスに送信されたパケットは、割り当て先のインターフェースだけに送達されます。以下のような、特別な目的を持つユニキャスト・アドレスが定義されています。

- ループバック・アドレス (::1): このアドレスは、ホストからそのホスト自体にパケットを送信する際に使用する仮想インターフェースに割り当てられます。このアドレスは、IPv4 のループバック・アドレス 127.0.0.1 と同等です。
- 未指定アドレス (::): このアドレスは、自動構成の実行中にホストでソース・アドレスとして使用されます。このアドレスは、IPv4 の未指定アドレス 0.0.0.0 と同等です。
- IPv4 互換可能アドレス (::<IPv4_address>): この種類のアドレスは、IPv6 トラフィックがトンネルを通して既存の IPv4 ネットワークを移動する必要がある場合に使用します。この種のトンネルの端点は、ホスト (自動トンネリング) かルーター (構成トンネリング) にすることができます。IPv4 互換可能アドレスは、96 ビットのゼロの後に有効な 32 ビットの IPv4 アドレスが続くという形式になります。たとえば、アドレス 1.2.3.4 (16 進数 01.02.03.04) は ::0102:0304 になります。
- IPv4 マップ・アドレス (::FFFF<IPv4_address>): この種類のアドレスは、IPv6 ホストと IPv4 ホストとの間で通信する必要がある場合に使用します。この場合、ヘッダー変換用に二重スタックのホストまたはルーターが必須です。たとえば、IPv6 ノードから IPv4 アドレス 1.2.3.4 のホストにデータを送信したい場合は、宛先アドレス ::FFFF:0102:0304 を使用します。
- リンク・ローカル・アドレス: この種類のアドレスは、ホストのインターフェースの接続先の物理ネットワーク専用です。
- サイト・ローカル・アドレス: この種類のアドレスは、インターネット中でルーティングできません。このアドレスは、IPv4 ネットワークの

専用アドレス (10.0.0.0, 176.16.0.0-176.31.0.0, 192.168.0.0-192.168.255.0) と同等です。

・ グローバル・ユニキャスト・アドレスの形式

グローバル・ユニキャスト・アドレスの形式は、RFC 2374 – An IPv6 Aggregatable Global Unicast Address *Format* で指定されているように、インターネットに接続している IPv6 ノードで主に使用されている形式に従うと想定されています。集合可能アドレスは、3つのレベルのインターネット階層に対応する3つのセクションに分割でき、それぞれの名前は以下のとおりです。

- 公用トポロジー：公用インターネット伝送サービスを提供するプロバイダーおよび交換局。
- サイト・トポロジー：サイトの外部のノードに公用伝送サービスを提供しない組織に限られます。
- インターフェース ID: リンク上のインターフェースを識別します。

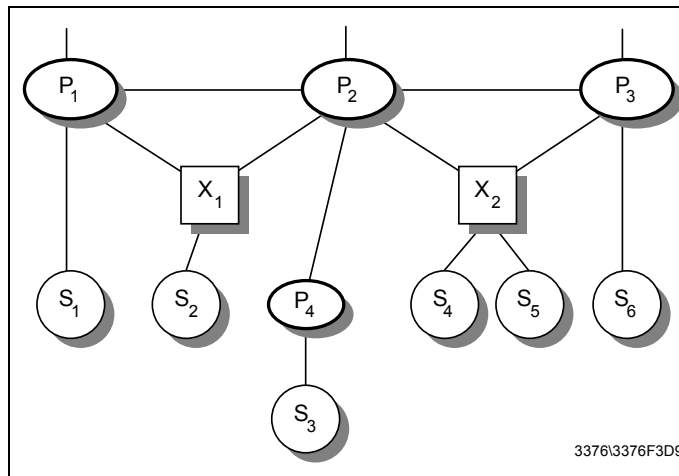


図 226 グローバル・ユニキャスト・アドレス形式 - 3つのレベルの階層

グローバル・ユニキャスト・アドレス形式は、図 226 に示されている基礎構造を想定して設計されています。P1、P2、および P3 は、バックボーン・プロバイダーです。P4 は中小プロバイダーで、P2 からサービスを受けています。交換局 (X1 および X2) は、電話網の交換局に似ていて、アドレスを割り振ります。加入者 (S1 ~ S6) は、プロバイダーに直接接続するか、交換局に接続するか (この場合はバックボーン・サービスのプロバイダーにも加入しなければならない) 選択できます。交換局を介して接続する企業には、IP アドレスを変更せずにバックボーン・プロバイダーを変更できるという柔軟性があります。

集合可能グローバル・ユニキャスト・アドレスの形式が、図 227 に示されています。

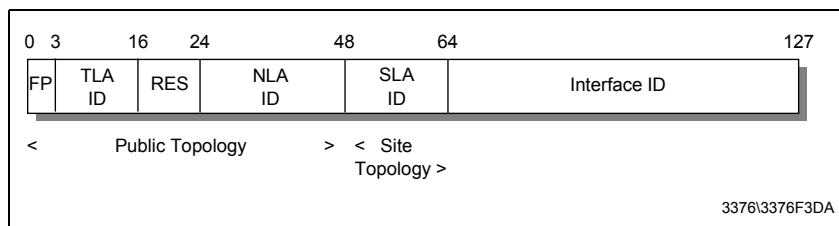


図 227 グローバル・ユニキャスト・アドレスの形式

ここで：

- FP** 形式設定接頭部 (001)。
- TLA ID** 最上位の集合 ID。これは最上位のルーティング階層です。インターネットの最上位のルーターには、アクティブな TLA ID ごとにルーティング・テーブル項目が必要です。項目数の最大値は 8,192 で、これは現在の IPv4 の最上位のルーター中の約 50,000 個の項目と比較されます。
- RES** 将来の利用のために予約済み。将来 TLA ID か NLA ID の数を増やす (必要が生じた場合) 際に使用できます。
- NLA ID** 次のレベルの集合 ID。TLA ID を割り当てられた組織 (プロバイダー) が、独自のアドレッシング階層を作成し、サイトを識別するのに使用します。個々の組織は、24 ビットの NLA ID スペースを使用して、現在 IPv4 によってサポートされているネットワークの合計数と同じ数のサイトにサービスを提供できます。
- SLA ID** サイト・レベルの集合 ID。このフィールドは、個々の企業が独自のローカル・アドレッシング階層を作成するのに使用します。この 16 ビットのフィールドを使用すると、最大 65,535 の個別サブネットを作成できます。

• マルチキャスト・アドレス

マルチキャスト・アドレスは、複数のホスト上のインターフェースのセットに割り当てられる ID です。このアドレスに送信されるパケットは、そのアドレスに対応するすべてのインターフェースに送達されます。(IP マルチキャストの詳細については、241 ページの 6.2 『Internet Group Management Protocol (IGMP)』を参照してください。)IPv6 にはブロードキャスト・アドレスはなく、その機能はマルチキャスト・アドレスによって置き換えられています。図 228 は、IPv6 マルチキャスト・アドレスのフォーマットを示しています。

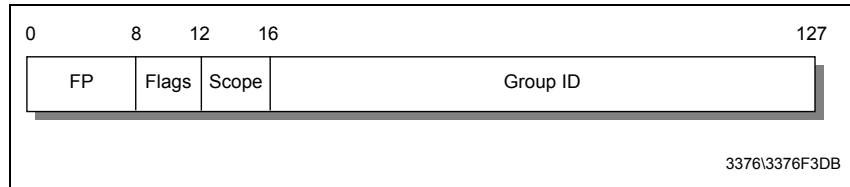


図 228 IPv6 マルチキャスト・アドレス・フォーマット

ここで：

FP フォーマット接頭部 - 1111 1111。

Flags 4 つのフラグ・ビットのセット。現在のところ低位のビットだけが意味があり、その内容は以下のとおりです。

0000 番号付け権限によって割り当てられた永住所。

0001 一時アドレス。この種のアドレスは必要に応じてアプリケーションによって設定できます。アプリケーションの終了時に、アドレスはそのアプリケーションによって解放され、再使用することができます。

Scope マルチキャストの有効範囲を示す 4 ビット値。指定可能な値は以下のとおりです。

0 予約済み

1 ローカル・ノード上のインターフェースに限定 (ノード・ローカル)

2 ローカル・リンク上のノードに限定 (リンク・ローカル)

5 ローカル・サイトに限定

8 組織に限定

E グローバル有効範囲

F 予約済み

グループ ID マルチキャスト・グループを示します。

たとえば、NTP サーバー・グループが、16 進数 101 のグループ ID で永続マルチキャスト・アドレスに割り当てられている場合は、以下のようになります。

•FF02::101 は、すべての NTP サーバーが送信側として同じリンク上にあることを意味します。

•FF05::101 は、すべての NTP サーバーが送信側として同じサイト上にあることを意味します。

特別な目的のマルチキャスト・アドレスは、以下のように事前定義されています。

- FF01::1** すべてのインターフェースはノード・ローカルです。ホストそのものの上のすべてのインターフェースを定義します。
- FF02::1** すべてのノードはノード・ローカルです。ローカル・ネットワーク上ですべてのシステムを定義します。
- FF01::2** すべてのルーターはノード・ローカルです。ホストそのものにローカルなすべてのルーターを定義します。
- FF02::2** すべてのルーターはリンク・ローカルです。同じリンク上のすべてのルーターをホストとして定義します。
- FF05::2** すべてのルーターはサイト・ローカルです。同じサイト上のすべてのルーターをホストとして定義します。
- FF02::B** モバイル・エージェントはリンク・ローカルです。
- FF02::1:2** すべての DHCP エージェントはリンク・ローカルです。
- FF05::1:3** すべての DHCP サーバーはサイト・ローカルです。

予約済みのマルチキャスト検出・アドレスのさらに詳細なリストは、RFC 2375 – IPv6 Multicast Address Assignments にあります。

この RFC も、*送信請求ノード・アドレス* と呼ばれる特別なマルチキャスト・アドレスを定義します。このフォーマットは FF02::1:FFxx:xxxx で、xx xxxx はノード・ユニキャスト・アドレスの最後の 24 ビットから取られます。たとえば、IPv6 アドレスが 4025::01:800:100F:7B5B のノードは、マルチキャスト・グループ FF02::1:FF 0F:7B5B に属しています。送信請求ノード・アドレスは、近隣ディスカバリー用の ICMP によって、重複アドレスを検出するために使用されます。詳細については、599 ページの 17.3 『Internet Control Message Protocol Version 6 (ICMPv6)』を参照してください。

• 任意キャスト・アドレス

任意キャスト・アドレスは、複数のホスト上のインターフェースに割り当てられる、特殊なタイプのユニキャスト・アドレスです。そのようなアドレスに送信されるパケットは、そのアドレスを持つ最も近いインターフェースに送達されます。ルーターは、距離の定義に基づいて最も近いインターフェースを判別します。たとえば、RIP の場合はホップ、OSPF の場合はリンク状態などです。

任意キャスト・アドレスは、ユニキャスト・アドレスと同じフォーマットを使用し、それらとは区別がつきません。ただし、任意キャスト・アドレスに割り当てられているノードは、この事実が分かるように構成する必要があります。

RFC 2373 は、現在のところ任意キャスト・アドレスに対して以下の制限を課します。

- 任意キャスト・アドレスは、パケットのソース・アドレスとして使用すべきではありません。
- 任意キャスト・アドレスはルーターにのみ割り当てることができます。

特殊任意キャスト・アドレス *subnet-router address* は事前定義されています。このアドレスは、特定のサブネットについてのサブネット接頭部と、それに続く後続ゼロで構成されます。このアドレスは、ノードが特定のサブネット上のルーターと接続する必要がある場合、およびそれがどのルーターと接続しても構わない場合（たとえば、モバイル・ノードが、その「ホーム」サブネット上のモバイル・エージェントの 1 つと通信する必要がある場合）に使用できます。

17.2.4 トラフィック・クラス

8 ビットのトラフィック・クラス・フィールドによって、アプリケーションは、生成するトラフィックの特定の優先順位を指定して、サービス・クラスの概念を導入することができます。これによって、Differentiated Services の場合のように、パケットの優先順位を決めることができます。優先順位トラフィックの IPv4 ネットワーク内での処理方法と比較するには、811 ページの 22.1 『QoS が必要な理由』を参照してください。

17.2.5 フロー・ラベル

IPv6 はフローの概念を導入していますが、これはソースから宛先への一連の関連パケットであり、中継のルーターによる、たとえばリアルタイム・サービスなどの、特殊なタイプの処理を必要とします。この処理の性質は、データグラムに付加されるオプション（つまり、IPv6 ホップ単位オプション・ヘッダー）、または独立プロトコル (Resource Reservation Protocol など；820 ページの 22.2.2 『Resource Reservation Protocol (RSVP)』を参照) のいずれかによって伝達できます。特定のフロー・ラベルの処理要件は、*状態情報* として知られています。これはルーターにおいてキャッシュに入れられません。既知のフロー・ラベルがあるパケットがルーターに着信すると、ルーターは、各パケットの残りのヘッダーを調べる必要なしに、効果的に経路を決定し、パケットを転送することができます。

ソースと宛先との間には、どのフローとも関連付けられていないトラフィックに加え、複数のアクティブ・フローが存在する場合があります。それぞれのフローは、IPv6 パケット内の 24 ビット・フロー・ラベル・フィールドで明確にラベルが付けられています。フロー・ラベルの使用の詳細については、RFC 2460 および RFC 1809 を参照してください。

17.3 Internet Control Message Protocol Version 6 (ICMPv6)

IP プロトコルは、あるノードから別のノードへのデータの移動に関係しています。ただし、IP がこのタスクを正常に実行するには、実行する必要がある他の数多くの機能があります。これにはいくつか挙げるだけでも、エラー報告、経路ディスカバリー、および診断があります。これらのすべてのタスクは、Internet Control Message Protocol によって実行されます (105 ページの

3.2 『Internet Control Message Protocol (ICMP)』を参照)。加えて、ICMPv6 はマルチキャスト・グループ・メンバーシップ情報の伝達タスクも実行します。これは、以前は IPv4 の IGMP プロトコルによって実行されていた機能と (241 ページの 6.2 『Internet Group Management Protocol (IGMP)』を参照)、以前は ARP によって実行されていたアドレス解決です (118 ページの 3.4 『Address Resolution Protocol (ARP)』を参照)。

ICMPv6 メッセージおよびその使用法は、RFC 2463 – Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification および RFC 2461 – Neighbor Discovery for IP Version 6 (IPv6) で指定されています。どちらの RFC も、随意選択の状況にあるドラフト規格です。

どの ICMPv6 メッセージにもその先頭に IPv6 ヘッダー (および場合によってはいくつかの IP 拡張ヘッダー) が付きます。ICMPv6 ヘッダーは、直前のヘッダーの「Next Header (次のヘッダー)」値 58 で識別されます。

すべての ICMPv6 メッセージには、図 229 に示されているのと同様のフォーマットがあります。

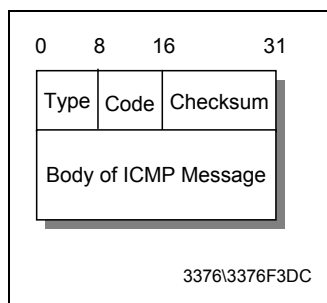


図 229 ICMPv6 汎用メッセージ・フォーマット

ここで：

Type

2 つのクラスの ICMPv6 メッセージがあります。エラー・メッセージには、タイプ 0 ~ 127 があります。通知メッセージには、タイプ 128 ~ 255 があります。

- 1 宛先に到達できない
- 2 パケットが大きすぎる
- 3 時間 (ホップ・カウント) が超過した
- 4 パラメーター問題
- 128 エコー要求
- 129 エコー応答
- 130 グループ・メンバーシップ照会

- 131 グループ・メンバーシップ・レポート
- 132 グループ・メンバーシップ縮小
- 133 ルーター送信請求
- 134 ルーター公示
- 135 近隣送信請求
- 136 近隣公示
- 137 リダイレクト・メッセージ

Code メッセージ・タイプに応じて異なります。

Checksum ICMPv6 メッセージと IPv6 ヘッダーの部分のデータ破壊を検出するために使用します。

メッセージの本体 メッセージ・タイプに応じて異なります。

すべてのタイプに対する ICMPv6 メッセージの完全な詳細については、RFC 2463 および RFC 2461 を参照してください。

17.3.1 近隣ディスカバリー

近隣ディスカバリーは ICMPv6 機能であり、これによってノードはそのリンク上の他のホストおよびルーターを識別できます。ノードは少なくとも 1 つのルーターを認識している必要があります。これによってノードは、ターゲット・ノードがそのローカル・リンク上にない場合に、パケットの転送先を知ります。近隣ディスカバリーによって、ルーターは、ノードが当初誤った選択をしていた場合に、さらに適切なルーターを使用するようにノードを指定変更することもできます。

17.3.1.1 アドレス解決

図 230 は、4 つの IPv6 ワークステーションがある、単純なイーサネット LAN の一部分を示しています。

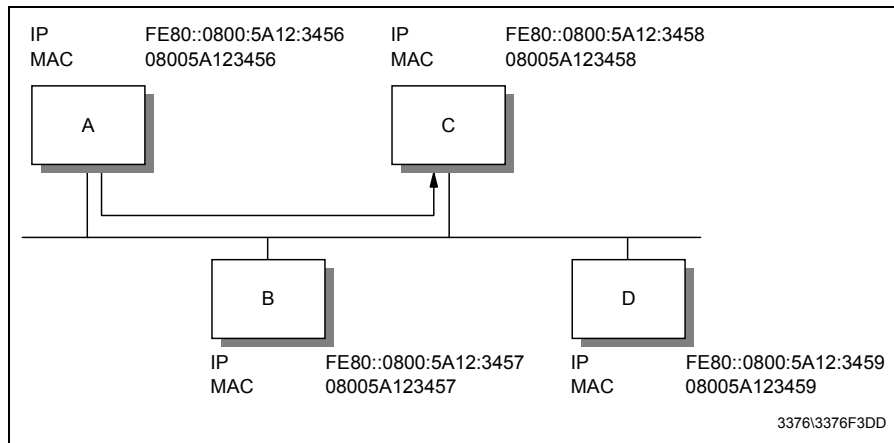


図 230 IPv6 アドレス解決

ワークステーション A は、データをワークステーション B に送信する必要があります。これはワークステーション B の IPv6 アドレスを知っていますが、その MAC アドレスを知らないため、パケットを送信する方法が分かりません。この情報を見つけるために、ワークステーション A は近隣送信請求メッセージを、図 231 で示すフォーマットで送信します。

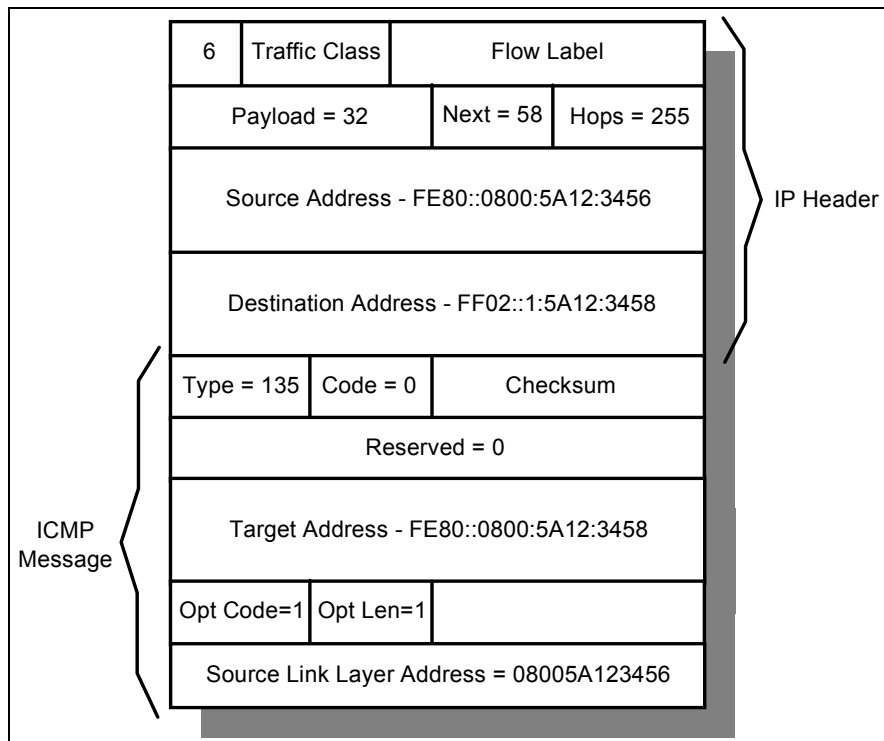


図 231 近隣送信請求メッセージのフォーマット

このパケットの IP ヘッダーにある以下の重要なフィールドに注目してください。

Next58 (続く ICMP メッセージ・ヘッダー用)。

Hops ホップが 255 に設定されていないすべての送信請求パケットは廃棄されます。これによって送信請求がルーターを通過していないことが保証されます。

Destination address このアドレスは、ターゲット・ワークステーションの送信請求ノード・アドレスです (マルチキャストの特殊タイプ。598 ページを参照)。すべてのワークステーションはその固有の送信請求ノード・アドレスに応答しなければなりません。他のワークステーションはそれを無視します。これは IPv4 の ARP に対する改善点であり、リンク上のすべてのノードによって処理される必要があるブロードキャスト・フレームを使用します。

ICMP メッセージそのものについては、以下の点に注目してください。

- Type

135 (近隣送信請求)。
- Target address

これはターゲット・ワークステーションの既知の IP アドレスです。
- Source link layer address

これはターゲット・ワークステーションにとって役立ち、パケットをソース・ワークステーションに送信して戻す場合に、その固有の近隣ディスカバリー・プロセスを開始する必要がなくなります。

近隣送信請求メッセージへの応答は近隣公示であり、これには以下のフォーマットがあります。

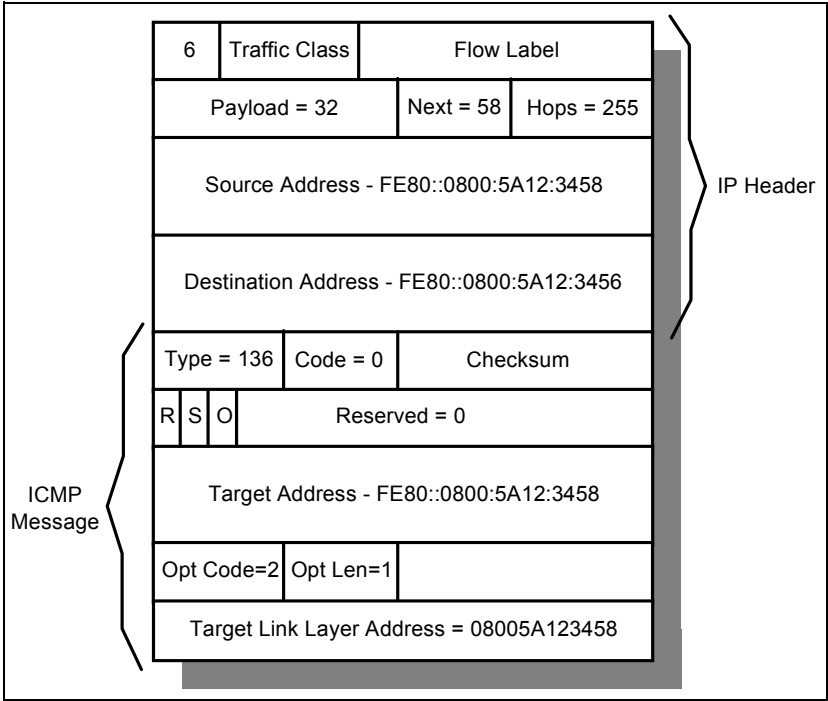


図 232 近隣公示メッセージ

近隣公示は、直接ワークステーション A に送信して戻されます。ICMP メッセージ・オプションには、ターゲット IP アドレスと、そのターゲットのリンク層 (MAC) アドレスが含まれています。公示メッセージの以下のフラグにも注目してください。

- R** ルーター・フラグ。公示の送信側がルーターである場合は、このビットはオンに設定されます。
- S** 送信請求フラグ。公示が送信請求の応答である場合は、このビットはオンに設定されます。
- O** オーバーライド・フラグ。このビットがオンに設定されている場合、受信ノードは、その近隣キャッシュにある、既存のキャッシュに入れられたリンク層エントリーを更新しなければなりません。

ワークステーション A は、このパケットを受け取ると、この情報を近隣キャッシュのメモリーにコミットして入れ、それから当初ワークステーション C に送信しようとしていたデータ・パケットを転送します。

近隣公示メッセージは、ノードが送信して、リンク層アドレスが変更されたことに気付いた場合に、近隣キャッシュへの更新を強制することもできます。

17.3.1.2 ルーターおよび接頭部ディスカバリー

602 ページの図 230 は、非常に単純なネットワークの例を示しています。さらに大きなネットワークや、特にインターネットに接続されているものでは、近隣ディスカバリー・プロセスが、同じリンク上にあるノードを検出するためにまったく同じ方法で使われます。ただし、かなり高い確率で、ノードは単に同じリンク上で他のノードと通信するだけでなく、世界のどこかにある他のネットワーク・セグメント上のノードとも通信する必要があります。この場合、ノードが知っている必要のある、以下の 2 つの重要な情報があります。

1. 世界の他の部分に到達するためにノードが利用できる、ルーターのアドレス
2. ルーターを経由せずに到達できるノードとして、同じリンク上の IP アドレスの範囲を定義する接頭部 (複数の場合あり)。

ルーターは ICMP を使用し、この情報をルーター公示によってホストに伝えます。ルーター公示メッセージのフォーマットは、図 233 に示されています。一般にメッセージには 1 つ以上の付加されたオプションがあります。指定可能な 3 つのオプションすべてがこの例に示されています。

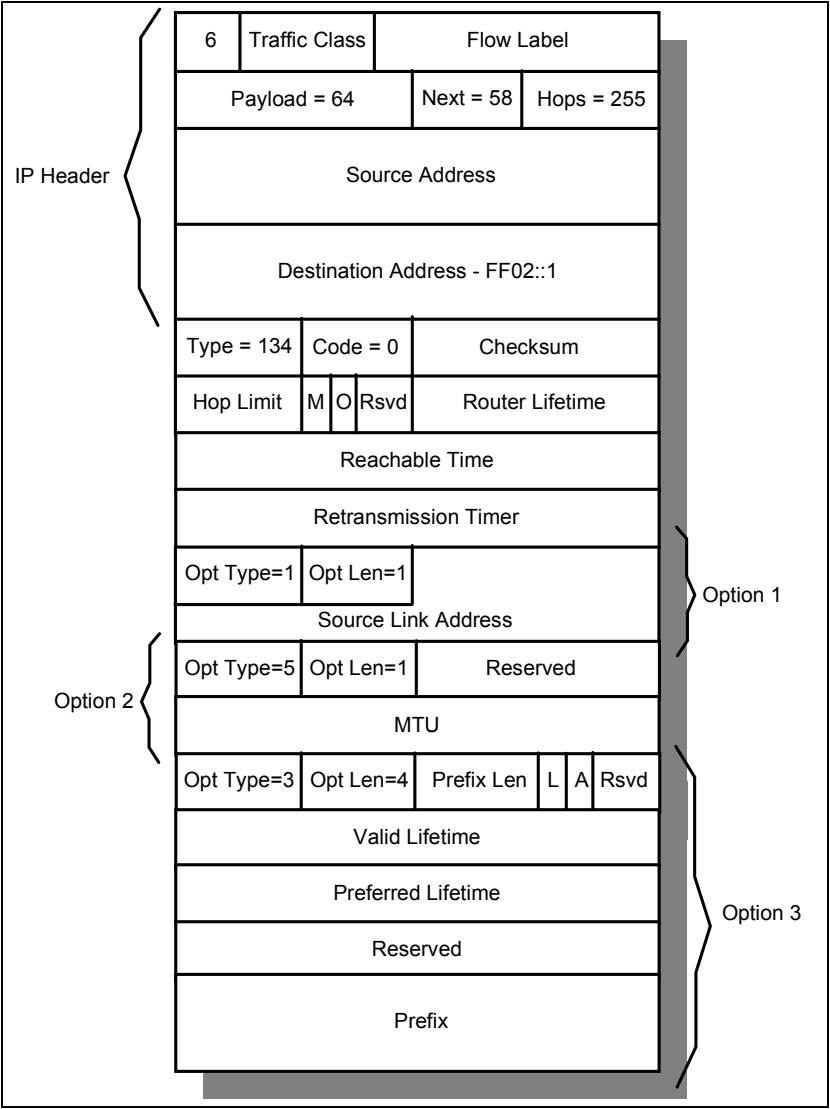


図 233 ルーター公示メッセージのフォーマット

このパケットの IP ヘッダーにある以下の重要なフィールドに注目してください。

Next 58 (続く ICMP メッセージ・ヘッダー用)。

Hops ホップが 255 に設定されていないすべての公示パケットは廃棄されます。これによってパケットがルーターを通過していないことが保証されます。

Destination address このアドレスは、ローカル・リンク上のすべてのシステムを定義する、特殊なマルチキャスト・アドレスです。

ICMP メッセージそのものについては、以下の点に注目してください。

Type	134 (ルーター公示)。
Hop limit	ノードがその発信 IP パケットの Hop Count フィールドに指定する必要があるデフォルト値。
M	1 ビットの管理アドレス構成フラグ (613 ページの 17.3.2 『ステートレス・アドレス自動構成』を参照)。
O	1 ビットのその他のステートフル構成フラグ (613 ページの 17.3.2 『ステートレス・アドレス自動構成』を参照)。
Router lifetime	ノードがこのルーターが使用可能であると見なせる期間。この期間が経過し、ノードが別のルーター公示メッセージを受け取っていないと、ノードはこのルーターが使用不可と見なします。
Reachable time	これは、ローカル・リンク上のすべてのノードのパラメーターを設定します。これはミリ秒単位の時間で、近隣送信請求への応答を受け取った後に、ノードが近隣を引き続き到達可能であると想定する時間です。
Retransmission timer	初期応答が受信されていない場合に、ノードが近隣送信請求メッセージの再送から次の再送まで待機する時間を、ミリ秒単位で設定します。

ルーター公示メッセージの 3 つの指定可能なオプションは以下のとおりです。

Option 1 (ソース・リンク・アドレス)

受信ノードに、近隣送信請求を実行する必要なしに、ルーターに直接応答することを許可します。

Option5 (MTU)

リンクの最大伝送ユニット・サイズを指定します。イーサネットなどの一部のメディアの場合、この値は固定されているので、このオプションは不要です。

Option 3 (接頭部)

リンクのアドレス接頭部を定義します。ノードはこの情報を使用して、ルーターの使用が必要かそうでないかを判別します。この目的

で使用される接頭部オプションでは、L (リンク) ビットがオンに設定されています。接頭部オプションは、アドレス構成の一部としても使用され、その場合は A ビットがオンに設定されています。詳細については、613 ページの 17.3.2 『ステートレス・アドレス自動構成』を参照してください。

ルーターは、ルーター構成で定義された頻度で、非送信請求公示を定期的には送信します。ただし、ノードでは、最も近いルーターについての情報を、スケジュールされた次の公示を待つ必要なしに入手したい場合があります (たとえば、ネットワークに付加されたばかりの新規ワークステーションなど)。この場合、ノードはルーター送信請求メッセージを送信することができます。ルーター送信請求メッセージのフォーマットは、図 234 に示されています。

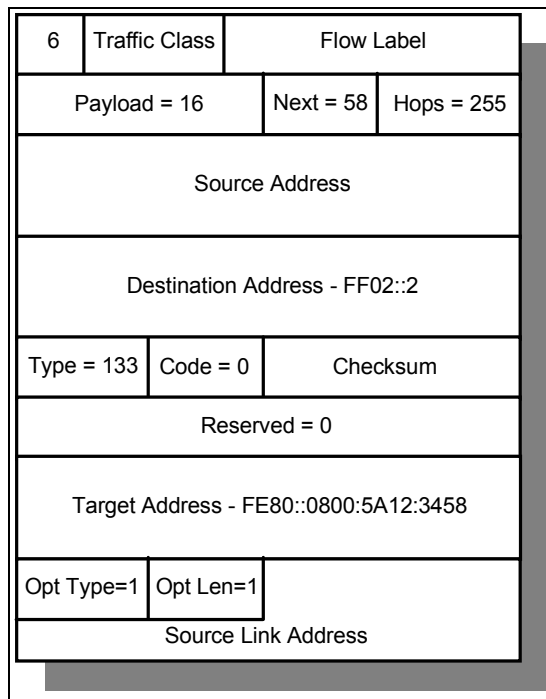


図 234 ルーター送信請求メッセージのフォーマット

このパケットの IP ヘッダーにある以下の重要なフィールドに注目してください。

Next 58 (続く ICMP メッセージ・ヘッダー用)。

Hops ホップが 255 に設定されていないすべての公示パケットは廃棄されます。これによってパケットがルーターを通過していないことが保証されます。

Destination address このアドレスは、ローカル・リンク上のすべてのルーターを定義する、特殊なマルチキャスト・アドレスです。

ICMP メッセージそのものについては、以下の点に注目してください。

Type 133 (ルーター送信請求)

Option 1 (ソース・リンク・アドレス)

受信ルーターに、近隣送信請求を実行する必要なしに、ノードに直接応答することを許可します。

送信請求メッセージを受け取る各ルーターは、(リンク・ローカルなマルチキャスト・アドレスのすべてのシステムではなく) その送信請求を送信したノードに 直接 に送信されるルーター公示で応答します。

17.3.1.3 リダイレクト

ルーター公示メカニズムによって、ノードは必ず 1 つ以上のルーターを認識することが保証され、それによってそのローカル・リンクの外部の装置に接続できます。ただし、ノードが複数のルーターを認識している状況では、データの送信時に選択されたデフォルトのルーターが、どのパケットに対しても選択された最適のルーターとは必ずしもならない可能性があります。この場合、ICMPv6 によって、特定の宛先の最も効果的なパスへの *リダイレクト* ができます。

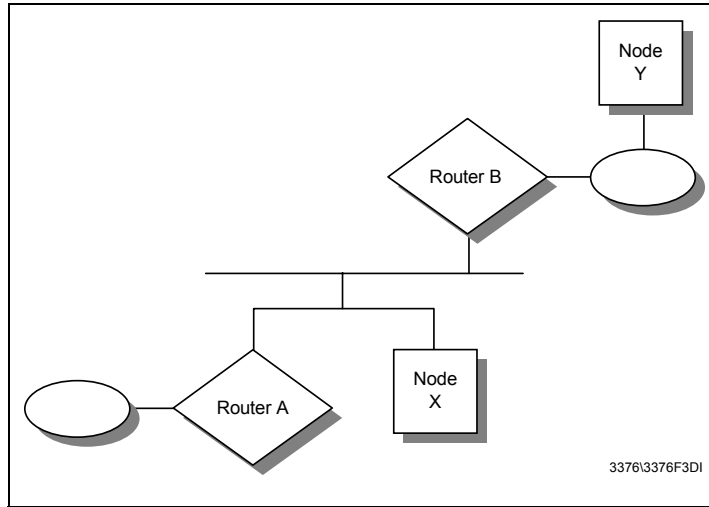


図 235 リダイレクト

図 235 で示されている簡単な例を考慮してください。ノード X はルーター A と B を認識しており、その両方からルーター公示メッセージを受け取っています。ノード X は、データをノード Y に送信しようとしています。ノード Y の IP アドレスとローカル・リンク接頭部を比較することによって、ノード X はノード Y がローカル・リンク上にはなく、ルーターを使用しているはずだということを知ります。ノード X は、デフォルトのルーターのリストからルーター A を選択し、パケットを転送します。明らかに、これはノード Y への最も効率的なパスではありません。ルーター A がパケットをノード Y に (ルーター B を経由して) 転送すると、ルーター A はメッセージをノード X にリダイレクトします。リダイレクト・メッセージのフォーマット (IP ヘッダーで終わる) は、図 236 に示されています。

6	Traffic Class	Flow Label	
Payload Length		Next = 58	Hops = 255
Source Address (Router A)			
Destination Address (Node X)			
Type = 137	Code = 0	Checksum	
Reserved = 0			
Target Address (Router B)			
Destination Address (Node Y)			
Opt Type=1	Opt Len=1		
Source Link Address (Router B)			
Opt Type=4	Opt Length	Reserved = 0	
Reserved = 0			
IP Header and Data			

図 236 リダイレクト・メッセージのフォーマット

このメッセージで注目すべきフィールドは以下のとおりです。

Type	137 (リダイレクト)。
Target address	これはノード Y への到達を試行する場合に使用する必要があるルーターのアドレスです。
Destination address	ノード Y の IP アドレス。
Option 2 (ターゲット・リンク層アドレス)	ルーター B のリンク・アドレスを指定し、ノード X が近隣送信請求なしにそれに到達できるようにします。
Option 4 (リダイレクト済みヘッダー)	ノード X によって送信された元のパケット、完全な IP ヘッダー、およびそれに収まる同量のデータを組み込み、リダイレクト・メッセージの合計サイズが 576 バイトを超えないようにします。

17.3.1.4 近隣未到達検出

ICMPv6 の近隣ディスカバリー機能の付加的な担当範囲は、*近隣未到達検出* (NUD) です。

ノード・アクティビティは、パケットの送信先の近隣の到達可能性状況を追跡します。これは次の 2 とおりの方法で実行できます。上層プロトコルをモニターして、接続が転送進行中 (たとえば TCP 承認を受信中) であるかを確認するか、または特定の近隣送信請求を発行して、ターゲット・ホストへのパスが引き続き使用可能かどうかを検査するかのいずれかの方法です。近隣へのパスに障害が発生していることが分かった場合、適切な処置が取られてリンクのリカバリーが試行されます。これには、アドレス解決処理の再始動または近隣キャッシュ・エントリーの削除が含まれることがあります。そうすると、ターゲットへの作業パスを見つけるように新規ルーターが試行されます。

NUD はノード間 (ホスト間、ホスト - ルーター間、ルーター - ホスト間を含む) のすべてのパスに使用されます。使用されるルーティング・プロトコルに同様のメカニズムがまだ組み込まれていない場合には、NUD がルーター間通信に使用されることもあります。近隣未到達検出の詳細については、RFC 2461 を参照してください。

17.3.2 ステートレス・アドレス自動構成

IPv6 の 128 ビット・アドレス・フィールドによって IPv4 固有の問題の多くが解決されましたが、TCP/IP 管理者にとってそのアドレス・サイズ自体が潜在的な問題を含んでいます。そのため IPv6 は、初期化時にインターフェースに対してアドレスを自動的に割り当てる機能を持つ設計になっています。その意図は、TCP/IP 管理者の役割のアクションを最小限にして、最小の場合はアクションがなくてもネットワークが作動可能になるようにすることです。IPv6 ノードは一般に常に自動構成を使って自身の IPv6 アドレスを取得します。そのためには、ステートフル自動構成と呼ばれる DHCP (621 ページの 17.5 『IPv6 での DHCP』を参照) を使用するか、または IPv6 の新機能で ICMPv6 を使用するステートレス自動構成によって行います。

ステートレス自動構成プロセスは、RFC 2462 – IPv6 Stateless Address Autoconfiguration で定義されています。このプロセスは以下のステップで構成されます。

1. システムの始動時、ノードがインターフェース・ハードウェアからインターフェース・トークンを取得することにより自動構成を開始します。インターフェース・トークンにはたとえば、トークンリングやイーサネット・ネットワークでの 48 ビット MAC アドレスがあります。
2. ノードが、リンク・ローカルの暫定ユニキャスト・アドレスを作成します。これは事前割り当てのリンク・ローカル接頭部 (FE80::/10) をインターフェース・トークンに結合して行われます。
3. ノードが、その暫定アドレスが固有であることの検証を試みます。そのためにその暫定アドレスをターゲットとして近隣送信請求メッセージを送出します。アドレスがすでに使用されている場合、ノードは応答で近隣公示を受信します。この時点で自動構成プロセスは停止します。(この場合はノードの手動構成が必要です。)
4. 応答を受信しなかった場合、ノードはリンク・レベル・アドレスを自身のインターフェースに割り当てます。次にホストが 1 つ以上のルーター送信請求を、全ルーターのマルチキャスト・グループに送信します。ルーターが存在している場合、ルーターはルーター公示で応答します。ルーター公示を受信しなかった場合、ノードは DHCP を使ってアドレスおよび構成情報の取得を試みます。DHCP サーバーが応答しなかった場合、ノードはリンク・レベル・アドレスを使い続けて、同一リンク上の他のノードだけと通信を行うことができます。
5. ルーター送信請求に対する応答でルーター公示を受信した場合、そのメッセージには、自動構成プロセスに進む方法をノードに通知するいくつかの情報が含まれています (図 233 を参照)。次の情報が含まれています。

M フラグ 管理対象アドレス構成

このビットがセットされていると、ノードは DHCP を使って自身の IP アドレスを取得する必要があります。

○ フラグ

その他のステートフル構成

このビットがセットされていると、ノードは DHCP を使って他の構成パラメーターを取得する必要があります。

接頭部オプション

ルーター公示の接頭部オプションに A ビット (自立アドレス構成フラグ) がセットされていると、ステートレス・アドレス自動構成に接頭部が使用されます。

6. ステートレス・アドレス構成が使用される場合はルーター公示から接頭部が取り入れられ、インターフェース・トークンに追加されてグローバル・ユニキャスト IP アドレスが形成されます。これがネットワーク・インターフェースに割り当てられます。
7. 作動中のノードは定期的なルーター公示を引き続き受信します。公示の中の情報が変わった場合、ノードは適切なアクションを起こさなければなりません。

ステートレスとステートフルの両方の構成を同時に使用できるということに注意してください。IP アドレスを取得するためにステートレス構成を使うが、さらに構成情報を取得するために DHCP を使用することはよくあります。ただし DHCP サーバーがなくても、小規模ネットワークでも大規模ネットワークでもプラグ・アンド・プレイ構成が可能です。

ステートレス・アドレス構成プロセスでは、同一インターフェースに複数のアドレスを割り振ることができるため、ネットワークに混乱を与えない、サイトのすべてのノードの安全な再番号付けも可能です (たとえば新しいネットワーク・プロバイダーに切り換えるときに新しいアドレッシングを必要とする場合)。詳細については、RFC 2462 を参照してください。

17.3.3 マルチキャスト・リスナー・ディスカバリー (MLD)

ルーターが特定のマルチキャスト・グループの数を発見するために使用するプロセスは、マルチキャスト・リスナー・ディスカバリー (MLD) と呼ばれます。MLD は ICMPv6 のサブセットであり、IPv4 の場合の IGMP (241 ページの 6.2 『Internet Group Management Protocol (IGMP)』を参照) と同等の機能があります。この情報はルーターによって、マルチキャスト・ルーティング・プロトコルが使用されているすべてのところへ送られるため、マルチキャスト・パケットは、該当するマルチキャスト・アドレスを待機しているノードがあるすべてのリンクに正しく送達されます。

MLD は、RFC 2710 - Multicast Listener Discovery (MLD) for IPv6 に規定されています。MLD は図 237 に示すフォーマットの ICMPv6 メッセージを使用します。

Vers.	Traffic Class	Flow Label	
Payload Length		Next = 58	Hops = 1
(Link Local)Source Address			
Destination Address			
Type	Code = 0	Checksum	
Max. Response Delay		Reserved	
IP Multicast Address			

図 237 MLD メッセージ・フォーマット

メッセージの IPv6 ヘッダーの次のフィールドに注意してください。

拡張ヘッダー情報 58 (ICMPv6 メッセージ拡張ヘッダー情報)。

最大ホップ数 常に 1 に設定されます。

ソース・アドレス リンク・ローカル・ソース・アドレスが使用されます。

MLD メッセージ自体では次に注意してください。

タイプ MLD メッセージには次の 3 つのタイプがあります。

130 マルチキャスト・リスナー照会

照会には次の 2 つのタイプがあります。

- 一般照会：リンク上で受信待機されているマルチキャスト・アドレスの検出に使用されます。
- マルチキャスト・アドレス特定照会：リンク上で特定のマルチキャスト・アドレスを

待機しているノードがあるかどうかの検出に使用されます。

131 マルチキャスト・リスナー・レポート

ノードによって使用され、そのノードがマルチキャスト・アドレスを待機していることが報告されます。

132 マルチキャスト・リスナー終了

ノードによって使用され、そのノードがマルチキャスト・アドレスの待機を中止していることが報告されます。

コード

送信側によって 0 に設定されますが、受信側では無視されます。

最大応答遅延

応答レポートが送信されるまでの最大許容遅延を設定します。このパラメーターは照会メッセージでのみ有効です。このパラメーターを大きくすると、ネットワーク上に応答側が多数ある場合の突然の高トラフィックのバーストを防ぐことができます。

マルチキャスト・アドレス

照会メッセージの場合、一般照会のときはこのフィールドはゼロに設定され、マルチキャスト・アドレス特定照会のときは特定の IPv6 マルチキャスト・アドレスが設定されます。

応答または終了メッセージの場合、このフィールドには受信待機されているマルチキャスト・アドレスが入ります。

ルーターは MLD を使って、自身が接続されている各リンク上で受信待機されているマルチキャスト・アドレスを調べます。ルーターが知る必要があるのは、特定のアドレスを待機しているノードがリンク上にあることです。待機しているそれらのノードのユニキャスト・アドレスや、待機しているノードの数は知る必要はありません。

ルーターは自身の各リンクについての一般照会を、すべてのノードのリンク・ローカル・アドレス (FF02::1) に対して定期的に送信します。マルチキャスト・アドレスを待機しているノードがその照会を受信すると、そのノードは自身が待機しているマルチキャスト・アドレスごとに遅延タイマー (0 から最大応答遅延の間で任意) を設定します。各タイマーが設定時間を超えるたびに、ノードは該当するマルチキャスト・アドレスを含むマルチキャスト・リ

スナー・レポート・メッセージを送信します。ノードがマルチキャスト・アドレスについての別のノードのレポートを受信したときに、ノードのそのアドレス用のタイマーがまだ設定時間内だったときは、ノードはそのタイマーを停止してそのアドレスについてのレポートは送信しません。このようにタイマー・メカニズムを使って、重複レポートが送信されるのを防ぎ、過剰なまたはバースト性のトラフィックが発生するのを防ぎます。

ルーターは、自身の各リンク上で分かっている各マルチキャスト・アドレスのリストを管理し、それらのマルチキャスト・アドレス用のタイマーを設定します。それらのタイマーのいずれかが、そのアドレスについてのレポートを受信しないうちに設定時間を超えると、ルーターはそのアドレスを待機しているノードはもうないと見なして、そのアドレスをリストから除去します。ルーターはレポートを受信するたびに、その特定のアドレス用のタイマーをリセットします。

ノードがマルチキャスト・アドレスの待機を終了したときに、そのノードがルーターにレポートを送信したリンク上の最後のノードだった（つまりそのタイマー遅延が別のノードのレポートの受信によって割り込みを受けなかった）場合は、そのノードはルーターに対してマルチキャスト・リスナー終アメッセージを送信します。ノードがタイマーの設定時間内に別のノードによって割り込みを受けた場合は、ノードはリンク上の他のノードがまだマルチキャスト・アドレスを待機しているの見なして、終了メッセージは送信しません。

ルーターは終了メッセージを受信すると、マルチキャスト・アドレス特定メッセージをリンク上に送信します。そのメッセージに対する応答でレポートを受信しなかった場合、ルーターはそのマルチキャスト・アドレスを待機しているノードはもうないと見なして、そのアドレスを自身のリストから除去します。

17.4 IPv6 における DNS

IPv6 では、128 ビット・アドレスの導入に伴い、ネットワーク・ユーザーが別のネットワーク・ユーザーをネットワーク装置の IP アドレスで識別するのは一層困難になっています。そのためドメイン・ネーム・サービスを使用することがさらに必要になります (291 ページの 8.1 『Domain Name System (DNS)』を参照)。

IPv6 アドレスの保管と検索をサポートするために多くの DNS 拡張が規定されています。それらは RFC 1886 – DNS Extensions to Support IP Version 6 で定義されています。この RFC は提案標準で、状態は「任意選択 (Elective)」です。ただし、この RFC の使用可能度の拡張の作業も進行中であり、それについては同名のインターネット・ドラフトに記述されています。

次の拡張が規定されています。

- 新規リソース・レコード・タイプの AAAA。これによりドメイン・ネームが IPv6 アドレスにマップされます。
- 新規ドメイン。アドレスからのドメイン・ネームのルックアップをサポートするとき 사용합니다。
- 既存の照会の定義の変更。これにより A と AAAA の両方のレコード・タイプの処理が適切に行えます。

17.4.1 IPv6 リソース・レコードのフォーマット

RFC 1886 では、データ・セクションにエンコードされる 128 ビット IPv6 アドレスとタイプ値の 28 (10 進) 以外は、AAAA レコードのフォーマットを A リソース・レコードと同様に定義しています。

逆 (アドレスからホスト名) ルックアップ用に特殊ドメイン IP6.INT が定義されています (IPv4 で使用される *in-addr.arpa* ドメインと同様)。IPv4 の場合と同様にアドレスは逆順に入力しなければなりませんが、10 進表記ではなく 16 進数字を使用します。

例として次の IPv6 アドレスの場合の逆ドメイン・ネーム項目を示します。

2222:0:1:2:3:4:5678:9abc

これが次のようになります。

c.b.a.9.8.7.6.5.4.0.0.0.3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.2.2.2.2.IP6.INT.

ここで上記のアドレスがノード ND1.test.com に関連付けられているとすると、ネーム・サーバーのゾーン・データの項目は次のようになります。

\$origin test.com.

ND1 99999 IN AAAA 2222:0:1:2:3:4:5678:9abc

cba98765400030002000100000002222.IP6.INT. IN PTR ND1 ²

17.4.1.1 リソース・レコード変更の提案

IPv6 のアドレッシング・システムは、単一インターフェースが複数のアドレスを持てるように、そしてアドレスの再番号付けが容易に行えるように設計されています (たとえば企業がサービス・プロバイダーの中の 1 つを変更する場合)。RFC 1886 で規定されている AAAA リソース・レコード・フォーマットを使用すると、再番号付けの変更のときの管理の手間は大きくなります。インターネット・ドラフト DNS Extensions to Support IP Version 6 の現

² この PTR 項目内の予約済み IPv6 アドレスを構成するすべての文字は、ピリオド (.) で区切らなければなりません。この例では、わかりやすくするためにピリオドを省略しています。

行バージョンでは、ネットワークの再番号付けを単純化するために、AAAA リソース・レコードのフォーマットの変更を提案しています。

AAAA レコードのデータ・セクションの提案フォーマットを図 238 に示します。

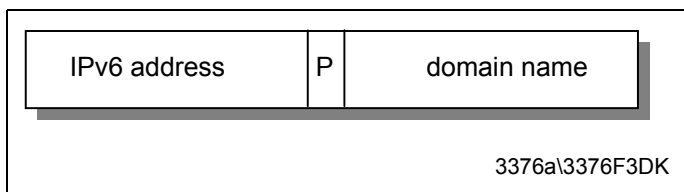


図 238 AAAA リソース・レコード- 提案データ・フォーマット

各部の説明：

IPv6 アドレス 128 ビット・アドレス (アドレスの下位ビットのみ)

P 接頭部の長さ (0-128)

ドメイン・ネーム 接頭部のドメイン・ネーム

このフォーマットの作用を、図 239 に示す例で考えてみます。

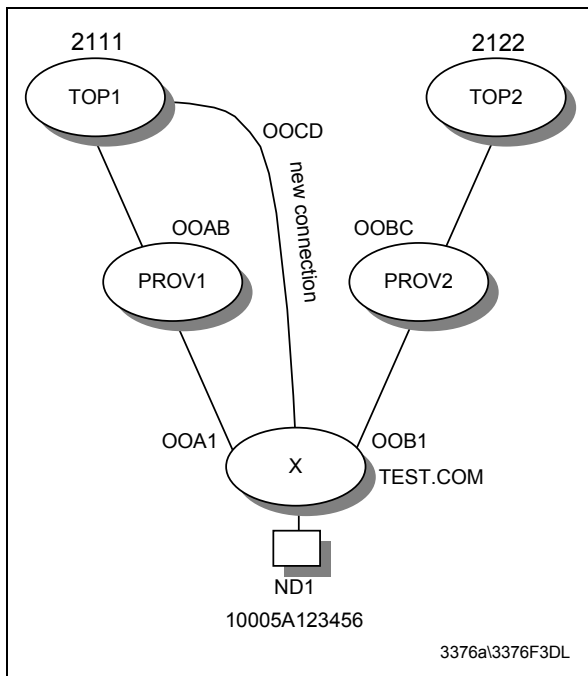


図 239 接頭部の番号付けの例

サイト X は 2 つのプロバイダー PROV1 および PROV2 とマルチホーミングを行っています。PROV1 は最上位プロバイダー TOP1 から伝送サービスを受けます。PROV2 は TOP2 からそのサービスを受けます。TOP1 の最上位アグリゲート ("TLA ID" + 「フォーマット接頭部」) は 2111 です (596 ページの図 227 を参照)。TOP2 の TLA は 2222 です。

TOP1 は次レベル・アグリゲート (NLA) として 00AB を PROV1 に割り当て済みです。PROV2 には NLA として 00BC が TOP2 によって割り当てられています。

PROV1 はサブスクライバー ID として 00A1 をサイト X に割り当て済みです。PROV2 はサブスクライバー ID として 00B1 をサイト X に割り当て済みです。

これらの結果、サイト X のノード ND1 (インターフェース・トークンは 10005A123456) には次の 2 つの IP アドレスが設定されます。

```
2111:00AB:00A1::1000:5A12:3456
2222:00BC:00B1::1000:5A12:3456
```

サイト X はドメイン・ネーム test.com で表されます。各プロバイダーには独自のドメインとして、top1.com、top2.com、prov1.com、および prov2.com

があります。これらの各ドメイン内に、接頭部を保持するための IP6 サブドメインが作成されます。ここでノード ND1 は DNS では次の項目で表すことができます。

```
ND1.TEST.COM AAAA ::1000:5A12:3456 80
IP6.TEST.COM

IP6.TEST.COM AAAA 0:0:00A1:: 32 IP6.PROV1.COM
IP6.TEST.COM AAAA 0:0:00B1:: 32 IP6.PROV2.COM

IP6.PROV1.COM AAAA 0:00AB:: 16 IP6.TOP1.COM

IP6.PROV2.COM AAAA 0:00BC:: 16 IP6.TOP2.COM

IP6.TOP1.COM AAAA 2111::

IP6.TOP2.COM AAAA 2222::
```

このフォーマットによって DNS 管理者のジョブがかなり単純化されて、再番号付けの変更のインプリメントも大幅に容易になります。たとえば、サイト X はプロバイダー PROV1 と PROV2 からのリンクを使うことをやめて、最上位サービス・プロバイダー TOP1 (サイト X に次レベル・アグリゲーション 00CD を割り振る) からの直接接続に投資することに決定したとします。DNS で必要な変更は、2 つの IP6.TEST.COM 項目を次の単一の項目に置き換えることだけです。

```
IP6.TEST.COM AAAA 0:00CD:: 16 IP6.TOP1.COM
```

注：

提案されている AAAA リソース・レコード・フォーマットは、現在作業が進行中であるに過ぎません。A6 リソース・レコードの代替ドラフトも現在計画中です。これらのテクノロジーのどれが標準になるかは今のところ分かりません。

17.5 IPv6 での DHCP

IPv6 では、ステートレス・アドレス自動構成が紹介されましたが、DHCP は、それぞれのアドレッシング方式で制御することを望むサイトのためのステートフルな代替方法として重要なものと見なしています。ステートレス自動構成と共に使うことにより、DHCP では、ノードがそれぞれのアドレスを入手するときに、そのノードに別の構成オプションを渡す手段が実現します。(DHCP の詳細については、131 ページの 3.7 『Dynamic Host Configuration Protocol (DHCP)』を参照してください。)

今のところ、IPv6 での DHCP をカバーした RFC はありません。ただし、2 つのインターネット・ドラフト *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* および *Extensions for the Dynamic Host Configuration Protocol for IPv6* で説明されている作業は進行中です。

17.5.1 DHCPv6 と DHCPv4 との間の相違点

DHCPv6 には、DHCPv4 と比べていくつかの重要な相違点があります。これは、IPv6 プロトコルで継承した拡張機能のいくつかを利用しています。プリンシパルの相違点のいくつかは、以下のとおりです。

- クライアントがブートすると、そこにはリンク・ローカル IP アドレスがあります。これは、DHCP サーバーまたはリレー・エージェントと通信するときに使えます。
- クライアントは、ブロードキャストではなくマルチキャスト・アドレスを使い、サーバーと接触します。
- IPv6 では、インターフェースごとに複数の IP アドレスを使うことができ、DHCPv6 では、要求された場合に、複数のアドレスを用意することができます。
- いくつかの DHCP オプションが不要になりました。たとえば、デフォルト・ルーターは、IPv6 隣接ディスカバリーを使うクライアント側で入手されるようになりました。
- DHCP メッセージ (アドレスの割り振りを含む) は、IPv4 では IP ヘッダーに示されましたが、IPv6 メッセージ拡張機能に示されるようになりました。
- BOOTP 互換性には要件はありません。
- 新しい再構成メッセージがあります。これは、構成変更をクライアントに送信するときにサーバーによって使われます (たとえば、アドレスの存続期間の縮小など)。クライアントは、初期構成を受け取ったら、引き続き再構成メッセージを待機する必要があります。

17.5.2 DHCPv6 メッセージ

今のところ、以下の DHCPv6 メッセージが定義されています。

DHCP Solicit

これは、IP マルチキャスト・メッセージです。DHCP クライアントは、メッセージを、すべての DHCP エージェント (リレーおよびサーバー) でウェルノウン・マルチキャスト・アドレス FF02::1:2 へ転送します。リレーに受信された場合、そのリレーは、メッセージを、すべての DHCP サーバーでウェルノウン・マルチキャスト・アドレス FF05::1:3 へ転送します。

DHCP Advertise

これは、DHCP Solicit への応答として送信されたユニキャスト・メッセージです。DHCP Solicit がリレーによって転送された場合で、DHCP サーバーが同じリンク上にあるかリレー・エージェントを経由している場合、DHCP サーバーは、送信請求をしているクライアントへ直接に応答します。公示メッセージには、1 つ以上の拡張機能が含まれる場合があります (DHCP オプション)。

DHCP Request

クライアントが DHCP サーバーを突き止めたら、DHCP 要求 (ユニキャスト・メッセージ) が送信され、アドレスまたは構成パラメーター (あるいはその両方) が要求されます。この要求は、サーバーがクライアントと同じリンクにない場合には、リレーによって転送されなければなりません。この要求には、サーバーで利用できるすべてのオプションのサブセットとすることができる拡張機能 (クライアントによって指定されたオプション) が含まれている可能性があります。

DHCP Reply

DHCP 要求 (クライアントへ直接に送信するか、リレーを経由して送信することが可能) への応答として送信された IP ユニキャスト・メッセージです。拡張機能には、クライアントへコミットされるアドレスまたはパラメーター (あるいはその両方) が含まれます。

DHCP Release

クライアントによってサーバーへ送信される IP ユニキャストです。これは、解放される予定のリソースをサーバーに通知します。

DHCP Reconfigure

サーバーによって 1 つ以上のクライアントに送信される IP ユニキャストまたはマルチキャスト・メッセージ。新しい構成情報が利用できることをクライアントに通知することを目的としています。クライアントは、サーバーからこれらの新しい変更を要求する DHCP 要求でこのメッセージに応答する必要があります。

DHCPv6 の詳細については、最新のインターネット・ドラフトを参照してください。ドラフトの正確な番号は変更されている可能性があるので注意してください。ドラフトは、以下のアドレスにあります。

<http://www.ietf.org/internet-drafts/draft-ietf-dhc-dhcpv6-19.txt>

17.6 IPv6 でのモバイル・サポート

本書の執筆時には、IPv6 でのモバイル・サポートをカバーする RFC はありません。しかし、Mobility Support in IPv6 という最新のインターネット・ドラフトで説明されている件が進行中です。

IPv6 プロトコルでの特定の拡張機能は、特にモバイル環境に適したものです。たとえば、Mobile IPv4 とは異なり、モバイル・ノードの代わりに「外部エージェント」の振りをするというようなルーターの要件はありません (655 ページの第 19 章『モバイル IP』を参照)。それで、隣接ディスカバリーおよびアドレスの自動構成では、ノードは、ローカル・ルーターからの特別なサポートなしで、ホームとは別に運用できるようになっています。さらに、モバイル・ノードがホーム位置から離れている間にそのモバイル・ノードへ送信されたほとんどのパケットは、Mobile IPv4 で使ったような完全なカプセル化を使うのではなく、IPv6 ルーティング (拡張機能) ヘッダーを使ってトンネリング化することができます。これにより、モバイル・ノードにパケットを送達するときのオーバーヘッドが減少します。

IPv6 でのモバイル・サポートの詳細については、次のアドレスにある最新のインターネット・ドラフトを参照してください。

<http://www.ietf.org/internet-drafts/draft-ietf-mobileip-ipv6-14.txt>

17.7 インターネットでの移行 - IPv4 から IPv6 へのマイグレーション

インターネットで IPv6 の利点が認められる場合、新しい IPv6 ホストおよびルーターを既存の IPv4 と共存させてデプロイするときに、移行期間が必要になります。RFC 2893– Transition Mechanisms for IPv6 Hosts and Routers および RFC2185 – Routing Aspects of IPv6 Transition では、古いシステムと新しいシステムとの間の互換性と、インターネットの機能に影響を与えない漸進的な移行の両方を実現するために採用されているさまざまなメカニズムが定義されています。このような技法は、まとめて *Simple Internet Transition (SIT)* と呼ばれることがあります。このような移行では、以下の技法が採用されています。

- IPv4 と IPv6 との間に相互運用しなければならないホストおよびルーター用の二重スタック IP 実装。
- IPv6 アドレスへの IPv4 アドレスの埋め込み。IPv6 ホストは、IPv4 と相互運用される割り当てられたアドレスで、IPv4 ホスト・アドレスは IPv6 に割り当てられます。
- IPv4 ルーター・ネットワーク経由で IPv6 パケットを運ぶことを目的とした IPv6-over-IPv4 トンネリング・メカニズム。

- IPv4/IPv6 ヘッダー変換。この技法は、IPv6 の実装が十分に進歩し、2、3 の IPv4 専用システムだけが残る場合に使うことを意図しています。

これらの技法は、特に Novell IPX のような他のプロトコルにも適しています。このプロトコルには、同じようなインターネットワーク・レイヤー・セマンティクスとアドレッシング方式があり、IPv6 アドレス・スペースの一部に容易に割り当てることができます。

17.7.1 二重 IP スタックの実装 - IPv6/IPv4 ノード

新しい IPv6 ノードが既存の IPv4 システムとの互換性を維持できるようにする最も簡単な方法は、二重 IP スタックの実装を実現することです。IPv6/IPv4 ノードは、通信相手となるシステムのタイプに応じて、IPv6 パケットか IPv4 データグラムのいずれかを送受信できます。このノードには、128 ビット IPv6 アドレスと 32 ビット IPv4 アドレスの両方があり、必ずしもこれらに関連させる必要はありません。図 240 には、同じリンクで IPv6 システムと IPv4 システムの両方と通信している二重スタック IPv6/IPv4 システムが示されています。

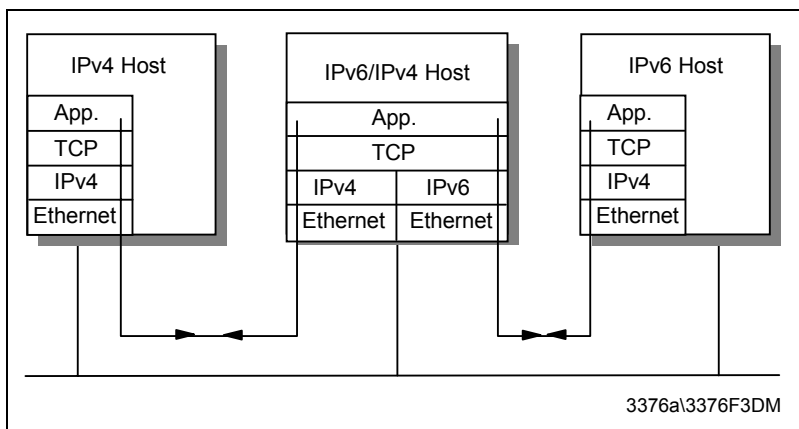


図 240 IPv6/IPv4 二重スタック・システム

IPv6/IPv4 ノードでは、IPv6 アドレスを入手するために、ステートレス自動構成かステートフル自動構成を使うことができます。さらに、IPv4 アドレスを入手するために、DHCP、BOOTP、または手動構成など、任意の方法を使うことも可能です。しかし、ノードが自動トンネリングを実行する予定の場合、IPv6 アドレスは IPv4 互換のアドレスでなければなりません。この場合、アドレスの低い位の 32 ビットが IPv4 アドレスとして機能します。(592 ページの 17.2.3 『IPv6 アドレッシング』を参照)

概念的には、二重スタック・モデルは、プロトコルをインターネットワーク・レイヤーの中でだけ 2 本立てにしたものと考えることができます。ただし、いずれかのスタックを使っているときに操作を行う場合には、すべてのトランスポート・レイヤー・プロトコルに対して関連した変更を加えることが必要なのは明らかです。アプリケーションが、IPv6 の増加したアドレス・スペースなど、IPv6 機能を利用する場合には、アプリケーションの変更も必要になります。

IPv6/IPv4 ノードで、別のシステムと通信する場合には、そのシステムの機能と送信する必要のあるパケットのタイプを知る必要があります。ここで、DNS が主要な役割を演じます。617 ページの 17.4 『IPv6 における DNS』で説明されているように、新しいリソース・レコード・タイプ AAAA は、ホスト名を IPv6 アドレスに割り当てるために定義されています。ネーム・サーバー検索の結果により、ノードが該当システムと通信するときの方法が決まります。ノードの DNS で見つかったレコードは、実行しているプロトコルによって異なります。

- IPv4 専用ノードには、DNS に IPv4 アドレスが含まれた A レコードだけがあります。
- IPv4 専用ノードと相互運用できる IPv6/IPv4 ノードには、IPv4 互換の IPv6 アドレスを含む AAAA レコードと、同等な IPv4 アドレスを含む A レコードがあります。
- IPv4 専用ノードと相互運用できない IPv6 専用ノードには、IPv6 アドレスを含む AAAA レコードだけがあります。

IPv6/IPv4 ノードは、DNS によって戻される情報に基づいて、使用する予定のプロトコルについての決定を行うため、AAAA レコードを DNS に組み込むことは、IPv6 システムと IPv4 システムとの間で相互運用するときの前提条件になります。ネーム・サーバーでは、必ずしも IPv6 を利用できるプロトコル・スタックを使う必要はありませんが、別のレコード・タイプをサポートしていなければならない点に注意してください。

17.7.2 トンネリング

IPv6 または IPv6/IPv4 システムが、以前の IPv4 ネットワークで通信しようとする他の同様なシステムとは区別されている場合、IPv6 パケットは IPv4 ネットワーク経由でトンネリングする必要があります。

IPv6 パケットは、IPv4 を経由して非常に簡単にトンネリングされます。IPv6 パケットは IPv4 データグラム内でカプセル化されます。つまり、IPv6 パケットに完全な IPv4 ヘッダーが追加されるということです。IPv4 データグラム内に IPv6 パケットが存在するということは、IPv4 ヘッダーの 41 というプロトコル値によって示されます。

IPv4 ネットワークを経由して IPv6 パケットをトンネリングする場合、*自動*と*構成済み*の 2 種類があります。

17.7.2.1 自動トンネリング

自動トンネリングは、IPv4 互換のアドレスに依存しています。トンネリングする時についての決定は、IPv4 をルーティングするネットワーク領域を経由して送信する必要のあるパケットを持つ IPv6/IPv4 ホストによって行われます。このホストは、以下の規則に従います。

- 宛先が IPv4 か IPv6 によって割り当てられたアドレスである場合、受信側は IPv6 を利用できないため、IPv4 を使ってパケットを送信します。それ以外の場合で、その宛先が同じサブネットにあるのであれば、受信側は IPv6 を利用できるため、IPv6 を使ってパケットを送信します。
- 宛先が同じサブネット上にはないが、IPv6 を利用できるサブネット上に少なくとも 1 つのデフォルト・ルーターが存在するか、その宛先の IPv6 ルーターに向かって構成された経路がある場合には、IPv6 を使ってパケットをそのルーターに送信します。それ以外の場合で、アドレスが IPv4 互換のアドレスである場合には、自動 IPv6-over-IPv4 トンネリングを使ってパケットを送信します。アドレスが IPv4 互換のアドレスではない場合には、宛先は IPv6 専用アドレスを持つノードです。ここには、IPv4 にルーティングされた領域を経由して接続します。この場合、さらに IPv6 にルーティングされることはありません。したがって、宛先に到達することは不可能です。

注

トンネルを使用するには、IP アドレスが IPv4 互換でなければなりません。IPv6 専用アドレスは IPv4 を使用してアドレッシングできないので、IPv6 専用アドレスに到達するために自動トンネリングを使用することはできません。IPv6/IPv4 ノードから IPv4 マップ・アドレスへのパケットは、IPv4 専用ノードを参照しているのでトンネリングできません。

上記の規則では、トンネルよりも IPv6 ルーターを優先して使うことが強調されていますが、これには以下の 3 つの理由があります。

- IPv4 ヘッダーをカプセル化することがないので、オーバーヘッドが少ない。
- IPv6 専用の機能が使える。
- IPv6 ルーティング・トポロジがデプロイされると、以前に存在した IPv4 トポロジよりも優先的に使われる。

ノードは、IPv6 にルーティングされた領域に付属しているのか IPv4 にルーティングされた領域に付属しているのかを知る必要はありません。IPv6 ルー

ターがサブネット上で構成されていれば必ずそれを使い、構成されていなければトンネリングを使います (この場合、IPv4 にルーティングされた領域に付属していると思われます)。

自動トンネリングは、ホスト間でも、ルーターとホストの間でも構いません。ソース・ホストは、可能であれば IPv6 パケットを IPv6 ルーターに送信しますが、そのルーターは同じことを実行できない可能性があるため、宛先ホスト自体への自動トンネリングを実行する必要があります。トンネリングよりも IPv6 ルーターを使うことが優先されるため、トンネルは常にできる限り「短い」ものにされます。しかし、IPv6 では、同じホップ単位ルーティング・パラダイムを使うため、トンネルは必ず宛先ホストに至るまでずっと伸ばされます。ホストは、パケットが宛先ホストに到達する前に、最終的に IPv6 互換領域に入るのかどうかを判別できません。受信側まで伸ばされていないトンネルを使うには、構成済みのトンネリングを使う必要があります。

自動トンネリングで使われるメカニズムは非常に単純です。

- カプセル化する IPv4 データグラムでは、IPv6 ソース・アドレスおよび宛先アドレスの下位 32 ビットを使い、同等の IPv4 アドレスを作成します。さらに、プロトコル番号を 41 (IPv6) にセットします。
- 受信側ノードのネットワーク・インターフェース・レイヤーは、着信パケット (または IPv4 データグラムがフラグメント化された場合にはパケット) が IPv4 に属するものと見なし、二重 IPv6/IPv4 インターネットワーク・レイヤーの IPv4 部分の上方へ渡します。
- 次に IPv4 レイヤーは、データグラムを通常の方法で受け取り、必要であればフラグメントを再アセンブルし、プロトコル番号である 41 を記録します。それから、IPv4 ヘッダーを除去し、元の IPv6 パケットをインターネットワーク・レイヤーの IPv6 部分の「横道」に渡します。
- IPv6 コードは、元のパケットを通常どおりに処理します。パケット内の宛先 IPv6 パケットは、ノードの IPv6 アドレス (IPv4 データグラムをカプセル化するときに使われた IPv4 アドレスと一致する IPv4 互換アドレス) であるため、このパケットは最終的な宛先に到達します。IPv6 は、どのような拡張ヘッダーであっても通常どおりに処理し、パケットの残りのペイロードを、最後の IPv6 ヘッダーにリストされている次のプロトコルに渡します。

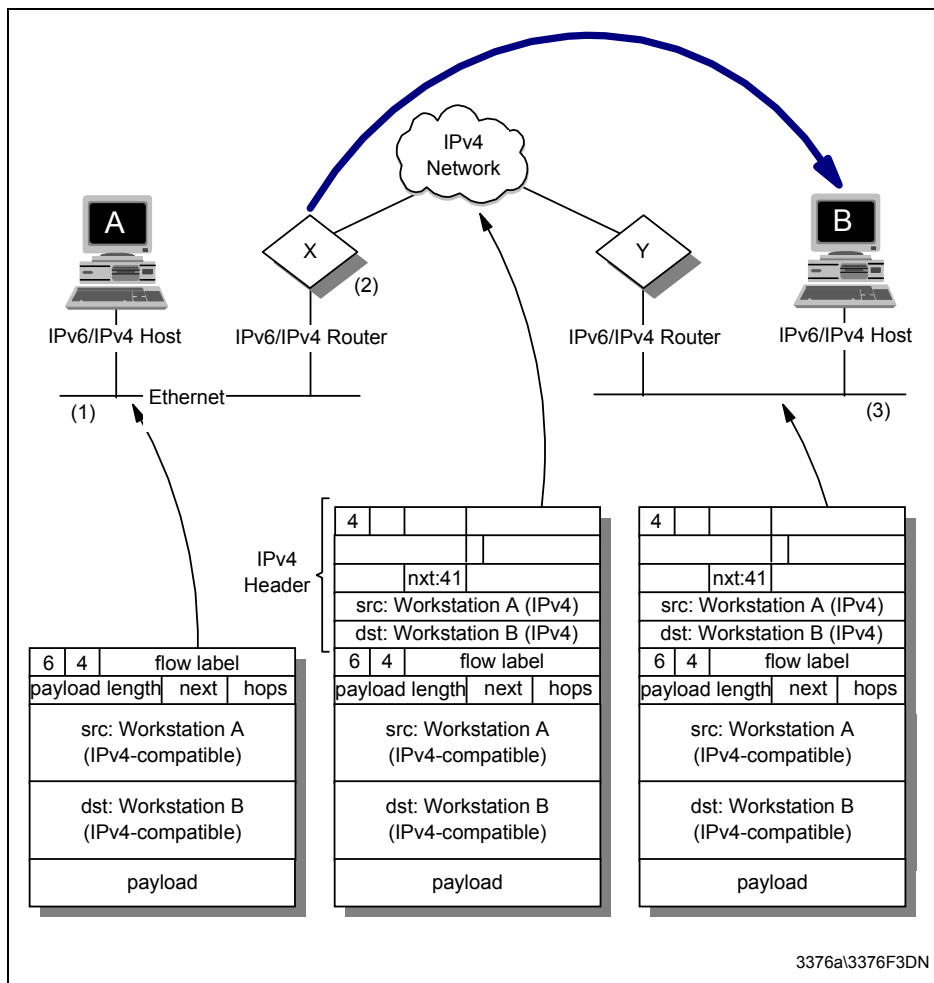


図 241 ルーター対ホストの自動トンネリング

図 241 は、IPv4 ネットワークで隔てられた 2 つの IPv6/IPv4 ノードを表しています。これらのワークステーションは両方とも IPv4 互換 IPv6 アドレスを持っています。ワークステーション A は以下のようにしてワークステーション B にパケットを送信します。

1. ワークステーション A は、そのローカル・リンク上の IPv6 対応ルーター (X) からルーター送信請求メッセージを受信します。これはこのルーターにパケットを転送します。
2. ルーター X は、IPv4 互換アドレスに由来する IPv4 の送信元アドレスと宛先アドレスを使用して、パケットに IPv4 ヘッダーを追加します。その後、

このパケットは IPv4 ネットワークを越えてワークステーション B まで転送されます。これがルーター対ホストの自動トンネリングです。

3. IPv4 データグラムがワークステーション B の IPv4 スタックに受信されます。プロトコル・フィールドは次のヘッダーが 41 (IPv6) であることを示しているので、IPv4 ヘッダーはデータグラムから除去され、次いで残りの IPv6 パケットが IPv6 スタックによって処理されます。

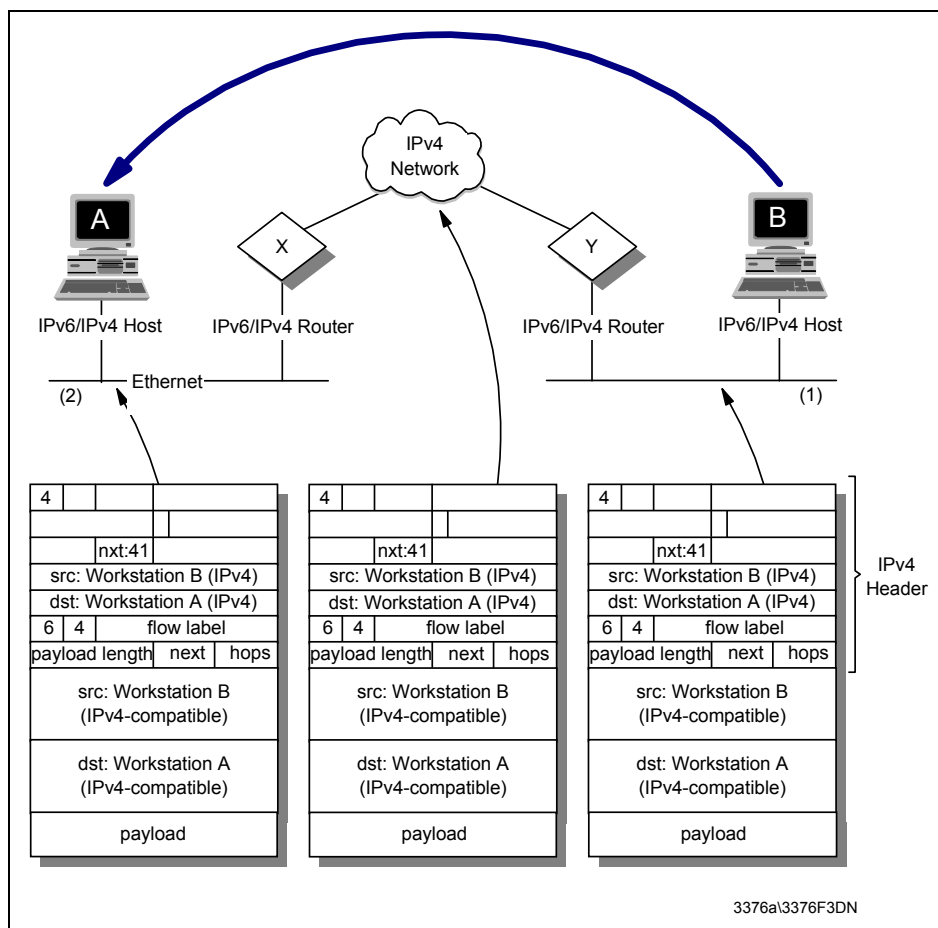


図 242 ホスト対ホストの自動トンネリング

図 242 では、ワークステーション B が以下のように応答しています。

1. ワークステーション B はそのローカル・リンク上に IPv6 対応ルーターを持っていません。したがって、これは IPv4 ヘッダーを自分の IPv6 フレームに追加し、その結果生じた IPv4 データグラムを IPv4 ネットワークを介

してワークステーション A の IPv4 アドレスに直接転送します。これがホスト対ホストの自動トンネリングです。

2. IPv4 データグラムがワークステーション A の IPv4 スタックに受信されます。プロトコル・フィールドは次のヘッダーが 41 (IPv6) であることを示しているので、IPv4 ヘッダーはデータグラムから除去され、次いで残りの IPv6 パケットが IPv6 スタックによって処理されます。

17.7.2.2 構成トンネリング

構成トンネリングは、IPv6-over-IPv4 のホスト対ルーターまたはルーター対ルーターのトンネルに使用されます。送信ホストまたは転送ルーターは、経路にネクスト・ホップのほかにトンネル終アアドレス (これは常に IPv4 互換アドレスです) も含まれるように構成されます。カプセル化のプロセスは自動トンネリングの場合と同じですが、IPv4 宛先アドレスは IPv6 宛先アドレスの下位 32 ビットではなく、トンネル終了の下位 32 ビットから取られます。この場合、IPv6 の宛先アドレスと送信元アドレスは IPv4 互換アドレスである必要はありません。

トンネルの終了のルーターは、IPv4 データグラムを受信すると、自動トンネリングの終了のノードと全く同じ方法でこれを処理します。元の IPv6 パケットがこのルーターの IPv6 層に渡されると、このルーターは自分が宛先ではないことを認識し、他の IPv6 パケットに対してするのと同様にそのパケットを最終宛先に転送します。

もちろん、トンネルから出た後にこの IPv6 パケットが再び他のルーターへトンネルされる可能性もあります。

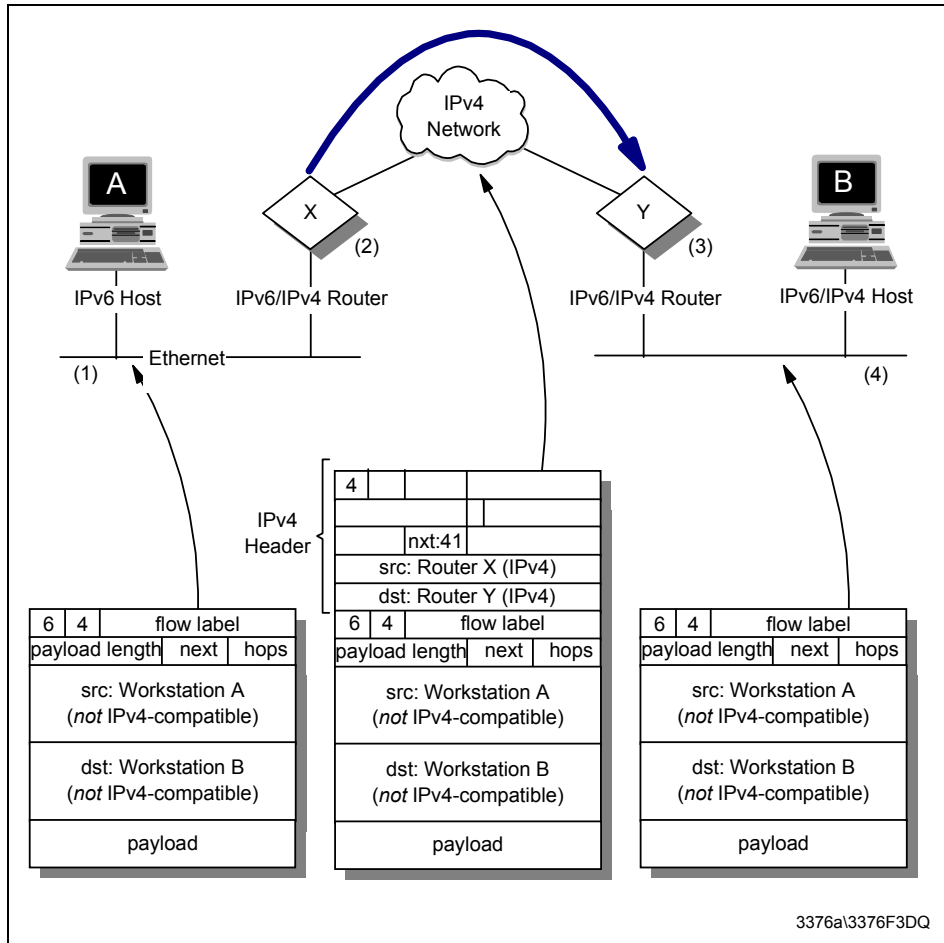


図 243 ルーター対ルーターの構成トンネリング

図 243 は、IPv4 ネットワークによって隔てられた IPv6 専用ノードを表しています。IPv6/IPv4 ルーターである X と Y の間にルーター対ルーターのトンネルが構成されています。

1. ワークステーション A はワークステーション B に送信する IPv6 パケットを構成します。これはそのローカル・リンク (X) 上で公示を行っている IPv6 ルーターにパケットを転送します。
2. ルーター X はパケットを受信しますが、宛先サブネットへの直接 IPv6 接続を持っていません。しかし、このサブネットへのトンネルが構成されています。したがって、ルーターは IPv4 ヘッダーをトンネル終了 (ルーター

Y) の宛先アドレスと共にパケットに追加し、このデータグラムを IPv4 ネットワークを介して転送します。

3. ルーター Y の IPv4 スタックがフレームを受信します。これは、プロトコル・フィールドの値が 41 であることを認識して IPv4 ヘッダーを除去し、残りの IPv6 パケットをその IPv6 スタックに渡します。IPv6 スタックは宛先 IPv6 アドレスを読み取って、パケットを転送します。
4. ワークステーション B が IPv6 パケットを受信します。

17.7.3 ヘッダーの変換

IPv6/IPv4 ノードのインストールでは、既存の IPv4 システムとの後方互換性が保たれます。しかし、IPv6 へのネットワークのマイグレーションがかなり進展した段階に達している場合、新しくインストールされるシステムは IPv6 専用になる可能性があります。したがって、IPv6 専用システムが残っている IPv4 専用システムと通信するための要件が発生します。IPv6 専用ノードが IPv4 専用ノードと相互操作するためには、ヘッダーの変換が必要です。ヘッダーの変換は、IPv6 によってルーティングされるエリアと IPv4 によってルーティングされるエリアの間の境界上の IPv6/IPv4 ルーターによって実行されます。

変換ルーターは IPv6 パケットから完全にヘッダーを除去し、それを対応する IPv4 ヘッダーで置き換えます (またはその逆を行います)。ルーターは、2 つのヘッダー内のフィールド間の正しいマッピングを行うことに加え、送信元アドレスと宛先アドレスを IPv4 マップ・アドレスから実際の IPv4 アドレスへと変換する (IP アドレスの下位 32 ビットを取ることによって) 必要があります。逆の方向では、ルーターは ::FFFF /96 接頭部を IPv4 アドレスに追加することによって IPv4 マップ・アドレスを作成します。送信元または宛先のいずれかの IPv6 アドレスが IPv6 専用の場合、ヘッダーは変換できません。

IPv4 ホストが 1 つしか含まれていないサイトの場合でも、これが通信する必要がある IPv6 ノードにはすべて IPv4 マップ・アドレスが含まれていなければならないことにご注意ください。

17.7.4 インターオペラビリティの要約

2 つのノードが相互操作できるかどうかは、それらの機能とアドレスによって決まります。

IPv4 ノードは以下と通信できます。

- ローカル・リンク上の任意の IPv4 ノード
- 任意の IPv4 ノードと、IPv4 ルーターを介して

- IPv4 マップ・アドレスを持つ任意の IPv6 ノードとヘッダーの変換を通して

IPv6 ノード (IPv6 専用アドレス) は以下と通信できます。

- ローカル・リンク上の任意の IPv6 ノード
- 任意の IPv6 ノードと、ローカル・リンク上の IPv6 ルーターを介して (ルーターからの IPv4 ネットワークを通るトンネリングが必要な可能性あり)

IPv6 ノード (IPv4 マップ・アドレス) は以下と通信できます。

- ローカル・リンク上の任意の IPv6 ノード
- 任意の IPv6 ノードと、ローカル・リンク上の IPv6 ルーターを介して (ルーターからの IPv4 ネットワークを通るトンネリングが必要な可能性あり)
- 任意の IPv4 ノードとヘッダーの変換を通して

IPv6/IPv4 ノード (IPv4 互換アドレス) は以下と通信できます。

- ローカル・リンク上の任意の IPv4 ノード
- 任意の IPv4 ノードと、ローカル・リンク上の IPv4 ルーターを介して
- ローカル・リンク上の任意の IPv6 ノード
- 任意の IPv6 ノードと、ローカル・リンク上の IPv6 ルーターを介して (ルーターからの IPv4 ネットワークを通るトンネリングが必要な可能性あり)
- 任意の IPv6/IPv4 ノード (IPv4 互換アドレス) と、ホスト対ホストのトンネルを介して

17.8 IPv6 への推進

IPv6 ネットワークの導入のための推進力は、IPv6 を必要とする新機能の必要性や IPv4 アドレス・スペースの逼迫である可能性があります。これらのどちらをより重視するかは、組織によって異なります。たとえば、大規模な古くからの内部 IPv4 ネットワークを持つ営利企業は、そのネットワーク内のアドレス・スペースに問題があるのでない限り、作動中の多数の IPv4 ホストとルーターをアップグレードするとは考えられません。しかし、ビジネスに欠くことのできない新しいアプリケーションに IPv6 でしか使用できない機能が必要な場合や、IPv6 専用アドレスを使用している他の組織に接続する必要がある場合には、IPv6 の配備に投資する可能性もあります。

しかし、初めて IP ネットワークをインプリメントする企業は、アドレス自動構成などの IPv6 のいくつかの機能に興味を抱く可能性があります。ただし、現在 IPv6 のインプリメントを考える人はだれであれ、このプロトコルが今もなお正に発展の途上にあるということを覚えておく必要があります。IPv4 も、新しい RFC やインターネット標準が絶えず作成されているので、まだ開発途上にあると言えますが、IPv6 の場合は、DHCP などの主要なプロトコルが本書執筆時点でもまだインターネット・ドラフトの段階にあります。現在の Internet バックボーンは IPv4 ルーターで構成されており、IPv6 プロトコルが広く使用され検査されるようになるまでは、これらの実動ルーターの所有者が IPv6 にアップグレードすることによってこれらを危険にさらすとは考えにくいものです。

ietf IPng (次世代) プロジェクトの 1 つの支流は、Internet 規模の IPv6 仮想ネットワークで、物理 IPv4 Internet の上の層になる 6Bone です。6Bone は、既存の IPv4 バックボーンを横断するトンネルでリンクされた、IPv6 パケットをサポートする多数の島々からなります。6Bone は、IPv6 プロトコルおよび製品の検査に広く使用されています。IPv6 の信頼性が大きくなり、IPv6 機能を持つ多くの製品が提供されるようになるにつれて、6Bone はいずれは ISP の実動バックボーンとユーザー・ネットワークの IPv6 対応ルーターで置き換えられることが予想されます。

17.9 リファレンス

IPv6 に関する詳細情報は以下の RFC の中にあります。

- RFC1752 – The Recommendation for the IP Next Generation Protocol
- RFC1883 – Internet Protocol, Version 6 (IPv6)
- RFC2460 – Internet Protocol, Version 6 (IPv6)
- RFC1191 – Path MTU Discovery
- RFC2373 – IP Version 6 Addressing Architecture
- RFC2374 – An IPv6 Aggregatable Global Unicast Address Format
- RFC2375 – IPv6 Multicast Address Assignments
- RFC2463 – Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification
- RFC2461 – Neighbor Discovery for IP Version 6 (IPv6)
- RFC2462 – IPv6 Stateless Address Autoconfiguration
- RFC1886 – DNS Extensions to Support IP Version 6
- RFC2893 – Transition Mechanisms for IPv6 Hosts and Routers
- RFC2185 – Routing Aspects of IPv6 Transition

第 18 章 Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) 規格は、多層スイッチングの進展途上にある現行の作業を表しています。MPLS の基本目標は、ラベル・スワッピング・パラダイムと従来のネットワーク層ルーティングとを統合することです。この統合によって、データ転送および拡張 QoS 機能のためのネットワークの位置決めが効率的になります。

初期の MPLS の作業は IPv4 だけを対象としていました。しかしそのコア・テクノロジーは、他のネットワーク層プロトコルにも拡張可能です。さらに、MPLS は特定の層テクノロジーに限定されていません。それはネットワーク層パケットを伝送することが可能なすべてのメディア上で機能します。

基本的な MPLS 仕様は、RFC 3031 に記述されています。本書を執筆している時点で、RFC は推奨される規格の状態になっています。テクノロジーの発展に従って、機能強化は後続の RFC で文書化される予定です。

18.1 MPLS の概要

MPLS 環境では、従来型のレイヤー 3 またはネットワーク層ルーティング（つまり IP ルーティング）を使用して、ネットワークを通過するためのパスを決めます。パスが決まると、データ・パケットがネットワークを横断する際に、ノードからノードへとスイッチされます。

18.1.1 従来型のルーティング・モデル

従来の非接続ネットワークでは、すべてのルーターがレイヤー 3 ルーティング・アルゴリズムを実行していました。パケットがネットワークを横断する際に、パスの途中にある各ルーターはそのパケットについて個別の転送判断を行います。パケット・ヘッダーに含まれる情報およびルーティング・アルゴリズムから取得した情報を使用して、ルーターはそのパケットが次に転送される宛先を選択します。IP ネットワークでは、このプロセスには各パケットの IP ヘッダーに含まれている宛先アドレスを IP ルーティング・テーブルから取得した最も詳細な経路にマッチングさせることが含まれます。この比較プロセスによって、そのパケットを次に転送する宛先が決まります。レイヤー 3 ヘッダーについてのこの分析および種別は、プロセッサで集中的に行うことができます。従来の非接続環境では、このアクティビティはエンドツーエンド・パスの途中にあるすべてのノードで生じていました。

18.1.2 MPLS 転送モデル

MPLS 環境では、ネットワーク内での最適なパスが事前に識別されます。その後、データ・パケットが MPLS ネットワークに入る際に、入口装置はレイ

ヤー 3 ヘッダーを使用して事前に決められたパスの 1 つをパケットに割り当てます。この割り当てを使用して、エンドツーエンド・パスを参照しているラベルをパケットに付加します。このラベルはネットワークを横断する際にデータ・パケットを伴います。パスの途中にある後続のルーターは、ラベル内の情報を使用して次に転送される装置を判別します。これらの装置はラベル内の情報だけを取り扱うので、プロセッサで集中的に行うレイヤー 3 の分析および種別は入口点だけで生じます。

18.1.3 その他の利点

ネットワークのコアにある装置上での処理要件を少なくすることに加えて、MPLS には従来型のレイヤー 3 ルーティングと比較していくつかの追加の利点があります。それらについては、続くいくつかのセクションで説明します。

18.1.3.1 トラフィック・エンジニアリング

トラフィック・エンジニアリングは、トラフィック・パターンがリソースをバランスよく使用するようにネットワーク・パスを選択するためのプロセスです。

従来型のインテリア・ゲートウェイ・プロトコル (IGP) アルゴリズムに基づくルーティングでは、リソースをバランスよく使用しないネットワーク・パスが選択されることがあります。これらの環境では、過剰に使用されるネットワーク・リソースや、過少に使用されるネットワーク・リソースが生じます。限られた程度のエンジニアリングは、ネットワーク・リンクに関連した IGP メトリックを取り扱うことによって提供されます。しかし、冗長パスの数が多き環境ではこの作業を管理することが困難になります。

トラフィック・エンジニアリングの利点を活用するために、MPLS を IGP アルゴリズムに関連して使用することができます。MPLS には、ネットワークの横断に使用する特定の経路データ・パケットを指定する機能があります。このようにデータ・パケットを明示的にルーティングすることによって、データの特定のストリームが必ず特定のパスを使用できるようになります。これらのデータ・ストリームをモニターおよび管理することにより、ネットワーク・リソースを効率的に使用できます。従来の IP ルーティングのソース・ルーティング・オプションによって、明示的なルーティングが使用可能となります。しかし、これはプロセッサ集約のアクティビティーなので、その使用は制限されています。MPLS は明示的なルーティングを可能な限り効率的に使用します。

MPLS には、データ・パケットの明示経路を判別するときに IP パケット・ヘッダーの外部にあるフィールドを分析する機能もあります。たとえば、ネットワーク管理者はパケットがネットワークに入れられた方法と場所に基づいて、トラフィック・フロー・ポリシーを作成することができます。従来のネットワークでは、この情報が使用可能なのは入口点だけでした。この追

加の分析によって、管理者は高水準の制御を行うことが可能になり、サービスのレベルがさらに予測可能になります。

18.1.3.2 Quality of Service ルーティング

QoS ルーティングは、パスが必要なレベルのサービスを提供できるように特定のデータ・ストリーム用に経路を選択する機能です。これらのサービス・レベルは、ネットワーク内で受け入れ可能なレベルの帯域幅、遅延、またはパケット・ロスを指定できます。これにより、全体的なネットワーク・ポリシーに基づいて、異なるレベルのサービスを提供することが可能になります。

必要な QoS を備えたネットワーク・パスを提供するためには、しばしば明示的ルーティングを使用することが必要になります。たとえば、特定の帯域幅を割り振る必要のある特定のストリームにパスを割り当てるのは直接的な方法です。しかし、複数のストリームによる結合された帯域幅が既存の容量を超過する可能性もあります。このシナリオでは、同じ入口ノードと出口ノードとの間にあるものであっても、個別のストリームに個別に経路指定しなければならない場合があります。これには、標準のトラフィック・エンジニアリングに備わっているものよりも高いレベルの細分性が必要となります。

MPLS 環境では QoS ルーティングを提供するための 2 つのアプローチがあります。

- MPLS ラベルには、サービス・クラス (COS) 情報が含まれています。トラフィックがネットワークをフローする際に、この情報を使用してネットワークでのホップごとにトラフィックの知的な優先順位付けを行うことができます。
- MPLS ネットワークは、入口装置と出口装置との間に複数のパスを提供することができます。それぞれのパスは、異なるレベルのパスを提供するように設計されています。その後、トラフィックはネットワークに入る際に適切なパスを知的に割り当てられます。

これらのアプローチによって、パケットをサービス・クラスのカテゴリに簡単に分類することができます。ローカルのネットワーク管理ポリシーは、各カテゴリに提供されるサービスを実際に決定します。

18.1.3.3 マルチプロトコル・サポート

Multiprotocol Label Switching 規格は、IPv4、IPv6、IPX、および AppleTalk を含む既存のネットワーク層プロトコルをサポートしています。この規格はさらに、イーサネット、トークンリング、FDDI、ATM、フレーム・リレー、およびポイント間リンクに対するリンク層サポートも提供しています。この規格を他のプロトコルおよびネットワーク・タイプにも拡張する活動が継続しています。

18.2 MPLS ネットワークのコンポーネント

このセクションでは、MPLS 環境で使用される主要な概念および用語について詳しく説明します。

18.2.1 用語

続くいくつかのセクションでは、MPLS で使用される用語を定義します。

18.2.1.1 転送等価クラス (FEC)

FEC は同じ方法で転送されるレイヤー 3 パケットのグループです。このパケット内のすべてのグループは、同じネットワーク・パスを経由し、同じ優先順位を持ちます。FEC 内のパケットは、異なるレイヤー 3 情報を持つこともあります。しかし、転送決定を簡単に行えるように、これらのパケットは互いに見分けがつかないようにになっています。

FEC グループの一般的な例を以下に示します。

- IP ルーティング・テーブル内で最も詳細な 2 つの経路が等しいパケットのセット。
- IP ルーティング・テーブル内で最も詳細な 2 つの経路が等しく、サービス設定の IP タイプも等しいパケットのセット。

MPLS ネットワークで、FEC はラベルによって識別されます。

18.2.1.2 ラベルおよびラベル付きパケット

上記のとおり、ラベルは固有の FEC を識別します。MPLS 装置は同じラベルが付けられたパケットすべてを同じ方法で転送します。

ラベルは一对の MPLS 装置間でローカルに有効です。それは、2 つの装置間でラベルと FEC との間のマッピングについて記述している合意事項を表しています。ラベルはローカルで有効なので、同じラベルをすべてのホップで使用する必要がないため、MPLS をより大きな環境で使用するための拡張容易性が増します。

転送に使用するレイヤー 2 テクノロジーに応じて、MPLS ラベルはデータ・フレーム内の異なる場所に配置することができます。レイヤー 2 テクノロジーがラベル・フィールドをサポートしている場合、MPLS ラベルはネイティブのラベル・フィールドにカプセル化されています。ATM ネットワークでは、VPI/VCI フィールドを使用して MPLS ラベルを保管することができます。同様に、DLCI フィールドを使用して MPLS ラベルをフレーム・リレー・ネットワーク内に保管することができます。

レイヤー 2 テクノロジーがラベルをネイティブにサポートしていない場合、MPLS ラベルは特にこの目的で付加されたカプセル化ヘッダー内に入れられます。このヘッダーはレイヤー 2 ヘッダーと IP ヘッダーとの間に存在します。このように専用のヘッダーを使用することにより、MPLS サービスがすべてのレイヤー 2 テクノロジーで可能になります (図 244 を参照してください)。

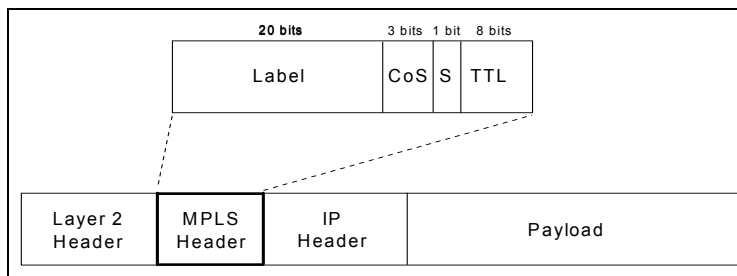


図 244 32 ビット MPLS ヘッダー

MPLS ヘッダーの内容には、以下のものが含まれます。

- MPLS ラベルの実際の値を含むラベル・フィールド。
- キューイングに影響を与え、パケットがネットワークを横断する際にそれに適用されるアルゴリズムを廃棄するために使用する COS フィールド。
- 階層ラベル・スタックをサポートしている S (スタック) フィールド。
- 従来型の IP TTL 機能をサポートしている TTL (存続時間) フィールド。

ラベル付きパケットは、ラベルがエンコードされているパケットです。拡張された MPLS 機能をサポートするため、実際にはパケットに複数のラベルが含まれることもあります。これはラベル・スタックと呼ばれます。このスタックは、個別のラベル相互間での順序のある関係を確立します。スタックは後入れ先出しモデルを使用してインプリメントされます。この機能の詳細については、645 ページの 18.2.4 『ラベル・スタックおよびラベル階層』で解説します。

18.2.1.3 ラベル・スタック・ルーター (LSR)

ラベル・スタック・ルーターは、同様にネイティブのレイヤー 3 を転送できる MPLS ノードです。MPLS ネットワークには、重要な 2 つのタイプの LSR があります。

- **入口ノード**は、MPLS ネットワークを MPLS 機能を実行しないノードに接続します。入口ノードはトラフィックが MPLS ネットワークに入るときに、それを処理します。

- ・ **出口ノード**は、MPLS ネットワークを MPLS 機能を実行しないノードに接続します。出口ノードはトラフィックが MPLS ネットワークから出るときに、それを処理します。

18.2.1.4 ネクスト・ホップ・ラベル転送項目 (NHLFE)

NHLFE は MPLS ノードによってパケットを転送するために使用されます。ノードをフローする FEC ごとに 1 つ以上の NHLFE があります。それぞれのノードは以下の情報を含む NHLFE 情報ベースの保守を担当します。

- ・ パケットのネクスト・ホップのアドレス。
- ・ ラベル・スタック上で実行される操作：
 - スタックの最上部にあるラベルを、指定した新規のラベルに置き換えます。これは、古いラベルのポップおよび新規ラベルのプッシュとして知られています。
 - スタックの最上部にあるラベルをポップします。
 - スタックの最上部にあるラベルを指定した新規のラベルに置き換えてから、指定した 1 つ以上の新規のラベルをラベル・スタック上にプッシュします。この処置が完了すると、スタックには 2 つ以上の MPLS ラベルが含まれています。
- ・ パケットの転送に使用されるデータ・リンクのカプセル化 (オプション)。
- ・ パケットの転送に使用されるラベル・スタックのエンコード (オプション)。
- ・ パケットを正常に処理するために必要な他の情報。

18.2.1.5 着信ラベル・マップ

ILM は MPLS ノードによってラベル付きパケットを転送するために使用されます。着信パケット内のラベルは、ILM への参照として使用されます。ILM 情報によって、ノードは転送情報を含む NHLFE のセットを選択することができます。

ILM はラベルを NHLFE のグループにマップすることがあります。これにより、コストの等しい複数のパスでロード・バランスを行うことが可能になります。

18.2.1.6 FEC から NHLFE へのマップ (FTN)

FTN は、ラベルなしで着信し転送前にラベルを付ける必要のあるパケットを処理するために、MPLS ノードによって使用されます。ラベルなしのデータ・パケットには、入口 MPLS ノードで特定の FEC が割り当てられます。この FEC は FTN への参照として使用されます。FTN マップによって、ノードは転送情報を含む NHLFE のセットを選択することができます。この活動は、MPLS ネットワークの入口ノードで実行されます。

FTN はラベルを NHLFE のグループにマップすることがあります。これにより、コストの等しい複数のパスでロード・バランスを行うことが可能になります。

18.2.2 ラベル・スワッピング

ラベル・スワッピングは、データ・パケットをネクスト・ホップ装置に転送するために MPLS ノードによって使用されます。このプロセスは、パケットがラベル付きで着信してもラベルなしで着信しても使用されます。このプロセスは、ATM およびフレーム・リレーがトラフィックをバーチャル・サーキットを介して転送する際に使用するメソッドに似ています。

18.2.2.1 ラベル付きパケットの転送

MPLS ノードは着信パケットのスタックの最上部にあるラベルを調べます。このノードは ILM を使用して、ラベルを NHLFE にマップします。NHLFE はパケットの転送先およびラベル・スタックに対して実行する操作を指示します。この情報を使用して、ノードは新規のラベル・スタックをエンコードしてから、生成されたパケットを転送します。

18.2.2.2 ラベルなしパケットの転送

MPLS ノードはネットワーク層ヘッダーおよび FEC の判別に必要な他の情報を調べます。このノードは FTN を使用して、FEC を NHLFE にマップします。処理はラベル付きパケットと同じになりました。NHLFE はパケットの転送先およびラベル・スタックに対して実行する操作を指示します。この情報を使用して、ノードは新規のラベル・スタックをエンコードしてから、生成されたパケットを転送します。

以下の図は、MPLS 環境でのラベル・スワッピングを示しています。

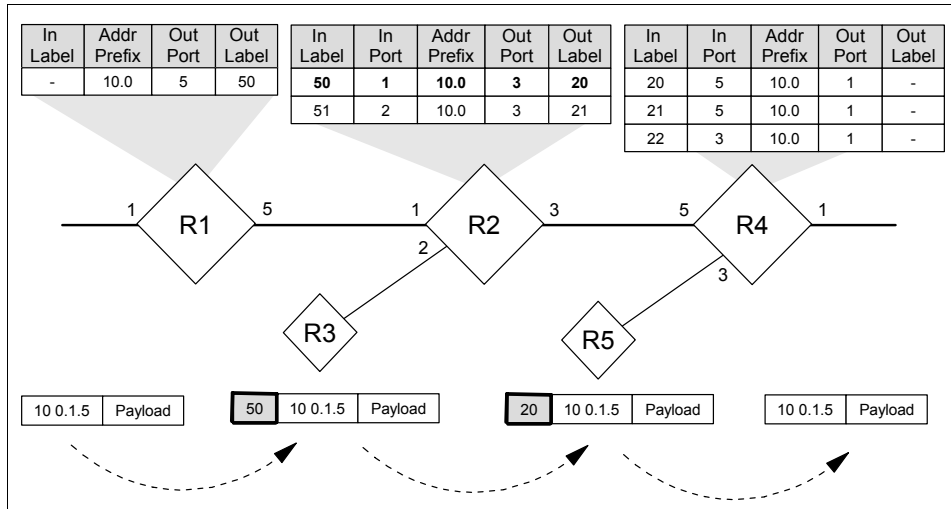


図 245 MPLS 環境でのラベル・スワッピング

注

ラベル・スワッピング環境で、ネクスト・ホップ・ルーターは常に MPLS 情報から判別されます。これにより、パケットが従来型のルーティング・アルゴリズムから取得されるパスとは異なるパスを横断することになる場合があります。

18.2.2.3 終わってから 2 番目のホップ・ポッピング

これは MPLS ラベルを出口ノードではなく終わってから 2 番目のノードでポップする機能です。アーキテクチャーの観点から、このタイプの処理も許可されています。ラベルの目的は、パケットをネットワークを介して出口ノードまで転送することです。終わってから 2 番目のノードがパケットを出口ノードに転送することを決定した後は、ラベルには機能がなくなります。それをパケットに含める必要はありません。

終わってから 2 番目のノードはスタックをポップして、NHLFE から取得したネクスト・ホップ・アドレスに基づいてパケットを転送します。出口ノードがパケットを受け取ると、以下の 2 つのアクティビティーのいずれかが生じます。

- ・パケットにラベルが含まれます。これが生じるのは、終わってから 2 番目のノードが 2 つ以上のラベルを持つノードを処理した場合です。このシナリ

オでは、今スタックの最上部にあるラベルが、出口ノードが転送決定を行うために必要なラベルです。

- パケットにラベルが含まれません。このシナリオでは、LSP 出口が標準のネットワーク層パケットを受け取ります。ノードはローカル IP ルーティング・テーブルを使用して、転送決定を行います。

18.2.3 ラベル交換パス (LSP)

LSP は、特定の FEC に属するパケットが横断する MPLS ノードのセットを表します。このセットは、順序がある、単一方向のリストです。トラフィックはリストのヘッドエンドにあるノードからリストのテールエンドにあるノードに向かってフローします。図 245 で示されているトラフィック・フローの LSP は、<R1, R2, R4> です。

MPLS ネットワークでは、LSP は以下の 2 つの方法のいずれかによって確立されます。

- 独立 LSP 制御：各 LSR はラベルを FEC にバインドする判断を独立して行います。その後、そのラベルを対等ノードに配布します。これは従来型の IP ルーティングに似ています。各ノードはパケットを転送する方法について独立して決定を行います。
- 順序 LSP 制御：LSR は FEC の出口 LSR であるか、または FEC へのネクスト・ホップから FEC にバインドされたラベルをすでに受け取っている場合、ラベルをその特定の FEC だけにバインドします。トラフィック・エンジニアリング・ポリシーがインプリメントされた環境では、順序 LSP 制御を使用して特定の FEC 内のトラフィックが特定のパスを必ず経由するようにします。

649 ページの 18.3 『ラベル配布プロトコル』は、MPLS 環境でラベル情報を交換するために使用されるプロシーチャーについて詳しく説明しています。

18.2.4 ラベル・スタックおよびラベル階層

ラベル付きパケットには、複数のラベルを含めることができます。ラベルは後入れ先出しのスタック内で保守されます。スタックは、ラベルのセット内に順序のある階層をインプリメントします。

この階層は、MPLS ノードがパケットをパートナー MPLS ノードに転送したとき、それらのノードがパケットのホップごとのパスで連続したルーターではない場合に使用されます。その場合、2 つの MPLS ノード間にトンネルが作成されます。トンネルが LSP としてインプリメントされ、ラベル交換を使用してトラフィックがトンネルを介して転送されます。

トンネルを介して送信されるトラフィックのセットは、FEC を形成します。トンネル内の各 LSR は、その FEC にラベルを割り当てる必要があります。

パケットをトンネルを介して送信するために、トンネル入口ノードはトンネル出口ノードが認識しているラベルをラベル・スタックにプッシュします。その後、トンネル入口ノードはネクスト・ホップ・ノードによって認識されているラベルをプッシュして、データ・パケットをトンネルを介して転送します。

たとえば、ネットワークに LSP <R1, R2, R3, R4> が含まれていると想定します。この例では、R2 および R3 は直接接続されていませんが、LSP トンネルのピア・エンドポイントです。ネットワークを横断する LSR の実際のシーケンスは、<R1, R2, R21, R22, R3, R4> となります。次の図は、この構成を示しています。

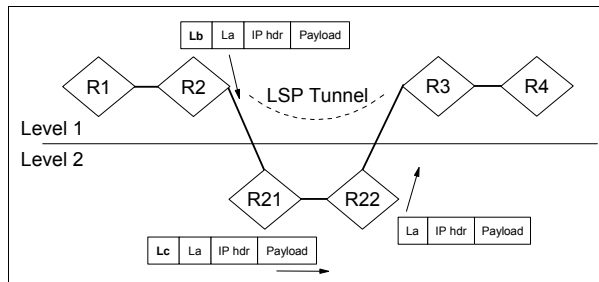


図 246 LSP トンネル

このネットワークを横断するパケットはレベル 1 LSP <R1,R2, R3, R4> を移動した後に、R2 から R3 に移動するときレベル 2 LSP <R2, R21, R22, R3> を使用します。レベル 1 LSP の視点からは、R2 のピア装置は R1 および R3 となります。レベル 2 の視点からは、R2 のピア装置は R21 となります。

この図を使用した場合、パケットが LSP トンネルを介して送信されるときに以下のアクションが生じます。

- R2 はラベル付きパケットを R1 から受け取ります。パケットには、単一のラベルが含まれています。ラベル・スタックの深さは 1 です。
- R2 はこのラベルをポップして、R3 によって認識されているラベルをプッシュします。このラベルは、La と呼ばれます。
- R2 には、R21 によって認識されているラベルも含める必要があります。R2 はラベルを既存のレベル 1 ラベルの上にプッシュします。このラベルは、Lb と呼ばれます。ラベル・スタックには、2 つの項目が含まれています。
- R2 はパケットを R21 に転送します。

- R21 は R2 によって付加されたレベル 2 のラベル (Lb) をポップして、R22 によって認識されているレベル 2 のラベルをプッシュします。このラベルは、Lc と呼ばれます。R21 はレベル 1 のラベルを処理しません。ラベル・スタックには 2 つの項目が含まれています。
- R21 はパケットを R22 に転送します。
- R22 は R21 が付加したレベル 2 のラベルを検討して、それが R2-R3 トンネル内の終わりから 2 番目のホップであることを知ります。R22 はレベル 2 のラベル (Lc) をポップして、パケットを R3 に転送します。ラベル・スタックには 1 つの項目が含まれています。

18.2.5 BGP 環境での MPLS スタック

図 247 に図示したネットワークは、3 つの自動システムを示しています。この環境には、IP ルーティングの 2 つのクラスが含まれています。

- 各自動システムは IGP を実行して、AS 内での接続性を保守しています。たとえば、R2、R21、R22、および R3 は OSPF を使用して AS 2 内の経路を保守することがあります。
- 各自動システムは BGP を実行して、自動システム相互間の接続性を保守します。たとえば、ボーダー・ルーター R1、R2、R3、および R4 は BGP を使用して、AS 間のルーティング情報を交換します。

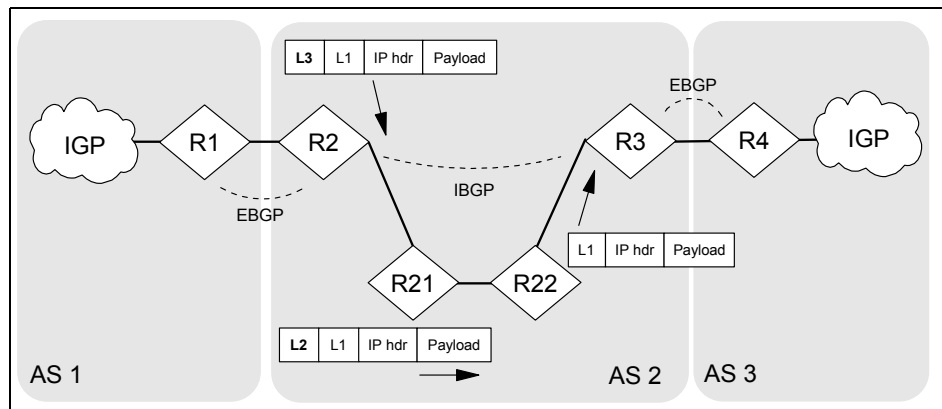


図 247 MPLS 環境での自動システムの接続

このサンプル・ネットワークでは、BGP によって知られている経路を BGP ボーダー・ルーターではない装置 (たとえば、R21、R22) に配布しないことをお勧めします。これにより、これらの装置上で IP ルーティング・テーブルを保守するために必要な CPU 処理が最小化されます。さらに、これらの装置上で BGP ルーティング・アルゴリズムを実行する必要がなくなります。

MPLS LSP スタックの使用は、この環境をインプリメントするために使用できます。この構成では、BGP 経路は BGP ピアにだけ配布されて、ピアとピアの間にあるホップごとのパスに沿った内部ルーターには配布されません。LSP トンネルは、以下の結果となるように構成されています。

- 各ピアが BGP 経由で配布するアドレス・プレフィックスごとに、ラベルを配布する。これらのラベルは、同じ AS 内でピアに配布されます。
- IGP が各 BGP ボーダー・ルーターのホスト経路を保守する。各内部ルーターは、各 IGP ネイバーのホスト経路のラベルを配布します。

R2 が AS 3 経由で接続されたネットワーク宛てのラベルのないパケットを受け取る状況を考えます。このパケットは、R2 にローカルに接続された LAN セグメントまたは AS 2 内の他の LAN セグメントから発信された可能性があります。パケットが AS1 から発信されたものであれば、以前にラベルが付けられていたと考えられます。

- R2 はローカルの IP 転送テーブルを検索して、必要な宛先アドレスのために最も特定された経路を判別します。この経路は、BGP を介して記憶されます。BGP ネクスト・ホップは R3 となります。
- R3 は前に最長マッチに対してラベルをバインドしていて、そのラベルを R2 に配布しています。このラベルは、L1 と呼ばれます。
- AS 2 内のすべての装置は IGP に参加しているので、R3 への経路は AS 2 内のすべての装置のルーティング・テーブルに出現します。
 - R22 は前に R3 に対してラベルをバインドしていて、そのラベルを R21 に配布しています。このラベルは、L2 と呼ばれます。
 - R21 は前に R3 に対してラベルをバインドしていて、そのラベルを R2 に配布しています。このラベルは、L3 と呼ばれます。
- R2 はラベル・スタックを作成することによって、AS 3 に宛てたデータ・パケットを準備します。スタック上の初期項目は、L1 ラベルをプッシュすることによって作成されます。スタック上の最上位項目は、L3 ラベルをプッシュすることによって作成されます。その後ラベル付きパケットは、ネクスト・ホップである R21 に送られます。
- R21 はラベル付きパケットを受け取って、スタック内の最上位項目を検討します。NHLFE 内の情報を使用して、R21 はスタック内の L3 ラベルを L2 ラベルに置き換えます。その後ラベル付きパケットは、ネクスト・ホップである R22 に送られます。
- R22 はラベル付きパケットを受け取って、最上位項目を検討します。R22 は R2-R3 トンネル上で終わりから 2 番目のポップなので、R22 は L2 ラベルをスタック上にポップして、データ・パケットを R3 に転送します。このとき、R3 に転送されるラベル・スタックには 1 つの項目が含まれています。

- R3 はラベル付きデータ・パケットを受け取って、スタック上の L1 ラベルを検討します。NHLFE 内の情報を使用することにより、R3 は古いラベルを R4 にバインドされたラベルに置き換えて、パケットを転送します。

注

MPLS ノードがすでにラベルが付いたパケット上にラベルをプッシュするときにはいつでも、新規のラベルは LSP 出口がその新規のラベルを割り当てたノードである FEC に対応していなければなりません。

18.3 ラベル配布プロトコル

ラベル配布プロトコルは、MPLS ノードがラベルを他のピア・ノードに配布できるようにするプロシーチャーのセットです。この仕様は、他の LSR に割り当てられたラベルおよび関連した意味について知らせるために、LSR によって使用されます。この交換はピアの間での共通の合意事項を確立します。

各 MPLS ノードはローカル IGP に参加して、ネットワーク・トポロジを判別し、ルーティング・テーブルに値を取り込みます。ラベル配布プロトコルはこの情報を使用してラベルを確立します。配布プロトコルが各ノードで実行した後、MPLS ネットワーク全体にはパスの完全なセットと関連したラベルがあるはずです。

ラベル配布プロトコルは、各ピアの MPLS 機能を知るために必要なピア間でのネゴシエーションをも含んでいます。

18.3.1 ラベル分散プロトコルのタイプ

MPLS アーキテクチャーは必要な配布プロトコルを指定することも、プロトコルが 1 つしかない想定することもしていません。そのため、複数の異なる規格が作られています。これらの規格は、以下の 2 つのカテゴリのいずれかに分類できます。

18.3.1.1 既存のプロトコルの拡張

ラベル配布情報が既存のデータ・フロー内に含められるように、既存のプロトコルに対する提議が行われてきました。以下に 2 つの例を示します。

- BGP 拡張：多くの場合、FEC は BGP ピアによって配布されたアドレス・プレフィックスの識別に使用されます。これらの装置が MPLS ラベルを配布することには利点があります。さらに、BGP 経路リフレクターを使用してラベルを配布すると、拡張容易性がかなり高まります。

- RSVP 拡張：この提議は、RSVP 規格を機能強化して LSP 情報の確立および配布をサポートするようにしたものです。これにより、エンドツーエンド・パスに沿ったリソースの割り振りが可能になります。

18.3.1.2 新規プロトコルの開発

さらに、ラベルを配布するためだけの目的で、新規のプロトコルが開発されています。これらの独立型プロトコルは、パスに沿ったすべてのホップで、特定のルーティング・プロトコルの存在に依存していません。このことは、機能拡張されてラベル配布機能を含むようになった既存のプロトコルの 1 つをサポートしていないノードを、LSP が横断しなければならない状況で役立ちます。

18.3.2 ラベル配布メソッド

MPLS ノード相互間でラベル情報を交換するために通信を開始するメソッドとしては、次の 2 つがあります。

- オンデマンドのダウンストリーム：LSR は特定の FEC のラベル・バインディングを要求できます。リクエストはその FEC のネクスト・ホップ MPLS ノードに対して行われます。
- 要求されていないダウンストリーム：LSR は情報を明示的に要求していない LSR へのバインディングを配布することができます。

これらの配布技法は共に、同じネットワークで同時に使用することができます。指定されたピアのセットで、アップストリーム LSR とダウンストリーム LSR は、使用される技法について同意していなければなりません。

18.4 ストリーム・マージ

ストリーム・マージは多数のデータ・フローを単一のダウンストリーム・フローに集約することです。マージを実行している装置は、個々のストリームを合併させて、それらが後続の MPLS ノードによって単一のストリームとして処理されるようにします。マージされたストリームは、単一のラベルによって表されます。マージされたパケットが転送された後は、異なる着信ラベルが付けられたパケットが到着したに関する情報がすべて失われます。

ストリーム・マージは、MPLS 拡張容易性の主要なコンポーネントです。

18.4.1 フレーム・ベースの環境でのマージ

フレーム・ベースの環境でのストリーム・マージは、直接的なものです。マージを実行している装置は複数のアップストリーム・ラベルを単一のダウンストリーム・ラベルにマップします。既存の MPLS スワッピング・プロシージャには変更がありません。

18.4.2 ATM 環境でのマージ

ATM 環境でのストリーム・マージは、より複雑です。ATM では、データ・パケットが AAL5 PDU にカプセル化されて、ATM セルとして送信されます。これらのセルには、特定の VPI/VCI 値があります。VPI/VCI 内のすべてのセルは、シーケンスとなって転送されます。データ・パスに沿ったすべての ATM スイッチがセルの順序を保存する必要があります。PDU を再組み立てする装置は、セルが連続していて、正しい順序で受け取られることを想定しています。

直接的な MPLS ストリーム・マージが ATM 環境で生じると、問題が発生します。このシナリオでは、複数の着信 VC からのセルがインターリーブされて、単一の発信 VC になります。ATM セル・ヘッダーには元の順序を復元するための情報が含まれていないので、元の PDU を再構成する際に問題が発生します。

ATM 環境でのストリーム・マージ中にセルのインターリービングを回避する方法には、以下の 2 つがあります。

- VC マージによって、複数の着信 VC が単一の発信 VC にマージされます。マージを実行している MPLS ノードは、1 つの AAL5 フレームからのセルが他の AAL5 フレームのセルから分離されているようにします。このために、ATM スイッチは 1 つのフレームからのセルの送信を他のフレームからのセルが転送されている間は遅らせます。フレーム標識の最後に達したときに、次のフレームの全体を転送できます。このタイプのバッファリングおよびストア・アンド・フォワード機能は、既存の ATM スイッチでは通常使用できませんでした。
- VP マージによって、複数の VP が単一の発信 VP にマージされます。マージされた VP 内にある異なる VCI は、フレームを複数のソースから区別するために使用されます。

VP マージはより大きな割合のデプロイ済み ATM ギアと互換性があるため、既存のネットワークにデプロイするのが容易であるという利点があります。さらに VP マージには、マージ点で遅延が生じないこと、および新規のバッファリング要件が生じないという利点もあります。

VP マージの主な短所は、各 VP 内での VCI 割り当てを調整する必要があるということです。

MPLS アーキテクチャは、VP マージおよび VC マージの両方のサポートを指定しています。MPLS に参加している ATM スイッチは、ネイバーのスイッチが VP マージまたは VC マージのどちらを実行するのか、あるいはどちらのマージも実行しないのかを判別する必要があります。

18.5 マルチプロトコル・ラムダ・スイッチング

データ・ネットワークがますます複雑になっているため、経済的で拡張が容易なソリューションを提供するアーキテクチャーが必要となっています。これらのソリューションには、パフォーマンスを最適化する機能も備わっている必要があります。これらの要求のいくつかは、MPLS 規格によって取り組まれています。これらの同じビジネス要件は、基礎となる光学式トランスポート・ネットワーク (OTN) にも適用されます。その結果、規格団体は MPLS 機能を光学式相互接続 (OXC) に組み込むための仕様の開発を始めています。この作業は、MP λ S と呼ばれます。

MPLS 機能を光学式ネットワークに拡張することには、以下に示すいくつかの利点があります。

- この作業では、MPLS 用に開発された技術にてこ入れして、光学式チャネルをリアルタイムで提供するために、それを活用できます。その結果、これらのテクノロジーの開発が加速します。
- この作業は、データ環境および光学式環境の両方のネットワーク管理にアプローチするための、一貫したアプローチを提供することができます。その結果、全体的な管理作業が単純化されます。
- この作業は LSP がルーターと OXC 装置が混在する場所を横断できるようにする単一のアーキテクチャーを提供できます。これにより、必要な設備での十分な帯域幅がネットワークによって備えられます。
- この作業により、IP ルーターは大容量の Dense Wave Division Multiplexing (DWDM) 機能を組み込むことになります。

MPLS 機能を光学式ネットワークに拡張することは、それら 2 つのタイプのネットワークにいくつかの類似点があることに基づいています。

- MPLS では、LSP がラベル付きパケットのセットによって横断されるポイントからポイントへのパスを記述しています。MP λ S では、光学式チャネルの証跡が 2 つのアクセス・ポイント間のポイントからポイントへの光学的接続を記述しています。
- MPLS ネットワークでは、LSR が <input port, input label> タプルと <output port, output label> タプルとの間の関係を確立します。同様に光学式トランスポート・ネットワークでは、OXC が <input port, input optical channel> タプルと <output port, output optical channel> タプルとの間の関係を確立します。これらの関係は確立後、データ・パケットの内容がその関係を変更することはできません。
- MPLS ネットワークでは、LSR がネットワークに関連した状態情報を発見、配布、および保守します。OXC は OTN 内でこれらと同じ機能を担当します。

- MPLS ネットワークでは、LSR が既存のトラフィック・エンジニアリング・ポリシーに関連した LSP の作成および保守を担当します。OXC は OTN 内でこれと同じ機能を担当します。
- MPLS ネットワークで、LSP は単一方向です。OTN では、光学式チャネル証跡は同じく単一方向です。

MPλS 環境では、OTN 内のファイバーをリンク（各リンクはチャネルのセットを表す）のセットとして表すことが可能です。IP ルーティング・プロトコル（拡張機能付き）は、光学式ネットワーク・トポロジーについての情報、使用可能な帯域幅、および関係する他の状態情報を配布します。この情報を使用して、光学式チャネル証跡のための明示経路が計算されます。その後、MPLS 配布プロトコルはこれらの証跡を作成します。

このテクノロジーの開発が続くにつれて、MPLS ネットワークと OTN との間の重要な相違に取り組む必要が生じます。

- MPLS ネットワークでは、転送情報は各データ・パケットに含まれるラベルの一部として伝達されます。OTN では、スイッチング情報は波長または光学式チャネルから暗示されます。
- OTN では、ラベル・マージの概念がありません。OXC は複数の波長を 1 つの波長にマージすることができません。
- OXC はラベルのプッシュまたはポップ操作を実行できません。光学の領域でラベルに相当するものは波長です。現在の光学テクノロジーには、波長をプッシュまたはポップするという概念はありません。

第 19 章モバイル IP

モバイル環境で作業する労働人口が増加しているため、モバイル・ネットワーク装置の構成および操作に関する問題が生じています。装置に複数の構成パラメーターのセットを割り振ることも可能ですが、この方法ではアドミニストレーターとユーザーの作業負荷が明らかに増大します。しかしながら、おそらくより重大な問題点は、このタイプの構成を行うと IP アドレスの数に関して無駄が生じるということです。

DHCP および DDNS 環境で、DHCP はそれがネットワークに接続するポイントでの有効な IP アドレスを装置に提供します。DDNS は、装置がネットワークのどこに接続したか、およびどの IP アドレスが割り振られたかには関係なく、その装置をホスト名で位置決めするための手段を提供します。モバイル装置を扱う際の問題に対する代替アプローチは、RFC 2002 – IP モビリティ・サポートで提供されています。IP モビリティ・サポートは一般にモバイル IP と呼ばれ、選択可能な状況にある、推奨される標準です。

19.1 モバイル IP の概要

モバイル IP によって、装置がネットワークに接続するたびに同じ IP アドレス（その ホーム・アドレス）を保持できるようになります。（もちろん、IP アドレスが不正なサブネットにプラグインした装置は、通常は到達できなくなります。）しかし、モバイル装置には気付アドレスもあり、このアドレスによって装置が現在存在しているサブネットに接続することができます。気付アドレスは、モバイル装置のホーム・サブネットに存在する装置であるホーム・エージェントによって管理されます。モバイル装置の IP アドレスにアドレス指定されたパケットはホーム・エージェントが代行受信してから、トンネルを介して気付アドレスに転送されます。それがトンネルの終端に到達すると、データグラムはモバイル装置に渡されます。モバイル・ノードは通常、送信するすべてのソース・アドレスとしてホーム・アドレスを使用します。

モバイル IP を使用すると、複数の場所でネットワークに接続する装置ごとに必要な IP アドレスが 1 つだけとなるので、アドレス不足の問題を解決し、管理の作業負荷を軽減するために役立ちます。

以下の用語がモバイル IP ネットワーク構成で使用されます。

ホーム・アドレス モバイル・ノードに割り振られた静的 IP アドレス。ノードがネットワークに接続する場所には関係なく、このアドレスは不変となります。

ホーム・ネットワーク

モバイル・ノードのホーム・アドレスと一致するネットワーク・プレフィックスのあるサブネット。モバイル・

ノードのホーム・アドレスに送るデータグラムは、常にこのネットワークに経路指定されます。

トンネル カプセル化されたデータグラムが通過するパス。

訪問先ネットワーク

モバイル・ノードが接続しているネットワーク (そのノードのホーム・ネットワークではないもの)。

ホーム・エージェント

ノードの現在場所情報を保守して、ノードがホームに存在しないときにデータグラムをトンネルを介してノードに送る、モバイル・ノードのホーム・ネットワーク上のルーター。

外部エージェント モバイル・ノードの存在を登録し、トンネルを解除してホーム・エージェントのモバイル・ノードからトンネルを介して送られてきたデータグラムをノードに転送する、訪問先ネットワーク上のルーター。

19.2 モバイル IP の操作

モビリティ・エージェント (ホーム・エージェントと外部エージェント) は、ネットワーク上での存在をエージェント公示メッセージによって公示します。これは拡張子のある ICMP ルーター公示メッセージです (658 ページの 19.3 『モビリティ・エージェント公示の拡張子』を参照してください)。モバイル・ノードがエージェント誘導メッセージを使用して、これらのメッセージの 1 つを明示的に要求することもできます。モバイル・ノードは、ネットワークに接続してこれらのメッセージの 1 つを受け取るとき、ホーム・ネットワーク上に存在しているのか、外部ネットワーク上に存在しているのかを判別できます。モバイル・ノードがホーム・ネットワーク上に存在していることを検出すると、モビリティ・サービスを使用しないで通常の方法で実行するようになります。さらに、直前まで他の場所で実行していたモバイル・ノードがホーム・ネットワークに戻ったときには、ホーム・エージェントに再登録することになります。これは登録要求と登録応答を交換することによって行います。

しかしモバイル・ノードが外部ネットワークに移動したことをエージェント公示から検出した場合、それは外部ネットワークの気付アドレスを取得します。このアドレスは外部エージェントから取得するか (外部エージェント自体のアドレスである外部エージェント気付アドレス)、または DHCP (この場合、共存気付アドレスと呼ばれる) などの他のメカニズムによって取得することができます。共存気付アドレスを使用すると、モバイル・ノードが訪問するすべてのネットワークに外部エージェントが存在する必要はなくなるとい

う利点がありますが、DHCP サーバーによって訪問先モバイル・ノードの IP アドレスのプールが使用可能であることが必要です。

モバイル・ノードと外部エージェントとの通信はリンク層レベルで行われることに注意してください。モバイル・ノードの IP アドレスはそのノードが現在存在するサブネットに属していないので、通常の IP ルーティングのメカニズムは使用できません。

モバイル・ノードが気付アドレスを受け取った後、そのノードをホーム・エージェントに登録する必要があります。これは外部エージェントを介して登録要求をホーム・エージェントに転送するか、またはホーム・エージェントに対して直接登録することにより行います (660 ページの 19.4 『モバイル IP の登録プロセス』を参照してください)。

ホーム・エージェントが気付アドレスを新しい位置に登録した後、モバイル・ノードのホーム・アドレスに対するデータグラムは、ホーム・エージェントが代行受信して、トンネルを介して気付アドレスに送られます。そのトンネルの終端は、外部エージェント (モバイル・ノードに外部エージェントの気付アドレスがある場合)、またはモバイル・ノード自体 (それに共存気付アドレスがある場合) となります。最初のデータグラムはトンネルから除去されて、モバイル・ノードに送られます。

モバイル・ノードは通常、標準の IP ルーティング・メカニズムを使用して受け取ったデータグラムに応答します。

モバイル IP の操作は図 248 に示されています。

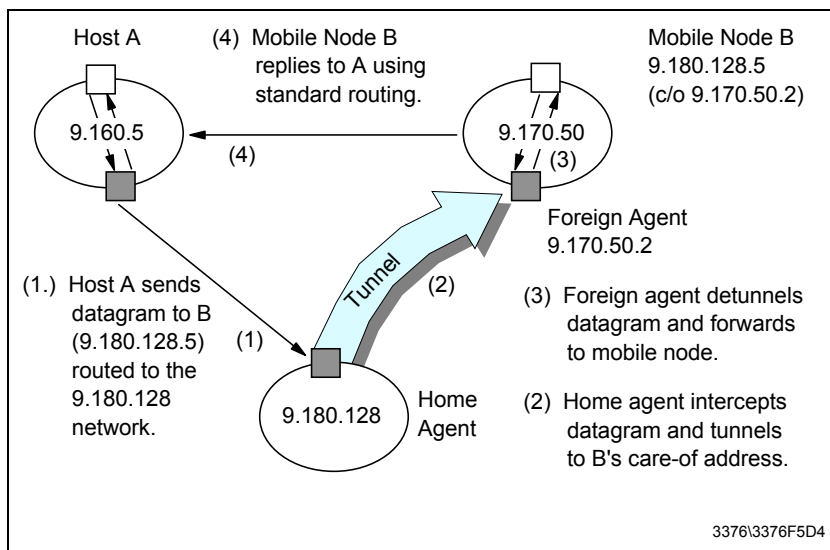


図 248 モバイル IP の操作

19.3 モビリティ・エージェント公示の拡張子

モビリティ・エージェント公示は、以下の 1 つ以上の拡張子を持つ ICMP ルーター公示から成ります。

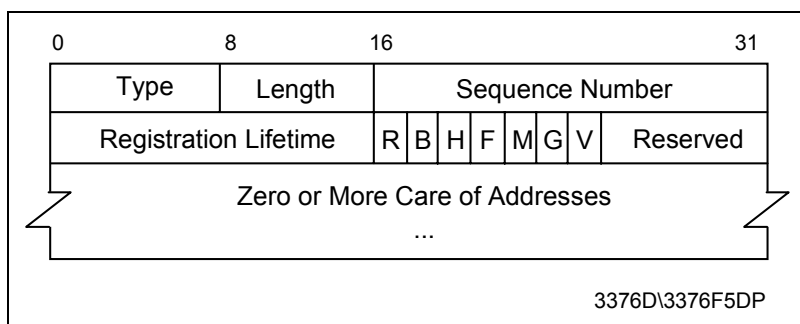


図 249 モビリティ・エージェント公示の拡張子

ここで、

Type 16

Length (6+[4*N])、N は公示された気付アドレスの数。

Sequence number	初期化されて以来、このエージェントによって送信された公示の数。
Registration lifetime	このエージェントが登録要求を受け入れる存続期間 (秒数) の最大値。値の 0xffff は無限大を表します。このフィールドは、ルーター公示そのものの存続期間フィールドとは関係がありません。
R	登録が必要。モバイル・ノードは共存気付アドレスを使用するのではなく、このエージェントに登録する必要があります。
B	ビジー。外部エージェントは追加の登録を受け入れることができません。
H	ホーム・エージェント。このエージェントは、このリンク上でホーム・エージェントとしてのサービスを提供します。
F	外部エージェント。このエージェントは、このリンク上で外部エージェントとしてのサービスを提供します。
M	最小のカプセル化。このエージェントは最小のカプセル化を使用するデータグラムをトンネルを介して受け取ります。
G	GRE カプセル化。このエージェントは GRE カプセル化を使用するデータグラムをトンネルを介して受け取ります。
V	Van Jacobson ヘッダー圧縮。このエージェントは登録済みモバイル・ノードに対してリンク上での Van Jacobson ヘッダー圧縮をサポートしています。
Reserved	この領域は無視されます。
Care-of Address(es)	このエージェントによって公示される気付アドレス (複数も可)。F ビットが設定されている場合、少なくとも 1 つが含まれている必要があります。

外部エージェントは特定の時点でビジーであるために、追加のモバイル・ノードにサービスを提供できないことがあることに注意してください。しかし、エージェント公示 (B ビットの設定された) の送信を続行して、エージェントが失敗したのではないこと、およびそれらが外部エージェントの範囲内に依然として存在することを、すでに登録済みのモバイル・ノードが判別できるようにしなければなりません。

プレフィックス長の拡張子を、モビリティ・エージェント公示の拡張子の後に配置することができます。この拡張子を使用して、ネットワーク・プレフィックスが移動の検出に使用されているとき、(メッセージの ICMP ルーター公示部分にある) 各ルーター・アドレスに適用するビットの数を示すことができます (665 ページの 19.7 『移動の検出』を参照してください)。詳細については、図 250 を参照してください。

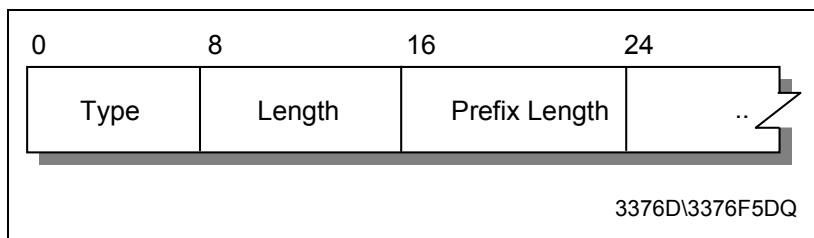


図 250 プレフィックス長の拡張子

ここで、

Type	19
Length	エージェント公示のルーター公示部分にあるルーター・アドレス項目の数。
Prefix length(s)	エージェント公示のルーター公示部分にある各ルーター・アドレスのネットワーク・プレフィックスを形成する先頭ビットの数。各プレフィックス長は、ルーター・アドレスがリストされる順序で別個のバイトによって示されます。

19.4 モバイル IP の登録プロセス

RFC 2002 は、モバイル IP を登録するための 2 つの異なる手順を定義しています。モバイル・ノードは外部エージェントを介して登録する、つまり外部エージェントが登録をモバイル・ノードのホーム・エージェントに中継するようにして登録するか、またはホーム・エージェントに直接登録することができます。以下の規則を使用して、これらの登録プロセスのどちらを使用するかを決めることができます。

- モバイル・ノードが気付アドレスを外部エージェントから取得した場合、そのノードは外部エージェントを介して登録する必要があります。
- モバイル・ノードが共存気付アドレスを使用しているものの、エージェント公示をこのサブネット上の外部エージェント (その公示で R ビット (登録が必要) が設定されている) から受け取っている場合、そのノードはエージェントを介して登録する必要があります。このメカニズムにより、

DHCP および共存気付アドレスが推奨されるアドレス割り振り方式である場合でも、外部サブネットで計算を行うことが可能になります。

- モバイル・ノードが共存気付アドレスを使用している場合、そのような公示を受け取っていない場合、そのノードは直接ホーム・エージェントに登録する必要があります。
- モバイル・ノードがホーム・ネットワークに戻った場合、そのノードは直接ホーム・エージェントに再登録する必要があります。

登録プロセスには、UDP データグラムである登録要求と登録応答メッセージの交換が含まれます。登録要求は、ポート 434 に送られます。登録は UDP ヘッダーと、それに続く図 251 に示されているフィールドから成ります。

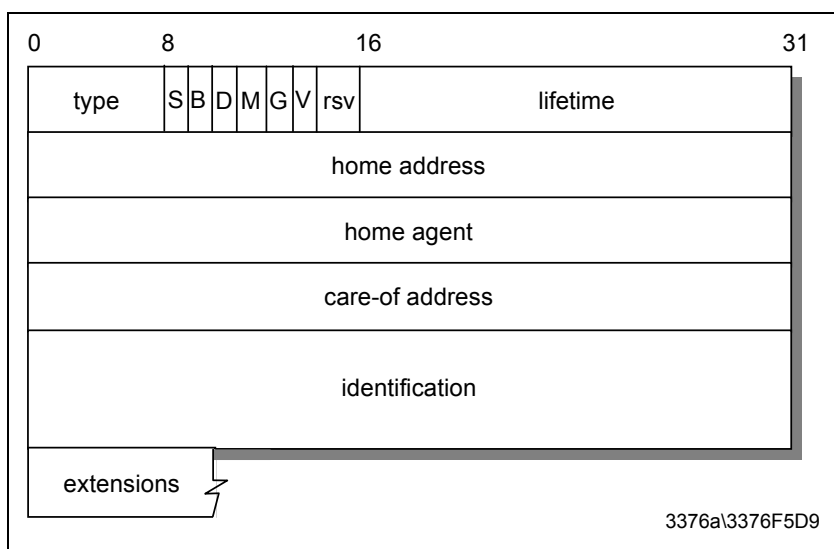


図 251 モバイル IP - 登録要求

ここで、

Type

1

S

同時結合。このビットが設定されている場合、ホーム・エージェントはこのノードに対する今までの結合および追加された新規の結合をすべて保存する必要があります。ホーム・エージェントはその後、ノードに対するデータグラムを複数の気付アドレスに転送します。この機能は特に、ワイヤレス・モバイル・ノードのためのものです。

B

ブロードキャスト・データグラム。このビットが設定されている場合、ホーム・エージェントはホーム・ネットワー

	ク上のブロードキャスト・データグラムをトンネルを開始してモバイル・ノードに送ります。
D	モバイル・ノードによるカプセル化解除。モバイル・ノードは共存気付アドレスを使用していて、送られてきたデータグラムをカプセル化解除します。
M	トンネルを介してモバイル・ノードに送るデータグラムに対して最小のカプセル化を使用します。
G	トンネルを介してモバイル・ノードに送るデータグラムに対して GRE カプセル化を使用します。
V	エージェントとモバイル・ノードとの間のリンクで Van Jacobson 圧縮を使用します。
rsv	予約済みビット。ゼロとして送信されます。
Lifetime	登録の有効期限が切れるまでの間に残されている秒数。値のゼロは、登録解除の要求を表します。0xffff は無限大を表します。
Home address	モバイル・ノードのホーム IP アドレス。
Home agent	モバイル・ノードのホーム・エージェントの IP アドレス。
Care-of address	トンネルの終端の IP アドレス。
Identification	モバイル・ノードによって作成され、登録要求を応答と一致させるために使用される、64 ビットの識別番号。
Extensions	拡張子の数は、すべて登録プロセスの認証に関連して定義されます。詳細については、RFC 2002 を参照してください。

モビリティ・エージェントは、登録要求に反応して登録応答を出します。それは登録要求のソース・ポートからコピーされた宛先ポートに送られます。登録応答のフォーマットを図 252 に示します。

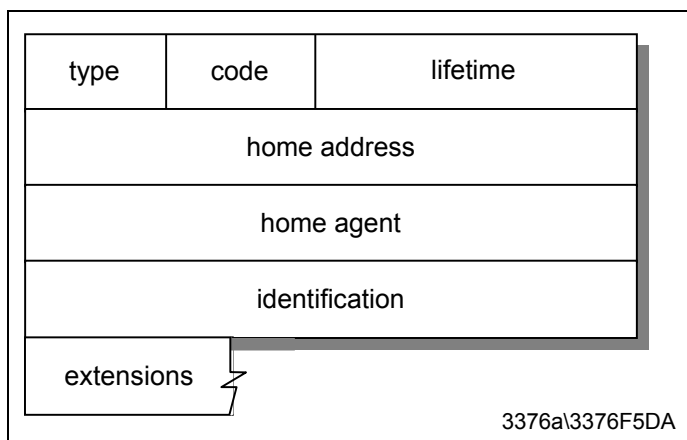


図 252 モバイル IP - 登録応答

ここで、

Type	3
Code	登録要求の結果を示します。
0	登録は受け入れられました。
1	登録は受け入れられましたが、同時結合はサポートされません。
64-88	登録は外部エージェントによって拒否されました。
128-136	登録はホーム・エージェントによって拒否されました。
Lifetime	登録の有効期限が切れるまでの間に残されている秒数。(コード・フィールドは 0 か 1 でなければなりません。)
Home address	モバイル・ノードのホーム IP アドレス。
Home agent	モバイル・ノードのホーム・エージェントの IP アドレス。
Identification	登録要求を応答と一致させるために使用される、64 ビットの識別番号。
Extensions	拡張子の数は、すべて登録プロセスの認証に関連して定義されます。

これらのメッセージの詳細については、RFC 2002 を参照してください。

19.5 トンネリング

ホーム・エージェントは、ホーム・ネットワークに到着するすべてのデータグラムの宛先 IP アドレスを検討します。そのアドレスが、現在ホームから離れているものとして登録されているモバイル・ノードのいずれかに一致する場合、ホーム・エージェントはデータグラムをそのモバイル・ノードの気付アドレスに (IP カプセル化の IP を使用して) トンネルを介して送ります。多くの場合、ホーム・エージェントはホーム・ネットワーク上のルーターとしても機能しています。そうであれば、多くの場合ホーム・エージェントは、現在ホームから離れているものとして登録されていないモバイル・ノードにアドレス指定されたデータグラムを受け取ります。この場合、ホーム・エージェントはモバイル・ノードがホームに存在すると想定して、データグラムをホーム・ネットワークに転送します。

外部エージェントは公示された気付アドレスからデータグラムを受け取ると、その内部宛先アドレスを登録済み訪問者のリストと比較します。一致するものが見つかった場合、外部エージェントはカプセル化解除されたデータグラムを適切なモバイル・ノードに転送します。一致するものが見つからない場合、そのデータグラムは廃棄されます。(外部エージェントがそのようなデータグラムを最初の IP ヘッダーに転送しないようにしてください。転送すると、ルーティング・ループが生じます。)

モバイル・ノードが共存気付アドレスを使用している場合、トンネルの終端はモバイル・ノード自体となります。モバイル・ノードには、ホーム・エージェントから受け取るデータグラムをカプセル化解除する役割があります。

19.6 ブロードキャスト・データグラム

ホーム・エージェントがブロードキャスト・データグラムを受け取るとき、モバイル・ノードがブロードキャストの転送を登録要求で明示的に要求しているの でなければ、そのデータグラムをモバイル・ノードに転送しないようにしてください。データグラムを転送する場合には、以下の方法の 1 つで行います。

- モバイル・ノードに共存気付アドレスがある場合、ホーム・エージェントは単にデータグラムをカプセル化して、トンネルを介して気付アドレスに直接送ります。
- モバイル・ノードに外部エージェントの気付アドレスがある場合、ホーム・エージェントはまずブロードキャストをカプセル化して、ノードのホーム・アドレスにアドレス指定したユニキャストにします。その後、このダイアグラムをカプセル化して、トンネルを介して気付アドレスに送ります。この方法により、外部エージェントがデータグラムをカプセル化解

除するときに、そのブロードキャストの転送先となる登録済みモバイル・ノードが判明します。

19.7 移動の検出

モバイル IP は、サイトからサイトへと定期的に移動してそのたびにラップトップを異なるサブネットに接続するユーザーだけではなく、本当に動的なモバイル・ユーザー（航空機から無線接続するユーザーなど）のためにも設計されています。モバイル・ノードがあるサブネットから別のサブネットに移動したとき、それを検出するための 2 つのメカニズムが定義されています。モバイル・ノードの移動が検出されたとき、そのノードを新しい外部ネットワークの気付アドレスに再登録する必要があります。移動を検出するための 2 つのメソッドは、以下のとおりです。

1. 外部エージェントはエージェント公示によって、ネットワーク上に存在することを常に公示しています。モバイル・ノードが外部エージェントからエージェント公示を受け取るとき、その公示の存続期間エージェント公示に基づくタイマーを開始します。モバイル・ノードが存続期間の満了する前に同じ外部エージェントから別の公示を受け取らない場合、モバイル・ノードはそのエージェントから切断されたと推定します。その間に、他の外部エージェントから公示を受け取る場合は、すぐにその新しいエージェントへの登録を試行することができます。他のエージェント公示を受け取らない場合は、登録するための新しい外部エージェントをエージェント誘導を使用して試行し、見つけるようにしてください。
2. モバイル・ノードは、新規に受け取ったエージェント公示が現行の気付アドレスと同じサブネット上にあるかどうかを検査します。ネットワーク・プレフィックスが異なる場合、モバイル・ノードはそれが移動したと推定します。現行の気付アドレスが満了するときに、モバイル・ノードは新規のエージェント公示を送信した外部エージェントに登録します。

19.7.1 ホームに戻る

モバイル・ノードがホーム・エージェントからエージェント公示を受け取るとき、ホーム・ネットワークに戻ったことが判明します。ホーム・エージェントに再登録する前に、モバイル・ノードはルーティング・テーブルをホーム・サブネット上での操作用に構成しなければなりません。

19.8 ARP に関する考慮事項

モバイル IP では、モバイル・ノードの移動に対処するために ARP に 2 つの拡張子が必要です。それらは、以下のとおりです。

プロキシ ARP ARP 要求に自分で応答できないかまたは応答する意思のないノードのために、他のノード送信した ARP 応答。

無償の ARP 1 つのノードによってローカル・ブロードキャスト・パケットとして送信される ARP パケットで、それを受け取るすべてのノードで ARP キャッシュ内の項目が更新されるようにするもの。

モバイル・ノードが外部ネットワーク上に存在するものとして登録されていると、そのホーム・エージェントはモバイル・ノードの MAC アドレスを問い合わせる ARP 要求に対して、プロキシ ARP を使用して応答します。ホーム・エージェントはその要求に対して、その MAC アドレスを示して応答します。

モバイル・ノードがホーム・ネットワークから移動して外部ネットワークに登録したとき、ホーム・エージェントは無償の ARP ブロードキャストを行って、ネットワーク上のすべてのローカル・ノードの ARP キャッシュを更新します。MAC アドレスとして、ホーム・エージェントの MAC アドレスが再び使用されます。

モバイル・ノードがホーム・ネットワークに戻るとき、以前に外部ネットワークに登録されていたので、すべてのローカル・ノードの ARP キャッシュを更新するために無償の ARP が、このたびはモバイル・ノードの本当の MAC アドレスと共に再び使用されます。

19.9 モバイル IP セキュリティに関する考慮事項

モバイル・コンピューティング環境は、盗聴に弱いワイヤレス・リンクを使用する場合には特に、セキュリティに関する多くの潜在的な危険にさらされています。ホーム・エージェントとモバイル・ノードの気付アドレスとの間のトンネルも、強力な認証メカニズムが登録プロセスの一部としてインプリメントされていなければ、代行受信されやすくなります。RFC 2002 は、認証プロトコルとしてキー付き MD5 のインプリメンテーションを指定して、完全なプライバシーが要求される環境では追加のメカニズム（暗号化など）の使用を推奨しています。

第 20 章 TCP/IP と他のプロトコルの統合

TCP/IP プロトコル・スイートの普及の増大にもかかわらず、現在多くのネットワークに TCP/IP とは別の複数のネットワーク・プロトコルがインプリメントされています。その例としてシステム・ネットワーク体系 (SNA)、拡張対等通信ネットワーク機能 (APPN)、NetBIOS があります。並列のネットワークの保守の費用を削減するためにそれらのネットワーク・プロトコルと TCP/IP の統合に使える技法が多数あります。接続性を確保するために通常は 1 つのネットワーク・インフラストラクチャーが選択されますが、他のプロトコルのデータもそのインフラストラクチャー上を伝送されます。IP が至るところに展開されていてそのプラットフォームが普及しているため、現在のネットワークではネットワーク・プロトコルとして IP が選択されるようになりました。

マルチプロトコル・トランスポート・ネットワーク (MPTN) ではネットワーク・プロトコル間のマッピングを試みっていますが、普及している統合メソッドの多くはデータのカプセル化を利用しています。これは、あるプロトコルのデータ単位を別のプロトコルのデータ単位の中に封入して、あるタイプのネットワーク・プロトコルのデータを別のネットワーク・プロトコル上で伝送できるようにするものです。この場合、カプセル化されたプロトコル・データは、その下にあるネットワーク・インフラストラクチャーにとってのペイロードのデータとして見ることができます。

20.1 Enterprise Extender

Enterprise Extender ネットワーク接続は、既存の High Performance Routing (HPR) オープン・テクノロジーの単純な拡張セットです。これにより UDP/IP パケットを使って HPR フレームの効率的な統合を行うことができます。HPR ネットワークにとって IP バックボーンは論理リンクです。IP ネットワークにとって SNA トラフィックは、ハードウェアやソフトウェアの変更なしに IP バックボーンに経路指定される UDP データグラムです。ゲートウェイと違ってそこではプロトコル変換は行われず、また一般的なトンネリング・メカニズムと違ってその統合は、トランスポート機能の追加というオーバーヘッドなしにルーティング層で行われます。先進テクノロジーによって、SNA LU タイプのものを使う SNA クライアントと同様に、SNA ベースのデータをアクセスする IP ベースのクライアント (たとえば 3270 エミュレーターや、IBM ホスト・オンデマンドなどのサービスを使用する Web ブラウザー) のサポートにも、イントラネット・インフラストラクチャーを効率的に使用することができます。

Enterprise Extender はネットワーク・プロトコル "edges" を使ってパケットのルーティングをシームレスに行います。そこでは時間を要するプロトコル

変換やトランスポート層の機能に関連したストア・アンド・フォワードを行う必要はありません。データ・リンク・スイッチ (DLSw) とは異なり、コネクションレス UDP を使用し、輻輳制御は終端システム間で行われるものとしているため、たとえば、ルーターの中に TCP 再送のためのバッファとタイマーや輻輳制御ロジックはありません。このような省略が行われているため、「エッジ」のルーターでは、必要な処理が軽くなりプロトコル変換のオーバーヘッドや多数の TCP 接続の保持がなくなって、パケットの転送というジョブを最適に行うことができます。したがってこのテクノロジーを使用すると、データ・センターのルーターで扱えるネットワークの規模とネットワーク・トラフィックの量を大きくできるため、費用に対する能力が高まります。

20.1.1 パフォーマンスとリカバリー

Enterprise Extender は、SNA ユーザーが期待していたトラフィック制御機能も多数備えています。SNA アプリケーションはサービス・クラス (COS) を使って、ネットワークに必要なサービスの特性（たとえば対話式かバッチか）を指定します。この指定には、必要なトラフィック優先順位も含まれます。ルーターの大半は何らかの形で優先順位付けキューイングをサポートしています。IBM ルーターではこの機能を帯域幅予約システム (BRS) と呼んでいます。Enterprise Extender は既存のこの機能を利用して、SNA 優先順位を UDP ポート番号にマッピングしています。ルーターはこれを使って適切な優先順位を容易に判別することができます。これらのポート番号のデフォルトは Internet Assigned Numbers Authority (IANA) に登録済みです。

ネットワーク障害は、テクノロジーが結合されているネットワークの場合でも、適切に処理することができます。データ・リンク・スイッチなどの他のテクノロジーと違って、Enterprise Extender は SNA および IP プロトコルのエッジで起きた障害を回避することができます。HPR は必要に応じて経路を切り替えることができ、Enterprise Extender にはそれらの経路に IP バックボーンを含めることができる柔軟性があるため、Enterprise Extender では SNA ネットワークで、SNA ネットワークの代替およびバックアップの経路として IP 接続を使用することができます。

20.2 データ・リンク・スイッチ

データ・リンク・スイッチ (DLSw) は IBM が 1993 年 3 月に発表したもので、RFC 1795 に文書化されています。その状態は Informational です。

20.2.1 概要

データ・リンク・スイッチは IBM SNA および IBM NetBIOS プロトコルのための転送メカニズムです。これは完全ルーティングは行いませんが、代わりにデータ・リンク層でのスイッチングと、インターネット上のトランスポートのための TCP/IP でのカプセル化を行います。他のベンダーのルーターも、

それが上記の RFC に準拠していれば加えることができます。また DLSw 接続は 2 つの終端ルーター間だけに存在するものであるため、DLSw 機能がないものは中間ルーターとして置くことができます。

20.2.2 機能の説明

DLSw は、マルチプロトコル・ルーターで SNA と NetBIOS をサポートするために開発されました。SNA と NetBIOS は基本的にはコネクション型プロトコルであるため、SNA と NetBIOS が LAN 上で使用するデータ・リンク制御手順は、IEEE 802.2 論理リンク制御 (LLC) タイプ 2 です。DLSw は WAN リンク上の SNA プロトコルも SDLC プロトコルによって適応します。

IEEE 802.2 LLC タイプ 2 は、ネットワーク伝送遅延が小さくて予測可能であるということ (たとえばローカル LAN) を前提に設計されています。そのため LLC では脱落フレームの検出に固定タイマーを使用します。広域の伝送路でブリッジングが (特に比較的低速で) 使用されると、ネットワーク遅延は大きくなり、また輻輳の状態に応じて大きく変わることがあります。遅延がタイムアウト値を超えると、LLC は再送を試みます。フレームが実際には脱落していなくて単に遅れているだけの場合、LLC タイプ 2 手順が混乱して、その結果リンクが最終的にダウンする可能性があります。

DLSw では、LLC タイプ 2 サービスを使用する場合の次のブリッジングの問題に対処しています。

- DLC のタイムアウト
- WAN 上の DLC 確認通知
- フローおよび輻輳の制御
- 検索パケットのブロードキャスト制御
- ソース経路のブリッジングのホップ・カウント制限

NetBIOS は LLC タイプ 1 を使うデータグラム・サービスも広く使用します。これに対して DLSw は、上記リストの中の最後の 2 つの問題に対処しています。DLSw とブリッジングの基本的な違いは、DLS でデータ・リンク制御が終了するのに対し、ブリッジングではそれがありません。図 253 は、LLC タイプ 2 サービスで動作する 2 つの終端システムの例でその違いを示したものです。

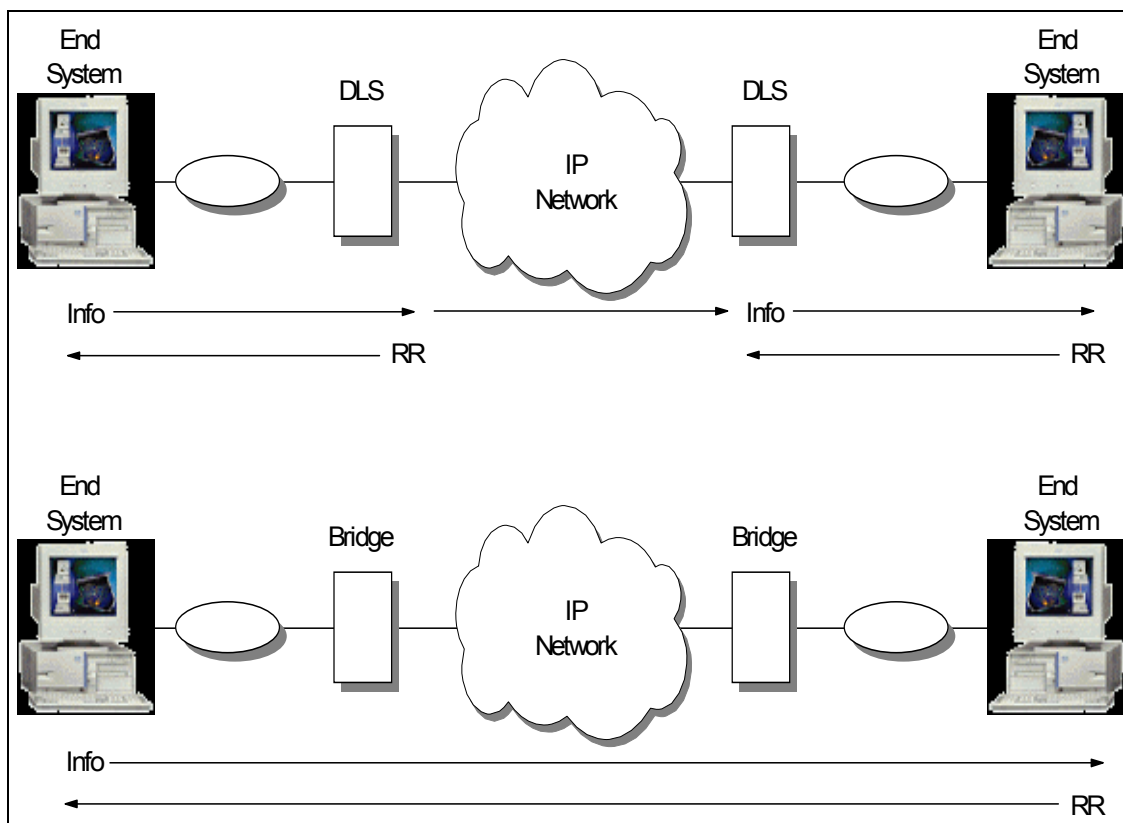


図 253 DLSw とブリッジングの比較

従来のブリッジングではデータ・リンク制御はエンドツーエンドです。DLSw は LLC タイプ 2 接続がスイッチで終了します。つまり LLC タイプ 2 接続は広域ネットワーク上を通らないことになります。DLS は LLC 接続を TCP 接続に多重化して他方の DLS に伝送します。そのため各終端の LLC 接続は相互に完全に独立しています。データ・リンク・スイッチが LLC 接続から受信したフレームを他の終端に送達するのは、そのデータ・リンク・スイッチの役割です。データ・リンク・スイッチ間では TCP を使用してフレームが確実に送達されるようにします。

この設計の結果、LLC タイムアウトはローカル LAN に限定されています。(つまり LLC タイムアウトは広域ネットワークまで及びません。) また、LLC タイプ 2 の確認通知 (RR) が WAN を通らないため、広域リンク上のトラフィックが減少します。SDLC リンクの場合、ポーリングとポーリング応答はローカルに行われ、WAN 上では行われません。検索フレームのブロードキャストは、ターゲット・システムのロケーションが見つかった後は、デー

タ・リンク・スイッチによって制御されます。そして最後に、スイッチはフローおよび輻輳の制御を終端システムに行わせることができます。

DLSw では LAN アドレスを使って SNA システム間の接続をセットアップします。SDLC 接続の装置は、LAN 接続の装置との通信が行えるように MAC アドレスで定義されます。NetBIOS システムの場合、DLSw はデータグラムの転送と NetBIOS セッションのための接続のセットアップに NetBIOS 名を使用します。SNA システムは伝送路の確立のために、TEST (または XID の場合もあり) フレームをヌル (x'00') SAP に送信します。NetBIOS システムには名前照会と名前認識フレームによるアドレス解決手順があり、これを使ってエンドツーエンド伝送路の確立が行われます。

DLSw はマルチプロトコル・ルーターにインプリメントできるため、ブリッジングとスイッチングの両方が使用可能な状態が考えられます。SNA フレームは、そのリンク SAP で識別することができます。SNA の通常の SAP 値は、x'04'、x'08'、および x'0C' です。NetBIOS は、リンク SAP 値として常に x'F0' を使用します。

詳細については、RFC 1795 を参照してください。

20.3 マルチプロトコル・トランスポート・ネットワーク (MPTN)

マルチプロトコル・トランスポート・ネットワーク (MPTN) は IBM によって開発されたオープン・アーキテクチャーであり、次のことを目的としています。

- 混合プロトコル・ネットワーク：さまざまなネットワーク・プロトコルの中の 1 つのネットワーク・プロトコルに関連したアプリケーションの実行。
- ネットワーク連結：さまざまなプロトコルのネットワーク間でのゲートウェイを介した接続マッチング・アプリケーション。

20.3.1 混合プロトコル・ネットワークの要件

一般的なネットワークの発達、特にローカル・エリア・ネットワークの発達に伴って、カスタマーの大規模なネットワークの多くが、さまざまなプロトコルが動作する別々のネットワークの連合状態になっています。この混成状態になったのには多くの理由があります。たとえば次のことが挙げられます。

- カスタマーの関心が特定のネットワーク体系を選択することからシフトして、特定のネットワーク・プロトコルの必要性にかかわりなくビジネス上の問題のソリューションを見つけることに関心が移った。

- 企業間の情報交換が必要になった。たとえば、直接製造、発注、請求の各システム。
- 企業合併のために相互接続が必要になった。

20.3.2 MPTN のアーキテクチャー

ネットワーク・プロトコルは一般に次の 3 つのタイプの機能を備えています。

- トランスポート
- ネーミングとアドレッシング
- 上位機能

トランスポート機能は、2 つのパートナーが、接続によってまたはデータグラムによるコネクションレスの方法で通信を行うための基本ファシリティーを提供するものです。ネーミングおよびアドレッシングの規則でエンティティーの呼び方とその検出方法が定義されます。上位機能には、ユーザーへの接続の割り振りとその使用の制御が含まれます。すべてのネットワーク・プロトコルが上位機能をサポートしているわけではありません。SNA と OSI ではサポートしていますが、TCP/IP と NetBIOS ではサポートしていません。

MPTN ではトランスポート機能を、上位機能およびプロトコルのユーザーから見えるアドレス・フォーマットから分離しています。その目的は、対応するアドレス構造を使用する上位プロトコルが、どのトランスポート機能でも実行されるようにするためです。トランスポートと上位の間での機能の分割が選択されたのは、プロトコル間で合理的にマップできる共通機能があるのはトランスポート・レベルが最上位のレベルだからです。他のレベルでは、実用的な分割を行うにはサービスの数が大きすぎるか小さすぎます。

20.3.3 MPTN の方法論

MPTN アーキテクチャーは、複数のネットワーク・プロトコル間で接続を連結する一連のトランスポート・サービスに対して新しい正規インターフェースを定義することにより、上記の要件を解決しています。アプリケーションが特定のトランスポート・サービスに合わせて作成されると、そのアプリケーションがその特定のトランスポート・サービスの前提事項になるように作成されることがあります。したがってそのアプリケーションは、それが使用するサービスではトランスポート固有のものになります。たとえば NetBEUI インターフェースに合わせて作成されたアプリケーションは、メッセージのブロードキャストを要求するかもしれません。特定のサービスがその基礎のトランスポート・ネットワークでもともとサポートされていない環境では、MPTN は補償を行います。本質的に MPTN ではアプリケーションは自由であるため、アプリケーションはさまざまなトランスポート・ネットワーク上で動作することができます。

別の観点から考慮すると、(OSI モデルでの) セッション層より上の機能は、トランスポート・サービスのユーザーつまりトランスポート・ユーザーです。それらのサービスはトランスポート層より下の機能で提供されます。MPTN ではトランスポート層プロトコル境界 (TLPB) と呼ぶ境界インターフェースを定義しており、そこでトランスポート・ユーザーとトランスポート・プロバイダーの区別を明確にしています。これを図 254 に図で示します。

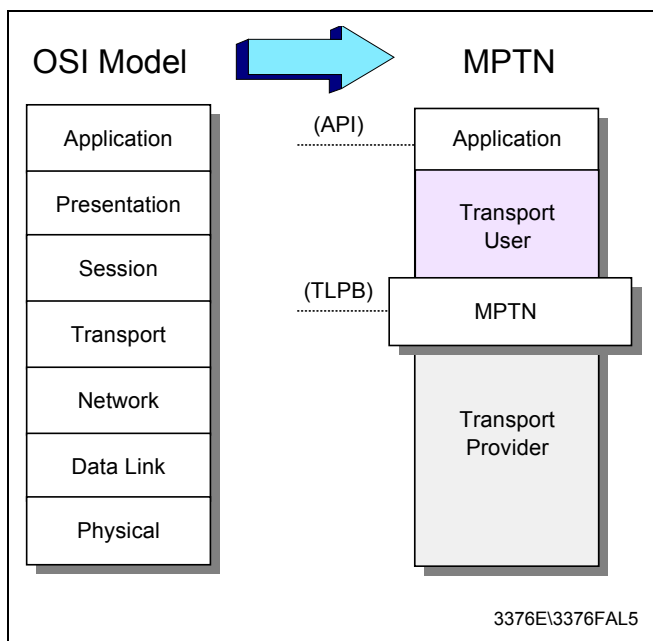


図 254 トランスポート層プロトコル境界

20.3.4 MPTN の主要コンポーネント

MPTN の機能は次の 4 つのタイプのノードに分けることができます。

- MPTN アクセス・ノード

MPTN アクセス・ノードにはトランスポート層プロトコル境界 (TLPB) が含まれます。これは上位プロトコルまたは特定のトランスポート・プロトコルのために作成されたアプリケーション・インターフェースを外部的には特に変更することなしに、別のプロトコルでそれらをトランスポートできるようにするためのセマンティック・インターフェースを備えます。このノードでは、アプリケーション・プログラムをその基礎のトランスポート・ネットワークから独立して実行することができ、また基礎のさまざまなトランスポート・ネットワーク上でアプリケーション・プログラムを実行することができます。

MPTN アクセス・ノードのアプリケーションは一般に、既存のアプリケーション・プログラミング・インターフェース (API) に合わせて作成されます。API ではネイティブ通信プロトコルの機能を使用します。トランスポート・レベルの MPTN 機能がアクセスされると、その機能にアクセスするために、ネイティブ通信プロトコルではなく API が変換されます。このときアプリケーション・プログラムとは同じインターフェースのままです。たとえば NetBEUI アプリケーション・インターフェースは NetBIOS 通信プロトコルを使用するように作成されます。トランスポート層より下の別のプロトコル・スタックを使用するために、NetBEUI API は MPTN 機能呼び出すように作成されます。これが行われた後は、この MPTN アクセス・ノード上の NetBEUI を使用するすべてのプログラムが、たとえば SNA、TCP/IP、OSI や他のプロトコルで、MPTN ネットワーク内の別の NetBEUI アプリケーションと通信を行うことができます。これらすべては、オリジナルのアプリケーション・プログラムを変更せずに実現可能です。

- MPTN アドレス・マッピング・サーバー・ノード

アドレス・マッピング・サーバー・ノードは特殊機能を持つ MPTN ノードです。ここでは同じトランスポート・プロバイダー・ネットワークに接続されている他の MPTN ノード (アクセス、ゲートウェイ) のアドレス・マッピング・クライアント・コンポーネントに対して汎用アドレス・マッピング・サービスを提供します。

- MPTN マルチキャスト・サーバー・ノード

マルチキャスト・サーバー・ノードは、マルチキャスト・サーバーという特殊機能を持つ MPTN ノードです。ここでは設計の結果としてもともとこのサービスをサポートしていないネットワークでの、データグラムのマルチキャストおよびブロードキャスト配布を管理します。たとえば NetBIOS トランスポート・ネットワークではデータグラムのマルチキャスト配布をサポートするように設計されていますが、SNA ネットワークではサポートされていません。マルチキャスト・サーバーはアドレス・マッピング・サーバーと連携して動作して、トランスポート・ネットワークで提供されるサービスの中にはこの機能が存在しないところに対してマルチキャスト・サービスを提供します。

- MPTN ゲートウェイ・ノード

MPTN ゲートウェイは 2 つのトランスポート・ネットワークを接続して、その両方に 1 つ以上のトランスポート・ユーザー・プロトコルで、エンドツーエンド・サービスを提供します。ゲートウェイには、(サポートされるトランスポート・ユーザー・プロトコルに関して) 非ネイティブ対非ネイティブのタイプと、(サポートされるトランスポート・ユーザー・プロトコルに関して) ネイティブ対非ネイティブのタイプの 2 つのタイプがあります。

非ネイティブ対ネイティブの場合、ゲートウェイ・ノードに接続されるトランスポート・プロバイダーの1つは、トランスポート・ユーザーにとってはネイティブです。このタイプのゲートウェイでは、MPTN 機能がないエンド・ノードへのアクセスが可能です。一方がネイティブのとき、(非ネイティブ・サイドの) MPTN プロトコルはゲートウェイで終了します (図 255 を参照)。

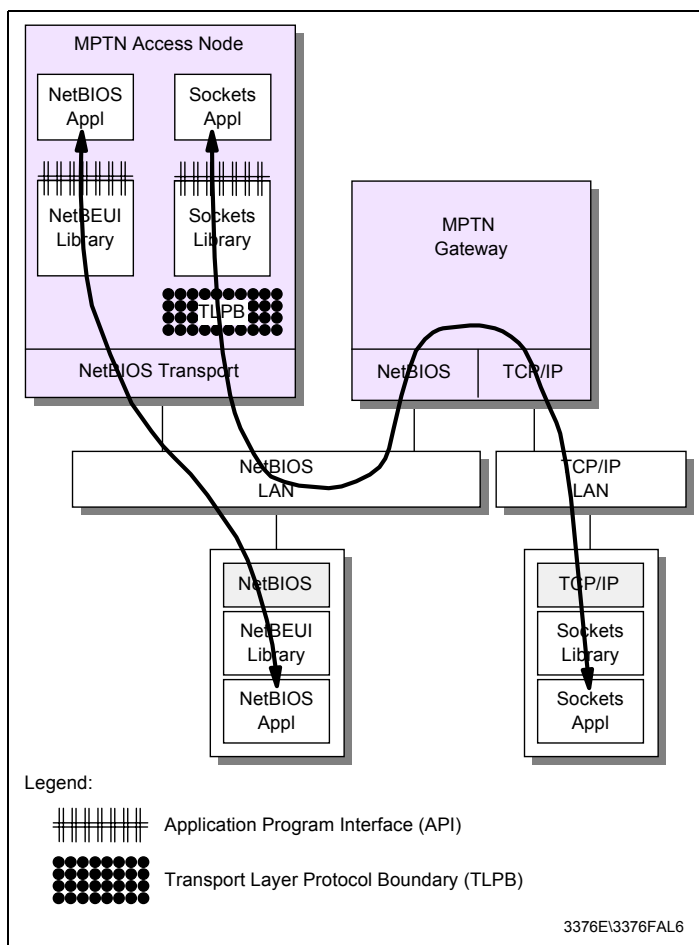


図 255 MPTN ネットワーク - 非ネイティブ対ネイティブのゲートウェイ

20.4 TCP/IP での NetBIOS

しばらくの間 NetBIOS サービスが LAN 環境のパーソナル・コンピュータのネットワーキングの主流メカニズムでした。NetBIOS はベンダーに依存し

ないソフトウェア・インターフェース (API) であり、プロトコルではありません。公式の NetBIOS 仕様はありませんが、実際には IBM 資料「*LAN Technical Reference: 802.2 and NetBIOS APIs* (SC30-3587)」に記述されている NetBIOS バージョンがリファレンスとして使用されています。

NetBIOS サービスには、次に示すように、広域ネットワーク環境によってはそこでの使用が制限される独特の特性がいくつかあります。

- NetBIOS ワークステーションは NetBIOS 名を使って相互にアドレスを指定します。NetBIOS はブロードキャスト手法を使ってネットワーク上の NetBIOS 名の登録と検出を行います。
- NetBIOS はフラット・ネーム・スペースを使用します。これには Domain Name System との関連は含まれません。
- NetBIOS は一般に 802.2 インターフェースを使用します。したがって経路指定できません。

これらの制約を解消するための 1 つのソリューションが、RFC 1001 – Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods と、RFC 1002 – Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications に記述されています。

タイトルが示すように、これらの RFC は NetBIOS サービスを Transmission Control Protocol (TCP) および User Datagram Protocol (UDP) の上位にインプリメントするメソッドを定義しています。

RFC 1001 および 1002 はカプセル化の手法は定義していません。そこでは NetBIOS サービスの UDP/IP と TCP/IP へのマッピングが定義されています。たとえば NetBIOS セッションが確立されると、NetBIOS セッション・データの送信に、TCP 接続上でソケット送信のコマンドが使用されます。

NetBIOS インターフェース用に作成されたネットワーク・アプリケーションから見れば、NetBIOS サービスについての変更はありません。アプリケーションは基礎のプロトコルでの変更は意識しません。

TCP と UDP の上位で実行される NetBIOS サービスは、論理的には図 256 のように表すことができます。

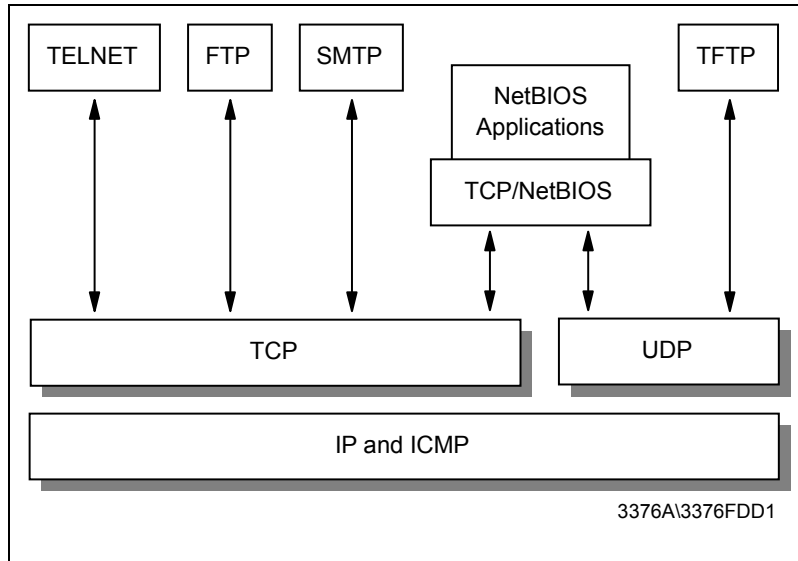


図 256 TCP/IP と NetBIOS

RFC 1001 は TCP/IP 環境での NetBIOS の操作を容易にする新しい概念をいくつか紹介しています。

- NetBIOS スコープ

NetBIOS スコープは NetBIOS 名が通用するコンピューターのグループです。NetBIOS の標準構成では、このグループ内のすべてのコンピューターが、ブロードキャスト手法を使って相互に接触することができます。NetBIOS over TCP/IP インプリメンテーションでは、このグループが相互に通信を行える代替メソッドを備えていなければなりません (特にそれらがサブネット間に広がっている場合)。インターネットワークで複数の非交差 NetBIOS スコープをサポートすることができます。各 NetBIOS スコープにはスコープ ID がありますが、これは Domain Name System と互換性のあるフォーマットの文字ストリングです。

- NetBIOS エンド・ノード

NetBIOS over TCP/IP を使用するコンピューターを NetBIOS エンド・ノードと呼びます。RFC 1001 ではエンド・ノードとして次の 3 つのタイプを定義しています。

- ブロードキャスト (B) ノード: ローカル・ブロードキャスト UDP データグラムを使って接続を確立してデータグラムを配布する。ブロードキャスト・データグラムは通常はルーターを通ることができないため、B ノード (RFC 1001 での定義) は単一のネットワーク・セグメント内でのみ通信を確立できます。

- Point-to-Point (P) ノード：NetBIOS ネーム・サーバー (NBNS) を使って他のノードとの接続を確立し、データグラムの配布には NetBIOS データグラム・ディストリビューター (NBDD) を使用します。P ノードではブロードキャスト UDP パケットの生成も受信も行わず、送信ユニキャスト・パケットだけを使用します。
- 混合 (M) ノード：混合ノードは B ノードと P ノードの組み合わせです。ここではユニキャスト (NBNS への) とブロードキャストの両方のパケットを使用します。

次の 4 番目のノード・タイプのハイブリッド (H) ノードは、RFC では定義されていませんが、これも広くインプリメントされています。

- ハイブリッド (H) ノード：ハイブリッド・ノードも B ノードと P ノードの組み合わせです。ただし H ノードは、最初から常にユニキャスト・パケットを使って NetBIOS ネーム・サーバーに接触を試み、それが失敗イベントになったときだけブロードキャスト・パケットを使用します。H ノードでは、不要なブロードキャスト・パケットの生成が制限されるという点で M ノードにない利点があります。

- NetBIOS ネーム・サーバー (NBNS)

NBNS は、P、M、または H の各ノードの初期化時に、その NetBIOS 名 / IP アドレスを登録します。ノードが NetBIOS 名で別のノードに接触するとき、ノードは常に NBNS を使ってそのノードの対応する IP アドレスを取得します。NetBIOS 名はそのスコープ ID と結合されて、エンコードされた形式 (有効なドメイン・システム名) で NBNS に保管されます。

- NetBIOS データグラム・ディストリビューター (NBDD)

- NBDD は、NetBIOS データグラム配布サービスをインターネット環境まで拡張するものです。エンド・ノードがデータグラムをブロードキャストする必要があるとき、ノードはブロードキャストのスコープを含むユニキャスト・データグラムを NBDD に送信します。NBDD は、指定された NetBIOS スコープ内の各エンド・ノードに NetBIOS データグラムを送信します。このとき、関係のある IP アドレスは NBNS のサービスを使って取得します。(実際には NBNS と NBDD が同じ装置にインプリメントされることがよくあります。)

20.4.1 NetBIOS ネーム・サーバー (NBNS) のインプリメンテーション

NetBIOS ネーム・サーバーは、P、H、および M ノード・ワークステーションの初期化時にそれらについての NetBIOS から IP へのマッピングを動的に登録して、それらのマッピングを求める要求に応答します。NBNS は管理が最も簡単な NetBIOS over IP ソリューションです。ここではクライアント・ワークステーションや DNS 上でファイルを手動で構成する必要がありません。

20.4.1.1 Microsoft Windows Internet Name Service (WINS)

Windows Internet Name Service (WINS) は NBNS の Microsoft でのインプリメンテーションです。WINS は、ネイティブ NetBIOS および NetBIOS over TCP/IP のインプリメンテーションによって、Microsoft 製品でのクライアントのみサポートします。

それぞれの Microsoft クライアントでは、プライマリー WINS サーバーの IP アドレスの構成と、オプションとしてセカンダリー WINS サーバーの IP アドレスの構成が必要です。クライアントは始動時に必ずプライマリー WINS サーバーに対して自身の NetBIOS 名と IP アドレスの登録を試みます。この登録はサービスまたはアプリケーション (たとえばワークステーションやメッセージャー) が開始されたときに行われ、プライマリー WINS サーバーに直接送信されます。その名前が別のクライアントにまだ登録されていない場合、サーバーは登録した NetBIOS 名とその名前の存続時間 (TTL) の詳細を含むメッセージで応答します。クライアントがプライマリー・サーバーへのその名前の登録を 3 回試みて失敗した場合、クライアントはセカンダリー・サーバーへのその名前の登録を試みます。セカンダリー・サーバーも応答に失敗した場合、クライアントは元に戻ってその名前を登録するためにブロードキャストを行います。

名前の登録は一時的に行われるものであるため、クライアントは登録名を契約したものとして維持しなければなりません。クライアントは TTL の 8 分の 1 のときに、プライマリー WINS サーバーに対して名前の登録のリフレッシュを試みます。クライアントがサーバーから応答を受信しない場合、クライアントは TTL の半分まで 2 分ごとに登録のリフレッシュの試みを続行します。この時点で、手順を繰り返しますが、セカンダリー WINS サーバーを使用します。

WINS が使用可能な場合、クライアントは名前の登録のときは H ノード・クライアントとして動作します。解決のときは H ノードが多少変更されたものになります。ネーム解決のときの WINS クライアントでのシーケンスは次のようになります。

1. それがローカル・マシン名かどうかを調べる。
2. ローカル・キャッシュを検査する。(解決済みの名前はキャッシュに 10 分間置かれます。)
3. プライマリー WINS サーバーの使用を試みる。(3 回の試行の後にもプライマリー・サーバーから応答がない場合はセカンダリー・サーバーを使用する。)
4. 名前照会のブロードキャストを試みる。
5. LMHOSTS ファイルを検査する (コンピューターが LMHOSTS を使用するように構成されている場合)。
6. HOSTS ファイルで試行する。
7. DNS で試行する。

NetBIOS over TCP/IP の詳細については、RFC 1001 および RFC 1002 を参照してください。

第 21 章 TCP/IP セキュリティー

この章では、TCP/IP ネットワークに関するセキュリティー問題を説明し、機密漏れや問題を、発生前に防ぐためのソリューションの概要を提供します。一般のネットワーク・セキュリティーの分野および特定の TCP/IP セキュリティーの分野は、範囲が広すぎて、本書に含まれるすべての方法でも扱いきれないため、この章では、最も一般的な機密漏れと、それを打ち消す方策に焦点を当てます。すべてではないにしても、多くのセキュリティー・ソリューションは暗号アルゴリズムに基づいているので、この章全体で提示される概念をより一層理解するために、暗号アルゴリズムについても簡単に概要を示します。

21.1 機密漏れとソリューション

このセクションでは、コンピューター・セキュリティーに対する最も一般的な攻撃をいくつか概説するとともに、このような機密漏れに対する実行可能なソリューションを示し、それについての実際の実装をリストします。

21.1.1 セキュリティーに対する一般的な攻撃

幾千年にもわたり、人類は、財宝や資産の保管場所の門を見張ってきました。見張りに失敗すると、通常は、盗難にあってしまったり、社会的な制裁を受けたり、あるいは処刑されることもありました。今でこそこれほど劇的なことにはなりません、非常に悪い結果になることは現在でもあります。現代では、IT 管理者は、内部と外部の両方からの侵入者および破壊活動家から自分たちの通信ネットワークを保護することが、それと同じほど重要であることに気が付いています。誇大妄想に駆られていなくとも、以下が現実となる十分な理由は見つかるものです。

- ・ 回線の盗聴：クリア・テキスト・データおよびパスワードへアクセスする目的で。
- ・ 偽名の使用：データに無許可アクセスする、または無許可の E メール、オーダーなどを作成する目的で。
- ・ サービス妨害：ネットワーク・リソースを機能不全にする目的で。
- ・ メッセージの再生：情報にアクセスし、転送中にその情報を変更する目的で。
- ・ パスワードの推測：通常は拒否される情報やサービスにアクセスする目的で（ディクショナリー攻撃）。
- ・ 鍵の推測：暗号化されたデータおよびパスワードにアクセスする目的で（力づく攻撃）。
- ・ ウィルス：データを破棄する目的で。

このような攻撃は、TCP/IP ネットワークだけに固有の攻撃ではありませんが、使われている最も有力なプロトコルである TCP/IP をベースにネットワークを構築しようとするユーザーすべてに対する潜在的な脅威であると見なす必要があります。TCP/IP はオープン・プロトコルであるため、ハッカーは前述のような攻撃をすることにより、簡単に標的のカモを見つけます。

21.1.2 ネットワーク・セキュリティ問題のソリューション

ネットワーク所有者は、侵入者がネットワークに侵入する方法を探するときと同じ熱意をもって自分を保護しようと努める必要があります。そのため、攻撃（特に、前述の攻撃）からネットワークを効果的に守るためのソリューションをいくつか用意しました。このようなソリューションはどれも、1 つの（または非常に限られた数の）セキュリティ問題を解決するに過ぎないことに注意してください。したがって、特定レベルの安全およびセキュリティを確保するには、そのようなソリューションをいくつか組み合わせることを考える必要があります。ソリューションには、以下のものがあります。

- ・ 暗号化：データとパスワードを保護する目的で。
- ・ デジタル署名および証明書による認証：ネットワークを介してデータを送信しているユーザーを確認する目的で。
- ・ 許可：不適切なアクセスを防ぐ目的で。
- ・ 整合性検査およびメッセージ確認コード：不適切なメッセージの変更から保護する目的で。
- ・ 否認防止：アクションを実行したユーザーによってそのアクションが拒否されないようにする目的で。
- ・ 一回限りパスワードおよび両方向乱数ハンドシェイク：会話のパーティーを相互に認証する目的で。
- ・ 頻繁な鍵のリフレッシュ、鍵の強化、および将来の鍵の派生防止：鍵の解読から保護する目的で（暗号解読）。
- ・ アドレスの隠蔽：サービス妨害攻撃から保護する目的で。

表 25 では、一般的な問題と機密漏れが、上でリストされているソリューションと対照されています。

表 25 機密漏れと保護策

問題 / 機密漏れ	救済策
回線の盗聴者がメッセージを読み取るとをどのように防ぎますか？	メッセージを暗号化します。通常は、共有秘密鍵を使います（秘密鍵には、公開 / 秘密鍵に比べて、非常に大きなパフォーマンス上の利点があります）。

問題 / 機密漏れ	救済策
鍵はどのような方法で安全に配布しますか？	別の暗号化技法を使います。通常は、公開 / 秘密鍵です。
鍵の不整合をどのように防ぎ、現行の鍵を解読することで将来の鍵を推測することからどのように保護しますか？	鍵を頻繁にリフレッシュし、以前の鍵から新しい鍵を派生させないようにします (Perfect Forward Secrecy (PFS) を使います)。
詐称者によるメッセージの再送をどのように防ぎますか (再生攻撃)？	シーケンス番号を使います (タイム・スタンプは、通常は、セキュリティの目的では信頼できません)。
メッセージが転送中に変更されていないことをどのように確認しますか？	メッセージ・ダイジェストを使います (ハッシュまたは一方方向機能)。
メッセージ・ダイジェストで暗号が漏えいしていないことをどのように確認しますか？	秘密鍵を使ってメッセージ・ダイジェストを暗号化することによるデジタル署名を使います (起点の認証、否認防止)。
メッセージと署名が適切なパートナーからのものであることをどのように確認しますか？	暗号化された乱数を含む両方向ハンドシェークを使います (相互認証)。
ハンドシェークが正しいパートナーと交換されたことをどのように確認しますか (中継ユーザー攻撃)？	デジタル証明書を使います (永続 ID に公開鍵をバインドします)。
別な方法で適切に認証されたユーザーによってサービスが不適切に使われることをどのように防ぎますか？	マルチ・レイヤー・アクセス制御モデルを使います。
ウィルスからどのように保護しますか？	外部リソースへのアクセスを制限します。外部データへ接触する各サーバーおよびワークステーションでアンチウィルス・ソフトウェアを実行し、そのソフトウェアを頻繁に更新します。
望まないまたは悪意のあるメッセージからどのように保護しますか (サービス妨害攻撃)？	フィルター、ファイアウォール、プロキシ、パケット認証、隠蔽内部アドレス、および名前構造などを使い、内部ネットワークへのアクセスを制限します。

一般には、ネットワークを外部からしっかり保護しますが、ほとんどの攻撃は企業ネットワーク内から開始されるので、内部に対してもしっかり監視するようにします。

21.1.3 セキュリティー・ソリューションの実装

以下のプロトコルおよびシステムは、コンピューター・ネットワークにさまざまな程度のセキュリティー・サービスを提供するときに、一般的に使われるものです。これらについては、この章の残りの部分で詳しく説明します。

- IP フィルター
- ネットワーク・アドレス変換 (NAT)
- IP Security Architecture (IPsec)
- SOCKS
- Secure Shell (SSH)
- Secure Sockets Layer (SSL)
- アプリケーション・プロキシ
- ファイアウォール
- Kerberos およびその他の認証システム (AAA サーバー)
- Secure Electronic Transactions (SET)

図 257 では、これらのセキュリティー・ソリューションが、TCP/IP レイヤー内のどの位置に存在するかを示しています。

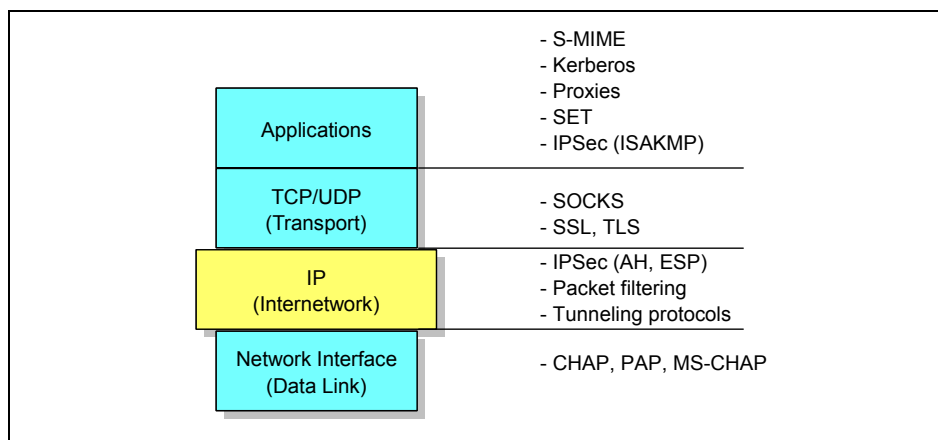


図 257 TCP/IP レイヤーでのセキュリティー・ソリューション

表 26 は、前述のいくつかのセキュリティー・ソリューションの特性を要約し、相互に比較しています。この表は、セキュリティー戦略を考案して、ど

のソリューションの組み合わせで希望するレベルの保護を実現できるかを判断すべきユーザーに役立ちます。

表 26 セキュリティー・ソリューションの実装 - 比較

	アクセス 制御	暗号化	認証	整合性検査	Perfect Forward Security	アドレス 隠蔽	セッション・モ ニター
IP フィル ター	y	n	n	n	n	n	n
NAT	y	n	n	n	n	y	y (接続)
IPsec	y	y (パケッ ト)	y (パケッ ト)	y (パケッ ト)	y	y	n
SOCKS	y	n	y (クライ アント / ユー ザー)	n	n	y	y (接続)
SSL	y	y (デー タ)	y (システ ム / ユー ザー)	y		n	y
アプリケー ション・プ ロキシー	y	通常は no	y (ユー ザー)	y	通常は no	y	y (接続および データ)
AAA サー バー	y (ユー ザー)	n	y (ユー ザー)	n	n	n	n

セキュリティ・ソリューションの全体は、ほとんどの場合、リストされているオプションの組み合わせで提供されるものに過ぎません。ユーザー固有のセキュリティ要件をセキュリティ・ポリシーに指定し、たとえば、ファイアウォールを使って実行する必要があります。

21.1.4 ネットワーク・セキュリティ・ポリシー

組織のセキュリティ・ポリシー全体は、セキュリティおよびビジネス・ニーズの分析に基づいて決定する必要があります。ファイアウォールはネットワーク・セキュリティのみに関連付けられるので、セキュリティ・ポリシー全体を適切に定義しない限り、ファイアウォールにはほとんど価値がありません。

ネットワーク・セキュリティ・ポリシーは、明示的に許可または拒否されるこれらのサービスについて、これらのサービスの使用法、およびこれらの規則の例外を定義します。ネットワーク・セキュリティ・ポリシーの各規則は、ファイアウォールまたはリモート・アクセス・サーバー (RAS) (また

はその両方)に実装する必要があります。一般に、ファイアウォールは、以下の方法のいずれかを用います。

特別に許可されていないものをすべてを拒否する

この方法では、許可されたサービスおよびアプリケーションを除いて、2つのネットワーク間のトラフィックがすべてブロックされます。したがって、希望するそれぞれのサービスおよびアプリケーションは、1つずつ実装しなければなりません。ファイアウォールのホールとなる可能性のあるサービスやアプリケーションは許可されません。これは最も安全な方法で、管理者によって明示的に許可されていない限り、サービスおよびアプリケーションは拒否されます。一方で、ユーザーの側からすると、より制限的であり便利でない可能性があります。

特別に拒否されていないものをすべてを許可する

この方法では、拒否されたサービスおよびアプリケーションを除いて、2つのネットワーク間のトラフィックがすべて許可されます。したがって、信頼されていないか有害だと思われるそれぞれのサービスやアプリケーションを、1つ1つ拒否する必要があります。ユーザーにとっては柔軟で便利な方法ですが、重大なセキュリティ問題が引き起こされる可能性もあります。

リモート・アクセス・サーバーでは、ユーザーの認証を行うとともに、特定のユーザーを企業イントラネット内の特定システムおよびネットワークの両方または一方に限定する(認証する)ことが理想的です。リモート・アクセス・サーバーは、ユーザーがローミング(複数のリモート・ロケーションから接続できる)なのか、定常(1つのリモート・ロケーションからのみ接続できる)なのかを判別し、特定ユーザーが適切に認証されたら、サーバーがそのユーザーのコールバックを使うかどうかを判別する必要があります。

一般に、匿名アクセスは、よくても非武装地帯(DMZ、719ページの21.3.6.4『選別サブネット・ファイアウォール(非武装地帯)』を参照)のサーバーに認可すべきです。企業イントラネット内のサービスではすべて、最低でもパスワード認証と適切なアクセス制御が必要です。外部からの直接アクセスは、必ず認証されてアカウントに入れられなければなりません。

21.2 暗号化についての簡単な概要

この章の目的は、用語を紹介し、TCP/IP セキュリティ実装に関連した主要な暗号の概念を簡単に説明することです。ここに示されている情報は、表面を扱っているだけです。問題によっては、未解決のままであったり、まったく言及されません。

21.2.1 用語

非常に基本的ないくつかの概念の定義から始めることにします。

暗号化

簡単に言うと、暗号化は、ユーザーのデータおよび通信の機密を保つ技術です。この目的を達成するには、**暗号化**、**復号**および**認証**のような技法が使われます。この分野での最近の進歩のために、暗号化の先駆者は影が薄くなっています。リバースするのが難しい方法に基づくデータの変換を構成する各手順は、暗号化と見なすことができます。強力な暗号化の主な要因は、リバース・エンジニアリングの難しさです。驚かれるかもしれませんが、パスワードでスクランブルをかけたワード・プロセッサ文書や圧縮アーカイブのような簡単な方法では、通常の PC を使うハッカーにほんの数分で破られます。強力な暗号化とは、適切な鍵を知らずにクリア・テキスト・メッセージを取り出すときに必要とされる計算方法によって、その取り出しを実行不可能にすることを意味します。ここでは、実行不可能というのは、次のようなことです。つまり、世界中のすべてのコンピューターに問題解決を割り当てたとして、それらのコンピューターは、解決できるまで、幾万年も計算しつづけなければならないということです。この取り出しの処理を、**暗号解読**といいます。試行された暗号解読のことを、**攻撃**といいます。

暗号化および復号 - 暗号アルゴリズム

暗号化とは、クリア・テキスト・メッセージを読み取り不能な形式に変換して、その意味を隠すことです。その逆の変換では、元のクリア・テキストが取り出されますが、これを復号といいます。暗号化および復号に使われる数学関数のことを、**暗号アルゴリズム**または**暗号**といいます。

暗号のセキュリティは、その機能を秘密にしておくことに全面的に基づいています。この場合は、**制限された暗号**になります。制限された暗号には、多くの欠点もあります。特定のアルゴリズムが多くユーザーに使われるときには、それを秘密にしておくことは非常に難しいことです。これが市販の製品に組み込まれている場合、時間とお金さえ掛ければ、すぐにリバース・エンジニアリングされます。このような理由のため、現在使われているアルゴリズムは、**鍵付き**です。すなわち、暗号化と復号で、**鍵**というパラメーターを利用します。鍵は、キー・スペースという、指定可能な値のセットから選択できます。キー・スペースは通常は大きなもので、さらに大きくなればそれだけ良くなります。このようなアルゴリズムのセキュリティは、内部の秘密ではなく、全面的に鍵に依存しています。実際、アルゴリズムそのものは普通は公用で、弱いと思われる部分については、広範囲に分析されています。

注：

新しい、知られていない、あるいは公開されていないアルゴリズムを信用しないでください。鍵付き暗号の原理は、図 258 に示されています。

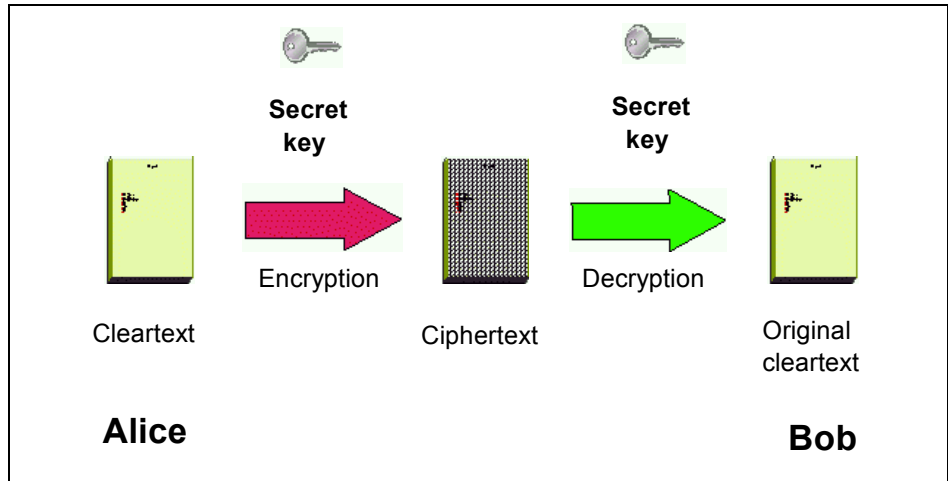


図 258 鍵付き暗号化および復号

注：

暗号について書かれた読み物では、プロトコルの最初の参加者をアリス、2番目の参加者をボブとすることが一般的です。彼らは「暗号カップル」です。

認証、整合性、および否認防止

暗号化により、メッセージに機密性が付加されます。インターネットのような信頼されないメディアを介して通信する場合、機密性に加えて、以下の点も必要になります。

- 認証 - メッセージの送信側が本当に名乗っている本人であることを検査するための方法。他の誰かになりすます侵入者も、認証によって検出されます。
- 整合性検査 - メッセージが通信パスの途中で変更されていないことを検査する方法。侵入者によって送信された改ざんメッセージがあれば、整合性検査で検出されます。副次作用として、通信エラーも検出されます。
- 否認防止 - 送信側がメッセージを本当に送信したことを証明する可能性。否認防止を実現するアルゴリズムを使うと、送信側は、問題のメッセージを送信したという事実を後で否定することができません。

21.2.2 対称または秘密鍵アルゴリズム

対称アルゴリズムは、鍵付きアルゴリズムで、復号鍵は暗号鍵と同様のものです。これらは、従来型の暗号アルゴリズムで、送信側と受信側とで安全な

通信を行う前に、両者がその鍵について同意していなければならないものです。図 258 には、対称アルゴリズムが示されています。対称アルゴリズムには、2 つのタイプがあります。数ビットのブロックでクリア・テキストを処理する **ブロック・アルゴリズム**、そして一度に 1 ビット (またはバイト) のクリア・テキストを処理する **ストリーム・アルゴリズム** です。

ブロック暗号は、さまざまなモードで使われます。電子コード・ブック・モード (ECB) は、いちばん簡単なモードです。クリア・テキストの各ブロックを個別に暗号化します。ブロック長を 64 ビットに指定する場合、 2^{64} の可能な入力クリア・テキスト・ブロックが存在し、それぞれは、 2^{64} の可能な暗号テキスト・ブロックの 1 つに正確に対応します。侵入者は、知られているクリア・テキスト / 暗号テキストのペアを使ってコード・ブックを構成し、攻撃を開始します。このように攻撃されやすいものであるため、暗号化ブロック・チェーン (CBC) モードがしばしば使われます。ここでは、前のブロックの暗号化の結果が、現在のブロックの暗号化に使われるため、各暗号テキスト・ブロックは、対応するプレーン・テキスト・ブロックに従属しているだけでなく、前のプレーン・テキスト・ブロックすべてに従属しています。

アルゴリズムは、しばしば初期設定ベクトル (IV) を利用します。これらは、鍵とは別の変数であり、アルゴリズムの初期状態を設定するときに便利です。

よく知られたブロック・アルゴリズムは、Data Encryption Standard (DES) です。これは、IBM が開発した世界的な標準暗号です。DES は、64 ビットのブロックを処理し、鍵の長さは 56 ビットです。しばしば、64 ビットの数値で表されますが、8 番目ごとのビットはパリティ・ビットとして機能します。この鍵から、16 の副鍵が派生され、16 回のアルゴリズム実行時に使われます。

DES は、クリア・テキストと同じ長さの暗号テキストを生成します。復号アルゴリズムは、暗号化のアルゴリズムとまったく同じですが、唯一の相違点は、副鍵のスケジュールです。このようなプロパティを持つため、ハードウェア実装に非常に適したものになります。

DES は長年使われていますが (起点は 70 年代初期にまでさかのぼります)、このアルゴリズムそのものは、まだ安全であると考えられています。このアルゴリズムに対する最も現実的な攻撃は、可能な鍵すべてを使って意味のある結果を探す、**力づくの復号**です。DES の問題点は、鍵の長さです。十分な時間とコンピューターが与えられれば、56 ビットの鍵に対する力づく攻撃が実行可能かもしれません。そのような理由で、最近では、Triple-DES と呼ばれる新しいモードの DES が一般的になってきています。Triple-DES を使う場合、元の DES アルゴリズムは、2 つか 3 つの異なる鍵を使って、3 回にわたって適用されます。この暗号化は、考え得る技術的な進歩を考慮に入れたとしても、当分の間破られることはないだろうと考えられています。

DES のエクスポート可能バージョンは、IBM の Commercial Data Masking Facility (CDMF) で、40 ビットの鍵を使います。

別の、さらに新しいブロック・アルゴリズムは、International Data Encryption Algorithm (IDEA) です。この暗号では、64 ビットのブロックと 128 ビットの鍵を使います。これは 90 年代初期に開発されたもので、DES を置き換えることを目的としていました。これは DES と比較して、暗号的に強くて速い暗号です。しかし、商業的には広く受け入れられていません。これは主に、比較的新しい暗号であり完全には分析されていないということが理由です。IDEA が最も意味のある形で使われているのは、フリーウェアのセキュア E メール・パッケージである Pretty Good Privacy (PGP) の中です。

ストリーム・アルゴリズムの一例は A5 です。これは、デジタル携帯電話トラフィックを GSM 規格で暗号化するときに使われるもので、ヨーロッパで広く使われています。

対称アルゴリズムの利点は、その効率性です。これは、ハードウェアへ簡単に実装することができます。主な欠点は、鍵管理の難しさです。鍵を交換する安全な方法は確かに存在しますが、実装が非常に難しい場合があります。

21.2.3 非対称または公開鍵アルゴリズム

これらのアルゴリズムは、対称暗号の主な欠点、つまり安全な鍵交換チャネルの要件に取り組んでいます。2 つの異なる鍵を使うというのが、そのアイデアです。公開鍵は、名前が示すようにすべての人に知らせておき、秘密鍵は、所有者が強固なセキュリティで保持します。公開鍵から秘密鍵を判別することはできません。公開鍵で暗号化されたクリア・テキストは、対応する秘密鍵でのみ復号できます。秘密鍵で暗号化されたクリア・テキストは、対応する公開鍵でのみ復号できます。したがって、受信側の公開鍵で暗号化されたメッセージを送信する場合、対象となった受信側だけがそのメッセージを読むことができます。このプロセスは、図 259 で示しています。アリスは、暗号化されたメッセージをボブに送信しています。

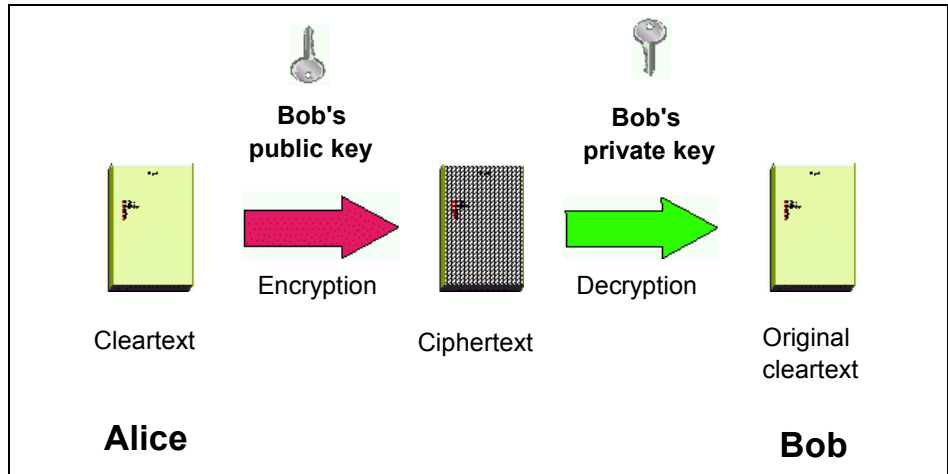


図 259 受信側の公開鍵を使った暗号化

公開鍵はだれでも使えるので、安全な鍵交換チャネルを用意しなくても、プライバシーは確保されます。通信することを望むパーティーは、相互の公開鍵を探します。

21.2.3.1 認証および否認防止

公開鍵アルゴリズムの興味深い特性の 1 つは、認証を提供できるという点です。暗号化のためには、秘密鍵が使われます。だれでも対応する公開鍵にアクセスしてメッセージを復号することができるため、ここにはプライバシーはありません。しかし、メッセージは認証されます。示された送信側の公開鍵を使って復号に成功する場合、そのメッセージは対応する秘密鍵で暗号化されています。この秘密鍵は、本当の送信側によってのみ知られます。そのため、送信側の ID が検査されます。秘密鍵での暗号化は、デジタル署名で使われます。その原理は、図 260 で示しています。アリスは、ボブがメッセージの認証性を検査できるようにするため、メッセージを自分の秘密鍵で暗号化（つまり「署名」）します。

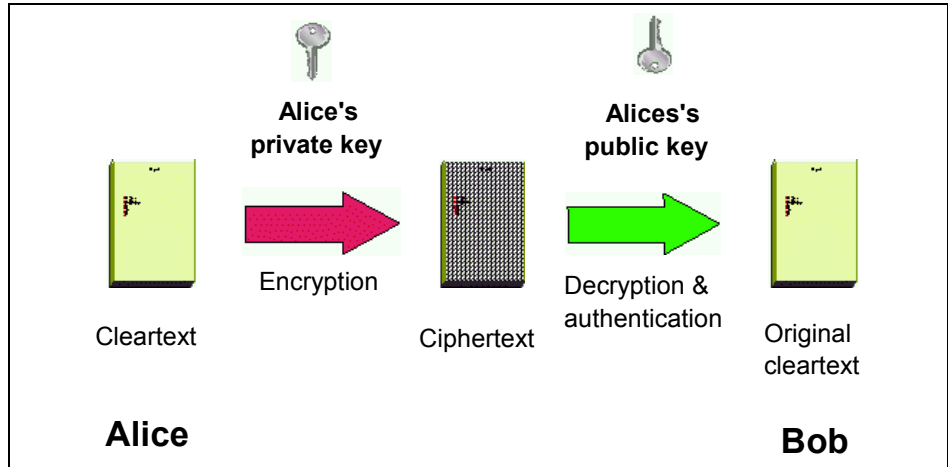


図 260 秘密鍵での暗号化による認証

さらに一歩進めると、秘密鍵で暗号化することにより、否認防止にもなります。メッセージを生成するときに秘密鍵を使えるのは該当するユーザーのみであるため、そのような暗号化されたメッセージが存在するだけで、発信元が本当にそのメッセージを送信したかどうかをテストすることになります。さらに、タイム・スタンプが含まれていれば、正確な日時も明らかになります。送信側が偽のタイム・スタンプを使えないようにする、信頼のおける第三者機関が関係するプロトコルが存在します。

21.2.3.2 公開鍵アルゴリズムの例

公開鍵を基にしたアルゴリズムは、さまざまな目的で使えます。以下のような、一般的な 2 つの適用例が存在します。

1. 暗号化 (693 ページの『RSA 公開鍵アルゴリズム』を参照)
2. 対称鍵アルゴリズムで使う共有鍵の生成 (694 ページの『Diffie-Hellman 鍵交換』を参照)

最も一般的な公開鍵アルゴリズムは、デファクト・スタンダードである RSA です。この名前は、Ron Rivest、Adi Shamir、および Leonard Adleman という 3 人の発明者にちなんで付けられました。RSA のセキュリティは、大きな数を因数分解することの難しさに依存しています。公開鍵と秘密鍵は、2 つの非常に大きい (200 桁以上) 素数の関数です。公開鍵と暗号テキストが与えられて、攻撃側が 2 つの素数の積を因数分解できると攻撃は成功します。RSA は、何年にもわたる広範な攻撃をしのいできています。計算能力が向上すると、RSA による安全を保つために、鍵の長さを長くすることになります (DES とは違います。DES では鍵の長さが固定されています)。

別の公開鍵アルゴリズムは、実際には最初に発明されたものですが、*Diffie-Hellman* です。これは鍵交換アルゴリズムです。つまり、安全ではないチャンネルで安全に共有秘密を確立するときに使います。通信するパーティーは、鍵を派生元となる公開情報を交換します。盗聴者は、安全ではないチャンネルに流された情報から鍵を再構成することはできません。さらに正確に言うと、再構成はコンピューター的に実行不可能なことです。*Diffie-Hellman* のセキュリティは、有限フィールドで離散対数を計算することの難しさに依存しています。共有秘密を確立後は、DES のような対称鍵アルゴリズムで使うために、共有秘密を用いて鍵を派生することができます。

Diffie-Hellman は、共有秘密鍵をできる限り安全に派生しますが、パーティーを認証するものではありません。認証するためには、RSA のような別の公開鍵アルゴリズムを使う必要があります。

残念なことに、公開鍵アルゴリズムは、鍵管理、プライバシー保護、認証、および否認防止の点では簡単なのですが、いくつかの欠点もあります。最も際立った点は、実行が遅く、ハードウェアに実装することが難しいということです。たとえば、RSA は、実装の状況によって異なりますが、DES よりも 100 倍から 10,000 倍遅くなります。このため、大量の暗号化では、普通は公開鍵アルゴリズムは使われません。このアルゴリズムを使う最も重要な場面は、鍵交換と認証です。別の際立った欠点は、対称アルゴリズムが耐える特定の暗号解読攻撃の影響を受けやすいという点です。

したがって、優れた暗号システム (暗号方式) では、両方のアルゴリズムを併用します。この方法では、認証と鍵交換のためのセッション確立フェーズには公開鍵アルゴリズムを使い、結果として生じるメッセージを暗号化するときには対称アルゴリズムを使います。

興味のある読者のために、モジュラー計算法が関係した、2 つの最も重要な非対称アルゴリズムを詳しく紹介します。算術演算モジュロ m (modulo m) は、その演算の結果が m で除算され、剰余を取ることを意味します。たとえば、 $3 * 6 \bmod 4 = 2$ のようになります。 $3 * 6 = 18$ であり、18 を 4 で除算すると剰余は 2 になるということです。

RSA 公開鍵アルゴリズム

RSA は、ISAKMP/Oakley フレームワークの中で、利用できる認証メソッドの 1 つとして使われます。RSA アルゴリズムの原理は、次のとおりです。

1. 2 つの大きな素数 p および q を取ります。
2. その積 $n = pq$ を求めます。 n はモジュラスといいます。
3. n より小さく $(p-1)(q-1)$ に対する比較的な素数である数値 e を選びます。
これは、 e と $(p-1)(q-1)$ は、1 以外の共通因数はないということです。

4. $d \bmod (p-1)(q-1)$ で、その逆を求めます。つまり、 $ed = 1 \bmod (p-1)(q-1)$ となります。

e と d は、それぞれ公開指数と秘密指数と呼ばれます。公開鍵は (n,e) のペアであり、秘密鍵は d です。因数 p と q は、秘密のままにしておくか、破棄する必要があります。

RSA 暗号化を簡単にした例は、次のとおりです。

1. アリスが秘密のメッセージ m をボブに送信するとします。アリスは、次のように累乗することにより、暗号テキスト c を作ります。

$$c = m^e \bmod n$$

e と n はボブの公開鍵です。

2. アリスは c をボブに送信します。
3. 復号するため、ボブは c を累乗します。

$$m = c^d \bmod n$$

さらに元のメッセージを回復します。 e と d の関連があるため、ボブは正確に m を回復します。ボブだけが d を知っているので、ボブだけが暗号テキストを復号することができます。

RSA 認証を簡単にした例は、次のとおりです。

1. アリスが署名付きのメッセージ m をボブに送信するとします。アリスは、次のように累乗することにより、デジタル署名 s を作ります。

$$s = m^d \bmod n$$

d と n はアリスの秘密鍵に属します。

2. アリスは s と m をボブに送信します。
3. 署名を検査するため、ボブは、結果を m と比較する場合に、次のように累乗して調べます。

$$m = s^e \bmod n$$

e と n はアリスの公開鍵に属します。

Diffie-Hellman 鍵交換

Diffie-Hellman 鍵交換は、ISAKMP/Oakley フレームワークの重要なコンポーネントです。鍵ネゴシエーション・セッションの初期の段階では、保護チャネルは機能していません。パーティーは、Diffie-Hellman アルゴリズムを使用して、共有秘密鍵を派生させます。これらの鍵は、鍵ネゴシエーション・プロトコルの次のステップで使われます。

アルゴリズムの概要は次のとおりです。

1. パーティー (アリスとボブ) は、2 つの公用値 (モジュラス m と整数 g) を共用しています。 m は、大きな素数でなければなりません。
2. アリスは、大きな乱数 a を生成し、次を計算します。

$$X = g^a \bmod m$$

3. ボブは、大きな乱数 b を生成し、次を計算します。

$$Y = g^b \bmod m$$

4. アリスは X をボブに送信します。

5. ボブは次を計算します。

$$K1 = X^b \bmod m$$

6. ボブは Y をアリスに送信します。

7. アリスは次を計算します。

$$K2 = Y^a \bmod m$$

$K1$ と $K2$ はどちらも、 $g^{ab} \bmod m$ と同じです。これは共有秘密鍵です。 a または b を知らずにこの値を生成できるユーザーはいません。この交換のセキュリティは、パーティーによって実行された累乗を逆算することはきわめて難しいという事実を基本としています。(つまり、サイズ m の有限フィールドで離散対数を計算すること。) RSA の場合と同様に、より大きな初期値 (この場合はより大きなモジュラス m) を選ぶことにより、敵の計算能力における進歩に対抗することができます。

ISAKMP/Oakley で Diffie-Hellman 交換を使う方法の詳細については、749 ページの 21.5.5 『Internet Key Exchange (IKE) プロトコル』を参照してください。

21.2.4 ハッシュ機能

ハッシュ機能 (メッセージ・ダイジェストとも言う) は、暗号化するときの基礎です。ハッシュ機能とは、可変長の入力データを取り、入力の「指紋」と見なされる固定長の出力データ (ハッシュ値) を生成する機能のことです。つまり、2 つのメッセージのハッシュが一致する場合、それらのメッセージは同じである可能性が高くなります。

暗号化に役立つハッシュ機能は、**一方向** でなければなりません。つまり、計算しやすいが、逆算は不可能でなければならないということです。一方向機能の日常の例としては、ポテトをマッシュする (つぶす) ことをあげられます。つぶすのは簡単ですが、つぶしてしまったら、元のポテトに戻すことは至難の業です。

優れたハッシュ機能は、**耐衝突** でもなければなりません。同じ値をハッシュする 2 つの異なる入力を見つけることは難しいはずで、何らかのハッシュ

機能が入力セットをより小さい出力セットへマッピングすると、理論的には衝突を見つけることは可能です。要点は、高い信頼性でメッセージを識別するために、メッセージに対して固有のデジタル「指紋」を付けることです。これはちょうど、実際の指紋で特定の人物を識別することと似ています。

2 番目の入力パラメーターとして鍵を使用するハッシュ機能とその出力は、図 261 で示しているように、メッセージとメッセージ確認コード (MAC) の 2 つに応じて異なります。

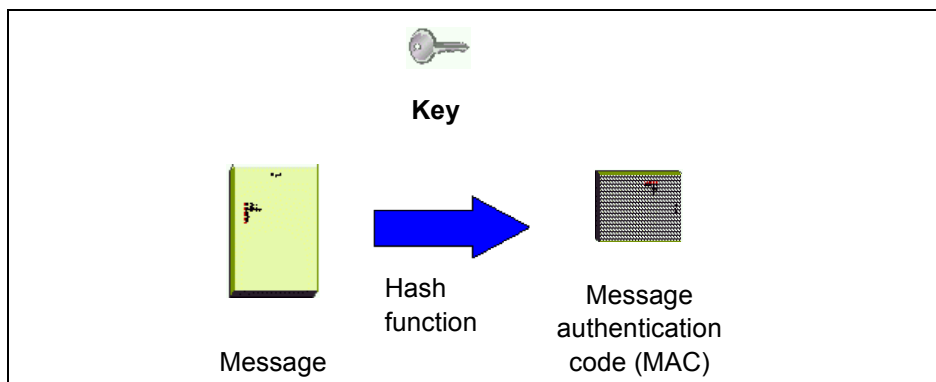


図 261 メッセージ確認コード (MAC) の生成

簡単に言うと、ハッシュを暗号化すると、MAC になります。メッセージに秘密鍵を追加してから連結をハッシュすると、結果は MAC になります。対称アルゴリズムと非対称アルゴリズムはどちらも、MAC の生成に使えます。

ハッシュ機能は、主に整合性と認証を確実にするために使われます。

- 送信側は、メッセージのハッシュを計算し、それをメッセージに添付します。
- 受信側は、受信メッセージのハッシュを計算してから、その結果と送信されたハッシュとを比較します。
- ハッシュが一致していれば、メッセージは改ざんされていません。
- 暗号鍵 (対称または非対称) が信頼されている送信側によってのみ知られている場合、正常に MAC が復号されるなら、自称の送信側と実際の送信側は同一ということです。

手順の図は、図 262 を参照してください。Message* 表記と MAC* 表記は、メッセージが信頼されていないチャネルを通過するときに変更される可能性があるという事実を表しています。

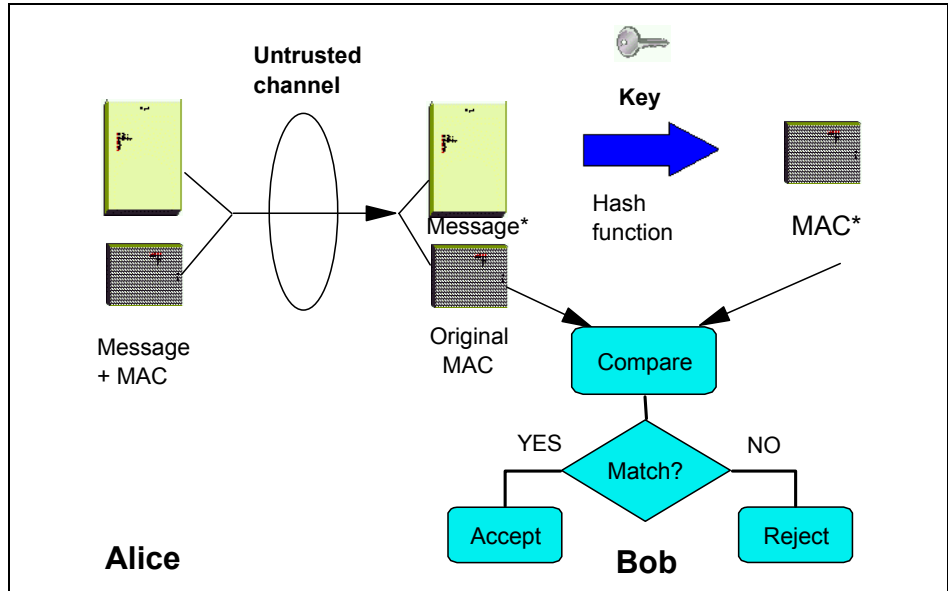


図 262 MAC を使った整合性および認証性の検査

侵入者が暗号化されたメッセージを変更した場合、復号は無意味となり、改ざんが検出できるため、どのような種類の暗号化でも結果は同じだと論じられる場合があります。メッセージのいくつかのフィールドに対する暗号化では、多くの場合、整合性または認証の両方または一方だけが必要というのが答えです。さらに、暗号化もプロセッサに大きく依存しています。例としては個人用バンキング・マシン・ネットワークがあります。ここでは、PIN だけが暗号化されますが、MAC は広く使われています。すべてのメッセージを暗号化することでの目覚ましい利益はありませんし、パフォーマンスは劇的に低下します。

秘密鍵でハッシュを暗号化することを、**デジタル署名**といいます。これは、特殊な MAC と考えることができます。秘密鍵でメッセージ全体を暗号化する代わりに、デジタル署名を使うと、かなりのパフォーマンスの向上と、新しく際立った特性を得られます。認証部分は、文書自体と切り離すことができます。この特性は、たとえば、Secure Electronic Transactions (SET) プロトコルで使われています。

公開鍵で秘密鍵を暗号化することを、**デジタル封筒**といいます。これは、対称アルゴリズムの秘密鍵を配布するときに使われる一般的な技法です。

21.2.4.1 ハッシュ機能の例

最も広く使われているハッシュ機能は、MD5 と Secure Hash Algorithm 1 (SHA-1) です。MD5 は、Ron Rivest (RSA の共同発明者) によって設計されました。SHA-1 は、MD5 の大きな影響を受け、Digital Signature Standard (DSS) に使う目的で、米国連邦情報・技術局 (NIST) と National Security Agency (NSA) によって設計されました。MD5 は 128 ビットのハッシュを生成するのに対し、SHA-1 は 160 ビットのハッシュを生成します。どちらの機能も、それぞれの出力にメッセージ長をエンコードします。SHA-1 は、より大きなハッシュを生成するため、より安全であると考えられています。

MD5 も SHA-1 も、入力パラメーターとして鍵を使いません。そのため、元の実装では、MAC 計算用に使うことはできません。しかしそうするために、鍵と入力データを連結し、機能をその結果に適用することは簡単です。

注：

実際には、たとえば IPsec では、さらに洗練された方式がよく用いられます。

鍵付き MD5 と鍵付き SHA-1

MD5 と SHA-1 を鍵付きのモードで使うことは簡単です。共有秘密鍵と保護するデータはどちらも、ハッシュ・アルゴリズムに対する入力です。次の IPsec の例では、図 263 で示しているように、データグラムは鍵と結合し、出力ハッシュ値は AH ヘッダーの Authentication Data フィールドに置かれます。

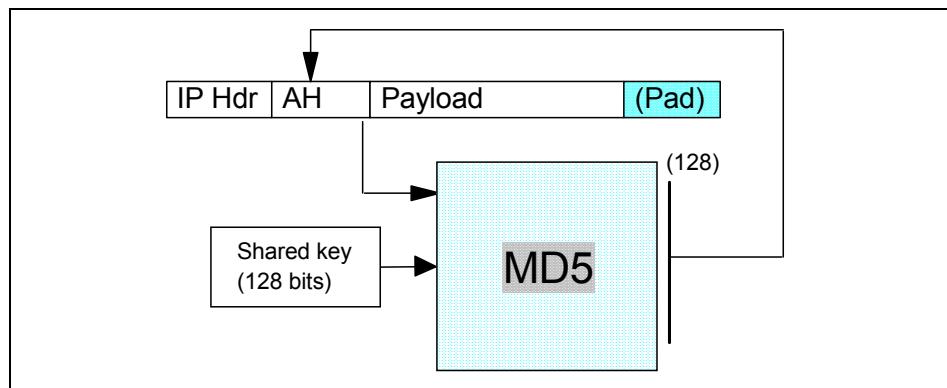


図 263 鍵付き MD5 の処理

鍵付き SHA-1 も同じ方法で機能します。唯一の違いは、ハッシュ値が 160 ビットより長いことです。

HMAC-MD5-96 と HMAC-SHA-1-96

より強力な方法としては、弊社で推し進めている Hashed Message Authentication Code (HMAC) があります。HMAC そのものはハッシュ機能ではなく、MAC 計算用の特定ハッシュ機能を使う暗号化は強力な方法です。

HMAC の機能を理解するため、一例として MD5 を考えてみます。基本の機能は、2 回連続して適用されます。1 回目は、MD5 への入力は、共有秘密鍵とデータグラムです。2 回目には、ハッシュ機能に対し、128 ビットの出力ハッシュ値と鍵が再度入力されます。結果のハッシュ値の左から 96 ビットまでは、データグラムの MAC として使われます。図は、図 264 を参照してください。

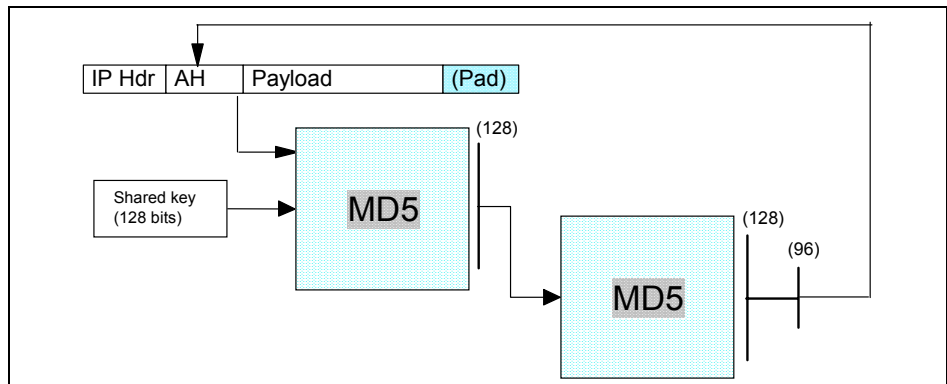


図 264 HMAC-MD5-96 の処理

HMAC-SHA-1-96 は、中間の結果が 160 ビットの長さである点を除き、同じ方法で機能します。

Digital Signature Standard (DSS)

前述のように、秘密鍵で暗号化されたハッシュ値は、デジタル署名といい、図 265 で示しています。

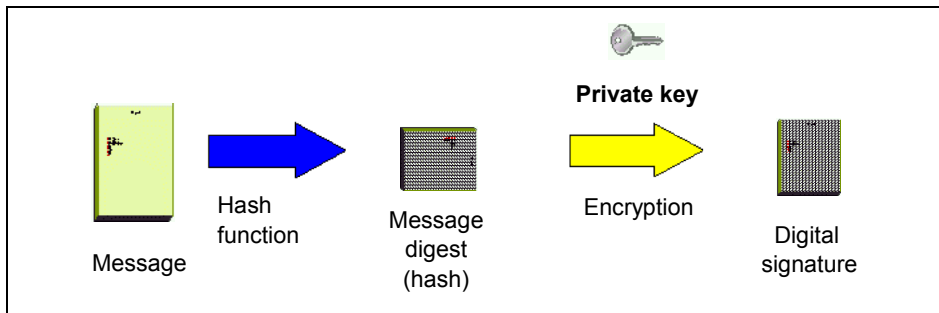


図 265 デジタル署名の生成

ISAKMP/Oakley で使うことのできる認証方式の 1 つは、米国政府のデジタル認証規格とするために NIST および NSA によって選定された DSS です。この規格は、SHA-1 で生成されるメッセージ・ダイジェストに署名し、署名を検査するときに使われる Digital Signature Algorithm (DSA) を記述しています。

以下で DSA を簡単に説明します。

1. 大きな素数 p を選びます。通常これは 512 ~ 1024 ビットの長さです。
2. $(p-1)$ の素因数 q を求めます。160 ビットの長さです。
3. 次を計算します。

$$g = h^{(p-1)/q} \bmod p$$

h は $(p-1)$ より小さい数値で、次が当てはまります。

$$h^{(p-1)/q} \bmod p > 1$$

4. 送信側の秘密鍵として、 q より小さい別の数値 x を選びます。
5. 次を計算します。

$$y = g^x \bmod p$$

これを送信側の公開鍵として使います。このペア (x, y) は、長期鍵ペアと言われることがあります。

6. 送信側は次のようにしてメッセージに署名します。

a. q より小さい乱数 k を生成します。

b. 次を計算します。

$$r = (g^k \bmod p) \bmod q$$

$$s = (k^{-1} (\text{SHA1}(m) + xr)) \bmod q$$

このペア (k, r) は、セッション別鍵ペアと言われることがあります。署名はペア (r, s) で表されています。

7. 送信側が (m,r,s) を送信します。
8. 受信側は次のようにして署名を検査します。
次を計算します。

$$\begin{aligned} w &= s^{-1} \bmod q \\ u_1 &= (\text{SHA1}(m) * w) \bmod q \\ u_2 &= (rw) \bmod q \\ v &= ((g^{u_1} y^{u_2}) \bmod p) \bmod q \end{aligned}$$

9. $v=r$ であれば、署名は検査されます。

21.2.5 デジタル証明書と認証局

691 ページの 21.2.3.1 『認証および否認防止』で説明されているように、公開鍵の暗号方式を用いると、パーティーは相互の公開鍵を取り出すことになります。しかし、ここで機密漏れが生じます。侵入者は、本物の公開鍵と侵入者の公開鍵を置き換えてから、いわゆる *中継ユーザー攻撃* を仕掛ける可能性があります。

たとえば、侵入者は、自分自身をアリスとボブの間に置きます。侵入者は、自分の公開鍵の 1 つをアリスの公開鍵に見せかけてボブに送信し、ボブをだますことができます。同じことはアリスにも当てはまります。アリスは自分がボブの公開鍵を使っていると思い込んでいますが、困ったことに、実際には侵入者の公開鍵を使っているのです。そのため、悪賢い侵入者は 2 人の間の機密トラフィックを復号することができ、しかもそのことは検出されません。たとえば、アリスによって送信され、ボブの公開鍵で暗号化されたメッセージが侵入者に届くと、侵入者はそれを復号し、内容を見てから、ボブの本物の公開鍵でそれを暗号化し直します。ボブは、アリスが偽の公開鍵を使っていることを知る由もありません。

さらに侵入者は、偽名を使って何か（たとえば、オンライン・ショッピング・モール）になりすまし、何も知らない買い物客をだますこともあります。

このような深刻な脅威に対するソリューションが *デジタル証明書* です。デジタル証明書とは、ID を関連する公開鍵へバインドするファイルのことです。このバインディングは、信頼のおける第三者機関、すなわち *認証局* (CA) によって検証されます。デジタル証明書は、認証局の秘密鍵で署名されているため、認証することができます。これは、申込者の検証後にのみ発行されます。通常、デジタル証明書には、公開鍵と ID 以外に、以下のような他の情報も含まれています。

- 発行の日付
- 有効期限
- 発行 CA からの各種情報（たとえば、シリアル番号）

注：

デジタル証明書には、ISO X.509 プロトコルという所定の国際規格があります。

パーティーは、相互のデジタル証明書を検索し、発行側の認証局の公開鍵を使い、その証明書を認証します。信頼における第三者機関がそれらの公開鍵を認証しているため、それらの公開鍵は本物であると言っていいでしょう。これにより、中継ユーザーと偽名の両方の攻撃から保護することができます。

1 つの CA ではすべての必要を満たすことはできないということは容易に想像が付きまゝ。ボブの証明書が、アリスの知らない CA によって発行される場合、どうなるでしょうか。アリスは、その認証局を信頼しても良いでしょうか。これは結局はアリスの決定次第なのですが、分かりやすく考えることができます。CA は 1 つの階層を形成することができ、これは *trust* チェーンと呼ばれることがあります。このチェーンの各メンバーには、それよりも上位の認証局によって署名された証明書があります。チェーン内の CA がより上位になると、より強力なセキュリティー手順が実施されます。ルート CA はすべてによって信頼され、その秘密鍵は最高機密になっています。

アリスは、信頼する CA が見つかるまで、チェーンを上方向に全探索できます。全探索では、より上位の CA によって発行された証明書を使い、従属 CA の公開鍵と ID が検査されます。

チェーンに信頼できる CA が見つかったら、アリスは、ボブの発行側 CA が信頼できるものと確信できます。これが、信頼の代行の全容です。私たちが信頼するだれかが署名したのであれば、その ID カードを信頼します。私たちがその署名者を知らなければ、上方を探索し、署名者がだれなのかといったことを確認できます。

この概念は、SET プロトコルで実装されています。主なクレジットカード・ブランドは、このプロトコルを利用して、共通のルートに集約される独自の CA 階層を運用しています。別の例として、Lotus Notes 認証も証明書に基づいていて、階層 *trust* チェーンを使って実装することができます。PGP でも同じような方法が用いられていますが、この *trust* チェーンはユーザーに基づいており、厳密な階層ツリーというよりは、分散した Web の形式になっています。

21.2.6 乱数生成プログラム

暗号方式の重要コンポーネントの 1 つとして、乱数生成プログラムがあります。乱数セッション鍵と乱数初期設定変数 (初期設定ベクトルと呼ばれることもある) は、何度も生成されます。たとえば、DES では、はっきりとした初

期設定ベクトルが必要であり、Diffie-Hellman では、鍵を導出するために、入力として機能する乱数を選択することが欠かせません。

品質、つまりこのような生成プログラムの乱数性は、想像以上にはるかに重要です。ほとんどのプログラム言語ライブラリーに付属する通常の乱数機能は、ゲームで使うには十分ですが、暗号化で使うには不十分です。そのような乱数生成プログラムは、どちらかという予測可能なものです。そのようなプログラムを使う場合、暗号解読者にとってはうれしいことに、暗号化された出力の中に関心のある相関を簡単に見つけられます。

乱数生成プログラムで直面する根本的な問題点は、コンピュータは究極の決定論マシンであるため、実際の乱数シーケンスの生成はできないということです。ジョン・フォン・ノイマンは、皮肉を込めてこのように言っています。「乱数を生成するための算術的な方法を考え出す人がいれば、まったく罪な話である。」このような理由で、*疑似乱数生成プログラム* という用語がより適しているわけです。

暗号的に強力な疑似乱数生成プログラムは、予測不能でなければなりません。次の乱数ビットを決定することは、生成プログラムの機能全体を使ったとしても、コンピュータ的に実行不可能でなければなりません。

疑似乱数生成プログラムでの一般的な実際の解決方法は、ハッシュ関数を使うことです。この方法では、十分な乱数性を得られ、効率的に実装することができます。軍事レベルの生成プログラムでは、物理現象に内在する乱数性を活用した、特別な装置を使います。PGP ソフトウェアには、興味深い解決方法があります。疑似乱数生成プログラムの初期シードは、ユーザーのキー・ストロークとキー・ストロークの間に経過した時間を計測することによって導き出されます。

21.2.7 暗号化の輸出 / 輸入制限

米国の輸出規定は 1996 に改変され、暗号化は商務省の統制下に置かれるようになりました。これは、以前は軍需品として扱われていたものです。このことは、暗号製品の輸出を自由化する上で、有効なステップです。

新しい輸出規定によれば、ある企業が承認された鍵リカバリー計画を持っていれば、その企業に対して 56 ビットの鍵暗号化アルゴリズムを輸出するライセンスを与えられるようになっていきます。この鍵リカバリー計画は 2 年以内に実装しなければならず、ライセンスは 6 か月単位で与えられます。金融機関は、鍵リカバリー計画の制限を免除してもらうことも可能です。

1997 年に、IBM は、承認されている他の製品と同じ方法で使う限り DES を輸出できるライセンスを与えられました。1998 年には、銀行業務アプリケーション用に、(Triple-DES のような) 強い暗号化製品の輸出が許可されました。

1998 年の 9 月には、ホワイトハウスは、暗号製品と鍵リカバリー要件に対する米国輸出制限のさらなる自由化を発表しました。要約すると以下のようになります。

- 56 ビットの DES およびそれに相当する製品を輸出するときの鍵リカバリー要件は除かれました。これには、56 ビットの対称鍵アルゴリズムとともに 1024 ビットの非対称鍵交換機能を使う製品も含まれます。
- 強力な暗号 (たとえば、3DES) を、(鍵リカバリーを使っても使わなくても) ライセンスの例外の下で制限なく輸出することは、45 か国で、金融業界以外の業界にも広がりつつあります。ここには、オンライン取引を保護する目的で、米国企業の子会社、保険、健康および医療 (生化学メーカーおよび製薬会社を除く)、およびオンライン商店が含まれています (ただし、軍需品と考えられるアイテムの流通業者は除く)。

後者の場合、リカバリー可能な製品には、外国の鍵リカバリー・エージェントに検討してもらわなくても、世界中で例外が適用されます (テロ支援国家は除く)。

- リカバリー可能な製品の輸出は、広範な国々で、主要な商業マーケットのほとんどの民間企業に許可されます (米国軍需品リストのアイテムを除く)。
- エンド・ユーザーに対する輸出ライセンスは、ケース・バイ・ケースで与えられます。

詳細は、米国商務省のホーム・ページから入手できます。

<http://www.bxa.doc.gov/Encryption/>

フランスの法律では、ユーザー・データを暗号化 / 復号できる製品には、マーケットに出す前に、フランス政府からのライセンスが与えられなければならないことになっています。顧客は、ケース・バイ・ケースですが、そのような製品を使う許可を得る必要があります。実際には、以下の 2 つの主要かつ便利な例外が存在します。

1. 通常のこととして、銀行が世界中で DES 製品を使えるようになるライセンスが与えられます (ケース・バイ・ケースの許可は必要ありません)。
2. 通常のこととして、だれでも弱い暗号化 (40 ビットの鍵を使った RC2/RC4) を使えるようになる世界的なライセンスが与えられます。

21.3 ファイアウォール

ファイアウォールには、組織のセキュリティ・ポリシーに有効な機能が備えられています。したがって、これらの機能を理解し、ネットワークへ適切に適用することが大切です。この章では、ファイアウォールの概念、ネット

ワーク・セキュリティー、ファイアウォール・コンポーネント、およびファイアウォールの例を説明します。

21.3.1 ファイアウォールの概念

ファイアウォールとは、保護された内部ネットワークとインターネットなどの信頼されていないネットワークの間のセキュリティー・ポリシーを実施するシステム（あるいはシステムのグループ）のことです。ファイアウォールは、インターネットとプライベート・ネットワークとの間を保護するものに見えることがあります。しかし、一般的に言って、ファイアウォールは、世界を2つ以上のネットワーク、つまり1つ以上の保護ネットワークと1つ以上の非保護ネットワークに分割する手段と考える必要があります。

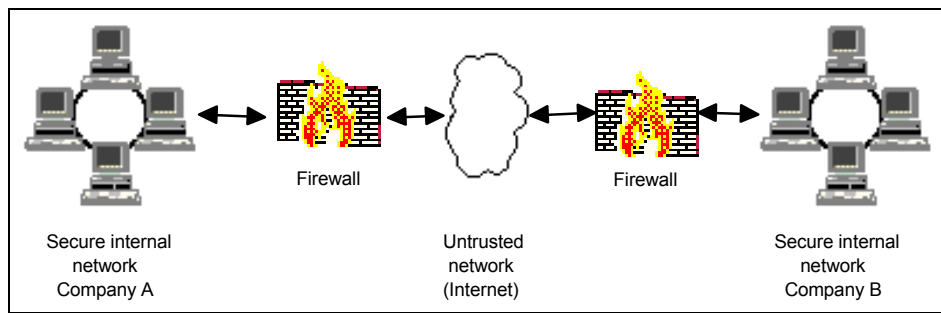


図 266 ファイアウォールの図

ファイアウォールは、PC、ルーター、中型機、メインフレーム、UNIX ワークステーション、またはこれらの組み合わせで構いませんが、外部からアクセスできる情報やサービス、その情報やサービスを外部から使えるユーザーを決定します。一般的に、ファイアウォールは、*チョーク・ポイント*とも呼ばれる、保護された内部ネットワークと信頼されていない外部ネットワークが接する位置にインストールされます。

ファイアウォールの仕組みを理解するため、ネットワークを、出入りを制御する必要のあるビルディングであると考えてください。このビルディングには、唯一の入り口であるロビーがあります。このロビーでは、受付係が来訪者を歓迎し、警備員が来訪者を見張り、ビデオ・カメラが来訪者の行動を記録し、バッジ読取装置がそのビルディングに入る来訪者を認証します。

このような手続きがよく機能してビルディングへの出入りを制御できるかもしれませんが、許可されていない人物が侵入に成功すると、この侵入者の行動に対してビルディングを保護する方法はありません。しかし、侵入者の動きが監視されていれば、疑わしい行動があったときにそれを検出できるかもしれません。

同様に、ファイアウォールは、内部の保護ネットワークと信頼されていない外部ネットワークの間のアクセスを制御することにより、組織の情報リソースを保護するように設計されています (図 267 を参照してください)。しかし、ファイアウォールが、信頼されているデータの通過を許可し、無防備なサービスを拒否し、外部攻撃から内部ネットワークを保護するよう設計されているとしても、新しく作成された攻撃が、そのファイアウォールをいつでも貫通する可能性があることを覚えておくことは大切です。ネットワーク管理者は、ファイアウォールによって生成されたすべてのログとアラームを定期的に調べる必要があります。一般的に言って、そのようにしないと、外部の攻撃から内部ネットワークを保護することは不可能です。

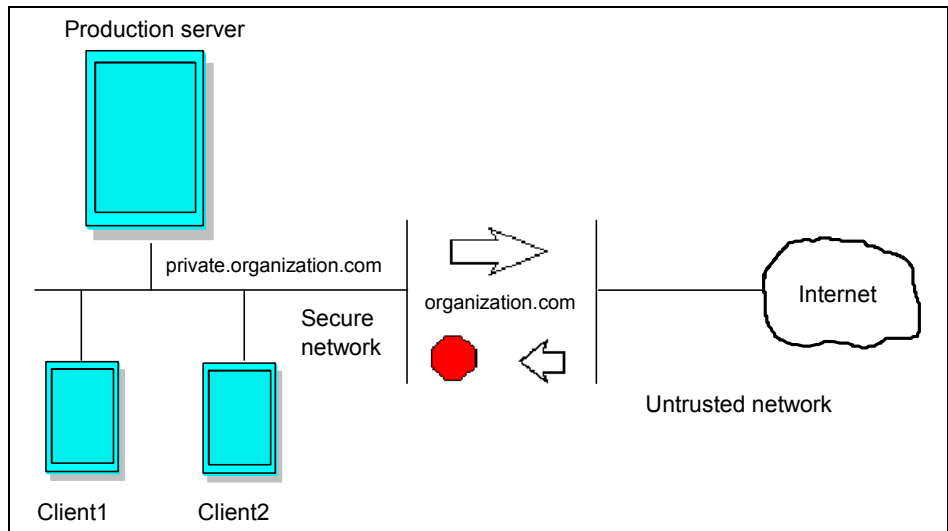


図 267 ファイアウォールは保護ネットワークとインターネットとの間のトラフィックを制御する

21.3.2 ファイアウォール・システムのコンポーネント

前述のように、ファイアウォールは、PC、中型機、メインフレーム、UNIX ワークステーション、ルーター、またはこれらの組み合わせで構いません。ファイアウォールは、要件に応じて、以下の 1 つ以上の機能コンポーネントで構成することができます。

1. パケット・フィルタ操作ルーター
2. アプリケーション・レベル・ゲートウェイ (プロキシ)
3. 回線レベル・ゲートウェイ

これらのコンポーネントのそれぞれに、異なる利点と欠点があります。一般に、有効なファイアウォールを構築するため、これらのコンポーネントを同時に使います。

21.3.3 パケット・フィルタ操作ルーター

ほとんどの場合、パケット・フィルタ操作は、フィルタ操作規則に従ってパケットを転送できるルーターを使うことによって実現します。パケットがパケット・フィルタ操作ルーターに到着すると、ルーターは、パケットのヘッダーから特定の情報を取り出し、そのパケットを通過させるか廃棄するかに関するフィルタ規則に従って決定します (図 268 を参照)。パケットのヘッダーからは、以下の情報を取り出すことができます。

- ソース IP アドレス
- 宛先 IP アドレス
- TCP/UDP ソース・ポート
- TCP/UDP 宛先ポート
- ICMP メッセージ・タイプ
- カプセル化されたプロトコル情報 (TCP、UDP、ICMP または IP トンネル)

パケット・フィルタ操作規則は、ネットワーク・セキュリティー・ポリシー (685 ページの 21.1.4 『ネットワーク・セキュリティー・ポリシー』を参照) に基づいています。そのため、パケット・フィルタ操作は、これらの規則を入力として使うことによって行われます。フィルタ操作規則を決定するときは、外部の攻撃を考慮に入れると同時に、サービス・レベル制限およびソース / 宛先レベル制限も考慮に入れる必要があります。

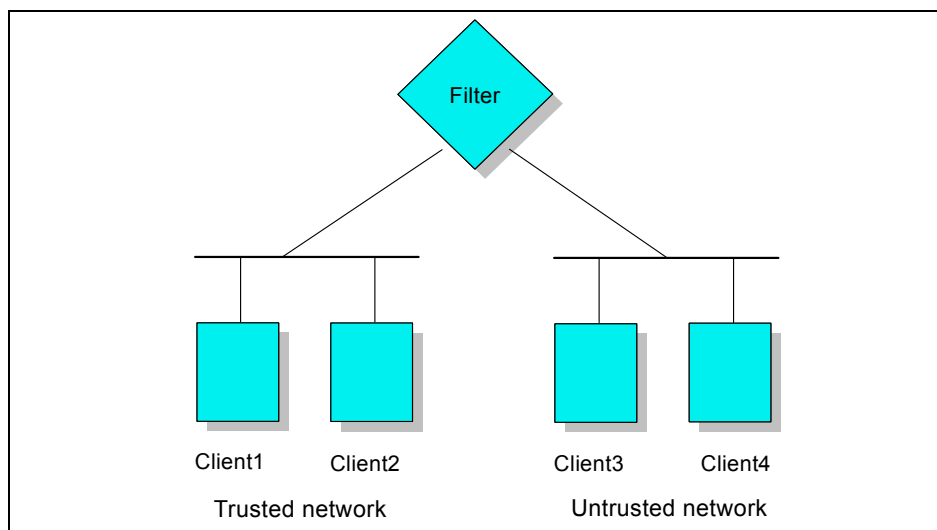


図 268 パケット・フィルタ操作ルーター

サービス・レベル・フィルター操作

ほとんどのサービスでは、ウェルノウン TCP/UDP ポート番号を使っているため、フィルターで関連するポート情報を使ってサービスを許可したり拒否することができます。たとえば、FTP サーバーは、TCP ポート 21 で接続を待機し、受動モード・クライアント以外の場合は、ポート 20 からアウトバウンド・データ接続を確立します。したがって、FTP 接続を保護ネットワークへ渡せるようにするには、ヘッダーに TCP ポートとして 20 と 21 が含まれるパケットを許可するようルーターを構成できます。一方で、RPC を用い、接続ごとに個別のポートを使う、NFS のようなアプリケーションもいくつか存在します。このような種類のサービスを許可すると、セキュリティ問題が引き起こされる可能性があります。

ソース / 宛先レベル・フィルター操作

パケット・フィルター操作規則では、ルーターは、パケットのヘッダーの宛先またはソース情報に従って、パケットを許可したり拒否したりできます。ほとんどの場合、あるサービスが利用できる場合、外部のユーザーがアクセスを許可されるのはそのサーバーだけです。ヘッダーに別の宛先が示されていたり宛先情報のない他のパケットは、廃棄されます。

拡張フィルター操作

前述のように (681 ページの 21.1.1 『セキュリティに対する一般的な攻撃』を参照)、プライバシーやネットワーク・セキュリティを脅かす、さまざまなタイプの攻撃があります。そのうちのいくつかは、IP オプションの検査、フラグメント・オフセットなどの拡張フィルター操作規則を用いることで廃棄できます。

パケット・フィルター操作の制限

パケット・フィルター操作規則は、非常に複雑になる場合があります。既存の規則に例外があると、さらに複雑になってしまいます。利用できるテスト・ユーティリティはいくつかあるのですが、それでもネットワーク・セキュリティにいくつかのホールが残されている可能性があります。パケット・フィルターは、ネットワークにとって絶対的な保護とはなりません。いくつかのケースでは、いくつかの情報のセット (たとえば、コマンド) を制限して、内部の保護ネットワークに入らないようにする必要があります。パケット・フィルターでは、特定サービスの内容を理解できないため、パケット・フィルターを指定したデータを制御することは不可能です。そのために、アプリケーション・レベルの制御が必要になるのです。

21.3.4 アプリケーション・レベル・ゲートウェイ (プロキシ)

アプリケーション・レベル・ゲートウェイは、*プロキシ*と呼ばれることがあります。アプリケーション・レベル・ゲートウェイは、2つのネットワーク間のトラフィックに対して、より高いレベルの制御を実現します。この制御では、特定サービスの内容を、ネットワーク・セキュリティ・ポリシーに従って監視してフィルターできます。したがって、必要なアプリケーションでは、ゲートウェイを通過するその特定サービスを管理するため、対応するプロキシ・コードをゲートウェイにインストールする必要があります (図 269 を参照)。

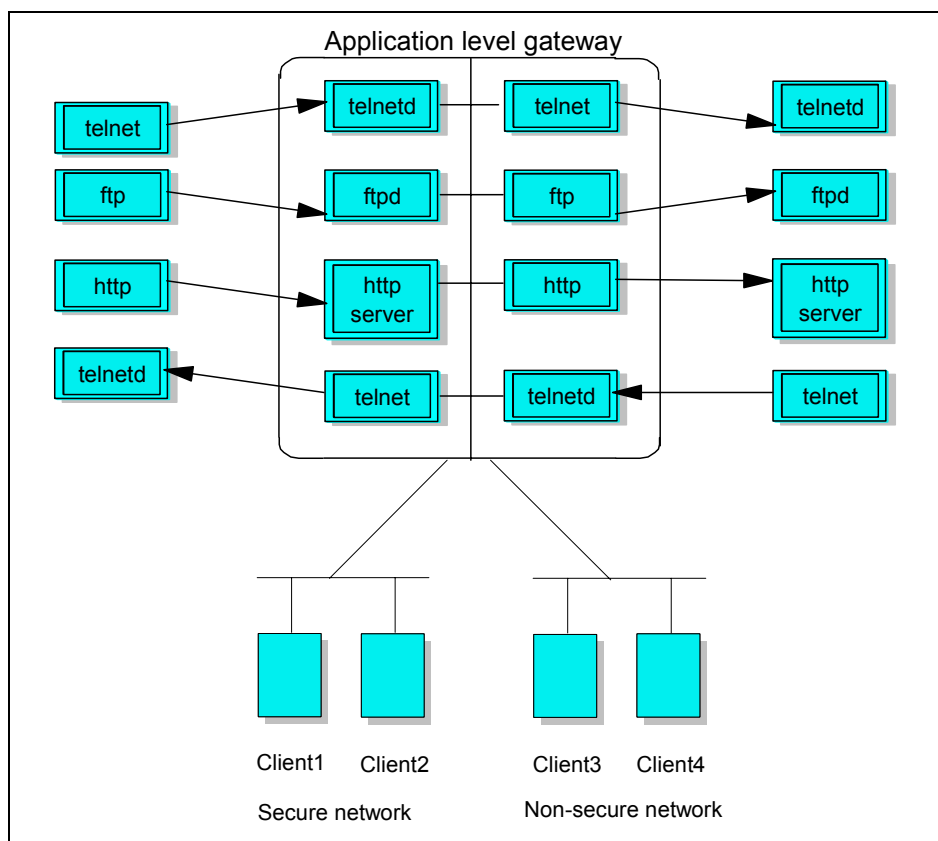


図 269 アプリケーション・レベル・ゲートウェイ

プロキシは、クライアントに対してはサーバーとして、宛先サーバーに対してはクライアントとして機能します。クライアントと宛先サーバーの間には、仮想接続が確立されます。プロキシは、クライアントとサーバー側からは、*透過的*に、つまりプロキシとの境界がないように見えますが、コマンドのような特定タイプのデータを宛先に送信する前に、それらのデータ

を監視してフィルター操作することができます。たとえば、FTP サーバーは、外部からアクセスすることを許可されています。考え得る攻撃からサーバーを保護するため、ファイアウォールの FTP プロキシを、PUT および MPUT コマンドを拒否するように構成できます。

プロキシ・サーバーとは、アプリケーション固有の中継サーバーで、保護および無保護ネットワークに接続するホスト上で稼働します。プロキシ・サーバーの目的は、IP レベルではなくアプリケーション・レベルで、2つのネットワーク間のデータの交換を制御することにあります。プロキシ・サーバーを使うことにより、プロキシ・サーバーで処理できるアプリケーション・プロトコル用の保護ネットワークと無保護ネットワークとの間で IP ルーティングを使用不可にできますが、プロキシ・サーバーでデータを中継することにより、引き続きネットワーク間でデータを交換することが可能です。図 270 には、FTP プロキシ・サーバーが示されています。

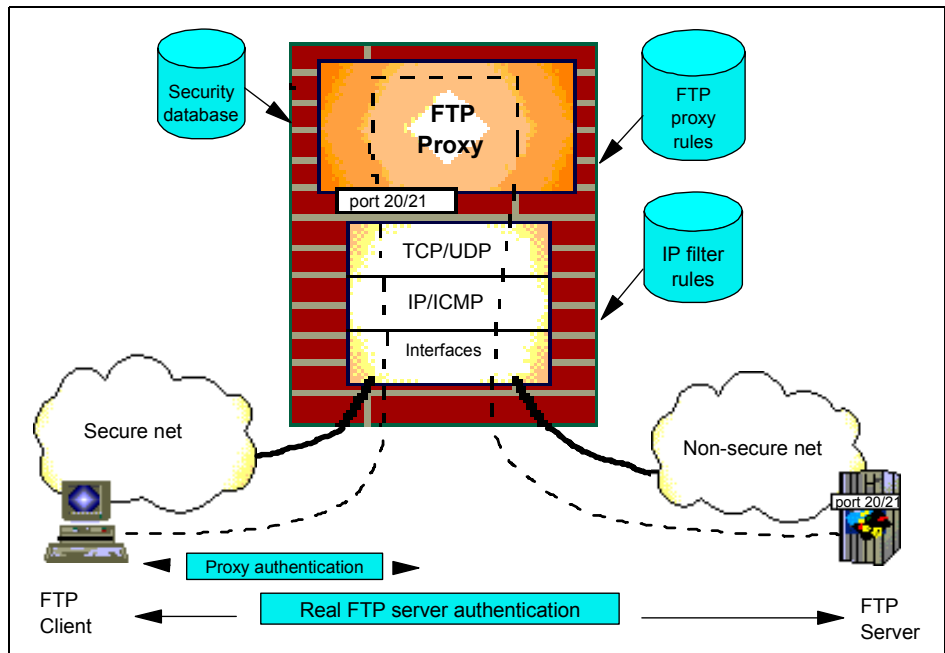


図 270 FTP プロキシ・サーバー

どのようなクライアントでも、プロキシ・サーバーへアクセスできるようにするには、そのクライアント・ソフトウェアを特別に変更する必要がある点に注意してください。つまり、クライアントおよびサーバー・ソフトウェアは、プロキシ接続をサポートしていなければならないということです。前の例では、FTP クライアントは、まずプロキシに対してそれ自体を認証する必要があります。認証が成功したら、プロキシの制限に基づく FTP

セッションが開始されます。ほとんどのプロキシ・サーバー実装システムでは、セキュリティー ID カードのような、さらに洗練された認証方法を用います。このメカニズムは、他の接続では再利用できない固有の鍵を生成します。IBM Firewall では、2 種類の ID カードがサポートされています。すなわち、Axent からの SecureNet カードと Security Dynamics からの SecureID カードです。

IP フィルターと比較して、アプリケーション・レベル・ゲートウェイには、接続のアプリケーション・データに基づいた、さらに広範囲のログギングが備えられています。たとえば、HTTP プロキシは、ユーザーが訪問した URL をログに記録できます。アプリケーション・レベル・ゲートウェイの別の機能は、強力なユーザー認証を使えるということです。たとえば、保護されていないネットワークから FTP および TELNET サービスを利用するときに、ユーザーはプロキシに対して強制的に認証させることができます。図 271 には、プロキシ・サーバー TCP セグメント・フローの例が示されています。

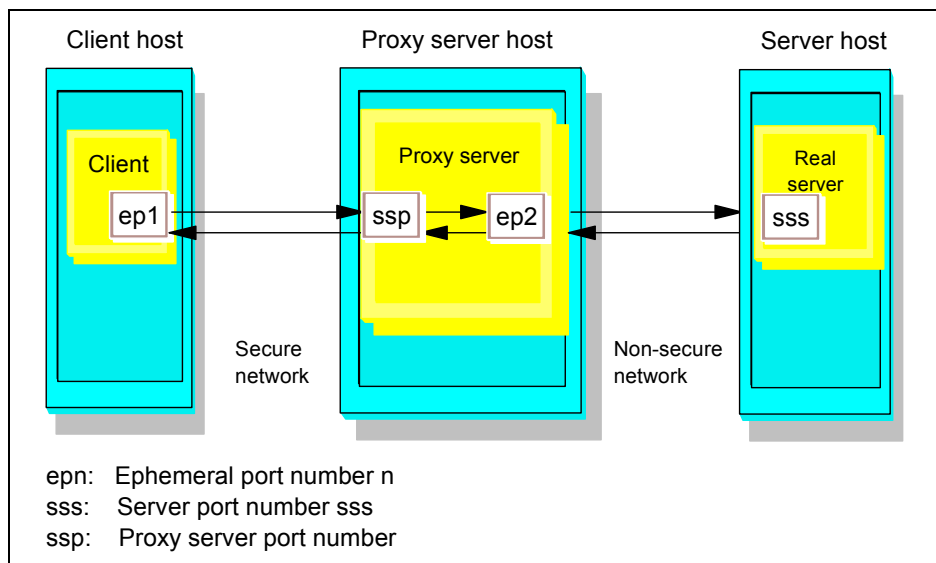


図 271 プロキシ・サーバー TCP セグメント・フロー

21.3.4.1 アプリケーション・レベル・ゲートウェイの制限

アプリケーション・レベル・ゲートウェイの欠点の 1 つは、プロキシ・サーバー経由で接続を確立するには、クライアント・ソフトウェアを変更して、そのプロキシ・サービスをサポートさせなければならないという点です。このことは、ソフトウェアを変更するのではなく、ユーザーの動作でいくらかの変更が加わることによって実現する場合があります。たとえば、プ

ロキシー経由で TELNET サーバーに接続するには、通常は、ユーザーはプロキシー・サーバーに認証され、その後で宛先 TELNET サーバーに認証されなければなりません。この場合、接続を確立するために、1 つではなく 2 つのユーザー・ステップが必要です。ただし、変更した TELNET クライアントでは、TELNET コマンドにプロキシー・サーバーではなく宛先ホストを指定することにより、ユーザーにとってそのプロキシー・サーバーを透過的にすることができます。

21.3.4.2 1 つの例：FTP プロキシー・サーバー

ほとんどの場合、FTP プロキシー・サーバーを使うために、ユーザーには有効なユーザー ID とパスワードが必要になります。UNIX システムでは、そのユーザーを UNIX システムのユーザーとして定義する必要もあります。

FTP は、以下の 2 つのいずれかのモードで使うことができます。

1. 通常モード
2. 受動モード

通常モードでは、FTP クライアントは、まず FTP サーバーのポート 21 へ接続し、制御接続を確立します。データ転送が必要な場合 (たとえば、DIR、GET、または PUT コマンドの結果として)、クライアントは PORT コマンドをサーバーへ送信し、サーバーのデータ・ポート (ポート 20) からクライアント・ホスト上の指定した一時ポート番号へのデータ接続を確立するよう、サーバーに指示を出します。

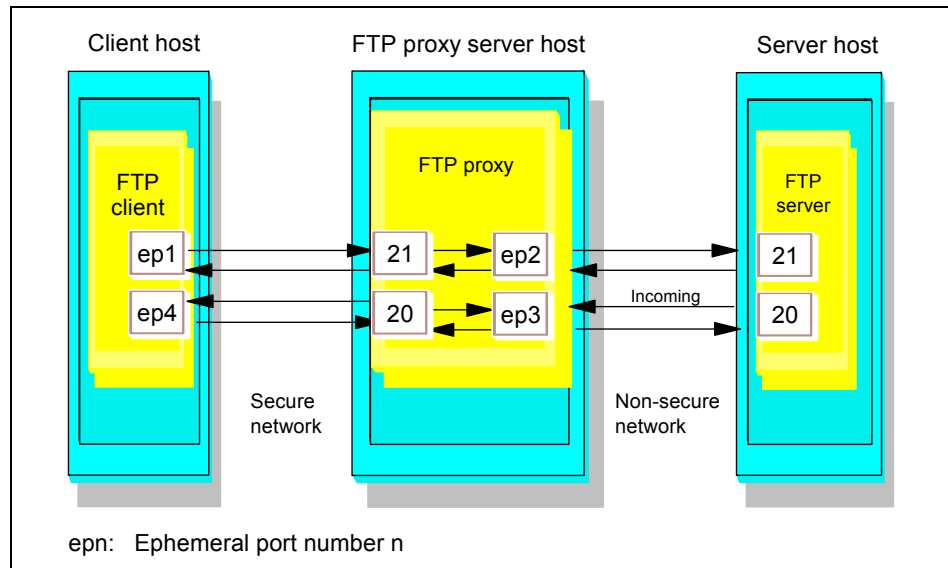


図 272 通常モードの FTP プロキシー

FTP プロキシ・サーバーの状態では、通常モードは、無保護ネットワークから FTP プロキシ・ホストへのインバウンド TCP 接続を許可しなければならないことを意味します。図 272 では、無保護ネットワークの FTP サーバー・ポート 20 から FTP プロキシ・サーバーの一時ポート番号への接続が確立される仕組みに注意してください。この仕組みを実現させるために、ポート 20 から FTP プロキシ・ホスト上の一時ポート番号へのインバウンド接続要求を許可する、IP フィルター操作規則が使われます。これは、通常は IP フィルター規則ではありません。IP フィルター規則では、クラッカーは、ポート 20 でプログラムを実行し、1023 より上のすべてのポート番号をスキャンできるので、最も簡単な形態であっても、サービス状態が拒否されてしまう可能性があります。そのため、カスタム・フィルタ規則の構成を追加するほうがよい場合があります。いくつかのファイアウォールは、発信する FTP 要求のテーブルを作成し、対応する着信データ転送要求と突き合わせることで、この方式を適切に処理します。

はるかにファイアウォール・フレンドリーなモードは、図 271 で示しているような、受動モードの操作です。このモードは、ファイアウォール・フレンドリー FTP と言われていて、RFC 1579 Firewall-Friendly FTP で説明されています。

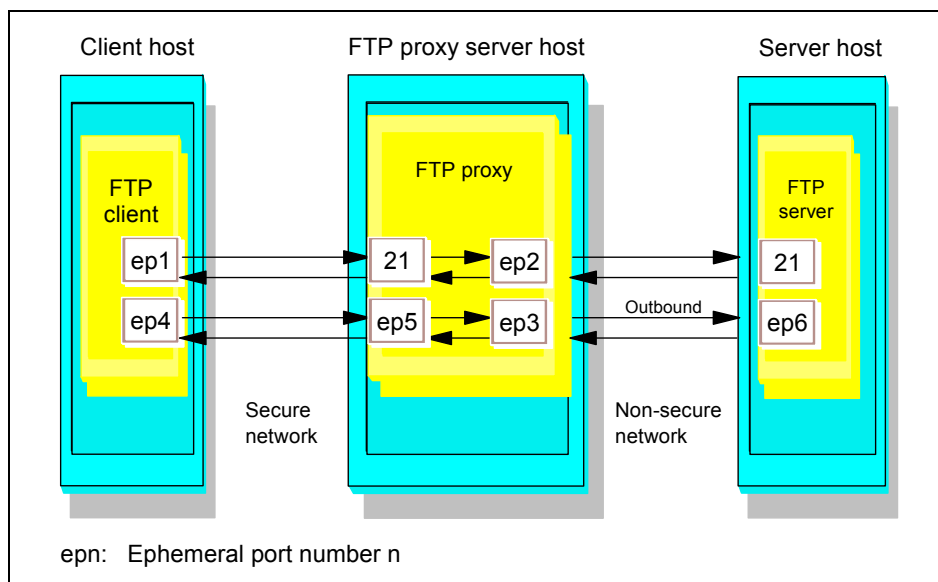


図 273 受動モードの FTP プロキシ (ファイアウォール・フレンドリー FTP)

受動モードでは、FTP クライアントはもう一度、サーバーのポート 21 に対する制御接続を確立します。データ転送が開始されると、クライアントは PASV コマンドをサーバーへ送信します。サーバーは、データ接続を確立す

るために、接触するクライアントにポート番号で応答します。その後、クライアントはデータ接続を開始します。

このセットアップでは、ポート 21 と無保護ネットワーク内の任意の一時ポート番号への接続を確立するため、FTP プロキシ・ホストでは一時ポート番号が使われます。ここでは、外部へ接続するので、一時ポート番号へのインバウンド接続を許可する規則は必要ありません。

21.3.5 回線レベル・ゲートウェイ

回線レベル・ゲートウェイは、TCP 接続を中継し、それ以上の処理またはフィルターを行いません。いくつかの回線レベル・ゲートウェイでは、UDP パケットを扱えます。回線レベル・ゲートウェイは、特別なタイプのアプリケーション・レベル・ゲートウェイと呼ばれることがあります。これは、アプリケーション・レベル・ゲートウェイでは、回線レベル・ゲートウェイでも同様に、ユーザーが認証されるとすべての情報を渡すよう構成できるからです (図 274 を参照)。しかし、この 2 つには、実際には以下のような重大な相違点があります。

- ・ 回線レベル・ゲートウェイでは、アプリケーションごとにクライアント側で余分な変更を加えなくても、複数の TCP/IP アプリケーションと UDP アプリケーションを処理できます。したがって、これにより回線レベル・ゲートウェイは、ユーザー要件を満たすにはよい選択となります。
- ・ 回線レベル・ゲートウェイでは、パケット処理またはフィルター操作は行われません。したがって、回線レベル・ゲートウェイは、一般に *透過的* ゲートウェイと呼ばれます。
- ・ アプリケーション・レベル・ゲートウェイには、UDP のサポートがありません。
- ・ 回線レベル・ゲートウェイは、しばしばアウトバウンド接続に使われます。一方、アプリケーション・レベル・ゲートウェイ (プロキシ) は、インバウンドとアウトバウンドの両方の接続に使われます。一般に、両方のタイプを組み合わせる場合、アウトバウンド接続には回線レベル・ゲートウェイを使い、インバウンド接続にはアプリケーション・レベル・ゲートウェイが使われるので、セキュリティとユーザーの要件を両方とも満たすことができます。

回線レベル・ゲートウェイでは、着信 UDP パケットまたは TCP 接続を処理できる場合もあります。しかし、保護側のクライアントは、そのようなパケットを予測するようにゲートウェイに通知しなければなりません。SOCKS v5 には、この機能があります。

回線レベル・ゲートウェイのよく知られた例の 1 つは、SOCKS です (詳細については、769 ページの 21.6 『SOCKS』を参照してください)。SOCKS を

介したデータ・フローは監視されることもフィルター操作されることもないため、セキュリティー問題が生じる可能性があります。セキュリティー問題を最小限にするため、外側のネットワーク（信頼されていないネットワーク）では、信頼されているサービスおよびリソースを使う必要があります。

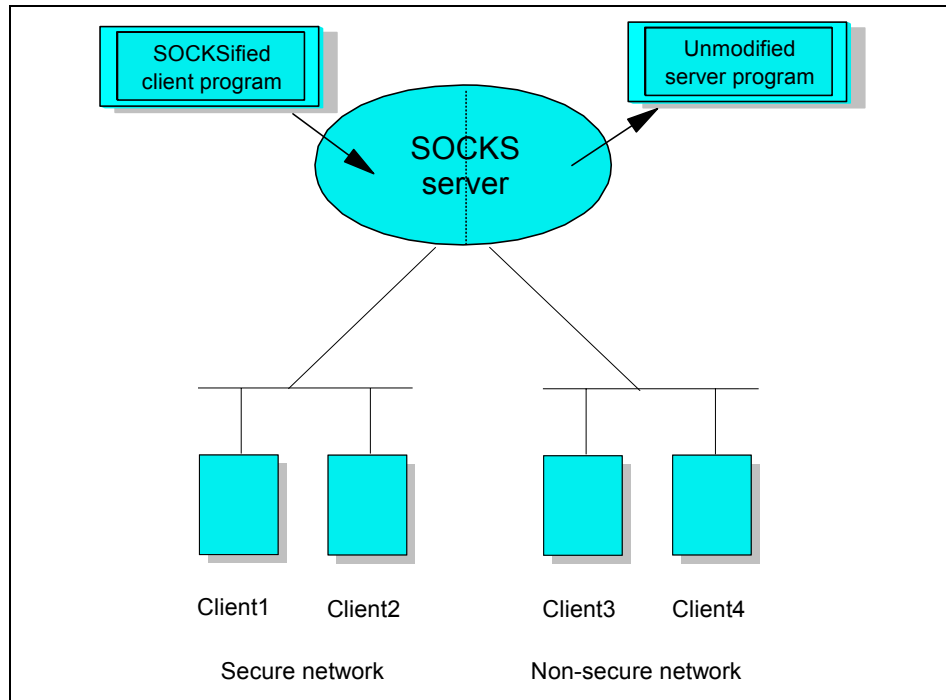


図 274 回線レベル・ゲートウェイ

21.3.6 ファイアウォールのタイプ

ファイアウォールは、1 つ以上のホストで稼働する、1 つ以上のソフトウェア・エレメントで構成されます。ホストは、汎用コンピューター・システムでも構いませんし、ルーターのような特化されたコンピューターでも構いません。重要なファイアウォールの例を 4 つ挙げることができます。それらは以下のとおりです。

1. パケット・フィルター操作ファイアウォール
2. デュアル・ホーム・ゲートウェイ・ファイアウォール
3. 選別ホスト・ファイアウォール
4. 選別サブネット・ファイアウォール

21.3.6.1 パケット・フィルタ操作ファイアウォール

パケット・フィルタ操作ファイアウォールは、コストがかからないため、一般的に使用されます(図 275 を参照してください)。このファイアウォールは、外部ネットワークと内部の保護ネットワークとの間に置かれている単なるルーターです。パケット・フィルタ操作規則は、トラフィックを許可したり拒否するように定義されています。707 ページの『パケット・フィルタ操作ルーター』を参照してください。

一般に、パケット・フィルタ操作ファイアウォールは、明示的に許可されていないサービスを拒否するように構成されます。この方法により、いくつかの潜在的な攻撃を防ぐことができますが、不適切なフィルタ規則の構成による攻撃には無防備なままです。

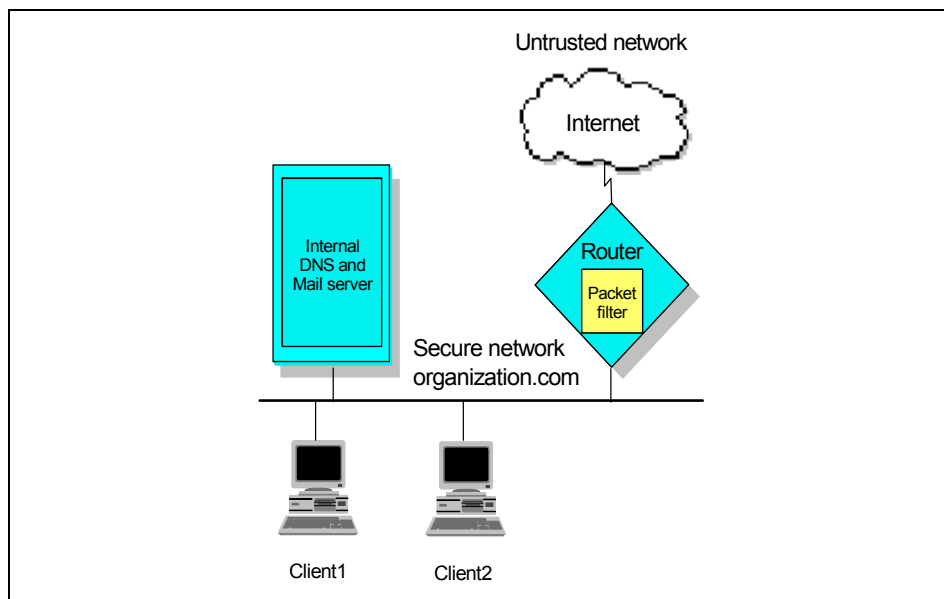


図 275 パケット・フィルタ操作ファイアウォール

このフィルタでは、内部ネットワーク上のいくつかのホストに対し、外部ネットワークから直接アクセスすることができます。そのようなホストは、独自の認証メカニズムを持ち、攻撃に備えて定期的に更新する必要があります。

21.3.6.2 デュアル・ホーム・ゲートウェイ・ファイアウォール

デュアル・ホーム・ホストには、少なくとも 2 つのネットワーク・インターフェースが備えられているため、少なくとも 2 つの IP アドレスがあります。このファイアウォールでは IP 転送ができないため、2 つのイン

ターフェースの間の IP トラフィックはすべて、ファイアウォールで中断されます (図 276 を参照してください)。したがって、ファイアウォールを通してパケットを送る場合、関連するプロキシまたは SOCKS サービスを経由する以外に方法がありません。パケット・フィルタ操作ファイアウォールとは異なり、デュアル・ホーム・ゲートウェイ・ファイアウォールを使うと、不明なサービスからの攻撃がある場合に、それらはブロックされます。デュアル・ホーム・ゲートウェイは、特別に許可されていないすべてのアクセスを拒否する方法を実装しています。

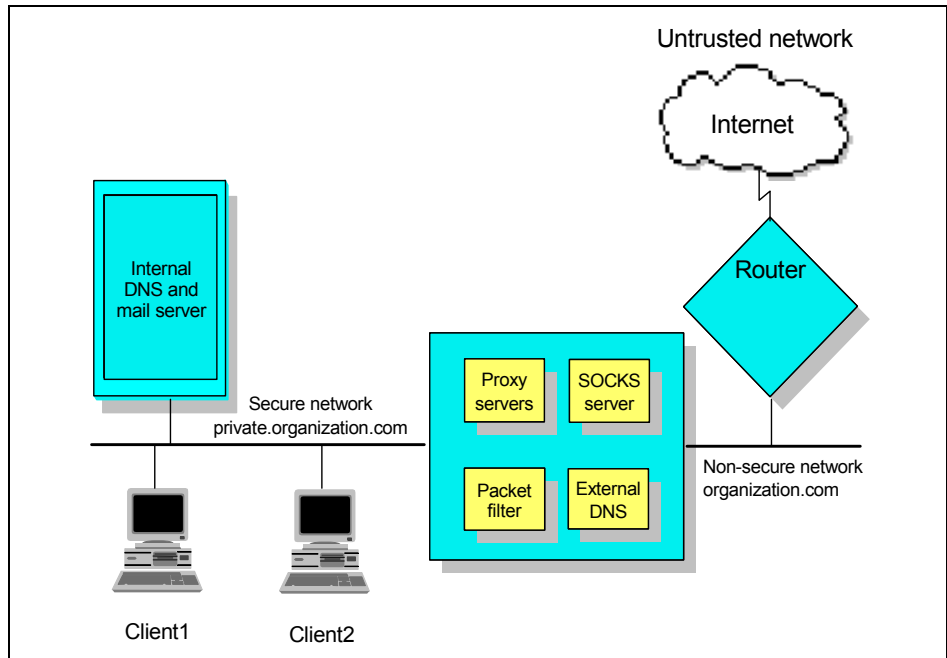


図 276 デュアル・ホーム・ファイアウォール

内部ユーザーと外部ユーザーの両方にアクセスできるように情報サーバー (Web サーバーや FTP サーバーなど) を配置する必要がある場合、サーバーを、保護されたネットワーク内にインストールするか、もしくは、相対的に保護は弱くなりますが、ファイアウォールとルーターの間にインストールできます。ファイアウォールの外側にインストールする場合、そのファイアウォールでは、保護ネットワークの内側から情報サーバーへアクセスできるように、関連したプロキシ・サービスが必要になります。情報サーバーが、ファイアウォールとルーターの間にインストールされる場合、ルーターは、パケット・フィルタ操作ができ、その結果に応じて構成できるものでなければなりません。このタイプのファイアウォールは、選別ホスト・ファイアウォールと呼ばれ、718 ページの 21.3.6.3 『選別ホスト・ファイアウォール』で説明されています。

21.3.6.3 選別ホスト・ファイアウォール

このタイプのファイアウォールは、パケット・フィルタ操作ルーターとアプリケーション・レベル・ゲートウェイとで構成されています。アプリケーション・レベル・ゲートウェイを含むホストは、要塞ホストと呼ばれます。ルーターは、信頼されていないすべてのトラフィックを要塞ホストへ転送し、場合によっては、情報サーバーへ転送するよう構成されています(図 277 を参照)。内部ネットワークは要塞ホストと同じサブネット上にあるため、セキュリティ・ポリシーにより、内部ユーザーは外部ネットワークへ直接にアクセスでき、内部ユーザーにプロキシ・サービスを使わせて外部ネットワークへアクセスさせることができます。このことは、ルーターが要塞ホストからのアウトバウンド・トラフィックのみを受け入れるようにルーター・フィルタ操作規則を構成することによって実現できます。

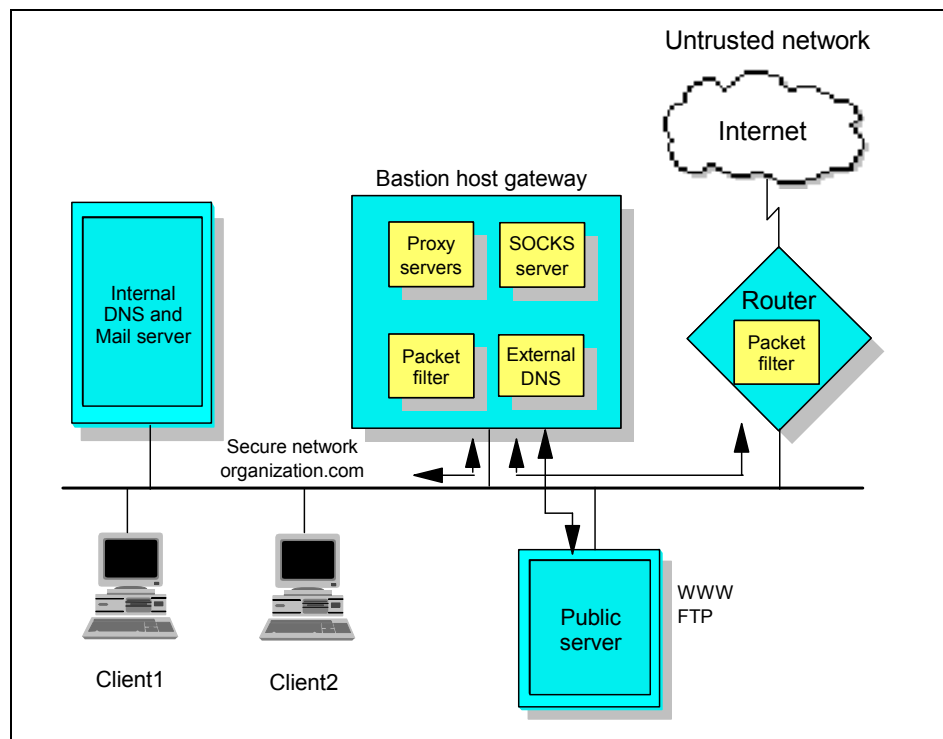


図 277 選別ホスト・ファイアウォール

この構成では、情報サーバーをルーターと要塞ホストの間に置くことができます。前述のように、セキュリティ・ポリシーによって、外部ユーザーと内部ユーザーのどちらかが情報サーバーへ直接にアクセスするのか、または要塞ホスト経由で情報サーバーへアクセスするかどうかが決定されます。強力なセキュリティが必要であれば、内部ネットワークから情報サーバーへ

のトラフィックと、外部ネットワークから情報サーバーへのトラフィックの両方を、要塞ホスト経由にすることができます。

この構成では、要塞ホストは標準のホストとすることができ、さらに強力に保護されたファイアウォール・システムが必要であれば、デュアル・ホーム・ホストとすることができます。この場合には、情報サーバーへの内部トラフィックや、ルーターを経由した外部への内部トラフィックはすべて、自動的にデュアル・ホーム・ホストのプロキシ・サーバーを通過することになります。要塞ホストは、外部からアクセスできる唯一のシステムになります。この要塞ホストへのログオンを許可されるユーザーはいません。そのようにしなければ、侵入者がシステムにログオンし、ファイアウォールをバイパスするように構成を変更する可能性があります。

21.3.6.4 選別サブネット・ファイアウォール (非武装地帯)

このタイプのファイアウォールは、2つのパケット・フィルター操作ルーターと1つの要塞ホストとで構成されています。選別サブネット・ファイアウォールでは、さまざまなファイアウォール・タイプの中でも最高レベルのセキュリティが提供されます (図 278 を参照してください)。このことは、外部ネットワークと内部ネットワークの間に非武装地帯 (DMZ) を作ることで実現します。このようにすると、外部ルーターは、外部から要塞ホスト (情報サーバーなど) へのアクセスのみを許可し、内部ルーターは、内部ネットワークから要塞ホストへのアクセスのみを許可します。ルーターでは、すべてのインバウンド・トラフィックおよびアウトバウンド・トラフィックを要塞ホスト経由にします。このようにすると、侵入者は、内部ネットワークに到達するまでに、3つの個別のシステムを突破しなければならないので、セキュリティは強力です。

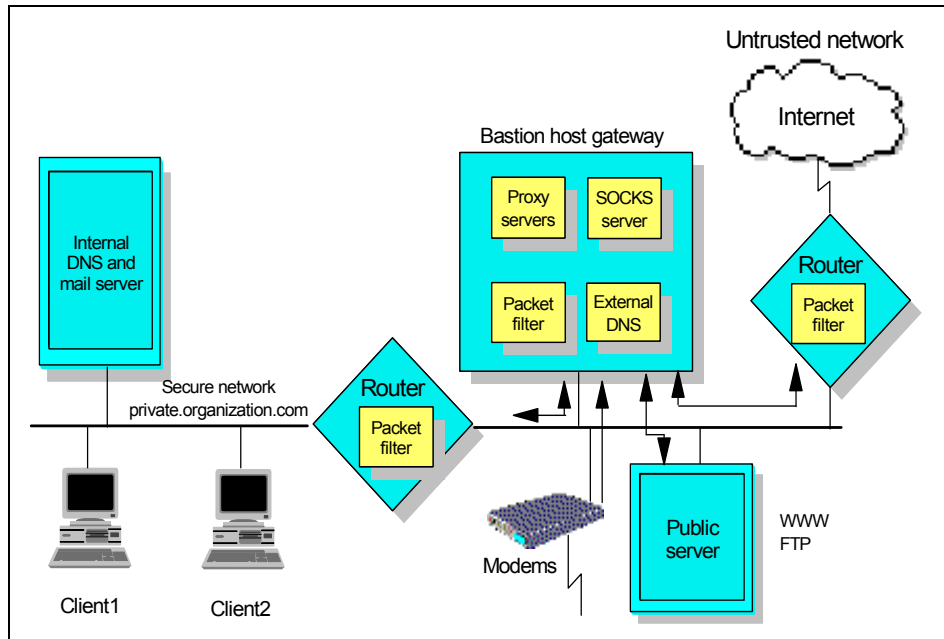


図 278 選別サブネット・ファイアウォール

DMZ の大きな利点の 1 つとして、ルーターは、外部ネットワーク上にあるシステムと内部ネットワーク上にあるシステムの両方が要塞ホストを使うようにするので、要塞ホスト側をデュアル・ホーム・ホストにする必要はないことを挙げられます。このようにすると、デュアル・ホーム・ホストで実現するよりも、より高速なスループットが得られます。もちろん、これは複雑なので、ルーター構成が不適切であれば、セキュリティ問題がいくらか生じる可能性はあります。

21.4 ネットワーク・アドレス変換 (NAT)

このセクションでは、ネットワーク・アドレス変換 (NAT) を説明します。NAT は、IP マスカレードとしても知られます。これは、内部 IP アドレスと公式に割り当てられた外部アドレスとのマッピングを提供します。

元々 NAT は、IP アドレスの不足問題に対する短期的なソリューションとして提案されたものです。また、過去に多くの組織が、インターネットの接続性が必要になることを予測せずに、ローカルに割り当てた IP アドレスを使ってきました。

NAT は、RFC 3022 で定義されています。

21.4.1 NAT の概念

NAT のアイデアは、プライベート・ネットワーク内の少数のホストだけが、そのネットワークの外部と通信するという事実に基づいています。ホストと通信する必要がある場合にのみ、公式な IP アドレス・プールから IP アドレスが各ホストに割り当てられる際、必要とされる公式アドレスの数は少なくて済みます。

専用のアドレス範囲か非公式のアドレスを持ち、インターネット上のホストと通信するネットワークには、NAT をソリューションとすることができます。このことは実際は、ほとんどの場合、ファイアウォールを実装することでも実現できるものです。そのため、プロキシまたは SOCKS サーバーを使うことによってインターネットと通信するクライアントは、それぞれのアドレスをインターネットに公開していません。ですから、そのようなアドレスを変換する必要はありません。しかし、何らかの理由でプロキシと SOCKS が利用できないか、特定の要件を満たせない場合には、NAT を使うことにより、内部ホスト・アドレスを公開しなくても、内部ネットワークと外部ネットワークとの間のトラフィックを管理できます。

専用 IP アドレス・スペースに基づいた内部ネットワークがあり、ユーザーは、アプリケーション・ゲートウェイが存在しないアプリケーション・プロトコルを使いたい場合を考えてみてください。唯一のオプションは、内部ネットワーク内のホストとインターネット上のホストの間に IP レベルの接続性を確立することです。インターネットのルーターは、IP パケットをルーティングして専用 IP アドレスに戻す方法を知らないので、ソース IP アドレスとしての専用 IP アドレスが付けられた IP パケットをルーター経由でインターネットに送信する方法はありません。

図 279 に示されているように、NAT は、発信パケットの IP アドレスを見て、公式に割り当てられたグローバル・アドレスに動的に変換します。着信パケットの場合は、割り当てられたアドレスを内部アドレスに変換します。

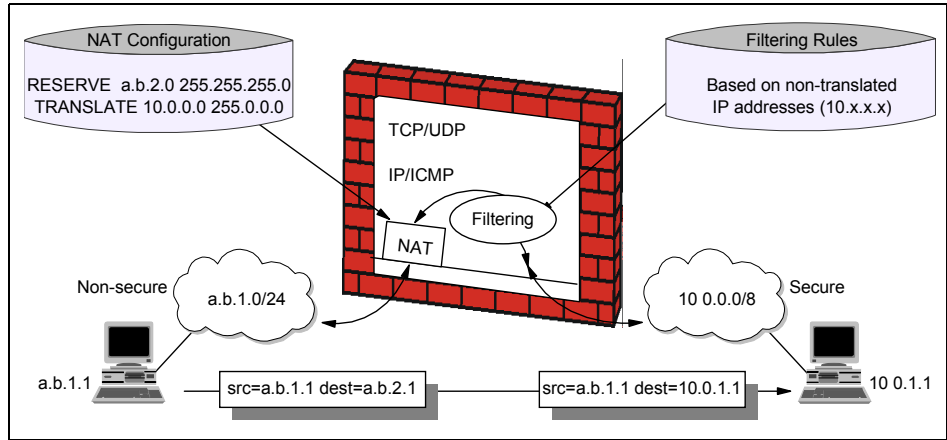


図 279 ネットワーク・アドレス変換 (NAT)

IP パケットを相互に交換する 2 つのホスト (1 つは保護ネットワークにあり、もう 1 つは無保護ネットワークにある) 側から見ると、NAT は、2 つのネットワーク・インターフェースの間で IP パケットを転送する標準の IP ルーターのように見えます (図 280 を参照してください)。

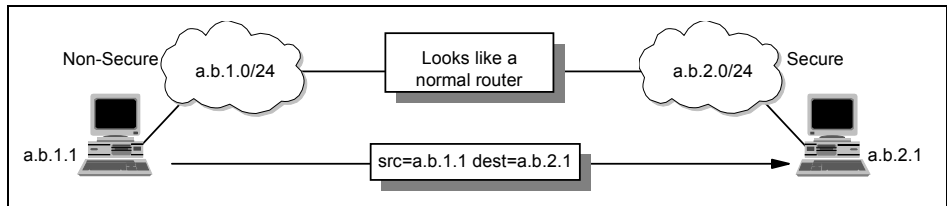


図 280 無保護ネットワークから見た NAT

21.4.2 変換メカニズム

発信 IP パケットごとに、NAT 構成規則によってソース・アドレスが調べられます。規則とソース・アドレスとが一致していれば、そのアドレスはアドレス・プールからのグローバル・アドレスに変換されます。事前定義されたアドレス・プールには、変換用に NAT で使えるアドレスが含まれています。着信パケットが NAT によって使われていれば、そのような着信パケットごとに宛先アドレスが調べられます。ここで一致すれば、そのアドレスは元の内部アドレスに変換されます。図 281 には、NAT の構成が示されています。

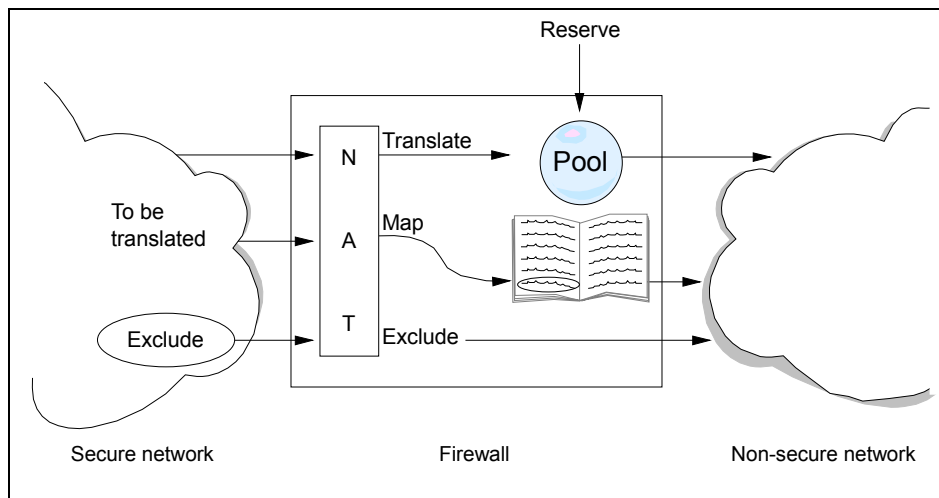


図 281 NAT の構成

NAT が、IP パケットのアドレスを変換する場合、チェックサムも調整されます。FTP パケットの場合、このパケットには、パケットのデータ内にアドレスが含まれている可能性があるため、作業はいくらか難しくなります。たとえば、FTP PORT コマンドには、ASCII 形式の IP アドレスが含まれています。このようなアドレスについても正確に変換する必要があり、その結果に応じて、チェックサムの更新および TCP シーケンスと確認通知の更新も行う必要があります。

NAT は、NAT を使うシステム側からは通常の IP ルーターのように見えます。ルーティング・テーブルを機能させるには、IP ネットワーク設計時に、2 つ以上の IP ネットワークまたはサブネットをルーター経由で接続するようにしてアドレスを選択しなければなりません。NAT IP アドレスは、個別のネットワークまたはサブネットからのものでなければならず、無保護ネットワーク内の他のネットワークまたはサブネットに関してははっきり明示する必要があります。無保護ネットワークがインターネットであれば、NAT アドレスは公衆ネットワークまたはサブネットからのものでなければなりません。つまり、NAT アドレスは IANA によって割り当てられる必要があります。

割り当てられたアドレスはプールで予約しておき、必要なときに使うようにします。保護ネットワークからの接続が確立された際、NAT は NAT プールから空いている次の公衆アドレスを選び、要求している保護ホストに割り当てることができます。NAT は、所定の時点で、どの内部 IP アドレスがどの外部 IP アドレスに割り当てられたのかを追跡し、外部ネットワークから受け取った応答を、対応する保護 IP アドレスに割り当てられるようにします。

NAT が IP アドレスを要求時に割り当てる場合、利用できる IP アドレスのプールに外部 IP アドレスを戻す時期を知っていなければなりません。IP レベルでは接続のセットアップまたは取り外しはないので、保護 IP アドレスと NAT 無保護 IP アドレス間の関連が必要なくなるときを判別するときに NAT が使うことのできるものは、IP プロトコル自体には何もあります。TCP は接続専用のプロトコルであるため、TCP ヘッダーから接続状況の情報（接続が終了しているかいないか）を入手することができますが、UDP にはそのような情報は含まれません。したがって、タイムアウト値（外部 IP アドレスを空いている NAT プールへ戻すまでに関連をアイドル状態にしておく期間を NAT に指示するタイムアウト値）を構成する必要があります。一般に、このパラメーターのデフォルト値は 15 分です。

ネットワーク管理者は、すべての保護ホストで NAT を使えるのかどうかを NAT に指示する必要もあります。このことは、対応する構成コマンドを使って行えます。無保護ネットワーク内のホストが、接続ネットワーク内のホストへの接続を開始しなければならない場合、どの無保護 NAT アドレスがどの保護 IP アドレスに一致するのかに関して、前もって NAT を構成する必要があります。したがって、静的な割り当てを定義して、無保護ネットワークから内部ネットワーク内の特定ホストへの接続を許可するようにします。外部名サーバーには、たとえば、保護ネットワーク内のコンピューター上で稼働するメール・ゲートウェイの項目が存在する場合があります。この外部名サーバーは、静的に割り当てられた IP アドレス（外部アドレス）への内部メール・ゲートウェイの公衆ホスト名を解決し、リモート・メール・サーバーは、接続要求をこの IP アドレスに送信します。その要求が無保護インターフェース上の NAT に到着すると、NAT は自分の割り当て規則を参照し、指定した無保護公衆 IP アドレスと保護 IP アドレスとの間に静的な割り当てがあるかどうかを確認します。ある場合には、その IP アドレスを変換し、その IP パケットを内部メール・ゲートウェイに続く保護ネットワークに転送します。

保護 IP アドレスへ静的に割り当てられた無保護 NAT アドレスは、NAT が必要に応じて使うことのできる無保護アドレスのプールに属するものとして指定したアドレスと、オーバーラップするものであってはならないことに注意してください。

21.4.3 NAT の制限

NAT は、IP ヘッダー内の IP アドレスの場合によく機能します。アプリケーション・プロトコルによっては、IP パケットのアプリケーション・データ部分の IP アドレス情報を交換しますが、NAT は一般的に、そのようなアプリケーション・プロトコルの IP アドレスの変換を処理することができません。現在のところ、ほとんどの実装では、FTP プロトコルを処理します。アプリケーション・データ内に IP 情報を含む特定アプリケーション向けの NAT の実装は、標準の NAT 実装よりも洗練されていることに注意してください。

NAT のもう一つの重要な制限は、NAT は、IP パケット内のいくつかのアドレス情報を変更するということです。終端間 IPsec 認証を使う場合、データグラム内のいずれかのビットに何らかの変更が加えられると、ソースに生成された整合性検査値は無効にされるため、アドレスが変更されたパケットは、AH プロトコルの下での整合性検査に必ず失敗します。IPsec プロトコルでは、以前に NAT によって扱われたアドレス問題に対するソリューションがいくつか提案されているため、所定の仮想プライベート・ネットワークを構成するすべてのホストが固有の（公衆）IP アドレスをグローバルに使う場合、NAT は必要なくなります。アドレスを隠すことは、IPsec のトンネル・モードで実現できます。ある企業が自分のイントラネット内で専用のアドレスを使う場合、IPsec のトンネル・モードでは、それらのアドレスが公衆インターネット内からクリア・テキストで示されることのないように設定することができます。そのため、NAT は必要がなくなります。（IPsec および VPN の詳細については、725 ページの 21.5 『IP Security Architecture (IPsec)』と 786 ページの 21.11 『仮想プライベート・ネットワーク (VPN) の概要』を参照してください。）

21.5 IP Security Architecture (IPsec)

このセクションでは、IPsec フレームワークとその 3 つの主要なコンポーネント（認証ヘッダー (AH)、暗号ペイロード (ESP)、および Internet Key Exchange (IKE)）を詳しく調べます。ヘッダー形式、特定の暗号機能、および異なるモードのアプリケーションが扱われます。

IPsec は、IP パケットに整合性検査、認証、暗号化、および再生保護を追加します。これは、終端間セキュリティーの場合と、ゲートウェイ間の保護トンネルを作成する場合に使われます。

IPsec は、相互に運用できるように設計されています。正常に実装されると、これをサポートしないネットワークおよびホストに影響を及ぼすことがなくなります。IPsec は、現在の暗号アルゴリズムとは別のものです。新しいアルゴリズムが使えるようになると、それらに適合させることができます。また、IPv4 と IPv6 のどちらでも機能します。実際、IPsec は IPv6 の必須コンポーネントです。

IPsec は、最新かつ最高の暗号アルゴリズムを使っています。IPsec プロトコルで使うためにアルゴリズムを特別に実装することは、*変形*と呼ばれることがあります。たとえば、ESP によって使われる DES アルゴリズムは、ESP DES-CBC 変形と呼ばれます。変形は、プロトコルのように、RFC で公開されます。

21.5.1 概念

2 つの主要な IPsec 概念、つまりセキュリティ・アソシエーションとトンネルを明らかにする必要があります。続くいくつかのセクションでは、これらの概念を説明します。

21.5.1.1 セキュリティー・アソシエーション

セキュリティ・アソシエーション (SA) の概念は、IPsec の基本です。SA とは、2 つの IPsec システム間の単方向 (シンプレックス) 論理接続のことで、以下の 3 つの組み合わせによって固有に識別されます。

`<Security Parameter Index, IP destination address, security protocol>`

メンバーの定義は、以下のとおりです。

Security Parameter Index (SPI) これは、同じ宛先アドレスおよびセキュリティ・プロトコルを持つ異なる SA を識別するときに使われる 32 ビットの値です。SPI は、セキュリティ・プロトコル (AH または ESP) のヘッダーに入れられて運ばれます。SPI には、SA の作成者によって定義された、ローカルな重みのみあります。1 ~ 255 の範囲の SPI 値は、Internet Assigned Numbers Authority (IANA) によって予約済みです。0 の SPI 値は、ローカル実装固有の目的のためだけに使う必要があります。RFC 2406 は、0 の値を伝送してはならないと述べています。一般に SPI は、SA の確立時に宛先システムによって選択されます。

IP destination address このアドレスは、単一キャスト、ブロードキャスト、またはマルチキャスト IP アドレスになります。しかし現在は、SA 管理メカニズムは、単一キャスト・アドレス用にのみ定義されています。

Security protocol これは、AH か ESP のいずれかになります。

SA は、その SA のプロトコルのモードに応じて、トランスポートまたはトンネルという 2 つのモードのいずれかになります。これらのプロトコル・モードの説明は、この章の後の部分に載せられています。

SA はシンプレックスであるため、2 つの IPsec システム間の双方向通信の場合は、それぞれの方向に 1 つずつ、定義された 2 つの SA が存在していなければなりません。

1 つの SA は、AH または ESP のいずれか (両方ではない) を使うことにより、伝送されるトラフィックに対してセキュリティー・サービスを提供します。つまり、AH と ESP の両方に保護されなければならない接続の場合、それぞれの方向に、2 つの SA を定義しなければならないということです。この場合、その接続を定義する SA のセットのことを、SA バンドルといいます。バンドルの SA は、同じエンドポイントで終了する必要はありません。たとえば、モバイル・ホストは、それ自体とファイアウォールとの間では AH SA を使い、ファイアウォールの背後ではホストに展開されるネストされた ESP SA を使えます。

IPsec 実装は、SA に関連した以下の 2 つのデータベースを保守します。

Security Policy Database (SPD)

Security Policy Database は、ソース、宛先、インバウンドかアウトバウンドか、といった要因に応じて、IP トラフィックに対して提供するセキュリティー・サービスを指定します。ここには、インバウンド・トラフィックとアウトバウンド・トラフィックとで区別したポリシー項目の番号付きリストが含まれています。これらの項目では、いくつかのトラフィックが IPsec 処理をう回しなければならないこと、いくつかのトラフィックを廃棄しなければならないこと、そして残りのトラフィックを IPsec モジュールで処理しなければならないことを指定している可能性があります。このデータベースの項目は、ファイアウォール規則やパケット・フィルターのようなものです。

Security Association Database (SAD) Security Association Database には、各 SA についてのパラメーター情報 (AH アルゴリズムと鍵または ESP アルゴリズムと鍵、シーケンス番号、プロトコル・モード、および SA 存続期間) が含まれます。アウトバウンド処理の場合、SPD 項目は SAD 内の項目を示します。つまり、SPD

が、所定の packets に使う SA を決定するということです。インバウンド処理の場合、packets の処理方法を決定するときには、SAD が参考にされます。

注：

IPsec 実装のユーザー・インターフェースは、通常は、これらのデータベースをもっと簡単な方法で隠したり表示します。

21.5.1.2 トンネリング

トンネリングまたはカプセル化は、packets 交換ネットワークにおいて一般的な技法です。このことは、ある packets を新しい packets にラッピングすることが関係します。すなわち、元の packets に新しいヘッダーが付加されるということです。図 282 に示されているように、元の packets 全体が、新しい packets のペイロードになります。

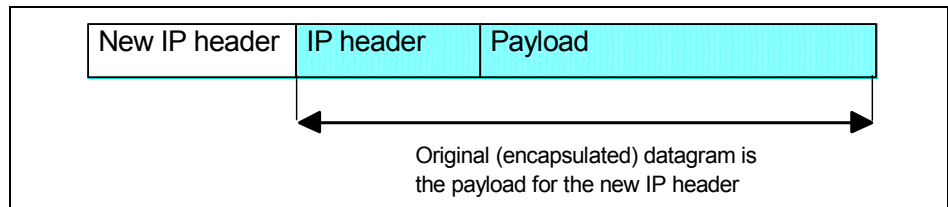


図 282 IP トンネリング

一般に、トンネリングは、あるプロトコルのトラフィックを、そのプロトコルを直接にサポートしないネットワークを経由して伝送するときに使われます。たとえば、NetBIOS や IPX を IP にカプセル化して、TCP/IP WAN リンク経由で伝送できます。IPsec の場合、IP は、カプセル化された packets のヘッダーを含めて全体的に保護するという少し違った目的のために、IP によってトンネリングされます。カプセル化された packets が暗号化されている場合、侵入者は、たとえばその packets の宛先アドレスを見つけることができません。(トンネリングを使わなくても実現可能です。) プライベート・ネットワークの内部構造をこのような方法で隠すことができます。

トンネリングでは、途中で元の packets の中間処理が必要です。外部ヘッダーに指定された宛先 (通常は IPsec ファイアウォールかルーター) は、トンネリングされた packets を受け取り、元の packets を取り出し、そしてそれを最終的な宛先へ送信します。処理のオーバーヘッドは、余分のセキュリティで補われます。

IP トンネリングの特筆すべき利点は、公衆インターネットを介する 2 つのイントラネット間 (グローバルで固有のアドレスが必要) で、専用 IP アドレスを持つパケットを交換できるということです。カプセル化されたヘッダーはインターネット・ルーターによって処理されないため、グローバルに割り当てられたアドレスを持たなければならないのは、トンネル (ゲートウェイ) のエンドポイントだけです。エンドポイントの背後にあるイントラネット内のホストには、専用アドレス (たとえば、10.x.x.x) を割り当てることができます。グローバルで固有な IP アドレスは不足しつつあるため、このような相互接続方式が重要になっています。

注:

IPsec トンネリングは、RFC 2003 *IP Encapsulation within IP* をモデルにしています。これは元々はモバイル IP (モバイル・ホストが、リモートまたは外部サブネットに付加された場合でも、ホーム IP アドレスを保持できるようにするアーキテクチャー) 向けに設計されたものです。655 ページの 19.1 『モバイル IP の概要』を参照してください。

21.5.2 認証ヘッダー (AH)

AH は、IP データグラムの整合性と認証を実現するために使われます。再生保護も可能です。再生保護サービスを使うかどうかはオプションですが、これはすべての IPsec 準拠システムに実装されていなければならないものです。このサービスはコネクションレスであるため、パケット単位で機能します。AH は、トランスポート・モードとトンネル・モードの 2 つのモードで使われます。

AH は、できるだけ多くの IP データグラムを認証します。トランスポート・モードでは、IP ヘッダー内のいくつかのフィールドが途中で変更されますが、受信側はその値を予測することはできません。これらのフィールドは、*可変*と呼ばれ、AH によって保護されません。可変 IPv4 フィールドは、以下のとおりです。

- Type of Service (TOS) (サービス・タイプ)
- Flags (フラグ)
- Fragment Offset (フラグメント・オフセット)
- Time To Live (TTL) (存続時間)
- Header checksum (ヘッダー・チェックサム)

これらのフィールドを保護する必要がある場合、トンネリングを使うようにします。IP パケットのペイロードは不変であると思なされるため、必ず AH によって保護されます。

AH は、IANA によって割り当てられたプロトコル番号 51 で識別されます。AH ヘッダーのすぐ前にあるプロトコル・ヘッダー (IPv4、IPv6、または拡張子) では、そのプロトコル (IPv4) または Next header (次のヘッダー) (IPv6、または拡張子) フィールドに、この値が示されています。

AH 処理は、フラグメント化されていない IP パケットに対してのみ適用されます。しかし、AH が適用されている IP パケットは、中間ルーターでフラグメント化することができます。この場合、まず宛先がパケットを再アセンブルしてから、それに AH 処理を適用します。(Fragment Offset (フラグメント・オフセット) フィールドがゼロ以外であるか、More Fragments (その他のフラグメント) ビットを設定することによって) フラグメントであると示される IP パケットが AH 処理に渡される場合、そのパケットは廃棄されます。これにより、いわゆる **重複フラグメント攻撃**を防ぐことができます。この攻撃は、フラグメント再アセンブル・アルゴリズムを誤用してねつ造パケットを作成し、ファイアウォールを通過させようとするものです。

認証に失敗するパケットは、廃棄されて上方のレイヤーに送達されることはありません。この操作モードを使うと、偽のパケットでホストやゲートウェイをあふれさせることにより通信をブロックすることを狙った、サービス妨害攻撃が成功するチャンスを激減させることができます。

21.5.2.1 AH の形式

AH の形式については、RFC 2402 に述べられています。

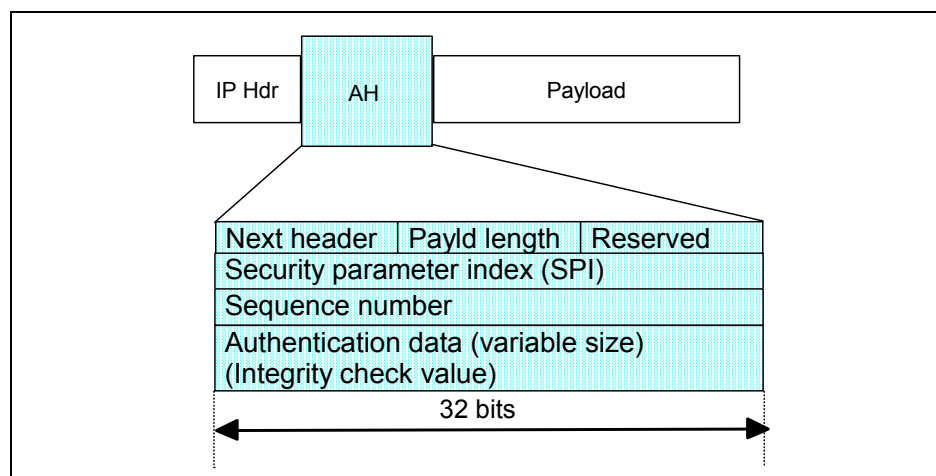


図 283 AH の形式

図 283 には、IP パケットでの認証ヘッダーのフィールドの位置が示されています。フィールドは、以下のとおりです。

Next header (次のヘッダー)

次のヘッダー t は、8 ビットのフィールドで、続くヘッダーのタイプを示します。このフィールドの値は、Internet Assigned Numbers Authority (IANA) によって出された最新の Assigned Numbers RFC で定義されている IP プロトコル番号のセットから選択したものです。つまり、IP ヘッダー・プロトコル・フィールドは 51 にセットされ、このプロトコル・フィールドから消された値は、AH の次のヘッダー・フィールドに入れられるということです。

Payload Length (ペイロードの長さ)

このフィールドは、8 ビットの長さで、32 ビット・ワードで表現される AH ヘッダーの長さから 2 を引いたものが示されます。これは、全体として、IP パケットの実ペイロード長に関連するものではありません。デフォルト・オプションを使う場合、値は 4 です (認証データの 3 つの 32 ビット固定ワードと 3 つの 32 ビット・ワードを足してそこから 2 を引いたもの)。

Reserved (予約)

このフィールドは、将来の利用のために予約済みです。その長さは 16 ビットで、ゼロにセットされています。

Security Parameter Index (SPI)

このフィールドは 32 ビットの長さです。定義については、726 ページの『Security Parameter Index (SPI)』これは、同じ宛先アドレスおよびセキュリティ・プロトコルを持つ異なる SA を識別するときに使われる 32 ビットの値です。SPI は、セキュリティ・プロトコル (AH または ESP) のヘッダーに入れられて運ばれます。SPI には、SA の作成者によって定義された、ローカルな重みのみあります。1 ~ 255 の範囲の SPI 値は、Internet Assigned Numbers Authority (IANA) によって予約済みです。0 の SPI 値は、ローカル実装固有の目的のためだけに使う必要があります。RFC 2406 は、0 の値を伝送してはならない

と述べています。一般に SPI は、SA の確立時に宛先システムによって選択されます。』を参照してください。

Sequence Number (シーケンス番号)

この 32 ビット・フィールドは、単調に増加するカウンターで、再生保護のために使われます。再生保護はオプションです。ただし、このフィールドは必須です。送信側は必ずこのフィールドを含めますが、これを処理するかどうかは受信側の判断に任されます。SA の確立時には、シーケンス番号はゼロに初期設定されます。SA を使って伝送された最初のパケットには、1 のシーケンス番号が付けられます。シーケンス番号を繰り返すことはできません。したがって、任意に指定した SA で伝送できる IP パケットの最大数は、 $2^{32}-1$ です。最大のシーケンス番号を使ってしまったら、新しい SA が確立され、その結果、新しい鍵が確立されます。送信側では、デフォルトで反再生が利用できるようになります。受信側が使わないことを選択した SA が確立されると、送信側はこのフィールドの値に注意を払う必要がなくなります。

注：

1. 一般には、反再生メカニズムは、人手による鍵管理と同時に使われることはありません。
2. RFC 1826 での元の AH 仕様では、シーケンス番号の概念は論じられていませんでした。したがって、この RFC に基づくさらに古い IPsec 実装には、再生保護が備えられていません。

Authentication Data (認証データ) これは、Integrity Check Value (ICV) を含む可変長フィールドで、IPv4 の場合は 32 ビットに埋め込まれ、IPv6 の場合は 64 ビットに埋め込まれます。各パケットの ICV は、SA 初期設定で選択したアルゴリズムを使って計算されます。名前に示されているように、着信パケットの

整合性を検査するために、受信側によって使われます。

理論的には、ICV を計算するためにどのような MAC アルゴリズムでも使うことができます。仕様では、HMAC-MD5-96 と HMAC-SHA-1-96 がサポートされていなければならないことになっています。以前の RFC 1826 では、鍵付き MD5 が必要でした。実際は、鍵付き SHA-1 も使われます。実装では、通常 2 ～ 4 のアルゴリズムがサポートされています。

ICV 計算を実行するときには、可変フィールドはゼロで埋められると考えられます。

21.5.2.2 AH の使い方

AH は、トランスポート・モードとトンネル・モードの 2 つの方法で使うことができます。

トランスポート・モードでの AH

このモードでは、図 284 に示されているように、認証ヘッダーは IP ヘッダーのすぐ後に挿入されます。データグラムにすでに IPsec ヘッダー（複数可）がある場合、AH はそれらの前に挿入されます。

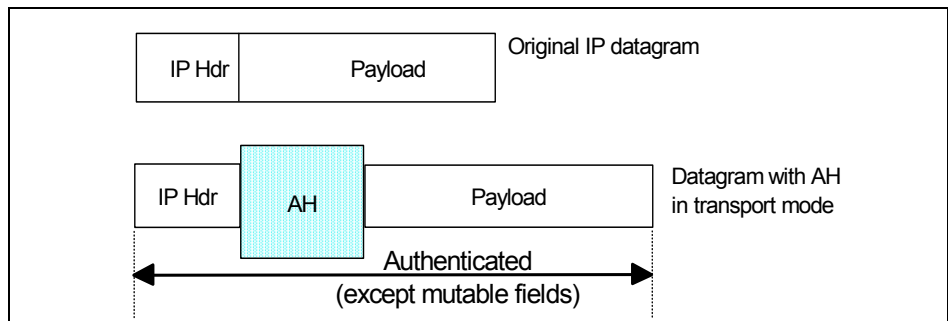


図 284 トランスポート・モードでの認証ヘッダー

トランスポート・モードは、ゲートウェイではなく、ホストによって使われます。ゲートウェイは、トランスポート・モードをサポートするときに必要ではありません。

トランスポート・モードの利点は、処理のオーバーヘッドが少ないということです。欠点は、可変フィールドは認証されないということです。

トンネル・モードでの AH

このモードでは、トンネリングの概念が適用されるので、新しい IP データグラムが構成され、元の IP データグラムはそのペイロードになります。トランスポート・モードでの AH は、結果データグラムに適用されます。図は、図 285 を参照してください。

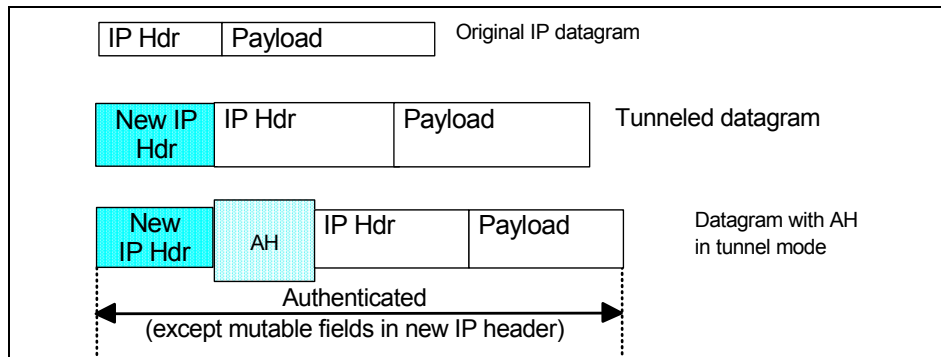


図 285 トンネル・モードでの認証ヘッダー

トンネル・モードは、セキュリティー・アソシエーションのいずれかの終端がゲートウェイである場合には、必ず使われます。したがって、2つのファイアウォールの間では、必ずトンネル・モードが使われます。

ゲートウェイは、トランスポート・モードもサポートする場合があります。このモードは、ゲートウェイがホストとして機能するときに許可されます。つまり、トラフィックがゲートウェイ自体に宛てられている場合です。たとえば、SNMP コマンドは、トランスポート・モードを使ってゲートウェイに送信することができます。

トンネル・モードでは、外部ヘッダーの IP アドレスは、内部ヘッダーのアドレスと同じである必要はありません。たとえば、2つのセキュリティー・ゲートウェイでは、相互に接続するネットワーク間のすべてのトラフィックを認証するために使われる 1つ

の AH トンネルを運用できます。これは、非常に一般的な運用モードです。

トンネル・モードの利点には、カプセル化された IP データグラムをまとめて保護できることと、専用アドレスを使えることなどがあげられます。ただし、このモードに関連して余分の処理オーバーヘッドがあります。

注：

RFC 1825 での元の AH 仕様では、トンネル・モードについて、付加的に述べられているに過ぎず、要件ではありません。このため、その RFC に基づき、AH をトンネル・モードではサポートしない IPsec 実装が存在します。

21.5.2.3 IPv6 の考慮事項

AH は、IPv6 の重要な部品の 1 つです (585 ページの 17.2.2 『拡張ヘッダー』を参照)。IPv6 環境では、AH は終端間ペイロードであると見なされ、ホップ・バイ・ホップ、ルーティング、およびフラグメント化拡張ヘッダーの後に示されます。宛先オプション拡張ヘッダー (複数可) は、認証ヘッダーの前と後のどちらに置いても構いません。図 286 には、一般的な IPv6 パケットでの、トランスポート・モードにおける AH の位置が示されています。* がマークされた拡張ヘッダーが存在する場合には、その位置は変更可能です。

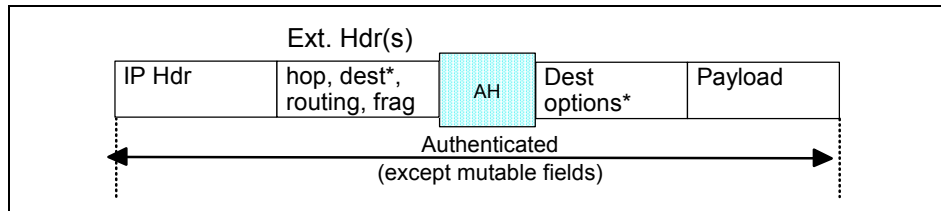


図 286 IPv6 でのトランスポート・モードの AH

IPv6 での AH の詳細については、RFC 2402 を参照してください。

21.5.3 暗号ペイロード (ESP)

ESP は、IP データグラムの整合性検査、認証、および暗号化を実現するために使われます。オプションで再生保護も可能です。これらのサービスはコネクションレスであるため、パケット単位で運用されます。希望するサービスのセットは、SA の確立時に選択できます。しかし、以下のようないくつかの制限が適用されます。

- 整合性検査と認証は一緒に使われます。
- 再生保護は、整合性検査および認証を共に選択する場合にだけ選択できます。
- 再生保護は受信側だけが選択できます。

暗号化は、他のサービスからは独立して選択できます。暗号化を使えるのであれば、整合性検査と認証をオンにしておくことを強くお勧めします。暗号化だけを使う場合、侵入者は、パケットをねつ造して暗号解読攻撃を開始する可能性があります。

(整合性検査を使った) 認証と暗号化はどちらもオプションですが、少なくともどちらか 1 つが必ず選択されています。そうしないと、ESP を使っていることにはなりません。

ESP は、IANA によって割り当てられたプロトコル番号 50 で識別されます。AH ヘッダーのすぐ前にあるプロトコル・ヘッダー (IPv4、IPv6、または拡張子) では、そのプロトコル (IPv4) または Next header (次のヘッダー) (IPv6、または拡張子) フィールドに、この値が示されています。

ESP 処理は、フラグメント化されていない IP パケットに対してのみ適用されます。しかし、ESP が適用されている IP パケットは、中間ルーターでフラグメント化することができます。この場合、まず宛先がパケットを再アセンブルしてから、それに ESP 処理を適用します。(Fragment Offset (フラグメント・オフセット) フィールドがゼロ以外であるか、More Fragments (その他のフラグメント) ビットを設定することによって) フラグメントであると示される IP パケットが ESP 処理に渡される場合、そのパケットは廃棄されます。これにより、729 ページの 21.5.2 『認証ヘッダー (AH)』に述べられている重複フラグメント攻撃を防ぎます。

整合性検査のある暗号化と認証の両方を選択する場合、受信側はまずそのパケットを認証し、このステップが成功した場合にだけ復号に進みます。このモードで運用することにより、計算リソースを節約することができ、サービス妨害攻撃に対するぜい弱さを小さくすることができます。

21.5.3.1 ESP パケット形式

現在の ESP の形式については、RFC 2406 に述べられています。ここには、前の ESP 仕様である RFC 1827 と比較して重要な変更が示されています。このセクションの情報は、RFC 2406 に基づくものです。

ESP パケットの形式は、AH パケットの形式よりさらに複雑です。ESP ヘッダーだけが存在しているわけではなく、ESP トレーラーと ESP 認証データも存在します (図 287 を参照)。ペイロードは、(カプセル化されて) ヘッダー

とトレーラーの間に置かれます。プロトコルの名前はこのことに由来しています。

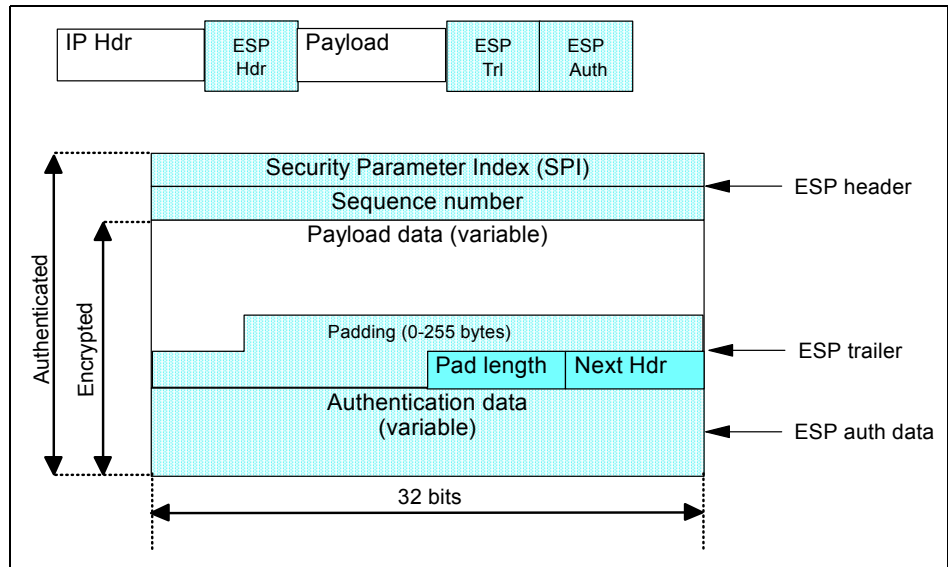


図 287 ESP ヘッダーおよびトレーラー

以下のフィールドは、ESP パケットの一部です。

Security Parameter Index (SPI)

このフィールドは 32 ビットの長さです。定義については、731 ページの『Security Parameter Index (SPI) このフィールドは 32 ビットの長さです。定義については、726 ページの『Security Parameter Index (SPI) これは、同じ宛先アドレスおよびセキュリティ・プロトコルを持つ異なる SA を識別するときに使われる 32 ビットの値です。SPI は、セキュリティ・プロトコル (AH または ESP) のヘッダーに入れられて運ばれます。SPI には、SA の作成者によって定義された、ローカルな重みのみあります。1 ~ 255 の範囲の SPI 値は、Internet Assigned Numbers Authority (IANA) によって予約済みです。0 の SPI 値は、ローカル実装固有の目的のためだけに使う必要があります。RFC 2406 は、0 の値を伝送してはならないと述べ

ています。一般に SPI は、SA の確立時に宛先システムによって選択されます。』を参照してください。』を参照してください。

Sequence Number (シーケンス番号)

この 32 ビット・フィールドは、単調に増加するカウンターです。定義については、732 ページの『Sequence Number (シーケンス番号) この 32 ビット・フィールドは、単調に増加するカウンターで、再生保護のために使われます。再生保護はオプションです。ただし、このフィールドは必須です。送信側は必ずこのフィールドを含めますが、これを処理するかどうかは受信側の判断に任せられます。SA の確立時には、シーケンス番号はゼロに初期設定されます。SA を使って伝送された最初のパケットには、1 のシーケンス番号が付けられます。シーケンス番号を繰り返すことはできません。したがって、任意に指定した SA で伝送できる IP パケットの最大数は、232-1 です。最大のシーケンス番号を使ってしまったら、新しい SA が確立され、その結果、新しい鍵が確立されます。送信側では、デフォルトで反再生が利用できるようになります。受信側が使わないことを選択した SA が確立されると、送信側はこのフィールドの値に注意を払う必要がなくなります。』を参照してください。

注：

1. 一般には、反再生メカニズムは、人手による鍵管理と同時に使われることはありません。
2. RFC 1827 での元の ESP 仕様では、シーケンス番号の概念は論じられていませんでした。したがって、この RFC に基づくさらに古い IPsec 実装には、再生保護が備えられていません。

Payload Data (ペイロード・データ)

ペイロード・データ・フィールドは必須です。これは、次のヘッダー・フィールドに記述されるデータの可変バイト数で構成されています。このフィールドは、SA 確立時に選択した暗号アルゴリズムで暗号化されます。アルゴリズムに初期設定ベクトルが必要な場合、それらのベクトルもここに含まれています。

ESP 仕様によると、CBC モードの DES アルゴリズムをサポートする必要があります (DES-CBC 変形)。IBM 製品では、多くの場合 Triple-DES や CDMF など、他の暗号化アルゴリズムもサポートされています。

Padding (埋め込み)

ほとんどの暗号化アルゴリズムでは、入力データはブロックの整数値でなければなりません。また、結果の暗号テキスト (埋め込み、埋め込み長、および次のヘッダー・フィールドを含む) は、4 バイトの境界で終了しなければならないため、次のヘッダー・フィールドは右寄せされます。このような理由のため、埋め込みが組み込まれています。さらに、元のメッセージの長さを隠すためにも使えます。しかしそのようにすると、逆に有効な帯域幅に影響する可能性があります。埋め込みは、オプションのフィールドです (ただしいくつかのアルゴリズムでは必要)。

注:

暗号化は、ペイロード・データ、埋め込み、埋め込み長、および次のヘッダー・フィールドに対応します。

Pad Length (埋め込み長)

この 8 ビットのフィールドには、先行する埋め込みバイトの数が表示されます。この数は必ず示されていて、値が 0 であれば埋め込みがないことを示します。

Next header (次のヘッダー)

次のヘッダーは、8 ビットの必須フィールドで、ペイロードに伝送されるデータ・タイプ、たとえば、TCP のような高レベルのプロトコル ID が示されます。この値は、IANA によって定義される IP プロトコル番号のセットから選択されます。

Authentication Data (認証データ)

このフィールドは、長さは可変であり、SPI から次のヘッダー・フィールドを含む範囲内の ESP パケット用に計算した ICV を含んでいます。認証データ・フィールドはオプションです。これは、SA の初期設定時に整合性検査と認証が選択された場合にだけ含まれます。

ESP 仕様によると、MD5 を使った HMAC と SHA-1 を使った HMAC の 2 つの認証アルゴリズムがサポートされている必要があります。さらに、IPsec 実装により、より簡素な鍵付きバージョンもサポートされています。

注：

- 1.IP ヘッダーは ICV では対応されません。
- 2.RFC 1827 での元の ESP 仕様では、ESP 内での認証の概念が暗号化変形概念と同時に論じられています。つまり、Authentication Data (認証データ) フィールドは存在せず、暗号化変形のために残されて、最終的に認証されることになりました。

21.5.3.2 ESP の使い方

AH のように、ESP はトランスポート・モードとトンネル・モードの 2 つの方法で使うことができます。

トランスポート・モードでの ESP

このモードでは、図 288 に示されているように、ESP ヘッダーは IP ヘッダーのすぐ後に挿入されます。データグラムにすでに IPsec ヘッダー (複数可) がある場合、ESP ヘッダーはそれらのいずれかの前に挿入されます。ESP トレーラーとオプションの認証データは、ペイロードに追加されます。

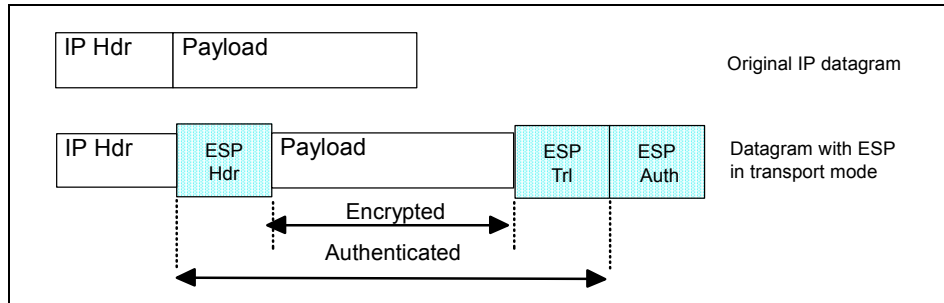


図 288 トランスポート・モードでの ESP

トランスポート・モードでの ESP では、IP ヘッダーのための認証も暗号化も実現しません。ESP 処理で偽のパケットが送達される可能性があるため、これは欠点です。トランスポート・モードの利点は、処理のオーバーヘッドが少ないということです。

AH の場合にそうであったように、トランスポート・モードは、ゲートウェイではなく、ホストによって使われます。ゲートウェイは、トランスポート・モードをサポートするときに必要ではありません。

トンネル・モードでの ESP

予想どおりに、このモードはトンネリングの原則を適用しています。新しい IP パケットは、新しい IP ヘッダーで作成されます。その後、トランスポート・モードで適用されたように、ESP が適用されます。このことは、図 289 に示されています。元のデータグラムは、新しい ESP パケットのペイロード・データになるため、暗号化と認証の両方が選択されていれば、完全に保護されます。ただし、新しい IP ヘッダーは保護されないままです。

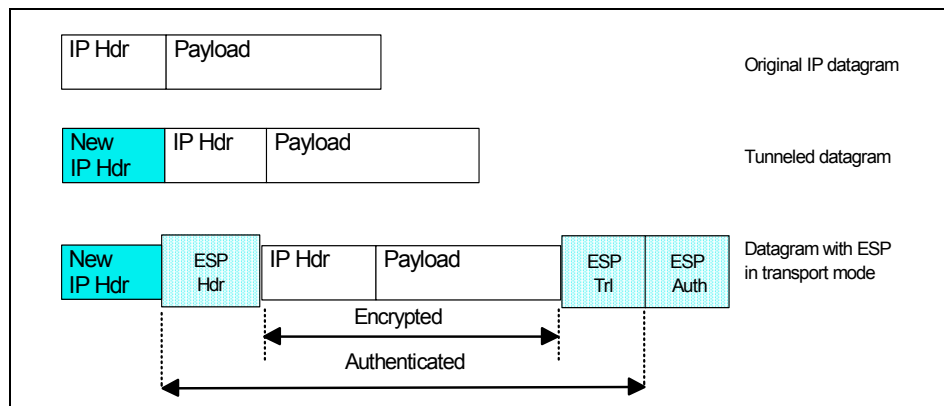


図 289 トンネル・モードでの ESP

トンネル・モードは、セキュリティー・アソシエーションのいずれかの終端がゲートウェイである場合には、必ず使われます。したがって、2つのファイアウォールの間では、必ずトンネル・モードが使われます。

ゲートウェイは、トランスポート・モードもサポートする場合があります。このモードは、ゲートウェイがホストとして機能するときに許可されます。つまり、トラフィックがゲートウェイ自体に宛てられている場合です。たとえば、SNMP コマンドは、トランスポート・モードを使ってゲートウェイに送信することができます。

トンネル・モードでは、外部ヘッダーの IP アドレスは、内部ヘッダーのアドレスと同じである必要はありません。たとえば、2つのセキュリティー・ゲートウェイでは、相互に接続するネットワーク間のすべてのトラフィックを保護するために使われる 1つの ESP トンネルを運用できます。ホストは、トンネル・モードをサポートするときに必要ではありません。

トンネル・モードの利点は、カプセル化された IP データグラムを全体的に保護できることと、専用アドレスを使うことができることがあります。しかし、このモードに関連して余分の処理オーバーヘッドがあります。

21.5.3.3 IPv6 の考慮事項

AH でそうであったように、ESP は IPv6 の重要な部品の 1つです (585 ページの 17.2.2 『拡張ヘッダー』を参照)。IPv6 環境では、ESP は終端間ペイロードであると見なされ、ホップ・バイ・ホップ、ルーティング、およびフラグメント化拡張ヘッダーの後に示されます。宛先オプション拡張ヘッダー (複数可) は、AH ヘッダーの前と後のどちらに置いても構いません。図 290 には、一般的な IPv6 パケットでの、トランスポート・モードにおける AH ヘッダーの位置が示されています。* がマークされた拡張ヘッダーが存在する場合には、その位置は変更可能です。

詳細については、RFC 2406 を参照してください。

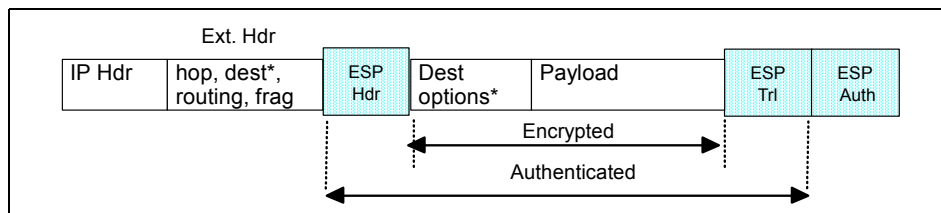


図 290 IPv6 でのトランスポート・モードの ESP

21.5.3.4 2つの認証プロトコル

ESP のセキュリティー・サービスについて知ると、本当に AH の必要性はあるのかと尋ねるユーザーもいるでしょう。ESP 認証は、なぜ IP ヘッダーにも

対応していないのでしょうか。これらの質問に対する公式の回答はありませんが、2つの異なる IPsec 認証プロトコルの存在の正当性を示すいくつかの理由を示します。

- ESP では、実際に使われるかどうかにかかわらず、強力な暗号アルゴリズムを実装する必要があります。いくつかの国では、強力な暗号化に関する制限規定が存在しています。そのような地域で ESP ベースのソリューションをデプロイするとトラブルになる可能性があります。しかし、認証は規制されておらず、AH は世界のどこでも自由に使うことができます。
- 多くの場合、必要なのは認証だけです。認証だけの場合、AH は、ESP と比較して形式がより簡単で処理オーバーヘッドも少ないため、パフォーマンスの点で優れています。このような場合には、AH を使うことに意味があります。
- 2つの異なるプロトコルを持っていると、IPsec ネットワークを経由した、さらにきめの細かい制御が可能になり、セキュリティ・オプションがより柔軟になります。AH と ESP をネストすると、たとえば、両方のプロトコルの威力を組み合わせた IPsec トンネルを実装できます。

21.5.4 IPsec プロトコルの組み合わせ

AH および ESP プロトコルは、単独でも組み合わせても適用できます。各プロトコルには2つのモードがあるので、可能な組み合わせはたくさん存在します。さらに複雑にするには、AH SA と ESP SA とで、同じエンドポイントを使わないようにします。幸いにも、実際の世界のシナリオでは、実際に起こり得るのは、数多くある可能性のうち2つか3つだけです。

注：

RFC 2406 には、各 IPsec 実装でサポートされていなければならない必須の組み合わせが述べられています。他の組み合わせもサポート可能ですが、相互運用性に影響する可能性があります。

726 ページの 21.5.1.1 『セキュリティ・アソシエーション』で、IPsec プロトコルの組み合わせは、SA バンドルで実現することを述べました。

SA バンドルを作成するには、以下の2つの方法があります。

- トランスポートの隣接化：どちらのセキュリティ・プロトコルも、同じ IP データグラムに対してトランスポート・モードで適用されます。この方法は、1 レベルだけの組み合わせの場合に实际的です。
- トンネリングの反復（ネスト）：このセキュリティ・プロトコルは、トンネル・モードで順番に適用されます。各アプリケーションの後に、新しい IP データグラムが作成され、そこに新しいプロトコルが適用されます。こ

の方法には、ネスト・レベルの制限がありません。ただし、4 つ以上のレベルは実際的ではありません。

これらの方法は組み合わせることができます。たとえば、トランスポートを隣接化した IPsec ヘッダーのある IP パケットは、トンネリングをネストすることによって送信することができます。

VPN を設計する場合、IPsec 処理ステージの数を制限する必要があります。観察したところ、3 つのステージが限界で、それ以上の処理をしても利点がありません。ほとんどすべてのケースで、2 つのステージがあれば十分です。

SA が異なるエンドポイントを持つ SA バンドルを作成できるようにするには、最低でも 1 つのレベルのトンネリングを適用する必要がある点に注意してください。トランスポートを隣接化すると、1 つの IP ヘッダーしか存在しないことになるので、複数のソース / 宛先アドレスは不可能になります。

組み合わせて使うときの実際的な原則としては、両方のプロトコル・ヘッダーを持つパケットを受け取ったら、IPsec の処理の順序は、認証してから復号するという順序でなければならないということをあげられます。発信元がはっきりしないパケットの復号に悩む必要がないのは言うまでもありません。

上記の原則に従い、送信側は、アウトバウンド・トラフィックに対して、まず ESP を適用し、その後で AH を適用します。実際、この順序は、トランスポート・モードの IPsec 処理のための明確な要件です。ESP と AH の両方を使っている場合には、ESP 認証をオンにすべきかどうかという新しい疑問が出てきます。いずれにしても、AH はパケットを認証します。答えは簡単です。

ESP 認証をオンにしても、これが有効になるのは ESP SA を AH SA の外側に拡張する場合だけです。たとえば、ESP は終端間で使えますが、AH はリモート・ゲートウェイに到達するだけです。このような場合、ESP 認証を使うことが理にかなっているというだけではなく、イントラネット内のスプーフィング攻撃を防ぐためにも、ESP 認証を使うことを強くお勧めします。

モードに関する限り、トランスポート・モードは、通常は接続のエンドポイント間で使われ、トンネル・モードは、2 つのマシンの少なくとも一方がゲートウェイである場合に、通常はそれらのマシンの間で使われます。

最も簡単なセットアップからより複雑にネストされたセットアップに至るまで、IPsec プロトコルを使ったさまざまな方法を考察しましょう。

21.5.4.1 ケース 1: 終端間のセキュリティ

図 291 に示されているように、2 つのホストがインターネット (またはイントラネット) 経由で接続されています。このホストの間には IPsec ゲート

ウェイはありません。この場合、ESP、AH あるいはその両方を使うことができます。トランスポート・モードとトンネル・モードのいずれかを適用できます。

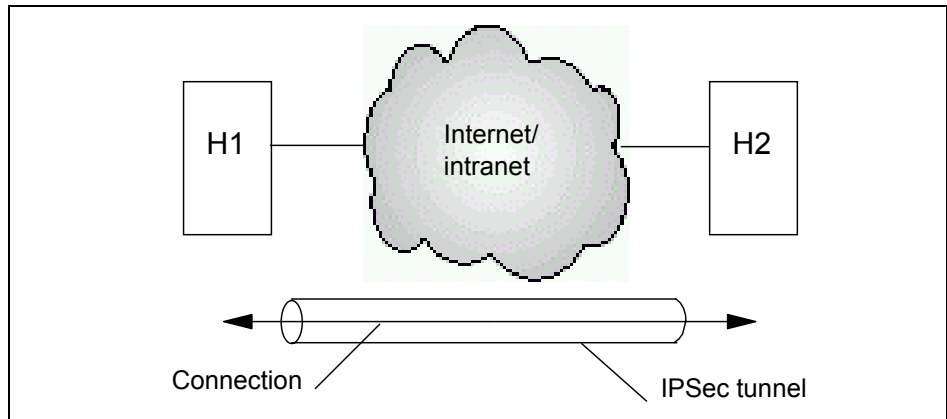


図 291 終端間のセキュリティ

すべての IPsec 実装でサポートされる必要のある組み合わせは、以下のとおりです。

- トランスポート・モード
 - AH 単独
 - ESP 単独
 - ESP の適用後に AH を適用 (トランスポートの隣接化)
- トンネル・モード
 - AH 単独
 - ESP 単独

21.5.4.2 ケース 2: 基本的な VPN サポート

仮想プライベート・ネットワーク (VPN) については、786 ページの 21.11 『仮想プライベート・ネットワーク (VPN) の概要』で説明されています。

図 292 には、最も簡単な IPsec VPN が示されています。ゲートウェイ G1 および G2 は、IPsec プロトコル・スタックを実行しています。イントラネット内のホストでは、IPsec をサポートする必要はありません。

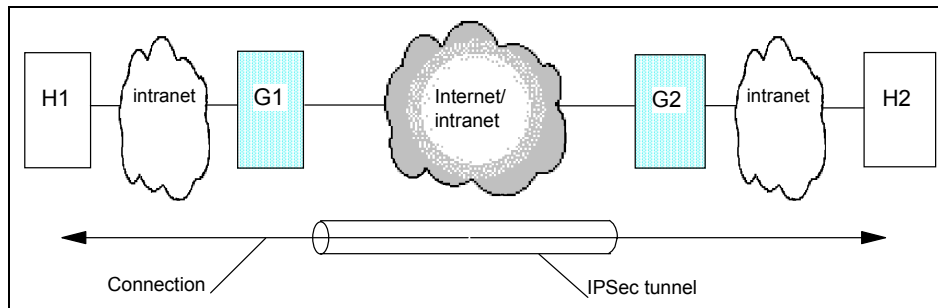


図 292 基本的な VPN サポート

このケースでは、ゲートウェイは、AH と ESP のいずれかに対するトンネル・モードだけをサポートする必要があります。

ゲートウェイ間で組み合わされたトンネル

ゲートウェイは、AH トンネルと ESP トンネルのいずれかをサポートする必要がありますが、しばしば、両方の IPsec プロトコルの機能を組み合わせたゲートウェイの間にトンネルを設けることが望ましい場合があります。

IBM IPsec 実装では、このタイプの組み合わされた AH-ESP トンネルをサポートしています。ヘッダーの順序は、トンネル・ポリシーを設定することにより、ユーザー側で選択できます。

ゲートウェイ間の組み合わされたトンネルとは、トンネリングを反復することではありません。トンネルを含む SA バンドルには、同一のエンドポイントがあるため、トンネリングを反復することでは十分ではありません。代わりに、ある IPsec プロトコルをトンネル・モードに適用し、別のプロトコルをトランスポート・モードに適用することにより、概念的に組み合わされた AH-ESP トンネルであると考えることができます。同様の方法として、元のデータグラムを IP トンネリングし、その後でそこにトランスポートの隣接化 IPsec 処理を適用することがあります。結果は、下記の図に示されていますが、トンネル・ポリシーで設定した順序のとおり、先頭に外部 IP ヘッダー、その次に IPsec ヘッダーが続き、それから元の IP パケットがきます。これが、2 つの IBM ファイアウォール間の組み合わされた AH-ESP トンネルでのパケット形式です。

注：

ESP 認証データは、IBM ファイアウォールの初期の実装には登場していません。

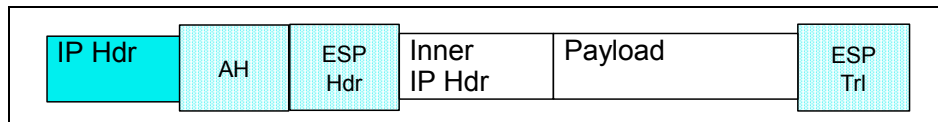


図 293 組み合わされた AH-ESP トンネル

21.5.4.3 ケース 3: VPN サポートを使った終端間セキュリティ

このケースは、ケース 1 とケース 2 を組み合わせたもので、関係するマシンで新しい IPsec 要件が必要になるわけではありません (図 294 を参照)。ケース 2 との大きな違いは、このケースでは IPsec をサポートするためにホストも必要になるという点です。

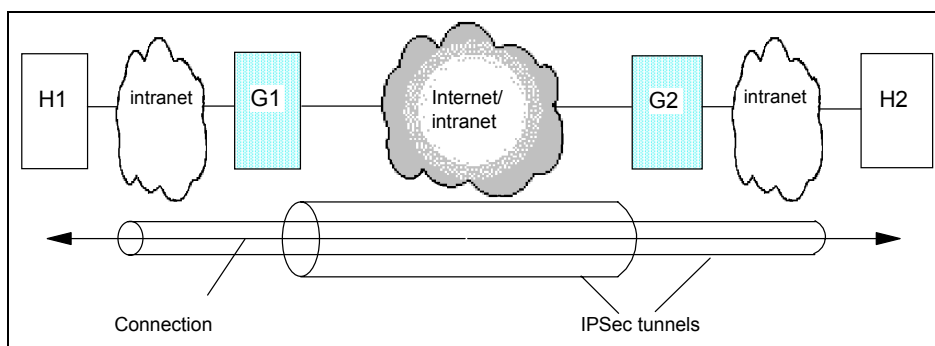


図 294 VPN サポートを使った終端間セキュリティ

一般的なセットアップでは、ゲートウェイは AH をトンネル・モードで使い、ホストは ESP をトランスポート・モードで使います。セキュリティが強化されたバージョンでは、ゲートウェイ間で組み合わせた AH-ESP トンネルを使うことができます。このようにして、最終的な宛先アドレスが暗号化され、インターネットを行き来するパケット全体が認証され、二重に暗号化されたデータが運ばれます。これは、3 つのステージによる IPsec の処理がコスト的に有効な唯一のケースですが、パフォーマンス的には影響は非常に大きくなります。

ESP トランスポートの AH トンネリング

AH のトンネリングを使ってトランスポート・モードの ESP トラフィックを保護するときの一般的な組み合わせを詳しく考察しましょう。

図 295 では、この組み合わせを実現する方法を詳しく示しています。図 294 のホスト H1 が IP パケットをホスト H2 に送信するとします。以下のことが生じます。

1. ホスト H1 は IP パケットを構成し、それに ESP トランスポートを適用します。次に、H1 はそのデータグラムをゲートウェイ G1 (宛先アドレスは H2) に送信します。
2. ゲートウェイ G1 は、このパケットが G2 にルーティングされるようにします。IPsec データベース (SPD および SAD) G1 を参照するとすぐに、パケットを送信する前にトンネル・モードの AH を適用しなければならないことが分かります。ここでは、必要なカプセル化が行われます。これで、IP パケットは、宛先として G2 のアドレスを持つことになり、最終的な宛先 H2 がカプセル化されます。
3. ゲートウェイ G2 は、AH トンネリングされたパケットを受け取ります。このパケットは、それ自身が宛先になっているため、データグラムを認証し、外部ヘッダーを取り除きます。G2 は、ペイロードは宛先が H2 になっている別の IP パケット (H1 で送信したパケット) であると見なすため、H2 へ転送します。G2 は、このパケットに ESP ヘッダーがあるかどうかは確認しません。
4. 最後に、H2 がパケットを受け取ります。これは宛先なので、ESP トランスポート処理が適用され、元のペイロードが取り出されます。

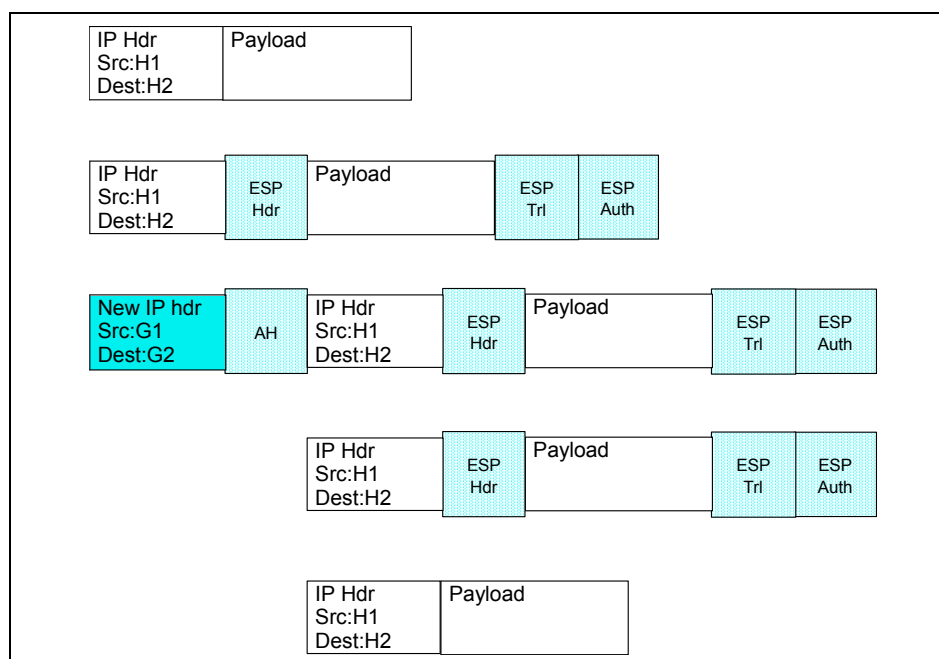


図 295 IPsec プロトコルのネスト

21.5.4.4 ケース 4: リモート・アクセス

図 296 に示されているこのケースは、ファイアウォールで保護されている組織内のサーバーに到達するためインターネットを使うリモート・ホストに適用されます。このリモート・ホストでは、一般的に ISP への PPP ダイアルイン接続が使われます。

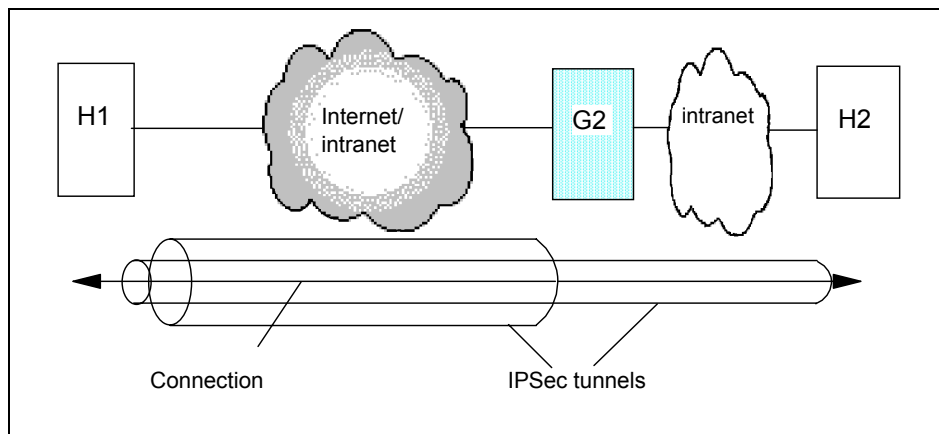


図 296 リモート・アクセス

リモート・ホスト H1 とファイアウォール G2 との間で必要なのは、トンネル・モードだけです。選択方法は、ケース 2 の場合と同じです。ホスト自体の間では、ケース 1 で選択したように、トンネル・モードとトランスポート・モードのいずれかを使えます。

一般的なセットアップは、H1 と G2 の間ではトンネル・モードの AH を使い、H1 と H2 の間ではトランスポート・モードの ESP を使います。トンネル・モードでの AH をサポートしていない以前の IPsec 実装では、これを実装することはできません。

また、リモート・ホスト H1 とゲートウェイ G2 との間に組み合わせた AH-ESP トンネルを作成することも一般的です。このケースでは、H1 は 1 つの SA バンドルを使うだけでイントラネット全体にアクセスできるため、図 296 に示されているセットアップを使う場合、1 つの SA バンドルを含む 1 つのホストにアクセスできるだけです。

21.5.5 Internet Key Exchange (IKE) プロトコル

以前は ISAKMP/Oakley と呼ばれていた Internet Key Exchange (IKE) フレームワークは、セキュリティー・アソシエーションの自動化されたネゴシエーションや、暗号鍵の自動化された生成およびリフレッシュをサポートしています。マシンに構成せずに、あるいは手動での構成なしに、このような機能

を実行するための機能は、どのような企業スケールの IPsec デプロイメントにとっても重大な要素となっています。

鍵交換およびメッセージ更新の詳細を説明する前に、いくつかの説明が必要です。

- Internet security association and key management protocol (ISAKMP)

セキュリティー・アソシエーション (ネゴシエーション、変更、削除) および鍵の管理を定義するフレームワークで、鍵生成データおよび認証データを交換するためのペイロードも定義します。ISAKMP そのものは、鍵交換プロトコルを定義することはありませんが、そこに備えられているフレームワークを、ネットワーク、トランスポート、またはアプリケーション・レイヤーのセキュリティー・メカニズムに適用し、さらにそれ自体に適用することができます。

- Oakley

セキュリティー・アソシエーションの鍵マテリアルを交換し更新するために、ISAKMP フレームワークで使える鍵交換プロトコル。

- Domain of interpretation (DOI)

特定の環境向けに ISAKMP フレームワークで使われるプロトコルのセットの定義。また、SA 属性およびペイロードの内容の構文、暗号変形のネーム・スペースなどに関してこれらのプロトコルと共用される一般的な定義。IPsec に関し、DOI は、IP で使うために ISAKMP をインスタンス化します。

- Internet key exchange (IKE)

ISAKMP の一部と Oakley および SKEME 鍵交換プロトコルの一部を使い、IPsec AH および ESP プロトコル用、また ISAKMP 自体用の鍵およびセキュリティー・アソシエーションを管理するプロトコル。

21.5.5.1 プロトコルの概要

ISAKMP では、すべての情報交換に対して、だれも鍵マテリアルを盗み見ることができないように、暗号化と認証の両方を行う必要があります。鍵マテリアルは、認証されたパーティー間でのみ交換されます。ISAKMP の手順は、鍵の初期設定を扱うことになり、セキュリティーが存在しないと思われるリンク経由で実行できなければならないため、このような取り決めが必要になります。

さらに、ISAKMP 方式は、以下のようないくつかの既知の漏えいに対する保護の提供という明確な目的を持って設計されています。

- サービス妨害：メッセージは、プロセッサに集中した暗号運用を実行しなくても、無効なメッセージをすぐに識別して拒否するために使える固有の cookies 付きで構成されます。

- 中継ユーザー：メッセージの削除、メッセージの変更、送信側へ戻されたメッセージの反映、以前のメッセージの再生、および意図しない受信側へのメッセージの転送といった一般的な攻撃に対して保護を提供します。
- Perfect Forward Secrecy (PFS): 以前の鍵が漏えいしても、他の鍵を破るために役立つ手掛かりは得られません。このことは、他の鍵を破ろうとする試みが、鍵の暗号が漏えいする前に発生しても後に発生しても同じです。つまり、リフレッシュされたそれぞれの鍵は、以前の鍵に依存することなく、生成されるということです。

IKE では、以下の認証方式が定義されています。

1. 事前共有鍵
2. デジタル署名 (DSS および RSA)
3. 公開鍵暗号化 (RSA および改訂された RSA)

暗号ベースのソリューションの堅ろう性は、選択した暗号アルゴリズムの実際の内容よりも、鍵を秘密にしておくことにより強く依存しています。そのため、IETF IPsec Working Group は、極めて堅固な Oakley 交換プロトコルを設定しています。ここでは、2 つのフェーズがある方法が使われます。

フェーズ 1

この一連のネゴシエーションは、ユーザーのデータ・トラフィックを保護するすべての暗号鍵が派生されるマスター秘密を確立します。最も一般的なケースでは、システム間で ISAKMP セキュリティー・アソシエーションを確立し、次のフェーズ 2 ネゴシエーションで流される ISAKMP メッセージを保護するのに使われる鍵を確立するために、公開鍵暗号方式が使われます。フェーズ 1 は、ISAKMP メッセージそのもののために保護一式を確立することにのみ関連したのですが、ユーザー・データを保護するために、何らかのセキュリティ・アソシエーションまたは鍵を確立することはありません。

フェーズ 1 では、暗号運用はプロセッサにもっとも負荷が集中するものですが、まれにしか実行する必要はなく、続くフェーズ 2 での複数の交換をフェーズ 1 での 1 回の交換でサポートできます。経験法則として、フェーズ 1 のネゴシエーションは、一日に 1 回か場合によっては一週間に 1 回実行すればよいのに対し、フェーズ 2 のネゴシエーションは 2、3 分おきに 1 回実行します。

フェーズ 2

フェーズ 2 の交換は、フェーズ 1 でネゴシエーションされたセキュリティ保護一式がアクティブになった後のみ使われるので、それほど複雑ではありません。やり取りするシステムの組が、ユーザー・データ交換を保護するセキュリティ・アソシエーションと鍵をネゴシエーションします。フェーズ 2 の ISAKMP メッセージは、フェーズ 1 で生成された ISAKMP セキュリ

ティー・アソシエーションによって保護されます。フェーズ 2 のネゴシエーションは、一般的にフェーズ 1 よりも頻繁に行われます。たとえば、フェーズ 2 ネゴシエーションの一般的な適用として、2、3 分おきに 1 回暗号鍵をリフレッシュすることをあげられます。

永続 ID

IKE プロトコルには、前もってリモート・ホストの IP アドレスが分かっている場合のソリューションも備えられています。ISAKMP を使うと、リモート・ホストは、名前や E メール・アドレスなどの永続 ID によって、それ自身を識別できるようになります。次に ISAKMP フェーズ 1 交換は、公開鍵暗号方式を使って、リモート・ホストの永続 ID を認証します。

- ・証明書は、永続 ID と公開鍵とのバインディングを作成します。したがって、ISAKMP の証明書ベースのフェーズ 1 メッセージ交換では、リモート・ホストの永続 ID を認証できます。
- ・ISAKMP メッセージそのものは IP データグラム内で運ばれるため、ISAKMP パートナー（たとえば、ファイアウォールまたは宛先ホスト）は、リモート・ホストの動的な IP アドレスを、認証された永続 ID と関連付けることができます。

21.5.5.2 IKE でのセキュリティー・アソシエーションの初期設定

このセクションでは、安全にやり取りすることを望む 2 つのシステム間で、ISAKMP/Oakley プロトコルを使ってまずセキュリティー・アソシエーションを確立し、鍵を交換する方法を概説します。

このセクションの残りの部分では、関係するパーティーを Host-A および Host-B とします。Host-A は、ISAKMP フェーズ 1 交換の起動側で、Host-B は応答側です。はっきりさせることが必要であれば、メッセージ交換内のさまざまなフィールドのソースを識別するときに、添え字 A または B が使われます。

21.5.5.3 IKE フェーズ 1 - ISAKMP セキュリティー・アソシエーションの

ISAKMP メッセージそのものを保護するセキュリティー・アソシエーションは、フェーズ 1 交換時に設定されます。「コールド」（前の鍵がないか、Host-A と Host-B の間で SA がネゴシエーションされた）で開始しているため、フェーズ 1 交換では、ISAKMP Identity Protect 交換 (Oakley メイン・モードとも言う) が使われます。交換を完了するには、以下の 6 つのメッセージが必要です。

- ・メッセージ 1 および 2 は、セキュリティー・アソシエーションの特性をネゴシエーションします。メッセージ 1 および 2 は、最初のフェーズ 1 交換で自由に流れ、認証されません。
- ・メッセージ 3 および 4 は、当座のもの（ランダムな値）を交換し、さらに Diffie-Hellman 交換を実行してマスター鍵 (SKEYID) を確立します。メッ

セージ 3 および 4 は、最初のフェーズ 1 交換で自由に流れ、認証されません。

- メッセージ 5 および 6 は、パーティーの実体を相互に認証するために必要な情報を交換します。メッセージ 5 および 6 のペイロードは、メッセージ 1 ～ 4 で確立された暗号化アルゴリズムと鍵マテリアルによって保護されます。

フェーズ 1 メッセージと交換された情報の詳細を続けます。

IKE フェーズ 1、メッセージ 1

Host-A は開始パーティーなので、クリア・テキストの ISAKMP メッセージ (メッセージ 1) を作成し、Host-B に送信します。ISAKMP メッセージそのものは、UDP パケットのペイロードとして運ばれますが、UDP パケットは、通常の IP データグラムのペイロードとして運ばれます (図 297 を参照)。

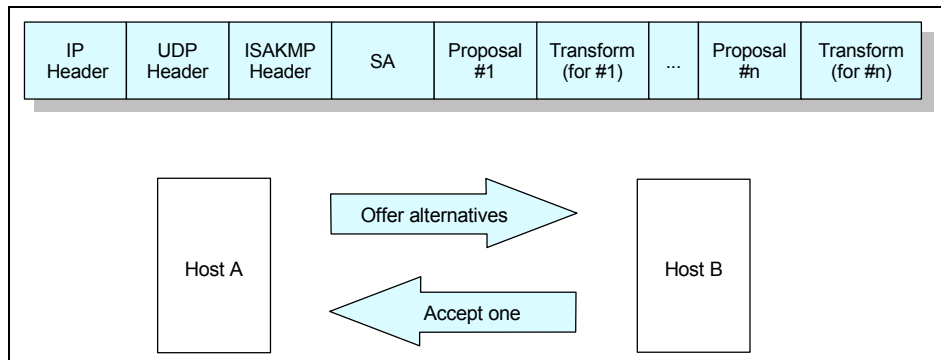


図 297 ISAKMP フェーズ 1 交換のメッセージ 1

IP ヘッダーに置くソースおよび宛先アドレスは、それぞれ Host-A (起動側) と Host-B (応答側) のアドレスです。UDP ヘッダーは、宛先ポートが 500であることを指定します。これは、ISAKMP プロトコルで使うために割り当てられたものです。UDP パケットのペイロードは、ISAKMP メッセージそのものを運びます。

メッセージ 1 では、起動側である Host-A は、応答側である Host-B による考慮事項のために、1 つ以上の保護一式のセットを提案します。そのため、ISAKMP メッセージでは、ペイロードに少なくとも以下のフィールドが含まれます。

ISAKMP header (ISAKMP ヘッダー)

メッセージ 1 の ISAKMP ヘッダーには、メイン・モードの交換タイプが示され、0 の Message ID (メッセージ ID) が含まれています。Host-A は、Responder Cookie (応答側 Cookie) フィールドを 0

にセットし、Initiator Cookie (起動側 Cookie) に選択したランダムな値を記入します。これは、Cookie-A と示されます。

Security Association (セキュリティー・アソシエーション)

Security Association (セキュリティー・アソシエーション) フィールドは、Domain of Interpretation (DOI) を示します。ホストでは、ホスト間で IPsec プロトコルを実行する予定であるため、この DOI は単なる IP です。

Proposal Payload (提案ペイロード)

Host-A の Proposal Payload (提案ペイロード) では、プロトコル PROTO_ISAKMP を指定し、SPI の値を 0 にセットします。

注 :

ISAKMP フェーズ 1 メッセージの場合、Proposal Payload (提案ペイロード) 内の実際の SPI フィールドは、ISAKMP セキュリティー・アソシエーションを示すために使われることはありません。フェーズ 1 の間、どちらもゼロ以外の値でなければならない <Initiator Cookie, Responder Cookie> の値の組の代わりに、ISAKMP SA が示されます。Responder Cookie (応答側 Cookie) は Host-B によって生成されていないため、ISAKMP SA が明白に示されることはありません。

Transform Payload (変形ペイロード)

Transform Payload (変形ペイロード) は、KEY_OAKLEY を指定します。KEY_OAKLEY 変形の場合、Host-A は関係のある属性、すなわち使う予定の認証方式、使う予定の疑似乱数機能、および使う予定の暗号化アルゴリズムを指定する必要があります。

注 :

メッセージ 1 には複数の提案を組み込むことができます。

IKE フェーズ 1 メッセージ 2

メッセージ 1 では、Host-A は、ISAKMP 交換を保護するために使う予定の保護一式の候補を 1 つ以上提案しました。Host-B は、メッセージ 2 を使い、どの保護 (存在する場合) をサポートするかを示します。Host-A が 1 つのオプ

ションだけを提案した場合、Host-B が行う必要のある作業は、その提案が受け入れられるかどうかを確認することだけです。

IP ヘッダーに置くソースおよび宛先アドレスは、それぞれ Host-B (応答側) と Host-A (起動側) のアドレスです。UDP ヘッダーは、宛先ポートが 500 であることを指定します。これは、ISAKMP プロトコルで使うために割り当てられたものです。UDP パケットのペイロードは、ISAKMP メッセージそのものを運びます。

メッセージの内容は、以下のとおりです。

ISAKMP Header (ISAKMP ヘッダー)

メッセージ 2 の ISAKMP ヘッダーには、メイン・モードの交換タイプが示され、0 の Message ID (メッセージ ID) が含まれています。Host-B は、Responder Cookie (応答側 Cookie) フィールドを、Cookie-B を呼び出すランダムな値にセットし、メッセージ 1 の Cookie-A フィールドで受信した値を Initiator Cookie (起動側 Cookie) フィールドにコピーします。値の組 <Cookie-A, Cookie-B> は、ISAKMP セキュリティー・アソシエーションの SPI の役割を果たします。

Security Association (セキュリティー・アソシエーション)

Security Association (セキュリティー・アソシエーション) フィールドは、Domain of Interpretation (DOI) を示します。ホストでは、ホスト間で IPsec プロトコルを実行する予定であるため、この DOI は単なる IP です。

Proposal Payload (提案ペイロード)

Host-B の Proposal Payload (提案ペイロード) では、プロトコル PROTO_ISAKMP を指定し、SPI の値を 0 にセットします。

Transform Payload (変形ペイロード)

Transform Payload (変形ペイロード) は、KEY_OAKLEY を指定します。KEY_OAKLEY 変形の場合、Host-A によって提供された提案で受け入れた属性は、適切なフィールドにコピーされます。

この時点で、ISAKMP セキュリティー・アソシエーションのプロパティーは、Host-A と Host-B に同意されています。ISAKMP SA の実体は、<Cookie-A, Cookie-B> の組と同じものにセットされています。しかし、Host-A および Host-B であると自称するパーティーの実体は、正式に検査されていません。

IKE フェーズ 1、メッセージ 3

フェーズ 1 ISAKMP 交換の 3 番目のメッセージは、最終的に暗号鍵を派生するときの情報の交換で始まります (図 298 を参照)。

重要:

メッセージそのものが実際の暗号鍵を運ぶことはありません。その代わり、鍵をローカルに派生するときに、Host-A と Host-B によって使われる入力を選びます。

ISAKMP ペイロードは、以下の 2 つのタイプの情報を交換するときに使います。

Diffie-Hellman 公開値

起動側からの Diffie-Hellman 公開値 g^x 。公開値の指数 x は、秘密にしておかなければならない秘密値です。

Nonce (臨時)

起動側からの臨時 N_i 。(Nonce (臨時) は、非常に厳格ないくつかの数学的ガイドラインに従って、乱数であると見なされる値の架空の名前です。)

ID

RSA 公開鍵を認証のために使う場合、臨時は他のパーティーの公開鍵で暗号化されます。いずれのパーティーの ID でも同じですが、この段階で交換されません。

改訂された RSA 公開鍵を使った認証が行われる場合、KE および ID ペイロードは、臨時から派生される秘密鍵とメッセージ 1 と 2 で同意された暗号化アルゴリズムで暗号化されるため、1 つの CPU に集中した公開鍵の運用を避けることができます。

証明書は、公開鍵の認証時でも公開鍵認証によるハッシュ値の場合でも、任意に交換できます。

これらの値は、それぞれ Key Exchange (鍵交換) フィールド、Nonce (臨時) フィールド、および ID フィールドで運ばれます。

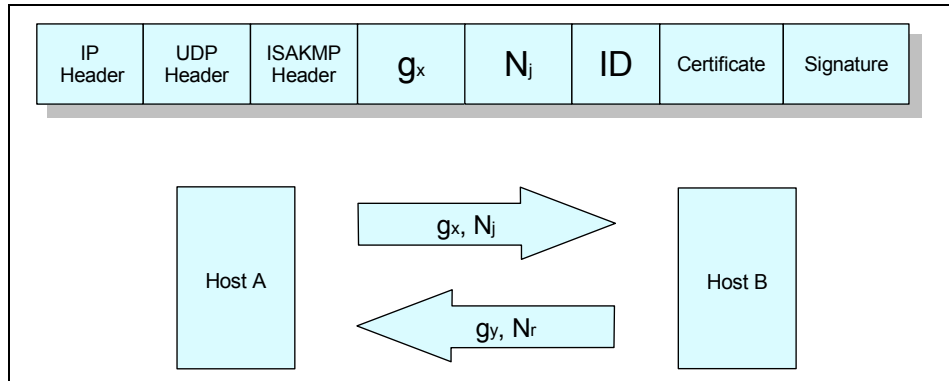


図 298 ISAKMP フェーズ 1 交換のメッセージ 3

IKE フェーズ 1、メッセージ 4

Host-A から Diffie-Hellman 公開値および臨時を受け取ったら、Host-B は、Host-A に対して独自の Diffie-Hellman 公開値 (応答側からの g^y) および臨時 (応答側からの N_r) を送信して応答します。

鍵の生成 (フェーズ 1)

この時点で、各ホストは、2 つの臨時 (N_i および N_r) の値を知っています。さらに、独自の秘密 Diffie-Hellman 値 (x および y) も知っていて、パートナーの公開値 (g^x または g^y) も知っています。したがって、それぞれの側で、複合値 g^{xy} を構成できます。最終的に、それぞれの側が、起動側 cookie と応答側 cookie の値を知ることになります。

これらすべての情報ビットを指定すると、それぞれの側は、以下の数量に関して同等の値を独立して計算することができます。

- SKEYID: このビット集合は、ときどき、鍵マテリアルと呼ばれることがあります。これは、処理の後の部分で、実際の暗号鍵を派生するときの未加工入力 that 提供されるためです。鍵マテリアルは、同意された鍵付き疑似乱数機能 (prf) を既知の入力に適用することによって取得されます。

- デジタル署名認証の場合：

$$\text{SKEYID} = \text{prf}(N_i, N_r, g^{xy})$$

- 公開鍵での認証の場合：

$$\text{SKEYID} = \text{prf}(\text{hash}(N_i, N_r), \text{CookieA}, \text{CookieB})$$

- 事前共有鍵を使った認証の場合：

$$\text{SKEYID} = \text{prf}(\text{pre-shared key}, N_i, N_r)$$

- SKEYID の値を計算したら、それぞれの側は、2 つの暗号鍵およびいくつかの余分な鍵マテリアルの生成作業に進みます。
 - SKEYID_d は、ユーザー・トラフィックを保護するために ISAKMP 以外の SA で使われる鍵を派生するために、引き続いてフェーズ 2 で使われる鍵マテリアルです。

$$\text{SKEYID\_d} = \text{prf}(\text{SKEYID}, g^{xy}, \text{CookieA}, \text{CookieB}, 0)$$

- SKEYID_a は、ISAKMP メッセージを認証するために使われる鍵です。

$$\text{SKEYID\_a} = \text{prf}(\text{SKEYID}, \text{SKEYID\_d}, g^{xy}, \text{CookieA}, \text{CookieB}, 1)$$

- SKEYID_e は、ISAKMP 交換を暗号化するために使われる鍵です。

$$\text{SKEYID\_e} = \text{prf}(\text{SKEYID}, \text{SKEYID\_a}, g^{xy}, \text{CookieA}, \text{CookieB}, 2)$$

プロトコルのこの時点では、Host-A および Host-B の両方とも、ISAKMP 交換を保護するために使う同等の認証鍵および暗号鍵を派生します。さらに、同等の鍵マテリアルも派生していて、そこから ISAKMP ネゴシエーションのフェーズ 2 でユーザー・データを保護するための鍵を派生します。しかし、この時点では、2 つのパーティーの実体は相互に認証されていません。

IKE フェーズ 1、メッセージ 5

フェーズ 1 フローのこの時点で、2 つのホストは実体情報を相互に交換し、相互に認証します。図 299 に示されているように、ISAKMP メッセージは、実体ペイロード、署名ペイロード、およびオプションの証明書ペイロードを運びます。Host-A は、メッセージ 5 を使って Host-B に情報を送信します。これにより、Host-B は Host-A を認証できるようになります。

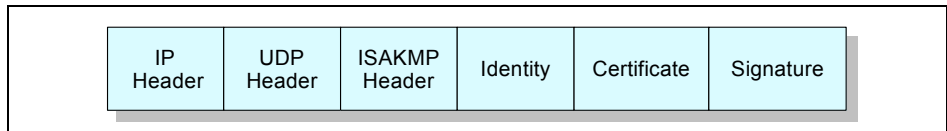


図 299 ISAKMP フェーズ 1 交換のメッセージ 5

実際の証明書が Certificate Payload (証明書ペイロード) フィールドに存在する場合、受信側は、信頼されている認証局の有効な署名が記されていること

を確認した後で、その情報を直接に使うことができます。メッセージに証明書がなければ、いくつかの実装方式を使って証明書を入手することは、受信側の役割です。たとえば、LDAP のようなプロトコルを使って信頼されている認証局へ照会を送信すること、保護 DNS サーバーを照会すること、以前に使われた証明書をそれぞれ対応した ID 値に割り当てる保護ローカル・キャッシュを維持すること、ISAKMP 認証要求メッセージをピアに送信し、証明書をすぐに要求側へ送信させることが可能です。

注：

証明書を入手する方法は、ローカル・オプションであって、IKE の一部として定義されているわけではありません。特に、当該証明書がまだ有効で失効していないことを確認することは、受信側のローカルな役割です。

留意しておくべき以下のいくつかのポイントがあります。

- 処理のこのステージでは、ISAKMP ペイロードはすべて、フェーズ 1 であってもフェーズ 2 であっても、(メッセージ 1 および 2 でネゴシェーションされた) 暗号化アルゴリズムや (メッセージ 3 および 4 の情報から派生した) 鍵を使って暗号化されます。しかし、ISAKMP ヘッダーそのものは、クリアに伝送されます。
- フェーズ 1 では、IPsec の ESP プロトコルは使われません。つまり、ESP ヘッダーは存在しません。受信側は、ISAKMP ヘッダーの Flags (フラグ) フィールドにある Encryption Bit (暗号化ビット) を使い、そのメッセージに暗号化が適用されているかどうかを判別します。フェーズ 1 交換の SPI として機能する値の組 <CookieA, CookieB> により、メッセージを復号するときに使う正しいアルゴリズムおよび鍵へのポインターが実現します。
- デジタル署名を使う場合、ISAKMP メッセージそのものには適用されません。その代わりに、Host-A と Host-B の両方で利用できる情報のハッシュに適用されます。
- 実体ペイロードで運ばれる実体は、必ずしもソース IP アドレスに対する関係を持つ必要はありません。しかし、実体ペイロードで運ばれる実体は、認証が使われている場合に、その認証を適用した実体でなければなりません。

Host-A (起動側) は、次のハッシュ機能を生成し、その結果を Signature Payload (署名ペイロード) フィールドに示します。

$$\text{HASH\_I} = \text{prf}(\text{SKEYID}, g^x, g^y, \text{CookieA}, \text{CookieB}, \text{SA}_p, \text{ID}_A)$$

認証にデジタル署名を使った場合、このハッシュは Host-A によっても署名されます。

ID_A は、Host-A の実体情報で、このメッセージの実体ペイロードで伝送されたものです。 SA_p は、SA ペイロード全体で、メッセージ 1 で Host-A によって送信されたものです。ここには、Host-A によって提案されたすべての提案およびすべての変形が含まれています。cookie、公開 Diffie-Hellman 値、および SKEYID は、メッセージ 1 ~ 4 で明示的に運ばれるか、それぞれの内容から派生されました。

IKE フェーズ 1、メッセージ 6

Host-A からメッセージ 5 を受信したら、Host-B は、そのハッシュを検査することによって Host-A の実体を検査します。

認証にデジタル署名を使っている場合、このハッシュの認証は Host-B によって検査されます。

これが成功したら、Host-B は、メッセージ 6 を Host-A に送信し、Host-A が Host-B の実体を検査できるようにします。

メッセージ 6 の構造は、メッセージ 5 の構造と同様ですが、はっきりとした変更点としては、実体ペイロードと Certificate Payload (証明書ペイロード) が Host-B に属するようになったことをあげられます。

$$HASH\_R = \text{prf}(SKEYID, g^y, g^x, \text{CookieB}, \text{CookieA}, SA_p, ID_B)$$

Diffie-Hellman 公開値と cookie が示される順序が変更されていて、最後の項目は、Host-B がメッセージ 6 で組み込んだ実体ペイロードであることに注意してください。

認証にデジタル署名を使った場合、このハッシュは Host-B によっても署名されます。この点は、Host-A によって署名された以前のハッシュと異なります。

Host-A がメッセージ 6 を受信して、ハッシュまたはデジタル署名を検査すると、フェーズ 1 交換が完了します。この時点で、それぞれの参加者は、自分のピアに対して自分自身を認証したことになります。どちらも、ISAKMP セキュリティー・アソシエーションの特性について同意し、同じ鍵のセット (または鍵マテリアル) を派生しました。

その他のフェーズ 1 要因

その他にも、注意すべき要因がいくつかあります。

- 使われる固有の認証メカニズムに関係なく、Oakley メイン・モードでは、交換されるメッセージは 6 つあります。しかし、個々のメッセージの内容は、認証方式に応じて異なります。
- Oakley 交換では、暗号化と認証の両方を利用しますが、IPsec の ESP または AH プロトコルのいずれも使いません。ISAKMP 交換は、ネットワーク・レイヤー・セキュリティ・メカニズムではなく、アプリケーション・レイヤー・セキュリティ・メカニズムで保護されています。
- ISAKMP メッセージは、UDP を使って送信されます。そのような送達には保証はありません。
- ISAKMP メッセージがフェーズ 2 フローではなくフェーズ 1 フローの一部であることを確認するための唯一の方法は、ISAKMP ヘッダーの Message ID (メッセージ ID) フィールドを調べることです。フェーズ 1 フローの場合、0 でなければならず、(ISAKMP 文書にはっきりと述べられていなくても) フェーズ 2 フローの場合、ゼロ以外でなければなりません。

21.5.5.4 IKE フェーズ 2 - プロトコルのセキュリティ・アソシエーションの設定

フェーズ 1 ネゴシエーション・プロセスを完了して ISAKMP セキュリティ・アソシエーションを設定したら、Host-A の次のステップは、Oakley フェーズ 2 メッセージ交換 (Oakley クイック・モードとも呼ばれる) を開始して、ユーザーのペアの間で交換される IP データグラムを保護するために使われるセキュリティ・アソシエーションと鍵を定義することです。(インターネットのドラフトでは、これらはいくらか控え目に "non-ISAKMP SAs (ISAKMP 以外の SA)" として言及されています。)

フェーズ 1 ネゴシエーションの目的は、ISAKMP メッセージを保護する方法についての同意を得ることであつたので、すべての ISAKMP フェーズ 2 ペイロード (ただし、ISAKMP ヘッダーそのものではない) は、フェーズ 1 ネゴシエーションで同意の得られたアルゴリズムを使って暗号化する必要があります。

フェーズ 2 で Oakley クイック・モードが使われる場合、認証は、いくつかの暗号ベースのハッシュ機能を使って行われます。ハッシュ機能への入力、一部はフェーズ 1 情報 (SKEYID) から、一部はフェーズ 2 で交換した情報から来ます。フェーズ 2 認証は証明書に基づくものですが、フェーズ 2 プロセスそのものでは証明書を直接には使いません。その代わり、フェーズ 1 からの SKEYID_a マテリアルを使います。このマテリアル自体は証明書で認証されたものです。

Oakley クイック・モードは、以下の 2 つの形式で実現します。

- 鍵交換属性を指定しなくても、クイック・モードを使って、暗号鍵をリフレッシュすることができます。ただし、Perfect Forward Secrecy (PFS) のプロパティは備えられていません。
- 鍵交換属性を指定すれば、クイック・モードを使って、PFS を提供する方法で暗号鍵をリフレッシュすることができます。このことは、公開 Diffie-Hellman 値の交換をメッセージ 1 と 2 に含めることで実現できます。

注：

PFS は、暗号化の専門家が待望していたプロパティのように思えますが、不思議なことに、仕様書は PFS をオプションとして扱っています。そこでは、システムは、Key Exchange (鍵交換) フィールドがクイック・モードのメッセージ内に存在する場合には、そのフィールドを処理できなければならないが、システムのメッセージ内にそのフィールドを含める必要はない、ということが示されています。

フェーズ 2 メッセージと交換された情報の詳細を続けます。

IKE フェーズ 2、メッセージ 1

クイック・モード交換のメッセージ 1 では、Host-A は、Host-A 自体を認証し、臨時を選択し、セキュリティ・アソシエーション (複数可) を Host-B に提案し、公開 Diffie-Hellman 値の交換を実行し、Host-A 自体のために機能しているのかそれとも別のエンティティのプロキシ・ネゴシエーション・プログラムとして機能しているのかを示すことができます。メッセージ 1 の形式の概要は、図 300 に示されています。

注：

鍵交換フィールドの組み込みはオプションです。しかし、Perfect Forward Secrecy が必要な場合には、このフィールドが存在していなければなりません。

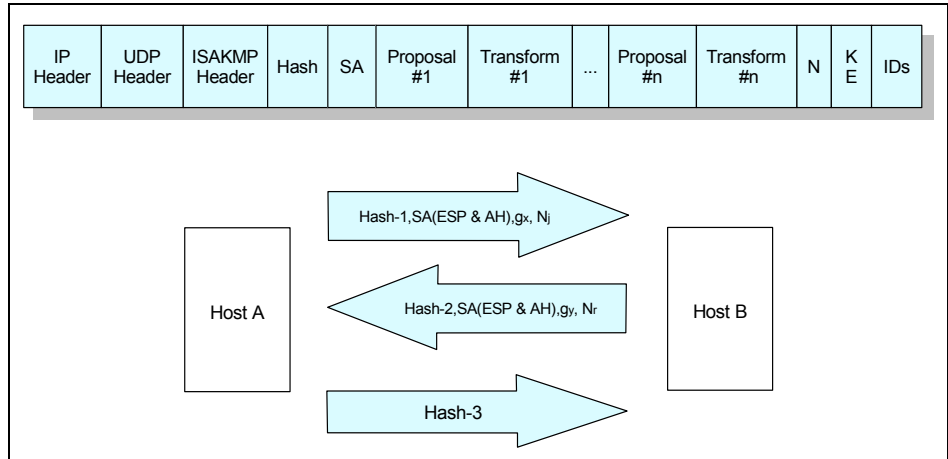


図 300 ISAKMP フェーズ 2 クイック・モード交換のメッセージ 1

Host-A と Host-B は、それぞれ自分のために機能することを前提としているため、図 300 に示されているユーザー識別フィールドは存在しません。メッセージは、以下のもので構成されます。

ISAKMP Header (ISAKMP ヘッダー)

ISAKMP Header (ISAKMP ヘッダー) は、クイック・モードの交換タイプを示し、Host-A によって選択されたゼロ以外の Message-ID (メッセージ ID) を含み、フェーズ 1 で選択した起動側および応答側 cookie の値 (つまり、Cookie-A および Cookie-B) を含み、暗号化フラグをオンにして、ISAKMP メッセージのペイロードがフェーズ 1 でネゴシエーションされたアルゴリズムと鍵に従って暗号化されることを示します。

Hash (ハッシュ)

Hash Payload (ハッシュ・ペイロード) は、ISAKMP ヘッダーのすぐ後に続く必要があります。HASH_1 は、フェーズ 1 交換でネゴシエーションされた鍵付き疑似乱数機能を使います。また、HASH_1 は、以下の情報から派生します。

- MSKEYID_a は、フェーズ 1 交換から派生したものです。
- M-ID は、このメッセージのメッセージ ID です。

- SA は、提案されたすべての提案を含め、このメッセージで運ばれる Security Association (セキュリティー・アソシエーション) ペイロードです。
- Nonce (臨時) は、フェーズ 1 で使われていた値とは異なる新しい値です。
- KE は、このメッセージで運ばれる公開 Diffie-Hellman 値です。この数量は、Host-A によって選択され、 g_{qm}^x として示されます。これは、フェーズ 1 交換で使われた g^x と同じ数量ではありませんので注意してください。
- ID は、ネゴシエーションする必要があるのはフェーズ 1 交換のエンドポイントなのか、プロトコル SA の代理をするエンドポイントなのかを識別できます (IKE がクライアント・モードで使われるときにはプロキシ ID)。これらの ID は、やはり、フェーズ 1 で使われた ID とは別である可能性があります。

注：

KE および ID を使うかどうかは、PFS が必要かどうかで応じて任意に決められます。

$$\text{HASH\_1} = \text{prf}(\text{SKEYID\_a}, \text{M-ID}, \text{SA}, \text{N}_{qmi}, \text{KE}, \text{ID}_{qmi}, \text{ID}_{qmr})$$

Security Association (セキュリティー・アソシエーション)

IP を Domain of Interpretation として示します。

Proposal, Transform (提案、変形) のペア

このメッセージには、このようなペアが 1 つ以上存在する可能性があります。最初の提案ペイロードは、1 の番号が付けられ、使う予定の IPsec プロトコルを識別し、そのプロトコルで使うために Host-A によってランダムに選択された SPI を含みます。提案ペイロードの後には、そのプロトコルで使う予定の暗号アルゴリズムを示す 1 つの変形ペイロードが続きます。2 番目の提案ペイロードに

は、2 の番号が付けられます。以後同様に番号が付けられていきます。

Nonce (臨時) ペイロード

ここには、Host-A によってランダムに選択された臨時 N_{qmi} が含まれます。

KE

これは、Host-A によって選択された公開 Diffie-Hellman 値 (g_{qm}^x) を運ぶ鍵交換ペイロードです。さらに、Group (グループ) というフィールドもありますが、これは Diffie-Hellman 交換で使われる素数と生成プログラムを示します。

ID ペイロード

この SA のエンドポイントを指定します。

IKE フェーズ 2、メッセージ 2

Host-B が Host-A からメッセージ 1 を受け取り、HASH_1 を使って正常に認証したら、応答としてメッセージ 2 を構成し、Host-A へ戻します。応答の Message ID (メッセージ ID) は、Host-A がメッセージ 1 で使ったメッセージ ID と同じです。

Host-B は、以下のものについて、新しい値を選択します。

Hash (ハッシュ)

ハッシュ・ペイロードは、値 HASH_2 を運ぶようになりました。これは、次のように定義されます。

$$\text{HASH\_2} = \text{prf}(\text{SKEYID\_a}, N_{qmi}, \text{M-ID}, \text{SA}, N_{qmr}, \text{KE}, \text{ID}_{qmi}, \text{ID}_{qmr})$$

Security Association (セキュリティー・アソシエーション)

Security Association (セキュリティー・アソシエーション) ペイロードは、選択した 1 つの提案と、それに関連した変形だけを記述します。Host-A によって提案される保護一式すべてが記述されるわけではありません。Host-B も、選択したプロトコルについて、1 つの SPI 値を選択します。Host-B の SPI は、どのような形であれ、Host-A が提案を提出したときにそのプロトコルに割り当てた SPI に依存することはありません。つまり、SPI_A を SPI_B と同じにする必要はないということです。必要なのは、それぞれをゼロ以外にすることと、それぞれをランダムに選択することだけです。

Nonce (臨時)

Nonce (臨時) ペイロードは、Host-B によって選択された乱数値である N_r を運ぶようになりました。

KE

鍵交換ペイロードは、Host-B の公開 Diffie-Hellman 値である g_{qm}^y を運ぶようになりました。

この時点で、Host-A と Host-B は、臨時と公開 Diffie-Hellman 値を交換しています。それぞれが、他の情報と共にこの情報を使い、伝送の方向ごとに、1 組の鍵を派生させることができます。

鍵の生成 (フェーズ 2)

臨時、公開 Diffie-Hellman 値、SPI、フェーズ 2 のメッセージ 1 と 2 で交換されたプロトコル・コード・ポイント、およびフェーズ 1 からの SKEYID 値を使うことにより、各ホストは、2 組の鍵マテリアルを派生するのに十分な情報を得ることができています。

• PFS を使う場合 :

- Host-A によって生成され、Host-B に受け取られるデータの場合、鍵マテリアルは次のようになります。

$$\text{KEYMAT}_{AB} = \text{prf}(\text{SKEYID\_d}, g_{qm}^{xy}, \text{protocol}, \text{SPI}_B, N_{qmi}, N_{qmr})$$

- Host-B によって生成され、Host-A に受け取られるデータの場合、鍵マテリアルは次のようになります。

$$\text{KEYMAT}_{BA} = \text{prf}(\text{SKEYID\_d}, g_{qm}^{xy}, \text{protocol}, \text{SPI}_A, N_{qmi}, N_{qmr})$$

• PFS を使わない場合 :

- Host-A によって生成され、Host-B に受け取られるデータの場合、鍵マテリアルは次のようになります。

$$\text{KEYMAT}_{AB} = \text{prf}(\text{SKEYID\_d}, \text{protocol}, \text{SPI}_B, N_{qmi}, N_{qmr})$$

- Host-B によって生成され、Host-A に受け取られるデータの場合、鍵マテリアルは次のようになります。

$$\text{KEYMAT}_{BA} = \text{prf}(\text{SKEYID\_d}, \text{protocol}, \text{SPI}_A, N_{qmi}, N_{qmr})$$

注：

特別なケースの場合、Host-A は、以下の目的で複数の鍵を派生しなければならない場合があります。

- 伝送したデータグラムの Integrity Check Value (ICV) を生成する。
- 受け取ったデータグラムの Integrity Check Value (ICV) を検査する。
- 伝送したデータグラムを暗号化する。
- 受け取ったデータグラムを復号する。

同様に、Host-B は、同じ鍵のミラー・イメージを派生する必要があります。たとえば、アウトバウンド・メッセージを暗号化するために Host-B が使う鍵は、そのインバウンド・メッセージを復号するために Host-A が使う鍵と同じにする場合があります。

IKE フェーズ 2、メッセージ 3

この時点で、Host-A と Host-B は、必要な鍵マテリアルを派生するのに必要なすべての情報を交換しています。クイック・モード交換での 3 番目のメッセージは、稼動状態であることを示すために Host-A によって使われます。このときには、メッセージ ID と、メッセージ 1 および 2 で交換された両方の臨時を網羅するハッシュ機能を生成します。メッセージ 3 は、ISAKMP ヘッダーと、以下を運ぶハッシュ・ペイロードだけで構成されています。

$$\text{HASH\_3} = \text{prf}(\text{SKEYID\_a}, 0, \text{M-ID}, N_{\text{qmi}}, N_{\text{qmr}})$$

Host-B がこのメッセージを受け取り、ハッシュを検査したら、それぞれのユーザー・データ・ストリームを保護するために、両方のシステムで、ネゴシエーションされたセキュリティ・プロトコルを使い始めることができます。

21.5.5.5 複数のセキュリティ・アソシエーションのネゴシエーション

1 つの 3 メッセージ・クイック・モード交換の中で、それぞれが独自の鍵マテリアルのセットを持つ複数のセキュリティ・アソシエーションをネゴシエーションすることも可能です。

メッセージ形式は、以前に示した形式と非常に似ているので、ここでは相違点だけに焦点を当てます。

- メッセージ 1 は、複数のセキュリティ・アソシエーション・ペイロードを運びますが、それぞれのペイロードは特定範囲の保護一式を提供しています。

- HASH_1 は、メッセージ 1 で運ばれる、提供された全セキュリティ・アソシエーションのセットの全体を網羅します。つまり、それぞれのセキュリティ・アソシエーションと、それによって提供されたすべての提案が含まれるということです。
- メッセージ 2 では、提供された SA ごとに、Host-B は 1 つの保護一式を選択します。つまり、n 個の SA がネゴシエーションのためにオープンしている場合、Host-B は、それぞれの提案ごとに 1 つの割合で、n 個の保護一式を選択します。
- HASH_1 のケースでそうであったように、HASH_2 は、メッセージ 1 で運ばれる、提供された全セキュリティ・アソシエーションのセットの全体を網羅するようになりました。つまり、それぞれのセキュリティ・アソシエーションと、それによって提供されたすべての提案が含まれるということです。
- メッセージ 1 と 2 を交換したら、Host-A と Host-B は、766 ページの の公式と同じ公式を使い、受け入れられる保護一式ごとに鍵マテリアルを生成できるようになります。このときの公式は、受け入れられる SA ごとに個別に適用されます。選択したすべての一式で臨時と公開 Diffie-Hellman 値が同じであるとしても、選択したそれぞれの保護一式ごとに派生される鍵マテリアルは異なっています。これは、それぞれの提案が異なる SPI を持つためです。
- 複数のセキュリティ・アソシエーションをネゴシエーションしたため、指定したデータグラムを保護するためにどれを使うかについては、それぞれが選択する問題になります。受信側のシステムは、ネゴシエーションされたすべての SA によって保護されるデータグラムを処理できなければなりません。つまり、指定したソース・ホストは、2 つの連続したデータグラムを宛先システムに送信することが正しいやり方です。その場合、それぞれのデータグラムは異なる SA によって保護されています。

21.5.5.6 リモート・アクセスでの IKE の使用

リモート・アクセスのシナリオで重要な要素は、Oakley を使い、動的に割り当てた IP アドレスではなく、名前によってリモート・ホストを識別することです。いったんリモート・ホストの識別を認証し、その動的に割り当てた IP アドレスへの割り当てを確認したら、残りの処理は他のシナリオで説明した処理と同じです。たとえば、企業のイントラネットが信頼できると見なされる場合、リモート・ホストは、それ自体とファイアウォールとの間に 1 つの SA を確立する必要があります。しかし、企業のイントラネットが信頼できないと見なされる場合には、リモート・ホストは、2 つの SA を設定する必要があります。1 つはそれ自体とファイアウォールとの間に、2 番目はそれ自体と宛先ホストとの間に設定します。

1 つの ISAKMP フェーズ 1 ネゴシエーションで、それに続く複数のフェーズ 2 ネゴシエーションを保護できることを思い出してください。フェーズ 1 ISAKMP ネゴシエーションでは、計算が集中する公開鍵暗号演算が使われるのに対して、フェーズ 2 ネゴシエーションでは、それほど計算が集中しない対称鍵暗号演算が使われます。そのため、計算が重くなるロードは、フェーズ 1 だけに生じ、ダイヤルアップ接続を最初に開始するときだけに実行されます。

リモート・アクセスのケースに関係する基本的な点は、以下のとおりです。

- ・ リモート・ホストの動的に割り当てられたアドレスは、すべての ISAKMP メッセージの IP ヘッダーに置かれたアドレスです。
- ・ リモート・ホストの永続 ID (E メール・アドレスなど) は、ISAKMP フェーズ 1 メッセージの ID フィールドに指定された数値です。
- ・ ISAKMP 交換で使われるリモート・ホストの証明書は、リモート・ホストの永続 ID に関連付ける必要があります。
- ・ トラフィックを生み出すデータグラムでは、リモート・ホストの動的に割り当てられた IP アドレスが使われます。このことが必要なのは、データグラムの IP ヘッダーに示される宛先 IP アドレスは、インバウンド・データグラムを処理するために、関係のある IPsec セキュリティー・アソシエーションを識別する SPI およびプロトコル・タイプと共に使われるためです。

21.6 SOCKS

SOCKS は、回線レベル・ゲートウェイの規格の 1 つです。この規格では、より標準的なプロキシ・サーバーのようなオーバーヘッドは必要ありません。標準的なプロキシ・サーバーでは、ユーザーは、宛先への 2 番目の接続を要求する前に、意識的にまずファイアウォールへ接続する必要があります (図 301 を参照してください)。

ユーザーは、宛先サーバーの IP アドレスでクライアント・アプリケーションを開始します。宛先サーバーとのセッションを直接に開始するのではなく、クライアントが、ファイアウォールの SOCKS サーバーへのセッションを開始します。SOCKS サーバーは、ソース・アドレスとユーザー ID が無保護ネットワークに対するそれ以上の接続を確立することを許可されているかどうか検査し、その後で、2 番目のセッションを作成します。

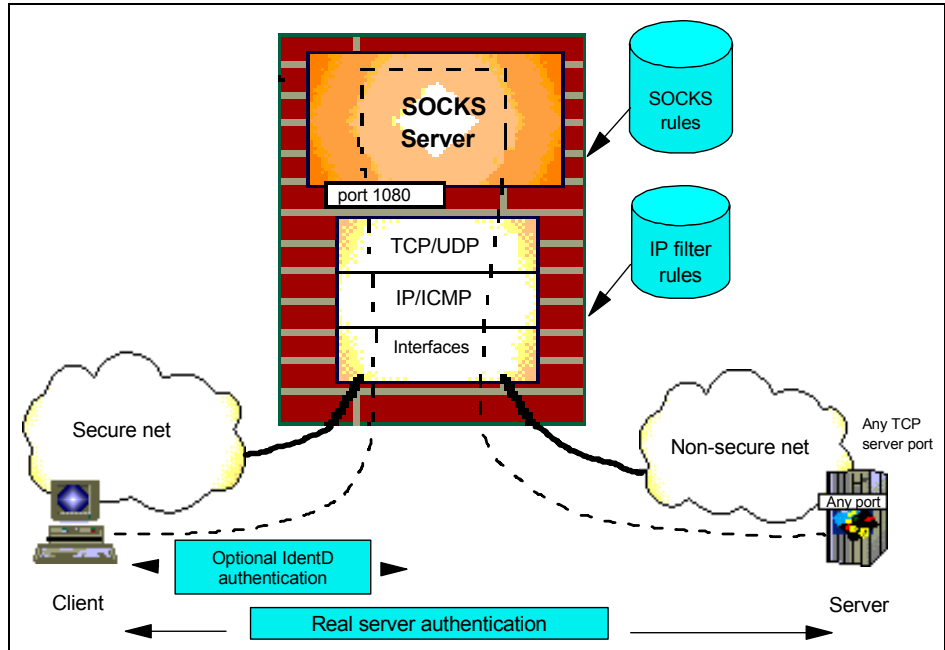


図 301 SOCKS サーバー

SOCKS には、新しいバージョンのクライアント・コード (SOCKS 化されたクライアントという) と、ファイアウォール上の別個の構成プロファイルのセットが必要です。ただし、サーバー・マシンでは変更は必要ありません。実際は、セッションが SOCKS サーバーによって中継されていることは分かりません。クライアントと SOCKS サーバーには、どちらにも SOCKS コードが必要です。SOCKS サーバーは、クライアントと実際のアプリケーション・サーバーとの間で、アプリケーション・レベルのルーターとして機能します。SOCKSv4 は、アウトバウンド TCP セッション専用です。プライベート・ネットワークのユーザーにとっては非常に簡単なのですが、保護パスワードの送達がないため、公衆ネットワーク・ユーザーとプライベート・ネットワーク・アプリケーションとの間のセッション向けではありません。SOCKSv5 には複数の認証方法が備えられているため、注意深さが求められるとはいえ、インバウンド接続にも使えます。SOCKSv5 は、UDP ベースのアプリケーションおよびプロトコルもサポートします。

Web ブラウザーの大多数は SOCKS 化されているので、ほとんどのプラットフォームに適した SOCKS 化された TCP/IP スタックを入手できます。その他の情報については、RFC 1928、RFC 1929、RFC 1961 および以下の URL を参照してください。

<http://www.socks.nec.com>

21.6.1 SOCKS バージョン 5 (SOCKSv5)

SOCKS バージョン 5 は、選択的な状況を持つ提案された規格プロトコルです。これについては、RFC 1928 で説明されています。

アプリケーション・レベル・ゲートウェイには、TELNET、FTP および SMTP といったいくつかのアプリケーション用の保護接続が備えられています。しかし、新しいアプリケーションごとに、プロキシー・コードを作成することは簡単ではありません。一般に、プロキシー・サービスを直接使うことができ、アプリケーション・レベル・ゲートウェイで UDP 接続が許可されていない場合でも、このプロキシー・サービスはしばらくしてから利用できるようになります。SOCKSv5 は、強力な認証メカニズムと無保護ネットワークからアドレス隠ぺいにおける短所と要件すべてを満たすものです。UDP のサポートは無防備に見えるかもしれませんが、特定のユーザーおよび特定のアプリケーションの UDP だけを通すように構成できます。

SOCKSv5 の概念は、SOCKSv4 に基づいていて、それに UDP サポート、新しい洗練されたさまざまな認証方式、そしてドメイン・ネームと IPv6 に対応する拡張アドレス方式など、いくつかの拡張機能が付加されたものです。SOCKSv5 は、以下を含む認証方式の範囲をサポートします。

1. ユーザー名 / パスワード認証
2. 一回限りパスワード・ジェネレーター
3. Kerberos
4. Remote Authentication Dial-In User Services (RADIUS)
5. Password Authentication Protocol (PAP)
6. IPsec 認証方式

SOCKSv5 は、以下の暗号化規格もサポートします。

1. DES
2. Triple DES
3. IPsec

以下のトンネリング・プロトコルがサポートされています。

1. PPTP
2. L2F
3. L2TP

以下の鍵管理システムがサポートされています。

1. SKIP

2. ISAKMP/Oakley

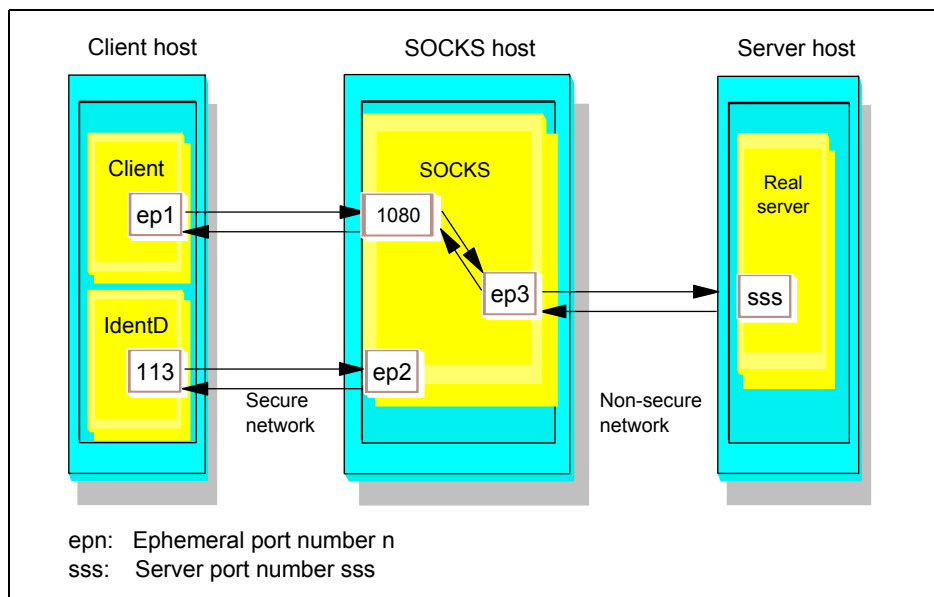


図 302 Socks TCP セグメント・フロー

SOCKSv5 サーバーは、指定したポート（通常は 1080）で接続を待機します。接続タイプ（TCP または UDP）に応じて、続くいくつかのセクションで説明されているステップを使い、接続を確立します。

21.6.1.1 SOCKSv5 TCP 接続

TCP を使って接続を確立するには、クライアントは、まずセッション要求情報を含む TCP パケットをポート 1080 経由でサーバーへ送信します（図 302 を参照してください）。アクセス権限によりこの操作が許可され、接続要求が成功すると、クライアントは認証のネゴシエーションに入ります。この状態では、認証タイプが決定され、その後でクライアントは中継要求を送信します。SOCKSv5 サーバーは要求を評価し、接続を確立するか拒否します。クライアントは、バージョン ID と方式オプションを含む次のメッセージを送信します。

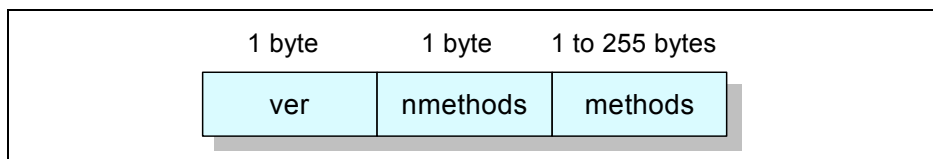


図 303 SOCKSv5 - バージョン ID および方式選択メッセージ形式

以下のような意味があります。

VER	SOCKS のバージョンを示します。SOCKSv5 の場合、値は 16 進数の X'05' です。
NMETHODS	methods (方式) フィールドに示される方式の数を示します。
METHODS	サポートされている認証およびカプセル化方式を示します。

サーバーは、以下のメッセージで応答します。

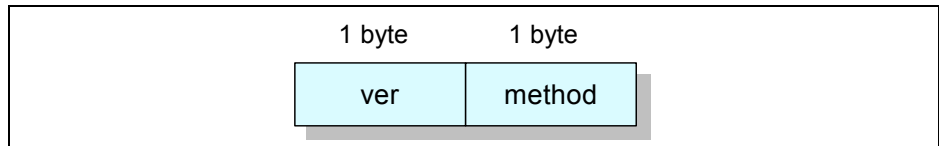


図 304 SOCKSv5 - 選択した方式メッセージ形式

現在の標準的な METHOD の 16 進値は、以下のとおりです。

X'00'	認証不要
X'01'	GSSAPI
X'02'	ユーザー名 / パスワード
X'03' ~ X'7F'	IANA でアサイン
X'80' ~ X'FE'	プライベート・メソッドのために予約済み
X'FF'	受け入れられない METHOD

すべての実装で、ユーザー名 / パスワード および GSSAPI 認証方式をサポートする必要があります。

SOCKSv5 接続

認証を正常に完了したら、クライアントは要求の詳細を送信します。方式のネゴシエーション時にカプセル化方式をネゴシエーションする場合、選択したカプセル化方式を以下のメッセージに適用しなければなりません。クライアントによって出される詳細な要求メッセージ形式は、以下のとおりです。

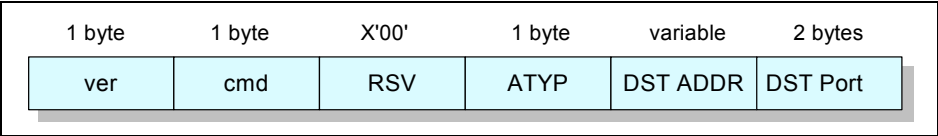


図 305 SOCKSv5 - 詳細な要求メッセージ形式

以下のような意味があります。

- VER** Socks プロトコル版番号。SOCKSv5 の場合、値は 16 進数の X'05' です。
- CMD** オクテットの SOCKS コマンド。
- X'01'** 持続
- X'02'** バインド
- X'03'** UDP ASSOCIATE
- RSV** 将来の利用のために予約済み。
- ATYP** オクテットのアドレス・タイプ。
- X'01'** IPv4 アドレス
- X'03'** ドメイン・ネーム
- X'04'** IPv6 アドレス
- DST.ADDR** 希望する宛先アドレス。
- DST.PORT** ネットワーク・オクテット順での希望する宛先ポート。

IPv4 アドレスは、4 バイトで保管されます。IPv6 アドレスは、16 バイトで保管されます。

ドメイン・ネームは、特定の長さバイトで保管されてから、完全修飾ドメイン・ネームとして保管されます。ドメイン・ネームの最後には末尾ヌルは付きません。

サーバーは、要求詳細メッセージを評価し、1 つ以上のメッセージで応答します。サーバーによって出される応答メッセージ形式は、以下のとおりです。

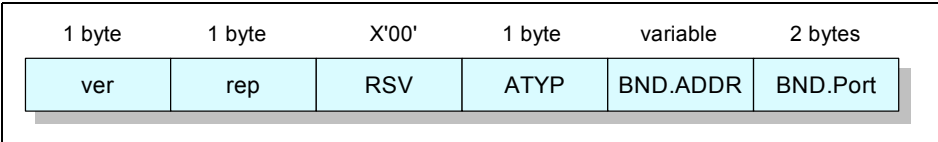


図 306 SOCKSv5 - サーバー応答メッセージ形式

以下のような意味があります。

VER Socks プロトコル版番号。SOCKSv5 の場合、値は 16 進数の X'05' です。

REP 応答フィールド：

X'00'	成功
X'01'	一般の SOCKS サーバー障害
X'02'	接続がルール・セットによって許可されていない
X'03'	ネットワークに到達不能
X'04'	ホストに到達不能
X'05'	接続が拒否された
X'06'	TTL 有効期限切れ
X'07'	コマンドがサポートされていない
X'08'	アドレス・タイプがサポートされていない
X'09' ~ X'FF'	割り当てられていない

RSV 将来の利用のために予約済み。

ATYPE オクテットのアドレス・タイプ。

X'01'	IPv4 アドレス
X'03'	ドメイン・ネーム
X'04'	IPv6 アドレス

BND.ADDR サーバーにバインドされたアドレス。

BND.PORT ネットワーク・オクテット順でのサーバーにバインドされたポート。

SOCKSv5 BIND

インターネットからの着信接続を受け入れるのであれば、SOCKSv5 接続に関して上記で説明されている形式と同じ要求および応答形式を使い、CMD フィールドを BIND に設定します。ただし、2 つの応答パケットを受け取ることになります。

最初の応答には、SOCKS サーバーが listener を関連付ける IP アドレスとポート番号が含まれています。

リモート・システムが SOCKS サーバーを呼び出すと、リモート・サーバーの詳細を含む BND.ADDR および BIND.Port フィールドを指定した 2 番目の応答を入手します。

21.6.1.2 SOCKSv5 UDP 接続

SOCKS サーバーを介して UDP 接続を使えるようにするには、クライアントは、まず UDP ASSOCIATE コマンドを SOCKSv5 サーバーに発行します。そうすると、SOCKSv5 サーバーは、クライアントがすべての UDP データグラムを送信する先の UDP ポートを割り当てます。各 UDP データグラムには、1 つの UDP 要求ヘッダーがあります。UDP 要求ヘッダー形式は、以下のとおりです。

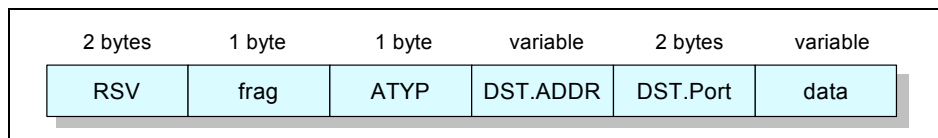


図 307 SOCKSv5 - UDP データグラムの要求ヘッダー形式

以下のような意味があります。

RSV 将来の利用のために予約済み。バイトはすべてゼロです。

FRAG 現在のフラグメント数。

ATYP オクテットのアドレス・タイプ。

X'01' IPv4 アドレス

X'03' ドメイン・ネーム

X'04' IPv6 アドレス

DST.ADDR 希望する宛先アドレス。

DST.PORT ネットワーク・オクテット順での希望する宛先ポート。

DATA ユーザー・データ。

UDP 中継サーバーは、UDP データグラムを DST.PORT で指定したポートへ送信するクライアントの IP アドレスを入手します。その後、別のソースからのすべてのデータグラムを廃棄します。

21.7 Secure Shell (I)

SSH は、システム同士の接続を保護するために使えます。これにより、TELNET、FTP POP3、または X Windows によって生成されたトラフィックのようなアプリケーション・トラフィックに対し、暗号化と圧縮の両方を実行できるようになります。圧縮は、低速モデム・リンクを経由しているときに便利です。実装では、ユーザーは暗号化方式を選択できます。

クライアント・ソフトウェアには、しばしば SSH1 サポートと SSH2 サポートの両方が装備されていることがあります。ユーザーは、パスワードか公開鍵 / 秘密鍵で認証されます。

SSH1 には、Blowfish、DES、3DES および RC4 といった暗号を解く鍵が備えられています。

SSH2 には、3DES、RC4、および Twofish といった暗号を解く鍵が備えられています。

21.7.1 SSH の概要

SSH は、クライアントからサーバーへの 1 つの TCP/IP 接続を確立します。この接続を介して送信されるトラフィックは暗号化され、オプションで LempelZiv 圧縮を使って圧縮されます。公開鍵 / 秘密鍵は、リモート・システムのユーザーと実体の両方を検査するときに使えます。

21.7.1.1 SSH および X Windows

X Window セッションは、SSH 接続を通過できます。SSH サーバーは、リモート X Window のクライアント向けに、新しい DISPLAY 変数 (および xauth 鍵) を生成します。SSH は、この X Windows トラフィックをユーザーのローカル X Server に転送します。ユーザーは、自分の X Server アプリケーションを所有していなければなりません。このアプリケーションは、ローカル・ホスト上で listening 状態にしておきます。

21.7.1.2 SSH ポート転送

SSH には、システム同士で TCP/IP ポートを割り当てる機能があります。たとえば、クライアントのローカル・ホスト上のポートとサーバーの POP3 ポートとの間でデータをコピーするよう SSH を構成できます。POP3 クライアントを実行し、ローカル・ホストで指定することにより、E メールを読むことのできる保護され暗号化されたセッションを確立します。

21.8 Secure Sockets Layer (SSL)

SSL は、Netscape Communications Corporation が RSA Data Security, Inc. と共同で開発したセキュリティー・プロトコルです。SSL プロトコルの主な目的は、通信するアプリケーション同士の専用チャネルを実現し、データのプライバシー、パートナーの認証、および整合性を確実にすることです。

21.8.1 SSL の概要

SSL は、それ自体にセキュリティーが実装されている標準の TCP/IP ソケット API に対して、それ以外の方法を提供します。したがって、理論的には、どのような TCP/IP アプリケーションであっても、アプリケーションを変更す

ることなく、保護された方法で実行することが可能です。実際には、SSL は HTTP 接続向けにのみ広く実装されている方法ですが、Netscape Communications Corp. では、NNTP や Telnet といった他のアプリケーション・タイプにも SSL の採用を呼びかけており、そのような実装のいくつかをインターネット上で自由に利用できるようになっています。たとえば、IBM では SSL を使い、Host On-Demand および eNetwork Communications Server 製品における TN3270 セッションのセキュリティーを強化しています。

SSL は、以下の 2 つのレイヤーで構成されます。

- 下のレイヤーで、さまざまな事前定義された暗号と認証の組み合わせを使ってデータを転送するためのプロトコルは、*SSL レコード・プロトコル* と呼ばれます。図 308 には、このプロトコルが示されていて、標準の HTTP ソケット接続と対比されています。この図では、SSL が簡単なソケット・インターフェースを提供するものとして示されていて、そこに他のアプリケーションを階層化することができる点に注意してください。実際には、現在の実装では、ソケット・インターフェースはアプリケーション内に組み込まれていて、他のアプリケーションで使える API が見えるわけではありません。
- 上のレイヤーで、暗号鍵を初期認証して転送するためのプロトコルは、*SSL ハンドシェーク・プロトコル* と呼ばれます。

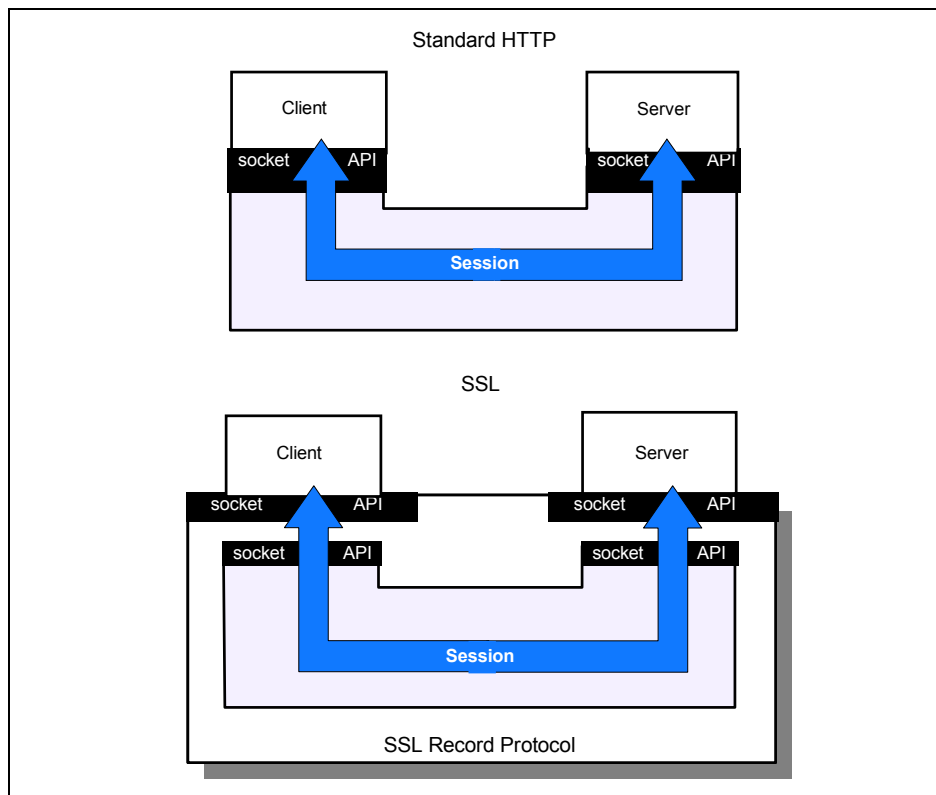


図 308 SSL - 標準セッションと SSL セッションの比較

SSL セッションは、以下のようにして開始します。

- クライアント (ブラウザ) で、ユーザーが `http:` ではなく `https:` で始まる特別な URL を持つ文書を要求します。このときには、その URL を URL 入力フィールドに入力するか、リンクをクリックします。
- クライアント・コードが SSL 要求を認識し、サーバー上の SSL コードへの接続を TCP ポート 443 経由で確立します。
- それからクライアントは、SSL レコード・プロトコルをキャリアとして使い、SSL ハンドシェイク・フェーズを開始します。この時点では、接続に組み込まれた暗号化検査または整合性検査はありません。

SSL プロトコルは、以下のセキュリティー問題に取り組んでいます。

プライバシー 初期ハンドシェイクで対称鍵が確立されたら、メッセージはこの鍵を使って暗号化されます。

整合性	メッセージには、メッセージの整合性を確認するメッセージ確認コード (MAC) が含まれます。
認証	ハンドシェイク時に、クライアントは、非対称鍵または公開鍵を使ってサーバーを認証します。これは、証明書に基づいて行うことも可能です。

SSL では、それぞれのメッセージを暗号化および復号することが必要であるため、ハイパフォーマンスであり、リソースのオーバーヘッドを持つようになります。

21.8.1.1 SSL V2.0 と SSL V3.0 との間の相違点

SSL V2.0 と SSL V3.0 との間には、下位互換性があります。SSL V3.0 サーバー実装では、SSL V2.0 クライアントからの接続要求を受け入れられなければなりません。SSL V2.0 と SSL V3.0 の主な相違点は、以下のとおりです。

- SSL V2.0 はクライアント認証をサポートしません。
- SSL V3.0 は CipherSpec での別の暗号タイプをサポートします。

21.8.2 SSL プロトコル

SSL プロトコルは、トランスポート・レイヤーの最上位に位置します。SSL は、それ自体が階層化されたプロトコルでもあります。これは、アプリケーション・レイヤーからデータを取り出し、それを再フォーマットし、トランスポート・レイヤーへ伝送するだけです。SSL は、メッセージを以下のように扱います。

• 送信側

以下の作業を実行します。

- 上位レイヤーからメッセージを取り出す
- データを管理しやすいブロックにフラグメント化する
- オプションでデータを圧縮する
- メッセージ確認コード (MAC) を適用する
- データを暗号化する
- 結果を下位レイヤーに伝送する

• 受信側

以下の作業を実行します。

- 下位レイヤーからデータを取り出す
- 復号する
- ネゴシエーションされた MAC 鍵でデータを検査する
- 圧縮が使われている場合はデータを解凍する
- メッセージを再アセンブルする
- メッセージを上位レイヤーに伝送する

SSL セッションは、それぞれ異なる状態で機能します。これらの状態は、セッション状態と、接続状態です。SSL ハンドシェーク・プロトコル (782 ページの 21.8.2.2 『SSL ハンドシェーク・プロトコル』を参照) は、クライアントとサーバーの状態を調整します。さらに、暗号スペック変更のメッセージに従って暗号化を調整するために定義された読み取り書き込み状態が存在します。

いずれかのパーティーが、暗号スペック変更のメッセージを送信すると、書き込み保留状態が、現在の書き込み状態に変更されます。また、いずれかのパーティーが、暗号スペック変更のメッセージを受信すると、読み取り保留状態が、現在の読み取り状態に変更されます。

セッション状態には、以下のコンポーネントが含まれます。

セッション ID	アクティブまたは再開可能なセッション状態を示すために、サーバーに選択された任意のバイト・シーケンス。
ピア証明書	ピアの証明書。このフィールドはオプションです。空のままでも構いません。
圧縮方式	圧縮アルゴリズム。
暗号スペック	データ暗号化アルゴリズム (ヌルや DES など) および MAC アルゴリズムを指定します。
マスター秘密	クライアントとサーバーとの間の 48 バイトの共有秘密。
再開可能	新しい接続にセッションを使えるかどうかを示すフラグ。

接続状態には、以下のコンポーネントが含まれます。

サーバーおよびクライアントの乱数

接続ごとに、クライアントとサーバーによって選択された任意のバイト・シーケンス。

サーバー書き込み MAC 秘密

サーバーによって、MAC 演算のために使われる秘密。

クライアント書き込み MAC 秘密

クライアントによって、MAC 演算のために使われる秘密。

サーバー書き込み鍵

サーバーにとってはデータを暗号化し、クライアントにとってはデータを復号するための暗号鍵。

クライアント書き込み鍵

クライアントにとってはデータを暗号化し、サーバーにとってはデータを復号するための暗号鍵。

初期設定ベクトル

初期設定ベクトルは、暗号化情報を保管します。

シーケンス番号

シーケンス番号は、最後の暗号スペック変更のメッセージ以降に伝送されたメッセージの番号を示します。クライアントおよびサーバーの両方が、シーケンス番号を保守します。

21.8.2.1 Change Cipher Spec プロトコル

Change Cipher Spec プロトコルには、暗号スペック変更のメッセージを送信する役割があります。クライアントは、ハンドシェーク鍵交換のような現在の暗号パラメーターを変更するために、いつでも要求を出すことができます。暗号スペック変更の通知に従い、クライアントはハンドシェーク鍵交換を送信し、利用できるのであれば、証明書検査メッセージを送信します。また、サーバーは、鍵交換メッセージを処理した後に暗号スペック変更のメッセージを送信します。その後、次の暗号スペック変更の要求が出されるまで、新しく同意された鍵を使います。暗号スペック変更のメッセージは、ネゴシエーション時の hello メッセージの後に送信されます。

21.8.2.2 SSL ハンドシェーク・プロトコル

SSL ハンドシェーク・プロトコルでは、クライアントおよびサーバーは、SSL 接続に必要なパラメーター（たとえば、プロトコル・バージョン、暗号アルゴリズム、オプションのクライアントまたはサーバー認証、および公開鍵暗号化方式など）を決定し、共有秘密を生成できます。この処理時には、ハンドシェーク・メッセージはすべて SSL レコード・レイヤーに転送され、特

別な SSL メッセージにカプセル化されます。図 309 には、SSL ハンドシェークの処理の一例が示されています。

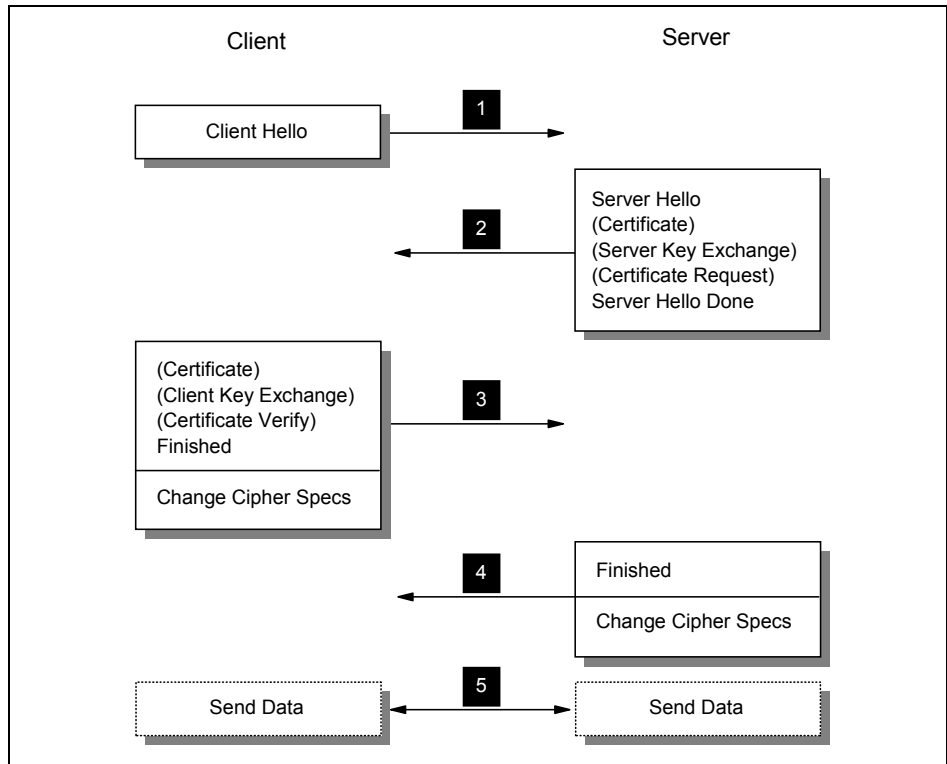


図 309 SSL - ハンドシェーク処理

図 309 に詳しく述べられている SSL ハンドシェーク処理は、以下の部分でさらに詳しく説明されています。

- ステップ 1: クライアントは、クライアント hello メッセージで接続要求を送信します。このメッセージには、以下のものが含まれます。
 - 必要なバージョン番号。
 - 時間情報 (標準の UNIX 32 ビット形式での現在時刻および日付)。
 - 任意でセッション ID。これを指定しない場合、サーバーは、以前のセッションを再開しようとするか、エラー・メッセージを戻します。
 - 暗号スイート。(クライアントによってサポートされている暗号オプションのリスト。これは、認証モード、鍵交換方式、暗号化および MAC アルゴリズムです。)
 - クライアントによってサポートされている圧縮方式。
 - 乱数値。

- ステップ 2: サーバーは、クライアント hello メッセージで送信されたパラメーターを評価し、サーバー hello メッセージを戻します。このメッセージには、SSL セッションで使うためにサーバーによって選択された以下のパラメーターが含まれます。

- バージョン番号
- 時間情報 (標準の UNIX 32 ビット形式での現在時刻および日付)
- セッション ID
- 暗号スイート
- 圧縮方式
- 乱数値

サーバー hello メッセージに従い、サーバーは以下のメッセージを送信します。

- サーバーを認証する必要がある場合のサーバー証明書
- 利用できる証明書が存在しないか証明書が署名専用である場合のサーバー鍵交換メッセージ
- クライアントを認証する必要がある場合の証明書要求

最後に、サーバーは、サーバー hello 完了メッセージを送信し、クライアントの応答を待機し始めます。

- ステップ 3: クライアントは、以下のメッセージを送信します。
 - サーバーが証明書要求を送信した場合、クライアントは、証明書を送信するか、証明書がない旨のメッセージを送信する必要があります。
 - サーバーがサーバー鍵交換メッセージを送信した場合、クライアントは、hello メッセージで判別した公開鍵アルゴリズムに基づいて、クライアント鍵交換メッセージを送信します。
 - クライアントが証明書を送信した場合、クライアントは、サーバー証明書を検査し、結果を示す証明書検査メッセージを送信します。

次に、クライアントは、ネゴシエーション部分が完了したことを示す完了メッセージを送信します。さらに、暗号スペック変更のメッセージも送信し、共有秘密を生成します。操作のこの部分は、ハンドシェーク・プロトコルによって制御されているものではなく、Change Cipher Spec プロトコルによって管理されていることを覚えておく必要があります。

- ステップ 4: サーバーは、ネゴシエーション部分が完了したことを示す完了メッセージを送信します。その後、サーバーは、暗号スペック変更のメッセージを送信します。
- ステップ 5: 最後に、セッションのパートナーは、個別に暗号鍵を生成します。このときに、これ以降の暗号化されたセッションで使う鍵をマスター鍵から派生します。ハンドシェーク・プロトコルは、状態を接続状態に変更します。アプリケーション・レイヤーから取り出されたデータはすべて、他のパーティーに対する特別メッセージとして伝送されます。

SSL セッションを開始するときには、通常の HTTP 接続を開始するときと比べて、余分のオーバーヘッドが多くなります。プロトコルでは、クライアントとサーバーがセッション鍵情報を保存し、2 回目にはネゴシエーションや認証せずにそのセッションを再開できるようにすることにより、このオーバーヘッドをいくらか回避します。

ハンドシェイクに従い、両方のセッション・パートナーが 1 つのマスター鍵を生成しました。その鍵から、他のセッション鍵を生成します。これらの鍵は、セッション・データの対称鍵暗号化とメッセージ・ダイジェストの作成に使われます。この方法で暗号化された最初のメッセージは、サーバーからの完了したメッセージです。クライアントが完了したメッセージを解釈できる場合、以下のような意味があります。

- メッセージが対称鍵バルク暗号 (DES や RC4 など) を使って暗号化されるため、プライバシーが実現しています。
- メッセージにはメッセージ確認コード (MAC) が含まれているため、メッセージ整合性が確実になっています。MAC は、メッセージそのものにマスター鍵から派生したマテリアルを加えたメッセージ・ダイジェストです。
- サーバーは、以前のマスター鍵からマスター鍵を派生できたので、認証されています。これは、サーバーの公開鍵を使って送信されたので、サーバーによってのみ (サーバーの秘密鍵を使って) 復号されることになっています。このことは、サーバーの公開鍵証明書の整合性にかかっていることに注意してください。

21.8.2.3 SSL レコード・プロトコル

マスター鍵を決定したら、クライアントおよびサーバーでは、アプリケーション・データを暗号化するためにその鍵を使えます。SSL レコード・プロトコルは、このようなメッセージの形式を指定します。一般に、このプロトコルには、変更されていないこととメッセージ全体が対称暗号を使って暗号化されたことを確認するための、メッセージ・ダイジェストが組み込まれています。通常ここでは、RC2 または RC4 アルゴリズムが使われています。ただし、仕様では、DES、Triple-DES、および IDEA もサポートされています。

米国連邦政府の一部門である米国国家安全保障局 (NSA) は、米国外に輸出されるソフトウェアで使える暗号鍵のサイズに制限を課していました。これらの規則は見直されましたが、当初は、鍵は、実サイズが 56 ビットに制限されていました。RC2 および RC4 アルゴリズムでは、56 ビット以外のすべてのビットが固定値に設定された鍵を使うことにより、このことを実現していました。国際 (輸出) 版のソフトウェア製品には、このような障害となるセキュリティが組み込まれていました。SSL は、ハンドシェイクのネゴシエーション・フェーズで、輸出版と非輸出版の間の不一致を調べます。たとえば、米国のブラウザーで SSL を使って輸出版のサーバーに接続しようとする場合、輸出強度の暗号化に同意することになります。暗号マテリアルの米国の

輸出規制を巡る最近の変更については、703 ページの 21.2.7 『暗号化の輸出 / 輸入制限』を参照してください。

21.9 Transport Layer Security (TLS)

Transport Layer Security 1.0 プロトコルは、SSL に基づいています。TLS 1.0 プロトコルについては、RFC 2246 に記述されています。2 つのアプリケーション (相互のコードについては知らない) で TLS を使い、保護された通信をすることができます。SSL 3.0 と TLS 1.0 の間には、目立った相違点はありません。それらは、メッセージ形式を少し変更するだけで、相互に運用できます。TLS 1.0 アプリケーションは、SSL 3.0 接続に降格することができます。

21.10 Secure Multipurpose Internet Mail Extension (S-MIME)

Secure Multipurpose Internet Mail Extension (S-MIME) は、非常に特異な SSL に似たプロトコルであると考えられます。S-MIME は、アプリケーション・レベルのセキュリティ構成ですが、その使用は、暗号化とデジタル署名で E メールを保護することに限定されます。これは、公開鍵テクノロジーに依存しているもので、X.509 証明書を使って通信パーティーの実体を確立します。S-MIME は、通信する終端システムに実装できます。中間ルーターやファイアウォールでは使われません。

21.11 仮想プライベート・ネットワーク (VPN) の概要

インターネットは、一般的で、低コストのバックボーン・インフラストラクチャーになっています。インターネットではどこにでも到達できるため、多くの企業は、公共のインターネットを経由して、保護された仮想プライベート・ネットワーク (VPN) の構築を考慮するようになっていきます。今日のグローバル・ビジネス環境に適した VPN の設計で挑戦となることは、企業内通信と企業間通信の両方のために公共のインターネット・バックボーンを活用すると同時に、従来のプライベートな自己管理企業ネットワークのセキュリティを提供することです。

この章では、仮想プライベート・ネットワーク (VPN) を定義し、その実装によって実現できる利点を説明することから始めます。セキュリティの考慮事項と計画の概要を説明したら、今日のマーケットで利用できる VPN ソリューションを説明します。

21.11.1 VPN の紹介および VPN の利点

インターネットの爆発的な成長に伴い、企業は「どのようにすれば我々のビジネスでインターネットを最もよく活用できるだろうか」と自問し始めています。当初は、企業はインターネットを使い、企業の Web サイトへ WWW アクセスできるようにすることにより、その企業のイメージ、製品、およびサービスを促進してきました。しかし、今日では、インターネットの可能性は無限であり、従来の IT システムに存在する主なビジネス・アプリケーションおよびデータに簡単にアクセスできる、インターネットのグローバルな到達性を使った e-business に焦点が移ってきています。企業は、保護された仮想プライベート・ネットワーク (VPN) ソリューションを実装することにより、アプリケーションおよびデータの到達範囲を安全にかつ低コストで世界に広げられるようになっていきます。

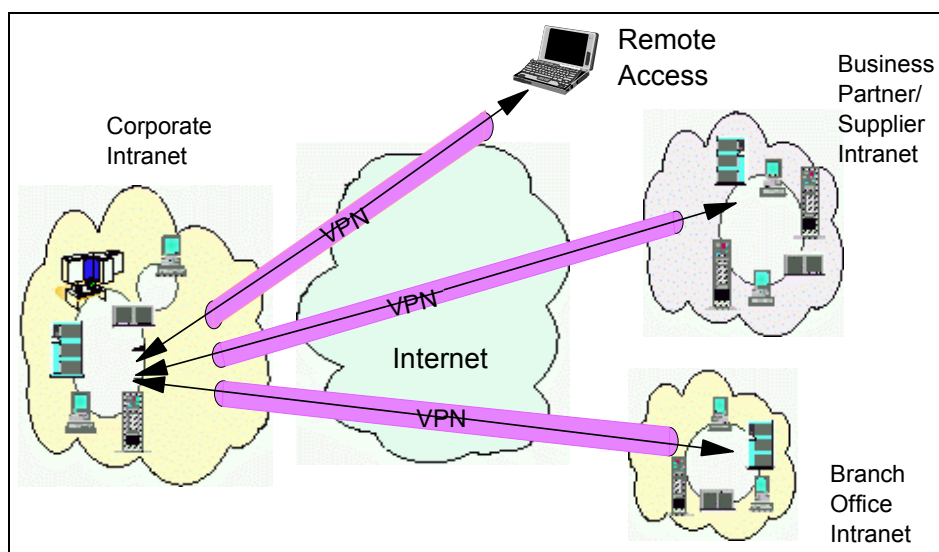


図 310 仮想プライベート・ネットワーク

仮想プライベート・ネットワーク (VPN) とは、インターネットなどの公開ネットワークを経由した企業のプライベート・イントラネットを強化したもので、基本的には、プライベートなトンネルを経由した、保護されたプライベート接続を作成します。VPN は、図 310 に示されているように、インターネットで接続しているリモート・ユーザー、支店、およびビジネス・パートナーを経由して、情報を強化された企業ネットワークへ安全に運びます。インターネット・サービス・プロバイダー (ISP) では、(直接回線またはローカルな電話番号経由で) インターネットへ低コストでアクセスできるので、企業は、現在の高価な専用回線、長距離電話、およびフリーダイヤル電話番号を廃止することができます。

Infonetics Research, Inc. による 1997 年度の VPN リサーチ・レポートでは、VPN を使ってリモート・サイトへの専用回線を置き換えることにより、広域ネットワーク (WAN) コストの 20% ~ 47% を抑えられるとの見積もりが出されています。さらに、リモート・アクセス VPN の場合、企業のリモート・アクセス・ダイヤルアップ・コストの 60% ~ 80% を節約できます。また、他の代替接続方法を利用できない場所でも、インターネット・アクセスは世界中で利用できます。

しかし、このような仮想プライベート・ネットワークを実装するためのテクノロジーは、標準化され始めたばかりです。今日のいくつかのネットワーキング・ベンダーは、標準化されていない VPN ソリューションを提供しているため、企業にとっては、従業員またはビジネス・パートナー / 仕入れ先 (またはその両方) すべてを、強化された企業ネットワークに統合することが難しくなっています。しかし、Internet Engineering Task Force (IETF) の標準に基づく VPN ソリューションは、相互運用がより容易で拡張機能がより高い全範囲の VPN シナリオをサポートします。

VPN の価値を最大にする鍵としては、企業が自分のビジネス・ニーズの変化に応じて自分の VPN を発展させたり、将来の TCP/IP テクノロジーへ容易にアップグレードできる機能をあげられます。広範囲のハードウェアおよびソフトウェア VPN 製品をサポートするベンダーには、このような要件を満たす柔軟性が備わっています。現在の VPN ソリューションは主に IPv4 環境で稼働するものですが、引き続きビジネス・パートナーまたは仕入れ先 (またはその両方) の VPN ソリューションと相互運用するため、IPv6 へアップグレードする機能を持っていることは重要です。また同じように重要なのは、VPN をデプロイする問題を理解しているベンダーと共同で作業する能力です。成功した VPN の実装には、テクノロジー以上のことが関係しています。この問題では、ベンダーのネットワーキングの経験が大きく物を言います。

21.12 Kerberos 認証および許可システム

Kerberos ネットワーク認証サービス バージョン 5 は、提案されている標準プロトコルです。この状況は、選択的で、RFC 1510 で説明されています。

Ambrose Bierce による *Enlarged Devil's Dictionary* によると、Kerberos は、「地獄の番人であり、はっきりと見えない人やものから門を守る務めがある。Kerberos は 3 つの頭を持つものとして知られている」ということです。

Kerberos サービスは、保護された領域で、通常は自分自身のシステムで稼働します。ユーザーは、ネットワークの他のサーバーへの接続を許可される前に、自分自身を Kerberos に検証してもらう必要があります。サーバーの実体も Kerberos に検査してもらうことが可能です。

Kerberos 認証および許可システムは、暗号化をベースにしたセキュリティ・システムで、ネットワーク環境内のユーザーとサーバーの間での相互認証を実現します。このシステムで想定する目的は、以下のとおりです。

- 機密でなければならないユーザーとサーバーとの間、および少なくとも 1 つのユーザーと 1 つのサービスのグループとの間で、不正な要求 / 応答を防ぐために認証を行うこと。
- 独自の許可システムを用意する予定のサービスごとに、許可を認証とは別に実装できるようにすること。許可システムでは、ユーザー / クライアントの認証を信頼できるものと見なせます。
- 「不渡り」または請求書作成発行の目的のためにモジュラー添付およびサポートに統合でき、保護されていて信頼できる会計システムの実装を許可すること。

Kerberos システムは、主に認証の目的で使われますが、許可情報を追加する柔軟性も持ち合わせています。

21.12.1 前提事項

Kerberos では、以下のことを前提としています。

- このセキュリティ・システムを使う環境には、最低限の物理的セキュリティを備えた区域に置くことのできる公開ワークステーションおよびプライベート・ワークステーション、リンク暗号化を使わずに、バックボーンまたはゲートウェイで接続された分散ローカル・ネットワークで構成できる大学ネットワーク、普通の物理セキュリティを備えた鍵の掛かった部屋にある集中運用サーバー、そして顕著な物理セキュリティを備えた集中運用サーバーが含まれます。
- ワークステーションを端末として利用すると、特定の条件をエミュレートすることができ、通常のデータが、暗号化保護なしで流れることになるため、銀行業務のような機密データまたは高リスク運用は、特別なセキュリティのないこの環境では扱えません。
- 使われている暗号システムの 1 つに、Data Encryption Standard (DES) があります。これは、Kerberos 設計担当者によってモジュラー化され置き換えられることを目的に開発されました。
- Kerberos では、システム全体のクロックの同期が不正確であると見なされるため、ワークステーションには、タイム・サーバーのような同期ツールが必要です。

21.12.2 命名

プリンシパル ID とは、Kerberos システムのクライアントまたはサービスを識別する名前です。

バージョン 4 では、ID は以下の 3 つのコンポーネントで構成されます。

- プリンシパル名は、Kerberos の管理機能によって割り当てられたクライアントおよびサービスごとに固有です。
- 明確な認証に使われるインスタンス名は、複数の形式で存在するクライアントおよびサービスの追加ラベルです。ユーザーの場合は、1 つのインスタンスで、個別の特権ごとに異なる ID を提供できます。サービスの場合は、通常 1 つのインスタンスで、このサービスを提供するマシンのホスト名を指定します。
- レルム名は、Kerberos サイトを個別に管理できるようにするために使われます。プリンシパル名とインスタンスは、属するレルムによって修飾され、そのレルム内でのみ固有になります。レルムは、一般的にはドメイン・ネームです。

バージョン 4 では、3 つのコンポーネントのそれぞれに 39 文字の長さ制限があります。規則によると、ピリオド (.) は受け入れられない文字です。

バージョン 5 では、ID は 2 つの部分 (レルム および 剰余) だけで構成されます。プリンシパルに名前を付けるときに多くのコンポーネントが必要になっても、このような順序になります。レルムと、剰余の各コンポーネントは、どちらも ASN.1 (Abstract Syntax Notation One、ISO 規格 8824) GeneralStrings として定義されます。ここでは、プリンシパル ID で利用できる文字についての制限はほとんどありません。

21.12.3 Kerberos 認証プロセス

Kerberos システムでは、サービスのためにサーバーへ接続するクライアントは、まず、相互に信頼のおける第三者機関 Kerberos Authentication Server (KAS) に、チケットを要請しなければなりません。このチケットは、機能として入手されます。この機能では、いずれかのコンポーネントが、サービスおよび Kerberos Authentication Server によってのみ知られている秘密鍵であるため、サービス提供側は、チケットの情報が Kerberos からのものであることを信じることができます。クライアントは、KAS にプリンシパル名 (c) として知らされます。秘密鍵 (K_c) は、ユーザーと Kerberos Authentication Server (KAS) にのみ知らされている認証鍵です。

この章では、 $\{X,Y\}$ の記号は、情報 (またはデータ) X および Y を含むメッセージを示しています。 $\{X,Y\}K_z$ は、データ X および Y を含むメッセージが、鍵 K_z を使って暗号化されたことを表します。

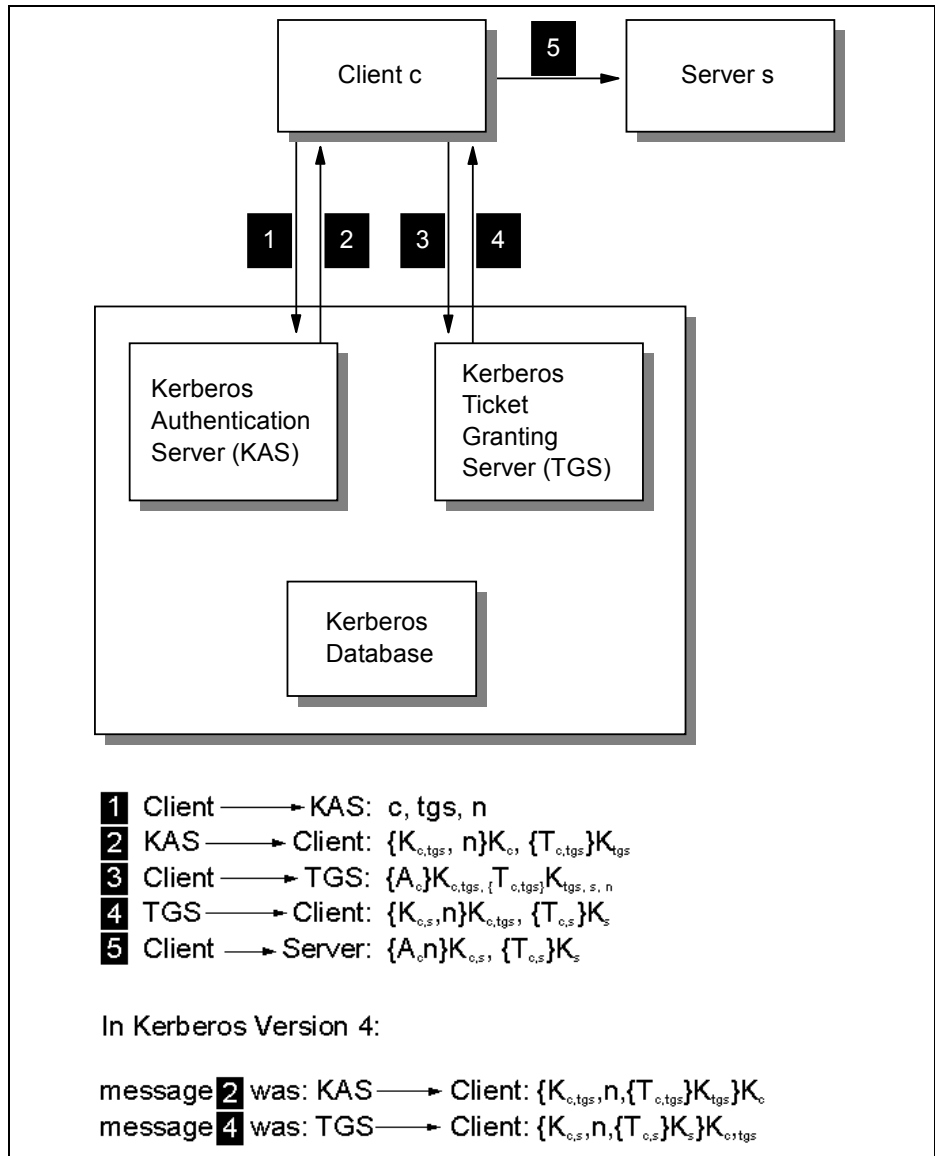


図 311 Kerberos 認証の構造

認証プロセスは、5つのメッセージを交換することで構成されています (図 311 を参照)。

1 クライアント $\rightarrow$ KAS

クライアントは、メッセージ $\{c, tgs, n\}$ を KAS に送信します。ここには、実体 (c)、臨時 (タイム・スタンプまたはこの要求を識別するための別の方法)、そしてチケット許可サーバー (TGS) で使うチケットの要求が含まれます。

2 KAS -> クライアント

認証サーバーは、Kerberos データベースからクライアント名 (c) とサービス名 (チケット許可サーバー、tgs) を探し、(K_c および K_{tgs}) ごとに、暗号鍵を入手します。

その後、KAS はクライアントへ送り返す応答を作成します。この応答には、初期チケット $T_{c,tgs}$ が含まれています。これは、要求されたサーバー (チケット許可サーバー) へのクライアント・アクセスを認可します。 $T_{c,tgs}$ には、 $K_{c,tgs}$ 、c、tgs、臨時、存続期間、および他の情報が含まれています。さらに KAS は、セッション鍵と呼ばれる、乱数暗号鍵 $K_{c,tgs}$ も生成します。その後、このチケットは、チケット許可サーバー (K_{tgs}) の暗号鍵を使って暗号化されます。このことにより、いわゆる *封印* チケット $\{T_{c,tgs}\}_{K_{tgs}}$ が生成されます。この封印チケットと TGS セッション鍵 $K_{c,tgs}$ で構成されるメッセージが作成されます。

注：

Kerberos バージョン 4 では、メッセージは次のとおりです。

$$\{K_{c,tgs}, n, \{T_{c,tgs}\}_{K_{tgs}}\}_{K_c}$$

それに対して Kerberos バージョン 5 では、メッセージはずっと簡単な形式です。

$$\{K_{c,tgs}, n\}_{K_c}, \{T_{c,tgs}\}_{K_{tgs}}$$

これにより、チケットの (不必要な) 二重暗号化が簡略化されます。

3 クライアント -> TGS

メッセージを受け取ると、クライアントは、クライアントと KAS にしか知らされていない秘密鍵 K_c を使ってメッセージを復号します。クライアントは、臨時 (n) が特定の要求に一致しているかどうかを確認し、今後 TGS とやり取りするときのために、セッション鍵 $K_{c,tgs}$ をキャッシュに入れます。

次に、クライアントは、メッセージを TGS へ送信します。このメッセージには、初期チケット $\{T_{c,tgs}\}_{K_{tgs}}$ 、サーバー名 (s)、臨時、および新しいオーセンティケーター A_c (タイム・スタンプを含む) が含まれます。 A_c は $\{c, nonce\}$ です。メッセージは、次のとおりです。

$\{A_c\}K_{c,tgs}, \{T_{c,tgs}\}K_{tgs}, s, n$

4 TGS -> クライアント

チケット許可サーバー (TGS) は、クライアント (c) から上記のメッセージを受け取り、まず自分の TGS 暗号鍵を使って封印チケットを復号します。(このチケットは、元々は、同じ鍵を使うステップ 2 の Kerberos 認証サーバーで封印されたものです。) 復号されたチケットから、TGS は TGS セッション鍵を入手します。TGS はこの TGS セッション鍵を使って、封印オーセンティケーターを復号します。(妥当性は、チケットとオーセンティケーターの両方のクライアント名、チケットの TGS サーバー名、そしてチケット、オーセンティケーター、受信メッセージで同じでなければならないネットワーク・アドレスを比較することによって検査されます。)

最後に、オーセンティケーター内の現在時刻を調べ、メッセージが最新であることを確認します。このときには、すべてのクライアントとサーバーは、指定されたある程度の許容範囲内でクロックを保守することが必要です。今度は、TGS は Kerberos データベースのメッセージからサーバー名を探し、指定したサービスの暗号鍵 (K_s) を入手します。

TGS は、クライアント (c) とサーバー (s) のために、新しい乱数セッション鍵 $K_{c,s}$ を構成し、以下を含む新しいチケット $T_{c,s}$ を作成します。

$K_{c,s}$ 、n、臨時、存続期間

次に、メッセージをアセンブルしてクライアントへ送信します。

注：

Kerberos バージョン 4 では、メッセージは次のとおりです。

$\{K_{c,s}, n, \{T_{c,s}\}K_s\}K_{c,tgs}$

それに対して Kerberos バージョン 5 では、メッセージはずっと簡単な形式です。

$\{K_{c,s}, n\}K_{c,tgs}, \{T_{c,s}\}K_s$

これにより、チケットの (不必要な) 二重暗号化が簡略化されます。

5 クライアント -> サーバー

クライアントは、このメッセージを受け取り、クライアントと TGS だけが共有している TGS セッション鍵を使ってメッセージを復号します。さらにクライアントは、このメッセージから、サーバー (s) と共有する新しいセッション

鍵 $K_{c,s}$ と、サーバーの秘密鍵 K_s を使って暗号化されるために復号できない封印チケットを入手します。

クライアントは、オーセンティケーターを作成し、新しいセッション鍵 $K_{c,s}$ を使って封印します。最後に、封印チケットとオーセンティケーターを含むメッセージをサーバー (s) へ送信し、サービスを要求します。

サーバー (s) は、このメッセージを受け取ったら、まず、サーバーと KAS だけが知っている暗号鍵を使って、封印チケットを復号します。次に、サーバーは、チケットに含まれている新しいセッション鍵を使ってオーセンティケーターを復号し、ステップ 4 で説明されていた同じ検査プロセスを実行します。

サーバーがクライアントを検査したら、クライアントがサーバーを検査するオプションがあります。これにより、侵入者がサーバーの偽名を使用することを防ぐことができます。クライアントでは、サーバーが (タイム・スタンプ値に示されたタイム・スタンプが付いた、クライアントのオーセンティケーターから) タイム・スタンプを含むメッセージを戻すことが必要です。このメッセージは、クライアントからサーバーへ渡されたセッション鍵を使って暗号化されます。

この構造での要点をいくつか要約しましょう。

- ワークステーションでエンド・サーバーを使う場合には、チケットが必要です。最初のチケット (初期チケットとも言う) 以外のすべてのチケットは、TGS から入手されます。最初のチケットは特別です。これは、TGS 自体のためのチケットであり、Kerberos 認証サーバーから入手されます。
- 各チケットは、チケットが割り振られるたびに割り当てられる 1 つのセッション鍵に関連付けられます。
- チケットは再利用できます。それぞれのチケットには存続期間があり、通常は 8 時間です。チケットの有効期限が切れたら、Kerberos に対してもう一度自分自身を識別させるため、ログイン名とパスワードを入力する必要があります。
- 再利用できるチケットとは異なり、新しいオーセンティケーターは、クライアントがサーバーとの新しい接続を開始するときに必要になります。オーセンティケーターは、その内部にタイム・スタンプを入れて運びます。オーセンティケーターは、発行後 2、3 分で有効期限が切れます。(このため、クライアントとサーバーとの間でクロックを同期する必要があります。)
- サーバーは、オーセンティケーターのタイム・スタンプがまだ有効な以前のクライアント要求のヒストリーを保守する必要があります。このようにすることにより、サーバーは、盗まれたチケットやオーセンティケーターから引き起こされた可能性のある重複する要求を拒否できます。

21.12.4 Kerberos データベース管理

Kerberos は、レルム内の各ユーザーおよびサービスのレコードを必要とします。各レコードには、以下のような必要な情報だけが保存されます。

- プリンシパル ID (c,s)
- このプリンシパルの秘密鍵 (K_c, K_s)
- この実体の有効期限の日付
- このレコードでの最終変更の日付
- このレコードを最後に変更したプリンシパルの実体 (c,s)
- このプリンシパルに付与されるチケットの最大存続期間 (存続期間)
- 属性 (未使用)
- 実装データ (外部には見えません)

秘密鍵フィールドは、マスター鍵がないときにデータベースを除去しても問題が生じないように、そのマスター鍵を使って暗号化されます。

このデータベースを管理する役割のエンティティは、Kerberos Database Manager (KDBM) です。1 つのレルムには 1 つの KDBM しかありませんが、複数の Kerberos Key Distribution Server (KKDS) を持ち、それぞれが Kerberos データベースのコピーを持つようにすることが可能です。これは、ユーザーが要求を送信する先の KKDS のグループの中から 1 つを選択できるように、可用性とパフォーマンスを向上させることを目的として行われます。KKDS は、読み取り専用操作を実行しますが、データベース全体を 1 日に 2、3 回コピーする、KDBM に対する現実化は行いません。これは、操作を単純化するために Kerberos で保護されたプロトコルを使って行われます。このプロトコルは、基本的に、ファイル転送操作を行う前の、チェックポイントおよびチェックサムを使った KDBM と KKDS との間の相互認証です。

21.12.5 Kerberos 許可モデル

Kerberos 認証モデルでは、リクエスターの実体を検査するためのサービスだけが許可されますが、リクエスターがそのサービスを使えるかどうかについての情報は与えられません。Kerberos 許可モデルは、各サービスがユーザーを知っているという原理に基づいているため、それぞれが自分自身の許可情報を保守するすることができます。ただし、Kerberos 認証システムは、許可の目的で使うことのできる情報およびアルゴリズムで強化することが可能です。(このことは、796 ページの 21.12.6 『Kerberos バージョン 5 の機能強化』に示されているように、バージョン 5 でより簡単になりました。) 次に、Kerberos は、ユーザー / クライアントが特定のサービスを使えるかどうかを検査できます。

明らかに、クライアント・アプリケーションとサーバー・アプリケーションの両方で、Kerberos 認証プロセスを処理できなくてはなりません。つまり、

クライアントとサーバーの両方が、Kerberos 化 されていなければなりません。

21.12.6 Kerberos バージョン 5 の機能強化

Kerberos バージョン 5 には、バージョン 4 と比較して数多くの強化点があります。重要な強化点のいくつかを紹介します。

- 暗号化を使うことは、個別のプログラム・モジュールに分けられたため、複数の暗号化システムをサポートできるようになりました。
- プロトコル・メッセージに示されるネットワーク・アドレスは、タイプおよび長さフィールドでタグ付けされるようになりました。これにより、複数のネットワーク・プロトコルをサポートできるようになりました。
- メッセージ・エンコードは、ISO 規格 8824 および 8825 に従った ASN.1 (Abstract Syntax Notation 1) 構文を使って記述されるようになりました。
- Kerberos バージョン 5 チケットが、新しい機能 (たとえば、レルム間の共同作用) をサポートするための強化された形式を備えるようになりました。
- 789 ページの 21.12.2 『命名』に述べられているように、プリンシパル ID の命名が変更されました。
- レルム間のサポートが強化されました。
- 許可およびアカウント作業の情報を、暗号化して、許可データ・フィールドのチケットの内部に入れて伝送できるようになりました。これにより、認証構造の機能が強化され、許可構造も組み込まれるようになりました。
- Kerberos バージョン 5 の実装への Generic Security Service API (GSSAPI) 向けのバインディングが提供されるようになりました。

21.13 リモート・アクセス認証プロトコル

企業イントラネットおよびインターネットヘリモート・ダイヤルインすることにより、リモート・アクセス・サーバーは、今日のインターネット・サービスの中でも非常に活気のある部分となりました。ますます多くのモバイル・ユーザーが、集約されたサイトのリソースだけではなく、インターネット上の情報源にアクセスできることを求めています。インターネットや企業のイントラネットが広く使われることによって、リモート・アクセス・サービスおよびデバイスの成長が活気付いています。ノートブック・コンピューターや E メール・アクセス用のパームトップ・デバイスといった、モバイル・コンピューティング・デバイスから、企業ネットワークのリソースへ簡単に接続したいという要求が高くなっています。

リモート・アクセスが出現したことにより、セキュリティの分野で重要な開発作業が生じました。リモート・アクセス・セキュリティの問題に取り

組むために、AAA (トリプル A) セキュリティー・モデルがセキュリティー業界で開発されました。だれ、何、そしていつという質問に対しては、認証、許可、そしてアカウント作業がそれぞれの答えになります。AAA セキュリティー・モデルにおけるこれら 3 つの A のそれぞれについて、簡単に説明します。

認証

これは、ユーザー (またはエンティティー) がだれかを判別するアクションです。認証は、多くの形式を取ることがあります。従来の認証では、名前と固定パスワードが使われています。ほとんどのコンピュータはこの方法で機能していますが、固定パスワードには、主にセキュリティーの分野で限界があります。現在の認証メカニズムの多くでは、一回限りのパスワードかユーザー確認のための質問への応答という照会が使われています。認証は、一般的に、ユーザーがマシンに初めてログインするとき、そのサービスを初めて要求するときに行われます。

許可

これは、ユーザーが許可されている行動を判別するアクションです。一般に、許可よりも認証が優先されますが、これは必要条件ではないことをもう一度強調しておきます。許可要求は、ユーザーが認証されていないことを示す場合があります。(こちら側は、ユーザーがだれかを知りません。) このようなケースでは、非認証ユーザーに対して問題のサービスを許可するかどうかを決定するのは、許可エージェントの役割です。現在のリモート認証プロトコルでは、許可は単に白か黒かの答えを出すだけでなく、特定ユーザー向けにサービスをカスタマイズすることも可能です。

アカウント作業

これは、通常は、認証と許可の後に行われる 3 番目のアクションです。しかし、認証または許可のいずれかがなければならぬわけではないことをもう一度強調しておきます。アカウント作業は、ユーザーが行っている行動、または行った行動 (あるいはその両方) を記録するアクションです。

分散したクライアント/サーバーのセキュリティー・データベース・モデルでは、数多くの通信サーバー (またはクライアント) が、1 つの集中データベース、または認証サーバーを使って、ダイヤルイン・ユーザーの実体を認証します。認証サーバーは、ユーザー、ユーザーのパスワード、およびアクセス権についてのすべての情報を保管します。セキュリティーが分散されている場合は、認証データのための中央設置場所が設けられるので、ユーザー情報がネットワークのあちこちのデバイスに散らばっているよりずっと安全です。1 つの認証サーバーで、数百の通信サーバーをサポートでき、数万のユーザーに対応することができます。通信サーバーは、認証サーバーに、ローカルにまたは WAN 接続を経由してリモートでアクセスすることができます。

いくつかのリモート・アクセス・ベンダーと Internet Engineering Task Force (IETF) は、このリモート・アクセス・セキュリティを促進する最前線におり、そのようなセキュリティ基準を標準化するための手段となっています。Remote Authentication Dial In User Service (RADIUS) と Terminal Access Controller Access Control System (TACACS) は、そのような共同での取り組みの中の 2 つであり、インターネット標準化組織とリモート・アクセス・ベンダーから生み出されたものです。

Remote Authentication Dial-In User Service (RADIUS) は、Livingston Enterprises 社によって開発された分散セキュリティ・システムです。RADIUS は、IETF の Network Access Server Working Requirements Group からの以前の勧告に基づいて設計されたものです。IETF Working Group for RADIUS は、RADIUS プロトコルの標準化に取り組むために、1996 年 1 月に結成されました。RADIUS は、現在、IETF 公認のダイヤルイン・セキュリティ・ソリューションになっています (RFC 2058 および RFC 2138)。

RADIUS と同様、Terminal Access Controller Access Control System (TACACS) は、業界標準のプロトコル仕様です (RFC 1492)。RADIUS と同様に、TACACS は、認証要求を NAS クライアントから受け取り、ユーザー名とパスワード情報を中央のセキュリティ・サーバーへ転送します。中央のサーバーは、TACACS データベースでも、外部のセキュリティ・データベースでも構いません。Extended TACACS (XTACACS) は、拡張機能付きの TACACS バージョンのことで、これは、Cisco 社が基本の TACACS プロトコルに追加して、高度な機能をサポートするようにしたものです。TACACS+ は、Cisco 社によるもう一つの拡張機能で、個別のアクセス・サーバー (TACACS+ サーバー) が、個別に認証、許可、およびアカウント作業のサービスを提供できるようにしたものです。

RADIUS および TACACS 認証サーバーは、関係するネットワークのセキュリティ機構に応じてさまざまな方法で設定できますが、ユーザーを認証するための基本的なプロセスは、本質的に同じものです。リモート・ダイヤルイン・ユーザーは、モデムを使い、アナログまたはデジタル・モデムが組み込まれたリモート・アクセス・サーバー (ネットワーク・アクセス・サーバー、つまり NAS と呼ばれる) に接続します。モデム接続が確立されたら、NAS はユーザーに名前とパスワードを要求します。次に NAS は、提供されたデータ・パケットからいわゆる認証要求を作成します。これは、認証要求を送信する特定の NAS デバイスを識別する情報、そのモデム接続で使われるポート、そしてユーザー名とパスワードで構成されます。

ハッカーによる盗聴から保護するため、RADIUS または TACACS クライアントとして機能する NAS は、パスワードを認証サーバーへ送信する前に暗号化します。主要なセキュリティ・サーバーに到達できない場合、セキュリティ・クライアントまたは NAS デバイスは、その要求を別のサーバーにルーティングできます。認証要求を受け取ったら、認証サーバーは、その要

求を検査してデータ・パケットを復号し、ユーザー名およびパスワード情報にアクセスします。ユーザー名とパスワードが正しければ、サーバーは認証確認通知パケットを送信します。この確認通知パケットには、ユーザーのネットワーク・リソース要件や許可レベルに関する情報など、別のフィルターを含めることができます。たとえば、セキュリティー・サーバーは、ユーザーが PPP を使った TCP/IP または IPX (あるいはその両方) を必要としていること、ユーザーがネットワークに接続するために SLIP を必要としていることなどを、NAS に通知できます。ここには、ユーザーがアクセスできる特定のネットワーク・リソースについての情報を含めることも可能です。

ネットワークでのスヌープを回避するために、セキュリティー・サーバーは、認証鍵または署名を送信し、自分自身をセキュリティー・クライアントに識別させます。NAS がこの情報を受け取ると、ネットワーク・サービスおよびリソースへの必要なアクセス権限をユーザーに許可する、必要な構成が可能になります。このログイン・プロセスの任意の時点で、必要なすべての認証条件が満たされると、セキュリティー・データベース・サーバーは、NAS デバイスに認証拒否メッセージを送信するため、ユーザーはそのネットワークへのアクセスを拒否されます。

21.14 Layer 2 Tunneling Protocol (L2TP)

L2TP では、PPP のトンネリングが可能です。PPP によってサポートされているプロトコルはすべてトンネリングできます。このプロトコルは、PPP 接続の範囲を広げたものです。リモート・ホストで開始してローカル ISP の存在位置 (PoP) で終了する代わりに、*仮想 PPP* リンクが、リモート・ホストから企業のゲートウェイに戻るまでの全経路を拡張するようになりました。L2TP トンネリングは、現在のところ IP/UDP 経由でサポートされています。この仕様については、RFC 2661 に述べられています。

L2TP は、Point-to-Point Tunneling Protocol (PPTP) と Layer 2 Forwarding (L2F; RFC 2341 で説明されている) という以前の 2 つのトンネリング・プロトコルを組み合わせることによって生み出された統一規格です。これら以前のプロトコルは、L2TP プロトコルほど完全なソリューションではありませんでした。片方は ISP によって作成されたトンネルを扱い、もう片方はリモート・ホストによって作成されたトンネルを扱っていました。L2TP では、ホスト作成のトンネルと ISP 作成のトンネルの両方がサポートされます。

L2TP には、仮想プライベート・ネットワークを作成する機能が追加されています。ここでは、複数のプロトコルや、内輪で対処がなされていた IP、IPX、および AT (AppleTalk) を利用することができます。さらに L2TP では、リモート・ユーザーが、ローカル ISP へ接続できるようになり、インターネットを経由してホーム・ネットワークへつなげられるようになりました。これにより、長距離の課金がなされなくなります。また、複数ボックス PPP 多重

リンクの問題を解決するメカニズムも備えられています。(同じ MP バンドルに宛てられた個別の物理ルーターへ接続されている呼び出しは、すべてのリンクで MP を終了できる同じエンドポイントへトンネリングすることが可能です。)

21.14.1 用語

プロトコルを説明する前に、いくつかの L2TP 用語を定義します。

L2TP アクセス・コンセントレーター (LAC)

PPP 操作と L2TP プロトコルの両方を処理できる 1 つ以上の公衆サービス電話網 (PSTN) かサービス総合デジタル網 (ISDN) 回線に付加されるデバイス。LAC は、L2TP で操作されるメディアを実装します。L2TP は、トラフィックを 1 つ以上の L2TP サーバー (LNS) に渡します。

L2TP ネットワーク・サーバー (LNS)

LNS は、PPP 端末にすることのできるすべてのプラットフォームで操作されます。LNS は、サーバー側の L2TP プロトコルを処理します。L2TP は、L2TP トンネルで到着する 1 つのメディアだけを利用するため、LNS は 1 つの LAN または WAN インターフェースだけを持つことができます。ただし、非同期、同期、ISDN、V.120 など、LAC でサポートされている任意の PPP インターフェースから到着する呼び出しを終了することは可能です。

ネットワーク・アクセス・サーバー (NAS)

一時的なオンデマンド・ネットワーク・アクセスをユーザーに提供するデバイス。このアクセスは、PSTN または ISDN 回線を使ったポイント間のアクセスです。

セッション (呼び出し)

L2TP は、ダイヤルイン・ユーザーと LNS との間で終端間 PPP 接続が試行されたとき、またはアウトバウンド呼び出しが開始されたときに、セッションを作成します。セッションのデータグラムは、LAC と LNS との間のトンネル経由

で送信されます。この LNS と LAC は、LAC に付属するユーザーごとに状態情報を保守します。

トンネル

トンネルは、LNS-LAC の組みによって定義されます。トンネルは、LAC と LNS との間で PPP データグラムを運びます。1 つのトンネルで、多くのセッションを多重化することができます。同じトンネル経由で制御接続を運用することにより、すべてのセッションとトンネル自体の確立、解放、および保守が制御されます。

属性値エアー (AVP)

メッセージ・タイプと本体をエンコードするときの統一された方式。この方式では、拡張性が最大になると同時に、L2TP の相互運用機能も許可されます。

21.14.2 プロトコルの概要

ホストとゲートウェイは同じ PPP 接続を共有するため、IP 以外のプロトコルをトランスポートする PPP の機能を利用できます。たとえば、L2TP トンネルを使い、リモート IP アクセスだけでなく、リモート LAN アクセスもサポートすることができます。図 312 には、基本的な L2TP 構成が示されています。

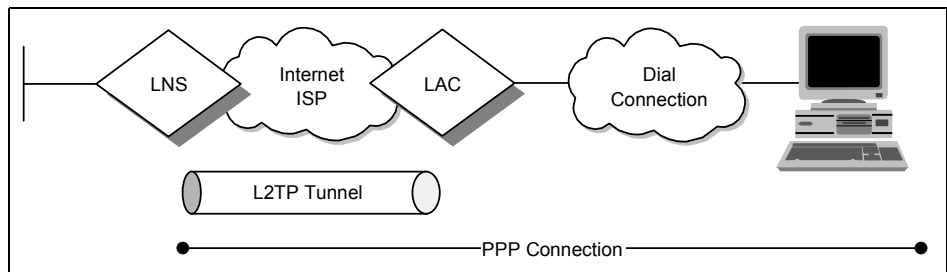


図 312 Layer 2 Tunnel Protocol (L2TP) シナリオ

図 312 によると、次のことが行われます。

1. リモート・ユーザーが PPP 接続を開始します。
2. NAS がその呼び出しを受け入れます。
3. NAS が許可サーバーを使ってリモート・ユーザーを識別します。
4. 許可が OK であれば、NAS/LAC は、企業の入り口で希望する LNS への L2TP トンネルを開始します。

5. LNS は、認証サーバーによってリモート・ユーザーを認証し、トンネルを受け入れます。
6. LNS が、呼び出しと L2TP トンネルの受け入れを確認します。
7. NAS が受け入れをログに記録します。
8. LNS が、リモート・ユーザーと PPP ネゴシエーションを交換します。
9. これで、リモート・ユーザーと LNS との間で端末間データがつながりました。

L2TP は、実際には別のバリエーションの IP カプセル化プロトコルです。

図 313 に示されているように、L2TP トンネルは、UDP パケット内で L2TP フレームをカプセル化し、ソースおよび宛先アドレスがトンネルのエンドポイントを定義している IP パケット内でさらにそれをカプセル化することによって作成されます。外部のカプセル化プロトコルが IP であるため、IPsec プロトコルをはっきりとこの複合 IP パケットに適用することができ、L2TP トンネル内を流れるデータが保護されます。AH、ESP、および ISAKMP/Oakley プロトコルは、すべてそのまま適用することができます。

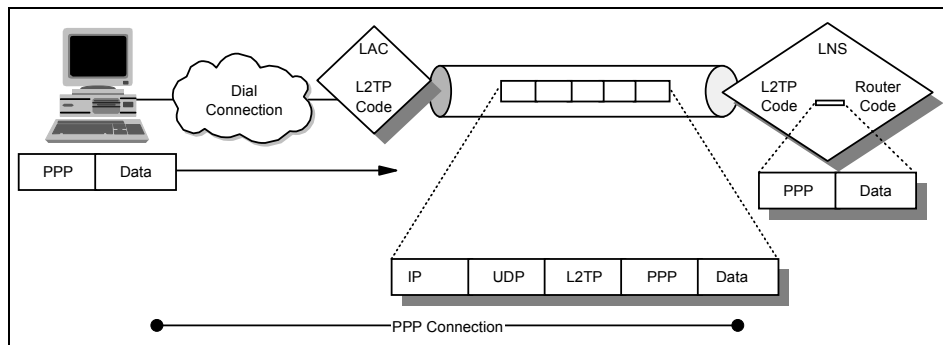


図 313 伝送時の L2TP パケット変更

L2TP は、UDP/IP 経由で運用でき、以下の機能をサポートしています。

- ・ 単一ユーザー・ダイヤルイン・クライアントのトンネリング
- ・ 認証されたユーザーのプロファイルに基づいて設定される 1 つの静的経路を持つルーターなど、小さなルーターのトンネリング
- ・ LAC から LNS への着信呼び出し
- ・ トンネルごとの多重呼び出し
- ・ PAP および CHAP 向けのプロキシ認証
- ・ プロキシ LCP
- ・ プロキシ LCP が LAC で使われていない場合の LCP 再始動
- ・ トンネルのエンドポイントの認証
- ・ プロキシ PAP パスワードを伝送するための隠された AVP
- ・ ローカル・レルム (つまり、user@realm) 参照テーブルを使ったトンネリング

- AAA サブシステムでの PPP ユーザー名ルックアップを使ったトンネリング (796 ページの 21.13 『リモート・アクセス認証プロトコル』を参照)

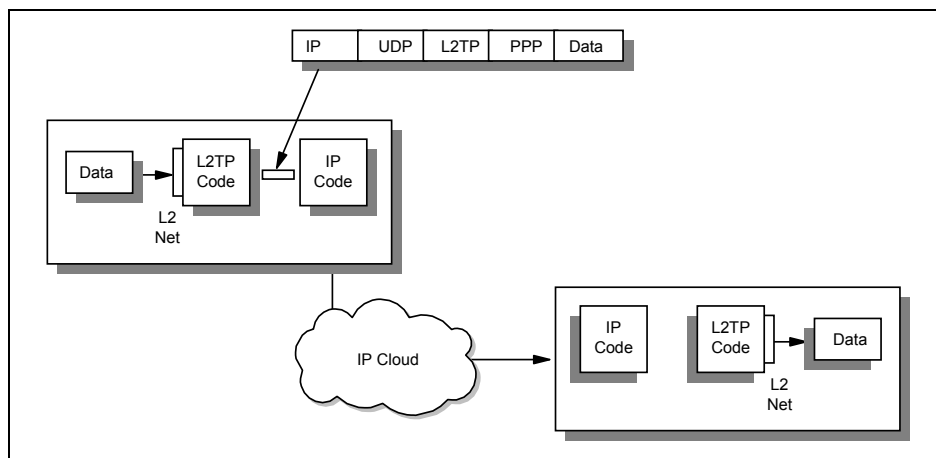


図 314 任意の IP ネットワークを経由した L2TP パケットの流れ

21.14.3 L2TP セキュリティ問題

L2TP では、低コストでのアクセス、マルチプロトコル・トランスポート、およびリモート LAN アクセスが可能です。暗号的に堅牢なセキュリティ機能が備わっていません。たとえば、以下のような点があります。

- 認証は、トンネルのエンドポイントの実体に対してだけ与えられますが、トンネル内を流れる個々のパケットごとに与えられるわけではありません。これでは、中継ユーザー攻撃やスプーフィング攻撃の場合に、トンネルが分かってしまう可能性があります。
- パケットごとの整合性がないため、L2TP トンネルが基礎となる PPP 接続のいずれかを終了できる偽の制御メッセージを生成することにより、サービス妨害攻撃を行うことが可能です。
- L2TP 自体には、ユーザー・データ・トラフィックを暗号化する機能は備わっていません。このため、データ機密性が問題となる場合に、障害となるデータ漏れが生じる可能性があります。
- PPP パケットのペイロードは暗号化できますが、PPP プロトコル一式には、自動的に鍵を生成したり自動的に鍵をリフレッシュするメカニズムがありません。このため、だれかが回線上で受信し、結局その鍵を破って、伝送されるデータにアクセスしてしまう可能性があります。

このような短所に気が付いたため、IETF の PPP Extensions Working Group は、これらの短所を直す方法を考え出しました。あちこちで重複した作業を

行うのではなく、L2TP 内で IPsec を使うことが推奨されています。このことは、RFC 2888 で説明されています。

要約すると、レイヤー 2 トンネル・プロトコルは、低コストのリモート・アクセスを提供する優れた方法であるということです。そして、IPsec と共に使うと、保護されたリモート・アクセスを可能にする優秀な技法となります。しかし、IPsec を補助的に使わないと、L2TP トンネル単独では適切なセキュリティが備わっていません。

21.15 Secure Electronic Transactions (SET)

SET は、1 つの電子クレジット・カード・システムの作成で協力することを目的として、MasterCard International と Visa International が合意に至った結論です。SET 以前は、各組織が独自のプロトコルを提示し、それぞれがさまざまなネットワーク会社およびコンピューター会社からの支援を受けてきました。現在は、主な会社のほとんどが SET 仕様を推し進めています (たとえば、IBM、Microsoft、Netscape および GTE など)。

続くいくつかのセクションでは、仕様を構成するコンポーネントとプロセスを上級者向けに説明します。SET の詳細については、MasterCard および Visa のホーム・ページを参照してください。

21.15.1 SET の役割

SET 仕様では、支払プロセスに関係したいいくつかの役割を定義しています。

マーチャント	これは、商品、サービス、または情報の販売者です。
決済機関	これは、クレジット・カード・サービスを提供し、お金のフローを維持する組織です。最も広く知られている決済機関は、MasterCard と Visa です。
発行者	これは、最初に購入者へカードを発行した組織です。通常は、これは仕様をよく知っているはずの銀行か他の金融機関です。
カード所有者	これは、発行者からクレジット・カードを与えられていて、自分の購買力を Web 上で行使したい Web サーファースのことです。
決済機関の支払ゲートウェイ	これは、マーチャントと、決済機関および発行者によって使われるバンク・カー

ド・ネットワークとの間に、インターフェースを提供します。バンク・カード・ネットワークはすでに存在していることを覚えておってください。決済機関の支払ゲートウェイは、確立されたバンク・カード・ネットワークに対し、インターネットとの間でのよく定義された保護インターフェースを提供します。決済機関の支払ゲートウェイは、決済機関に代わって運用されていますが、インターネット・サービス・プロバイダー (ISP) のような第三者組織によって提供されることがあります。

認証局

SET 処理では、公開鍵暗号方式を使用するので、システムの各要素は、1 つ以上の公開鍵証明書を必要としています。CA のいくつかのレイヤーについては、仕様の中で説明されています。(SET 証明書については、808 ページの 21.15.3 『SET 証明書の構造』で説明されています。)

21.15.2 SET 取引

SET 仕様では、購入、認証、返金といったさまざまな取引の流れが説明されています。図 315 には、一般的なオンライン購入に関係した取引が示されています。

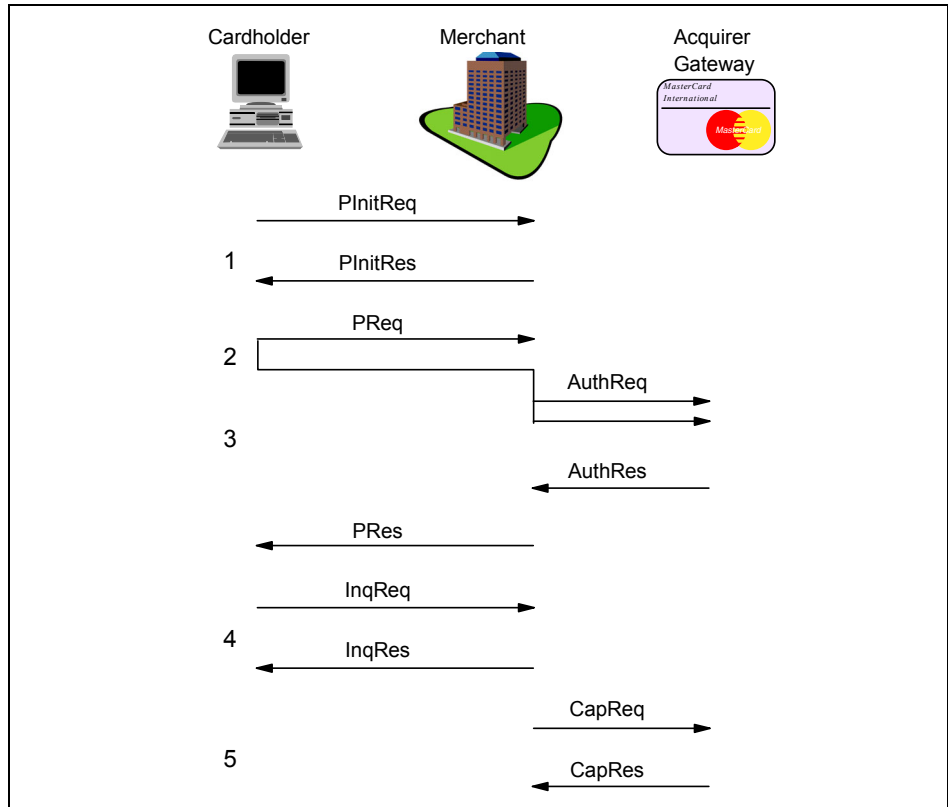


図 315 一般的な SET 取引の順序

この図は、以下の取引について示しています (それぞれの取引は、要求 / 応答の組みで構成されています)。

1. PInit

これは、使われるカードのブランドや、カード所有者が保持する証明書などの詳細を含め、システムを初期設定します。SET では、カード所有者が署名付きの証明書を持つことを絶対条件とはしていませんが、そうすることを強くお勧めします。カード所有者の証明書は、クレジット・カードのアカウント番号を公開鍵の所有者にバインドします。決済機関が、カード所有者の公開鍵で署名された所定のカード番号に関する要求を受け取る場合、その要求は本物のカード所有者からのものであることが分かります。細かく言うと、その要求は、カード所有者の鍵リングがインストールされていてその鍵リングを利用できるコンピューターからのものであることが分かります。その場合でも、そのコンピューターを盗んで鍵リング・パスワードを破った泥棒である可能性はあります。

2. 購入注文

これは、何かを購入するカード所有者からの実際の要求です。この要求メッセージは、実際には2つのメッセージが組み合わさったものです。1つは注文命令 (OI) で、マーチャントへクリアで送信されるもので、もう1つは購入命令 (PI) で、マーチャントが決済機関の支払ゲートウェイに渡すものです。PI は決済機関の公開鍵で暗号化されるため、マーチャントはそれを読むことができません。マーチャントは、後で決済機関へ伝送するために、メッセージを保管しておきます。PI には OI のハッシュも含まれているため、ペアとして処理できるのは、この2つのメッセージだけです。カード番号は、要求の PI 部分にだけ記されていることに注意してください。つまり、マーチャントがそれにアクセスすることはないということです。したがって、詐欺を働くユーザーが偽のストアフロントを設定してクレジットカード情報を集めることを防ぐことができます。

購入注文には応答があります。これは通常は、決済機関の承認が与えられてから、(ここで示されているように) 送信されます。しかし、マーチャントは、許可が得られる前にカード所有者との取引を完了させることができません。この場合、カード所有者には、要求の許可が受け入れ保留中であることを示すメッセージが表示されます。

3. 許可

この要求では、マーチャントは、決済機関の支払ゲートウェイを介して、要求を許可することを決済機関に要請します。メッセージには、購入商品と価格の説明が含まれます。さらに、カード所有者が送信した購入注文からの PI も含まれます。このようにして、決済機関は、マーチャントとカード所有者の両方が購入商品と数量について合意していることを知ることになります。

決済機関が要求を受け取ると、決済機関は既存のバンク・カード・ネットワークを使って要求を許可し、適切な応答を戻します。

4. 照会

カード所有者は、自分の要求がどのように処理されているかを知ることができます。SET 仕様では、そのような目的での照会取引が用意されています。

5. 資金化

ここまです、お金の移動はありません。マーチャントからの資金化要求があると、決済機関は、以前に許可した金額をマーチャントの口座に振り込みます。

実際には、資金化は、許可要求 / 応答の一部に組み込むことができます (上記を参照)。しかし、マーチャントが資金を後で資金化できる状況も存在します。たとえば、ほとんどのメール注文操作では、商品が実際に出荷

されるまで、クレジット・カード口座から引き落とされることはありません。

SET 仕様には他の取引もいくつか記されていますが、上記の要約では、ベースとなる原則を紹介しました。

21.15.3 SET 証明書の構造

SET 仕様では、世界中の何十万もの参加者を想定しています。参加者のそれぞれは、少なくとも 1 つの公開鍵証明書を持っている可能性があります。実際に、いくつかのケースでは、プロトコルがエンティティーに複数の証明書を持たせる場合があります。たとえば、決済機関の支払ゲートウェイでは、署名付きメッセージ用に 1 つと、暗号化用に 1 つ必要です。

そのようにスケールが大きいときの鍵管理では、簡単で平易な証明書構造以上のものが必要になります。SET 向けの認証局の組織は、図 316 に示されています。

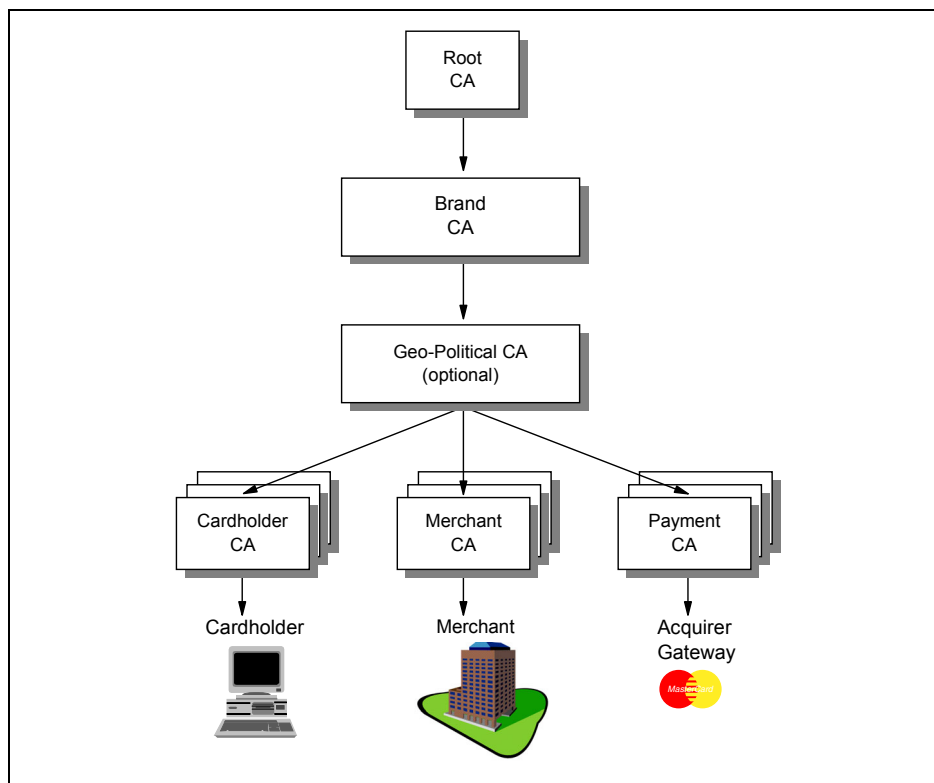


図 316 SET 認証局

証明書チェーンの最上位では、極端に厳しい仕組みになっていて、ルート認証局はオフラインのままにされています。ここには、新しいクレジット・カード・ブランドが SET コンソーシアムに加わる場合にのみアクセスされます。階層の次のレベルでは、ブランド・レベル CA も非常に安全です。これらは、各クレジット・カード・ブランドごとに独立して管理されます。

別の運用ポリシーを持つそれぞれのブランドの下では、いくらかの柔軟性が許可されています。たとえば、地域や国に基づいて CA を設定することが可能です。CA 階層の基底にあるのは、マーチャント、カード所有者、および決済機関の支払ゲートウェイに証明書を提供する CA です。SET 仕様には、マーチャントとカード所有者が証明書をオンラインで要求するためのプロトコルが用意されています。SET では、カード所有者が自分自身の証明書を所持することを勧めているため、そのプロセスを簡単にすることは大切なことです。また、カード所有者が CA の Web サイトにサーフィンし、要求証明書オプションを選択してブラウザで認証要求アプリケーションを起動し、それからフォームに記入して証明書要求を送受信することを想定しています。

もちろん、システムで証明書を簡単に作成できる場合、盗難が生じたときや他のセキュリティー・ブリーチ（抜け穴）が生じたときに、その証明書を簡単に取り消せるようにする必要もあります。SET 仕様には、この目的での証明書更新プロトコルと証明書取り消しプロトコルがいくつか含まれています。証明書を要求するときのメカニズムは簡単かもしれませんが、それでもこの点でユーザーを教育する必要があります。たとえば、カード所有者は、自分の財布が盗まれたときにクレジット・カード会社へ知らせるべきであることを知っていますが、自分のコンピューターが盗まれたときにもクレジット・カード会社へ知らせるべきであることはあまり知られていません。しかし、そのコンピューターに、秘密鍵と証明書が含まれた鍵リング・ファイルがある場合、泥棒はカード所有者の費用でショッピングすることも可能になります。

21.16 参照資料

以下の RFC には、この章で示された TCP/IP セキュリティー・ソリューションについての詳細が示されています。

- RFC 1492 – An Access Control Protocol, Sometimes Called TACACS
- RFC 1510 – The Kerberos Network Authentication Service (V5)
- RFC 1579 – Firewall-Friendly FTP
- RFC 1928 – SOCKS Protocol Version 5
- RFC 1929 – Username/Password Authentication for SOCKS V5
- RFC 1961 – GSS-API Authentication Method for SOCKS Version 5

- RFC 2003 – IP Encapsulation within IP
- RFC 2104 – HMAC: Keyed-Hashing for Message Authentication
- RFC 2138 – Remote Authentication Dial In User Service (RADIUS)
- RFC 2246 – The TLS Protocol Version 1.0
- RFC 2313 – PKCS 1: RSA Encryption Version 1-5
- RFC 2314 – PKCS 10: Certification Request Syntax Version 1-5
- RFC 2315 – PKCS 7: Cryptographic Message Syntax Version 1-5
- RFC 2401 – Security Architecture for the Internet Protocol
- RFC 2402 – IP Authentication Header
- RFC 2403 – The Use of HMAC-MD5-96 within ESP and AH
- RFC 2404 – The Use of HMAC-SHA-1-96 within ESP and AH
- RFC 2405 – The ESP DES-CBC Cipher Algorithm With Explicit IV
- RFC 2406 – IP Encapsulating Security Payload (ESP)
- RFC 2407 – The Internet IP Security Domain of Interpretation for ISAKMP
- RFC 2408 – Internet Security Association and Key Management Protocol
- RFC 2409 – The Internet Key Exchange (IKE)
- RFC 2410 – The NULL Encryption Algorithm and Its Use With IPsec
- RFC 2411 – IP Security Document Roadmap
- RFC 2412 – The OAKLEY Key Determination Protocol
- RFC 2661 – Layer Two Tunneling Protocol "L2TP"
- RFC 2888 – Secure Remote Access with L2TP
- RFC 3022 – The IP Network Address Translator (NAT)

第 22 章 サービス品質

インターネットを含む IP ベースのネットワークを使用する機会が増えるにつれて、特定のアプリケーションに必要なネットワーク・リソースを備えることがますます重要になっています。このことによってアプリケーションの優劣が決まるようになってきているので、インターネットワークの処理がアプリケーションに優先的に要求されるようになっていきます。さらにアプリケーションには、待ち時間の短縮や帯域幅の拡大のリアルタイム要件などさまざまな要求があります。

この章では、トラフィック優先順位、つまりサービス品質 (QoS) について説明します。インターネットやイントラネットでは QoS が求められる理由について説明し、TCP/IP ネットワークで QoS を実装する以下の 2 つの主要な方法を取り上げます。

- 統合サービス
- DiffServ

22.1 QoS が必要な理由

今日のインターネットやイントラネットでは、帯域幅が重要な問題になっています。インターネットを個人用や社用で使用する人はますます増えています。インターネットを介して伝送されるデータの量は、指数関数的に増えています。IP 電話およびビデオ会議システムなどのマルチメディア・アプリケーションには、インターネットの初期に使用されていたアプリケーションよりもはるかに大きな帯域幅が必要です。WWW、FTP、または TELNET などの従来のインターネット・アプリケーションは、パケット・ロスを経容できませんでしたが、さまざまな遅延の影響はあまり受けませんでした。これとは正反対に、最新のアプリケーションは、適切な量であればパケット・ロスを補正しますが、通常はさまざまな遅延の影響を非常に受けやすくなっています。

したがって、帯域幅の制御がないと、これらのリアルタイム・ストリームの品質は、現在使用できる帯域幅に依存することになります。帯域幅が狭かったり不安定だったりして、ドロップアウトやハングなどが起こると、リアルタイムの伝送の品質が悪くなります。リアルタイム・プロトコル (RTP) を使用した伝送の品質でさえ、基礎となる IP 送達サービスの使用状況に依存します。

したがって、インターネットに関してリアルタイム・アプリケーションの一定のサービス品質 (QoS) を保証するには、特定の概念が必要です。QoS とは、特定のデータ・ストリームの品質 (帯域幅、バッファ使用量、優先順位、CPU 使用量など) を記述したパラメーターの集合といえます。基本的な

IP プロトコル・スタックには 1 つだけ QoS が備えられており、ベストエフォートと呼ばれています。パケットは、特別な帯域幅や最小限の時間遅延が保証されていなくても、ポイント間を伝送されます。ベストエフォート・トラフィック・モデルを使用すると、インターネット要求は先着順サービス・ストラテジーで処理されます。つまり、すべての要求の優先順位が同じで、1 つずつ順番に処理されます。特定の接続用に帯域幅が予約されたり、特殊な要求に優先順位が付けられることはありません。したがって、予測可能なインターネット・サービスを備えるために、新しいストラテジーが開発されました。

今日では、インターネットや IP ベースのインターネットワークで QoS を実現するために、統合サービスと DiffServ という 2 つの主要な基本原理があります。

統合サービス

統合サービスは、IP ネットワーク・モデルを機能強化し、特定のフローについてリアルタイム伝送と保証された帯域幅をサポートします。この場合のフローとは、単一のユーザー・アクティビティーの結果生じた、同一の QoS が必要な、固有の送信側から固有の受信側への関連データグラムの区別可能なストリームと定義しています。

たとえば、あるフローが特定のホストのペア間の 1 つのビデオ・ストリームで構成されるとします。両方向のビデオ接続を確立するには、2 つのフローが必要です。データ・フローを開始するアプリケーションごとに、このフローに必要な QoS を指定できます。この接続の QoS を予約済みにするなどしてビデオ表示の持続を確約するには、ビデオ会議ツールには、最低でも 128 kbps の帯域幅と、100 ミリ秒のパケット遅延が必要です。

DiffServ

DiffServ 機構では、フロー単位のシグナル方式は使用されないで、結果として、ルーティングの基礎構造でフロー単位の状態が消費されるということはありません。DiffServ レベルはさまざまなユーザーのグループに割り振ることができます。したがって、すべてのトラフィックを、さまざまな QoS パラメーターを指定したグループやクラスに分散できます。そのため、統合サービスと比較すると、保守のオーバーヘッドが少なくなります。

22.2 統合サービス

統合サービス (IS) モデルは、計画されていた IS インターネットの根本原理になるよう、IETF 作業チームによって定義されました。このインターネット方式モデルには、現在使用されているベストエフォート・サービスや、新しいリアルタイム・サービスが組み込まれており、インターネットやインターネットワーク上で帯域幅を予約する機能を備えています。

IS は、QoS を保証する必要がある、リアルタイム・マルチメディアなどの新しいアプリケーションの、ネットワークとリソースの使用効率を最適化するために開発されました。ルーティング遅延と輻輳ロスのため、現行のベストエフォート・インターネットでは、リアルタイム・アプリケーションは作動状況は良好ではありません。ビデオ会議、ビデオ・ブロードキャスト、および音声会議ソフトウェアには、許容できる品質のビデオや音声を備えるには、保証された帯域幅が必要です。統合サービスを使用すると、インターネット・トラフィックを標準的なベストエフォート・トラフィックに分割して、従来どおり使用できるようにし、アプリケーション・データは保証された QoS でフローします。

統合サービス・モデルをサポートするには、インターネット・ルーターが、サービス・モデルに従ってフローごとに該当する QoS を提供できるようになっていなければなりません。さまざまなサービス品質を提供するルーター機能のことを、**トラフィック制御**といいます。トラフィック制御は以下のコンポーネントで構成されています。

パケット・スケジューラー

パケット・スケジューラーは、キュー管理とさまざまなスケジューリング・アルゴリズムを使用して、ホストやルーター中のさまざまなパケット・ストリームの転送を、それらのパケット・ストリームのサービス・クラスに基づいて管理します。パケット・スケジューラーは、フローごとにパケット送達が QoS パラメーターに対応していることを確認しなければなりません。またスケジューラーは、トラフィックのポリシングやシェーピングを行って、特定のレベルのサービスに準拠させることもできます。パケット・スケジューラーは、パケットがキューに入れられるポイントに実装されなければなりません。普通このポイントは、出力ドライバー・レベルのオペレーティング・システムで、リンク層プロトコルに対応しています。

パケット分類機能

パケット分類機能は、特定のレベルのサービスを受け取る、ホストやルーター中の IP フローのパケットを識別します。トラフィック制御を効果的に行うために、分類機能により個々の着信パケットが特定のクラスにマップされます。同じクラスに分類されたすべてのパケットには、スケジューラーによって同じ処理が行われます。クラスの選択は、既存のパケット・ヘッダー中のソースおよび宛先 IP アドレスとポート番号、および個々のパケットに追加しなければならない追加の種別番号に基づいて行われます。クラスを広範囲なフローのカテゴリーに対応させることができます。

たとえば、参加者が複数のビデオ会議からのビデオ・フローを、すべて 1 つのサービス・クラスに所属させることができます。さらに 1 つのフローだけ特定のサービス・クラスに所属させることもできます。

アドミッション制御 アドミッション制御には、ルーターで使用され、新しいフローの QoS 要求を受諾できるだけのルーティング・リソースがあるかどうかを判断する、決定アルゴリズムが含まれています。空いているルーティング・リソースが足りない場合は、新しいフローを受諾するとそれ以前の保証に悪影響があるため、その新しいフローを拒否しなければなりません。新しいフローを受諾すると、ルーター中の予約インスタンスによりパケット分類機能とパケット・スケジューラーが割り当てられ、このフローの QoS 要求が予約されます。アドミッション制御は、予約パスにのって個々のルーターで呼び出され、ホストでリアルタイム・サービスが要求される時点で受諾 / 拒否の決定が行われます。アドミッション制御アルゴリズムは、サービス・モデルとの整合性がなければなりません。

時折アドミッション制御はポリシー制御と混同されることがあります。ポリシー制御はパケット・スケジューラーによって処理されるパケット単位の機能です。ポリシー制御により、約束されているトラフィック特性にホストが違反していないことが確約されます。それでも、QoS が確実に保証されるように、アドミッション制御によりリソースの予約に関する管理ポリシーが実施されます。ポリシーの一部は、要求された予約に関するユーザー認証の検査に使用されます。無許可の予約要求は拒否できます。結果として、アドミッション制御は、将来のインターネット・リソースに関するコストを計算する上で重要な役割を果たすことができます。

図 317 は、ホストとルーターにおける統合サービス・モデルの操作を示しています。

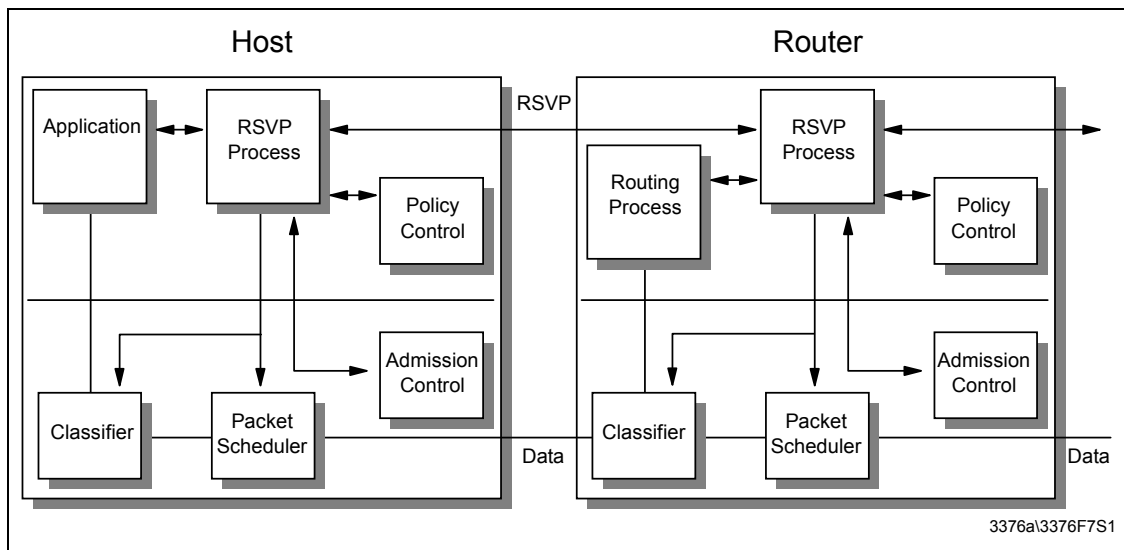


図 317 統合サービス・モデル

統合サービスでは、予約メッセージのシグナルとして、Resource Reservation Protocol (RSVP) を使用します。IS インスタンスは、RSVP を介して通信し、フローのパスにのってエンドポイント・ホストとルーターでフロー固有の状態を作成して保守します。RSVP プロトコルの詳細記述については、820 ページの 22.2.2 『Resource Reservation Protocol (RSVP)』を参照してください。

図 317 に図示されているように、予約済みのフロー中でデータ・パケットを送信したいアプリケーションは、予約インスタンス RSVP と通信します。RSVP プロトコルは、要求されている QoS を使用してフロー予約のセットアップを試行します。この QoS は、アプリケーションがポリシー制限を満たしており、ルーターがこの要求されている QoS を処理できる場合に受諾されます。RSVP は、個々のノード中のパケット分類機能とパケット・スケジューラーに、このフローのパケットを適切に処理するように指示します。アプリケーションがデータ・パケットを 1 つ目のノード (要求されている QoS に従って特定のサービス・クラスにこのフローをマップしている) 中の分類機能に送達すると、このフローは送信側の IP アドレスによって認識され、パケット・スケジューラーに伝送されます。パケット・スケジューラーはパケットを、それらのパケットのサービス・クラスに従って、次のルーター (最終的には受信側のホスト) に転送します。

RSVP はシンプレックス・プロトコルなので、QoS 予約は送信ノードから受信ノードへの 1 方向だけ行われます。例のアプリケーションで、データ・フローの予約を取り消したい場合は、メッセージを予約インスタンスに送信し

ます。予約インスタンスはパスにのっとってすべてのルーター中の予約済みの QoS リソースを解放します。解放後、他のフローでリソースを使用できます。IS 仕様は RFC 1633 で定義されています。

22.2.1 サービス・クラス

統合サービス・モデルでは、統合サービス IETF 作業チームによって定義された、さまざまなサービス・クラスが使用されます。これらのサービス・クラスは、アプリケーションに応じて、QoS 制御に関する密または疎の境界を備えます。現行の IS モデルには、*保証されたサービス* が組み込まれており、これは RFC 2212 で定義されています。また、*負荷コントロール・サービス* も組み込まれており、これは RFC 2211 で定義されています。これらのサービス・クラスについて理解するには、いくつかの用語について知っている必要があります。

IS モデルにはフロー単位の予約が備えられているので、個々のフローはフロー記述子に割り当てられます。フロー記述子は、特定のデータ・パケットのフローのトラフィックと QoS 特性を定義します。図 318 に図示されているように、IS 仕様では、フロー記述子はフィルター仕様 (filterspec) とフロー仕様 (flowspec) から構成されています。

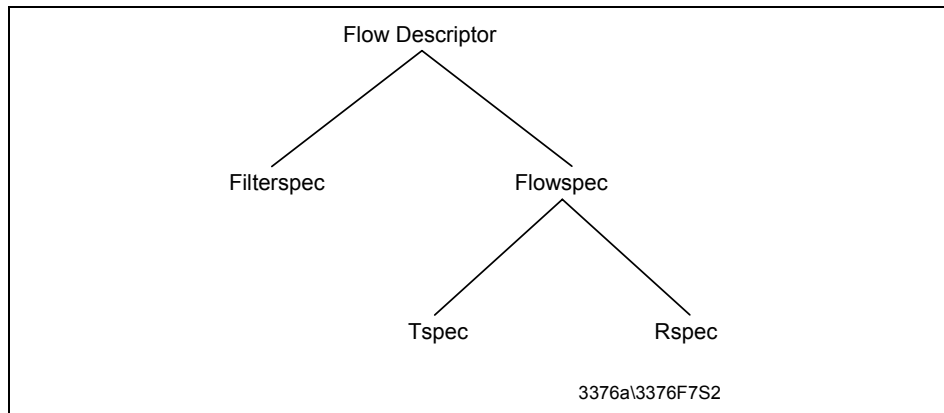


図 318 フロー記述子

filterspec は、送信側の IP アドレスとソース・ポートを使用して、特定のフローに属するパケットを識別するために使用されます。filterspec からの情報はパケット分類機能で使用されます。flowspec には、*呼び出し情報*というパラメータの集合が含まれています。呼び出し情報は以下の 2 つのグループに分類できます。

- トラフィック仕様 (Tspec)
- サービス要求仕様 (Rspec)

Tspec は、要求されているサービスのトラフィック特性を記述します。IS モデルでは、この Tspec はトークン・バケット・フィルタとして表されています。この原理は、周期的な時間間隔で文字 (トークン) をバッファ (バケット) に追加し、バケット中のトークンの数がデータ・パケットのバケット長以上の場合だけ、データ・パケットが送信側から送信されるようにします。このストラテジーを使用すると、ネットワーク上の 2 つのデータ・パケットの間の時間間隔を正確に制御できます。トークン・バケット・システムは 2 つのパラメーターで指定します。1 つは トークン率 r で、これはトークンをバケットに入れる際の比率を表し、もう 1 つは バケット能力 b です。 r と b の両方とも正でなければなりません。図 319 には、トークン・バケット・モデルが図示されています。

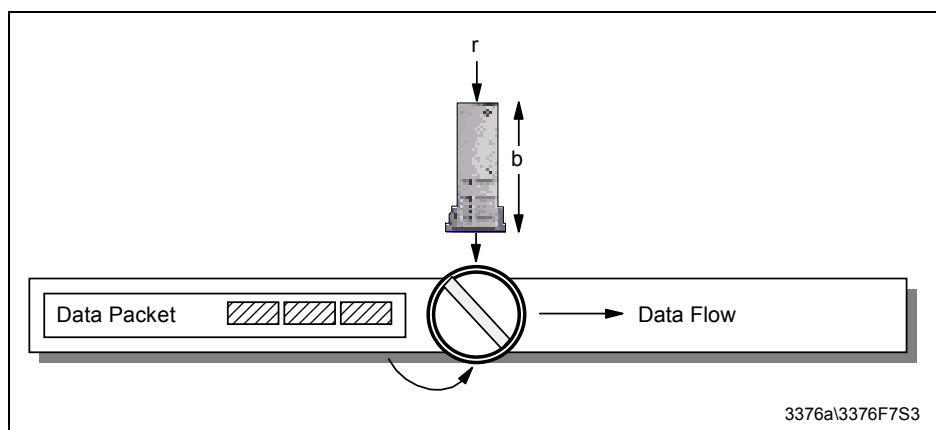


図 319 トークン・バケット・フィルタ

パラメーター r は、長期データ転送速度を指定し、1 秒当たりの IP データグラムのバイト数で測定されます。このパラメーターの値の範囲は、1 バイト / 秒 ~ 40 テラバイト / 秒です。パラメーター b は、システムで許可されるバースト・データ転送速度を指定し、バイト数で測定されます。このパラメーターの値の範囲は、1 バイト ~ 250 ギガバイトです。これらのパラメーターで指定できる値の範囲は、将来のネットワーク・テクノロジーを想定して、意図的に大規模にされています。ネットワーク・エレメントは、全範囲の値をサポートすることが想定されていません。トークン・バケット・フィルタを通るトラフィックは、時間枠 T (秒) 全体で、送信されるデータの量が、 $rT + b$ (r および b はトークン・バケットのパラメーター) を超えないという規則に従っていなければなりません。

Tspec の一部として、他にも 2 つのトークン・バケット・パラメーターがあります。それは、最小ポリシング単位 m と最大パケット・サイズ M です。パラメーター m は、最小 IP データグラム・サイズをバイト単位で指定します。トークン・バケット・フィルタよりも小さいパケットは、サイズ m として

カウントされます。パラメーター M は、Tspec に準拠する最大パケット・サイズをバイト単位で指定します。要求された最大パケット・サイズがリンクの MTU サイズより大きい場合は、ネットワーク・エレメントは、サービス要求を拒否しなければなりません。要約すると、トークン・バケット・フィルターとはポリシング機能の一種で、トラフィック仕様に準拠したパケットを準拠しないパケットから分離します。

サービス要求仕様 (Rspec) は、アプリケーションで要求される特定のフローのサービス品質を指定します。この情報は、サービスのタイプと、QoS を要求するアプリケーションの需要に依存します。この情報は特定の帯域幅、最大パケット遅延、または最大パケット・ロス率から構成されます。IS を実装すると、Tspec および Rspec からの情報はパケット・スケジューラーで使用されます。

22.2.1.1 負荷コントロール・サービス

負荷コントロール・サービスは、リアルタイム・アプリケーションなどの、インターネット中の過負荷条件の影響を非常に受けやすいアプリケーションのクラスをサポートすることを想定しています。この種のアプリケーションは、負荷の少ないネットワークでは良好ですが、過負荷条件では即時に性能が低下します。アプリケーションで負荷コントロール・サービスを使用すると、ネットワーク負荷が増えてもパフォーマンスデータ・フローのパフォーマンスは低下しません。

負荷コントロール・サービスのサービス・レベルは 1 つのだけになっており、意図的に最小限に抑えられています。仕様中にオプションの機構や機能はありません。このサービスには 1 つだけ機能があります。負荷の軽いネットワークでは、このサービスはベストエフォート・サービスに似ています。つまり、負荷の軽い条件では、負荷コントロール・サービスを使用して QoS 予約するアプリケーションには、制御されていない (ベストエフォート) トラフィックに備えられるサービスとほとんど同じサービスが備えられます。これに関連して、*負荷の軽い条件* とは、伝送されるパケットが非常に高いパーセント比率で正常に宛先に送達し、送達したパケットの伝送遅延が非常に高いパーセント比率で最小伝送遅延を大幅に超過しないということです。

ネットワーク中の、負荷コントロール・サービスに関する要求を受諾する個々のルーターで、適切な帯域幅とパケット処理リソースが QoS 予約要求の処理に使用できるようになっていることを確認しなければなりません。アクティブなアドミッション制御を使用してこの操作を行えます。Tspec で表されている新しい QoS 予約をルーターで受諾する前に、リンク帯域幅、ルーター、スイッチ・ポート・バッファ・スペースなどのすべての重要なリソースや、パケット転送の計算能力を考慮しなければなりません。

負荷コントロール・サービス・クラスでは、帯域幅、遅延、またはロスなどの制御パラメーターに特定のターゲット値を受諾したり使用したりできません。

ん。負荷コントロール・サービスを使用するアプリケーションは、少量のパケット・ロスやパケット遅延に対して保護しなければなりません。

負荷コントロール・サービスを使用した QoS には、トークン・パケット・パラメーター r と b 、および最小ポリシング単位 m と最大パケット・サイズ M から成る Tspec が必要です。Rspec は必要ありません。なぜなら負荷コントロール・サービスには、固定帯域幅を予約したり最小パケット遅延を保証したりする機能は備えられていないからです。セットアップ時に指定した Tspec に準拠したトラフィックに限り、負荷コントロール・サービスにより QoS 制御が備えられます。したがって、サービスの保証は、時間枠 T 全体で送信されるデータの量が $rT+b$ を超えないというトークン・パケット規則を守っているパケットだけに適用されます。

負荷コントロール・サービスは、音声およびビデオ会議ソフトウェアなどの、妥当な量のパケット・ロスや遅延を許容できるアプリケーション用に設計されています。

22.2.1.2 保証されたサービス

保証されたサービス・モデルは、保証された送達時間内にデータグラムが着信することを確認する機能を備えています。つまり、トラフィック仕様に準拠するフローのすべてのパケットは、フロー記述子で指定された最大遅延時間以内に着信します。保証されたサービスは、データグラムがソースから伝送されてから特定の時間以内に受信側に着信する必要のあるアプリケーションで使用されます。

たとえば、ストリーミング・テクノロジーを使用するビデオおよび音声ブロードキャスト・システムなどの、リアルタイム・マルチメディア・アプリケーションでは、適切な再生時刻より後に着信したデータグラムは使用できません。金融データ（共用部分）のリアルタイム配布などの、厳しいリアルタイム要件があるアプリケーションも、保証されたサービスは必要ありません。保証されたサービスにより、ジッター（データグラム遅延の最大値と最小値の差）は最小になりませんが、最大キューイング遅延を制御できます。

保証されたサービス・モデルは、ネットワークの究極の遅延制御を表します。他のサービス・モデルに備えられている遅延制御の方が、遅延の制限が緩くなっています。したがって、保証されたサービスは、予約パスにのっとった個々のルーターによって備えられる場合だけ効果を発揮します。

保証されたサービスにより、アプリケーションで遅延に対する強力な制御を行えるようになります。IP ネットワーク中の遅延には、固定の伝送遅延と可変のキューイング遅延の 2 つの部分があることを知っておくのは重要です。固定遅延は、選択されたパスに依存します。このパスは保証されたサービスではなくセットアップ機構によって決まります。IP ネットワーク中のすべてのデータ・パケットの遅延は、光の速さと、ルーティング・パス上のすべて

のルーター中のデータ・パケットの送受反転時間によって限定され、最小になります。キューイング遅延は、保証されたサービスによって決まり、2つのパラメーターによって制御されます。1つはトークン・バケット (特にバケット・サイズ b) で、もう1つは予約用に要求される帯域幅 R です。これらのパラメーターは液体モデルを構成するために使用され、このモデルは保証されたサービスを使用するフローの終端間の振る舞いを制御します。

液体モデルは、帯域幅が R の送信側と受信側の間の専用リンクによって設けられるサービスを指定します。液体モデルでは、このフローのサービスは、他のフローのサービスから完全に独立しています。保証されたサービスの定義は、トークン・バケット (r, b) に従い、帯域幅 R の回線によってサービス提供される液体フロー遅延が、 b/R によってバインドされる結果に依存します (R が r 以上の場合)。保証されたサービスは、この動作をサービス率 R で概算します。この場合の R は、専用回線の帯域幅ではなくルーティング・パスを介した共用帯域幅です。

保証されたサービス・モデルでは、 T_{spec} と R_{spec} を使用してフロー予約がセットアップされます。 T_{spec} はトークン・バケット・パラメーターによって表されます。 R_{spec} には、パラメーター R が含まれていて、このパラメーターはフロー予約の帯域幅を指定します。保証されたサービス・モデルは RFC 2212 で定義されています。

22.2.2 Resource Reservation Protocol (RSVP)

統合サービス・モデルでは、Resource Reservation Protocol (RSVP) を使用して、QoS 予約のセットアップや制御を行います。RSVP は、RFC 2205 で定義されており、その状況は提案された規格に従っています。RSVP はインターネット・プロトコルの一種で、ルーティング・プロトコルではないので、RSVP を操作するには既存のルーティング・プロトコルが必要です。RSVP プロトコルは IP や UDP より優先的に実行され、予約パス上のすべてのルーターに実装されます。RSVP の鍵となる概念は、フローと予約です。

RSVP 予約は、ルーターを介した特定のパス上の特定のデータ・パケット・フローに適用されます。811 ページの 22.1 『QoS が必要な理由』で説明されているように、フローは、固有の送信側から固有の受信側への関連データグラムの区別可能なストリームと定義されています。受信側がマルチキャスト・アドレスの場合は、フローを複数の受信側に送達できます。RSVP は、ユニキャスト・フローやマルチキャスト・フローと同じサービスを提供します。RSVP からの個々のフローは、宛先 IP アドレスと宛先ポートによって識別されます。すべてのフローには専用のフロー記述子があり、この記述子には特定のフローに必要な QoS が含まれています。RSVP プロトコルはフロー記述子の内容を認識しません。これは RSVP によって不透明オブジェクトとして転送され、ルーターのトラフィック制御機能 (パケット分類機能とパケット・スケジューラー) に送達されて処理されます。

RSVP はシンプレックス・プロトコルなので、予約は 1 方向だけ行われます。ビデオおよび音声会議などの全二重接続（個々の送信側が受信側でもある）の場合、端末ごとに 2 つの RSVP セッションをセットアップする必要があります。

RSVP プロトコルは受信側で開始されます。RSVP シグナル・メッセージを使用して、送信側は特定の QoS を受信側に送信し、受信側は RSVP 予約メッセージを送り返します。その際、送信側から受信側へのフロー用に予約する必要がある QoS が使用されます。大規模なマルチキャスト・グループの異機種混合の受信側の場合、この動作の QoS 要件は別々に考慮します。送信側は、予約を構成する際に、受信側になり得る全員の特性を知っている必要はありません。

RSVP を使用して予約を確立するには、受信側はシステム機能に応じた予約要求を送信側に送信します。たとえば、ワークステーションが高速で PC が低速の場合に、1 秒当たり 30 フレームの高品質 MPEG ビデオ・ストリームを受信したいとします。このデータ転送速度は 1.5 Mbps とします。ワークステーションにはビデオ・ストリームをデコードできる CPU パフォーマンスが十分ありますが、PC は 1 秒当たり 10 フレームしかデコードできません。ビデオ・サーバーから 2 つの受信側に 1.5 Mbps のビデオ・ストリームを備えられるというメッセージを送信した場合、ワークステーションからは 1.5 Mbps をいっぱい活用するという要求を戻せます。しかし、PC はすべてのフレームをデコードできないので、このフローでこの最大帯域幅は必要ありません。したがって PC は、1 秒当たり 10 フレームかつ 500 Kbps というフローの予約要求を送信できます。

22.2.2.1 RSVP の操作

リソース予約の基本的な部分は、パスです。パスとは、送信側から受信側へのさまざまなルーターを介したパケット・フローの方法のことです。特定のフローに属するすべてのパケットに同じパスが使用されます。パスにより、フローと同じ方向で送信されるメッセージが送信側で生成されるかどうかが決まります。個々の送信側のホストは、発信するデータ・フローごとに path メッセージを周期的に送信します。path メッセージには、特定のフローの QoS を記述したトラフィック情報が含まれます。RSVP 自体ではルーティングは処理されないで、個々のルーター中のルーティング・テーブル中にあるこの情報を使用して、RSVP メッセージが転送されます。

path メッセージが 1 つ目の RSVP ルーターに達すると、このルーターは、メッセージ中の *last hop* フィールドにある IP アドレスを格納します。この IP アドレスは、送信側のアドレスです。このルーターは、*last hop* フィールドに独自の IP アドレスを挿入し、path メッセージを次のルーターに送信します。このプロセスは、メッセージが受信側に達するまで繰り返されます。このプロセスが終了すると、個々のルーターは直前のルーターのアドレスを認識し、

パスが逆方向にアクセスできるようになっています。図 320 には、パス定義のプロセスが示されています。

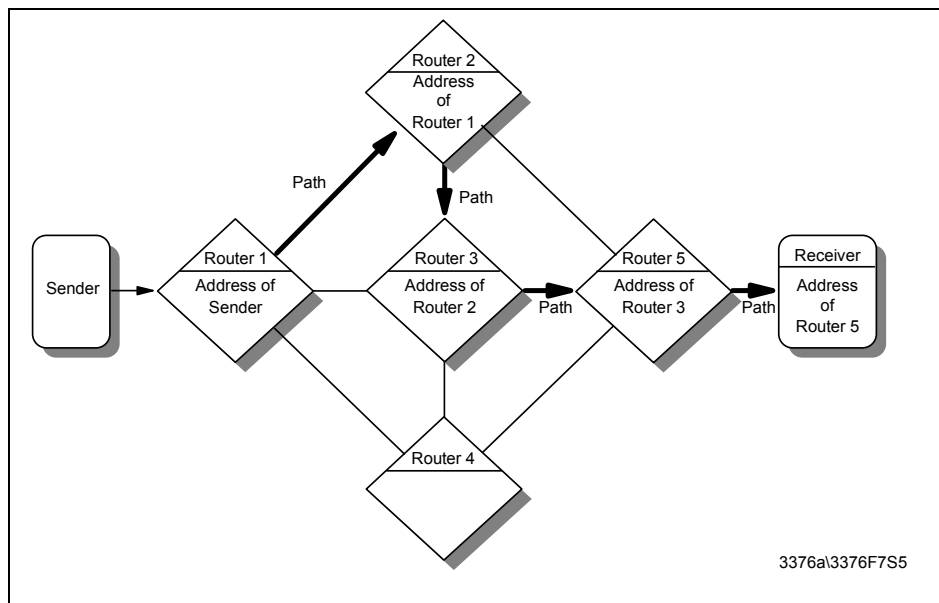


図 320 RSVP パス定義のプロセス

path メッセージを受信したルーターでは、フローのリソース予約を処理する準備が行われます。これらのルーター全体で、このフローに属するすべてのパケットが同じ方法 (path メッセージで定義されている方法) で処理されます。

path メッセージ送信後のシステムの状況は以下のとおりです。送信側がフローに特殊な QoS を設ける可能性があることが受信側で認識され、このフローの有効なリソース予約がすべてのルーターで認識されます。

この時点で、受信側でこのフローの QoS を予約したい場合は、予約 (resv) メッセージを送信します。予約メッセージにはこの受信側から要求する特定のフローの QoS が含まれています。このメッセージはフロー記述子を構成する filterspec と flowspec で表されます。path メッセージから受け取ったアドレスを使用して、受信側からパス中の最後のルーターに resv メッセージが送信されます。すべての RSVP 使用可能装置でパス上の直前の装置のアドレスが認識されているので、予約メッセージは送信側に対してパス上を逆方向に送信され、すべてのルーターでリソース予約が確立されます。図 321 には、ルーターを介した予約メッセージのフローが示されています。

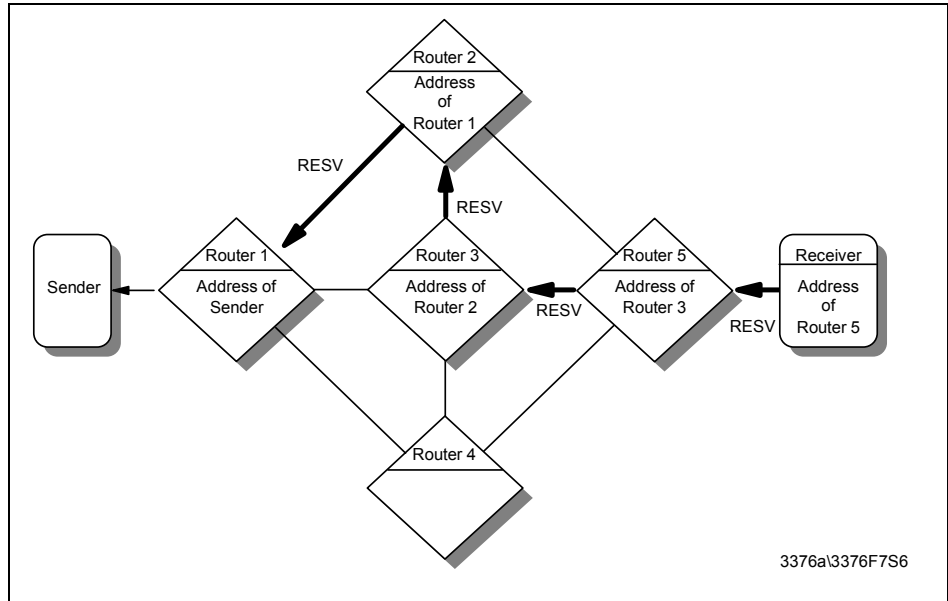


図 321 RSVP resv メッセージのフロー

個々のノードで、予約要求により 2 つのアクションが開始されます。

1. このリンク上の QoS 予約

RSVP プロセスにより、要求がノード上のアドミッション制御とポリシー制御に渡されます。アドミッション制御により、新しい QoS 予約を確立するために必要なリソースがルーターにあるかが検査され、ポリシー制御により、QoS 要求を行う権限がアプリケーションにあるかが検査されます。これらのテストの一方が失敗すると、予約は拒否され、RSVP プロセスにより *ResvErr* エラー・メッセージが該当する受信側に戻されます。両方の検査とも成功すると、ノードで resv メッセージ中の *filterspec* 情報を使用してパケット分類機能が設定され、*flowspec* 情報を使用してパケット・スケジューラーが設定されます。その後、パケット識別機能によりこのフローに属するパケットが認識され、パケット・スケジューラーによって *flowspec* で定義された当該 QoS が入手されます。

図 322 には、RSVP ルーター中の予約プロセスが示されています。

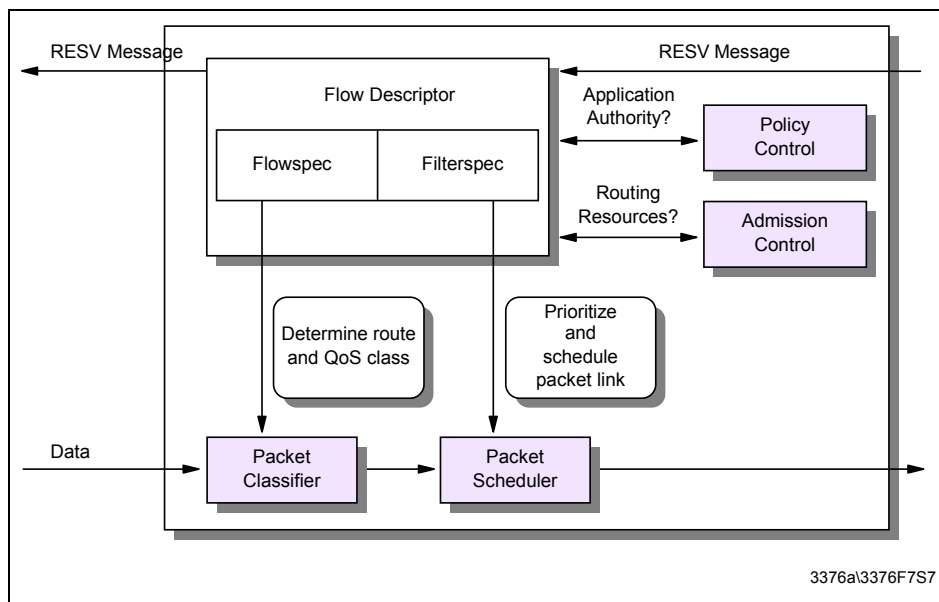


図 322 RSVP 予約プロセス

2. 予約要求の転送

アドミッションとポリシーの検査が成功したら、送信側に対して予約要求がアップストリームで伝搬されます。マルチキャスト環境では、受信側が複数の送信側からデータを受け取れます。当該予約要求の伝搬先の送信側ホストの集合のことを、この要求の**有効範囲**といいます。予約の成功後にノードによって転送される予約要求は、直前のホップ・ダウンストリームから受信した要求と違う場合があります。1つの考えられる理由としては、トラフィック制御機構によりホップ単位で flowspec に変更が加えられることがあることが挙げられます。さらに重要なもう1つの理由として、マルチキャスト環境では、同じ送信側の別々のダウンストリーム・ブランチから送信された予約は、アップストリーム・パスを送信されていくうちにマージされることが挙げられます。このマージ処理は、ルーター中でリソースを保存するために必要です。

正常な予約要求は、既存の予約がその要求されている予約以上になるポイントまで、マルチキャスト・ツリーに沿ってアップストリームに伝搬されます。このポイントで、着信した要求は、既存のそれ以上転送する必要がない予約とマージされます。

図 323 では、マルチキャスト・フローの場合の予約のマージについて説明しています。

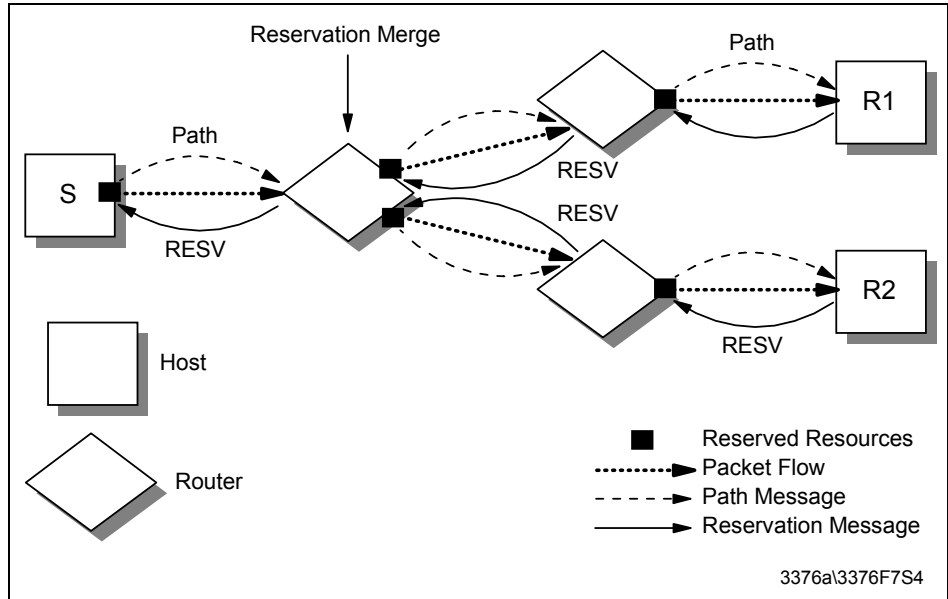


図 323 マルチキャストの場合の RSVP 予約のマージ

予約要求が送信側に達すると、パス上のすべてのルーターで QoS 予約が確立され、アプリケーションでダウンストリームでの受信側へのパケットの送信を開始できるようになります。個々のルーター中のパケット分類機能とパケット・スケジューラーにより、要求された QoS に従ってパケットが転送されていることが確認されます。

このタイプの予約は、パス上のすべてのルーターで RSVP がサポートされている場合だけ有効です。リソース予約をサポートしていないルーターが 1 つだけある場合は、通常のルーターの「ベストエフォート」特性のために、パス全体のサービスは保証できません。パス上の RSVP をサポートしていないルーターは、フローの障害になります。

予約要求を発信する受信側は、その要求がネットワーク中にインストールされたことを示す確認メッセージも要求します。受信側は、Resv メッセージに確認要求を組み込み、予約が正常に確立された場合に ResvConf メッセージを受信します。

RSVP リソース予約では、ルーターとホストでソフト状態が維持されます。この状態の場合、RSVP によりパスにのっとって既存の予約のリフレッシュ・メッセージが送信されないと、新しい予約が取り消されます。したがって、経路を変更してもプロトコル・オーバーヘッドが生じません。タイムアウト

期間後にルーター中のパス状態フィールドがリセットされるので、path メッセージもなければなりません。

RSVP 取り除きメッセージを使用して、パスと予約を削除することもできます。取り除きメッセージには以下の 2 種類あります。

- PathTear メッセージ

PathTear メッセージは、開始ポイントからすべての受信側にダウンストリームに送信され、個々の RSVP 使用可能装置のパス状態と従属するすべての予約状態を削除します。

- ResvTear メッセージ

ResvTear メッセージは、開始ポイントからすべての送信側にアップストリームに送信され、すべてのルーターとホストの予約状態を削除します。

送信側、受信側、または状態タイムアウトを通知するルーターから取り除き要求を開始できます。RSVP 予約のソフト状態の原理からすると、古い予約を明示的に取り除く必要は実際にはありません。しかしながら、既存の予約が必要なくなった時点で、エンド・ホストで取り除き要求を送信することをお勧めします。

22.2.2.2 RSVP 予約スタイル

マルチキャスト・マルチメディア・アプリケーションのユーザーは、さまざまな送信側から頻繁にフローを受信します。821 ページの 22.2.2.1 『RSVP の操作』で説明されている予約プロセスでは、受信側は、受信したいフローごとに別々の予約要求を開始しなければなりません。しかし RSVP には、さまざまな送信側からのフローの QoS を予約する方法として、さらに柔軟性の高い方法が備えられています。予約要求には、**予約スタイル**というオプションの集合が組み込まれています。これらの 1 つに、同一セッション中のさまざまな送信側の予約を処理するオプションがあります。受信側は、送信側ごとに**特殊予約**を確立したり、1 つのセッション中の送信側から送信されるすべてのパケットについて 1 つの**共用予約**を確立したりできます。

別のオプションに、予約要求の送信側を選択する方法を定義するものがあります。**明示リスト**を指定したり、**ワイルドカード**を指定して 1 つのセッションに属する送信側を選択したりできます。明示的に送信側を選択する予約では、filterspec で正確に 1 つの送信側を識別しなければなりません。ワイルドカードを使用して送信側を選択する場合は、filterspec は必要ありません。図 324 には、この予約オプションで定義される予約スタイルが示されています。

Sender Selection	Distinct Reservation	Shared Reservation
Explicit	Fixed-Filter (FF) Style	Shared-Explicit (SE) Style
Wildcard	(Not Defined)	Wildcard-Filter (WF) Style

図 324 RSVP 予約スタイル

ただし：

ワイルドカード・フィルター (WF)

ワイルドカード・フィルター・スタイルでは、共用予約オプションとワイルドカード送信側選択オプションが使用されます。この予約スタイルの場合、1つのセッション中のすべての送信側について1つの予約が確立されます。さまざまな送信側からの予約はパスにのってマージされるので、最大の予約要求だけが送信側に送達されます。

ワイルドカード予約は、すべての送信側ホストに対してアップストリームに転送されます。セッション中に新しい送信側が加わった場合（たとえば、新しいメンバーがビデオ会議に加わった場合）、予約はこれらの新しい送信側に拡張されます。

固定フィルター (FF)

固定フィルター・スタイルでは、特殊予約オプションと明示送信側選択オプションが使用されます。したがって、特定の送信側からのデータ・パケット用に、特殊な予約が作成されます。同一セッション中のさまざまな送信側からのパケットで予約が共用されません。

共用明示 (SE)

共用明示スタイルでは、共用予約オプションと明示送信側選択オプションが使用されます。つまり、1つの予約の対象は、指定した送信側のサブセットからのフローになります。したがって、受信側からの予約要求に送信側リストを組み込まなければなりません。

共用スタイル (WF と SE) で確立された予約は、ほとんどマルチキャスト・アプリケーションで使用されます。このタイプのアプリケーションの場合には、複数のデータ・ソースでデータを同時に送信する場合とは違って、送信側ごとに QoS を予約する必要はありません。

たとえば、5 人の参加者から成る音声会議で、すべての端末で 64 kbps でデータ・ストリームが送信されるとします。固定フィルター・スタイルの予約を使用すると、会議のすべてのメンバーが個別に他の送信側からのフロー用に 64 kbps の予約を確立しなければなりません。しかし、音声オーディオでは普通は同時に 1 人か 2 人しか話しません。したがって、すべての送信側について 128 kbps の帯域幅を 1 つ予約すれば十分です。なぜなら、大部分の音声会議ソフトウェアでは無音抑止が使用されているので、話していないときにはパケットは送信されないからです。つまり、個々の受信側がすべての送信側について 128 kbps の共用予約を 1 つ作成すればよいことになります。

共用明示スタイルを使用する場合は、すべての受信側は会議中の他の送信側すべてを明示的に識別しなければなりません。ワイルドカード・フィルター・スタイルを使用すると、予約は、その予約の仕様と合致するすべての送信側にとって有効になります。たとえば、音声会議ツールによってデータ・パケットが特殊な TCP/IP ポートに送信される場合に、受信側はこの宛先ポートを使用するすべてのパケットに関するワイルドカード・フィルター予約を作成できます。

22.2.2.3 RSVP メッセージの形式

基本的に、RSVP メッセージの構成は、先頭に共通ヘッダーがあり、その後に変数のオブジェクトから成る本文が続いています。これらのオブジェクトの数と内容は、メッセージのタイプに応じて変わります。メッセージ・オブジェクトには、リソース予約を認識するために必要な情報（フロー記述子や予約スタイルなど）が含まれます。ほとんどの場合、RSVP メッセージ中のオブジェクトの順序には論理的な違いはありません。RFC 2205 では、RFC 中で定義されているオブジェクト順序を RSVP の実装システムで使用するよう勧められていますが、どのような順序のオブジェクトでも受諾されます。図 325 には、RSVP メッセージの共通ヘッダーが示されています。

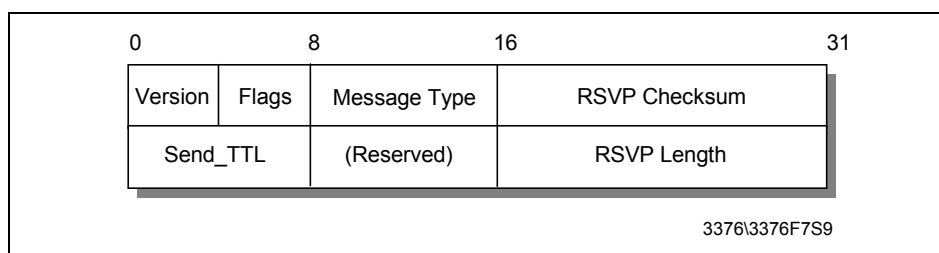


図 325 RSVP 共通ヘッダー

ただし：

バージョン 4 ビットの RSVP プロトコル番号。現行バージョンは 1 です。

- フラグ

フラグ用に予約されている 4 ビットのフィールド。現時点ではフラグは定義されていません。
- メッセージ・タイプ

メッセージ・タイプを指定する 8 ビットのフィールド。

 - Path
 - Resv
 - PathErr
 - ResvErr
 - PathTear
 - ResvTear
 - ResvConf
- RSVP 検査合計

16 ビットのフィールド。検査合計は、RSVP メッセージの受信側によって、このメッセージの伝送時のエラーを検出するために使用されます。
- Send_TTL

8 ビットのフィールド。送信されるメッセージの IP TTL 値が含まれます。
- RSVP 長

RSVP メッセージの全長 (共通ヘッダーとその後のすべてのオブジェクトを含む) を示す 16 ビットのフィールド。この長さはバイト単位でカウントされます。

共通ヘッダーの後の RSVP オブジェクトは、32 ビットのヘッダーと 1 つ以上の 32 ビットのワードで構成されます。図 326 には、RSVP オブジェクトのヘッダーが示されています。

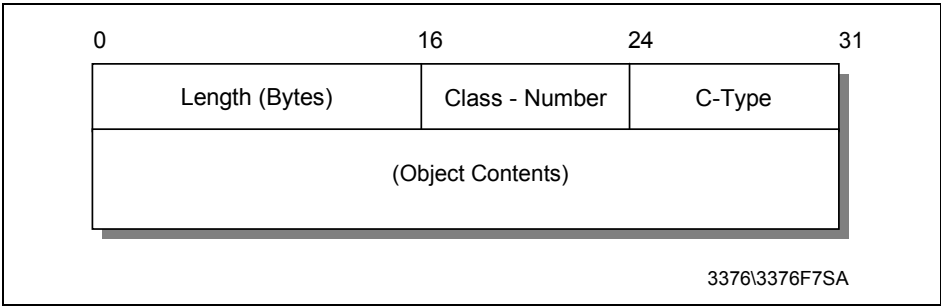


図 326 RSVP オブジェクト・ヘッダー

ただし：

長さ	バイト単位のオブジェクト長を含む 16 ビットのフィールド。この値は 4 の倍数でなければなりません。最小長は 4 バイトです。
クラス番号	オブジェクト・クラスを識別します。以下のクラスが定義されています。 - Null NULL オブジェクトのクラス番号はゼロです。このオブジェクトの長さは 4 以上ですが、任意の 4 の倍数でかまいません。NULL オブジェクトは、RSVP メッセージのオブジェクト・シーケンス中のどこにあってもかまいません。この内容は受信側では無視されます。 - Session session オブジェクトには、IP 宛先アドレス、IP プロトコル ID、および宛先ポートが含まれており、後続の他のオブジェクトの特定のセッションを定義します。session オブジェクトは、すべての RSVP メッセージで必須です。 - RSVP_HOP RSVP_HOP オブジェクトには、このメッセージを送信したノードの IP アドレスと、論理発信インターフェース・ハンドルが含まれています。RSVP_HOP オブジェクトは、ダウンストリーム・メッセージ (path メッセージなど) の場合は PHOP (直前のホップ) オブジェクトを表し、アップストリーム・メッセージ (resv メッセージなど) の場合は NHOP (ネクスト・ホップ) オブジェクトを表します。 - Time_Values Time_Values オブジェクトには、path メッセージと予約メッセージのリフレッシュ期間が含まれています。指定した時間枠内にこれらのメッセージが更新されないと、パス状態や予約状態は取り消されます。 - Style style オブジェクトは、予約のスタイルと、Flowspec や Filterspec にないスタイル固有の情報を定義します。style オブジェクトは、すべての resv メッセージで必須です。 - Flowspec

このオブジェクトは、予約メッセージ中に必要な QoS を指定します。

- Filterspec

Filterspec オブジェクトは、Flowspec で指定された QoS を受信するデータ・パケットを定義します。

- Sender_Template

このオブジェクトは、送信側の IP アドレスと追加のデマルチプレクシング情報が含まれています。この情報は、送信側の識別に使用されます。Sender_Template は、すべての path メッセージで必須です。

- Sender_Tspec

このオブジェクトは、送信側からのデータ・フローのトラフィック特性を定義します。Sender_Tspec は、すべての path メッセージで必須です。

- Adspec

adspec オブジェクトは、パスにのっとして RSVP ノード中のトラフィック制御モジュールに公示情報を備えるために使用されます。

- Error_Spec

このオブジェクトは、PathErr または ResvErr メッセージ中のエラーや ResvConf メッセージ中の確認を指定します。

- Policy_Data

このオブジェクトには、ポリシー・モジュールで、関連した予約の管理を許可するかどうか決定するための情報が含まれます。このオブジェクトは Path、Resv、PathErr、または ResvErr メッセージで使用できます。

- Integrity

integrity オブジェクトには、起点ノードを認証し RSVP メッセージの内容を検証するための暗号データが含まれています。

- Scope

Scope オブジェクトには、メッセージ中の情報を送信する送信側ホストの明示リストが含まれます。このオブジェクトは、Resv、ResvErr、または ResvTear メッセージで使用できます。

- Resv_Confirm

このオブジェクトには、予約の確認を要求する受信側の IP アドレスが含まれます。このオブジェクトは、Resv または ResvConf メッセージで使用できます。

C タイプ

C タイプは、クラス番号中のオブジェクト・タイプを指定します。IPv4 および IPv6 でさまざまなオブジェクト・タイプが使用されています。

オブジェクトの内容

オブジェクトの内容は、オブジェクト・タイプによって変わり、最大長は 65528 バイトです。

すべての RSVP メッセージは可変数のオブジェクトで構成されます。最も重要な RSVP メッセージである path メッセージと resv メッセージの、推奨されているオブジェクト順序が、図 327 に示されています。この図は、RSVP path メッセージの形式に関する概要を示しています。path メッセージ中で使用できるが必須でないオブジェクトは、括弧で囲まれています。

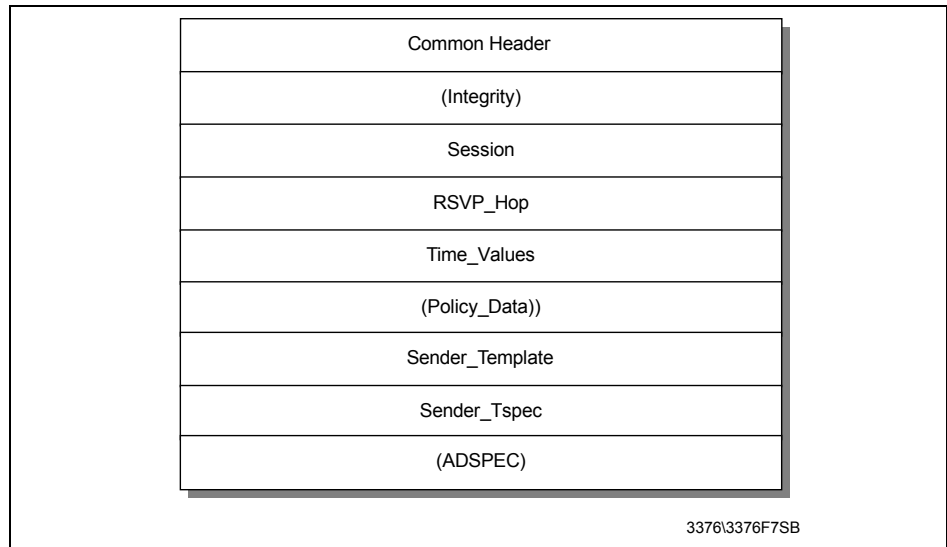


図 327 RSVP path メッセージの形式

Integrity を path メッセージ中で使用する場合は、共通ヘッダーの直後にしなければなりません。他のオブジェクトの順序は RSVP の実装に応じて変わる場合もありますが、図 327 に示されている順序は RFC で推奨されているものです。

RSVP Resv メッセージは、path メッセージに似ています。図 328 には、予約メッセージで使用されるオブジェクトが示されています。

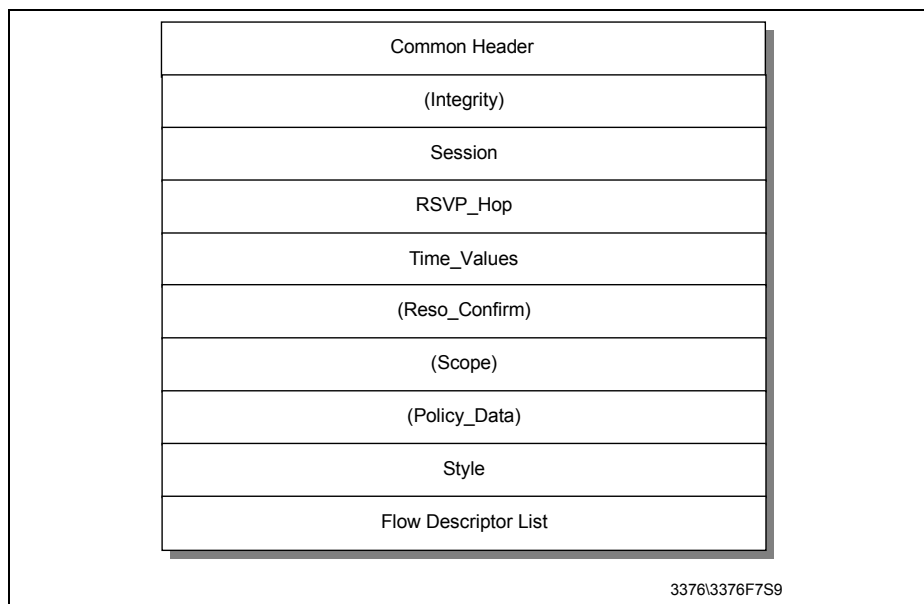


図 328 RSVP resv メッセージの形式

path メッセージ中と同様に、Integrity オブジェクトを使用する場合は共通ヘッダーの後でなければなりません。Style オブジェクトとその後のフロー記述子リストには、別の制限が適用されます。これらはメッセージの末尾でなければなりません。他のオブジェクトの順序は RFC の推奨に従っています。

RSVP メッセージの構造と、予約メッセージ中のさまざまな予約スタイルの取り扱いに関する詳細な説明は、RFC 2205 を参照してください。

22.2.3 統合サービスの見通し

統合サービスは、異機種ネットワークを介して終端間のサービス品質 (QoS) をアプリケーションに備える目的で設計されています。したがって、統合サービスは、さまざまなネットワーク・タイプやネットワーク装置でサポートされる必要があります。さらに、ルーターなどのネットワーク中のエレメントで、終端間 QoS フローの要求サービスを備えるための情報が必要になります。ルーター中でこの情報をセットアップするには、Resource Reservation Protocol (RSVP) が使用されます。RSVP は、シグナル・プロトコルの一種で、統合サービス情報を転送できます。

RSVP はネットワーク中のリソースを要求するために使用しますが、統合サービスは、リソース要件を定量化し、要求リソースの可用性を判断して、必要なサービス・タイプを定義します。

RSVP の配置を (つまり、インターネット中の統合サービスの配置を) 困難にしてきた要因がいくつかあります。以下の要因があります。

- 現在 RSVP シグナルを生成するホストがごく少数である。近々数が増えることが予想されるものの、多くアプリケーションは RSVP シグナルを生成できません。
- 統合サービスはフロー状態とフロー処理に基づいている。大規模ネットワークの場合、フロー処理が劇的に向上しないと、拡張容易性の問題が生じる可能性があります。
- アクセス制御認証やアカウントティングなどの、必須のポリシー制御機構は、最近使用できるようになったばかりである。

サービス提供者が統合サービスと RSVP を使用するよう動かされるかどうかは、市場の需要にかかっています。しかし、需要を増やすためには、ネットワーク装置 (ルーターなど) に必須のソフトウェアをサポートする必要があります。

別の局面も考慮する必要があります。DiffServ ネットワークを介して統合サービスのサポートを実行することも考慮できます。このソリューションには、以下のものが備えられています。

- アプリケーション用の終端間 QoS (IP テレフォニーなど)、およびビデオ・オンデマンド。
- Intserv を使用して、終端間パスに従って、ホストにフロー単位の定量化可能なリソースを要求できる (リソースに対するアドミッションに関するフィードバックを含む)。
- Diffserv によって、フロー状態とフロー処理の必要をなくすことにより、大規模ネットワーク間での拡張容易性を可能にする。

このソリューションは、845 ページの 22.3.2 『DiffServ ネットワーク上での統合サービス (Intserv)』で説明されています。

22.3 DiffServ

DiffServ (DS) の概念は、現在 IETF DS 作業チームが開発しています。DS の仕様は、特定の IETF インターネット・ドラフトで定義されており、現時点では RFC は使用できません。以下に、インターネット中にサービスの差異化を備えることに関する基本原理と着想の概要を示します。この概念は依然として開発中なので、本書で言及されている仕様は DiffServ の最終定義では変更されている可能性があります。

DS 開発の目標は、インターネット・トラフィック用に差異化されたサービス・クラスを備えて、さまざまなタイプのアプリケーションをサポートし、特定の業務要件を満たすようにすることです。DS を使用すると、特定の時間の特定の負荷に関するパフォーマンス（遅延、スループット、パケット・ロスなど）を予測できます。812 ページの 22.2 『統合サービス』で説明されている統合サービスと DiffServ の間の違いは、DS の場合は、フロー単位状態であったりすべてのホップでシグナルを送出したりしなくても、インターネット中で拡張が容易なサービスの差異化を備えることです。フローごとに固有の QoS 予約を実行する必要はありません。DS を使用すると、インターネット・トラフィックは、さまざまな QoS 要件のさまざまなクラスに分割されます。

DS の中心的なコンポーネントは、サービス・レベル・アグリーメント (SLA) です。SLA とは、顧客とサービス提供者との間のサービス契約の一種で、トラフィックの分類と、それに対応して顧客が受信する必要がある転送サービスの詳細を指定します。顧客はユーザーの組織でも別の DS ドメインでもかまいません。サービス提供者は、SLA を結んでいる顧客のトラフィックが、契約した QoS を満たすことを確約しなければなりません。したがって、サービス提供者は、ネットワーク管理の際に、該当するサービス・ポリシーをセットアップし、ネットワークのパフォーマンスを測定して、合意したトラフィック・パフォーマンスを保証しなければなりません。

DS を使用できるネットワーク装置で、さまざまな顧客からのデータ・パケットを区別するには、IP パケットの特定のフィールドに変更を加えます。個々の IP パケット中にある、DS フィールドという小さなビット・パターンを使用して、個々のネットワーク・ノードで特定の転送処理を受信するパケットにマークを付けます。DS フィールドは、IPv4 IP ヘッダーの場合は旧 TOS オクテットのスペースを使用し (93 ページの 3.1.8.1 『IP データグラム・フォーマット』を参照)、IPv6 ヘッダーの場合はトラフィック・クラス・オクテットを使用します。ドメイン中のすべてのネットワーク・トラフィックは、DS フィールドで指定したトラフィック・クラスに応じたサービスを受信します。

SLA に準拠したサービスを備えるには、ネットワーク中で以下の機構を結合しなければなりません。

- ネットワークの端と管理境界で、DS フィールド (TOS オクテット) 内にビットを設定する。
- これらのビットを使用して、ネットワーク中のルーターでパケットが処理される方法を決める。
- 個々のサービスの QoS 要件に従って、ネットワーク境界でマークされたパケットの条件付けをする。

現在定義されている DS アーキテクチャーでは、1 方向で非対称のサービス差異化しか備えられていません。相補型の対称アーキテクチャーの開発は、現在研究中の問題です。DS アーキテクチャーの詳細について以下に説明します。

22.3.1 DiffServ アーキテクチャー

統合サービスとは違って、DiffServ で行われる QoS 保証は静的で、ルーター中に長期間とどまります。したがって、DS を使用するアプリケーションで、特定のデータ・パケットの QoS 予約をセットアップする必要はありません。DS 使用可能ネットワークを通過するすべてのトラフィックで、特定の QoS を受信できます。DS フィールドを使用してデータ・パケットにマークを付けなければならない、これがネットワーク中のルーターによって解釈されます。

22.3.1.1 ホップごとの転送優先順位付け (PHB)

DS フィールドでは、6 ビットを使用して DiffServ コード・ポイント (DSCP) が判別されます。ネット中の個々のノードで、このコード・ポイントを使用して、PHB が選択されます。2 ビットの現在使用されていない (CU) フィールドは予約済みです。受信パケットに PHP を使用する場合は、DiffServ 準拠のノードで CU ビットの値は無視されます。図 329 には、定義済みの DS フィールドの構造が示されています。

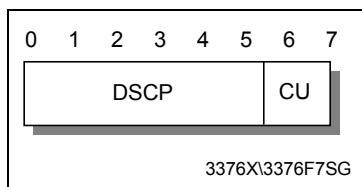


図 329 DS フィールド

個々の DS 使用可能ネットワークに、DS フィールドのあるパケットを処理する方法に関する情報がなければなりません。DS 仕様では、この情報のことを **ホップごとの転送優先順位付け (PHB)** といいます。PHB とは、特定のネットワーク・ノードで受信されるパケットの転送処理に関する説明のことです。DS フィールド内の DSCP 値は、個々のノードでのパケットの処理に関する PHB を選択するために使用されます。予測可能なサービスを備えるには、DiffServ ネットワーク中のすべてのルーターでホップごとの転送優先順位付けが使用できる必要があります。ルーター中でパラメーターの集合として PHB を記述し、出力インターフェース上でパケットをスケジュールする方法を制御するために使用できます。PHB は、優先順位を設定できる複数の独立したキュー、キュー長に関するパラメーター、またはパケットの除去アルゴリズムと除去優先度重みのいずれかです。

DS には、キューのスケジューリングと管理をサポートして、アウトバウンド・パケットの優先順位付けを行い、ネットワーク上で輻輳を最小化してキュー項目数を制御するルーターが必要です。共通インターネット・ルーター中の従来の FIFO キューイングには、サービスの差異化は備えられていないので、パフォーマンス上の問題が生じることがあります。ルーター中でのパケットの処理は、ルーターの機能と特定の構成に応じて異なり、IP パケットの DS フィールドによって選択されます。たとえば、優先順位の異なる 8 種類のキューのあるルーターに IP パケットが送達される場合に、DS フィールドを使用して、このパケットのルーティングを行うキューを選択できます。目盛りは、最も低い優先順位のゼロから、最も高い優先順位の 7 までです。

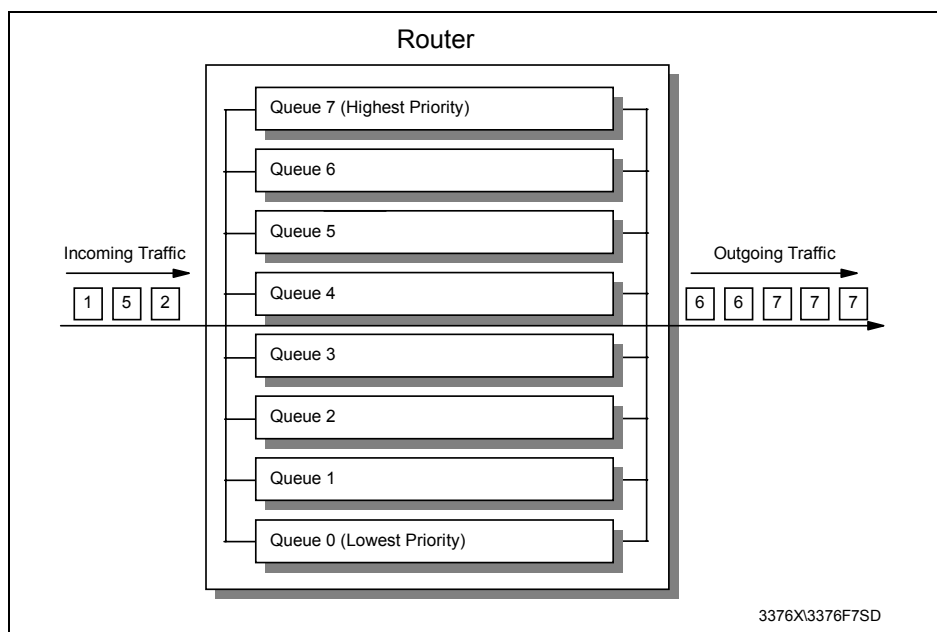


図 330 DS によるルーティングの例

別の例として、データ・パケットの優先順位が複数ある 1 つのキューを持つルーターが考えられます。このルーターは、DS フィールドを使用して、キュー中のパケットの除去優先度を選択します。値ゼロは「このパケットが除去される可能性が一番高い」ことを意味し、7 は「このパケットが除去される可能性が一番低い」ことを意味します。別の例として、4 つのキューのそれぞれに 2 つのレベルの除去優先度がある場合も考えられます。

個々のルーター中のホップごとの転送優先順位付けの機能が同等であることを確認するには、将来の DS 仕様で特定の共通 PHB を定義して、DS フィールド値が同じなのに 1 つの DS ドメインのさまざまなルーターで別々の転送優先順位付けが行われることのないようにしなければなりません。つまり、

将来の DS 仕様で、特定のサービス・クラスを表す固有の PHB 値を定義しなければなりません。1 つの DS ドメイン中のすべてのルーターで、特定の PHB を指定したパケットで受信されるサービスが認識されていなければなりません。DiffServ 作業チームは、DiffServ を備えるために使用する PHB を提供しています。これらの提供されている PHB には標準化されているものや、広範囲に使用されているものや、依然として実験段階のものがあります。

PHB はグループとして定義されます。PHB グループは、1 つ以上の PHB の集合で、指定や実装は同時にしか行えません。すべての PHB に適用されるキューの保守や管理のポリシーが 1 つのグループになっているからです。デフォルトの PHB が、すべての DS 準拠のノードで使用できるようになっていなければなりません。デフォルトの PHB は、既存のルーターで利用できるベストエフォートの転送動作を表します。他に合意がない場合は、パケットはこのサービス・レベルに属していると想定されます。IETF 作業チームは、DS フィールド内で DSCP 値として 000000 を使用して、デフォルトの PHB を定義することを推奨しています。

標準化されて提供されている別の PHB に、優先転送(EF) PHB があります。これは優先順位の高い動作で、代表的にはルーティングの更新などのネットワーク制御トラフィックで使用されます。EF PHB の場合は、DS フィールドの DSCP フィールド内に値 101100 を指定することが推奨されています。

22.3.1.2 DSCP の編成

DSCP に関する IANA の考慮事項がいくつかあります。DSCP のコード・ポイント・スペースは、64 のコード・ポイント値を区別します。このスペースを 3 つのプールに分割することが提案されています。プール 1 は標準のアクションに使用できます。他のプールはローカルで実験的に使用できます。近々、2 つのプールの 1 つがローカルで実験的に使用するために備えられます(838 ページの表 27 を参照)。

このスペースを以下の 3 つのプールに分割することが提案されています。

表 27DSCP プール

プール	コード・ポイント・スペース	割り当て
1	xxxxx0	標準のアクション
2	xxxx11	実験的 / ローカル使用
3	xxxx01	将来の実験的 / ローカル使用

22.3.1.3 DiffServ のドメイン

QoS 保証は、特定の終端間接続ではなく、良質な定義の DiffServ ドメイン用にセットアップされます。IETF 作業チームは、DiffServ ドメインのことを、

インターネットに隣接した部分であり、これを介して DiffServ ポリシーの一貫した集合を整合性のある方法で管理されるものと定義しています。このドメインは、さまざまな管理可能ドメインや自律システム、さまざまな承認領域、さまざまなネットワーク・テクノロジー（セルやフレーム・ベースの手法など）、ホスト、およびルーターを表すことができます。DS ドメインは、さまざまな DS ドメインの相互接続に使用する境界コンポーネントと、ドメイン中だけで使用する内部コンポーネントで構成されます。図 331 には、2 つの DS ドメインでの境界コンポーネントと内部コンポーネントの使用法が示されています。

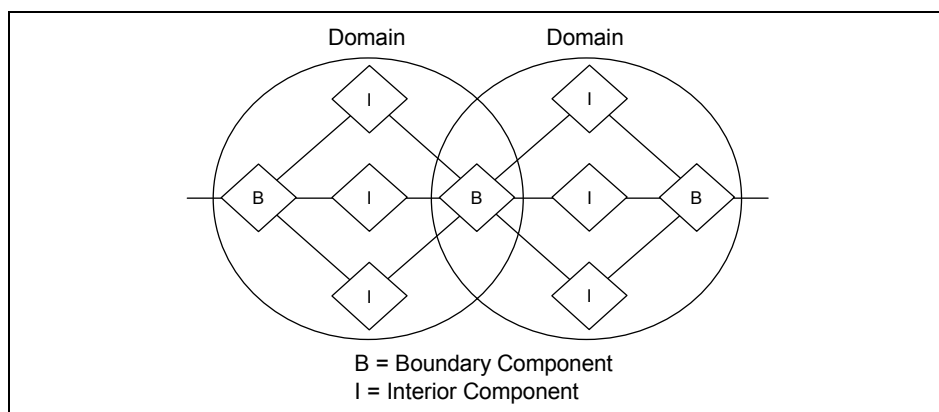


図 331 DS ドメイン

通常 DS ドメインは、同一の管理下にある 1 以上のネットワークで構成されます。たとえば、社内のイントラネットや、インターネット・サービス・プロバイダー (ISP) にすることができます。ドメインを管理することにより、ドメインによって提供されている SLA をサポートするために適切なリソースが用意されて予約されているかを確認することになります。ネットワーク管理者は、該当する測定の手法を使用して、DS ドメイン中のネットワーク・リソースがすべての許可された QoS 要求を十分満たしているかどうかモニターしなければなりません。

22.3.1.4 DS 境界ノード

DS ドメイン間を転送されるすべてのデータ・パケットは、境界ノードを通過しなければなりません。境界ノードはルーター、ホスト、ファイアウォールのいずれでもかまいません。DS ドメインを出るトラフィックを処理する DS 境界ノードのことを**出口ノード**といい、DS ドメインに入るトラフィックを処理する DS 境界ノードのことを**入り口ノード**といいます。通常、DS 境界ノードは、トラフィックの方向に応じて、入り口ノードと出口ノードの両方の働きをします。入り口ノードでは、ドメインに入るパケットが、それ以前に通過したドメインと同じ QoS を受信しているか確認しなければなりません。

ん。DS 出力ノードでは、直接接続している同位のドメインに転送されるトラフィックに対して条件付け機能が実行されます。トラフィックの条件付けは、**トラフィック条件付け機能**によって境界ノードの内部で実行されます。この機能は、DS ドメインを出入りするパケットを分類したり、マークを付けたり、条件付けしたりします。トラフィック条件付け機能は以下のコンポーネントで構成されています。

分類機能

分類機能は、パケット・ヘッダーに基づいてパケットを選択し、分類規則に合致したパケットをその後の処理のために転送します。DS モデルでは、以下の 2 種類のタイプ分類機能が指定されています。

- 複数フィールド (MF) 分類機能。これは、DS フィールドや、RSVP 分類機能と同様に他の IP ヘッダー・フィールド (IP アドレスやポート番号など) に基づいて分類できます。
- 動作集約 (BA) 分類機能。これは、DS フィールド内のビットだけに基づいて分類します。

計量

トラフィック計量は、分類機能で選択されたパケットの転送が、顧客とサービス提供者の間の SLA に関する Qos を記述したトラフィック・プロファイルに対応しているかどうか測定します。計量により、状態情報が他の条件付け機能に渡され、要求されている QoS の要件に従っていてもいなくても、パケットごとに特定のアクションが起動されます。

マーカ

DS マーカは、着呼 IP パケットの DS フィールドを、特定のビット・パターンに設定します。PHB は DS フィールドの先頭 6 ビットで設定されているので、マーク付けられたパケットは、サービス提供者と顧客の間の SLA に従って、DS ドメインの内部に転送されます。

シェーパー / ドロッパー

パケット・シェーパーとパケット・ドロッパーを使用すると、構成済みのトラフィックのプロパティの一部 (たとえば、816 ページの 22.2.1 『サービス・クラス』で説明されているトークン・バケット・フィルタ) を規格合致させることができます。これらのコンポーネントは、さまざまな方式を使用して、ストリームをトラフィック・プロファイルに準拠させます。シェーパーを使用すると、パケットの一部またはすべてが遅れます。普通シェーパーには有限サイズのバッファがあるので、遅れたパケットを保持できるだけのバッファ・スペースがない場合は、パケットが廃棄されることがあります。ドロッパーを使用すると、パケットの一部またはすべてが廃棄されます。この

プロセスのことを、ストリームのポリシングといいます。シェーパのバッファ・サイズをゼロ個の packets に設定することにより、特殊なシェーパとしてドロップを実装できます。

トラフィック条件付け機能は、主に DS 境界コンポーネントで使用されますが、内部コンポーネントにも実装できます。図 332 には、トラフィック条件付け機能コンポーネントの組み合わせが示されています。

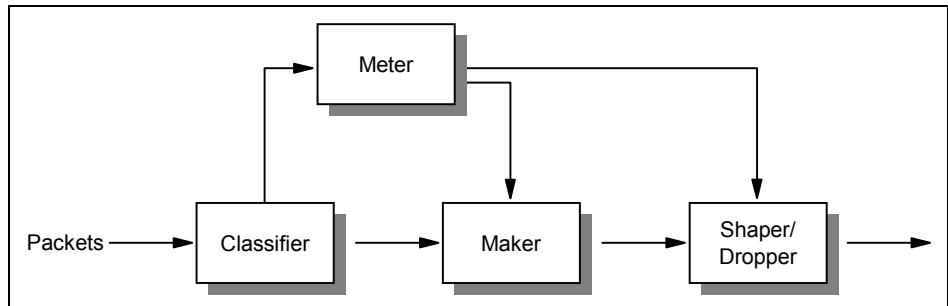


図 332 DS トラフィック条件付け機能

境界コンポーネント中のトラフィック条件付け機能は、ドメインを伝送される packets に、そのドメインでサポートされている PHB グループの 1 つから PHB を選択するためのマークが正しく付けられていることを確認します。この処理は必須です。なぜなら、DS ドメインによって PHB のグループが異なる場合があるため、DS フィールド中の同一のエントリーの解釈がドメインによって異なる可能性があるからです。

たとえば、packets が伝送される 1 つ目のドメインには、すべてのルーターに 4 つのキューがあり、それぞれのキュー優先順位は異なります (0 ~ 3)。PHB 値が 3 の packets が最も高い優先順位でルーティングされます。しかし、packets が伝送される 2 つ目のドメインには、すべてのルーターに 8 種類のキューがあり、PHB 値が 7 の packets がすべて最も高い優先順位でルーティングされます。最も高い優先順位で 1 つ目のドメインに転送される packets は、2 つ目のドメインでは中間の優先順位になるに過ぎません。このことは、顧客とサービス提供者の間の SLA 契約に違反します。したがって、2 つのドメインに接続している境界ルーター中のトラフィック条件付け機能は、packets が 1 つ目のドメインから 2 つ目のドメインに伝送される場合に、PHB が 3 から 7 に再マーク付けされることを確認します。図 333 には、2 種類のドメインを伝送されるデータ・packets の再マーク付けの例が示されています。

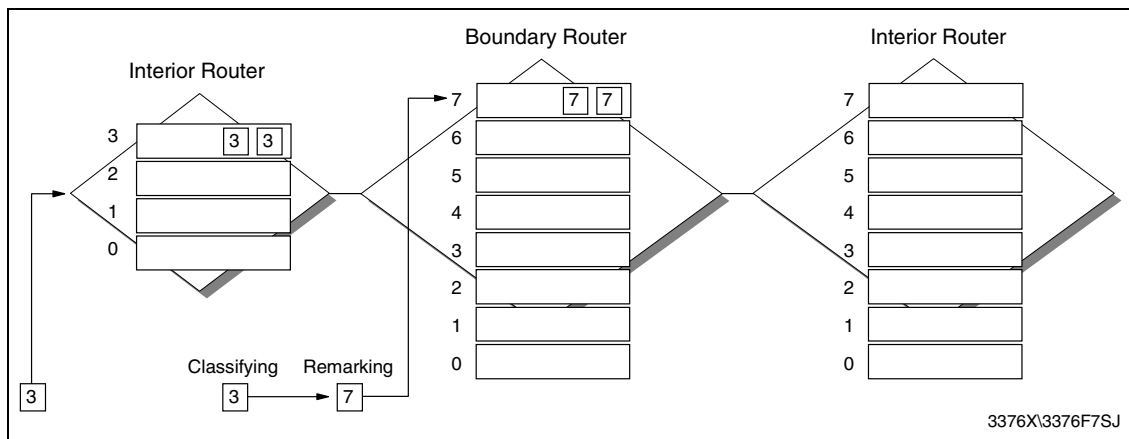


図 333 データ・パケットの再マーク付け

データ・パケットが複数のドメインを伝送される場合、すべての境界コンポーネントで DS フィールドを再マーク付けすることにより、SLA で契約した QoS を保証できます。SLA には、トラフィック条件付け合意 (TCA) の詳細が含まれています。TCA は、トラフィック・ストリームの分類規則と一時プロパティを指定しています。TCA には、SLA を満たすために、トラフィック条件付け機能で行われなければならないパケットの計量、マーク付け、廃棄、およびシェーピングの方法に関する情報が含まれています。さまざまな DS ドメインを通過するパケットが個々のドメインで同一のサービスを受けることを保証するには、TCA 情報が、DS ネットワークのすべての境界コンポーネントで使用できるようになっていなければなりません。

22.3.1.5 DS 内部コンポーネント

DS ドメイン中の内部コンポーネントは、DS フィールドに基づいて、パケットの転送動作を選択します。普通は、内部コンポーネントは、トラフィック優先順位付けアルゴリズムを含むルーターです。通常 DS ドメインの内部で DS フィールドの値は変更されないで、QoS の合意に従うには、すべての内部ルーターで同一のトラフィック転送ポリシーが使用されていなければなりません。DS フィールド中の PHB 値が違うデータ・パケットは、PHB の QoS 定義に従って、別個の QoS を受け取ります。1 つのドメイン中のすべてのルーターで着信トラフィックに関するポリシー機能は同じなので、内部ノード中のトラフィック条件付けは、パケット分類機能だけで行われます。パケット分類機能により、パケットの PHB 値や他の IP ヘッダー・フィールドに基づいてパケットが選択され、ノードのキュー管理とスケジューリングのインスタンスにパケットが転送されます。図 334 には、内部ノード中のトラフィック条件付けが示されています。

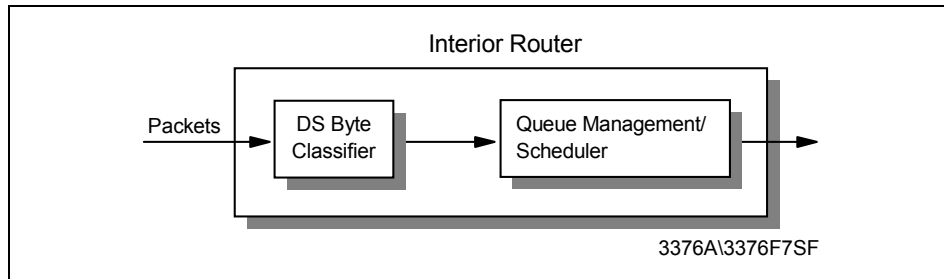


図 334 DS 内部コンポーネント

トラフィックの分類と優先順位付けのルーティングは、DS ドメインのすべての内部コンポーネントで行われます。データ・パケットがドメインを通過すると、次のドメインの境界ルーターに送達され、要求されている QoS でこのドメインを通過するよう再マーク付けする必要があるかもしれません。

22.3.1.6 ソース・ドメイン

IETF DS 作業チームは、ソース・ドメインのことを、特定のサービスを受け取るトラフィックを発信する 1 つ以上のノードを含むドメインと定義しています。ソース・ドメイン中のトラフィック・ソースと中間ノードで、トラフィックの分類と条件付けの機能を実行できます。ソース・ドメインから送信されるトラフィックがソース・ドメインから出る前に、トラフィック・ソースで直接マークを付けたり、中間ノードでマークを付けたりできます。

これに関連して、送信側のアプリケーション自体で、データ・パケットの最初の PHB マーク付けが行われないということは重要です。アプリケーションでは、ネットワーク中の DiffServ の可用性は認識されません。したがって、DS ネットワークを使用するアプリケーションを、DS をサポートするように再作成することはできません。このことと統合サービスとの相違点は重要です。統合サービスでは、コードを一部変更する必要がある場合には、大部分のアプリケーションで直接 RSVP プロトコルがサポートされます。

アプリケーションから送信されるパケットの最初の PHB マーク付けは、ソース・ホストか、パケットが通過する 1 つ目のルーターで行えます。パケットは IP アドレスとソース・ポートで識別されます。たとえば、顧客とサービス提供者との SLA で、音声アプリケーションによって送信されるパケットに高い優先順位が保証されているとします。音声アプリケーションでは特定のポートを介してデータ・パケットが送信され、複数フィールドの分類機能でこのアプリケーションを認識できます。この分類タイプにより、パケットの IP アドレスとポート番号が認識され、別のアプリケーションから送信されるパケットと区別できます。ホストにトラフィック条件付け機能と MF 分類機能が含まれている場合は、IP パケットに該当する PHB 値のマークを付けることにより、顧客が要求する QoS を受け取ることができます。ホストにトラ

フィック条件付け機能が含まれていない場合は、パケットの初期マーク付けは、トラフィック条件付け機能をサポートしているソース・ドメイン中の 1 つ目のルーターで行われます。図 335 には、ホストおよびルーター中でのパケットの初期マーク付けが示されています。

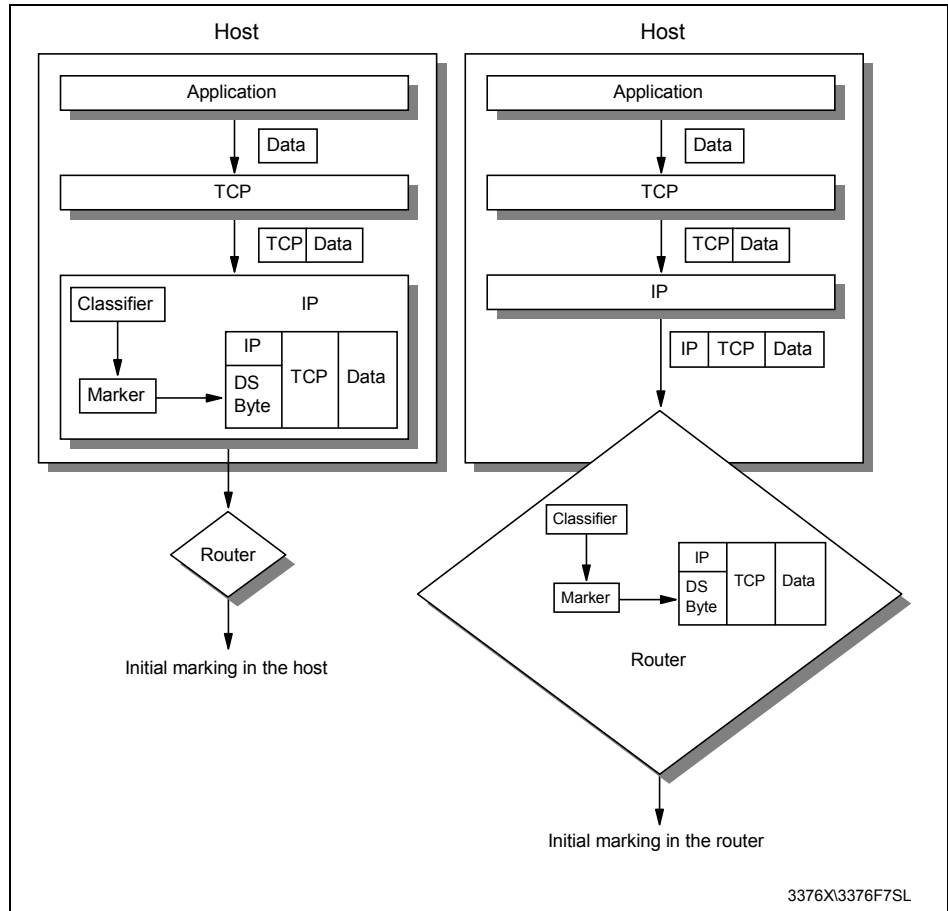


図 335 データ・パケットの初期マーク付け

本書の例では、DS ネットワークには、音声アプリケーションからのパケットの優先順位は他のパケットより高い必要があるというポリシーがあります。送信側ホストで、すべての発信パケットの DS フィールドに、優先順位が高いことを示す DS コード・ポイントでマークを付けることができます。または、その代わりに、送信側のホストに直接接続している 1 つ目のホップ・ルーターで、トラフィックを分類し、正しい DS コード・ポイントでパケットにマークを付けることもできます。ソース DS ドメインでは、この提供者の DS ドメイン宛ての集約トラフィックが、顧客とサービス提供者の間の

SLA に準拠していることを確認することになります。ソース・ドメインの境界ノードでも、提供されるサービスが、要求されているサービスに準拠しているかモニターし、必要に応じてパケットのポリシング、シェーピング、または再マーク付けを行います。

22.3.2 DiffServ ネットワーク上での統合サービス (Intserv)

この基本的な着想は、両方のアーキテクチャーを使用して、終端間の定量化 QoS を備え、これによって拡張容易性も可能にするというものです。このことを実現するには、Intserv モデル終端間を、1 つ以上の Diffserv 領域を含むネットワーク全体に適用します。

Intserv では、Diffserv 領域が、Intserv を実行している Intserv 使用可能ルーターまたはホストに接続している仮想リンクとして表示されます。Diffserv 領域中では、ルーターは特定の PHB 定義によって実装され、集約トラフィック制御が備えられます。DiffServ ネットワークの端でポリシーを決定して、DiffServ 領域に入るトラフィックの総量を限定できます。Intserv のトラフィックを、DiffServ 領域の制限に適合させる必要があります。

Intserv ネットワークと Diffserv ネットワークを接続する方法には、以下の 2 つがあります。

1. Diffserv のネットワークまたは領域中のリソースに、RSVP 認識装置を組み込む。この装置は RSVP シグナルに関係します。
2. DiffServ 領域中のリソースに、RSVP シグナルを組み込まない。

本書では最新の方法を使用することを想定しているので、本書の例 (図 336) では 2 つ目の構成について説明しています。

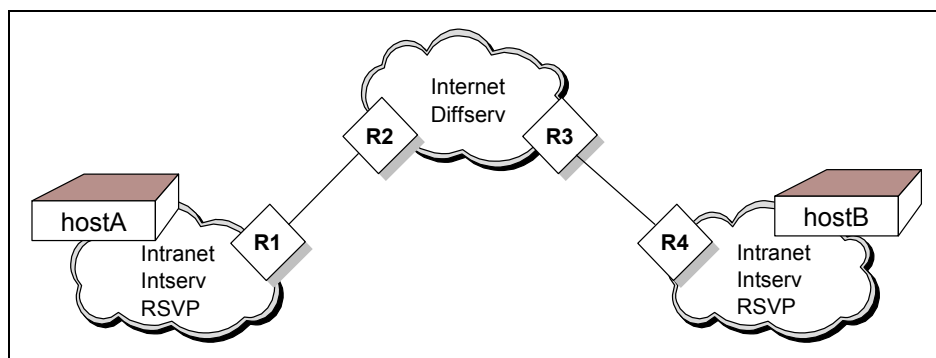


図 336 DiffServ を使用した統合サービスと RSVP

この構成では、2 つの RSVP 使用可能イントラネットが DiffServ 領域に接続しています。一方のイントラネットのホストでは RSVP を使用して定量化

QoS 要件を通信しており、他方のイントラネット中のパートナー・ホストでは QoS 認識アプリケーションが実行されています。

これらのイントラネットにはエッジ・ルーター R1 と R4 が含まれており、これらのルーターは DiffServ 領域インターフェースに隣接しています。これらのルーターは、DiffServ 領域にあるボーダー・ルーター R2 と R3 に接続しています。

ホスト (ホスト A など) 中のサービス要求アプリケーションによって、RSVP シグナル・プロセスが開始されます。ホスト中のトラフィック制御により、伝送されるパケット中に DSCP のマークが付けられ、伝送されるトラフィックが使用中の Intserv サービスの要件に合わせてシェーピングされます。

イントラネット中のホストの間で、RSVP 終端間シグナル・メッセージが交換されます。したがって、RSVP/Intserv リソース予約は、DiffServ 領域の外側で行われます。

エッジ・ルーターは、DiffServ ネットワークのアドミSSION制御エージェントの働きをします。これらのルーターによって、両方のイントラネット中でホストからのシグナル・メッセージが処理され、アドミSSION制御が適用されます。アドミSSION制御は、DiffServ 領域中で使用できるリソースに基づいています。また、会社のイントラネットのポリシー中で定義することもできます。

サンプル中のボーダー・ルーター (R2 および R3) では RSVP が認識されないため、これらのルーターは純粋に DiffServ ルーターの働きをします。これらのルーターにより、指定された DSCP とホストの集約トラフィック制御の合意に基づいて、パケットの制御や実行依頼が行われます。

Diffserv ネットワークまたは領域では、集約トラフィック制御がサポートされており、MF 分類を使用できないと想定されています。したがって、RSVP メッセージは、DiffServ ネットワーク領域を透過的に通過し、パフォーマンス上の影響は微々たるものになります。

次に考慮する必要がある点として、Intserv のサービス・タイプと定量化パラメーターの集合 (flowspec) のマッピングがあります。DiffServ ネットワークでは PHB か PHB の集合が使用されるので、これに従って Intserv flowspec のマッピングを定義する必要があります。マッピング値は、DSCP 中のビットの組み合わせになっています。ただし、DiffServ ネットワークの帯域幅管理に関する考慮事項の観点からこのマッピングを考察する必要があります。

DSCP 値は DiffServ ネットワーク中のすべてのルーターに認識されるようになっています。そこで、DSCP がこれらのルーターに伝搬される方法に関する疑問が生じます。

以下の 2 つの選択項目があります。

1. DiffServ 領域の入り口 (境界ルーター) で DSCP のマークを付けることができる。この場合、DiffServ 領域の出口 (他の境界ルーター) で再マーク付けすることもできます。
2. イントラネットのホストまたはルーターで、DSCP のマーク付けを行える。この場合、該当するマッピングを行うには、マーク付けを行う装置と通信する必要があります。この処理は RSVP に備えられています。

以下の手順は、アプリケーションに終端間 QoS サポートを備える方法を示しています。

1. イントラネットに接続しているホスト A から、別のイントラネットに接続しているホスト B に要求が出されます。両方のイントラネットとも、DiffServ ネットワークを介して接続しています (845 ページの図 336 を参照)。
2. ホスト A で RSVP PATH メッセージが生成されます。このメッセージには、送信側のアプリケーションで設けられるトラフィックについて記述されています。
3. この PATH メッセージが、イントラネットを介してルーター R1 に送信されます。イントラネット中のネットワーク装置によって、標準的な RSVP / Intserv 処理が行われます。
4. ルーター R1 で PATH 状態が定義され、DiffServ ネットワーク中のルーター R2 に転送されます。
5. PATH は Diffserv ネットワーク中の R2 と R3 では無視されます。これはイントラネット中の R4 に送信され、ホスト B にも送信されます。
6. ホスト B で PATH メッセージが受信されると、RSVP RESV メッセージが作成されます。このメッセージは、特定の Intserv サービス・タイプの、設けられたトラフィックを示します。
7. RESV メッセージが Diffserv ネットワークに送信されます。
8. このメッセージは、Diffserv ネットワークを透過的に送信され、ルーター R1 に伝送されます。
9. R1 で、RESV メッセージによりアドミSSION制御処理が起動されます。つまり、初期 RSVP / Intserv 要求中で要求されたリソースが、DiffServ ネットワーク中で使用できる、対応する DiffServ サービス・レベルのリソースと比較されます。対応するサービス・レベルは、Intserv から Diffserv へのマッピング機能によって判別されます。リソースの可用性は、SLA に定義されている能力によって決まります。
10. R1 で要求が承認されると、RESV メッセージの受け入れが許可され、送信側のホスト A に送信できるようになります。R1 では、この特定の伝送

インターフェース上で、受け入れ許可されたサービス・レベルで使用できる引き下げ能力で、テーブルを更新します。

11. RESV メッセージがイントラネット中の RSVP ノードで拒否されると、ホスト A に受信されます。QoS プロセスでは、メッセージの受信が、指定された Intserv サービス・タイプの、指定されたメッセージ・フローが受け入れ許可されたと解釈されます。また、DSCP のマーク付けの方法が学習されます。この方法が、これ以降このフローのパケットを送信する際に使用されます。

22.3.3 LDAP を使用した DS の構成と管理

DiffServ ネットワークでは、サービス・レベルの情報がすべてのネットワーク・エレメントに備えられることにより、特定の顧客フローの帯域幅、遅延、または除去の優先度が確実に正しく管理され、制御されなければなりません。すべての DS 境界コンポーネントに、定義されているサービス・レベルに関する同一のポリシー情報がなければなりません。同じ情報があれば、DS フィールドにマークを付けられたパケットが、すべての DS ドメインで確実に同じサービスを受け取ることができます。DS ネットワーク中の 1 つのドメインだけポリシー情報が違うと、このドメインを通過するデータ・パケットは、顧客とサービス提供者との間の SLA で契約したサービスを受け取れません。

ネットワーク管理者は、さまざまな顧客のためにさまざまなサービス・レベルを定義し、この情報をすべての境界コンポーネントに手操作で備えることができます。次回手操作で変更を加えるまで、このポリシー情報はネットワーク・コンポーネント中で静的になっています。

しかし、動的なネットワーク環境では、クラス・ベースのパケット処理動作とクラス・ベースのポリシー制御の柔軟な定義が、使用できるようになっている必要があります。管理ポリシーは稼働中の環境によって変わることがあるので、ディレクトリー・ベースのリポジトリに管理ポリシーを格納する必要があります。ポリシー情報をディレクトリーから複数の物理サーバーに配布できますが、その管理はネットワーク管理者が 1 つのエンティティに対して行います。ディレクトリー情報は、DS ネットワーク中でトラフィックの条件付けに関するポリシー情報を使用するすべてのネットワーク・エレメント、たとえばホスト、プロキシ、ルーターなどに伝搬しなければなりません。

現在の異機種混合の環境では、ネットワーク装置と管理ツールが別のベンダーによって開発されている場合がよくあります。したがって、標準化された形式を使用して、標準化された機構でディレクトリー・サーバー機能に管理ポリシーを格納し、ディレクトリー情報を DS 境界コンポーネント (ディレクトリー・クライアントの働きをする) に備える必要があります。これらの機能は、*Lightweight Directory Access Protocol (LDAP)* に備えられています。

LDAP は、一般的に使用されている業界標準のディレクトリー・アクセス手段です (330 ページの 8.4 『Lightweight Directory Access Protocol (LDAP)』を参照)。LDAP は、広く使用されているディレクトリー・アクセス用の単純なプロトコルです。さまざまなサービス・レベルのポリシー規則が LDAP スキーマとしてディレクトリーに保管されているので、ポリシーを実装している装置 (ホスト、ルーター、ポリシー・サーバー、プロキシーなど) にダウンロードできます。図 337 には、DS ネットワーク・エレメントと LDAP サーバーが協働する様子が示されています。

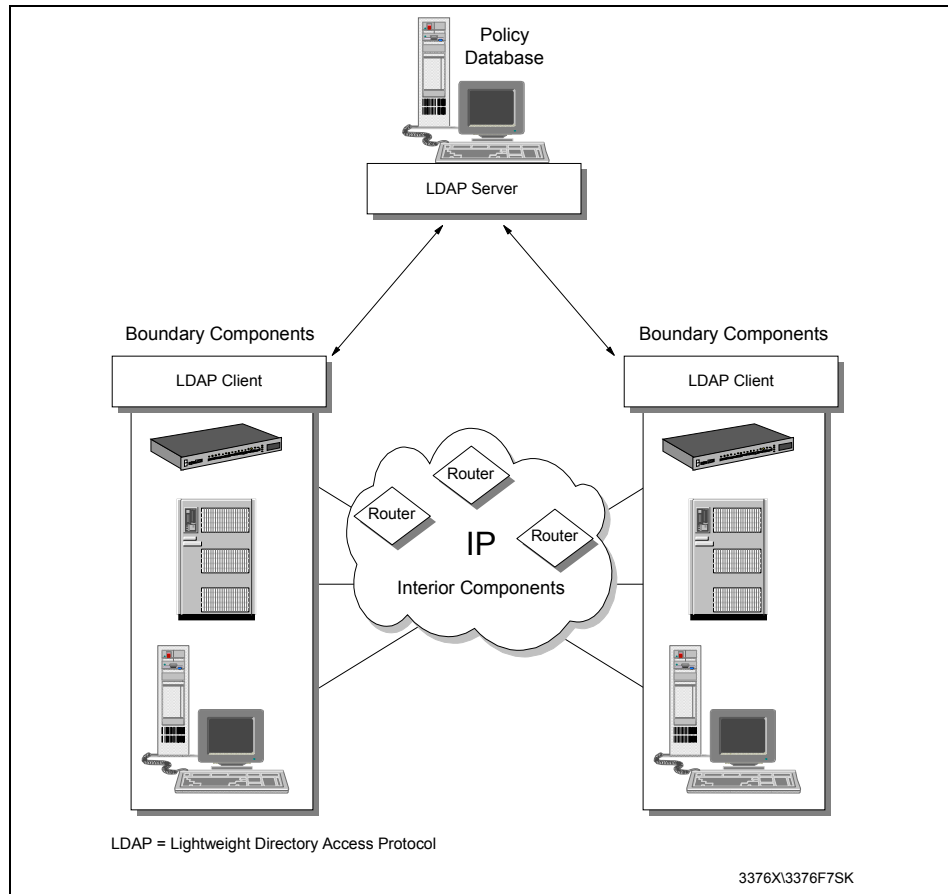


図 337 LDAP を使用した DS コンポーネントの管理

22.3.4 DiffServ と IPSec の使用

725 ページの 21.5 『IP Security Architecture (IPsec)』で説明されている IPsec プロトコルでは、暗号計算に IP ヘッダー中の DS フィールドは使用されません。したがって、ネットワーク・ノードで DS フィールドに変更を加

えても、IPsec の整合性検査は失敗しないので、IPsec の終端間セキュリティには影響はありません。したがって、DS ネットワーク中で IPsec 保護パケットを使用できます。

IPsec のトンネル・モードは、カプセル化された IP ヘッダーの DS フィールドに関するセキュリティを備えています。トンネル・モードの IPsec パケットには、トンネル開始点によって備えられる外部ヘッダーと、パケットの送信元のホストによって備えられるカプセル化された内部ヘッダーが含まれています。

IPsec トンネルがある場合の DS フィールドの処理は、以下のようになります。

1. IPsec トンネルの開始位置のノードで、外部 IP ヘッダーを付加して着呼 IP パケットがカプセル化され、ローカル DS ドメイン中の SLA に従ってこの外部ヘッダーの DS フィールドが設定されます。
2. 保護パケットが DS ネットワークを伝送され、該当する場合は中間ノードで外部 IP ヘッダー中の DS フィールドに変更が加えられます。
3. パケットが IPsec トンネルの終わりに達すると、トンネル・エンド・ノードで外部 IP ヘッダーがはく奪され、内部 (オリジナル) IP ヘッダーに含まれている情報を使用してパケットが転送されます。
4. オリジナルのデータグラムの DS ドメインが、IPsec トンネルの終了位置の DS ドメインと違う場合は、トンネル・エンド・ノードで内部ヘッダーの DS フィールドに変更を加えて、オリジナルのドメイン中の SLA に合致させなければなりません。この変更を加えると、トンネル・エンド・ノードは効果的に DS 入り口ノードの働きをします。
5. パケットが DS ネットワーク中を順方向に進んで、IPsec トンネルの終了位置に達すると、中間ノードでオリジナルの IP ヘッダーを使用して DS フィールドに変更が加えられます。

22.4 参照情報

インターネット中の QoS の詳細については、以下の RFC を参照してください。

- RFC 1349 – Type of Service in the Internet Protocol Suite
- RFC 1633 – Integrated Services in the Internet Architecture: An Overview
- RFC 2205 – Resource Reservation Protocol (RSVP) – Version 1 Functional Specification
- RFC 2206 – RSVP Management Information Base Using SMIv2
- RFC 2207 – RSVP Extensions for IPSEC Data Flows

- RFC 2208 – Resource Reservation Protocol (RSVP) – Version 1 Applicability Statement
- RFC 2209 – Resource Reservation Protocol (RSVP) – Version 1 Message Processing Rules
- RFC 2210 – The Use of RSVP with IETF Integrated Services
- RFC 2211 – Specification of the Controlled Load Network Element Service
- RFC 2212 – Specification of Guaranteed Quality of Service
- RFC 2474 – Definition of the Differentiated Services Field (DS Field)
- RFC 2988 – A Framework for Integrated Services (Intserv) Operation over Diffserv Networks).

第 23 章 可用性、拡張容易性、および負荷平準化

インターネット・ビジネスは非常に急速に成長しているので、サーバー上の主幹業務のデータやアプリケーションの連続可用性は、企業にとって重要な要件になっています。インターネットの導入によって、企業は収益を増やしたり、製品や配送に関する詳細な情報を提供するための新しいストラテジーを開発することが挑戦となります。その結果は、顧客やビジネス・パートナーを満足させるものでなければなりません。さらに、作業要員がインターネットを介してアクセスする企業内部のビジネス・プロセスの管理を、より一層最適化する必要もあります。

顧客、ビジネス・パートナー、および従業員による、アプリケーションやデータへのアクセスの需要が増えることは、新しいサーバーやネットワーキングのストラテジーやサービスを開発する際の挑戦になります。以下の 3 つの主要な局面を考慮する必要があります。

- 企業の情報の可用性を、週 7 日 24 時間有効にする方法
- 顧客やビジネス・パートナーの照会が急激に増えたために、トランザクションの数が急激に増えた場合でも、これらのサービスを保証できるようにする方法
- 並列にインストールしたサーバーの間で、サーバーのアプリケーションやデータに対するアクセスを共用できるようにする方法

可用性、拡張容易性、および負荷平準化がその答えです。この章では、可用性、拡張容易性、および負荷平準化を実現するために使用できる手法について説明します。これらのソリューションは標準化されていないので、基本的に以下の説明は、Cisco ルーターと IBM OS/390 (z/OS) で使用できる機能のみにとどめていますが、これらのプラットフォームで活用できる手法は、場合によっては他のプラットフォームにも適用できます。

23.1 可用性

アプリケーション・インスタンス、ネットワーク・インターフェース、およびマシンには障害が起こることがあります (保守のために予定されている場合と、アプリケーションやシステムのエラーのために予定外の場合があります)。このような場合に、エンド・ユーザーがサービスを受けられなくなることはありません。単にアプリケーションを再始動するという簡単な作業だけで、アプリケーション・インスタンスの障害からリカバリーできます。仮想 IP アドレス (VIPA) を使用すれば、ネットワーク・インターフェース障害も許容できます。VIPA は特定の物理インターフェースに関連付けられていないので、障害が起きることがないからです。

ただし、マシンの障害は多少複雑です。別のシステム上の代替アプリケーション・イメージを使用していることをユーザーは認識しないまま、即時にサービスに再接続できるようにする必要があります。さらに、他のシステムへのパスが自動的に変更されたことがユーザーには分からないようにする必要もあります。

参照されている可用性の要件を満たすために、別のシステム編成を適用する必要があります。そのため、複数のアプリケーション・インスタンスを、TCP/IP ネットワークへの並列接続を備えた TCP/IP スタックを組み込んだ、複数のマシン上で実行することになります。一般用語でこのソリューションのことをクラスター化の手法といい、負荷平準化の目的で使用されますが、高可用性の要件のソリューションとしても有効です。

クラスター化の手法では、接続を受け付けられるターゲット・サーバーのリストから、障害が起きたサーバー以外のターゲット・サーバーへの接続をディスパッチします。この方法によって、ディスパッチング機能では、このような要求を満たせないサーバーへの接続のルーティングは回避されます。

クラスター化の手法では、複数のマシン上で実行されている同等のアプリケーション・インスタンスが実装されている必要があります。アプリケーション、TCP/IP スタックのあるオペレーティング・システム、またはマシンに障害が起きた場合、ディスパッチングの手法を使用すると即時にバックアップできます。

特定のサーバーからサービスを要求したユーザーは、特定のサーバーのアプリケーションにアドレッシングしなくなります。これ以降は、サーバーのグループにアドレッシングします。接続要求はディスパッチャーに送信されるようになり、ディスパッチャーで転送先の使用可能なアプリケーション・サーバーが決定されます。したがって、ユーザーには接続先の (グループ中の) アプリケーション・サーバーは認識されません。

クラスター化の手法には、アプリケーションのグループを参照するアドレスが必要です。これは、仮想 IP アドレスの使用によって解決できます。仮想 IP アドレス (VIPA) とは、アプリケーション・サーバー (Telnet サーバーなど) のグループの IP アドレスのことです。この VIPA が接続要求用に使用されます。ディスパッチャーは、ユーザーからの接続要求の受信側になります。ディスパッチャーにより、使用可能なサーバーのリストから実サーバーが選択され、このサーバーに要求が転送されます。

さまざまな種類の分散規則を使用したディスパッチャーによって、使用可能なアプリケーション・サーバーを選択するプロセスを拡張することができます。接続要求の分散については、負荷平準化の項で説明します。

可用性の別の局面として、ディスパッチャーに障害が起こる場合についても考慮します。この場合、同じ IP アドレスのバックアップ・ディスパッチャーを実装し、ユーザーがバックアップ・ディスパッチャーに接続要求を送信できるようにする必要があります。バックアップ・ディスパッチャーも、その IP アドレスをネットワークに伝搬します。したがって、ルーターは、ユーザーの接続要求をバックアップ・ディスパッチャーに送信する新しいパスを使用します。

ディスパッチャーでクライアント / サーバー接続が維持されている場合は、バックアップ・ディスパッチャーでも現在実行中の接続を引き継ぐ必要があります。実行中の接続を主ディスパッチャーに戻すには、取り消しプロセスを実装する必要があります。

主分散マネージャーのリカバリー後に、稼働を止めずに取り消しプロセスを実行する例は、860 ページの 23.4.6 『DVIPA アドレスの引き継ぎ / 取り消し』で説明しています。

23.2 拡張容易性

拡張容易性によって、業務が拡大して追加のシステム能力が必要になった場合にソリューションが提供されます。顧客やビジネス・パートナーからの新しい接続要求がさらに増えたために、作業負荷の能力が低下している場合は、稼働を止めずに現行システム環境を増強することが必要になります。

従来の単一システム環境（クラスター化されていないシステム）では、稼働を止めずにシステムをアップグレードする操作は比較的制限されています。能力を上げるには、そのシステムをダウンしてから新しい機能をインストールする必要があります。したがって特定の期間、利用不能になります。

クラスター・システムを実装するのは、もっとよい方法です。新しいシステムを、同等のアプリケーション・インスタンスを実行しているクラスターに追加しても、クラスター中の他のシステムには影響はありません。このソリューションにより、業務の拡大をシームレスに実施する能力が追加されます。従来のシステムと比較すると、クラスター・サーバー環境では、ユーザーは特定のシステムに拘束されません。したがって、ユーザーがサーバーへの接続を管理する際の柔軟性が増します。新しいシステムがオンラインになると、そのマシンへの新しい接続が行われ、新しい作業負荷が引き継がれます。

23.3 負荷平準化

ユーザー接続を持つアプリケーションを特定のシステムに割り当てると、そのシステムの能力には過負荷がかかり、一方、他のシステムでは別のシステ

ム内の別のアプリケーションへの接続要求が少なくなり、能力をもて余すという場合があります。

すべてのシステムの負荷を等しいレベルにするという目標に到達するには、これらのシステムを、クラスター化されたシステム・グループに編成する必要があります。このクラスター中のすべてのシステムで、作業負荷に関する情報をディスパッチャー（分散マネージャーともいう）に通知できます。それから、このマネージャーによって、作業負荷の情報に基づいて、ユーザーからアプリケーション・サーバーのシステムへの接続要求が分散されます。

ユーザーはこのようなクラスター化には気付きません。分散マネージャーのマシンで実行されていると想定して、サービスへの接続が試行されます。分散マネージャーにより、クラスター中のすべてのシステムの現行の作業負荷に基づいて、実サービス・プロバイダーに接続要求が転送されます。すべてのターゲット・システム中のワークロード・マネージャーなどの機能により、作業負荷の状態に関する情報が提供されます。

ターゲット・システムからの作業負荷情報がない場合は、分散マネージャーで以下のような分散規則を使用できます。

- 単純なラウンドロビン分散
- 複数の分散接続

23.4 この章で使用されている用語

ソリューションについて説明する前に、この章で使用されている用語の一部について説明する必要があります。以下の用語について説明します。

- シスプレックス
- Workload Manager (WLM)
- 仮想 IP アドレス (VIPA)

23.4.1 シスプレックス

この章の文脈では、シスプレックスという用語は、疎結合の S/390 または z/Series (OS/390 または z/OS) イメージのグループを指します。たとえば、チャネルや S/390 カップリング・ファシリティ・テクノロジーを介して接続している複数の論理区画 (LPAR) や物理ホストはシスプレックスです。シスプレックスでは、これらの OS/390 または z/Series ホストを互いに協働させて、DASD (バックアップ機能を備える) を共用したり、作業負荷を分散したりできます。シスプレックスという用語は、システムまたは LPAR のクラスターにも使用されています。

詳細については、「*並列シスプレックス 概要* (GC88-6613)」を参照してください。

23.4.2 Workload Manager (WLM)

WLM とは、作業のスケジューリング、負荷平準化、およびパフォーマンス管理の制御に使用する OS/390 のコンポーネントです。OS/390 V2R10 用の TCP/IP では、WLM に対する独自のインターフェースが使用されています。このインターフェースにより、Sysplex Distributor に、負荷平準化を実行するための情報が提供されます。WLM から周期的に受け取る更新情報によって、シスプレックス中のホストで使用可能な処理リソースが最も多いものが Sysplex Distributor に通知されます。

詳細については、「OS/390 MVS 計画: ワークロード管理 (GC88-6582)」を参照してください。

23.4.3 仮想 IP アドレス (VIPA)

実際には、VIPA アドレスはアプリケーション・グループ・アドレスです。ユーザーはこのアドレスを使用して、接続要求をディスパッチャーつまり分散マネージャーに送信します。分散マネージャーにより、負荷平準化の規則に基づいてこの要求がターゲット・サーバーに転送されます。

IBM OS/390 または z/OS TCP/IP 環境では、VIPA アドレスの使用法は異なります。VIPA アドレスには以下の 3 種類があります。

- 静的 VIPA
- 動的 VIPA (DVIPA)
- 分散 DVIPA

23.4.3.1 静的 VIPA アドレス

IP アドレスは、ISO/OSI ネットワーク層アドレスで、ネットワーク層への論理接続点を表します。IP アドレスは、物理アドレスを表すデータ・リンク層上に、LAN MAC アドレスなどのハードウェア・アダプター・アドレスをマップします。

仮想 IP アドレス (VIPA) もネットワーク層アドレスですが、このアドレスはハードウェア・アダプターにマップしません。ハードウェアには障害が起こることがあり、障害が起きたアダプターを介してシステムに接続することはできません。VIPA アドレスはハードウェア・アダプターをマップしないので、OS/390 TCP/IP スタックからインターネットやイントラネットへのパス上で、接続端点として常に使用できます。ハードウェア・アダプターに障害が起きても、ルーター・ソフトウェアによって、別のアダプターを介した VIPA アドレスへの 2 番目のパスが自動的に選択されます。

VIPA アドレスは OS/390 TCP/IP プロファイル内で定義されます。これらとその後続の定義は TCP/IP スタックに属しているので、静的 VIPA アドレスといえます。

23.4.3.2 動的仮想 IP アドレス (DVIPA)

1 つの TCP/IP スタック専用として定義される静的 VIPA とは対照的に、動的 DVIPA (DVIPA) は、主 TCP/IP スタックに障害が起きた場合は別の TCP/IP スタックに移動できます。したがって、この DVIPA を介してアクセスしているアプリケーションには (立ち上げられている場合)、別のシステムからも同じ DVIPA アドレスを使用してアクセスできます。ユーザーがアプリケーションに再接続しても、この変更には気付きません。

ウェルノウン・ポート番号があるたいのアプリケーション・サーバーの場合、ネットワーク管理者は、主に TCP/IP スタックで使用するのためにのみ DVIPA を定義します。管理者はバックアップ DVIPA も定義します。これは、システム障害の場合に他の TCP/IP スタックから引き継ぐことができます (例については、859 ページの 23.4.5 『シスプレックス中の動的 IP アドレッシング』を参照してください)。

DVIPA の引き継ぎ用に複数の TCP/IP スタックを使用する場合は、バックアップ・スタック中に定義されているその DVIPA のランク番号によって、最初に DVIPA を引き継ぐべきスタックが判別されます。

基本的に、すべての VIPA はルーティング・デーモンを使用するよう設計されています。OS/390 TCP/IP は、IP ネットワークへの経路を確立するために、ルーティング・デーモン OMPROUTE または ROUTED を提供します。

23.4.3.3 分散仮想 IP アドレス

分散 DVIPA は特殊なタイプの DVIPA です。これは、シスプレックス内の接続を、同じまたは別のオペレーティング・システム・イメージ上の、別の TCP/IP スタックに分散できます。Sysplex Distributor はこれを使用して、アプリケーション・サーバーがある選択済みのターゲット・システムに接続を転送します。Sysplex Distributor は、この章で前述されているディスパッチャーつまり分散マネージャーの働きをします。

23.4.4 動的 XCF

動的 XCF とは、“システム間カップリング・ファシリティを介した動的通信” のことです。この機能を使用すると、相互通信機能では、システム間カップリング・ファシリティの接続機構を使用している同じシスプレックス中の他の TCP/IP スタックに対する接続を動的に生成できます。システム・パートナーで新しい IP スタックが検出されると、その新規スタックへの接続が自動的に定義されます。DYNAMICXCF を OMPROUTE や OROUTED などのルーティング・プログラムで使用する場合は、構成データとルーティング・テーブルも自動的に作成されます。

23.4.5 シスプレックス中の動的 IP アドレッシング

860 ページの図 338 には、3 つの IP スタックが示されており、それぞれが 1 つのシステム内にあります。システムは別のハードウェア上に存在させることも、または 1 つ以上のシステム内の論理区画 (LPAR) に存在させることもできます。管理者は、他の TCP/IP スタックに引き継ぐことができる DVIPA アドレスを定義しています。スタック間で引き継ぐことができる定義済みの DVIPA BACKUP アドレスもあります。別々のシステム中に等しい DVIPA アドレスが複数ある場合、高いランクで定義されているシステムに DVIPA アドレスが引き継がれます。DVIPA アドレスをアプリケーション・インスタンスと関連付けることもできます。

本書の例 (860 ページの図 338 を参照) では、システム TCP1 で DVIPA アドレス 10.1.1.1 および 10.1.1.2 が主に使用されるよう定義されています。TCP1 に障害が起こると、相互カップリング・ファシリティーを介して、システム 2 では障害に関する情報を入手します。これによって、アプリケーション・サービスを備えるためには、バックアップ・アドレス 10.1.1.1 および 10.1.1.2 を活動化する必要があることが認識されます。TCP3 では、10.1.1.1 の定義済みのバックアップのランクが TCP2 より低いので、このバックアップは活動化されません。

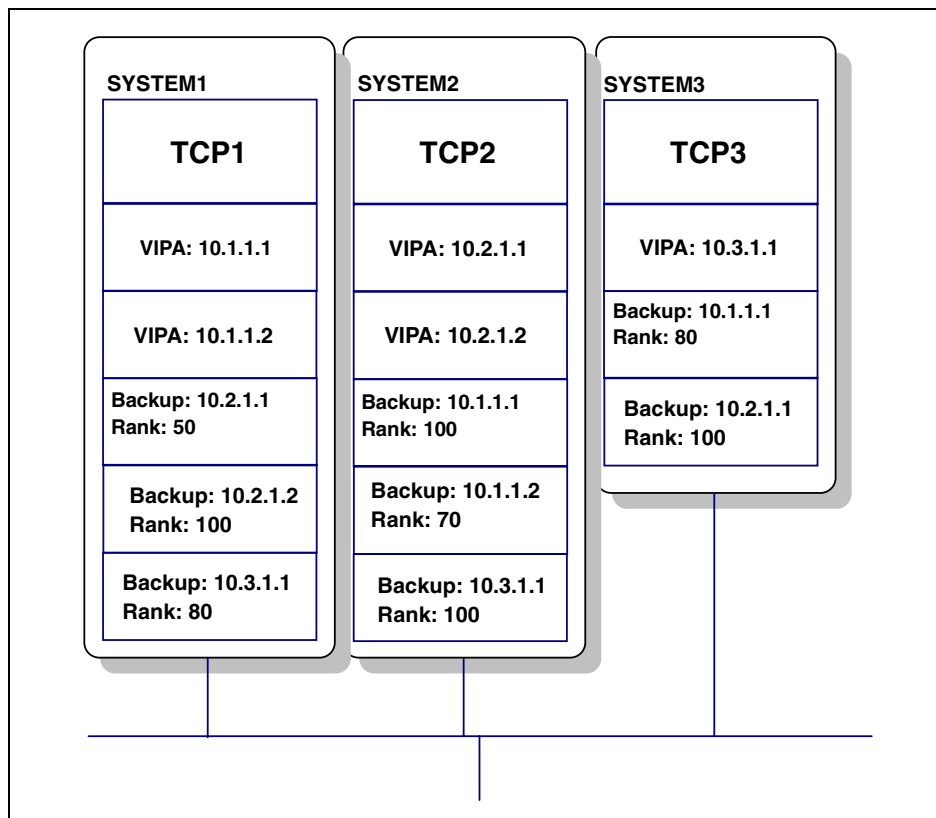


図 338 動的 VIPA アドレスと他のシステムにあるその定義済みバックアップ

DVIPA を活動化すると、バックアップ・システムは自動的に TCP/IP プロファイル中に定義を作成し、バックアップが必要なくなるとこれらの定義を削除することが必要になります。管理者の作業は、この動的 DVIPA アドレスのバックアップ・システムを定義することのみです。

23.4.6 DVIPA アドレスの引き継ぎ / 取り消し

引き継ぎ (図 339 の 1): TCP/IP スタックやシステムに障害が起きた場合には、別の TCP/IP スタック上にこの DVIPA がバックアップとして定義されていれば、このスタックへの DVIPA の引き継ぎが自動的に実行されます。同一の DVIPA を複数のバックアップ・システムに定義できます。指定されたランクによって、どのシステムが DVIPA を引き継ぐ必要があるかの順番が決まります。

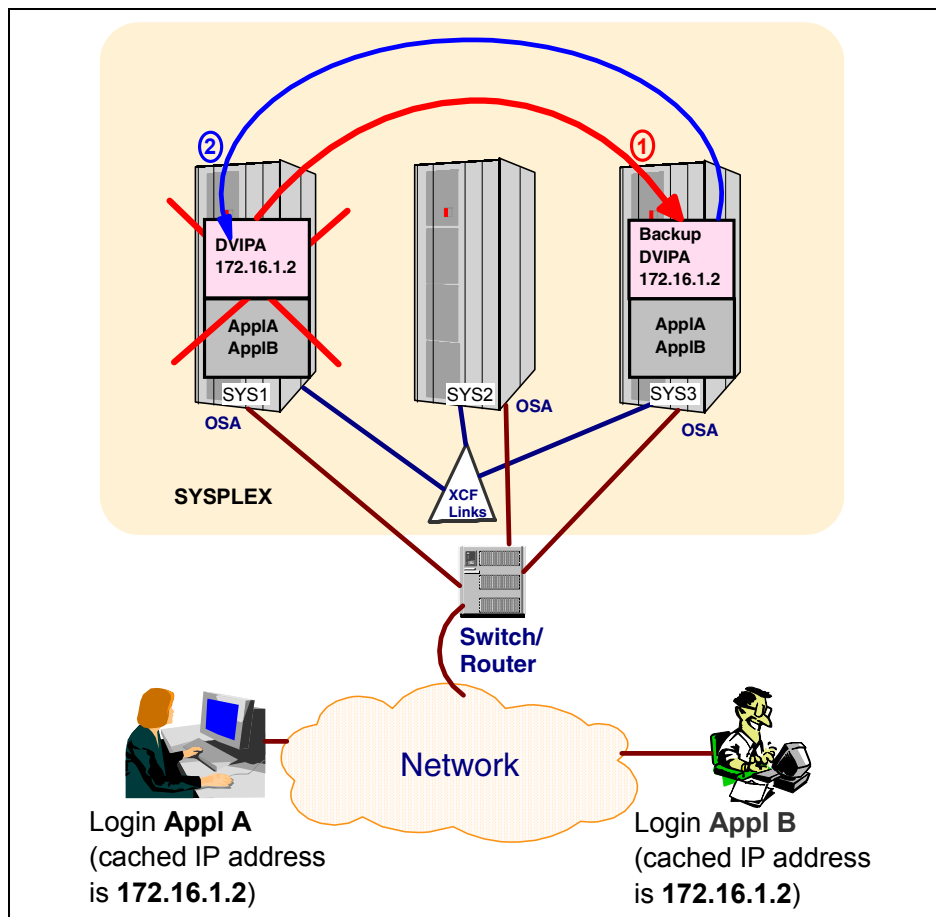


図 339 DVIPA の引き継ぎ/取り消し

図 339 では、3 つのシステムが 1 つのシスプレックス・クラスターに属しています。すべてのシステムは、相互カップリング・ファシリティーを介してリンクで接続されています。SYS1 では、2 つのアプリケーション (ApplA および ApplB) を実行しています。両方とも DVIPA 172.16.1.2 をマップしています。システム障害の場合、SYS3 が DVIPA 172.16.1.2 を引き継ぐ必要があります。

実際に TCP/IP スタックまたはシステムの障害が起きたとします。バックアップ定義とランク定義に基づいて、SYS3 は自動的に DVIPA 172.16.1.2 を引き継ぎます。SYS3 の TCP/IP スタック中のプロファイル定義が動的に更新されて、アプリケーション ApplA および ApplB は DVIPA の活動化を開始します。このスタックは、ルーティング・プロトコルに新しい DVIPA を通知し、これ

によってそのルーティング・テーブルが更新されます。このルーティング・プロトコルにより、他の近隣ルーターに DVIPA の引き継ぎが通知されます。この時点でクライアントからアプリケーションに再接続できます。クライアントでは、アプリケーションが別のシステム上で実行されるようになったことを認識しません。

取り消し (図 339 の 2): 取り消し手順はさらに複雑です。まず障害が起きたシステムを再度立ち上げる必要があります。シスプレックス・クラスター中の他のすべてのシステムで、障害が起きたシステムの再始動に関する情報が受信されます。この情報は、相互カップリング・ファシリティーのリンクを介して送信されます。動的 XCF 機能により、シスプレックス内のシステム間の IP パスが動的に定義されます (本書の例では、SYS2 から SYS1 へのパスと SYS3 から SYS1 へのパス、およびこれらの逆)。SYS1 中の DVIPA 172.16.1.2 も使用可能ですが、アクティブではありません。

次に、取り消しを編成する方法を決定する必要があります。問題は、現行接続を中断せずに作業負荷を SYS1 に送り戻すということです。以前の OS/390 のソリューションでは、SYS1 中の新しい VIPA アドレスに作業負荷を送信することしかできませんでした。しかし、この場合には、ネーム・サーバーは 1 つの古い VIPA アドレスのみにマップしているので、クライアントは、アプリケーション名 (ApplA など) を使用するよりも、VIPA アドレスにログオンする必要がありました。

OS/390 バージョン 2.10 の TCP/IP では、重複した DVIPA を使用できます。SYS1 TCP/IP スタックを再始動する際には、DVIPA 定義に基づいて以下の 2 つの選択肢があります。

1. 中断せずに DVIPA を即時に移動する必要があるかどうか
2. バックアップ・スタックで事前に接続を引き継がなくなった場合に DVIPA を移動する必要があるのか、それとも中断せずに取り消しプロセスを行う必要があるかどうか。

1 番目の場合には、以下の機能が備えられます。

- このスタックは、DVIPA 172.16.1.2 の所有者である。これは IP ネットでルーター・デーモンに伝搬されます。つまり、新しい接続要求が SYS1 にルーティングされます。
- 現行接続はバックアップ・スタック (本書の例では SYS3) に残るが、SYS1 上のスタックで管理される。クライアントは、中断せずに、アプリケーション・サーバー (ApplA または ApplB) との処理を実行します。

2 番目の場合には、DVIPA で SYS3 上の端点として実行されている接続がない場合に限り、この DVIPA が SYS1 に送り戻されます。つまり、DVIPA が *活動停止* 状態になっている場合か、または中断の方法によって DVIPA を SYS1 に管理者が戻したい場合かということです。

23.5 使用可能なソリューションの紹介

依然としていくつかの疑問が残っています。たとえば、以下のような疑問です。

- 言及されている状況で、クライアント / サーバー接続要求の処理に使用できるディスパッチャーつまり分散マネージャーのハードウェアとソフトウェアの種類。
- ディスパッチャーつまり分散マネージャーのインストール場所。クラスター内、IP ネットワーク中のマシン、クラスターを制御するルーター (1 次および 2 次) のいずれか。

IBM および Cisco の独自のソリューションについては、続くセクションで説明します。まず、クラスターの外部に実装されるディスパッチャーの説明から始めます。以下のディスパッチャーについて説明します。

- Network Dispatcher (WebSphere Edge Server の一部)
- Cisco Local Director

次に、完全なクラスター内部ソリューションである、IBM OS/390 Sysplex Distributor について説明します。さらに、Cisco MultiNode Load Balancing (MNLB) が提供する内部機能と外部機能を比較します。

両社の共同の調査開発プロジェクトに基づいた共同ソリューションもあります。これは、コーディングを改善して追加したハードウェアとソフトウェアの両方の製品の利点を最大限に活用することを目指しています。このソリューションは、IBM Sysplex Distributor CASA Manager と Cisco Forwarding Agent で構成されます。

最後に、従来のソフトウェア製品を使用した他のソリューションについて説明します。以下のソフトウェアについて説明します。

- TCP/IP for OS/390 Domain Name Server、Workload Manager を使用 (DNS/WLM)
- Virtual Router Redundancy Protocol (VRRP)
- ラウンドロビン DNS
- NAT ベースの手法

23.6 Network Dispatcher

Network Dispatcher (以前の Network Dispatcher (ND)) は、負荷平準化に関する新しい方法の基礎になります。続くいくつかのセクションでは、このディスパッチャーについて説明し、他のソリューションと比較します。

23.6.1 Network Dispatcher のコンポーネント

Network Dispatcher は、前述の制限に対処するために開発され、顧客のサイトの拡張容易性と可用性の必要を満たすための拡張機能を備えています。これは以下の 2 つのコンポーネントで構成されています。

Interactive Session Support 前述のラウンドロビン DNS の制限に対処しつつ、クライアントに以前と同じ DNS インターフェースを提供します。すでにラウンドロビン DNS を使用してデプロイされたアプリケーション用に、中断を最小限にとどめるマイグレーション・パスを提供します。

Dispatcher ラウンドロビン DNS の代わりにインストールできる拡張 IP レベル負荷平準化機構を提供します。Dispatcher は、インストールし終わると、クライアントには完全に見えませんが、負荷平準化、管理、および可用性に関する優れた機能を送達できます。

これらの 2 つのコンポーネントは、さまざまな構成中に別々にデプロイしたり一緒にデプロイしたりできるので、顧客の広範囲なアプリケーションの要件に合わせることができます。

23.6.1.1 Interactive Session Support

Interactive Session Support (ISS) は、DNS ベースのコンポーネントです。これは負荷モニター・デーモンを提供し、ご使用のシステムの構成部分になっているサーバーごとにインストールできます。このデーモンのグループのことをセルといいます。セルのメンバーの 1 つは、負荷モニター・サービスの「スポークスマン」になります。

標準メモリーやプロセッサの使用状況の図を使用して、ISS デーモンに伴う負荷を測定したり、あるいは、一例としてアプリケーションが認識できる実行可能モジュールや単純なシェルスクリプトに、基準のカスタム・セットを提供して、ISS に使用方法を指示することもできます。これを、カスタム測定基準といいます。

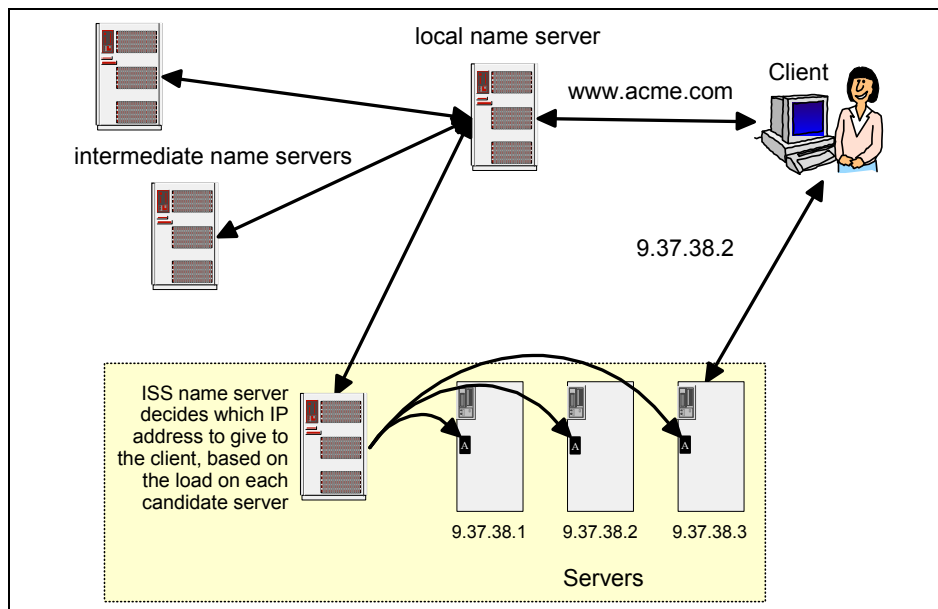


図 340 Interactive Session Support

ISS には監視インターフェースが備えられており、他のアプリケーションで負荷モニター・サービスを使用できるようになっています。監視アプリケーションにより、セルが監視され、負荷に基づいてアクションが開始されます。Network Dispatcher には、ネーム・サーバー監視とディスパッチャー監視という、2 種類の監視アプリケーションが備えられています。

1. 図 340 のように、ネーム・サーバー監視により、負荷モニター・サービスにインテリジェント DNS ネーム解決サービスを提供することができます。このサービスは、(gethostbyname() 呼び出しの応答として) 厳密なラウンドロビンのローテーションではなく、個々のサーバー・マシン上で実行されている ISS デーモンからのフィードバックに基いて、クライアントに IP アドレスを戻します。また、サーバーがダウンしている場合は、IP アドレスは戻されません。
2. ディスパッチャー監視は、Dispatcher コンポーネントと共に使用されます。これについては後述します。

PING 三角測量モードで ISS を構成することもできます。この場合は、ISS デーモンはそれぞれからの PING 時刻に基づいてクライアントに応答します。したがって、ネットワーク・トポロジーに基づく負荷平準化を実現できます。

23.6.1.2 Dispatcher

Dispatcher は IP レベルのロード・バランサーです。これは、IBM 社の調査部が特許を持つテクノロジーに基づいた、基本的に他と異なる負荷平準化の方法を使用します。Dispatcher では、いかなる方法でも DNS を使用しませんが、Dispatcher の前では普通は通常の静的 DNS が依然として使用されています。Dispatcher がインストールされて構成を終えると、実際には Dispatcher が、クライアントからすべてのパケットが送信されるサイト IP アドレスになります。この外部に公示されているアドレスが、クラスター・アドレスと呼ばれます。クラスター・アドレスは必要な数のみ定義できます。個々のクラスター中でサポートされる必要があるポートを構成してから、これらのポートごとにサポートを提供する実際のサーバーを構成できます。オプションとして、ゲートウェイ・ルーターでフィルター操作して、クラスター中のサーバーの実 IP アドレスをクライアントから隠すこともできます。このオブジェクト指向のクラスター / ポート / サーバー構造によって、真の 24 x 7 演算まで可能な、動的に作成および変更可能な単純な構成インターフェースが備えられます。詳細については、図 341 を参照してください。

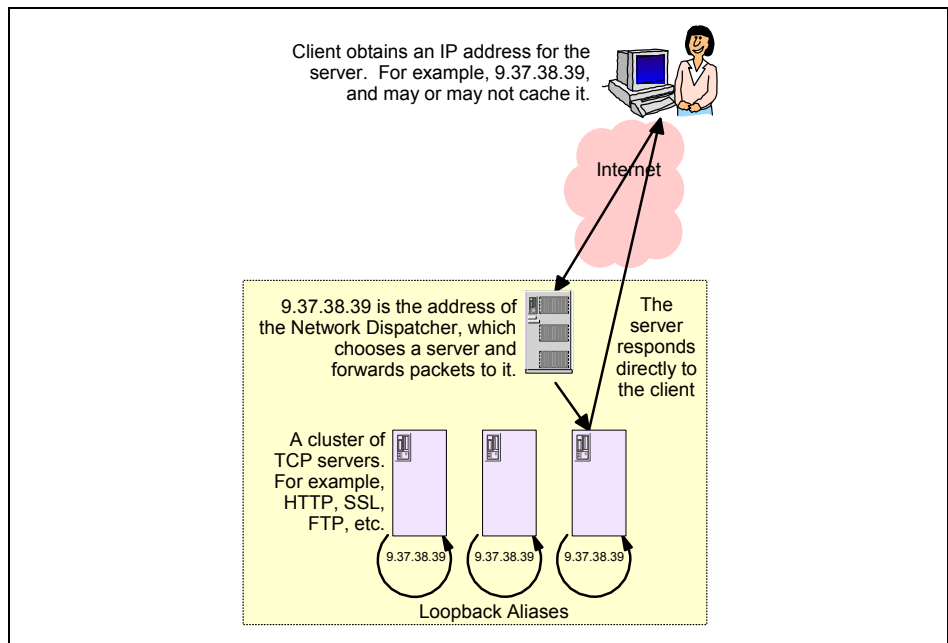


図 341 Dispatcher

Dispatcher のコア機能は Executor です。この機能は、カーネル・レベルの機能（つまり装置ドライバー）で、個々のパケットのヘッダーのみ検査し、パケットが既存の接続に属しているのか、それとも新しい接続要求を表しているのかを判別します。これを実現するために、この機能では、メモリー中に

格納されている単純接続テーブルを使用します。Dispatcher のマシン (クライアントとサーバーの間にあるので、Dispatcher がインストールされていないような状態になる) 上で接続が実際にセットアップされることはありませんが、接続テーブルに、その存在とその送信先のサーバーのアドレスが記録されることに注意してください。

接続がすでに存在する (メモリー内の接続テーブルに既存の項目がある) 場合は、追加処理は行われずに、パケットは初期接続要求で選択されたのと同じサーバーに即時に転送されます。フローしているパケットのほとんどはこのタイプなので、負荷平準化処理全体のオーバーヘッドは最小限に保たれます。これは、パフォーマンスと拡張容易性の点で Dispatcher が他の製品より優れている理由の 1 つです。

パケットが新しい接続要求である場合は、Executor が構成を調べ、要求されたクラスター・アドレスで、クライアントから要求されたポート上の要求をサポートできるサーバーを確認します。それから、選択されたサーバーごとに格納されている重みを使用して、接続の転送先として正しいサーバーが判別されます。このサーバーについて説明する項目が接続テーブル中に作成され、この接続に関する後続のパケットは選択済みのサーバーに確実に正しく転送されます。

適格なすべてのサーバーによって負荷が共用されることが望ましいので、正しいサーバーが必ずしも最善のサーバーとは限らないことに注意してください。最悪のサーバーが負荷を担う必要が生じることさえあります。トラフィックが常に最善のサーバーにのみ転送されるとすると、そのサーバーはすぐに最善でなくなることは確実です。その結果、負荷割り振りが変動するので、最適な平準化は実現しません。Dispatcher には正しいサーバーの選択に関する専用のアルゴリズムがあり、この拡張された平滑化技法により最短の時間で最適な平衡化が実現します。

Executor では、転送時にはクライアントの IP パケットに変更は加えません。Dispatcher はそのクラスター・サーバーと同じサブネット上にあるので、パケットを通常の IP パケットと同様に、選択されたサーバーの IP アドレスに明示的に転送するのみです。Dispatcher の TCP/IP スタックは、オペレーティング・システムで承認されている方法でパケットの MAC アドレスのみに変更を加え、選択したサーバーにパケットを送信します。

このサーバー上の TCP/IP スタックで、Dispatcher から未変更のパケットを受諾し、通常のアプリケーション処理のために選択済みポートに渡すことができるようにするには、Dispatcher マシンの IP アドレスも、非公示の別名として個々のクラスター・サーバー上にインストールします。これを実現するには、ループバック・アダプター上で別名を構成します。

それから、サーバーの TCP は、独自の基本構成からソース・アドレスとターゲット・アドレスを判別するのではなく、単にクライアントから示されたこれらのアドレスをスワッピングすることにより、標準的な TCP のセマンティクスに従って、サーバーからクライアントへの接続を確立します。これは Dispatcher の IP アドレスを使用してクライアントに応答することを意味します。直接の結果として、平衡化機能はクライアントとクラスター・サーバーのどちらにもわからなくなります。このような不可視性により、サーバーのプラットフォームに標準的な TCP/IP プロトコルが実装されていたとしても、Dispatcher はこれらのプラットフォームに依存しません。

パフォーマンスと拡張容易性における Dispatcher の別の主要な利点は、アプリケーション・サーバーは、クライアントの要求に対する応答を、Dispatcher を介さずに直接にクライアントに戻すことです。実際には、オリジナルの物理パスを使用して戻す必要さえなく、独立した高帯域幅の接続を使用できます。多くの場合、サーバーからクライアントへのアウトバウンド・トラフィックのボリュームは、実質的にインバウンド・トラフィックより大きくなります。たとえば、サーバーから送信する Web ページの HTML と組み込みイメージは、一般に要求元のクライアントの URL のサイズの少なくとも 10 倍以上になります。この機能は、1998 年長野冬季オリンピックの Web サイトのスケーリング要件に取り組む上で重要でした。

Dispatcher は本当の汎用 TCP/IP アプリケーションなので、この機能は HTTP や FTP のトラフィックのみでなく、他の規格準拠タイプである TCP や UDP のトラフィックにも適用できます。

23.6.2 負荷平準化と重み

Dispatcher には Manager 機能があり、この機能により Executor が従う重みを設定します (図 342 を参照)。Manager でこれらの重みを設定する際には、以下の 4 つのメトリックが使用されます。

- 個々のポート上のサーバーごとのアクティブな接続の数。この数は、Executor の内部で保持されます。
- 個々のポート上のサーバーごとの新しい接続の数。この数は、Executor の内部で保持されます。
- 個々のサーバーが稼働中かどうかのチェック。この機能は Advisor によって実行されます。
- 個々のサーバーに代行能力があるかどうかのチェック。つまり、現実には作業を処理できるかどうかチェックされます。この機能は ISS Load Monitor によって実行されます。

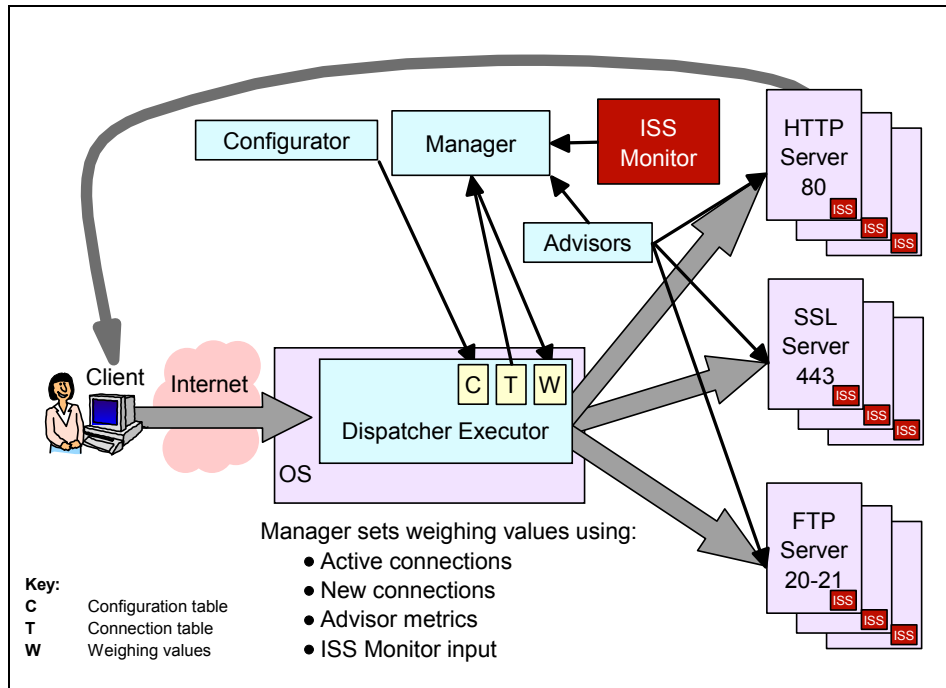


図 342 Dispatcher のコンポーネント

Advisor は、Dispatcher の一部として実行される軽量クライアントですが、実際に実コマンドを個々のアプリケーション・サーバーに渡します。これは単純なコマンドのみ実行しますが、TCP/IP プロトコル・スタックが起動しているかどうかのみを検証する PING よりは、意味のあるものにする必要があります。この要求が正常実行された場合は、サーバーが起動されていると想定されます。失敗した場合は、Advisor から Manager に通知され、そのサーバーからの再応答がなければ Manager によりそのサーバーの重みがゼロに設定されます。

アプリケーション・サーバーに ISS Load Monitor デーモンがインストールされていると（カスタム・メトリックがあってもなくても）、Dispatcher 監視を介して負荷に関するレポートを定期的に Dispatcher に渡すことができます。これらのレポートの結果を、Dispatcher の個々のサーバー重みの設定プロセスに利用できます。

23.6.3 高可用性

Dispatcher にはすでに高可用性が備えられています。なぜなら、その基本機能の 1 つは、その背後でクラスター化されている、障害が起きたサーバーを選択しないようにすることだからです。高可用性を実現するようにハード

ウェアとソフトウェアの包括的な実装の一部として、機能停止点となるロード・バランサー自体を除去するように、Dispatcher を構成することもできます。

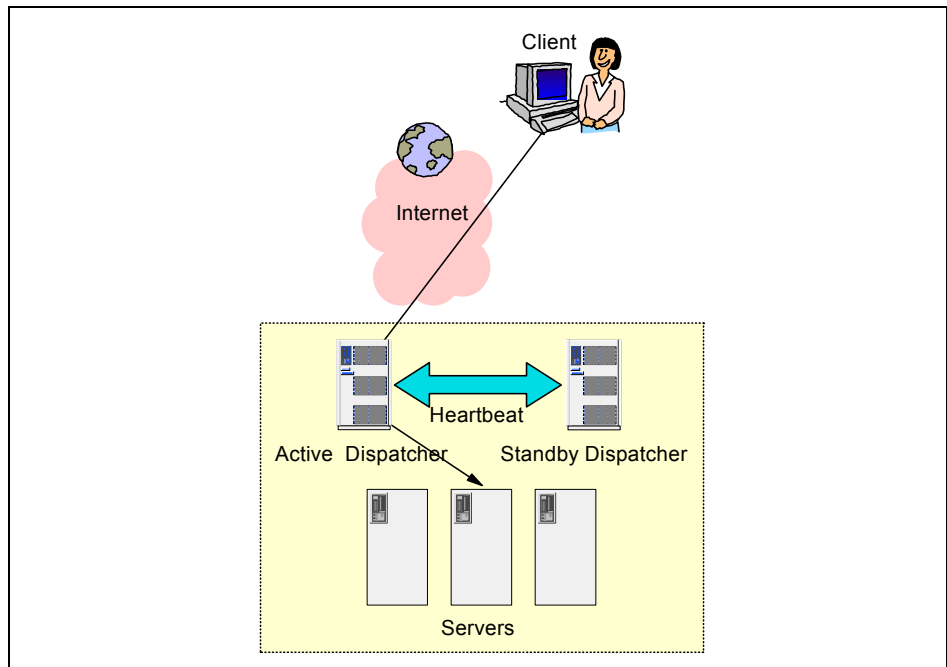


図 343 高可用性

図 343 のように、Dispatcher はオプションで、その状態（およびその接続テーブルの状態と、1 次 / アクティブ・マシンの状態）と同期する同じサブネット上の 2 次 / 待機マシンで構成することができ、1 次 / アクティブ・マシンからのハートビートを受信します。

さまざまなユーザー定義のアクセス可能性基準も追加指定できます。これはたとえば、重複したアダプターやネットワーク間でのゲートウェイ・ルーターへのアクセスなどです。ハートビートが失敗したり、定義されているアクセス可能性基準が満たされなかったりすると、1 次マシンがダウンしていると想定され、パケット転送の役割は待機マシンに引き継がれます。一般には、フェイルオーバー・プロセスに要するのは 5 秒未満で、発生する可能性がある接続障害の回数は最小限になります。ARP 項目の経過時間切れに基づく他のフェイルオーバー方法は、完了するまでに 1 分程度要することがあります。

障害が起きた場合には、待機マシン上の接続テーブルは、障害が起きている 1 次マシンの接続テーブルと密接に同期化されるので、使用中の既存の接続の大多数には障害後にも継続します。受信したすべてのパケットの送信先は、

新しくアクティブになったマシンでも認識されており、実際のフェイルオーバー中に失われた個々のパケットは TCP によって自動的に再送されます。障害が起こる可能性が最も高い接続は、開始または終了プロセス中だった接続です。開始要求の場合、クライアントには障害を認識されずに、クライアントの TCP/IP スタックまたはアプリケーションは再試行して正常に実行することができます。終了要求の場合、たいていはデータは失われていません。

23.6.4 サーバーの類縁性

この機能のことを、スティッキー・オプションともいいます。この機能は、クライアントに代わって接続間で特定の種類の永続状態を保存し、クライアントの再接続時にこの状態を失わないアプリケーションを負荷平準化するために備えられています。

元々クライアントがサイトと接続していた場合、通常の方法で選択されたサーバーに関連付けられます。スティッキー・オプションを構成した場合、相違点は、構成可能なタイムアウト値が満了するまで、クライアントから送信される後続の接続がこの同じサーバーにディスパッチされるということです。Dispatcher を使用して、ポート単位でこのスティッキー機能を構成できます。

23.6.5 規則ベースの平衡化

規則ベースの平衡化は、規則と、その規則に従う場合に負荷平準化する一連のサーバーという概念を導入します。以下の規則を使用できます。

- クライアント IP アドレス
- クライアント・ポート
- 時刻
- ポートの秒当たりの接続数
- ポートのアクティブ接続数

これを使用すると、負荷平準化のポリシーをセットアップする際に、サイトにおいて、そのトラフィックや、そのサイトにアクセスするクライアントの識別を考慮することができます。必要に応じて範囲を指定できます。たとえば、特定のサブネット中のすべてのクライアントの IP アドレスを特定のサーバーのセットに転送したり、「午前 8 時から午後 5 時の間は 10 個のサーバーのセットを使用し、それ以外の時間は 5 つのサーバーのセットを使用」したりできます。これによって、クライアント / サーバーの関係について固有のまたは独自のセマンティクスやプロトコルを使用しなくても、個人、クライアントのグループ、または時刻に対するサービス品質を簡単に実装できます。

23.6.6 広域 Network Dispatcher

以前のリリースの Dispatcher では、すべてのクラスター・サーバーが Dispatcher と同じサブネット上にある必要がありました。バージョン 2 の Dispatcher では、私設ネットワーク内でも、さらにはインターネットを介してでも、リモート・サーバーに対する距離上の制限はありません。数千キロ離れていても、地球上の別の場所に分散していてもサイトを提供することができます。リモート・ロケーションではもう 1 つの Dispatcher をインストールしなければなりません。

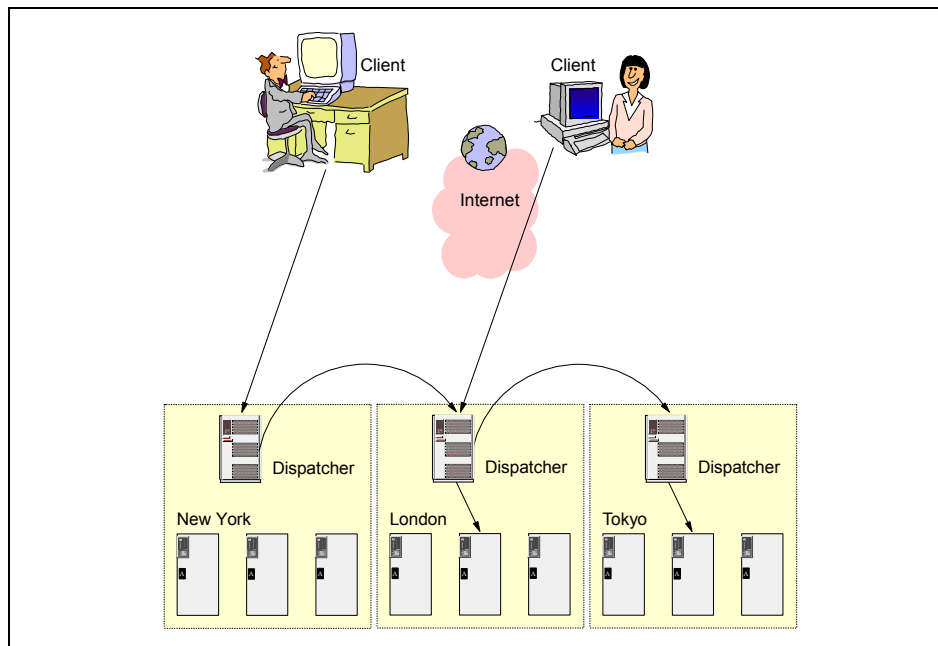


図 344 広域 Network Dispatcher

図 344 のように、このオプションを使用すると、正真正銘の分散した広域ネットワーク・サイトを構成できます。任意のクライアント・データを、任意の場所にある任意の構成済みサーバーに転送できます。サーバーの負荷、規則、または両方の組み合わせに基づいて選択できます。複数の方法で、最初にクライアントのパケットを受信するサイトを選択できます。その中には、PING 三角測量モードで実行している ISS に備えられている、「最も近接しているサイト」モード (873 ページの 23.6.7 『ISS と Dispatcher の結合』を参照) も含まれます。

リモート・サイトにパケットを伝送するには、発信側の Dispatcher で未変更のクライアント・パケットをカプセル化してから、受信側の Dispatcher でカプセル化解除します。サーバーからクライアントへのデータ・フローは、

ローカル・エリア・ネットワークの Dispatcher を使用した場合と同じように直接送信されるので、オーバーヘッドは最小限にとどめられます。

23.6.7 ISS と Dispatcher の結合

Network Dispatcher 製品の 2 つのコンポーネントを効果的に一緒に配置して、地理的に分散しているサイトをサポートできます。

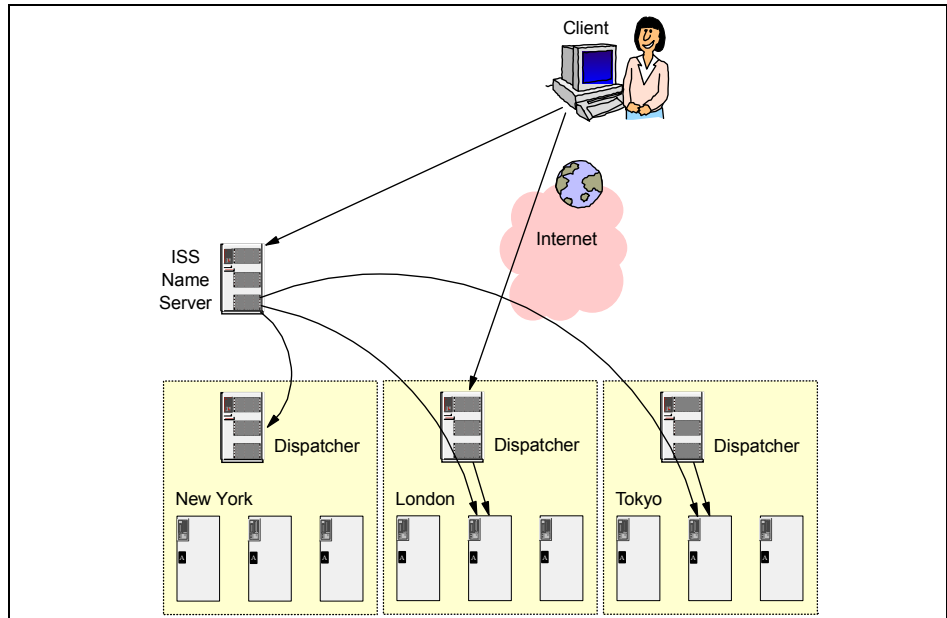


図 345 ISS と Dispatcher の協働

図 345 のように、アプリケーション・プログラムからの標準的な名前解決要求に対する応答として、個々のサイトで稼働している Dispatcher マシン上で実行されている ISS デーモンからのフィードバックに基づいて、選択したサイトの IP アドレスが ISS ネーム・サーバーからクライアントに戻されます。以下の 2 つの方法のうちの 1 つで結果を戻すように、ISS デーモンを構成できます。

1. システム負荷に基づいて、最も負荷の小さいサイトをクライアントに指示する。必要に応じて、ISS デーモンでカスタム測定基準を使用できます。
2. PING 三角測量に基づいて、ネットワーク・トポロジーの視点から最も近接したサイトをクライアントに指示する。

それから、接続要求がクライアントから、選択した場所の選択した Dispatcher マシンに送信されます。次に、独自の重みの集合に基づいてこのサイトのクラスターから個々のサーバーが選択されます。この構成に、前述

の広域ネットワーク Dispatcher サポートを組み込むことにも注意してください。

23.6.8 Advisor およびカスタム・アドバイザー

Dispatcher には HTTP、HTTPS (SSL)、FTP、NNTP、SMTP、POP3、および Telnet 用の標準的なアドバイザーが備えられています。顧客は、Dispatcher で他の規格や専用のプロトコルを使用したり、標準的なアドバイザーに特定の拡張を実装したりできます。Custom Advisor 機能と、高品質なサンプル・ソース・コードがこの目的で備えられています。カスタム・アドバイザーは、Java 開発キット リリース 1.1.2 以上の Java で作成されていなければなりません。これは Dispatcher に備えられていません。

この強力な拡張機能と、作成したカスタム・コードを結合して、アプリケーション・サーバー・マシン上で実行し、平衡化している Dispatcher とアプリケーションの間の協同の度合いを高くすることができます。この方法を使用している本書の例では、Web サーバー中で実行しているカスタム・アドバイザーと Java サーブレットの間のリンクになっています。個々のサーバーからパフォーマンス・データを徹底的に抽出して Dispatcher に戻すように、サーブレットをコード化できます。このようにすると、顧客の実アプリケーション環境に合わせて細部にわたって調整された結果を Dispatcher で利用できます。

23.6.9 SNMP サポート

Dispatcher データの Simple Network Management Protocol (SNMP) 管理情報ベース (MIB) が備えられています。この MIB には、Dispatcher の状態と現行のパフォーマンスに関連した包括的な値の集合が含まれています。Tivoli NetView や他の同種の装備ツールなどの SNMP を使用できる管理ツールから、この MIB にアクセスできます。さらに、クラスター・サーバーに障害が起きたり、Dispatcher 自体の待機がフェイルオーバーしたりすると、SNMP トラップも生成されます。

23.6.10 場所共有オプション

企業が小規模なサイトからスタートして、その後急速に拡張する計画を立てている場合、拡張の初期段階ではコストを低く抑えながら、Network Dispatcher の高可用性と拡張容易性のオプションの利点を活用し、拡張していく時点でその場所も決められるという、単純なオプションがあります。

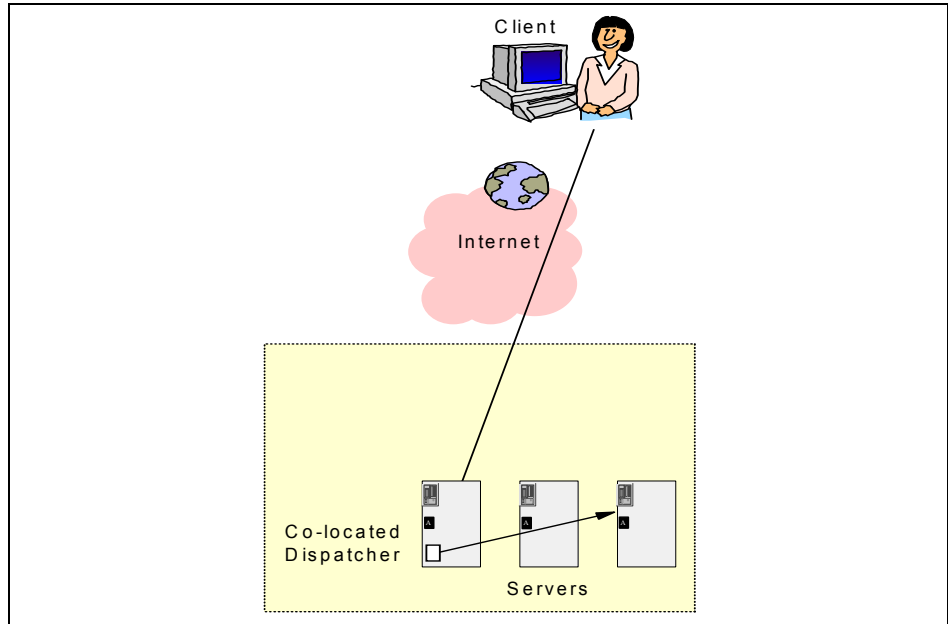


図 346 場所共有オプション

図 346 のように、複数のサーバーでトラフィックを処理する必要があるほどサイトの負荷が大きくなると、この会社は、アプリケーション・サーバーがあるマシンの 1 つに Dispatcher ソフトウェアをインストールすることにより、最小限のハードウェアとソフトウェアの投資で、既存のネットワークの基礎構造に Dispatcher を追加できます。この方法の場合、既存のネットワーク構成に変更を加える必要は全くありません。最も単純な場合、Dispatcher を構成し、アプリケーション・サーバーの内容を別のサーバーに複製するか、ファイル共有システムを使用して共有するだけで、すべての必要を満たせます。

Dispatcher の場所共有オプションを使用すると、会社は時間をかけずに最小限のコストでスタートし、トラフィックのボリューム上の必要が生じた際に、独立型の Dispatcher が高可用性の構成に発展させることができます。専用 Dispatcher が配置されている場合でも、待機 Dispatcher とアプリケーション・サーバーの 1 つの共有場所を選択できます。

23.6.11 ISP の構成

インターネット・サービス・プロバイダー (ISP) が、大規模なバックボーン・ネットワークを使用していたり、使用頻度の高い検索エンジンのサービスを提供している場合は、高可用性モードの Network Dispatcher と、大容量のキャッシング・プロキシ・サーバーや会社のファイル共有システムを結合

することにより、拡張容易性と可用性の要件に対処できます。図 347 のように、これらのコンポーネントは IBM WebSphere Edge Server に付属しています。

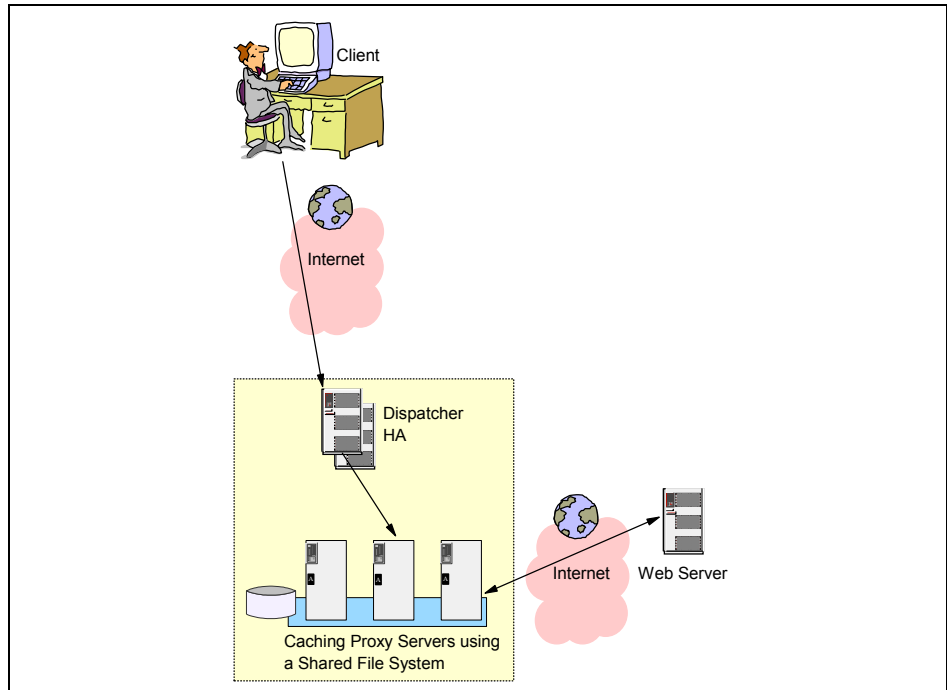


図 347 IBM WebSphere Edge Server

キャッシュを効率よく管理すると、バックボーン・ネットワークの輻輳を大幅に減らすことができるので、クライアントに送達される応答の時間が短くなり、顧客サービスが向上する一方で、トラフィックのコストを減らすことができます。

23.6.12 OS/390 の並列シスプレックスのサポート

OS/390 並列シスプレックス上にエンタープライズ・アプリケーション・サーバーを配備している会社では、OS/390 の Workload Manager (WLM) からの情報を、Dispatcher 機能の負荷平準化プロセスに対する入力データとして使用できます。

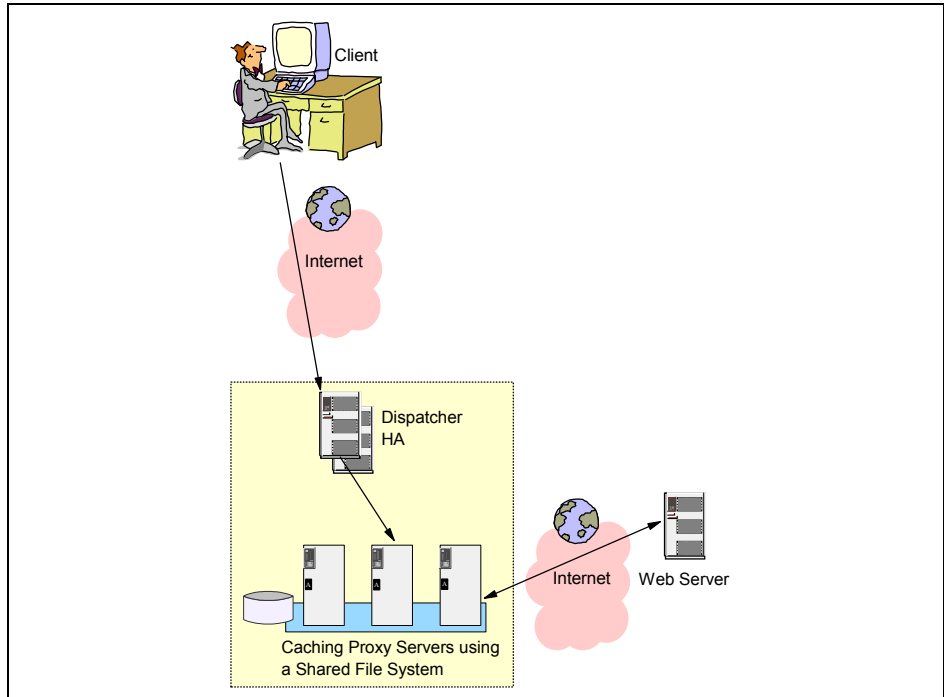


図 348 Dispatcher と OS/390 並列シスプレックス

このバージョンの Dispatcher には、WLM Advisor という追加のアドバイザーがあります。図 348 のように、WLM Advisor は OS/390 Workload Manager と対話し、ISS Dispatcher 監視と同じ形式で Dispatcher の Manager に調整済みの結果を送ります。WLM Advisor の測定基準により、構成されているサーバーごとに以下のものが測定されます。

- サーバーが使用可能かどうか。
- サーバーに代行能力があるかどうか。
- 重要性が低いと想定されている作業が現在サーバーで行われているかどうか。
- プロセッサの速度。
- 不足条件がないかどうか。

WLM の詳細については、857 ページの 23.4.2 『Workload Manager (WLM)』を参照してください。

23.7 Cisco LocalDirector

LocalDirector のクライアント接続とデータ・フローの機能性と説明について、以下に概要を示します。

23.7.1 概説

LocalDirector は、冗長なフォールト・トレラント・サーバー・ファームを作成するために設計された、特殊なネットワーク装置です。これは透過的な確認ブリッジのような働きをして、着信データ・パケットを IP ネットワーク（インターネットまたはイントラネット）から、サーバー・ファーム中の選択されたアプリケーション・サーバーに転送します。LocalDirector では、負荷平準化の方式を使用して、TCP や UDP の内容を含む IP データグラムが送信されます。

図 349 には、IP ネットワーク中の LocalDirector の場所が示されています。

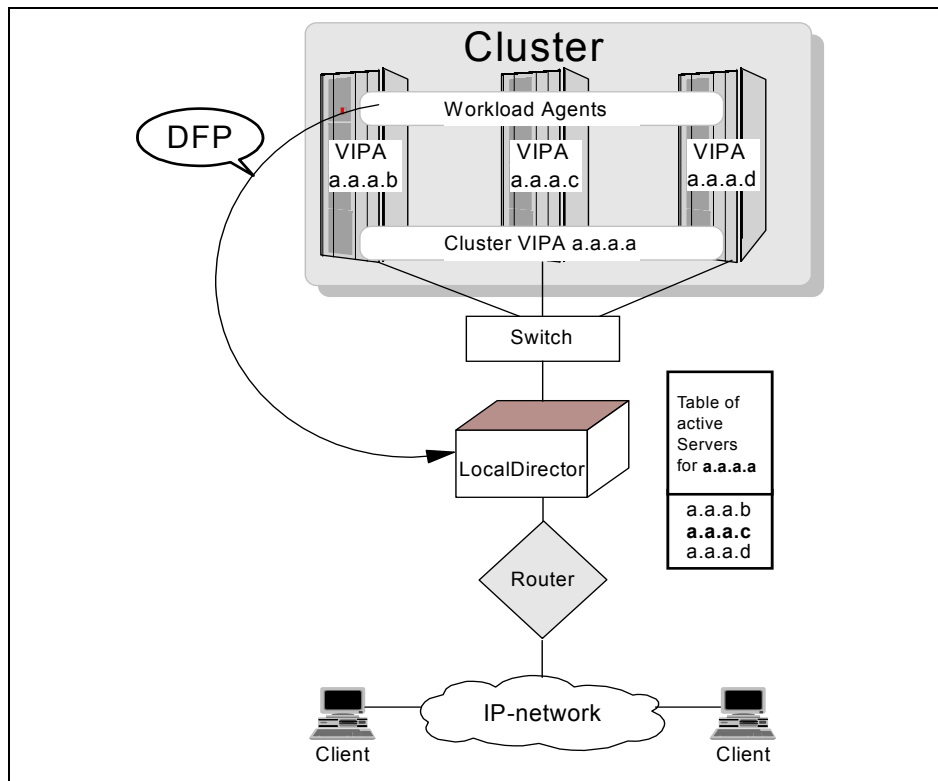


図 349 IP ネットワークとサーバー・クラスターのための LocalDirector ブリッジ

すべてのサーバーが 1 つの仮想サーバーとして表されます。したがって、サーバー・クラスターのすべてのメンバーに対して、1 つの IP アドレス (a.a.a.a など) と 1 つの URL のみ必要です。

LocalDirector はクラスター中のサーバーと通信し、アプリケーション・サーバーの状態やサーバーの現行の作業負荷などの情報を入手します。この情報は、Dynamic Feedback Protocol (DFP) を介してサーバーから LocalDirector に転送されます。IBM OS/390 には、LocalDirector の DFP パートナーである Workload Agent があります。

DFP は、サーバーの可用性と負荷に関する情報を、ネットワークのリモート・サイトにある Cisco Distributed Director (図 349 にはなし) に配布するためにも使用されます。この機能により、Web サイトからクライアントへの応答時間が短縮されます。

LocalDirector は、HTTP メッセージをリダイレクトできます。この機能は主に、サーバーの障害やサーバー・クラスターの障害が起きた場合に別の場所に Web トラフィックをリダイレクトするために使用されます。実サーバーがすべて使用できなくなった場合のために、別の Web サイトへのクライアントのリダイレクトが備えられています。

LocalDirector には、待機 LocalDirector に即時フェイルオーバーを準備する機能もあります。障害状態になると、フェイルオーバー機構により、発信元の LocalDirector の構成が待機 LocalDirector に自動的に複製されます。発信元の LocalDirector で保守されていた、サーバーに対する現行ユーザー接続は、2 次 LocalDirector によって保守されるようになります。クライアントとサーバーの間の接続は除去されません。

23.7.2 接続とデータグラムのフロー

クライアントからサーバーへのアクセスは、ドメイン・ネーム・サーバーによってサーバー・クラスター IP アドレスに変換された単一の URL を使用して行われます。このアドレスのことを、クラスターの仮想 IP ともいいます。静的 VIPA も IBM シスプレックス中で使用できます。

サーバー・クラスターを表す LocalDirector の IP アドレスが認識されると、クライアントによって TCP ハンドシェイク・プロセスが開始されます。

通常は、ハンドシェイク・プロセスは以下の操作で構成されています。

- クライアントからサーバーへの、接続の第 1 段階を要求する SYN 要求の送信
- サーバーからクライアントへの、SYN ACK 応答 (肯定応答) の送信

- サーバーからクライアントへの、接続の第 2 段階 (サーバーからクライアントへの接続) を要求する SYN 要求の送信
- クライアントからサーバーへの、ACK 応答の送信

LocalDirector を介したサーバー・クラスターへの接続のセットアップは異なります。

実サーバーはクライアントによって認識されないため、サーバー・クラスターの IP アドレスが使用されます。しかし、このアドレスは、サーバー・クラスターの代行機能として LocalDirector に接続するためにのみ使用されず、リソースでクライアント要求を使用できるかどうかを認識するのは、LocalDirector のみです。使用できる場合は、サーバー・クラスターの IP アドレスが実サーバーの IP アドレスに割り当てられます。

この手順では、LocalDirector で最初の SYN 要求に割り込む必要があります。まず、LocalDirector で負荷平準化の規則に基づいて、選択する必要のある実サーバーを検出しなければなりません。選択プロセスが終了したら、実宛先 IP アドレスを割り当て、選択したサーバーに SYN を転送します。また、この LocalDirector を介したすべてのクライアント / サーバー接続のリストに、この接続を追加します。この情報は、2 次 LocalDirector への引き継ぎが生じた場合に重要になります。

TCP ハンドシェイク・プロセスは、実サーバーとクライアントの間で継続されます。LocalDirector を介してパケットが流れるたびに、接続が依然として実行中であることと、アプリケーションとシステムが作業中であることが LocalDirector で認識されます。

23.8 IBM Sysplex Distributor

以下の内容について説明しています。

- Sysplex Distributor のエレメント
- 開始タスクと引き継ぎ / 取り消しタスク
- 負荷平準化の規則
- 接続要求の処理

23.8.1 Sysplex Distributor のエレメント

Sysplex Distributor の環境では、1 つのシスプレックス・クラスター中に複数のシステム / LPAR があります。すべての TCP/IP スタックは、他のシステムには相互カップリング・ファシリティーによって接続し、IP リンクは、LAN スイッチ / ルーターに OSA アダプター (OSA Express など) またはチャネル接続ルーターを介して接続します。

885 ページの図 350 に、Sysplex Distributor のコンポーネントと、クライアントとアプリケーション・サーバーの間でやり取りされるデータ・フローに関する概要が示されています。

Sysplex Distributor は、TCP/IP プロファイル中に定義されている TCP/IP 機能です。これには、DVIPA と、その TCP/IP アプリケーションとの関連に関する情報が含まれます。本書の例では、DVIPA は 172.7.1.1 で、ポート 20 と 21 を介してアクセスしているファイル転送デーモン (FTP) に関連付けられています。Sysplex Distributor には、この DVIPA の分散先になるターゲット・システムに関する情報も含まれます。本書の例では、FTP に関連した DVIPA はシステム A、B、および C に分散されます。これらのターゲット・システムには、同等の DVIPA (FTP 関連の 172.7.1.1 など) があります。ターゲット・システム上の DVIPA は隠れた DVIPA で、Sysplex Distributor のみがアドレッシングできます。ルーティングのためにネットワークに伝搬されません。

ネットワークの IP アドレス 172.16.0.0 は、相互カップリング・リンクに属しています。図が煩雑にならないようにするために、スイッチ / ルーターに対する接続の IP アドレスは図示されていません。

23.8.2 Sysplex Distributor の初期設定および引き継ぎ / 取り消し

Sysplex Distributor (本書の例では SYS1) を開始すると、相互カップリング・リンクの動的 XCF アドレスと、分散 DVIPA を、定義されているすべてのターゲット・スタックに通知します (858 ページの 23.4.3.3 『分散仮想 IP アドレス』を参照)。

ターゲット・スタックでは、分散 DVIPA が独自の TCP/IP プロファイルにルーティング不能 DVIPA として追加されます。これらの分散 DVIPA はネットワークに伝搬されません。ターゲット・スタックから分散スタックに、分散 DVIPA をマッピングするサーバー・アプリケーションの状態が通知されます。この状態や他の特定の情報は、Sysplex Distributor で接続要求をターゲット・システムに転送するために使用されます。

システムが TCP/IP に障害が起きた場合は、バックアップの Sysplex Distributor (本書の例では SYS2) により、SYS1 のすべての活動が引き継がれます。この処理が行われる方法については、860 ページの 23.4.6 『DVIPA アドレスの引き継ぎ / 取り消し』で説明しています。

引き継ぎの時間を短縮するために、1 次 Sysplex Distributor (本書の例では SYS1) からバックアップ Sysplex Distributor (SYS2) に、ターゲット・スタックの分散 DVIPA について通知されます。したがって、1 次 Sysplex Distributor で保守されている現行接続は、バックアップ Sysplex Distributor で常に認識されています。1 次で障害が起きた場合でも、これらの接続を保守しなければなりません。

保守には以下の作業が含まれます。

- ターゲット・スタック上のアプリケーションにアクセスするには、Sysplex Distributor で、ポートに関連付けられているアクティブ DVIPA に関する宛先ポート・テーブルを作成して更新する必要がある。
- 接続要求を分散するには、“ 現行の ” ルーティング・テーブルも保守する必要がある。これらのテーブルの内容は、オペレーター表示画面で以下のものなどを表示するのに使用されます。
 - 作業負荷を受け入れようとしているアプリケーションで、現在使用できる DVIPA ターゲット・スタックとポート
 - このターゲット・スタックを送達するのに使用する動的 XCF IP アドレス
 - DVIPA およびポート・ターゲット・スタックにルーティングされる接続の数
 - DVIPA およびポート・ターゲット・スタックで現在有効な WLM 重みと QoS 値

23.8.3 Sysplex Distributor の負荷平準化の規則

着信接続要求に関するさまざまな負荷平準化の規則があります。以下の規則に基づいて分散できます。

- ターゲット・システムとアプリケーションの状態に関する Workload Manager (WLM) の情報
- 追加の TCP/IP アプリケーションであるポリシー・エージェントによって提供される、WLM 情報とサービス品質 (QoS) 情報
- 順次負荷平準化である “ ラウンドロビン ”

23.8.4 接続要求の処理

クライアントからの接続要求がインターネットかイントラネットを介して Sysplex Distributor で受信されると (885 ページの図 350 の 1 を参照)、Sysplex Distributor ではテーブルが検索され、要求された分散 DVIPA を含む、希望するアプリケーションがあるターゲット・システムが検出されます。負荷平準化の規則に基づいて、該当するターゲット・システムが選択されます。本書の例では、Sysplex Distributor により、ターゲット・システム C が FTP 接続用に選択されています。次に、相互カップリング・ファシリティーを介したリンクを使用して、分散 DVIPA に接続要求が転送されます (885 ページの図 350 の 2 を参照)。

アプリケーション・サーバーにより、最初の SYN 要求がクライアントに送信されて、接続確立プロセスが開始されます。クライアントへの戻りは Sysplex Distributor を介しません (885 ページの図 350 の 3 を参照)。ネットワーク、

スイッチ、またはルーターに直接（本書の例では、OSA アダプターを介して）戻されます。

23.8.5 接続確立後のデータ・パス

確立後、この接続に関するデータ・フローには、クライアントから常に Sysplex Distributor と相互カップリング・リンクを介してアプリケーション・サーバーに達するパスが設けられます。アプリケーション・サーバーからクライアントへも、OSA アダプターを介したスイッチ / ルーターへの直接パスが設けられます。

インバウンド・トラフィックの場合、Sysplex Distributor のために多少のボトルネックが生じることがあります。しかし、アウトバウンド・トラフィックではネットワークに対する別のパスを使用しているので、このボトルネックは重要なこととは見なされません。Web サーバー・トラフィックなどのほとんどの接続では、インバウンド・データの量はアウトバウンド・データの量よりはるかに少なくなります。

このソリューションの利点は、すべての機能がシスプレックス・クラスター中に内部的に配置されることです。外部システムやルーター上の機能はありません。

システムの定義や変更は、OS/390 管理による処理のみで行われます。

23.8.6 引き継ぎ / 取り消し

23.8.6.1 引き継ぎ

1 次 Sysplex Distribution スタックに障害が起きると、この障害に関する情報が、相互カップリング・ファシリティを介してすべてのバックアップ分散スタックに通知されます。ランクが最も高いバックアップ分散スタックにより、1 次分散スタックの開始時にこのスタックで確認されたすべての分散 DVIPA が活動化されます。

新しい分散スタック（バックアップ・スタック）からすべてのターゲット・スタックに、以後このスタックがすべての“隠れた”DVIPA の分散スタックになることが通知されます。再度ターゲット・スタックからバックアップ分散スタックに、アプリケーション・サーバーと現行接続を表す DVIPA とポートの状態が通知されます。

バックアップ分散スタックで、操作サポート用の宛先ポート・テーブルと現行ルーティング・テーブルが作成されます。最後に、引き継がれた分散 DVIPA がネットワークに伝搬されます。この処理により、ルーターのルーティング・テーブルとパスを、シスプレックス・クラスター中の新しい入り口点に更新できます。新しい入り口点は、バックアップ分散スタックの IP アドレス（本書の例では SYS2 の OSA アダプター）になります。

23.8.6.2 取り消し

1 次シスプレックス分散スタックが再始動されます。定義されている分散 DVIPA はすべて活動化されます。1 次シスプレックス分散スタックから、バックアップ分散スタックとすべての定義済みのターゲット・スタックに、接続の取り消しの準備ができたことが通知されます。ターゲット・スタックから 1 次分散スタックに、“隠れた” DVIPA とポートの状態が通知されます。

バックアップ分散スタックから 1 次分散スタックに、現行のルーティング・テーブルの内容とすべての接続が転送されます。また、戻された DVIPA を削除して分散ルーティング・テーブルが更新されます。ただし、この処理はこれらの DVIPA が“即時に移動可能”と事前定義されている場合に限り行われます。

最後に、1 次分散スタックの分散 DVIPA がネットワークに伝搬されます。ルーターのルーティング・テーブルとパスが更新され、再び 1 次シスプレックス分散スタックに接続できるようになります。

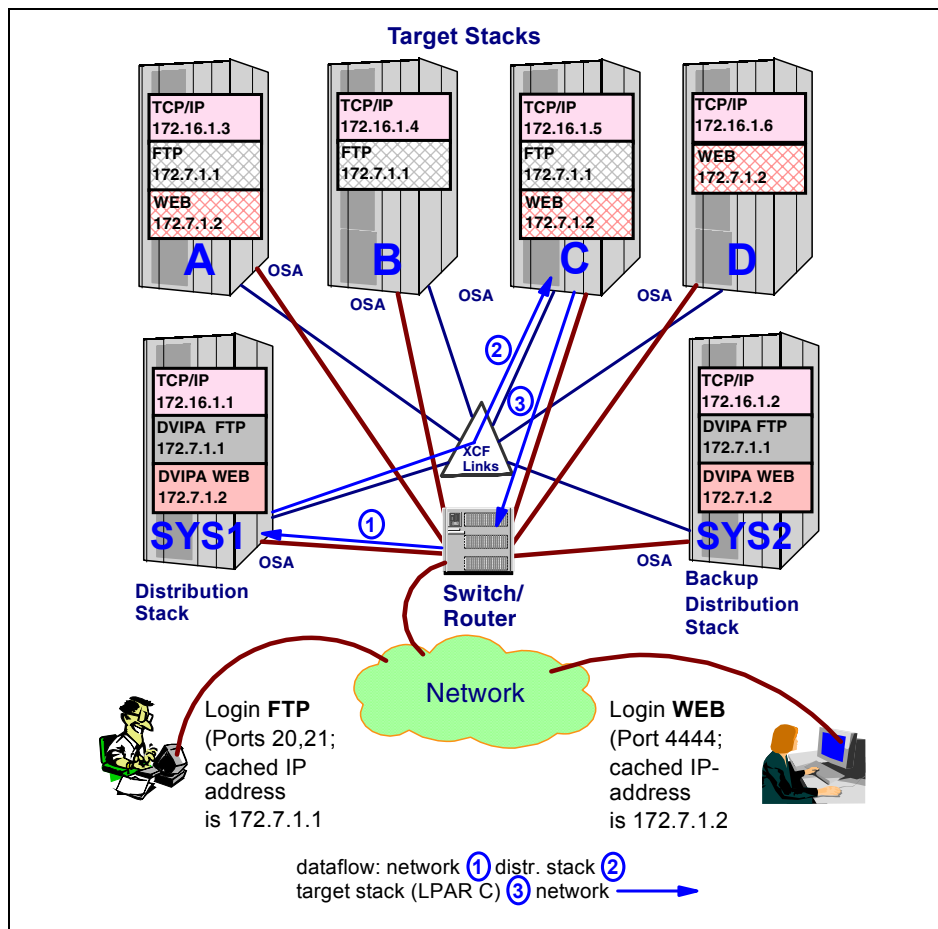


図 350 Sysplex Distributor 環境とデータ・パス

23.8.7 可用性、拡張容易性、および負荷平準化の達成

アプリケーションの可用性は、システム /LPAR 上に複数のターゲット・スタックを実装し、同等のアプリケーション・インスタンスを並列稼働することによって達成されます。1つのシステム /LPAR やアプリケーションに障害が起きても、エンド・ユーザーは同じアプリケーション名を使用して即時に再接続できます。このアプリケーション名は、別のシステム /LPAR 中にある、同一の“隠れた”DVIPAを指します。

1つ以上のバックアップ Sysplex Distributor スタックにより、Sysplex Distributor スタックへの永続アクセスも提供されます。定義済みのバック

アップ・スタックと 1 次分散スタックの間で稼働している接続の引き継ぎと取り消しは、再接続せずに自動的に処理されます。

すべての分散スタックでは、ワークロード・マネージャーやポリシー・エージェントからの情報を使用して、シスプレックス・クラスター中のターゲット・アドレスのうち現時点で最善のものを選択する際に、適切な決定を行います。

23.9 Cisco MultiNode Load Balancing (MNLB)

Cisco 社の可用性、拡張容易性、および負荷平準化の目標を達成する方法は、IBM Sysplex Distributor のソリューション (880 ページの 23.8 『IBM Sysplex Distributor』を参照) と比べると、完全に異なっています。

アプリケーション・サーバーの可用性は、サーバー・クラスターに接続されたネットワークの視点から考慮されます。サーバー・クラスターへのアクセスは、スイッチかルーターを介して行われます。

サーバー・クラスターのみでなく、以下の制御全体が外部的に行われます。

- サーバー・クラスター中の使用可能なアプリケーション・サーバーへのアクセス
- サーバー・クラスターへのアクセスの拡張に関する拡張容易性
- サーバー・クラスター中の個々のサーバーの作業負荷の管理
- 要求されたクライアント / サーバー接続にとって現時点で最善なマシンを選択する負荷平準化プロセス
- 選択されたサーバー・ターゲットへのこの接続要求の転送

この制御を行うには、クラスター内のサーバー中の情報が、イントラネットに接続したマシン (スイッチ、ルーター、または他のサーバー) に実装された特別なサービス・アプリケーションに転送される必要があります。

これらのサービス・アプリケーションのことを、MNLB コンポーネントといいます。以下の 4 つのコンポーネントを使用できます。

- Service Manager
- Forwarding Agent
- Workload Agent
- Backup Service Manager

887 ページの図 351 には、MNLB コンポーネントの場所が示されています。

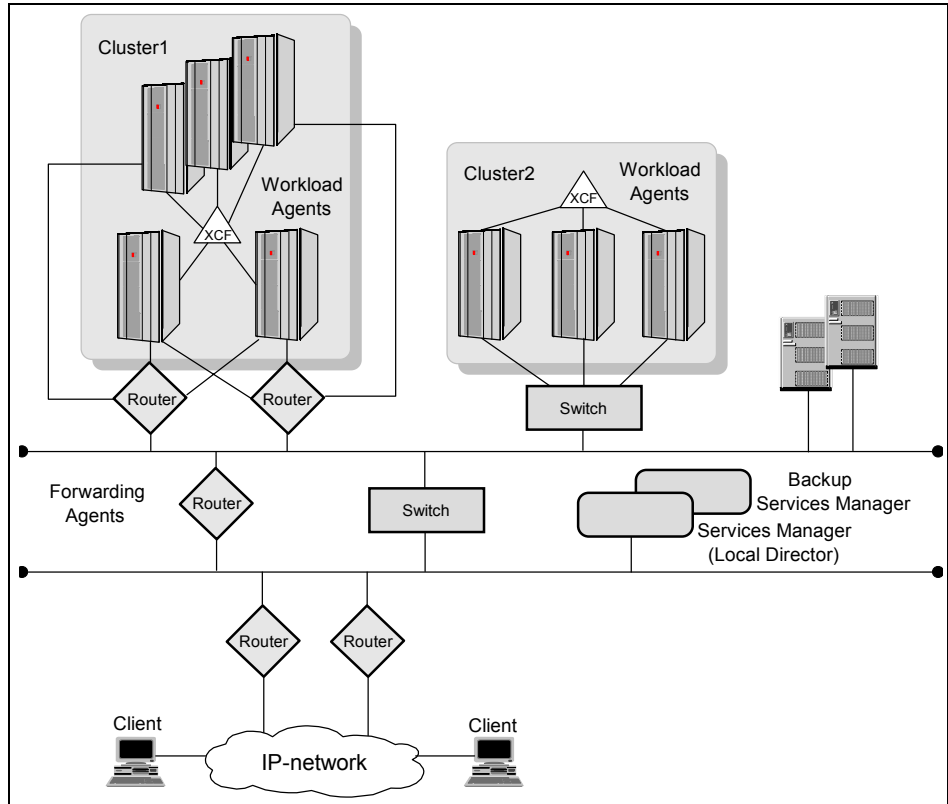


図 351 MNLB コンポーネント

23.9.1 MultiNode Load Balancing 機能の概要

MNLB は、新しい e-commerce アプリケーションや e-business アプリケーション用の、サーバー負荷平準化のソリューションとして設計されています。MNLB は、Cisco のルーターやスイッチ、Cisco LocalDirector、およびアプリケーション・サーバー・プラットフォーム内で実行されるソフトウェアで構成されています。これは、多数のルーター間で負荷平準化情報に基づいて TCP 接続要求と IP データグラムを分散し、サーバー・アプリケーション (これらも IBM シスプレックス環境中にある) の可用性、拡張容易性、およびパフォーマンスが最大レベルになるようにします。

Service Manager は、Cisco LocalDirector 中にあり、接続要求の分散を行います。この処理の際には、アプリケーションの可用性、サーバー・プロセッサの能力、および負荷平準化のアルゴリズム (ラウンドロビン、最小接続数、Dynamic Feed Back Protocol (FDP) を介して受信した情報など) に関する

る情報が使用されます。このプロトコルは、IBM OS/390 Workload Manager (WLM) の情報などを提供します。

Workload Agent は、サーバー選択時に使用する最善の負荷平準化の結果を *Service Manager* で計算するのに必要な情報を提供します。*Workload Agent* は、サーバー・プラットフォーム、またはサーバー・ファームやクラスター化サーバー環境を管理するマシン上で実行されるソフトウェアです。

Cisco Workload Agent for OS/390 では、IBM OS/390 Workload Manager (WLM) データが使用されます。これらのデータは、共通使用される Dynamic Feed Back Protocol (DFP) に変換されてから、*Service Manager* に送信されます。Cisco Workload Agent for OS/390 は、IBM シスプレックス環境での負荷平準化を最適化します。

Forwarding Agent は、*Service Manager* の指示に基づいてパケットを転送するパケット・リダイレクターとして使用されます。*Forwarding Agent* は、Cisco ルーターか経路切り替えモジュール中で実行されるソフトウェアです。

Backup Service Manager は、1 次 *Service Manager* に障害が起きた場合に、接続を確立します。

23.9.2 接続とその後のデータ・フローの確立

接続とその後のデータ・フローを確立するステップは以下のとおりです。

1. クライアントからアプリケーション・サーバーに接続要求が送信され、接続が開始されます。
2. ネーム・サーバーで、ドメイン・ネームが仮想 IP アドレスに解決されます。このアドレスのことをクラスター IP アドレス、クラスター VIPA、または分散 VIPA ともいいます。これは総称 IP アドレスで、クラスター化環境中で実行されているアプリケーションをアドレッシングします。負荷平準化プロセスの後には、このアドレスを実 IP アドレスに割り当てる必要があります。
3. *Forwarding Agent* でこの接続要求が受信されると、この要求が *Service Manager* に転送されます。*Forwarding Agent* では、SYN フラグがオンの状態で TCP ヘッダー検査を行うことにより、接続要求が検出されます。
4. *Workload Agent* によって周期的に提供される情報に基づいて、*Service Manager* で負荷平準化が決定されます。サーバーのクラスターから最適なアプリケーション・サーバーが選択されます。
5. *Service Manager* から *Forwarding Agent* に、最適なアプリケーション・サーバーの IP アドレスを含む接続要求が送り返されます。
6. したがって、*Forwarding Agent* で特定の接続要求の宛先が確認されます。この接続要求がアプリケーション・サーバーに転送されます。

7. サーバーで接続要求が受け取られ、受け取った SYN 要求の肯定応答を送信して接続の確立が試行されます。また、SYN 要求がクライアントに送信されます。
8. クライアントで SYN 要求が受け入れられると、受け取られた SYN が確認され、Service Manager によって提供される IP アドレスを使用して最初のデータが直接アプリケーション・サーバーに送信されます。

既存の接続で以後 Service Manager との連絡は行われません。

以下の図は、これまで説明してきた接続フローを示しています。

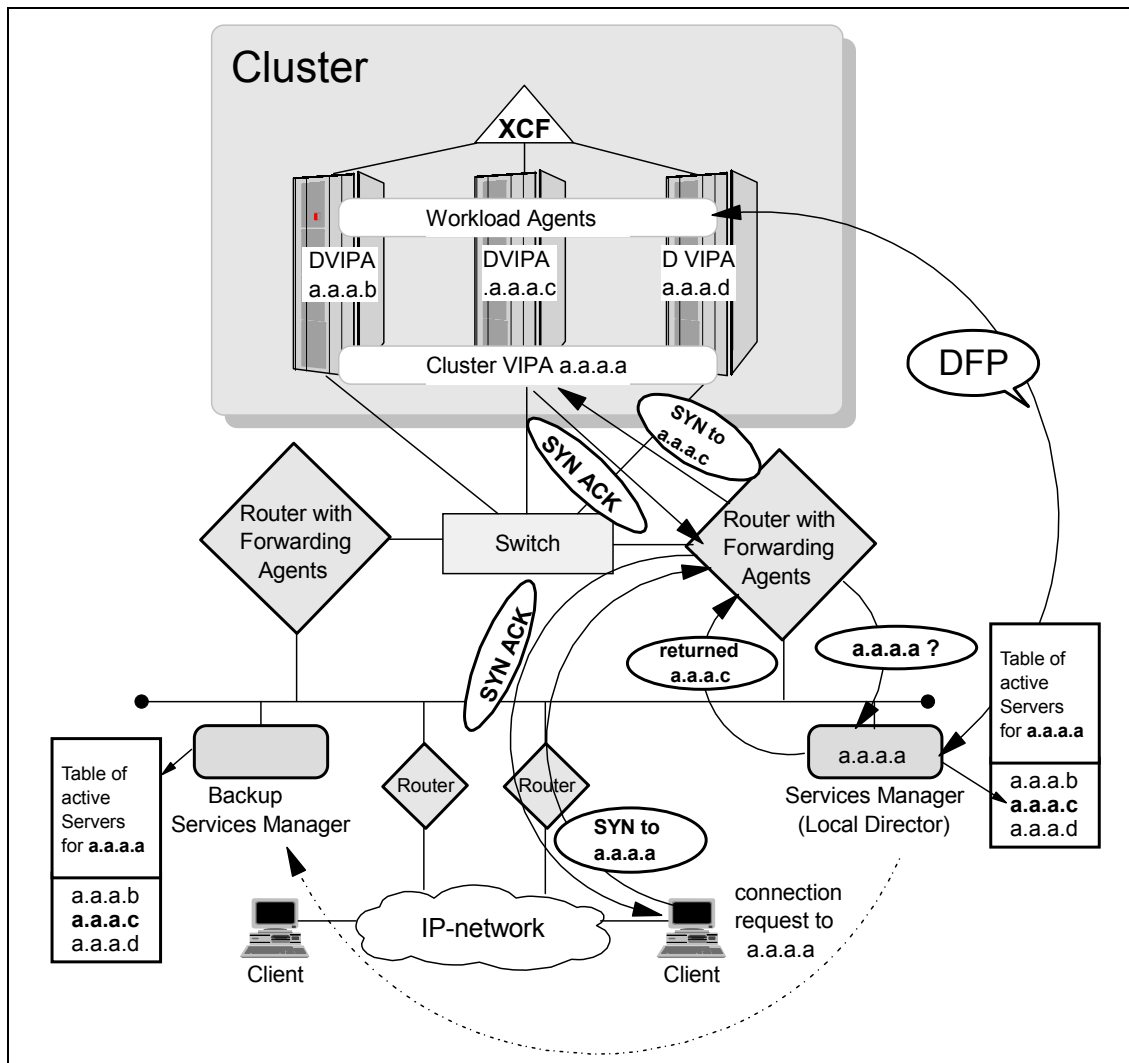


図 352 接続セットアップのフロー

注：

クライアントはアプリケーションのホスト IP アドレスは使用しませんが、代わりにクラスター VIPA アドレスを使用します。このアドレスは、クラスター内のすべてのホストの専用のグループ・アドレスです。アプリケーションの実ホスト IP アドレスは、Workload Management の情報に基づいて、Service Manager によって決定されます。

23.9.3 クライアント / サーバー接続の再始動

システム /LPAR、TCP/IP スタック、またはアプリケーションに障害が起きて、エンド・ユーザーは同じクラスター VIPA アドレスを使用して別の TCP/IP スタック上の別のアプリケーション・インスタンスに即時に再接続できます。エンド・ユーザーは、既知のアプリケーションのドメイン・ネームを再度使用できます。この名前はドメイン・ネーム・サーバーによってクラスター VIPA アドレスに変換されるからです。

ルーター中の Forwarding Agent により、再度 SYN 要求が検出され、Service Manager に対して実ホスト VIPA が求められます。

障害が起きたシステム /LPAR、TCP/IP スタック、またはアプリケーションの情報が、DFP メッセージ交換を介して Workload Manager から Service Manager に送信されます。この情報には、現在実行中のアプリケーションの状態が含まれます。これは Forwarding Agent に、負荷平準化のポリシーに基づいてご希望のアプリケーションの新規 VIPA アドレスを使用するように指示します。

残りの接続確立プロセスは、888 ページの 23.9.2 『接続とその後のデータ・フローの確立』で説明されているとおりに行われます。

23.9.4 可用性、拡張容易性、および負荷平準化の達成

以下の 2 つの部分に分けて分析します。

- 最初の部分では、シスプレックス環境中のクラスター化アプリケーション・サーバーと、OSA アダプターを介したスイッチへの接続について考慮する。
- 2 番目の部分では、Service Managers、Forwarding Agent、およびルーターなどの、イントラネットのコンポーネントについて考慮する。

23.9.4.1 シスプレックス環境

Cisco のソリューションでは、動的 VIPA は必ずしも必要ありません。アプリケーションが、ホストごとに定義された静的 VIPA のメンバーになることもできます。複数のアプリケーション・インスタンスが、同一のクラスター VIPA 中の別個のシステム /LPAR 上で開始されます。DVIPA には、起点のスタックに障害が起きたら別の TCP/IP スタックに移動できるという利点があります。

最終的にクライアント / サーバー接続用に使用される DVIPA/VIPA アドレスは、Service Manager での負荷平準化の決定結果に応じて決まります。

したがって、アプリケーション・リソースの使用状況の制御は、Service Manager によってシスプレックス環境にとって外部的に行われ、個々のシステム /LPAR 中にある Workload Agent の内部機能も関係します。

アプリケーション・サーバーの可用性は、シスプレックス中のマシンの負荷と、実行中のアプリケーション・インスタンスの数によって決まります。静的 VIPA の場合は引き継ぎや取り消しは自動的に行われません。接続の分散は外部 Service Manager によって行われるので、自動化の必要はありません。したがって、シスプレックス中のコンポーネントの作業負荷は、Service Manager によって判別されます。

シスプレックスからイントラネットへの接続に複数のバックアップを作成して、代替パスを提供することもできます。

可用性と負荷平準化は、シスプレックス中のコンポーネントと外部 Service Manager によって適切に処理されます。

業務が拡大するにつれて、拡張容易性に対する容易なソリューションを考慮する必要が生じてきます。その方法として、たとえば、現行接続に割り込まずにシスプレックス中に追加のサーバーをインストールできます。これを実行するには、シスプレックス中のすべてのシステムで動的 XCF を使用して新しいシステムを追加します。新しい DVIPA/VIPA を既存のクラスター DVIPA/VIPA に追加して、Service Manager に伝搬する必要があります。

23.9.4.2 イントラネットのコンポーネント

イントラネットのコンポーネントは以下のとおりです。

Service Manager

最も重要なコンポーネントは Service Manager で、これは LocalDirector のマシンにあります。Service Manager は、すべてのクライアント接続要求のアドレスです。このコンポーネントに障害が起きたり、ハードウェア障害（アダプターや回線など）のためにアクセスできない場合は、クライアントからアプリケーション（シスプレックス中で実行されているものなど）への接続要求は処理できません。

したがって、バックアップの Service Manager をインストールする必要があります。1 次 Service Manager に障害が起きた場合は、バックアップの Service Manager に即時に切り替えられます。高速切り替えの前提条件は、1 次 Service Manager にあるシスプレックスのコンポーネントの状態に関する情報が、バックアップの Service Manager にもあることです。つまり、1 次 Service Manager とバックアップの Service Manager との間で、作業負荷データやアプリケーション・サーバーの状態などのデータの交換が行われる必要があります。

Service Managers で扱われる IP アドレスとクラスター DVIPA/VIPA はブロードキャストされます。これらのアドレスは Forwarding Agent によって使用され、アプリケーションの DVIPA/VIPA を割り当てるための接続要求の送信先の Service Manager が判断されます。

Service Manager を 1 つのみ実装すると困難な事態になることがあります。なぜなら、特に Web サービスの要求時にボトルネックが生じることがあるからです。Web サービスでは HTTP が使用されますが、HTTP では Web サーフィン中に多数の TCP 接続が使用されます。したがって、アプリケーション・サーバーへの即時クライアント・アクセスの点で、外部 Service Managers の可用性が困難になることがあります。Forwarding Agent で接続要求および関連した DVIPA/VIPA を繰り返すことによるキャッシングにより、Service Manager の負荷を減らすことができます。

拡張容易性が必要な場合は、Service Manager 機能のある追加のマシンをイントラネットにインストールすることもできます。このようにすると、Service Manager で扱われるクラスター VIPA が多すぎるという事態を避けられます。

Forwarding Agent

Forwarding Agent では接続要求の検出が行われるので、Service Manager と同じく重要です。Forwarding Agent の障害は、イントラネット中にバックアップを備えるというソリューションで解決できます。この状態の場合、希望するクラスター VIPA へのパス上の別の Forwarding Agent により、接続要求が検出されます。これにより Service Manager の IP アドレスに要求が送信されます。これはアドレス先のクラスター VIPA を扱います。

アプリケーション・サーバーへの即時クライアント・アクセスの点で、外部 Forwarding Agent の可用性も困難になります。

Forwarding Agent のある追加のルーターをイントラネットにインストールすると、拡張容易性を容易に実現できます。このようにしても、現在稼働中の接続には影響はありません。Forwarding Agent へのパスの過負荷を避けるために、クライアントから Forwarding Agent への並列パスを実装する方法を知っている必要があります。このパスは、現行のデータグラム・トラフィックにも使用されます。この方法についてはヒントしか記述していません。この問題は本書の対象ではないからです。

23.10 IBM Sysplex Distributor および Cisco MNLB

IP アプリケーションの爆発的な成長と近い将来の予測に基づき、また従来の OS/390 や z/OS によって制御されているトランザクションやデータベースへのアクセスも考慮に入れて、IBM 社と Cisco 社は協力して、“高可用性 Web サービス”というソリューションについて研究しました。これは z/OS V1R2 で利用可能になることを想定しています。このソリューションには、IBM シスプレックス・サーバーおよびそのすべての動的機能と、Cisco 社の MultiNode Load Balancing (MNLB) 機能との間で協働するように拡張および適応された、ハードウェアとソフトウェアのパッケージが備えられています。

このソリューションには、以下のような利点があります。

- Sysplex Distributor を介したインバウンド・トラフィック・フローを避ける。
- 負荷平準化の情報を遅れずに得られる。
- ポリシー情報やサービス品質情報を使用して、“最善な”サーバーを選択する。
- シスプレックス・トラフィック用に LocalDirector をインストールする必要がない。

23.10.1 何を意味するか

IBM Sysplex Distributor は負荷平準化を備えています。Cisco MNLB はルーティングを備えています。

- Sysplex Distributor は、Cisco MNLB の Cisco Services Manager の機能を受け取る。
 - これは、個々のシステムの WLM 情報、サービス品質 (QoS) データ、またはポリシー・エージェント (PAGENT) によって備えられるポリシー情報に基づいて、該当する“最善の”サーバーを選択します。Sysplex Distributor ではクラスター中の WLM によって備えられるデータに基づいて負荷平準化が行われるので、Dynamic Feedback Protocol (DFP) を使用する必要はなくなります。
 - これには、スイッチ (Cisco CAT 6500 など) の MNLB 機能である、Forwarding Agent に対する接続情報が備えられています。この情報は、CASA (Cisco Appliance Services Architecture) という専用の Cisco プロトコルを介して転送されます。
- その後スイッチ (またはルーター) では、この情報を使用して、クライアント / サーバー・データがシスプレックス中の選択されたサーバーに直接転送されるので、すべてのインバウンド・トラフィックが LocalDirector や the Sysplex Distributor などの単一点を通過するのを避けることができる。
 - 既存の MNLB Forwarding Agent が使用されます。Sysplex Distributor や相互カップリング・リンクを介したインバウンド・トラフィックのパスは使用されなくなります。スイッチではサーバーに直接転送する方法が使用されます (OSA Express アダプターを使用するなど)。OSA Express アダプターの共用については後述されます。

注：

Sysplex Distributor は、OS/390 V2.10 および新機能と並行使用できるように設計されています。したがって、Cisco Services Manager 機能と共に使用する DVIPA がある一方で、旧 V2.10 の作業負荷分散ソリューションで使用する DVIPA も TCP/IP プロファイル内に定義できます。

23.10.2 IBM Sysplex Distributor と Service Manager の概要

Cisco 独自の MNLB 構成において、Service Manager には負荷平準化を分散するアルゴリズムが備えられています。負荷平準化処理の作業負荷データを入力するために、Services Manager は、OS/390 の下のアドレス・スペースである、個々のサーバー内の Cisco Workload Agent と通信します。このプログラムにより、OS/390 WLM 中の情報が検索されます。Services Manager と Workload Agent との間の通信は、Dynamic Feedback Protocol (DFP) を使用して行われます。

負荷平準化処理の後で、Services Manager により、サーバー・クラスターから該当するアプリケーション・サーバーが選択されます。最後に、接続要求の転送先の実サーバー IP アドレスが、Forwarding Agent に指示されます。Services Manager と Forwarding Agent との間でこの情報を交換する際には、Cisco Appliance Services Architecture (CASA) プロトコルが使用されます。

IBM Cisco ソリューションでは、Sysplex Distributor で Cisco Services Manager の機能を使用できるようになりました。Sysplex Distributor では前のリリースと同じテクノロジーが使用されています。したがって、クラスター・アドレスとして分散 DVIPA が使用されます。また、OS/390 Workload Manager (WLM) の重み情報に基づいて、" 最善の " アプリケーション・サーバーが選択されます。ただし、この選択処理は、QoS や、ポリシー・エージェントのデータベース (LDAP サーバーや個人用のデータベースなど) に定義されているポリシーに関する情報を使用するように拡張されています。

Sysplex Distributor とスイッチまたはルーター中の Forwarding Agent との間の通信の際にも、CASA プロトコルを使用して Forwarding Agent との間でデータが交換されます。Sysplex Distributor から Forwarding Agent に、接続要求によって有効なアプリケーション・サーバーの実 IP アドレスが指示されます (896 ページの 23.10.5 『接続確立プロセス』を参照)。

Sysplex Distributor で保守されているクラスター・サーバーの IP アドレスとポートを Forwarding Agent に通知するには、最初の接続要求を解決する前に装置活動化プロセスを開始する必要があります。そのためには、CASA パケットと情報を Forwarding Agent に送信します。たとえば、以下のような情報を送信します。

- 宛先 IP アドレスとポートの DVIPA。これはクラスター・アドレスを定義します。
- インタレスト・アドレス。これは Forwarding Agent で使用される CASH パケットのアドレスを定義する動的 XCF IP アドレスとポートです。

Forwarding Agent は、この情報によって宛先 DVIPA と指定ポートを含むパケットを監視し、そのパケットが接続要求の場合は、指定された動的 XCF IP アドレスのクラスター・アドレスについて、実サーバー IP アドレスが Sysplex Distributor から入手されます。

23.10.3 Cisco Forwarding Agent: 概要と機能

Forwarding Agent は、887 ページの 23.9.1 『MultiNode Load Balancing 機能の概要』で説明されている働きをします。

これにより、すべての着呼接続要求が検出され、Sysplex Distributor 中の実サーバーの IP アドレスが要求され、接続要求とそれ以降のすべてのデータグラムが実サーバーに直接ルーティングされます。

Forwarding Agent によりすべての接続に関する類縁性レコードが作成され、このレコードは Services Manager から実サーバー IP アドレスが戻される際に更新されます。

23.10.4 Cisco Workload Agent

これはアプリケーション・サーバーのシステム内にあります。Workload Agent は作業負荷と可用性に関するフィードバックを Services Manager に提供します。Workload Agent では、DFP プロトコルを使用して、Services Manager に作業負荷情報を提供します。Cisco OS/390 Workload Agent は、Workload Agent のインプリメンテーションです。これは、OS/390 アドレス・スペース内で実行するプログラムで、IBM OS/390 Workload Manager (WLM) 中の作業負荷情報を検索します。

シスプレックス構成では、Sysplex Distributor と個々のシステム /LPAR の IBM Workload Managers の間で直接通信されます。したがって、Cisco OS/390 Workload Agent は必須ではありません。

23.10.5 接続確立プロセス

TCP 接続確立プロセスの概要について以下に説明します。899 ページの図 353 には、関係が図示されています。

1. クライアントから、サーバー (TN3270E サーバー、FTP サーバー、Web サーバーなど) にログインして、TCP 接続要求が開始されます。クライアントは、ホスト名として、実アプリケーション・サーバーではなく、シス

プレックス・クラスター中のサーバー・グループまたはクラスター名を選択します。ドメイン・ネーム・サーバーによって、ホスト名がシスプレックス・クラスターの IP アドレスに変換されます。

2. 接続要求が IP ネットに伝送されます。これは TCP SYN 要求です。最初のルーター (クライアントのデフォルト・ルーター) のルーティング・テーブル内で、シスプレックス・クラスターの IP アドレスが検索されます。このアドレスは、要求されたシスプレックス・クラスターを担当する Forwarding Agent のテーブル内では、仮想 IP アドレスとして定義されます。このシスプレックス・クラスター IP アドレス (動的仮想 IP アドレス (DVIPA)) を、Forwarding Agent のテーブルに登録し、ネットワークに伝搬する方法については後述します。
3. Forwarding Agent で IP パケットが受信されると、これらのパケットの内容が活用されます。接続要求 (TCP ヘッダー中の SYN ビットでマークされている) が受信されると、CASA 要求が Services Manager (IBM Sysplex Distributor 内にある) に送信され、シスプレックス中のアプリケーション・サーバーの実 IP アドレスが要求されます。
4. 接続確立プロセスの最初の部分として、Forwarding Agent によって、ソース IP アドレス / ポート - 宛先 IP アドレス / ポートの移送プロトコル類縁性が作成されます。これはキャッシュ・テーブル中で、要求された接続の項目になります。この類縁性キャッシュの目的については後述されます。
5. Sysplex Distributor 中の Services Manager で、WLM、QoS、またはポリシー・エージェントの情報をを使用してシスプレックス中の "最善な" アプリケーション・サーバーが選択され、CASA 応答中にクライアント / サーバー接続の実 IP アドレスが戻されます。

この実 IP アドレスは、動的に作成される IP アドレスです。このアドレスは、TCP/IP スタックが起動され、シスプレックス中の他の TCP/IP スタックへの接続が開始される際に作成されます。また Services Manager で、前述のソース - 宛先の類縁性が作成され、TCP 接続が制御されます。たとえば、接続ルーティング・テーブルの表示など、コンソール・ディスプレイに関する情報が提供されます。
6. CASA 応答の受信後に、Forwarding Agent の類縁性キャッシュが実サーバー・アドレスに更新され、接続要求パケットがアプリケーション・サーバー中の TCP/IP スタックに転送されます。
7. アプリケーション・サーバー・システム / LPAR 上の TCP/IP スタックにより、IP と TCP ヘッダーが有効かどうかを検査されます。すべて有効な場合は、TCP により、受信した SYN 要求に関する確認通知 (ACK) が戻され、SYN 要求がクライアントに送信されて全二重接続が確立されます。
8. Forwarding Agent で、直前の接続要求の SYN/ACK がクライアントに送信中であることが検出されると、実行中の接続の情報で類縁性レコードが更新されます。他の Forwarding Agent へのマルチキャストのために、類縁

性キャッシュが記録されます。クライアント / サーバー双方間のパス上で、他の Forwarding Agent を使用することもできます。

参照されているアプリケーション・サーバーの IP パケットが別の Forwarding Agent で受信されると、その Forwarding Agent で既存のソース - 宛先類縁性の類縁性テーブルが検査されます。合致するレコードが検出されると、クライアント・パケットは、すでに接続しているが、現在は別の Forwarding Agent を介して並列パスを使用している実サーバー IP アドレスを指すようにされます。

9. SYN/ACK を受信した Forwarding Agent から、Sysplex Distributor 中の Services Manager に、ソース - 宛先接続情報も通知されます。これは、接続ルーティング・テーブルなどの、z/OS コンソール・ディスプレイのソースでもあります。
10. クライアントで SYN/ACK が受信されると、接続確立プロセスは終了し、最初のアプリケーション・データを送信できます。

これまで説明してきた接続確立プロセスでは、簡単な概要にとどめるために、セキュリティのためのプロトコルについては意図的に触れていません。

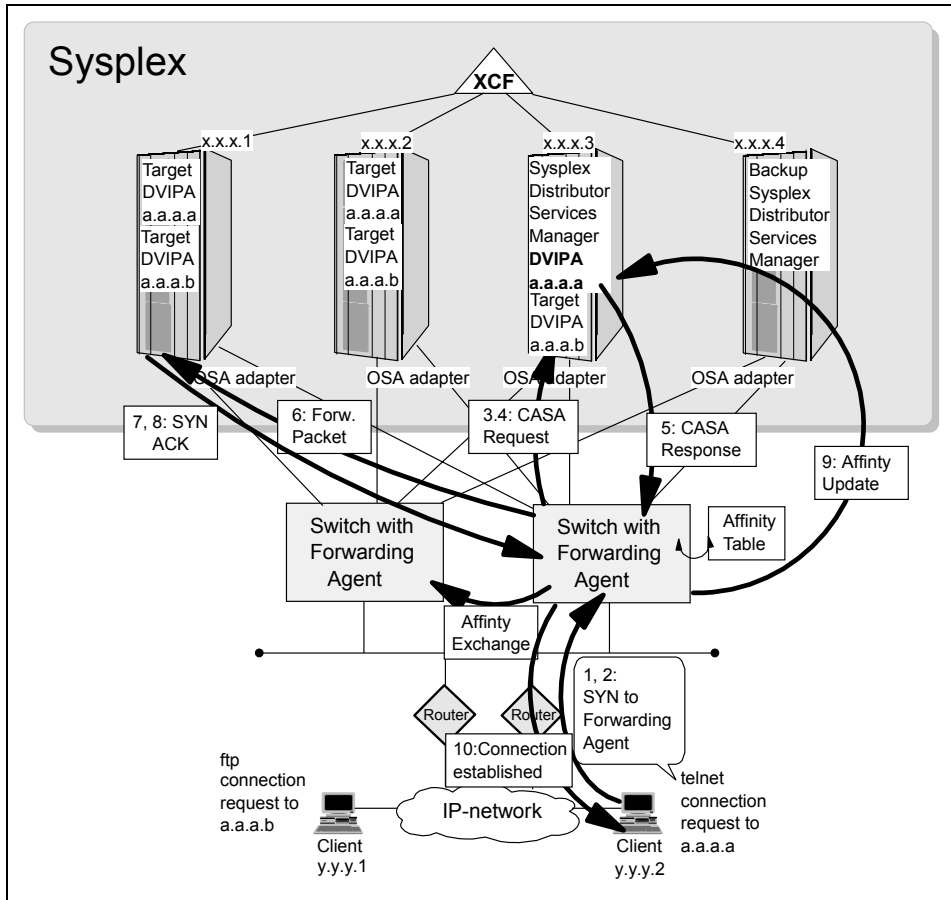


図 353 Telnet 接続の確立

図 353 には、クライアントから Telnet サーバー・グループ a.a.a.a への Telnet 接続が示されています。この接続に際して選択された Sysplex Distributor の実サーバー IP アドレスは x.x.x.2 です。負荷平準化プロセスに基づいて、別のクライアント接続を x.x.x.1 にルーティングできます。

他のクライアントから実サーバー x.x.x.1 への接続があります。

23.10.6 スタック、サーバー、または LPAR の障害

アプリケーション・サーバー、TCP/IP スタック、またはシステム /LPAR に障害が起きますと、実 IP アドレスは使用できず、クライアント接続は失われます。この障害は、WLM 情報によって Sysplex Distributor に通知されます。同じクラスター・アドレス (たとえば、Telnet の場合は a.a.a.a) を使用して、ク

クライアントから即時に新しく再接続を開始できます。Forwarding Agent により、Sysplex Distributor に対する新しい CASA 要求がセットアップされます。Sysplex Distributor により、別の TCP/IP スタック上で代替の実行中のアプリケーション・インスタンスが検索され、別の実 IP アドレス (x.x.x.1 など) が Forwarding Agent に戻されます。接続確立プロセスが前述のように続行されます。

23.10.7 Sysplex Distributor の障害

Sysplex Distributor に障害が起きると、その 1 次 Sysplex Distributor の作業はすべてバックアップ Sysplex Distributor に引き継がれます。相互カップリング・ファシリティーにより、障害が起きたアプリケーションの WLM か TCP/IP スタックを介して、バックアップ Sysplex Distributor に情報が提供されます。システム全体がダウンした場合は、バックアップ Sysplex Distributor の WLMPOLL はタイムアウトになります。これは引き継ぎプロセスのシグナルです。860 ページの 23.4.6 『DVIPA アドレスの引き継ぎ / 取り消し』の、引き継ぎと取り消しのプロセスに関する詳細情報を参照してください。

23.10.8 パケットのルーティング

インバウンド・パケットのルーティングは、Forwarding Agent (Cisco スイッチにインストールされている) によって行われます。インバウンド・パケットには、アプリケーション・サーバーのクラスター IP アドレスがあります。Forwarding Agent によりパケットの内容が参照され、その情報が類縁性キャッシュと比較されます。合致した項目が検出された場合は、実行中の接続があると認識されます。パケットは、関連したアプリケーション・サーバーの実 IP アドレスにルーティングされます。

アウトバウンド・パケットは、スイッチかルーターを介してイントラネットに送信されます。OSPF 負荷平準化機構を使用して、コストの等しい並列パスを介して負荷を分散できます。

類縁性キャッシュ中の情報は、ソース IP アドレス / ポート、宛先 IP アドレス / ポート、および実 IP アドレスで構成されます。

図 354 には、前述の接続例を基にした類縁性キャッシュの内容が示されています。

Affinity Table						internal connection information
Source Address	Port	Dest. Address	Port	Prot		
y.y.y.2	4213	a.a.a.a	23	TCP		real server y.y.y.2
a.a.a.a	23	y.y.y.2	4213	TCP		y.y.y.2
y.y.y.1	4178	a.a.a.b	20	TCP		y.y.y.1
y.y.y.1	4178	a.a.a.b	21	TCP		y.y.y.1
a.a.a.b	20	y.y.y.1	4178	TCP		y.y.y.1
a.a.a.b	21	y.y.y.1	4178	TCP		y.y.y.1

図 354 類縁性テーブル・キャッシュ

23.10.9 MNLB コンポーネントの追加タスク

23.10.9.1 Services Manager

Services Manager は、実サーバー・アドレスをマップするクラスター・アドレスを指定した DVIPA で構成されます。

Services Manager では、マルチキャスト・アドレスを使用し、Internet Group Management Protocol (IGMP) IP アドレスおよびポートを使った CASA プロトコルを介して、マッピング情報を Forwarding Agents に送信します。

この初期マルチキャストには、クラスター・アドレスのサービスを提供するワイルドカード情報が含まれます (本書の例では a.a.a.a and a.a.a.b)。この情報により、Forwarding Agent によって監視されるインバウンドとアウトバウンドのトラフィックに関する、以下のようなワイルドカード・キャッシュ項目が作成されます。

Sysplex Distributor により TCP プロトコルが提供されます。

Source Address	Source Mask	Port	Dest Address	Dest Mask	Port	Prot
0.0.0.0	0.0.0.0	0	a.a.a.a	255.255.255.255	23	TCP
0.0.0.0	0.0.0.0	0	a.a.a.b	255.255.255.255	20	TCP
0.0.0.0	0.0.0.0	0	a.a.a.b	255.255.255.255	21	TCP
a.a.a.a	255.255.255.255	23	0.0.0.0	0.0.0.0	0	TCP
a.a.a.b	255.255.255.255	20	0.0.0.0	0.0.0.0	0	TCP
a.a.a.b	255.255.255.255	21	0.0.0.0	0.0.0.0	0	TCP

図 355 ワイルドカード・キャッシュ

901 ページの図 355 の最初の 3 行により、それらの行で定義されている宛先およびポート・アドレス (a.a.a.a、23 など) を持つクラスター・アドレス用

の IP アドレスおよびソース・ポートからの着信クライアント・パケットが、Forwarding Agent で受け入れられることが決まります。

4 ～ 6 行目には、使用できるアウトバウンド・アドレスの組み合わせが記述されています。

すべての情報は Forwarding Agent で使用され、すべての IP パケットの内容検査が行われます。

23.10.9.2 バックアップ Services Manager

CASA アーキテクチャー全体が Services Manager に従属します。Services Manager に障害が起きると、接続のセットアップに関するクライアント要求を実行できます。したがって、1 つ以上のバックアップ Services Managers を実装することが非常に重要になります。

バックアップ Services Manager は、ワイルドカード類縁性を介して、フロー情報を周期的に Forwarding Agent に登録します。Services Manager に障害が起きると、Forwarding Agent では、優先順位の選択に基づいて、次に高いバックアップ優先順位フローがあるバックアップ Services Manager が自動的に選択されます。

23.10.9.3 Forwarding Agent

Forwarding Agent は、IP パケットのルーティングを行います。ルーティングは、Services Manager の情報に基づいて行われます。Services Manager から受信したワイルドカードを介して、実行する機能が判別されます。したがって、Forwarding Agent では IP パケットを参照して、ソースと宛先の IP アドレスと、受信したワイルドカードを介して Services Manager から Forwarding Agent に通知されたポートを調べます。

したがって、Forwarding Agent で、インバウンドとアウトバウンドのトラフィックの IP ヘッダーがすべて検査され、新しい接続要求が着信しているかどうかや、すでに登録されているかどうか調べられます。登録すると、類縁性キャッシュ中に項目が作成されます。登録されていない場合は、“最善の”ターゲット・サーバーの IP アドレスを提供するよう Services Manager に指示されます。このアドレスがキャッシュ中の類縁性項目のうち 1 つの情報と合致する場合は、接続はすでに存在しています。

Forwarding Agent では、Services Manager と同じマルチキャスト IGMP アドレスを介して、ワイルドカード情報を受信できます。したがって、CASA プロトコルを実行している Sysplex Distributor は、Forwarding Agent と同じマルチキャスト IGMP アドレスとポートで構成されている必要があります。

23.11 OS/390 DNS/WLM

非常に複雑な Sysplex Distributor/MNLB ソリューションに加えて、OS/390 ドメイン・ネーム・サーバー (DNS) を使用して、シスプレックス環境中の複数のアプリケーション・サーバー・インスタンスに負荷平準化された接続を分散する単純な方法もあります。この方法の場合は、シスプレックス中で OS/390 BIND Domain Name System (DNS) を実行します。¹ これは OS/390 Workload Manager (WLM) と協働します。この場合、WLM をゴール・モードで実行する必要があります。このモードでは、システム /LPAR 上の作業負荷の状態に関する重み値が提供されます。

BIND DNS の詳細については、291 ページの第 8 章『ディレクトリーおよびネーミングのプロトコル』を参照してください。

23.11.1 シスプレックス環境中の DNS

23.11.1.1 理由

ドメイン・ネームは、階層ネーム・スペース構造で構成されています。たとえば、以下のようにになっています。

```
mvs1.itso.raleigh.ibm.com
```

このドメイン・ネームは、*mvs1* という LPAR を指します。この LPAR は、ローリーにある IBM 社の ITSO 部門にあり、高水準の修飾子 *com* が付けられています。

クライアントで *mvs1* マシンとの Telnet 3270 接続を模倣したい場合は、クライアントの PC で以下のように入力します。

```
tn3270 mvs1
```

ネーム解決の場合、PC での解決によりドメイン *itso.raleigh.ibm.com* が追加され、以下のアドレスの IP アドレスを入手して解決要求がドメイン・ネーム・サーバーに送信されます。

```
tn3270 mvs1.itso.raleigh.ibm.com
```

アプリケーションが使用できない場合や、*mvs1* がダウンしている場合は、クライアントが希望するサービスにアクセスすることはできなくなります。クライアントは、たとえば *tn3270 mvs2* を使用して、別のマシン上でのアプリケーション *tn3270* の検索を試行することもできます。しかしながら、実行中のアプリケーションを見つけるまでに多数のシステムを検索する必要がある場合、この方法は煩雑なものになる可能性があります。

¹ OS/390 Berkeley Internet Name Domain (BIND) Domain Name System (DNS) は、UNIX ネーム・サーバーです。

代替方法として、クラスター・アドレスを使用し、ドメイン・ネーム・サーバーにより、クラスター・アドレス tn3270 が定義されたマシン上で実行中のアプリケーション・インスタンスを検索することができます。

23.11.1.2 ソリューション

ドメイン・ネーム・サーバーに、クラスター中の最初のサーバーを選択し、マシンの負荷を調べる能力がある場合、そのネーム・サーバーからクライアントに、最も負荷が少ない希望するサーバーの IP アドレスが戻されます。この場合、tn3270 との TCP 接続中の応答時間が短くなるので、クライアントにとって好都合です。この方法は、DNS/WLM ソリューションを使用して行われます。

しかしながら、2 つの新しい用語について考慮する必要があります。

1. 新しい用語の 1 つのクラスター・アドレスを実装する必要があります。たとえば、tn3270grp などです。このグループには、1 つのシスプレックス中の、同じアプリケーション・プログラムを実行するすべてのアプリケーションが含まれます (tn3270 サーバーなど)。
2. クラスター・アドレスは、もう 1 つの新しい用語のシスプレックス・サブドメインに属します。たとえば、tn3270grp.mvsplex などです。

クライアントでは完全修飾のドメイン・ネームを使用して、ネーム・サーバーにアクセスします。たとえば、mvsplex というシスプレックス中のグループ tn3270grp の tn3270 サーバーにアクセスするには、以下のように入力します。

```
tn3270 tn3270grp.mvsplex.itso.raleigh.ibm.com
```

こうすると、itso.raleigh.ibm.com ドメインに属している指定されたシスプレックス *mvsplex* 中で実行しているマシンのクラスターから、アプリケーション tn3270 が判別されます。

23.11.1.3 構成のサンプル

以下の図は、3 つのシステム /LPAR 上で 3 つの TCP/IP スタックを実行していることを示しています。スタック TCP1 と TCP3 で TN3270 サーバーが実行されています。どちらのアプリケーションともクラスター TN3270GRP に属しています。また FTPGRP もありますが、これは FTP サーバーのクラスターです。WLM から DNS に作業負荷情報が提供されます。

1 次ネーム・サーバーと、バックアップ用の 2 次ネーム・サーバーの、2 つの DNS がインストールされています。どちらも、Workload Manager と通信して、負荷平準化の決定に使用する重み情報を入手します。

負荷平準化の決定は、TCP 接続の確立時のみ行われます。接続をセットアップすると、クライアント / サーバー接続はクライアントで接続が終了されるまで存続します。

アプリケーション・サーバーが TCP/IP スタックの障害や、システム異常終了のために、接続が失敗すると、クライアントで別のシステム上のアプリケーションを入手して、ネーム・サーバーを介して再接続できます。1 次ネーム・サーバーに障害が起きたら、2 次ネーム・サーバーにより、ご希望のアプリケーションの負荷平準化された新しい IP アドレスが備えられます。

DNS/WLM ネーム・サーバー (1 次および 2 次) を、OS/390 システムにインストールすることは重要です。DNS/WLM コードを実行する機能はそれらのシステムにしかないからです。

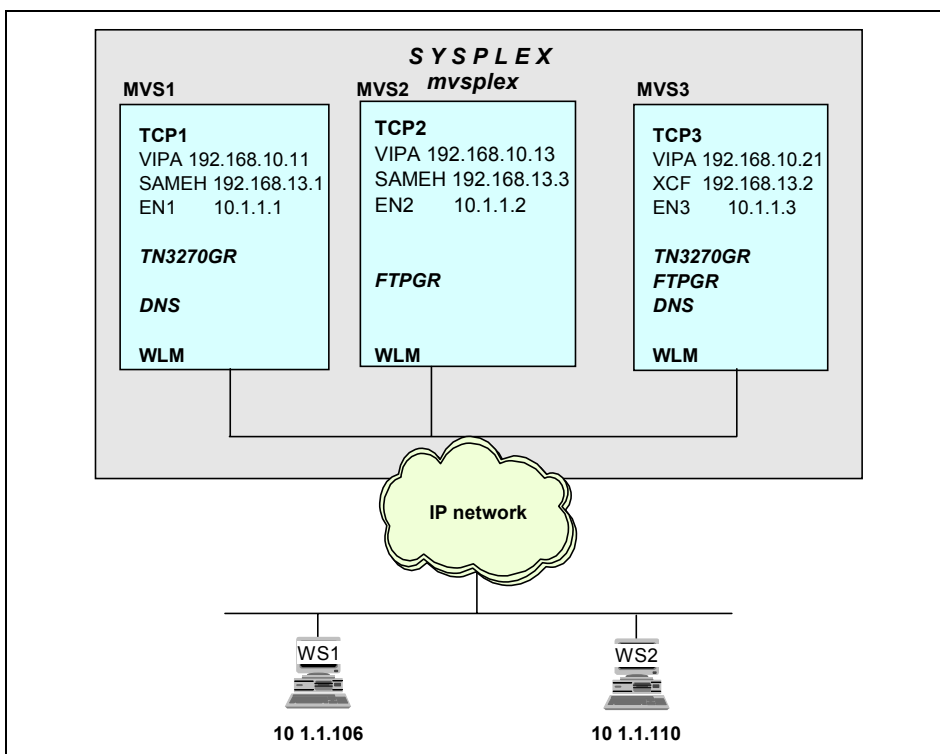


図 356 DNS/WLM ソリューション

23.11.1.4 アプリケーションのネーム・サーバーへの登録のプロセス

アプリケーションをネーム・サーバーに登録するには、以下のステップに従ってください。

1. アプリケーションを開始すると、WLM で登録されます。その結果、WLM でアプリケーションの存続期間、状態、および負荷を制御できます。
登録後に WLM で以下のことが認識されます。
 - グループ名、サーバー名、ホスト名、IP アドレス
 - CPU 負荷、使用可能なストレージ、入出力率など
2. WLM と他のシステムとの間で、相互カップリング・ファシリティーを介してこの情報が交換されます。
3. 個々の WLM に、シスプレックス中のすべてのシステムのリソースと負荷のイメージがあります。
4. 定義に従って、1 分から 5 分までの間、毎分 DNS で、DNS に接続しているすべての WLM の重みの図を更新したものが受信されます (1 つの TCP/IP スタックを介して)。その結果、DNS ではアクティブなアプリケーションと、その DNS サーバー名および IP アドレスが認識されます。転送テーブルの項目が動的に定義されます。908 ページの図 359 を参照してください。
5. この更新された重み情報に基づいて、負荷平準化プロセスにより、着信 TCP 接続要求に関する“最善の”IP アドレスが選択されます。
6. クライアントの解決要求と、グループ名を含む完全修飾ドメイン・ネームが DNS に着信します。DNS で、WLM から受信した最終更新内容に基づいて、“最善の”アプリケーション・サーバーが選択され、このサーバーの IP アドレスがクライアントに戻されます。
7. TCP 接続要求がクライアントからこの戻された IP アドレスに直接送信され、接続セットアップが続行されます。

図 357 には、上記のフローが示されています。

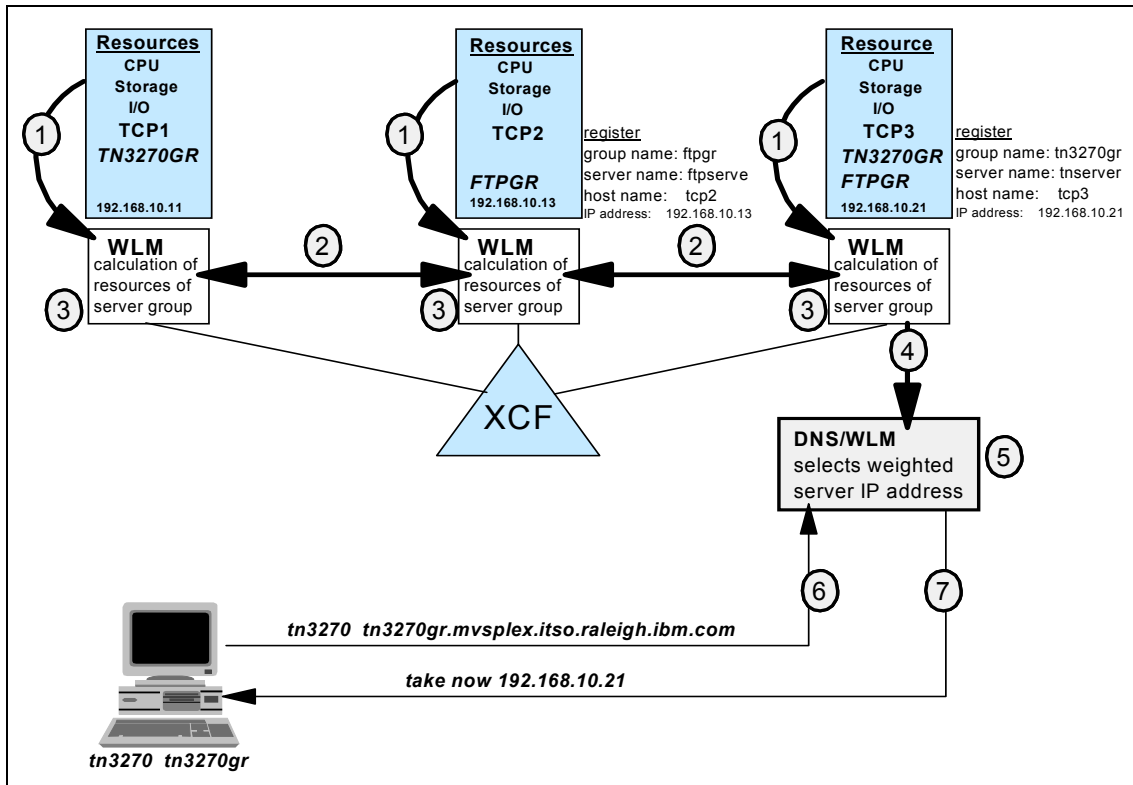


図 357 アプリケーション登録と解決要求

23.11.2 DNS/WLM とリモート・ネーム・サーバー

DNS/WLM の実装に関する決定をする場合には、この OS/390 ネーム・サーバー・システムを、UNIX インストールに基づく既存のネーム・サーバー・システムと統合する必要があります。この方法は簡単です。シスプレックス中のサーバー・グループに対する項目として、1 次 DNS/WLM を指す項目を、リモート・ネーム・サーバーに追加する必要があります。1 次 DNS/WLM への接続に障害が起きた場合には、2 次項目が 2 次ネーム・サーバーを指している必要があります。例については、908 ページの図 358 を参照してください。

```
mvsplex NS tcp1.mvsplex.itso.raleigh.ibm.com
mvsplex NS tcp3.mvsplex.itso.raleigh.ibm.com
```

図 358 リモート・ネーム・サーバー上の新規項目

リモート・ネーム・サーバーに解決要求が着信すると、このサーバーで新しいサブドメイン *mvsplex* が認識されます。このサーバーでは名前を解決できない (このサブドメイン項目の \$origin が無い) ので、この要求を別のネーム・サーバー *tcp1.mvsplex.itso.ibm.com* に宛て、名前が解決されるようになります。tcp1 に障害が起きると、2 次ネーム・サーバーの tcp3 に連絡されます。

tcp1 と tcp3 のどちらのシステムの場合も、DNS/WLM ネーム・サーバー中でこれらのシステムの IP アドレスを指すには、ホスト項目も必要です。

1 次および 2 次ネーム・サーバーには、mvsplex に関連付けられたクラスター名と従属の IP アドレスに関する項目が必要です。図 359 の例を参照してください。

CNAME はサブドメインに関連付けられている正規名です。クラスター名にもホスト項目 (A という項目) が必要です。これらの項目は登録プロセスの実行時に動的に作成されます。

```
$ORIGIN mvsplex.itso.raleigh.ibm.com

                IN NS tcp1
                IN NS tcp3

tcp1             IN A   192.168.10.11
tcp3             IN A   192.168.10.21

tn3270gr        IN cname tn3270gr.mvsplex.itso.raleigh.ibm.com
ftpgr           IN cname ftpgr.mvsplex.itso.raleigh.ibm.com

----- dynamically created entries by the -----
----- name server after WLM registration -----

tn3270          IN A   192.168.10.11
                IN A   192.168.10.21
ftpgr           IN A   192.168.10.13
                IN A   192.168.10.21
```

図 359 動的に作成されたホストのクラスター項目

23.12 Virtual Router Redundancy Protocol (VRRP)

Virtual Router Redundancy Protocol (VRRP) は、1998 年 4 月に IBM、Ascend Communications、Microsoft、および Digital Equipment Corporation の各社によって IETF に提出され、RFC 番号 2338 として文書化されています。この状態は提案標準です。

23.12.1 概要

ホスト IP 構成では、静的に構成されたデフォルト経路を使用することが非常に一般的です。この方法を使用すると、エンド・ホストでの構成と処理のオーバーヘッドを最小限にとどめることができます。またこの方法は、すべての IP インプリメンテーションで仮想的にサポートされています。DHCP (131 ページの 3.7 『Dynamic Host Configuration Protocol (DHCP)』を参照) などの動的ホスト構成プロトコルは、この操作モードでデプロイされるはずですが、この場合、通常はエンド・ホスト IP アドレスとデフォルト・ゲートウェイの構成を備えています。ただし、この構成では単一の障害が作成されます。デフォルト・ルーターがないと、使用可能な代替パスを検出できないすべてのエンド・ホストが分離され、重大な問題が発生します。

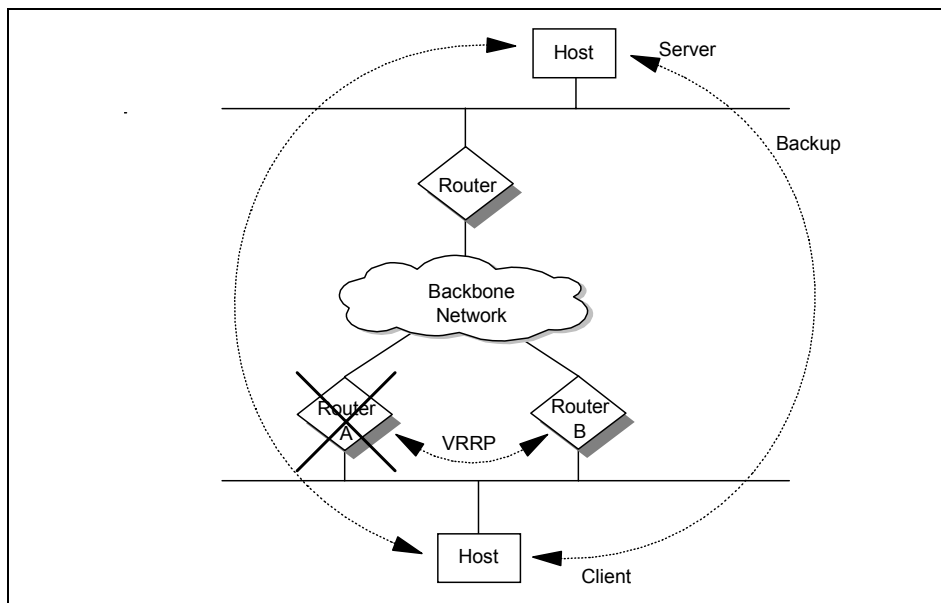


図 360 VRRP の図

VRRP は、静的なデフォルト経路環境に固有の、単一の障害を除去するように設計されています。VRRP は、LAN 上の VRRP ルーターの 1 つに仮想ルーターの役割を動的に割り当てる選択プロトコルを指定します。仮想ルーター

に関連付けられた IP アドレスを制御する VRRP ルーターのことをマスターと呼び、マスターはこれらの IP アドレスに送信されたパケットを転送します。マスターが使用不能になった場合、選択プロセスによって、転送の役割に関する動的フェイルオーバーが備えられます。それからエンド・ホストで、LAN 上の仮想ルーターの IP アドレスをデフォルトの先頭ホップ・ルーターとして使用できます。VRRP を使用する場合の利点としては、すべてのエンド・ホストで動的ルーティングやルーター検出プロトコル (105 ページの 3.2 『Internet Control Message Protocol (ICMP)』中のルーター検出プロトコルを参照) を構成しなくても、デフォルト・パスの高可用性を実現できることがあります。

23.12.2 VRRP の定義

VRRP で使用される用語の一部を以下に示します。

VRRP ルーター	Virtual Router Redundancy Protocol を実行するルーター。1 つ以上の仮想ルーターに関係することができます。
仮想ルーター	VRRP によって管理される抽象オブジェクトで、共用 LAN 上でホストのデフォルト・ルーターの働きをします。これは共通 LAN 全体において、定義に応じて、仮想ルーター ID と、関連した IP アドレスの集合で構成されます。1 つの VRRP ルーターを 1 つ以上の仮想ルーターのバックアップにすることができます。
IP アドレス所有者	仮想ルーターの IP アドレスが実インターフェース・アドレスになっている VRRP ルーター。ICMP PING や TCP 接続などの場合、このルーターは、起動時に、これらの IP アドレスの 1 つにアドレッシングされているパケットに応答します。
1 次 IP アドレス	実インターフェース・アドレスの集合から選択された IP アドレス。選択アルゴリズムの 1 つとして、常に最初のアドレスを選択することもできます。VRRP 通知が送信される際には、常に 1 次 IP アドレスが IP パケットのソースとして使用されます。
仮想ルーター・マスター	仮想ルーターに関連付けられた IP アドレスに送信されたパケットを転送し、これらの IP アドレスに関する ARP 要求に応答することが想定されている VRRP ルーター。IP アドレス所有者が使用可能な場合は、これが常にマスターになることに注意してください。

仮想ルーター・バックアップ

現行のマスターに障害が起きた場合に、仮想ルーターに関する転送の役割が想定されている VRRP ルーターの集合。

23.12.3 VRRP の概要

VRRP は、前述の仮想ルーター機能を備えている選択プロトコルを指定します。プロトコル・メッセージングは、すべて IP マルチキャスト・データグラム (237 ページの第 6 章『IP マルチキャスト』を参照) を使用して実行されるので、IP マルチキャストをサポートしているさまざまなマルチアクセス LAN テクノロジーでこのプロトコルを稼働できます。個々の VRRP 仮想ルーターには、1 つの MAC アドレスが事前に割り当てられています。仮想ルーター MAC アドレスは、マスター・ルーターによって周期的に送信されるすべての VRRP メッセージ中でソースとして使用されるので、拡張 LAN 内でブリッジを認識できます。

仮想ルーターは仮想ルーター ID (VRID) と IP アドレスの集合によって定義されます。仮想ルーターを VRRP ルーターのインターフェース上の実アドレスと関連付けたり、追加の仮想ルーター・マッピングや仮想ルーターのバックアップになる優先順位を指定して VRRP ルーターを構成したりできます。VRID とアドレスの間のマッピングは、LAN 上のすべての VRRP ルーター間で整合性がなければなりません。しかしながら、さまざまな LAN 上のさまざまなアドレス・マッピングで VRID を再利用することに対する制限はありません。

個々の仮想ルーターの有効範囲は、単一の LAN に制限されます。ネットワーク・トラフィックを最小限にするために、個々の仮想ルーターのマスターのみで周期的な VRRP 通知メッセージが送信されます。優先順位の低いバックアップ・ルーターが、マスターを優先使用することはありません。これによって、現行より優先順位の高いパスが使用可能にならない限り、サービスが中断することはなくなります。優先使用しようとする試みをすべて禁止するように管理することもできます。ただし例外が 1 つのみあり、それは VRRP ルーターが所有するアドレスに関連付けられた仮想ルーターのマスターは、常にその VRRP ルーターであるということです。マスターが使用できなくなると、短時間の遅延の後で、最も優先順位の高いバックアップがマスターに遷移するので、サービスの中断を最小限にとどめて仮想ルーターの役割の遷移を制御できます。

VRRP プロトコルの設計には、マスターからバックアップへの即時遷移が備えられており、サービスの中断が最小限にとどめられていることに加えて、最適化も組み込まれています。これにより、プロトコルが複雑にならないようにされる一方で、代表的な操作シナリオでのマスター遷移の制御が保証されます。最適化により、選択プロトコルの実行時の状態に関する要件は最小

限にとどめられ、アクティブ・プロトコルの状態も最小限で済み、メッセージ・タイプと送信側は1つで済みます。代表的な操作シナリオは、2つの冗長ルーター、ルーターごとに異なるパス優先順位、またはこれらの両方として定義されています。この前提事項に違反している場合（たとえば、3つ以上の冗長パスの優先順位がすべて同じ場合など）の副次作用としては、マスター選択の際に短時間で重複パケットを転送できることがあります。しかしながら、大部分のデプロイメントは代表的なシナリオの前提事項に該当すると考えられ、この場合にはマスター・ルーターの障害がまれに起こり、その際にマスター選択の収束に予期される所要時間は非常に短いものです（< 1 秒）。したがって VRRP を最適化すると、プロトコル設計は非常に単純化できますが、短時間のネットワーク性能低下が生じる可能性がわずかながらあります。

23.12.4 構成の例

図 361 には、単純なネットワークの例が示されており、2つの VRRP ルーターに1つの仮想ルーターが実装されています。

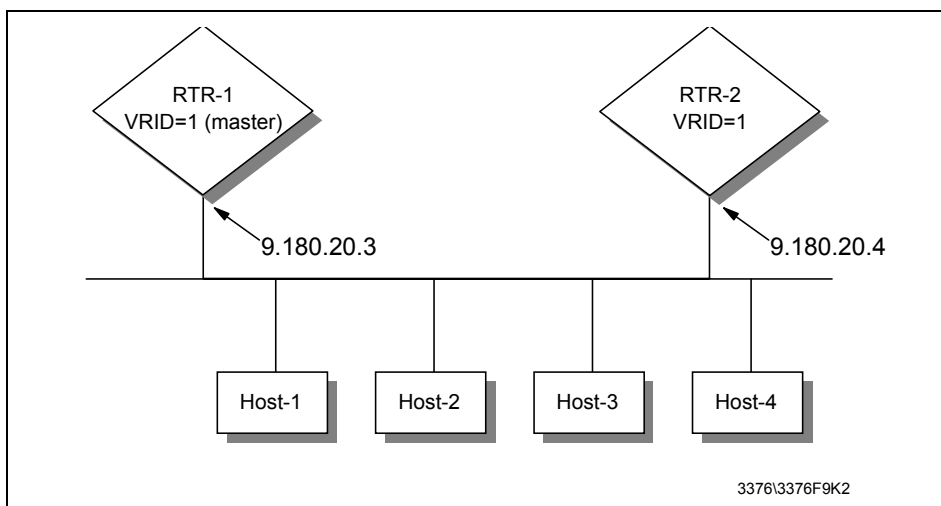


図 361 VRRP の単純な構成の例

上記の構成は、非常に単純な VRRP のシナリオを示しています。この構成では、エンド・ホストには仮想ルーター #1 の IP アドレス (IP アドレス 9.180.20.3) に対するデフォルト経路がインストールされており、両方のルーターとも VRRP を実行しています。左側のルーターは仮想ルーター #1 (VRID=1) のマスターになっており、右側のルーターは仮想ルーター #1 のバックアップになっています。左側のルーターに障害が起きると、右側のルーターに仮想ルーター #1 とその IP アドレスが引き継がれるので、ホストへのサービスが中断しません。この例で、IP アドレス 9.180.20.4 は左側のルーターからバックアップされたものではないことに注意してください。IP

アドレス 9.180.20.4 は、右側のルーターのみでインターフェース・アドレスとして使用されます。IP アドレス 9.180.20.4 のバックアップを作成するには、2 つ目の仮想ルーターを構成する必要があります。この構成が図 362 に示されています。

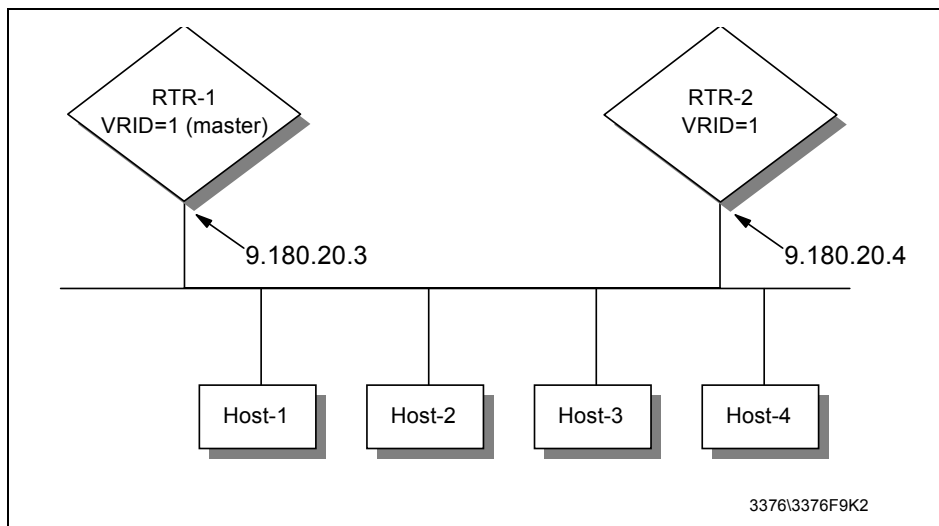


図 362 VRRP の単純な負荷分割構成の例

図 362 には、2 つの仮想ルーターと、その間のトラフィックを分割するホストから成る構成が示されています。この例は、現実の場面では非常に一般的であると想定されています。上記の構成では、半数のホストには仮想ルーター #1 (IP アドレス 9.180.20.3) へのデフォルト経路がインストールされており、残りの半数のホストには仮想ルーター #2 (IP アドレス 9.180.20.4) へのデフォルト経路がインストールされています。その結果、ホストからルーターを介するトラフィックの負荷平準化が実現する一方で、完全な冗長性も保たれます。

23.12.5 VRRP パケットの形式

VRRP パケットの目的は、仮想ルーター ID に関連付けられているマスター・ルーターの優先順位と状態を、すべての VRRP ルーターに通信することです。VRRP パケットは IP パケット中にカプセル化されて送信されます。VRRP パケットは、VRRP に割り当てられた IPv4 マルチキャスト・アドレスに送信されます。IANA によって割り当てられた VRRP の IP アドレスは、224.0.0.18 です。これはリンク・ローカル有効範囲のマルチキャスト・アドレスです。ルーターは、TTL に関係なく、この宛先アドレスを使用してデータグラムを転送できません (67 ページの 3.1 『インターネット・プロトコル (IP)』を参

照)。TTL は 255 に設定しなければなりません。VRRP ルーターで TTL が 255 でないパケットが受信された場合は、このパケットを廃棄しなければなりません。

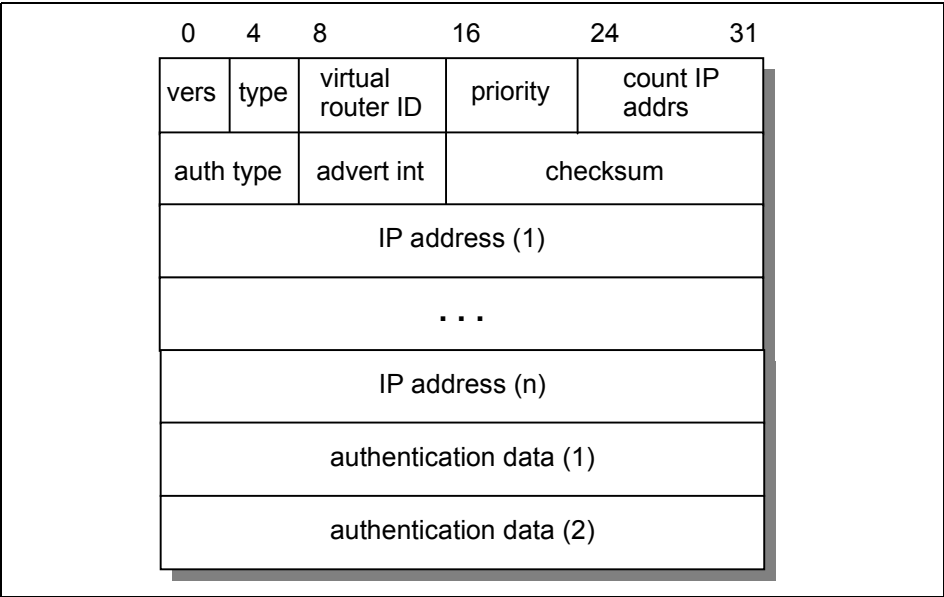


図 363 VRRP パケットの形式

VRRP ヘッダーのフィールドは以下のように定義されています。

Version (バージョン)

バージョンのフィールドは、このパケットの VRRP プロトコル・バージョンを指定します。(RFC 2338 では、バージョンは 2 です。)

Type (タイプ)

タイプのフィールドは、この VRRP パケットのタイプを指定します。このバージョンのプロトコル中に定義されているパケット・タイプは、1 のみです。

Virtual Router ID (VRID) (仮想ルーター ID (VRID))

仮想ルーター ID (VRID) フィールドは、このパケットで状況が報告されている仮想ルーターを識別します。

Priority (優先順位)

優先順位のフィールドは、送信側 VRRP ルーターの仮想ルーターに関する優先順位を指定します。値が大きい

ほど優先順位が高くなります。仮想ルーターに関連付けられた IP アドレスを所有する VRRP ルーターの優先順位の値は、255 でなければなりません。仮想ルーターのバックアップとなる VRRP ルーターには、1 ~ 254 の間の優先順位の値を使用しなければなりません。仮想ルーターのバックアップとなる VRRP ルーターのデフォルトの優先順位の値は、100 です。優先順位の値ゼロ (0) には特別な意味があり、VRRP に関する現行のマスターが停止していることを示します。この値は、現行マスターのタイムアウトを待機せずに、バックアップ・ルーターを起動して即時にマスターへ遷移する場合に使用します。

Count IP Addr (IP アドレスのカウント)

この VRRP 通知に含まれている IP アドレスの数。

Auth Type (認証タイプ)

認証タイプのフィールドは、使用する認証方式を識別します。認証タイプは、インターフェースごとに固有です。認証タイプ・フィールドは 8 ビットの符号なし整数です。認証タイプが不明なパケットや、ローカルで構成された認証方式と合致しないパケットは、廃棄しなければなりません。以下の認証方式が現在定義されています。

0 - 認証なし

1 - 単純テキストのパスワード

2 - IP 認証ヘッダー

Adver Int (通知間隔)

デフォルトは 1 秒です。このフィールドは、ルーターの構成の誤りをトラブルシューティングするために使用します。

Checksum (検査合計)

VRRP メッセージ中のデータ破損を検出するために使用します。

IP address(es) (IP アドレス)

仮想ルーターに関連付けられている 1 つ以上の IP アドレス。

Authentication data (認証データ)

認証ストリングは、現在は単純テキストの認証のみに使用されます。

23.13 ラウンドロビン DNS

負荷平準化の問題に対する初期のソリューションとして、ホスト名を実際の IP アドレスに変換することがよくありました。これが Domain Name System です (291 ページの 8.1 『Domain Name System (DNS)』を参照)。特定のサービスに関する代替 IP アドレスのテーブル全体が回転されるので、ある程度の負荷平準化が達成されます。この方法のことを、ラウンドロビン DNS ということがよくあります (図 364 を参照)。この方法の利点は、プロトコルに準拠していることと、クライアントと宛先ホストの両方に認識されないことです。さらに、この方法はトランザクションの開始時に一度だけ実行されます。

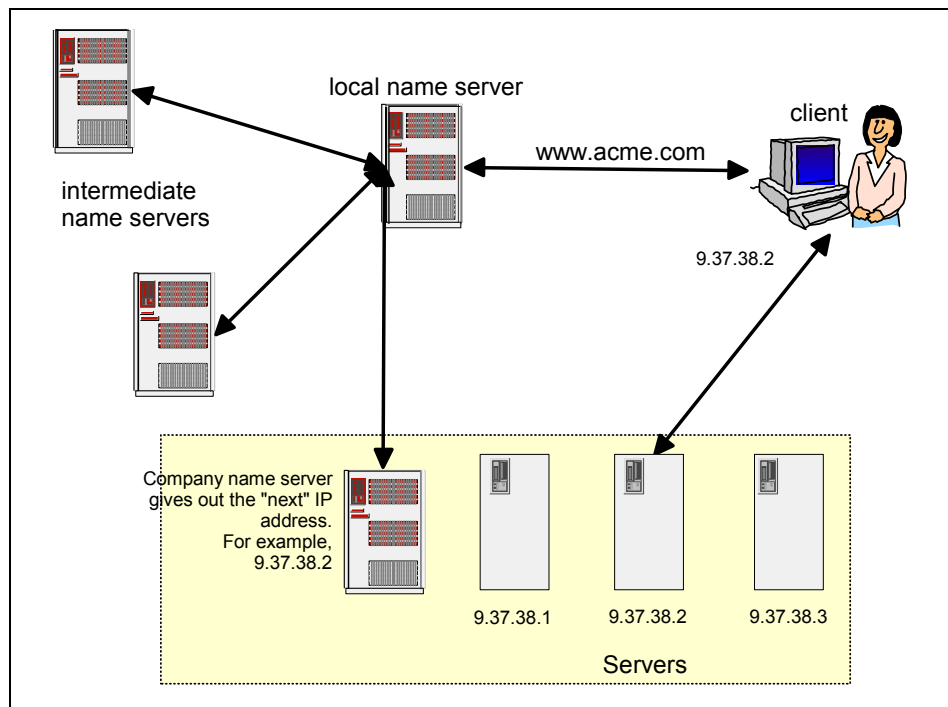


図 364 ラウンドロビン DNS

中間ネーム・サーバーとクライアントのソフトウェア（ほとんどの代表的なブラウザを含む）で、DNS サービスによって戻される IP アドレスがキャッシュに入れられ、明示的に指定された存続時間 (TTL) 値 (67 ページの 3.1 『インターネット・プロトコル (IP)』を参照) が無視される（特に TTL が小さいかゼロの場合）ので、残念ながらこの方法は無効になる場合が時折あります。その結果、クライアントでは平準化が解決される代わりに、キャッシュに入れられた IP アドレスが引き続き使用されるので、DNS によって備えられる平準化機能はう回されます。クライアントで IP アドレスがキャッシュに入れられない場合でも、基本的なラウンドロビン DNS には以下の制限があります。

- ポートごとに区別する機能は備えられません。
- サーバーの可用性を認識しません。
- サーバー上の作業負荷を考慮しません。

23.14 負荷平準化に関する代替ソリューション

負荷平準化に関するハードウェアやソフトウェアの製品を販売している会社は多数あります。非常に多様な手法が使用されており、それぞれ利点も欠点もあります。以下に、これらの代替方法を、Network Dispatcher で使用されている方法と比較しながら考慮します。

23.14.1 ネットワーク・アドレス変換

ネットワーク・アドレス変換 (NAT) は、クライアントからサーバーへのインバウンド・パケット中のソースおよびターゲット IP アドレスに変更を加え、サーバーからクライアントへのアウトバウンド・パケット中でその IP アドレスを元の値に復元します。(NAT の詳細については、720 ページの 21.4 『ネットワーク・アドレス変換 (NAT)』を参照してください。)

NAT がサーバーで認識されないと、サーバー上で特殊なエージェント・コードが必要なくなるので、元々クライアントで使用されていた IP アドレスを復元するには、ロード・バランサーを介してすべてのパケットをクライアントに送り返さなければならないことに注意してください。この点については、図 365 を参照してください。このことは、ロード・バランサーと、ロード・バランサーで管理されるリソースのあるサーバーにさまざまな影響があるので、大きなオーバーヘッドが生じます。

この追加のオーバーヘッドと待ち時間のために、ネットワーク遅延やロード・バランサー事態のキューイング遅延が生じることがあります。この遅延は、NAT ソリューションの拡張容易性の可能性を著しく制限します。この種の遅延が起きないようにするには、ロード・バランサーに、インバウンドとアウトバウンド

トバウンドの両方のパケットを処理できるだけでなく、異常にボリュームが大きいアウトバウンド・トラフィックも処理する能力がなければなりません。

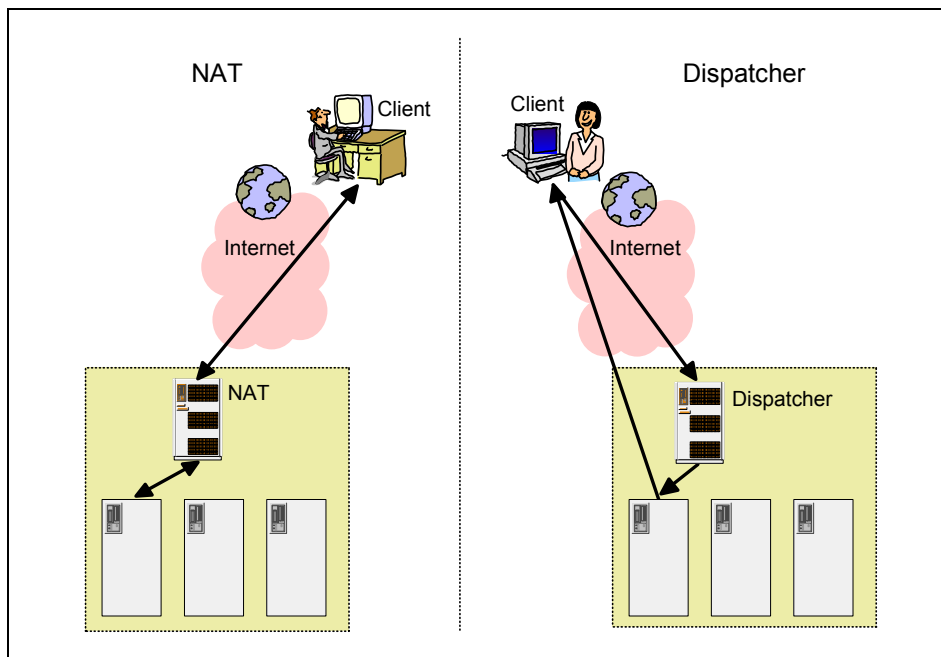


図 365 ネットワーク・アドレス変換と Network Dispatcher - 1

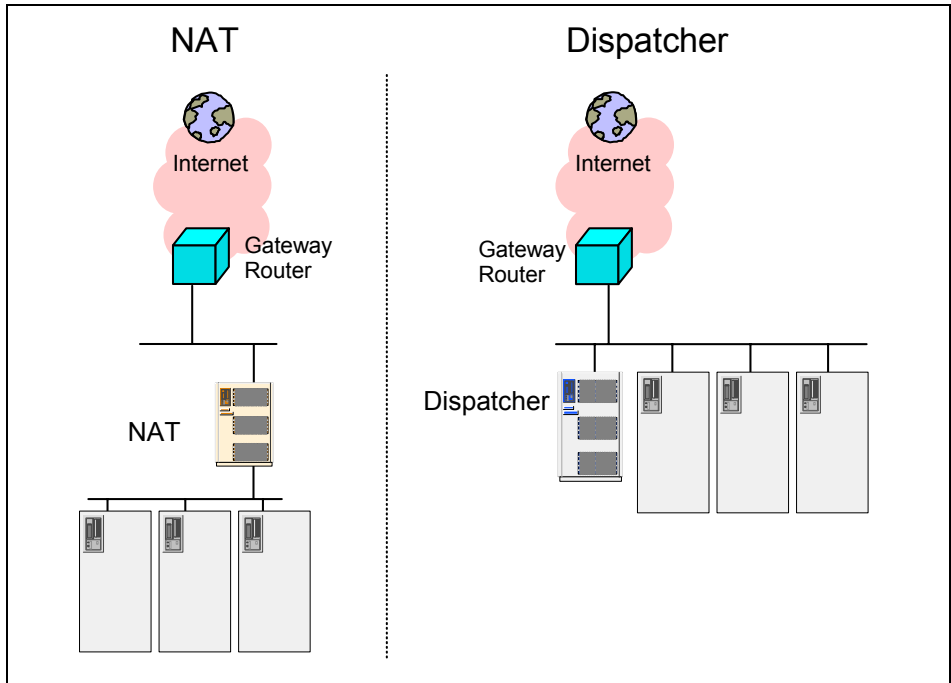


図 366 ネットワーク・アドレス変換と Network Dispatcher - 2

図 366 のように、NAT を使用すると、NAT 装置をブリッジとしてインストールすることになり（他の種類のブリッジは使用できない）、サーバーが実質的に専用のセグメントになってしまうために、インバウンド要求とアウトバウンド要求の両方を参照する必要が生じることが時折あります。この場合、既存のネットワークに大幅な物理的な変更を加える必要があるので、インストールが複雑になる可能性があります。これらのサーバーに関係するすべてのトラフィックは、負荷平準化されるかどうかに関係なく、ロード・バランサーを介して移動しなければなりません。

逆に Dispatcher は、既存のネットワーク基礎構造を破壊せずに即時かつ簡単にインストールできます。さらに、Dispatcher の構成には柔軟性があるので、非対称ネットワークを構成して非常に広い帯域幅を処理できます。

本来認められていた NAT の利点の 1 つ（広域ネットワークを介してリモート宛先にパケットを転送する機能）は、有効にデプロイできません。なぜなら、広域ネットワーク接続はブリッジの背後にあるため、サイトの私設ネットワーク内にしか存在できないからです。さらに、同一の NAT 装置が、広域ネットワーク・リンクからの単一の出口でなければなりません。Dispatcher の広域ネットワーク・サポートにはこのような制限はありません。

これらの制限を解決しようとして、NAT ソリューションの中には、不要な追加機能を備えて、NAT にとって必須のオーバーヘッドを追加しているものもあります。たとえば、ポート・アドレス間のマッピングを行う機能などが該当します。これは、ウェルノウン・ポートの規格と暗黙的に対立します。この機能は、NAT ベースのソリューションの利点として頻繁に宣伝されてきましたが、ポート・マッピングのいわゆる「利点」はそれほど価値はなく、より規格に準拠した他の方法で同様の機能をデプロイできます。

NAT ベースの負荷平準化ソリューションでは、サーバーが起動しているかどうかを検査する際には、実際のクライアント要求を中断する必要があるので、通常はサーバーが停止すると、実クライアント要求の 1 つのタイムアウトの結果と見なすだけにとどまります。Dispatcher で使用される特殊なアドバイザーの方が、破壊される可能性が小さく、障害に対する反応も速くなります。

NAT 装置では、ポート・レベルではなくクライアントの IP アドレスのみに基づいて、類縁性またはスティッキー性がマップされる場合がよくあります。この場合、クライアントからサーバーに接続すると、そのクライアントからの他のアプリケーション用のトラフィックが、すべて同じサーバーに転送されることになります。現実にはスティッキー機能のレンダリングは使用できない場合が多いので、構成の柔軟性が著しく制限されることになります。

23.14.2 カプセル化

負荷平準化の別の方法として、パケットに変更を加えずにカプセル化してからサーバーに渡すプロキシがあります。この方法にはいくつかの長所があります。特にこの方法を使用すると、NAT ソリューションのブリッジングの場合とは違って、ロード・バランサーで広域ネットワーク全体にトラフィックを転送できます。しかし、他のインプリメンテーションでは、すべてのトラフィックでカプセル化が使用されるので、個々のサーバーにロード・バランサーのエージェントをインストールする必要があります。このエージェントは、カプセル化の逆のプロセスを行います。したがって、サーバー・プラットフォームを選択する際には、個々の定義によって、サーバー・エージェントを使用できるプラットフォームに制限されます。また、NAT と同様に、パケットの追加処理が必要なので、大多数のサイトで必要なレベルの拡張容易性が得られない可能性が高くなります。

一方、Dispatcher の場合は、その背後でクラスター化されたサーバーにコードをインストールする必要はありません。さらに、サーバーが稼働しているハードウェアやオペレーティング・システム・プラットフォームを知らなくてもかまいません。

付録 A プラットフォームの実装

IBM オペレーティング・システムには何年もの間 TCP/IP 機能が装備されており、TCP/IP の適用範囲が継続的に増大し、向上してきました。この章では、いくつかの IBM オペレーティング・システムおよびハードウェア・プラットフォームにおける、主な TCP/IP の機能とプロトコルの例について説明します。プラットフォームや機能をすべて概略するのではなく、IBM 社から入手できるさまざまな TCP/IP スタック機能に注目しています。

A.1 IBM Communications Server for OS/390 V2R10

OS/390 を稼働する IBM エンタープライズ・サーバーは、主幹業務の作業負荷を担うのに役立つ能力、拡張容易性、セキュリティ、および可用性を備えています。CS for OS/390 の広範囲に及ぶ優れた機能の中には QoS、RSVP、および高機能なセキュリティ機構が含まれていますが、これらの機能について詳しく説明することが本書の目的ではありません。以下では、これを実装した場合の能力と柔軟性に注目します。完全な説明については、「*IBM Communications Server for OS/390 TCP/IP Implementation Guide Volume 1: Configuration and Routing* (SG24-5227)」を参照してください。

IBM Communications Server for OS/390 IP は、OS/390 UNIX システム・サービスを使用するように、OS/390 TCP/IP を発展させる作業の第 2 段階になります。現在の CS for OS/390 IP は、スタックが再設計され、パフォーマンスが向上しています。その中には、Communications Storage Management (CSM) と、VTAM のマルチパス・チャネル (MPC) や Queued Direct I/O (QDIO) 機能の利点を活用することがあります。このような VTAM との密接な結合により、パフォーマンスや保守容易性が向上します。CS for OS/390 IP では、新旧のネットワーク接続機構を介して、OS/390 UNIX システム・サービスおよび従来の MVS 環境へアクセスするという、2 つの環境が備えられています。

A.1.1 サポートされている接続プロトコルと装置

DLC は、TCP 排他 DLC と共用 DLC の 2 つのカテゴリーに分類できます。TCP 排他 DLC は、CS for OS/390 IP スタック専用で、CS for OS/390 IP の複数インスタンス間で共用することはできません。CS for OS/390 IP でサポートされている TCP 排他 DLC には、以下のチャネル・プロトコルが含まれています。

- Channel Data Link Control (CDLC)

このプロトコルは、CS for OS/390 IP と、374x 実行ネットワーク制御プログラム (NCP) 内または 9x0 チャネル接続ルーター内でコード化された IP ルーターとの間の、固有の IP 接続をサポートします。

- Common Link Access to Workstation (CLAW)

このプロトコルは、CS for OS/390 IP を、3172 実行 ICCP、RS/6000、およびこのインターフェースをサポートしている Cisco ルーターに接続するために使用します。

- Channel-to-Channel (CTC)

このプロトコルは、2 つの CS for OS/390 IP システム間でサポートされ、1 つの読み取り / 書き込みチャネルの対を使用します。並列チャネルと ESCON チャネルの両方ともサポートされています。

- Hyperchannel

このプロトコルは、NSC A220 Hyperchannel Adapter とその下位アダプターを介して接続するのに使用します。

- LAN Channel Station (LCS)

このプロトコルは、OSA、3172 実行 ICP、2216、および 3746-9x0 MAE で使用されます。

さらにもう 1 つの TCP 排他 DLC プロトコルがありますが、このプロトコルでは S/390 チャネルを使用できません。SAMEHOST DLC (PTP Samehost と混同しないでください) を使用すると、同一の MVS イメージ上で実行されている CS for OS/390 IP と他のサーバーの間で通信できます。以前にはこの通信は IUCV によって提供されていました。現在は、以下の 3 つのサーバーで SAMEHOST DLC が活用されています。

- SNALINK LU0

このサーバーには、LU0 トラフィックを使用する SNA ネットワークを介した接続が備えられています。このサーバーは、OS/390 VTAM に対するアプリケーションの働きをします。

- SNALINK LU6.2

このサーバーには、LU6.2 トラフィックを使用する SNA ネットワークを介した接続が備えられています。このサーバーも、OS/390 VTAM に対するアプリケーションの働きをします。

- X.25

このサーバーには、NCP パケット交換インターフェース (NPSI) を使用した X.25 ネットワークへの接続が備えられています。

共用 DLC は、マルチプロトコル・スタックの複数インスタンスで同時に使用できます。たとえば、CS for OS/390 IP の 1 つ以上のインスタンスと OS/390 VTAM の 1 つ以上のインスタンスとの間で、共用 DLC を使用できます。この種の共用 DLC には、以下のものが含まれます。

- Multipath Channel+ (MPC+)

MPC+ は Multipath Channel (MPC) プロトコルの拡張バージョンです。これを使用すると、複数の読み取りチャネルや書き込みチャネルを効率よく使用できます。High Performance Data Transfer (HPDT) では、MPC+ と Communication Storage Manager (CSM) を一緒に使用して、データ伝送に必要なデータ・コピーの数を少なくします。このタイプの接続を使用する方法は 2 つあります。

1 つ目の方法は MPCPTP というもので、この場合 CS for OS/390 IP は、2 地点間方式で対等の IP スタックに接続されます。この方法では、CS for OS/390 IP を以下のいずれかに接続できます。

- 別の CS for OS/390 IP スタック
- 2216
- RS/6000
- 3746-9x0 MAE
- Cisco ルーター (Cisco Channel Interface Processor (CIP) または Cisco Channel Port Adapter (CPA) を介して)

2 つ目の方法は、MPC+ を使用して Open Systems Adapter (OSA) に接続するというものです。この構成中の OSA は、MPCPTP の場合のように対等 IP スタックの働きをするのではなく、CS for OS/390 IP スタックの拡張機能の働きをします。この方法に関連して、以下のものがサポートされています。

- OSA-2 固有の ATM (RFC1577)
- OSA-2 高速イーサネットおよび FDDI (MPCOSA)
- OSA-Express QDIO では、MPC+ を使用して、CS for OS/390 IP と OSA-Express の間の制御信号を交換します。

• MPCIPA (QDIO)

OSA-Express には、Queued Direct I/O (QDIO) という新しい通信用の機構が備えられています。QDIO では MPC+ プロトコルを使用して制御信号が交換されますが、QDIO インターフェースはチャネル・プロトコルとは全く違います。これは直接メモリー・アクセス (DMA) を使用するので、チャネル・プログラムに関連したオーバーヘッドが生じません。元々、OSA-Express と CS for OS/390 IP には、ギガビット・イーサネット接続機構しかサポートされていませんでした。CS for OS/390 V2R10 IP では、高速イーサネットと ATM LAN エミュレーションの QDIO サポートも備えられています。

CS for OS/390 IP と OSA-Express Adapter が協働することにより、S/390 からアダプターへの数値計算機能のオフロードが備えられます。このインターフェースのことを IP Assist (IPA) といいます。オフロードにより、既存の OSA-2 インターフェースと比べて、ネットワーク・インターフェー

スに必要な S/390 サイクルが少なくなり、OS/390 OSA-Express 環境が全体的に改善されます。

- XCF

XCF DLC を使用すると、システム間カップリング・ファシリティ（XCF）を介して、並列シスプレックス中の複数の CS for OS/390 IP スタック間で通信できます。XCF DLC は、従来の DLC と同様に定義できますが、XCF Dynamics もサポートされています。XCF Dynamics では XCF リンクが自動的に起動されます。

- PTP Samehost

この接続タイプは、IUTSAMEH と呼ばれることもあり、同一の MVS イメージ上で実行されている複数の CS for OS/390 IP スタックの接続に使用されます。また、これらの CS for OS/390 IP スタックを OS/390 VTAM に接続し、Enterprise Extender を使用することもできます。

A.1.2 サポートされているサポート・アプリケーション

CS for OS/390 IP には、ORouteD と OMROUTE という 2 つのルーティング・アプリケーションが付属しています。OMROUTE と ORouteD を同一の TCP/IP スタック上で並行して実行することはできません。これらのアプリケーションにより、ルーティング・テーブル内のルーティング項目の追加、削除、変更が行われます。また、CS for OS/390 IP プロファイル内の Gateway または BEGINROUTES 定義を介して作成された静的経路の代わりに、これらのアプリケーションを使用できます。

A.1.2.1 ORouteD

ORouteD は、Routing Information Protocol (RIP) バージョン 1 (RFC 1058) および RIP バージョン 2 (RFC 1723) を、CS for OS/390 IP に実装したものです。ORouteD は、OMROUTE よりはるかに古いアプリケーションなので、制限があります。ORouteD にはゼロ・サブネットはサポートされていません。また ORouteD には、宛先ネットワークやホストに対する、同じコストのマルチパス経路がサポートされていません。したがって、OMROUTE をルーティング・アプリケーション（ルーティング・デーモンともいう）としてお勧めします。

A.1.2.2 OMROUTE

OS/390 V2R6 IP 以降では、OMROUTE には、RFC 1583 で説明されている Open Shortest Path First (OSPF) プロトコル (OSPF バージョン 2) と、RIPv1 および RIPv2 が実装されています。適切に構成すると、OMROUTE と共に稼働している OS/390 ホストが、TCP/IP ネットワーク中のアクティブな OSPF ルーター、RIP ルーター、またはその両方になります。これらの 2 つのルーティング・プロトコルのどちらか（または両方）を使用して、ホストのルーティング・テーブルを動的に保守できます。また、OS/390 V2R7 IP に

は新しい OMPROUTE サブエージェントも備えられており、このサブエージェントには OSPF プロトコルと状態情報を含む OSPF MIB 変数が実装されています。この MIB 変数は RFC 1850 中で定義されています。

A.1.3 Enterprise Extender

OS/390 V2R6 以降には、Enterprise Extender (HPR/IP ともいう) がサポートされています。Enterprise Extender には、サービス・クラスや伝送優先順位などの、IP ネットワーク中での APPN (拡張対等通信ネットワーキング機能) の従来の利点が備えられています。Enterprise Extender を使用して、以下の特性の 3 層ネットワークを構成できます。

- 専用 IP バックボーン・ネットワーク
- 主幹業務の SNA アプリケーション用の継続的で信頼性の高いサービス
- 費用効果の高いネットワーク・サービス提供

Enterprise Extender では、User Datagram Protocol (UDP) を使用して IP ネットワークにアクセスします。

A.1.4 仮想 IP アドレッシング (VIPA)

(静的) 仮想 IP アドレッシング (VIPA) の本来の目的は、特定のネットワーク接続機構上でホスト・アプリケーションの従属関係をなくすことにありました。サーバーに接続するクライアントでは、通常は複数のネットワーク・インターフェース (IP アドレス) から 1 つ選択します。選択したインターフェースがダウンすると、接続もダウンするので、別のインターフェースを介して再確立する必要があります。また、インターフェースがダウンしている間は、障害が起きたインターフェース (および IP アドレス) に対する新しい接続を確立できません。

VIPA を使用する際には、物理的な接続機構やインターフェースに対応しない仮想 IP アドレスを定義します。その後、CS for OS/390 IP は、IP ネットワークに対して、VIPA アドレスは独立したサブネットワーク上にあり、CS for OS/390 自体はこのサブネットワークに対するゲートウェイになることを示します。VIPA アドレスを選択してサーバーに接続するクライアントでは、使用可能な実ホスト・インターフェースの 1 つを介してパケットを VIPA にルーティングします。このインターフェースに障害が起きると、別のアクティブ・インターフェースを使用して、非介入でパケットが VIPA アドレスに転送されます。

CS for OS/390 V2R8 IP では、VIPA の可用性の適用範囲に関する概念が拡張され、障害が起きたシステム・イメージや TCP/IP スタック全体をリカバリーできます。特に、以下の 2 つの VIPA の拡張が導入されています。

- VIPA 自動引き継ぎ機能を使用すると、シスプレックス中の複数の TCP/IP スタック上に同一の VIPA アドレスを定義できます。あるスタックを VIPA の 1 次つまり所有スタックとして定義し、別のスタックを 2 次つまりバックアップ・スタックとして定義できます。1 次スタックだけが IP ネットワークに認識されます。所有スタックに障害が起きると、2 次スタックの 1 つに引き継がれ、VIPA の所有権が想定されます。ネットワーク内では、ルーティング・テーブル中だけで変更が確認されます。この場合、これらの DVIPA に関連付けられたアプリケーションがバックアップ・システム上でアクティブになるので、サービスのホット・スタンバイが備えられます。
- 動的 VIPA (アプリケーション・インスタンスの) を使用すると、アプリケーションで TCP/IP スタックに独自の VIPA アドレスを登録できます。この場合、シスプレックス・イメージ間でアプリケーション・サーバーを移動しても、そのサーバーを名前やアドレスで認識しているクライアントには影響はありません。アプリケーション・インスタンスの物理的な場所が移動しても、名前やアドレスは一定です。この方法を使用すると、アプリケーションのホストにしたいシステム・イメージ上で動的に VIPA をアクティブにできます。アプリケーション・インスタンスは一度にシスプレックス上の 1 つのイメージでのみアクティブになるので、他のイメージにはサービスのコールド・スタンバイが備えられることになります。

XCF を使用して TCP/IP スタック間で通信すると、これらの VIPA 拡張を使用できます。つまり、XCF はシスプレックス中の VIPA に関する通信の基礎になります。XCF の動的性によって構成が容易になるので、多くのさらに新しい VIPA 機能によって構成は容易になります。

A.1.5 Sysplex Distributor

Sysplex Distributor これは、S/390 IP サーバー間の接続をディスパッチする最先端のテクノロジーです。特に Sysplex Distributor により、動的 VIPA と VIPA の自動引き継ぎが拡張され、シスプレックス中のターゲット・サーバー間で負荷を分散できます。Network Dispatcher で使用されている着信接続の分散のテクノロジーと、動的 VIPA とを結合して、シスプレックス中の特定のサービスの高可用性を確実に実現します。

Sysplex Distributor では、1 つの IP エンティティーで、特定のサービスを認識するための特定の IP アドレスの所有権が公示されます。この方法により、Sysplex Distributor の単一システム・イメージも、特殊な IP アドレスの単一システム・イメージになります。この IP アドレスのことを、分散 DVIPA といいます。さらに、Sysplex Distributor では、分散 VIPA を公示し、自分に対する接続をディスパッチする IP エンティティー自体が、シスプレックス中のシステム・イメージになり、分散元スタックと呼ばれます。

Sysplex Distributor では、ワークロード・マネージャー (WLM) とその機能を使用して、サーバーの負荷が測定されます。この枠組みで、WLM から分散元スタックにこのサーバーの負荷が通知されるので、分散元スタックでは着呼接続要求の送信先について最善の決定が下されます。また Sysplex Distributor には、ポリシー・エージェント中の特定のポリシーを指定する機能もあるので、WLM のサーバー負荷情報に加えて、ターゲット・スタックから QoS 情報も使用できます。さらに、これらのポリシーにより、特定のサブネットワーク中でクライアントのターゲット・スタックも候補として指定できます。

接続要求は、Sysplex Distributor の分散先スタックに送信されます。スタックでは、個々の要求の受信先として最善なターゲット・サーバーが候補として選択され、その要求がルーティングされます。状態が保守されるので、この接続に関連したデータ・パケットを正しいスタックに転送できます。また、シスプレックス中のサーバーから送信されるデータは、分散元スタックを介する必要はありません。

Sysplex Distributor は、SecureWay Communications Server for OS/390 V2R8 IP に導入されている動的 VIPA と自動 VIPA 引き継ぎの機能も拡張します。この拡張により、VIPA を別のスタックに接続を破壊せずに移動できます。つまり、以前は VIPA はシスプレックス中の 1 つのスタック上だけでアクティブにすることができました。この場合、あるスタック上に接続があると、別のスタックに VIPA を移動しようとしただけで、サービス中に破壊が生じる可能性がありました。Sysplex Distributor を使用すると、元々 VIPA を所有しているスタック上の既存の接続を破壊せずに、VIPA を移動できます。

要約すると、Sysplex Distributor には以下の利点があります。

- 構成が簡単

Sysplex Distributor を使用すると、別のレベルへの構成が簡単になります。分散の初期構成は非常に簡単に行えます。さらに、構成しなくてもサーバーを分散に追加できます。

- サーバー負荷を測定する際の正確さの向上

Sysplex Distributor では、WLM によって備えられるサーバー負荷情報が使用されます。しかし、ターゲット・サーバーからの QoS パフォーマンス測定も使用して、着信接続のターゲット・サーバーの選択に活用することもできます。また、ターゲット・スタック候補の集合を、接続を要求しているクライアントに応じて変えることもできます。したがって、簡単な方法で、特定のクライアントのサブセットに対して特定のスタックを予約できます。

- 究極の可用性

Sysplex Distributor と自動 VIPA 引き継ぎを協働させて、特定のサービスに関連した接続の分散を確実に行えます。シスプレックス分散中のすべて

のスタックを分散元スタックのバックアップにすることができるので、システム・イメージが1つだけあれば、サービスの可用性は保証されます。この方法で、ターゲット・サーバーを着信接続の分散用のバックアップ・スタックにすることができます。

- 統合が簡単

エンド・ユーザーは、Sysplex Distributor によって分散が実行されていることに気付きません。単に新しいサーバー IP アドレス (またはホスト名) に接続すると認識します。

- DNS からの独立性

Sysplex Distributor は、負荷平準化のホスト名による解決に従属していません。

- 追加のハードウェアが必要ない

Sysplex Distributor の機能はすべてシスプレックス中に含まれているので、ハードウェアを追加しなくてもこの機能の利点を活用できます。

- パフォーマンス

Sysplex Distributor はシスプレックス中に含まれるので、CS for OS/390 IP でクラスター中の同一性の利点を活用できます。したがって、分散元スタックを介する接続やデータの転送を非常に高速に行えます。また、シスプレックス中のスタック間の通信が向上するので、VIPA の障害を速く認識し、バックアップ機能を拡張できます。

A.1.6 サービス品質 (QoS)

OS/390 UNIX サービスのポリシー・エージェント (PAGENT) は、構成ファイルからサービス・レベルのポリシーを読み取って TCP/IP スタックにインストールするサーバーです。

ネットワーク管理者は、PAGENT を使用して、ユーザーの便宜を図って以下のサービス・レベル・ポリシーを定義できます。

- TCP スループットの制御
- クライアントからの接続要求のアドミッション制御
- 不要なデータグラムのブロッキング
- 日時に基づくポリシーの仕様の活動化
- 優先キューや選択的廃棄に TOS 設定を使用する、CS for OS/390 内と IP ルーター内のすべての IP トラフィックの優先順位付け
- OS/390 V2R8 IP 以降、ポリシー・エージェントでは、RSVP アプリケーションによって要求される RSVP 予約パラメーターを制限しており、アクティブな RSVP トラフィック・フローの合計を制限しています。

- CS for OS/390 V2R10 IP では、ポリシー・エージェントには Traffic Regulation と Sysplex Distributor のポリシーがサポートされています。

ポリシー・エージェントでは、ポリシー構成ファイルか LDAP サーバーから サービス・ポリシーの規則とステートメントを取り出して、(すべての MVS イメージ間の) CS for OS/390 スタックにインストールします。

PAGENT で発信 IP パケットの TOS 値を設定し、必要に応じてこの値を使用してスタック中の既存のポリシーをフラッシュアウトしたり更新したりできます。また、PAGENT を使用して、HFS ファイル中に情報やエラー・メッセージを記録することもできます。さらにポリシー・エージェントにより、ローカル・ファイルや LDAP サーバー中で、TCP/IP スタック中に保守されている更新済みポリシーを動的にモニターすることもできます。

A.2 IBM OS/400 V5R1

IBM @server iSeries および AS/400 サーバーは、中性能の分野において信頼性の高さが知られています。多年にわたり、このサーバーのハードウェアとソフトウェアは、64 ビットのアーキテクチャーで構築されてきました。その後の複数の OS/400 バージョンおよびリリースで、このサーバー・プラットフォームは拡張され、高性能な e-business サーバーになりました。IBM @server iSeries サーバーには、完全かつ堅固な TCP/IP プロトコル、サーバー、およびサービスの一式が付属しています。

OS/400 には、TCP/IP 通信プロトコル機能および関連した管理機能と構成機能が組み込まれ、パッケージ化されています。TELNET、Simple Mail Transfer Protocol (SMTP)、ファイル転送プロトコル (FTP)、Routing Information Protocol 1 および 2 (RIP)、および LPR/LPD (リモート印刷サポート) などの TCP/IP アプリケーションは、Pascal ベースの API を含む TCP/IP ユーティリティとして残っています。これらの TCP/IP ユーティリティは、OS/400 を注文したすべての顧客に自動的に配送されます。

TCP/IP はネットワーク・コンピューティングの枠組みの基礎になります。大半の新しい IBM @server iSeries e-business のインフラストラクチャーは、TCP/IP 上で排他的に稼働します。Lotus Domino、Java、WebSphere、Web サービング、および IBM Network Stations もこの対象になります。最近の TCP/IP 拡張により、IBM @server iSeries がさらに強力な e-business サーバーになっています。

以下に、OS/400 V5R1 に組み込まれているいくつかの機能の例を取り上げます。機能をすべてリストすることは意図されていません。

A.2.1 GUI 構成サポート

IBM @server iSeries オペレーション・ナビゲーターに備えられているグラフィカル・ユーザー・インターフェースを使用して、IBM @server iSeries TCP/IP の構成の全体を管理できます。TCP/IP の構成をステップ形式で分かりやすく手引きするグラフィック・ウィザードが組み込まれています。その他にも、さまざまなサーバー構成のセットアップをサポートする構成ウィザードを使用できます。IBM @server iSeries サーバー・ネットワーク管理は容易にはなっていません。

A.2.2 TCP/IP Connectivity Utilities for IBM @server iSeries

TCP/IP Connectivity Utilities for IBM @server iSeries には、以下を含む、多数のサーバーとサービスの一式が組み込まれています。

- GUI 構成サポート
- ファイル転送プロトコル (FTP) クライアントおよびサーバー
- Simple Mail Transfer Protocol (SMTP)
- Post Office Protocol (POP) バージョン 3 サーバー
- Web ベースの管理サーバー
- ネットワーク・ファイル・システム (NFS) クライアントおよびサーバー
- ドメイン・ネーム・システム (DNS) サーバー
- Dynamic Domain Name System (DDNS) サーバー
- Dynamic Host Configuration Protocol (DHCP) サーバー
- IP 印刷
- ライン・プリンター・リクエスター (LPR) およびライン・プリンター・デーモン (LPD)
- 5250/HTML ワークステーション・ゲートウェイ (WSG) サーバー
- Telnet クライアントおよびサーバー
- リモート EXECution (REXEC) クライアントおよびサーバー
- リモート IPL サポート
- BOOTP サーバー
- TFTP サーバー

IBM @server iSeries サーバーでは、以下を含む広範囲の物理インターフェースがサポートされています。

- IBM トークンリング LAN
- イーサネット LAN
- イーサネット 100 Mb LAN
- 1 Gbps のイーサネット
- フレーム・リレー
- 無線 (LAN)
- 統合 PC サーバー LAN
- 非同期サポート

- 同期サポート
- サービス総合ディジタル網 (ISDN)
- ATM (LAN エミュレーション)

A.2.3 動的 IP ルーティング (RIP および RIP2)

Routing Information Protocol (RIP) は、動的 IP ルーティング・プロトコルの一種で、隣接しているルーターと通信し、それぞれのネットワーク接続について互いに通知します。動的ルーティング・プロトコルにより、ネットワークの保守が容易になり、ネットワークのパフォーマンスと信頼性が向上します。IBM @server iSeries には、RIP バージョン 1 と RIP バージョン 2 の両方が組み込まれています。RIP のバージョン 2 には、セキュリティー機能と効率化機能が追加されています。

A.2.4 拡張機能

AS/400 プラットフォームには、多数の TCP/IP 拡張機能が組み込まれていますが、その一部について以下で説明します。

A.2.4.1 Simple Network Management Protocol (SNMP)

Simple Network Management Protocol (SNMP) には、インターネット環境を管理する機能が備えられています。SNMP は、TCP/IP ネットワークの個々のノードで使用され、SNMP マネージャーによってノードのモニターや管理が行われます。IBM @server iSeries SNMP エージェントには、ホスト間でのネットワーク管理メッセージや情報の交換がサポートされています。OS/400 には管理情報ベース (MIB-II) がサポートされています。IBM @server iSeries 用の SNMP には、以下の機能が組み込まれています。

- アプリケーション管理用の SNMP API には、ローカルまたはリモートの SNMP エージェントを介して管理データを操作する機能があります。AnyNet/400 サポートを使用して、SNA ネットワークや TCP/IP ネットワーク上のシステムから情報を取り出せるので、ネットワーク中の潜在的な問題を簡単に発見したり管理したりできます。
- SNMP エージェントは拡張可能で、サブエージェントを動的に追加できる API が備えられているので、管理上の決定をする際に必要な追加情報を表示できます。また OS/400 には、IBM @server iSeries サーバーのハードウェアおよびソフトウェアのインベントリー用に、ホスト・リソース管理情報ベース (MIB) もサポートされています。Independent Software Vendor (ISV) が SNMP API を使用して、IBM @server iSeries 管理アプリケーションを作成し、インベントリー・データの収集、モニター、ネットワーク中のリソースの変更、およびさまざまなその他のタスクを行えます。また、サブエージェントを作成して、他のプラットフォーム上で実行されている管理アプリケーションから、追加の IBM @server iSeries 管理情報にアクセスできます。

- SNMP 管理機能は 2 種類のエンティティに分かれます。1 つは“マネージャー”で、もう 1 つは“エージェント”です。SNMP のエージェント機能は、IBM @server iSeries サーバー上で実行され、SNMP マネージャー機能が実装されているネットワーク管理ステーションで管理できます。OS/400 SNMP エージェントにより、TCP/IP に関する構成データ、パフォーマンス・データ、および問題管理データが SNMP マネージャーに提供されます。以下の管理情報ベースがサポートされています。

- MIB-II
- 伝送グループ
- APPN
- 専用

- 現在、LAN 上の TCP/IP ネットワーク中にプリンターを配置することが一般的に行われています。SNMP 管理には、このネットワーク中で、印刷ファイルをこれらのプリンターに送信し、これらのプリンターからの応答を管理するオプションがあります。このオプションは、OS/400 プリンター記述中で構成され、業界標準の Simple Network Management Protocol (SNMP) に基づいています。このオプションは、プリンター・メッセージ情報ブロックをサポートする SNMP プリンターをサポートしています。
- V4R5 より前は、現行のプリンター・ジョブ言語 (PJI) ドライバーからライン・プリンター・リクエスト (LPR) を使用し、ホスト印刷変換からの出力を使って ASCII プリンターに印刷していました。新しい SNMP ドライバーを使用すると、直接制御機構プリンターやプリンター・ジョブ言語 (PJI) ドライバーで同じ機能を参照しながら、IBM @server iSeries サーバーからもっと多くのプリンターにアクセスできます。

A.2.4.2 ネットワークのサービス品質 (QoS)

IBM @server iSeries の V5R1 には、ネットワーク内の TCP/IP トラフィックの制御と管理を行い、ルーターやスイッチに含まれている最先端のネットワーク・サービス品質 (QoS) の利点を活用する機能が備えられています。IBM @server iSeries の、TCP/IP トラフィックを管理する QoS 機能により、適用される QoS ポリシーに基づいて TCP/IP トラフィックの廃棄、マーキング、およびシェーピングを行うことができます。さらに、帯域幅管理要求の制御のために QoS アドミッション制御機能が備えられています。以下の QoS 機能がサポートされています。

- IBM @server iSeries RSVP エージェントを含む Resource Reservation Protocol (RSVP)
- アプリケーション用 RSVP API (X/Open 規格 API)
- DiffServ

- TCP/IP 5 タプル (ソース IP アドレス、宛先 IP アドレス、ソース・ポート、宛先ポート、およびプロトコル)、アドレス範囲、およびワイルドカードに基づく QoS ポリシー。このサポートには、ポリシー・エージェントと、オペレーション・ナビゲーター中の QoS ポリシー構成用のウィザード・ベースの GUI が含まれます。
- QoS ポリシーの効果性をモニターする QoS モニター API および GUI

A.2.5 Proxy Address Resolution Protocol (Proxy ARP)

Proxy ARP は IP ネットワーキング技法で、これを使用すると、あるマシン、つまりプロキシ・エージェントが別のマシンに代わって ARP 要求に応答できます。これは SLIP、PPP、および平衡型接続の場合に便利です。なぜなら、装置が論理的に同一のローカル LAN サブネット中にあるかのように見えるので、動的ルーティング・プロトコルや静的経路定義を実装する必要がなくなるからです。Proxy ARP は、フレーム・リレー・ネットワーク中での透過性のあるサブネットの作成や、LPAR 区画間の仮想 LAN 通信にも使用します。

A.2.6 Point-to-Point Protocol (PPP)

Point-to-Point Protocol (PPP) は、広域ネットワーク TCP/IP 接続用の開放型プロトコルの 1 つで、ダイヤル回線と専用回線の両方をサポートできます。これを使用して、複数の場所にわたる企業のイントラネットを拡張できます。また、これはインターネット・サービス・プロバイダー (ISP) を介したインターネット接続に関するデファクト・スタンダードです。PPP をダイヤルアップ・プロトコルとして使用すると、SLIP (Serial Line Internet Protocol) に代わる、より堅固なプロトコルになります。PPP の ISDN サポートを使用すると、IBM @server iSeries で、ISDN 交換網に接続できます。IBM @server iSeries で PPP を使用すると、リモート LAN アクセスのための、また企業内のイントラネット内へのリモート・オフィス・ゲートウェイとしての、優れた統合ソリューションが備えられます。認証ユーザーや応答側の接続プロファイルに基づくフィルター操作により、IBM @server iSeries サーバー中またはその下のリソースに対するアクセスを制御できます。IBM @server iSeries には、多重リンク接続もサポートされており、これを使用すると、複数の回線を束ねて、待ち時間を短くしたり、帯域幅の必要に応じて回線の追加や除去を動的に行ったりできます。

A.2.7 セキュリティー機能

AS/400 では多数の機能を使用してセキュリティーに対応していますが、それらの一部について概略します。

A.2.7.1 TCP/IP パケット・セキュリティー

TCP/IP パケット・セキュリティーは、フィルター操作やパケット規則ともいいますが、アプリケーションやサービスに対するネットワーク・アクセスを選択的に制限する、つまりジャーナル処理します。このセキュリティーを使用すると、機密アプリケーションを実行していたり、Web サーバーの働きをしたりしている IBM @server iSeries サーバーに関する保護を追加できます。TCP/IP パケット・セキュリティーは、IBM @server iSeries がカジュアル・ルーターの働きをする場合に、サブネットワーク全体を保護するのに役立ちます。ネットワーク・セキュリティーを層の形式で実装する場合は、TCP/IP パケット・セキュリティーを 2 次レベルの防御として実装することもできます。

A.2.7.2 仮想プライベート・ネットワーク

IBM @server iSeries の仮想プライベート・ネットワーキング・サポートは、業界標準に基づいており、以下のものが含まれています。

- IP Security Protocol (IPSec)
- Internet Key Exchange (IKE)
- Layer 2 Tunneling Protocol (L2TP)

IBM @server iSeries の VPN (仮想プライベート・ネットワーキング) サポートは、V4R4 で導入されて V5R1 で拡張されましたが、追加のセキュリティー、信頼性の向上、パフォーマンスの改善、および使いやすさの向上を実現しています。オペレーション・ナビゲーターが再設計され、VPN の構成を直感的に分かる仕方でナビゲートできるようになったので、VPN ウィザードを使用してネットワーク・セキュリティー・ポリシーのセットアップや実装を行えます。デジタル証明書により、暗号操作に関する拡張容易性と保護機構が備えられています。現行の V5R1 では、デジタル証明書を VPN 構成中で使用して、VPN 端点の識別を認証できます。事前に共有されている秘密や RSA 署名を使用して、メイン・モードやアグレッシブ・モードで、IKE ネゴシエーションを実行できます。OS/400 VPN には、プラットフォームごとに固有の VPN ネットワーク・アドレス変換 (VPN NAT) も備えられているので、IPSec プロトコルの AH や ESP を使用している場合に、代表的な NAT の制限を無効にできます。通常、VPN 接続中で NAT を適用すると、セキュリティー・アソシエーション (SA) のセットアップのために、AH を使用した認証が中断され、トンネル・モードで ESP が処理されず、トランスポート・モードでも ESP と共に使用できません。IP パケットのフィルター操作は、IBM @server iSeries VPN の肝要な部分です。V5R1 では、このコンポーネントが拡張され、インターフェース単位でフィルターの活動化や非活動化を行えるようになりました。

A.2.7.3 Layer 2 Tunneling Protocol (L2TP)

Layer 2 Tunneling Protocol (L2TP) は、拡張リンク・プロトコルの 1 つで、インターネットを介したマルチホップ・バーチャル・サーキットを備えていま

す。L2TP のことを“仮想 PPP”ともいいます。なぜなら、L2TP ではリンク層にバーチャル・サーキットを作成してから、Point-to-Point Protocol (PPP) を使用してネットワーク層での接続を完了するからです。L2TP は、普通は仮想プライベート・ネットワーク (VPN) と一緒に使用して、インターネットを介した保護接続を備えます。

Layer 2 Tunneling Protocol 用に以下のモードがサポートされています。

- リモート・オンデマンドのダイヤル (必須トンネル): このモード・タイプは、ISP と IBM @server iSeries サーバーの間に L2TP トンネルをセットアップします。リモート・ユーザーは、トンネリングが行われたことに気付きません。必須トンネルが理想的ですが、選択した ISP でこの L2TP モードがサポートされていなければなりません。
- オンデマンドの起動側 (任意トンネル): このモードでは、IBM @server iSeries を使用して、L2TP トンネルを介してリモート・システムに自動的に接続できます。アウトバウンド IP トラフィックをリモート・システムにルーティングする必要がある場合、この IBM @server iSeries サーバーで、自動的にトンネルが開始され、リモート・システムに接続されます。つまり L2TP Network Server (LNS) の働きをします。
- マルチホップ: このモード・タイプを使用すると、IBM @server iSeries サーバーで、クライアントの L2TP Access Concentrator (LAC) や L2TP Network Server (LNS) に代わって L2TP トラフィックをリダイレクトできます。L2TP マルチホップ接続を確立するために、この IBM @server iSeries サーバーは、特定の LNS に対する LAC の働きをするのと同時に、1 つ以上の LAC に対する LNS の働きもします。

A.2.7.4 ネットワーク・アドレス変換 (NAT)

ネットワークの TCP/IP アドレッシング体系が競合している場合や、ネットワーク・トポロジーのすべてまたは一部を隠蔽したい場合は、ネットワーク・アドレス変換 (NAT) 機能がソリューションを提供します。さらに、TCP/IP アドレスを隠蔽すると、1 つの TCP/IP アドレスを共用して、あるネットワーク上のすべてのコンピューターから、別のネットワーク上のサーバーにアクセスできます。ダイヤルアップ・リンクを使用して、インターネットなどの別のネットワークに接続する場合には、マスカレードが特に便利です。

A.2.8 仮想 IP アドレッシング (VIPA)

仮想 IP アドレッシングにより、IBM @server iSeries 上の物理ネットワーク・インターフェースに関連付けられない仮想 IP アドレスが作成されます。インストールされているすべての物理インターフェースを介して、ネットワークから IBM @server iSeries サーバー上のこの仮想アドレスにアクセスできます。したがって、1 つの IP アドレスを使用して複数の物理インターフェース間で負荷平準化し、大ボリュームの IBM @server iSeries

e-business サーバーのネットワーク・パフォーマンスを動的に改善できます。さらに VIPA により、IBM @server iSeries クラスタやバックアップ・システム環境内の耐障害性も改善できます。

A.2.9 アプリケーション・プログラミング・インターフェース (API)

企業には、私設ネットワークに関する固有のインターオペラビリティ要件が何度も発生します。企業が固有の要件を満たすには、独自のアプリケーションを備えなければなりません。IBM @server iSeries には、この作業を行うためのプログラミング・インターフェースが備えられています。Sockets API を使用すると、分散アプリケーション間で、データをローカルに、またはネットワークを介して交換できます。Sockets API には、コネクション型とコネクションレスの両方の通信がサポートされています。IP に加えて Sockets API を使用して、直接 Internetwork Packet Exchange (IPX) プロトコルを介して通信するアプリケーションを作成できます。

A.2.9.1 RPC

分散アプリケーションの開発者は、Remote Program Call (RPC) インターフェースも使用できます。本来、この方法ではリモート・アプリケーションは呼び出し可能プログラムとして表示されます。さらに、Java と Lotus Domino にも、TCP/IP ネットワークを介して分散アプリケーションを実装するためのさまざまなプログラミング・オプションが備えられています。

A.2.9.2 ソケットと SSL のサポート

ソケット・プログラミングでは、ソケット・アプリケーション・プログラム・インターフェース (API) を使用して、リモート・プロセスとローカル・プロセスの間の通信リンクを確立します。ソケット API は、アプリケーション層とトランスポート層の間に通信モデルとして配置されます。ソケット API を使用すると、アプリケーションにより、代表的な通信モデル上のトランスポート層やネットワーク層とのインターフェースを設けられます。ソケット API は OS/400 の一部として付属しています。ソケット API は、IBM @server iSeries サーバー上の開放型環境の一部です。ソケット API と統合ファイル・システムを使用すると、UNIX アプリケーションを IBM @server iSeries サーバーに移動する作業が簡単になります。Secure Sockets Layer (SSL) は、セキュリティ・プロトコルの一種で、開放型通信ネットワーク (インターネットなど) 上のプライバシーを確保します。SSL プロトコルを使用すると、クライアント / サーバー・アプリケーションにより、盗聴、改ざん、およびメッセージの偽造を防ぐように設計された方法で通信できます。V4R4 では、IBM @server iSeries サーバー上の多数のアプリケーションで SSL を使用できるようになっています。Telnet、HTTP サーバー、CA/400 ホスト・サーバー、システム管理、および LDAP がこれに該当します。さらに、OS/400 SSL サポートには API の集合が組み込まれており、既存の OS/400 ソケット API に加えてこれらの API も使用すると、アプリケーションで保護通信を確

立するのに必要な機能を備えることができます。SSL API や新しいグローバル・セキュア・ツールキット (GSKit) API を使用して、ソケット・アプリケーションを作成して接続を保護できます。

A.2.9.3 非同期 IOCP

TCP/IP には、OS/400 ソケット API の実装の一部として、非同期入出力完了ポート (Async IOCP) のサポートが組み込まれています。これらの新しいソケット API により、マルチスレッドのインターネットおよび e-business サーバー・アプリケーションでクライアント・データを処理する際に、拡張が容易で、効率的で、強力な機能が得られます。

インターネット、e-business、および他のネットワーク・サーバー・アプリケーションで、非同期入出力完了ポートの設計モデルと API を使用すると、他の設計モデルや API を使用する場合と比べて、クライアントから作業を処理する際のスレッドのプールが少なく済みます。代表的なスレッド化サーバー・アプリケーション設計モデルでは、普通はクライアント接続当たりのスレッドは 1 つです。対話式アプリケーション (Telnet など) のような、断続的にデータ・フローが交換されるアプリケーションの場合は特に、個々のスレッドでデータを待機する時間は長くなります。つまり、個々のスレッドでは、データが使用できない場合はスリープ状態になる CPU サイクルを消費し、最終的にデータが使用可能になった場合はウェイクアップ状態になる CPU サイクルを消費します。

アプリケーションで非同期 IOCP の設計と API を使用すると、任意のアプリケーション・スレッドにより任意のクライアントにサービスを提供できます。つまり、サーバー・アプリケーション当たりの必要なスレッド数は少なくなり、個々のスレッドは常に使用中になります。したがって、サーバー・アプリケーションのスレッドが、スリープ状態やウェイクアップ状態でデータを待機している場合、システム・リソースはほとんど消費されないか、全く消費されません。つまり、インターネット・アプリケーションや e-business アプリケーションを、非同期 IOCP API を使用するように設計して作成すると、多くの場合、CPU の使用率を大幅に減らすことができます。通常、非同期 IOCP を使用できるアプリケーションの方が、必要なシステム CPU リソース、ページング・リソース、システム・メモリー、およびシステム・ストレージの量は少なくなります。

システム上のソケットは、バークレイ・ソフトウェア・ディストリビューション (BSD) 4.3 ソケットに基づいており、これと互換可能です。C/400 や ILE RPG などの、統合言語環境プログラム (ILE) で作成されたアプリケーション・プログラムで、ソケット API を使用できます。

A.2.9.4 TLS と SSL のサポート

ほとんどの OS/400 TCP/IP サーバー・アプリケーションを SSL 用に使用できます。V4R5 より前のリリースでは、SSL V2.0 と SSL V3.0 がサポートされていました。現リリースでは、SSL サポートが拡張され、TLS プロトコルが組み込まれています。TLS のサポートは最新のプロトコルで、Secure Sockets Layer (SSL) サポートの業界標準の定義です。TLS プロトコルは、SSL バージョン 3.0 プロトコルを発展させたアップグレード・バージョンです。TLS バージョン 1 と SSL バージョン 3 で、同じ基本レコード構造と回線フローを共用できます。TLS には、SSL と同じ機能があり、SSL と互換性がありますが、プロトコル・フローに関する新しい機能と説明が組み込まれています。TLS では、SSL プロトコルの保護機能を向上させ、SSL プロトコルの仕様をより簡潔で完全なものにすることにより、SSL の定義と実装の標準化を進めています。

顧客やビジネス・パートナーは、IBM @server iSeries サーバー上で TLS サポートを使用すると、インターネット・アプリケーションでセキュリティを使用できるようにする場合に、業界の最新のテクノロジーの利点を継続的に利用できます。TLS サポートは、自動的に HTTP、FTP、および Telnet サーバーなどの SSL を使用できるアプリケーションの一部になります。SSL および GSKit API のパラメーター値を使用すると、ビジネス・パートナーや顧客が作成した SSL アプリケーションで SSL や TLS を使用できるようになります。HTTP サーバー (オリジナルおよび Apache 社の強化バージョンの両方) には、サーバー・ディレクティブがサポートされており、クライアント接続に使用できる暗号スイートやプロトコルを制御できます。

IBM @server iSeries 上で SSL/TSL サポートを使用するには、Cryptographic Access Provider 製品の 1 つ (5722-AC2 または 5722-AC3) をインストールする (暗号機能を使用可能にするため) 必要があります。OS/400 V5R1 の場合は、IBM 4758-023 Cryptographic Coprocessor カードをインストールして、SSL ハンドシェイクのパフォーマンスを向上させることもできます。

A.2.10 サポートされているアプリケーション

以下に、AS/400 プラットフォームに備えられているアプリケーションの一部について簡単に紹介します。

A.2.10.1 LDAP

OS/400 には、LDAP にアクセス可能なディレクトリー・サーバーと、他の LDAP ディレクトリー・サーバーと通信するための対応する API が備えられています。Java、C、および C++ で作成された OS/400 アプリケーション用と Windows アプリケーション用の API が備えられています。インターネット・メール・クライアントなどの LDAP を使用できるアプリケーションで、

IBM @server iSeries ディレクトリーのアクセス、更新、および管理を行います。

IBM プラットフォーム用と IBM 以外のプラットフォーム用の LDAP ディレクトリーを使用することにより、LDAP を使用してインターネットやイントラネットを介して分散情報を管理するように、OS/400 アプリケーションを開発できます。電子メール・アドレスなどの IBM @server iSeries ユーザー情報で、メール・クライアントや他の LDAP アプリケーションにアクセスできます。

ディレクトリー・サービスには SecureWay Directory for OS/400 が実装されています。これは LDAP V3 をサポートしています。LDAP V3 には内部文字 (UTF-8) のサポートが組み込まれています。UTF-8 は混合文字のマルチバイト・コード・ページで、各国語データをサポートしています。LDAP V3 は動的スキーマもサポートしています。動的スキーマは、ディレクトリー中に格納され、LDAP サーバーによって管理されます。新しい Directory Management Tool を、Lightweight Directory Access Protocol Data Interchange Format (LDIF) ファイルか LDAP コマンド行ユーティリティからインポートして使用し、このスキーマを更新できます。

OS/400 V5R1 LDAP サポートを使用すると、IBM @server iSeries プリンターや NetServer 印刷共用に関する情報を、LDAP ディレクトリー中に公開できます。公開を実行すると、ユーザーは、LDAP ディレクトリー中にある、プリンターの場所や機能などの IBM @server iSeries プリンターに関する情報を照会するアプリケーションを作成できます。またユーザーは、Windows 2000 で使用できる「Add Printer (プリンターの追加)」ウィザードを使用し、プリンターを公開するのに必要な情報を Windows 2000 アクティブ・ディレクトリーから入手できるように指定して、Windows 2000 上で直接 IBM @server iSeries プリンターを構成できます。

A.2.10.2 NFS

IBM @server iSeries の統合ファイル・システムには、ネットワーク・ファイル・システムのサポートが組み込まれています。NFS は、UNIX ベースのクライアント/サーバー・コンピューティング環境で使用される一般的なプロトコルです。NFS サーバーと NFS クライアントの間では Remote Program Calls を使用して通信されます。NFS クライアントから、ローカル・アプリケーションのリモート・ファイルにシームレスにアクセスできます。リモート・ファイルは、リモート UNIX マシン、Linux、または NFS サーバーを実行しているどのマシン上にあってもかまいません。IBM @server iSeries NFS サーバーを介してエクスポートした IBM @server iSeries ディレクトリーを、NFS クライアントをサポートしている他のシステム上にマウントできます。

A.2.10.3 Simple Mail Transfer Protocol (SMTP)

Simple Mail Transfer Protocol は、電子メールの送受信に使用されます。他の IBM @server iSeries のメール機能との整合性を保つために、SMTP とシステム・ネットワーク体系 (SNA) 配布サービス (SNADS) は AnyMail/400 を介して相互運用します。SNADS と AnyMail/400 は OS/400 の一部です。

SMTP には、SMTP 配布のメール・オブジェクト (最大 2 GB)、MIME、システム配布ディレクトリーおよび別名テーブル中への受信メール送信側の自動登録オプション、OfficeVision for AS/400 メモ、メッセージ、および接続機構がサポートされています。システム上のメール負荷に応じて SMTP を調整できるので、拡張容易性は向上します。

A.2.10.4 POP3 サーバー

Post Office Protocol (POP) サーバーは、POP3 メール・サーバーを IBM @server iSeries に実装したものです。このサーバーを使用すると、IBM @server iSeries は、POP メール・プロトコルをサポートしているクライアント (Windows、OS/2、AIX、および Macintosh 上で稼働する Netscape や Eudora などの主要な電子メール・クライアントが含まれる) に対する POP サーバーの働きをします。ユーザーは、POP サーバーを使用して、AnyMail/400 メール・サーバー・フレームワーク (OS/400 の一部) を介して、OfficeVision for AS/400 と POP の間で、メール (Multipurpose Internet Mail Extensions (MIME) メールを含む) を交換できます。

A.2.10.5 DNS と DDNS

OS/400 には DNS サーバーの全機能が組み込まれています。これは、1 次、2 次、およびキャッシングの役割用に構成できます。他のプラットフォームからの DNS 構成データを、簡単に IBM @server iSeries DNS サーバーにマイグレーションできます。また、既存の IBM @server iSeries ホストのテーブル情報を、DNS 構成データベースに移動するマイグレーション・ユーティリティも備えられています。

V5R1 では、OS/400 DNS (ドメイン・ネーム・システム) サービスが大幅に拡張されています。DNS サービスは、広く使用されている業界標準の DNS を参照して、これに基づいて実装されています。新しい動的更新機能が拡張のリストに含まれており、この機能により DNS が動的 DNS (DDNS) に変換されます。

IBM @server iSeries Dynamic Host Configuration Protocol (DHCP) サーバーも、動的 DNS 更新トランザクションを送信するように構成できるように拡張されており、IBM @server iSeries では、これらを組み合わせて、ネットワーク上で TCP/IP アドレスや関連した DNS ホスト名を自動的に管理する統合動的 IP ソリューションをサポートしています。

OS/400 V5R1 を使用している場合は、BIND バージョン 4.9.3 か BIND バージョン 8.2.3 のどちらかに基づいて DNS サーバーを実行するよう選択できます。V5R1 では、DNS サーバー (BIND 8.2.3) は、UNIX アプリケーションとしてポータブル・アプリケーション・ソリューション環境 (PASE) の下で稼働します。PASE には、UNIX アプリケーションのランタイム環境が備えられており、IBM @server iSeries の基本プロセッサが OS/400 RISC モードから UNIX RISC モードに切り替えられます。この実装内容を使用して、BIND の新規バージョンが入手可能になった時点で即時に OS/400 に実装できます。

A.2.10.6 DHCP

DHCP をデプロイしてすべての TCP/IP ワークステーション構成タスクを集中制御すると、TCP/IP ネットワークの管理コストを大幅に減らせます。DHCP は、Windows 95/NT、UNIX、および IBM Network Station などの一般的なワークステーションのほとんどで固有にサポートされている標準プロトコルです。DHCP を使用すると、新しいワークステーションをネットワークに追加する際に、すべての IP 構成データ (IP アドレス、サブネット・マスク、デフォルトのルーターなど) を動的に割り当てることができます。さらに、ワークステーションをネットワークから除去する際に、DHCP によりネットワーク・リソースを自動的にリカバリーしたりリサイクルしたりできます。これらの機能により、手動でのワークステーション構成という、時間を浪費したりエラーになったりする作業がなくなります。

OS/400 には、直感的に分かる形式の GUI 管理インターフェースを含む、DHCP サーバーの全機能が組み込まれています。OS/400 には DHCP リレー・エージェント (BOOTP リレー・エージェントともいう) も付属しており、これをデプロイすると DHCP 要求を複数のサブネットワークから 1 つ以上の中央 DHCP サーバーにルーティングできます。

A.2.10.7 Telnet クライアントおよびサーバー

Telnet プロトコルを使用すると、Telnet クライアントのワークステーションからリモート・システムにローカルに接続するかのように、システム (Telnet クライアント) からリモート・システム (Telnet サーバー) のリソースにアクセスして使用できます。IBM @server iSeries の Telnet には、Telnet クライアント機能と Telnet サーバー機能が備えられています。Telnet プロトコルには、クライアントとサーバーの間で、Telnet 接続の操作特性を制御するオプションをネゴシエーションする機構が備えられています。この種のネゴシエーションには、多数の機能がありますが、クライアントとサーバーの両方でサポートされている端末タイプのうち最善のものを判断する機能も含まれています。

ネゴシエーションされた端末タイプに応じて、IBM @server iSeries の Telnet クライアントでは 3270、5250、VT100、または VT220 のフルスクリーン・モードが稼働します。IBM @server iSeries の Telnet サーバーで

は、ASCII 行モードか、3270、5250、または VT100 のフルスクリーン・モードで稼動します。Telnet セッションで使用できる機能は、操作モードによって異なります。セキュリティー機能と自動化機能は、IBM @server iSeries Telnet 5250 サーバーに組み込まれています。

登録されている Telnet サーバーは、セッションの初期化時およびセッション終了時に終了します。このように終了するために必要なことは、顧客作成のユーザー出口プログラムを適切な出口点に登録することだけです。接続している Telnet クライアント・エミュレーターには変更を加える必要がないので、既存のクライアントにとってこの機能は即時に役立ちます。

接続しているクライアント（または登録されている Telnet サーバー出口プログラム）で仮想装置を選択すると、優先のサブシステムに一般性の高いジョブ・ルーティングが備えられるので、関連したワーク・マネージメントを調整できます。仮想装置を選択すると、優先コード・ページ、文字セット、およびキーボードの属性をセッション単位で確立できるので、各国語サポートの柔軟性を高めることができます。

プリンター・パススルー・サポートは、TCP/IP 環境用の追加プリンターをサポートする 2 つの新しい端末 (IBM 3812-1 と IBM 5553-B01) で構成されています。このサポートを使用すると、Telnet サーバーにより、クライアントで拡張ネゴシエーションを介したり初期化出口プログラムを割り当てたりして、仮想プリンター装置を動的に作成または選択する（およびその両方を実行する）際の柔軟性を備えることができます。

Telnet サーバーには、SSL を介した保護 Telnet セッションがサポートされています。Telnet サーバーには、デジタル証明書による、サーバーだけの認証と、サーバーとクライアントの両方の認証がサポートされています。

A.3 Linux

Linux オペレーティング・システムは、小規模の（小さすぎる場合は含まれない）Intel システムから大規模な zSeries サーバーに至るまで、広範囲のプラットフォームで使用できます。Linux が人気のある理由の多くは、可用性があることと、開放型のプロトコルや規格を採用しているに基づく開放性によります。その結果、世界中の開発者が基本的なカーネルのソースに貢献し、既存の多数の Linux の配布先に送っています。

このタイプの開発環境を使用すると、Linux に最新の TCP/IP 規格を備えることができます。多くの場合、Linux 中へのプロトコルの実装は、最初はプロトコルの実行可能性を証明するという方法で行われます。Linux に備えられている最も注目すべき機能の 1 つに、Linux ファイアウォールがあり、これは NAT サポートも備えています。

A.3.1 Linux ファイアウォール

Linux カーネルは IP マスカレードをサポートしています。このサポートを使用すると、インターネットに対するモデム接続を他の複数のコンピューターと共用できます。この拡張機能の 1 つに VPN マスカレードがあり、これには IPsec ベースと PPTP ベースのサポートが追加されています。

カーネルにより、接続先が追跡されます。この機能のことを `ipchains` サポートといいます。したがって、Linux ファイアウォールは主に Linux カーネル中にあります。`ipchains` コマンドを使用すると、IP アドレスと指示に基づいて、通常の接続や VPN 接続の許可 / 拒否 / マスカレードを行えます。

詳細については、以下のサイトにある VPN Masquerade howto ファイルを参照してください。

<http://www.redhat.com/mirrors/LDP/HOWTO/VPN-Masquerade-HOWTO-3.html>

A.4 ネットワーク・コンピューター

作業員の移動に伴う問題を解決する別の方法として、ユーザーのマシンを企業のさまざまな場所に移動させるという問題に取り組むのではなく、企業のさまざまな場所に固定されたワークステーションを配置し、すべてのユーザーがどこからでも自分のデータやアプリケーションにアクセスできるようにするという方法があります。これはネットワーク・コンピューターの原理です。

一般用語では、ネットワーク・コンピューターとは、普通は以下のものを含むマイクロプロセッサ・ベースのシステムのことです。

- ローカル・メモリー (NVRAM)
- ディスクはなし (ハード・ディスクもディスクレットもなし)
- 複数の形式のネットワーク接続
- 端末エミュレーション機能
- 以下を含む IP ベースのプロトコル
 - TCP
 - FTP
 - Telnet
 - NFS
- Web ブラウザー
- Java 仮想マシン

要するに、ネットワーク・コンピュータは 1 つの高性能な端末と見なすことができます。

ネットワーク・コンピュータに必要なソフトウェアは、オペレーティング・システムも含めて、ネットワーク中のサーバーに格納され、ネットワーク・コンピュータの電源がオンになったりユーザーが新しい機能をアクティブにした時点で、オンデマンドでダウンロードされます。

ネットワーク・コンピュータの独自のローカル NVRAM (ブート・モニター・プログラム) には、サーバーに対する BOOTP 接続か DHCP 接続を行うのに必要なコードだけが格納されており、オペレーティング・システム・カーネルをダウンロードできるようになっています。その後、実装や構成の内容に応じて、TFTP か NFS を使用して、オペレーティング・システムの残りの部分がダウンロードされます。

通常は、ネットワーク・コンピュータの電源をオンにすると、ログオン・パネルが表示されます。ログオンが正常に完了すると、ユーザー独自のアプリケーション・アイコンの設定を含む個別設定のデスクトップが表示されます。これらのアイコンのいずれかを選択すると、サーバーからアプリケーションがダウンロードされます。(数種類のサーバーにアクセスできます。)

ネットワーク・コンピュータのすべての接続には、TCP/IP プロトコルが使用されます。アプリケーションをダウンロードする場合には NFS か TFTP が使用されますが、アプリケーション自体では、通信する必要がある場合は TCP/IP が使用されます。たとえば、SNA エミュレーターでは TN3270 か TN5250 のうち該当する方が使用されます。

通常は、ネットワーク・コンピュータは Intel プロセッサに基づいていませんが、ICA (独立コンソール・アーキテクチャー) クライアントをロードすると、Windows Terminal Server または Citrix Systems Metaframe 上で Windows アプリケーションにアクセスできます。この場合、Windows アプリケーションの処理は NT サーバー自体で実行され、画面変更だけがクライアントに送達されます。X Window サーバーをダウンロードすると、ネットワーク・コンピュータで、リモートに UNIX ベースのシステム上でアプリケーションを実行できます。

サーバー上に格納されているネットワーク・コンピュータのソフトウェアすべてを一元管理し、オペレーティング・システムやアプリケーション・コードの更新を単純化できます。

ネットワーク・コンピュータを実装した IBM 製品は、IBM Network Station で、これは Windows NT、AIX、OS/400、OS/390、および OS/2 サーバーからブートしたり管理したりできます。IBM Network Station の詳細については、以下の IBM Network Station ホーム・ページを参照してください。

<http://www.pc.ibm.com/us/netvista/thinclient.html>

付録 B 特記事項

本書の目的は、TCP/IP 製品で提供されているプログラミング・インターフェースの仕様を提供することではありません。

本書において、IBM 製品、プログラム、またはサービスについて言及または説明する場合があります。しかし、このことは、IBM が営業を行っているすべての国においてこのような製品、プログラム、またはサービスが利用可能であることを必ずしも示すものではありません。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。IBM 製品、プログラムまたはサービスに代えて、IBM の知的所有権を侵害することのない機能的に同等のプログラムを使用することができます。

本書の中にある情報は、指定する機器を使用して開発されました。従ってその適用は特定のハードウェアとソフトウェア・プロダクトとレベルに限定されます。

IBM は、本書に記載されている内容に関して特許権（特許出願中のものを含む）を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。使用許諾については、下記の宛先に書面にてご照会ください。

〒106-0032 東京都港区六本木 3-2-31

IBM World Trade Asia Corporation

Licensing

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム（本プログラムを含む）との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation, Dept. 600A, Mail Drop 1329, Somers,
NY 10589 USA

本プログラムに関する上記の情報は、適切な使用条件の下で 사용할ことができますが、有償の場合もあります。

本書に含まれる情報は、IBM の正式なテストを受けておらず、現存するままの状態 で配布されます。この情報の利用またはこうした技法の導入は、お客様の責任であるとともに、これを評価しお客様の稼働環境への統合するお客様の能力に依存します。個々の項目は、特定の状況における正確性について IBM によって検討されていますが、全く同一または同様な結果が得られる保証はありません。お客様自身の環境にこれらの技法を適用しようとする場合は、お客様自身のリスクにおいて行っていただきます。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。

以下は、IBM Corporation の商標です。

AIX	AnyNet
Application System/400	APPN
AS/400	AT
C/400	CICS
C/370	CT
Current	DB2
Domino	DPI
e (ロゴ) [®] 	ESCON
IBM [®]	IBM.COM
IBM Global Network	Integrated Language Environment
IPDS	Language Environment
Lotus	Lotus Notes
MVS/ESA	Netfinity
Network Station	Notes
Redbooks	Redbooks Logo 
OfficeVision	Operating System/2
OS/2	OS/390
OS/400	PAL
Parallel Sysplex	Presentation Manager
RACF	RISC System/6000
RS/6000	SecureWay
S/390	System/390
SPVTAM	WebSphere
Wave	
XT	

以下は、それぞれ各社の商標または登録商標です。

Tivoli、Manage. Anything. Anywhere.、The Power To Manage.、Anything.Anywhere.、TME、NetView、Cross-Site、Tivoli Ready、Tivoli Certified、Planet Tivoli、および、Tivoli Enterprise は IBM corporation の商標です。

C-bus は、Corollary Inc. の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

PC Direct は、Ziff Communications Company の米国およびその他の国における商標であり、IBM Corporation がライセンスを得て使用しています。

ActionMedia、LANDesk、MMX、Pentium、および ProShare は、Intel Corporation の米国およびその他の国における商標です。

UNIX は、The Open Group がライセンスしている米国およびその他の国における登録商標です。

SET、SET ロゴ、SET Secure Electronic Transaction および Secure Electronic Transaction は、SET Secure Electronic Transaction LLC の商標です。

他の会社名、製品名およびサービス名などはそれぞれ各社の商標または登録商標です。

付録 C 関連資料

以下にリストされている資料は、本書で取り上げられているトピックの詳細について参照するのに特に適しているはずです。

C.1 IBM のレッドブック

- *Administering DCE and DFS 2.1 for AIX (and OS/2 Clients)*、SG24-4714
- *AS/400 Mail: Multiple SMTP Domains Behind a Firewall*、SG24-5643
- *AS/400 TCP/IP Autoconfiguration: DNS and DHCP Support*、SG24-5147
- *Beyond DHCP - Work Your TCP/IP Internetwork with Dynamic IP*、SG24-5280
- *A Comprehensive Guide to Virtual Private Networks, Volume I: IBM Firewall, Server and Client Solutions*、SG24-5201
- *Data Sharing: Using the OS/390 Network File System*、SG24-5262
- *DCE Cell Design Considerations*、SG24-4746
- *IBM Communications Server for OS/390 TCP/IP Implementation Guide, Volume 1*、SG24-5227
- *IBM Communications Server for OS/390 V2R10 TCP/IP Implementation Guide, Volume 2*、SG24-5228
- *IBM Router Interoperability and Migration Examples*、SG24-5865
- *IP Network Design Guide*、SG24-2580
- *Internetworking over ATM: An Introduction*、SG24-4699
- *Local Area Network Concepts and Products: Routers and Gateways*、SG24-4755
- *Load-Balancing Internet Servers*、SG24-4993
- *Managing OS/390 TCP/IP with SNMP*、SG24-5866
- *Migrating Subarea Networks to an IP Infrastructure Using Enterprise Extender*、SG24-5957
- *MPTN Architecture Tutorial and Product Implementations*、SG24-4170
- *OS/390 eNetwork Communications Server V2R7 TCP/IP Implementation Guide, Volume 3*、SG24-5229
- *OSA-Express Implementation Guide*、SG24-5948
- *Secure e-business in TCP/IP Networks on OS/390 and z/OS*、SG24-5383

- *Security on the Web Using DCE Technology*, SG24-4949
- *The Technical Side of Being an Internet Service Provider*, SG24-2133
- *TCP/IP in a Sysplex*, SG24-5235
- *Understanding LDAP*, SG24-4986
- *Understanding OSF DCE 1.1 for AIX and OS/2*, SG24-4616
- *Understanding Optical Communications*, SG24-5230

C.2 IBM レッドブック・コレクション

以下の CD-ROM 上のレッドブックも使用できます。すべての CD-ROM の内容、更新状況、および形式に関する情報は、ibm.com/redbooks の、CD-ROM のボタンをクリックしてください。

CD-ROM のタイトル	コレクション ・キットの番号
IBM System/390 Redbooks Collection	SK2T-2177
IBM Networking Redbooks Collection	SK2T-6022
IBM Transaction Processing and Data Management Redbooks Collection	SK2T-8038
IBM Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
IBM AS/400 Redbooks Collection	SK2T-2849
IBM Netfinity Hardware and Software Redbooks Collection	SK2T-8046
IBM RS/6000 Redbooks Collection	SK2T-8043
IBM Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

C.3 その他のリソース

以下の資料も、追加の情報源として関係があります。

- *iSeries Handbook*, GA19-5486
- *LAN Technical Reference: 802.2 and NetBIOS Application Program Interfaces*, SC30-3587
- *Network and e-business Products Reference*, GX28-8002
- Albitz, et al., *DNS and BIND, Fourth Edition*, O'Reilly & Associates, Inc., 2001, ISBN 0-596-00158-4
- Bates, *Broadband Telecommunications Handbook*, McGraw-Hill Professional Publishing, 1999, ISBN 0-0713-46481
- Black, *IP Routing Protocols: RIP, OSPF, BGP, PNNI and Cisco Routing Protocols*, Prentice Hall, 2000, ISBN 0-130-14248-4

- Comer, *Internetworking with TCP/IP, Volume I, Principles, Protocols and Architecture, Fourth Edition*, Prentice-Hall, Inc., 2000, ISBN 0-13-018380-6
- Comer, et al., *Internetworking with TCP/IP, Volume II, Design, Implementation, and Internals, Third Edition*, Prentice-Hall, Inc., 1998, ISBN 0-13-973843-6
- Comer et al., *Internetworking with TCP/IP, Volume III, Client Server Programming and Applications, Windows Sockets Version*, Prentice-Hall, Inc., 1997, ISBN 0-13-848714-6
- Costales, et al., *sendmail, Second Edition*, O'Reilly & Associates, Inc., 1997, ISBN 1-56592-222-0
- Cypser, *Communications for Cooperating Systems - OSI, SNA and TCP/IP*, Addison-Wesley, Publishing Company, Inc., 1992, ISBN 0-201-50775-7
- December, et al., *HTML 3.2 & CGI Unleashed*, Sams.net Publishing, 1996, ISBN 1-57521-177-7
- Halabi, et al., *Internet Routing Architectures, Second Edition*, Cisco Press, 2000, ISBN 1-57870-233-X
- Huitema, *IPv6: The New Internet Protocol, Second Edition*, Prentice-Hall, Inc., 1998, ISBN 0-13-850505-5
- Hunt, et al., *TCP/IP Network Administration, Second Edition*, O'Reilly & Associates, Inc., 1995, ISBN 1-56592-322-7
- Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, second edition, John Wiley & Sons, Inc., 1995, ISBN 0-471-11709-9
- Stevens, *TCP/IP Illustrated, Volume I, The Protocols*, Addison Wesley, 1994, ISBN 0-201-63346-9
- Stevens, et al., *TCP/IP Illustrated, Volume II, The Implementation*, Addison Wesley, 1995, ISBN 0-201-63354-X
- Stevens, et al., *TCP/IP Illustrated, Volume III, TCP for Transactions, HTTP, NNTP, and the UNIX Domain Protocols*, Addison Wesley, 1996, ISBN 0-201-63495-3
- Thomas, *IPng and the TCP/IP Protocols*, John Wiley & Sons, Inc., 1996, ISBN 0-471-13088-5
- Wireless Application Protocol Forum Ltd, *Official Wireless Application Protocol: The Complete Standard with Searchable CD-ROM*, John Wiley & Sons, 1999, ISBN 0-471-32755-7
- Zwicky, et al., *Building Internet Firewalls, Second Edition*, O'Reilly & Associates, Inc., 2000, ISBN 1-565-92871-7

C.4 参照 Web サイト

以下の Web サイトも、追加の情報源として関係があります。

- <http://www.ietf.org/>
- <http://www.iana.org/>
- <http://www.irtf.org/>
- <http://www.isoc.org/>
- <http://www.iab.org/>

省略語と頭字語

AAA	認証、許可、およびアカウント リング (Authentication, Authorization and Accounting)	ASN.1	抽象構文記法 1 (Abstract Syntax Notation 1)
AAL	ATM アダプテーション・レイヤー (ATM Adaptation Layer)	AS/400	Application System/400
AFS	Andrews File System	ATM	非同期伝送モード (Asynchronous Transfer Mode)
AH	認証ヘッダー (Authentication Header)	BGP	Border Gateway Protocol
AIX	拡張対話式エグゼクティブ・オペレーティング・システム (Advanced Interactive Executive Operating System)	BIND	バークレイ・インターネット・ネーム・ドメイン (Berkeley Internet Name Domain)
API	アプリケーション・プログラミング・インターフェース (Application Programming Interface)	BNF	バックス正規形式 (Backus-Naur Form)
APPN	拡張対等通信ネットワークワーキング機能 (Advanced Peer-to-Peer Networking)	BRI	基本インターフェース (Basic Rate Interface)
ARP	Address Resolution Protocol	BSD	バークレイ・ソフトウェア・ディストリビューション (Berkeley Software Distribution)
ARPA	Advanced Research Projects Agency	CA	認証局 (Certification Authority)
AS	自律システム (Autonomous System)	CBC	暗号化ブロック・チェーン (Cipher Block Chaining)
ASCII	情報交換用米国標準コード (American Standard Code for Information Interchange)	CCITT	国際電信電話諮問委員会 (Comit・Consultatif International TM Hraphique et TM Qhonique) (現在の ITU-T)
		CDMF	Commercial Data Masking Facility
		CERN	Conseil Européen pour la Recherche Nucléaire
		CGI	コモン・ゲートウェイ・インターフェース (Common Gateway Interface)

CHAP	Challenge Handshake Authentication Protocol		Circuit-terminating Equipment)
CICS	顧客情報管理システム (Customer Information Control System)	DDN	国防データ・ネットワーク (Defense Data Network)
CIDR	無クラス・ドメイン間ルーティング (Classless Inter-Domain Routing)	DDNS	Dynamic Domain Name System
CIX	商用インターネット交換点 (Commercial Internet Exchange)	DEN	ディレクトリー対応ネットワーク (Directory-Enabled Networking)
CLNP	Connectionless Network Protocol	DES	Digital Encryption Standard
CORBA	共通オブジェクト・リクエスト・ブローカー・アーキテクチャ (Common Object Request Broker Architecture)	DFS	分散ファイル・サービス (Distributed File Service)
COS	サービス・クラス (Class of Service)	DHCP	動的ホスト構成プロトコル (Dynamic Host Configuration Protocol)
CPCS	Common Part Convergence Sublayer	DLC	データ・リンク制御 (Data Link Control)
CPU	中央演算処理装置 (Central Processing Unit)	DLCI	データ・リンク接続 ID (Data Link Connection Identifier)
CSMA/CD	キャリア・センス多重アクセス / 衝突検出 (Carrier Sense Multiple Access with Collision Detection)	DLL	ダイナミック・リンク・ライブラリー (Dynamic Link Library)
DARPA	国防総省国防高等研究事務局 (Defense Advanced Research Projects Agency)	DLSw	データ・リンク・スイッチ (Data Link Switching)
DCE	分散コンピューティング環境 (Distributed Computing Environment)	DLUR	従属 LU リクエスター (Dependent LU Requester)
DCE	データ回線終端装置 (Data	DLUS	従属 LU サーバー (Dependent LU Server)
		DME	分散管理環境 (Distributed Management Environment)
		DMI	Desktop Management Interface

DMTF	Desktop Management Task Force	ESCON	エンタープライズ・システム接続 (Enterprise Systems Connection)
DMZ	非武装地帯 (Demilitarized Zone)	ESP	暗号ペイロード (Encapsulating Security Payload)
DNS	ドメイン・ネーム・サーバー (Domain Name Server)	FDDI	光ファイバー分散データ・インターフェース (Fiber Distributed Data Interface)
DOD	米国国防総省 (U.S. Department of Defense)	FQDN	完全修飾ドメイン・ネーム (Fully Qualified Domain Name)
DOI	変換処理用ドメイン (Domain of Interpretation)	FR	フレーム・リレー (Frame Relay)
DOS	ディスク・オペレーティング・システム (Disk Operating System)	FTP	ファイル転送プロトコル (File Transfer Protocol)
DSA	デジタル署名アルゴリズム (Digital Signature Algorithm)	GGP	ゲートウェイ間プロトコル (Gateway-to-Gateway Protocol)
DSAP	宛先サービス・アクセス・ポイント (Destination Service Access Point)	GMT	グリニッジ標準時 (Greenwich Mean Time)
DSS	デジタル署名規格 (Digital Signature Standard)	GSM	Group Special Mobile
DTE	データ端末装置 (Data Terminal Equipment)	GUI	グラフィカル・ユーザー・インターフェース (Graphical User Interface)
DTP	データ転送プロセス (Data Transfer Process)	HDLC	ハイレベル・データ・リンク制御 (High-level Data Link Control)
DVMRP	Distance Vector Multicast Routing Protocol	HMAC	ハッシュ・メッセージ確認コード (Hashed Message Authentication Code)
EBCDIC	拡張 2 進 10 進コード (Extended Binary Communication Data Interchange Code)	HPR	高性能ルーティング (High Performance Routing)
EGP	Exterior Gateway Protocol	HTML	ハイパーテキスト・マークアップ言語

	(Hypertext Markup Language)	IGP	Interior Gateway Protocol
HTTP	Hypertext Transfer Protocol	IIGP	Internet Inter-ORB Protocol
IAB	インターネット・アクティビティー・ボード (Internet Activities Board)	IKE	Internet Key Exchange
		IMAP	Internet Message Access Protocol
IAC	コマンドとして解釈する (Interpret As Command)	IMS	情報管理システム (Information Management System)
IANA	InternetAssigned Numbers Authority	IP	インターネット・プロトコル (Internet Protocol)
IBM	International Business Machines Corporation	IPC	プロセス間通信 (Interprocess Communication)
ICMP	Internet Control Message Protocol	IPSec	IP Security Architecture
ICSS	Internet Connection Secure Server	IPv4	インターネット・プロトコル バージョン 4 (Internet Protocol Version 4)
ICV	整合性検査値 (Integrity Check Value)		
IDEA	国際データ暗号化アルゴリズム (International Data Encryption Algorithm)	IPv6	インターネット・プロトコル バージョン 6 (Internet Protocol Version 6)
IDLC	統合データ・リンク制御 (Integrated Data Link Control)	IPX	Internetwork Packet Exchange
IDRP	Inter-Domain Routing Protocol	IRFT	Internet Research TaskForce
IEEE	米国電気電子学会 (Institute of Electrical and Electronics Engineers)	ISAKMP	Internet Security Association and Key Management Protocol
IESG	Internet Engineering Steering Group	ISDN	サービス総合ディジタル網 (Integrated Services Digital Network)
IETF	Internet Engineering Task Force	ISO	国際標準化機構 (International Organization for Standardization)
IGMP	Internet Group Management Protocol		
IGN	IBM Global Network	ISP	インターネット・サービス・プロバイダー

	(Internet Service Provider)	LLC	論理リンク層 (Logical Link Layer)
ITSO	International Technical Support Organization	LNS	L2TP ネットワーク・サーバー (L2TP Network Server)
ITU-T	国際電気通信連合 - Telecommunication Standardization Sector (International Telecommunication Union - Telecommunication Standardization Sector) (旧 CCITT)	LPD	ライン・プリンター・デーモン (Line Printer Daemon)
		LPR	ライン・プリンター・リクエスト (Line Printer Requester)
IV	初期設定ベクトル (Initialization Vector)	LSAP	リンク・サービス・アクセス・ポイント (Link Service Access Point)
JDBC	Java Database Connectivity	L2F	Layer 2 Forwarding
		L2TP	Layer 2 Tunneling Protocol
JDK	Java 開発キット (Java Development Toolkit)	MAC	メッセージ確認コード (Message Authentication Code)
JES	ジョブ入カシステム (Job Entry System)		
JIT	Java Just-in-Time コンパイラ (Java Just-in-Time Compiler)	MAC	メディア・アクセス制御 (Medium Access Control)
JMAPI	Java 管理 API (Java Management API)	MD2	RSA メッセージ・ダイジェスト 2 アルゴリズム (RSA Message Digest 2 Algorithm)
JVM	Java 仮想マシン (Java Virtual Machine)		
JPEG	Joint Photographic Experts Group	MD5	RSA メッセージ・ダイジェスト 5 アルゴリズム (RSA Message Digest 5 Algorithm)
LAC	L2TP Access Concentrator	MIB	管理情報ベース (Management Information Base)
LAN	ローカル・エリア・ネットワーク (Local Area Network)	MILNET	軍事ネットワーク (Military Network)
LAPB	Link Access Protocol Balanced	MIME	Multipurpose Internet Mail Extensions
LCP	Link Control Protocol		
LDAP	Lightweight Directory Access Protocol	MLD	マルチキャスト・リスナー・ディスカバリー (Multicast Listener Discovery)
LE	LAN エミュレーション (LAN Emulation) (ATM)		

MOSPF	Multicast Open Shortest Path First	NCP	Network Control Protocol
MPC	マルチパス・チャネル (Multi-Path Channel)	NCSA	National Computer Security Association
MPEG	Moving Pictures Experts Group	NDIS	ネットワーク・ドライバ ー・インターフェ ース仕様 (Network Driver Interface Specification)
MPLS	マルチプロトコル・ラ ベル交換 (Multiprotocol Label Switching)	NETBIOS	ネットワーク基本入出 力システム (Network Basic Input/Output System)
MPOA	ATM を介したマルチプ ロトコル (Multiprotocol over ATM)	NFS	ネットワーク・ファイ ル・システム (Network File System)
MPTN	マルチプロトコル・ト ランспорт・ネット ワーク (Multiprotocol Transport Network)	NIC	ネットワーク情報セン ター (Network Information Center)
MS-CHAP	Microsoft Challenge Handshake Authentication Protocol	NIS	Network Information System
MTA	メッセージ転送エー ジェント (Message Transfer Agent)	NIST	米国連邦情報・技術局 (National Institute of Standards and Technology)
MTU	最大伝送単位 (MTU: Maximum Transmission Unit)	NMS	ネットワーク管理ス テーション (Network Management Station)
MVS	多重仮想記憶オペレー ティング・システム (Multiple Virtual Storage Operating System)	NNTP	Network News Transfer Protocol
NAT	ネットワーク・アドレ ス変換 (Network Address Translation)	NRZ	非ゼロ復帰 (Non-Return-to-Zero)
NBDD	NETBIOS データグラ ム・ディストリビュー ター (NetBIOS Datagram Distributor)	NRZI	非ゼロ復帰反転 (Non-Return-to-Zero Inverted)
NBNS	NETBIOS ネーム・サー バー (NetBIOS Name Server)	NSA	国家安全保障局 (National Security Agency)
NCF	Network Computing Framework	NSAP	ネットワーク・サービ ス・アクセス・ポイン ト (Network Service Access Point)

NSF	国立科学財団 (National Science Foundation)		Assembler/ Disassembler)
NTP	Network Time Protocol	PAP	Password Authentication Protocol
NVT	ネットワーク仮想端末 装置 (Network Virtual Terminal)	PDU	プロトコル・データ単 位 (Protocol Data Unit)
ODBC	Open Database Connectivity	PGP	Pretty Good Privacy
ODI	オープン・データ・リ ンク・インターフェー ス (Open Datalink Interface)	PI	プロトコル・インター プリター (Protocol Interpreter)
OEM	相手先商標製造会社 (Original Equipment Manufacturer)	PIM	プロトコル独立マルチ キャスト (Protocol Independent Multicast)
ONC	オープン・ネットワー ク・コンピューティン グ (Open Network Computing)	PKCS	Public Key Cryptosystem
ORB	オブジェクト・リクエ スト・ブローカー (Object Request Broker)	PKI	Public Key Infrastructure
OSA	Open Systems Adapter	PNNI	私設ネットワーク間イ ンターフェース (Private Network-to-Network Interface)
OSI	オープン・システム間 相互接続 (Open Systems Interconnect)	POP	Post Office Protocol
OSF	オープン・ソフトウェ ア・ファウンデーショ ン (Open Software Foundation)	POP	存在点 (Point-of-Presence)
OSPF	Open Shortest Path First	PPP	Point-to-Point Protocol
OS/2	オペレーティング・シ ステム /2	PPTP	Point-to-Point Tunneling Protocol
OS/390	Operating System for the System/390 platform	PRI	Primary Rate Interface
OS/400	Operating System for the AS/400 platform	PSDN	パケット交換データ網 (PSDN: Packet Switching Data Network)
PAD	パケット組み立て / 分 解機能 (PAD: Packet	PSTN	公衆交換電話網 (Public Switched Telephone Network)
		PVC	パーマネント・バー チャル・サーキット (Permanent Virtual Circuit)

QLLC	修飾論理リンク制御 (Qualified Logical Link Control)	RSH	リモート・シェル (Remote Shell)
QoS	サービス品質 (Quality of Service)	RSVP	Resource Reservation Protocol
RACF	リソース・アクセス管 理機能 (Resource Access Control Facility)	RS/6000	IBM RISC システム /6000 (IBM RISC System/6000)
RADIUS	Remote Authentication Dial-In User Service	RTCP	Realtime Control Protocol
RAM	ランダム・アクセス・ メモリー (Random Access Memory)	RTP	Realtime Protocol
RARP	Reverse Address Resolution Protocol	SA	セキュリティ・アソ シエーション (Security Association)
RAS	リモート・アクセス・ サービス (Remote Access Service)	SAP	サービス・アクセス・ ポイント (Service Access Point)
RC2	RSA Rivest 暗号 2 アル ゴリズム (RSA Rivest Cipher 2 Algorithm)	SDH	同期ディジタル階層 (Synchronous Digital Hierarchy)
RC4	RSA Rivest 暗号 4 アル ゴリズム (RSA Rivest Cipher 4 Algorithm)	SDLC	同期データ・リンク制 御 (Synchronous Data Link Control)
REXEC	遠隔実行プロトコル (Remote Execution Command Protocol)	SET	Secure Electronic Transaction
RFC	Request For Comments	SGML	Standard Generalized Markup Language
RIP	Routing Information Protocol	SHA	セキュア・ハッシュ・ アルゴリズム (Secure Hash Algorithm)
RIPE	Réseaux IP Européens	S-HTTP	Secure Hypertext Transfer Protocol
RISC	縮小命令セット・コン ピューター (Reduced Instruction-Set Computer)	SLA	サービス・レベル・ア グリーメント (Service Level Agreement)
ROM	読み取り専用メモリー (Read-only Memory)	SLIP	Serial Line Internet Protocol
RPC	リモート・プロシー ジャー・コール (Remote Procedure Call)	SMI	管理情報の構造 (Structure of Management Information)
		S-MIME	Secure Multipurpose Internet Mail Extension

SMTP	Simple Mail Transfer Protocol	TCP/IP	伝送制御プロトコル / インターネット・プロトコル (Transmission Control Protocol/Internet Protocol)
SNA	システム・ネットワーク体系 (System Network Architecture)		
SNAP	Subnetwork Access Protocol	TFTP	Trivial File Transfer Protocol
SNG	Secured Network Gateway (IBM eNetwork Firewall の旧製品名)	TLPB	トランスポート層プロトコル境界 (Transport-Layer Protocol Boundary)
SNMP	Simple Network Management Protocol	TLS	TransportLayer Security
SOA	Start of Authority	TOS	Type of Service
SONET	同期光学式ネットワーク (Synchronous Optical Network)	TRD	伝送ルーティング・ドメイン (Transit Routing Domain)
SOCKS	SOCK-et-S (リリース後に残る NEC 内部開発名)	TTL	存続時間 (Time to Live)
SPI	Security Parameter Index	UDP	User Datagram Protocol
SSL	Secure Sockets Layer	UID	固有 ID (Unique Identifier)
SSAP	ソース・サービス・アクセス・ポイント (Source Service Access Point)	URI	Uniform Resource Identifier
SSP	Switch-to-Switch Protocol	URL	Uniform Resource Locator
SSRC	同期ソース (Synchronization Source)	UT	世界時 (Universal Time)
SVC	スイッチド・バーチャル・サーキット (Switched Virtual Circuit)	VC	バーチャル・サーキット (Virtual Circuit)
TACACS	Terminal Access ControllerAccess Control System	VM	仮想計算機オペレーティング・システム (Virtual Machine Operating System)
TCP	伝送制御プロトコル (Transmission Control Protocol)	VPN	仮想プライベート・ネットワーク (Virtual Private Network)
		VRML	バーチャル・リアリティ・モデル化言語 (Virtual Reality Modeling Language)

VRRP	Virtual Router Redundancy Protocol	X.509	ITU および ISO のデジタル証明書規格 (ITU and ISO Digital Certificate Standard)
VTAM	仮想記憶通信アクセス方式 (Virtual Telecommunications Access Method)	3DES	
WAE	ワイヤレス・アプリケーション環境 (Wireless Application Environment)		Triple Digital Encryption Standard
WAP	Wireless Application Protocol		
WSP	Wireless Session Protocol		
WTP	Wireless Transaction Protocol		
WAN	広域ネットワーク (Wide Area Network)		
WWW	ワールド・ワイド・ウェブ (World Wide Web)		
XDR	外部データ表現 (External Data Representation)		
XML	Extensible Markup Language		
X11	X Window システムバージョン 11 (X Window System Version 11)		
X.25	CCITT パケット交換規格 (CCIT Packet Switching standard)		
X.400	CCITT および ISO のメッセージ処理サービス規格 (CCITT and ISO Message-handling Service Standard)		
X.500	ITU および ISO のディレクトリー・サービス規格 (ITU and ISO Directory Service Standard)		

索引

数字

2210 668

2210/2216 Router

Enterprise Extender 667

High Performance Routing (HPR) 667

帯域幅予約システム (BRS) 668

2216 668

6Bone 635

A

A5 アルゴリズム 690

AAA セキュリティー・モデル 796

Abstract Syntax Notation One (ASN.1) 546

Address Resolution Protocol (ARP)

BOOTP 126

IEEE 802.2 標準 31

IEEE 802.x 標準 119

Proxy-ARP 122

アルゴリズム 122

イーサネット 31, 119

応答 118

キャッシュ 118

パケット 119, 124

フレーム・リレー 43

プロキシー ARP 666

無償の ARP 666

モバイル IP 665

要求 118

AIX 312

Andrew File System (AFS) 378

Kerberos 403

アクセス制御リスト (ACL) 403

管理ユーティリティー 403

クライアント・キャッシュ 403

セル 402

認証 403

ネーム・スペース 402

複製 403

リモート・プロシージャ・コール (RPC)
403

Apache 449, 467

ARPANET 14

ネットワーク・ワーキング・グループ 13

ASN.1 546

Assigned Numbers RFC 731

ATM 45, 50, 52, 53, 56, 58

atm 49

ATM LAN エミュレーション 56

ATM 接続ホスト機能グループ (AHFG) 64

ATMARP 46

B

base-64 458

Berkeley Internet Name Domain (BIND) 317

Berkeley Software Distribution (BSD) 13

BGMP 264

BGP 189

経路リフレクター 205

集約 202

同期 200

同盟 203

プロトコルの説明 194

マルチプロトコル拡張 263

Bluetooth 28

BNF 409

BOOTP サーバー 125

BOOTP 転送 129, 546

BOOTP リレー・エージェント 546

Bootstrap Protocol (BOOTP) 125

ARP 126

BOOTPREQUEST メッセージ 129

BOOTREPLY メッセージ 128

DHCP インターオペラビリティ 141

forwarding 129

TFTP 126

転送 84

ネットワーク・コンピューター 944

メッセージ・フォーマット 127

Border Gateway Multicast Protocol 264

Border Gateway Protocol 189

Border Gateway Protocol (BGP)

バージョン 4 (BGP-4) 92

プロトコルの説明 194

C

CBC 689

CCITT 13

CDMF 376, 690

CERN 449

Channel Data Link Control (CDLC)

説明 921

Channel-to-Channel (CTC) 922

CICS 288

CICS ソケット・インターフェース 288

Cisco

Local Director 863
Classical IP over ATM 49, 61
CLNP 39, 41
Commercial Data Masking Facility (CDMF) 690
Commercial Internet Exchange (CIX) 17
Common information Model (CIM) 349
Common Link Access to Workstation (CLAW) 922
Communication Storage Manager (CSM) 923
CSNET
 メール・ゲートウェイ・サービス 407

D

DARPA 13
Data Encryption Standard (DES) 689, 789
DCA 14
DDN Network Information Center 291
DDNS 312
Defense Advanced Research Projects Agency (DARPA) 13
Defense Communication Agency (DCA) 14
DES 376, 689, 693, 702, 789
DHCPv6 621
Differentiated Services
 IPsec 849
Diffie-Hellman 693, 694, 702, 756, 757, 760, 762, 764, 766, 768
DiffServ
 DS 境界ノード 839
 DS ドメイン 835, 838, 842, 848
 DS バイト 835, 840, 842, 848, 850
 DS マーカー 840
 LDAP 848
 PHB 836
 QoS 835, 838, 842
 SLA 839, 842, 848
 TOS オクテット 835
 入り口ノード 839, 850
 拡張が容易なサービス選別 835
 境界ノード 844
 サービス・クラス 835
 サービス・レベル・アグリーメント (SLA) 835
 セキュリティ 850
 ソース・ドメイン 843
 ディレクトリー情報 848
 ディレクトリー・クライアント 848
 ディレクトリー・サーバー 848
 出口ノード 839
 動作集約 (BA) 分類機能 840
 トークン・パケット・フィルター 840

トラフィック計量 840
トラフィック条件付け機能 841, 843
トラフィック条件付け合意 (TCA) 842
トラフィック優先順位付けアルゴリズム 842
トラフィック・クラス・オクテット 835
トラフィック・プロファイル 840
内部コンポーネント 842
バイト、DS 842
パケット・シェーパー 840
パケット・ドロッパー 840
複数フィールド (MF) 分類機能 840
分類機能 840, 843
ホップごとの転送優先順位付け (PHB) 836
ポリシー情報 848
ポリシング 840
優先転送 (EF) PHB 838
予測可能なパフォーマンス 835

dig 312

Digital Signature Standard 698, 751

Directory Access Protocol (DAP) 331

Distance Vector Multicast Routing Protocol 248

DIXIE 332

DLSW 668

DNS

DNS セキュリティー 312

階層ネーム・スペース 293

クライアント認証 312

動的 DNS 312

トップレベル・ドメイン 293

ドメイン・ネーム解決 297

トランスポート 297

ネーム・サーバー 295

ネーム・リゾルバー 295

DOI 754, 755

Domain Internet Groper 312

Domain Name System (DNS) 25, 346

Berkeley Internet Name Domain (BIND) 317

DDNS 拡張 313

IPv6 拡張 617

IPv6 逆ルックアップ 618

IPv6 用リソース・レコード 618

print=q.in-addr.arpa ドメイン q.in-addr.arpa 295

TTL 917

アプリケーション 312

 dig 312

 host 312

 nslookup 312

一般ドメイン 294

- 階層ネーム・スペース 292
- 完全修飾ドメイン・ネーム 293
 - 例 310
- キャッシュ 300, 917
- 国別ドメイン 294
- 権限のゾーン 295
- 再帰的照会 299
- 信頼できない応答 299
- 信頼できる応答 299
- ゾーン 295
- ゾーン転送 300
- ゾーン・データ 310
- 存続時間 (TTL) 300, 302
- トランスポート 311
- ネーム・サーバー 297
 - キャッシングのみ 300
 - セカンダリー 300
 - 操作 300
 - プライマリー 300
- ネーム・リゾルバー 297
 - 操作 299
- ネットワーク・オブジェクト 301
- 反復照会 299
- 非再帰的照会 299
- 分散ネーム・スペース 295
- ポインター照会 295
- メッセージ 303
 - 説明 303, 308
- メッセージ圧縮 307
- メッセージ・フォーマット 315
- ラウンドロビン DNS 916
- リソース・レコード (RR)
 - 説明 301, 303
 - フォーマット 301, 321, 618
- ルート・ネーム・サーバー 296
- 例 308
- Domain of Interpretation (DOI) 754, 755
- DSA 700
- DSS 698, 700
- DUAL アルゴリズム 186
- DVMRP 248
- Dynamic Domain Name System (DDNS) 655
 - A レコード更新 327
 - Berkeley Internet Name Domain (BIND) 317
 - DDNS クライアント 318
 - DHCP オプション 81 328
 - DHCP サーバー 318
 - DNS 拡張 313
 - DNS 更新メッセージ 315
 - IBM DDNS UPDATE メッセージ・フォーマット 326
 - IBM Dynamic IP 322
 - IBM インプリメンテーション 324
 - KEY リソース・レコード 322
 - KEY リソース・レコード・フォーマット 322
 - MD5 322
 - PTR レコード更新 327
 - RSA 322
 - RSA 公開鍵 317
 - SIG リソース・レコード 322
 - TTL 323
 - 公開鍵 322
 - 混合環境 329
 - セキュリティ 320
 - デジタル署名 317
 - 動的事前セキュア・モード 320
 - 動的セキュア・モード 320
 - プロキシ A レコード更新 327
 - メカニズム 318
 - リソース・レコード・フォーマット 321
- Dynamic Host Configuration Protocol (DHCP) 125, 131, 328
 - BOOTP インターオペラビリティ 141
 - BOOTP 転送 129
 - BOUND 状態 137
 - DHCPACK メッセージ 134
 - DHCPDECLINE メッセージ 134
 - DHCPDISCOVER メッセージ 134
 - DHCPINFORM メッセージ 135
 - DHCPNACK メッセージ 134
 - DHCPRELEASE メッセージ 134
 - DHCPREQUEST メッセージ 134
 - DHCPv6
 - DHCP Advertise 623
 - DHCP Reconfigure 623
 - DHCP Release 623
 - DHCP Reply 623
 - DHCP Request 623
 - DHCP Solicit 622
 - IBM インプリメンテーション 324
 - INIT 状態 135
 - PTR レコード更新 327
 - REBINDING 状態 138
 - RENEWING 状態 137
 - アドレス割り振り 131
 - オプション 81 328
 - クライアント / サーバー対話 135, 137, 139
 - タイマー 137

ネットワーク・コンピューター 944
プロキシ A レコード更新 327
メッセージ・タイプ 134
メッセージ・フォーマット 132
リース時間 137

E

E メール 405
EBGP 192
EBONE 3
e-business 787
ECB 689
EGP 145, 189
EIGRP 151, 182
EJB 465
Enhanced Interior Gateway Routing Protocol 182
Enterprise Extender 667, 925
Enterprise JavaBean 465
Entity EJB 466
Extended TACACS 798
Extensible Markup Language (XML) 461
Exterior Gateway Protocol 145, 189

F

FDDI 32, 33
finger 574
Flags (フラグ) フィールド 729
For Your Information 文書 (FYI) 26
Fragment Offset (フラグメント・オフセット)
729

G

gated 208
GetBulkRequest 558
GIF 427
GUI 567

H

Hashed Message Authentication Code (HMAC)
699
Header checksum (ヘッダー・チェックサム)
729
High Performance Data Transfer (HPDT) 45, 923
High Performance Routing (HPR) 667
HMAC 699
host (プログラム) 312
HTML 450, 461, 462
HTTP
 cookie 451
 Secure Sockets Layer (SSL) 451
 Uniform Resource Identifier (URI) 453
 Uniform Resource Locator (URL) 453

Uniform Resource Name (URN) 453
Web ブラウザー 450
エンティティ 455
応答メッセージ 455
基本認証 458
キャッシング 453, 459
状況コードの定義 456
ダイジェスト認証 458
妥当性検査メカニズム 460
通信 450
トランザクション 450
認証 458
ネゴシエーション 459
プライバシーの露出 451
プロキシ 452, 711
プロトコル・パラメーター 453
メソッド 456
メッセージ・フォーマット 454
有効期限メカニズム 459
要求メッセージ 455

Hypertext Transfer Protocol (HTTP) 450, 868

I

IAB 23
IASG 調整機能グループ (ICFG) 64
IBGP 192
IBM Dynamic IP 322
IBM HTTP Server 449
IBM Web アプリケーション・サーバー 467
ICMP
 宛先到達不能 207
 時間超過 208
 始点抑制 208
 リダイレクト 207
ICMPv6 599
IDEA 690
IEEE 802.11 28
IEEE 802.x 標準 30
IESG 22
IGMP 241
 ローカル・グループ・データベース 243
IGP 145
IIOP 465
IMS ソケット・インターフェース 288
InATMARP 46
InformRequest 560
Interior Gateway Protocols 145
International Data Encryption Algorithm (IDEA)
690
Internet Architecture Board (IAB) 21, 23

- Internet Assigned Numbers Authority (IANA) 21, 26, 86, 237, 420, 428, 731
- Internet Control Message Protocol (ICMP) 105
 - Ping 116
 - 宛先到達不能 109
 - アドレス・マスク応答 116
 - アドレス・マスク要求 116
 - エコー 108
 - エコー応答 108
 - 時間超過 113
 - 情報要求 115
 - 情報応答 115
 - ソース消失 110
 - タイム・スタンプ応答 114
 - パス MTU ディスカバリー 105
 - パラメーター問題 114
 - リダイレクト 111
 - ルーター公示 111
 - ルーター送信請求 111
 - ルーター・ディスカバリー 105
 - ルーター・ディスカバリー・プロトコル 112
- Internet Control Message Protocol (ICMPv6) 621
 - DHCP サーバー 613
 - MTU 607
 - 近隣キャッシュ 605
 - 近隣公示メッセージ 604
 - 近隣送信請求メッセージ 602, 613
 - 近隣ディスカバリー 601
 - 近隣未到達検出 (NUD) 612
 - 暫定アドレス 613
 - 自動構成 613
 - ステートフル自動構成 613, 621, 625
 - ステートレス自動構成 613, 621
 - マルチキャスト・グループ 614
 - マルチキャスト・リスナー終了メッセージ 617
 - マルチキャスト・リスナー・ディスカバリー (MLD) 614
 - マルチキャスト・リスナー・レポート 616
 - メッセージ・フォーマット 600
 - リダイレクト・メッセージ 609
 - ルーター公示メッセージ 605, 613
 - ルーター送信請求メッセージ 608, 613
- Internet Engineering Steering Group (IESG) 22
- Internet Engineering Task Force (IETF) 21, 23, 788
- Internet Group Management Protocol 241
- Internet Group Management Protocol (IGMP) 118, 241, 599
- マルチキャスト・リスナー・ディスカバリー (MLD) 614
- Internet Inter-ORB Protocol 465
- Internet Message Access Protocol バージョン 4 (IMAP4) 439
- Internet Network Information Center (InterNIC) 68
- Internet Society (ISOC) 21
- Internet2
 - 参加者 19
- Internet2 の使命 19
- IP 49
 - 次世代 (IPng) 635
 - プロトコル・スタック 125
- IP Assist 923
- IP Security Architecture (IPSec)
 - AH と ESP の組み合わせ 743
 - Diffie-Hellman アルゴリズム 694
 - Diffie-Hellman 鍵交換 695
 - DiffServ 849
 - Digital Signature Algorithm 700
 - Hashed Message Authentication Code (HMAC) 699
 - HMAC 699
 - IPSec モジュール 727
 - RSA アルゴリズム 693
 - SA バンドル 727, 743
 - Security Association Database (SAD) 727
 - Security Parameter Index (SPI) 726
 - Security Policy Database (SPD) 727
 - 暗号の概念 686
 - 概念 726
 - カプセル化 728
 - 組み合わせられたトンネル 746
 - 公開鍵 694
 - 公開指数 694
 - 処理の順序 744
 - 整合性検査 849
 - セキュリティ・アソシエーション (SA) 726
 - トランスポートの隣接化 743, 746
 - トンネルリング 728
 - トンネルリングのネスト 743
 - トンネルリングの反復 743
 - トンネル・モード 850
 - 秘密鍵 694
 - 秘密指数 694
 - 変形 725
 - モジュラス 693, 695
- IP Version 6 (IPv6) 599, 601, 613
- 6Bone 635

DNC 拡張 617
 IPv4 からの移行 624
 IPv4 互換可能アドレス 594
 IPv4 ネットワーク経由のトンネリング 627
 IPv4 マップ・アドレス 594
 IPv6 アドレス 592
 アドレス・スペース 592
 オプション 584
 拡張ヘッダー 585
 ESP 592
 redid=ipv6ext. タイプ - 長さ - 値 (TLV) オプション形式 585
 宛先オプション 592
 タイプ - 長さ - 値 (TLV) オプション形式 587
 認証 592
 フラグメント 591
 ホップ単位 587
 ルーティング 589
 逆 DNS ルックアップ 618
 形式設定接頭部 593
 構成 トンネリング 631
 サイト・ローカル・ユニキャスト・アドレス 594
 自動トンネリング 627
 送信請求ノード・マルチキャスト・アドレス 598
 データグラム 581
 ドメイン・ネーム・サービス 617
 トンネリング 587, 626
 自動 627
 構成 631
 二重スタック 625
 任意キャスト・アドレス 598
 ノード 581
 パケット 581
 ヘッダー・フォーマット 581
 パケット・サイズ 585
 フラグメント・ヘッダー 591
 フロー 599
 フロー・ラベル 599
 ヘッダーの変換 633
 ヘッダー・チェックサム 584
 ホスト 581
 ホップ単位ヘッダー 587
 モバイル・サポート 624
 リンク・ローカル・ユニキャスト・アドレス 594
 ルーター 581
 IP アドレス 5
 枯渴 85, 579
 ネットワーク番号 5
 ホスト番号 5
 IP アドレスのこ渴 579
 IP ゲートウェイ 11
 IP スタック 125
 IP 接頭部 90
 IP データグラム 92
 TOS オクテット 835
 概要 7
 IP データグラム・ヘッダー 93
 IP バージョン 6 (IPv6)
 VPN 788
 トラフィック・クラス・オクテット 835
 IP ルーター 5
 IP6.INT ドメイン 618
 IPA 923
 IP-FDDI 32
 IPv6 788
 IPv6 用 RIPng 162
 IPX 728
 ISAKMP/Oakley
 Certificate Payload (証明書ペイロード) 760
 Diffie-Hellman 752, 756, 757, 760, 762, 764, 768
 Diffie-Hellman アルゴリズム 694
 Digital Signature Algorithm 700, 758
 Digital Signature Standard 751
 DOI 754, 755
 Domain of Interpretation (DOI) 754, 755, 764
 Encryption Bit (暗号化ビット) 759
 Encryption Flag (暗号化フラグ) 763
 Flags (フラグ) フィールド 759
 Hash Payload (ハッシュ・ペイロード) 763, 765, 767
 ID 756
 ID ペイロード 758
 Identity Payload (ID ペイロード) 759, 760
 Identity Protect 交換 752
 Initiator Cookie (起動側 Cookie) 753, 755, 757
 Integrity Check Value (ICV) 767
 ISAKMP header (ISAKMP ヘッダー) 753, 755, 761, 763, 767
 ISAKMP ヘッダー 759
 Key Exchange (鍵交換) フィールド 756, 762
 Key Exchange (鍵交換) ペイロード 765, 766
 KEY_OAKLEY 754, 755
 LDAP 758

Message ID (メッセージ ID) 753, 755, 765, 767
Message ID (メッセージ ID) フィールド 761
Message-ID (メッセージ ID) 763
Nonce (臨時) フィールド 756
Nonce (臨時) ペイロード 766
Oakley クイック・モード 761
Oakley メイン・モード 752, 761
Perfect Forward Secrecy (PFS) 751, 762
PFS 751, 762
Proposal Payload (提案ペイロード) 754, 755, 764
PROTO_ISAKMP 754, 755
Responder Cookie (応答側 Cookie) 753, 755, 757
RSA アルゴリズム 693
RSA 公開鍵暗号化 751
RSA 公開鍵認証 756
Security (セキュリティ) ペイロード 765
Security Association (セキュリティ・アソシエーション) フィールド 754, 755
Security Association (セキュリティ・アソシエーション) ペイロード 764, 767
SKEYID 752, 757, 760, 766
SKEYID_a 758
SKEYID_d 758
SKEYID_e 758
SPI 759, 766, 768, 769
Transform Payload (変形ペイロード) 754, 755, 764
宛先ポート 753
アプリケーション・レイヤー・セキュリティ 761
暗号化 761
暗号化アルゴリズム 754
暗号鍵 751, 756, 758, 762
永続 ID 752, 769
改訂された RSA 公開鍵認証 756
鍵交換属性 762
鍵マテリアル 750, 757, 758, 766, 768
疑似乱数機能 754, 763
公開鍵 751, 752
公開値 756, 757, 760, 762, 764, 766, 768
サービス妨害 750
事前共有鍵 751
指数 756
証明書 752, 758, 761, 769
証明書ペイロード 758
署名ペイロード 758

セキュリティ・アソシエーション 751, 752, 755, 760, 761
セキュリティ保護一式 751
中継ユーザー 751
デジタル署名 759, 760
認証 693, 761
認証鍵 758
認証局 758
認証方式 754
認証メカニズム 751, 761
認証要求メッセージ 758
ハッシュ機能 761
秘密値 756, 757
フェーズ 1 751
フェーズ 2 751
複合値 757
プロキシ・ネゴシエーション・プログラム 762
プロトコル・コード・ポイント 766
保護 DNS サーバー 758
保護一式 753, 768
保護ローカル・キャッシュ 758
マスター鍵 752
マスター秘密 751
メッセージ 1 753, 762, 767
メッセージ 2 754, 765, 768
メッセージ 3 756, 767
メッセージ 4 757
メッセージ 5 758
メッセージ 6 760
リモート・アクセス 768
リモート・ホスト 752, 768
臨時 756, 757, 762, 765, 766, 768
ISDN
B チャンネル 37
Basic Rate Interface (BRI) 37
D チャンネル 37
NRZ エンコード 37
PPP カプセル化 37
Primary Rate Interface (PRI) 37
最大伝送単位 (MTU) 38
ISO 13
ISO 3166 294
print=q.arpa ドメイン q.arpa 295
ISO 8859 422
ISP 749
ITU-T 13
IV 689

J

Java

- Java Server Pages (JSP) 463
- JavaScript 462
- アプレット 462
- サーブレット 463
- 署名付きアプレット 462
- ネットワーク・コンピューター 463, 943

- Java Server Pages (JSP) 463

- JavaBean 464

- JavaScript 462

- JPEG 427

K

- KAS 791

- KDBM 795

- Kerberos Authentication Server (KAS) 791

- Kerberos Database Manager (KDBM) 795

- Kerberos Key Distribution Server (KKDS) 795

- Kerberos システム 345, 369, 403

- 許可モデル 795

- 前提事項 789

- 想定する目的

- 会計 789

- 許可 789

- 認証 789

- データベース管理 795

- 認証プロセス 790, 795

- 命名 789

- インスタンス名 790

- プリンシパル名 790

- レルム名 790

- KKDS 795

L

- LAN Channel Station (LCS) プロトコル 922

- LAN エミュレーション (LANE) 61

- LAN エミュレーション・サーバー 58

- LAN セグメント 125

- LAN 置換 49

- LANE

- 層 62

- Layer 2 Forwarding (L2F) 799

- Layer 2 Tunneling Protocol (L2TP) 799

- LAC 800

- LNS 800

- NAS 800

- アクセス・コンセントレーター 800

- セキュリティ機能 803

- セッション 800

- トンネル 801

- ネットワーク・アクセス・サーバー 800

- ネットワーク・サーバー 800

- LDAP 758

- Lightweight Directory Access Protocol (LDAP)

- DiffServ 848

- Directory Access Protocol (DAP) 331

- Kerberos 345

- LDAP クライアント 333

- LDAP サーバー 333, 340, 346

- Name Service Interface (NSI) 348

- Simple Authentication and Security Layer (SASL) 344

- X.500 331, 347

- X/Open Directory Service (XDS) 348

- アクセス権 335

- 暗号化 335, 343

- 機能モデル 341

- 許可 343

- ゲートウェイ・プロセス 333

- 検索 341

- 検索フィルター 335, 342

- 構文 337

- 識別名 (DN) 335, 338

- 情報モデル 336

- スキーマ 338

- スキーマ検査 338

- セキュリティ 343

- セキュリティ・メソッド 335

- 相対識別名 (RDN) 335, 338

- 通信プロトコル 333

- ディレクトリー 334

- ディレクトリー項目 335

- ディレクトリー情報ツリー (DIT) 335, 339

- 認証 343

- ネーミング・モデル 338

- バインディング 335

- LIS 47

- LocalDirector 863, 878

- Logical IP Subnetwork (LIS) 52

- Lotus Domino 468

- Lotus Notes 702

- LSAP 30

M

- MA 552

- MAC 696, 699

- Management Information Base

- IBM に特定の部分 551

- MIB-II グループ定義 548

オブジェクト定義の例 547
概要 548
Management Information Base (MIB) 547
MD5 314, 322, 666, 698, 699
Metropolitan Fiber Systems (MFS) 18
Metropolitan Area Ethernet (MAE) 18
MIB 547
Microsoft Internet Explorer 448
MILNET 14
Mosaic 447
MOSPF 253
MOTIF 567
MOUNT コマンド 396
MPC+ 923
MPEG 427
MPOA 61, 65
機能グループ層 62
クライアント 61
サーバー 61
操作 65
利点 61
論理コンポーネント 61
MPTN 671
mrouted 248
MSDP 261
Multicast Backbone (MBONE) 265
Multicast OSPF 253
Multicast Source Discovery Protocol 261
MultiNode Load Balancing (MNLB) 863, 886
および Sysplex Distributor 893
MultiNode Load Balancing (MNLB) Forwarding Agent 888
MultiNode Load Balancing (MNLB) Service Manager 887
MultiNode Load Balancing (MNLB) Workload Agent 888
Multi-Path Channel+ (MPC+) 45
Multi-Purpose Internet Mail Extensions (MIME)
MIME の新規フィールド値の登録 420
SMTP との関係 406
エンコードされた語 436
エンコード方式 429
概要 417
内容タイプ 422
ヘッダー・フィールド 421
メールボックスのアドレッシング 415
文字セット 436
ISO-8859 422
US-ASCII 422

N

NAS 798
National Science Foundation (NSF) 15
National Science Foundation Network (NFSNET) 15
National Security Agency (NSA) 698
NCF
Lotus Domino 468
WebSphere Application Server 467
WebSphere Performance Pack 467
NCSA httpd 449
NE 552
NetBIOS 728
エンド・ノード 677
サービス 676
スコープ 677
データグラム・ディストリビューター (NBDD) 678
データ・リンク・スイッチ
スイッチ対スイッチ・プロトコル 668
ネーム・サーバー (NBNS) 678
Netscape Navigator 448
NETSTAT 575
Network Access Points (NAPs) 18
Network Control Program (NCP) 13
Network Dispatcher 863
Advisor 869, 874
Dispatcher 864, 866, 872
Executor 866, 868
Interactive Session Support 864
Manager 868
PING 三角測量 873
Workload Manager (WLM) 876
重み 868, 873
監視 865
規則 871
クラスター 866
高可用性 869
スティッキー・オプション 871
測定基準 868, 873
待機マシン 870
パフォーマンス 868
負荷平準化 864
負荷モニター・サービス 864
平衡化 871
リモート・サーバー 872
Network File System (NFS)
MOUNT コマンド 396
オプション 396

- NFS URL 402
- NFS プロトコル 398
- 概念 394
- キャッシュ・ファイル・システム 400
- ファイル・システム 399
- マウント・プロトコル 395
- リモート・プロシージャー・コール 394
- ロック・マネージャー・プロトコル 399
- Network Information System (NIS) 329
- Network News Transfer Protocol 574
- NewsReader/2 574
- Next Generation Internet (NGI) イニシアチブ 18
- NIS 329
- NIST 698
- NLRI 191
- NMA 552
- NMS 552
- NNTP 574
- Not-So-Stubby エリア 180
- NSA 698
- NSFNET 3
- nslookup 312
- NVT 352
- O**
- Oakley クイック・モード 761
- Oakley メイン・モード 761
- OMPROUTE 924
- Open Look 567
- Open Shortest Path First 165, 924
- Open System Interconnection (OSI) 5
- OS/2 Warp Server 312
- OS/390 312
 - Workload Manager (WLM) 876
 - シスブレックス 856
- OSA-Express 923
- OSF 567
- OSI 546
- OSI プロトコル・スタック 331
- OSPF 237, 924
- OSPF エリア 166
- OSPF 仮想リンク 177
- over-the-air プロトコル (OTA) 511
- P**
- Perfect Forward Secrecy (PFS) 751, 762
- PFS 751, 762
- PGP 690
- PIM 256, 258
- PIM 希薄モード 258
- PIM 高密度モード 256
- PIM-DM 256
- PIM-SM 258
- Ping 116
- Point-to-Point Protocol (PPP) 798
 - IP Control Protocol (IPCP) 36
 - IPCP 36
 - L2TP トンネル 801
 - LCP 35
 - Link Control Protocol (LCP) 35
 - NCP 35
 - Network Control Protocol (NCP) 35
 - Synchronous Digital Hierarchy (SDH) 44
 - Synchronous Optical Network (SONET) 44
 - Synchronous Payload Envelope (SPE) 44
 - Van Jacobson Header Compression 37
 - 認証 36
- Point-to-Point Tunneling Protocol (PPTP) 799
- poison reverse 249
- Post Office Protocol 437
- PostScript 405, 419, 428
- Pretty Good Privacy (PGP) 690
- Project Athena 567
- Protocol Independent Multicast 256
- protocol virtual LAN (PVLAN) 60
- ProxyArec 328
 - ProxyArec 328
- Proxy-ARP 81
 - 概念 122
- PVC 47
- Q**
- QoS
 - ベストエフォート 811
- Queued Direct I/O 923
- Queued Direct I/O (QDIO) 923
- R**
- RADIUS 798
- RC2 704
- RC4 704
- Real-Time Transport Protocol 469
- Remote Authentication Dial In User Service 798
- Remote Exec Daemon (REXECD) 363
 - REXEC コマンド 363
 - REXECD 363
 - 動作原理 405
- Remote Shell Protocol (RSH) 363
- Request for Comments (RFC)
 - Internet Standards Track 22

RFC 0768 212, 214
RFC 0791 67
RFC 0792 105, 106
RFC 0793 215, 229
RFC 0821 405, 408, 416, 417, 418
RFC 0822 405, 409, 417, 418, 425
RFC 0826 46, 50, 66, 118, 119
RFC 0854 362, 393
RFC 0855 362
RFC 0877 38
RFC 0894 31, 66
RFC 0903 124
RFC 0904 189
RFC 0906 126, 129
RFC 0919 67
RFC 0922 67, 84
RFC 0925 122
RFC 0934 419
RFC 0948 31, 32
RFC 0950 67, 105, 106
RFC 0951 23, 125, 128
RFC 0959 383
RFC 0974 405, 415, 417
RFC 0977 574
RFC 1001 676
RFC 1002 676
RFC 1010 31
RFC 1013 574
RFC 1027 122
RFC 1028 546
RFC 1032 313
RFC 1033 313
RFC 1034 25, 291, 302, 313
RFC 1035 25, 291, 302, 313
RFC 1042 32, 53, 66
RFC 1049 405, 417, 419
RFC 1052 545, 565
RFC 1055 34, 66
RFC 1057 281
RFC 1058 151, 157, 158, 159, 924
RFC 1075 248
RFC 1085 565
RFC 1101 313
RFC 1112 117
RFC 1122 56
RFC 1123 392, 406, 468
RFC 1138 468
RFC 1144 34, 37, 66
RFC 1148 468

RFC 1149 23
RFC 1155 546, 565
RFC 1156 546, 548
RFC 1157 546, 552, 553, 554, 565
RFC 1166 68, 69
RFC 1179 567
RFC 1183 313
RFC 1188 66
RFC 1191 105, 585, 635
RFC 1202 332
RFC 1206 26
RFC 1213 546, 548, 549, 565
RFC 1215 565
RFC 1228 565
RFC 1239 565
RFC 1249 332
RFC 1256 105, 111
RFC 1288 574
RFC 1293 46
RFC 1325 26
RFC 1327 468
RFC 1349 67, 95, 850
RFC 1350 391, 392, 393
RFC 1351 565
RFC 1352 565
RFC 1356 38, 66
RFC 1390 32
RFC 1408 356
RFC 1437 23
RFC 1441 556, 565
RFC 1466 88
RFC 1480 294, 313
RFC 1483 53, 66
RFC 1487 332, 349
RFC 1492 798, 809
RFC 1494 419
RFC 1496 419
RFC 1510 788, 809
RFC 1518 89, 90
RFC 1518 - 1520 89
RFC 1519 89
RFC 1520 89, 92
RFC 1542 23, 125
RFC 1577 46, 61
RFC 1579 713, 809
RFC 1583 924
RFC 1584 253
RFC 1591 313
RFC 1592 284, 565

RFC 1594	26	RFC 1970	601
RFC 1618	37, 66	RFC 1995	313, 314
RFC 1619	44, 66	RFC 1996	313, 314
RFC 1631	810	RFC 2002	655, 660, 662, 663, 666
RFC 1633	815, 850	RFC 2003	729, 810
RFC 1652	406, 417	RFC 2011	549, 566
RFC 1661	35, 66	RFC 2012	549, 566
RFC 1662	35, 66	RFC 2013	549, 566
RFC 1700	36, 39, 47, 66, 237	RFC 2026	22
RFC 1706	302, 313	RFC 2045	406, 418, 437
RFC 1723	159, 924	RFC 2046	406, 418, 437
RFC 1733	439	RFC 2047	406, 418, 435, 437
RFC 1734	438	RFC 2048	406, 419, 420, 428, 437
RFC 1748	565	RFC 2049	406, 419, 437
RFC 1752	580, 635	RFC 2050	88
RFC 1755	56, 66	RFC 2054	402
RFC 1771	189, 194	RFC 2055	402
RFC 1777	332, 349	RFC 2058	798
RFC 1778	332, 350	RFC 2060	439
RFC 1779	332, 350	RFC 2065	325
RFC 1795	66, 668	RFC 2080	162
RFC 1809	599	RFC 2090	394
RFC 1812	26, 165	RFC 2104	810
RFC 1813	394, 402	RFC 2109	468
RFC 1823	333, 350	RFC 2131	131
RFC 1825	735	RFC 2132	125, 129, 131, 134, 137
RFC 1826	732	RFC 2136	312, 313, 314, 324, 325
RFC 1827	736, 738, 740	RFC 2137	325
RFC 1869	406, 411	RFC 2138	798, 810
RFC 1870	407	RFC 2156	419, 437, 468
RFC 1883	580, 635	RFC 2159	437
RFC 1886	617, 618, 635	RFC 2183	437
RFC 1889	469	RFC 2185	624, 635
RFC 1890	469	RFC 2203	401, 402
RFC 1901	556, 565	RFC 2205	820, 828, 833, 850
RFC 1904	565	RFC 2206	850
RFC 1905	556, 558, 566	RFC 2207	850
RFC 1906	556, 566	RFC 2208	851
RFC 1907	556, 566	RFC 2209	851
RFC 1909	556, 566	RFC 2210	851
RFC 1910	556, 566	RFC 2211	816, 851
RFC 1918	88	RFC 2212	816, 820, 851
RFC 1928	770, 771, 809	RFC 2222	344, 350
RFC 1929	770, 809	RFC 2223	23
RFC 1939	437	RFC 2224	402
RFC 1945	450, 468	RFC 2225	46, 49, 53, 66
RFC 1959	332, 350	RFC 2228	383
RFC 1960	332, 350	RFC 2231	437
RFC 1961	770, 809	RFC 2236	117, 240

RFC 2246 786
RFC 2251 332, 334, 350
RFC 2252 332, 350
RFC 2253 332, 350
RFC 2254 332, 350
RFC 2255 333, 346, 350
RFC 2256 333, 350
RFC 2283 264
RFC 2313 810
RFC 2314 810
RFC 2315 810
RFC 2328 165, 171
RFC 2338 909, 914
RFC 2341 799
RFC 2347 391, 392
RFC 2355 360, 362
RFC 2362 256
RFC 2373 592, 598, 635
RFC 2374 595, 635
RFC 2375 598, 635
RFC 2390 46, 66
RFC 2396 453, 468
RFC 240 810
RFC 2400 26, 49, 66
RFC 2401 810
RFC 2402 730, 735
RFC 2403 810
RFC 2404 810
RFC 2405 810
RFC 2406 736, 742, 810
RFC 2407 810
RFC 2408 810
RFC 2409 810
RFC 2410 810
RFC 2411 810
RFC 2412 810
RFC 2427 40, 66
RFC 2460 580, 599, 635
RFC 2461 600, 601, 612, 635
RFC 2462 613, 614, 635
RFC 2463 600, 601, 635
RFC 2474 851
RFC 2535 313, 314
RFC 2570 563, 566
RFC 2571 563, 566
RFC 2572 563, 566
RFC 2573 563, 566
RFC 2574 563, 566
RFC 2575 563, 566

RFC 2578 556, 566
RFC 2579 556, 566
RFC 2580 556, 566
RFC 2616 450, 451, 453, 454, 455, 460, 468
RFC 2617 458, 468
RFC 2661 799, 810
RFC 2664 26
RFC 2742 566
RFC 2800 26
RFC 2822 453, 468
RFC 2845 325
RFC 2888 803, 810
RFC 2893 624, 635
RFC 2909 264
RFC 2988 851
RFC 3007 312, 313, 314
RFC 3010 394, 402
RFC 3022 720, 810
RFC 922 84
状態 23, 24
記録 24
限定使用 24
実験 24
情報 24
推奨 24
提案標準 24
ドラフト標準 23
任意選択 24
非推奨 24
必須 24
標準 23
目的 23
Request for Comments(RFC)
RFC 2012 549
Resource Reservation Protocol (RSVP) 599, 815, 820
Filterspec 831
filterspec 822, 826
Flowspec 830
flowspec 822
path メッセージ 822
PathTear メッセージ 826
QoS 820, 822, 827
QoS 予約 823, 825
ResvTear メッセージ 826
Tspec 831
共通ヘッダー 828
共用明示スタイル 827
共用予約 826

- 権限 823
- 固定フィルター・スタイル 827
- 受信側 821
- 送信側 821
- ソフト状態 825
- 特殊な予約 826
- 取り除き要求 826
- パス 821
- パス定義 821
- フロー 820, 821
- フロー記述子 822
- ポリシー制御 823
- マルチキャスト・アドレス 820
- メッセージ・オブジェクト 828
- 予約 820
- 予約スタイル 826
- 予約メッセージ 822
- 予約要求 825
- ルーター 821
- ルーティング・テーブル 821
- ワイルドカード・フィルター・スタイル 827
- Reverse Address Resolution Protocol (RARP)
 - 応答 124
 - パケット・フォーマット 124
 - 命令コード・フィールド 124
 - 要求 124
- REXX ソケット 289
- RIP 151, 924
 - パケット・フォーマット 152
- RIP バージョン 1 159
- RIP バージョン 2 159
- routed 208
- Routing Information Protocol 151
- RPC
 - 外部データ表現 (XDR) 278
- RPCGEN 282
- RSA 314, 322, 692, 693
- RSA アルゴリズム 693
- RSH 363
- RTCP 469
- RTE 163
- RTP 469
 - Control Protocol 469
 - ヘッダー形式 471
- R T e a u x I P E u r o p F n s (R I P E) 86
- S
- SA バンドル 727, 746
- Secure Electronic Transactions (SET) 697
 - カード所有者 804
- 鍵管理 808
- 決済機関 804
- 支払ゲートウェイ 804
- 取引 805
- 認証局 805
- 発行者 804
- マーチャント 804
- Secure Hash Algorithm 1 (SHA-1) 698
- Secure Sockets Layer (SSL) 777
 - Change Cipher Spec プロトコル 782
 - Handshake Protocol 778, 782
 - HTTP 451
 - Record Protocol 778, 785
 - TCP ポート 443 779
 - 暗号鍵の生成 784
 - 公開鍵 785
 - 互換性 780
 - 状態 781
 - 証明書 784
 - セキュリティ問題 779
 - セッション 781
 - セッション状態 781
 - 接続状態 782
 - 対称鍵 785
 - ハンドシェイク・フェーズ 779
 - マスター鍵 785
 - メッセージ確認コード (MAC) 785
 - レコード・レイヤー 782
- Security Association Database (SAD) 727
- Security Parameter Index (SPI) 726, 731, 737
- Security Policy Database (SPD) 727
- Serial Line IP (SLIP) 798
 - Van Jacobson Header Compression 34
 - アドレッシング 34
 - インプリメンテーション 35
 - 概要 34
- Session EJB 467
- SET 702
- SGMP 546
- SHA-1 698
- Shortest-Path First アルゴリズム 149
- Simple Gateway Monitoring Protocol (SGMP) 546
- Simple Internet Transition (SIT) 624
- Simple Mail Transfer Protocol (SMTP)
 - Domain Name System 405, 414
 - MX リソース・レコード 415
- 宛先アドレス 412
- 概要 405
- 各国語サポート 405

- シナリオ 413
- 非テキスト・データ 405
- メール交換プロトコル 410
- メールボックス 409
- メールボックス・アドレス 412
- メール・ヘッダーのフォーマット 408
- メール・ヘッダー・フォーマット
 - MIME 拡張機能 421
 - 非 ASCII 文字の使用 435
- Simple Network Management Protocol (SNMP)
 - MA 552
 - NE 552
 - NMA 552
 - NMS 552
 - 管理エージェント (MA) 552
 - コミュニティ名 554
 - コンポーネント 553
 - ネットワーク管理アプリケーション (NMA) 552
 - ネットワーク管理ステーション (NMS) 552
 - ネットワーク・エレメント (NE) 552
 - プロトコル・データ単位 554
 - メッセージ・フォーマット 556
- SKEYID 757, 766
- SKEYID_a 758
- SKEYID_d 758
- SKEYID_e 758
- SMI 546
- SMTP サービス拡張機能 406, 411
- SNA 668
 - LCP 35
 - Link Control Protocol (LCP) 35
 - NCP 35
 - Network Control Protocol (NCP) 35
- SNMP 556
- SNMPv2 556
- SNMPv2 エンティティ 557
- SNMPv2 パーティ 557
- SOCKS
 - SOCKS 化された TCP/IP スタック 770
 - SOCKS 化されたクライアント 770
 - SOCKS サーバー 769
 - SOCKSv4 770
 - SOCKSv5 770
 - TCP 接続 772
 - UDP サポート 771
 - UDP 接続 776
 - UDP 中継サーバー 776
 - UDP ポート 776
- 暗号化規格 771
- 回線レベル・ゲートウェイ 769
- 鍵管理システム 771
- カプセル化方式 773
- トンネリング・プロトコル 771
- 認証方式 771
- バージョン ID 772
- ファイアウォール 714, 769
- 方式オプション 772
- 要求詳細メッセージ 774
- source-routing bridges 546
- SPF アルゴリズム 166
- Split horizon 157
- split horizon with poison reverse 158
- Standard Protocol Numbers (STD) 25
 - STD 01 26, 35
 - STD 02 96
 - STD 05 67, 105
 - STD 06 212
 - STD 07 215
 - STD 51 35
- Sub-Network Access Protocol (SNAP) 30
- Subnetwork Access Protocol (SNAP) 39, 41
- Sun HotJava ブラウザー 448
- SVC 47
- Synchronous Digital Hierarchy (SDH) 44
- Synchronous Optical Network (SONET) 44
- Synchronous Payload Envelope (SPE) 44
- Sysplex Distributor 863, 880, 926
 - QoS 情報の使用 927
 - および Cisco MNLB 893
 - 負荷平準化の規則 882
 - 分散元スタック 926
 - 利点 927
- Systems Network Architecture (SNA) 5
- T
- TACACS 798
- TACACS+ 798
- TCP
 - SOCKS 化されたスタック 770
- TCP/IP
 - 層 5
- TCP/IP プロトコル・スイート 3
- TELNET 351
 - NVT 352
 - NVT プリンター 353
 - オプションのネゴシエーション 358
 - 基本コマンド 359
 - コマンド構造 356

- 操作の原則 351
- ネットワーク仮想端末装置 352
- ネットワーク仮想端末装置 (NVT) 352
- ネットワーク・コンピューター 943
- フルスクリーン機能 356
- プロキシ・サーバー 711
- Telnet 6
- Terminal Access Controller Access Control System 798
- TGS 792
- Time To Live (TTL) (存続時間)
 - IPSec 認証ヘッダー (AH) 729
- TN3270 359
- Transmission Control Protocol (TCP) 868
 - 3 ウェイ・ハンドシェーク 229
 - アプリケーション・プログラミング・インターフェース 229
 - ウィンドウ 220
 - 確認通知 226
 - 可変タイムアウト間隔 228
 - 高速再送 234
 - 再送 226
 - 信頼性 217
 - ストリーム・データ転送 216
 - スロー・スタート 231
 - セグメント 216, 221
 - IP データグラムでマップされた 229
 - オプション 224
 - フォーマット 222
 - 接続の確立 228
 - 全二重 217
 - ソケット 212
 - 多重化 217
 - チェックサム 223
 - 定義 215
 - 特性 216
 - ハーフ・アソシエーション 212
 - 輻輳制御 232
 - 輻輳制御アルゴリズム 230
 - プッシュ 216
 - フロー制御 217
 - 論理接続 217
- Transmission Control Protocol/Internet Protocol (TCP/IP) 3
- Triple-DES 689, 703
- Trivial File Transfer Protocol (TFTP)
 - BOOTP 126
 - NetASCII 393
 - Sorcerer's Apprentice Syndrome 392
 - TFTP RFC 392
- 概要 391
- データ・モード 393
- ネットワーク・コンピューター 944
- パケット・タイプ 392
- プロトコル 392
- trust チェーン 702
- Type of Service (TOS) (サービス・タイプ) 729
- U
- UMOUNT コマンド 398
- Uniform Resource Identifier (URI) 453
- Uniform Resource Locator (URL) 453
- Uniform Resource Name (URN) 453
- University Corporation for Advanced Internet Development (UCAID) 19
- Usenet News 574
- User Datagram Protocol (UDP) 212, 868
 - アプリケーション・プログラミング・インターフェース 214
 - チェックサム 214
 - データグラム・フォーマット 213
 - ポート 213
- V
- VIPA
 - 概要 925
 - コールド・スタンバイ 926
 - 動的 VIPA 926
 - 引き継ぎ 926
 - ホット・スタンバイ 926
- VIPA の自動引き継ぎ 926
- Virtual Router Redundancy Protocol (VRRP)
 - 1 次 IP アドレス 910
 - IP アドレス所有者 910
 - VRRP ルーター 910
 - 仮想ルーター 910, 911
 - 仮想ルーター ID 910
 - 仮想ルーター・バックアップ 911
 - 仮想ルーター・マスター 910
 - 選択プロトコル 911
 - デフォルト・ルーター 909
 - 認証 915
 - マルチキャスト・アドレス 913
 - 優先順位 914
- VRML 450
- W
- Web サーバー 449
- Web ブラウザー 447
- WebSphere Application Server 467

WebSphere Edge Server 467
 Windows Internet Name Service 679
 WINS 679
 Winsock インターフェース 273
 Wireless Application Protocol (WAP) 28, 499
 Wireless Session プロトコル (WSP) 518
 Wireless Telephony Application (WTA) 516, 518
 Wireless Transaction プロトコル (WTP) 531
 WLM 927
 World Wide Web
 Mosaic 447
 WWW 447
 Network News Transfer プロトコル 574
 NNTP 574
X
 X Window 567
 Mosaic 447
 X Toolkit 570
 X Window Manager 570
 X クライアント 569
 X サーバー 569
 X プロトコル 570
 Xlib 570
 ウィジェット 571
 概念 568
 使用される TCP ポート 573
 ネットワーク・コンピューター 944
 プロトコル 573
 X.25
 Subnetwork Access Protocol (SNAP) 39
 後続プロトコル ID (SPI) 38
 組織固有 ID (OUI) 40
 ネットワーク層プロトコル ID (NLPID) 38
 バーチャル・サーキット 38
 プロトコル ID (PID) 40
 プロトコル・データ単位 (PDU) 38
 呼び出し要求パケット 38
 呼び出し利用者データ (CUD) 38
 X.500 331, 347, 368
 X.509 702
 X.509 証明書 786
 XCF 926
 XML 461
 XNS 151
 XTACACS 798
あ
 アーキテクチャー 3
 アーキテクチャー・モデル 6, 8
 アプリケーション 6
 インターネットワーク 7
 層 5
 トランスポート 6
 ネットワーク・インターフェース 7
 アカウント作業 797
 宛先オプション拡張ヘッダー 735, 742
 アドレス解決 (ATMARP および InATMARP) 46
 アドレス割り振り 131
 アノニマス FTP 390
 アプリケーション層 6
 アプリケーション・プログラミング・インター
 フェース (API)
 TCP 229
 UDP 214
 ソケット 211
 アプリケーション・プロトコル 8
 アプリケーション・レベル・ゲートウェイ 706
 アルゴリズム
 Diffie-Hellman 694
 Digital Signature 700, 758
 RSA 693
 TCP 輻輳制御 230
 鍵交換 693
 公開鍵 691, 692, 693
 ストリーム 688
 対称 688, 690
 ブロック 688
 暗号 687
 制限された 687
 暗号アルゴリズム 687
 暗号化 335, 343, 682, 687, 695, 697, 771
 強力 687
 暗号化アルゴリズム 703, 754
 暗号解読 682, 687
 暗号解読者 703
 暗号鍵 687
 暗号カップル 688
 暗号化ブロック・チェーン (CBC) 689
 暗号化、強力な 687
 暗号テキスト 692
 暗号ペイロード (ESP)
 AH との組み合わせ 743
 Authentication Data (認証データ) 740
 CDMF 739
 DES-CBC 変形 739
 ESP トレーラー 736
 ESP 認証データ 736
 ESP ヘッダー 736

- HMAC-MD5 740
- HMAC-SHA-1 740
- Integrity Check Value (ICV) 740
- IP フラグメント 736
- IPv6 環境 742
- Next Header (次のヘッダー) フィールド 740
- Pad Length (埋め込み長) 739
- Padding (埋め込み) 739
- Payload Data (ペイロード・データ) 739
- Security Parameter Index (SPI) 737
- Sequence Number (シーケンス番号) 738
- 暗号化 736, 739
- 再生保護 739
- 整合性検査 736
- トランスポート・モード 740, 744
- トンネル・モード 740, 744
- 認証 736, 744
- 変形 725
- 暗号方式 693, 702

い

- イーサネット
 - 802.2 論理リンク制御 (LLC) 30
 - ARP 31, 119
 - DIX 29, 119
 - DSAP 30
 - IEEE 802.3 標準 29, 31
 - IEEE 802.4 標準 31
 - IEEE 802.5 標準 31
 - IEEE 802.x 標準 128
 - LSAP 30
 - SNAP 30
 - SSAP 30
 - Sub-Network Access Protocol (SNAP) 30
 - フレーム・フォーマット 29
 - プロトコル・タイプ番号 29
 - ヘッダー・フィールド 29
- 一時ポート 10, 210
- 一方向機能 695
- 一回限りパスワード 682
- イメージ 450
- 因数分解 692
- インターネット 3, 4, 13, 14
 - Acceptable Use Policy (AUP) 17
 - Advance Network and Services (ANS) 17
 - ANS CO+RE 17
 - Commercial Internet Exchange (CIX) 17
 - ソース・ルーティング 546
 - の営利使用 17
 - ブリッジされるセグメント 546

- インターネット層 7
- インターネットワーキング 3, 13
- インターネットワーキング層プロトコル 61
- インターネットワーク 3
- インターネットワーク・アドレス・サブグループ (IASG) 63
- インターネット・ゲートウェイ 11
- インターネット・サービス・プロバイダー (ISP) 18
- インターネット・プロトコル (IP) 7, 131
 - High Performance Routing (HPR) 667
 - IP 接頭部 90
 - MTU 99
 - TTL 96
 - 気付アドレス 655
 - 経路記録オプション 103
 - 厳密ソース・ルーティング・オプション 102
 - 専用 IP アドレス 721
 - タイム・スタンプ 104
 - データグラム 92
 - データグラム・ヘッダー 93
 - トンネリング 664
 - 非厳密ソース・ルーティング・オプション 101
 - フラグメント化 99
 - モバイル IP 655
 - モバイル IP 登録 660
 - モバイル IP の操作 657
 - ルーティング・オプション 101
- インターネット・プロトコル・スイート 3

う

- ウィルス 681
- ウィンドウ
 - TCP に適用されている 220
 - 原則 217, 220
- ウェルノウン・サービス 281, 415
- ウェルノウン・ポート 10, 210

え

- エージェント誘導メッセージ 656
- エニーキャスト 85
- エンティティ 557

お

- オープン
 - 能動 / 受動 228, 230
- オープン・ソフトウェア・ファウンデーション (OSF) 567
- オブジェクト・マネージメント・グループ (OMG) 465

オブジェクト・リクエスト・ブローカー (ORB) 465
オフライン・モデル 440
音声 450
オンライン・モデル 440
か
回線の盗聴 681
回線レベル・ゲートウェイ 706
外部エージェント 656
鍵管理 690, 771, 808
鍵交換 690
鍵交換アルゴリズム 693
鍵付きアルゴリズム 687
鍵の長さ 692
鍵のリフレッシュ 682, 762
鍵マテリアル 750, 757, 766, 768
仮想現実 450
仮想接続 VC 50
仮想ルーター 909, 911
カプセル化 728
可変フィールド 729
可用性 874, 917
管理 556
管理エージェント (MA) 552
管理情報の構造と識別 (SMI) 546
き
キー・スペース 687
ギガビット・イーサネット 923
疑似ヘッダー
 IPv6 584
疑似乱数機能 754
疑似乱数生成プログラム 703
気付アドレス 655
偽名の使用 681
逆パス転送アルゴリズム 246
キャッシュ 378, 403, 917
 ARP キャッシュ 118
 HTTP のキャッシング 453, 459
 ICMPv6 近隣キャッシュ 605
共通オブジェクト・リクエスト・ブローカー・
 アーキテクチャー (CORBA) 465
共有鍵 692
共有秘密 693, 695, 698, 699
強力な暗号化 743
許可 343, 368, 682, 797
 Simple Authentication and Security Layer
 (SASL) 344
距離ベクトル・プロトコル 151

距離ベクトル・ルーティング 148
近隣ディスカバリー 601
く
組み合わされたトンネル 746
クライアント / サーバー・モデル 9, 271
クラスレス・ドメイン間ルーティング (CIDR)
 IP 接頭部 90
 伝送ルーティング・ドメイン (TRD) 90
グラフィカル・ユーザー・インターフェース (GUI) 567
クリア・テキスト 687
け
計算法、モジュラー 693
継承適任者 183
経路記録 103
経路サーバー機能グループ (RSFG) 64
経路再配布 178
経路要約 181
ゲートウェイ 11, 125
ゲートウェイ・アドレス 125
権限 823
厳密ソース・ルーティング 102
こ
公開鍵 322, 690, 700, 701, 751, 785
公開鍵アルゴリズム 691, 692, 693
攻撃 682
コード・ブック 689
顧客情報管理システム (CICS) 288
コモン・ゲートウェイ・インターフェース (CGI) 462
さ
サーバー、LDAP 340
サーバー・サイド・インクルード 463
サービス品質 (QoS) 811
サービス妨害攻撃 681, 682, 730, 750, 803
サービス・レベル・アグリーメント (SLA) 835
サブレット 463
再構成、DHCP 623
再生攻撃 681
再生保護 732
最大伝送単位 (MTU) 33, 99
 ICMPv6 607
 ISDN 38
 パス MTU の判別 585
サブエージェント 284
サブネット 71
サブネット化
 可変長 73

静的 73
サブネット番号 72
サブネット・マスク 72
判別 76

し
システム間カップリング・ファシリティー (XCF) 924
シスプレックス 856
ジッター 819
自動構成 613
指紋 695
集結点 (RP) 258
収束 154
終端装置機能グループ (EDFG) 63
静的ルーティング 146, 147
衝突回避 695
証明書 682, 752, 784
省略語 955
初期設定ベクトル 702
初期設定ベクトル (IV) 689
自立走行式システム 144
信頼性のあるトランスポート・プロトコル 185

す
スーパーネット化 90
スタブ・エリア 180
ストリーム・アルゴリズム 688
スプーフィング攻撃 803

せ
制限された暗号 687
整合性検査 682, 688, 696
セキュリティ ID カード 710
セキュリティ・ポリシー 685
セッション別鍵 700

接続
ATM 45
DLSW 668
FDDI 32
ISDN 37
MPC+ 45
MPOA 60
PPP 35
SLIP 34
SONET 44
X.25 38
フレーム・リレー 40
切断使用モデル 440
セル 50
専用 IP アドレス 729

そ
素因数 700
層 5
ソケット
CICS ソケット・インターフェース 288
IMS ソケット・インターフェース 288
REXX ソケット 289
アソシエーション 211
アドレス 211
インターフェース 211
会話 211
システム呼び出し 275
ストリーム・タイプ 273, 277
定義 211
データグラム・タイプ 273, 277
トランスポート・アドレス 212
ハーフ・アソシエーション 211
ロー・タイプ 273, 277

素数 692
存在位置 799
存続時間
DDNS 323
IP 96
IPv6 ホップ限度 583
NetBIOS 名 679
RSVP 829

た
帯域幅予約システム (BRS) 668
対称アルゴリズム 688, 690
対称鍵 785
タイプ - 長さ - 値 (TLV) 587
タイム・スタンプ 692
大量の暗号化 693

ち
チェックサム
IPv6 584
TCP データグラム 223
UDP データグラム 214
力づく攻撃 681, 689
チケット許可サーバー (TGS) 792
チャネル・プロトコル 921
中継ユーザー 751
中継ユーザー攻撃 701, 803
中心基点ツリー・アルゴリズム 247
長期鍵 700
重複フラグメント攻撃 730
直接メモリー・アクセス (DMA) 923
値、ハッシュ 698

て

- ディクショナリー攻撃 681
- デジタル証明書 701
- デジタル署名 317, 691, 697, 759, 760
- デジタル封筒 697
- ディスクレス・ホスト 126
- データ・リンク接続 ID (DLCI) 40
- データ・リンク層 7
- データ・リンク・スイッチ
 - スイッチ対スイッチ・プロトコル 668
 - IEEE 802.2 LLC Type 2 669
 - NetBIOS 668
 - SDLC 669
- デフォルト転送機能機能グループ (DFFG) 64
- デフォルト・ルーター 64, 909
- 電子コード・ブック・モード (ECB) 689
- 電子メール 405
- 電子メール・モデル 440
- 転送能力 61

と

- 透過的なサブネット化 122
- 統合サービス
 - filterspec 816
 - flowspec 816
 - QoS 815
 - Resource Reservation Protocol (RSVP) 815
 - Rspec 816, 818, 819, 820
 - Tspec 817, 818, 820
 - アドミッション制御 814
 - 液体モデル 819
 - サービス率 820
 - サービス・クラス 816
 - 最小ポーリング単位 817
 - 最大パケット・サイズ 817
 - トークン率 817
 - トークン・パケット・フィルター 817, 820
 - トラフィック制御 813
 - パケット能力 817
 - パケット分類機能 813, 815
 - パケット・スケジューラー 813, 815
 - ビデオ会議 812
 - 負荷コントロール・サービス 816, 818
 - フロー 812, 814, 815, 819, 820
 - フロー記述子 816
 - 保証されたサービス 816, 819
 - 保証された帯域幅 812
 - 予約 820
 - リアルタイム伝送 812

- リアルタイム・サービス 814
- リソース予約 814
- 統合サービス (IS) 812
- 頭字語 955
- 動的 DNS 312
- 動的 VIPA 926
- 動的ホスト構成プロトコル (DHCP) 655
 - ICMPv6 613
- 動的ルーティング 146
- トークンリング LAN 32
- トポロジー・データベース 170
- ドメイン・ネーム・システム (DNS)
 - IPv6 626
- トランスポート層 6
- トランスポート層プロトコル境界 673
- トランスポートの隣接化 743, 746
- トリガー更新 159
- トンネリング 627, 728, 771
- トンネルリングのネスト 743
- トンネルリングの反復 743
- トンネル 801
 - トンネル・モード 850

に

- 偽ヘッダー
 - TCP チェックサム 223
 - UDP チェックサム 214
- ニュースグループ 574
- ニュース・エージェント 574
- ニワトリが先か卵が先かの問題 126
- 認証 36, 343, 368, 403, 438, 666, 682, 687, 688, 691, 696, 697, 710, 771, 777, 797, 805, 915
 - IP Control Protocol (IPCP) 36
 - IPCP 36
- 認証局 758, 805
- 認証局 (CA) 701
- 認証ヘッダー (AH)
 - Authentication Data (認証データ) 732
 - Authentication Data フィールド 698
 - ESP との組み合わせ 743
 - Flags (フラグ) フィールド 729
 - Fragment Offset (フラグメント・オフセット) 729
 - Header checksum (ヘッダー・チェックサム) 729
 - HMAC-MD5-96 732
 - HMAC-SHA-1-96 732
 - Integrity Check Value (ICV) 732
 - IP フラグメント 730
 - IPv6 環境 735

Next Header (次のヘッダー) フィールド 731
Payload Length (ペイロードの長さ) 731
Reserved (予約) フィールド 731
Security Parameter Index (SPI) 731
Sequence Number (シーケンス番号) 732
Time To Live (TTL) (存続時間) 729
Type of Service (TOS) (サービス・タイプ) 729
鍵付き MD5 732
可変フィールド 729, 734
再生保護 732
トランスポート・モード 733, 744
トンネル・モード 733, 744
ヘッダー形式 730
変形 725
認証方式 754
ね
ネーム・サーバー
権限の委任 296
ゾーン・データの例 310
ネットワーク仮想端末装置 (NVT) 352
ネットワーク管理
SGMP 546
オブジェクト ID 551
管理情報の構造と識別 546
ネットワーク管理アプリケーション (NMA) 552
ネットワーク管理ステーション (NMS) 552
ネットワーク層 7
ネットワーク層到達可能性情報 191
ネットワーク・アクセス・サーバー 798
ネットワーク・アドレス変換 (NAT)
FTP パケット 723
IP ネットワーク設計 723
IPSec 725
TCP パケット 723
アドレス・プール 722
セキュア・ネットワーク 722
専用 IP アドレス 721
タイムアウト値 724
変換メカニズム 722
無保護ネットワーク 722
ロード・バランサー 917
ネットワーク・インターフェース層 7
ネットワーク・エレメント (NE) 552
ネットワーク・オブジェクト 301
ネットワーク・コンピューター 463, 943
ネットワーク・ファイル・システム (NFS)
ネットワーク・コンピューター 943

は

バージョン 2、Simple Network Management Protocol (SNMPv2) 556
パーティー 557
ハイパーテキスト文書 450
ハイパーテキスト・マークアップ言語 (HTML) 447, 450, 461
ハイブリッド・ルーティング 150
パケット・フィルタ操作 706
パケット・フィルタ操作ルーター 706
パス MTU ディスカバリー 105
パス属性 197
ハッカー 682, 687
バックス正規形式 (BNF) 409
ハッシュ機能 695, 696, 699, 761
ハッシュ値 695, 698, 699

ひ

非厳密ソース・ルーティング 101
ビデオ 450
非同期伝送モード 45
否認防止 682, 688, 692
非武装地帯 (DMZ) 719
秘密鍵 690, 700
秘密、共有 693
標準プロトコル番号 (STD)
STD 02 358, 420
STD 03 311
STD 08 351
STD 09 383
STD 10 405, 417
STD 11 405
STD 13 291, 437
STD 14 405
STD 33 391

ふ

ファイアウォール 12, 685, 727, 749
DMZ 719
FTP 受動モード 712
FTP 通常モード 712
FTP プロキシ 710, 712
HTTP プロキシ 711
IBM Firewall 710
SOCKS 714
TELNET プロキシ 711
アウトバウンド接続 714
アプリケーション・レベル・ゲートウェイ 718, 771
インバウンド接続 714

- 回線レベル・ゲートウェイ 714, 769
- 概要 12
- 拡張フィルター操作 708
- サービス・レベル・フィルター操作 708
- セキュリティ ID カード 710
- 選別サブネット・ファイアウォール 719
- 選別ホスト・ファイアウォール 718
- ソース / 宛先レベル・フィルター操作 708
- デュアル・ホーム・ゲートウェイ 716
- 認証 710
- ネットワーク・アドレス変換 (NAT) 720
- パケット・フィルター操作 707
- パケット・フィルター操作規則 707
- パケット・フィルター操作ファイアウォール 716
- パケット・フィルター操作ルーター 719
- 非武装地帯 719
- フィルター規則 707
- プロキシ 709, 714, 716
- 保護ネットワーク 705
- ポリシー 685, 707
- 無保護ネットワーク 705
- 要塞ホスト 718, 719
- ロギング 711
- ファイル転送プロトコル (FTP) 6, 383, 868
 - アノニマス FTP 390
 - 応答コード 387
 - サブコマンド
 - cd 385
 - close 387
 - dir 385
 - get 386
 - lcd 385
 - ls 385
 - mode 386
 - open 385
 - pass 385
 - put 386
 - quit 387
 - site 385
 - type 386
 - user 385
 - シナリオ 388
 - 受動モード 386, 712
 - 操作 384
 - 通常モード 712
 - 転送セッションの終了 387
 - ネットワーク・コンピューター 943
 - プロキシ 387
 - プロキシ・サーバー 710, 712
 - ログイン 385
 - 復号 687
 - ブッシュ・テクノロジー 509
 - 物理層 62
 - フラグメント化 99
 - フラグメント化拡張ヘッダー 735, 742
 - フラット・ネーム・スペース 291
 - 組織ドメイン 294
 - 地理的ドメイン 294
 - ブリッジ 11
 - プリンシパル ID 789
 - プル・テクノロジー 509
 - フレーム・リレー
 - ARP 43
 - Subnetwork Access Protocol (SNAP) 41
 - データ・リンク接続 ID (DLCI) 40
 - ネットワーク・レベル・プロトコル (NLPID) 41
 - バーチャル・サーキット 40
 - プロトコル・データ単位 (PDU) 41
 - ブロードキャスト
 - 逆パス転送アルゴリズム 84
 - 説明 83
 - プロキシ 387, 706
 - プロキシ ARP 666
 - プロキシ・サーバー 769
 - ブロック・アルゴリズム 688
 - プロトコル番号
 - IPv6 ヘッダー内の 583
 - プロトコル・スタック 5
 - 分散 DVIPA 926
 - 分散コンピューティング環境 (DCE) 341
 - DCE アーキテクチャー 330
 - DFS 管理可能ドメイン 378
 - Distributed Time Service 365
 - Interface Definition Language (IDL) 375
 - Kerberos 369
 - NFS インターオペラビリティ 378
 - X.500 347, 368
 - X/Open Directory Service (XDS) 348
 - アーキテクチャー上のコンポーネント 365
 - アクセス制御リスト・ファシリティ 370
 - アプリケーション・プログラム・インターフェース 365
 - インターネット・ドメイン・ネーム・サービス (DNS) 368
 - エピソード・ファイル・システム 379
 - 拡張特権属性証明書 (EPAC) 372

キャッシュ・マネージャー 378
許可 368
クーリエ・タイム・サーバー 376
グローバル・タイム・サーバー 376
グローバル・ディレクトリー・エージェント
(GDA) 346, 366
グローバル・ディレクトリー・サービス (GDS)
366
公開鍵のサポート 374
スレッド 365
セキュリティ・サービス 365
セル 346, 366, 378
セル・ディレクトリー・サービス (CDS) 346,
366
チケット許可チケット (TGT) 371
ディレクトリー・サービス 330, 365
特権サービス 369
特権チケット許可チケット (PTGT) 372
トランスポート・サービス 365
認証 368
認証サービス 369, 370
バックアップ・クーリエ・タイム・サーバー
376
ファイルセット・ロケーション・データベース
(FLDB) 367
ファイルのネーミング 379
分散ファイル・サービス 365
マスター・レプリカ 366
読み取り専用レプリカ 366
リモート・プロシージャ・コール 365
レジストリー 369
レジストリー・サービス 370
ローカル・タイム・サーバー 376
ローカル・ファイル・システム (LFS) 379
ログイン・ファシリティ 370

へ

平衡化 864, 916
米国防総省 (DoD) 14
米国防情報・技術局 (NIST) 698
変形 725

ほ

ポインター照会 295
ポート 209
一時 210
ウェルノウン 210
ポート・マップ 398
ホーム・エージェント 655
保護 DNS サーバー 758

保護ローカル・キャッシュ 758
ホップ限度 583
ホップ・バイ・ホップ拡張ヘッダー 735, 742
ポリシー・エージェント 927, 928

ま

マスター鍵 785
マスター秘密 751
待ち時間 61
マルチキャスト 85
DVMRP 248
RSVP 820
TTL 値 240
VRRP 913
アドレス 237
サーバー (MCS) 64
全ホストのグループ 243
ホスト・グループ 85, 237
マルチキャスト・グループ 614
マルチキャスト・リスナー終了メッセージ
617
マルチキャスト・リスナー・ディスカバリー
(MLD) 614
マルチキャスト・リスナー・レポート 616
Multicast Backbone (MBONE) 265
マルチキャスト・ドメインの相互接続 261
マルチキャスト送達ツリー 244
マルチキャスト・アドレス解決サーバー (MARS)
64
マルチプロトコルのカプセル化 53
マルチプロトコル・トランスポート・ネットワー
ク体系 (MPTN)
MPTN ゲートウェイ 674
アドレス・マッピング・サーバー・ノード
674
機能の補償 672
上位機能 672
トランスポート機能 672
トランスポート層プロトコル境界 (TLPB) 673
ネーミングおよびアドレッシング機能 672
マルチキャスト・サーバー・ノード 674
マルチホーミング 68
マルチホーム 76, 91, 92, 243
マルチメディア 469
む
無限カウント 154
無償の ARP 666
め
メールボックス 409

メッセージ確認コード 682
メッセージ確認コード (MAC) 696, 785
メッセージ・シーケンス番号 (IMAP4) 442
も
モジュラー計算法 693
モバイル IP 655, 729
 エージェント公示 658
 登録 660
 トンネル 664
 ブロードキャスト・データグラム 664
モバイル IP の操作 657
ゆ
ユニキャスト
 アドレス 83
ら
乱数機能 703
乱数生成プログラム 702
り
離散対数 693, 695
リモート印刷 (LPR/LPD) 567
リモート転送機能機能グループ (RFFG) 64
リモートプロシージャ呼び出し (RPC)
 応答メッセージ 280
リモート・アクセス 768
リモート・アクセス・サーバー (RAS) 685
リモート・ダイヤルイン 796
リモート・プロシージャ・コール (RPC) 278
 概念 278
 トランスポート 279
 ポート・マップ 281
 呼び出しメッセージ 279
リモート・ホスト 749, 752
リモート・メソッド呼び出し (RMI) 466
両方向乱数ハンドシェイク 682
リンク状態 149
リンク状態データベース 149
リンク状態ルーティング 149
リンク層 7
臨時 756, 757, 762, 765, 766, 768
る
ルーター 5, 10, 11, 546
 RSVP 821, 825
 仮想ルーター 909, 911
 ソフト状態 825
 デフォルト・ルーター 909
 ルーター・ディスカバリー・プロトコル 112
ルーター・ディスカバリー 105

ルーター・ディスカバリー・プロトコル 112
ルーティング 77
 間接 77
 直接 77
 部分的なルーティング情報 77
ルーティング拡張ヘッダー 735, 742
ルーティング・テーブル 78
 RSVP 821
ルーティング・テーブル爆発 89
ルーティング・プロトコル 143
ろ
論理接続 212
わ
ワールド・ワイド・ウェブ (WWW) 3, 447
 Apache 449
 CERN 449
 FTP クライアント 447
 HTML インタープリター 447
 HTTP クライアント 447
 IBM Web アプリケーション・サーバー 467
 Java Server Pages (JSP) 463
 Lotus Domino 468
 Microsoft Internet Information Server API (ISAPI) 463
 NCSA httpd 449
 Netscape Navigator 448
 Netscape Server API (NSAPI) 463
 NNTP クライアント 447
 POP クライアント 447
 SMTP クライアント 447
 Sun HotJava ブラウザー 448
 Uniform Resource Identifier (URI) 453
 Uniform Resource Locator (URL) 453
 Uniform Resource Name (URN) 453
 Web サーバー 449
 Web ブラウザー 447, 450
 WebSphere Application Server 467
 WebSphere Performance Pack 467
 クライアント / サーバー処理モデル 448
 コモン・ゲートウェイ・インターフェース (CGI) 462
 サーバー・サイド・インクルード 463
 サプレット 463
ワイヤレス・アプリケーション環境 (WAE) 516

IBM



Redbooks

TCP/IP チュートリアル および技術解説書



TCP/IP チュートリアル および技術解説書

TCP/IP プロトコ ル・スイートの ネットワーキング の基本について

QoS やセキュリ ティーなどの拡張 概念

最新の TCP/IP プ ロトコル

TCP/IP プロトコル・スイートは、今日のネットワークで結ばれた世界におけるコンピューター通信のデファクト・スタンダードになっています。特定のネットワーク標準のユビキタス・インプリメンテーションにより、それによって使用可能になっているアプリケーションに非常に依存した状態が生じています。今日では、TCP/IP プロトコルとインターネットは、娯楽や情報を得るためだけでなく、取引を実行し、商品を売買し、顧客にサービスを提供することによってビジネスを運営するためにも使用されています。TCP/IP を使用する一連のアプリケーションは拡張を続けており、さらなるインフラストラクチャー・サポートのニーズが拡大しています。

この TCP/IP チュートリアルおよび技術解説書では、TCP/IP プロトコル・スイートを詳細に考察します。第 1 部では、TCP/IP について紹介し、このプロトコルに不可欠な基礎概念についての基本的な理解を得られるようにします。続いて第 2 部では、新たに開発されたワイヤレスおよびマルチメディア・アプリケーションを含む、今日の最もよく知られた TCP/IP アプリケーション・プロトコルの概要について解説します。

最後に第 3 部では、IPv6、セキュリティ、Quality of Service、IP モビリティ、および MPLS を含む、ネットワーキングにおける拡張された概念と最近のインフラストラクチャーの傾向について説明します。TCP/IP が現在直面している課題とそれを克服するために開発されているテクノロジーについても説明します。

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

実際の経験に 基づいた テクニカル情報の構築

IBM レッドブックは、IBM International Technical Support Organization によって作成されています。IBM のエキスパート、世界各地のカスタマーおよびパートナーが、実際のシナリオに基づいたタイムリーなテクニカル情報を作成しています。それぞれの環境で IT ソリューションをより効果的にインプリメントするための、具体的な推奨事項を提示しています。

詳細については、以下を参照してください。
ibm.com/redbooks

GG88-4005-00

