Performance and Fault Management

Chris Elliott

Escalation Engineer

chelliot@cisco.com

Cisco Systems, Inc.

NETWORLD + INTEROP

CISCO SYSTEMS

EMPOWERING THE INTERNET GENERATION℠

InteropNet

iLabs

Introduction

- Who am I?
  - 1975—Started working on the ARPAnet
  - 1977—Member of the DARPA TCP/IP committee
  - 1993—Started participation with InteropNet
  - 1996—Cisco Certified Internetwork Expert (CCIE)
  - 1999—Member of the InteropNet Advisory Board
  - 2000—Author of “*Performance and Fault Management*”, Cisco Press, June 2000
  - Active member of the IETF, concentrating on Network Management issues
  - iLabs team member for 2 years, concentrating on IP Telephony

Introduction

- Networld+Interop InteropNet Labs
  - Interoperability testing, demo, and education at N+I shows in Las Vegas and Atlanta
- Four areas of specialization:
  - IP Telephony
  - MPLS
  - SANs
  - Internet 2
- Booth 1549

Agenda

- What Is Performance and Fault Management?
- How SNMP Object Types Work
- Data Collection Best Practices
- How to Select the Best Objects for Your Network
- Best Objects for Different Devices and Technologies
- Polling vs. Notifications
- How to “Roll Your Own”
- Q&A

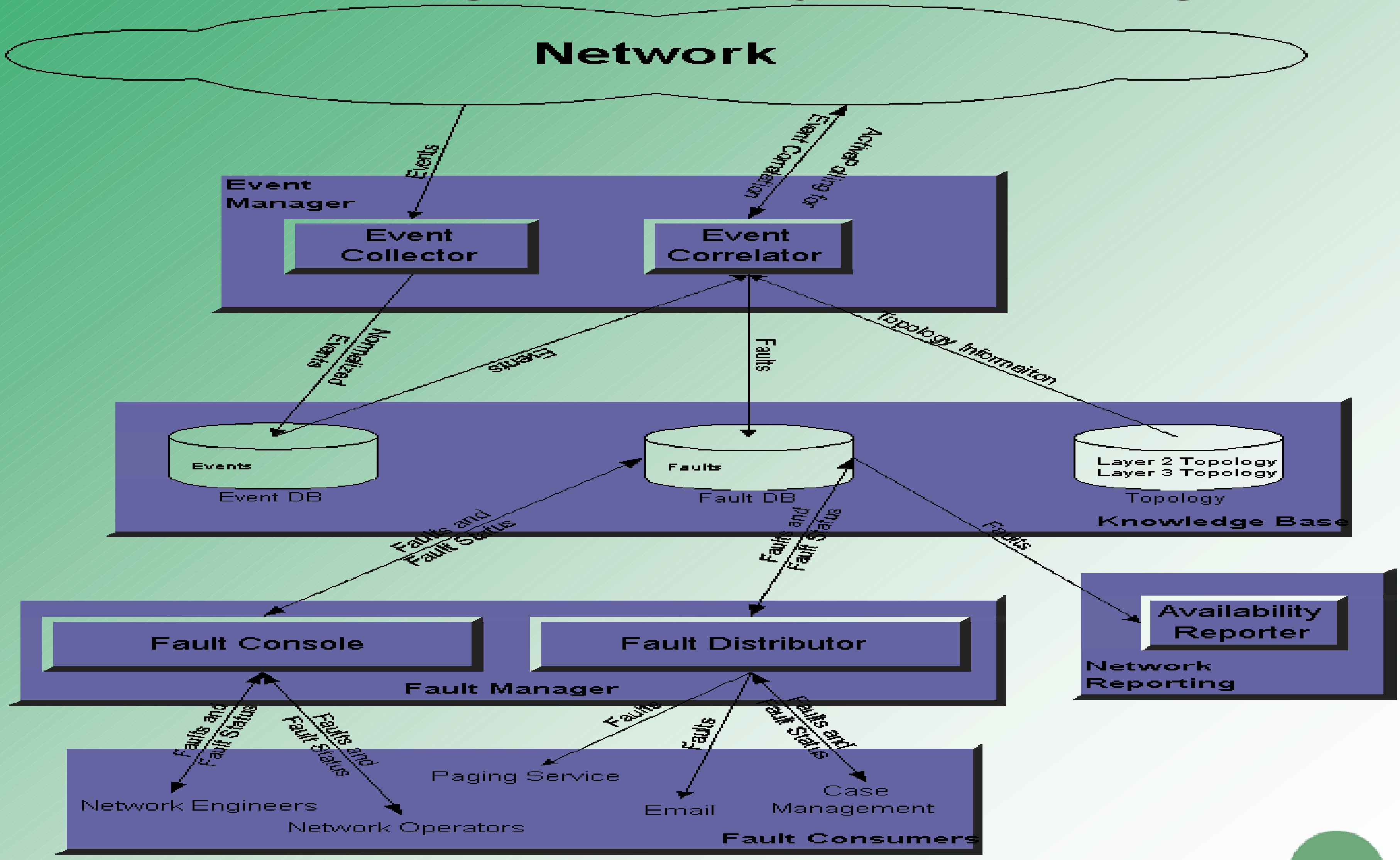
Performance Management

- Measuring and reporting on network performance so that performance can be maintained at an acceptable level
- Not real-time
- Example measurement points: network throughput, user response times, and line utilization
- Steps to Performance Management
  - –Data collection
  - –Baselining the network to establish typical performance
  - –Determining thresholds of acceptable performance

Fault Management

- Detecting, reporting, and (to the extent possible) automatically fixing network problems as they occur
- Real-time
- Example events input: syslog messages, SNMP notifications (traps or informs), events from NMS
- Steps to Fault Management
  - –Setting thresholds that trigger events
  - –Event collection
  - –Event correlation to determine Faults
  - –Fault reporting and tracking

Fault Management System Diagram



InteropNet
iLabs

How Performance and Fault Management Intersect

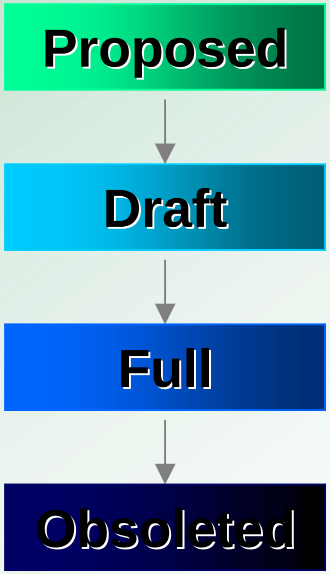
- Processing performance data may uncover network faults
  - –This may lead you to add event thresholds to more quickly report these issues
- Excessive or repeated faults may lead you to change what is being monitored for performance
  - –Monitor additional objects
  - –Modify the thresholds of acceptable performance

Agenda

- What Is Performance and Fault Management?
- How SNMP Object Types Work
- Data Collection Best Practices
- How to Select the Best Objects for Your Network
- Best Objects for Different Devices and Technologies
- Polling vs. Notifications
- How to “Roll Your Own”
- Q&A

Standard MIBs

- Beware, an RFC is not necessarily a standard.
  - Internet Drafts (I-D) (118 MIB modules in 101 drafts currently)
  - Standards Track Process defined in RFC 2026
    - Proposed (111 MIB modules in 105 RFCs)
    - Draft   (25 MIB modules in 21 RFCs)
    - Full (11 MIB modules in 9 RFCs)
    - Obsoleted (83 MIB modules in 72 RFCs)
  - Non-standards-track MIB modules
    - Experimental (9 MIB modules in 9 RFCs)
    - Informational (9 MIB modules in 8 RFCs)
    - Historic (6 MIB modules in 5 RFCs)
  - IANA maintained documents: IANA-IF-TYPES



How to Stay Informed About MIBs

- IETF Operations and Management Area
  - –http://www.ietf.org http://www.rfc-editor.org
  - –Specific web site for O&M
  - –http://www.ops.ietf.org
- A mailing list:
  - –mibs@ietf.org
- Bill Fenner’s site:
  - –http://www.aciri.org/fenner/mibs/
- Cisco’s MIB site:
  - –http://www.cisco.com/public/sw-center/netmgmt/cmtk/
  - –ftp://ftp.cisco.com/pub/mibs

SNMP Object Identification

- Need a scheme that allows two vendors or products within a vendor to compare like items
  - –Object Identifiers (OID) were chosen as the identification scheme
  - –An OID is an ordered sequence of non-negative integers written left to right, containing at least two elements (0.0)
  - –Bound to simple names in MIB Modules:
    - •"ifInOctets" is 1.3.6.1.2.1.2.2.1.10

SNMP Object Identification

- OIDs are not limited to SNMP protocol
  - Are useful, globally unique values that can be used for identifying anything
- Once a MIB module is published, OIDs are bound for all time to the objects defined
  - Objects can not be deleted! See RFC 2665
    - Can only be made obsolete
    - Even minor changes to an object are discouraged

SNMP Object Identification

- Most common prefixes are:
  - 1.3.6.1.2.1—contains MIB-II/std. objects
  - 1.3.6.1.3—experimental MIB modules
  - 1.3.6.1.4.1—contains vendor’s objects
- IEEE is now defining MIBs under their space:
  - 1.2.840.10006.300.43—802.3ad Link Aggregation
  - 1.2.840.10036—802.11 Wireless
- Enterprise OIDs are delegated by IANA

Tools for Managing OIDs

- Useful tools for managing OID/names
  - –libsmi (open source)
    - http://www.ibr.cs.tu-bs.de/projects/libsmi/
      - –smidump -f identifiers
  - –SMICng (commercial)
    - http://www.snmpinfo.com
      - –smicng –L
  - –Cisco OID files
    - ftp://ftp.cisco.com/pub/mibs/oid
  - –Cisco SNMP Translate tool
    - http://jaguar.ir.miami.edu/~marcus/snmptrans.html

Example OID Report

```
-- List format from SMICng version 2.2.0.7
1.3.6.1.2.1.31.1.1 TOT: ifXTable[IF-MIB]
1.3.6.1.2.1.31.1.1.1 ROT: ifXEntry[IF-MIB] aug: ifEntry[IF-MIB]
1.3.6.1.2.1.31.1.1.1.1 COT: ifName[IF-MIB] syn: DisplayString[SNMPv2-TC]acc:ro
1.3.6.1.2.1.31.1.1.1.2 COT: ifInMulticastPkts[IF-MIB] syn: Counter32 acc: ro
1.3.6.1.2.1.31.1.1.1.3 COT: ifInBroadcastPkts[IF-MIB] syn: Counter32 acc: ro
1.3.6.1.2.1.31.1.1.1.6 COT: ifHCInOctets[IF-MIB] syn: Counter64 acc: ro
1.3.6.1.2.1.31.1.1.1.7 COT: ifHCInUcastPkts[IF-MIB] syn: Counter64 acc: ro
1.3.6.1.2.1.31.1.1.1.8 COT: ifHCInMulticastPkts[IF-MIB] syn: Counter64 acc: ro
1.3.6.1.2.1.31.1.1.1.9 COT: ifHCInBroadcastPkts[IF-MIB] syn: Counter64 acc: ro
1.3.6.1.2.1.31.1.1.1.14 COT: ifLinkUpDownTrapEnable[IF-MIB]
         syn: ENUM{ enabled(1) disabled(2) } acc: rw
1.3.6.1.2.1.31.1.1.1.15 COT: ifHighSpeed[IF-MIB] syn: Gauge32 acc: ro
1.3.6.1.2.1.31.1.1.1.17 COT: ifConnectorPresent[IF-MIB] syn: TruthValue  1.3.6.1.2.1.31.1.1.1.18
    COT: ifAlias[IF-MIB] syn: DisplayString  1.3.6.1.2.1.31.1.1.1.19 COT:
    ifCounterDiscontinuityTime[IF-MIB] syn: TimeStamp
```

TOT—table object type
ROT—row object type
COT—columnar object type
SOT—scalar object type

Some SNMP Object Types

- Integer32
  - –Length is 32 bits, can be negative
- UInteger32
  - –Still 32-bit, but non-signed
- Gauge32
  - –An integer reflecting a current value
- Counter32 and 64
  - –Only counters come in 64-bit size
  - –Counts something until it reaches it’s maximum value, then wraps

Some SNMP Object Types-continued

- TimeTicks
  - Length is also 32-bit
  - A measurement of time in hundredths of a second
- OctetString
  - 0 or more bytes
  - A string of 8-bit characters
  - DisplayString is an OctetString of printable characters

How Gauges and Counters Work

- Gauge
  - – Like a Speedometer
  - – Used for rates like load (CPU, interface)
- Counter
  - – Like an Odometer
  - – Used to store counts, can be converted to a rate using deltas

Why Counters?

- Counter are most flexible objects for many performance and fault management tasks:
  - –Subsystem performance monitoring
    - •Errors
    - •Utilization/measure of activity
  - –Most debugging activities require counters
    - •Fault isolation
  - –Resource usage evaluation/planning
    - •Trending and thresholds
  - –Basis for most billing applications

CLI Counters

- Command Line Interfaces
  - –No standards body currently defines one
    - •Yet most CLIs have common traits
  - –Each counter is named
    - •packets input, packets output
  - –CLI Counters start at zero and increase in value
    - •base starting point undefined, usually system start
  - –CLI Counters may also decrease in value
    - •Telco style event performance counters

CLI Counters

- The definition of what a given counter counts is dependent on vendor documentation
  - and on independent observation
- Are formatted for direct human consumption
  - 0 packets input, 0 packets output
- Many implementations provide command to clear/reset counter
  - clear interface ethernet 3

CLI Counters

- Show commands and expect scripts remain the basic way of life in element management

```
c4500#sh int e1
Ethernet1 is up, line protocol is down
Last clearing of "show interface" counters never
Output queue 0/40, 0 drops; input queue 0/75, 0 drops
0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants
   0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
   0 input packets with dribble condition detected
   187352 packets output, 11347294 bytes, 0 underruns
   187352 output errors, 0 collisions, 3 interface resets
```

SNMP Counters

- Allow you to compare apples to apples
  - –Counters have standard definitions
    - •As defined by IETF, IEEE, some vendors…
    - •Regardless of network element type or vendor
  - –And globally unique, hard to pronounce names
    - •1.3.6.1.2.1.17.2.4 dot1dStpTopChanges
- Have a well specified size
  - –32 or 64 bits wide
    - •64 bit data-type available in SNMP v2c or v3
    - •Hacks for SNMPv1 include split counters

SNMP Counters

- Counters do not necessarily start at zero
  - Vendor implementation friendly
- Are not for direct human consumption
  - Require a DELTA function to compute rate
- Can tell if the counter value polled is valid
  - Each counter has a well defined indicator that represents the validity of the sample taken known as a “discontinuity”

SNMP Counters

- Have well defined semantics

```
ifHCInOctets OBJECT-TYPE
SYNTAX      Counter64
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The total number of octets received on the interface,
    including framing characters.  This object is a 64-bit
    version of ifInOctets.

    Discontinuities in the value of this counter can occur at
    re-initialization of the management system, and at other
    times as indicated by the value of
    ifCounterDiscontinuityTime."
            ::= { ifXEntry 6 }
```

SNMP Counters

- Good counters are generally derived from underlying protocol specification

```
dot1dTpPortInFrames OBJECT-TYPE
          SYNTAX  Counter
          ACCESS  read-only
          STATUS  mandatory
          DESCRIPTION
                "The number of frames that have been received by
                this port from its segment. Note that a frame
                received on the interface corresponding to this
                port is only counted by this object if and only if
                it is for a protocol being processed by the local
                bridging function, including bridge management
                frames."
          REFERENCE
                "IEEE 802.1D-1990: Section 6.6.1.1.3"
```

Units specified

Clearly specifies what to count

SNMP Counter Types

- Structure of Management Information
    - Version 1 RFC 1155
    - Version 2 RFC 2578-2580
  - Counter32 / Counter64
  - ZeroBasedCounter32
  - Integer32, Gauge32, are **not** counters
    - but can be the basis for new counter Textual-Conventions

RFC 2493

PerfCurrentCount

PerfIntervalCount

PerfTotalCount

RFC 2856

CounterBasedGauge64

ZeroBasedCounter64

SNMP Counter Types

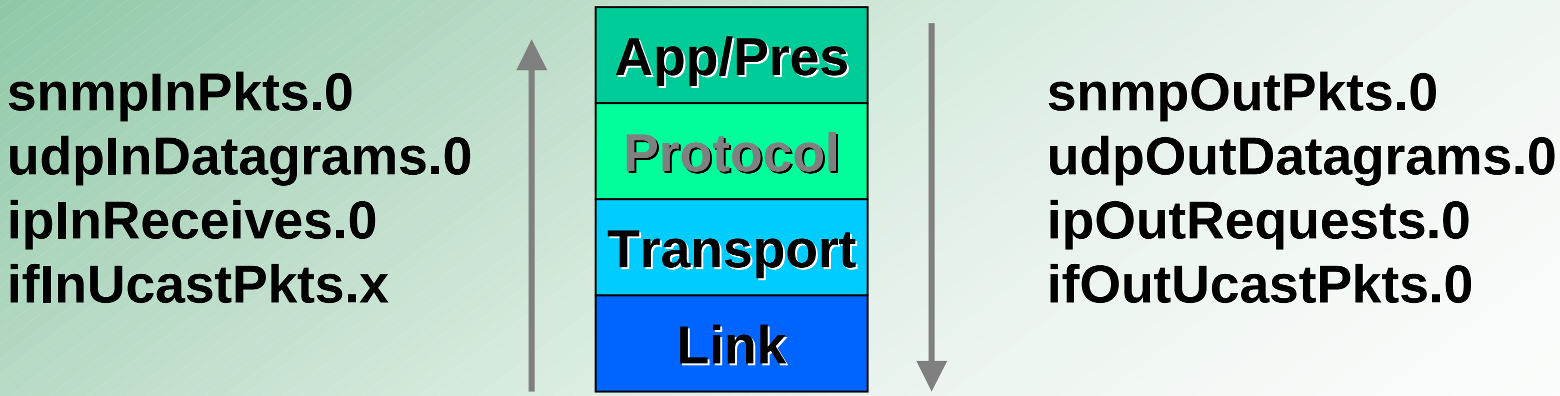
- RFC 2578 Section 7.1.6. Counter32

The Counter32 type represents a non-negative integer which monotonically increases until it reaches a maximum value of 2^32-1 (4294967295 decimal), when it wraps around and starts increasing again from zero

Counters have no defined “initial” value, and thus, a single value of a Counter has (in general) no information content.

SNMP Counter Types

- Beware: retrieving counters can affect the values one is retrieving in-band/out-of-band.
  - –A given SNMP get/getnext to a network element will increment at least these counters:



snmpInPkts.0
udpInDatagrams.0
ipInReceives.0
ifInUcastPkts.x

App/Pres
Protocol
Transport
Link

snmpOutPkts.0
udpOutDatagrams.0
ipOutRequests.0
ifOutUcastPkts.0

Getting Counters—PDU Size

- Understand how large your PDU's are
  - –Standard specifies agent must support 484
  - –MTU of most networks is 1500 bytes
  - –The max SNMP/UDP/IP PDU can be 65518 with ip fragmentation, but is very, very costly and may not be supported by many agents and managers.
  - –Agents have a max PDU size they accept and create
    - •else snmpInTooBig, snmpOutTooBig will increment
    - •and tooBig error returned in packet

Getting Counters—PDU Size

- one ifTable counter, community = 5 bytes
  - net-snmp 4.0 (open source)/snmpget
    - Can fit 80 32-bit integer varbinds per 1500 byte MTU

```
# snoop -S between mgr agent
  mgr -> agent     length: 1498  UDP D=161 S=37913 LEN=1464
  agent -> mgr     length: 1447  UDP D=37913 S=161 LEN=1413
```

  - SNMP Research 15.1.0.8(commercial)/getone
    - Can fit 83 32-bit integer varbinds per 1500 byte MTU

```
  mgr -> agent      length: 1402  UDP D=161 S=53411 LEN=1368
  agent -> mgr      length: 1513  UDP D=53411 S=161 LEN=1479
```

Agenda

- What Is Performance and Fault Management?
- How SNMP Object Types Work
- Data Collection Best Practices
- How to Select the Best Objects for Your Network
- Best Objects for Different Devices and Technologies
- Polling vs. Notifications
- How to “Roll Your Own”
- Q&A

Data Collection Best Practices

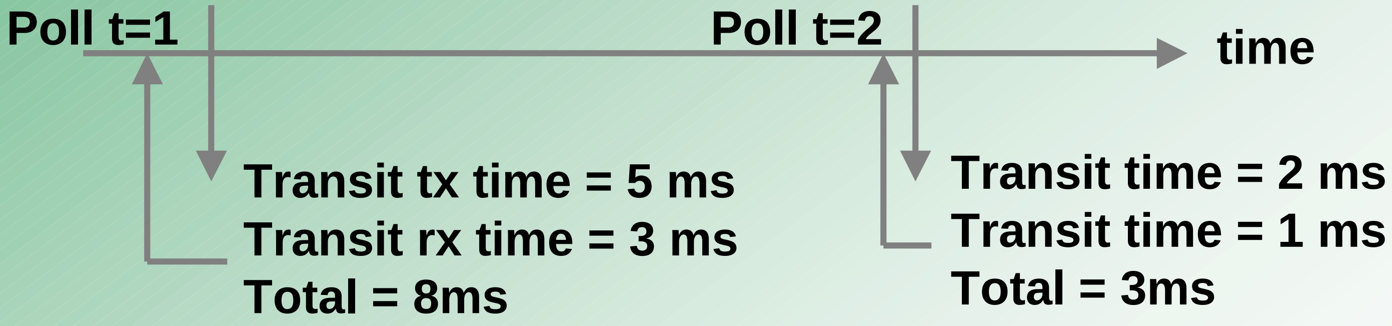
- How you poll counters/form requests can impact the quality of the data for analysis
- How you poll for counters can skew your information/graphs
- Skew defined per Webster's dictionary:
  - To give a bias to; distort
- Time is the major factor causing skew

Data Collection Best Practices
Time Skew

- Group multiple objects in a given Get or GetNext request to minimize time differences in sampling like objects.
- $\text{GET}_{t=1,2,3,..}$ { (x = ifIndex, y = time)
  - ifInUCastPkts.x, ifOutUCastPkts.x,
  - ifInDiscards.x, ifOutDiscards.x
  - ifInErrors.x, ifOutErrors.x
  - sysUpTime.0, ifCounterDiscontinuityTime.x }

Data Collection Best Practices
Time Skew

- When calculating the delta time between two polling requests, use sysUpTime from the device itself and not the management station to avoid transit time skew



Transit tx time = 5 ms
Transit rx time = 3 ms
Total = 8ms

Transit time = 2 ms
Transit time = 1 ms
Total = 3ms

Skew = 5ms

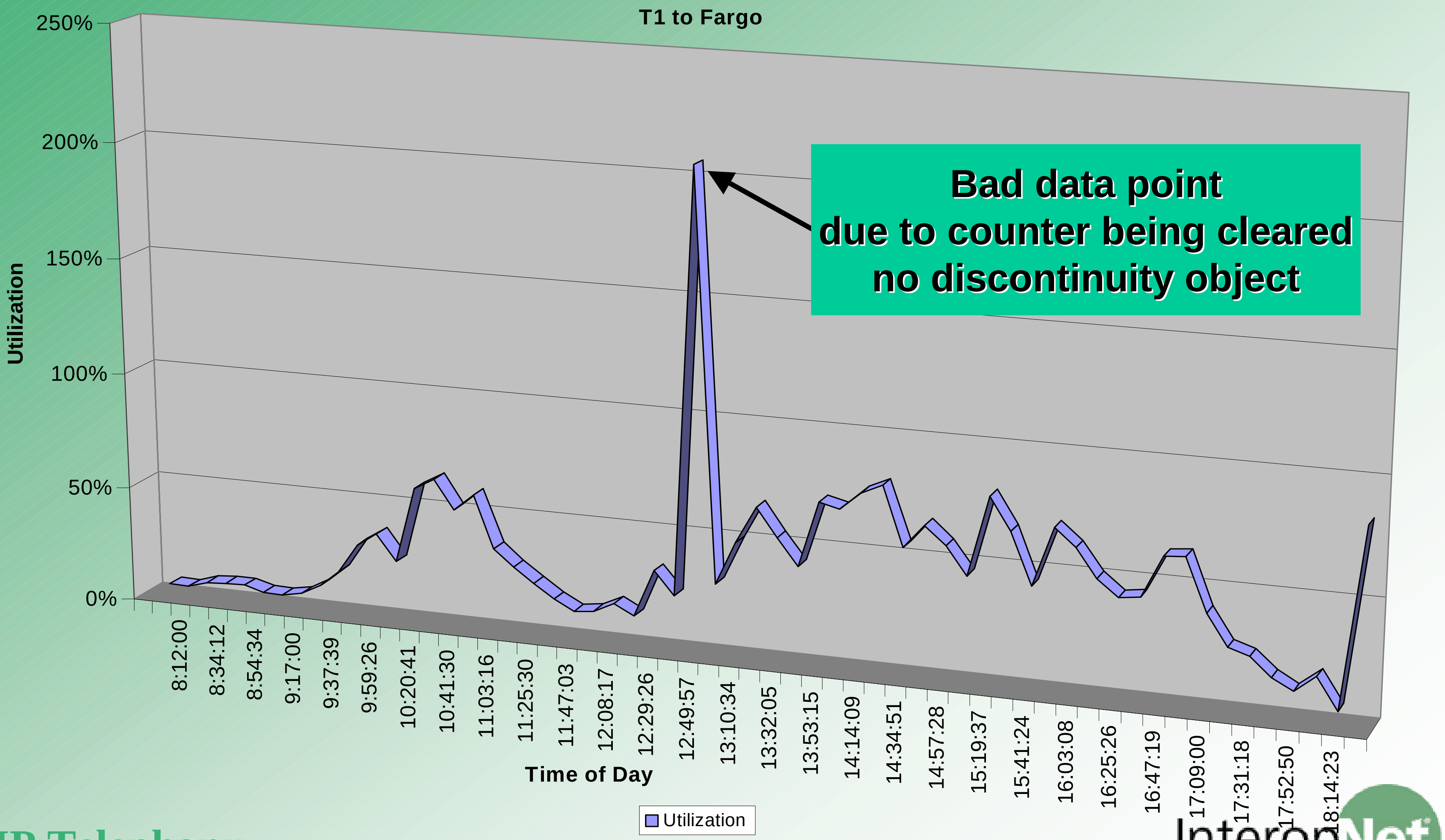
Data Collection Best Practices Using Perf Counters

- All digital circuit interfaces (DS0, DS1, E1, DS3, E3, SONET, SDH) use time based counters
  - –PerfCurrentCount (RFC 2493)
    - •Current Interval counters can decrease in value
    - •Must align polling with device on 15 minute boundaries
  - –PerfIntervalCount
    - •Provides history up to 24 hours in 96 15-minute intervals
- All devices and management stations need to be in time sync—use NTP

Data Collection Best Practices Counter Discontinuity

- Counters should not be reset without a way to determine the reset
  - Leads to inaccurate delta calculations
- Two ways to determine a counter reset:
  - Polling sysUpTime for reset
    - Reset every time SNMP agent is reset
    - Note: sysUpTime wraps every 1.36 years
  - Poll the discontinuity timer if it exists
    - Look in the description of the counter in the MIB module

Data Collection Best Practices
Counter Discontinuity



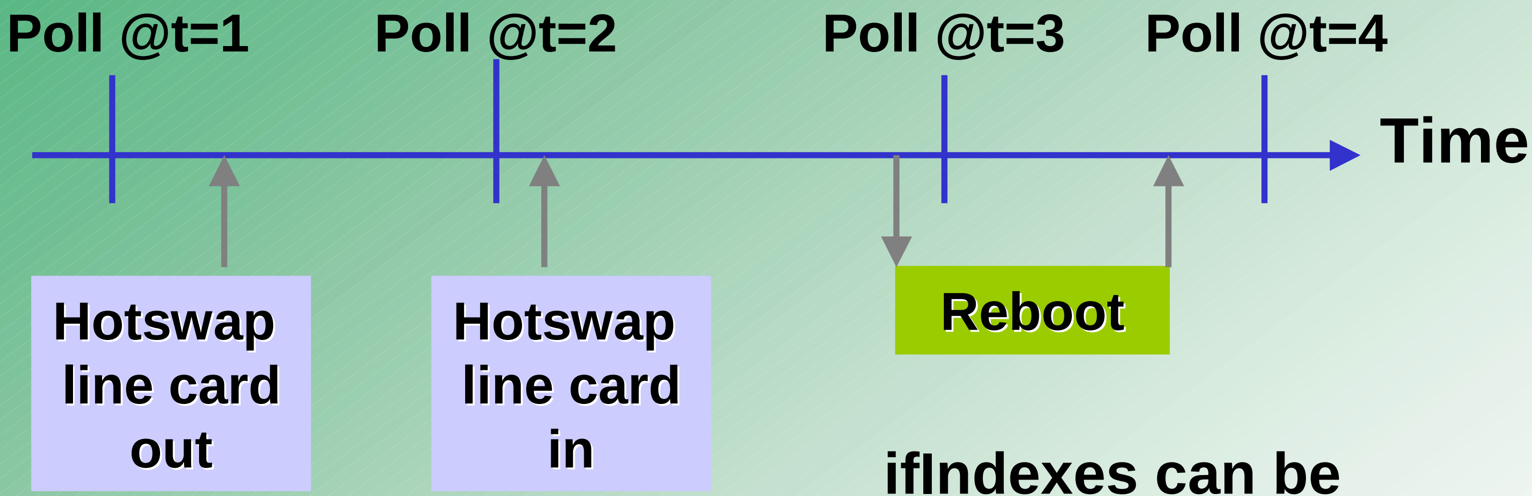
Bad data point
due to counter being cleared
no discontinuity object

Data Collection Best Practices
Counter Discontinuity

- For each counter polled:
  - –collect the discontinuity managed object
- GET$_{y=1,2,3,..}$ { (x == ifIndex, y=time)
  - •*ifCounterDiscontinuityTime.x*,
  - •sysUpTime.0
  - •ifInUCastPkts.x, ifOutUCastPkts.x,
  - •ifInDiscards.x, ifOutDiscards.x
  - •ifInErrors.x, ifOutErrors.x }

Throw out **deltas where discontinuity does not match previously polled value**

Data Collection Best Practices
ifIndex Changing



Poll @t=1

Poll @t=2

Poll @t=3

Poll @t=4

Time

Hotswap line card out

Hotswap line card in

Reboot

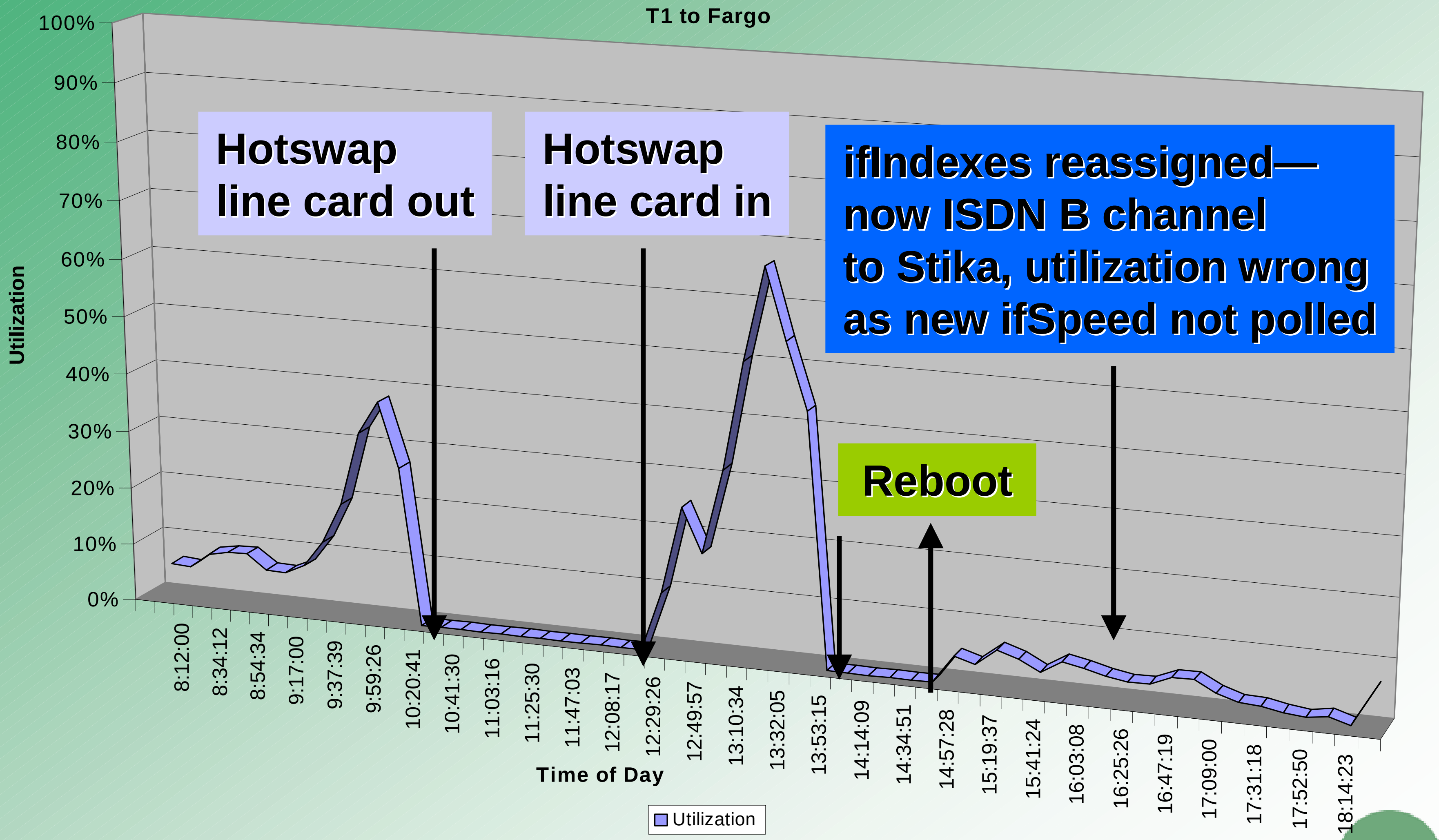
If same type of line-card is reinserted into same slot, ifIndex must be reused.

RFC 2863

ifIndexes can be reassigned across reboots.

Use ifAlias to track reassignment

Data Collection Best Practices
ifIndex Changing



Hotswap line card out

Hotswap line card in

ifIndexes reassigned—now ISDN B channel to Stika, utilization wrong as new ifSpeed not polled

Reboot

Data Collection Best Practices
Setting Polling Interval

- Additional checks in determining minimum poll interval
  - –Verify CPU Load on device is acceptable
  - –Verify management traffic load created is acceptable
  - –Wrap time for a given counter:

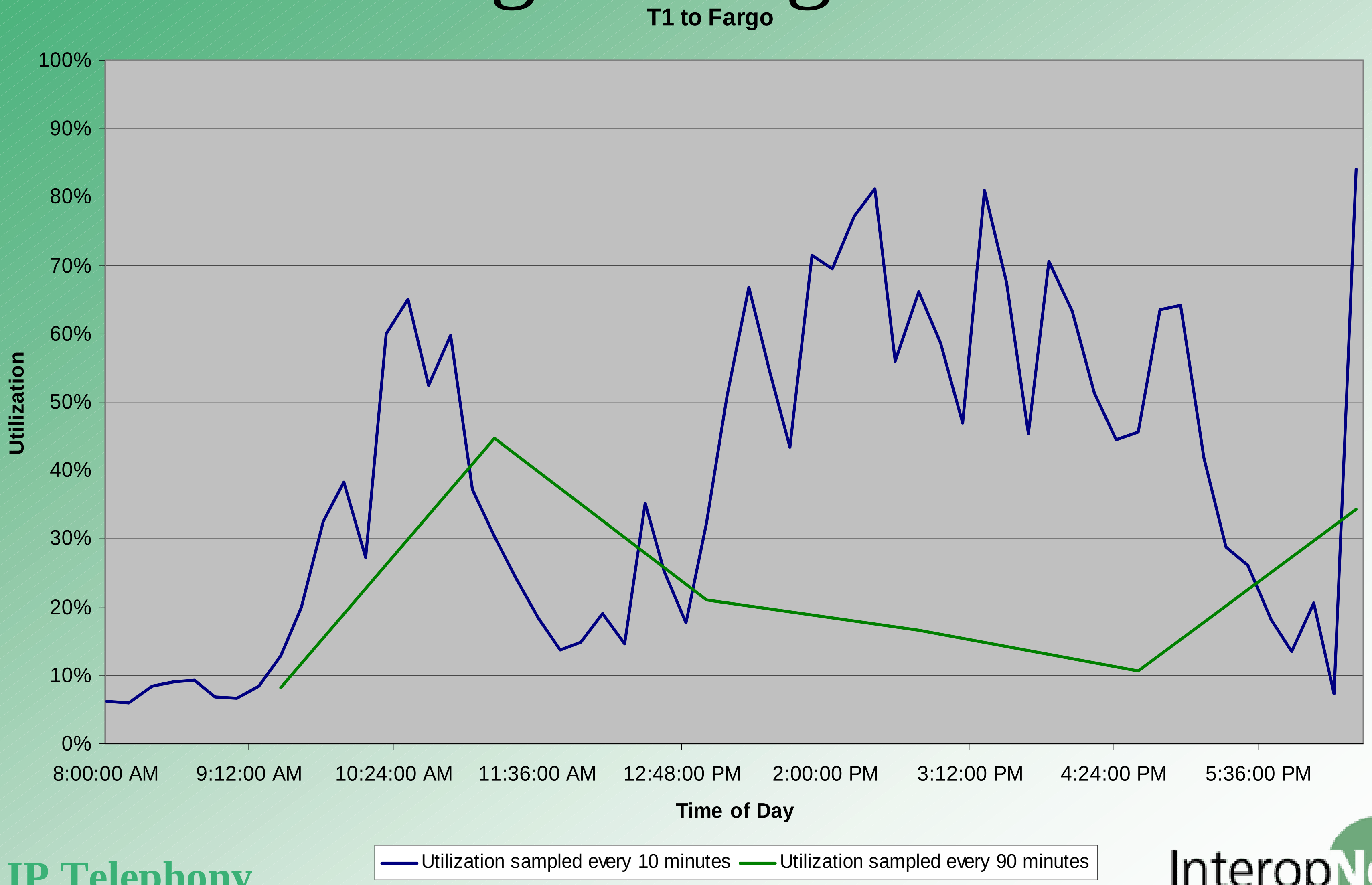
32-bit counters by link speed/sec:

10M	**57.26 minutes**
100M	**5.73 minutes**
155M	**3.69 minutes**
1Gig	**34 seconds**

64-bit counters by link speed/sec:

1 Terabit	**~5 years**
81,000,000 Terabits	**30 minutes**

Data Collection Best Practices
Setting Polling Interval



Data Collection Best Practices Counter Size

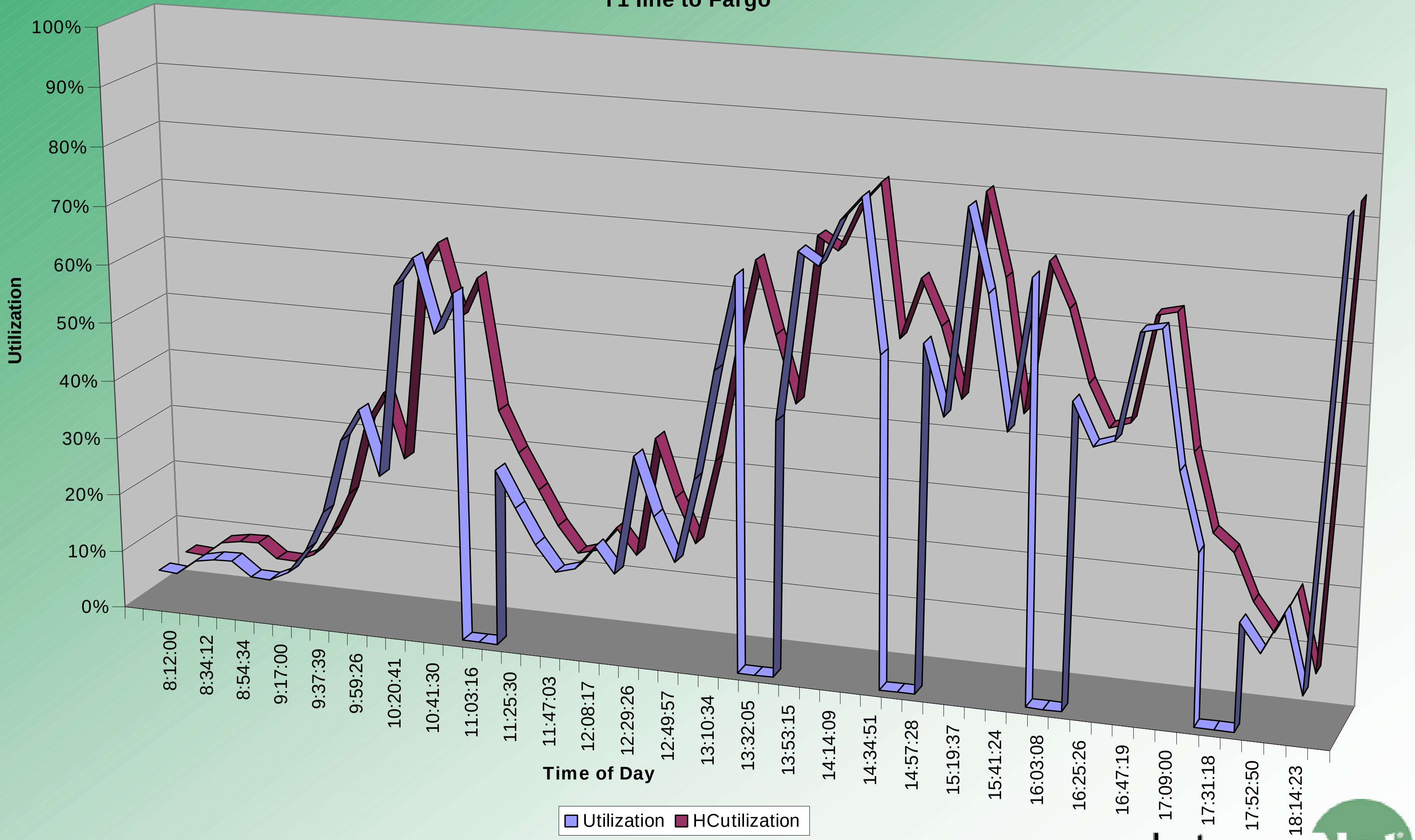
- Determine which counter size to poll for a given managed object
  - 64 bit counters are often named “High Capacity” or HC as in ifHCInOctets
  - Another strategy is high/low 32 bit objects
  - Section 3.1.6 of RFC 2863 provides to vendors as follows for IETF Standards abiding Agent implementations for byte/packet counters

Data Collection Best Practices
Counter Size

- Abiding IETF Implementations will provide byte/packet counters at widths of:
  - –ifSpeed <= 20 Mbps
    - •32-bit byte and packet counters
  - –ifSpeed > 20 Mbps && < 650 Mbps
    - •32-bit packet counters and 64-bit byte counters
  - –ifSpeed >= 650 Mbps
    - •64-bit byte and packet counters
  - –Implementations may provide additional counters, i.e. 64-bit byte counters for 10M interfaces

Data Collection Best Practices
Visualization of Wraps

T1 line to Fargo



Agenda

- What Is Performance and Fault Management?
- How SNMP Object Types Work
- Data Collection Best Practices
- How to Select the Best Objects for Your Network
- Best Objects for Different Devices and Technologies
- Polling vs. Notifications
- How to "Roll Your Own"
- Q&A

How to Select the Best Objects for Your Network

- Find objects
  - Use the OID files
    - ftp://ftp.cisco.com/pub/mibs/oid
  - Look for Standards-based MIBs first
- Drill down using the MIB module
  - Look at MIB to determine if objects meet requirements

How to Select the Best Objects for Your Network

- Determine if MIB is supported in your software release
  - Use supportlists
    - ftp://ftp.cisco.com/pub/mibs/supportlists
  - Use MII to determine specific IOS release and feature set support
    - mii@external.cisco.com
    - No subject, one line for each image name

Agenda

- What Is Performance and Fault Management?
- How SNMP Object Types Work
- Data Collection Best Practices
- How to Select the Best Objects for Your Network
- Best Objects for Different Devices and Technologies
- Polling vs. Notifications
- How to “Roll Your Own”
- Q&A

Polling vs. Notifications

- Polling puts more load on:
  - Network Devices
  - Network Links
  - Network Management Stations

Polling vs. Notifications

- Polling needed for:
  - Availability Monitoring
  - Utilization and Forecasting
- Polling not needed for:
  - Operational Monitoring
  - Fault Monitoring

Polling vs. Notifications

- Notifications puts more load on:
  - Network engineer

Polling vs. Notifications

- How to setup Notifications
  - –RMON Events/Alarms
  - – Available in IOS since 11.1
  - – Available in CatOS always
  - –Event MIB
  - – Available in IOS since 12.1(3)T
- Customize objects for use in Notifications
  - –Expression MIB
  - – Available in IOS since 12.0(5)T, enhanced in 12.1(3)T

How to Reduce the Device and Network Load Caused by Polling

- Network devices have a primary job
  - Routers route
  - Switches switch
- Network Management should improve availability and reliability of the network
  - Should not impact the primary job of network devices

How to Reduce the Device and Network Load Caused by Polling

- SNMP requires lexicographic ordering
- Routing and ARP tables are stored in best order for routing, not SNMP
  - Must be sorted for each SNMP request
  - CPU time required exponentially increases with table size

How to Reduce the Device and Network Load Caused by Polling

- Reduce the number of requests
  - PDU packing
  - get-bulk operator in SNMPv2c or above
  - Use proprietary methods
  - CISCO-BULK-FILE-MIB
  - CISCO-FTP-CLIENT-MIB

How to Reduce the Device and Network Load Caused by Polling

- Eliminate access to tables
  - Use SNMP view
- May have an operational impact
  - Discovery will be slower in OpenView NNM or NetView without access to routing table
  - CiscoWorks 2000 User Tracking will resolve users to only MAC address without access to ARP table

How to Reduce the Device and Network Load Caused by Polling

- Use CEF switching
  - – CEF table is in lexicographic order
  - – SNMP can avoid all routing table-related sorts by using CEF table

How to Reduce the Device and Network Load Caused by Polling

- Reduce duplicate traffic
  - Group devices and manage each device from one or two locations
  - Use trap exploders to reduce notification device and network load

Agenda

- What Is Performance and Fault Management?
- How SNMP Object Types Work
- Data Collection Best Practices
- How to Select the Best Objects for Your Network
- Best Objects for Different Devices and Technologies
- Polling vs. Notifications
- How to “Roll Your Own”
- Q&A

Rolling your Own

- Sometimes the desired object doesn’t exist but can be derived from multiple other objects
  - –Use the Expression MIB to create new objects
- Sometimes the exact notification doesn’t exist but the SNMP objects to trigger the trap do exist
  - –Use the Event MIB to create new notifications
- Sometimes neither the object nor the notification exists, but the object can be derived
  - –Use both the Expression MIB and the Event MIB

Expression MIB

- Based on IETF draft and numbered in Cisco's namespace, not RFC 2982
- Allows you to create new SNMP objects based upon formulas
- Available in IOS since 12.0(5)T, added delta and wildcard support in 12.1(3)T

Expression MIB

- Delta and wildcard support allows, for example:
  - –Calculating Utilization for all interfaces with one expression
    - •This will allow you to recreate the locIfInOctets and locIfOutOctets objects in a standards-based manner—with more flexibility
  - –Calculating errors as a percentage of traffic

Expression MIB

- Show commands
  - show management expression
- Debug commands
  - debug management expression ?

evaluator	Expression MIB evaluator
mib operations	Expression MIB SNMP
parser	Expression MIB parsing

- No CLI configuration support yet

Event MIB

- Based on IETF draft and numbered in Cisco’s namespace, not RFC 2981
- Allows you to create custom Notifications and log them and/or send them as SNMP traps or informs
- Available in IOS since 12.1(3)T

Event MIB

- Show commands
  - – show management event
- Debug commands
  - – debug management event mib
- No CLI configuration support yet

Q&A

Presentation is available at

www.ilabs.interop.net