On fortifying key negotiation schemes with poorly chosen passwords

Indexing term: Information Theory

Abstract: *Key exchange schemes such as Diffie Hellman are vulnerable to middleperson attacks, and thus are often augmented by means of shared secrets. Where these secrets must be memorised, they will usually be vulnerable to guessing attacks. We show how collision rich hash functions can be used to detect such attacks while they are in progress and thus frustrate them.*

Introduction: In communications security design, one of the most important questions is whether an opponent will ever have unsupervised access to the equipment. If the answer is no, then we can greatly simplify the design by storing long term secrets. However, the equipment will then have to be well guarded at all times.

This may be feasible for military equipment, but in the commercial world, physical security procedures are generally insufficient to stop an opponent from getting occasional access. It follows that we must either use tamper resistant hardware, or avoid using long term secrets. In the latter case, the well known Diffie Hellman key negotiation scheme [1] is very useful, and has indeed been used in secure telephone designs.

The problem with Diffie Hellman is of course the middleperson attack; Eve interposes herself into the communications link between Alice and Bob, so that when Alice and Bob try to set up a secure channel, they actually end up with two: one between Alice and Eve, and another between Eve and Bob.

In some telephones, authentication is provided by the parties recognising each others' voices. There are applications, however, where more security is needed. Of course, if the users have the capacity to remember secret keys, then they can use Diffie Hellman followed by a challenge-response protocol to check that there is no intruder in the circuit [2]; but this kind of solution is not always feasible.

Remote login: Consider for example the problem of remote login to a computer system, where a user U wishes to access a host system H. It is well known that humans cannot in general remember good keys, and that the passwords which they are able to remember are likely to succumb to guessing attacks. We shall therefore assume that U and H share a password P with n bits of entropy, while the eavesdropper E can perform 2^n computations.

We shall also assume that although the user's equipment cannot store long term secrets safely, it will not usually have been subverted so as to capture his password directly. This is often realistic, as the user may have a large choice of workstations which he can use, and it will not be economic for the opponent to tap a significant number of them; and the user can always load new software from a commercial distribution diskette, or from an archive on the network (for a secure way to do this see [3]).

However, it is sensible to assume that the network will be compromised. Router information is easy to subvert [4], and the passage of time appears to increase the number of people who are in a position to mount active attacks on networks.

Proposed method: Let g be the generator of a group in which the Diffie Hellman problem is believed to be hard, and let r_I be a random number generated by participant I. Then in the Diffie Hellman scheme, user U would set up a secure session with the host H by sending it g^{r_U}, the host would send g^{r_H}, and they would use $g^{r_U r_H}$ as a session key.

Now if Eve has managed to infiltrate the protocol, U's session key will be $k_U = g^{r_U r_E}$, and H's will be $k_H = g^{r_E r_H}$. Thus the attack can be detected if U and H have some means of comparing session keys. However, as all their communications pass through Eve, who can modify their messages at will, the problem of authentication is not straightforward.

This is especially the case if the user only shares a password with the host. If this password is P, and he simply sends the host $h(P, k_U)$ (where h is a suitable one-way hash function), then Eve can find P by a dictionary search, and pass $h(P, k_H)$ on to the host.

This problem has already been the motivation for other protocols which use passwords to augment Diffie Hellman key exchange, most notably EKE [5]. We propose instead to use collision rich hash functions [3]. Let the function $q(k, x)$ be defined by

$$q(k, x) = h(h(k \mid x) \bmod 2^m) \mid x) \tag{1}$$

where h is a suitable one-way collision-free hash function and $\mid$ denotes concatenation. This function has the feature that it has many collisions on the first variable k, while having no collisions on the second variable x.

The point of this is that U and H can now use their shared password P to check whether their two session keys are the same, without giving this password away to Eve. Firstly, U sends H $q(P, k_U)$; then, if this agrees with H's version of P, he returns $q(h(P), k_H)$. Now if Eve is in fact present, she can perform a dictionary search and finds that 2^{n-m} of the possible passwords satisfy it. The best she can do is to choose one of them, P' at random, and send $q(P', k_H)$ to H.

The probability that $P = P'$ is 2^{m-n}, and H will accept P' if and only if $h(P \mid k_H) = h(P' \mid k_H) \bmod 2^m$. The probability of this is $2^{-m} + 2^{m-n}$, which will be minimised at $2^{1-n/2}$ if we set $m = n/2$.

It would not of course be prudent for the user to continue using a password which had been rejected, as he would have to assume that the rejection was caused by an attack, which reduced the entropy of his password by m bits. Thus an eavesdropper can carry out a denial of service attack. However, this is always possible (for example, using dynamite); the purpose of computer security mechanisms is not to prevent such attacks, but to detect them [5].

In many application environments (such as payment systems), the security manager will want to know at once if such an attack is attempted, and so the procedure will be for the user to call a security hotline for a new password. There might be a mechanism in the user software for dealing with mistyped passwords, in order to minimise the false alarm rate; there are a number of options ranging from typing passwords twice, through displaying a password checksum, to issuing machine generated passwords which contain a controlled amount of redundancy.

Using authentication servers: The above protocol ignores the fact that the host H may use a separate authentication server A to check passwords. For example, H might be a merchant, and A a credit card company which guarantees to pay H for services purchased by properly identified customers.

In this environment, a sensible card issuer would not trust merchant hosts with user passwords, because of the ease with which these hosts can be subverted, and the risk that the opponent will register as a merchant in order to harvest passwords and carry out a major fraud; such frauds are becoming increasingly common against ATM and eftpos systems [7]. Thus the initial session will take place between U and A, with H acting as a communications channel. Once U has set up a key with A, this can be used to set up a key with H.

Conclusions: Collision rich hash functions are a useful tool for building secure communications protocols, especially where one must rely on low quality shared secrets such as passwords. In particular, they enable authenticated key exchange protocols to be simpler than was previously the case.

R J Anderson and T M A Lomas
University Computer Laboratory
Pembroke Street, Cambridge CB2 3QG

References

[1] Diffie W and ME Hellman, "New Directions in Cryptography", in *IEEE Transactions on Information Theory*, **IT-22** no 6 (November 1976) p 650

[2] Diffie W, "Authenticated Key Exchange and Secure Interactive Communications", in *SECURICOM 90*

[3] Lomas TMA and B Christianson, "To Whom am I Speaking?", to appear in *IEEE Computer*

[4] Kumar B, and J Crowcroft, "Integrating Security in Inter-Domain Routing Protocols", in *Computer Communication Review v 39 no 5 (Sep 93) pp 36 - 51*

[5] Bellovin SM and M Merritt, "Augmented Encrypted Key Exchange", in *Proceedings of the 1st ACM Conference on Computer and Communications Security* (1993) pp 244 - 250

[6] Needham RM, "Denial of Service", to appear in *Communications of the ACM*

[7] Anderson RJ, "Why Cryptosystems Fail", in *Proceedings of the 1st ACM Conference on Computer and Communications Security* (1993) pp 215 - 227