The S/Key Protocol

Setup

The user selects an S/Key password K and a maximum sequence number N. The system supplies a seed k. These are hashed together (specifically: k and K are concatenated, run through the MD4 cryptographic hash function, and shortened to 64 bits by xoring the two 64-bit halves) to form h. The system then computes the following sequence of passwords $p_0, \ldots, p_{N-1}$:

$$p_0 = f^N(h);\ p_1 = f^{N-1}(h);\ \ldots;\ p_m = f^{N-m}(h);\ \ldots;\ p_{N-1} = f(h).$$

Note that $p_i = f^{N-i}(h)$, or $p_i = f(p_i-1)$, for $0 \le i < N$.

The system stores the current count m, the seed k, and the last validated password p_{m-1} in a file called *skeykeys*

Validation Algorithm

In the following algorithm, *localhost* is the host which the user is logged in to and *remotehost* is the host that the user is trying to log in to from *localhost*. The S/Key mechanism is to be used.

1. User supplies login name, which is sent to *remotehost*
2. The *remotehost* sends m and k to the *localhost*.
3. User supplies his or her S/Key key K; from that, m, and k, the user (or the *localsystem*) computes h and the next password $p_{\mathrm{m}} = f^{N-m}(h)$. This is transmitted to *remotehost*.
4. The *remotehost* uses the new password to compute $f(p_m) = f(f^{N-m}(h)) = f^{N-m+1}(h) = f^{N-(m-1)}(h) = p_{m-1}$.
5. If the computed password p_{m-1} is the same as the one stored in the *skeykeys* file, the user supplied the correct password, and the *skeykeys* file is updated. If not, the user did not supply the correct password, and the login is denied.