One-time Signature Protocols for Signing Routing Messages

Kan Zhang

Computer Laboratory

Cambridge University

kz200@cl.cam.ac.uk

Attacks on Routing Protocols

- **Replay of old routing messages**
- **Inserting bogus routing messages**

Securing Routing Protocols

Current protection (RIP, OSPF, ISIS, IDRP):

- **Clear-text passwords**

Perlman and others proposed stronger protection mechanisms in which public-key digital signatures are used to provide:

- **Authenticity**
- **Integrity**

of routing messages.

FLS by Hauser, Przygienda and Tsudik

Hash table computed by a router for link L_1 to L_n:

$$
\begin{array}{cccccc}
 & \multicolumn{2}{c}{L_1} & \cdots & \multicolumn{2}{c}{L_n} \\
 & up & down & \cdots & up & down \\
1 & h^1(x_1) & f^1(x_1) & \cdots & h^1(x_n) & f^1(x_n) \\
2 & h^2(x_1) & f^2(x_1) & \cdots & h^2(x_n) & f^2(x_n) \\
\vdots & & \vdots & \ddots & & \vdots \\
k & h^k(x_1) & f^k(x_1) & \cdots & h^k(x_n) & f^k(x_n)
\end{array}
$$

where h and f are two hash functions and x_i are random values.

Limitations

- **Very frequent state changes**
- **Clock drifts**
- **Multiple-valued link costs**
- **Large or changing number of links**
- **Applicability to other routing messages**

One-time Signature Schemes

- **Lamport's original scheme**
  To sign a single bit m, choose x_0 and x_1 and publish $h(x_0)$ and $h(x_1)$

$$s_m = \begin{cases} x_0 & \textbf{if } m = 0 \\ x_1 & \textbf{if } m = 1 \end{cases}$$

- **Improvement by Merkle**

message	00101100
sign	00101100 101

- **Improvement by Winternitz**
- **Authentication tree by Merkle, Vaudenay, Bleichenbacher and Maurer**

Chained One-time Signature Protocol (COSP)

- **Choose at random as secret key components**

$$x_j, \quad j = 1, ..., n.$$

- **Prepare a table of n hash chains of length k:**

$$\begin{array}{llllll} 0 & h^0(x_1), & h^0(x_2), & \cdots, & h^0(x_n) \\ 1 & h^1(x_1), & h^1(x_2), & \cdots, & h^1(x_n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ k & h^k(x_1), & h^k(x_2), & \cdots, & h^k(x_n) \end{array}$$

- **Sign and broadcast the kth row of the table .**

COSP Signing

1. **Obtain a n-bit binary string g by concatenating $f(M_i)$ with a count field using Merkle's method as explained above.**
2. **Form the one-time signature by concatenating the hash values $h^{k-i}(x_j)$ in the $(k-i)$th row of the table for all j such that $g_j = 1$, where g_j is the jth bit of string g.**

COSP Verification

1. **Obtain the n-bit binary string g by concatenating $f(M_i)$ with a count field using Merkle's method as explained above.**
2. **For all j such that $g_j = 1$, check if**

$$h^{i-i'}(r_j) = v_j, \tag{1}$$

   where r_j and v_j are the received and stored value for the jth bit, respectively, and v_j is last updated for message i'.
3. **If true, accept the message and update v_j with value r_j so that when he evaluates Eq. (1) for message $i'' > i$ in the future he only needs to perform $i'' - i$ hash computations.**

Delay-and-Forge Attack

message M_i	00101100	101
message M_{i+1}	01101100	100
fake message M_i'	01101000	101

$$x_2^i = h(x_2^{i+1})$$

- **Signature are sent at pre-set time interval** T
- **Clocks have to be synchronized within time window** T
- **Signatures are valid within time window** T

Independent One-time Signature Protocol (IOSP)

- **To sign message M_i, choose at random as secret key components for next message x'_j, $j = 1, ..., n$ and compute one-time public key P' for next message as $P' = h(h(x'_1)\|\cdots\|h(x'_n))$**
- **Obtain a n-bit binary string g by concatenating $f(M_i\|P')$ with a count field using Merkle's method as explained above.**
- **Compute one-time signature S by concatenating signature components s_j, $j = 1, \cdots, n$, given by**

$$s_j = \begin{cases} h(x_j) & \textbf{if } g_j = 0 \\ x_j & \textbf{if } g_j = 1 \end{cases}$$

where g_j is the jth bit of string g.

IOSP Verification

- **Obtain the n-bit binary string g by concatenating $f(M_i \| P')$ with a count field using Merkle's method as explained above.**
- **Compute $V = h(v_1 \| v_2 \| \cdots \| v_n)$, where v_j, $j = 1, \cdots, n$ is given by**

$$v_j = \begin{cases} r_j & \textbf{if } g_j = 0 \\ h(r_j) & \textbf{if } g_j = 1 \end{cases}$$

  where r_j is the received jth signature component and g_j is the jth bit of string g.
- **If $V = P$, accept the message and update P with value P'.**

Performance

- **COSP verification needs** $l + \lfloor \log_2 l \rfloor + 2$ **hash computations while IOSP needs about half of that.**
- **Signature verification using IOSP runs more than 10 times faster than RSA (MD5 vs. 1024/8 RSA on 200MHz/64MB Pentium PC using CryptoLib 1.1)**
- **Both COSP and IOSP signature generation takes negligible time, whereas RSA signature generation is about 100 times slower than verification**

Comparison of COSP and IOSP

- **Advantages of IOSP**
  - **Signature verification runs twice as fast as COSP**
  - **Less memory for storing keys**
  - **No timing constraint**

- **Advantages of COSP**
  - **The signature size of COSP is roughly half of that of IOSP (2KB for IOSP and 1KB for COSP using MD5)**
  - **Easy to catch up**

Applicability as efficient alternatives to public-key signatures

- **Fast signature generation and verification**
- **Non-interactive**

As a general approach, the way our protocols being used with public-key systems for message signing is similar to that of secret-key cryptography being used with public-key systems for data encryption.