Reducing the Cost of Security in Link-State Routing

R. Hauser	A. Przygienda	G. Tsudik
McKinsey Consulting	Fore Systems	USC-ISI
Zürich, Switzerland	Bethesda, MD	Marina Del Rey, CA
hauser@acm.org	prz@fore.com	gts@isi.edu

ISOC Symposium on Network and Distributed System Security, Feb 10-11, 1996, San Diego, CA

Introduction

Routing Protocols:

- Ford/Fulkerson's Max Flow $\longrightarrow$ *Distance Vector*
  e.g., OSPF, IDPR, ATM-PNNI
- Dijkstra's Shortest Path $\longrightarrow$ *Link State*,
  e.g., RIP, BGP, IDRP

Focus of this work:

$\boxed{\longrightarrow \textit{ How to minimize the cost of security in Link State Routing?}}$

Link State Security

Link State (LS) Security Requirements:

1. Origin Authentication
2. Non-repudiation
3. Data integrity
4. Timeliness and Ordering

Building Blocks:

- Public key-based digital signatures
  (RSA, DSS, El Gamal, Schnorr, etc.)
- (Conjectured/alleged) one-way hash functions
  (MD5, 8-pass SNEFRU, SHA, etc.)
- Hash chain constructs
  (e.g., S/KEY one-time authentication, micro-payments, etc.)

Hash Chains

Example:

1. *Alice generates a secret* R

2. *Computes a hash chain of length* n*:*
$$\mathcal{H}^1(R),\ \ldots,\ \mathcal{H}^i(R),\ \ldots\ \mathcal{H}^n(R)$$
where $\mathcal{H}^0(R) = R$ *and* $\mathcal{H}^i(R) = \mathcal{H}(\mathcal{H}^{i-1}(R))$ *for* $0 < i < n$

3. *Initially, Bob receives* $\mathcal{H}^n(R)$

4. *Alice releases* $\mathcal{H}^{n-1}(R)$

5. *Bob checks that* $\mathcal{H}(\mathcal{H}^{n-1}(R))$ *matches* $\mathcal{H}^n(R)$*.*

Last two steps can be repeated $n-1$ times

Stable Link State – SLS

Observation

A large percentage (50%, by some estimates) of LSU-s are simply re-statements of previous LSU-s, i.e., an LSU often carries no new information other than its timing since links and nodes go up and down infrequently.

LSU types:

1. **Anchor** LSU – $ALSU$
   generated whenever a link state change occurs or the current hash chain is depleted. Signed, sequenced, timestamped.
   Contains: $\mathcal{H}^n(R)$, T_0, n, $\{LINKS\}$, SIG
2. **Chained** LSU – $CLSU_i$
   generated periodically or upon explicit request Unsigned, sequenced, timestamped.
   Contains: $\mathcal{H}^{n-i}(R)$, T_i, i

SLS contd.

Issues:

- Missing CLSU-s?
- Storage Requirements?
- Only effective in **STABLE** routing environments!

Fluctuating Link State – FLS

Observation

The state of a link is (typically) a *binary* value.

- Each node generates $(n \times\ k \times\ s)$ hash chains

  n - chain length

  k - # of incident links

  s - # of possible link states (typically 2: UP and $DOWN$)

- All R_j $(1 \leq\ j \leq\ k)$ must be random and unique,
- ALSU contains:

  $[nodeID, T_n, \mathcal{H}^n(R_1), \mathcal{G}^n(R_1), ..., \mathcal{H}^n(R_j), \mathcal{G}^n(R_j), ..., \mathcal{H}^n(R_k), \mathcal{G}^n(R_k)]^{SK}$

Hash Table

	L_1		...	L_j		...	L_k	
	up	down	...	up	down	...	up	down
1	$\mathcal{H}^1(R_1)$	$\mathcal{G}^1(R_1)$	...	$\mathcal{H}^1(R_j)$	$\mathcal{G}^1(R_j)$	...	$\mathcal{H}^1(R_k)$	$\mathcal{G}^1(R_k)$
.	.	.	...	.	.	...	.	.
.	.	.	...	.	.	...	.	.
.	.	.	...	.	.	...	.	.
i	$\mathcal{H}^i(R_1)$	$\mathcal{G}^i(R_1)$	...	$\mathcal{H}^i(R_j)$	$\mathcal{G}^i(R_j)$	...	$\mathcal{H}^i(R_k)$	$\mathcal{G}^i(R_k)$
.	.	.	...	.	.	...	.	.
.	.	.	...	.	.	...	.	.
.	.	.	...	.	.	...	.	.
n	$\mathcal{H}^n(R_1)$	$\mathcal{G}^n(R_1)$	...	$\mathcal{H}^n(R_j)$	$\mathcal{G}^n(R_j)$	...	$\mathcal{H}^n(R_k)$	$\mathcal{G}^n(R_k)$

CLSU Construction

For each link L_j, $(1 \leq \ j \leq \ k)$ and for each $CLSU_i$, $(1 \leq \ i < \ n)$ link state flags (LSF_i) is defined as:

$$LSF_i = [LF_i(1), ..., LF_i(k)]$$

where:

$$LF_i(j) = \begin{cases} 1 \text{ if } L_j \text{ is UP} \\ 0 \text{ if } L_j \text{ is DOWN} \end{cases}$$

For each link L_j, $(1 \leq \ j \leq \ k)$ and for each $CLSU_i$, $(1 \leq \ i < \ n)$ link state vector (LSV_i) is defined as:

$$LSV_i = [LS_i(1), ..., LS_i(k)]$$

where:

$$LS_i(j) = \begin{cases} \mathcal{H}^{n-i}(R_j) \text{ if } LF_i(j) = 1 \\ \mathcal{G}^{n-i}(R_j) \text{ if } LF_i(j) = 0 \end{cases}$$

$CLSU_i$ contains: $[nodeID, i, T_i, LSF_i, LSV_i]$

CLSU Processing

1. Looks up the current entry for $nodeID$

2. Validates T_i and i:

   Checks that T_i is valid (reasonably close to current time), $i > p$ and $T_i > T_p$ (last stored timestamp from $CLSU_p$.)

3. For each link L_j reflected in $CLSU_i$ $(0 < j < k)$:

   a) if L_j's state is unchanged $(LF_i(j) = LF_p(j))$, compute:

   $$\begin{array}{ll} \mathcal{G}^{i-p}(LS_i(j)) & \text{if } LF_i(j) = 0 \\ \mathcal{H}^{i-p}(LS_i(j)) & \text{if } LF_i(j) = 1 \end{array}$$

   and compare to $LS_p(j)$; reject upon mismatch.

   b) if L_j's state has changed $(LF_i(j) \neq LF_p(j))$, compute:

   $$\begin{array}{ll} \mathcal{G}^{i}(\mathcal{G}^{n-i}(R_j)) & \text{if } LF_i(j) = 0 \\ \mathcal{H}^{i}(\mathcal{H}^{n-i}(R_j)) & \text{if } LF_i(j) = 1 \end{array}$$

   and compare to $LS_n(j)$; reject upon mismatch.

Replace LSV_p with LSV_i.

Analysis

Security (both SLS and FLS):

1. Strength of the underlying signature function (wrt ALSUs)
2. Strength of the underlying hash function (wrt CLSUs)
3. Randomness of the starting values
4. Loose clock synchronization: maximum skew $= (2 \times t)$

Limitations:

- Very frequent state oscillations
- Clock synchronization impossible
- Multiple-valued (or continuous) link state

Conclusions

Related Work:

[MB-96] S. Murphy and M. Badger, *Digital Signature Protection of the OSPF Routing Protocol*, 1996 Symposium on Network and Distributed Systems Security (SNDSS'96), February 1996.

[P-88] R. Perlman, *Network Layer Protocols with Byzantine Robustness*, Ph.D. Dissertation, MIT LCS TR-429, October 1988.

Future Work:

- Experimental Results (OSPF)
- SLS/FLS Hybrid
- Other Constructs?