A Logical Reconstruction of SPKI

Joe Halpern
Cornell University

Ron van der Meyden
University of New South Wales

SDSI/SPKI

SDSI (Simple Distributed Security Infrastructure) [Rivest/Lampson]

- principals identified with public keys
- each principal has *local names*
  - In Ron's name space, `Joe's poker-buddies` refers to the set of principals associated with `poker-buddies` in Joe's name space
  - In earlier work [CSFW '99/J. Computer Security '01], we gave a logic LLNC (Logic of Local Name Containment) for capturing SDSI's operational name resolution algorithm.

SDSI has been incorporated into SPKI (Simple Public Key Infrastructure):

- allows expiry dates for certificates and revocation
- deals with authorization and delegation

Goal of this work: extend the earlier approach to dealing with this new features.

Monotonicity and Revocation

SDSI is monotonic: more certificates $\rightarrow$ more keys may be bound to a given name.

Revocation means that that extra information could result in fewer bindings.

- LLNC is monotonic
  - [Ninghui Li (CSFW '00) erroneously claimed LLNC is nonmonotonic.]
- Do we need nonmonotonicity to handle revocation?
  - No!

We give a monotonic logic with natural semantics that can capture SPKI's tuple reduction rules.

SPKI Syntax I: Names

- SPKI views authority as being associated with principals = public keys.
- Instead of global names, SPKI has local name spaces, like SDSI.

A SPKI name is either

- a key in some set K of keys,
- a local name (byte string) in some set N, or
- a *compound name* $(\texttt{name}\ \texttt{n}_1\ \texttt{n}_2 \ldots \texttt{n}_k)$, $\texttt{n}_i \in K \cup N$

For simplicity we ignore other ways SPKI has of describing principals, like hashes and threshold subjects.

- There are a few other minor simplification and white lies in talk, to simplify the presentation.

SPKI Syntax II: Certificates

A *naming certificate* has the form

$$(\texttt{cert k n p V k}_r).$$

- Certificate binds name `p` to the local name `n` in `k`'s local name space during the interval $\texttt{V} = [t_1, t_2]$ provided that $\texttt{k}_r$ does not revoke the certificate.
  - $\texttt{k}_r$ is optional

Authorization certificates have the form

$$(\texttt{cert k p A D V k}_r)$$

- `k` allows `p` to perform the actions in `A` (and to delegate this authority, if Boolean $\texttt{D} = \textit{true}$) during interval `V`, provided that $\texttt{k}_r$ does not revoke the certificate.

A *certificate revocation list* (CRL) issued by `k` has the form

$$(\texttt{crl k (canceled } \texttt{c}_1, \ldots, \texttt{c}_\texttt{n}\texttt{) V})$$

- according to `k`, the certificates $\texttt{c}_1, \ldots, \texttt{c}_\texttt{n}$ are revoked during the interval V.

SPKI's Tuple Reduction Rules

To see if collection C of certificates authorizes certain actions,

1. first remove tuples c which are legitimately revoked in a CRL in C
2. convert each remaining naming/authorization certificate $\mathtt{c}$ to 4/5 tuple $\tau_{\mathtt{C}}$ by removing word $\mathtt{cert}$ and "revoker" $\mathtt{k}_r$
   - if $\mathtt{c} = (\mathtt{cert\ k\ n\ p\ V\ k}_r)$, then $\tau_{\mathtt{C}}$ is $\langle \mathtt{k}, \mathtt{n}, \mathtt{p}, \mathtt{V} \rangle$.
3. rewrite tuples according to rules below:

R1. $\langle \mathtt{k}_1, \mathtt{k}_2, \mathit{true}, \mathtt{A}_1, \mathtt{V}_1 \rangle + \langle \mathtt{k}_2, \mathtt{p}, \mathtt{D}_2, \mathtt{A}_2, \mathtt{V}_2 \rangle$
$\longrightarrow \langle \mathtt{k}_1, \mathtt{p}, \mathtt{D}_2, \mathtt{A}_1 \cap \mathtt{A}_2, \mathtt{V}_1 \cap \mathtt{V}_2 \rangle$

R2. $\langle \mathtt{k}_1, \mathtt{n}, \mathtt{k}_2\text{'s } \mathtt{m}\text{'s } \mathtt{p}, \mathtt{V}_1 \rangle + \langle \mathtt{k}_2, \mathtt{m}, \mathtt{k}_3, \mathtt{V}_2 \rangle$
$\longrightarrow \langle \mathtt{k}_1, \mathtt{n}, \mathtt{k}_3\text{'s } \mathtt{p}, \mathtt{V}_1 \cap \mathtt{V}_2 \rangle$

R3. $\langle \mathtt{k}_1, \mathtt{k}_2\text{'s } \mathtt{n}\text{'s } \mathtt{p}, \mathtt{D}, \mathtt{A}, \mathtt{V}_1 \rangle + \langle \mathtt{k}_2, \mathtt{n}, \mathtt{k}_3, \mathtt{V}_2 \rangle$
$\longrightarrow \langle \mathtt{k}_1, \mathtt{k}_3\text{'s } \mathtt{p}, \mathtt{D}, \mathtt{A}, \mathtt{V}_1 \cap \mathtt{V}_2 \rangle$

A Logic for Reasoning about SPKI: Syntax

Primitives:

- *principal expressions*: either an element of $K \cup N$ or has the form $\mathtt{p}$'s $\mathtt{q}$, where $\mathtt{p}$, $\mathtt{q}$ are principal expressions;
- the set $\mathcal{C}$ of certificates;
- special constant $\mathtt{now}$;
- *validity intervals* $[t_1, t_2]$, $t_1 \leq t_2 \leq \infty$.

Formulas:

- $\mathtt{p} \longmapsto \mathtt{q}$, for principal expressions $\mathtt{p}, \mathtt{q}$ is a formula;
- $\mathtt{c}$ and $\mathit{valid}(\mathtt{c})$, for $\mathtt{c} \in \mathcal{C}$;
- $\mathit{Perm}(\mathtt{k}, \mathtt{p}, A)$ and $\mathit{Del}(\mathtt{k}, \mathtt{p}, A)$, for key $\mathtt{k}$, principal expression $\mathtt{p}$, set A of actions;
- $\mathtt{now} \in V$;
- $\neg\varphi$, $\varphi \wedge \psi$.

Call the resulting language $\mathcal{L}_{SPKI}$.

LLNC is the fragment of $\mathcal{L}_{SPKI}$ with only naming certificates:

- ($\texttt{cert k n p}$) corresponds to the LLNC formula $\texttt{k}\ \mathit{cert}\ \texttt{n} \longmapsto \texttt{p}$.

LLNC does not deal with time, permission, delegation, or revocation.

A Logic for Reasoning about SPKI: Semantics

The semantics for $\mathcal{L}_{SPKI}$ extends that of LLNC. Major components:

- a *run*: a function $r : I\!N \to \mathcal{P}(\mathcal{C}))$.
  - $\mathtt{c} \in r(\mathtt{t})$ if certificate $\mathtt{c}$ issued at time $\mathtt{t}$ in r
- a *local name assignment*: a function $L : K \times N \times I\!N \to \mathcal{P}(K)$.
  - $L(\mathtt{k}, \mathtt{n}, \mathtt{t})$ contains the keys associated at time $\mathtt{t}$ with the name $\mathtt{n}$ in $\mathtt{k}$'s name space.
- a *permission/delegation assignment*: a function $P : K \times I\!N \to \mathcal{P}(K \times Act_{\mathcal{A}} \times \{0, 1\})$ such that
  1. if $\langle \mathtt{k}', \mathtt{a}, 0\rangle \in P(\mathtt{k}, \mathtt{t})$ then $\langle \mathtt{k}', \mathtt{a}, 1\rangle \notin P(\mathtt{k}, \mathtt{t})$,
  2. if $\langle \mathtt{k}_2, \mathtt{a}, 1\rangle \in P(\mathtt{k}_1, \mathtt{t})$ and $\langle \mathtt{k}_3, \mathtt{a}, i\rangle \in P(\mathtt{k}_2, \mathtt{t})$ then $\langle \mathtt{k}_3, \mathtt{a}, i\rangle \in P(\mathtt{k}_1, \mathtt{t})$.
  - If $\langle \mathtt{k}', \mathtt{a}, i\rangle \in P(\mathtt{k}, \mathtt{t})$, $\mathtt{k}$ has granted permission to $\mathtt{k}'$ to perform action $\mathtt{a}$ at time $\mathtt{t}$; if $i = 1$, $\mathtt{k}$ has delegated authority to $\mathtt{k}'$ to propagate the right to perform action $\mathtt{a}$.

An *interpretation* π is a pair $\langle L, P\rangle$.

Interpreting Names

Given a local name assignment L, a key$\mathtt{k}$, and a time $\mathtt{t}$, each principal expression $\mathtt{p}$ is assigned a set of keys $[\![\mathtt{p}]\!]_{L,\mathtt{k},\mathtt{t}}$:

- $[\![\mathtt{k}']\!]_{L,\mathtt{k},\mathtt{t}} = \{\mathtt{k}'\}$, if $\mathtt{k}' \in K$ is a key,
- $[\![\mathtt{n}]\!]_{L,\mathtt{k},\mathtt{t}} = L(\mathtt{k}, \mathtt{n}, \mathtt{t})$, if $\mathtt{n} \in N$ is a local name,
- $[\![\mathtt{p}\text{'s } \mathtt{q}]\!]_{L,\mathtt{k},\mathtt{t}} = \cup\{[\![\mathtt{q}]\!]_{L,\mathtt{k}',\mathtt{t}} \mid \mathtt{k}' \in [\![\mathtt{p}]\!]_{L,\mathtt{k},\mathtt{t}}\}$.

This definition is essentially identical to that in [Abadi98,HM99/01].

Interpreting Formulas

Truth of a formula is defined with respect to a run r, interpretation $\pi = \langle L, P \rangle$, key $\mathtt{k}$, and time $\mathtt{t}$.

Define $r, \pi, \mathtt{k}, \mathtt{t} \models \varphi$ by induction on structure of φ:

- $r, \pi, \mathtt{k}, \mathtt{t} \models \mathtt{p} \longmapsto \mathtt{q}$ if $[\![\mathtt{p}]\!]_{L,\mathtt{k},\mathtt{t}} \supseteq [\![\mathtt{q}]\!]_{L,\mathtt{k},\mathtt{t}}$,
- $r, \pi, \mathtt{k}, \mathtt{t} \models \mathtt{c}$ if $\mathtt{c} \in r(\mathtt{t}')$ for some $\mathtt{t}' \leq \mathtt{t}$,
- $r, \pi, \mathtt{k}, \mathtt{t} \models \mathit{Perm}(\mathtt{k}_1, \mathtt{p}, A)$ if for all $\mathtt{k}_2 \in [\![\mathtt{p}]\!]_{L,\mathtt{k}_1,\mathtt{t}}$ and $\mathtt{a} \in A$, $\langle \mathtt{k}_2, \mathtt{a}, i \rangle \in P(\mathtt{k}_1, \mathtt{t})$ for some $i \in \{0, 1\}$,
- $r, \pi, \mathtt{k}, \mathtt{t} \models \mathit{Del}(\mathtt{k}_1, \mathtt{p}, A)$ if for all $\mathtt{k}_2 \in [\![\mathtt{p}]\!]_{L,\mathtt{k}_1,\mathtt{t}}$ and $\mathtt{a} \in A$, we have $\langle \mathtt{k}_2, \mathtt{a}, 1 \rangle \in P(\mathtt{k}_1, \mathtt{t})$,
- $r, \pi, \mathtt{k}, \mathtt{t} \models \mathtt{now} \in V$ if $\mathtt{t} \in V$,
- $r, \pi, \mathtt{k}, \mathtt{t} \models \mathit{valid}(\mathtt{c})$ if $\mathtt{c}$ is *valid*: it was issued before time $\mathtt{t}$ in r and not revoked,
- $r, \pi, \mathtt{k}, \mathtt{t} \models \varphi \wedge \psi$ if $r, \pi, \mathtt{k}, \mathtt{t} \models \varphi + r, \pi, \mathtt{k}, \mathtt{t} \models \psi$,
- $r, \pi, \mathtt{k}, \mathtt{t} \models \neg\varphi$ if not $r, \pi, \mathtt{k}, \mathtt{t} \models \varphi$.

Consistency

So far, there is no connection between the run and the interpretation.

- We want the meaning of local names and information about permissions and delegations given in the interpretation to be determined by the information given in the run.

$\pi = \langle L, P \rangle$ is *consistent* with r if, for all times $\mathtt{t} \in I\!N$,

1. if naming certificate $(\mathtt{cert\ k\ n\ p\ V\ k}_r)$ is valid at $\mathtt{t}$ in r, then $[\![\mathtt{n}]\!]_{L,\mathtt{k},\mathtt{t}} \supseteq [\![\mathtt{p}]\!]_{L,\mathtt{k},\mathtt{t}}$;
2. if authorization certificate $(\mathtt{cert\ k\ p\ A\ D\ V\ k}_r)$ is valid at $\mathtt{t}$ in r, then
   (a) $\langle \mathtt{k}', \mathtt{a}, i \rangle \in P(\mathtt{k}, \mathtt{t})$ for some $i \in \{0, 1\}$,
   (b) if $\mathtt{D} = \mathtt{true}$ then $\langle \mathtt{k}', \mathtt{a}, 1 \rangle \in P(\mathtt{k}, \mathtt{t})$.

Consistency by itself is not enough:

- the run in which no certificates are ever issued is consistent with an interpretation where every key is permitted to perform every action.

Minimal Interpretations

Want the interpretation to capture what is forced by the certificates and no more.

Define an order $\leq$ on interpretations:

$\langle L, P\rangle \leq \langle L', P'\rangle$ if $L(\mathtt{k},\mathtt{n},\mathtt{t}) \subseteq L'(\mathtt{k},\mathtt{n},\mathtt{t})$ for all $\mathtt{k},\mathtt{n},\mathtt{t}$, and if $(\mathtt{k}',\mathtt{a},i) \in P(\mathtt{k},\mathtt{t})$, then $(\mathtt{k}',\mathtt{a},i') \in P'(\mathtt{k},\mathtt{t})$ for some $i' \geq i$.

Proposition: For every run r there exists a unique interpretation π_r minimal in the set of interpretations consistent with r.

Definition: $r,\mathtt{k},\mathtt{t} \models_c \varphi$ if $r,\pi_r,\mathtt{k},\mathtt{t} \models \varphi$.

- φ is *c-valid* (wrt set K of keys), written $\models_{c,K} \varphi$, if $r,\mathtt{k},\mathtt{t} \models_c \varphi$ for all r, $\mathtt{k} \in K$, and $\mathtt{t}$.
  - Sometimes K matters; we make it explicit if it does.

Characterizing Certificates

A certificate $\texttt{c}$ has an associated formula φ_{tc}

- If $\texttt{c}$ is the naming certificate $(\texttt{cert k n p V})$, then $\varphi_{\texttt{C}}$ is

$$\texttt{now} \in V \Rightarrow (\texttt{k}\text{'s } \texttt{n} \longmapsto \texttt{p}).$$

- If $\texttt{c}$ is the authorization certificate $(\texttt{cert k p A D V})$, then $\varphi_{\texttt{C}}$ is

$$\texttt{now} \in V \Rightarrow [\mathit{Perm}(\texttt{k}, \texttt{p}, \texttt{A}) \land (\texttt{D} \Rightarrow \mathit{Del}(\texttt{k}, \texttt{p}, \texttt{A}))].$$

Proposition: If $\texttt{c} \in \mathcal{C}$ then $\models_c \texttt{c} \land \mathit{valid}(\texttt{c}) \Rightarrow \varphi_{\texttt{C}}$.

If a certificate was issued in a run r and remains valid, then the associated formula is true in the minimal interpretation consistent with r.

Conversely, the minimal interpretation consistent with a run is the minimal one satisfying all the formulas associated with the currently valid certificates that have been issued.

Proposition: An interpretation π is consistent with a run r if, for all times $\texttt{t}$, keys $\texttt{k}$, and certificates $\texttt{c}$:

$$r, \pi, \texttt{k}, \texttt{t} \models \texttt{c} \land \mathit{valid}(\texttt{c}) \Rightarrow \varphi_{\texttt{C}}.$$

What we have so far:

- An expressive logic for reasoning about SPKI:
  - The logic can talk about permission, delegation, validity of certificates, names
  - It has a natural semantics.
- A way of translating certificates into the logic.

What we want:

- To connect the tuple reduction process to reasoning in the logic.

Soundness of Tuple Reduction

Given naming and authorization certificates C and CRLs C_R, let $\texttt{Tuples}(C, C_R)$ be the tuples corresponding to certificates in C that are guaranteed not to have been revoked:

- E.g., if
  - $\texttt{c} = (\texttt{cert k n p V k}_r) \in C$,
  - $(\texttt{crl k}_r\ (\texttt{canceled c}_1, \ldots, \texttt{c}_n)\ \texttt{V}') \in C_R$, and
  - $\texttt{c} \neq \texttt{c}_i$, $i = 1, \ldots, n$,

  then $\texttt{c} = (\texttt{k n p V} \cap \texttt{V}') \in \texttt{Tuples}(C, C_R)$.
  - Important assumption: CRLs are issued for *non-overlapping* intervals.
- Key point: $\texttt{Tuples}(C, C_R)$ is *monotonic* in both C and C_R

Theorem: If $\texttt{Tuples}(C, C_R) \longrightarrow^* \tau_{\mathsf{c}}$, then

$$\models_c \left(\bigwedge_{\mathsf{c}' \in C \cup C_R} \mathsf{c}' \right) \Rightarrow \varphi_{\mathsf{c}}.$$

Completeness

Completeness is somewhat more subtle.

A *concrete certificate* has a corresponding tuple of the form $\langle \mathtt{k}, \mathtt{n}, \mathtt{k}', [\mathtt{t}, \mathtt{t}] \rangle$ (in the case of naming certificates) or $\langle \mathtt{k}, \mathtt{k}', \mathtt{D}, \{\mathtt{a}\}, [\mathtt{t}, \mathtt{t}] \rangle$ (in the case of authorization certificates).

- concrete certificates talk about the keys that are bound to names and the keys that are authorized to perform single actions at a single point in time.

$\langle \mathtt{k}, \mathtt{n}, \mathtt{k}', \mathtt{V} \rangle$ subsumes $\langle \mathtt{k}, \mathtt{n}, \mathtt{k}', [\mathtt{t}, \mathtt{t}] \rangle$ if $\mathtt{t} \in \mathtt{V}$.

Completeness Theorem I: If $\mathtt{c}$ is a concrete certificate and $\models_c \left(\wedge_{\mathtt{c}' \in C \cup C_R} \; \mathtt{c}' \right) \Rightarrow \varphi_{\mathtt{c}}$, then $\mathtt{Tuples}(C, C_R) \longrightarrow^* \tau_{\mathtt{c}'}$ for some $\mathtt{c}'$ that subsumes $\mathtt{c}$.

Conclusion: R1–R3 suffice for concrete certificates.

Getting Full Completeness

There are two major impediments to getting full completeness.

Impediment 1: Want conclusions about names other than keys. R2 and R3 do not suffice. Suppose

- $\mathtt{c}_1$ is $(\mathtt{cert\ k_1}, \mathtt{n}, \mathtt{k_2}\text{'s } \mathtt{m}\text{'s } \mathtt{p}, [\mathtt{t}, \mathtt{t}])$,
- $\mathtt{c}_2$ is $(\mathtt{cert\ k_2}, \mathtt{m}, \mathtt{q}, [\mathtt{t}, \mathtt{t}])$, and
- $\mathtt{c}_3$ is $(\mathtt{cert\ k_1}, \mathtt{n}, \mathtt{q}\text{'s } \mathtt{p}, [\mathtt{t}, \mathtt{t}])$.

Clearly $\models_c \mathtt{c}_1 \wedge \mathtt{c}_2 \Rightarrow \varphi_{\mathtt{c}_3}$. But tuple reduction can't get this.

Problem: R2 applies only if third component is a key.

$$\text{R2. } \langle \mathtt{k}_1, \mathtt{n}, \mathtt{k}_2\text{'s } \mathtt{m}\text{'s } \mathtt{p}, \mathtt{V}_1 \rangle + \langle \mathtt{k}_2, \mathtt{m}, \mathtt{k}_3, \mathtt{V}_2 \rangle \longrightarrow \langle \mathtt{k}_1, \mathtt{n}, \mathtt{k}_3\text{'s } \mathtt{p}, \mathtt{V}_1 \cap \mathtt{V}_2 \rangle$$

Generalize R2 to R2′:

$$\text{R2}'. \langle \mathtt{k}_1, \mathtt{n}, \mathtt{k}_2\text{'s } \mathtt{m}\text{'s } \mathtt{p}, \mathtt{V}_1 \rangle + \langle \mathtt{k}_2, \mathtt{m}, \mathtt{q}, \mathtt{V}_2 \rangle \longrightarrow \langle \mathtt{k}_1, \mathtt{n}, \mathtt{q}\text{'s } \mathtt{p}, \mathtt{V}_1 \cap \mathtt{V}_2 \rangle.$$

Similarly generalize R3 to R3′.

Impediment 2: Want conclusions about arbitrary time intervals. Add following rule:

R4(a). $\langle \mathtt{k}, \mathtt{n}, \mathtt{p}, \mathtt{V}_1 \rangle + \langle \mathtt{k}, \mathtt{n}, \mathtt{p}, \mathtt{V}_2 \rangle \longrightarrow \langle \mathtt{k}, \mathtt{n}, \mathtt{p}, \mathtt{V}_3 \rangle$
if $\mathtt{V}_1 \cup \mathtt{V}_2 \supseteq \mathtt{V}_3$.

R4(b). $\langle \mathtt{k}, \mathtt{p}, \mathtt{D}_1, \mathtt{A}_1, \mathtt{V}_1 \rangle + \langle \mathtt{k}, \mathtt{p}, \mathtt{D}_2, \mathtt{A}_2, \mathtt{V}_2 \rangle \longrightarrow \langle \mathtt{k}, \mathtt{p}, \mathtt{D}_3, \mathtt{A}_3, \mathtt{V}_3 \rangle$
if $\mathtt{D}_3 \Rightarrow \mathtt{D}_1 \wedge \mathtt{D}_2$ is a tautology, $\mathtt{V}_1 \cup \mathtt{V}_2 \supseteq \mathtt{V}_3$, and $\mathtt{A}_1 \cup \mathtt{A}_2 \supseteq \mathtt{A}_3$.

Completeness Theorem II: If $|K| > |C| + |\mathtt{c}|$, and $\models_{c,K} \left(\wedge_{\mathtt{c}' \in C \cup C_R}\ \mathtt{c}'\right) \Rightarrow \varphi_{\mathtt{c}}$, then

$$\mathtt{Tuples}(C, C_R) \longrightarrow^*_{\{R1, R2', R3', R4\}} \tau_{\mathtt{c}}.$$

It seems reasonable to assume that in practice $|K| \gg |C| + |\mathtt{c}|$. Some restriction on $|K|$ is necessary.

Example: Suppose that $K = \{k\}$.

- $\mathtt{c}$ is $(\mathtt{cert}\ \ \mathtt{k}, \mathtt{n}, \mathtt{k}, \mathtt{V})$
- $\mathtt{c}'$ is $(\mathtt{cert}\ \ \mathtt{k}, \mathtt{n}, \mathtt{k}\text{'s}\ \mathtt{m}, \mathtt{V})$.

$\models_{c,K} \mathtt{c} \Rightarrow \varphi_{\mathtt{c}'}$ (since $\models_{c,K} \mathtt{n} \longmapsto \mathtt{k} \Rightarrow \mathtt{k}\text{'s}\, \mathtt{n} \longmapsto \mathtt{k}\text{'s}\, \mathtt{m}$).

- There are no rules that let us derive this.
- Cardinality of K also an issue in completeness theorems for LLNC.

Conclusions

We have provided a semantic basis for SPKI.

- The logic shows the sense in which the tuple reduction rules are complete.
- New reduction rules are needed full completeness
- Translating the English description to the logic forces us to clarify some ambiguities.
- No need for nonmononotonicity to handle revocation.
- Focus here is on reduction rules, but the logic should be useful for general reasoning about names and authorization.
  - Can translate queries about names and actions to the logic, and use Logic Programming technology to answer them (cf. [HM99/01])
    - which principals are authorized to perform a certain action,
    - which actions is a principal allowed to perform,
    - which names have a particular principal bound to them.