Safely composing security protocols via tagging

Stéphanie Delaune

LSV, ENS Cachan & CNRS & INRIA project SECSI

February, 25, 2008

⟶ joint work with Véronique Cortier, Jérémie Delaitre, Myrto Arapinis and Steve Kremer

Context: cryptographic protocols

Cryptographic protocols

- small programs designed to secure communication (*e.g.* secrecy)
- use cryptographic primitives (*e.g.* encryption, signature, ...)

The network is unsecure!

Communications take place over a public network like the Internet.

Context: cryptographic protocols

Cryptographic protocols

- small programs designed to secure communication (*e.g.* secrecy)
- use cryptographic primitives (*e.g.* encryption, signature, ...)

The network is unsecure!

Communications take place over a public network like the Internet.

Cryptographic protocols (formal approach)

Messages are abstracted by terms

- pairing $\langle m_1, m_2 \rangle$,
- symmetric $\mathsf{enc}(m, k)$ and public-key encryption $\mathsf{enca}(m, \mathsf{pub}(A))$,
- signature $\mathsf{sign}(m, \mathsf{priv}(A))$.

Cryptographic protocols (formal approach)

Messages are abstracted by terms

- pairing $\langle m_1, m_2 \rangle$,
- symmetric $\mathrm{enc}(m, k)$ and public-key encryption $\mathrm{enca}(m, \mathrm{pub}(A))$,
- signature $\mathrm{sign}(m, \mathrm{priv}(A))$.

Presence of an idealized attacker

- may read, intercept and send messages,
- may build new messages following deduction rules (symbolic manipulation on terms).

Cryptographic protocols (formal approach)

Messages are abstracted by terms

- pairing $\langle m_1, m_2 \rangle$,
- symmetric $\mathsf{enc}(m, k)$ and public-key encryption $\mathsf{enca}(m, \mathsf{pub}(A))$,
- signature $\mathsf{sign}(m, \mathsf{priv}(A))$.

Presence of an idealized attacker

- may read, intercept and send messages,
- may build new messages following deduction rules (symbolic manipulation on terms).

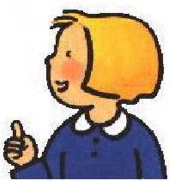
Examples:

$$\frac{m \quad k}{\mathsf{enc}(m,k)} \qquad \frac{\mathsf{enc}(m,k) \quad k}{m} \qquad \frac{\mathsf{enca}(m,\mathsf{pub}(a)) \quad \mathsf{priv}(a)}{m}$$

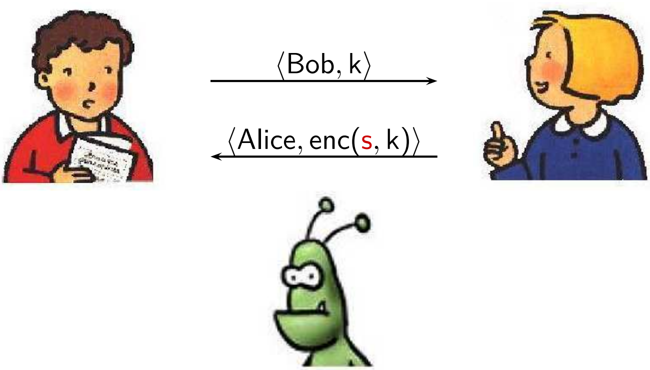
A simple protocol

$$\xrightarrow{\langle \mathsf{Bob}, \mathsf{k} \rangle}$$

$$\xleftarrow{\langle \mathsf{Alice}, \mathsf{enc}(\color{red}{\mathsf{s}}, \mathsf{k}) \rangle}$$



A simple protocol



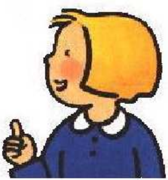
Question?

Can the attacker learn the secret s?

A simple protocol

$$\xrightarrow{\langle \mathsf{Bob}, \mathsf{k} \rangle}$$

$$\xleftarrow{\langle \mathsf{Alice}, \mathsf{enc}(\mathsf{s}, \mathsf{k}) \rangle}$$



Answer: Of course, Yes!

$$\frac{\dfrac{\langle \mathsf{Alice}, \mathsf{enc}(\mathsf{s}, \mathsf{k}) \rangle}{\mathsf{enc}(\mathsf{s}, \mathsf{k})} \qquad \dfrac{\langle \mathsf{Bob}, \mathsf{k} \rangle}{\mathsf{k}}}{\mathsf{s}}$$

Composition problem (part 2 of this talk)

$\longrightarrow$ sessions coming from the same protocol

$$\begin{array}{ll} A \rightarrow B: & \mathrm{enca}(\langle A, K, Na\rangle, \mathrm{pub}(B)),\ \mathrm{sign}(\mathrm{enca}(\langle A, Na\rangle, \mathrm{pub}(B)), \mathrm{priv}(A)) \\ B \rightarrow A: & Na, \mathrm{enc}(s, K) \end{array}$$

Question?

What about the secrecy of s?

Composition problem (part 2 of this talk)

$\longrightarrow$ sessions coming from the same protocol

$$
\begin{array}{ll}
A \rightarrow B: & \mathrm{enca}(\langle A, K, Na\rangle, \mathrm{pub}(B)),\ \mathrm{sign}(\mathrm{enca}(\langle A, Na\rangle, \mathrm{pub}(B)), \mathrm{priv}(A)) \\
B \rightarrow A: & Na, \mathrm{enc}(s, K)
\end{array}
$$

Question?

What about the secrecy of s?

Composition problem (part 2 of this talk)

$\longrightarrow$ sessions coming from the same protocol

$$\begin{array}{rl} A \rightarrow B: & \mathsf{enca}(\langle A, K, Na\rangle, \mathsf{pub}(B)),\ \mathsf{sign}(\mathsf{enca}(\langle A, Na\rangle, \mathsf{pub}(B)), \mathsf{priv}(A)) \\ B \rightarrow A: & Na, \mathsf{enc}(s, K) \end{array}$$

Attack with 2 sessions:

$$\begin{array}{rl} A \rightarrow B: & \mathsf{enca}(\langle A, K, Na\rangle, \mathsf{pub}(B)),\ \mathsf{sign}(\mathsf{enca}(\langle A, Na\rangle, \mathsf{pub}(B)), \mathsf{priv}(A)) \\ B \rightarrow A: & Na, \mathsf{enc}(s_1, K) \\ I(A) \rightarrow B: & \mathsf{enca}(\langle A, Ki, Na\rangle, \mathsf{pub}(B)),\ \mathsf{sign}(\mathsf{enca}(\langle A, Na\rangle, \mathsf{pub}(B)), \mathsf{priv}(A)) \\ B \rightarrow A: & Na, \mathsf{enc}(s_2, Ki) \end{array}$$

Question?

What about the secrecy of s?

Composition problem (part 1 of this talk)

Protocol 1

$$P_1 : \quad A \to B : \quad \mathrm{enca}(s, \mathrm{pub}(B))$$

Question?

What about the secrecy of s?

Composition problem (part 1 of this talk)

$\longrightarrow$ sessions coming from different protocols

Protocol 1

$$P_1: \quad A \rightarrow B: \quad \mathrm{enca}(s, \mathrm{pub}(B))$$

Protocol 2

$$\begin{array}{lll} P_2: & A \rightarrow B: & \mathrm{enca}(N_a, \mathrm{pub}(B)) \\ & B \rightarrow A: & N_a \end{array}$$

Question?

What about the secrecy of s?

Motivations

Verification of security protocols

- Existing tools allow us to verify relatively small protocols and sometimes only for a bounded number of sessions
- Most often, we verify them in isolation
  $\longrightarrow$ this is not sufficient

Motivations

Verification of security protocols

- Existing tools allow us to verify relatively small protocols and sometimes only for a bounded number of sessions
- Most often, we verify them in isolation
  $\longrightarrow$ this is not sufficient

Our Goals

1. propose a general and simple transformation that maps a protocol that is secure for one session into a protocol that is secure for an unbounded number of sessions;
2. investigate sufficient and rather tight conditions for a protocol to be safely used in an environment where other protocols may be executed as well;

$\longrightarrow$ protocols may share identities and keys (*e.g.* public keys, long-term symmetric keys)

Outline of the talk

1. Introduction
2. Preliminaries
3. Composition result (1st part)
4. Composition result (2nd part): ongoing work
5. Conclusion

Outline of the talk

1. Introduction
2. Preliminaries
3. Composition result (1^{st} part)
4. Composition result (2^{nd} part): ongoing work
5. Conclusion

Deduction capabilities of the attacker

Composition rules

$$\frac{T \vdash u \quad T \vdash v}{T \vdash \langle u, v \rangle} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathsf{f}(u, v)} \ \textit{with}\ \mathsf{f} \in \{\mathsf{enc}, \mathsf{enca}, \mathsf{sign}\}$$

Decomposition rules

$$\frac{}{T \vdash u}\ u \in T \qquad \frac{T \vdash \langle u, v \rangle}{T \vdash u} \qquad \frac{T \vdash \langle u, v \rangle}{T \vdash v} \qquad \frac{T \vdash \mathsf{enc}(u, v) \quad T \vdash v}{T \vdash u}$$

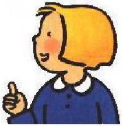
$$\frac{T \vdash \mathsf{enca}(u, \mathsf{pub}(v)) \quad T \vdash \mathsf{priv}(v)}{T \vdash u} \qquad \frac{T \vdash \mathsf{sign}(u, \mathsf{priv}(v))}{T \vdash u}\ \textit{(optional)}$$

Deducibility relation

A term u is deducible from a set of terms T, denoted by $T \vdash u$, if there exists a prooftree witnessing this fact.

Protocol – Example: Needham Schroeder protocol (1978)

Needham Schroeder protocol:



$$\begin{array}{rcrr} A & \rightarrow & B: & \{N_a, A\}_{\mathrm{pub}(B)} \\ B & \rightarrow & A: & \{N_a, N_b\}_{\mathrm{pub}(A)} \\ A & \rightarrow & B: & \{N_b\}_{\mathrm{pub}(B)} \end{array}$$

A protocol is a finite set of roles:

Exemple:

role $\Pi(1)$ corresponding to the 1st participant played by a talking to b:

$$\mathrm{init} \xrightarrow{N} \mathrm{enca}(\langle N, a\rangle, \mathrm{pub}(b))$$
$$\mathrm{enca}(\langle N, x\rangle, \mathrm{pub}(a)) \rightarrow \mathrm{enca}(x, \mathrm{pub}(b)).$$

Secrecy via constraint solving

Constraint systems are used to specify secrecy preservation under a particular, finite scenario.

Scenario

$$
\begin{array}{ccc}
\mathrm{rcv}(u_1) & \xrightarrow{N_1} & \mathrm{snd}(v_1) \\
\mathrm{rcv}(u_2) & \xrightarrow{N_2} & \mathrm{snd}(v_2) \\
 & \dots & \\
\mathrm{rcv}(u_n) & \xrightarrow{N_n} & \mathrm{snd}(v_n)
\end{array}
$$

Constraint System

$$
\mathcal{C} = \begin{cases} T_0 \Vdash u_1 \\ T_0, v_1 \Vdash u_2 \\ \quad \dots \\ T_0, v_1, .., v_n \Vdash s \end{cases}
$$

Secrecy via constraint solving

Constraint systems are used to specify secrecy preservation under a particular, finite scenario.

Scenario

$$\begin{array}{ccc} \mathsf{rcv}(u_1) & \xrightarrow{N_1} & \mathsf{snd}(v_1) \\ \mathsf{rcv}(u_2) & \xrightarrow{N_2} & \mathsf{snd}(v_2) \\ & \ldots & \\ \mathsf{rcv}(u_n) & \xrightarrow{N_n} & \mathsf{snd}(v_n) \end{array}$$

Constraint System

$$\mathcal{C} = \begin{cases} T_0 \Vdash u_1 \\ T_0, v_1 \Vdash u_2 \\ \quad \ldots \\ T_0, v_1, .., v_n \Vdash s \end{cases}$$

Solution of a constraint system

A substitution σ such that

for every $T \Vdash u \in \mathcal{C}$, $u\sigma$ *is deducible from* $T\sigma$.

Secrecy via constraint solving

Constraint systems are used to specify secrecy preservation under a particular, finite scenario.

Scenario

$$
\begin{array}{c}
\mathsf{rcv}(u_1) \xrightarrow{N_1} \mathsf{snd}(v_1) \\
\mathsf{rcv}(u_2) \xrightarrow{N_2} \mathsf{snd}(v_2) \\
\dots \\
\mathsf{rcv}(u_n) \xrightarrow{N_n} \mathsf{snd}(v_n)
\end{array}
$$

Constraint System

$$
\mathcal{C} = \left\{ \begin{array}{l}
T_0 \Vdash u_1 \\
T_0, v_1 \Vdash u_2 \\
\quad \dots \\
T_0, v_1, .., v_n \Vdash s
\end{array} \right.
$$

Well-formed constraint system

- monotonicity: intruder never forgets information
- origination: a variable first appear in a right hand side.

$\longrightarrow$ to discard some weird protocols, we also require plaintext origination

Procedure due to H. Comon-Lundh

Input: A (well-formed) constraint system
Output: Either $\perp$ or a constraint system in solved form
$\longrightarrow$ systems in solved form always have a solution

R_5 : $\mathcal{C} \wedge T \Vdash f(u,v) \rightsquigarrow \mathcal{C} \wedge T \Vdash u \wedge T \Vdash v$
for $f \in \{\langle\rangle, \mathrm{enc}, \mathrm{enca}, \mathrm{sign}\}$

R_4 : $\mathcal{C} \wedge T \Vdash u \rightsquigarrow \perp$ if $vars(T,u) = \emptyset$ and $T \nvdash u$

R_1 : $\mathcal{C} \wedge T \Vdash u \rightsquigarrow \mathcal{C}$ if $T \cup \{x \mid T' \Vdash x \in \mathcal{C}, T' \subsetneq T\} \vdash u$

R_2 : $\mathcal{C} \wedge T \Vdash u \rightsquigarrow_\sigma \mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma$ $u' \in st(T)$

R_3 : $\mathcal{C} \wedge T \Vdash v \rightsquigarrow_\sigma \mathcal{C}\sigma \wedge T\sigma \Vdash v\sigma$ $u, u' \in st(T)$
if $\sigma = \mathrm{mgu}(u,u')$, $u, u' \notin \mathcal{X}$, $u \neq u'$

These simplification rules give us an algorithm to decide satisfiability of a well-formed constraint system.

Procedure due to H. Comon-Lundh

Input: A (well-formed) constraint system
Output: Either $\perp$ or a constraint system in solved form
$\longrightarrow$ systems in solved form always have a solution

$$\begin{array}{llcl}
R_5: & \mathcal{C} \wedge T \Vdash f(u,v) & \rightsquigarrow & \mathcal{C} \wedge T \Vdash u \wedge T \Vdash v \\
 & & & \text{for } f \in \{\langle\rangle, \mathrm{enc}, \mathrm{enca}, \mathrm{sign}\} \\
R_4: & \mathcal{C} \wedge T \Vdash u & \rightsquigarrow & \perp \quad \text{if } vars(T,u) = \emptyset \text{ and } T \not\vdash u \\
R_1: & \mathcal{C} \wedge T \Vdash u & \rightsquigarrow & \mathcal{C} \quad \text{if } T \cup \{x \mid T' \Vdash x \in \mathcal{C}, T' \subsetneq T\} \vdash u \\
R_2: & \mathcal{C} \wedge T \Vdash u & \rightsquigarrow_\sigma & \mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma \qquad u' \in st(T) \\
R_3: & \mathcal{C} \wedge T \Vdash v & \rightsquigarrow_\sigma & \mathcal{C}\sigma \wedge T\sigma \Vdash v\sigma \qquad u, u' \in st(T) \\
 & & & \text{if } \sigma = \mathrm{mgu}(u,u'),\ u, u' \notin \mathcal{X},\ u \neq u'
\end{array}$$

These simplification rules give us an algorithm to decide satisfiability of a well-formed constraint system.

Procedure due to H. Comon-Lundh

Input: A (well-formed) constraint system
Output: Either $\perp$ or a constraint system in solved form
$\longrightarrow$ systems in solved form always have a solution

$$
\begin{array}{llcll}
R_5: & \mathcal{C} \wedge T \Vdash f(u,v) & \rightsquigarrow & \mathcal{C} \wedge T \Vdash u \wedge T \Vdash v & \\
 & & & & \text{for } f \in \{\langle\rangle, \mathrm{enc}, \mathrm{enca}, \mathrm{sign}\} \\
R_4: & \mathcal{C} \wedge T \Vdash u & \rightsquigarrow & \perp & \text{if } vars(T,u) = \emptyset \text{ and } T \not\vdash u \\
R_1: & \mathcal{C} \wedge T \Vdash u & \rightsquigarrow & \mathcal{C} & \text{if } T \cup \{x \mid T' \Vdash x \in \mathcal{C}, T' \subsetneq T\} \vdash u \\
R_2: & \mathcal{C} \wedge T \Vdash u & \rightsquigarrow_\sigma & \mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma & u' \in st(T) \\
R_3: & \mathcal{C} \wedge T \Vdash v & \rightsquigarrow_\sigma & \mathcal{C}\sigma \wedge T\sigma \Vdash v\sigma & u, u' \in st(T) \\
 & & & & \text{if } \sigma = \mathrm{mgu}(u,u'),\ u,u' \notin \mathcal{X},\ u \neq u'
\end{array}
$$

These simplification rules give us an algorithm to decide satisfiability of a well-formed constraint system.

Refinement of the procedure

$$\mathsf{R}'_2: \quad \mathcal{C} \wedge T \Vdash u \quad \rightsquigarrow_\sigma \quad \mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma \qquad\qquad u' \in st(T)$$

$$\mathsf{R}'_3: \quad \mathcal{C} \wedge T \Vdash v \quad \rightsquigarrow_\sigma \quad \mathcal{C}\sigma \wedge T\sigma \Vdash v\sigma \qquad\qquad u, u' \in st(T)$$

$$\text{if } \sigma = \mathrm{mgu}(u, u'),\ u, u' \notin \mathcal{X},\ u \neq u'$$

<u>u, u' are not pairs</u>

Proposition - Cortier et al.,FSTTCS'07

These simplification rules, i.e. R_1, R_4, R_5, R'_2 and R'_3, still forms a complete decision procedure.

This result is of independent interest:

- we provide a more efficient procedure for solving constraint systems
  $\longrightarrow$ of course, the theoretical complexity remains the same, i.e. NP

Refinement of the procedure

$$\begin{array}{llllr} R'_2: & \mathcal{C} \wedge T \Vdash u & \rightsquigarrow_\sigma & \mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma & u' \in st(T) \\ R'_3: & \mathcal{C} \wedge T \Vdash v & \rightsquigarrow_\sigma & \mathcal{C}\sigma \wedge T\sigma \Vdash v\sigma & u, u' \in st(T) \end{array}$$

if $\sigma = \mathrm{mgu}(u, u')$, $u, u' \notin \mathcal{X}$, $u \neq u'$

u, u' are not pairs

Proposition - Cortier et al.,FSTTCS'07

These simplification rules, i.e. R_1, R_4, R_5, R'_2 and R'_3, still forms a complete decision procedure.

This result is of independent interest:

- we provide a more efficient procedure for solving constraint systems
  $\longrightarrow$ of course, the theoretical complexity remains the same, i.e. NP

Refinement of the procedure

$$\mathsf{R}_2' : \quad \mathcal{C} \wedge T \Vdash u \quad \rightsquigarrow_\sigma \quad \mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma \qquad\qquad u' \in st(T)$$

$$\mathsf{R}_3' : \quad \mathcal{C} \wedge T \Vdash v \quad \rightsquigarrow_\sigma \quad \mathcal{C}\sigma \wedge T\sigma \Vdash v\sigma \qquad\qquad u, u' \in st(T)$$

$$\text{if } \sigma = \mathrm{mgu}(u, u'),\ u, u' \notin \mathcal{X},\ u \neq u'$$

$\underline{u, u' \text{ are not pairs}}$

Proposition - Cortier et al.,FSTTCS'07

These simplification rules, i.e. R_1, R_4, R_5, R_2' and R_3', still forms a complete decision procedure.

This result is of independent interest:

- we provide a more efficient procedure for solving constraint systems
  $\longrightarrow$ of course, the theoretical complexity remains the same, i.e. NP

Outline of the talk

1. Introduction
2. Preliminaries
3. Composition result (1^{st} part)
4. Composition result (2^{nd} part): ongoing work
5. Conclusion

Condition 1 - Tagging

Condition 1 (well-tagged protocol)

Each protocol is given an identifier (*e.g.* the protocol's name). This identifier has to appear in any encrypted and signed message.

$\longrightarrow$ this tagging policy will avoid interaction between two differents protocols.

Example: P_1 is 1-tagged whereas P_2 is 2-tagged

Protocol P_1

$$A \rightarrow B : \mathrm{enca}(\langle 1, s \rangle, \mathrm{pub}(B))$$

Protocol P_2

$$A \rightarrow B : \mathrm{enca}(\langle 2, N_a \rangle, \mathrm{pub}(B))$$
$$B \rightarrow A : N_a$$

Condition 1 - Tagging

Condition 1 (well-tagged protocol)

Each protocol is given an identifier (*e.g.* the protocol's name). This identifier has to appear in any encrypted and signed message.

$\longrightarrow$ this tagging policy will avoid interaction between two differents protocols.

Example: P_1 is 1-tagged whereas P_2 is 2-tagged

Protocol P_1

$$A \rightarrow B : \mathsf{enca}(\langle 1, s\rangle, \mathsf{pub}(B))$$

Protocol P_2

$$A \rightarrow B : \mathsf{enca}(\langle 2, N_a\rangle, \mathsf{pub}(B))$$
$$B \rightarrow A : N_a$$

Condition 2 - No critical key in plaintext

Protocol P_1

$$A \rightarrow B : \mathsf{enca}(\langle 1, s\rangle, \mathsf{pub}(B))$$

Protocol P_2

$$B \rightarrow A : \mathsf{priv}(B)$$

Condition 2 (no critical key in plaintext)

Let KC be the set of *critical keys*, *i.e.* constants and long-term keys used in P_1 or P_2 and not publicly known.

$$\mathsf{KC} \cap (\mathit{plaintext}(P_1) \cup \mathit{plaintext}(P_2)) = \emptyset.$$

Example: We have that $\mathsf{KC} = \{\mathsf{priv}(B)\}$.

$\longrightarrow$ Condition 2 (no critical key in plaintext) is not satisfied by P_2.

Condition 2 - No critical key in plaintext

Protocol P_1

$$A \rightarrow B : \mathsf{enca}(\langle 1, s \rangle, \mathsf{pub}(B))$$

Protocol P_2

$$B \rightarrow A : \mathsf{priv}(B)$$

Condition 2 (no critical key in plaintext)

Let KC be the set of *critical keys*, *i.e.* constants and long-term keys used in P_1 or P_2 and not publicly known.

$$\mathsf{KC} \cap (\mathit{plaintext}(P_1) \cup \mathit{plaintext}(P_2)) = \emptyset.$$

Example: We have that $\mathsf{KC} = \{\mathsf{priv}(B)\}$.

$\longrightarrow$ Condition 2 (no critical key in plaintext) is not satisfied by P_2.

Condition 2 - No critical key in plaintext

Protocol P_1

$A \rightarrow B : \mathsf{enca}(\langle 1, s\rangle, \mathsf{pub}(B))$

Protocol P_2

$B \rightarrow A : \mathsf{priv}(B)$

Condition 2 (no critical key in plaintext)

Let KC be the set of *critical keys*, *i.e.* constants and long-term keys used in P_1 or P_2 and not publicly known.

$$\mathsf{KC} \cap (plaintext(P_1) \cup plaintext(P_2)) = \emptyset.$$

Example: We have that $\mathsf{KC} = \{\mathsf{priv}(B)\}$.

$\longrightarrow$ Condition 2 (no critical key in plaintext) is not satisfied by P_2.

Condition 2 - No critical key in plaintext

Protocol P_1

$$A \rightarrow B : \mathsf{enca}(\langle 1, s\rangle, \mathsf{pub}(B))$$

Protocol P_2

$$B \rightarrow A : \mathsf{priv}(B)$$

Condition 2 (no critical key in plaintext)

Let KC be the set of *critical keys*, *i.e.* constants and long-term keys used in P_1 or P_2 and not publicly known.

$$\mathsf{KC} \cap (\mathit{plaintext}(P_1) \cup \mathit{plaintext}(P_2)) = \emptyset.$$

Example: We have that $\mathsf{KC} = \{\mathsf{priv}(B)\}$.

$\longrightarrow$ Condition 2 (no critical key in plaintext) is not satisfied by P_2.

Main result - Composition theorem

Let P_1 and P_2 be two well-tagged protocols such that

1. P_1 is α-tagged and P_2 is β-tagged with $\alpha \neq \beta$,
2. critical keys do not appear in plaintext position, *i.e.*

$$\mathsf{KC} \cap (plaintext(P_1) \cup plaintext(P_2)) = \emptyset$$

where $\mathsf{KC} = (\mathsf{ExtNames}(P_1) \cup \mathsf{ExtNames}(P_2)) \smallsetminus T_0$

Let s be a α-tagged term such that $vars(s) \subseteq vars(P_1)$.

Then P_1 preserves the secrecy of s for the initial knowledge T_0 if and only if $P_1 \mid P_2$ preserves the secrecy of s for T_0.

Main result - Composition theorem

Let P_1 and P_2 be two well-tagged protocols such that

1. P_1 is α-tagged and P_2 is β-tagged with $\alpha \neq \beta$,
2. critical keys do not appear in plaintext position, *i.e.*

$$\mathsf{KC} \cap (plaintext(P_1) \cup plaintext(P_2)) = \emptyset$$

where $\mathsf{KC} = (\mathsf{ExtNames}(P_1) \cup \mathsf{ExtNames}(P_2)) \smallsetminus T_0$

Let s be a α-tagged term such that $vars(s) \subseteq vars(P_1)$.

Then P_1 preserves the secrecy of s for the initial knowledge T_0 if and only if $P_1 \mid P_2$ preserves the secrecy of s for T_0.

Main result - Composition theorem

Let P_1 and P_2 be two well-tagged protocols such that

1. P_1 is α-tagged and P_2 is β-tagged with $\alpha \neq \beta$,
2. critical keys do not appear in plaintext position, *i.e.*

$$\mathsf{KC} \cap (plaintext(P_1) \cup plaintext(P_2)) = \emptyset$$

where $\mathsf{KC} = (\mathsf{ExtNames}(P_1) \cup \mathsf{ExtNames}(P_2)) \smallsetminus T_0$

Let s be a α-tagged term such that $vars(s) \subseteq vars(P_1)$.

Then P_1 preserves the secrecy of s for the initial knowledge T_0 if and only if $P_1 \mid P_2$ preserves the secrecy of s for T_0.

Main steps of the proof

Proposition

Let sc be a scenario of $\Pi_1 \mid \Pi_2$, T_0 the intruder's knowledge, s the secret.
Let
- $\mathcal{C}$ be the constraint system associated to sc, T_0 and s,
- $\mathcal{C}'$ be the constraint system associated to $\mathrm{sc}|_{\Pi_1}$, T_0 and s.

We have that $\mathcal{C}$ satisfiable implies $\mathcal{C}'$ satisfiable

1. If $\mathcal{C}$ satisfiable, there exists a solution θ without any mixing, i.e. terms in $\mathcal{C}\theta$ will be either α-tagged or β-tagged.
   $\longrightarrow$ refinement of the constraint solving procedure due to H. Comon-Lundh
2. Removing β-tagged terms from a left hand side of a constraint is safe

$$T_0, T_\alpha\theta, T_\beta\theta \vdash u_\alpha\theta \;\Rightarrow\; T_0, T_\alpha\theta \vdash u_\alpha\theta$$

$\longrightarrow$ proved by induction on the prooftree witnessing $T_0, T_\alpha\theta, T_\beta\theta \vdash u_\alpha\theta$

Main steps of the proof

Proposition

Let sc be a scenario of $\Pi_1 \mid \Pi_2$, T_0 the intruder's knowledge, s the secret.
Let

- $\mathcal{C}$ be the constraint system associated to sc, T_0 and s,
- $\mathcal{C}'$ be the constraint system associated to $\mathrm{sc}|_{\Pi_1}$, T_0 and s.

We have that $\mathcal{C}$ satisfiable implies $\mathcal{C}'$ satisfiable

1. If $\mathcal{C}$ satisfiable, there exists a solution θ without any mixing, i.e. terms in $\mathcal{C}\theta$ will be either α-tagged or β-tagged.
   $\longrightarrow$ refinement of the constraint solving procedure due to H. Comon-Lundh
2. Removing β-tagged terms from a left hand side of a constraint is safe

$$T_0, T_\alpha\theta, T_\beta\theta \vdash u_\alpha\theta \;\Rightarrow\; T_0, T_\alpha\theta \vdash u_\alpha\theta$$

$\longrightarrow$ proved by induction on the prooftree witnessing $T_0, T_\alpha\theta, T_\beta\theta \vdash u_\alpha\theta$

Main steps of the proof

Proposition

Let sc be a scenario of $\Pi_1 \mid \Pi_2$, T_0 the intruder's knowledge, s the secret.
Let
- $\mathcal{C}$ be the constraint system associated to sc, T_0 and s,
- $\mathcal{C}'$ be the constraint system associated to $\mathsf{sc}|_{\Pi_1}$, T_0 and s.

We have that $\mathcal{C}$ satisfiable implies $\mathcal{C}'$ satisfiable

1. If $\mathcal{C}$ satisfiable, there exists a solution θ without any mixing, i.e. terms in $\mathcal{C}\theta$ will be either α-tagged or β-tagged.
   $\longrightarrow$ refinement of the constraint solving procedure due to H. Comon-Lundh
2. Removing β-tagged terms from a left hand side of a constraint is safe

$$T_0, T_\alpha\theta, T_\beta\theta \vdash u_\alpha\theta \;\Rightarrow\; T_0, T_\alpha\theta \vdash u_\alpha\theta$$

$\longrightarrow$ proved by induction on the prooftree witnessing $T_0, T_\alpha\theta, T_\beta\theta \vdash u_\alpha\theta$

A little bit further ...

In the journal version of the paper (currently under submission)

- we add a new primitive: hash function h(m),
- we relax the condition "well-tagged" to non-unifiability,
- we deal with a class of security properties
  $\longrightarrow$ we introduce a logic for which the composition result holds

A little bit further ...

In the journal version of the paper (currently under submission)

- we add a new primitive: hash function h(m),
- we relax the condition "well-tagged" to non-unifiability,
- we deal with a class of security properties
  $\longrightarrow$ we introduce a logic for which the composition result holds

A little bit further ...

In the journal version of the paper (currently under submission)

- we add a new primitive: hash function h(m),
- we relax the condition "well-tagged" to non-unifiability,
- we deal with a class of security properties
  $\longrightarrow$ we introduce a logic for which the composition result holds

$$\begin{array}{rcl}\psi & := & \mathsf{true} \mid P(t_1, \ldots, t_n) \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2 \mid \mathrm{Y}\psi \mid \psi_1 \,\mathrm{S}\, \psi_2 \\ & & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mid \exists x.\psi \mid \forall x.\psi \\ \phi & := & \psi \mid \mathsf{learn}(m) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \exists x.\phi \mid \forall x.\phi\end{array}$$

This logic allows us to express:

- secrecy of a nonce: $\forall x. (\Box\, \mathsf{nonce}(x)) \Rightarrow \neg \mathsf{learn}(x)$
- several notions of authentication, e.g. aliveness:
  $\mathsf{end}(a, b) \Rightarrow \Box\, \mathsf{start}(b)) \wedge (\mathsf{end}(b, a) \Rightarrow \Box\, \mathsf{start}(a))$

Related Works

The idea of adding an identifier is not novel:

- Principle 10 in the prudent engineering paper,
  [Abadi & Needham, 1995]
- ...

There are also some formal results about this composition problem:

- *Protocol independence through disjoint encryption* [Guttman & Thayer,00]
  $\longrightarrow$ asymmetric condition allowing one to deal with protocols with ticket (e.g. Neuman-Strubblebine protocol)
  $\longrightarrow$ their condition has to hold on any valid execution of the protocol
- *Sufficient conditions for composing security protocols* [Andova *et al.*,07]
  $\longrightarrow$ different kinds of composition (parallel, sequential)
  $\longrightarrow$ they have to assume typing hypothesis, they can not deal with protocols with ciphertext forwarding

Related Works

The idea of adding an identifier is not novel:

- Principle 10 in the prudent engineering paper, [Abadi & Needham, 1995]
- ...

There are also some formal results about this composition problem:

- *Protocol independence through disjoint encryption* [Guttman & Thayer,00]
  $\longrightarrow$ asymmetric condition allowing one to deal with protocols with ticket (e.g. Neuman-Strubblebine protocol)
  $\longrightarrow$ their condition has to hold on any valid execution of the protocol

- *Sufficient conditions for composing security protocols* [Andova *et al.*,07]
  $\longrightarrow$ different kinds of composition (parallel, sequential)
  $\longrightarrow$ they have to assume typing hypothesis, they can not deal with protocols with ciphertext forwarding

Outline of the talk

1. Introduction
2. Preliminaries
3. Composition result (1^{st} part)
4. Composition result (2^{nd} part): ongoing work
5. Conclusion

Summary

Our Goal

We propose a transformation which maps a protocol P that is secure for a single session to a protocol $\overline{P}$ that is secure for an unbounded number of sessions.

$\longrightarrow$ side-effect: we also caracterise a class of protocols for which secrecy for an unbounded number of sessions is decidable

Main Difficulty

We can not assume that a (static) tag is already shared between the different participants of one session.

$\longrightarrow$ we will use dynamic tags

Summary

Our Goal

We propose a transformation which maps a protocol P that is secure for a single session to a protocol $\overline{P}$ that is secure for an unbounded number of sessions.

$\longrightarrow$ side-effect: we also caracterise a class of protocols for which secrecy for an unbounded number of sessions is decidable

Main Difficulty

We can not assume that a (static) tag is already shared between the different participants of one session.
$\longrightarrow$ we will use dynamic tags

Our transformation

Let P be a protocol with ℓ participants as given below:

$$\begin{array}{rcl} A_{i_1} \longrightarrow A_{j_1} & : & m_1 \\ A_{i_2} \longrightarrow A_{j_2} & : & m_2 \\ & \vdots & \\ A_{i_k} \longrightarrow A_{j_k} & : & m_k \end{array}$$

Our transformation

The protocol $\overline{P}$ (with ℓ participants) is decribed below:

Initialisation phase: broadcast of fresh nonces

$$
\begin{array}{rl}
A_1 \rightarrow All: & A_1, N_1 \\
A_2 \rightarrow All: & A_2, N_2 \\
\vdots & \\
A_\ell \rightarrow All: & A_\ell, N_\ell
\end{array}
$$

Our transformation

The protocol $\overline{P}$ (with ℓ participants) is decribed below:

Initialisation phase: broadcast of fresh nonces

$$\begin{array}{rl} A_1 \rightarrow All: & A_1, N_1 \\ A_2 \rightarrow All: & A_2, N_2 \\ & \vdots \\ A_\ell \rightarrow All: & A_\ell, N_\ell \end{array}$$

Every particicpant obtain a tag $= \langle A_1, N_1, A_2, N_2, \ldots, A_\ell, N_\ell \rangle$

Our transformation

The protocol $\overline{P}$ (with ℓ participants) is decribed below:

Initialisation phase: broadcast of fresh nonces

$$\begin{array}{rl} A_1 \rightarrow All: & A_1, N_1 \\ A_2 \rightarrow All: & A_2, N_2 \\ \vdots & \\ A_\ell \rightarrow All: & A_\ell, N_\ell \end{array}$$

Every particicpant obtain a tag $= \langle A_1, N_1, A_2, N_2, \ldots, A_\ell, N_\ell \rangle$

Main phase:

$$\begin{array}{rl} A_{i_1} \rightarrow A_{j_1}: & \overline{m_1} \\ A_{i_2} \rightarrow A_{j_2}: & \overline{m_2} \\ \vdots & \\ A_{i_k} \rightarrow A_{j_k}: & \overline{m_k} \end{array}$$

where the function $\overline{m}$ is defined by:

$$\begin{cases} \overline{\langle u_1, u_2 \rangle} & \rightarrow \quad \langle \overline{u_1}, \overline{u_2} \rangle \\ \overline{f(u_1, u_2)} & \rightarrow \quad f(\langle \mathsf{tag}, \overline{u_1} \rangle, \overline{u_2}) \\ & \qquad \text{when } f \in \{\mathsf{enc}, \mathsf{enca}, \mathsf{sign}\} \\ \overline{u} & \rightarrow \quad u \qquad \text{otherwise} \end{cases}$$

Example

Consider again the protocol $\overline{P}$ between A and B

$$\begin{array}{rl} A \rightarrow B: & \mathsf{enca}(\langle A, K, Na\rangle, \mathsf{pub}(B)), \\ & \mathsf{sign}(\mathsf{enca}(\langle A, Na\rangle, \mathsf{pub}(B)), \mathsf{priv}(A)) \\ B \rightarrow A: & Na, \mathsf{enc}(s, K) \end{array}$$

$\longrightarrow$ there is an attack involving 2 sessions between A and B.

The protocol $\overline{P}$ is as follows:

$$\begin{array}{rl} A \rightarrow B: & A, N_1 \\ B \rightarrow A: & B, N_2 \\ A \rightarrow B: & \mathsf{enca}(\langle \mathsf{tag}, \langle A, K, Na\rangle\rangle, \mathsf{pub}(B)), \\ & \mathsf{sign}(\langle \mathsf{tag}, \mathsf{enca}(\langle \mathsf{tag}, \langle A, Na\rangle\rangle, \mathsf{pub}(B))\rangle, \mathsf{priv}(A)) \\ B \rightarrow A: & Na, \mathsf{enc}(\langle \mathsf{tag}, s\rangle, K) \end{array}$$

where $\mathsf{tag} = \langle A, N_1, B, N_2\rangle$

Example

Consider again the protocol $\overline{P}$ between A and B

$$
\begin{array}{rl}
A \rightarrow B: & \mathrm{enca}(\langle A, K, Na\rangle, \mathrm{pub}(B)), \\
 & \mathrm{sign}(\mathrm{enca}(\langle A, Na\rangle, \mathrm{pub}(B)), \mathrm{priv}(A)) \\
B \rightarrow A: & Na, \mathrm{enc}(s, K)
\end{array}
$$

$\longrightarrow$ there is an attack involving 2 sessions between A and B.

The protocol $\overline{P}$ is as follows:

$$
\begin{array}{rl}
A \rightarrow B: & A, N_1 \\
B \rightarrow A: & B, N_2 \\
A \rightarrow B: & \mathrm{enca}(\langle \mathrm{tag}, \langle A, K, Na\rangle\rangle, \mathrm{pub}(B)), \\
 & \mathrm{sign}(\langle \mathrm{tag}, \mathrm{enca}(\langle \mathrm{tag}, \langle A, Na\rangle\rangle, \mathrm{pub}(B))\rangle, \mathrm{priv}(A)) \\
B \rightarrow A: & Na, \mathrm{enc}(\langle \mathrm{tag}, s\rangle, K)
\end{array}
$$

where $\mathrm{tag} = \langle A, N_1, B, N_2\rangle$

Composition result

Conjecture (almost established)

Under the same kind of hypothesis than the previous composition result (i.e. no critical key in plaintext, plaintext origination property), we have that

> *If P preserves the secrecy of s for a single honest session then $\overline{P}$ preserves the secrecy of s for an unbounded number of sessions.*

$\longrightarrow$ we prove this result by contradiction and we rely on the refinement of the procedure due to H. Comon-Lundh.

Remark: In each constraint system obtained after several simplification steps of the procedure, the terms are always uniquely tagged (even if there are not necessarily tagged as expected by a normal execution (i.e. no intervention of the attacker)

Composition result

Conjecture (almost established)

Under the same kind of hypothesis than the previous composition result (i.e. no critical key in plaintext, plaintext origination property), we have that

> *If P preserves the secrecy of s for a single honest session then $\overline{P}$ preserves the secrecy of s for an unbounded number of sessions.*

$\longrightarrow$ we prove this result by contradiction and we rely on the refinement of the procedure due to H. Comon-Lundh.

Remark: In each constraint system obtained after several simplification steps of the procedure, the terms are always uniquely tagged (even if there are not necessarily tagged as expected by a normal execution (i.e. no intervention of the attacker)

Composition result

Conjecture (almost established)

Under the same kind of hypothesis than the previous composition result (i.e. no critical key in plaintext, plaintext origination property), we have that

> *If P preserves the secrecy of s for a single honest session then $\overline{P}$ preserves the secrecy of s for an unbounded number of sessions.*

$\longrightarrow$ we prove this result by contradiction and we rely on the refinement of the procedure due to H. Comon-Lundh.

Remark: In each constraint system obtained after several simplification steps of the procedure, the terms are always uniquely tagged (even if there are not necessarily tagged as expected by a normal execution (i.e. no intervention of the attacker)

Related Works

Another compiler

- *Synthesizing secure protocols* [Cortier *et al.*,07]
  $\longrightarrow$ their notion of security for P is very weak (essentially with no adversary)
  $\longrightarrow$ their transformation is heavier than ours

Some other decidability classes for an unbounded number of sessions

- *On the security of ping-pong protocols* [Dolev *et al.*,83]
  $\longrightarrow$ PTIME decision procedure
  $\longrightarrow$ the class of protocols they consider is very restrictive
- *Towards a completeness result ... of security protocols* [Lowe,98]
- *Tagging makes secrecy decidable for unbounded nonces as well* [Rammanujam *et al.*,03]
  $\longrightarrow$ notion of secrecy that disallow temporary secrets
  $\longrightarrow$ no ciphertext forwarding (e.g. Yahalom)

Related Works

Another compiler

- *Synthesizing secure protocols* [Cortier *et al.*,07]
  $\longrightarrow$ their notion of security for P is very weak (essentially with no adversary)
  $\longrightarrow$ their transformation is heavier than ours

Some other decidability classes for an unbounded number of sessions

- *On the security of ping-pong protocols* [Dolev *et al.*,83]
  $\longrightarrow$ PTIME decision procedure
  $\longrightarrow$ the class of protocols they consider is very restrictive

- *Towards a completeness result ... of security protocols* [Lowe,98]
- *Tagging makes secrecy decidable for unbounded nonces as well* [Rammanujam *et al.*,03]
  $\longrightarrow$ notion of secrecy that disallow temporary secrets
  $\longrightarrow$ no ciphertext forwarding (e.g. Yahalom)

Outline of the talk

1. Introduction
2. Preliminaries
3. Composition result (1st part)
4. Composition result (2nd part): ongoing work
5. Conclusion

How to combine both results ?

$\longrightarrow$ by using tags of the form tag $= \langle id_\alpha, A_1, N_1, \ldots, A_\ell, N_\ell \rangle$.

Remark: dynamic tagging is not sufficient to compose different protocols.

Protocol 1

$$\begin{array}{ll} A \rightarrow B: & A, N_1 \\ B \rightarrow A: & B, N_2 \\ A \rightarrow B: & \mathrm{enca}(\langle A, N_1, B, N_2, s \rangle, \\ & \qquad \mathrm{pub}(B)) \end{array}$$

There is an attack on s:

- role B of P_2 with the tag $\langle A, N_1, B, N_2' \rangle$,
- role A of P_1 with the tag $\langle A, N_1, B, N_2' \rangle$.

How to combine both results ?

$\longrightarrow$ by using tags of the form tag $= \langle id_\alpha, A_1, N_1, \ldots, A_\ell, N_\ell \rangle$.

Remark: dynamic tagging is not sufficient to compose different protocols.

Protocol 1

$$\begin{array}{rl} A \rightarrow B: & A, N_1 \\ B \rightarrow A: & B, N_2 \\ A \rightarrow B: & \mathrm{enca}(\langle A, N_1, B, N_2, s\rangle, \\ & \qquad \mathrm{pub}(B)) \end{array}$$

Protocol 2

$$\begin{array}{rl} A \rightarrow B: & A, N_1' \\ B \rightarrow A: & B, N_2' \\ A \rightarrow B: & \mathrm{enca}(\langle A, N_1', B, N_2', N_a\rangle, \\ & \qquad \mathrm{pub}(B)) \\ B \rightarrow A: & N_a \end{array}$$

There is an attack on s:

- role B of P_2 with the tag $\langle A, N_1, B, N_2' \rangle$,
- role A of P_1 with the tag $\langle A, N_1, B, N_2' \rangle$.

How to combine both results ?

$\longrightarrow$ by using tags of the form tag $= \langle id_\alpha, A_1, N_1, \ldots, A_\ell, N_\ell \rangle$.

Remark: dynamic tagging is not sufficient to compose different protocols.

Protocol 1

$$\begin{array}{rl} A \rightarrow B: & A, N_1 \\ B \rightarrow A: & B, N_2 \\ A \rightarrow B: & \mathrm{enca}(\langle A, N_1, B, N_2, s\rangle, \\ & \hfill \mathrm{pub}(B)) \end{array}$$

Protocol 2

$$\begin{array}{rl} A \rightarrow B: & A, N'_1 \\ B \rightarrow A: & B, N'_2 \\ A \rightarrow B: & \mathrm{enca}(\langle A, N'_1, B, N'_2, N_a\rangle, \\ & \hfill \mathrm{pub}(B)) \\ B \rightarrow A: & N_a \end{array}$$

There is an attack on s:

- role B of P_2 with the tag $\langle A, N_1, B, N'_2 \rangle$,
- role A of P_1 with the tag $\langle A, N_1, B, N'_2 \rangle$.

Conclusion

Conclusion: Two composition results

- one that can be used to compose protocols that satisfy disjoint encryption
  $\longrightarrow$ this can be obtained with static tags
- one that is useful to compose sessions of the same protocol (general class of protocols)
  $\longrightarrow$ this can be obtained with dynamic tags

Both results are based on a refinement of the procedure due to H. Comon

Yet another composition result: with S. Kremer and M. Ryan

- another class of protocols: password based protocols
- another notion of security: resistance against guessing attacks

$\longrightarrow$ we use another notion of tagging

Conclusion

Conclusion: Two composition results

- one that can be used to compose protocols that satisfy disjoint encryption
  $\longrightarrow$ this can be obtained with static tags
- one that is useful to compose sessions of the same protocol (general class of protocols)
  $\longrightarrow$ this can be obtained with dynamic tags

Both results are based on a refinement of the procedure due to H. Comon

Yet another composition result: with S. Kremer and M. Ryan

- another class of protocols: password based protocols
- another notion of security: resistance against guessing attacks

$\longrightarrow$ we use another notion of tagging