Trace equivalence of protocols with randomised encryption for an unbounded number of sessions

Rémy Chrétien[1,2], sous la supervision de Véronique Cortier[3] et Stéphanie Delaune[2]

[1]Université Paris 7

[2]LSV, ENS Cachan & CNRS & INRIA Saclay Île-de-France

[3]LORIA - CNRS

7th November 2012

Plan

Introduction

Two families of formal properties for protocols:

- **reachability:** for secrecy or authentication, "is the attacker able to reach some term?"
- **equivalence:** for anonymity properties, "are two protocols similar to the attacker?"

Decidability of protocols with one variable

Existing results:

- Reachability is undecidable in most cases
- Reachability becomes decidable under specific constraints, *e.g.* bounded number of sessions, one variable, tagging
- Accessibility is decidable for a bounded number of sessions

Our goal: to provide a decidability result for equivalence with an unbounded number of sessions.

Signature

Only randomised encryption, symmetric or asymmetric, plus public/private constants with an equational theory:

Signature Σ

$$\Sigma = \Sigma_{\text{pub}} \uplus \Sigma_{\text{prv}}$$
$$\Sigma_{\text{pub}} = \{\text{rsenc}/3, \text{rsdec}/3, \text{raenc}/3, \text{radec}/3\} \cup A_{\text{pub}}$$
$$\Sigma_{\text{prv}} = \{\text{pk}/1\} \cup A_{\text{prv}}$$

Equational Theory E

$$E = \{\text{rsdec}(\text{rsenc}(x, y, z), y) = x, \text{radec}(\text{raenc}(x, \text{pk}(y), z), y) = x\}$$

Syntax

Syntax

$P, Q :=$	0	empty process
	$\text{in}(c, u).P$	reception on channel c
	$\text{out}(c, u).P$	emission on channel c
	$P \mid Q$	parallel composition
	$!P$	replication
	$\text{new } n.P$	fresh name generation

In the following u will contain *only one variable* and *no nonces* (except for random seeds) .

Semantics

Example: Input

$$(\mathcal{E}; \mathsf{in}(c,u).P \cup \mathcal{P}; \sigma) \xrightarrow{\mathsf{in}(c,R)} (\mathcal{E}; P\theta \cup \mathcal{P}; \sigma)$$

$$\text{where } \begin{cases} R \text{ is a public term (recipe) built on } \Sigma_{\mathsf{pub}} \text{ and } dom(\sigma) \\ \theta = mgu(u, R\sigma\downarrow) \neq \bot \end{cases}$$

where $(\mathcal{E}; \mathcal{P}; \sigma)$ is a configuration:

- $\mathcal{E}$ is a set of restricted names
- $\mathcal{P}$ is a multiset of processes
- σ is a substitution representing the exchanged messages

Trace equivalence

Definition (Traces)

$$\mathsf{trace}(K) = \{(s, \mathsf{new}\,\mathcal{E}.\sigma) \mid K \stackrel{s}{\Longrightarrow} (\mathcal{E}; \mathcal{P}; \sigma)\}.$$

Definition (Trace equivalence)

Let K_A and K_B configurations, $K_A \sqsubseteq_t K_B$ if for every $(\mathsf{tr}, \phi) \in \mathsf{trace}(K_A)$, there exists $(\mathsf{tr}, \phi') \in \mathsf{trace}(K_B)$ such that $\phi \sim_s \phi'$.
$K_A \approx_t K_B$ if $K_A \sqsubseteq_t K_B$ and $K_B \sqsubseteq_t K_A$.

A dummy e-passport (I)

Alice (A), owner of an electronic passport with her name (id_A) wants to identify herself to an authority.

Informal specification

$$\begin{array}{lll} A \to S: & \{id_A\}^n_{k_{AS}} & (a) \\ S \to A: & \{x\}^-_{k_{AS}} \to h(x), \{x\}^n_{k_{\mathbf{auth}}} & (b) \end{array}$$

- k_{AS} and k_{auth} are symmetric secret keys.
- h is a public hash function.

A dummy e-passport (II)

Formal specification

$$
\begin{array}{lll}
P_A = & \text{in}(c_A^A, \text{start}).\text{new } n.\text{out}(c_A^A, \text{rsenc}(id_A, k_{AS}, n)) & (1)\\
\mid\ ! & \text{in}(c_A^S, \text{rsenc}(x, k_{AS}, \_)).\text{out}(c_A^S, h(x)) & (2)\\
\mid\ ! & \text{in}(c_S, \text{rsenc}(x, k_{AS}, \_)).\text{new } n.\text{out}(c_S, \text{rsenc}(x, k_{\text{auth}}, n)) & (3)
\end{array}
$$

The same protocol can be defined for another distinct agent, B. Does our e-passport preserve the anonymity of its holder?

Proposition

$$P_A \not\approx_t P_B$$

The class R_1

R_1 is the class of protocols (P, ϕ_0)

$$P = \bigmid_{i=1}^{n} \bigmid_{j=1}^{p_i} \epsilon!\mathrm{in}(c_i, u_j^i).\mathrm{new}\ \tilde{n}_j^i.\mathrm{out}(c_i, u_j'^i)$$

such that:

- any constant used as a symmetric or private key in P must either be in ϕ_0 or not be reachable,
- input and output terms do not contain decryption symbols,
- blind-copy is only allowed once, for a plaintext,
- output terms $u_j'^i$ have to be cyphertexts with non-reachable keys or constants.

At this point, the protocols are *not* deterministic.

Undecidability of R_1

Theorem (Undecidability of R_1)

Trace equivalence in R_1 is undecidable.

Idea

We encode an instance of the Post Correspondence Problem: given two sets of tiles u_i and v_i, we build two protocols, P and Q such that $P \approx_t Q$ iff PCP has no solution.

Defining P:

- P chooses (non-deterministically) a set of tiles (the u_i or the v_i).
- The attacker, through P, builds a word u.
- P can then unstack the tiles from the set previously selected.
- If P chose the v_i **and** reached the empty word, it outputs a public constant "success".

Defining Q:

- Q automatically selects the v_i.
- The attacker, through Q, builds a word u.
- Q can then unstack the tiles v_i.
- If Q reached the empty word, it outputs the same public constant "success".

Why trace equivalence fails iff PCP has a solution:

- If PCP has a solution, u: the trace of P selecting the u_i and building u has no equivalent in Q.
- If PCP has no solution:
  - any trace of Q can be matched by P
  - any trace of P selecting the v_i can be matched by Q
  - any trace of P selecting the u_i can be matched as Q will never ouput "success"

A closer look to P and Q:

Non-deterministic choice

$$
\begin{array}{rlll}
P := & | & ! & \mathsf{in}(c, x).\mathsf{new}\ n.\mathsf{new}\ m.\mathsf{out}(c, \mathsf{rsenc}(\mathsf{rsenc}(\epsilon, k_u, n), k, m)) \\
& | & ! & \mathsf{in}(c, x).\mathsf{new}\ n.\mathsf{new}\ m.\mathsf{out}(c, \mathsf{rsenc}(\mathsf{rsenc}(\epsilon, k_v, n), k, m)) \\
& | & & \mathsf{in}(c', \mathsf{rsenc}(x, k, \_).\mathsf{new}\ n.\mathsf{out}(c, \mathsf{rsenc}(x, k', n)) \\
& & & \ldots
\end{array}
$$

In the end, the attacker knows either $\{\{\epsilon\}_{k_u}\}_{k'}$ or $\{\{\epsilon\}_{k_v}\}_{k'}$ **and** does not know which one.

Building the word u

$$
\begin{aligned}
P := \quad & \ldots \\
& |!\mathsf{in}(c_a, \mathsf{senc}(\mathsf{senc}(x, k_u, \_), k, \_)).\mathsf{out}(c_a, \mathsf{senc}(\mathsf{senc}(\mathsf{senc}(x, a, l), k_u, n), k, m)) \\
& |!\mathsf{in}(c_b, \mathsf{senc}(\mathsf{senc}(x, k_u, \_), k, \_)).\mathsf{out}(c_b, \mathsf{senc}(\mathsf{senc}(\mathsf{senc}(x, b, l), k_u, n), k, m)) \\
& \ldots
\end{aligned}
$$

and the same with key k_v.

Unstacking the tiles u_i

$$
\begin{aligned}
P_U(k) := \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot u_i, k_0, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(x, k_1, n), k, m) \\
& |!\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot u_i, k_1, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(x, k_1, n), k, m)) \\
& |!\mathsf{in}(c', \mathsf{senc}(\mathsf{senc}(\epsilon, k_1, \_), k, \_)).\mathsf{out}(c', \mathsf{senc}(\mathsf{senc}(\epsilon, k_2, n), k, m))
\end{aligned}
$$

Note: for the sake of brevity, name generation for randomised encryption is here forgotten.

Unstacking the tiles v_i

$$
\begin{aligned}
P_V(k) := \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot v_i, k'_0, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(x, k'_1, n), k, m)) \\
| \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot v_i, k'_1, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(x, k'_1, n), k, m)) \\
| \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot w, k'_0, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot w, k'_3, n), k, m)) \\
| \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot w, k'_1, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(x \cdot w, k'_3, n), k, m)) \\
| \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(w', k'_0, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(w', k'_3, n), k, m)) \\
| \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(w', k'_1, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(w', k'_3, n), k, m)) \\
| \quad & !\mathsf{in}(c_i, \mathsf{senc}(\mathsf{senc}(x, k'_3, \_), k, \_)).\mathsf{out}(c_i, \mathsf{senc}(\mathsf{senc}(x, k'_3, n), k, m)) \\
| \quad & !\mathsf{in}(c', \mathsf{senc}(\mathsf{senc}(\epsilon, k'_1, \_), k, \_)).\mathsf{out}(c', \mathsf{success}) \\
| \quad & !\mathsf{in}(c', \mathsf{senc}(\mathsf{senc}(x \cdot \alpha, k'_1, \_), k, \_)).\mathsf{out}(c', \mathsf{senc}(\mathsf{senc}(x \cdot \alpha, k'_2, n), k, m)) \\
| \quad & !\mathsf{in}(c', \mathsf{senc}(\mathsf{senc}(x, k'_3, \_), k, \_)).\mathsf{out}(c', \mathsf{senc}(\mathsf{senc}(x, k'_2, n), k, m))
\end{aligned}
$$

The class R_2

R_2 is the class of protocols (P, ϕ_0),

$$P = \prod_{i=1}^{n} \prod_{j=1}^{p_i} \epsilon!\mathrm{in}(c_i, u_j^i).(\prod_{l=1}^{m_j^i} \mathrm{new}\ n_l).\mathrm{out}(c_i, u_j'^i)$$

such that:

- $(P, \phi_0) \in R_1$
- for all $i \in \{1, \ldots, n\}$ and $j, j' \in \{1, \ldots, p_i\}$, if $j \neq j'$ then for any renaming of variables, $mgu(u_j^i, u_{j'}^i) = \bot$.

At this point, the protocols *are* deterministic.

Decidability of R_2

Theorem (Decidability of R_2)

Trace equivalence in R_2 is undecidable.

Idea

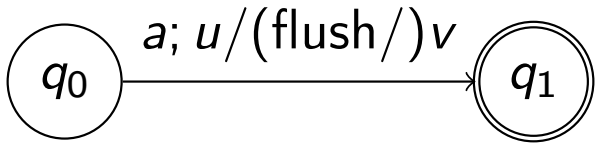
We reduce the problem of trace equivalence of protocols to the equivalence of deterministic pushdown automata (Sénizergues, 2001)

Real-time Flushing Generalized Deterministic Pushdown Automata

A subset of the class of deterministic pushdown automata (DPA):

- each transition can unstack / stack a finite word on the stack alphabet
- a transition can additionnaly flush the whole stack

RFGDPA strictly contain real-time deterministic pushdown automa (as the latter cannot empty their stack) and are strictly less expressive than deterministic pushdown automata (because of ϵ-transitions).



From protocols of R_2 to RFGDPA

1. Trace equivalence in R_2 can be simplified: two traces are equivalent if they share the same labels and the same equalities hold
   - equalities between an element of the frame and a public constant
   - equalities between two elements of the frames equal to a private constant
2. Those kinds of equalities can easily be verified by an automaton.

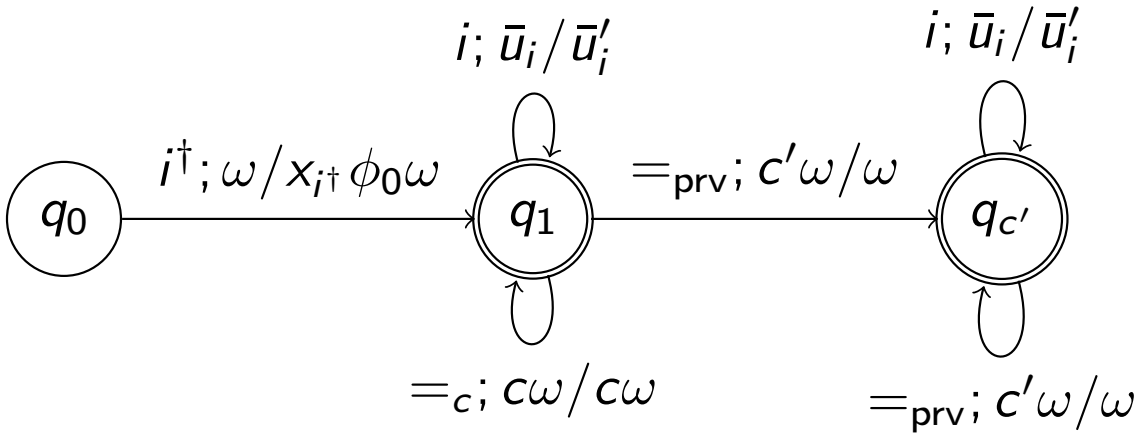


Figure : Automaton $\mathcal{A}_P$

where c is a public constant and c' is private.
And because P is deterministic, $\mathcal{A}_P$ is deterministic too.

From trace to language equivalence

Let (P, ϕ_0) and (Q, ψ_0) be two elements of R_2. Then $(P, \phi_0) \approx_t (Q, \psi_0)$ if, and only if, $L(\mathcal{A}_P) = L(\mathcal{A}_Q)$.

L(a)=l(b)? decidability results form complete formal systems, Sénizergues, 2001

Equivalence of deterministic pushdown automata is decidable

Which leads to the desired result:

Decidability of R_2

Trace equivalence in R_2 is decidable.

A converse?

From language to trace equivalence

Let $\mathcal{A}$ and $\mathcal{A}'$ be two RFGDPA. Then $L(\mathcal{A}') = L(\mathcal{A}')$ if, and only if, $(P, \phi_0) \approx_t (P', \phi_0')$.

Given the FGDPA $\mathcal{A}$, we now define the protocol (P, ϕ_0) in the following way:

$$\phi_0 = \text{new } n.\{x_1 \triangleright \text{rsenc}(\omega, q_0, n)\}$$

$$\begin{array}{rlll} P := & ! & \text{in}(c_a, \text{rsenc}(x, u.q, \_)).\text{new } \tilde{n}.\text{out}(c_a, \text{rsenc}(x, v.q', \tilde{n})) & (1_{q,a,u}) \\ | & ! & \text{in}(c_a, \text{rsenc}(x, u'.q, \_)).\text{new } \tilde{n}.\text{out}(c_a, \text{rsenc}(\omega, v'.q', \tilde{n})) & (2_{q,a,u'}) \\ | & ! & \text{in}(c_f, \text{rsenc}(x, q_f, \_)).\text{out}(c_f, \text{success}) & (3_{q_f}) \end{array}$$

A complexity gap

Complexity-wise, only little is known for deterministic pushdown automata:

- Equivalence is primitive recursive (Stirling, 2002)
- No non-trivial lower bound (no NP/coNP-hardness result known)

Still, because our transformation is not polynomial, we get the following:

NP-hardess

Trace equivalence in R_2 is coNP-hard.

Idea: encode the Hamiltonian path problem into a reachability problem in R_2.

To sum up

- There is a tight link between equivalence of protocols and RFG(D)PA.
- Trace equivalence in R_1 is undecidable because of the non-determinism of its protocols.
- Trace equivalence in R_2 is on the other hand decidable thanks to their determinism.
- Complexity results are scarce, but still coNP-hard bound.

In the future

To do:

- extend R_2: get back to simple processes but enlarge our signature, weaken the other conditions (output terms should not be cyphertexts only)
- get a better lower bound on the complexity of R_2: is trace equivalence PSPACE-hard?