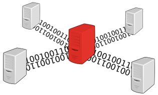
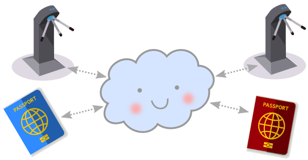
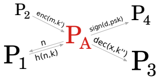
Formal Proofs of Cryptographic Protocols

Modelling and Verifying Unlinkability







David Baelde

LMF, ENS Paris-Saclay

March 9, 2021

Qui suis-je?

Parcours

- 2000: MPSI/MP*, ENS Lyon, MPRI, thèse à Polytechnique:
  A linear approach to the proof theory of least and greatest fixed points
- 2009: Postdoc à U. Minnesota, Paris-Sud, ITU Copenhagen
- 2012: Maître de conférences à l'ENS Paris-Saclay

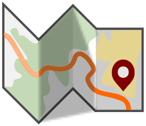
Recherche au Laboratoire Méthodes Formelles (LMF = LSV + LRI/VaIs)

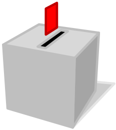
- Théorie de la preuve: preuves infinitaires, hyperséquents
- Vérification de protocoles cryptographiques

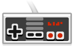
Enseignement à l'ENS Paris-Saclay

- Enseignements en L3, M1, M2 et prépa agrégation,
  notamment en logique, sécurité, programmation et génie logiciel

Increasingly many activities are becoming digitalized.

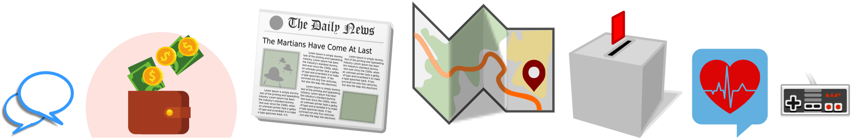






Security & Privacy

Increasingly many activities are becoming digitalized.

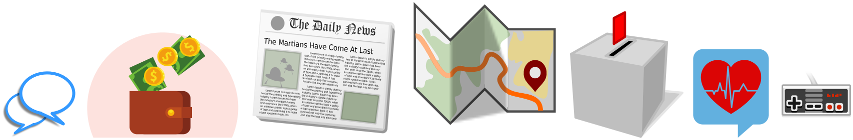


These systems must ensure important properties:

- security: secrecy, authenticity, no double-spending...
- privacy: anonymity, absence of tracking...

Security & Privacy

Increasingly many activities are becoming digitalized.



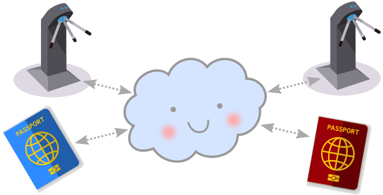
These systems must ensure important properties:

- security: secrecy, authenticity, no double-spending...
- privacy: anonymity, absence of tracking...

Frequent flaws at various levels can be avoided using science:

- hardware, software and specifications;
- cryptographic primitives and protocols.

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

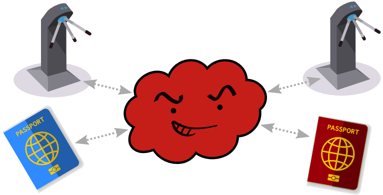
Reader (R) knows all legitimate keys.

$$\begin{array}{ccccl} R & \rightarrow & T_i & : & n_R \\ T_i & \rightarrow & R & : & \mathsf{h}(n_R, k_i) \end{array}$$

Scenario under consideration:

- roles R, T_1, ..., T_n; arbitrary number of sessions for each role

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

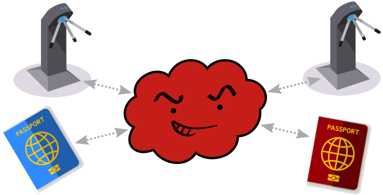
Reader (R) knows all legitimate keys.

$$\begin{array}{llll} R & \rightarrow & T_i & : & n_R \\ T_i & \rightarrow & R & : & \mathsf{h}(n_R, k_i) \end{array}$$

Scenario under consideration:

- roles R, T_1, ..., T_n; arbitrary number of sessions for each role
- attacker can intercept messages, inject new messages

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

Reader (R) knows all legitimate keys.

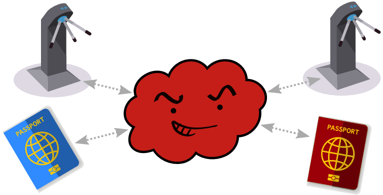
$$\begin{array}{lcccl} R & \to & T_i & : & n_R \\ T_i & \to & R & : & \mathsf{h}(n_R, k_i) \end{array}$$

Scenario under consideration:

- roles R, T_1, ..., T_n; arbitrary number of sessions for each role
- attacker can intercept messages, inject new messages

Readers correctly authenticate tags.

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

Reader (R) knows all legitimate keys.

$$\begin{array}{lllll} R & \rightarrow & T_i & : & n_R \\ T_i & \rightarrow & R & : & \mathsf{h}(n_R, k_i) \end{array}$$

Scenario under consideration:

- roles R, $T_1, \ldots, T_n$; arbitrary number of sessions for each role
- attacker can intercept messages, inject new messages

Readers correctly authenticate tags.

Tags can be tracked: privacy violation.

- The attacker can obtain the pseudonym $\mathsf{h}(0, k_i)$ from a tag.

This talk

Part 1: Formal proofs of cryptographic protocols

- The computational and symbolic models
- Existing verification techniques

Part 2: Modelling and verifying unlinkability

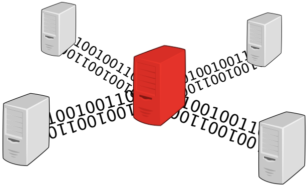
- A formal definition of *strong unlinkability*
- Synthesizing sufficient conditions from attacks
- Verifying conditions using state-of-the-art tools

This is based on joint work with Lucca Hirschi (LORIA), Stéphanie Delaune (IRISA) and Solène Moreau (IRISA).

Part 1/2

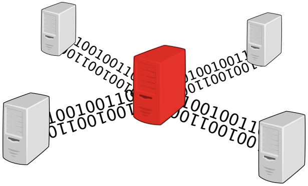
Formal Proofs of Cryptographic Protocols

The computational model



Messages	=	bitstrings
Secrets	=	random samplings
Primitives	=	PTIME Turing machines
Participants	=	PPTIME Turing machines

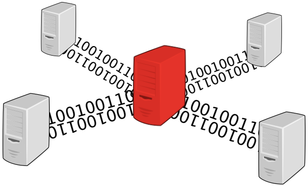
The computational model



Messages	$=$	bitstrings
Secrets	$=$	random samplings
Primitives	$=$	PTIME Turing machines
Participants	$=$	PPTIME Turing machines

In general, properties only hold with overwhelming probability, under some assumptions on cryptographic primitives.

The computational model



Messages	=	bitstrings
Secrets	=	random samplings
Primitives	=	PTIME Turing machines
Participants	=	PPTIME Turing machines

In general, properties only hold with overwhelming probability, under some assumptions on cryptographic primitives.

Example (Unforgeability, EUF-CMA)

There is a negligible probability of success for the following game, for any attacker $\mathcal{A}$:

- Draw k uniformly at random.
- $\langle u, v\rangle := \mathcal{A}^{\mathcal{O}}$ where $\mathcal{O}$ is the oracle $x \mapsto \mathsf{h}(x, k)$.
- Succeed if $u = \mathsf{h}(v, k)$ and $\mathcal{O}$ has not been called on v.

Naive protocol in the computational model

Authentication

Attacker can interact with tags and readers,
wins if some reader accepts a message that has not been emitted by a tag.

Naive protocol in the computational model

Authentication

Attacker can interact with tags and readers,
wins if some reader accepts a message that has not been emitted by a tag.

- Attacker has to obtain some $\mathsf{h}(n_R, k_i)$ without querying T_i on n_R.
- Impossible if h is unforgeable.

Naive protocol in the computational model

Authentication

Attacker can interact with tags and readers,
wins if some reader accepts a message that has not been emitted by a tag.

- Attacker has to obtain some $\mathsf{h}(n_R, k_i)$ without querying T_i on n_R.
- Impossible if h is unforgeable.

Privacy

Attacker interacts with either T_A, T_B or T_A, T_A
wins if he guesses in which situation he is
(with probability significantly different from $\frac{1}{2}$).

- Success with probability almost 1 thanks to pseudonyms.

Naive protocol in the computational model

Authentication

Attacker can interact with tags and readers,
wins if some reader accepts a message that has not been emitted by a tag.

- Attacker has to obtain some $\mathsf{h}(n_R, k_i)$ without querying T_i on n_R.
- Impossible if h is unforgeable.

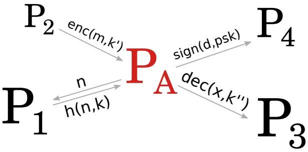
Privacy

Attacker interacts with either T_A, T_B or T_A, T_A
wins if he guesses in which situation he is
(with probability significantly different from $\frac{1}{2}$).

- Success with probability almost 1 thanks to pseudonyms.

Proofs in computational model are tedious, error-prone.
Formal verification techniques have been developed first for more abstract models...

A symbolic model: messages

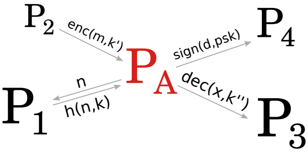


Messages	=	terms modulo equations
Secrets	=	fresh *names*
		(no probabilities)

Example (Equational theories)

- Hash functions: no equations.
- Xor: associativity, commutativity, neutrality and cancellation.

A symbolic model: messages



Messages	=	terms modulo equations
Secrets	=	fresh *names*
		(no probabilities)

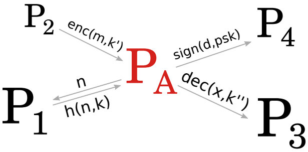
Definition (Deduction)

Given a set of private names E and known messages $\sigma = \{x_i \mapsto m_i\}_{i \in [1;n]}$, message s is *deducible* when there R such that $R\sigma =_{\mathsf{E}} s$ and R does not contain any name of E.

Example

With $E = \{n, k\}$ and $\sigma = \{\ x \mapsto n \oplus \mathsf{h}(n,k),\ y \mapsto n\ \}$, deduce $\mathsf{h}(n,k)$ using $R = \dots$

A symbolic model: messages



Messages	=	terms modulo equations
Secrets	=	fresh *names*
		(no probabilities)

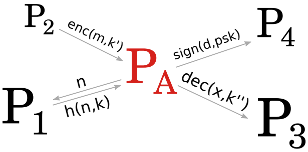
Definition (Deduction)

Given a set of private names E and known messages $\sigma = \{x_i \mapsto m_i\}_{i \in [1;n]}$, message s is *deducible* when there R such that $R\sigma =_{\mathsf{E}} s$ and R does not contain any name of E.

Example

With $E = \{n, k\}$ and $\sigma = \{\ x \mapsto n \oplus \mathsf{h}(n, k),\ \ y \mapsto n\ \}$,
deduce $\mathsf{h}(n, k)$ using $R = x \oplus y$.

A symbolic model: messages



Messages	=	terms modulo equations
Secrets	=	fresh *names*
		(no probabilities)

Definition (Static equivalence, $\sigma \sim \sigma'$)

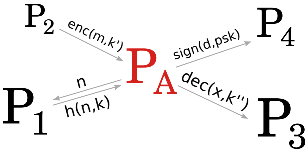
Given a set of private names E, two *frames* σ and σ' with same domain are statically equivalent when, for any R_1 and R_2,

$$R_1\sigma =_{\mathsf{E}} R_2\sigma \quad \text{iff} \quad R_1\sigma' =_{\mathsf{E}} R_2\sigma'$$

Example (Empty E, or no equation involving h (and names))

Let $E = \{k, n, m\}$, $\sigma = \{x \mapsto \mathsf{h}(n,k), y \mapsto n\}$ and $\sigma' = \{x \mapsto m, y \mapsto n\}$.
We have $\sigma \overset{?}{\sim} \sigma'$

A symbolic model: messages



Messages = terms modulo equations

Secrets = fresh *names*

(no probabilities)

Definition (Static equivalence, $\sigma \sim \sigma'$)

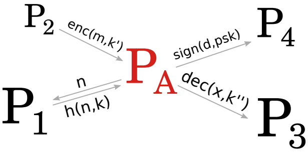
Given a set of private names E, two *frames* σ and σ' with same domain are statically equivalent when, for any R_1 and R_2,

$$R_1\sigma =_{\mathsf{E}} R_2\sigma \quad \text{iff} \quad R_1\sigma' =_{\mathsf{E}} R_2\sigma'$$

Example (Empty E, or no equation involving h (and names))

Let $E = \{k, n, m\}$, $\sigma = \{x \mapsto \mathsf{h}(n, k), y \mapsto n\}$ and $\sigma' = \{x \mapsto m, y \mapsto n\}$.
We have $\sigma \ \sim \ \sigma'$ and $\sigma \cup \{z \mapsto k\} \sim^{?} \sigma' \cup \{z \mapsto k\}$.

A symbolic model: messages



Messages = terms modulo equations

Secrets = fresh *names*

(no probabilities)

Definition (Static equivalence, $\sigma \sim \sigma'$)

Given a set of private names E, two *frames* σ and σ' with same domain are statically equivalent when, for any R_1 and R_2,

$$R_1\sigma =_{\mathsf{E}} R_2\sigma \quad \text{iff} \quad R_1\sigma' =_{\mathsf{E}} R_2\sigma'$$

Example (Empty E, or no equation involving h (and names))

Let $E = \{k, n, m\}$, $\sigma = \{x \mapsto \mathsf{h}(n, k), y \mapsto n\}$ and $\sigma' = \{x \mapsto m, y \mapsto n\}$. We have $\sigma \ \sim \ \sigma'$ and $\sigma \cup \{z \mapsto k\} \ \not\sim \ \sigma' \cup \{z \mapsto k\}$.

A symbolic model: participants

Participants modelled using a process algebra, e.g. applied π-calculus.

Example (Naive protocol)

$$
\begin{array}{lcl}
T_i & \stackrel{def}{=} & \mathbf{in}(c,x).\mathbf{out}(c,\mathsf{h}(x,k_i)) \\
R & \stackrel{def}{=} & \mathbf{new}\ n.\mathbf{out}(c,n).\mathbf{in}(c,y).\mathbf{if}\ \exists i.\ \ y=\mathsf{h}(n,k_i)\ \mathbf{then}\ \mathbf{out}(c,\mathsf{ok}) \\
S & \stackrel{def}{=} & \mathbf{new}\ k_1,\ldots,k_n.(!T_1 \mid \ldots \mid !T_n \mid !R)
\end{array}
$$

A symbolic model: participants

Participants modelled using a process algebra, e.g. applied π-calculus.

Example (Naive protocol)

$$T_i \stackrel{def}{=} \mathbf{in}(c, x).\mathbf{out}(c, \mathsf{h}(x, k_i))$$
$$R \stackrel{def}{=} \mathbf{new}\ n.\mathbf{out}(c, n).\mathbf{in}(c, y).\mathbf{if}\ \exists i.\ \ y = \mathsf{h}(n, k_i)\ \mathbf{then}\ \mathbf{out}(c, \mathsf{ok})$$
$$S \stackrel{def}{=} \mathbf{new}\ k_1, \ldots, k_n.(!T_1 \mid \ldots \mid !T_n \mid !R)$$

Accessibility

Given a system S, does there exist an attacker process A such that $S \mid A$ executes towards a *bad situation*: secret is revealed, agent accepts inauthentic message, etc.

Example

For any A, $S \mid A \not\rightarrow^* (\mathbf{out}(\_, k_i) \mid \_)$ i.e. keys remain secret.

A symbolic model: participants

Participants modelled using a process algebra, e.g. applied π-calculus.

Example (Naive protocol)

$$
\begin{aligned}
T_i &\stackrel{def}{=} \mathbf{in}(c,x).\mathbf{out}(c,\mathsf{h}(x,k_i)) \\
R &\stackrel{def}{=} \mathbf{new}\ n.\mathbf{out}(c,n).\mathbf{in}(c,y).\mathbf{if}\ \exists i.\ \ y=\mathsf{h}(n,k_i)\ \mathbf{then}\ \mathbf{out}(c,\mathsf{ok}) \\
S &\stackrel{def}{=} \mathbf{new}\ k_1,\ldots,k_n.(!T_1 \mid \ldots \mid !T_n \mid !R)
\end{aligned}
$$

May-testing equivalence $S \approx_m S'$

A system S satisfies a test A when $S \mid A$ may "execute successfully".
Two systems are may-testing equivalent when they satisfy the same tests.

Example

$T_1 \mid T_2$ and $T_1 \mid T_1$ are not equivalent.

A symbolic model: labelled transition system (LTS)

Avoid explicit attacker by studying interaction traces.

$(P, \Phi) \xrightarrow{\alpha} (Q, \Psi)$ where $\begin{cases} \textit{States} \text{ combine process } P \text{ with frame } \Phi = E.\sigma. \\ \textit{Actions } \alpha \text{ of the form } \mathbf{in}(c, R) \text{ or } \mathbf{out}(c, w). \end{cases}$

Example

$$(T_i \mid T_j, \Phi_0) \xrightarrow{\mathbf{in}(c,0).\mathbf{in}(c,0).\mathbf{out}(c,w).\mathbf{out}(c,w')} (0, \Phi_0 \cup \{\, w \mapsto \mathsf{h}(0, k_i), w' \mapsto \mathsf{h}(0, k_j) \,\})$$

A symbolic model: labelled transition system (LTS)

Avoid explicit attacker by studying interaction traces.

$(P,\Phi) \xrightarrow{\alpha} (Q,\Psi)$ where $\begin{cases} \textit{States} \text{ combine process } P \text{ with frame } \Phi = E.\sigma. \\ \textit{Actions } \alpha \text{ of the form } \mathbf{in}(c,R) \text{ or } \mathbf{out}(c,w). \end{cases}$

Example

$$(T_i \mid T_j, \Phi_0) \xrightarrow{\mathbf{in}(c,0).\mathbf{in}(c,0).\mathbf{out}(c,w).\mathbf{out}(c,w')} (0, \Phi_0 \cup \{\ w \mapsto \mathsf{h}(0,k_i), w' \mapsto \mathsf{h}(0,k_j)\ \})$$

Equivalences

- Trace equivalence $(P,\Phi) \approx_t (Q,\Psi)$, almost coincides with $\approx_m$.

A symbolic model: labelled transition system (LTS)

Avoid explicit attacker by studying interaction traces.

$(P, \Phi) \xrightarrow{\alpha} (Q, \Psi)$ where $\begin{cases} \textit{States} \text{ combine process } P \text{ with frame } \Phi = E.\sigma. \\ \textit{Actions } \alpha \text{ of the form } \mathbf{in}(c, R) \text{ or } \mathbf{out}(c, w). \end{cases}$

Example

$$(T_i \mid T_j, \Phi_0) \xrightarrow{\mathbf{in}(c,0).\mathbf{in}(c,0).\mathbf{out}(c,w).\mathbf{out}(c,w')} (0, \Phi_0 \cup \{\ w \mapsto \mathsf{h}(0, k_i), w' \mapsto \mathsf{h}(0, k_j)\ \})$$

Equivalences

- Trace equivalence $(P, \Phi) \approx_t (Q, \Psi)$, almost coincides with $\approx_m$.
- Bisimilarity $\approx_s$ is "the finest reasonable equivalence".
  Comes with logical characterization, and bisimulation proof technique.
  Coincides with trace equivalence on *determinate* processes.

A symbolic model: labelled transition system (LTS)

Avoid explicit attacker by studying interaction traces.

$(P, \Phi) \xrightarrow{\alpha} (Q, \Psi)$ where $\begin{cases} \textit{States} \text{ combine process } P \text{ with frame } \Phi = E.\sigma. \\ \textit{Actions } \alpha \text{ of the form } \mathbf{in}(c, R) \text{ or } \mathbf{out}(c, w). \end{cases}$

Example

$$(T_i \mid T_j, \Phi_0) \xrightarrow{\mathbf{in}(c,0).\mathbf{in}(c,0).\mathbf{out}(c,w).\mathbf{out}(c,w')} (0, \Phi_0 \cup \{\ w \mapsto \mathsf{h}(0, k_i), w' \mapsto \mathsf{h}(0, k_j)\ \})$$

Equivalences

- Trace equivalence $(P, \Phi) \approx_t (Q, \Psi)$, almost coincides with $\approx_m$.
- Bisimilarity $\approx_s$ is "the finest reasonable equivalence".
  Comes with logical characterization, and bisimulation proof technique.
  Coincides with trace equivalence on *determinate* processes.
- Diff-equivalences, even stronger, are equivalence notions expressed as reachability problems for *bi-processes*.

Verification in the symbolic model: accessibility

Accessibility problems are undecidable in general:

- unbounded protocol executions (unbounded sessions);
- unbounded recipes (message derivations by the attacker).

Verification in the symbolic model: accessibility

Accessibility problems are undecidable in general:

- unbounded protocol executions (unbounded sessions);
- unbounded recipes (message derivations by the attacker).

Verification techniques for bounded sessions:

- Symbolic execution + decidable constraint solving for some primitives.

For unbounded sessions:

- Semi-decision based on Horn clause abstraction (Proverif).
- Semi-automated prover based on multiset rewriting (Tamarin).

Verification in the symbolic model: accessibility

Accessibility problems are undecidable in general:

- unbounded protocol executions (unbounded sessions);
- unbounded recipes (message derivations by the attacker).

Verification techniques for bounded sessions:

- Symbolic execution + decidable constraint solving for some primitives.

For unbounded sessions:

- Semi-decision based on Horn clause abstraction (Proverif).
- Semi-automated prover based on multiset rewriting (Tamarin).

Some mature tools with industrial successes

- Casper, Proverif, AVISPA, Scyther, Tamarin
  (Oxford, Inria Paris & Nancy, ETH Zürich, CISPA)
- Breaking/fixing/proving Google SSO, 3G/5G authentication, Neuchatel & Belenios e-voting, WPA2, Signal, TLS 1.3, etc.

Verification in the symbolic model: equivalence

Equivalence also undecidable in general: it subsumes secrecy.

For bounded sessions
it is possible to (semi)decide trace equivalence for some primitives:

- Symbolic execution and constraint solving:
  SPEC (ANU), Apte (LSV & Inria Nancy) and DeepSec (Inria Nancy)
  (protocol equivalence is coNEXP-complete)
- Horn-clause resolution: Akiss (Inria Nancy)
- Planning and SAT-solving: SAT-Equiv (LSV & Inria Nancy)

Verification in the symbolic model: equivalence

Equivalence also undecidable in general: it subsumes secrecy.

For bounded sessions
it is possible to (semi)decide trace equivalence for some primitives:

- Symbolic execution and constraint solving:
  SPEC (ANU), Apte (LSV & Inria Nancy) and DeepSec (Inria Nancy)
  (protocol equivalence is coNEXP-complete)
- Horn-clause resolution: Akiss (Inria Nancy)
- Planning and SAT-solving: SAT-Equiv (LSV & Inria Nancy)

For unbounded sessions:

- Proverif and Tamarin can verify diff-equivalence.
- More specialized techniques e.g. based on type systems,
  small attack properties.

Part 2/2

Modelling and Verifying Unlinkability

An informal definition of unlinkability

ISO/IEC standard 15408

> *ensuring that a user may make multiple uses of a service or resource without others being able to link these uses together*

This is stronger than anonymity, and prevents any form of tracking.

Strong unlinkability with generic readers

Definition from [B., Delaune & Moreau, 2020]
inspired by [Arapinis et al., 2010]:

$$!R \mid !\, \textbf{new}\ \overline{k}.\ !T(\overline{k}) \qquad \approx_t \qquad !R \mid !\, \textbf{new}\ \overline{k}.\ T(\overline{k})$$

multiple-session/real scenario *single-session/ideal scenario*

Key contribution:

- A precise model of readers with shared database of credentials.

Strong unlinkability with generic readers

Definition from [B., Delaune & Moreau, 2020]
inspired by [Arapinis et al., 2010]:

$$!R \;\mid\; !\,\textbf{new}\ \overline{k}.\ !T(\overline{k}) \qquad \approx_t \qquad !R \;\mid\; !\,\textbf{new}\ \overline{k}.\ T(\overline{k})$$

multiple-session/real scenario — *single-session/ideal scenario*

Key contribution:

- A precise model of readers with shared database of credentials.

Remarks:

- All tags (resp. readers) on same channel: $\approx_t$ and $\approx_s$ a priori differ.
- Tag sessions can be made sequential using alternative construct i T.

The problem

In its general formulations, strong unlinkability cannot be directly verified using off-the-shelf verification tools:

Tamarin and Proverif's diff-equivalences are too constraining.

Our approach:

Identify reasonable conditions
that imply unlinkability
and are easier to verify using existing tools.

Condition 1

Our naive protocol fails unlinkability
because messages leak information about the tags' identity.

Definition (Frame opacity)

For any execution of the multiple-session system $(S_m, \emptyset) \xrightarrow{t} (S'_m, \Phi)$,

$$\Phi \sim \Phi_{\mathsf{ideal}}(t)$$

where messages of the ideal frame $\Phi_{\mathsf{ideal}}(t)$
may depend on session nonces but not on identity parameters.

Example (Basic Hash protocol)

$$T_i \to R : \langle n_T, \mathsf{h}(n_T, k_i) \rangle \quad \text{idealized into} \quad \langle n_T^1, n_T^2 \rangle$$

Condition 2

Definition (Well-authentication)

In any execution of the multiple-session system,
when some agent evaluates a test successfully,
it must have had an *honest interaction* with a dual agent.

Condition 2

Definition (Well-authentication)

In any execution of the multiple-session system,
when some agent evaluates a test successfully,
it must have had an *honest interaction* with a dual agent.

Consider the LAK protocol:

$$\begin{array}{lcl} R & \to & T & : & n_R \\ T & \to & R & : & \langle n_T, \mathsf{h}(n_R \oplus n_T, k)\rangle \\ R & \to & T & : & \text{(* not useful *)} \end{array}$$

Readers do not properly authenticate tags: why?
This leads to a failure of unlinkability: why?

Condition 3

Consider the OSK protocol,
using unkeyed hash functions g and h and a parameter $b \in \mathbb{N}$:

- Each tag has a secret k, readers have a database of known secrets.
- At each round the tag emits $\mathsf{g}(\mathsf{h}(k))$ and updates $k := \mathsf{h}(k)$.
- Readers accept messages of the form $\mathsf{g}(\mathsf{h}^n(k))$ for $n \in [0; b]$ and k in the database, which is then replaced by $\mathsf{h}^{n+1}(k)$.

It is not unlinkable: why?

Condition 3

Consider the OSK protocol,
using unkeyed hash functions g and h and a parameter $b \in \mathbb{N}$:

- Each tag has a secret k, readers have a database of known secrets.
- At each round the tag emits $\mathrm{g}(\mathrm{h}(k))$ and updates $k := \mathrm{h}(k)$.
- Readers accept messages of the form $\mathrm{g}(\mathrm{h}^n(k))$ for $n \in [0; b]$ and k in the database, which is then replaced by $\mathrm{h}^{n+1}(k)$.

It is not unlinkable: why?

Definition (No desynchronization)

In any execution of the multiple-session system, if a tag and reader have an honest interaction, then all of their tests must evaluate successfully.

Conditions are sufficient

Theorem

Strong unlinkability holds for all protocols that satisfy frame opacity, well-authentication and no desynchronization.

Proof

There is essentially only one way to map a multiple-session execution to a single-session execution. That execution is feasible and indistinguishable:

Conditions are sufficient

Theorem

Strong unlinkability holds for all protocols that satisfy frame opacity, well-authentication and no desynchronization.

Proof

There is essentially only one way to map a multiple-session execution to a single-session execution. That execution is feasible and indistinguishable:

- If a test passes in the multiple-session execution, it results from an honest interaction. It is still honest in the single-session execution, and thus passes.

Conditions are sufficient

Theorem

Strong unlinkability holds for all protocols that satisfy frame opacity, well-authentication and no desynchronization.

Proof

There is essentially only one way to map a multiple-session execution to a single-session execution. That execution is feasible and indistinguishable:

- If a test passes in the multiple-session execution, it results from an honest interaction. It is still honest in the single-session execution, and thus passes.
- If a test fails in the multiple-session execution, it must result from a dishonest interaction. The dishonest interaction must lead to a failed test on the single-session side.

Conditions are sufficient

Theorem

Strong unlinkability holds for all protocols that satisfy frame opacity, well-authentication and no desynchronization.

Proof

There is essentially only one way to map a multiple-session execution to a single-session execution. That execution is feasible and indistinguishable:

- If a test passes in the multiple-session execution, it results from an honest interaction. It is still honest in the single-session execution, and thus passes.
- If a test fails in the multiple-session execution, it must result from a dishonest interaction. The dishonest interaction must lead to a failed test on the single-session side.
- The multiple and single-session frames are indistinguishable from their idealizations, which coincide.

Conditions are reasonable and verifiable

We have been able to formally verify our conditions in the symbolic model for several protocols, using Proverif and Tamarin.

- Several RFID protocols, including fixed versions of LAK and OSK
- E-passport protocols BAC and PACE (with minor fixes)
- Some proofs of more complex protocols, involving counters or advanced primitives such as zero-knowledge proofs

First-time proofs!

Conditions are reasonable and verifiable

We have been able to formally verify our conditions in the symbolic model for several protocols, using Proverif and Tamarin.

- Several RFID protocols, including fixed versions of LAK and OSK
- E-passport protocols BAC and PACE (with minor fixes)
- Some proofs of more complex protocols, involving counters or advanced primitives such as zero-knowledge proofs

First-time proofs!

We could not carry out some analyses due to insufficient support for xor in these tools.

Conditions are reasonable and verifiable

We have been able to formally verify our conditions in the symbolic model for several protocols, using Proverif and Tamarin.

- Several RFID protocols, including fixed versions of LAK and OSK
- E-passport protocols BAC and PACE (with minor fixes)
- Some proofs of more complex protocols, involving counters or advanced primitives such as zero-knowledge proofs

First-time proofs!

We could not carry out some analyses due to insufficient support for xor in these tools.

We encountered a single case of incompleteness:
Tamarin finds a failure of well-authentication for PACE, but this failure seems harmless for unlinkability.
(Interestingly, well-auth. holds for Proverif due to weaker equational theory.)

Discussion: earlier work

Two-agent games

Game where attacker chooses two tags and must distinguish them.

- Proposed in [Avoine, 2005], with incorrect privacy claim for OSK.
- Strengthened in [Juels & Weis, 2006].

Discussion: earlier work

Two-agent games

Game where attacker chooses two tags and must distinguish them.

- Proposed in [Avoine, 2005], with incorrect privacy claim for OSK.
- Strengthened in [Juels & Weis, 2006].

Three-agent games

Attacker has to distinguish between T_1, T_1 and T_2, T_3.

- Two-agent games miss attacks involving concurrent tag sessions.
- Formal (bounded) verification of OSK in [Brusó et al., 2010] due to abusive removal of reader.

Discussion: earlier work

Two-agent games

Three-agent games

Weak and strong unlinkability [Arapinis et al., 2010]

- Weak unlinkability proposed as reasonable definition,
  but actually misses attacks.
- Strong unlinkability viewed as a proof technique, hence bisimilarity.
- Incorrect claim of unlinkability for BAC e-passport protocol.
  Bisimilarity actually leads to systematic failure of unlinkability for identity-specific readers!

Discussion: earlier work

Two-agent games

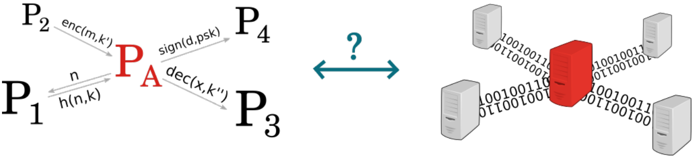
Three-agent games

Weak and strong unlinkability [Arapinis et al., 2010]

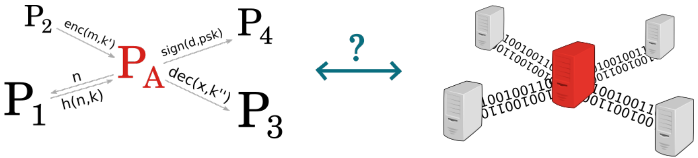
- Weak unlinkability proposed as reasonable definition, but actually misses attacks.
- Strong unlinkability viewed as a proof technique, hence bisimilarity.
- Incorrect claim of unlinkability for BAC e-passport protocol. Bisimilarity actually leads to systematic failure of unlinkability for identity-specific readers!

  (That part of the story is not over, as as [Filimonov et al., 2019] reports on an attack against BAC that our trace-equivalence-based definition misses.)

Discussion: computational model



Discussion: computational model

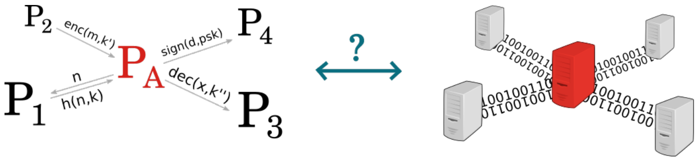


Some computational soundness results show that symbolic abstractions imply computational indistinguishability.

They remain limited by strong assumptions.

- No sound symbolic abstraction of xor in presence of replication.
- Most verification in symbolic model is disconnected from these results.

Discussion: computational model



Some computational soundness results show that
symbolic abstractions imply computational indistinguishability.

They remain limited by strong assumptions.

- No sound symbolic abstraction of xor in presence of replication.
- Most verification in symbolic model is disconnected from these results.

Alternative: direct verification in the computational model.

- Cryptoverif mimicks the cryptographer's *game-hopping* proofs.
- Easycrypt relies on *probabilistic relational Hoare logic*.
- With several colleagues, current work on the Squirrel prover...

Conclusion

Conclusion

This talk

- Formal models for cryptographic protocols.
- All kinds of problems: modelling, theory & practice.

What's next

- Currently developing a new prover in the computational model.
- Proofs of unlinkability with stronger guarantees,
  also new proofs for protocols involving xor.

Privacy

- A crucial need in modern societies, slowly being recognized as such.
- Requires broader analysis involving probabilities, time, data, ...
  See side-channel attacks, differential privacy, etc.

References

Baelde, Delaune, Jacomme, Koutsos & Moreau,
An Interactive Prover for Protocol Verification in the Computational Model,
S&P 2021 [PDF]

Baelde, Delaune & Moreau,
A Method for Proving Unlinkability of Stateful Protocols,
CSF 2020 [PDF]

Hirschi, Baelde & Delaune,
A Method for Unbounded Verification of Privacy-Type Properties,
JCS 2019 [PDF]

Hirschi, Baelde & Delaune,
A method for Verifying Privacy-Type Properties: the Unbounded Case,
S&P 2016 [PDF]

I have cited many tools and papers, don't hesitate to ask me for references.