Automatic Verification of Unlinkability and Anonymity for Unbounded Sessions

Sequoia Meeting

Lucca Hirschi

January 25, 2016

joint work with

David Baelde and Stéphanie Delaune

LSV and LSV
we need formal verification of crypto protocols covering privacy
we need formal verification of crypto protocols covering privacy

Goal:
- checking unlinkability and anonymity
- in the symbolic model (Dolev-Yao)
- for unbounded sessions
we need formal verification of crypto protocols covering privacy

Goal:

- checking unlinkability and anonymity
- in the symbolic model (Dolev-Yao)
- for unbounded sessions

- **Unlinkability** (≡untraceability) [ISO/IEC 15408]:
  Ensuring that a user may make multiple uses of a service or resource without others being able to link these uses together.

- **Anonymity** [ISO/IEC 15408]:
  Ensuring that a user may use a service or resource without disclosing the user’s identity. [...]
Context

Strong unlinkability [Arapinis, Chothia, Ritter, Ryan CSF’10]:

\[
\{ ! \nu k ! \nu n(T | R) \approx ! \nu k. \nu n(T | R) \}_{\mathcal{M}} \approx \{ ! \nu k. \nu n(T | R) \}_{S}
\]

- \(\mathcal{M}\): \(\infty\) many different \(T - R\) playing \(\infty\) many sessions
- \(S\): \(\infty\) many different \(T - R\) playing at most one session

Checking this is undecidable (because of replication)

Existing approaches:
- manual: need to exhibit huge bisimulations
- automatic (ProVerif/Maude-NPA/Tamarin): rely on abstraction (diff-equivalence) not precise enough
  \(\Rightarrow\) always fail to prove unlinkability

\(\Rightarrow\) there is a need for dedicated abstraction targeting unlinkability
Context

Strong unlinkability [Arapinis, Chothia, Ritter, Ryan CSF’10]:

\[ \exists \nu \vec{k} \exists \nu \vec{n}(T | R) \approx \exists \nu \vec{k}.\nu \vec{n}(T | R) \]

- \(\mathcal{M}\): \(\infty\) many different \(T - R\) playing \(\infty\) many sessions
- \(S\): \(\infty\) many different \(T - R\) playing at most one session

- Checking this is undecidable (because of replication)

Existing approaches:

- **manual**: need to exhibit huge bisimulations
- **automatic** (ProVerif/Maude-NPA/Tamarin):
  rely on abstraction (diff-equivalence) not precise enough
  \(\therefore\) always fail to prove unlinkability
Strong unlinkability [Arapinis, Chothia, Ritter, Ryan CSF’10]:

\[
\vdash \nu T \nu n(T \mid R) \approx \nu k. \nu n(T \mid R)
\]

- \(\mathcal{M}\): \(\infty\) many different \(T - R\) playing \(\infty\) many sessions
- \(S\): \(\infty\) many different \(T - R\) playing at most one session

- Checking this is undecidable (because of replication)

Existing approaches:

- **manual**: need to exhibit huge bisimulations
- **automatic** (ProVerif/Maude-NPA/Tamarin):
  - rely on abstraction (diff-equivalence) not precise enough
  - \(\leadsto\) always fail to prove unlinkability

\(\leadsto\) there is a need for dedicated abstraction targeting unlinkability
We identify:

- 2 conditions implying unlinkability and anonymity
- for a class of 2-agents protocols including our target case studies

We make sure:

- our conditions can be checked automatically using ProVerif
- they correspond to good design practices

⇒ sound approach to check automatically privacy properties working well in practice
I : What could go *wrong* 🙄?
R1: Messing with messages

Tag
$k, id$

Reader
$k$

\[\text{enc}(id, k)\]

Condition 1: Frame Opacity (FO)
▶ Goal: messages do not leak info about involved agents
▶ Intuitively: outputs are (statically) indistinguishable from \(\neq\) nonces

\[\{\text{enc}(id, k), \text{enc}(id, k)\} \sim \{nf_1, nf_2\}\]
R1: Messing with messages

Practical examples (RFID protocols): HB\(^+\), DM, KCL, LBV, LD, . . .
R1: Messing with messages

**Condition 1: Frame Opacity (FO)**

- **Goal**: messages do not leak info about involved agents
- **Intuitively**: outputs are (statically) indistinguishable from $\neq$ nonces

\[
\{\text{enc}(id, k), \text{enc}(id, k)\} \not\sim \{n_1^f, n_2^f\}
\]
R2: Messing with conditionals

Practical examples: BAC (ePassport), some versions of PACE (new version of ePassport), LAK, CH
R2: Messing with conditionals

Condition 2: Well-Authentication (WA)

- **Goal**: conditionals do not leak info about involved agents
- **Intuitively**: if a party does not abort then the attacker did not interfere
II : Big picture
<table>
<thead>
<tr>
<th>UK/ANO</th>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Equivalence?</td>
<td>Active Attacker?</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
</tbody>
</table>

↑ Theorem: implies ↑

<table>
<thead>
<tr>
<th>FO</th>
<th>“Messages are without relations”</th>
</tr>
</thead>
<tbody>
<tr>
<td>WA</td>
<td>“Conditionals hold only for honest interactions”</td>
</tr>
<tr>
<td></td>
<td>Equivalence?</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------</td>
</tr>
<tr>
<td>UK/ANO</td>
<td>✓</td>
</tr>
<tr>
<td>FO</td>
<td>✓</td>
</tr>
<tr>
<td>WA</td>
<td>□</td>
</tr>
</tbody>
</table>

Theorem: implies
<table>
<thead>
<tr>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>UK/ANO</strong></td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

\[\text{Theorem: implies}\]

<table>
<thead>
<tr>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FO</strong></td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>WA</strong></td>
<td></td>
</tr>
</tbody>
</table>

\[\text{can be checked}\]

- FO: automatic check of **diff-equivalence** using Proverif
- WA: automatic check of **correspondence prop.** using Proverif
<table>
<thead>
<tr>
<th></th>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UK/ANO</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>FO</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>WA</td>
<td></td>
<td>✔</td>
</tr>
</tbody>
</table>

**Theorem:** implies

**can be checked**

- FO: automatic check of **diff-equivalence** using Proverif
- WA: automatic check of **correspondence prop.** using Proverif

**Tight enough to conclude on our case studies:**

(BAC, LAK, Hash-Lock, EKE, SPKE)
III : Model and Problem
\( \Sigma \)-algebra + equational theory \( E \) + reduction rules (\( \text{à la Proverif} \))

\[
\begin{align*}
\Sigma_c &= \{\text{df}/2, \langle\_\_\_\rangle/2, \text{enc}/2, \text{ok}/0, \text{no}/0\} \\
\Sigma_d &= \{\pi_1/1, \pi_2/1, \text{dec}/2\} \\
E &= \{(\text{df}(\text{df}(x, y), z) = \text{df}(\text{df}(x, z), y))\} \\
\text{def}_\Sigma(\text{dec}) &= \{\text{dec}(\text{enc}(x, y), y) \rightarrow x\} \\
\text{def}_\Sigma(\pi_i) &= \{\pi_i(\langle x_1, x_2 \rangle) \rightarrow x_i\}
\end{align*}
\]

induce

\[
\begin{align*}
\text{a congruence } &= \quad \text{e.g., } g^{xyz} =_E g^{zyx} \\
\text{a “computation” relation } &\Downarrow \quad \text{e.g., } \text{dec}(\text{enc}(n, g^{ab}), g^{ba}) \Downarrow n
\end{align*}
\]
### Process

\[
P, Q := \begin{align*}
0 & \quad \text{null} \\
in(c, x).P & \quad \text{input} \\
out(c, u).P & \quad \text{output} \\
\text{let } \vec{x} := \vec{v} & \quad \text{evaluation/test} \\
P | Q & \quad \text{parallel} \\
!P & \quad \text{replication} \\
\nu n.P & \quad \text{restriction}
\end{align*}
\]
Applied-$\pi$ - Syntax

**Process**

\[ P, Q ::= \begin{align*}
0 & \quad \text{null} \\
\text{in}(c, x).P & \quad \text{input} \\
\text{out}(c, u).P & \quad \text{output} \\
\text{let } \vec{x} := \vec{v} \text{ in } P \text{ else } Q & \quad \text{evaluation/test} \\
P \mid Q & \quad \text{parallel} \\
!P & \quad \text{replication} \\
\nu n.P & \quad \text{restriction}
\end{align*} \]

**Configuration**

\[ A = (P; \Phi) \]

\[ \Phi = \{ w_1 \mapsto u_1, \ldots, w_n \mapsto u_n \} \]
### Applied-π - Semantics

\[
\begin{align*}
(in(c, x).P \cup P; \phi) & \xrightarrow{in(c,R)} (P\{x \mapsto u\} \cup P; \phi) & \text{if } R\phi \Downarrow u \\
(out(c, u).P \cup P; \phi) & \xrightarrow{out(c,w)} (P \cup P; \phi \cup \{w \mapsto u\}) & \text{if } w \text{ fresh} \\
(let \, \vec{x} := \vec{v} \in P \text{ else } Q \cup P; \phi) & \xrightarrow{t_l} (P\{\vec{x} \mapsto \vec{u}\} \cup P; \phi) & \text{if } \vec{v} \Downarrow \vec{u} \\
(let \, \vec{x} := \vec{v} \in P \text{ else } Q \cup P; \phi) & \xrightarrow{e} (Q \cup P; \phi) & \text{if } \exists i \, v_i \not\Downarrow \\
(\nu \, n.P \cup P; \phi) & \xrightarrow{t} (P \cup P; \phi) & \text{if } n \text{ fresh} \\
!P \cup P; \phi) & \xrightarrow{t} (P \cup !P \cup P; \phi) \\
\{P_1 \mid P_2\} \cup P; \phi) & \xrightarrow{t} (\{P_1, P_2\} \cup P; \phi)
\end{align*}
\]
Applied-$\pi$ - Semantics

$$(\text{in}(c, x).P \cup P; \phi) \xrightarrow{\text{in}(c,R)} (P\{x \mapsto u\} \cup P; \phi) \quad \text{if } R\phi \Downarrow u$$

$$(\text{out}(c, u).P \cup P; \phi) \xrightarrow{\text{out}(c,w)} (P \cup P; \phi \cup \{w \mapsto u\}) \quad \text{if } w \text{ fresh}$$

$$(\text{let } \overrightarrow{x} := \overrightarrow{v} \text{ in } P \text{ else } Q \cup P; \phi) \xrightarrow{\tau_t} (P\{\overrightarrow{x} \mapsto \overrightarrow{u}\} \cup P; \phi) \quad \text{if } \overrightarrow{v} \Downarrow \overrightarrow{u}$$

$$(\text{let } \overrightarrow{x} := \overrightarrow{v} \text{ in } P \text{ else } Q \cup P; \phi) \xrightarrow{\tau_e} (Q \cup P; \phi) \quad \text{if } \exists i v_i \Downarrow$$

$$(\nu n.P \cup P; \phi) \xrightarrow{\tau} (P \cup P; \phi) \quad \text{if } n \text{ fresh}$$

$$!P \cup P; \phi) \xrightarrow{\tau} (P \cup !P \cup P; \phi)$$

$$({P_1} \mid {P_2}) \cup P; \phi) \xrightarrow{\tau} ({P_1}, {P_2}) \cup P; \phi$$
Applied-$\pi$ - Semantics

\[
(in(c, x).\mathcal{P} \cup \mathcal{P}; \phi) \xrightarrow{in(c, R)} (\mathcal{P}\{x \mapsto u\} \cup \mathcal{P}; \phi) \quad \text{if } R\phi \Downarrow u
\]

\[
(out(c, u).\mathcal{P} \cup \mathcal{P}; \phi) \xrightarrow{out(c, w)} (\mathcal{P} \cup \mathcal{P}; \phi \cup \{w \mapsto u\}) \quad \text{if } w \text{ fresh}
\]

\[
(let \bar{x} := \bar{v} in P else Q \cup \mathcal{P}; \phi) \xrightarrow{t_i} (\mathcal{P}\{\bar{x} \mapsto \bar{u}\} \cup \mathcal{P}; \phi) \quad \text{if } \bar{v} \Downarrow \bar{u}
\]

\[
(let \bar{x} := \bar{v} in P else Q \cup \mathcal{P}; \phi) \xrightarrow{t_e} (Q \cup \mathcal{P}; \phi) \quad \text{if } \exists i v_i \not\Downarrow
\]

\[
(\nu n.\mathcal{P} \cup \mathcal{P}; \phi) \xrightarrow{=} (\mathcal{P} \cup \mathcal{P}; \phi) \quad \text{if } n \text{ fresh}
\]

\[
(!\mathcal{P} \cup \mathcal{P}; \phi) \xrightarrow{=} (\mathcal{P} \cup !\mathcal{P} \cup \mathcal{P}; \phi)
\]

\[
(\{P_1 | P_2\} \cup \mathcal{P}; \phi) \xrightarrow{=} (\{P_1, P_2\} \cup \mathcal{P}; \phi)
\]
Applied-$\pi$ - Trace Equivalence

### Static Equivalence

$\Phi \sim \Psi$ when

- $\text{dom}(\Phi) = \text{dom}(\Psi)$ and
- $\forall M, (M\Phi \nRightarrow M\Psi)$ and
- $\forall M, N, (M\Phi \Downarrow =_E N\Phi \iff M\Psi \Downarrow =_E N\Psi)$.

### Trace Equivalence

$A \sqsubseteq B$ when, for any $A \xrightarrow{\text{tr}} A'$ there exists $B \xrightarrow{\text{tr}'} B'$ such that $\text{obs}(\text{tr}) = \text{obs}(\text{tr}')$ and $\Phi(A') \sim \Phi(B')$.

$A \approx B$, when $A \sqsubseteq B$ and $B \sqsubseteq A$. 
Our class of protocols & our problem

Our class

- Intuitively, a party $P$ is a process of the form:

  ```
  out( , ).
  in( , ).
  let := in
    out( , )
    in( , )
    let [...] 
  else out( , )
  ```
Our class of protocols & our problem

Our class

- Intuitively, a party $P$ is a process of the form:
  
  ```
  out( , ).
in( , ).
let := in
  out( , )
in( , )
  let [...] else out( , )
  ```

- A protocol $\Pi$ is a tuple $(\vec{k}, \vec{n}_T, \vec{n}_R, T, R)$ where:
  - $T$ and $R$ are parties
  - $\vec{k}$: identity names and $\vec{n}_T/\vec{n}_R$: session names
  - $fn(T) \subseteq \vec{k} \cup \vec{n}_T$ (resp. for $R$)
Our class of protocols & our problem

Our class

- Intuitively, a party $P$ is a process of the form:

  ```
  out( , ).
  in( , ).
  let := in
  out( , )
  in( , )
  let [...]
  else out( , )
  ```

- A protocol $\Pi$ is a tuple $(\overrightarrow{k}, \overrightarrow{n}_T, \overrightarrow{n}_R, T, R)$ where:
  - $T$ and $R$ are parties
  - $\overrightarrow{k}$: identity names and $\overrightarrow{n}_T / \overrightarrow{n}_R$: session names
  - $fn(T) \subseteq \overrightarrow{k} \sqcup \overrightarrow{n}_T$ (resp. for $R$)

Unlinkability

$$
\begin{align*}
& \vdash \nu \overrightarrow{k} \neg (\nu \overrightarrow{n}_T T \mid \nu \overrightarrow{n}_R R) \\
& \equiv \vdash \nu \overrightarrow{k}.(\nu \overrightarrow{n}_T T \mid \nu \overrightarrow{n}_R R)
\end{align*}
$$
IV : Sufficient conditions
For any execution $A \xrightarrow{t} B$, we have that $\Phi(B) \sim [\Phi(B)]_{\text{nonce}}$. 
Frame opacity

For any execution $A \xrightarrow{t} B$, we have that $\Phi(B) \sim [\Phi(B)]^{\text{nonce}}$.

Require that all outputs are $\sim$ from nonces is too strong:

- $\Phi = \{ w \mapsto \langle \text{enc}(n_1, k), \text{enc}(n_2, k) \rangle \}$
- if $[\Phi]^{\text{nonce}} = \{ w \mapsto n \}$ then $\Phi \not\sim [\Phi]^{\text{nonce}}$
- if $[\Phi]^{\text{nonce}} = \{ w \mapsto \langle n, n' \rangle \}$ then $\Phi \sim [\Phi]^{\text{nonce}}$
Frame opacity

For any execution $A \xrightarrow{t} B$, we have that $\Phi(B) \sim [\Phi(B)]^{\text{nonce}}$. 

Require that all outputs are $\sim$ from nonces is too strong:

- $\Phi = \{ w \mapsto \langle \text{enc}(n_1, k), \text{enc}(n_2, k) \rangle \}$
- if $[\Phi]^{\text{nonce}} = \{ w \mapsto n \}$ then $\Phi \not\sim [\Phi]^{\text{nonce}}$
- if $[\Phi]^{\text{nonce}} = \{ w \mapsto \langle n, n' \rangle \}$ then $\Phi \sim [\Phi]^{\text{nonce}}$

Transparent function symbols

$f \in \Sigma_c$ is \textit{transparent} if:

- attacker can extract its arguments and
- does not appear in E.
Frame opacity

For any execution $A \xrightarrow{t} B$, we have that $\Phi(B) \sim [\Phi(B)]^\text{nonce}$.

- $\Phi = \{ w \mapsto \langle \text{enc}(n_1, k), \text{enc}(n_2, k) \rangle \}$
- $[\Phi]^\text{nonce} = \{ w \mapsto \langle n, n' \rangle \}$ and $\Phi \sim [\Phi]^\text{nonce}$

Transparent function symbols

$f \in \Sigma_c$ is transparent if:
- attacker can extract its arguments and
- does not appear in $E$.

Idealization

There exists a function $[\cdot]^\text{ideal} : \mathcal{T}(\Sigma_c, \mathcal{N}) \rightarrow \mathcal{T}(\Sigma_t, \{ \square \})$ such that:
- $[u]^\text{ideal} = f([u_1]^\text{ideal}, \ldots)$ if $u =_E f(u_1, \ldots)$ for some $f \in \Sigma_t$,
- and $[u]^\text{ideal} = \square$ otherwise.
Well-Authentication

\[ \Pi = (\vec{k}, \vec{n}_I, \vec{n}_R, T, R) \] is well-authenticating if, for any execution

\[ (M; \emptyset) \xrightarrow{t, T[\vec{k}, \vec{n}_1]} (P; \Phi) \]

there must be a \( R(\vec{k}, \vec{n}_2) \) such that \( T(\vec{k}, \vec{n}_1) \) and \( R(\vec{k}, \vec{n}_2) \) were having an honest execution in \( (t, \Phi) \).

A trace \( t \) is honest for a frame \( \Phi \) if

- \( \tau_e \notin t \) and
- \( \text{obs}(t) = \text{out}(. \cdot, w_0).\text{in}(. \cdot, M_0).\text{out}(. \cdot, w_1) \ldots \text{with } M_i\Phi \Downarrow =_{E} w_i\Phi. \)
Main Theorem

If $\Pi = (k, n_T, n_R, T, R)$ is well-authenticating and $M$ ensures frame opacity, then $\Pi$ ensures unlinkability.

A similar theorem for both unlinkability and anonymity.
V : Applications
Tool: UKano
We wrote UKano: a tool built on top of ProVerif that automatically checks our two sufficient conditions

New proofs of Unlinkability & Anonymity for:
▶ BAC + PA + AA (ePassport);
▶ (fixed) LAK (RFID auth.);
▶ Hash-Lock (RFID auth.);
▶ EKE, SPKE (PAKE protocols).

When conditions fail to hold: no direct attacks but still...

Flaws/attacks discovered:
▶ some versions of PACE (¬ UK);
▶ LAK (¬ UK).

Had some issues with PACE (ePassport v2)

Sources of UKano and ProVerif files at
https://sites.google.com/site/ukanosp/
Tool: UKano

We wrote UKano: a tool built on top of ProVerif that automatically checks our two sufficient conditions

New proofs of Unlinkability & Anonymity for:

- BAC+PA+AA (ePassport);
- (fixed) LAK (RFID auth.);
- Hash-Lock (RFID auth.);
- EKE, SPKE (PAKE protocols).
Tool: UKano

We wrote UKano: a tool built on top of ProVerif that automatically checks our two sufficient conditions

New proofs of Unlinkability & Anonymity for:

- BAC+PA+AA (ePassport);
- (fixed) LAK (RFID auth.);
- Hash-Lock (RFID auth.);
- EKE, SPKE (PAKE protocols).

When conditions fail to hold: no direct attacks but still...

Flaws/attacks discovered:

- some versions of PACE (¬ UK);
- LAK (¬ UK).

Had some issues with PACE (ePassport v2)

Sources of UKano and ProVerif files at
https://sites.google.com/site/ukanosp/
VI: Conclusion
<table>
<thead>
<tr>
<th>UK/ANO</th>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Theorem:** implies

- FO: “Messages are without relations”
- WA: “Conditionals hold only for honest interactions”

**can be checked**

- FO: automatic check of **diff-equivalence** using Proverif
- WA: automatic check of **correspondence prop.** using Proverif

**Tight** enough to conclude on our **case studies:**

- (BAC, LAK, Hash-Lock, EKE, SPKE)
## Future Work

### Improve the method

- tackle memory (often used in RFID)
- move to other tools as backends (Tamarin, Maude-NPA)
- allow more flexibility for honest interactions
Future Work

Improve the method

- tackle memory (often used in RFID)
- move to other tools as backends (Tamarin, Maude-NPA)
- allow more flexibility for honest interactions

\[
\nu \vec{k} ! (\nu \vec{n}_T T \mid \nu \vec{n}_R R) \approx \nu \vec{k} ! (\nu \vec{n}_T^1 T^1 \mid \nu \vec{n}_R^1 R^1) \\
\approx \ldots \\
\approx \nu \vec{k} ! (\nu \vec{n}_T^{\text{ideal}} T^{\text{ideal}} \mid \nu \vec{n}_R^{\text{ideal}} R^{\text{ideal}})
\]

Reusing core ideas

- exploit our conditions to obtain other properties
- investigate transformations-based approaches: interactive, modular, compositional
- extract guidelines from our conditions
Future Work

Improve the method
- tackle memory (often used in RFID)
- move to other tools as backends (Tamarin, Maude-NPA)
- allow more flexibility for honest interactions

\[
! \nu \vec{k} ! (\nu \vec{n}_T T \mid \nu \vec{n}_R R) \approx \nu \vec{k} ! (\nu \vec{n}_T^1 T^1 \mid \nu \vec{n}_R^1 R^1) \\
\approx \ldots \\
\approx ! \nu \vec{k} ! (\nu \vec{n}_T^{\text{ideal}} T^{\text{ideal}} \mid \nu \vec{n}_R^{\text{ideal}} R^{\text{ideal}})
\]

Reusing core ideas
- exploit our conditions to obtain other properties
- investigate transformations-based approaches: interactive, modular, compositional
- extract guidelines from our conditions

Thank you!