Automatic Verification of Privacy Protection for Unbounded Sessions
Shonan

Lucca Hirschi

October 28, 2015

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

we need formal verification of crypto protocols covering privacy

Goal:

- checking privacy (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. [...]
Strong unlinkability [Ryan et al. CSF’10]:

\[
\begin{align*}
\nu \vec{k} \, & \nu \vec{n}(T \mid R) \approx \nu \vec{k}.\nu \vec{n}(T \mid R) \\
\mathcal{M} & \subseteq \mathcal{S}
\end{align*}
\]

Intuition: \(\mathcal{M} \subseteq \mathcal{S}\)

\[\forall \text{ and behaviour of } (\mathcal{M} \parallel \vec{a}) \text{ producing observable } D \Rightarrow \exists \text{ behaviour of } (\mathcal{S} \parallel \vec{a}) \text{ producing observable } D' \sim D\]
Context

Strong unlinkability [Ryan et al. CSF’10]:

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

\[\mathcal{M} \approx \mathcal{S}\]

Intuition: \(\mathcal{M} \subseteq \mathcal{S}\)

\(\forall \text{ and behaviour of } (\mathcal{M} \parallel \text{REF}) \text{ producing observable } \mathcal{D}\)

\(\Rightarrow \exists \text{ behaviour of } (\mathcal{S} \parallel \text{REF}) \text{ producing observable } \mathcal{D'} \sim \mathcal{D}\)

- Checking this is \textbf{undecidable} (because of !)

Existing approaches:

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

Strong unlinkability [Ryan et al. CSF’10]:

\[ \nu k! \nu \overline{n}(T \mid R) \approx ! \nu k. \nu \overline{n}(T \mid R) \]

Intuition: \( \mathcal{M} \sqsubseteq S \)

\( \forall \mathcal{M} \text{ and behaviour of } (\mathcal{M} \parallel \overline{\mathcal{M}}) \text{ producing observable } \mathcal{D} \)
\( \Rightarrow \exists \text{ behaviour of } (S \parallel \overline{S}) \text{ producing observable } \mathcal{D}' \sim \mathcal{D} \)

- Checking this is **undecidable** (because of !)

Existing approaches:

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

\( \Rightarrow \) there is a need for **dedicated** abstraction targeting privacy
Contribution

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
Messing with messages & $C_{\text{data}}$

$C_{\text{data}}$: “Messages are without relations”

Tag
$k, id$

Reader
$k$

$\text{enc}(id, k)$

Goal: messages do not leak info about involved agents

Intuitively: outputs are (statically) indistinguishable from $\neq$ nonces
Messing with messages & $C_{\text{data}}$

$C_{\text{data}}$: “Messages are without relations”

Practical examples (RFID protocols): HB$^+$, DM, KCL, LBV, LD, …
Messing with messages & $C_{\text{data}}$

$C_{\text{data}}$: “Messages are without relations”

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

\[
\text{Tag} \quad k, id \\
\text{Reader} \quad k \\
\text{enc}(id, k) \\
\{\text{enc(ok, } k\}, \text{enc(ok, } k\}) \not\sim \{n_1^t, n_2^t\}
\]
Messing with conditionals & $C_{\text{test}}$

$C_{\text{test}}$: “Conditionals hold only for honest interactions”

Practical examples: BAC (ePassport), some versions of PACE (new version of ePassport), LAK, CH
Messing with conditionals & $C_{test}$

\[ C_{test} : \text{"Conditionals hold only for honest interactions"} \]

- **Goal**: conditionals do not leak info about involved agents
- **Intuitively**: if Tag goes to a Then branch then the attacker just forwarded messages between this Tag and some Reader
A taste of $C_{\text{data}} \& C_{\text{test}}$

<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>
</tbody>
</table>

Theorem: implies

$C_{\text{data}} \Rightarrow C_{\text{test}}$ can be checked

$\triangleright$ $C_{\text{data}}$: automatic check of diff-equivalence using Proverif

$\triangleright$ $C_{\text{test}}$: automatic check of correspondence prop. using Proverif
A taste of $C_{\text{data}} & C_{\text{test}}$

<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

- $C_{\text{data}}$: “Messages are without relations”
- $C_{\text{test}}$: “Conditionals hold only for honest interactions”
A taste of $C_{\text{data}} \& C_{\text{test}}$

<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

<table>
<thead>
<tr>
<th>$C_{\text{data}}$</th>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$C_{\text{test}}$</th>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
A taste of $C_{data} \& C_{test}$

<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>

$\uparrow \text{Theorem: implies} \uparrow$

<table>
<thead>
<tr>
<th>$C_{data}$</th>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td>□</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$C_{test}$</th>
<th>Equivalence?</th>
<th>Active Attacker?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>□</td>
<td>✓</td>
</tr>
</tbody>
</table>

$\uparrow \text{can be checked} \uparrow$

- $C_{data}$: automatic check of \textit{diff-equivalence} using Proverif
- $C_{test}$: automatic check of \textit{correspondence prop.} using Proverif
Applications

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

New proofs of Unlinkability & Anonymity for:

- BAC+PA+AA (ePassport);
- PACE+PA+AA (ePassport v2);
- (fixed) LAK (RFID auth.);
- Hash-Lock (RFID auth.).
Applications

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

**New proofs of Unlinkability & Anonymity for:**

- BAC + PA + AA (ePassport);
- PACE + PA + AA (ePassport v2);
- (fixed) LAK (RFID auth.);
- Hash-Lock (RFID auth.).

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

**Flaws/attacks discovered:**

- some versions of PACE (¬ UK);
- LAK (¬ UK).
... still looking for other case studies ...

Thank You!