Automatic Verification of Privacy Protection for Unbounded Sessions
CSF 5'

Lucca Hirschi

July 16, 2015
we need formal verification of crypto protocols covering privacy
Introduction

~~ we need formal verification of crypto protocols covering privacy

Goal:

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

Goal:
- checking privacy (unlinkability and anonymity)
- in the symbolic model
- for unbounded sessions.

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

\[
\| k \| n(T | R) \approx \| k \cdot n(T | R)
\]
we need formal verification of crypto protocols covering privacy

Goal:

- checking privacy (unlinkability and anonymity)
- in the symbolic model
- for unbounded sessions.

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

\[ \nu \vec{k} \parallel \nu \vec{n} (T \mid R) \approx \nu \vec{k}. \nu \vec{n} (T \mid R) \]

Existing approaches:

- manual: need to exhibit huge bisimulations;
- automatic (ProVerif/Maude-NPA): abstractions yield false attacks.
Contribution

We identify:
- 2 conditions implying unlinkability and anonymity
- for a class of 2-agents protocols including some 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
A taste of $C_{\text{rel}} \& C_{\text{honest}}$

<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>
A taste of $C_{\text{rel}}$ & $C_{\text{honest}}$

<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>

$\uparrow$ implies $\uparrow$

$C_{\text{rel}}$: all outputs are indistinguishable from “nonces”

$C_{\text{honest}}$: test of $A$ holds $\Rightarrow A$ had an honest interaction
### A taste of $C_{\text{rel}}$ & $C_{\text{honest}}$

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

|$\uparrow$ implies $\uparrow$|

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

<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>

⇑ implies ⇑

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

⇑ can be checked ⇑

- $C_{\text{rel}}$: automatic check of **diff-equivalence** using Proverif
- $C_{\text{honest}}$: automatic check of **correspondence prop.** using Proverif
Applications

New proofs of UK & Ano for:

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

New proofs of UK & Ano 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!