Verifying Accountability for Unbounded Sets of Participants

34th IEEE Computer Security Foundations Symposium

Kevin Morio, Robert Künnemann
CISPA Helmholtz Center for Information Security
kevin.morio@cispa.de, robert.kuennemann@cispa.saarland
Trust

Security Protocol 

Accountability

must be justified

Parties

Trust strengthens in

requires
Accountability by Causation

- Accountability notion of Künneleman et. al. (2019)
- Based on sufficient causation

- **Accountability for $\varphi$**
  - Meta property of a protocol
  - Allows identifying all parties causing a violation of $\varphi$

When is a party’s behavior the cause of a violation?
Accountability by Causation

- This is what happened
  - exposed the data  \( \implies \text{verdict} = \{\text{user}\} \)

- How did this happen?
  - misconfigured server  \( \implies \text{verdict} = \{\text{user}, \text{devil}\} \)

- Both causes are required
  - executed exploit

- is leaked
Counterfactual Relation

Actual Situation

leaked jointly by and

Counterfactual Situation

leaked by

But that's not what happened!

Need relation between actual and counterfactual world!
Automated Verification of Accountability

- How can accountability be verified?

\[ \text{verdict}(t) \; \text{provides accountability for} \; \phi \]
Case distinction on different verdicts

\[ \text{verdict}(t) := \begin{cases} V_1 & \text{if } \omega_1(t) \\ \vdots \\ V_n & \text{if } \omega_n(t) \end{cases} \]

Cases are **exhaustive** and **exclusive**

**Problems**

1. Finite number of verdicts
2. Verdicts must be stated explicitly

\[ \Rightarrow \text{Only a bounded number of parties are supported} \]
- **Case tests**: Trace properties with free variables

\[
\tau_1 := \exists data, i. \text{LeakEmployee}(e, data)@i \\
\tau_2 := \exists data, i. \text{LeakAdminHacker}(a, h, data)@i
\]

- **Verdict function**: Union over instantiated case tests

\[
\text{verdict}(t) := \bigcup_{\tau \in \text{tests}} \{ fv(\tau \rho) \mid \exists \rho. t \models \tau \rho \}
\]

- **Example**

\[
\text{verdict}(t) = \{ E_1 \}, \quad \{ A, H \}, \quad \{ E_2 \}
\]
Challenge: Counterfactual Relation

Actual Trace

Counterfactual Traces
Implementation

- SAPIC / MSRs
- Case tests
- Accountability lemmas

translate

- MSRs
- Standard lemmas

Tamarin

- Verification
- Counterexample
- Timeout

test evidence:
"Ex #i. Blame(m)@i"

lemma missing: evidence accounts for
"All sid s ms #i. Send(<sid, s>, ms)i
c=> Ex m #j. Post(<sid, m>, '0', ms)@j"

missing_evidence_suff: verified (16 steps)

missing_evidence_single: ...

missing_evidence_suff: verified (16 steps)

missing_verif_empty: falsified - found trace (16 steps)
Case Studies

- 8 case studies (4 from prior work, 4 new)
- Prior work

### New case studies

<table>
<thead>
<tr>
<th>Our proposal</th>
<th>[21]</th>
</tr>
</thead>
<tbody>
<tr>
<td>WhoDunit (fixed)</td>
<td>✓ 7 52 s</td>
</tr>
<tr>
<td>Certificate Transparency (extended)</td>
<td>✓ 27 17 s</td>
</tr>
<tr>
<td>OCSP Stapling (trusted resp.)</td>
<td>✓ 7 18 s</td>
</tr>
<tr>
<td>OCSP Stapling (untrusted resp.)</td>
<td>✗ 7 18 s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1 role</th>
<th>2 roles</th>
<th>3 roles</th>
<th>4 roles</th>
<th>5 roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic DMN (duplicate ciphertexts)</td>
<td>—</td>
<td>—</td>
<td>✓ 13 26 s</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>DMN + message tracing (first)</td>
<td>✓ 7 8 s</td>
<td>✓ 7 124 s</td>
<td>✓ 7 137 s</td>
<td>✓ 7 14178 s</td>
<td>✓ 7 134160 s</td>
</tr>
<tr>
<td>DMN + message tracing (all)</td>
<td>✓ 7 6 s</td>
<td>✗ 7 12 s</td>
<td>✗ 7 22 s</td>
<td>✗ 7 100 s</td>
<td>✗ 7 355 s</td>
</tr>
<tr>
<td>MixVote (unbounded)</td>
<td>✓ 14 6 s</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1 party</th>
<th>2 parties</th>
<th>3 parties</th>
<th>4 parties</th>
<th>5 parties</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMN + message tracing (first)</td>
<td>✓ 7 7 s</td>
<td>✓ 17 133 s</td>
<td>✓ 46 2146 s</td>
<td>✓ 149 23827 s</td>
<td>—* 544</td>
</tr>
<tr>
<td>DMN + message tracing (all)</td>
<td>✓ 7 4 s</td>
<td>✗ 17 23 s</td>
<td>✗ 46 115 s</td>
<td>✗ 149 548 s</td>
<td>✗ 544</td>
</tr>
<tr>
<td>MixVote (unbounded)**</td>
<td>✓ 14 5 s</td>
<td>✓ 34 58 s</td>
<td>✓ 92 2721 s</td>
<td>—* 298</td>
<td>—* 1112</td>
</tr>
</tbody>
</table>

* No verification results due to memory exhaustion. ** Each party acts in the same role, that of the server.
Conclusion

- Automated verification of accountability supporting an unbounded number of participants
  - Necessary for analyzing real-world protocols
- Case tests as the key concept
  - Flexible definition of verdict functions
  - Improved readability

- Implemented in Tamarin (github.com/kevinmorio/tamarin-prover)

- Up to 5x faster than the previous approach
- Less effort to formulate accountability lemmas