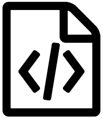
How to Verify Privacy Automatically

Laouen Fernet

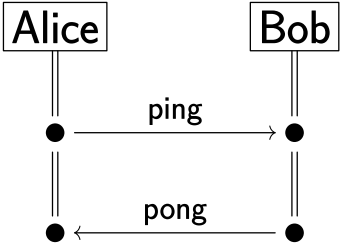
DTU Compute
lpkf@dtu.dk

November 13, 2023

Control of privacy



Automated verification



Security protocols

Take-aways

1. We should formally verify privacy in many applications

Electronic voting

Contact tracing

Mobile payments, transport, e-passports...

Take-aways

2. We can define privacy goals in a declarative and intuitive way with logic

(α, β)-privacy

Take-aways

3. Automated verification of (α, β)-privacy is practical

Decision procedure and prototype tool [1]

At the bar

Challenge 25 by the bartender: private information disclosed

The Bourne Identity, 2002

At the bar

Challenge 25 by the bartender: private information disclosed

The Bourne Identity, 2002

Protocol specification

Tag	Reader
$\star\ T \in Tags.$	$\mathsf{rcv}(x).$
$K := r(T).$	try $T = \mathsf{getT}(x)$ in
$r(T) := h(K).$	$\quad s := \mathsf{state}(T).$
$\mathsf{snd}(g(K)).0$	$\quad$ try $s' = \mathsf{extract}(x, s)$ in
	$\quad\quad \mathsf{state}(T) := h(s').$
	$\quad\quad \mathsf{snd}(\mathsf{ok}).0$

- Processes with atomic transactions
- Intruder controlling the network
- Crypto API

(α, β)-privacy

- Formula α = payload, over alphabet $\Sigma_0 \subset \Sigma$
- Formula β = technical information, over alphabet Σ
- Violation of privacy = β excludes some models of α

Example: unlinkability for two sessions

$\alpha \equiv T_1, T_2 \in$ Tags   Question: $T_1 \stackrel{?}{=} T_2$

Example: voting

- $\alpha \equiv v_1, v_2, v_3, v_4 \in \{0, 1\} \wedge v_1 + v_2 + v_3 + v_4 = 2$
- β includes α and encrypted ballots etc.

Example: voting

v_1 v_2 v_3 v_4

- $\alpha \equiv v_1, v_2, v_3, v_4 \in \{0, 1\} \land v_1 + v_2 + v_3 + v_4 = 2$
- β includes α and encrypted ballots etc.
- If $\beta \Rightarrow v_1 = v_4 \land v_2 = v_3$: privacy violation

Protocol excerpt

```
...
* x in Agent.
* y in {yes, no}. # Flip a coin
receive M.
try N := dcrypt(inv(pk(s)),M) in
if y = yes then
  new R. send crypt(pk(x),pair(yes,N),R)
else
  new R. send crypt(pk(x),no,R)
...
```

Multi message-analysis problem

Several possibilities

$$(y = \mathsf{yes}, [\ldots, l \mapsto \{\mathsf{yes}, N\}_{\mathsf{pk}(x)}])$$
$$(y = \mathsf{no}, [\ldots, l \mapsto \{\mathsf{no}\}_{\mathsf{pk}(x)}])$$

The concrete execution corresponds to one of them

One multi message-analysis problem = one pair (α, β)

Automated verification

Challenges:

- Undecidable problem
- Infinite state space
- Proofs!

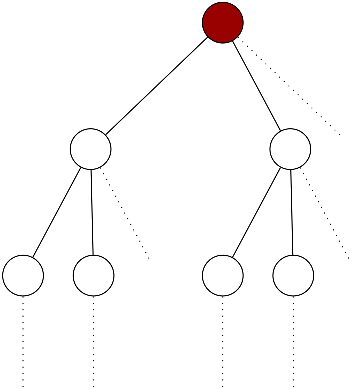
Methods:

- Restriction to a (large) class of protocols + bound
- Constraints solving with abstractions

Idea of the procedure

- We verify privacy as a *reachability property* in a transition system
- We use the *lazy intruder* (finite branching when the intruder is sending messages)
- In each state, we represent several executions
- We normalize states so that, in each state, all possibilities are “equivalent”

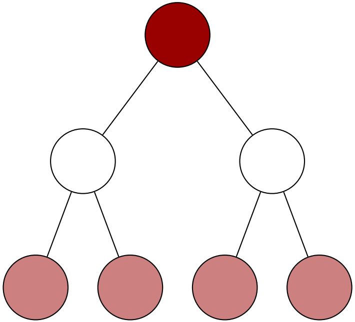
Symbolic representation



(α, β) in every state

Abstraction

$\longrightarrow$



Several (α_i, β_i) in every symbolic state

Recent work

- Paper on decision procedure with correctness and termination proofs
- Prototype tool implemented in Haskell
- Models and case study for several existing protocols
- Paper on typing result for guaranteeing well-typed attacks

Current and future work

- Compositionality result: how to combine protocols securely?
- Development of the tool (user-friendliness)
- Support for a larger class of protocols

Take-aways

1. We should formally verify privacy in many applications
2. We can define privacy goals in a declarative and intuitive way with logic, using (α, β)-privacy
3. Automated verification of (α, β)-privacy is practical

References

L. Fernet, S. Mödersheim, and L. Viganò.
A decision procedure for alpha-beta privacy for a bounded number of transitions.
In *CSF 2024 (to appear)*. IEEE, 2024.
Extended version at `https://people.compute.dtu.dk/lpkf`.

S. Mödersheim and L. Viganò.
Alpha-beta privacy.
ACM Trans. Priv. Secur., 22(1):1–35, 2019.

S. Gondron, S. Mödersheim, and L. Viganò.
Privacy as reachability.
In *CSF 2022*. IEEE, 2022.