Engineering Secure Software Systems

December 8, 2020: The Rusinowitch-Turuani Theorem: Proof and Limitations

Henning Schnoor

Institut für Informatik, Christian-Albrechts-Universität zu Kiel

Part I: Crypto Protocols

Overview

Part I: Crypto Protocols

- Foundations
- Cryptography
- An Example and an Attack

More Examples

Formal Protocol Model

Automatic Analysis: Theoretical Foundations

- Short Attacks
- NP hardness

Overview

Part I: Crypto Protocols

Foundations

Cryptography

An Example and an Attack

More Examples

Formal Protocol Model

Automatic Analysis: Theoretical Foundations

Short Attacks

NP hardness

Size of a Minimal Attack

theorem

(σ, o) minimal successful attack on P, then $|\sigma(x)|_{\mathsf{DAG}} \leq |\{r_0, \ldots, r_n, s_0, \ldots, s_n\}|_{\mathsf{DAG}}$.

completes NP membership proof

- each x: representation of $\sigma(x)$ bound by protocol length
- number of variables is polynomial

therefore: polynomial representation of attack

proof (sketch)

- for every x with $\sigma(x) > 1$ ex. subterm t of P with $\sigma(t) = \sigma(x)$, t no variable
- every "long" $\sigma(x)$ is obtained by applying terms from the protocol
- replace long terms with references to protocol

$\rightsquigarrow$ small DAG size

Proof of Rusinowitch Turuani Theorem

theorem

(σ, o) minimal successful attack on P, then $|\sigma(x)|_{\mathrm{DAG}} \leq |\{r_0, \ldots, r_n, s_0, \ldots, s_n\}|_{\mathrm{DAG}}$.

proof

U set of variables, then $\overline{U} = \{\sigma(x) \mid x \in U\}$.

- construct $S_i \subseteq \mathrm{Sub}(\{r_0, \ldots, s_n\})$, $V_i \subseteq \mathcal{V}$:

$$|\sigma(x)|_{\mathrm{DAG}} \leq \left|S_i \cup \overline{V_i}\right|_{\mathrm{DAG}}$$

- 1. $i = 0$: choose $S_0 = \emptyset$, $V_0 = \{x\}$.
  2. $i \to i + 1$
     - choose $y \in V_i$, t from protocol with $\sigma(y) = \sigma(t)$
     - $S_{i+1} = S_p \cup \{t\}$
     - $V_{i+1} = V_p \setminus \{y\} \cup \mathcal{V}(t)$
     - ind: $|\sigma(x)|_{\mathrm{DAG}} \leq \left|S_i \cup \overline{V_i}\right|_{\mathrm{DAG}}$
     - now: $|\sigma(x)|_{\mathrm{DAG}} \leq \left|S_i \cup \overline{V_i}\right|_{\mathrm{DAG}} \leq \left|S_{i+1} \cup \overline{V_{i+1}}\right|_{\mathrm{DAG}}$

usage

- S_i: references into protocol
- V_i: TODO-list: variables to be replaced by references

start/end, termination

- $i = 0$: $|\sigma(x)|_{\mathrm{DAG}} \leq \left|\overline{\{x\}}\right|_{\mathrm{DAG}}$
- $V_i = \emptyset$: $|x|_{\mathrm{DAG}} \leq |S_i|_{\mathrm{DAG}} \leq |r_0, \ldots, s_n|_{\mathrm{DAG}}$
- why do we reach $V_i = \emptyset$?
  - $\sum_{z \in V_i} |\sigma(z)|$ decreases

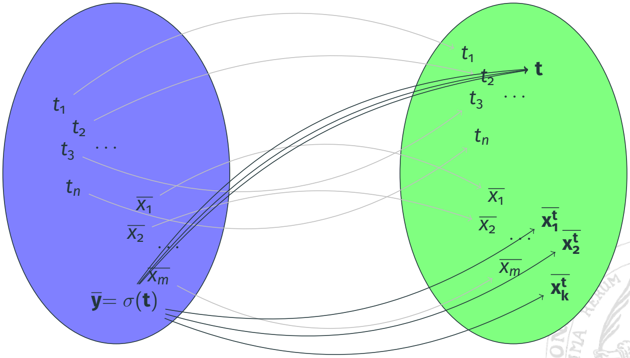
Proof of Rusinowitch-Turuani Theorem

claim

$\left|S_i \cup \overline{V_i}\right|_{\mathrm{DAG}} \leq \left|S_{i+1} \cup \overline{V_{i+1}}\right|_{\mathrm{DAG}}$ via injection $f\colon \mathbf{Sub}\left(\mathsf{S_i} \cup \overline{\mathsf{V_i}}\right) \to \mathbf{Sub}\left(\mathsf{S_{i+1}} \cup \overline{\mathsf{V_{i+1}}}\right)$

induction

- $S_{i+1} = S_i \cup \{t\}$
- $V_{i+1} = V_i \cup \mathcal{V}(t) \setminus \{y\}$



Exercise

Task (applying the Rusinowitch Turuani Theorem)

In the lecture, we discussed how to model the Needham-Schroeder protocol formally as an input to INSECURE such that the attack can be detected. However, this modeling required us to already specify the “correct” sessions (“Alice with Charlie, Charlie with Bob”) as the input. For a complete automatic analysis, such a manual step should not be necessary. Can you come up with a general mechanism translating a natural representation of a protocol (for example, as the list of “intended instances” for a single session) into an instance that can be used as input for INSECURE? If not, why not?

Overview

Part I: Crypto Protocols

Foundations

Cryptography

An Example and an Attack

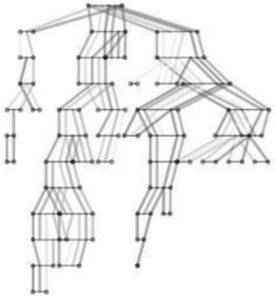
More Examples

Formal Protocol Model

Automatic Analysis: Theoretical Foundations

Short Attacks

NP hardness



theorem

INSECURE is NP-complete.

two parts

1. INSECURE $\in$ NP
2. INSECURE is NP-hard

second part: INSECURE is NP-hard

recall TGI: reduce from NP-complete problem

NP hardness proof

TGI refresher

L_1, L_2 languages, L_1 NP-hard and $L_1 \leq_m^p L_2$, then L_2 NP-hard.

reduction

$L_1 \leq_m^p L_2$ means:

L_1-question can be "efficiently translated" into L_2-questions

formally

there is a total, P-computable function $f : \Sigma^* \to \Sigma^*$ with

$$x \in L_1 \textit{ iff } f(x) \in L_2 \textit{ for all } x.$$

NP-complete problem: 3SAT

SAT for formulas $\varphi = \bigwedge_{i=1}^{n} (l_1^i \vee l_2^i \vee l_3^i)$, where l_j^i literals over $\{x_1, \dots, x_m\}$

NP hardness reduction I

3SAT

- **nondeterministic step**: guess assignment I for variables of formula φ
- **deterministic step**: check that assignment satisfies φ

INSECURE

- **nondeterministic step**: guess *one* adversary message (encoding I)
- **deterministic step**: let honest participants check that assignment satisfies φ

note

φ can be hard-coded into INSECURE instance

issues

- adversary can interfere with communication between honest principals
- use cryptography to ensure secure communication

NP hardness reduction II

crypto

needed for proof?

3SAT instance

$\varphi = \bigwedge_{i=1}^{n}(l_1^i \vee l_2^i \vee l_3^i)$, each l_j^i literal over $\{x_1, \ldots, x_m\}$, $l_j^i \in \{x_{r_{i,j}}, \overline{x_{r_{i,j}}}\}$

protocol instances

- A expects and distributes $\{x_1, \ldots, x_m\}$-assignment I:
  $$[x_1, x_2, \ldots, x_m] \to [m_1, \ldots, m_n], \text{ with } m_i = \mathrm{enc}_k^s([i, x_{r_{i,1}}, x_{r_{i,2}}, x_{r_{i,3}}])$$
- for $i \in \{1, \ldots, n\}$, satisfying assignment (α, β, γ) of clause i: instance $B^i_{(\alpha,\beta,\gamma)}$
  $$\mathrm{enc}_k^s([i, \alpha, \beta, \gamma]) \to k_i$$
- final assembly F: if all clauses satisfied, release FAIL
  $$[k_1, \ldots, k_n] \to \mathrm{FAIL}$$

correctness

FAIL $\leftrightarrow$ adv gets all k_i $\leftrightarrow$ for each i, one $B^i_{(\alpha,\beta,\gamma)}$ releases k_i

$\leftrightarrow$ $(\alpha, \beta, \gamma) \models$clause $\leftrightarrow$ all clauses satisfied by I $\leftrightarrow$ φ satisfiable