Protocol analysis using ProVerif

ProVerif

- http://www.proverif.ens.fr
- Static analysis for cryptographic protocols under the perfect cryptography assumption
- Can check secrecy and correspondence properties
- Errs only to the safe side
  - If a protocol is insecure, then says so
  - If a protocol is secure, then sometimes may claim to have found an attack
- Principle: translate the protocol to a set of Horn clauses
  - Involves a little bit of abstraction
- There is an attack $\Rightarrow$ this set is satisfiable

Horn clauses

$$p_1(t_{11},\ldots,t_{1k_1}) \wedge \cdots \wedge p_n(t_{n1},\ldots,t_{nk_n}) \Rightarrow q(t'_1,\ldots,t'_m)$$

- $p_1,\ldots,p_n,q$ — predicate symbols
  - from a fixed set; each with fixed arity
- t_*, t'_* — terms
  - countable number of atoms
  - constructors from a fixed set
- terms may contain term variables as subterms
- $\bigwedge_i p_i(\ldots X \ldots) \Rightarrow q(\ldots X \ldots)$ means
  $\forall t \in \mathbf{Term} : \left(\bigwedge_i p_i(\ldots t \ldots) \Rightarrow q(\ldots t \ldots)\right)$
  - $\mathbf{Term}$ — the set of all ground terms (without variables)

Examples

- A translation of a protocol always contains a unary predicate a
  - $\mathrm{a}(t)$ means that the attacker can learn t
- A translation contains rules for composing and decomposing messages:
  - $\mathrm{a}(pair(X, Y)) \Rightarrow \mathrm{a}(X) \qquad \mathrm{a}(pair(X, Y)) \Rightarrow \mathrm{a}(Y)$
  - $\mathrm{a}(X) \wedge \mathrm{a}(Y) \Rightarrow \mathrm{a}(pair(X, Y))$
  - $\mathrm{a}(senc(K, X)) \wedge \mathrm{a}(K) \Rightarrow \mathrm{a}(X)$
  - $\mathrm{a}(penc(pk(K), X)) \wedge \mathrm{a}(K) \Rightarrow \mathrm{a}(X)$
  - $\mathrm{a}(K) \wedge \mathrm{a}(X) \Rightarrow \mathrm{a}(sign(K, X))$
  - $\mathrm{a}(sign(K, X)) \Rightarrow \mathrm{a}(X)$
  - $\mathrm{a}(X) \Rightarrow \mathrm{a}(h(X))$

Recall our example

$$A \longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B}$$
$$B \longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A}$$
$$A \longrightarrow B : \{[N_A, N_B]\}_{K_B}$$
$$B \longrightarrow A : \{M\}_{K_{AB}}$$

- The attacker can have the first message by starting a new session

$$\mathrm{a}(pk(A)) \wedge \mathrm{a}(pk(B)) \Rightarrow \mathrm{a}(penc(pk(B), triple(pk(A), n, k)))$$

Recall our example

$$A \longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B}$$
$$B \longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A}$$
$$A \longrightarrow B : \{[N_A, N_B]\}_{K_B}$$
$$B \longrightarrow A : \{M\}_{K_{AB}}$$

- The attacker can have the first message by starting a new session

$$\mathrm{a}(pk(A)) \wedge \mathrm{a}(pk(B)) \Rightarrow \mathrm{a}(penc(pk(B), triple(pk(A), n, k)))$$

  Something is very wrong here... What n? What k?
- n and k would be different in each session. There must be a parameter "session ID".

The first message

$$A \longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B}$$
$$B \longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A}$$
$$A \longrightarrow B : \{[N_A, N_B]\}_{K_B}$$
$$B \longrightarrow A : \{M\}_{K_{AB}}$$

- The attacker can have the first message by starting a new session

$$\mathrm{a}(pk(A)) \wedge \mathrm{a}(pk(B)) \wedge \mathrm{a}(Id) \Rightarrow \mathrm{a}(penc(pk(B), triple(pk(A), n[Id], k[Id])))$$

The first message

$$
\begin{aligned}
A \longrightarrow B &: \{[K_A, N_A, K_{AB}]\}_{K_B} \\
B \longrightarrow A &: \{[N_A, N_B, K_B]\}_{K_A} \\
A \longrightarrow B &: \{[N_A, N_B]\}_{K_B} \\
B \longrightarrow A &: \{M\}_{K_{AB}}
\end{aligned}
$$

- The attacker can have the first message by starting a new session

$$
\mathrm{a}(pk(A)) \wedge \mathrm{a}(pk(B)) \wedge \mathrm{a}(Id) \Rightarrow \mathrm{a}(penc(pk(B), triple(pk(A), n[Id], k[Id])))
$$

- Attacker: "Dear Alice, please start session 5 with Bob"
  - $k(5)$ will be exchanged
- Attacker "Dear Alice, please start session 5 with me"
  - Attacker learns $k(5)$

The first message

$$
\begin{aligned}
A \longrightarrow B &: \{[K_A, N_A, K_{AB}]\}_{K_B} \\
B \longrightarrow A &: \{[N_A, N_B, K_B]\}_{K_A} \\
A \longrightarrow B &: \{[N_A, N_B]\}_{K_B} \\
B \longrightarrow A &: \{M\}_{K_{AB}}
\end{aligned}
$$

- Session ID must contain the roles of the parties.

$$
\begin{aligned}
\mathrm{a}(pk(A)) \wedge \mathrm{a}(pk(B)) \wedge \mathrm{a}(Id) \Rightarrow \\
\mathrm{a}(penc(pk(B), triple(pk(A), \\
n[pk(A), pk(B), Id], k[pk(A), pk(B), Id])))
\end{aligned}
$$

The second message

$$
\begin{aligned}
A &\longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B} \\
B &\longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A} \\
A &\longrightarrow B : \{[N_A, N_B]\}_{K_B} \\
B &\longrightarrow A : \{M\}_{K_{AB}}
\end{aligned}
$$

- When Bob gets the first message, he responds with the second

$$
\begin{aligned}
&\mathrm{a}(Id) \wedge \mathrm{a}(penc(pk(B), triple(pk(A), N, K))) \Rightarrow \\
&\qquad \mathrm{a}(penc(pk(A), triple(N, n'[pk(A), pk(B), Id], pk(B))))
\end{aligned}
$$

The third message

$$A \longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B}$$
$$B \longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A}$$
$$A \longrightarrow B : \{[N_A, N_B]\}_{K_B}$$
$$B \longrightarrow A : \{M\}_{K_{AB}}$$

- When Alice gets the second message, she responds with the third

$$\mathrm{a}(penc(pk(A), triple(n[pk(A), pk(B), Id], N', pk(B)))) \Rightarrow$$
$$\mathrm{a}(penc(pk(B), pair(n[pk(A), pk(B), Id], N')))$$

The fourth message

$$
\begin{aligned}
A \longrightarrow B &: \{[K_A, N_A, K_{AB}]\}_{K_B} \\
B \longrightarrow A &: \{[N_A, N_B, K_B]\}_{K_A} \\
A \longrightarrow B &: \{[N_A, N_B]\}_{K_B} \\
B \longrightarrow A &: \{M\}_{K_{AB}}
\end{aligned}
$$

- When Bob gets the third message, he responds with the fourth...
- But only if he has participated in the session from the beginning

The fourth message

$$
\begin{aligned}
A &\longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B} \\
B &\longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A} \\
A &\longrightarrow B : \{[N_A, N_B]\}_{K_B} \\
B &\longrightarrow A : \{M\}_{K_{AB}}
\end{aligned}
$$

- When Bob gets the third message, he responds with the fourth...
- But only if he has participated in the session from the beginning
- When Bob has received the first and third messages, he can respond with the fourth.

$$
\begin{aligned}
&\mathrm{a}(penc(pk(B), triple(pk(A), N, K))) \wedge \\
&\qquad \mathrm{a}(penc(pk(B), pair(N, n'[pk(A), pk(B), Id]))) \Rightarrow \\
&\qquad\qquad \mathrm{a}(senc(K, m))
\end{aligned}
$$

The fourth message

$$A \longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B}$$
$$B \longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A}$$
$$A \longrightarrow B : \{[N_A, N_B]\}_{K_B}$$
$$B \longrightarrow A : \{M\}_{K_{AB}}$$

- When Bob gets the third message, he responds with the fourth...
- But only if he has participated in the session from the beginning
- When Bob has received the first and third messages, he can respond with the fourth.

$$\mathrm{a}(penc(pk(B), triple(pk(A), N, K))) \wedge \mathrm{a}(penc(pk(B), pair(N, n'[pk(A), pk(B), Id]))) \Rightarrow \mathrm{a}(senc(K, m))$$

What is wrong here?

The fourth message

$$A \longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B}$$
$$B \longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A}$$
$$A \longrightarrow B : \{[N_A, N_B]\}_{K_B}$$
$$B \longrightarrow A : \{M\}_{K_{AB}}$$

Only Bob will send M, and only to Alice.

$$\mathrm{a}(penc(pk(sB), triple(pk(sA), N, K))) \wedge$$
$$\mathrm{a}(penc(pk(sB), pair(N, n'[pk(sA), pk(sB), Id]))) \Rightarrow$$
$$\mathrm{a}(senc(K, m))$$

Solving the system

- Is $\mathrm{a}(m)$ derivable?
- You may ask a Prolog system. And it will answer...

Solving the system

- Is $\mathrm{a}(m)$ derivable?
- You may ask a Prolog system. And it will answer...
- ...infinite loop.
  - To get $\mathrm{a}(m)$, we could use some $\mathrm{a}(f(m))$
  - To get $\mathrm{a}(f(m))$, we could use some $\mathrm{a}(f(f(m)))$
  - To get...
- The unification strategy of ProVerif is more geared towards such protocol representations.

Try to run ProVerif

- Demo

Try to run ProVerif

- Demo
- Try to reconstruct the attack

What went wrong

- Alice sent the first message to Bob
- Bob received it twice, responding to it both times
    - Fair enough

What went wrong

- Alice sent the first message to Bob
- Bob received it twice, responding to it both times
  - Fair enough
- But the adversary repeated the session identifier
  - Not good
  - To avoid that, newly generated values must contain all received messages so far.

The second message

$$A \longrightarrow B : \{[K_A, N_A, K_{AB}]\}_{K_B}$$
$$B \longrightarrow A : \{[N_A, N_B, K_B]\}_{K_A}$$
$$A \longrightarrow B : \{[N_A, N_B]\}_{K_B}$$
$$B \longrightarrow A : \{M\}_{K_{AB}}$$

- When Bob gets the first message, he responds with the second

$$\mathrm{a}(Id) \wedge \mathrm{a}(penc(pk(B), triple(pk(A), N, K))) \Rightarrow \mathrm{a}(penc(pk(A), triple(N, n'[pk(A), pk(B), Id, penc(pk(B), triple(pk(A), N, K))], pk(B))))$$

The fourth message

$$
\begin{aligned}
A \longrightarrow B &: \{[K_A, N_A, K_{AB}]\}_{K_B} \\
B \longrightarrow A &: \{[N_A, N_B, K_B]\}_{K_A} \\
A \longrightarrow B &: \{[N_A, N_B]\}_{K_B} \\
B \longrightarrow A &: \{M\}_{K_{AB}}
\end{aligned}
$$

$$
\mathrm{a}(penc(pk(sB), triple(pk(sA), N, K))) \wedge \mathrm{a}(penc(pk(sB), pair(N, n'[pk(sA), pk(sB), Id, penc(pk(sB), triple(pk(sA), N, K))]))) \Rightarrow \mathrm{a}(senc(K, m))
$$

Try to run ProVerif

- Demo

Try to run ProVerif

- Demo
- A similar-looking attack...

Try to run ProVerif

- Demo
- A similar-looking attack. . .
- We actually have a type flaw! Let us correct it.

Try to run ProVerif

- Demo
- A similar-looking attack. . .
- We actually have a type flaw! Let us correct it.
- OK

Correspondence assertions

- Two more predicates, b and e, for $\mathbf{begin}$ and $\mathbf{end}$.
- After a party has executed $\mathbf{begin}(M)$, its following messages are translated with $b(M)$ as a premise.
  - ... contains session IDs and received messages.
- Emitting $\mathbf{end}(M)$ is adversary's goal, hence it is the conclusion of a rule.
  - $\mathrm{a}(m_1) \wedge \cdots \mathrm{a}(m_k) \Rightarrow e(m)$
- If $b(X)$ is necessary for $e(X)$, then we have (non-injective) agreement.

ISO 3-pass mutual authentication

Draft:

$$\begin{aligned}
&1.\ A \longrightarrow B : N_{A1}\\
&2.\ B \longrightarrow A : [\![\{N_{A1}, N_B, K_A\}]\!]_{K_B}\\
&3.\ A \longrightarrow B : [\![\{N_B, N_{A2}, K_B\}]\!]_{K_A}
\end{aligned}$$

Final:

$$\begin{aligned}
&1.\ A \longrightarrow B : {\color{blue}N_A}\\
&2.\ B \longrightarrow A : [\![\{{\color{blue}N_A}, N_B, K_A\}]\!]_{K_B}\\
&3.\ A \longrightarrow B : [\![\{N_B, {\color{blue}N_A}, K_B\}]\!]_{K_A}
\end{aligned}$$

- From signature find the message.
- Public key $\equiv$ principal's name.
- $\mathbf{end}(K_A, K_B)$ executed by B in the very end.
- $\mathbf{begin}(K_A, K_B)$ executed by A before 3rd message.

Injective agreement

- Add the session identifier to the argument of e.
- Add the session identifiers and received messages to the argument of b.
- If $b((X, II))$ is necessary for $e((X, I))$, and I appears in II, then we have injective agreement.
- Example:

$$\begin{array}{l} 1.\ A \longrightarrow B : (A, B) \\ 2.\ B \longrightarrow A : [\{N\}]_{K_B} \end{array}$$

has agreement, which is not injective. Indeed, A's signature verification fails, if B has never signed anything.