ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

S. Mödersheim
Ch. Sprenger
AS 2009

Formal Methods for Information Security

Exercise Sheet 3

Hand-in date: Oct 19, 2009

Assignment 3.1: Station-to-station protocol

The Station-to-station (STS) protocol is specified as follows:

$$\begin{array}{ll} A \to B: & \exp(g, X) \\ B \to A: & \exp(g, Y), \{|\{\exp(g, Y), \exp(g, X), A\}_{\mathsf{inv}(\mathsf{pk}(B))}|\}_{K_{AB}} \\ A \to B: & \{|\{\exp(g, X), \exp(g, Y), B\}_{\mathsf{inv}(\mathsf{pk}(A))}|\}_{K_{AB}} \end{array}$$

where $K_{AB} = \exp(\exp(g, X), Y)$ is the derived shared key. We want to verify that the term $\exp(\exp(g, X), Y)$ is a secret shared by A and B. In OFMC (latest verion "2009b"), there are two ways to specify that a message M is secret between A and B:

- In "classical" notation: `M secret between A, B` from both A and B's point of view, and
- In channel notation: `A ->* B : M` (from A's point of view) and `B ->* A : M` (from B's point of view).

However, there is a subtle difference between these two specifications. Whereas in the first case, the secrecy claim (signal) is placed at the latest possible point in the given role, in the second case, it is placed in the earliest possible position in the role.

(a) Formalize the STS protocol as an Alice&Bob specification in OFMC.

(b) Verify the protocol with each of the two secrecy claims and explain the difference.

Solution

(a) Here is the formalization of the STS protocol:

```
Protocol: STS       # Station-to-station protocol

Types:
  Agent A, B;
```

```
  Number g, X, Y, Msg;
  Function pk

Knowledge:
  A: A, B, g, pk, inv(pk(A));
  B: A, B, g, pk, inv(pk(B))

Actions:
  A -> B: exp(g, X)
  B -> A: exp(g, Y),
          {|{exp(g, Y), exp(g, X), A}inv(pk(B))|}exp(exp(g,X),Y)
  A -> B: {|{exp(g, X), exp(g, Y), B}inv(pk(A))|}exp(exp(g,X),Y)

Goals:
  exp(exp(g,X),Y) secret between A, B
  A ->* B: exp(exp(g,X),Y)
  B ->* A: exp(exp(g,X),Y)
```

Note that we assume here that each role knows its peer before the protocol starts.

(b) In the first case, secrecy holds (verified for $N = 3$ sessions).

In second case, the claim `B ->* A: exp(exp(g,X),Y)` fails, i.e., from B's point of view the derived shared key is not secret. Here is the attack found by OFMC (with some renaming for better readability):

```
ATTACK TRACE
i -> (bob,1): g
(bob,1) -> i: exp(g,Y(1)).
              {{exp(g,Y(1)).g.alice}_inv(pk(bob))}_(exp(g,Y(1)))
i -> (i,17): exp(g,Y(1))
i -> (i,17): exp(g,Y(1))
```

In this attack, the intruder sends `g` in the first message. As a consequence, the public message `exp(g,Y(1))` contributed by Bob in the second message is at the same time the derived shared key and therefore known to the intruder.

The problem here is that the secrecy of the shared key, in this case `exp(g,Y(1))`, is claimed at the earliest possible point where it becomes constructible: after the reception of the first message. At this point, Bob has not yet verified that `g` is the half-key of `alice`. She has indeed never meant to use `g` as her half key and will never authenticate it; also, the intruder is not able to forge her signature. Thus, `bob` is stuck with the protocol execution at this point and can never finish the protocol run.

Since `bob` cannot finish the protocol run, he will never use the key `exp(g,Y(1))` to encrypt any data, so one may argue that there is nothing wrong with this partial protocol run where the intruder found out a secret that is not yet confirmed—because it never will be. This example shows that it can matter where we put a secrecy signal/claim and

in fact, one should in general use the first notion of secrecy that only "fires" when an agent has successfully finished its protocol run.

Assignment 3.2: Semantics of Alice&Bob notation

The following protocol is an excerpt from a contract signing protocol by Asokan, Shoup, and Waidner:[1]

$$\begin{array}{lll} A \to B: & \{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))} & (= m_1) \\ B \to A: & \{\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}, h(N_B)\}_{\mathsf{inv}(\mathsf{pk}(B))} & (= m_2) \\ A \to B: & N_A & \\ B \to A: & N_B & \end{array}$$

where $text$ is the text of the contract that is being signed and h is a public hash function. In the first round, the agents sign and countersign the contractual text and exchange their *public commitments* $h(N_A)$ and $h(N_B)$. In the second round, they exchange the corresponding secret commitments N_A and N_B. The main goal of the protocol is that at the end of the protocol run each participant is in possession of a *valid contract* of the form m_1, m_2, N_A, N_B.

Initially, role A knows A, B, the hash function h, pk and its own signing key, $\mathsf{inv}(\mathsf{pk}(A))$. The situation for B is analogous. For our purposes, we can consider $text$ as a nonce generated by A. Roles A and B also generate their respective secret commitments N_A and N_B.

(a) Translate the role A from Alice&Bob notation to a role script.

(b) Can you reformulate the result using pattern matching? Justify your answer.

(c) What is the initial knowledge of the intruder if we consider $Agent = \{a, b, i\}$?

Solution

(a) The set of labeled messages containing the initial knowledge of role A and the messages she created is:

$$M = \{A^{\mathcal{X}_1}, B^{\mathcal{X}_2}, h^{\mathcal{X}_3}, \mathsf{pk}^{\mathcal{X}_4}, \mathsf{inv}(\mathsf{pk}(A))^{\mathcal{X}_5}, text^{\mathcal{X}_6}, N_A^{\mathcal{X}_7}\}$$

A's AnB role is:

$$\begin{array}{rl} r = & \mathsf{snd}(\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}) \cdot \\ & \mathsf{rcv}(\{\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}, h(N_B)\}_{\mathsf{inv}(\mathsf{pk}(B))}) \cdot \\ & \mathsf{snd}(N_A) \cdot \\ & \mathsf{rcv}(N_B) \end{array}$$

Let us now translate this AnB role into an executable role.

[1] The full specification and the actual goal of this protocol are beyond the limitations of AnB notation. We consider only this excerpt as an exercise.

- First event $\mathsf{snd}(\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))})$: We check that

$$m_1{}^{\{\mathcal{X}_1,\mathcal{X}_2,\mathcal{X}_6,\mathcal{X}_3(\mathcal{X}_7)\}_{\mathcal{X}_5}} \in \mathcal{DY}_L(M)$$

  so the resulting send event is

$$\mathsf{snd}(\{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_6, \mathcal{X}_3(\mathcal{X}_7)\}_{\mathcal{X}_5}).$$

- Second event $\mathsf{rcv}(\{\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}, h(N_B)\}_{\mathsf{inv}(\mathsf{pk}(B))})$: We define $M' = M \cup \{m_2^{\mathcal{X}_8}\}$ and note that there are two ways to derive the message m_1:

$$m_1^{\pi_1(\mathsf{open}(\mathcal{X}_8))}, m_1^{\{\mathcal{X}_1,\mathcal{X}_2,\mathcal{X}_6,\mathcal{X}_3(\mathcal{X}_7)\}_{\mathcal{X}_5}} \in \mathcal{DY}_L(M')$$

  Similarly, there are two ways to derive true:

$$\mathsf{true}^{\mathsf{verify}(\mathcal{X}_4(\mathcal{X}_2),\mathcal{X}_8)}, \mathsf{true}^{\mathsf{true}} \in \mathcal{DY}_L(M')$$

  Similar observation holds for the submessages $A, B, text, h(N_A)$ of m_1, but the related checks are redundant in this case. Therefore, the resulting receive event is

$$\mathsf{rcv}(\mathcal{X}_8 \mid \pi_1(\mathsf{open}(\mathcal{X}_8)) \approx \{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_6, \mathcal{X}_3(\mathcal{X}_7)\}_{\mathcal{X}_5}, \mathsf{verify}(\mathcal{X}_4(\mathcal{X}_2), \mathcal{X}_8) \approx \mathsf{true})$$

- Third event $\mathsf{snd}(N_A)$: This is translated to $\mathsf{snd}(\mathcal{X}_7)$, since $N_A^{\mathcal{X}_7} \in \mathcal{DY}_L(M')$.
- Fourth event $\mathsf{rcv}(N_B)$: We define $M'' = M' \cup \{N_B^{\mathcal{X}_9}\}$. There are two distinct ways to derive the message $h(N_B)$:

$$h(N_B)^{\pi_2(\mathsf{open}(\mathcal{X}_8))}, h(N_B)^{h(\mathcal{X}_9)} \in \mathcal{DY}_L(M'')$$

  Hence, the resulting receive event is:

$$\mathsf{rcv}(\mathcal{X}_9 \mid \pi_2(\mathsf{open}(\mathcal{X}_8)) \approx h(\mathcal{X}_9))$$

Overall, the translation of role A yields:

$$\begin{aligned} trans(M, r) = \; & \mathsf{snd}(\{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_6, \mathcal{X}_3(\mathcal{X}_7)\}_{\mathcal{X}_5})\cdot \\ & \mathsf{rcv}(\mathcal{X}_8 \mid \pi_1(\mathsf{open}(\mathcal{X}_8)) \approx \{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_6, \mathcal{X}_3(\mathcal{X}_7)\}_{\mathcal{X}_5}, \\ & \qquad \mathsf{verify}(\mathcal{X}_4(\mathcal{X}_2), \mathcal{X}_8) \approx \mathsf{true})\cdot \\ & \mathsf{snd}(\mathcal{X}_7)\cdot \\ & \mathsf{rcv}(\mathcal{X}_9 \mid \pi_2(\mathsf{open}(\mathcal{X}_8)) \approx h(\mathcal{X}_9)) \end{aligned}$$

We now substitute the variables appearing as labels in M by the terms they are labeling:

$$\begin{aligned} trans(M, r) = \; & \mathsf{snd}(\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))})\cdot \\ & \mathsf{rcv}(\mathcal{X}_8 \mid \pi_1(\mathsf{open}(\mathcal{X}_8)) \approx \{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}, \\ & \qquad \mathsf{verify}(\mathsf{pk}(B), \mathcal{X}_8) \approx \mathsf{true})\cdot \\ & \mathsf{snd}(N_A)\cdot \\ & \mathsf{rcv}(\mathcal{X}_9 \mid \pi_2(\mathsf{open}(\mathcal{X}_8)) \approx h(\mathcal{X}_9)) \end{aligned}$$

Finally, we replace the generalized receive events with pattern matching by substituting $\mathcal{X}_8 = \{\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}, HB\}_{\mathsf{inv}(\mathsf{pk}(B))}$ and simplifying the resulting role:

$$\begin{aligned} trans(M, r) = \; & \mathsf{snd}(\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}) \cdot \\ & \mathsf{rcv}(\{\{A, B, text, h(N_A)\}_{\mathsf{inv}(\mathsf{pk}(A))}, HB\}_{\mathsf{inv}(\mathsf{pk}(B))}) \cdot \\ & \mathsf{snd}(N_A) \cdot \\ & \mathsf{rcv}(\mathcal{X}_9 \mid HB \approx h(\mathcal{X}_9)) \end{aligned}$$

By the substitution for $\mathcal{X}_8$ the consistency check in the first receive becomes satisfied and can be removed. Likewise, the check in the second receive simplifies to $HB \approx h(\mathcal{X}_9)$ by rewriting algebraic equations. Overall, we have thus removed all projections from the checks in the translated role (i.e., the projections and the open destructor).

(b) The consistency check in the second receive event cannot be transformed into pattern matching: if we replace HB with $h(\mathcal{X}_9)$ in the entire role script, then this check is placed into the first receive, i.e. the agent would accept only values of the form $h(\cdot)$ at that point. This is however unrealistic as the nonce $\mathcal{X}_9$ is not learned before the second receive and the agent should accept any value HB in the first receive.

(c) The initial knowledge of the intruder is $IK_0 = \{a, b, i, h, \mathsf{pk}, \mathsf{inv}(\mathsf{pk}(i))\}$.

Assignment 3.3: Intruder models

In the lecture, the receive transition rule was defined as follows:

$$\frac{th(tid) = \mathsf{rcv}(t) \cdot tl \qquad dom(\sigma) = vars(t) \qquad t\sigma \in \mathcal{DY}(IK)}{(tr, IK, th) \rightarrow (tr \cdot (tid, \mathsf{rcv}(t\sigma)), IK, th[tid \mapsto tl\sigma])}$$

(a) Modify this rule to model a passive attacker who only eavesdrops communication, but does not actively participate by modifying messages.

(b) Does your solution still allow replays?

(c) Under which assumptions may you restrict the message derivation rules for a passive intruder to destruction rules only (i.e., remove the Composition rule)?

Solution

(a) Here is a modified rule:

$$\frac{th(tid) = \mathsf{rcv}(t) \cdot tl \qquad dom(\sigma) = vars(t) \qquad t\sigma \in IK}{(tr, IK, th) \rightarrow (tr \cdot (tid, \mathsf{rcv}(t\sigma)), IK, th[tid \mapsto tl\sigma])}$$

The intruder knowledge IK is exactly the set of all messages that have been sent on the network by send events. With this rule, the intruder send new messages agents executing receive events.

(b) Replays are still possible, and also messages can appear at a wrong receiver, or get lost in the network. This things can indeed happen in networks without an active intruder, so one may accept this as a model.

(c) The passive intruder needs composition rules only for decrypting messages that have a composed key, or for deriving composed secrets. For instance, if the intruder knows

$$M = \{\, \{|m|\}_{h(na,nb)},\ na,\ nb \,\}$$

and h is public, but $h(na, nb) \notin M$. Then, he needs the composition rule to first obtain this key $h(na, nb)$ before he can decrypt the message. However, in cases where all keys and secrets are atomic values, the passive intruder does not need composition.[2]

[2] Actually, for keys like $\mathsf{pk}(A)$ we also not necessarily need composition if the intruder initially knows $\mathsf{pk}(A)$ for every agent A that he knows.