Formal Methods for Information Security

Project report

Mathias Woringer
mworinger@student.ethz.ch

Gian-Luca Piras
gpiras@student.ethz.ch

May 25, 2021

1 The PACE protocol

The referenced files are the theory files, each of which contains an executability lemma for the described protocols as well as additional lemmas that map the security properties required in the task.

1.1 A simple challenge-response protocol

The following simple challenge-response protocol was formalized in Tamarin:

$$A \to B : x$$
$$B \to A : [x]_{k(a,b)}$$

The MAC function was realised by an user-defined function with two parameters in Tamarin. The details can be found in theory file **PACE_1.spthy**.

1.2 Mutual authentication

The challenge-response protocol is extended in this subtask in that now it is not just Alice who sends a nonce x to Bob. Bob now also generates and sends a nonce y, which is sent to A. The goal of the protocol is to reach an agreement between the two roles A and B with both nonces. Formally, the protocol now looks like this:

$$A \to B : x$$
$$B \to A : y$$
$$A \to B : [y]_{k(b,a)}$$
$$B \to A : [x]_{k(a,b}$$

For the first revision of the protocol, we implemented two different protocol approaches. Firstly, we designed the protocol sequentially. This means that Alice must have received something from Bob before Alice can send something to Bob. This variant can be found in the file **PACE_2a_seq.spthy**. Furthermore, we have implemented a parallel variant, which can be found under **PACE_2a.spthy**. To be able to carry out the MIM attack, the sequential variant is needed.

a) If we assume that there can be a network attacker who can eavesdrop on all the information exchanged. So we can assume that this protocol is vulnerable to *MIM attacks*. Since neither Alice nor Bob include any information in their messages to authenticate the message, it can be intercepted by an eavesdropper. Alice thinks she is talking to Bob. However, she is talking to Bob. Bob, on the other hand, thinks he is talking to Alice although he is talking to the attacker who is pretending to be Alice. An attack scenario could look like following:

$$\begin{aligned} A &\rightarrow A'_{eaves} : x \\ A'_{eaves} &\rightarrow B : x' \\ B &\rightarrow A : y \\ A &\rightarrow B : [y]_{k(a,b)} \end{aligned}$$

b) The problem with the protocol in a) is the fact that neither Alice nor Bob authenticate their messages. Thus, neither Alice nor Bob can actually know whether the message they receive comes from the person they want. In order to eliminate this weakness, an identifier of the respective identity is given to the message. Thus, the property of the injective agreement now holds for both.

$$\begin{aligned} A &\rightarrow B : x \\ B &\rightarrow A : y \\ A &\rightarrow B : [< A, x, y >]_{k(b,a)} \\ B &\rightarrow A : [< B, x, y >]_{k(a,b)} \end{aligned}$$

As before we have created a a sequential protocol (see theory file **PACE_2b_seq.spthy**) and a parallel protocol (see theory file **PACE_2b.spthy**).

1.3 Introducing a session key

a) Solution a

b) Solution b

1.4 Replace the password by a nonce

1.5 Introducing Diffie-Hellman: The PACE protocol

a) Solution a

b) Solution b

c) Solution c

d) Solution d

2 The Off-the-Record Messaging Protocol

2.1 Modeling the original OTR Key Exchange

2.2 Authentication Failure

2.3 Improvement

2.4 SIGMA