CHRISTIAN-ALBRECHTS-UNIVERSITÄT ZU KIEL
Institut für Informatik · AG Software Engineering
Priv.-Doz. Henning Schnoor

Engineering Secure Software Systems Winter 2020/21
Exercise Sheet 9

issued: January 12, 2021 **due:** January 21, 2021

Exercise 9.1, ProVerif example I (10 Points)

Consider the following protocol:

1. $A \to B$ $\quad \mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N_A)$
2. $B \to A$ $\quad [\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N_B), N_A]$
3. $A \to B$ $\quad N_B$

Here, k_{AB} is a long-term symmetric key shared by Alice and Bob. Is the protocol secure in the sense, that it can only be completed correctly if both Alice and Bob participate in the protocol run? Analyse the protocol "by hand" and using ProVerif.

Note: If you use the standard ProVerif **query attacker**(FAIL) modeling, you need to express the "participation property" as secrecy property. We will study a different method using events later in the lecture.

Solution First we need to clarify what it means that "both Alice and Bob" participate in the protocol run. Given that the protocol only uses symmetric keys, and identities do not appear in the protocol, this question is not formally well-defined. In particular, if we model this protocol formally, there is not even a notion of an identity that runs the protocol. Therefore, the security property under discussion should be formalized as follows:

- an honest protocol session is one in which the value k_{AB} appears
- we say the protocol is secure if for every honest responder (initiator) session, there is also an honest initiator (responder) session
- more precisely: we define the following two instances, using receive/send rules:

  initiator^i

  $\epsilon \quad \to \quad \mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N^i_A)$

  $(\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(y^i), N^i_A) \quad \to \quad y^i$

  responder^i

  $\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(x^i) \quad \to \quad (\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N^i_B), x^i)$

  $N^i_B \quad \to \quad \epsilon$

We say that an intiator or responder instance is *active* in a protocol run if it completes its second (first) step. The security property then can be formalized as follows: If the responder (initiator) session finishes, then the initiator (responder) session was active as well. This security property is obviously satisfied, since the adversary cannot generate terms of the form $\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(t)$ itself:

- If the initiator instance finishes, then some other instance must have decrypted N^i_A, as this value is received in the second step. In the case that only one instance is running, this "other instance" can only be a responder instance.
- If the responder instance finishes, then some instance must have generated the term received in the first step, which the adversary cannot do. In the case that only one responder instance is running, this must have been an initiator instance.

However, this analysis only considers the case where there is only one initiator and one responder session. A stronger security requirement would be the following: In a protocol with some finite number of initiator and responder sessions (parametrized with different values of i), for every responder (initiator) session that finishes, there is a **unique** initiator (responder) session that is active.

This security property does not hold, as the responder can essentially be used as a "decryption oracle." Therefore, an attack can be performed as follows, using a single intiator instance (A) and three responder instances (B_1, B_2, and B_3):

A: $\epsilon \rightarrow \mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N_A)$

B_1: $\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N_A) \rightarrow (\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N_{B_1}), N_A)$

at this point, the attacker simply uses the instance B_2 to decrypt N_{B_1}:

B_2: $\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N_{B_1}) \rightarrow (\mathsf{enc}^{\mathsf{s}}_{k_{AB}}(N_{B_2}), N_{B_1})$

this allows the attacker to complete the session B_1:

B_1: $N_{B_1} \rightarrow \epsilon$.

To finish the session B_2, the adversary needs the nonce N_{B_2}. For this, he can simply use the session B_3 in a similar way.

Exercise 9.2, ProVerif example II (10 Points)

Choose a cryptographic protocol and use ProVerif to analyze its security properties, that is:

1. Specify the protocol in ProVerif (including the required cryptographic primitives),
2. specify the security property in ProVerif,
3. run ProVerif to search for attacks. Does the result match with your expectations?

You can use any protocol you find interesting—all the protocols mentioned in the course so far are good candidates. The following is an incomplete list:

- your modeling of the WhatsApp authentication protocol in the first exercise,
- the (broken) authentication protocols presented in the first exercise class and their fixes,
- the Needham-Schroeder(-Lowe) protocol,
- the Woo-Lam protocol,
- the ffgg protocol,
- the repaired version of the handshake-protocol from the ProVerif tutorial (the broken version was discussed in the lecture).

Remember that most protocols are only insecure if the "correct" sessions are initiated. You can use a generic mechanism to start sessions, or use hard-coded sessions in your evaluation.