Exercise for Engineering Secure Software Systems

December 10, 2020: Exercises 4, 5

Henning Schnoor

Institut für Informatik, Christian-Albrechts-Universität zu Kiel

Leftovers

Exercise

Task (Security Modeling Issues: Are we Missing Something?)

In the lecture, we defined security of a protocol as, essentially, unreachability of a state in which the adversary learns the constant FAIL. However, this FAIL-constant obviously does not have a correspondance in a real implementation of a protocol. In particular, the rules releasing the FAIL-constant are removed from the protocol in a real implementation. As a consequence, a potential security proof of a protocol in our formal model treats a different protocol than the protocol running in a real implementation.

Are there cases where this difference results in an insecure protocol that can be proven secure in our formal model? If this is the case, how can we circumvent this issue?

Discussion: Tasks for this week

Exercise

Task (Formal Protocol Model: Features and Omissions)

There are a couple of usually assumed properties of cryptographic systems that are not explicitly expressed in our protocol model. Which of the following properties are implicitly expressed in our model, and which are not? Are any of the “omissions” problematic?

- Nonces are indeed used only once, and are freshly generated for each session.
- Private keys are never sent over the network.
- There is a complete public-key infrastructure PKI available.
- Nonces are long enough so that the adversary cannot guess them correctly.
- The adversary knows the involved algorithms, including the protocol (Kerckhoffs’s principle).
- In the absence of an adversary, the network simply forwards the protocol participant’s messages as intended.

Exercise

Task (exponential attack size)

For $i \in \mathbb{N}$, the protocol P_i is defined as follows:

- There are two instances:
  1. $\mathcal{I}_1$ has a single receive/send action $[x_1, \ldots, x_i] \to \mathsf{enc}^{\mathsf{s}}_k([t_1, t_2])$, with
     $t_1 = [x_1, [x_2, [x_3, [x_4, [\ldots, [x_{i-1}, [x_i, \mathsf{o}]] \ldots]]]]]$
     $t_2 = [[[[[\ldots[[\mathsf{o}, x_i], x_{i-1}], \ldots], x_4], x_3], x_2], x_1]$.
  2. $\mathcal{I}_2$ has a single receive/send action $\mathsf{enc}^{\mathsf{s}}_k(y, y) \to \mathsf{FAIL}$.
- The initial adversary knowledge is the set $\{\mathsf{o}, \mathsf{1}\}$.

Show that each protocol P_i is insecure, but a successful attack requires terms of exponential length. How can you use DAGs to obtain a shorter representation of the involved terms?

Exercise

Task (no unique successful minimal attack)

Show that in general, there is no unique minimal successful attack on a protocol. That is, construct a protocol and two different successful attacks on it that both have minimal size.

Exercise

Task (parsing lemma proof)

In the proof of the Parsing Lemma, we showed that in that particular setting, the term $\sigma(x)$ is constructed by the adversary. Is this generally true? More precisely: Is there a protocol P with initial knowledge I and a successful minimal attack (o, σ) such that there is a variable x with $\sigma(x) \neq x$ and $\sigma(x) \notin \mathsf{DY}(S)$, where S is the set of terms available to the adversary at the step where the first term containing $\sigma(x)$ is sent?