Universität des Saarlandes
Lehrstuhl für Softwaretechnik
Prof. Andreas Zeller, Stephan Neuhaus
Design of Secure Software Systems
Solution 11

1 Challenge-Response

1.1 Reversible Crypto Needed?

The standard protocol looks like this:

$$\begin{aligned} \text{Alice} \longrightarrow \text{Bob} &: \textit{Alice} \\ \text{Bob} \longrightarrow \text{Alice} &: R \\ \text{Alice} \longrightarrow \text{Bob} &: \{R\}_K, \end{aligned}$$

Bob doesn't need to decrypt $\{R\}_K$, he only needs to encrypt R with K and see whether it matches the value in Alice's message.

1.2 Mutual Authentication With Challenge-Response

Mutual Authentication. We already know that Bob knows that he is talking to Alice. It remains to show that Alice also knows that she is talking to Bob.

Alice decrypts $\{R\}_K$ to get R. By assumption, she can distinguish R from a random number. Therefore, the principal that has encrypted R knew the secret key K. Since we assume that only Alice and Bob know K, Alice concludes that it must have been Bob who encrypted R.

The challenge R must have a limited lifetime because otherwise Eve can replay a previous conversation with Bob. This works by exchanging "Eve" for "Bob" in the protocol.

Generating Recognizable Challenges with Limited Lifetimes. One way would be for Bob to generate a random number, prepend (or append) to it his local time and encrypt that. Alice then accepts R as authentic of the decrypted timestamp is within a predefined distance of her own local time.

A second possibility would be for Bob to send an encrypted message containing a random number concatenated with a serial number. Alice would then have to keep track of the most recent serial number she has received from Bob. She would accept the challenge as authentic if the decrypted serial number is larger than the one she has stored. (This raises the necessity of storing the last serial securely. However, since we assume that Alice can store K securely, then she can probably also store another number securely as well.)

2 Timestamp-Synchronized Authentication

Exploting Stateless Servers. If Bob never remembers past messages, then Eve could listen to a message by Alice and quickly (i.e., within the timestamp's lifetime) replay it to Bob:

$$\begin{aligned} \text{Alice} \longrightarrow \text{Bob} &: \textit{Alice}, \{t\}_K \\ \text{Eve} \longrightarrow \text{Bob} &: \textit{Alice}, \{t\}_K. \end{aligned}$$

Since the timestamp is authentic, it will be accepted by Bob.

Multiple Servers. One solution is to include Bob's identity in the message from Alice:

$$\text{Alice} \longrightarrow \text{Bob} \quad : \quad Alice, \{Bob, t\}_K .$$

This way, a message cannot be reused for other servers.