Security

Assignment 7, Wednesday October 29, 2008

Goals: After successfully completing this exercise you should have a working grasp of encrypted key exchange, and you should understand how to use cryptographic hash functions to achieve forward secrecy.

Deadline: Friday November 7, during the lecture or earlier in Flavio's or Olha's post box (in the white cabinet near the printer at the station-side-end of the corridor on the second floor of the Huygens building).

1. (3 points) Alice and Bob have agreed upon a common key k_{AB}. They are worried that it will eventually become known to others (for example because a judge orders them to make it public or because an attacker breaks into their computer). It seems that after the leak, their messages will inevitably not be secret anymore. But they at least want to have some forward secrecy. That is, they want to prevent that an attacker who has stored old messages can now decrypt them. Alice and Bob make the following pact. For the first message, they use k_{AB} as a key. For the second $h(k_{AB})$, for the third $h^2(k_{AB}) = h(h(k_{AB}))$, where h is a hash function that is known to everyone, like SHA256.
   - Explain why this does the job.
   - Which properties of a cryptographic hash function (efficiency, one-wayness, collision-resistance) are important here?
   - What must Alice and Bob to do after sending a message to ensure that the forward secrecy will be maintained from then on?

2. (3 points). Suppose Alice want to identify herself to server Bob via her terminal with her password. She could follow the following protocol:

$$A \to B : (\text{“Alice”}, \text{password})$$

   i.e. she sends her name and password to Bob, who verifies (a hash of) it against his table. This protocol has several serious drawbacks:

   (a) Eve could simply listen in on the communication channel to intercept Alice's password.

   (b) Eve could try to guess Alice's password online.

   (c) If Eve could get at Bobs table, she could crack it offline.

   So Alice and Bob share a password P, that could be weak. We could “strengthen it” as follows. First we generate a secret shared key K_P from P, and then follow this protocol:

$$\begin{aligned} A \to B &: \text{“Hi''} \\ B \to A &: \{R\}_{K_P} \\ A \to B &: R \end{aligned}$$

   where R is a random nonce. So Bob sends Alice something that only she should be able to decrypt with her password.

   - Which of the above disadvantages (a)-(c) does this solve? Does Eve have an attack on this protocol? We could also modify the protocol as follows. Alice generates via her terminal a random public/secret key pair (e_A, d_A) for one-time use, and then follows this protocol:

$$\begin{aligned} A \to B &: \{e_A\}_{K_P} \\ B \to A &: \{\{R\}_{e_A}\}_{K_P} \\ A \to B &: \{A\}_R \end{aligned}$$

So Alice sends the public key to Bob, encrypted via her password. Bob then answers with a random number R, encrypted via the public key and the password. Now Alice should be the only one that can decrypt this, and return $\{A\}_R$.

- Which of the above disadvantages (a)-(c) does this solve? Does Eve have an attack on this protocol?

3. (4 points) Consider Otway-Rees authentication protocol (Tanenbaum, fig. 8-41.) Remove the identifiers from the encrypted ticktes on the steps 1 and 2, so that instead of $\{A, B, R, R_A\}_{K_A}$ one has $\{R, R_A\}_{K_A}$ and instead of $\{A, B, R, R_B\}_{K_B}$ one has $\{R, R_B\}_{K_B}$. Now Eve can perform a "man-in-the-middle" attack. How? At the end of the attack Alice will have a shared session key K_S with Eve, but she will be sure that she has a session with Bob.