Security
Assignment 6, Monday, October 17, 2011

Handing in your answers: There are two options:

1. Put your solutions in Pim's post box (in the white cabinet, lower door, near the printer at the station-side-end of the corridor on the second floor of the Huygens building). Before handing in, make sure:
   - your names and student numbers are written on the document.
2. Send your solutions by e-mail to `pim@cs.ru.nl` with subject '*assignment 6*'. This e-mail should only contain a single PDF document as attachment. Before handing in, make sure:
   - your names and student numbers are included in the document (since it will be printed),
   - your names are part of the filename (example: PimVullers_FlavioGarcia-6.pdf), and
   - the file is a PDF document.

Deadline: Monday, November 7, 8:30 sharp!

Properties of cryptographic hash functions: A cryptographic hash is a function h that assigns to a bit string another bit string (usually of fixed length, but this is not a formal requirement here), and has the following four properties:

(E) h can be computed efficiently (efficiency),

(O1) *given a bit string* y it is infeasible to determine a bit string x such that $h(x) = y$ (pre-image resistance),

(O2) *given a bit string* x it is infeasible to determine a different bit string x' ($x \neq x'$) such that $h(x) = h(x')$ (2nd pre-image resistance),

(C) it is infeasible to determine *any two different bit strings* x *and* x' ($x \neq x'$) such that $h(x) = h(x')$ (collision resistance).

1. **(3 points)** The following authentication protocol makes use of the Lamport hash. The Lamport hash basically is repeatedly hashing an input value. We write $h^n(x)$ for n times hashing x, e.g. $h^3(x) = h(h(h(x)))$. Now, the server stores a triple $(\text{user}, n, Y = h^n(\text{password}))$ for each user in a database and initially takes $n = 10.000$.

   $$\begin{array}{lcl} 1. & A \longrightarrow S & : \quad A \\ 2. & S \longrightarrow A & : \quad n \\ 3. & A \longrightarrow S & : \quad X = h^{n-1}(\text{password}) \end{array}$$

   The server checks if $h(X) = Y$, then decrements n and sets $Y := X$. So after a successful run of this protocol the Server holds a new triple $(\text{user}, n-1, Y = h^{n-1}(\text{password}))$.

   (a) Can you think of an attack (here, relay is not considered an attack)? Can the attacker learn the password? Use the arrow notation ($A \longrightarrow B$: message) in your explanation.

   (b) At some point $n = 0$. Is it safe to start over again and put n back to 10.000? Briefly explain your answer.

2. **(2 points)** One way to construct a one-time pad uses a hash function h and starts with an initial key K and appends iterated applications of h:

   $$K \cdot h(K) \cdot h(h(K)) \cdot h(h(h(K))) \cdots$$

   Is this way to construct a one-time pad secure? If yes, briefly motivate your answer, otherwise give an attack.

3. **(3 points)** Alice and Bob live together but work at different places. On the way home from work they try to decide who has to cook dinner, and use the following protocol, where h is a hash function, and n_A, n_B are numbers chosen by Alice and Bob.

$$\begin{array}{lll} 1. & A \longrightarrow B & : \quad h(n_A) \\ 2. & B \longrightarrow A & : \quad n_B \\ 3. & A \longrightarrow B & : \quad n_A \end{array}$$

If $n_A + n_B$ is even, Alice has to cook, and Bob otherwise.

   (a) Does the size of the chosen numbers affect the security of this protocol. Give short answers, *both* for Alice and for Bob's perspectives.

   (b) Which security properties of the hash function h are relevant for this protocol? Give brief answers.

4. **(2 points)** Consider the following block encryption mechanism (due to Karn), based on only a secure hash function h and bitwise XOR $\oplus$. Assume h produces hash values of length 256 bits. Assume a symmetric/shared key k of even length.

   Let m be a plaintext message, of length 512 bits. Split m halfway in two parts m_1, m_2, both of length 256, so that $m = m_1 \cdot m_2$, where $\cdot$ denotes concatenation. Split also the key k halfway in two parts $k = k_1 \cdot k_2$.

   Now define $k\{m\} = c_1 \cdot c_2$ where

$$c_1 = m_1 \oplus h(m_2 \cdot k_1) \qquad \text{and} \qquad c_2 = m_2 \oplus h(c_1 \cdot k_2).$$

   Show how to decrypt: how to obtain m from $k\{m\}$, assuming that you know k.