CHALMERS TEKNISKA HÖGSKOLA
Datavetenskap
Björn von Sydow DIT250/TDA351

Solutions to exam in Cryptography, December 16, 2010

1. (a) A collision attack is an attempt to find two different messages m_1 and m_2, such that $h(m_1) = h(m_2)$, where h is the hash function under consideration.

   (b) Any hash function is vulnerable to the birthday attack. This is the attack where the attacker finds a collision by generating random messages, hashing them and storing the results in a dictionary, checking each time if the new hash value is already in the dictionary. If the hash values are uniformly distributed n-bit strings, one can show that a collision can be expected in $O(\sqrt{2^n}) = O(2^{n/2})$ messages. A cryptographic primitive with an attack that takes $2^{n/2}$ steps is said to offer $n/2$ bits of security.

2. (a) DES.

   (b) Its key length, 56 bits, is too short, permitting brute force attacks.

   (c) One can use triple DES (3DES), where a message is encrypted three times with DES, using different keys. This gives key length 168 bits, which is more than adequate. Often, the second step is decryption rather than encryption and often the first and last keys are identical, reducing key length to 112 bits.

   (d) Some problems:

   - Resending of (parts of) a message can be recognized.
   - An active adversary can reorder blocks with predictable results, if message structure is known.
   - Patterns, fixed parts of message structure, etc are recognizable.

3. The user computes $\Phi(N) = (p-1)(q-1)$ and then d such that $d \cdot e = 1(\bmod \Phi(N))$, using the extended Euclidean algorithm. (If $\gcd(e,N) \neq 1$, this may fail and a new e need to be chosen.)

4. (a) Let $k_1 k_2 k_3 \ldots$ be the keystrem generated by the LFSR. The bits in the stream satisfy

$$k_n = k_{n-16} \oplus k_{n-17}, \quad n = 18, 19 \ldots \qquad (*)$$

   Using his knowledge of the plaintext bits, the adversary first computes k_i for $i = 1001, 1002, \ldots 1100$ as $k_i = c_i \oplus m_i$. He can then compute the rest of the keystream using (*). He can also compute the first 1000 bits of the keystream by rewriting (*) as $k_{n-17} = k_{n-16} \oplus k_n$. When he knows the entire keystream, he can decrypt and find $m = c \oplus k$.

   (b) The maximal period is $2^{17} - 1$ bits.

5. Cedric uses message 2 from the run and initiates a new run with Trent, pretending to be B:

$$\begin{array}{lll} 1. & C(B) \rightarrow T & : \; B, C, \{K\}_{K_{BT}} \\ 2. & T \rightarrow B & : \; \{K\}_{K_{CT}} \end{array}$$

   Cedric picks up Trent's reply, which gives him K encrypted for himself. He decrypts, gets K and can now decrypt $\{M\}_K$.

   An improvement to the protocol would be to make messages more explicit, i.e. by including the identities of the parties in the encrypted part of the messages:

$$\begin{array}{lll} 1. & A \rightarrow T & : \; A, B, \{A, B, K\}_{K_{AT}} \\ 2. & T \rightarrow A & : \; \{B, A, K\}_{K_{BT}} \\ 3. & A \rightarrow B & : \; \{B, A, K\}_{K_{BT}}, \{M\}_K \end{array}$$

   If Trent checks this, he will not accept message 1 of Cedric's attempt.

6. (a) For a correctly signed message we have that

$$r = c^{x'} = (s^a X^b)^{x'} = (H(m)^{xa} g^{xb})^{x'} = H(m)^a g^b,$$

   where the last equality makes use of the fact that $xx' = 1 (\bmod q)$. So, Victor checks that $r = H(m)^a g^b$.

   (b) If Alice is honest, $r = s^{ax'} g^b$, (but we will then not have $s = H(m)^x$). Thus $(rg^{-b})^{a_1} = s^{x' a a_1}$. Similarly, also $(r_1 g^{-b_1})^a = s^{x' a a_1}$.

   (c) As for all discrete-log based problems, an elliptic curve group currently offers the best efficiency for a given security level.

7. (a) We use the same notation as in home assignment 2. Let $R_i =< r_1, r_2, r_3, r_4, r_5, r_6, r_7, i >$ where the first seven bytes are arbitrary and the last byte i is systematically varied by the adversary. Let the sought block $D_{K_{AB}}(C_k)$ be $< b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8 >$. The adversary first constructs a sequence of messages ending with the two blocks $R_i || C_k$ and sends them to Bob one by one. Bob views each of them as a message encrypted in CBC mode and decrypts it. The last block becomes $D_{K_{AB}}(C_k) \oplus R_i$ and the last byte of this block is $b_8 \oplus i$. Bob now checks padding and informs the adversary. If padding is correct, the adversary guesses that the padding is of type 0, and thus $b_8 \oplus i = 0$, which gives $b_8 = i$. The adversary now proceeds to find b_7 by constructing messages ending with $S_i || C_k$, where $S_i =< r_1, r_2, r_3, r_4, r_5, r_6, i, b_8 \oplus 1 >$. If padding is correct, the last byte of the pad block is 1, hence also the 7th byte is 1 and the adversary gets $b_7 \oplus i = 1$, i.e. $b_7 = i \oplus 1$. The adversary proceeds, just as in home assignment 2, to find b_6, b_5, by forcing message to have pad blocks of type 2, 3, ...

   (b) We apply the decryption function to the CBC equation given in the problem to get

$$D_{K_{AB}}(C_i) = A_i \oplus C_{i-1},$$

   from which we get

$$C_{i-1} = D_{K_{AB}}(C_i) \oplus A_i.$$

   Put $i = k$ and we get directly how C_{k-1} should be chosen.

   (c) To construct C_{k-2}, the adversary uses Bob as a padding oracle in order to find $D_{K_{AB}}(C_{k-1})$, just as in (a). He then uses the last equation in (b) above for $i = k - 1$ and constructs C_{k-2}. Proceeding in the same way, he can construct the ciphertext blocks in reverse order.