Institut for Matematik og Datalogi
Syddansk Universitet

April 7, 2011
JFB

Cryptology – F11 – Week 9

Lecture, April 4

We discussed subliminal channels and begin on chapter 8, covering sections 8.1 and 8.4 and up through and including Theorem 8.2 in section 8.2.

Lecture, April 7

We finished section 8.2, and covered sections 8.3 and 7.6 (the latter from some notes).

Lecture, April 13

We will finish with undeniable signatures from the notes (the denial protocol) and begin on zero-knowledge (from the notes by Ivan Damgård and Jesper Buus Nielsen, available through the course's homepage).

No lecture, April 18

Lecture, April 27

We will begin on zero-knowledge (from the notes by Ivan Damgård and Jesper Buus Nielsen, available through the course's homepage).

Problem sessions April 14 and 27

1. For SHA-1, give an estimate for how many bits of the original 512 bits of input are XORed together to get the different bits of the different W_i.
2. For SHA-1, what happens to the low-order bit of W_{79}?

3. In the verification protocol for undeniable signatures (in the textbook), the verifier chooses randomly two values e_1 and e_2. Why are there two values? Why not just let $e_2 = 0$ always?

4. Give a protocol for digital signatures in which the verification (which can be shown to the judge) does not reveal to the judge the contents of the document which was signed.

5. Some applications are sensitive to *replay attacks*, where an adversary takes a copy of an original signed message and sends it again later. (For example, it should not be possible to repeat a request to transfer money from one bank account to another.) Design a protocol (using signatures) to prevent replay attacks.

6. According to Ivan Damgård, the essence of SSL (authentication between a server S and a client C, predessor of TLS) is as follows:

   (a) C sends a hello message containing a nonce (a random challenge) n_C.

   (b) S sends a nonce n_S and its certificate *Cert(S)* (issued by a certification authority and containing the public key K_S of S.)

   (c) C verifies *Cert(S)* and chooses a pre-master secret pms at random. C sends $E(K_S, pms)$, its certificate *Cert(C)* to S, and its signature sig_C on the concatenation of n_C, n_S, and $E(K_S, pms)$.

   (d) S sends C a MAC on all messages sent so far in this protocol, using pms as the secret key.

   (e) C verifies the MAC. IF OK, it send S a MAC on all messages sent so far in this protocol.

   (f) Use a shared function to compute keys for authentication and encryption from n_S, n_C, and pms.

   In this protocol, how does S authenticate itself? How does C authenticate itself. Why do the keys depend on n_S and n_C, instead of just pms? Is it important that C actually send a MAC at the end, or would OK be enough?

7. Consider the following proposal for a hash function, where $E(K, M)$ is encryption of the 128 bits of M using a 128-bit key K in Rijndael

(AES). Let IV be a 128-bit random string. Pad the document to be hashed with zeros so that the number of bits is divisible by 128. Let the resulting document be $M = m_1||m_2||...||m_r$, where each block m_i contains exactly 128 bits and the operation $||$ is concatenation.

$$
\begin{array}{rcl}
H_0 & \leftarrow & IV \\
H_1 & \leftarrow & E(m_1, H_0) \\
H_2 & \leftarrow & E(m_2, H_1) \\
 & . & \\
 & . & \\
 & . & \\
H_r & \leftarrow & E(m_r, H_{r-1}) \\
H & \leftarrow & E(m_1 \oplus m_2 \oplus ... \oplus m_r, H_r)
\end{array}
$$

The operation $\oplus$ is bit-wise exclusive-or, and the output of the hash function is (H_0, H). (Note that there a message which has zeros at the end will hash to the same value as that message with the zeros truncated (removed). Thus, finding collisions is trivial, but we will ignore that type of collision in the following.)

a. Using the Birthday Paradox, define and analyze (how many calls to Rijndael) an algorithm for finding a collision.

b. The above hash function would be much less secure if the steps with the ByteSub transformations were simply removed from Rijndael. Which is the hardest of the three problems Preimage, Second Preimage, and Collision, which could now be solved efficiently? How would you solve that problem?

8. Do problems 7.3a and 7.4a in the textbook.

9. Do problem 8.1 in the textbook. In part b, it should say "$s_o = \frac{-b}{a-1} \pmod{M}$".

10. Recall the quadratic residuosity implementation of probabilistic encryption, from the original paper by Goldwasser and Micali. Design a subliminal channel for use with this cryptosystem.

    Here we are assuming that the two prisoners are allowed to send encrypted messages to each other, but the warden always forces the receiver to decode the message for him (and the sender suspects that this is happening). With the subliminal channel, the receiver will decrypt an innocuous (or even deceptive) message for the warden, but the warden will never know about the true message which the receiver gets at the same time.

If the warden actually carries the message, he can defeat this plan and eliminate the subliminal channel. To do this, the warden takes the encoded message from the sender and changes it. Afterwards the new cryptogram will still be an encryption of the same message, but the subliminal channel will be gone.

Explain how this can all be done even though the prisoners are not allowed to use a redundant representation of the original message.

11. Do problem 8.5. Argue that if the Discrete Logarithm Problem is hard, then this generator is secure, i.e. there is no probabilistic polytime ϵ-Distinguisher.