IND426 Tutorial 10 (Solutions)  
(Roberto Togneri 1998)

1. (a) In the a product cipher the P-boxes and S-boxes alternate. Is this arrangement any more secure than having all the P-boxes and then all the S-boxes?  
(b) Suppose that a message has been encrypted using DES in ciphertext block chaining mode. One bit of the ciphertext in block Ci is accidentally transformed from a 0 to a 1 during transmission. How much plaintext will be garbled as a result?

Solution

(a) A sequence of P-boxes is effectively just one P-box and a sequence of S-boxes is effectively just one S-box, so in the end you would just have the same encrypting "effect" as one P-box followed by one S-box. By alternating P-boxes and S-boxes more complex encodings are possible.

(b) $P_i$ and $P_{i+1}$ only will be garbled, hence two blocks of plaintext will be garbled if one block of ciphertext is modified.

2. (a) You and your friend want to confirm that you each have the same session key without allowing an eavesdropper listening to your interaction somehow deriving what the key. Your friend suggests that you create a random bit string that is the length of the key, XOR it with the key, and send the result over the channel. Your friend will then XOR the incoming block with the key and sends the result back to you. You check, and if what you receive is your original random string, you have verified that your friend has the same session key, yet neither of you has ever transmitted the key. Is there a flaw in this scheme?

(b) Your lecturer has had the bright idea of doing away with the complexities of DES and CBC by simply using the session key as the seed to a random number generator to generate a one-time pad encryption of the complete plain-text. Discuss the merits or otherwise of this scheme.

Solution

(a) Silly idea! Since $C = K$ (session key) $\oplus R$ (random bit-string) and since an eavesdropper will be able to pick up both $C$ (which you send) and $R$ (which your friend returns to you) then $K = C \oplus R$!

(b) It may work but why bother? Random number generators are, at best, as complex and as expensive to use as DES with CBC and more typically not optimised for speed like DES and probably less secure! It only sounds like a good idea because of the relative ease in accessing a random number generator which is available with most programming libraries.

3. Consider the Needham-Schroeder authentication protocol as a solution to the replay attack which can occur with the simple KDC solution to the authentication problem:

(a) Briefly explain what the messages 1 and 2 achieve?

(b) Briefly explain how messages 3, 4 and 5 convince Alice that she is talking to Bob and vice versa? Hence show why this protocol is resistant to the replay attack.

(c) Why is $B$ included in message 2 given Alice knows she wants to talk to Bob?

(d) What is the point of the random number, $R_{kA}$, in messages 1 and 2?

(e) Is there any need to encrypt $R_{kA}$ in message 3?

(f) Would $K_s(R_{kA})$ in message 5 be just as effective as $K_s(R_{kA}-1)$?

Solution

(a) Messages 1 and 2 allow Alice to communicate with the KDC to obtain the session key ($K_s$) as well as the information that would be sent to Bob ($K_s(A,K_s)$) to securely transmit the secret session key to him. Alice can use this to challenge Bob as shown in (b)

(b) Alice challenges Bob with $K_s(R_{kA})$ and only Bob (who can decrypt $K_s(A,K_s)$ to retrieve $K_s$) can correctly reply with $K_s(R_{kA}-1)$. Similarly, Bob challenges Alice with $R_{kA}$ and only Alice can correctly produce $K_s(R_{kA}-1)$. Because the $R_{kA}$ and $R_{kA}$ act as nonce’s this protocol is resistant to replay attack.

(c) Prevents Trudy from substituting $B$ for $T$ in message 1, and hence $K_s$ for $K_T$ in message 2 without Alice being aware of the switch (the KDC would return $T$ instead of $B$ in message 2 and Alice would realise something was wrong, without including this Alice would be blissfully ignorant).

(d) Makes message 2 from the KDC resistant to replay attack.

(e) On the face of it no, but consider a reflection attack. Trudy can simply send, say, $R_{kA}$ to Bob and magically obtain $K_s(P)$ (and attempt to fool Bob). Obtaining $K_s(P-1)$ is useless since this changes the decrypted message.

(f) On the face if it yes. But this does prevent Trudy from sending, say, plain-text message $P$ to Alice in order to obtain $K_s(P)$ (and attempt to fool Bob). Obtaining $K_s(P-1)$ is useless since this changes the decrypted message.

4. (a) Perform RSA encryption with $p = 3$ and $q = 11$ on $P = 19$ assuming the smallest values for $d$ and $e$ are chosen. Verify your result by the corresponding decryption. What are the public and private keys?

(b) Alice picks $n = 47$ and $g = 3$ and these will be sent to Bob in plaintext. Alice secretly picks $x = 8$ and Bob secretly picks $y = 10$. Show how the Diffie-Hellman exchange allows a common session key to be derived by both Alice and Bob in a secure manner.

Solution

(a) $n = 33$ and $z = 20$. Now we need to find the smallest $d$ which is relatively prime to 20: $d = 3!$! Now we need to find $e$ such that $(3e) \mod 20 = 1$: hence $e = 7$!

So $C = P^e \mod 33 = (19)^7 \mod 33 = 13$, and $P = C^d \mod 33 = (13)^7 \mod 33 = 12$.

The public-key is $\{7, 33\}$ and the private key is $\{3, 33\}$

The public key is $(7, 33)$ and the private key is $(3, 33)$.
Alice calculates $g^x \pmod{n} = 38 \pmod{47} = 28$ and sends (47, 3, 28) to Bob. Bob extracts the values for $n$ and $g$ and then calculates $g^y \pmod{n} = 310 \pmod{47} = 17$ and sends (17) to Alice. Now Alice will derive $K_S = 17 \pmod{47} = 4$, and Bob will derive $K_S = 28^{10} \pmod{47} = 4$ and they both have the same session key $K_S = 4$.

5. (a) Comment on using the check-bits of error-detecting codes (e.g. checksum, CRC) as message authentication codes.

(b) Public-key encryption using, say RSA, is about 1000 times slower than private-key encryption using, say, IDEA. Briefly explain why this result is not surprising. Furthermore, public-key encryption techniques also require special arithmetic routines. What special arithmetic do you think is required?

(c) Compare and contrast message authentication of electronic messages using an MD5 hash signed with the private-key of the sender with normal signatures on hardcopy messages?

Solution

(a) Since any change in the document would be construed as a bit errors then an error-detecting code would pick these up and indicate the document was incorrect. However error-detecting codes are designed for channel errors not message authentication and standard error-detecting codes are designed for speed, specific rather than total coverage of all errors, and would not be suitable for message authentication codes.

(b) IDEA uses a complex combination bit-wise XOR operations, modulo-2^{16} addition and modulo-2^{16} multiplication which can all be easily implemented directly in hardware logic or the corresponding 16-bit microprocessor instructions. RSA on the other hand requires floating-point operations to carry out the exponentiation operation and modulo-n arithmetic operations. Indeed the since $p$ and $q$ are typically (very!) large numbers (order of $10^{100}$) a special set of arithmetic routines to deal with such large numbers on a limited precision CPU are needed. And big number arithmetic operations are bound to be even slower than normal operations.

(c) A MAC allows the integrity of the message itself to be checked whereas this is not the case in a hardcopy message (although the physical cut&paste changes may be visibly apparent!). On the other hand the MAC is requires possession of the public-key of the sender in order to authenticate the message whereas a visible signature on a hardcopy message can be easily verified by inspection (at least by close colleagues). Message authentication is more secure than hardcopy signatures but it depends on the trust people give to the software and systems available to actually authenticate and verify the messages. In most cases blind trust in the software is needed so visible inspection is not possible. Given the level of current software engineering practice a signed hardcopy message is probably still more secure!