6.915 Computer and Network Security **Fall Term, 1995**

Lecture 5 : September 21,1995

Lecturer: Ron Rivest *Scribe: Dinko Zidaric*

1 Modes of operation of block ciphers

In the last lecture, we've covered how to encrypt a 64-bit code block with DES, but we haven't discuss how to effectively use the algorithm to encrypt messages which don't happen to be 64 bits long. To make DES work for arbitrarily long messages and to prevent some of the security breaches present in the most straightforward implementation of DES, other modes of operations have been invented. The most popular ones are Electronic Code Book (ECB), Cipher Block Chaining (CBC), and Output Feedback Mode (OFM). In the following sections we discuss each of them in turn.

1.1 Electronic Code Book (ECB)

Figure 1: Electronic Code Book Mode

This is the most straightforward way of using DES and is shown in Figure 1. To encrypt the message we divide the plaintext message into 8-byte blocks, encrypt every block separately, and concatenate the resulting blocks. If we encrypt two identical blocks of cyphertext, the resulting ciphertext blocks will be identical as well.

Therefore, we give an eavesdropper some information about the encrypted data, and depending on the context, that information might be sufficient to compromise the secrecy.

1.2 Cipher Block Chaining (CBC)

Figure 2: Cipher Block Chaining Mode

CBC method avoids some of the possible problems encountered with the ECB method. CBC method produces different ciphertext blocks even if the plaintext blocks are identical. As shown in Figure 2, the input to a single DES encryption module is not just the block of a plaintext, but it is also the XOR of the plaintext block and the cyphertext produced in the calculation of the previous plaintext block. Since the first encryption module does not have a predecessor, a random number known as the Initialization Vector (IV) is fed into the first XOR.

In addition to using CBC method for data encryption, CBC can be used in the calculation of a message authentication code (MAC). For example, we can use last S bits of CBC encryption of a message M under a key K as the $MAC(M, K)$ of length S. To make this method work, we need to also prepend the length of the message to the message before we produce the MAC. As we will see in the homework, if the length is appended or is missing altogether, the MAC is not secure.

1.3 Ouput Feedback Mode (OFM)

Ouput feedback mode acts like a pseudorandom number generator. A message is encrypted by XOR-ing the message with the output stream of the OFM module. To create a stream of pseudorandom bits, a 64-bit Initialization Vector X is encrypted under key K creating Y_0. If we are encrypting b bits of the message at the time (usually 8 bits as in the figure) the left-most b bits of Y_0 are used to XOR the b bits of plaintext and then appended to the IV vector. The IV vector is then shifted b bits to the left and the whole process is repeated to encrypt the next b bits. So Y_1 is created and so on ...

Figure 3: Output Feedback Mode

1.4 Error Propagation

When using ECB method for data encryption, if we make an error during transmission of a block, only the ciphertext and its corresponding decrypted plaintext will be corrupted. All other blocks will not be affected.

If an error is made while transmitting a block encrypted using the cipher block chaining method, some limited error propagation will occur. If you look back at Figure 2, you will notice that every ciphertext block C_i is involved in decryption of two message blocks M_i and M_{i+1}. Therefore, the decryption of a single corrupted ciphertext block will produce two incorrect plaintext blocks.

The output feedback mode has 1-to-1 error propagation property similar to electronic code book. A single corrupted ciphertext block will produce a single incorrect plaintext block.

1.5 Padding Techniques

Since not all messages have lengths in multiple of DES block length (i.e. 64 bits), different padding techniques were developed to extend plaintext messages. One technique involves always appending the message with a bit 1 and a necessary number of zeros producing a plaintext message of required length.

2 Application of Secret Key Cryptography: Pay - TV

Most of today's pay - TV programming is distributed in the following way: The studio produces TV program which is encrypted with a program key and is transmitted over the satellite to a local cable company. The cable company distributes the signal over the wires to the individual customers and it is also responsible to provide the program keys to its customers subscribed to a particular program. Every customer has a decoder box with the secret key known only to the box and to the cable company. In order to distribute a program key, the cable company encrypts the program with the secret key of a customer decoder box and sends it to the decoder box over a separate control channel. The decoder box decrypts the message using its secret key, and extracts the program key.

3 Public-key Cryptography

The main idea behind the public-key cryptography is that the encryption and decryption capabilities are decoupled. Alice can have two keys P_A and S_A which are her public and secret keys, respectively. Alice can now give Bob (as well as everybody else) her public key P_A, and Bob can use it to encrypt a message M producing ciphertext $C = E(P_A, M)$. Although Bob encrypted the message for Alice, he nor anybody else can decrypt it. Alice, in possesion of her private key, is the only party that can do it. Alice takes her private key S_A and decrypts the ciphertext C producing the original message M. $M = D(S_A, C) = D(S_A, E(P_A, M))$

Note that the secrecy of P_A is not required. Anyone can encrypt messages for Alice.

In general, it should not be possible to figure out how to decrypt, even if you know everything how to encrypt, though "dictionary attacks" are still possible. Dictionary attacks involve searching through the space of possible plaintexts, encrypting all the candidates, and comparing the resulting ciphertexts against the ciphertext we are trying to decrypt. One of the ways to protect against these kind of attacks is through something called **Randomized** Public-key Encryption. The ciphertext C of a message M is equal to $(R \oplus M, E(P_A, R))$, where R is a random number. For longer messages a hybrid scheme between public and secret key cryptography is used most often. The ciphertext is a pair $(DES(K, M), E(P_A, K))$. This scheme offers the advantages of both public-key cryptography (flexibility) and of classical cryptography (speed).

3.1 Authentication - Digital Signatures

Digital signatures are the public-key solution to authentication. Digital signature reverse the roles of public and private keys. Assuming that S_A can be used to encrypt as well as decrypt (as in RSA), $Signature_A(M) = E(S_A, M)$. To verify the signature, anybody can use Alice's public key P_A to check whether the message M is equal to $D(P_A, Signature_A(M))$. Digital signatures are electronic equivalent for handwritten signatures for electronic checks, contracts, etc. Digital signatures can also be combined with the public key encryption schemes in a way that a message can be first signed, by using the secret key of the sender, and then sealed by using the public key of the receiver of the message.

3.2 Certificates

In our previous discussions about encryption and authentication using public-key cyrptography, we have assumed that Bob was given Alice's public key P_A. Now we should discuss how Bob can be assured that the key he has is definitely Alice's key and not of an imposture. One simple way is to get it through a secure channel (e.g. directly from Alice). Another way is by having a "Certification Authority" (CA) which would sign a "certificate" saying that Alice's public key really is P_A. Although we have a signed digital certificate of Alice's public key, we still have to be able to verify the signature on the certificate. One way to do this is to use Certification Hierarchy. As explained in great detail in S. Chockani's paper (part of today's reading material), there is a tree of CAs at different levels. At the root level is the main CA whose public key is known to everybody and is not checked. The root level CA issues certificates to Level 1 CAs, Level 1 CAs issues certificates to Level 2 CAs, etc. To check a signature on a certificate, we just follow certificate links up the tree until we reach the root level CA.

4 Handouts and Readings

There are 4 handouts: The first handout is Readings for Lecture 5 including "How to Share a Secret" by Adi Shamir, "Toward a National Public Key Infrastructure" by Santosh Chokhani, and "Key Escrowing Today" by Dorothy Denning and Miles Smid. The other three are scribe notes for Lecture 2, Problem Set 2, and a New York Times article "Fraud Can Florish Without the Internet".

The assigned reading is Chapter 3 in the textbook.