Chapter 1 – Overview

OSI Security Architecture:

- **Security attacks**
  - **Passive attack:** Passive attacks are in the nature of eavesdropping on, or monitoring of, transmissions. Passive attacks are very difficult to detect, because they do not involve any alteration of the data
  - **Active attack:** Active attacks involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories: masquerade, replay, modification of messages, and denial of service.

- **Security Services**
  - Authentication, Access Control, Data Confidentiality, Data Integrity, Nonrepudiation, Availability Service

- **Security Mechanisms**
  - specific security mechanisms:
    - Encipherment, digital signatures, access controls, data integrity, authentication exchange, traffic padding, routing control, notarization
  - pervasive security mechanisms:
    - trusted functionality, security labels, event detection, security audit trails, security recovery

Chapter 2 – Classical Encryption Techniques

Playfair Cipher

In this case, the keyword is monarchy. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter. Plaintext is encrypted two letters at a time, according to the following rules:

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

1. Repeating plaintext letters that are in the same pair are separated with a filler letter, such as x, so that balloon would be treated as ba lx lo on.
2. Two plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.
3. Two plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the column circularly following the last. For example, mu is encrypted as CM.
4. Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM (or JM, as the encipherer wishes).

Vigenere cipher

$$C = C_0, C_1, C_2, \ldots, C_{n-1} = \mathrm{E}(K, P) = \mathrm{E}[(k_0, k_1, k_2, \ldots, k_{m-1}), (p_0, p_1, p_2, \ldots, p_{n-1})]$$
$$= (p_0 + k_0) \bmod 26, (p_1 + k_1) \bmod 26, \ldots, (p_{m-1} + k_{m-1}) \bmod 26,$$
$$(p_m + k_0) \bmod 26, (p_{m+1} + k_1) \bmod 26, \ldots, (p_{2m-1} + k_{m-1}) \bmod 26, \ldots$$

```
key:          deceptivedeceptivedeceptive
plaintext:    wearediscoveredsaveyourself
ciphertext:   ZICVTWQNGRZGVTWAVZHCQYGLMGJ
```

One time pad

Same as vigenere but the key's length is the message length.

Steganography

- **Character marking**: Selected letters of printed or typewritten text are overwritten in pencil. The marks are ordinarily not visible unless the paper is held at an angle to bright light.
- **Invisible ink**: A number of substances can be used for writing but leave no visible trace until heat or some chemical is applied to the paper.
- **Pin punctures**: Small pin punctures on selected letters are ordinarily not visible unless the paper is held up in front of a light.
- **Typewriter correction ribbon**: Used between lines typed with a black ribbon, the results of typing with the correction tape are visible only under a strong light.

Why Stream Cipher is not recommended to use same key?

If two plaintexts are encrypted with the same key using a stream cipher, then cryptanalysis is often quite simple. If the two ciphertext streams are XORed together, the result is the XOR of the original plaintexts. If the plaintexts are text strings, credit card numbers, or other byte streams with known properties, then cryptanalysis may be successful.

Chapter 3 – Block Ciphers and the DES

DES

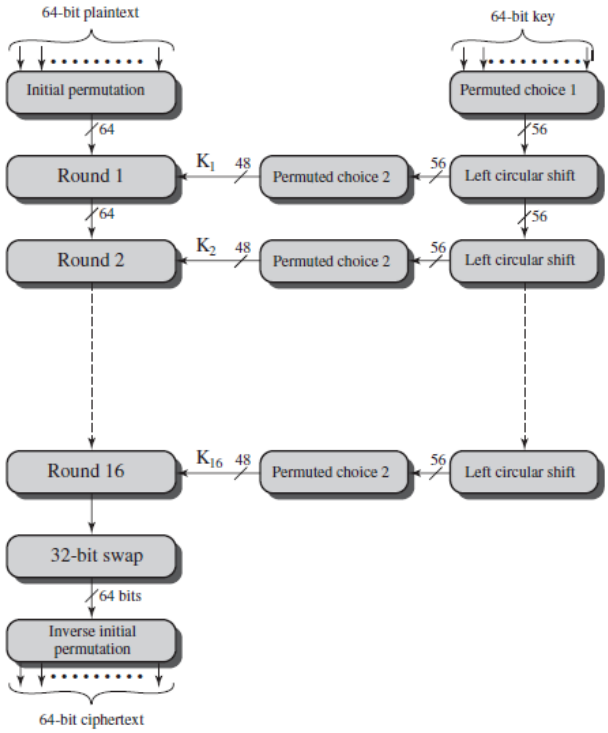


Figure 3.5 General Depiction of DES Encryption Algorithm

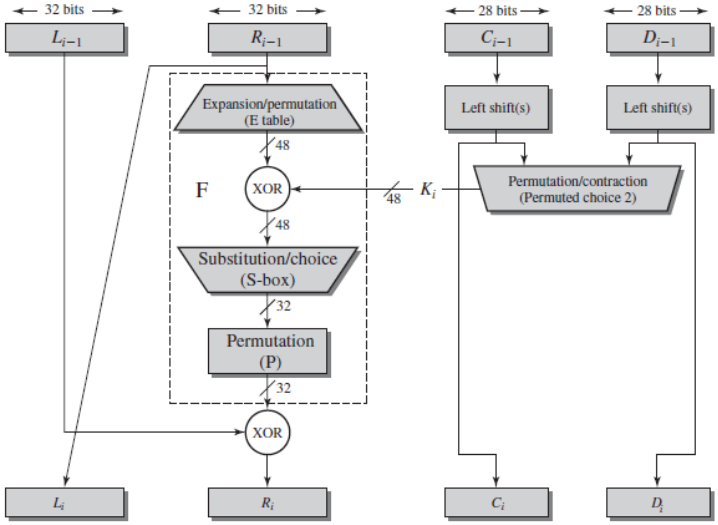


Figure 3.6 Single Round of DES Algorithm

SBOX

The outer two bits of each group select one of four possible substitutions (one row of an S-box). Then a 4-bit output value is substituted for the particular 4-bit input (the middle four input bits). The 32-bit output from the eight S-boxes is then permuted, so that on the next round, the output from each S-box immediately affects as many others as possible.

What is Avalanche effect?

A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext.

What is timing attack?

In essence, a timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts.A timing attack exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs.

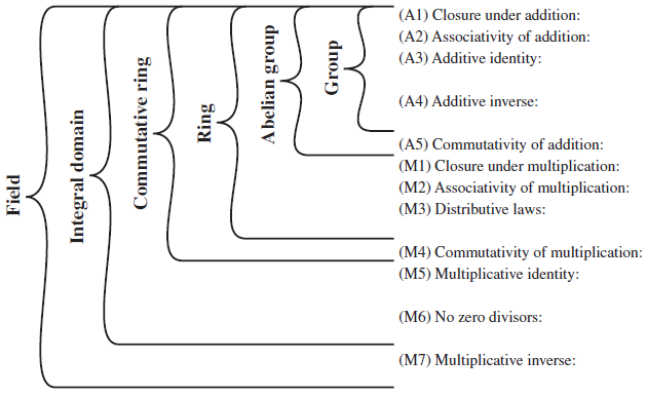
Diffusion

Dissipates statistical structure of plaintext over bulk of ciphertext

Confusion

Makes relationship between ciphertext and key as complex as possible

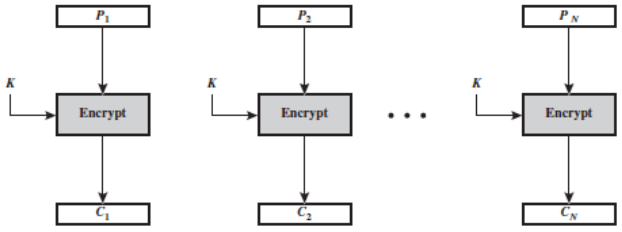
Chapter 4 – Basic Concepts in Number Theory and Finite Fields



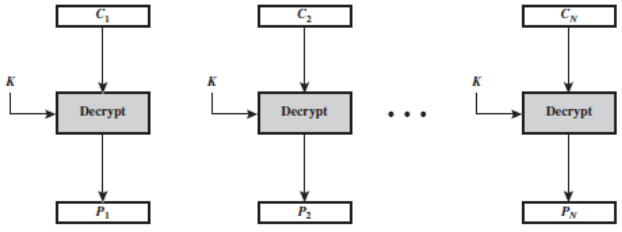
Solve examples for this chapter

Chapter 6 – Block Cipher Operation

Electronic Code Book

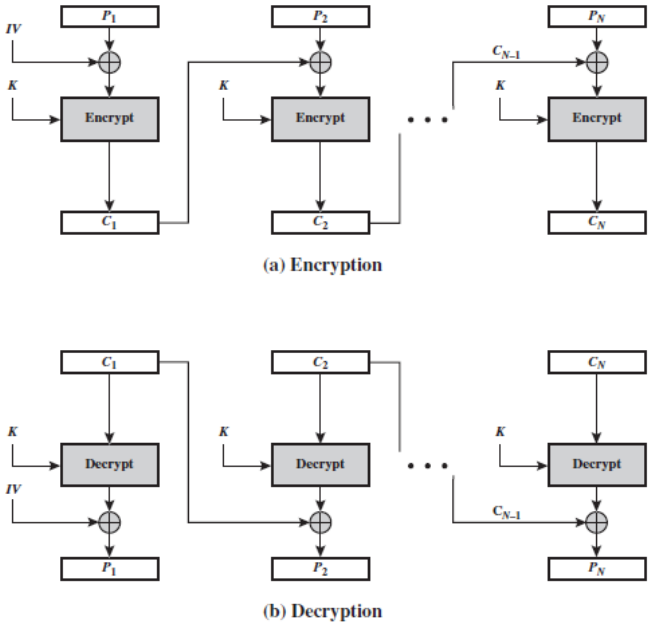


(a) Encryption

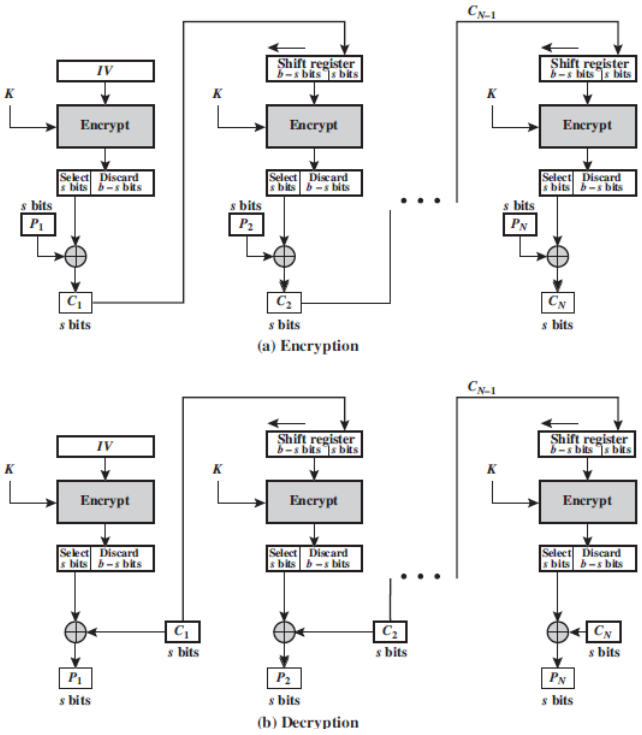


(b) Decryption

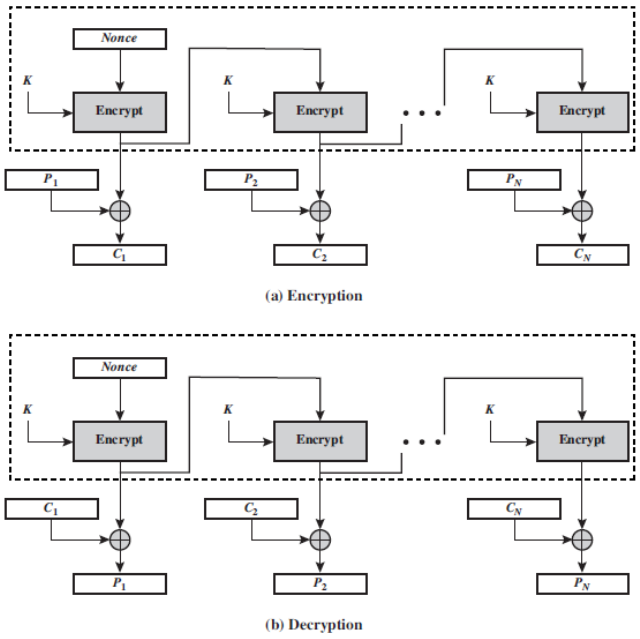
Cipher Block Chaining Mode



Cipher Feedback Mode



Output Feedback Mode



Counter Mode

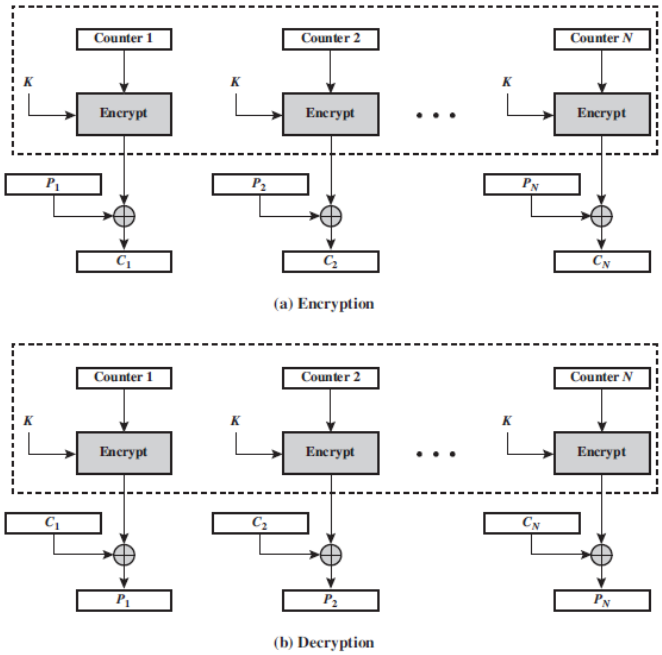


Figure 6.7 Counter (CTR) Mode

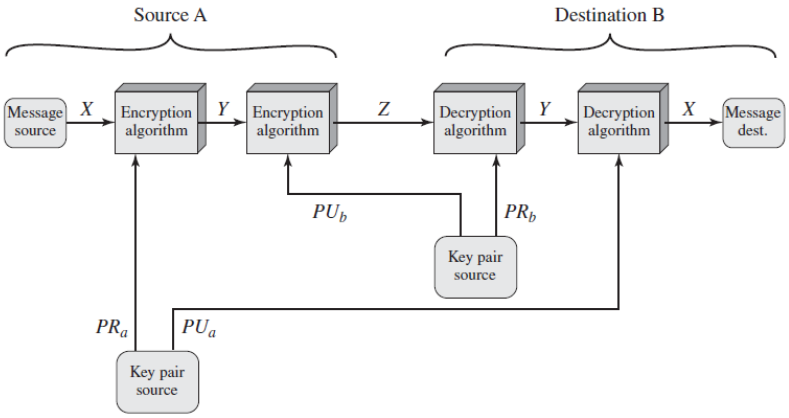
Chapter 9 – Public-Key Cryptography and RSA

Requirements of public Key Encryption

1. It is computationally easy for a party B to generate a pair (public key PU_b, private key PR_b).
2. It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext: $C = E(PU_b, M)$
3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message: $M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$
4. It is computationally infeasible for an adversary, knowing the public key, PU_b, to determine the private key, PR_b.

5. It is computationally infeasible for an adversary, knowing the public key, PU_b, and a ciphertext, C, to recover the original message. We can add a sixth requirement that, although useful, is not necessary for all public-key applications:
6. The two keys can be applied in either order: $M = D[PU_b, E(PR_b,M)] = D[PR_b, E(PU_b,M)]$

Authentication and Secrecy



RSA

Key Generation Alice

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calcuate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d = e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Public Key

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Security of RSA

Four possible approaches to attacking the RSA algorithm are

- **Brute force:** This involves trying all possible private keys.
- **Mathematical attacks:** There are several approaches, all equivalent in effort to factoring the product of two primes.
- **Timing attacks:** These depend on the running time of the decryption algorithm.
- **Chosen ciphertext attacks:** This type of attack exploits properties of the RSA algorithm.

Chapter 10 – Other Public-Key Cryptosystems

Diffie-Hellman Key exchange algorithm

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{XA} \bmod q$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{XB} \bmod q$

Calculation of Secret Key by User A
$K = (Y_B)^{XA} \bmod q$

Calculation of Secret Key by User B
$K = (Y_A)^{XB} \bmod q$

ElGamal Cryptographic System

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

Key Generation by Alice	
Select private X_A	$X_A < q - 1$
Calculate Y_A	$Y_A = \alpha^{XA} \bmod q$
Public key	$PU = \{q, \alpha, Y_A\}$
Private key	X_A

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < q$
Select random integer k	$k < q$
Calculate K	$K = (Y_A)^k \bmod q$
Calculate C_1	$C_1 = \alpha^k \bmod q$
Calculate C_2	$C_2 = KM \bmod q$
Ciphertext:	(C_1, C_2)

Decryption by Alice with Alice's Private Key	
Ciphertext:	(C_1, C_2)
Calculate K	$K = (C_1)^{XA} \bmod q$
Plaintext:	$M = (C_2 K^{-1}) \bmod q$

Diffie-Hellman Man in the middle attack

1. Darth prepares by creating two private / public keys
2. Alice transmits her public key to Bob
3. Darth intercepts this and transmits his first public key to Bob. Darth also calculates a shared key with Alice
4. Bob receives the public key and calculates the shared key (with Darth instead of Alice)
5. Bob transmits his public key to Alice
6. Darth intercepts this and transmits his second public key to Alice. Darth calculates a shared key with Bob
7. Alice receives the key and calculates the shared key (with Darth instead of Bob)
8. Darth can then intercept, decrypt, re-encrypt, forward all messages between Alice & Bob

Chapter 11 – Cryptographic Hash Functions

Requirements of a strong hash function

1. **Variable input** size H can be applied to a block of data of any size.
2. **Fixed output** size H produces a fixed-length output.
3. **Efficiency** H(x) is relatively easy to compute for any given, making both hardware and software implementations practical.
4. **Preimage resistant (one-way property)** For any given hash value, it is computationally infeasible to find y such that H(y) = h.
5. **Second preimage resistant** (weak collision resistant) for any given block, it is computationally infeasible to find y!= x with H(y) = H(x).
6. **Collision resistant (strong collision resistant)** It is computationally infeasible to find any pair (x, y) such that H(x) = H(y).
7. **Pseudorandomness** Output of H meets standard tests for pseudorandomness.

Chapter 12 – Message Authentication Codes

Message Authentication Requirements:

In the context of communications across a network, the following attacks can be identified.

1. **Masquerade**: Insertion of messages into the network from a fraudulent source. This includes the creation of messages by an opponent that are purported to come from an authorized entity. Also included are fraudulent acknowledgments of message receipt or non receipt by someone other than the message recipient.
2. **Content modification**: Changes to the contents of a message, including insertion, deletion, transposition, and modification.
3. **Sequence modification**: Any modification to a sequence of messages between parties, including insertion, deletion, and reordering.
4. **Timing modification**: Delay or replay of messages. In a connection-oriented application, an entire session or sequence of messages could be a replay of some previous valid session, or individual messages in the sequence could be delayed or replayed. In a connectionless application, an individual message (e.g., datagram) could be delayed or replayed.

Ways of doing Message Authentication Functions:

- **Hash function:** A function that maps a message of any length into a fixedlength hash value, which serves as the authenticator
- **Message encryption:** The ciphertext of the entire message serves as its authenticator
- **Message authentication code (MAC):** A function of the message and a secret key that produces a fixed-length value that serves as the authenticator

HMAC Design Objectives

- To use, without modifications, available hash functions. In particular, to use hash functions that perform well in software and for which code is freely and widely available.
- To allow for easy replaceability of the embedded hash function in case faster or more secure hash functions are found or required.
- To preserve the original performance of the hash function without incurring a significant degradation.
- To use and handle keys in a simple way.
- To have a well understood cryptographic analysis of the strength of the authentication mechanism based on reasonable assumptions about the embedded hash function.

HMAC Algorithm

H = embedded hash function (e.g., MD5, SHA-1, RIPEMD-160)
IV = initial value input to hash function
M = message input to HMAC (including the padding specified in the embedded hash function)
Y_i = i^{th} block of M, 0 < = i <= (L – 1)
L =number of blocks in M
b = number of bits in a block
n = length of hash code produced by embedded hash function
K = secret key; recommended length is n; if key length is greater than b,the key is input to the hash function to produce an n-bit key
K^+ = K padded with zeros on the left so that the result is b bits in length
ipad = 00110110 (36 in hexadecimal) repeated b/8 times
opad = 01011100 (5C in hexadecimal) repeated b/8 times

$$\text{HMAC}(K, M) = \text{H}[(K^+ \oplus \text{opad}) \| \text{H}[(K^+ \oplus \text{ipac}$$

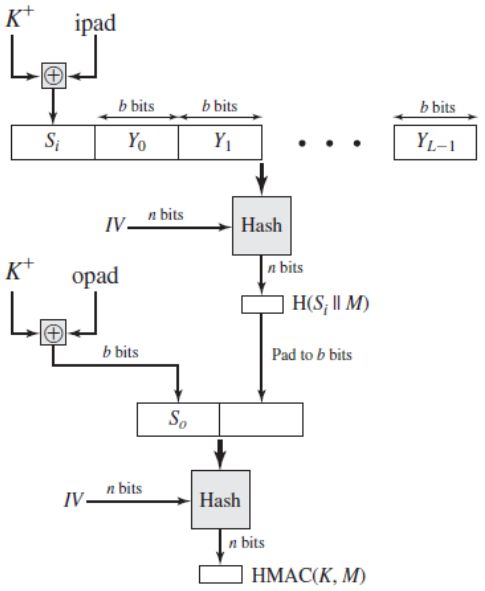
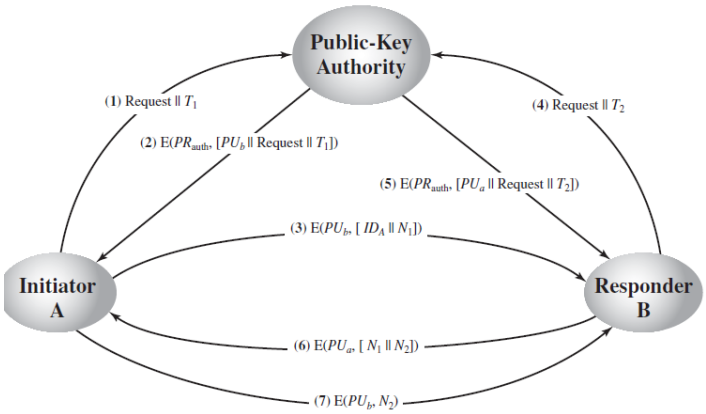
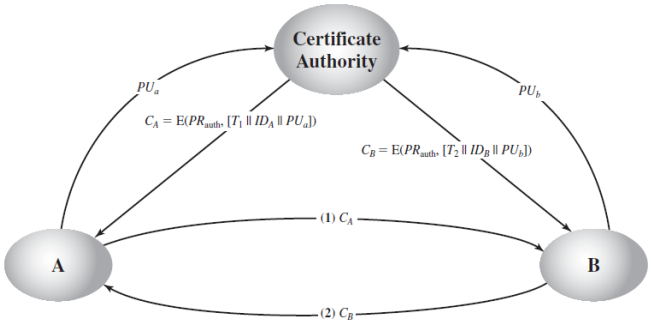


Figure 12.5 HMAC Structure

Chapter 13 – Digital Signatures

Chapter 14 – Key Management and Distribution





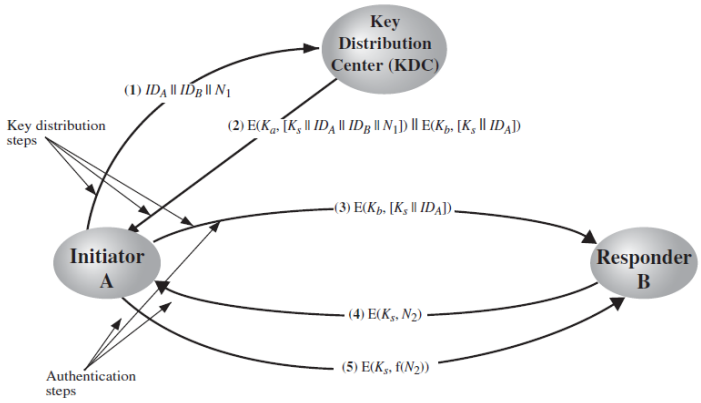


Figure 14.3 Key Distribution Scenario

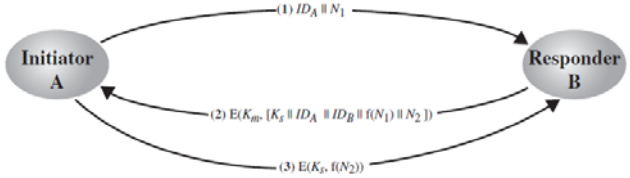


Figure 14.5 Decentralized Key Distribution

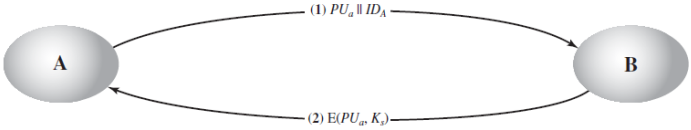


Figure 14.7 Simple Use of Public-Key Encryption to Establish a Session Key

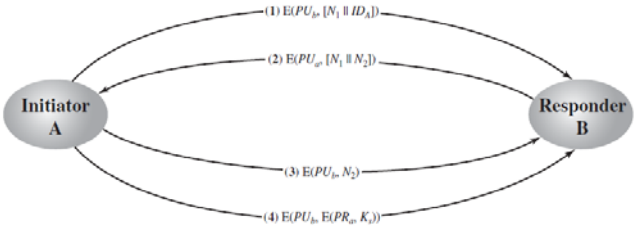


Figure 14.8 Public-Key Distribution of Secret Keys

Chapter 15 – User Authentication

Needham and Schroeder Equations:

1. A → KDC: $ID_A \| ID_B \| N_1$
2. KDC → A: $E(K_a, [K_s \| ID_B \| N_1 \| E(K_b, [K_s \| ID_A])])$
3. A → B: $E(K_b, [K_s \| ID_A])$
4. B → A: $E(K_s, N_2)$
5. A → B: $E(K_s, f(N_2))$

Denning Equations:

1. $A \rightarrow KDC: \quad ID_A \| ID_B$
2. $KDC \rightarrow A: \quad E(K_a, [K_s \| ID_B \| T \| E(K_b, [K_s \| ID_A \| T])])$
3. $A \rightarrow B: \quad E(K_b, [K_s \| ID_A \| T])$
4. $B \rightarrow A: \quad E(K_s, N_1)$
5. $A \rightarrow B: \quad E(K_s, f(N_1))$

Chapter 16 – Transport Level Security

Difference between the SSL session and the SSL connection

- Connection: A connection is a transport (in the OSI layering model definition) that provides a suitable type of service. For SSL, such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.
- Session: An SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol. Sessions define a set of cryptographic

SSL Features that prevent the following attacks

1. **Brute Force Cryptanalytic Attack**: The conventional encryption algorithms use key lengths ranging from 40 to 168 bits.
2. **Known Plaintext Dictionary Attack**: SSL protects against this attack by not really using a 40-bit key, but an effective key of 128 bits. The rest of the key is constructed from data that is disclosed in the Hello messages. As a result the dictionary must be long enough to accommodate 2128 entries.
3. **Replay Attack**: This is prevented by the use of nonces.
4. **Man-in-the-Middle Attack**: This is prevented by the use of pubic-key certificates to authenticate the correspondents.
5. **Password Sniffing**: User data is encrypted.
6. **IP Spoofing:** The spoofer must be in possession of the secret key as well as the forged IP address..
7. **IP Hijacking**: Again, encryption protects against this attack.
8. **SYN Flooding**: SSL provides no protection against this attack

Chapter 20 – Intruders

Types of Intruders

- **Masquerader**: An individual who is not authorized to use the computer and who penetrates a system's access controls to exploit a legitimate user's account
- **Misfeasor**: A legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuses his or her privileges
- **Clandestine user**: An individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection

Difference Between rule based intrusion detection and statistical intrusion detection

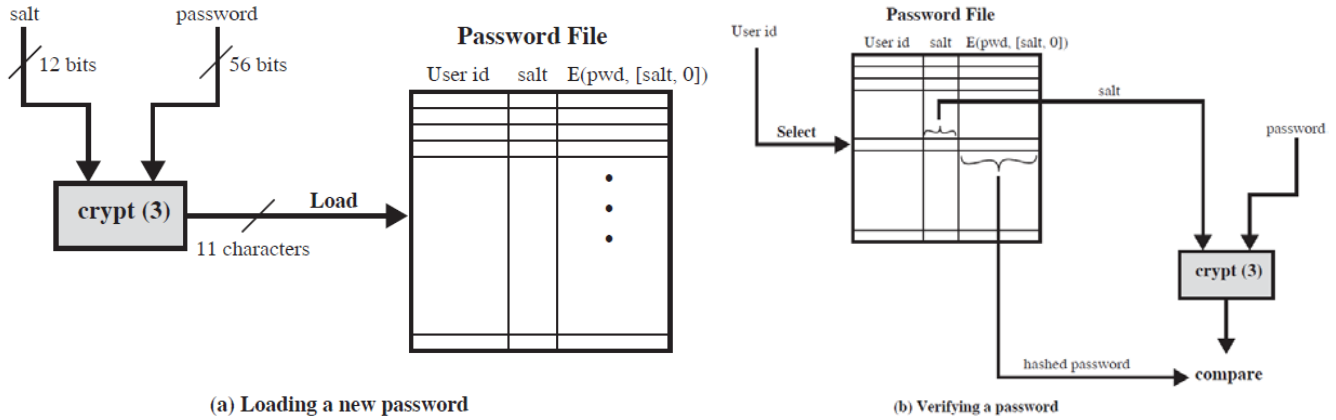
1. **Statistical anomaly detection**: Involves the collection of data relating to the behaviour of legitimate users over a period of time. Then statistical tests are applied to observed behaviour to determine with a high level of confidence whether that behaviour is not legitimate user behaviour.
    a. **Threshold detection**: This approach involves defining thresholds, independent of user, for the frequency of occurrence of various events.
    b. **Profile based**: A profile of the activity of each user is developed and used to detect changes in the behaviour of individual accounts.
2. **Rule-based detection**: Involves an attempt to define a set of rules that can be used to decide that a given behaviour is that of an intruder.
    a. **Anomaly detection**: Rules are developed to detect deviation from previous usage patterns.

b. **Penetration identification**: An expert system approach that searches for suspicious behavior.

Password Scheme used in UNIX CRYPT(3)

Each user selects a password of up to eight printable characters in length. This is converted into a 56-bit value (using 7-bit ASCII) that serves as the key input to an encryption routine. The encryption routine, known as crypt(3), is based on DES.The DES algorithm is modified using a 12-bit "salt" value.
Typically, this value is related to the time at which the password is assigned to the user. The modified DES algorithm is exercised with a data input consisting of a 64-bit block of zeros. The output of the algorithm then serves as input for a second encryption. This process is repeated for a total of 25 encryptions. The resulting 64-bit output is then translated into an 11-character sequence. The hashed password is then stored, together with a plaintext copy of the salt, in the password file for the corresponding user ID. This method has been shown to be secure against a variety of cryptanalytic attacks [WAGN00].
The salt serves three purposes:

- It prevents duplicate passwords from being visible in the password file. Even if two users choose the same password, those passwords will be assigned at different times. Hence, the "extended" passwords of the two users will differ.
- It effectively increases the length of the password without requiring the user to remember two additional characters. Hence, the number of possible passwords is increased by a factor of 4096, increasing the difficulty of guessing a password.
- It prevents the use of a hardware implementation of DES, which would ease the difficulty of a brute-force guessing attack.



(a) Loading a new password

(b) Verifying a password

Protection of password files

The password file can be protected in one of two ways:

- **One-way function**: The system stores only the value of a function based on the user's password. When the user presents a password, the system transforms that password and compares it with the stored value. In practice, the system usually performs a one-way transformation (not reversible) in which the password is used to generate a key for the one-way function and in which a fixed-length output is produced.
- **Access control**: Access to the password file is limited to one or a very few accounts.

Password Selection Strategies

- **User education**: Users can be told the importance of using hard-to-guess passwords and can be provided with guidelines for selecting strong passwords. This user education strategy is unlikely to succeed at most installations, particularly where there is a large user population or a lot of turnover. Many users will simply ignore the guidelines. Others may not be good judges of what is a strong password.
- **Computer-generated passwords**: also have problems. If the passwords are quite random in nature, users will not be able to remember them.
- **Reactive password checking**: is one in which the system periodically runs its own password cracker to find guessable passwords. It is resource intensive if the job is done right. Because a determined opponent who is able to steal a password file can devote full CPU time to the task for hours or even days, an effective reactive password checker is at a distinct disadvantage. Furthermore, any existing passwords remain vulnerable until the reactive password checker finds them.
- **Proactive password checking**: In this scheme, a user is allowed to select his or her own password. However, at the time of selection, the system checks to see if the password is allowable and, if not, rejects it.

- Rule enforcement
- Large Dictionary of bad words
- Markov

Chapter 21 – Malicious Software

Phases of worms and viruses:

- **Dormant phase**: The virus is idle. The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit. Not all viruses have this stage.
- **Propagation phase**: The virus places a copy of itself into other programs or into certain system areas on the disk. The copy may not be identical to the propagating version; viruses often morph to evade detection. Each infected program will now contain a clone of the virus,which will itself enter a propagation phase.
- **Triggering phase**: The virus is activated to perform the function for which it was intended. As with the dormant phase, the triggering phase can be caused by a variety of system events, including a count of the number of times that this copy of the virus has made copies of itself.
- **Execution phase**: The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files.

Difference Between worm and virus

A worm is a program that can replicate itself and send copies from computer to computer across network connections. A computer virus is a piece of software that can “infect” other programs by modifying them; the modification includes injecting the original program with a routine to make copies of the virus program,

What are the functions performed by the propagation phase of a worm?

1. Search for other systems to infect by examining host tables or similar repositories of remote system addresses.
2. Establish a connection with a remote system.
3. Copy itself to the remote system and cause the copy to be run.