19CSE311 - COMPUTER SECURITY
UNIT-2- PART 4
Key Management & Distribution

February 14, 2022

Dr.Remya S/ Mr. Sarath R
Assistant Professor
Department of Computer Science

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY
School of Engineering

Key Management

- The main aim of key management is to generate a secret key between two parties and store it to prove the authenticity between communicating users.
- Key management is the technique which support key generation, storage and maintenance of the key between authorized users
- It plays an important role in cryptography as the basis of securing goals like confidentiality, authentication, data integrity and digital signatures
- It is not the case where the communication parties are using the same key for encryption and decryption or whether two different keys are used for encryption and decryption
- Basic purpose of key management is: key generation, key distribution, controlling the use of keys etc

- Key Distribution
  - Symmetric Key Distribution
    - Symmetric encryption
    - Asymmetric encryption
  - Public Key Distribution
    - Public Announcement of Public Keys
    - Publicly Available Directory
    - Public-Key Authority
    - Public-Key Certificates

Symmetric Key Distribution using symmetric encryption

- When two parties share the same key (i.e. symmetric key) that protect from access by others, the process between two parties that exchanges that key called as symmetric key distribution.
- For two parties A and B, key distribution can be achieved in a number of ways, as follows:
  1. A can select a key and physically deliver it to B.
  2. A third party can select the key and physically deliver it to A and B.
  3. If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
  4. If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B.

- Options 1 and 2 call for manual delivery of a key.
- But it is difficult in a wide-area distributed system
- Option 3 is a possibility for either link encryption or end to end encryption.
- But if an attacker ever succeeds in gaining access to one key, all subsequent keys will be revealed
- For end-to-end encryption variation of option 4 has been adopted
- Here a key distribution centre is responsible for distributing the keys to pair of users as needed
- Each user must share a unique key with the distribution centre for the process of key distribution

- The use of a key distribution center is based on the use of a hierarchy of keys.
- At a minimum, two levels of keys are used. Communication between end systems is encrypted using a temporary key, often referred to as a session key.
- Typically, the session key is used for the duration of a logical connection, such as a frame relay connection or transport connection, and then discarded.
- Each session key is obtained from the key distribution center over the same networking facilities used for end-user communication.
- Accordingly, session keys are transmitted in encrypted form, using a master key that is shared by the key distribution center and an end system or user.

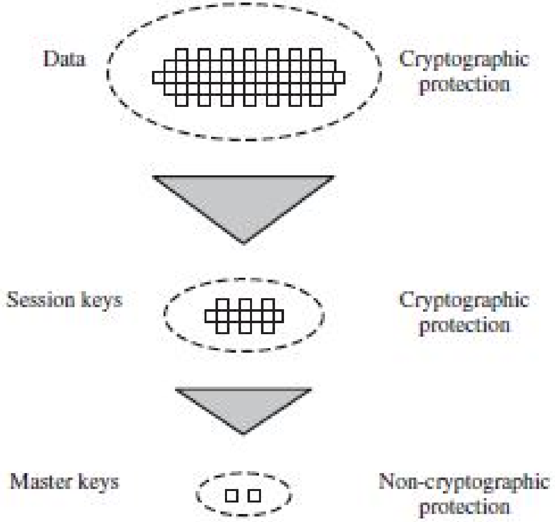


Figure 14.2 The Use of a Key Hierarchy

Key Distribution Scenario

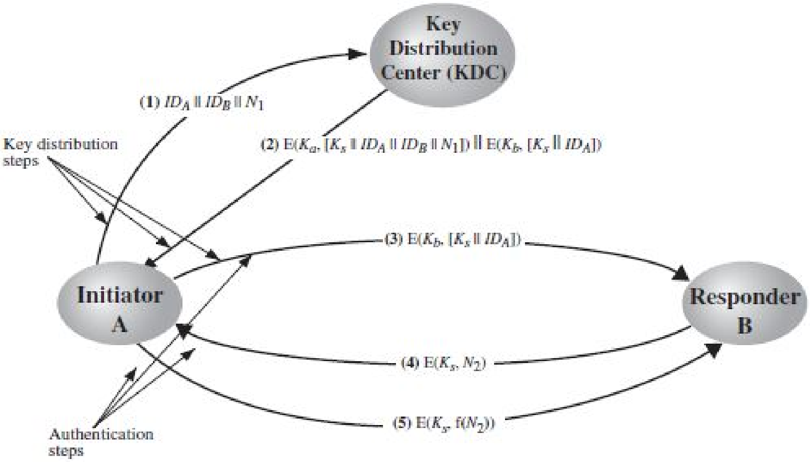
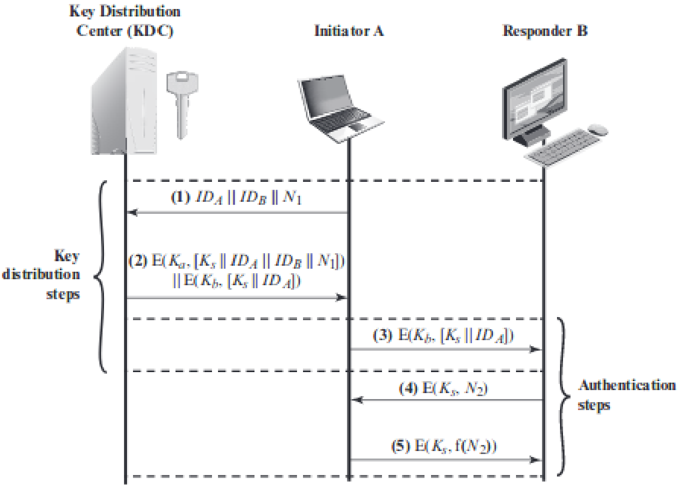


Figure 14.3 Key Distribution Scenario

Key Distribution Scenario



- The scenario assumes that each user shares a unique master key with the key distribution center (KDC).
- Let us assume that user A wishes to establish a logical connection with B and requires a one-time session key to protect the data transmitted over the connection.
- A has a master key K_a, known only to itself and the KDC; similarly, B shares the master key K_b with the KDC.

Steps

1. A issues a request to the KDC for a session key to protect a logical connection to B.
   $ID_A||ID_B||N_1$
   - The message includes the identity of A and B and a unique identifier, N1, for this transaction, which we refer to as a nonce.
   - The nonce may be a time- stamp, a counter, or a random number; the minimum requirement is that it differs with each request
   - To prevent masquerade, it should be difficult for an opponent to guess the nonce. Thus, a random number is a good choice for a nonce.

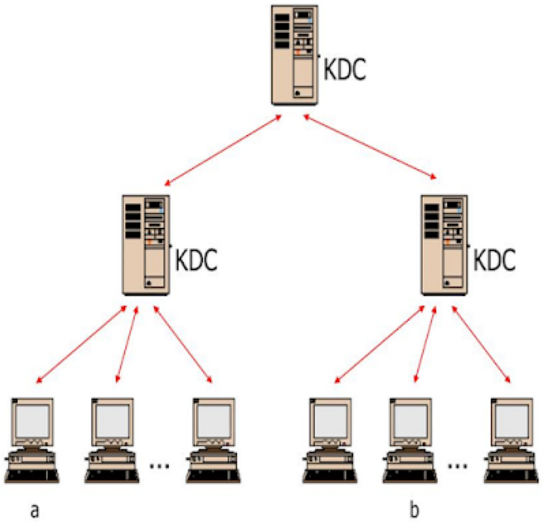
2. The KDC responds with a message encrypted using K_a. Thus, A is the only one who can successfully read the message, and A knows that it originated at the KDC.
$E(K_a, [K_s||ID_A||ID_B||N_1])||E(K_b, [K_s||ID_A])$

- The message includes two items intended for A: The one-time session key, Ks, to be used for the session. The original request message, including the nonce, to enable A to match this response with the appropriate request.
- Thus, A can verify that its original request was not altered before reception by the KDC and, because of the nonce.
- In addition, the message includes two items intended for B: The one-time session key, K_s, to be used for the session. An identifier of A (e.g., its network address), ID_A These last two items are encrypted with Kb(the master key that the KDC shares with B).
- They are to be sent to B to establish the connection and prove A's identity.

3. A stores the session key for use in the upcoming session and forwards to B the information that originated at the KDC for B, namely, $E(K_b, [K_s||ID_A])$.
   - Because this information is encrypted with Kb, it is protected from eavesdropping.
   - B now knows the session key (K_s), knows that the other party is A (from ID_A), and knows that the information originated at the KDC (because it is encrypted using K_b).
   - At this point, a session key has been securely delivered to A and B, and they may begin their protected exchange.
4. Using the newly generated session key for encryption, B sends a nonce, N2, to A. $E(Ks, N_2)$
5. Also, using Ks, A responds with f (N2), where f is a function that performs some transformation on N2. $E(Ks, f(N2))$.

- There are 4 different methods are used:
  1. Hierarchical Key Control
  2. Session key life time
  3. A transparent key control scheme
  4. Decentralized key control.

Hierarchical Key Control

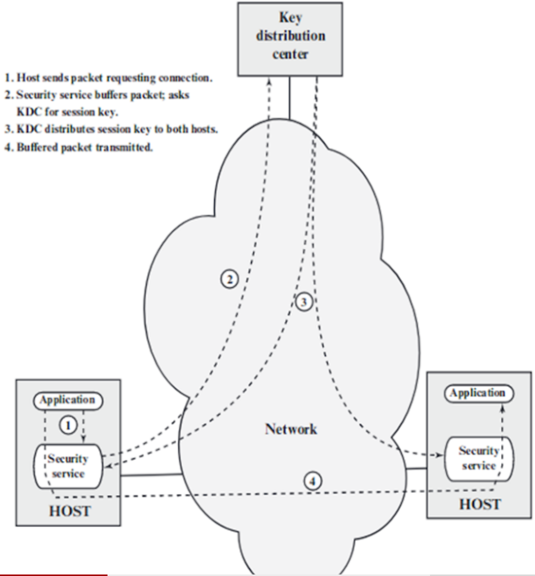


- It is not necessary to limit the key distribution function to a single KDC.
- Indeed, for very large networks, single KDC is not enough to distribute keys among all users. As an alternative, a hierarchy of KDCs can be established.
- For example, there can be local KDCs, each responsible for a small domain of the overall internetwork, such as a single LAN or a single building.
- For communication among entities within the same local domain, the local KDC is responsible for key distribution.
- If two entities in different domains desire a shared key, then the corresponding local KDCs can communicate through a global KDC.
- In this case, any one of the three KDCs involved can actually select the key.
- The hierarchical concept can be extended to three or even more layers, depending on the size of the number of users and the geographic scope of the internetwork.

Session key life time

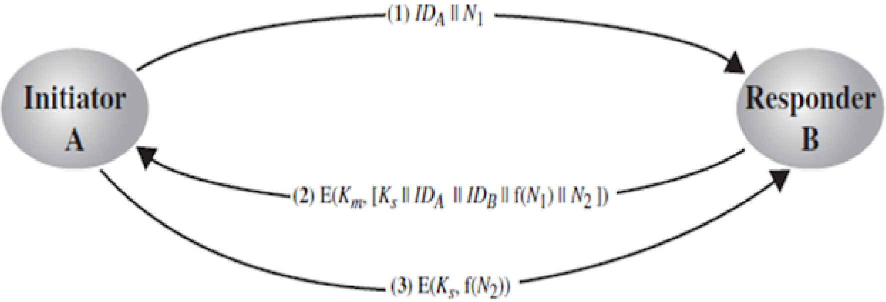
- The more frequently session keys are exchanged, the more secure they are, because the attacker has to capture session key every time to decrypt cipher text.
- Short session key life time: Key exchange frequently & more secure.
- Long session key life time: Reduce Key exchange time & less network bandwidth used.
- For connection-oriented protocols, new session key established for each new connection. Then Update key periodically, if the connection has long time.
- For connection less protocols, not to use a new key for each session but use a given session key for a fixed period of time.

A transparent key control scheme



- The SSM (Session security module) saves that packet and applies to the KDC for permission to establish the connection (step 2).
- The communication between the SSM and the KDC is encrypted using a master key shared only by this SSM and the KDC.
- If the KDC approves the connection request, it generates the session key and delivers it to the two appropriate SSMs, using a unique permanent key for each SSM (step 3).
- The requesting SSM can now release the connection request packet, and a connection is set up between the two end systems (step 4).
- All user data exchanged between the two end systems are encrypted by their respective SSMs using the onetime session key.

Decentralized Key Control



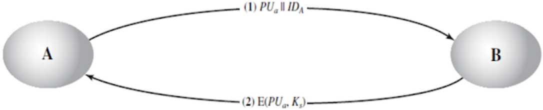
1. A issues a request to B for a session key and includes a nonce, N_1.
2. B responds with a message that is encrypted using the shared master key. The response includes the session key selected by B, an identifier of B, the value $f(N_1)$, and another nonce, N_2.
3. Using the new session key, A returns $f(N_2)$ to B.

Symmetric Key Distribution using Asymmetric Encryption - Secret Key Distribution using Asymmetric Encryption

- There are two approaches:
  1. Simple Secret Key Distribution
  2. Secret key Distribution with Confidentiality and Authentication.

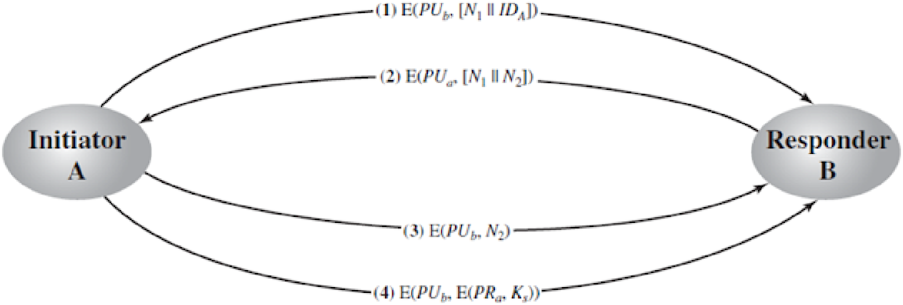
Simple Secret Key Distribution

- If A wishes to communicate with B, the following procedure is employed:
- A generates a public/private key pair PU_a, PR_a and transmits a message to B consisting of PU_a and an identifier of A, ID_A. B generates a secret key, K_s, and transmits it to A, which is encrypted with A's public key.



- A decrypt message using, $D(PRa, E(PUa, Ks))$ to recover the secret key. Because only A can decrypt the message, only A and B will know the identity of K_s.
- A discards $PU_a and PR_a$ and B discards PU_a.
- A and B can now securely communicate using conventional encryption and the session key K_s. At the completion of the exchange, both A and B discard K_s

Secret Key Distribution with Confidentiality and Authentication

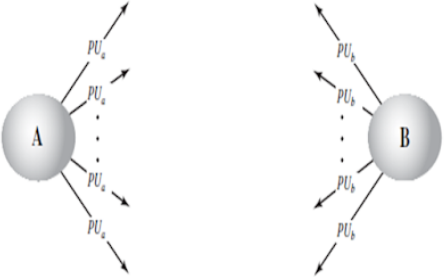


- A uses B's public key to encrypt a message to B containing an identifier of $A(ID_A)$and a nonce (N_1), which is used to identify this transaction uniquely.
- B sends a message to user A encrypted with PU_a and containing A's nonce as (N_1) well as a new nonce generated by $B(N_2)$. Because only B could have decrypted message (1), the presence of N1 in message (2) assures A that the correspondent is B.
- A returns N2, encrypted using B's public key, to assure B that its correspondent is A.
- A selects a secret key and sends $M = E(PU_b, E(PR_a, K_s))$ to B.
- Encryption of this message with B's public key ensures that only B can read it; encryption with A's private key ensures that only A could have sent it.
- B decrypt the message and get secret key K_s.
- The result is that this scheme ensures both confidentiality and authentication in the exchange of a secret key.

Public Key Distribution in Network Security

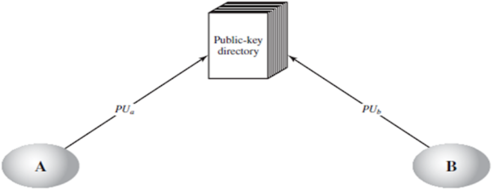
- There are four methods of public key distribution:
  1. Public announcement of Public Keys
  2. Publicly Available Directory
  3. Public Key Authority
  4. Public Key Certificates.

Public Announcement of Public Keys



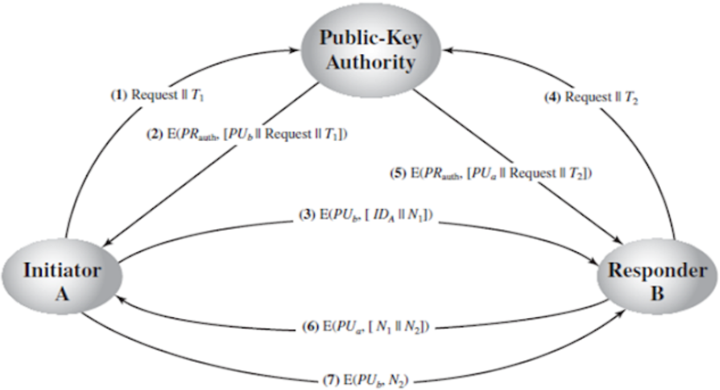
- In a public key cryptography, such as RSA, any user can send his/her key to any other user or broadcast it to the group
- This type of approach is having a biggest drawback. Any user can pretend to be a user A and send a public to another user or broadcast it.
- Until user A has got this thing and alerts to other user, a pretender is able to read all encrypted message of other users.

Publicly Available Directory



- A dynamic publicly available directory is used to achieve the security. Maintenance and distribution of public directory is controlled by a trust entity.
- A trusted entity maintains a directory for each user as $< name, publickey >$. Each user has to register a public key with the directory.A user can replace the existing key with a new one at any time for any particular reason.
- It is more secure than public announcement but still having some weakness. A hacker can obtain the private key of directory or temper with the information kept by directory.

Public-Key Authority



- It gives stronger security. As shown in figure a central authority keeps a dynamic directory of public keys of all users.
- Additionally, each user knows the public key of authority.

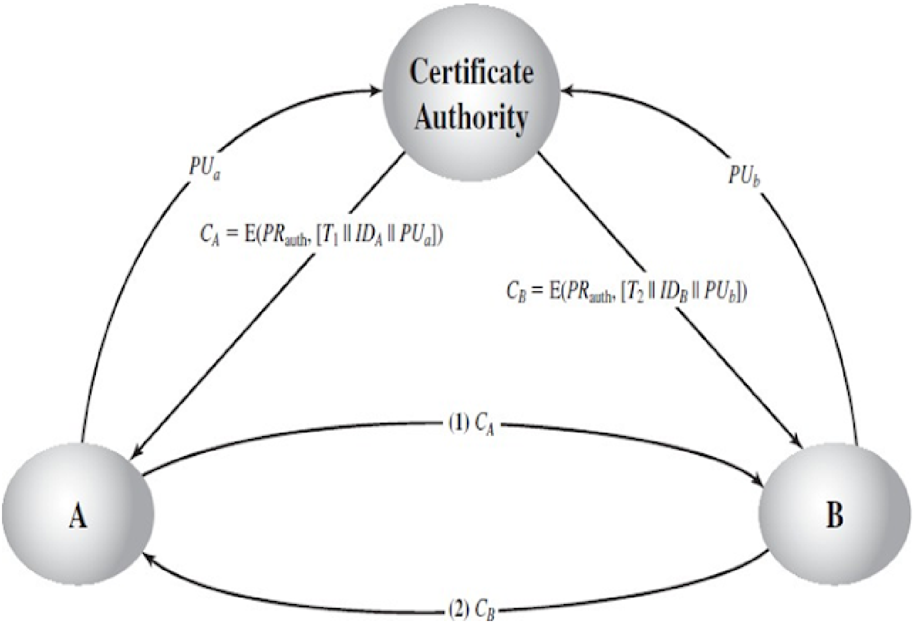
Steps

1. A sends a time stamped message to the public-key authority containing a request for the current public key of B.
2. The authority responds with a message that is encrypted using the authority's private key, PR_{auth}. Thus, A is able to decrypt the message using the authority's public key. Therefore, A is assured that the message originated with the authority.
   - The message includes the following: B's public key, PU_b, which A can use to encrypt messages destined for B.
   - The original request used to enable A to match this response with the corresponding earlier request and to verify that the original request was not altered before reception by the authority.
   - The original timestamp given so A can determine that this is not an old message from the authority containing a key other than B's current public key.

3. A stores B's public key and also uses it to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1), which is used to identify this transaction uniquely.
4. &
5. B retrieves A's public key from the authority in the same manner as A retrieved B's public key.
6. B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B (N_2). Because only B could have decrypted message (3), the presence of in message (6) assures A that the correspondent is B.
7. A returns N_2, which is encrypted using B's public key, to assure B that its correspondent is A.

Public-Key Certificates

- directory of names and public keys maintained by the authority is vulnerable to tampering.
- An alternative approach, first suggested by Kohn Felder, is to use certificates.
- In essence, a certificate consists of a public key, an identifier of the key owner, and the whole block signed by a trusted third party. Typically, the third party is a certificate authority, such as a government agency or a financial institution that is trusted by the user community.
- A user can present his or her public key to the authority in a secure manner and obtain a certificate. The user can then publish the certificate.
- Anyone needing this user's public key can obtain the certificate and verify that it is valid by way of the attached trusted signature.
- A participant can also convey its key information to another by transmitting its certificate. Other participants can verify that the certificate was created by the authority.


Certificate Authority
PU_a
PU_b
$C_A = E(PR_{auth}, [T_1 \| ID_A \| PU_a])$
$C_B = E(PR_{auth}, [T_2 \| ID_B \| PU_b])$
(1) C_A
(2) C_B
A
B

Requirements

1. Any participant can read a certificate to determine the name and public key of the certificate's owner.
2. Any participant can verify that the certificate originated from the certificate authority and is not counterfeit.
3. Only the certificate authority can create and update certificates.
4. Any participant can verify the certificate.