Cryptography

Cunsheng DING, HKUST

COMP3631

Lecture 14: Two Key Establishment Protocols

Outline of This Lecture

1. Recall the passive and active attacks.
2. Recall the discrete logarithm problem.
3. Study the Diffie-Hellman key exchange protocol.
4. Study the digital envelop key distribution protocol.

Part I: Passive and Active Attacks

A Recall of Passive and Active Attacks

Passive attacks: Any attack on a security system under the assumption that the attacker can only intercept texts exchanged over a communication channel is called a **passive attack**.

Active attacks: Any attack on a security system under the assumption that the attacker can stop, intercept, delete, modify, and replay texts exchanged over a communication channel or insert his/her texts into the channel is called an **active attack**. In such a scenario, we say that the attacker has **full control** over the communication channel.

Remark: By definition, active attacks are more powerful than passive attacks.

Remark: A system may be secure with respect to passive attacks, but insecure with respect to active attacks. For example, the DH protocol covered today is such a protocol.

Part II: Key Distribution and Key Agreement

Key Establishment: Necessity

- For symmetric encryption, the two parties must share the same key.
- The key must be protected from access by others.
- The key should be changed regularly (an adversary or enemy may learn the key in some way).

Key distribution: One party generates a key and delivers it to others in a secure way. Today we will see a key distribution protocol.

Key agreement: The communication parties first exchange key materials and then compute a common key with their exchanged key materials. Such a key is not determined by a single party and has contributions from more than one parties. Today we will see a key agreement protocol.

Part III: The Diffie-Hellman Key Exchange Protocol

A Key Agreement Protocol

A Recall of Primitive Roots of a Prime p

Primitive root: Let p be a prime. A positive integer α is called a **primitive root** of p if each nonzero element $a \in \mathbf{Z}_p$ can be expressed as

$$a = \alpha^i \bmod p$$

for a unique integer i with $0 \leq i \leq p-2$.

Attention: The concept of primitive roots of a prime was introduced in Lecture 12.

A Recall of the Discrete Logarithm Problem

The discrete logarithm: Let p be a prime and α be a **primitive root** of p. By definition, each nonzero element $a \in \mathbf{Z}_p$ can be expressed as

$$a = \alpha^i \bmod p$$

for a unique integer i with $0 \leq i \leq p-2$.

The exponent i is referred to as the **discrete logarithm**, or **index**, of a for the base α, and is denoted $\log_\alpha a$ or $\mathrm{ind}_\alpha(a)$.

The discrete logarithm problem is the computation of the unique exponent i, given a, α and p. This problem was studied in Lecture 12 about the ElGamal public key cipher.

The Diffie-Hellman Key Exchange Protocol

User A		User B
Generate random $X_A < p$		Generate random $X_B < p$
calculate $Y_A = \alpha^{X_A} \bmod p$	$Y_A \longrightarrow$	Calculate $Y_B = \alpha^{X_B} \bmod p$
	$\longleftarrow Y_B$	
Calculate $k = (Y_B)^{X_A} \bmod p$		Calculate $k = (Y_A)^{X_B} \bmod p$

The Diffie-Hellman Key Exchange Protocol

- Both p and α are publicly known. But X_A and X_B must be kept confidential to A and B, respectively.
- $k = \alpha^{X_A X_B} \bmod p$ is the established common value by A and B.
- Certain bits in the binary representation of k can be used as a secret key of a one-key cipher for encrypting messages between A and B.
- In some real-world security systems, the established key k is used as a master key or seed key for computing encryption keys and authentication keys.
- Certain bits in the binary representation of k can be used as an initial value of a one-key block cipher in the CBC mode or the Counter mode for encrypting messages between A and B.

Security of the Diffie-Hellman Key Exchange Protocol

The security of the protocol with respect to passive attacks:
Both p and α are publicly known.

- An adversary C finds X_A by solving the equation $Y_A = \alpha^{X_A} \bmod p$ and then computes $k = (Y_B)^{X_A} \bmod p$. This is to solve the discrete logarithm problem (see Lecture 12).
- An adversary C finds X_B by solving the equation $Y_B = \alpha^{X_B} \bmod p$ and then computes $k = (Y_A)^{X_B} \bmod p$. This is to solve the discrete logarithm problem (see Lecture 12).
- The security with respect to **passive attacks** depends on the difficulty of solving the discrete logarithm problem, which is believed to be very hard in general.

Security of the Diffie-Hellman Key Exchange Protocol

The security of the protocol with respect to active attacks (I): A man-in-the-middle attack on this protocol goes as follows:

- A $\leftrightarrow$ C $\leftrightarrow$ B. With her Y_C in hand, an adversary C is standing in the middle of the communication channel and waiting for a message.
- When A and B are running the DH protocol, C impersonates B and establishes a common key k_1 with A, then C impersonates A and establishes a common key k_2 with B. Details are below:

$$\text{A} \rightarrow Y_A \rightarrow \text{C} \rightarrow Y_C \rightarrow \text{B}$$

$$\text{A} \leftarrow Y_C \leftarrow \text{C} \leftarrow Y_B \leftarrow \text{B}$$

Then A and C compute a common key k_1 with Y_A and Y_C, but A thinks that she has established k_1 with B. B and C compute a common key k_2, but B thinks that he has established k_2 with A.

Security of the Diffie-Hellman Key Exchange Protocol

The security of the protocol with respect to active attacks (II): The consequence of the man-in-the-middle attack on this protocol:

- A $\leftrightarrow$ C $\leftrightarrow$ B. After establishing k_1 with A and k_2 with B, the adversary C is standing in the middle of the communication channel and waiting for an encrypted message.
- Later, when A and B are exchanging encrypted data, C can use k_1 and k_2 to decrypt and encrypt their data as follows:

$$\text{A} \rightarrow E_{k_1}(m) \rightarrow \text{C} \rightarrow E_{k_2}(m) \rightarrow \text{B}$$

$$\text{A} \leftarrow E_{k_1}(m') \leftarrow \text{C} \leftarrow E_{k_2}(m') \leftarrow \text{B}$$

In this way, C can recover every message exchanged between A and B.

Security of the Diffie-Hellman Key Exchange Protocol

The security of the protocol with respect to active attacks:
The man-in-the-middle attack on this protocol:

- Why is the man-in-the-middle attack on this protocol successful?

  —— Pause for discussion!

- How do you modify the DH protocol so that the man-in-the-middle attack cannot work?

  —— Pause for discussion!

Remark: This DH key exchange protocol is widely used in real-world security systems, where a mutual authentication mechanism is added. See Appendix A for such a protocol.

Remark: Partially due to the invention of this protocol, Diffie and Hellman received the Turing award!

Online Demo of the Diffie-Hellman Protocol

In the following URL, you find the demo of the Diffie-Hellman key exchange protocol:

`https://8gwifi.org/DHFunctions.jsp`

Part IV: The Digital Envelop Key Distribution Protocol

The Digital Envelop Key Distribution Protocol

Session key: Two parties want to communicate using a one-key cipher for encryption. They need a session key for each session of communication.

Session key distribution with a public-key cipher: Assume that Alice and Bob have exchanged their public keys.

- Alice generates a session key k and then sends $E_{k_e^{(B)}}(k)||ID_A||ID_B$ to Bob.
- Bob uses his private key $k_d^{(B)}$ to decrypt $E_{k_d^{(B)}}(k)$ and recovers k.

Remark: The encrypted session key $E_{k_e^{(B)}}(k)$ is called a **digital envelope**.

Security of the Digital Envelop Key Distribution Protocol

Question: Is the digital envelop key distribution protocol secure with respect to passive attacks?

Answer: The answer depends on the security of the public key cipher.
The protocol is secure if the public-key cipher is computationally secure.
The protocol is insecure if the public-key cipher is computationally insecure.

Justification: An attacker can only intercept the digital envelop $E_{k_e^{(B)}}(k)$ by the definition of passive attacks.

Security of the Digital Envelop Key Distribution Protocol

Question: Is the digital envelop key distribution protocol secure with respect to active attacks?

Answer: No. Anyone who has knowledge of Bob's public key can impersonate Alice and can establish a session key k with Bob. But Bob thinks that he has established the key k with Alice.

Remark: Such an attack is called an **impersonation attack**. It works because there is no mechanism for Bob to verify the identity of the sender of the digital envelop!

Attention: The data integrity service is not provided by this digital envelop key distribution protocol. This protocol is widely used in real-world security systems, where mutual authentication and data integrity mechanisms are added (see Appendix A for such protocols).

Appendix A: Combining Two Application Protocols

Question: How do you combine an authentication protocol in Lecture 13 with a key establishment protocol in this lecture?

A Combining Method for Two Protocols

The protocol: Alice and Bob share a long-term secret key k for a secure one-key cipher, and have computed their DH values Y_A and Y_B.

$$\text{Alice } \rightarrow E_k(Y_A||Timestamp_1||ID_A||ID_B)||ID_A||ID_B \rightarrow \text{ Bob},$$

$$\text{Alice } \leftarrow ID_A||ID_B||E_k(ID_A||ID_B||Timestamp_2||Y_B) \leftarrow \text{ Bob},$$

where $Timestamp$ is the current time, ID_A and ID_B are the ID of Alice and Bob, respectively. They know the ID of each other.

- It provides the data confidentiality, the data integrity, the antirelay, the data origin authentication and the mutual authentication services.
- It combines a mutual authentication protocols and the DH protocol.
- The common integer computed from Y_A and Y_B may be used as a session key for the one-key cipher or an IV for the CBC mode.

Another Combining Method of Two Protocols

The protocol: Alice and Bob have exchanged their public keys for a secure public-key cipher and Alice has generated a session key or IV denoted by k.

$$\text{Alice } \rightarrow E_{k_e^{(B)}}(N_1)||ID_A||ID_B \rightarrow \text{ Bob},$$

$$\text{Alice } \leftarrow ID_A||ID_B||E_{k_e^{(A)}}(N_1||N_2) \leftarrow \text{ Bob},$$

$$\text{Alice } \rightarrow E_{k_e^{(B)}}(k||N_2)||ID_A||ID_B \rightarrow \text{ Bob},$$

where N_i are nonces, ID_A and ID_B are the ID of Alice and Bob, respectively. They know the ID of each other.

Question: How should these received texts be processed by their receivers? What security services are provided?

Remark: It combines a mutual authentication protocol in Lecture 13 and the digital envelope protocol of this lecture.

Appendix B:

A Key Distribution Protocol with a Key Distribution Center

Students are not required to read this protocol. It is for some students who wish to learn more.

A Key Distribution Protocol

Parties involved: A key distribution center (KDC), a group of people to communicate with each other.

Requirements: Whenever A wants to communicate with B, the KDC should generate a one-time key (called **session key**) and distribute it to A and B. Both confidentiality and authenticity must be achieved.

Remark: The session key (one-time key) is established only for this session of communication between A and B.

A Key Distribution Protocol – Continued

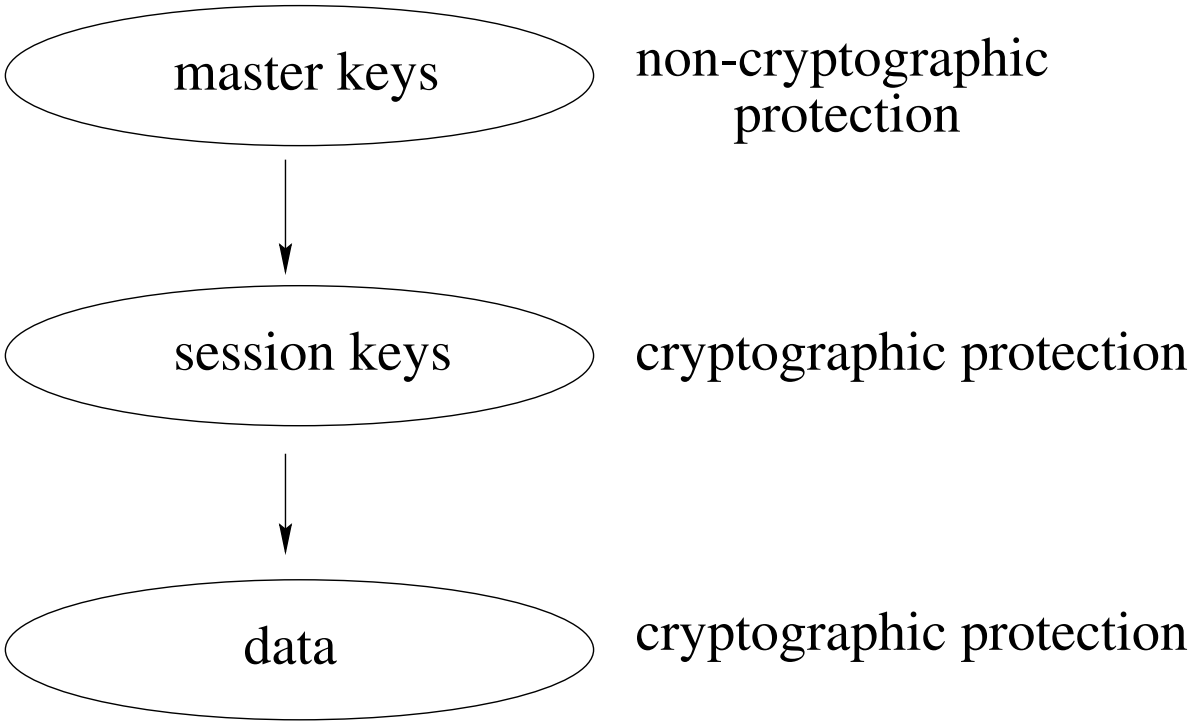
Building blocks needed:

- The KDC and all parties involved in this communication system use a one-key block cipher.
- The KDC and each party A share a secret key k_a, which is called a **master key**.
- The KDC is a trustful party.

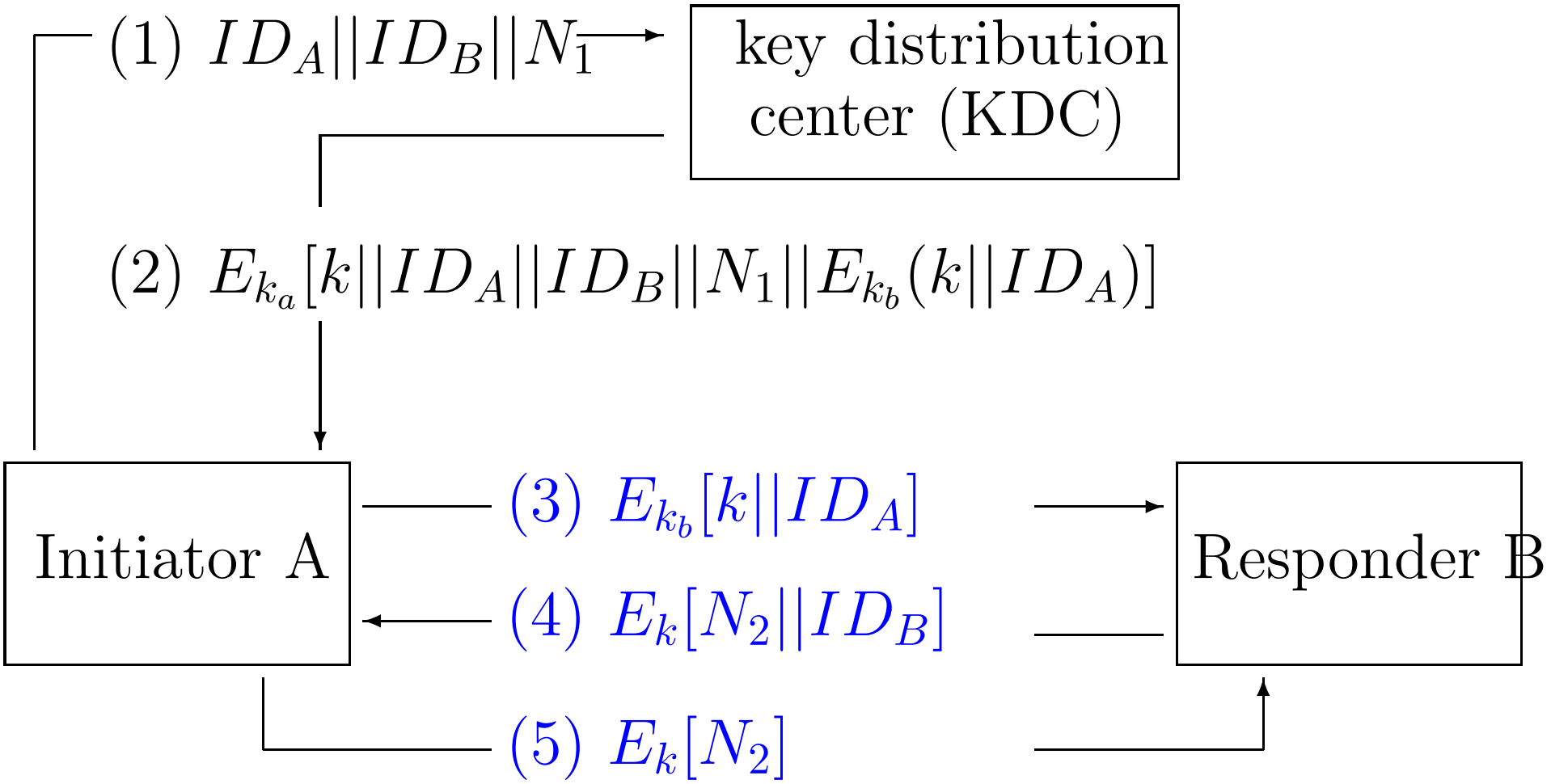
Remark: The master keys are used to protect the sessions keys when they are distributed.

A Key Distribution Protocol – Continued

Pictorial description of use of the key hierarchy:



A Key Distribution Protocol



Parameters in the Key Distribution Protocol

- N_i is a nonce (a string of characters), used as identifier for that transaction and is a challenge to the other party.
- k_a, k_b master keys, k secret key.
- ID_A, the identifier of A.
- ID_B, the identifier of B.

Explaining the Key Distribution Protocol (1)

- The first message may be a forged or modified one. But after receiving Message (2), A will detect such a forgery or modification.
- After receiving Message (2), A can authenticate the creator and sender of Message (2) (sender by $ID_A||ID_B$ and antireplay by N_1).
- Steps (3), (4) and (5) are for mutual authentication. Why?
  - The creator of Message (3) must be the KDC as people other than the KDC and B do not know k_b and cannot embed ID_A inside the ciphertext.
  - If Message (3) is a replayed message, A will discover this replay after processing Message (4) (see next page for justification).

  Hence, Message (3) is the KDC's current data, but its sender may be A or somebody else. Therefore, Message (5) is necessary.

Explaining the Key Distribution Protocol (2)

If Message (3) is a replayed message, A will discover this replay after processing Message (4).

- If Message (3) is a replayed message, then Messages in (3) and (4) should be $E_{k_b}(k'||ID_A)$ and $E_{k'}(N_2||ID_B)$, where $k' \neq k$ and k' is an earlier key. Then after computing $D_k(E_{k'}(N_2||ID_B))$, A cannot find ID_B in the decrypted text $D_k(E_{k'}(N_2||ID_B))$.

Steps (3), (4) and (5) are for mutual authentication. Why?

- The challenge to B is k, and the challenge to A is N_2.
- Mutual authentication is done via a challenge-response sub-protocol.