安全协议设计与分析

第三讲：安全协议的需求分析与目标描述

李晖 网络空间安全学院
lihuill@bupt.edu.cn

What we learned

- Lecture 1:
  - Simple notion for protocols
  - Needham-Schroeder
- Lecture 2:
  - A high level overview of cryptography
    - Symmetric key Encryption
    - Public key encryptions and signing
    - Hash
  - Abstract equation for modelling encryption

The Needham-Schroeder Protocol

An (in) famous authentication protocol

1. $A \rightarrow B : E_{KUb}(N_a, A)$
2. $B \rightarrow A : E_{KUa}(N_a, N_b)$
3. $A \rightarrow B : E_{KUb}(N_b)$

N_a and N_b can then be used to generate a symmetric key

An Attack Against the Needham-Schroeder Protocol

The attack acts as a man in the middle:

1. $A \rightarrow C : E_{KUc}(N_a, A)$

   1`. $C(A) \rightarrow B : E_{KUb}(N_a, A)$

   2`. $B \rightarrow C(A) : E_{KUa}(N_a, N_b)$

2. $C \rightarrow A : E_{KUa}(N_a, N_b)$

3. $A \rightarrow C : E_{KUc}(N_b)$

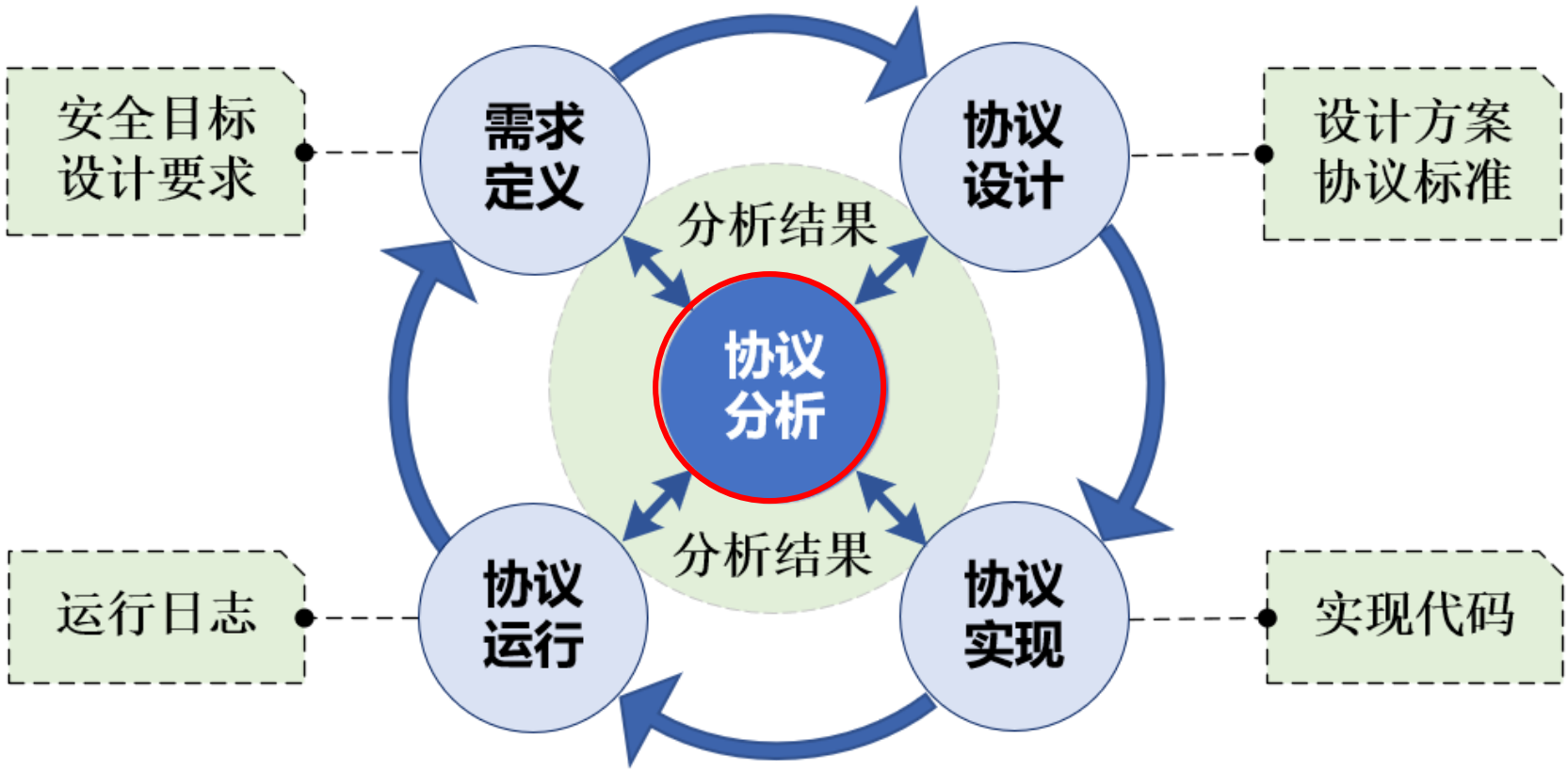
   3`. $C(A) \rightarrow B : E_{KUb} (N_b)$

Today:

- First Part: Analyzing requirements of Security Protocol

- Second Part: Goals for security protocol
  - To know if a protocol is secure you must know what is aims to achieve.
  - Example: Diffie-Hellman & STS Protocol.

Security protocol lifecycle



Analyzing the requirements

- **Functional analysis**
  - Participating entities?
  - Capabilities of each entity?
  - Including Trusted Third-Party?
  - Authentication?
  - Key agreement?
  - … …
- **Security threats**
  - Possible attacks?
  - Who,When,Where,What,How?
- **Security requirements**
  - Security goals

Is This An Attack?

The Needham-Schroeder key establishment protocol is the bases for Kerberos

- A and B use the trusted 3rd party S to establish a key K_{ab}:

1. $A \rightarrow S : A, B, N_a$
2. $S \rightarrow A : E_{Kas} \{ N_a , B, K_{ab}, E_{Kbs} \{K_{ab} , A \} \}$
3. $A \rightarrow B : E_{kbs} \{ K_{ab} , A \}$

4. $B \rightarrow A : E_{kab} \{ N_b \}$
5. $A \rightarrow B : E_{kab} \{ N_b + 1 \}$

Is This An Attack?

If Eve knows $oldK_{ab}$, forces B to use an old key:

-999. $A \rightarrow S : A, B, N_a$

-998. $S \rightarrow A : E_{Kas}\{ N_a , B, oldK_{ab}, E_{Kbs}\{oldK_{ab} , A \}\}$

.....

.....

3. $E (A) \rightarrow B : E_{Kbs}\{ oldK_{ab} , A \}$

4. $B \rightarrow E (A) : E_{oldKab}\{ N_b \}$

5. $E (A) \rightarrow B : E_{oldKab}\{ N_b + 1 \}$

What is a Protocol Trying to Achieve?

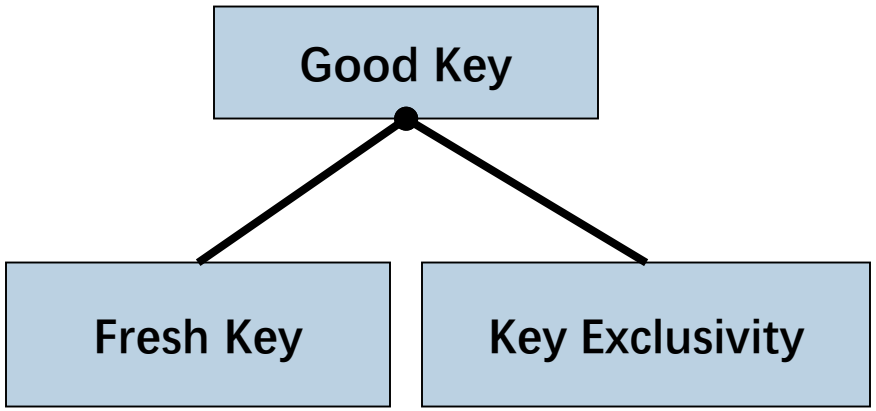
We MUST be clear about what we are trying to do with a protocol:

- Security
  - If we can securely establish a fresh key then secure transfer of data is easy.
- Key establishment
  - Yes, but we saw the problem in the last example
- Authentication
  - Again we have to be exact.

Some Key Establishment Goals

- **Key Freshness**: the key established is new
  - either from some trusted 3rd party
  - or because it uses a new nonce.
- **Key Exclusivity**: the key is only known to the principals in the protocol.
- **Good Key**: the key is **both** fresh and exclusive

A Hierarchy of Goals



A Good Key is Not Enough

- A "Good Key" provides a secure communication channel....

  ... but we have no idea who is on the other end of the channel.

- We need authentication as well.

Authentication Goals

The most simple:

- Far-end Operative: A knows that "B" is currently active:
  - For instance B might have signed a nonce generated by A e.g.

1. $A \rightarrow B : N_a$
2. $B \rightarrow A : Sign_B (N_a)$

Authentication Goals

and

- Once Authentication: A knows that B wishes to communicate with A
  - For instance, B might have the name A in a message e.g.

1. $B \rightarrow A : Sign_B (A)$

Entity Authentication

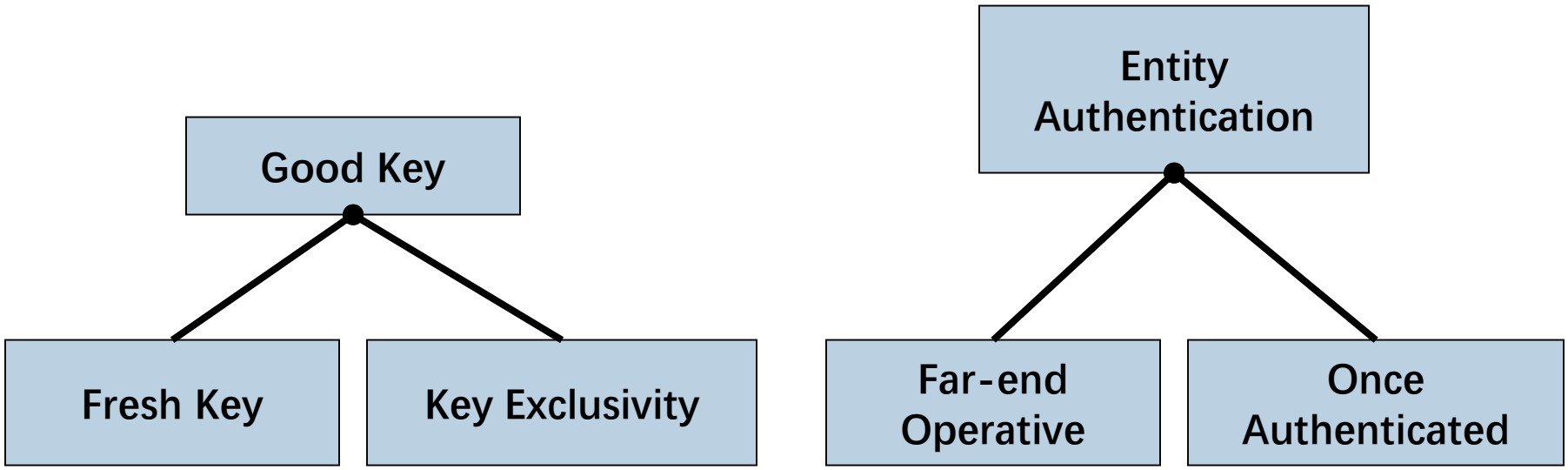
Both of these together give:

- Entity Authentication: A knows that B is currently active ***and*** wants to communicate with A.

e.g.

1. $A \rightarrow B : N_a$
2. $B \rightarrow A : Sign_B (A, N_a)$

A Hierarchy of Goals

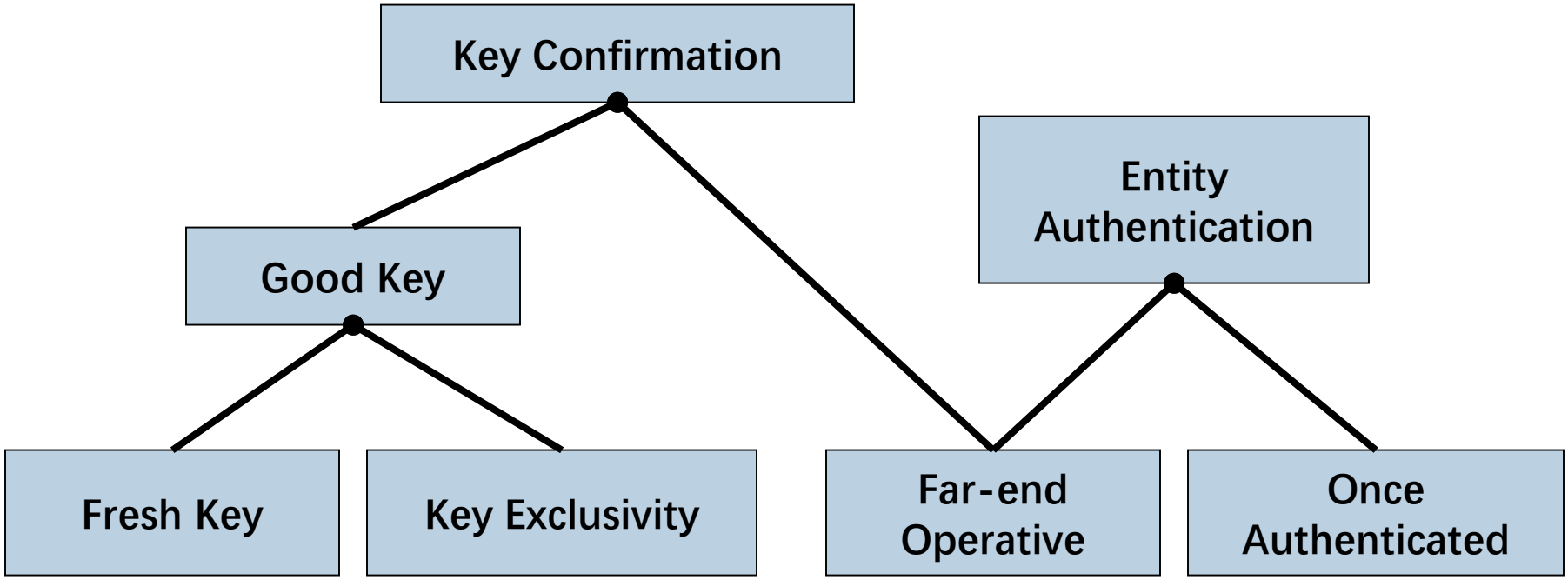


Key Confirmation

A common goal is:

- **Key Confirmation** of A to B is provided if
  - B can be sure that "K" is a good key to communicate with A
  - and that A knows "K".

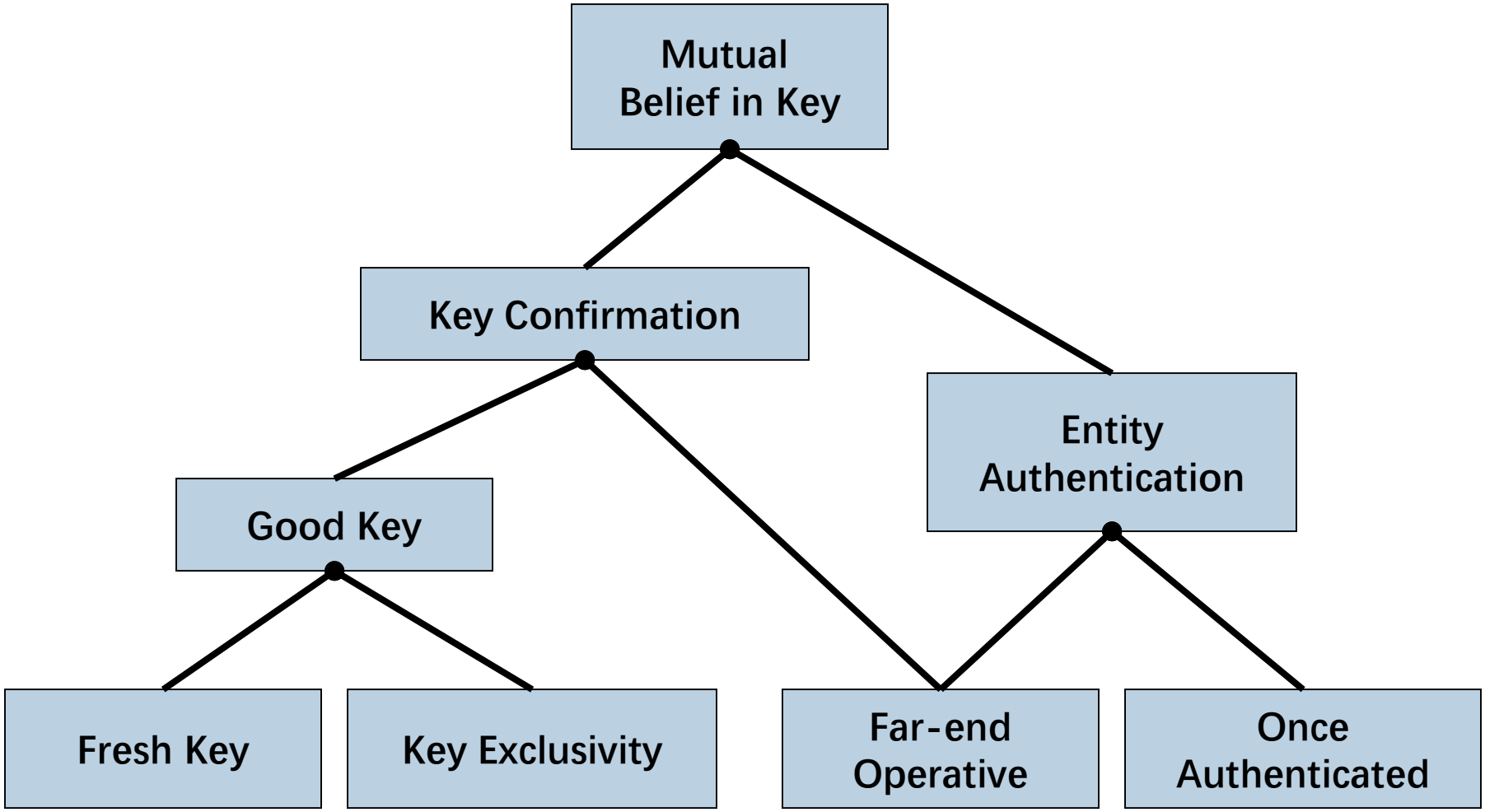
A Hierarchy of Goals



The Highest Goal

- **Mutual Belief in Key**: is provided for B only if
  - “K” is a good key with A
  - A believes that “K” is a good key for “B”
  - and A wishes to communicate with B using “K”

A Hierarchy of Goals



Example: Diffie-Hellman

- The Diffie-Hellman is a widely used key agreement protocol.

- It relies on some number theory:
  - a mod b = n where ∃m s.t. a = m*b + n

- The protocol uses two public parameters
  - generator "g" (often 160 bits long)
  - prime "p" (often 1024 bits long)

Diffie-Hellman

- A and B pick random numbers r_A and r_B and calculate

  "$t_A = g^{r_A} \bmod p$" and

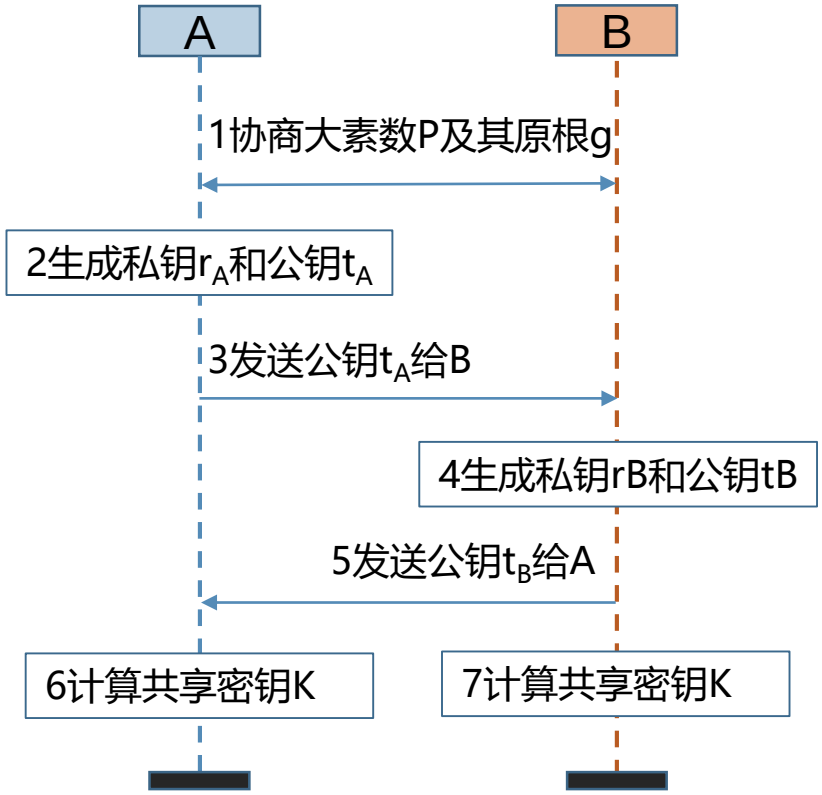
  "$t_B = g^{r_B} \bmod p$"

- The protocol just exchanges these numbers:
  1. $A \rightarrow B : t_A$
  2. $B \rightarrow A : t_B$

- A calculates "$t_B^{r_A} \bmod p$" and B calculates "$t_A^{r_B} \bmod p$"

this is the key:

$$K = g^{r_A r_B} \bmod p$$



Diffie-Hellman

- An observer cannot work out r_A and r_B from t_A and t_B therefore the attacker cannot calculate the key.
- The values of "g" and "p" must be big enough to make it intractable to try all possible combinations.
- So we have a "**Good Key**" but know nothing about the participants.
- **We did not need to share any keys at the start, therefore this is a very powerful protocol.**
- Vulnerable to a Man in the Middle (MitM) attack.

Basic Station-to-Station Protocol

- The Basic Station-to-Station (STS) protocol adds authentication:

1. $\mathbf{A \rightarrow B : t_A}$
2. $\mathbf{B \rightarrow A : t_B , \{ Sign_B(t_A, t_B) \}_{Kab}}$
3. $\mathbf{A \rightarrow B : \{ Sign_A(t_A, t_B) \}_{Kab}}$

($t_A = g^{rA} \bmod p$; $t_B = g^{rB} \bmod p$; $K_{ab} = g^{rArB} \bmod p$)

- The **good key** goal is achieved for A:

1. The signature in message 2 can only be formed by B.

2. It is not a replay from an old protocol run since A knows that t_A (g^{rA}) was fresh.

3. The signature alone does not imply that B knows K_{ab}. Therefore the encryption with K_{ab} is necessary to provide assurance that B really knows K_{ab} .

Full Station-to-Station Protocol

- Strong entity authentication is more problematic since there is no explicit inclusion of identifiers in the signed messages which could be used to deduce the identity of the desired communications partner.

Lowe gives the modified version:

1. $A \rightarrow B : A, B, t_A$
2. $B \rightarrow A : B, A, t_B , \{ Sign_B(t_A, t_B) \}_{Kab}$
3. $A \rightarrow B : A, B, \{ Sign_A(t_A, t_B) \}_{Kab}$

- Good Key: as before
- Key Confirmation: A knows that B knows the K_{ab}.
- Any attacks?

Attack on Mutual Belief

1. $A \rightarrow E(B) : A, B, t_A$

   $1'. E \rightarrow B : E, B, t_A$

   $2'. B \rightarrow E : B, E, t_B, \{Sign_B(t_A,t_B)\}_{Kab}$

2. $E(B) \rightarrow A : B, A, t_B, \{Sign_B(t_A,t_B)\}_{Kab}$

3. $A \rightarrow E(B) : A, B, \{Sign_A(t_A, t_B)\}_{Kab}$

What does STS provide?

- Attacker E does NOT learn the key.
- B does not accept the key.
- But A does accept the key.

This can be fixed by changing line 2 to:

$$\mathbf{2.\ B \rightarrow A : t_B\ ,\ \{\ A,\ Sign_B(t_A,\ t_B\ )\ \}_{Kab}}$$

This is not done because this attack does not pose a real risk.

In this case **Key Confirmation** is enough.

Very Important Homework:

- Whenever you do something secure over the Internet (i.e., logon remotely, check your bank balance, pay a bill), ***think about the protocol used.***

Is the communication secure?

- Have you established a key? Is it fresh?

Think about who has been authenticated:

- Have you been authenticated?
- Has the computer you're talking to?

Today: Goals for security protocol

- To know if a protocol is secure you must know what it aims to achieve.
- Example: Diffie-Hellman & STS Protocol.

Next lecture: Attacks and Principles

- Common types of attacks on protocols
- Good design principles for protocol.

References

- Tom Chothia, The Modelling and Analysis of Security Protocols, PPT
- David Basin, Protocols for Authentication and Key Establishment(2020), Book

谢谢大家！欢迎提问！