06 Authentication and Key Management

2020 Spring

Information Security

Teacher: Po-Wen Chi

neokent@gapps.ntnu.edu.tw

March 26, 2020

Department of Computer Science and Information Engineering,

National Taiwan Normal University

Before We Start

Review

What we have learned:

- Symmetric Key Encryption.
- Asymmetric Key Encryption.
- Hash/MAC.

Where is the key??

TTP: Trusted Third Party

In the Beginning

In the past, the sender and the receiver share the same codebook.

- Or use the newspaper.

Quiz

Given n people, if one member wants to talk with others securely, how many keys does the member need?

Quiz

Given n people, if one member wants to talk with others securely, how many keys does the member need?

Answer: $n - 1$.

And there will be total C_2^n keys in this system.

Quiz

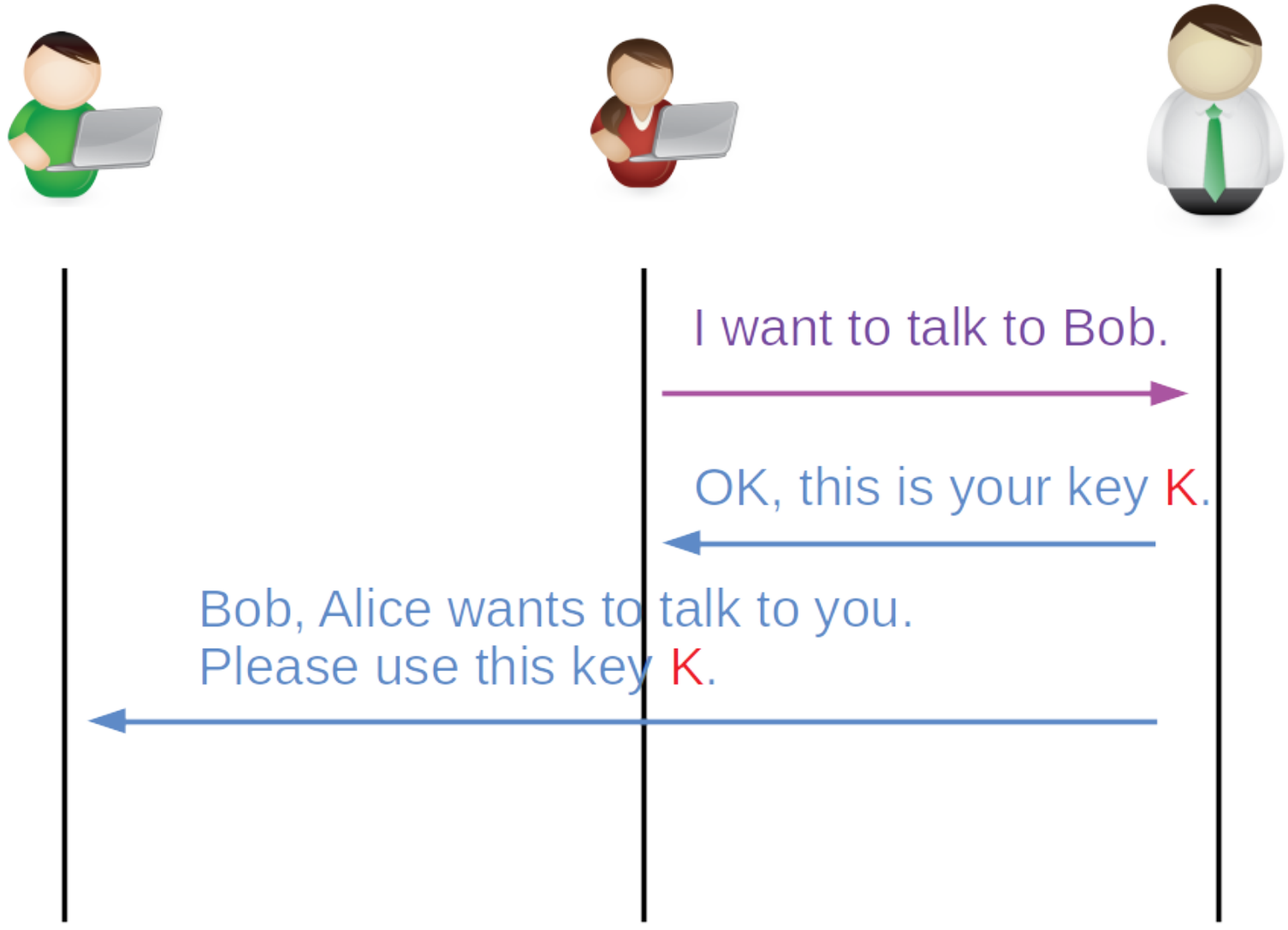
Given n people, if one member wants to talk with others securely, how many keys does the member need?

Answer: $n - 1$.

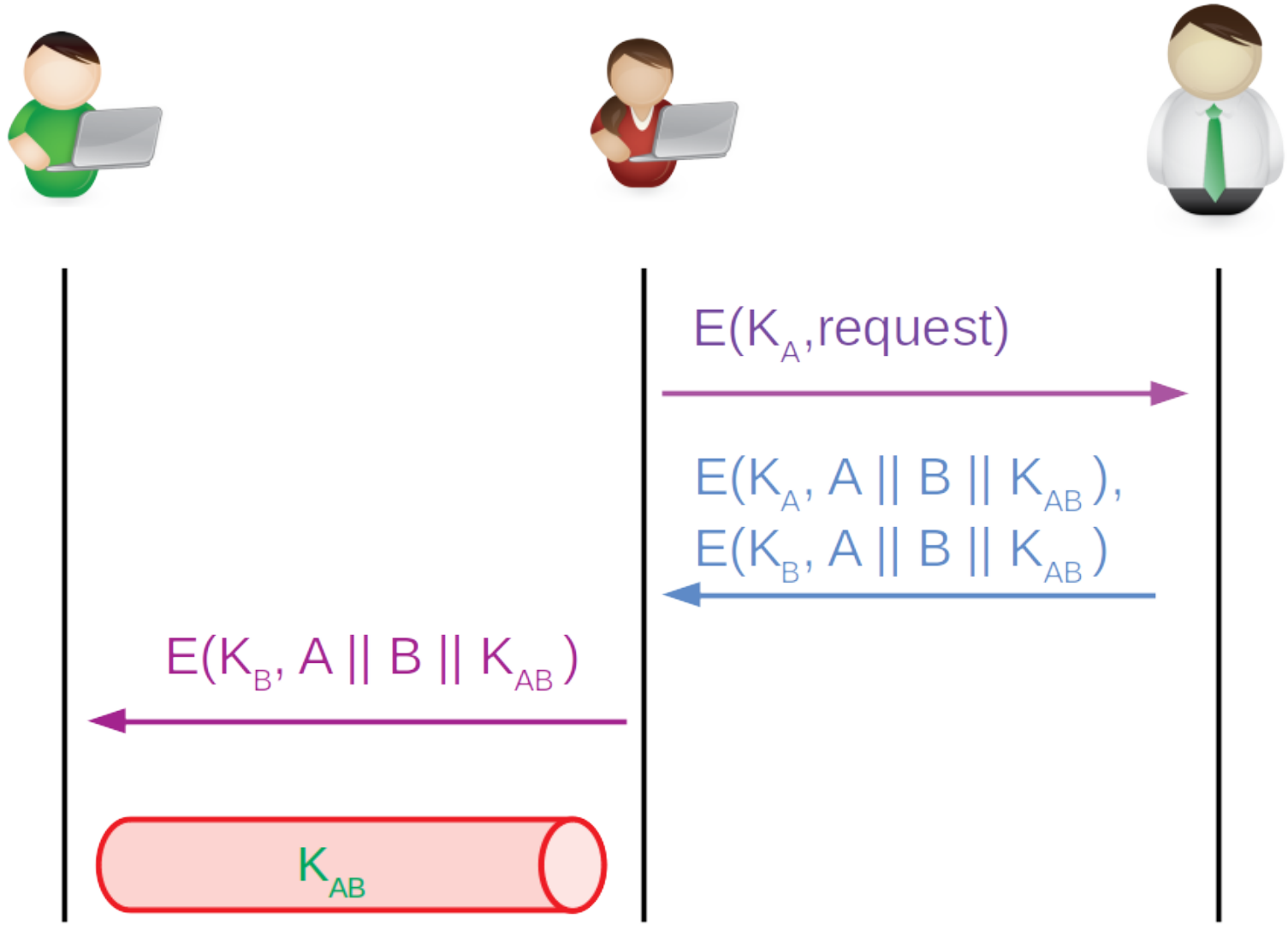
And there will be total C_2^n keys in this system.

Key management is really a big issue.

A Simple Solution: TTP



A Simple Solution: TTP



Quiz

How to attack??

Quiz

How to attack??

Replay Attack.

Quiz

How to attack??

Replay Attack.

How to defense?

Quiz

How to attack??

Replay Attack.

How to defense?

Add timestamp.

Is it Possible to Exchange Key without TTP

- TTP is a very simple solution.
- However, in some cases, we do not like TTPs.
  - TTP may not be always there.
  - Because there is no TTP in the real world.
    - Example: **Prism Project, Huawei**.
- How can we do this?
  - Remember, we always assume that the attacker can see your communication.

Some Solutions

- **Merkle Puzzles**.
- **Diffie Hellman**.
- **Public Key Solution**.
  - Since the key is public, of course we can release to the attacker.
  - Yes, but how to release your public key?

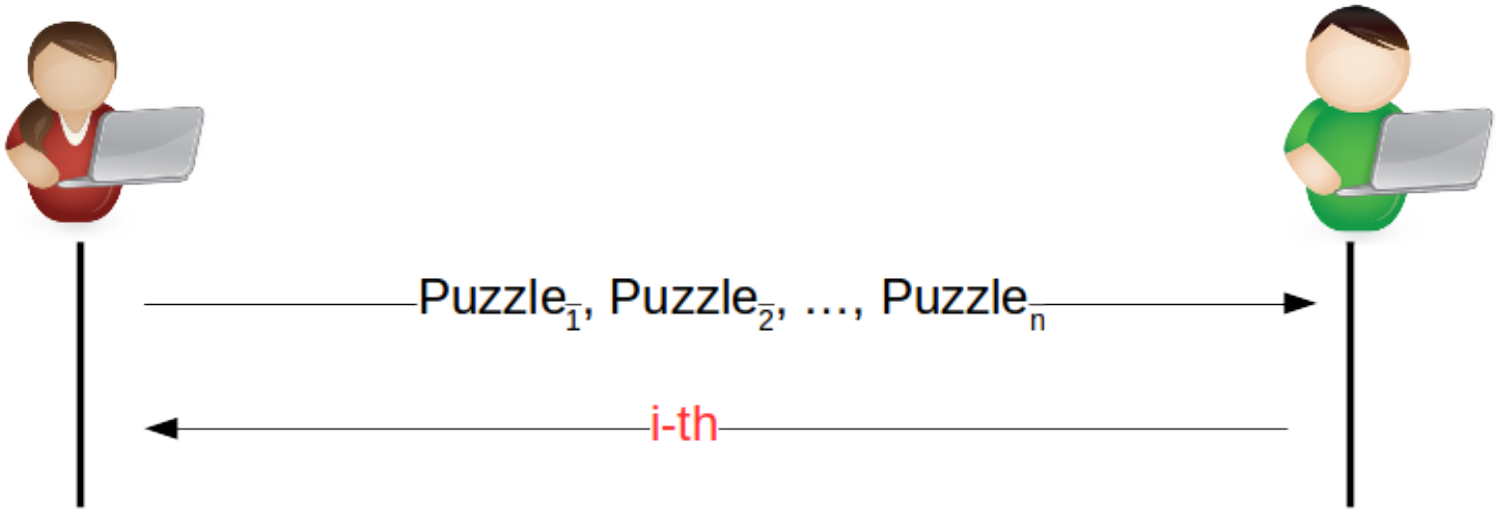
Merkle Puzzle

Puzzle-based Solution

1. Alice prepares n puzzles and sends all of them to Bob.
2. Bob randomly selects one puzzle and solves it.
3. After Bob solves it, Bob told Alice that I have solved i-th puzzle, we can communicate with the puzzle answer as a key.

How about the attacker??

Puzzle-based Solution



- What is a puzzle?
  - A symmetric key encryption where key is n-bits.
  - Message: "This is message X. This is the symmetrical key Y".
  - Example: $n = 32$
- For Alice: $O(n)$.
- For Bob: $O(n)$.
- For Attacker: $O(n^2)$.

Diffie Hellman Key Exchange (Important)

Before We Start

We need to start with some assumptions.

What does Assumption Mean?

- In the crypto field, lots of mechanism are based on assumptions.
- If someone can crack our crypto system, it means (s)he can solve a hard problem.
  - The problem is assumed to be hard since we have no idea how to solve it.

What does Assumption Mean?

- In the crypto field, lots of mechanism are based on assumptions.
- If someone can crack our crypto system, it means (s)he can solve a hard problem.
  - The problem is assumed to be hard since we have no idea how to solve it.
  - But the reason may be that we are not smart enough.

Some Assumptions

Discrete Logarithm Assumption

Given $g, a \in \mathbb{G}$, it is hard to find an integer x that $g^x = a$.

Computational Diffie-Hellman Assumption

Given $g^x, g^y \in \mathbb{G}$, it is hard to compute g^{xy}.

Decisional Diffie-Hellman Assumption

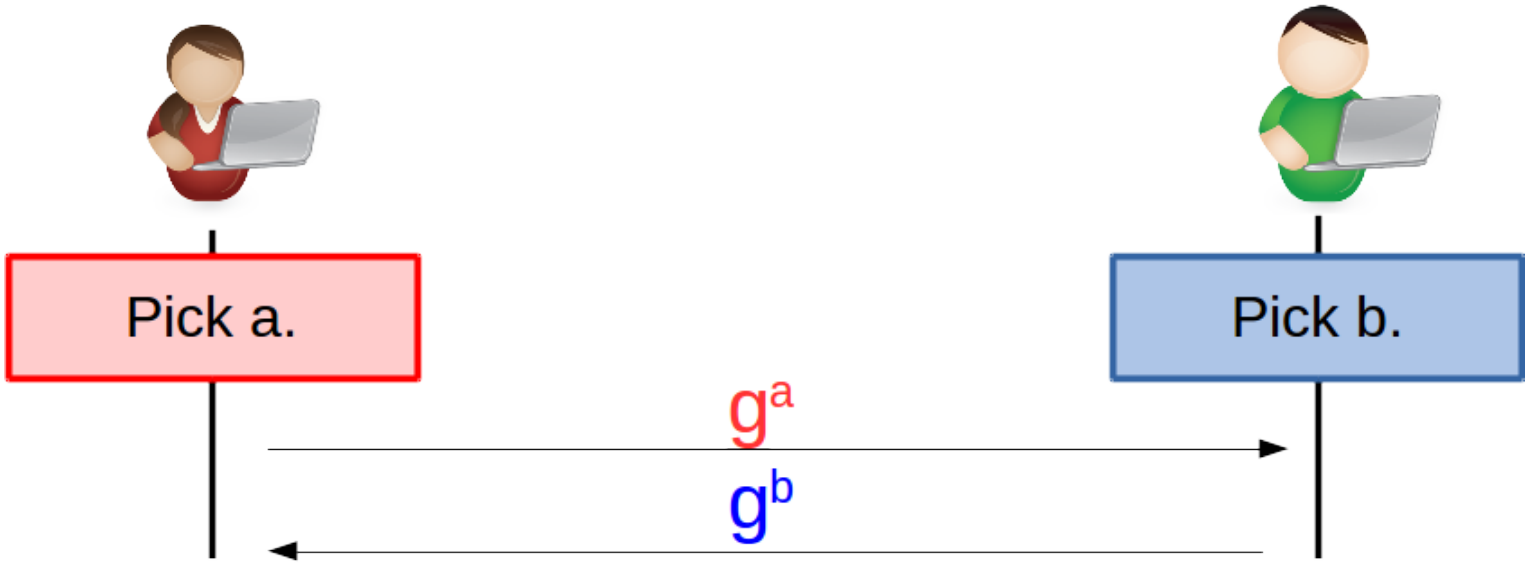
Given $g^x, g^y, t \in \mathbb{G}$, it is hard to verify if $t = g^{xy}$.

Please do not tell me that given 2 and 8, the answer is 3.

Quiz

Which one is the hardest among three?

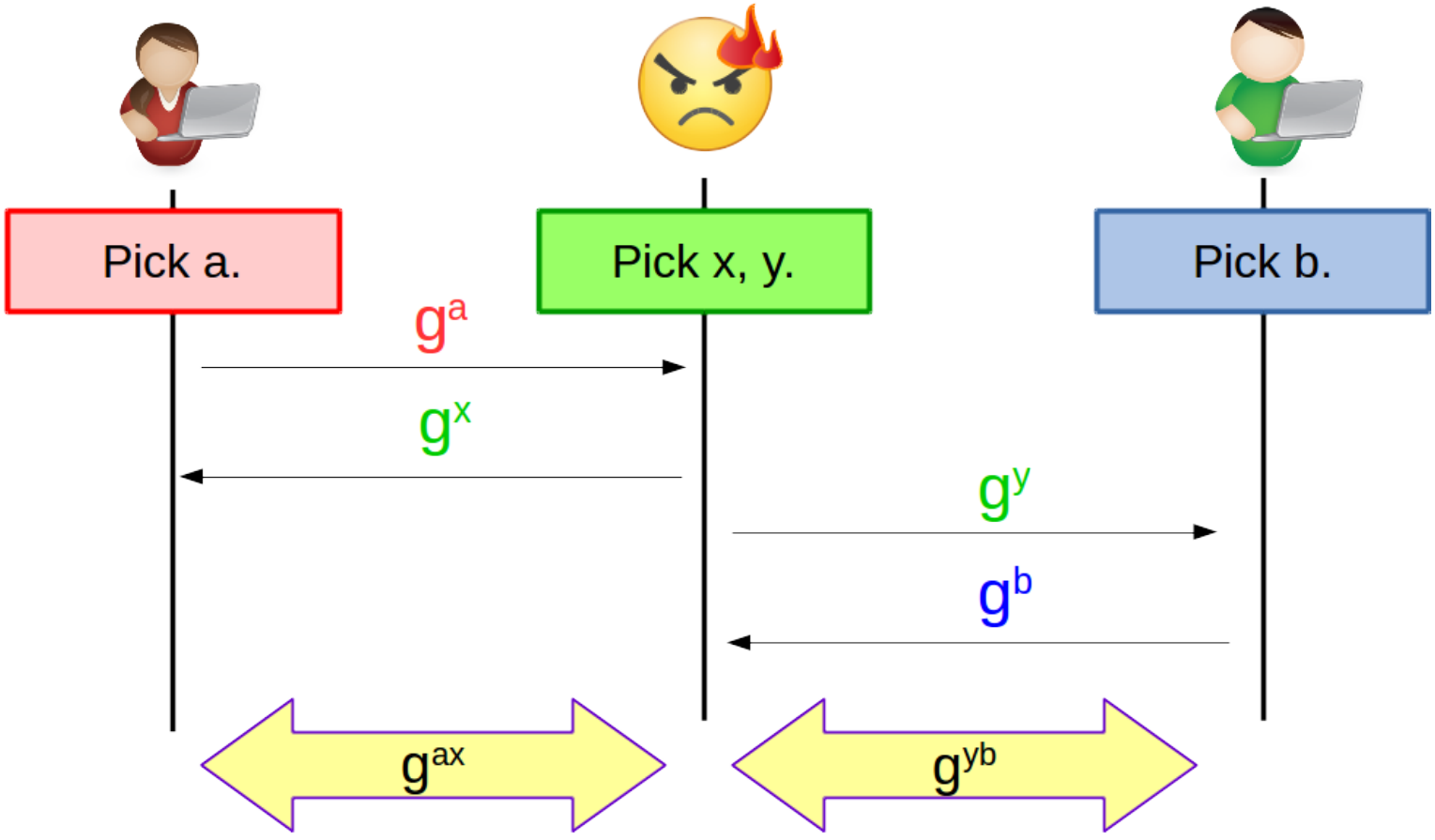
Diffie-Hellman Protocol



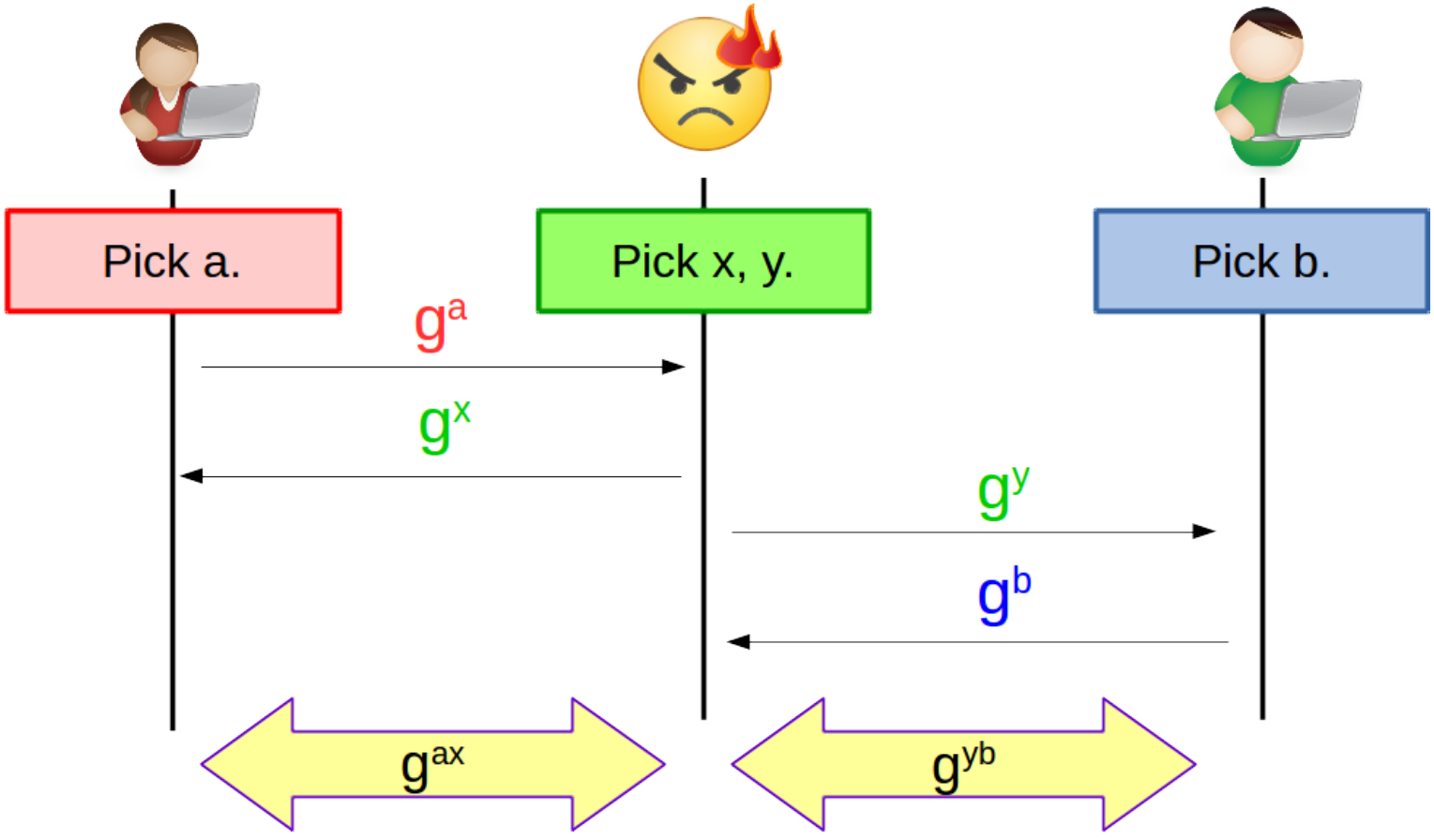
The key will be g^{ab}.

How to Attack??

Man in the Middle Attack

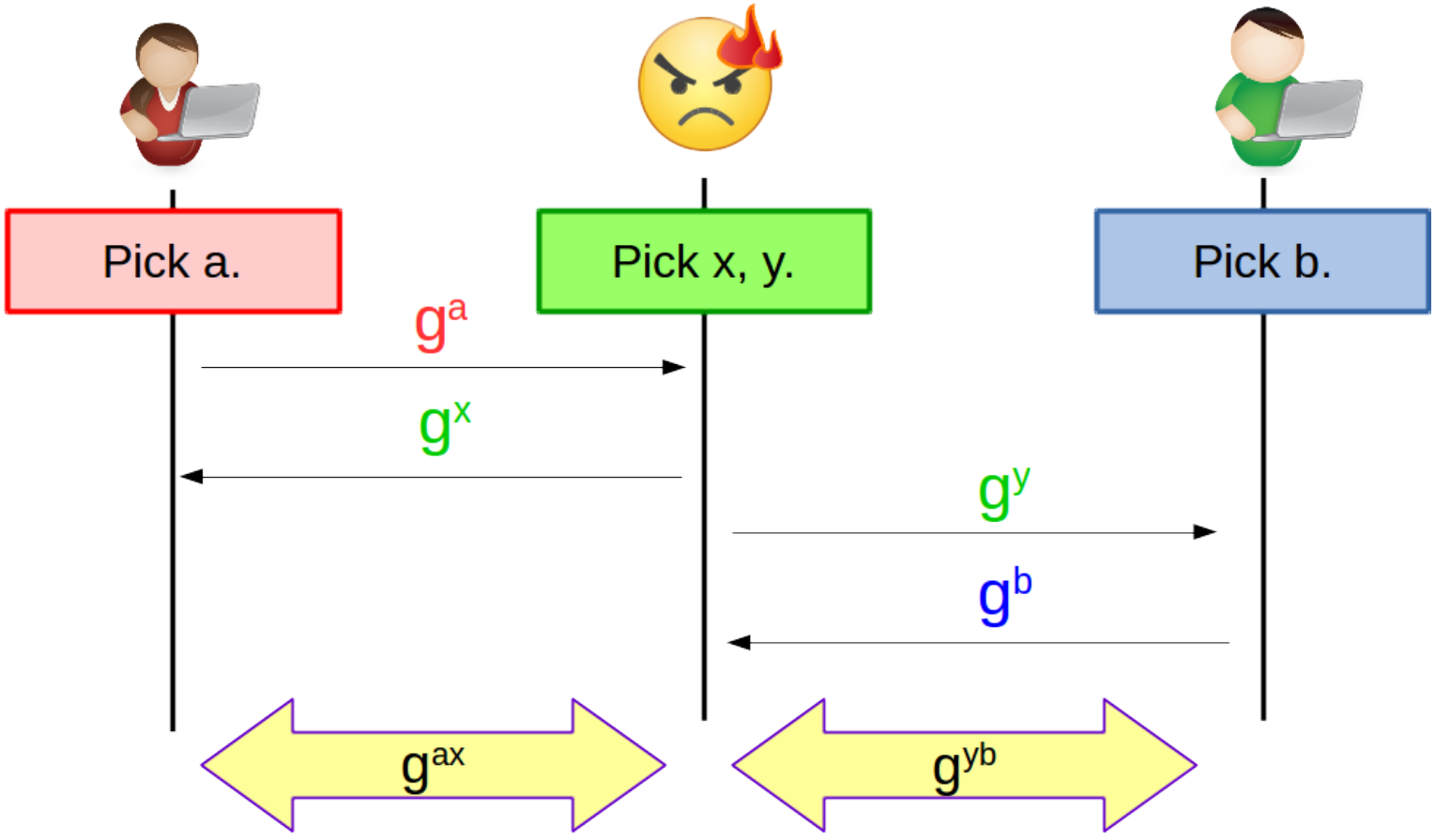


Man in the Middle Attack



Note that this attack does not solve the hard problem.

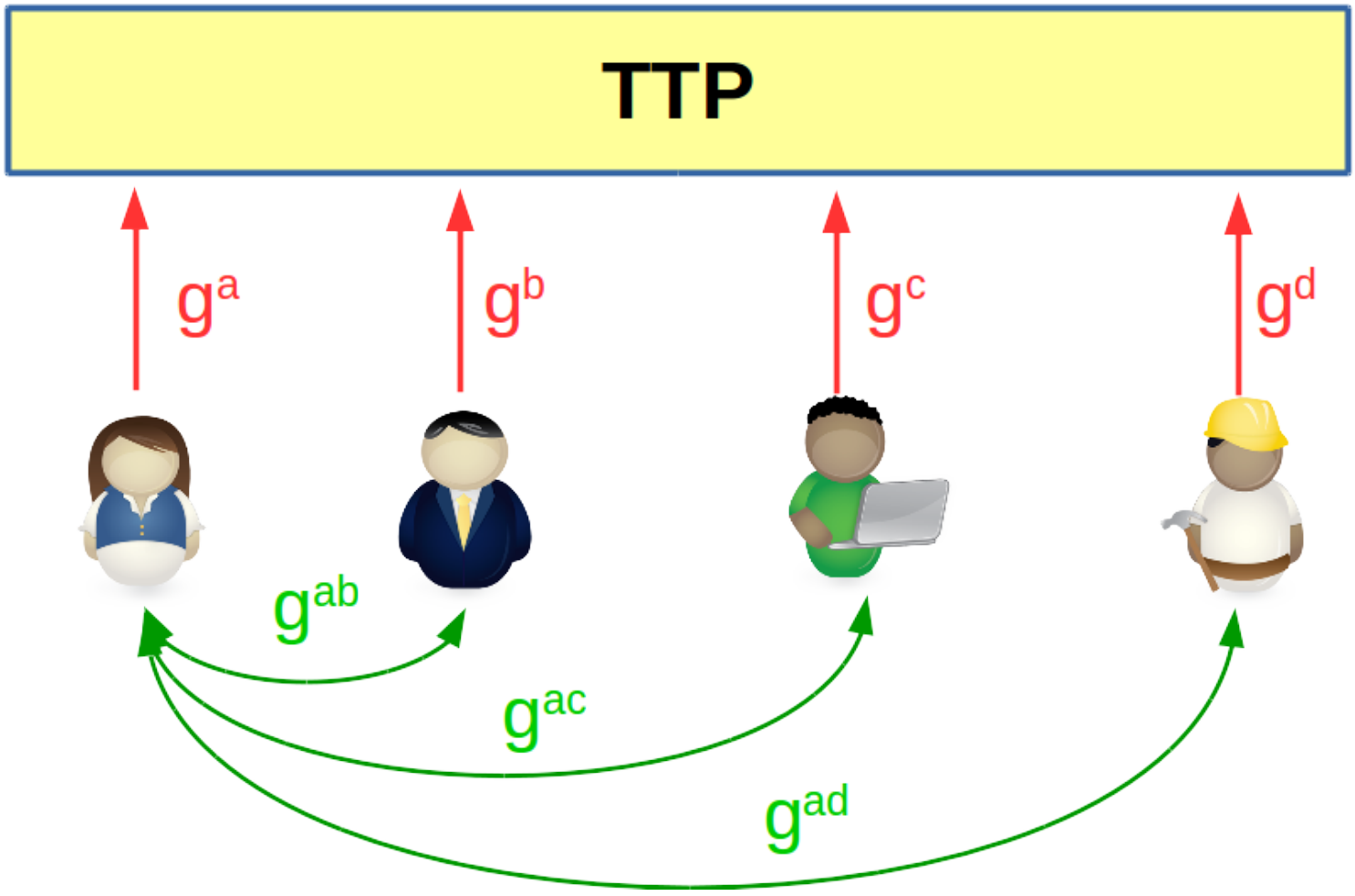
Man in the Middle Attack



Note that this attack does not solve the hard problem.

How to defense?

Another Usage



Questions

- How to establish a key between Alice, Bob and Charlie?

Questions

- How to establish a key between Alice, Bob and Charlie?
  - **Bilinear Map Group**.

Questions

- How to establish a key between Alice, Bob and Charlie?
  - **Bilinear Map Group**.
- How to establish a key between Alice, Bob, Charlie, Dennis?

Questions

- How to establish a key between Alice, Bob and Charlie?
  - **Bilinear Map Group**.
- How to establish a key between Alice, Bob, Charlie, Dennis?
  - **Multi-linear Map Group**.
  - **Open Question**.

ElGamal Encryption

ElGamal Encryption

- ElGamal is a famous public key encryption system and ...

ElGamal Encryption

- ElGamal is a famous public key encryption system and ...
- Wait a moment! I think we have already completed the public key encryption topic.

ElGamal Encryption

- ElGamal is a famous public key encryption system and ...
- Wait a moment! I think we have already completed the public key encryption topic.
- Yes, but there is something we skip ... **the security proof**.

ElGamal Encryption

- ElGamal is a famous public key encryption system and ...
- Wait a moment! I think we have already completed the public key encryption topic.
- Yes, but there is something we skip ... **the security proof**.
- Now we have some assumptions, we can prove the encryption system.

ElGamal Encryption Scheme

- **KeyGen**$(1^\lambda) \rightarrow \{PK, SK\}$:
  - A cyclic group $\mathbb{G}$ of order p with generator g.
  - $x \xleftarrow{R} \mathbb{Z}_p^*$.
  - $h \leftarrow g^x$.
  - $PK : \{\mathbb{G}, p, g, h\}$.
  - $SK : \{x\}$.
- **Enc**$(PK, m) \rightarrow C$:
  - $y \xleftarrow{R} \mathbb{Z}_p^*$.
  - $C = (c_1, c_2) = (g^y, m \cdot h^y)$.
- **Dec**$(SK, C) \rightarrow m$:
  - $s = c_1^x = g^{xy}$.
  - $c_2 \cdot s^{-1} = m \cdot h^y \cdot s^{-1} = m \cdot g^{xy} \cdot g^{-xy} = m$.

How to Prove its Security?

How to Prove its Security?

Which assumption should you use?

Security Proof

Theorem

If the decisional Diffie-Hellman assumption (DDH) holds in $\mathbb{G}$, then ElGamal achieves semantic security.

Proof:

If I can break ElGamal, given g^x, g^y, t, I know how to check if $t = g^{xy}$.

Security Proof

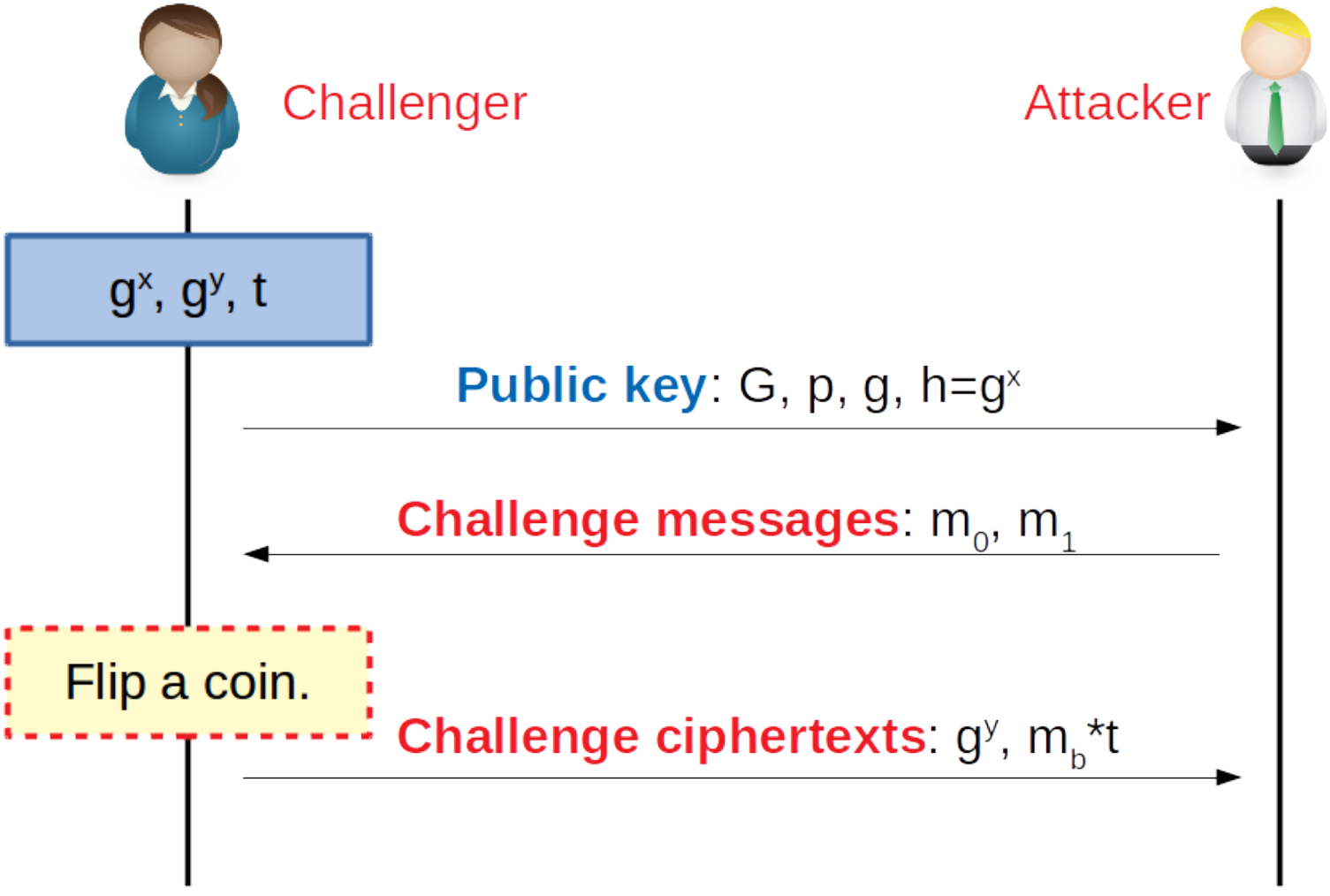
I give you g^x, g^y, t, would you please

answer me if $t=g^{xy}$?

I do not know.

(Wait … I remember that Bob knows how to break ElGamal. Maybe I can ask him for help.)

Security Proof



Security Proof

- If Bob cannot break the game:
  - The probability that Bob wins the game is $\frac{1}{2}$.
- If Bob can break the game:
  - The probability that Bob wins the game is $\frac{1}{2} + \delta$.

So if δ is not negligible, that is, the challenge ciphertext is a **valid** Elgamal ciphertext, which implies $t = g^{xy}$.

DSA: Digital Signature Algorithm

DSA

- **Setup**:
  - A large prime p.
  - A prime q that divides $p-1$.
  - A hash function H, like SHA-256.
  - $g = h^{\frac{p-1}{q}}$.
    - h is a random number $\in \{2, \ldots, p-2\}$. Commonly h $= 2$ is used.
    - $g^q \equiv ??$.

DSA

- **Setup**:
  - A large prime p.
  - A prime q that divides $p - 1$.
  - A hash function H, like SHA-256.
  - $g = h^{\frac{p-1}{q}}$.
    - h is a random number $\in \{2, \ldots, p-2\}$. Commonly h $= 2$ is used.
    - $g^q \equiv 1 \bmod p$.

DSA

- **Setup**:
  - A large prime p.
  - A prime q that divides $p-1$.
  - A hash function H, like SHA-256.
  - $g = h^{\frac{p-1}{q}}$.
    - h is a random number $\in \{2, \ldots, p-2\}$. Commonly h $= 2$ is used.
    - $g^q \equiv$.
- **KeyGen**:
  - Private key: $x, x < q$.
  - Public key: g^x.

DSA

- **Sign**: given a message m and a private key x.
  - $k \xleftarrow{R} \mathbb{Z}_q$.
  - $r = (g^k(\bmod p)) \bmod q$.
  - $s = (H(m) + xr) \cdot k^{-1} \bmod q$.
  - The signature is (r, s).
- **Verify**: ??

DSA

- **Sign**: given a message m and a private key x.
  - $k \xleftarrow{R} \mathbb{Z}_q$.
  - $r = (g^k(\bmod p)) \bmod q$.
  - $s = (H(m) + xr) \cdot k^{-1} \bmod q$.
  - The signature is (r, s).
- **Verify**: message m, signature (r, s), public key g^x:
  - $a = H(m) \cdot s^{-1} \bmod q$.
  - $b = r \cdot s^{-1} \bmod q$.
  - $v = (g^a (g^x)^b \bmod p) \bmod q$.
  - Check if $v = r$.

Correctness

$$
\begin{aligned}
v &= (g^a (g^x)^b \bmod p) \bmod q \\
&= (g^{H(m)\cdot s^{-1} \bmod q} (g^x)^{r\cdot s^{-1} \bmod q} \bmod p) \bmod q \\
&= (g^{(H(m)+xr)\cdot s^{-1} \bmod q} \bmod p) \bmod q \\
&= (g^k(\bmod p)) \bmod q
\end{aligned}
$$

Quiz

How to prove DSA is secure?

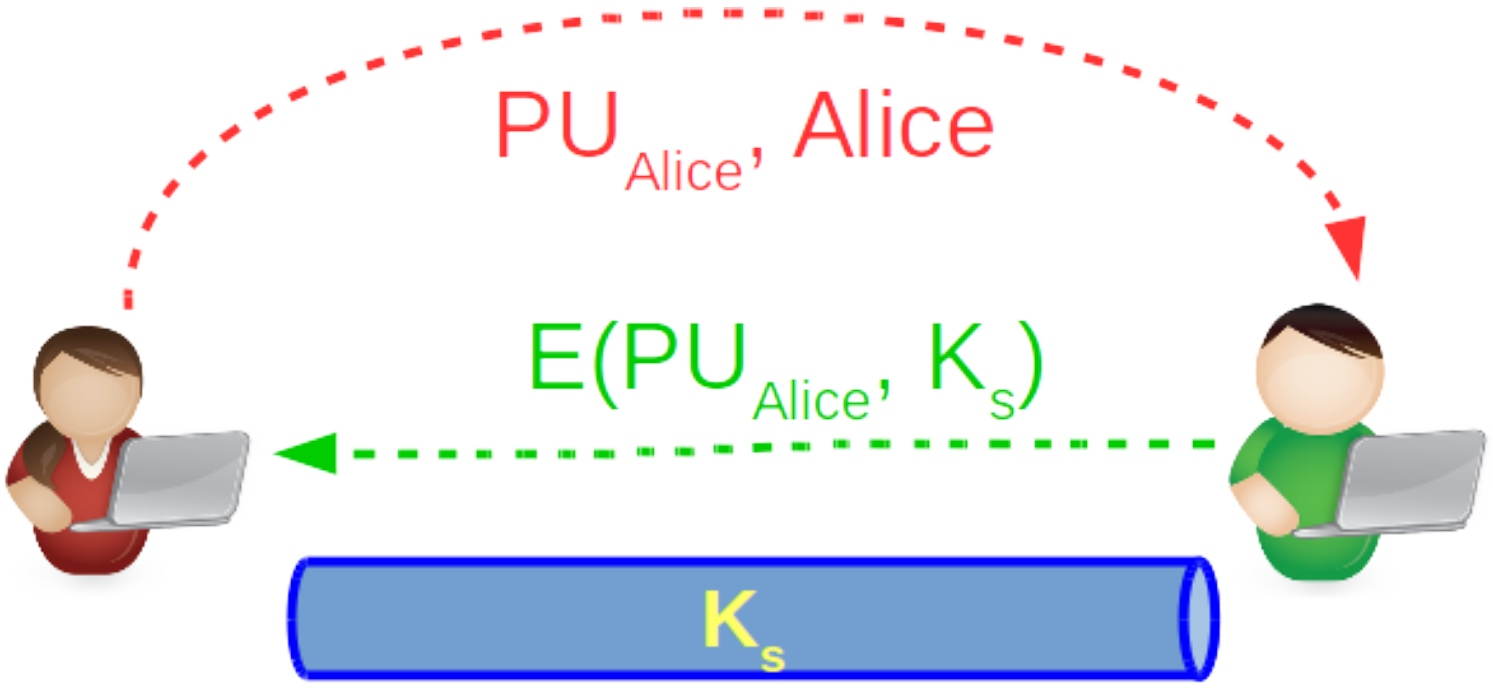
Quiz

How to prove DSA is secure?

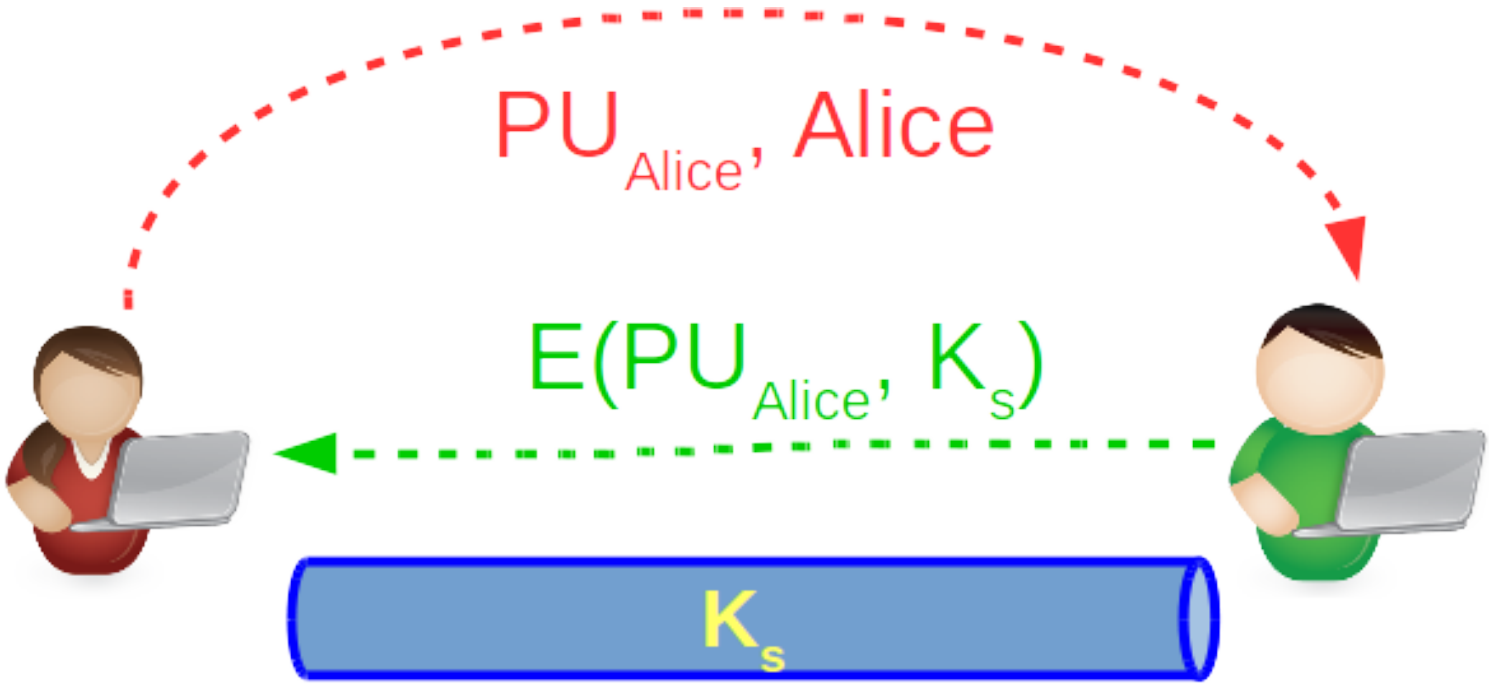
No one knows...

Key Distribution through Public Key Encryption

A Trivial Way

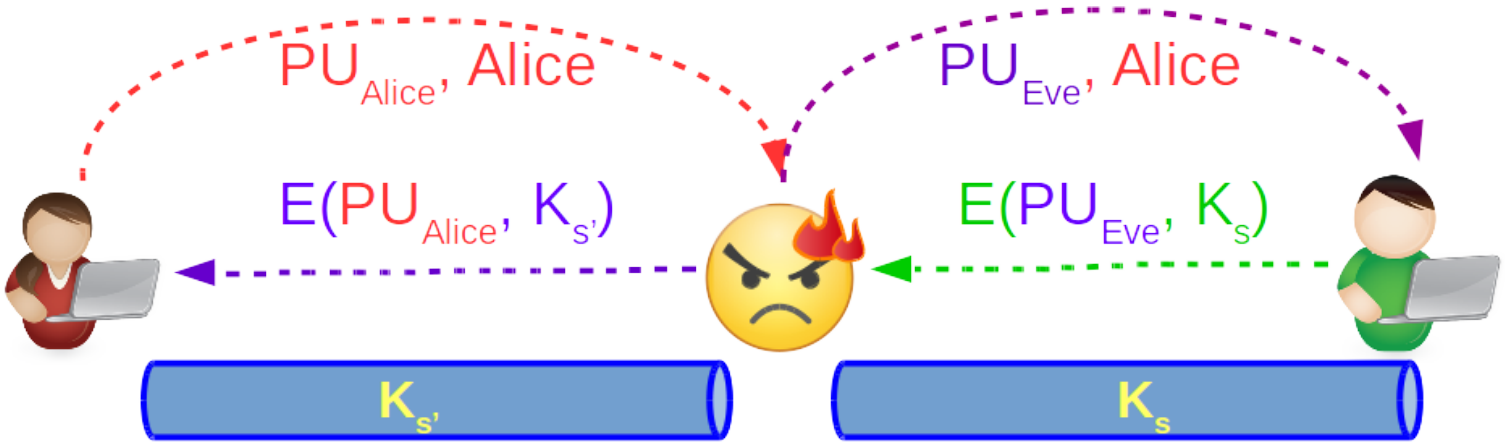


A Trivial Way



Quiz: Is there any problem in this scheme?

Man in the Middle

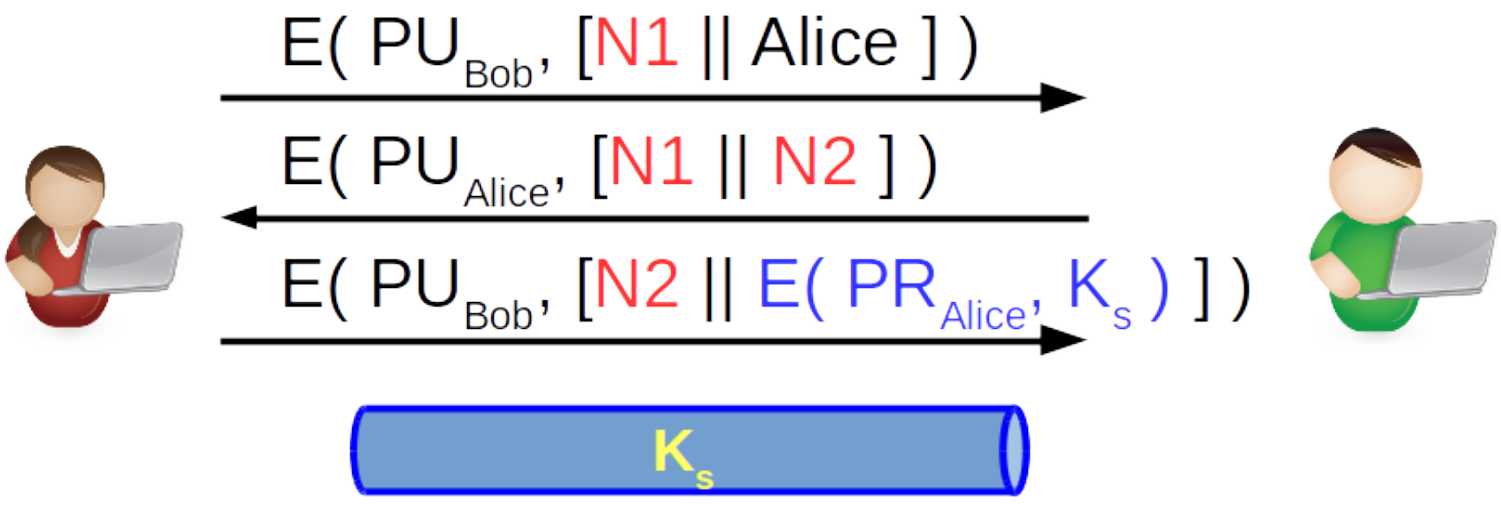


We do not know the sender really has the private key.

Quiz

Why do not we use public key encryption always?

Key Distribution with Authentication



One Important Problem

- Wait a moment! This scheme assumes that the public keys are well-known.
- How to distribute your public key?

One Important Problem

- Wait a moment! This scheme assumes that the public keys are well-known.
- How to distribute your public key?

So we will see how to distribute public keys.

One Important Problem

- Wait a moment! This scheme assumes that the public keys are well-known.
- How to distribute your public key?

So we will see how to distribute public keys.

BTW, is there any other problem about this scheme?

One Important Problem

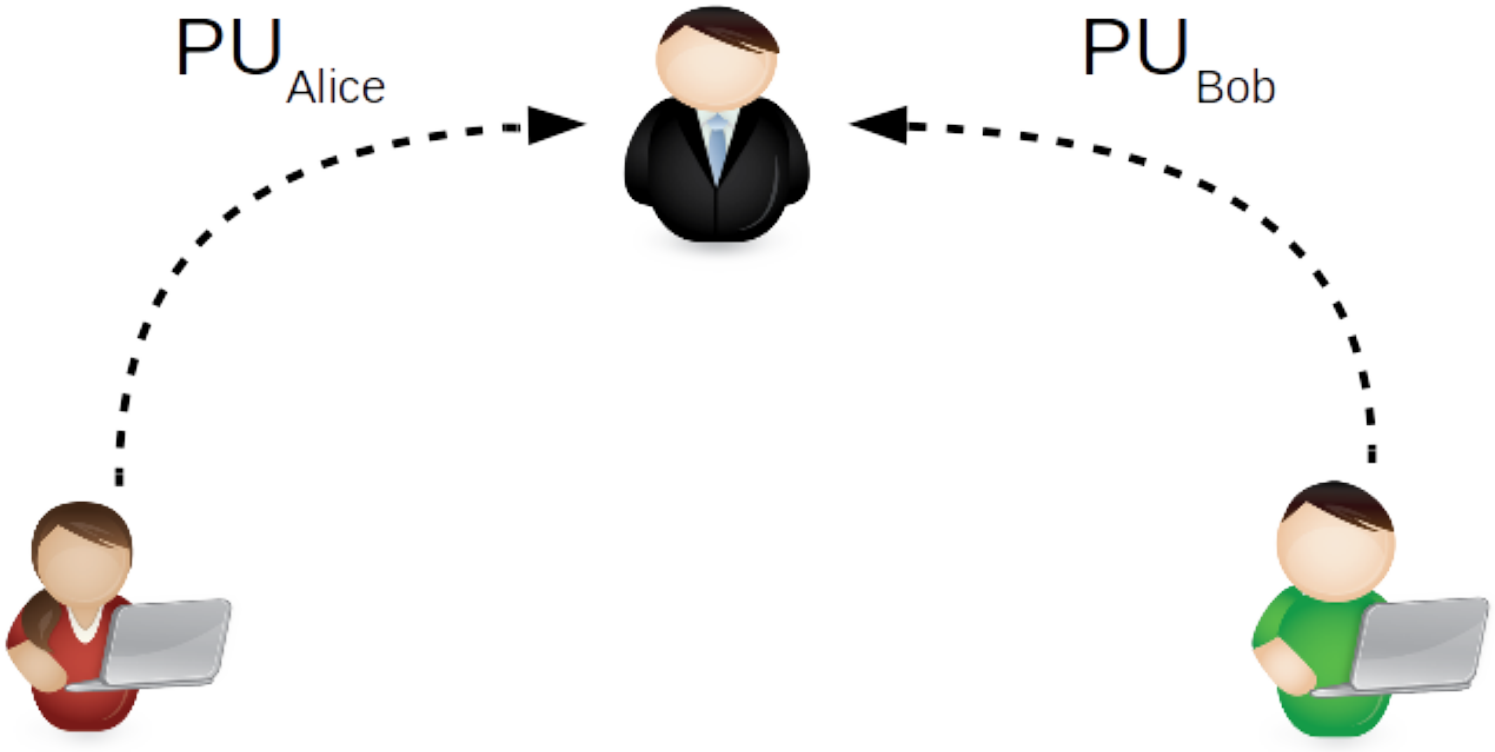
- Wait a moment! This scheme assumes that the public keys are well-known.
- How to distribute your public key?

So we will see how to distribute public keys.

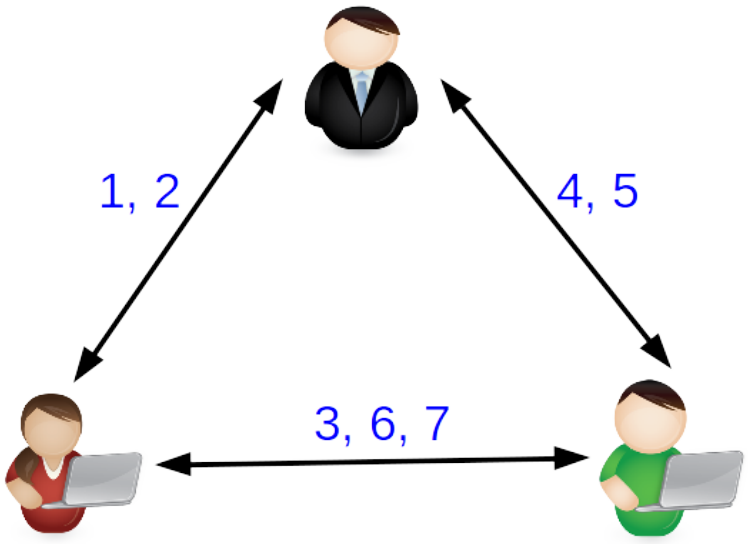
BTW, is there any other problem about this scheme? Replay attack.

X509 Certificate

Again, TTP is Always a Solution



TTP-based Solution

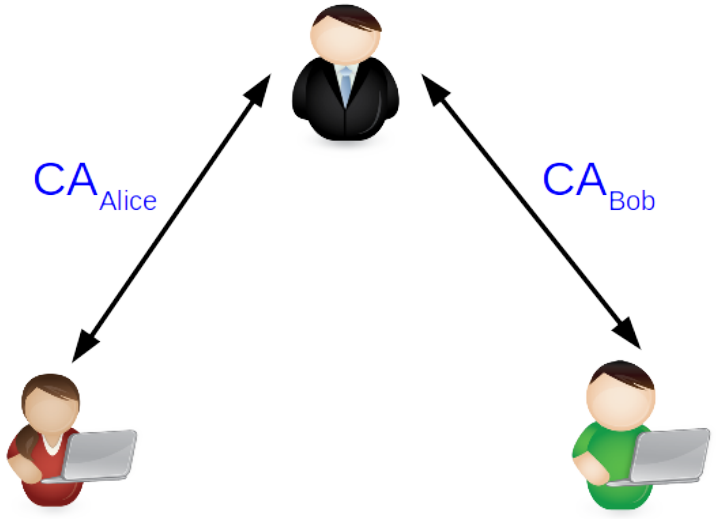


1. $\text{Req}||T_1$.
2. $E(PR_{\text{TTP}}, PU_{\text{Bob}}||\text{Req}||T_1)$.
3. $E(PU_{\text{Bob}}, ID_{\text{Bob}}||N_1)$.
4. $\text{Req}||T_2$.
5. $E(PR_{\text{TTP}}, PU_{\text{Alice}}||\text{Req}||T_2)$.
6. $E(PU_{\text{Alice}}, N_1||N_2)$.
7. $E(PU_{\text{Bob}}, N_2)$.

Some Drawbacks

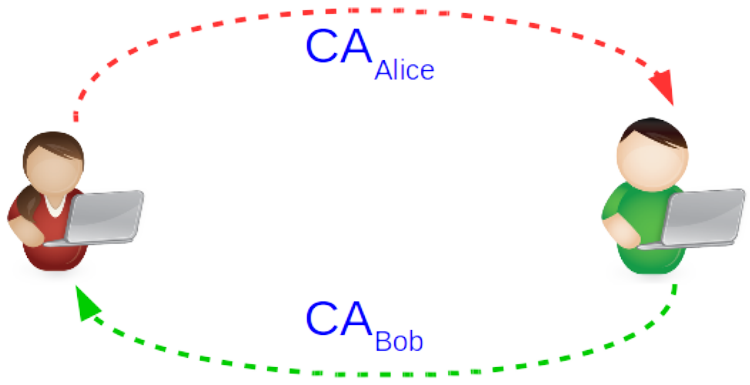
- Everytime when Alice and Bob need to talk, TTP should always be there.
- TTP may be the bottleneck.

Certificate-based Solution



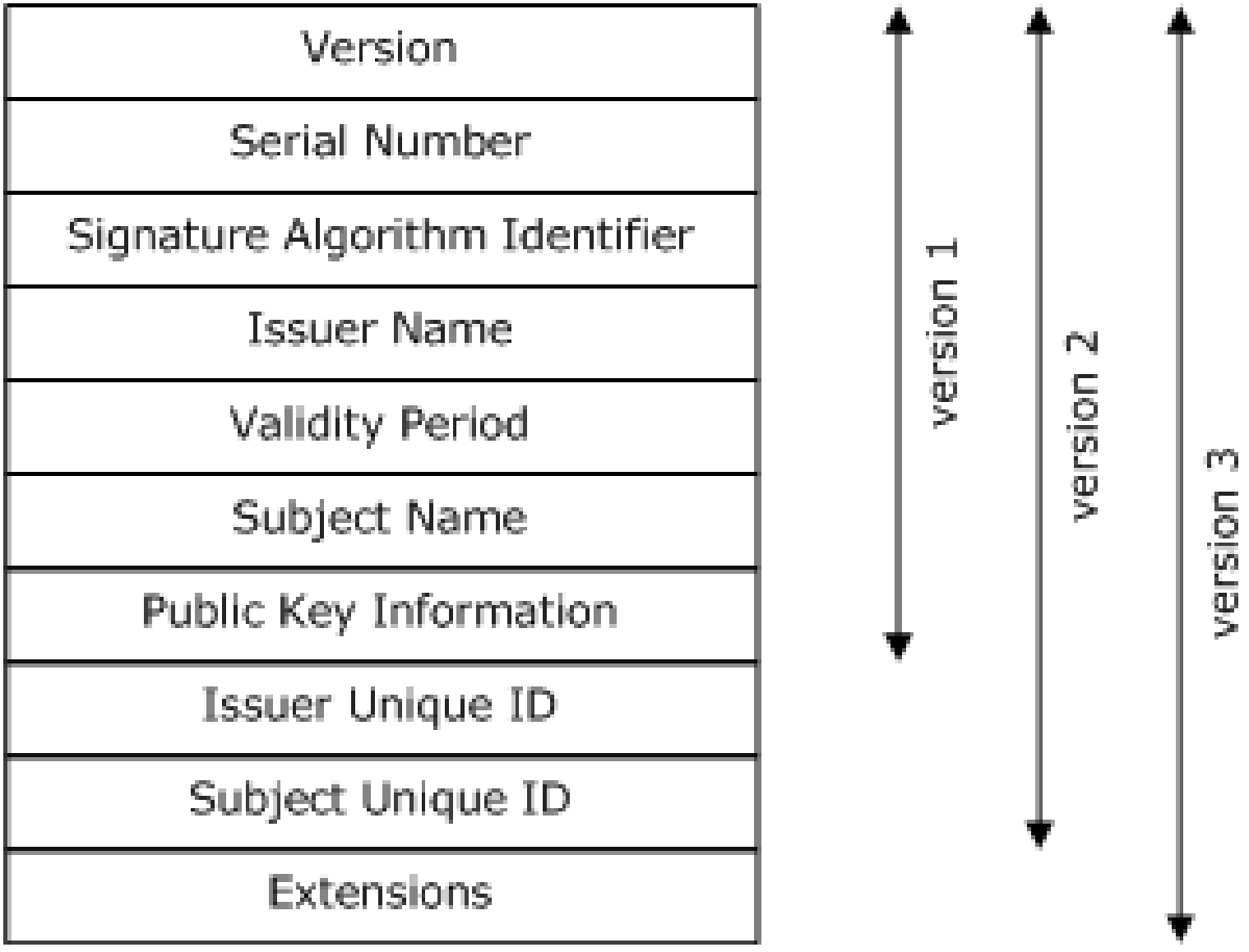
- $E(PR_{\text{TTP}}, T||ID||PU_{\text{ID}})$.

Certificate-based Solution



- $E(PR_{\text{TTP}}, T||ID||PU_{\text{ID}})$.

X509 Certificate



Certificate Example

Let's check some certificate.

- `http://w1.csie.ntnu.edu.tw/`
- `https://www.google.com/`
- `https://ftp.rsa.com/`

Who can Issue a Certificate?

- Actually, everyone can issue a certificate.
  - `http://www.study-area.org/tips/certs/certs.html`
- However, is there anyone who dares to accept certificates issued by you?

Note

If you want to have a CA, maybe you can try **Let's Encrypt**.

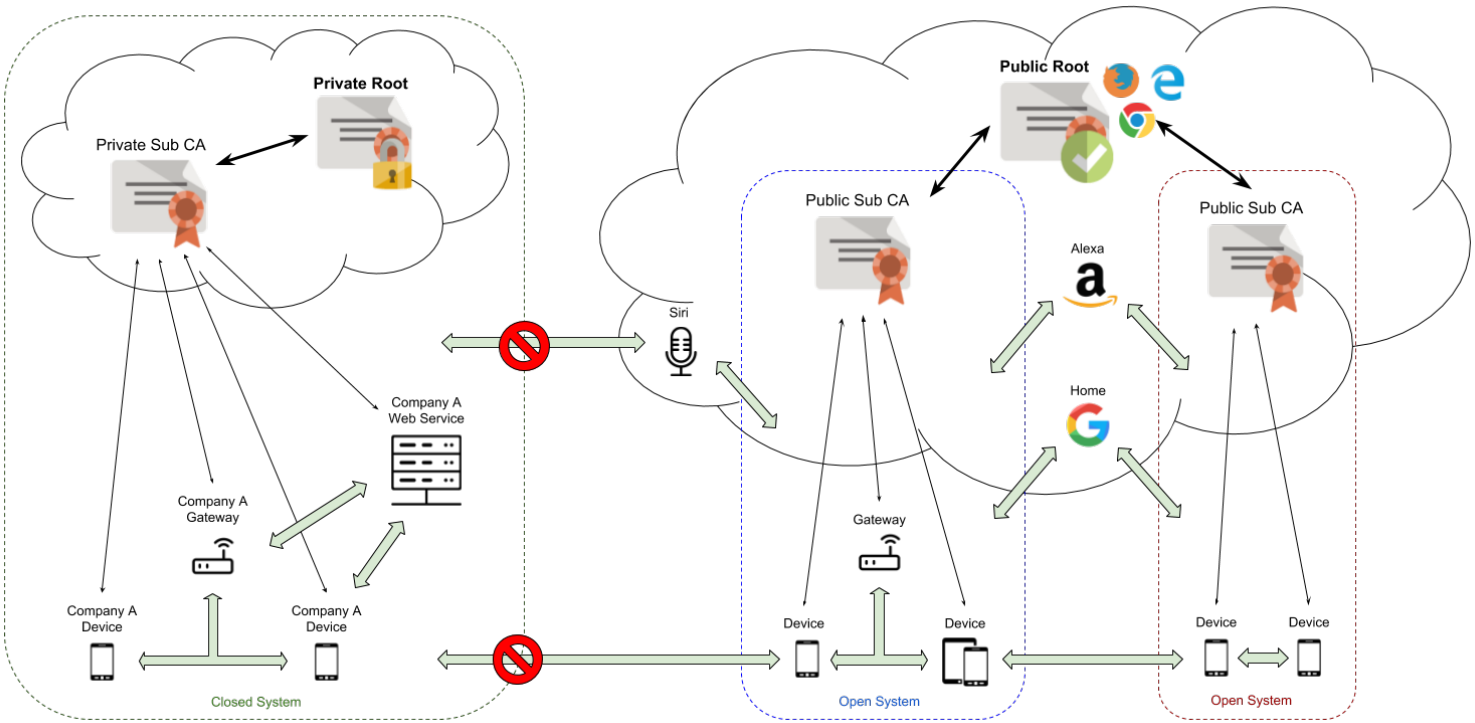
- **Free**.
- Since it is free, lots of people will use this CA, including bad guys.

PKI: Public Key Infrastructure

Public Key Infrastructure

A public key infrastructure (PKI) is a system for the creation, storage, and distribution of **digital certificates** which are used to verify that a particular public key belongs to a certain entity.

PKI


Private Root
Private Sub CA
Company A
Web Service
Company A
Gateway
Company A
Device
Company A
Device
Closed System
Siri
Public Root
Public Sub CA
Alexa
Home
Gateway
Device
Device
Open System
Public Sub CA
Device
Device
Open System

PKI

More details can be found at https://www.e-reading.club/bookreader.php/142115/Choudhury_-_Public_Key_Infrastructure_implementation_and_design.pdf.

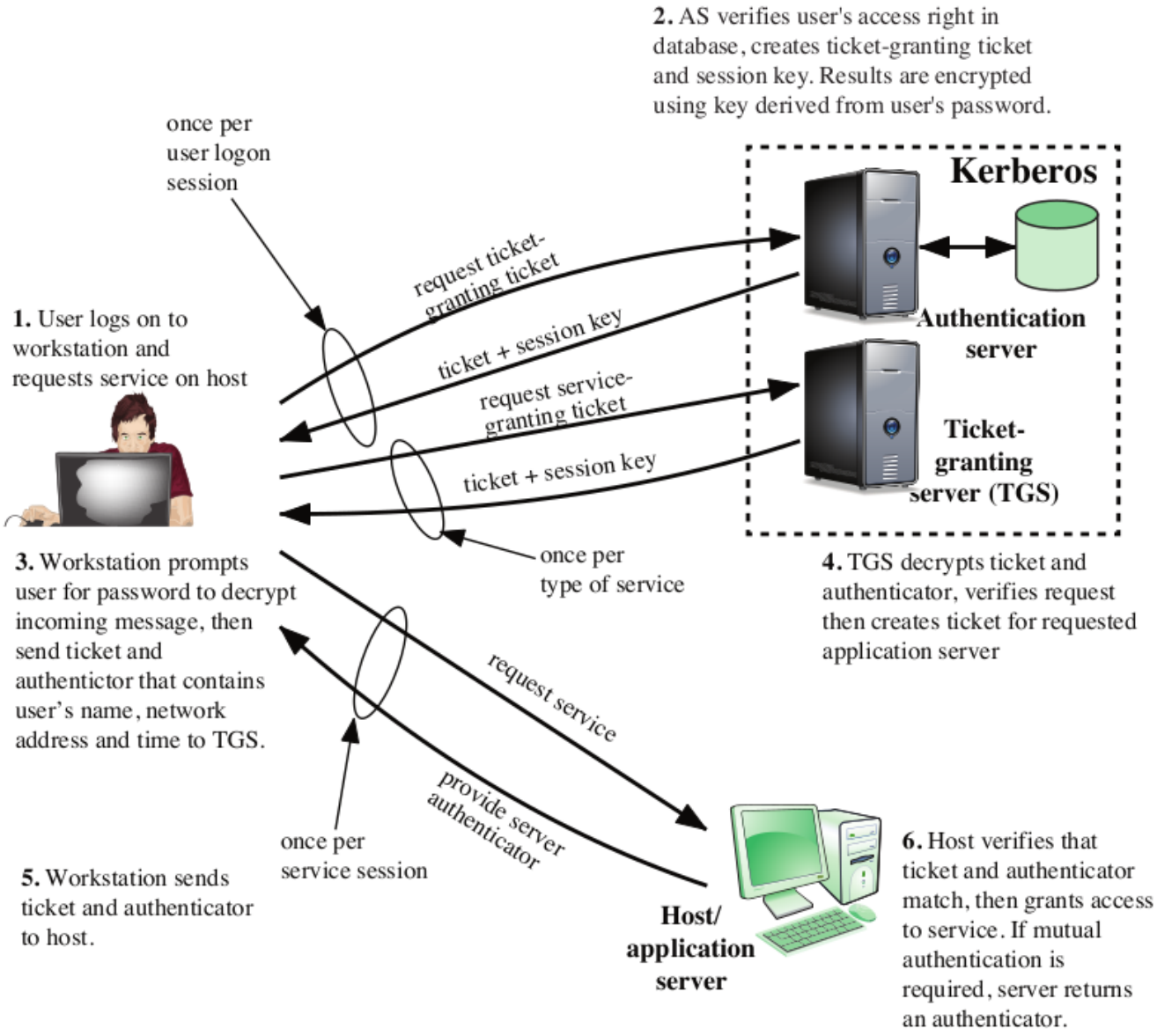
Kerberos

Kerberos

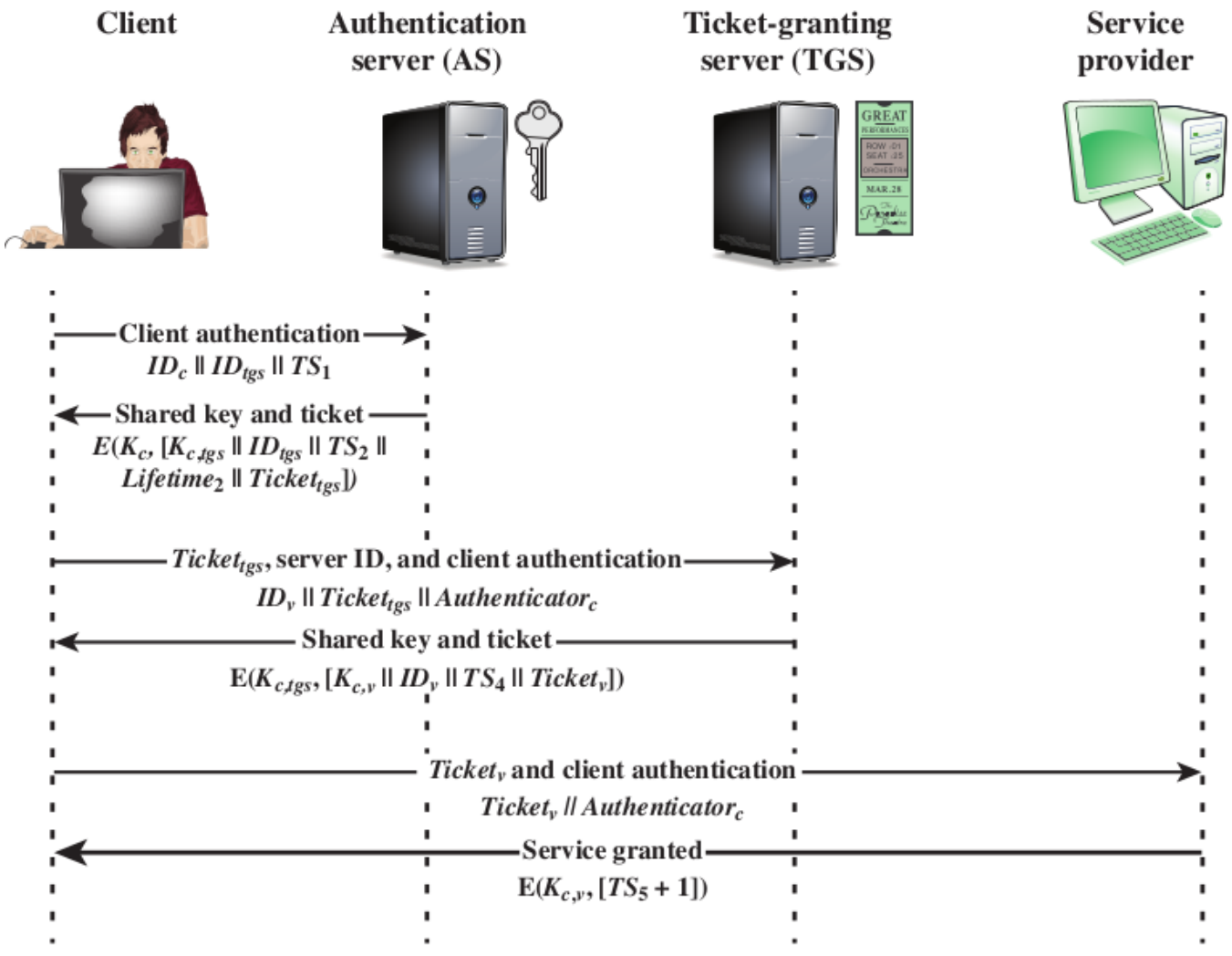
Kerberos

- Designed by MIT.
- Scenario:
  Assume an open distributed environment in which users at workstations wish to access services on servers distributed throughout the network.

Kerberos


2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.
once per user logon session
Kerberos
request ticket-granting ticket
ticket + session key
Authentication server
1. User logs on to workstation and requests service on host
request service-granting ticket
ticket + session key
Ticket-granting server (TGS)
once per type of service
4. TGS decrypts ticket and authenticator, verifies request then creates ticket for requested application server
3. Workstation prompts user for password to decrypt incoming message, then send ticket and authentictor that contains user's name, network address and time to TGS.
request service
provide server authenticator
once per service session
5. Workstation sends ticket and authenticator to host.
Host/ application server
6. Host verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

Kerberos


Client
Authentication server (AS)
Ticket-granting server (TGS)
Service provider
Client authentication
$ID_c \| ID_{tgs} \| TS_1$
Shared key and ticket
$E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$
$Ticket_{tgs}$, server ID, and client authentication
$ID_v \| Ticket_{tgs} \| Authenticator_c$
Shared key and ticket
$E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_4 \| Ticket_v])$
$Ticket_v$ and client authentication
$Ticket_v \| Authenticator_c$
Service granted
$E(K_{c,v}, [TS_5 + 1])$

The End of Crypto

Farewell, Crypto

What you should know:

1. Symmetric key encryption: DES, 3DES, AES.
2. Asymmetric key encryption: RSA.
3. Hash function.
4. Key Exchange: Diffie-Hellman.
5. How to read security protocols.
6. How to define the security of crypto.

So Now I Know What Crypto is

Unfortunately, you are not even in the crypto gateway. I skip a lot of things:

1. Prime.
2. Number theory.
3. Other public key schemes, like ECC.

More Crypto Topics

1. Zero knowledge commitment.
2. Identity-based encryption.
3. Attribute-based encryption.
4. Homomorphic encryption.
5. Post-quantum encryption.
6. Blockchain.