Public-key Cryptography

Dr. Chen Zhang

Department of Computer Science
The Hang Seng University of Hong Kong

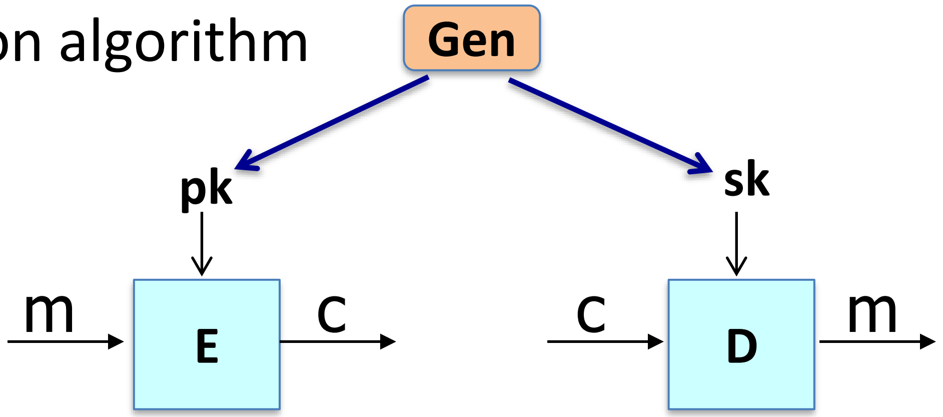
Slides partially adapted from lecture notes by M. Goodrich&R. Tamassia, W. Stallings&L. Brown, and Dan Boneh.

Recall Symmetric Encryption

- Main issue in symmetric encryption: secure key exchange
- Some methods to solve the key exchange issue:
  - Share the secret key before encrypting
    - Problem: Need to store too much keys (n people has n(n-1)/2 keys)
  - Use key distribution center (KDC): allocate a secret key for each user. Then generate a session key for each session. Use each user's secret key to encrypt the session key.
    - Problem: If the KDC is corrupted, all data will be leaked.
  - Public key encryption

Public Key Encryption

- Each person has two keys (also known as key pair)
  - Private key sk
  - Public key pk
- The public key encryption consists of Gen, E, and D
  - Gen is used to generate public key and secret key
  - E denotes the encryption algorithm
  - D denotes the decryption algorithm

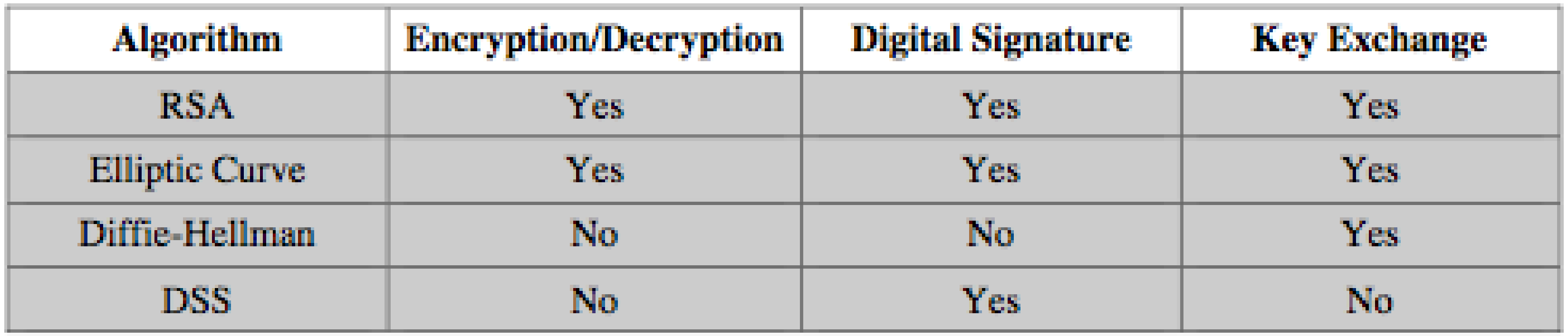


Public-Key Requirements

- Public-Key algorithms rely on two keys where:
  - it is computationally infeasible to find decryption key knowing only algorithm & encryption key
  - it is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
  - either of the two related keys can be used for encryption, with the other used for decryption (optional)
- These are difficult requirements which only a few algorithms have satisfied

Public-Key Applications

- Can classify uses into 3 categories:
  - **key exchange** (of session keys)
  - **encryption/decryption** (provide secrecy)
  - **digital signatures** (provide authentication)
- Some algorithms are suitable for all uses, others are specific to one



Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Diffie-Hellman **Key Exchange**

- Its goal is to securely establish a channel to create and share a key for symmetric key algorithms.
- First published public-key algorithm
- By Diffie and Hellman in 1976 along with the exposition of public key concepts
- Used in a number of commercial products
- Practical method to exchange a secret key securely that can then be used for subsequent encryption of messages
- Security relies on difficulty of computing **discrete logarithms**

Mod operation

- A mod B: The remainder after dividing A by B
  - E.g., 25 mod 12=1 , 25=12*2+1

    3 mod 12=3 , 12*0+3
- Logarithm: the exponent or power to which a base must be raised to yield a given number.
  - E.g., x is the logarithm of n to the base b if $b^x = n$
  - Its not very difficult to compute logarithm.
- **Discrete Logarithms problem**
  - $b^x \ mod \ p = n$, Given (b,p,n), its difficult to know x (p is a large prime)

Diffie-Hellman Setup

Suppose Alice and Bob would like to exchange key

- They agree on global parameters:
  - large prime integer P
  - A generator g of P
- Alice and Bob generate their keys
  - Alice chooses a random number as secret key: $S_A < P$
  - Bob chooses a random number as secret key : $S_B < P$
- Alice and Bob compute the public key P_A
  - Alice's public key: $P_A = g^{S_A} \, mod \, P$
  - Bob's public key: $P_B = g^{S_B} \, mod \, P$

* Prime number : a natural number greater than 1 that has no positive divisors other than 1 and itself.

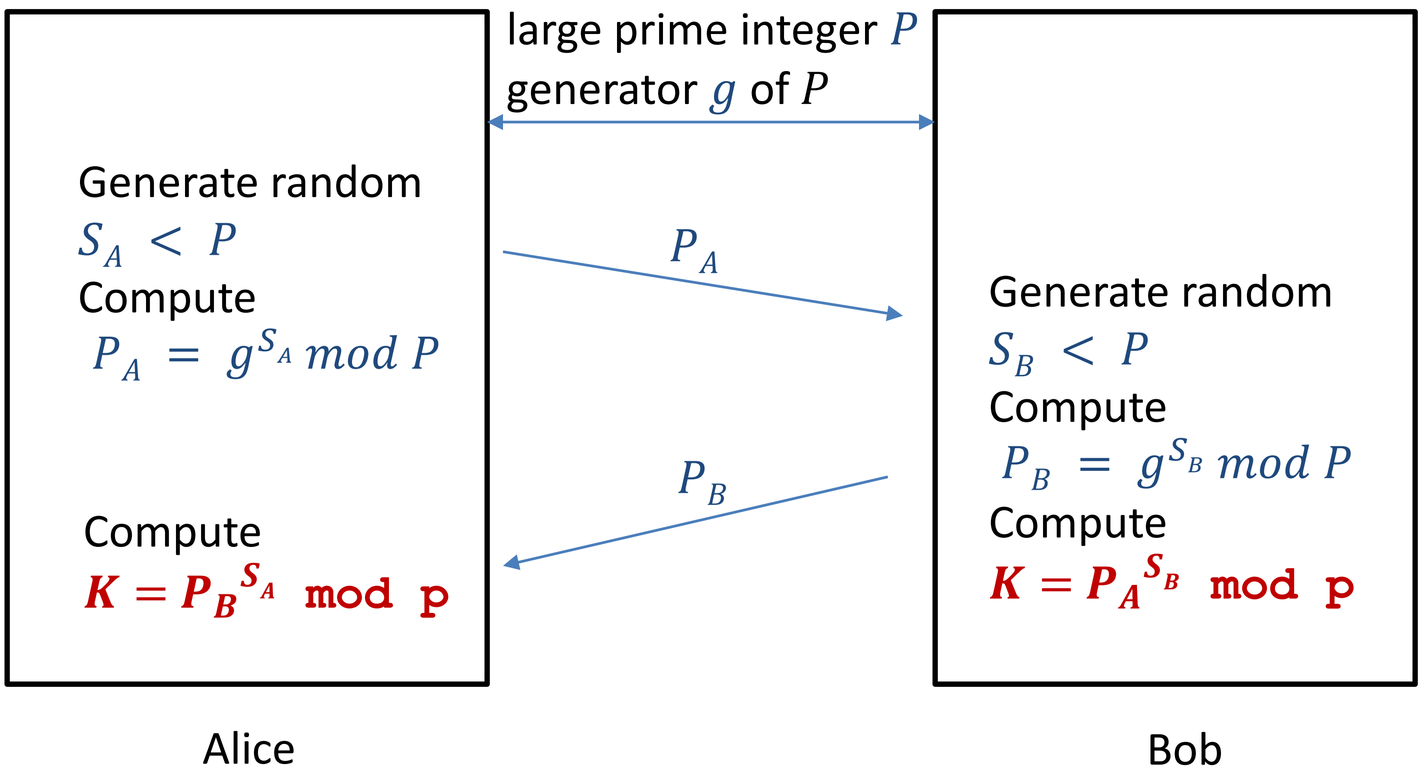
Diffie-Hellman Key Exchange

- Compute the session key K_{AB} for Alice and Bob:

$$\mathrm{K_{AB}} = g^{\mathrm{S}_A \cdot \mathrm{S}_B} \bmod q$$
$$= {P_B}^{S_A} \; mod \;\; p \quad \text{(which Alice can compute)}$$
$$= {P_A}^{S_B} \; mod \;\; p \quad \text{(which Bob can compute)}$$

- $\mathrm{K_{AB}}$ is used as session key in private-key encryption scheme between Alice and Bob
- if Alice and Bob subsequently communicate, they will have the **same** key as before, unless they choose new public-keys.
- To get S_A, must solve discrete log problem.

Key Exchange Protocols



Diffie-Hellman Key Exchange

Diffie-Hellman Example

- users Alice & Bob who wish to swap keys:
- agree on prime $p=353$ and $g=3$
- select random secret keys:
  - A chooses $S_A=97$, B chooses $S_B=233$
- compute respective public keys:
  - $P_A=3^{97} \bmod 353 = 40$ (Alice)
  - $P_B=3^{233} \bmod 353 = 248$ (Bob)
- compute shared session key as:
  - $K_{AB}= P_B^{S_A} \bmod 353 = 248^{97} = 160$ (Alice)
  - $K_{AB}= P_A^{S_B} \bmod 353 = 40^{233} = 160$ (Bob)

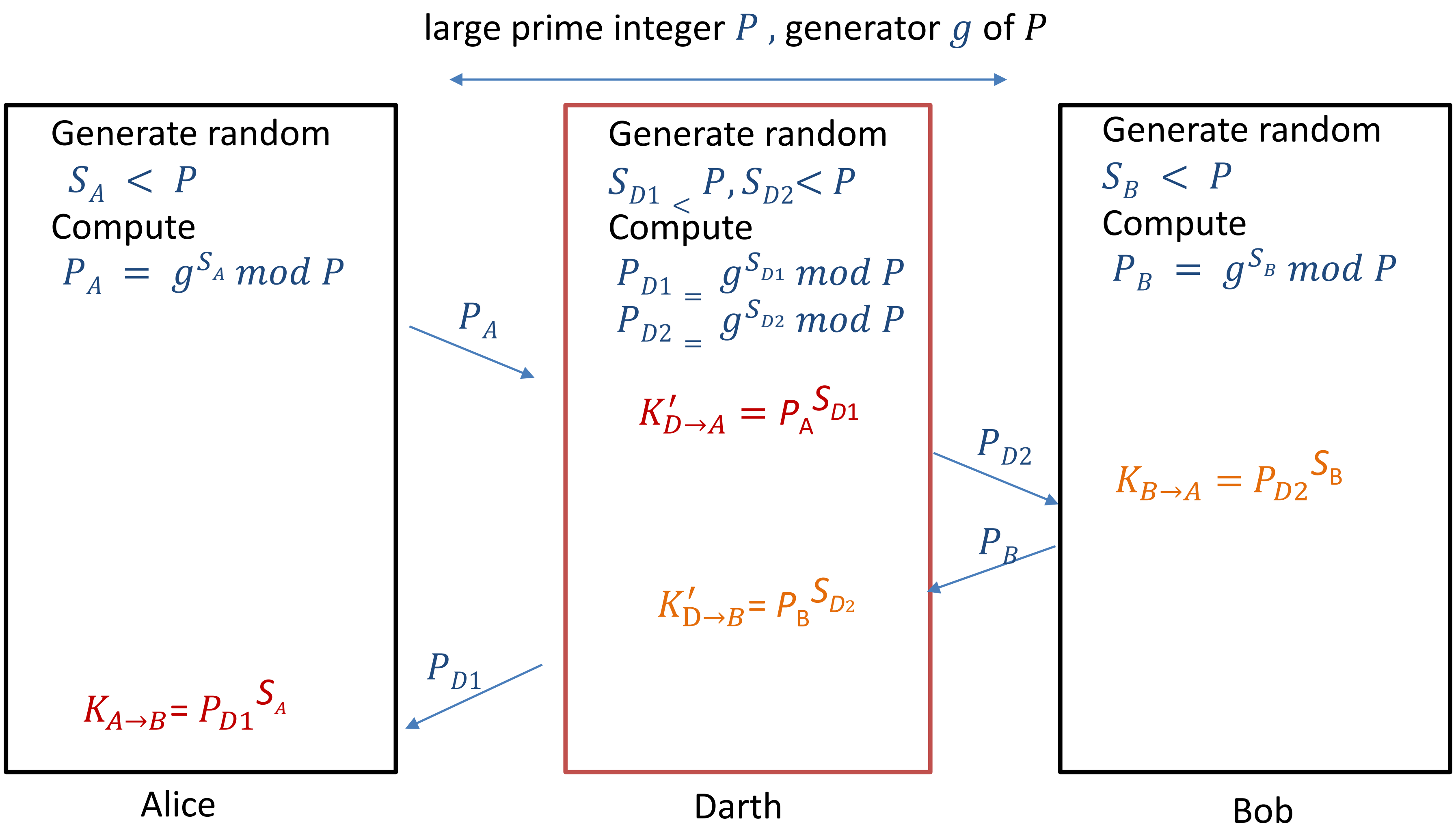
Remarks on Key Exchange Protocols

- Users could create random private/public D-H keys each time they communicate
- Users could create a known private/public D-H key and publish in a directory, then consulted and used to securely communicate with them
- Both of these are vulnerable to a **man-in-the-middle** Attack
- Authentication of the keys is needed

Man-in-the-Middle Attack

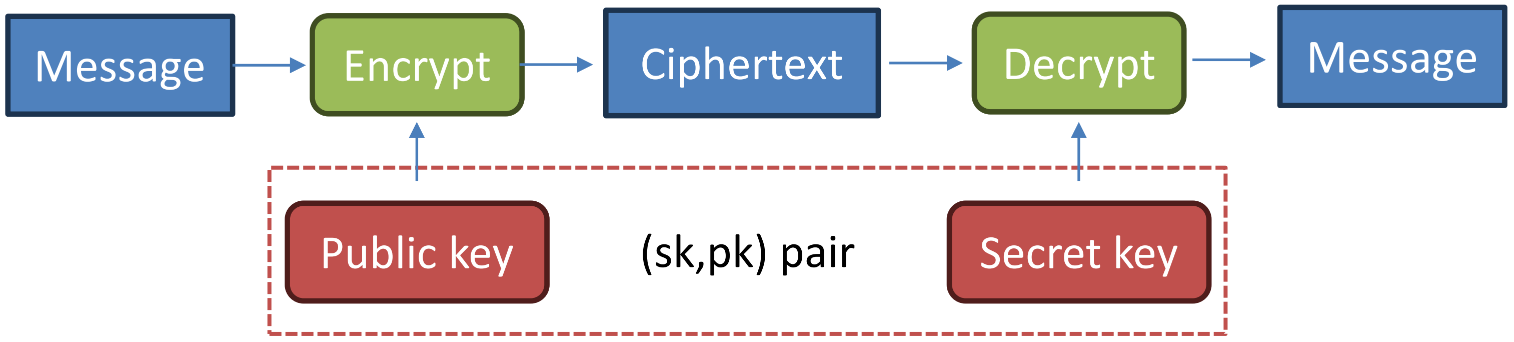
- The attacker secretly relays and possibly alters the communications between two parties who believe that they are directly communicating with each other, as the attacker has inserted themselves between the two parties.
- Attack is:
  1. Attacker Darth generates private keys S_{D1} & S_{D2}, and their public keys P_{D1} & P_{D2}
  2. Alice transmits P_A to Bob
  3. Darth intercepts P_A and transmits P_{D2} to Bob. Darth also calculates $K'_{D\rightarrow A} = {P_A}^{S_{D1}}$
  4. Bob receives P_{D2} and calculates $K_{B\rightarrow A} = {P_{D2}}^{S_B}$
  5. Bob transmits P_B to Alice
  6. Darth intercepts P_B and transmits P_{D1} to Alice. Darth calculates $K'_{D\rightarrow B} = {P_B}^{S_{D2}}$
  7. Alice receives P_{D1} and calculates $K_{A\rightarrow B} = {P_{D1}}^{S_A}$
- all subsequent communications compromised

Man-in-the-Middle Attack


large prime integer P , generator g of P
Generate random
$S_A < P$
Compute
$P_A = g^{S_A} \bmod P$
$K_{A\to B} = P_{D1}^{S_A}$
Generate random
$S_{D1} < P, S_{D2} < P$
Compute
$P_{D1} = g^{S_{D1}} \bmod P$
$P_{D2} = g^{S_{D2}} \bmod P$
$K'_{D\to A} = P_A^{S_{D1}}$
$K'_{D\to B} = P_B^{S_{D2}}$
Generate random
$S_B < P$
Compute
$P_B = g^{S_B} \bmod P$
$K_{B\to A} = P_{D2}^{S_B}$
P_A
P_{D1}
P_{D2}
P_B
Alice
Darth
Bob

Public-key Encryption/Decryption

- Alice with public key and secret key (pk,sk)
  - Alice publishes pk.
  - Anyone can send messages encrypted by pk.
  - Only Alice can decrypt messages using sk.



Public-key Encryption

RSA Public-Key Encryption

- By Rivest, Shamir & Adleman of MIT in 1977
- Best known and widely used public-key algorithm
- Uses exponentiation of integers modulo a prime
- Encrypt: $C = M^e \bmod n$
- *Decrypt:* $M = C^d \bmod n = (M^e)^d \bmod n = M$
- both sender and receiver know values of *n* and *e*
- only receiver knows value of *d*
- public-key encryption algorithm with
  - public key $PU = \{e, n\}$ and private key $PR = \{d, n\}$.

optional

RSA Cryptosystem

- Setup:
  - ***Step1***: Generate two primes p and q, set n = pq.
  - ***Step2***: Compute L=lcm(p-1,q-1), lcm denotes least common multiple.
  - ***Step3:*** compute e: 1<e<L and gcd(e,L)=1, gcd denotes greatest common divisor.
  - ***Step4:*** Compute d: 1<d<L, e*d mod L=1 (d is the inverse of e)
- Keys:
  - Public key: $\boldsymbol{K_E} = (\boldsymbol{e, n})$
  - Private key: $\boldsymbol{K_D} = \boldsymbol{d}$
- Encryption:
  - Plaintext $\boldsymbol{M}$ in $\boldsymbol{Z_n}$
  - $\boldsymbol{C} = \boldsymbol{M}^{\boldsymbol{e}} \bmod \boldsymbol{n}$
- Decryption:
  - $\boldsymbol{M} = \boldsymbol{C}^{\boldsymbol{d}} \bmod \boldsymbol{n}$

- Example
  - Setup:
    - $\boldsymbol{p} = 17$, $\boldsymbol{q} = 19$, $\boldsymbol{n} = 17\cdot19 = 323$
    - L=lcm(p-1,q-1)=lcm(16,18)=144
    - Compute ***e*** : gcd(***e***, L)=1, some numbers maintain that 5,7,11,13,17,19,25…. ,we set ***e*** = 5
    - Compute ***d: e*** * ***d*** mod L=1, we set d=29, because 5*29 mod 144=1
  - Keys:
    - public key: (***e, n***)=(5,323)
    - private key: 29
  - Encryption:
    - $\boldsymbol{M} = 123$
    - $\boldsymbol{C} = 123^5 \bmod 323 = 225$
  - Decryption:
    - $\boldsymbol{C} = 225^{29} \bmod 323 = 123$

optional

Algorithmic Issues

- The implementation of the RSA cryptosystem requires various algorithms
- Overall
  - Representation of integers of arbitrarily large size and arithmetic operations on them
- Encryption
  - Modular power
- Decryption
  - Modular power
- Setup
  - Generation of random numbers with a given number of bits (to generate candidates $\boldsymbol{p}$ and $\boldsymbol{q}$)
  - Primality testing (to check that candidates $\boldsymbol{p}$ and $\boldsymbol{q}$ are prime)
  - Computation of the GCD (to verify that $\boldsymbol{e}$ and L are relatively prime)
  - Computation of the multiplicative inverse (to compute $\boldsymbol{d}$ from $\boldsymbol{e}$)

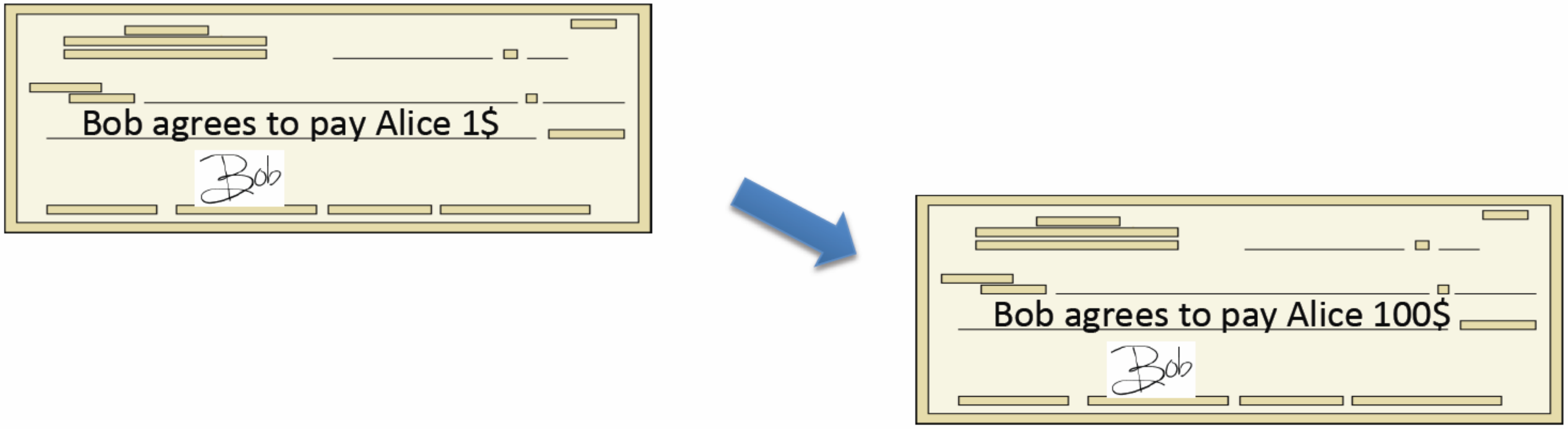
Security

- Security of RSA based on difficulty of **factoring** (For a large number N=pq. Only knowing N, its very difficult to know p and q)
  - Best known algorithm takes exponential time
- Estimated resources needed to factor a number within one year.

Length (bits)	PCs	Memory
430	1	128MB
760	215,000	4GB
1,020	342×10^6	170GB
1,620	1.6×10^{15}	120TB

Digital signatures

Goal: bind document to author

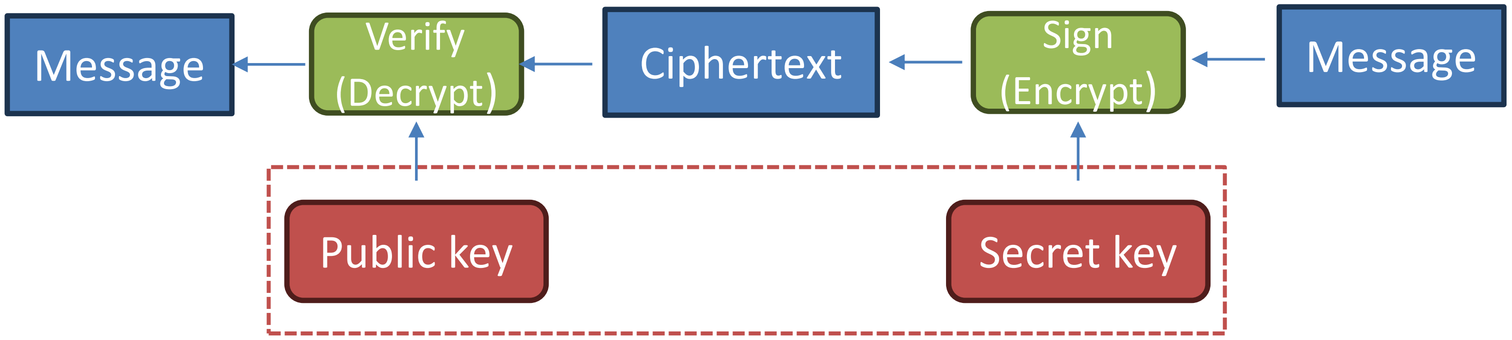


Problem: attacker can copy Bob's sig from one doc to another

Solution: make signature depend on document

Digital Signatures

- Alice with public key and secret key (pk,sk)
  - Alice publishes pk for verifying signatures.
  - Only Alice can send messages signed with sk.
  - Anyone can use pk to check messages signed by Alice.



Digital Signature

Building Block: Trapdoor Functions (TDF)

- Def: a trapdoor function over X is a triple of efficient algs. (G, F, F^{-1}):
- G(): randomized alg. Outputs a key pair (sk, pk)
- F(pk,·): defines a function $X \rightarrow Y$
- F^{-1}(sk, ·): defines a function $Y \rightarrow X$ that inverts F(pk,·)

For all x in X: F^{-1}(sk,F(pk,x))=x Encryption/Decryption

For all y in Y: F(pk,F^{-1}(sk,y))=y Digital Signature

Security: (G, F, F^{-1}) is secure if F(pk,·) is a "one-way" function:

- Given F(pk,x), it is difficult to find x (just given pk)

Digital Signatures from TDFs

- (G, F, F^{-1}): secure TDF $X \longrightarrow Y$
- $H: M \longrightarrow Y$ a hash function

Sign(sk, m∈Y) :

output

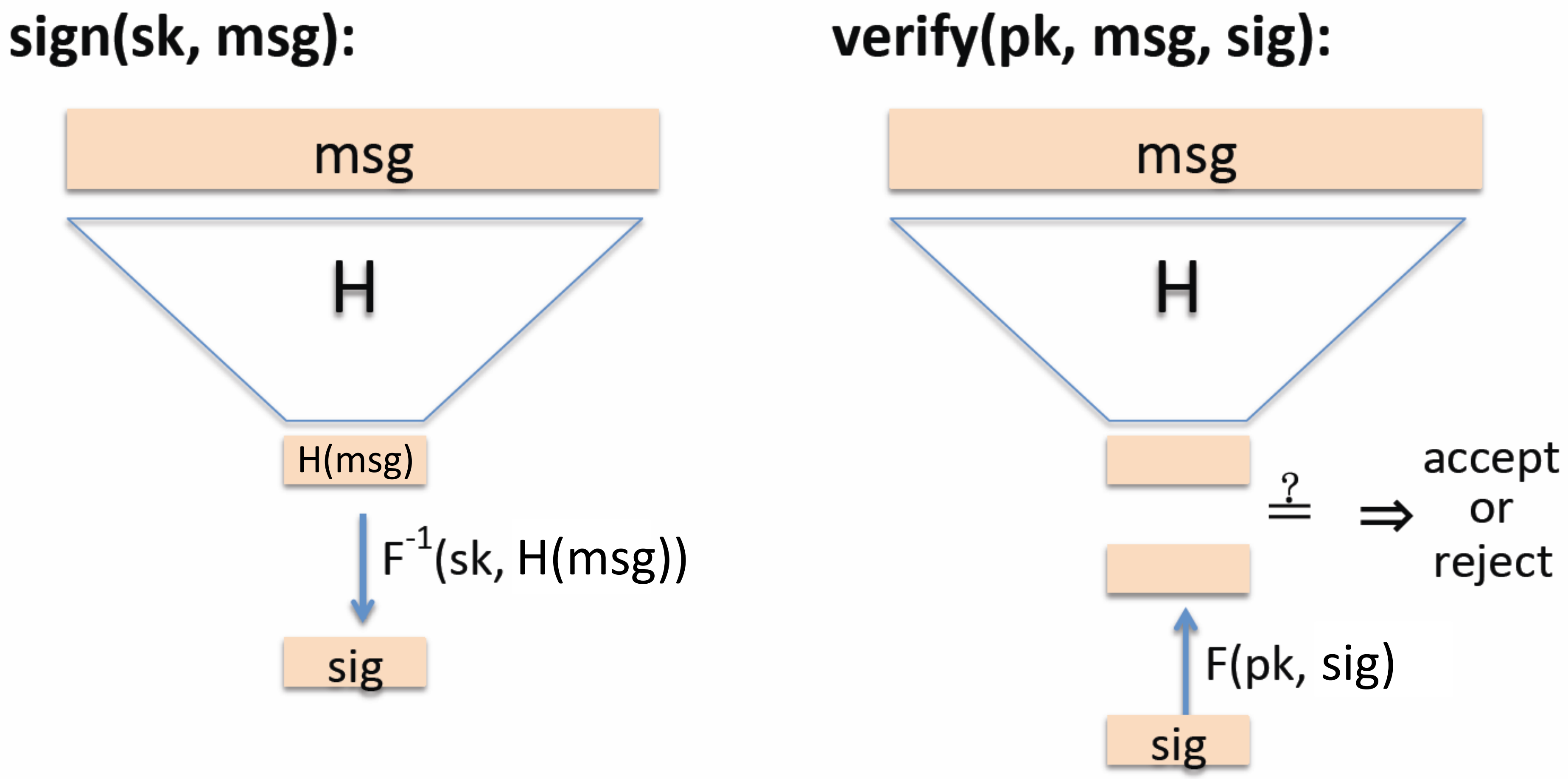
$$\text{sig} = F^{-1}\big(\text{sk}, H(m)\big)$$

Verify(pk, m, sig) :

output

$$\begin{cases} 1 & \text{if} \quad H(m) = F(\text{pk}, \text{sig}) \\ 0 & \text{otherwise} \end{cases}$$

Digital Signatures from TDFs



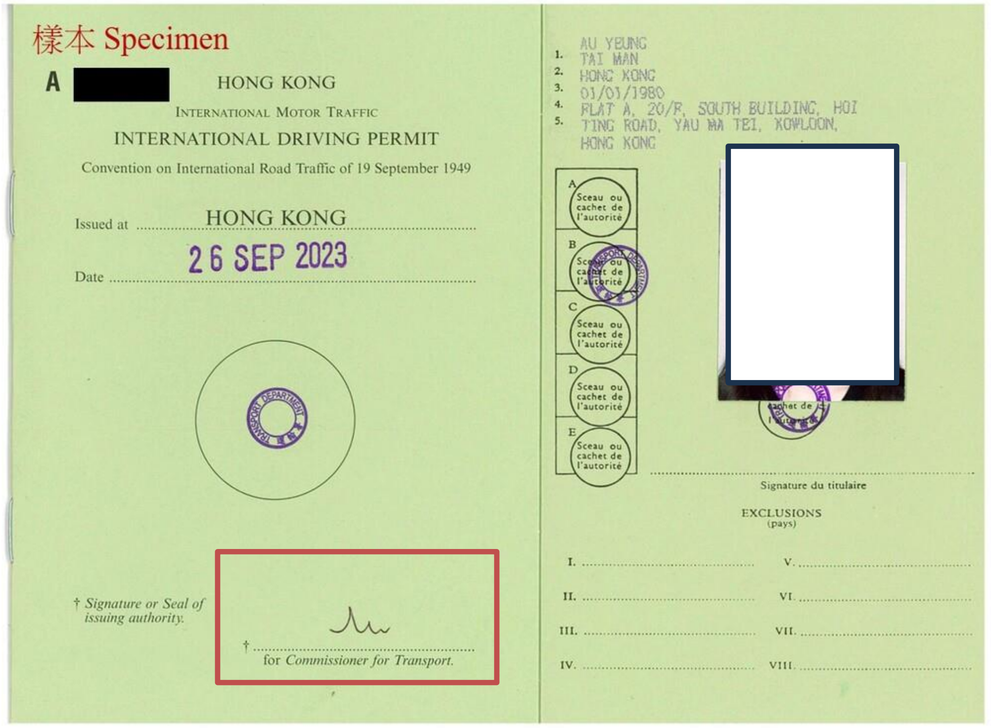
Certificate and PKI

Ensure the public key is from the real sender!

Certificates: Bind User ID with PK

- Each entity is assigned a public key certificate (PKC).
- The certificate includes some information of the entity such as name, issuer, public key, **digital signature generated by certification authority (CA), etc.**

Endorsed by a trusted third party!



樣本 Specimen

A [redacted]

HONG KONG

INTERNATIONAL MOTOR TRAFFIC

INTERNATIONAL DRIVING PERMIT

Convention on International Road Traffic of 19 September 1949

Issued at HONG KONG

Date 26 SEP 2023

† Signature or Seal of issuing authority.

† [illegible]

for Commissioner for Transport.

AU YEUNG
1. TAI MAN
2. HONG KONG
3. 01/01/1980
4. FLAT A, 20/F, SOUTH BUILDING, HOI
5. TING ROAD, YAU MA TEI, KOWLOON, HONG KONG

A Sceau ou cachet de l'autorité

B Sceau ou cachet de l'autorité

C Sceau ou cachet de l'autorité

D Sceau ou cachet de l'autorité

E Sceau ou cachet de l'autorité

cachet de l'autorité

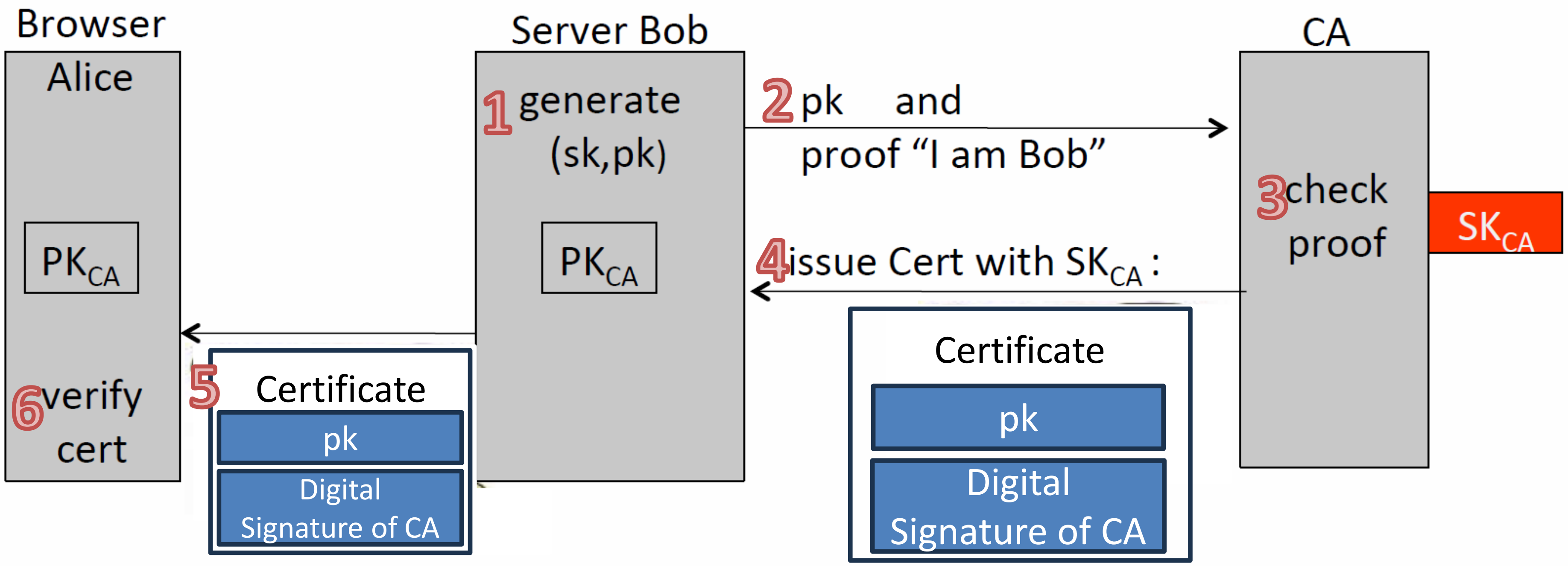
Signature du titulaire

EXCLUSIONS
(pays)

I.	V.
II.	VI.
III.	VII.
IV.	VIII.

Certificates: bind Bob's ID to his PK

How does Alice (browser) obtain Bob's public key pk_{Bob} ?



Bob uses Cert for an extended period (e.g. one year)

Who can issue the certificate?
How to revoke the certificate if the secret key is leaked?

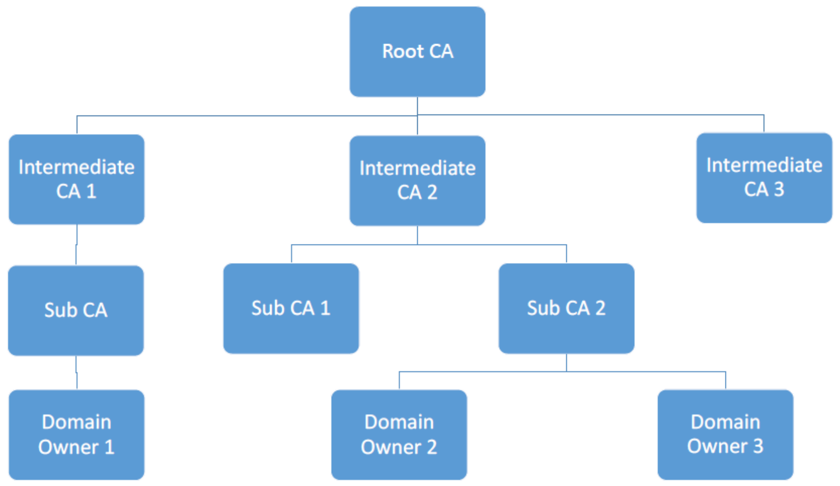
Public-Key Infrastructure

Public key infrastructure (PKI): a framework of technologies, policies, and procedures used to manage and secure the distribution of digital certificates.

- Main components of PKI:
  - User (Entity)
  - CA – issue certificates
    - Generate key pairs for users (can also be done by users)
    - Verify user identify
    - Issue certificate
    - Revoke certificate
  - Repository – the database used to store certificates

Public-Key Infrastructure

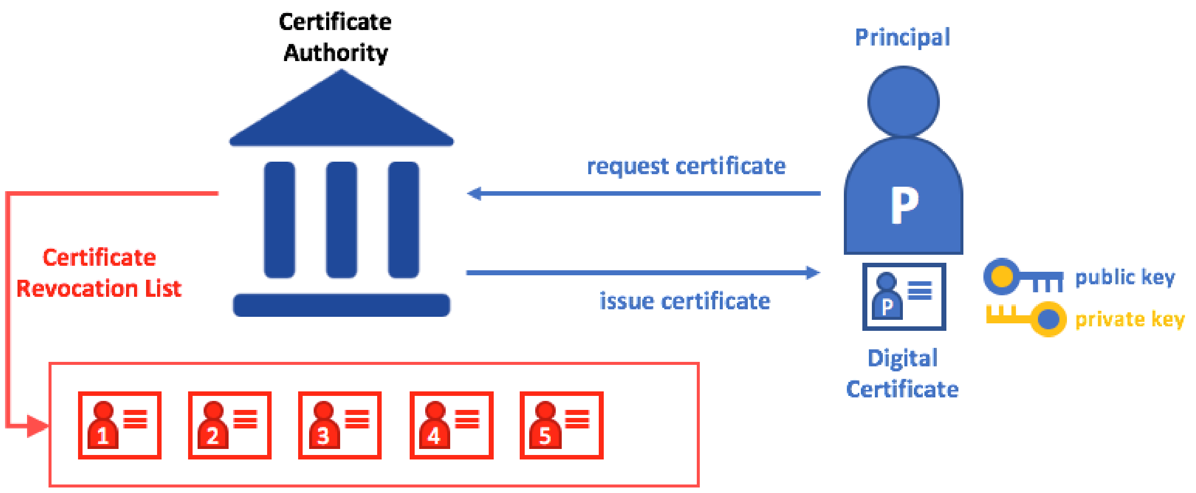
- How to verify the public key of CA?
  - Trusted root authority (VeriSign, IBM, United Nations)
    - Everyone must know the verification key of root authority
    - Check your browser; there are hundreds!!!
  - Root authority can sign certificates
  - Certificates identify others, including other authorities
  - Leads to certificate chains



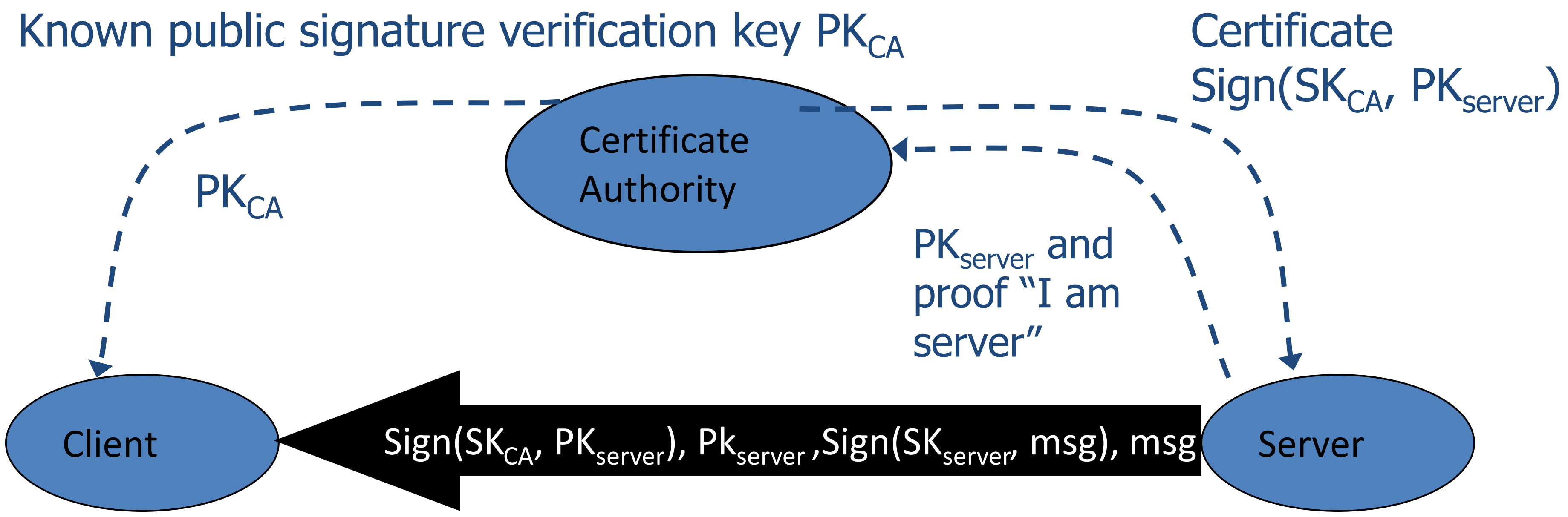
Hierarchy

Public-Key Infrastructure

- Certificate revocation list (CRL): includes all certificates have been revoked by the CA.
- Verify whether a certificate is valid: In addition to verifying the validity period and digital signature of the certificate, you also need to check whether the certificate has been revoked (i.e. download the latest CRL and check whether the certificate is in the CRL).



PKI flow chart

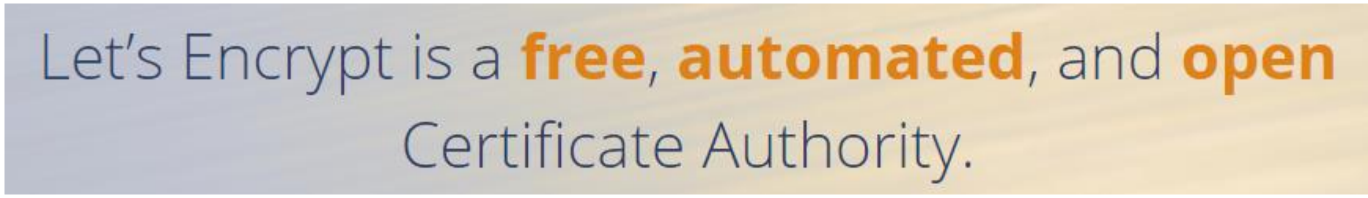

Known public signature verification key PK_{CA}
Certificate
$Sign(SK_{CA}, PK_{server})$
Certificate Authority
PK_{CA}
PK_{server} and proof "I am server"
Client
$Sign(SK_{CA}, PK_{server}), Pk_{server}, Sign(SK_{server}, msg), msg$
Server

Server certificate can be verified by any client that has CA public key $PK_{CA.}$

Certificate authority is "offline"

PKI cannot provide absolute security

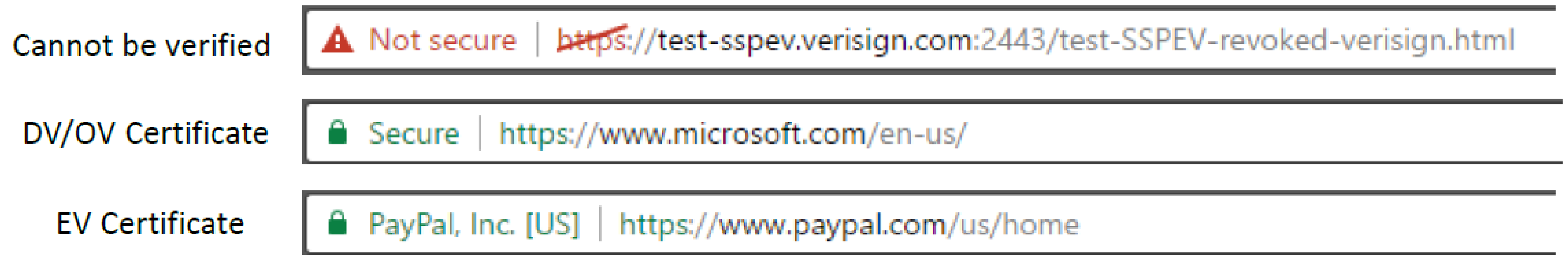
- It is very cheap, even free to get one Digital Certificate



Let's Encrypt is a **free**, **automated**, and **open** Certificate Authority.

  - Malicious website holders can easily get Digital Certificates

Chrome browser


Cannot be verified
Not secure | ~~https~~://test-sspev.verisign.com:2443/test-SSPEV-revoked-verisign.html

DV/OV Certificate
Secure | https://www.microsoft.com/en-us/

EV Certificate
PayPal, Inc. [US] | https://www.paypal.com/us/home

1.DV (domain validated): only check the validity of domain name, confirm the legitimacy of the domain but do not validate the identity of the organization or individual.

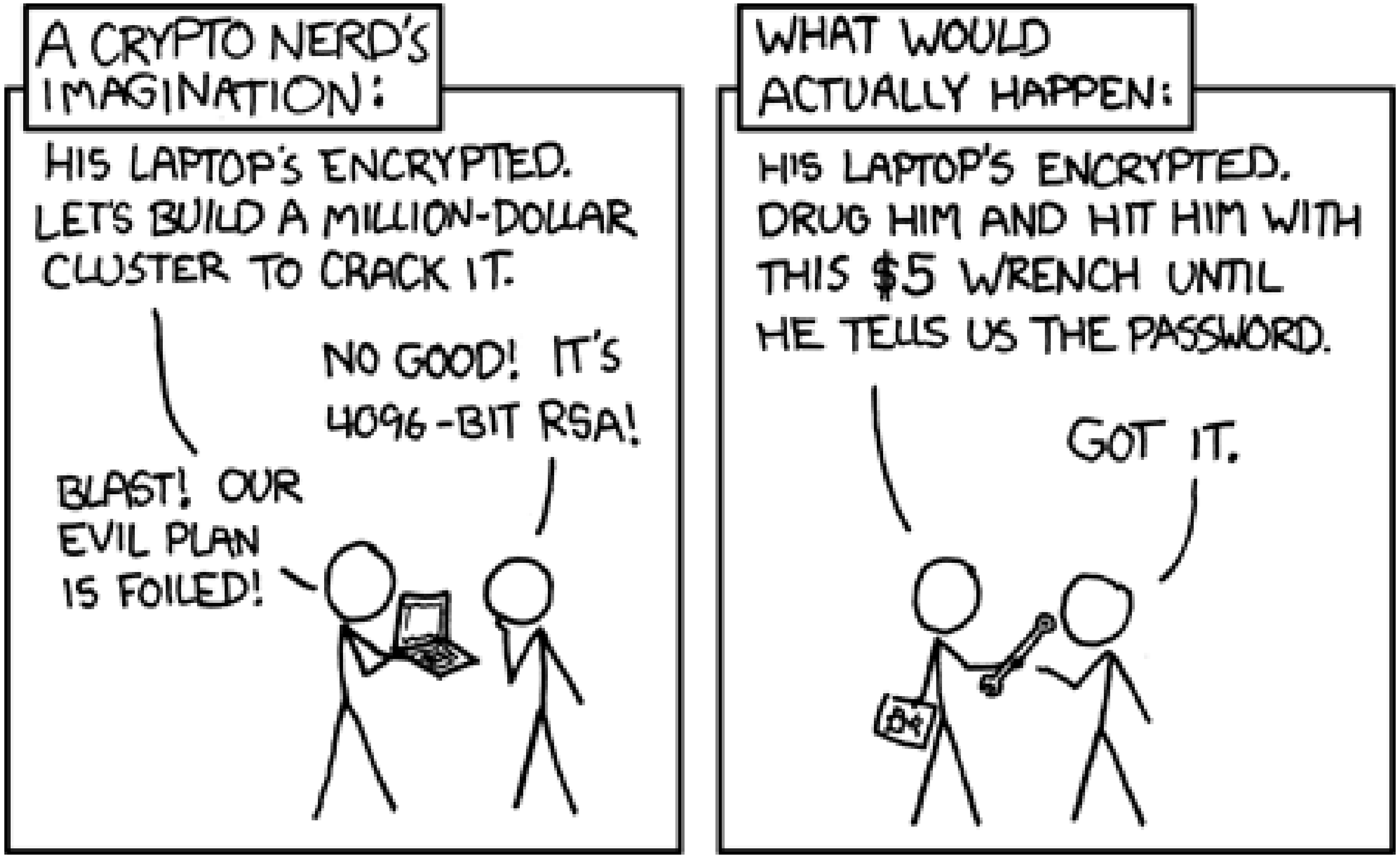
2.OV (organization validated): involve stricter validation of the applicant's identity. The CA verifies domain ownership and performs additional organization identity checks to ensure the certificate holder is a legitimate organization.

3.EV (extended validated): the highest level of validation and trust. When issuing an EV certificate, the CA conducts rigorous verification to ensure the legitimacy and true identity of the certificate holder.

Limitations of Cryptography

Limitations of cryptography

- Most security problems are not crypto problems
  - This is good
    - Cryptography works!
  - This is bad
    - People make other mistakes; crypto doesn’t solve them
- Misuse of cryptography is dangerous for security


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.