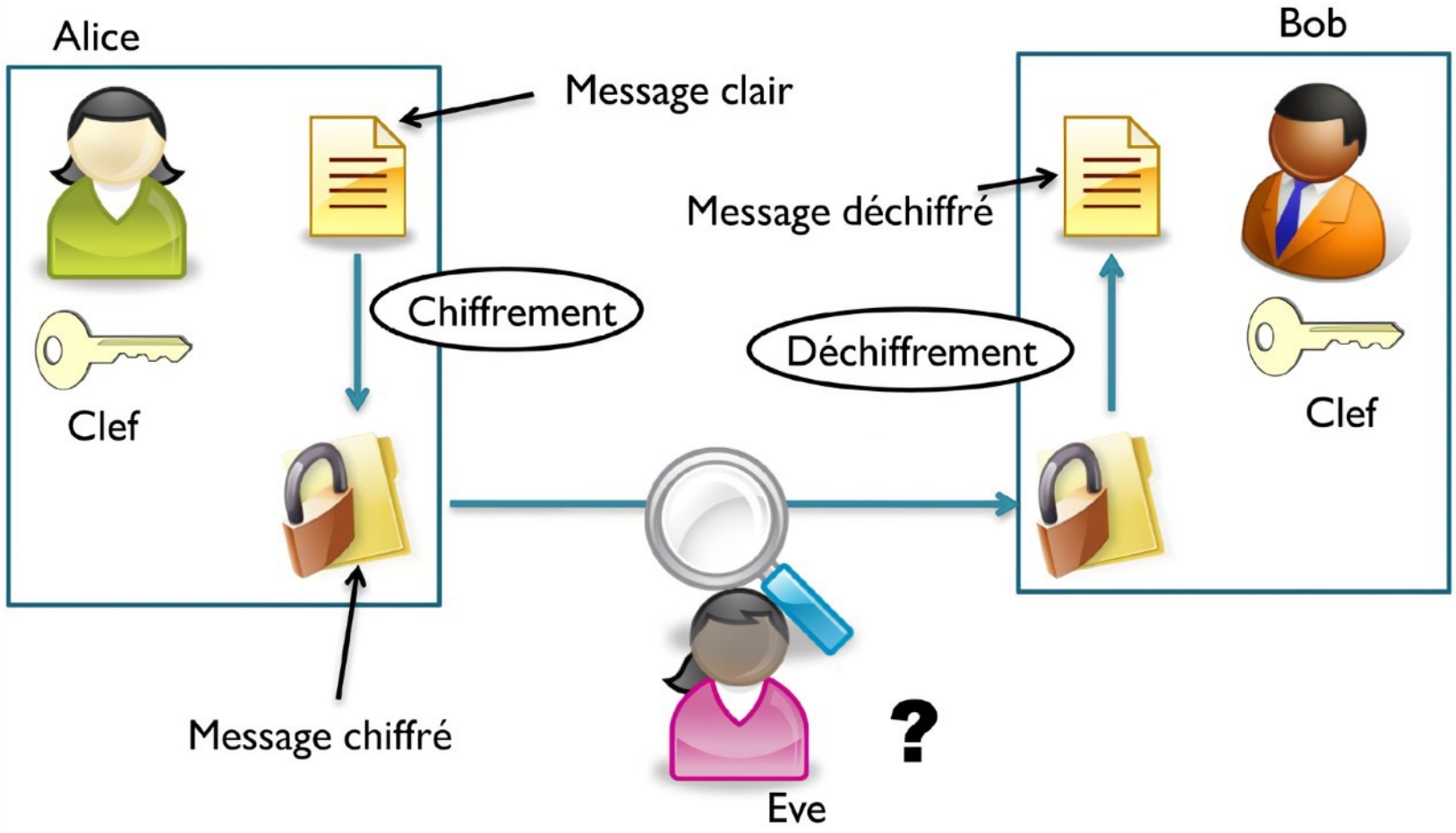
# Technologies for E-Commerce

**CAN302**

**Department of Communications and Networking**

**Xi'an Jiaotong-Liverpool University (XJTLU)**

# Week10 – Https

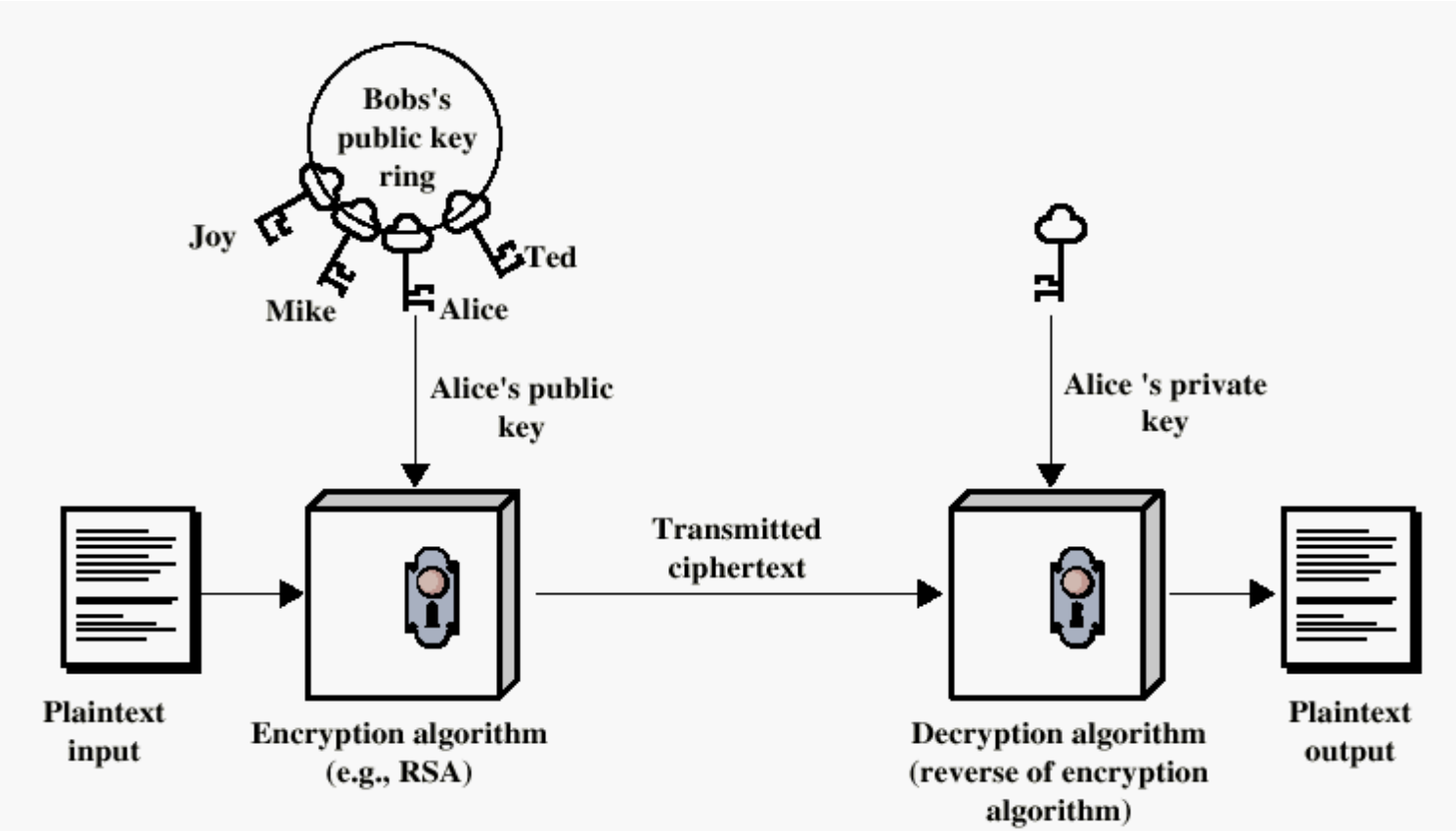



Apply encryption technologies to build a safe channel

# Some ideas about https

- Symmetric encryption is quick and safe but difficult to share the key.
- If we can find a way to share the key, we can build a secure channel.
- DH is a safe way to negotiate a key but cannot against middle-man.
- RSA/ECC can share a public key, it is a potential candidate to negotiate the key.

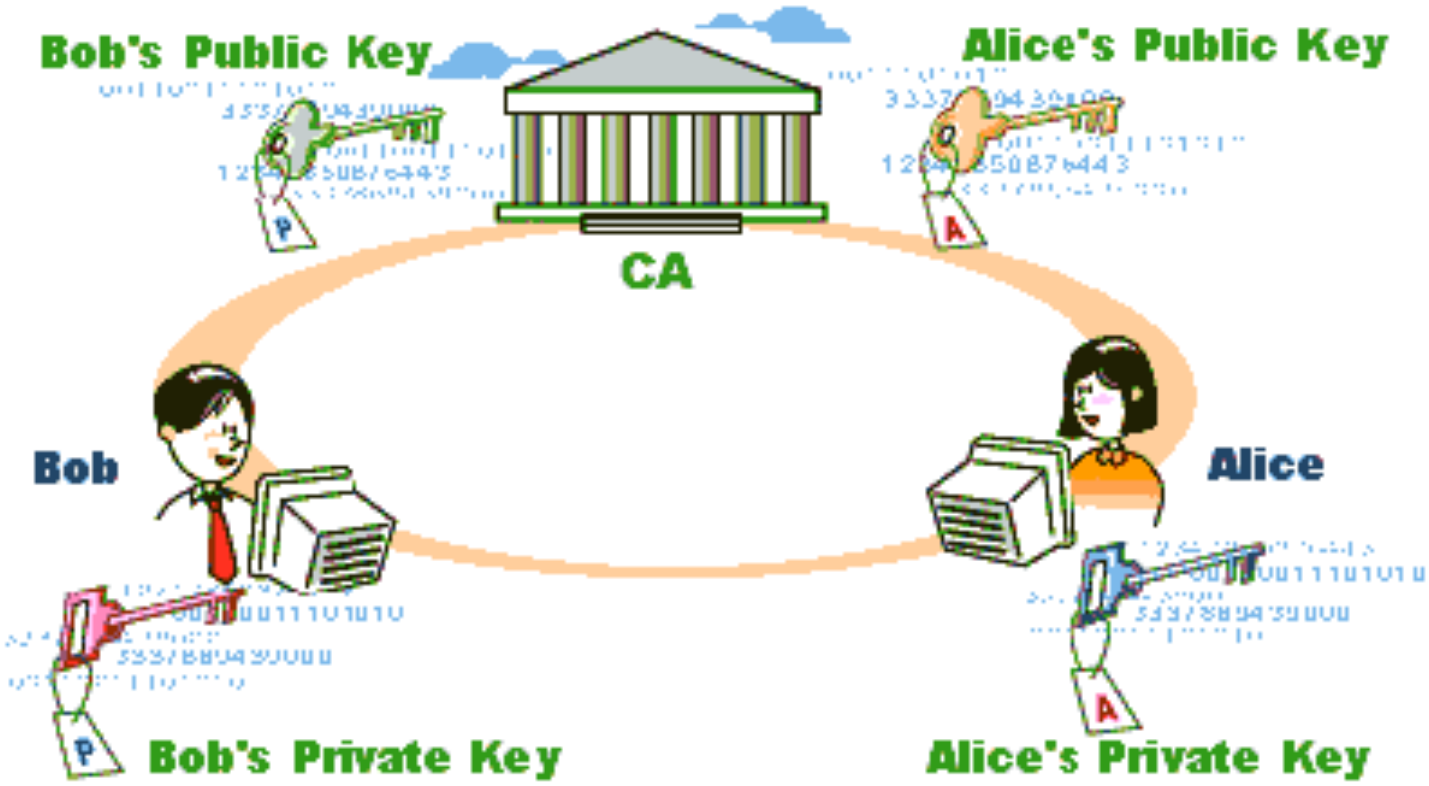
# Challenge

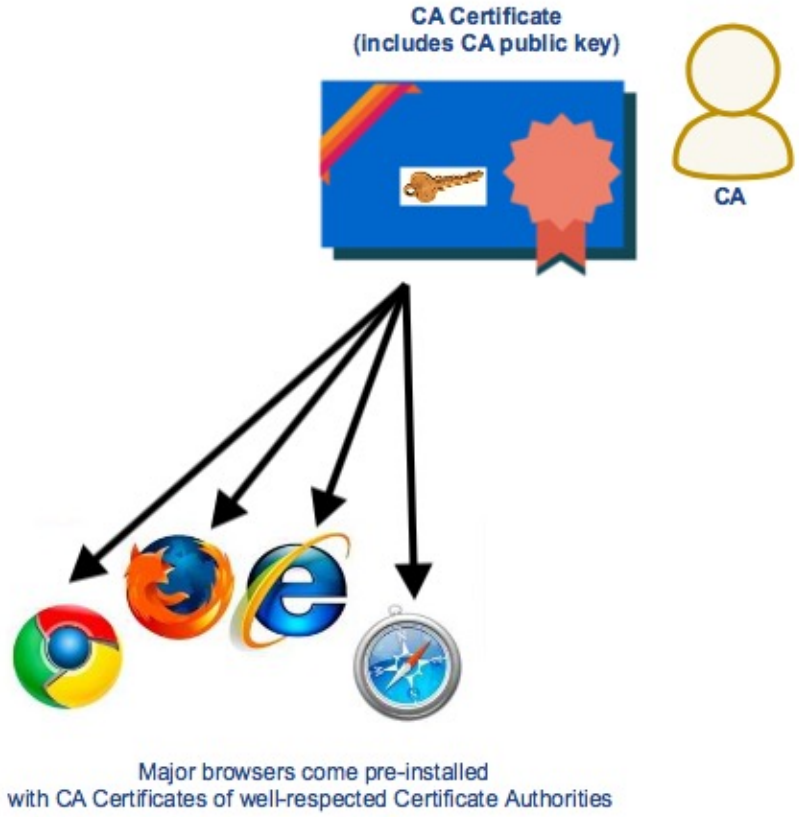



"On the Internet, nobody knows you're a dog."

- How to safely collect public keys of Alice, Ted, Mike and so on.
- How to trust {5, 2407} is a public key of the target web that never visited before?

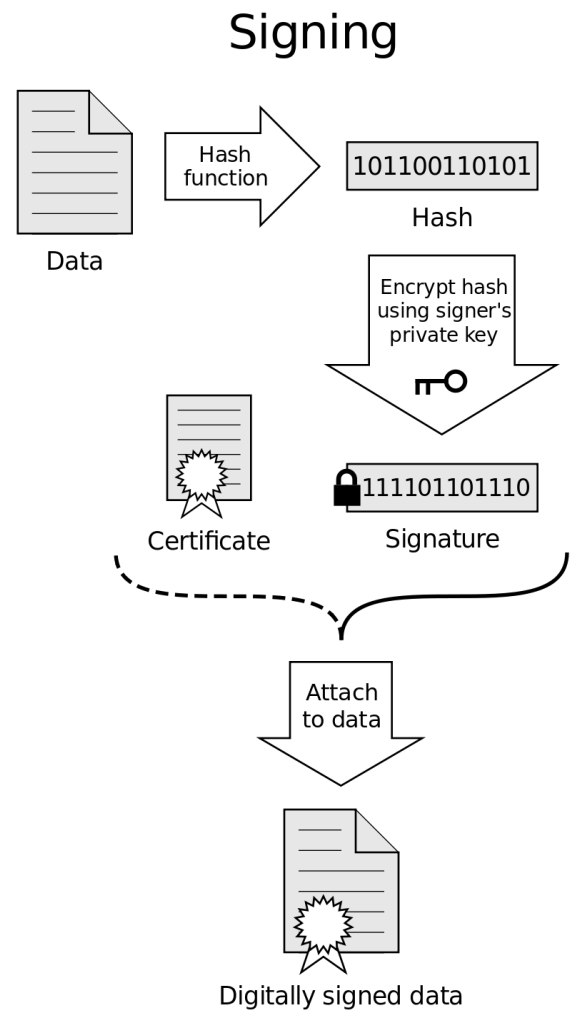
# Certificate Authority

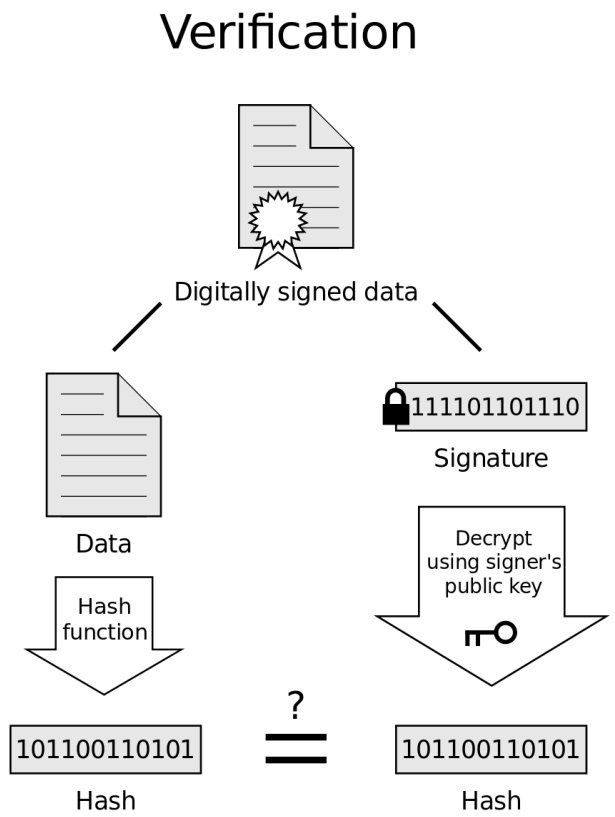



https://www.comodo.com/resources/small-business/digital-certificates5.php
https://www.jscape.com/blog/an-overview-of-how-digital-certificates-work

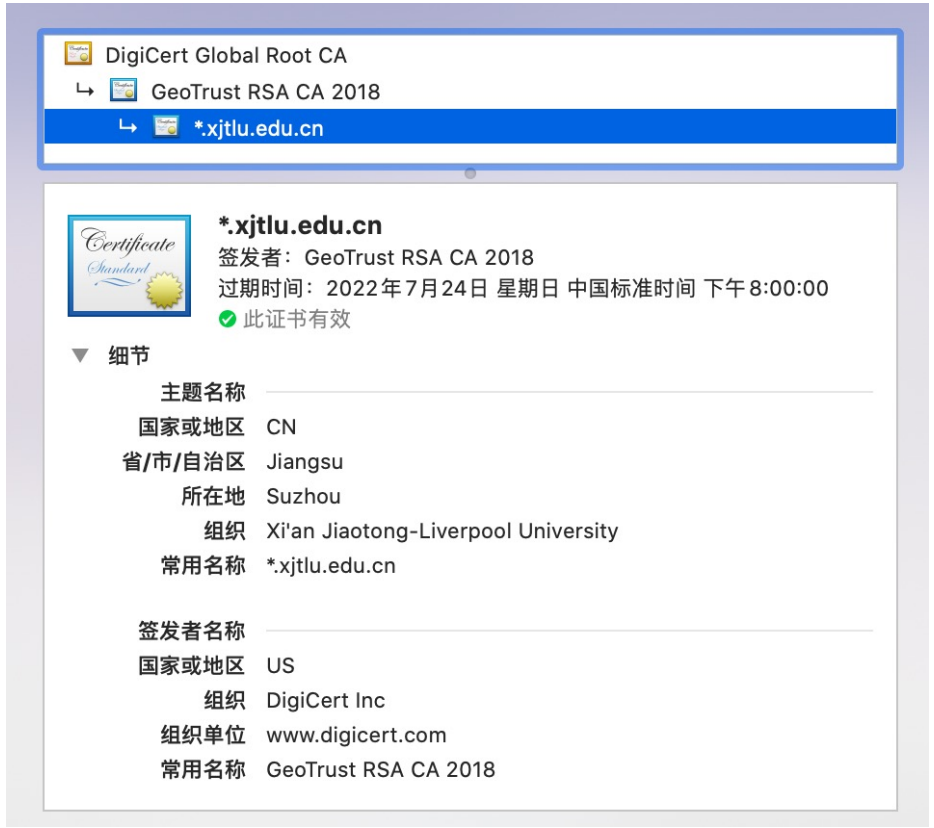
# Digital signature

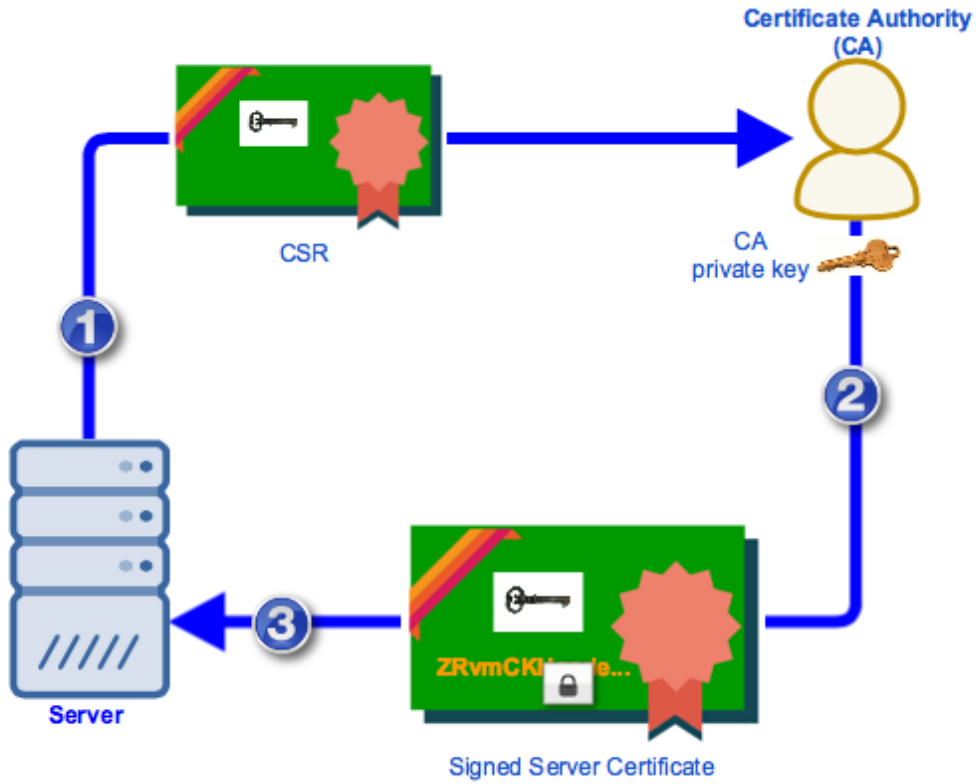



https://www.pubnub.com/blog/chat-security-user-identification-with-digital-signature-message-verification/

# The certificate

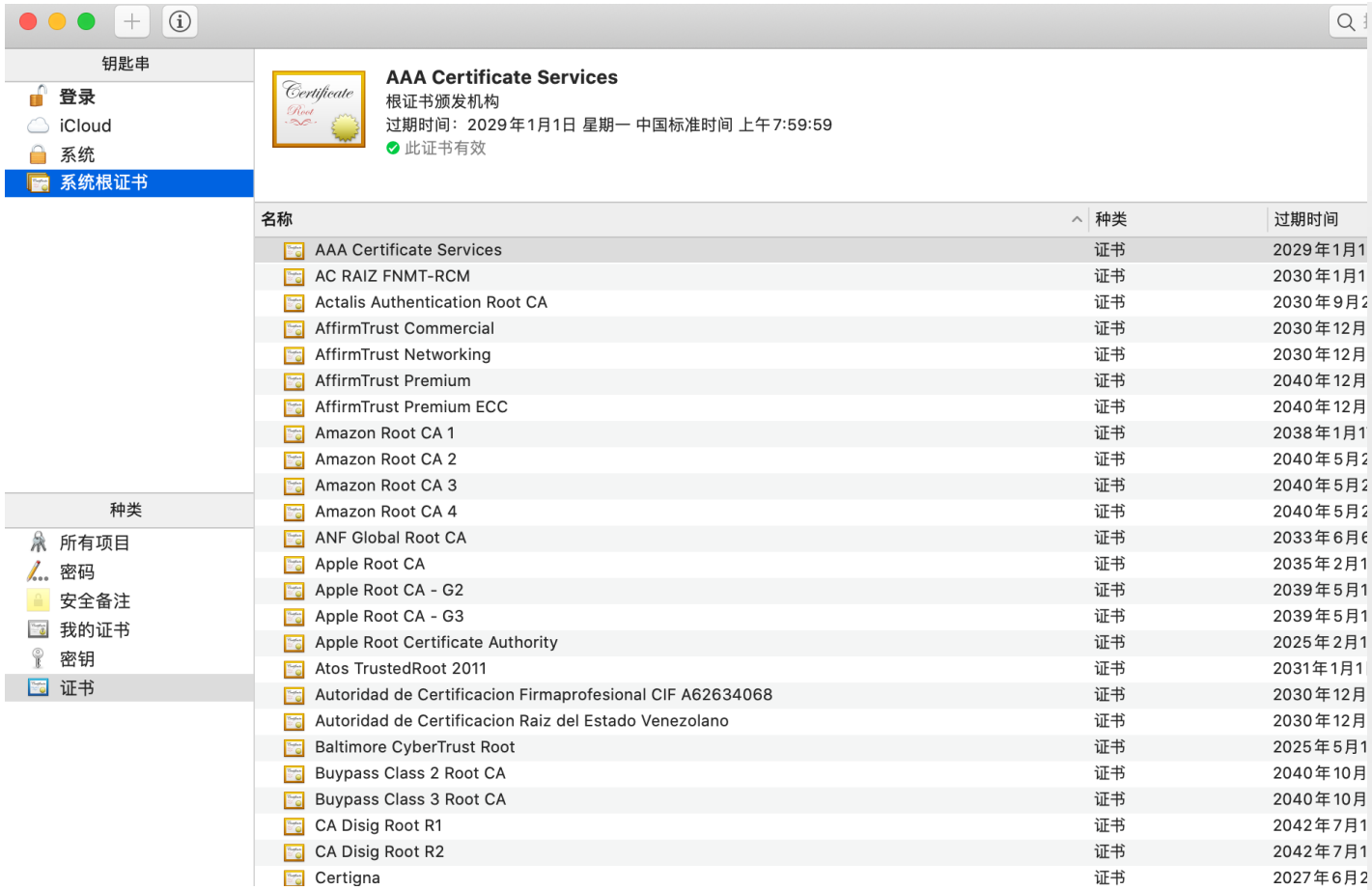


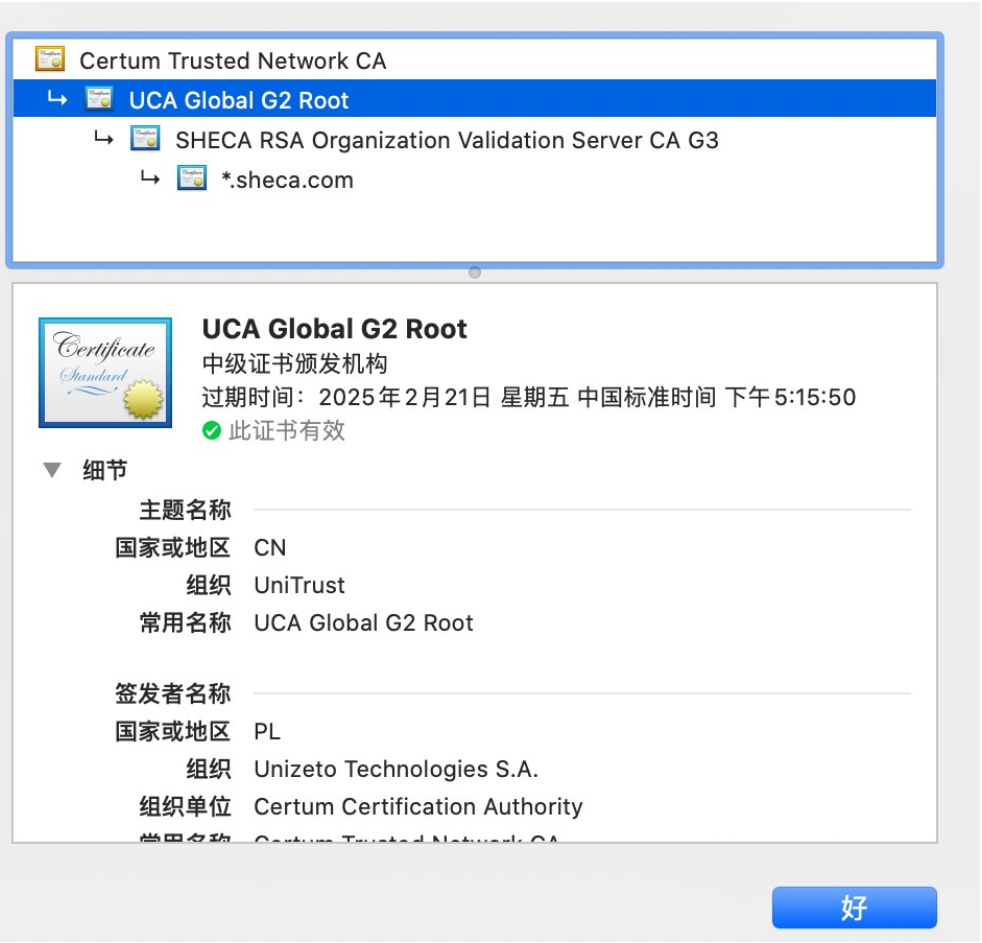




- Webmaster create the key pairs and only send the public key to CA for registration.
- CA need to verify the authority of CRS application.

https://www.comodo.com/resources/small-business/digital-certificates5.php
https://www.jscape.com/blog/an-overview-of-how-digital-certificates-work
https://www.myssl.cn/tools/

# Major CA in our browsers



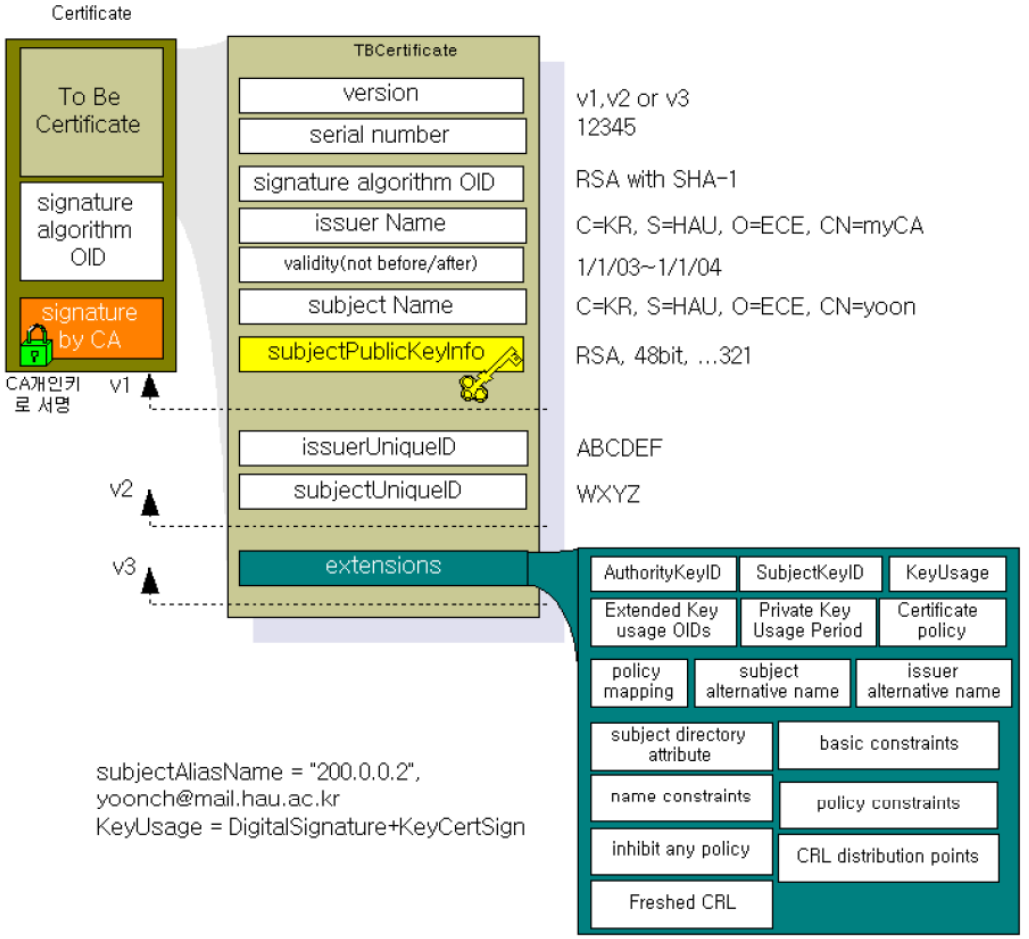

- Dominated by US and European companies.
- 上海CA has been accepted by most browsers.

# ITU-T X.509

SectigoRSADomainValida

SectigoRSADomainValidationSecureServerCA.crt

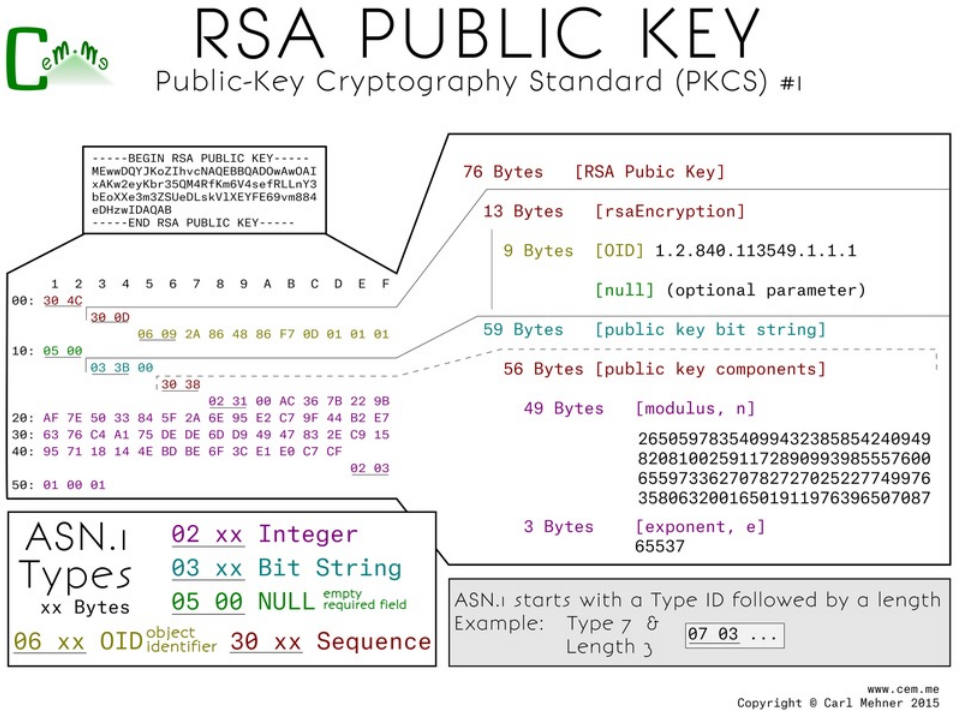
```
-----BEGIN CERTIFICATE-----
MIIGEzCCA/ugAwIBAgIQfVtRJrR2uhHbdBYLvFMNpzANBgkqhkiG9w0BAQwFADCB
iDELMAkGA1UEBhMCVVMxEzARBgNVBAgTCk5ldyBKZXJzZXkxFDASBgNVBAcTC0pl
cnNleSBDaXR5MR4wHAYDVQQKExVUaGUgVVNFUlRSVVNUIE5ldHdvcmsxLjAsBgNV
BAMTJVVTRVJUcnVzdCBSU0EgQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkwHhcNMTgx
MTAyMDAwMDAwWhcNMzAxMjMxMjM1OTU5WjCBjzELMAkGA1UEBhMCR0IxGzAZBgNV
BAgTEkdyZWF0ZXIgTWFuY2hlc3RlcjEQMA4GA1UEBxMHU2FsZm9yZDEYMBYGA1UE
ChMPU2VjdGlnbyBMaW1pdGVkMTcwNQYDVQQDEy5TZWN0aWdvIFJTQSBEb21haW4g
VmFsaWRhdGlvbiBTZWN1cmUgU2VydmVyIENBMIIBIjANBgkqhkiG9w0BAQEFAAOC
AQ8AMIIBCgKCAQEA1nMz1tc8INAA0hdFuNY+B6I/x0HuMjDJsGz99J/LEpgPLT+N
TQEMgg8Xf2Iu6bhIefsWg06t1zIlk7cHv7lQP6lMw0Aq6Tn/2YHKHxYyQdqAJrkj
eocgHuP/IJo8lURvh3UGkEC0MpMWCRAIIz7S3YcPb11RFGoKacVPAXJpz9OTTG0E
oKMbgn6xmrntxZ7FN3ifmgg0+1YuWMQJDgZkW7w33PGfKGioVrCSo1yfu4iYCBsk
Haswha6vsC6eep3BwEIc4gLw6uBK0u+QDrTBQBbwb4VCSmT3pDCg/r8uoydajotY
uK3DGReEY+1vVv2Dy2A0xHS+5p3b4eTlygxfFQIDAQABo4IBbjCCAWowHwYDVR0j
BBgwFoAUU3m/WqorSs9UgOHYm8Cd8rIDZsswHQYDVR0OBBYEFI2MXsRUrYrhd+mb
+ZsF4bgBjWHhMA4GA1UdDwEB/wQEAwIBhjASBgNVHRMBAf8ECDAGAQH/AgEAMB0G
A1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjAbBgNVHSAEFDASMAYGBFUdIAAw
CAYGZ4EMAQIBMFAGA1UdHwRJMEcwRaBDoEGGP2h0dHA6Ly9jcmwudXNlcnRydXN0
LmNvbS9VU0VSVHJ1c3RSU0FDZXJ0aWZpY2F0aW9uQXV0aG9yaXR5LmNybDB2Bggr
BgEFBQcBAQRqMGgwPwYIKwYBBQUHMAKGM2h0dHA6Ly9jcnQudXNlcnRydXN0LmNv
bS9VU0VSVHJ1c3RSU0FBZGRUcnVzdENBLmNydDAlBggrBgEFBQcwAYYZaHR0cDov
L29jc3AudXNlcnRydXN0LmNvbTANBgkqhkiG9w0BAQwFAAOCAgEAMr9hvQ5Iw0/H
ukdN+Jx4GQHcEx2Ab/zDcLRSmjEzmldS+zGea6TvVKqJjUAXaPgREHzSyrHxVYbH
7rM2kYb2OVG/Rr8PoLq0935JxCo2F57kaDl6r5ROVm+yezu/Coa9zcV3HAO4OLGi
H19+24rcRki2aArPsrW04jTkZ6k4Zgle0rj8nSg6F0AnwnJOKf0hPHzPE/uWLMUx
RP0T7dWbqWlod3zu4f+k+TY4CFM5ooQ0nBnzvg6s1SQ36yOoeNDT5++SR2RiOSLv
xvcRviKFxmZEJCaOEDKNyJOuB56DPi/Z+fVGjmO+wea03KbNIaiGCpXZLoUmGv38
sbZXQm2V0TP2ORQGgkE49Y9Y3IBbpNV9lXj9p5v//cWoaasm56ekBYdbqbe4oyAL
l6lFhd2zi+WJN44pDfwGF/Y4QA5C5BIG+3vzxhFoYt/jmPQT2BVPi7Fp2RBgvGQq
6jG35LWjOhSbJuMLe/0CjraZwTiXWTb2qHSihrZe68Zk6s+go/lunrotEbaGmAhY
LcmsJWTyXnW0OMGuf1pGg+pRyrbxmRE1a6Vqe8YAsOf4vmSyrcjC8azjUeqkk+B5
yOGBQMkKW+ESPMFgKuOXwIlCypTPRpgSabuY0MLTDXJLR27lk8QyKGOHQ+SwMj4K
00u/I5sUKUErmgQfky3xxzlIPK1aEn8=
-----END CERTIFICATE-----
```




https://www.itu.int/rec/T-REC-X.509/en

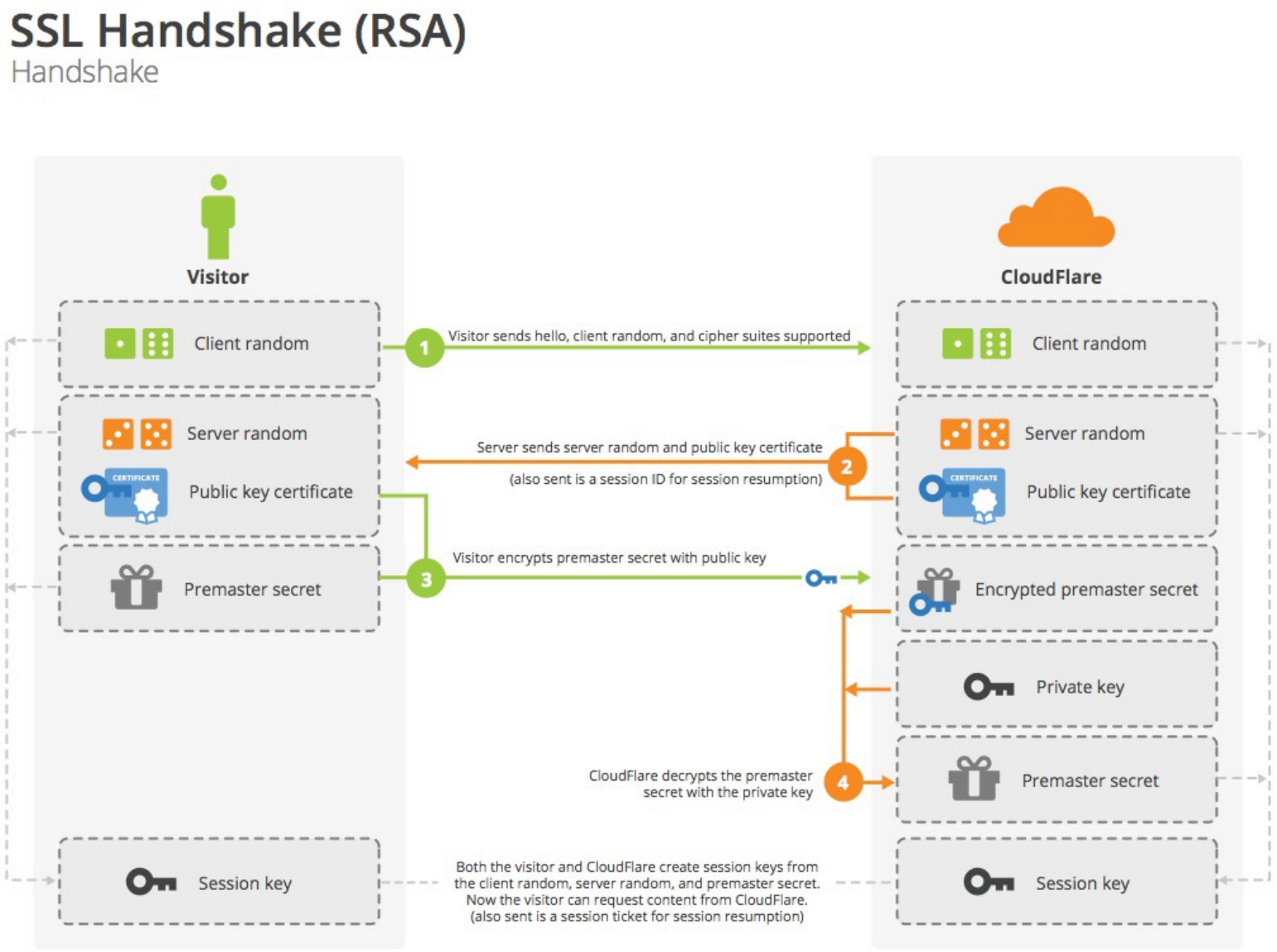
# PKCS (Public-Key Cryptography Standards)

| No. | PKCS Title | Comments |
|---|---|---|
| 1 | RSA Cryptography Standard | |
| 2,4 | | incorporated into PKCS #1 |
| 3 | Diffie-Hellman Key Agreement Standard | superseded by IEEE 1363a. |
| 5 | Password-Based Cryptography Standard | |
| 6 | Extended-Certificate Syntax Standard | never adopted |
| 7 | Cryptographic Message Syntax Standard | superseded by RFC 3369 |
| 8 | Private-Key Information Syntax Standard | |
| 9 | Selected Object Classes and Attribute Types | |
| 10 | Certification Request Syntax Standard | |
| 11 | Cryptographic Token Interface Standard | referred to as CRYPTOKI |
| 12 | Personal Information Exchange Syntax Standard | |
| 13 | (reserved for ECC) | never been published |
| 14 | (reserved for pseudo-random number generation) | never been published |
| 15 | Cryptographic Token Information Syntax Standard | |



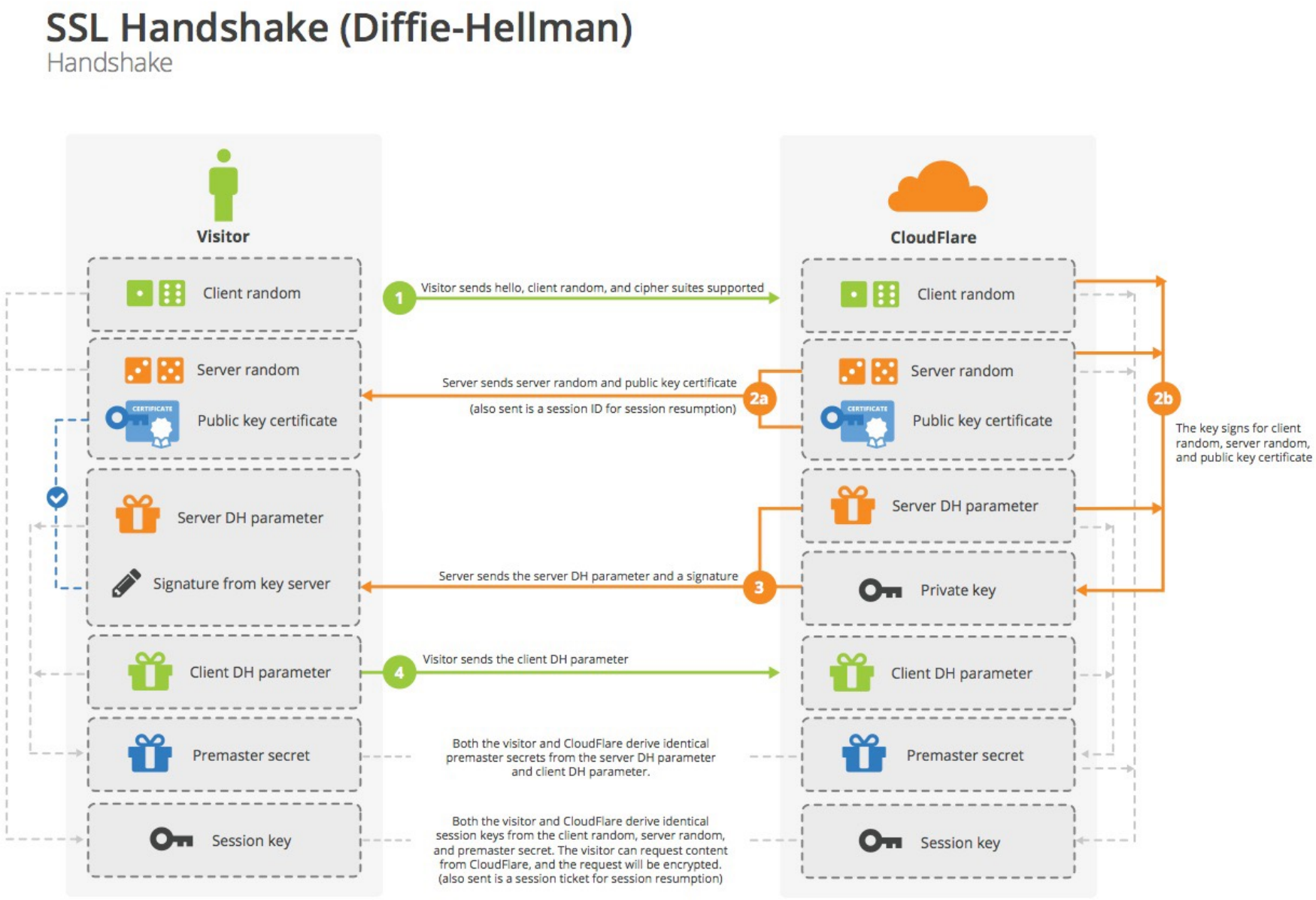

https://www.educba.com/pkcs/
https://www.encryptionconsulting.com/public-key-cryptography-standards/

# Actual https




- Session key is generated by server random, client random and pre-master

# Actual https




- DH algorithm is used to further enhance the security of pre-master.

http://www.ruanyifeng.com/blog/2014/09/illustration-ssl.html

# Transport Layer Security (TLS) Parameters

```
Cipher Suites (26 suites)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x009f)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x009e)
    Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
    Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 (0xc024)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 (0xc023)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
    Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA256 (0x003d)
    Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA256 (0x003c)
    Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
    Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
    Cipher Suite: TLS_DHE_DSS_WITH_AES_256_CBC_SHA256 (0x006a)
    Cipher Suite: TLS_DHE_DSS_WITH_AES_128_CBC_SHA256 (0x0040)
    Cipher Suite: TLS_DHE_DSS_WITH_AES_256_CBC_SHA (0x0038)
    Cipher Suite: TLS_DHE_DSS_WITH_AES_128_CBC_SHA (0x0032)
    Cipher Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000a)
    Cipher Suite: TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA (0x0013)
    Cipher Suite: TLS_RSA_WITH_RC4_128_SHA (0x0005)
    Cipher Suite: TLS_RSA_WITH_RC4_128_MD5 (0x0004)
```



- SSL (Secure Sockets Layer) stander for the “s” in Https.
- SSL has 1.0, 2.0 and 3.0 version.
- A new name TLS (Transport Layer Security） replaced SSL.
- Currently TLS1.3 is the latest version and TLS1.2 is still widely used.
- Cipher suite is an important concept, it contains the method to exchange pre-master, the digital sign algorithm of server, the symmetric encryption algorithm and block cipher modes.
- For example, TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 means using DHE to exchange the pre-master, the digital sign is RSA algorithm, AES_256 is the symmetric encryption algorithm and the block cipher mode is GCM.
- ECDH means using EC (math) to do the DH algorithm.
- ECDHE means the session key need to be replaced after certain packages.

https://www.jianshu.com/p/580e92dc4fdd
https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml
https://blog.csdn.net/herongoal/article/details/83414888

# MAC - Message Authentication Code

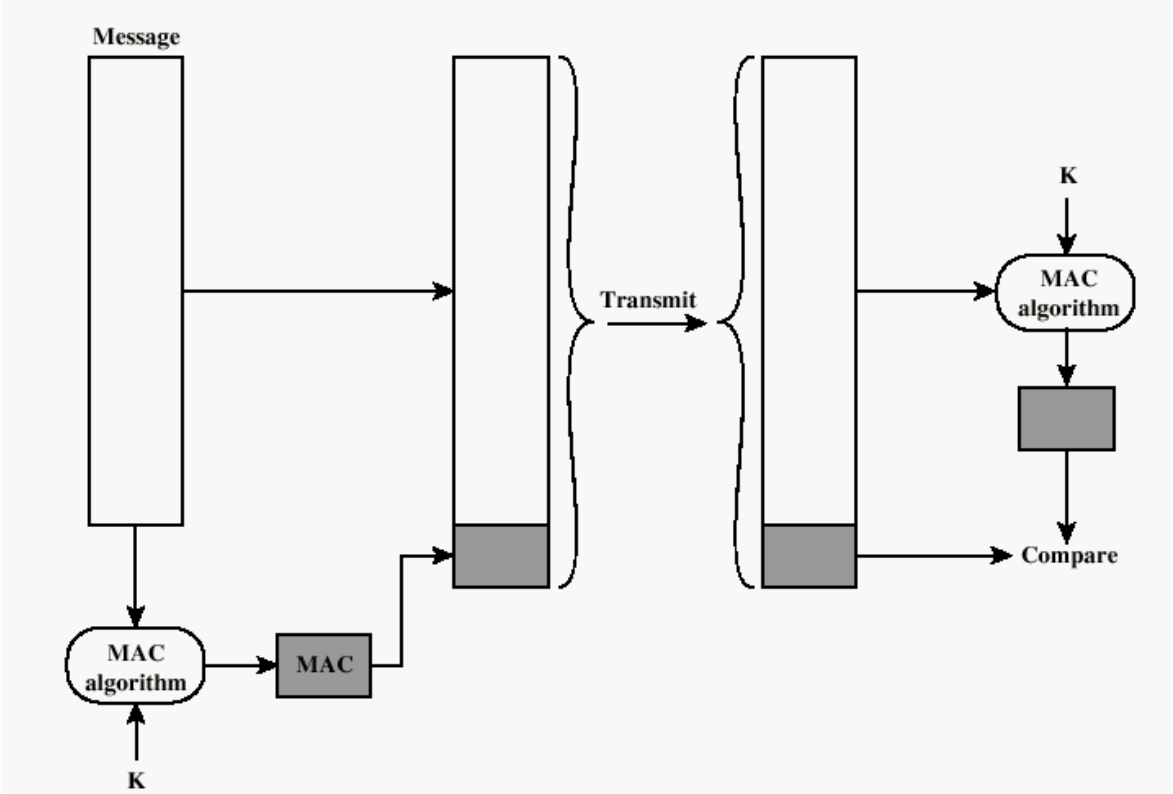
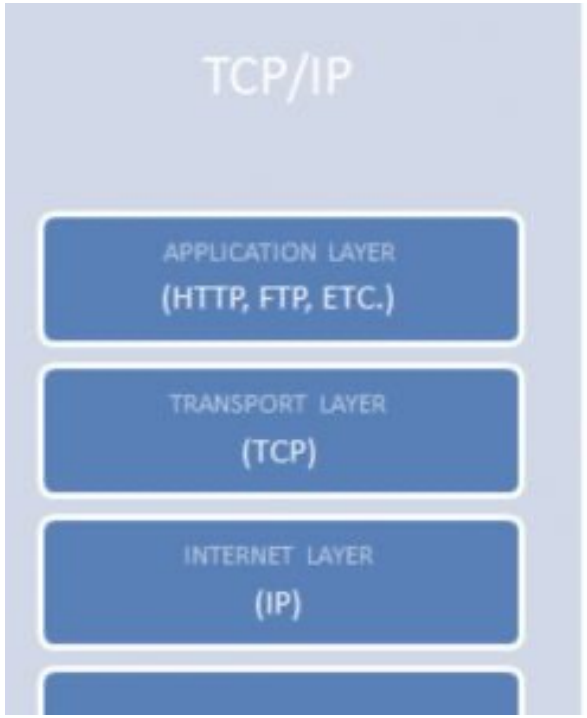
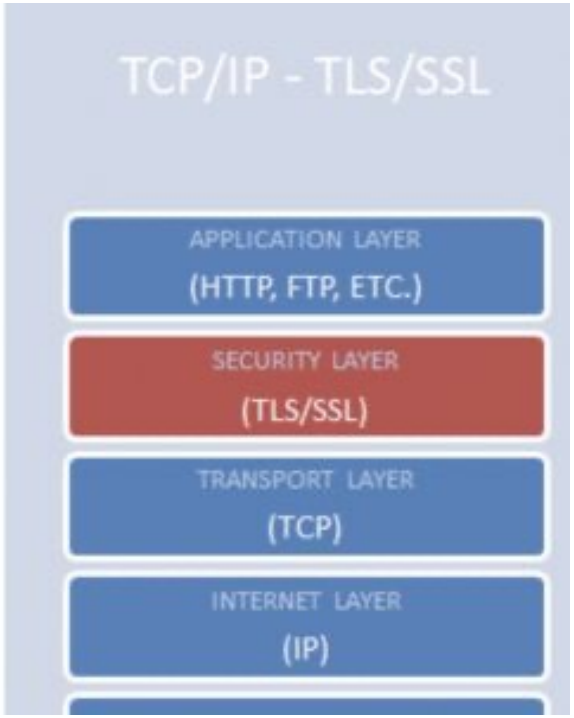


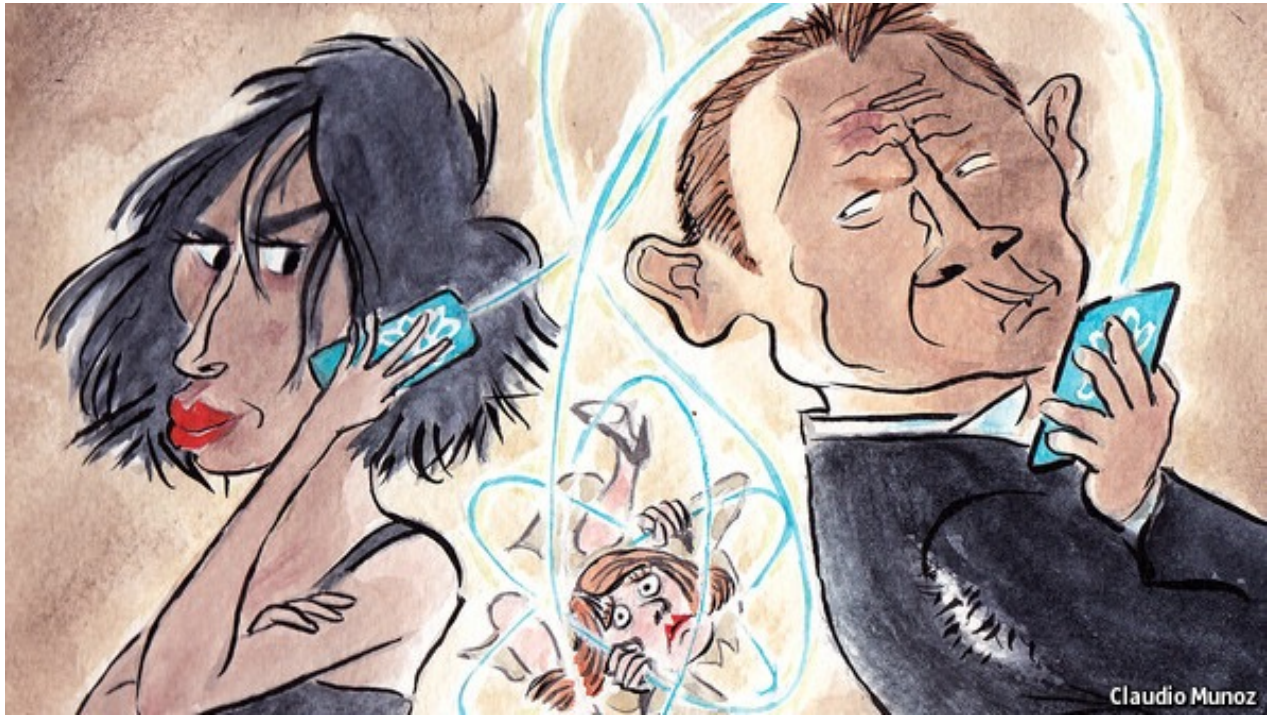

Figure 3.1 Message Authentication Using a Message Authentication Code (MAC)

- In TLS, MAC algorithm of symmetric encryption is also a part of cipher suite.
- MAC means how to check the integrate of a message, normally a HASH function.

# Summary of SSL/TLS

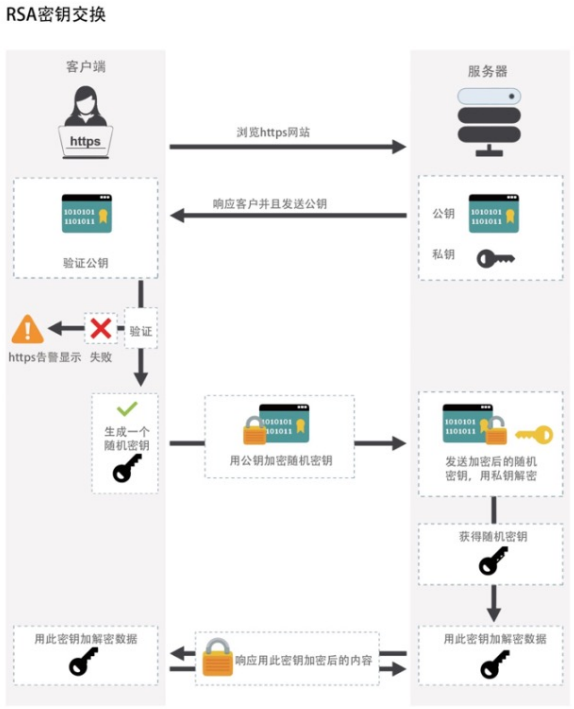


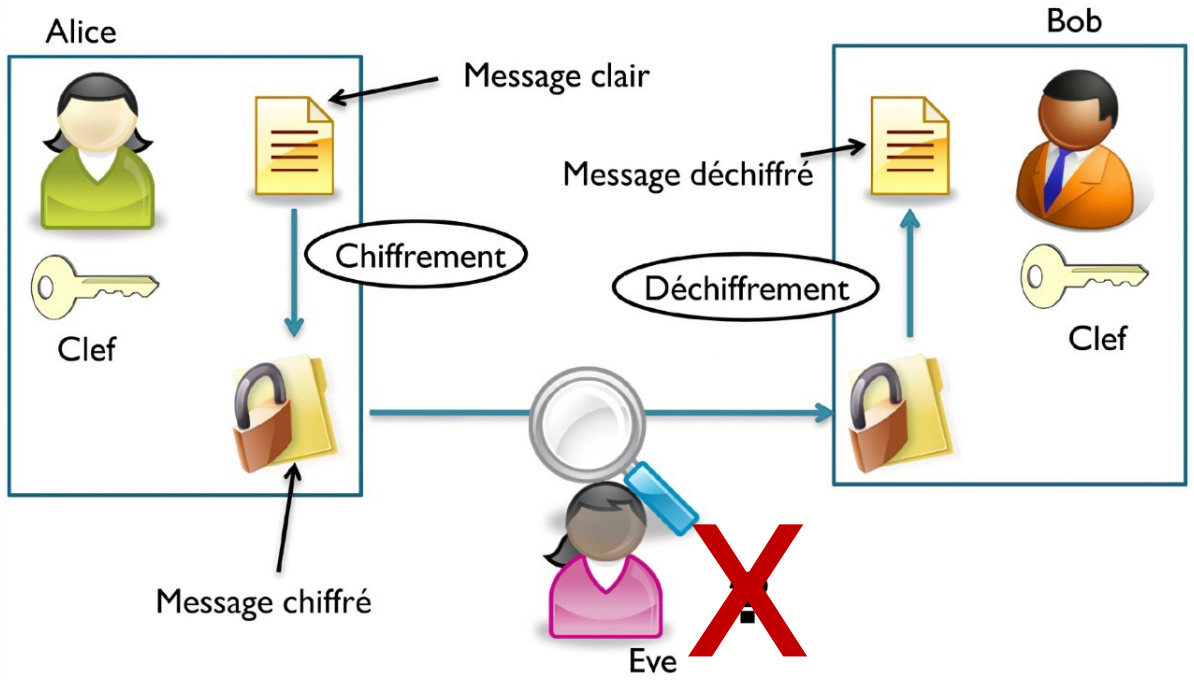




- TLS is not limited to Http.
- TLS can also apply for FTP, SMTP …
- TLS cannot work if only use symmetric algorithms or otherwise, it combines both.

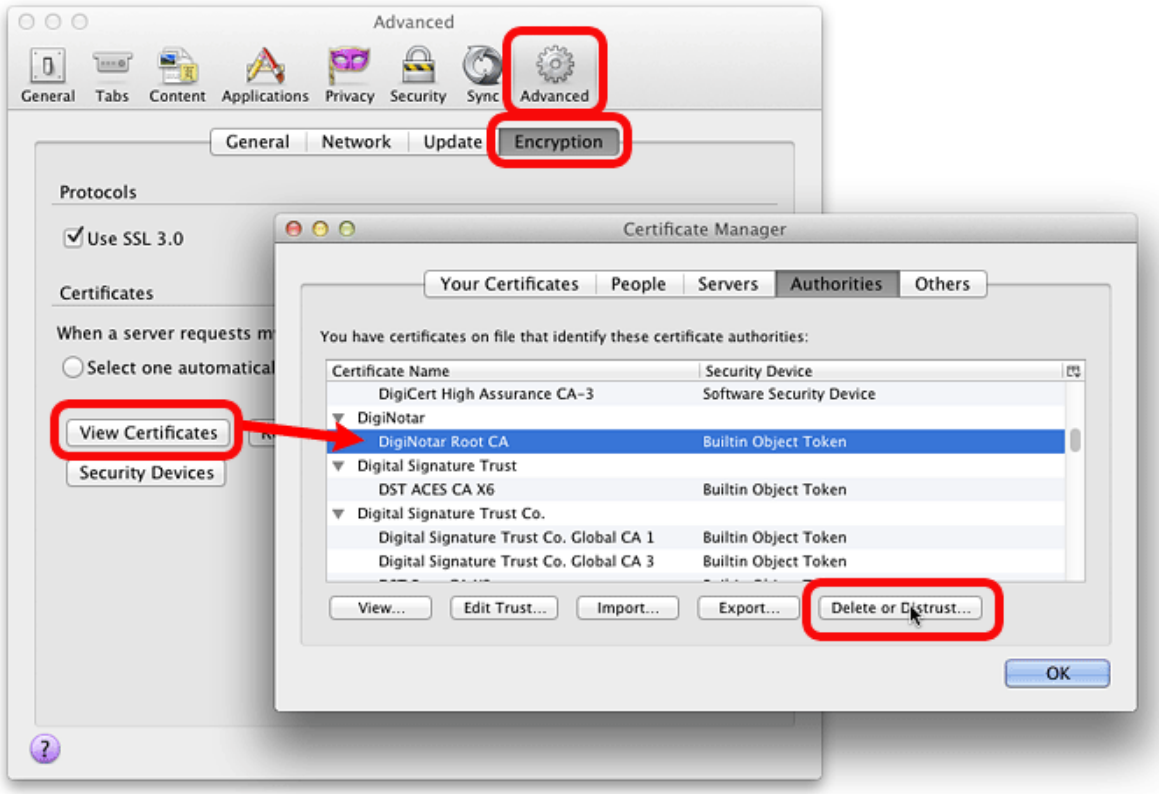
# It seems Eve is out!




- Remember, the biggest risk is "We think we are safe!"

# Be attention to this

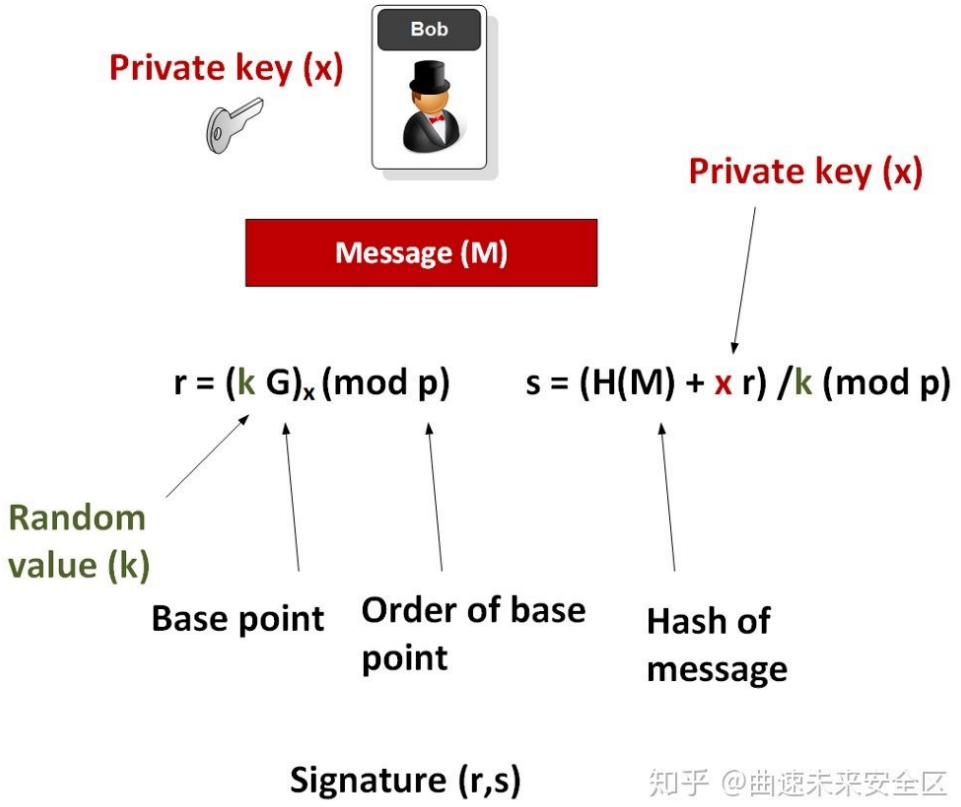

- CA DigiNotar has been hacked and fake “google” was online in 2011

https://www.zdnet.com/article/diginotar-files-for-bankruptcy-following-hack-attack/

# How about, the private key is disclosive!

- It is time to decrypt all previous securities!

# Tools cannot guarantee safe, but  with people


- **Again**, But more important thing is about people and related regulations.
- Sony PS3 use a fixed “random” number, the door opens for everyone.

https://zhuanlan.zhihu.com/p/49568984

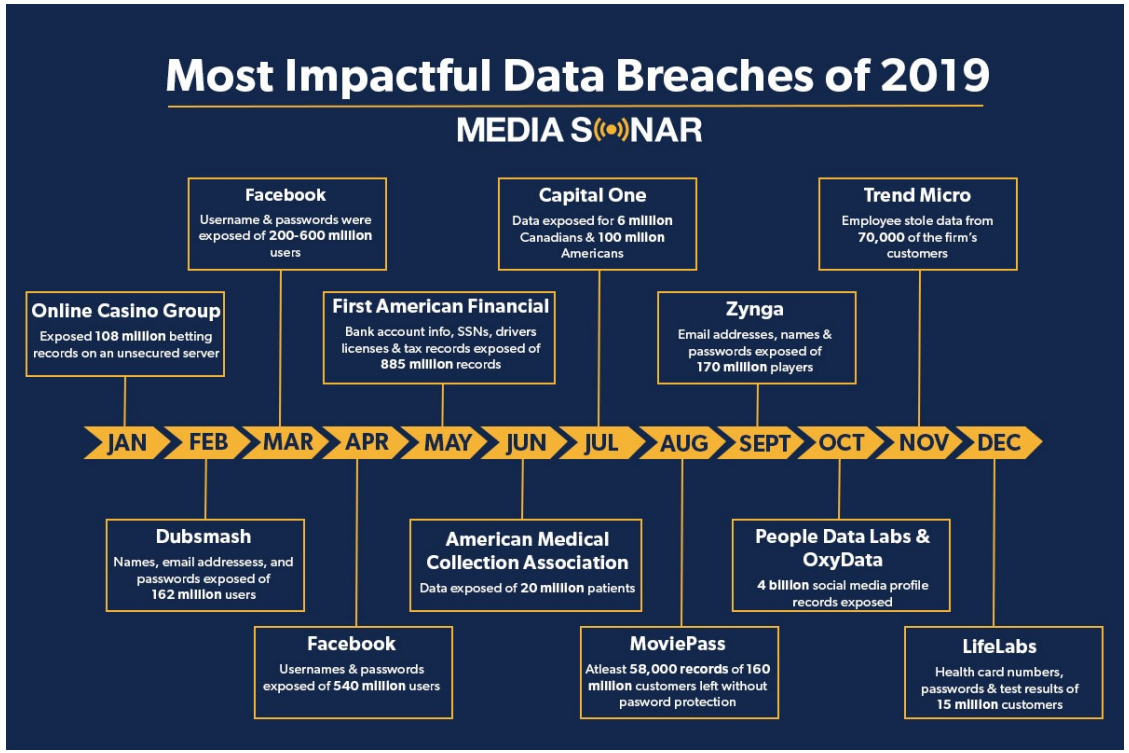
# Password or account credential

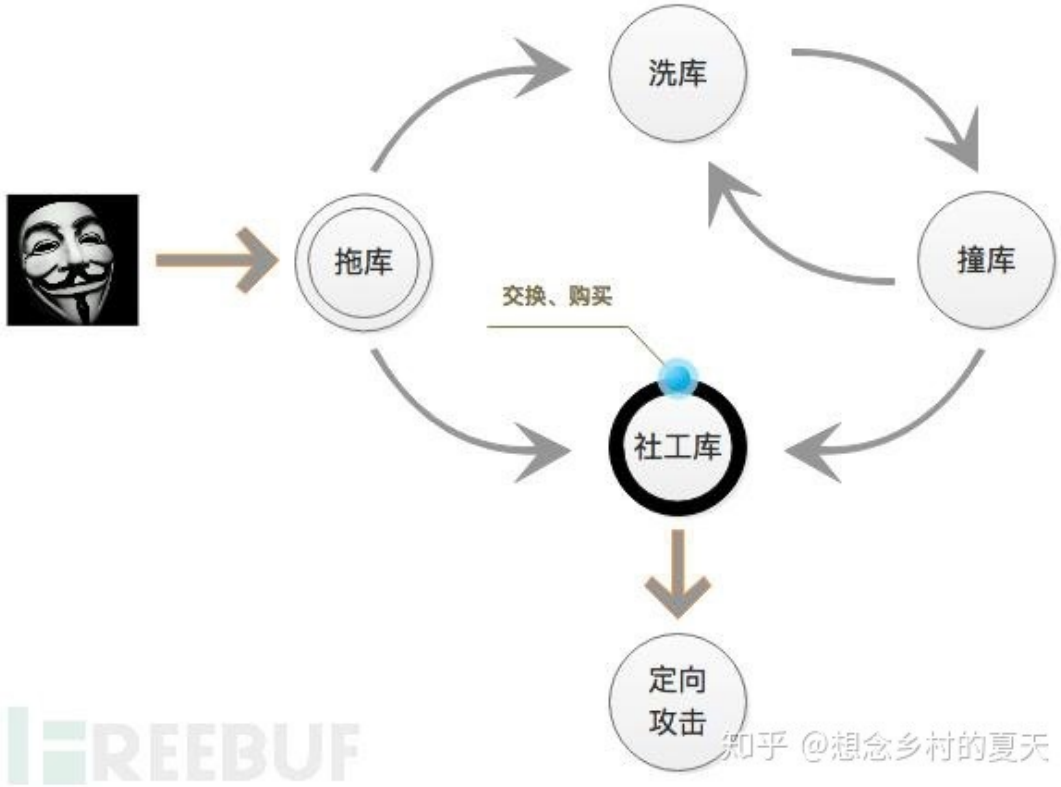

- **Password:** something that enables one to pass or gain admission
- **Credential:** testimonials or certified documents showing that a person is entitled to credit or has a right to exercise official power

**Username + password is the most widely used authority credential on internet**

https://www.merriam-webster.com/dictionary/password

# Attacks never sleep

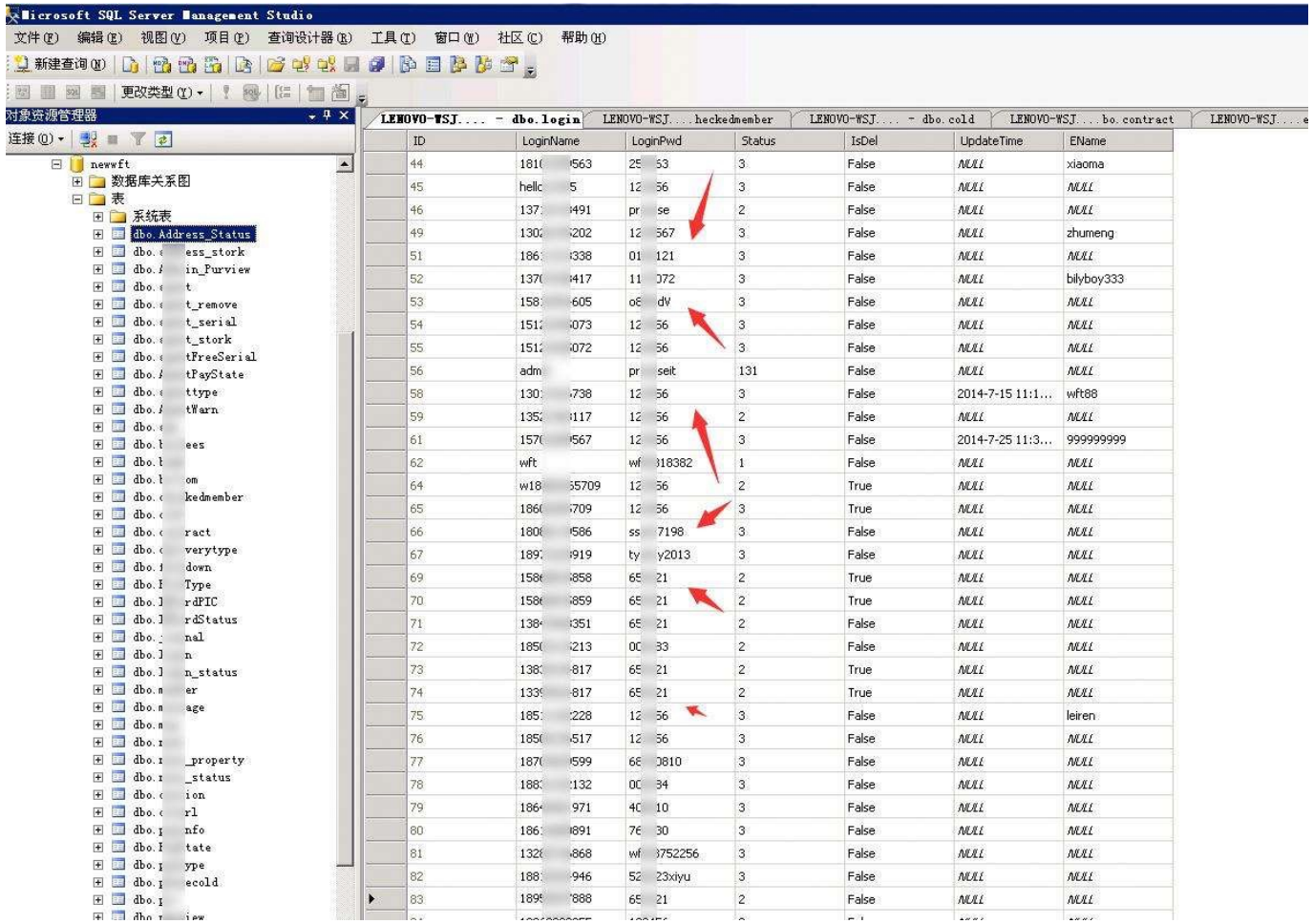



- **Harvard Business Review: *"You Can't Secure 100% of Your Data 100% of the Time"***
- **But we need to do something to make it hard or worthless.**

https://hbr.org/2017/12/you-cant-secure-100-of-your-data-100-of-the-time

# Never store passwords in plain text




- **Password** is only for comparison.
- On design, even the Admin should NOT has any chance to access customers passwords in plain text.
- **HASH** is a good choice while is is NOT an encryption algorithm.
- For example, you can read 123456 very easy. But you may not know what does “8d969eef6ecad3c29a3a629280e686cf0c3f5d5a86aff3ca12020c923adc6c92” means.
- It is SAH256(“123456”)

http://wy.zone.ci/bug_detail.php?wybug_id=wooyun-2015-095231

# Hash collision attack

55

These following two different 128 byte sequences hash to the same:

**MD5 Hash**: 79054025255fb1a26e4bc422aef54eb4

The differences below are highlighted (bold). Sorry it's kind of hard to see.

```
d131dd02c5e6eec4693d9a0698aff95c 2fcab58712467eab4004583eb8fb7f89
55ad340609f4b30283e488832571415a 085125e8f7cdc99fd91dbdf280373c5b
d8823e3156348f5bae6dacd436c919c6 dd53e2b487da03fd02396306d248cda0
e99f33420f577ee8ce54b67080a80d1e c69821bcb6a8839396f9652b6ff72a70
```

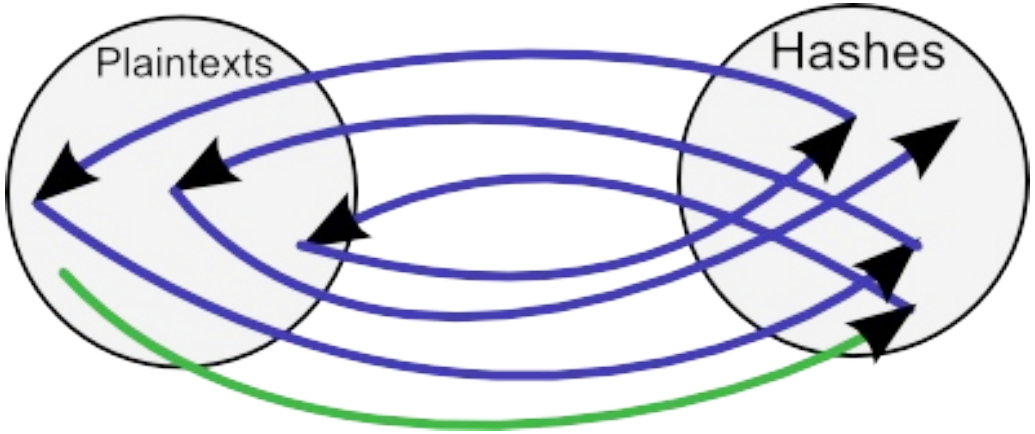
and

```
d131dd02c5e6eec4693d9a0698aff95c 2fcab50712467eab4004583eb8fb7f89
55ad340609f4b30283e4888325f1415a 085125e8f7cdc99fd91dbd7280373c5b
d8823e3156348f5bae6dacd436c919c6 dd53e23487da03fd02396306d248cda0
e99f33420f577ee8ce54b6708028 0d1e c69821bcb6a8839396f965ab6ff72a70
```

- Dr. 王小云 proved collisions can be found in MD5, SHA-1 algorithms
- She lead the design of SM-3

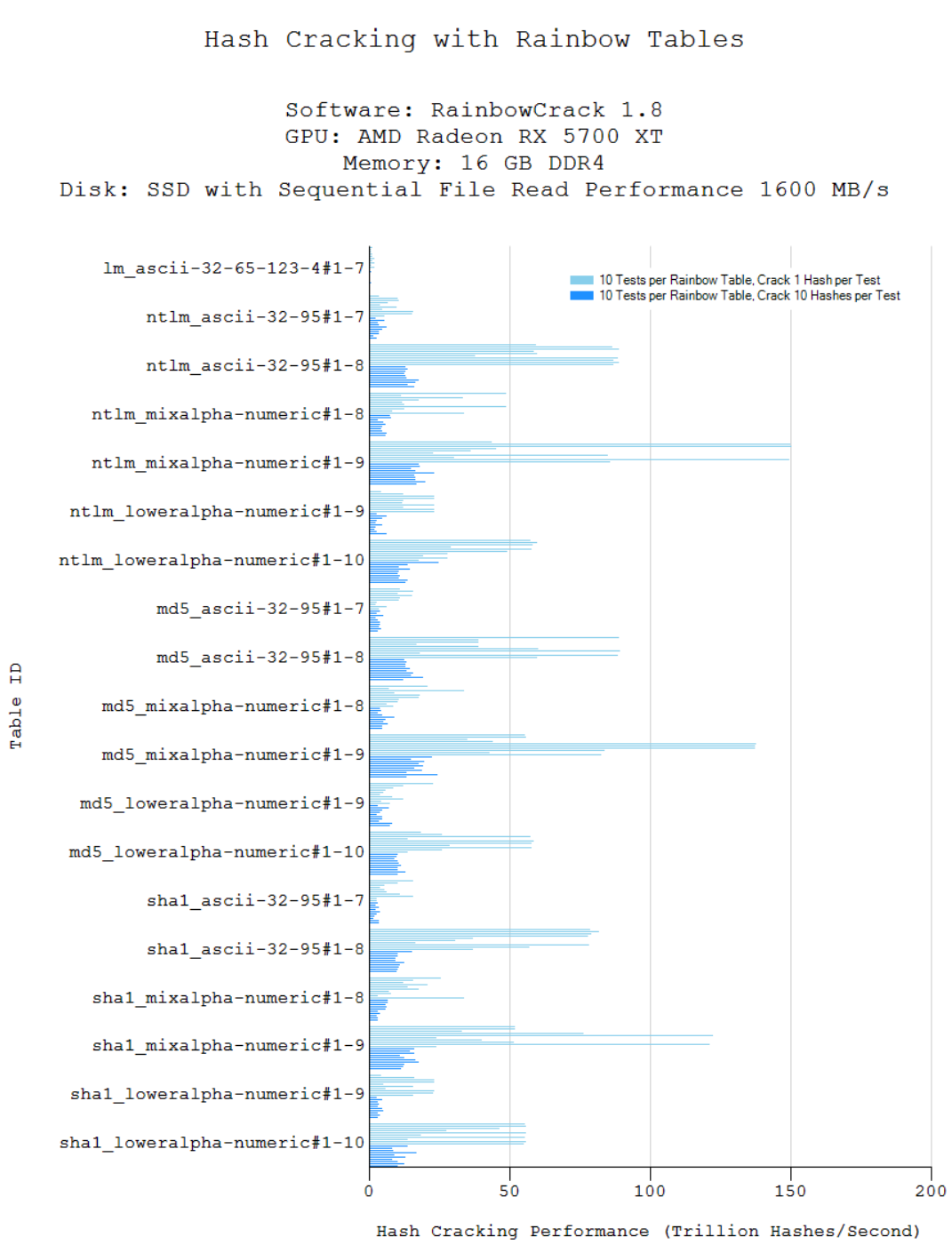
https://baike.baidu.com/item/%E7%8E%8B%E5%B0%8F%E4%BA%91/29050
https://zhuanlan.zhihu.com/p/377468857

# Hash only is NOT safe




- Rainbow table help to speed up the hash finding.
- Add "**salt**" can improve the difficulty.

https://kestas.kuliukas.com/RainbowTables/
https://www.ionos.com/digitalguide/server/security/rainbow-table
https://www.jianshu.com/p/732d9d960411
http://project-rainbowcrack.com/table.htm



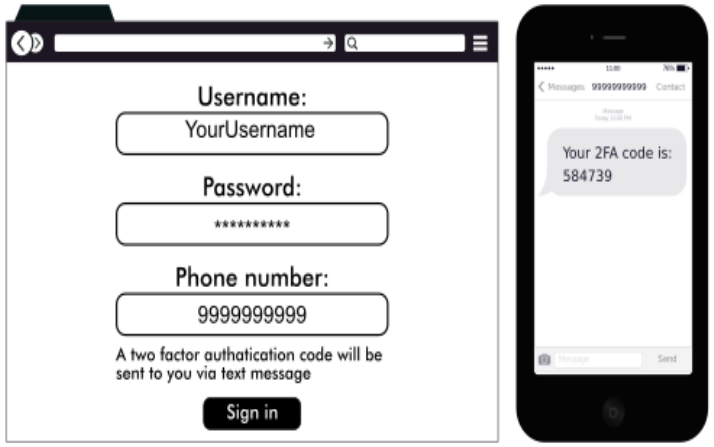
# Salt can NOT against database searching

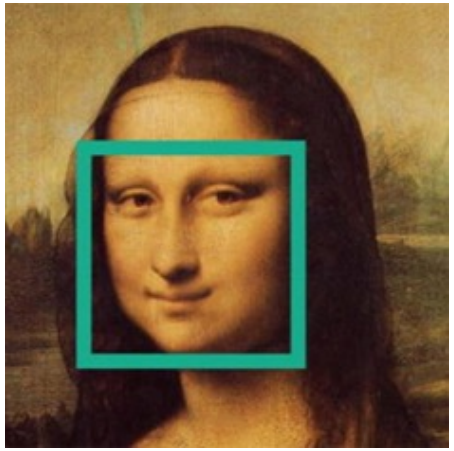


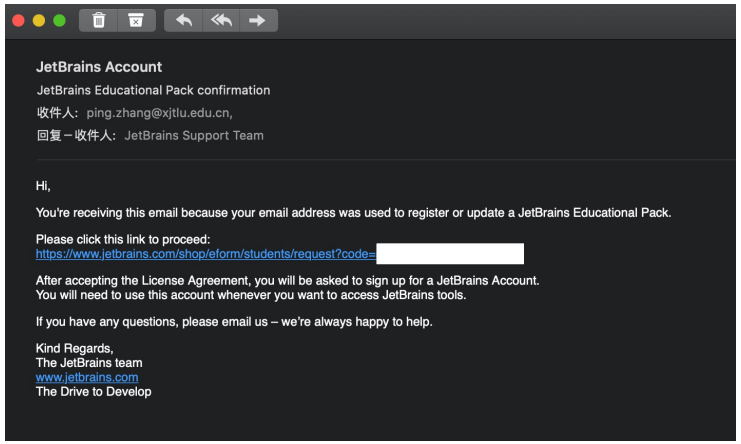
- According to public report, over 1.4 B passwords breach.
- Let customers check if they are in the list.

https://www.sohu.com/a/210357960_804262

# MFA: Multi-Factor Authentication



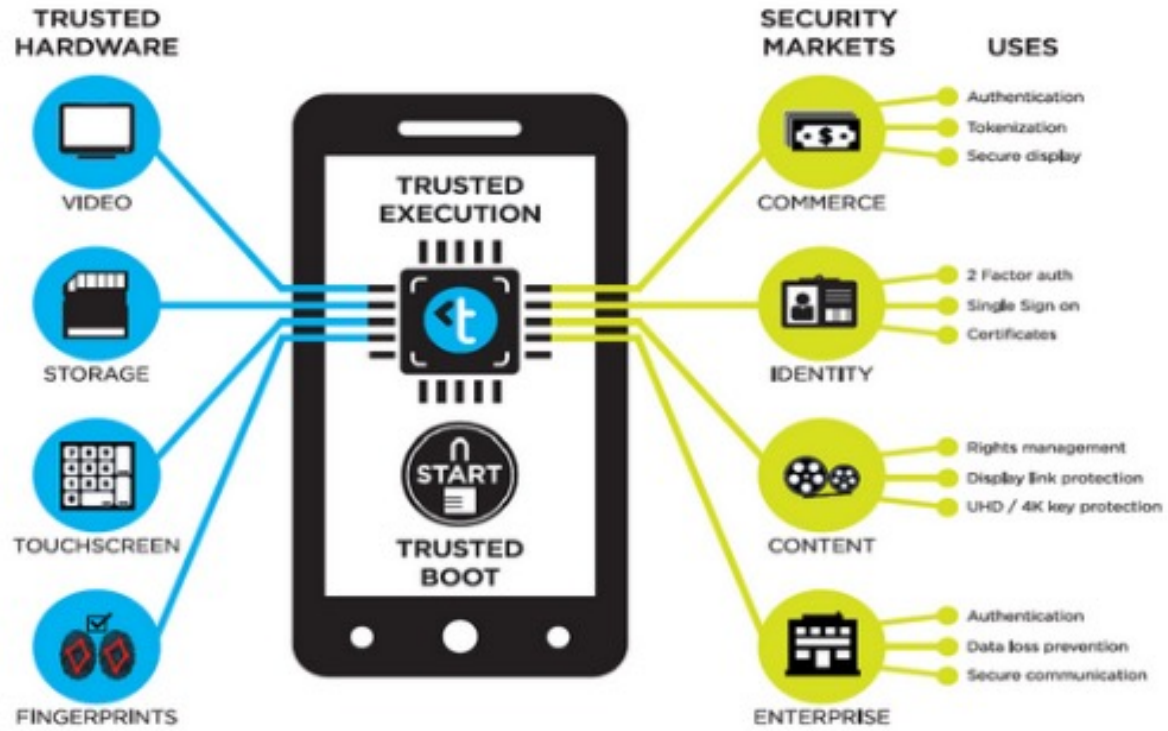




**MFA** is a security system that requires more than one method of authentication to verify user's identity for a login or other transaction.
Normally, 2 factors are used. **2FA** (Two-factor authentication) is a subset of MFA.

https://authonet.com/authentication.php

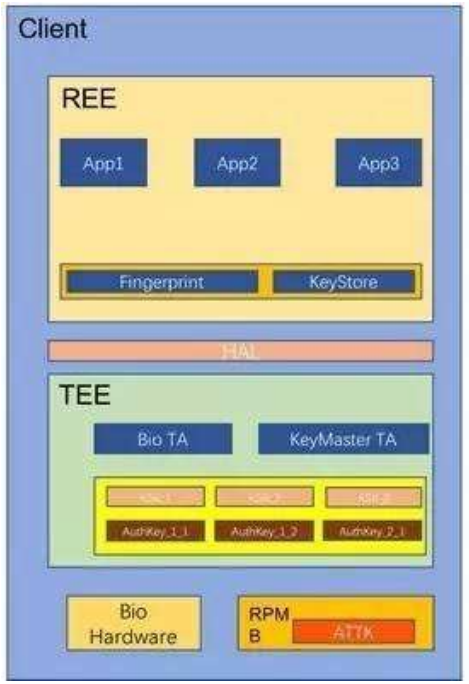
# And these

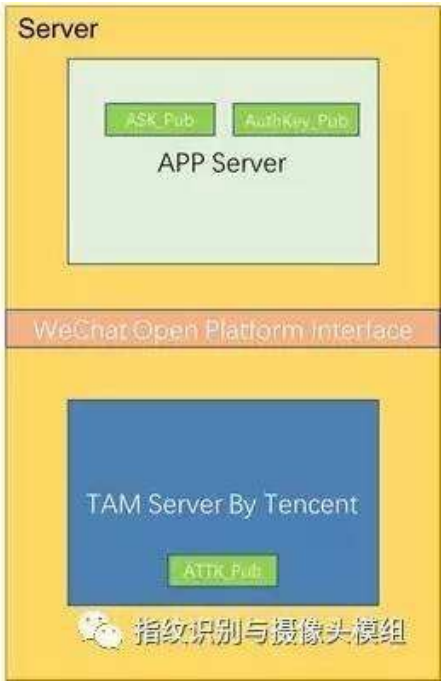




- Public Key Infrastructure（PKI) can also used at client side.
- Normally the private keys are protected by chips.
- CA play an important role in USB stick for PC application

https://zhuanlan.zhihu.com/p/24222064

# Tencent ‘soter’



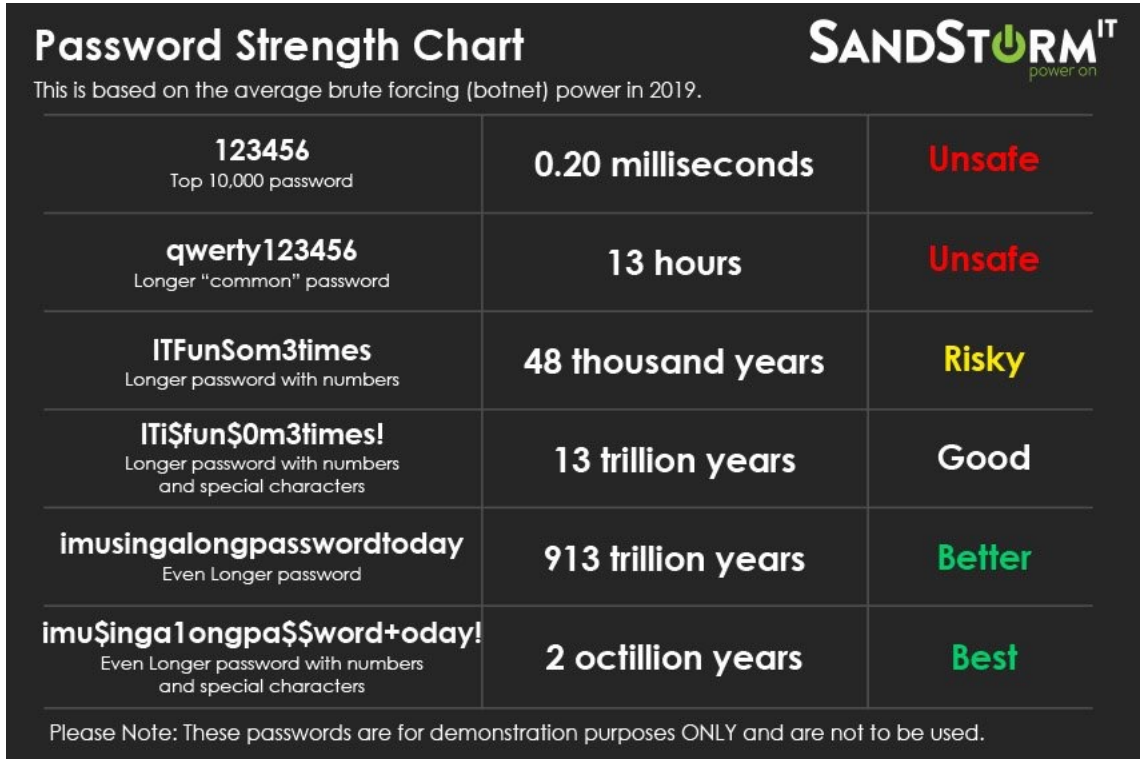

| 密钥名称 | 唯一性 | 作用描述 | 生成算法 | 公钥位置 | 私钥位置 |
|---|---|---|---|---|---|
| ATTK | 设备唯一 | 验证设备合法性 | RSA-2048 | TAM Server | RPMB |
| ASK | 应用唯一 | 验证应用合法性 | RSA-2048 | 第三方应用程序 Server | File system as other keys（加密存储、权限控制） |
| Auth Key | 不限数量，根据业务需要生成 | 验证业务逻辑唯一性 | RSA-2048 | 第三方应用程序 Server | File system as other keys（加密存储、权限控制） |



- It is a correct BUT difficult way to go.
- get all mobile vendors’ support to send root certificate to Tencent.
- Tencent TAM server play like the root CA server.

https://github.com/Tencent/soter/wiki

# Some suggestions on Password/Credential



**Password Strength Chart**

SANDSTORM IT power on

This is based on the average brute forcing (botnet) power in 2019.

| | | |
|---|---|---|
| **123456**<br>Top 10,000 password | **0.20 milliseconds** | **Unsafe** |
| **qwerty123456**<br>Longer "common" password | **13 hours** | **Unsafe** |
| **ITFunSom3times**<br>Longer password with numbers | **48 thousand years** | **Risky** |
| **ITi$fun$0m3times!**<br>Longer password with numbers and special characters | **13 trillion years** | **Good** |
| **imusingalongpasswordtoday**<br>Even Longer password | **913 trillion years** | **Better** |
| **imu$inga1ongpa$$word+oday!**<br>Even Longer password with numbers and special characters | **2 octillion years** | **Best** |

Please Note: These passwords are for demonstration purposes ONLY and are not to be used.

- Weak password is the NO.1 factor.
- Do not install "dangers" APP/software, like 180 * 2.
- If you have to, use the "dangers" software in VM.
- Do not use the same password for different website/APPs, at least add some "salt".
- Do not register account in "*unsafe*" websites, for example, those do not use 2FA.
- Do not share password.