Key Management

- Key management is the set of techniques and procedures supporting the establishment and maintenance of keying relationships between authorized parties.

- Key Management deals with the creation, exchange, storage, deletion, and refreshing of keys.

Key Management Lifecycle

Key Management Lifecycle

- **Key generation:** generating a cryptography key using an approved set of rules, including the use of a pseudo-random generator.

- **Key installation:** the process of setting up, configuring and testing keying material, including hardware, software and cryptomodules.

- **Key establishment:** the distribution of keys between two or more entities involved in the communication.

Key Management Lifecycle

- **Key certification:** the key must be certified — an authentication using digital signatures (issued by third party certification authority) that unambiguously associate the key with the appropriate sources.

- **Key usage:** the process of ensuring operational availability of keying material during the applicable cryptoperiod of the keys.

- **Key storage:** the keys must be stored with a high degree of Confidentiality, Integrity and Availability (CIA).

Key Management Lifecycle

- **Key update & recovery:** mechanisms that allow authorized entities to update and retrieve the keys stored in the operational memory.

- **Key revocation:** the key is destroyed or deregistered when no further key management operations are applicable to the associated source entities.

Key Distribution

- Key distribution in cryptography is a system that is responsible for providing keys to the users in a network that shares sensitive or private data.

- Public Key Distribution
- Private Key Distribution

Public Key Distribution

1. Public Announcement
2. Publicly Available Directory
3. Public Key Authority
4. Public Key Certificate

Public Announcement

- Users distribute public keys to recipients or broadcast to community at large.
- Example: append keys to email messages or post to news groups or email list.

Public Announcement

- Major weakness is forgery.
- Anyone can create a key claiming to be someone else and broadcast it
- Until forgery is discovered can masquerade as claimed user for authentication

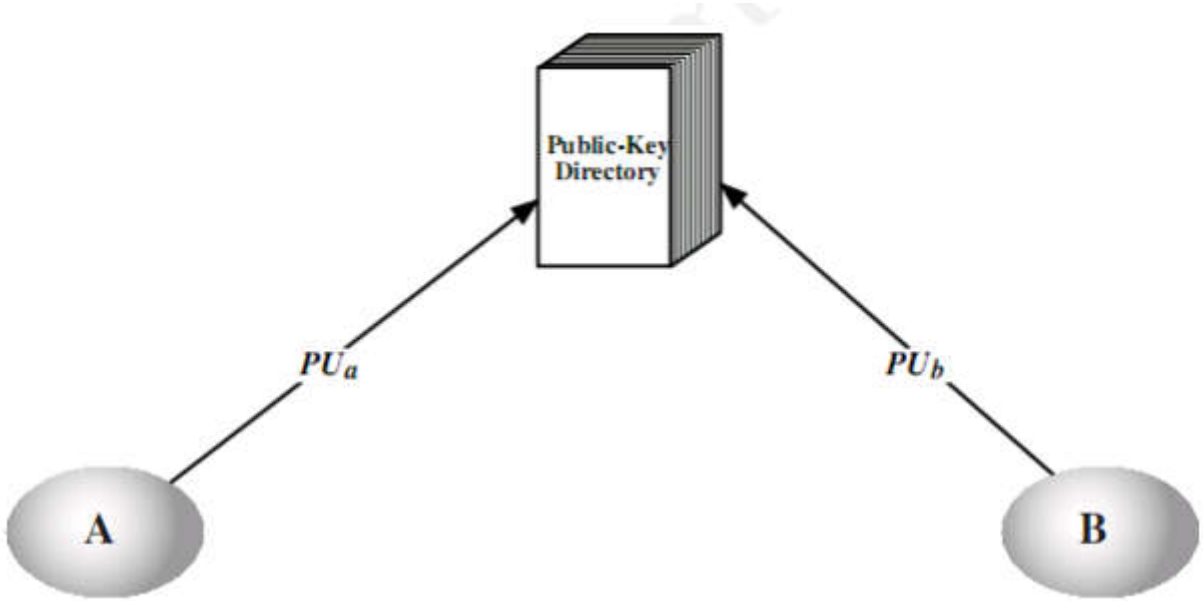
Publicly Available Directory

- We can obtain greater security by registering keys with a public directory

- directory must be trusted with properties:
  - contains {name, public-key} entries
  - participants register securely with directory
  - participants can replace key at any time
  - directory is periodically published
  - directory can be accessed electronically

Publicly Available Directory

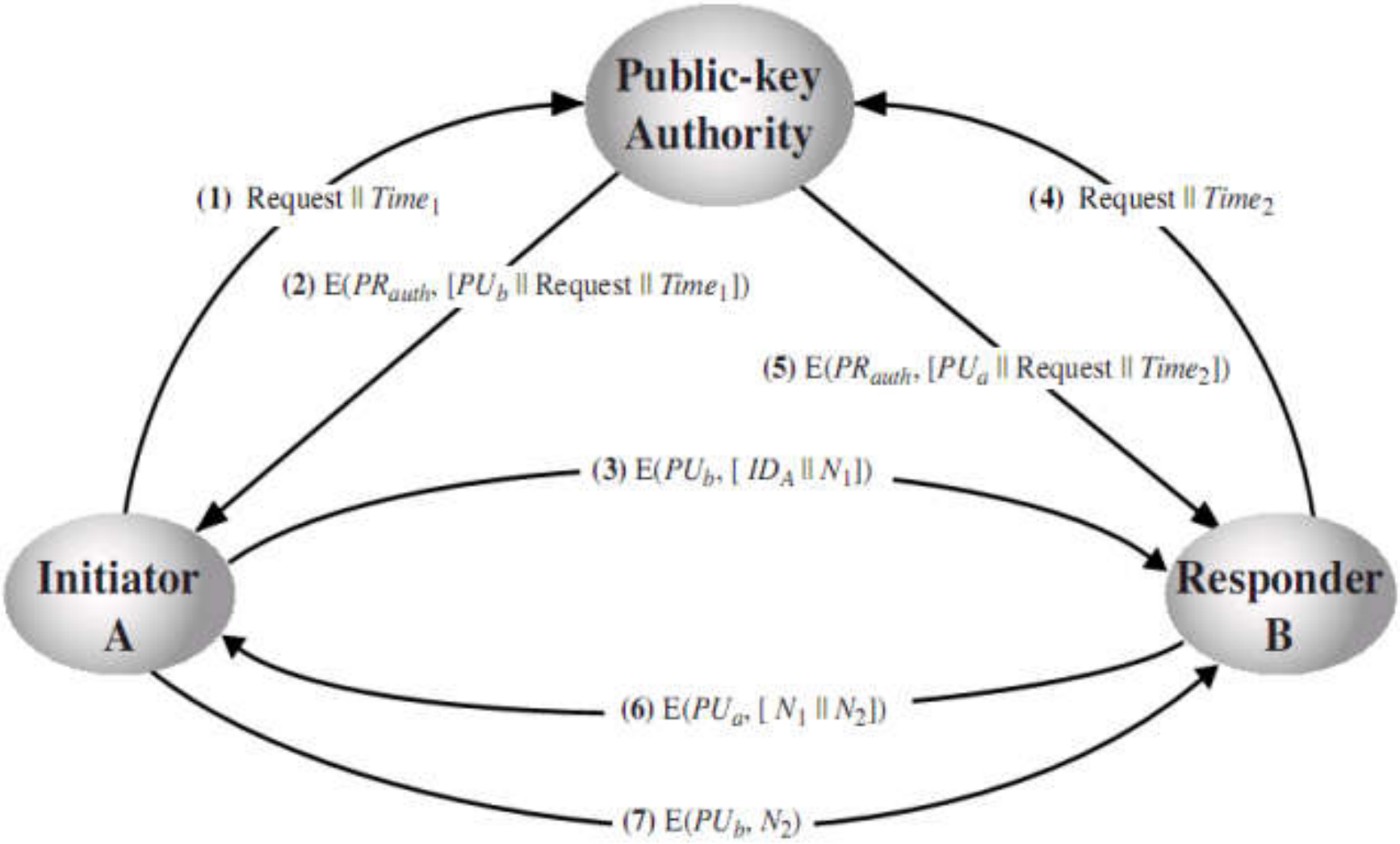
- We can obtain greater security by registering keys with a public directory



Public Key Authority

- Stronger security for public-key distribution can be achieved by providing tighter control over the distribution of public keys from the directory.
- It requires users to know the public key for the directory, and that they interact with directory in real-time to obtain any desired public key securely.
- Totally seven messages are required.

Public Key Authority



Public Key Authority

1. A sends a timestamped message to the public-key authority containing a request for the current public key of B.

2. The authority responds with a message that is encrypted using the authority's private key, PR_{auth} Thus, A is able to decrypt the message using the authority's public key. Therefore, A is assured that the message originated with the authority. The message includes the following:

    - B's public key, PU_b which A can use to encrypt messages destined for B
    - The original request, to enable A to match this response with the corresponding earlier request and to verify that the original request was not altered before reception by the authority.
    - The original timestamp, so A can determine that this is not an old message from the authority containing a key other than B's current public key.

Public Key Authority

3. A stores B's public key and also uses it to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1), which is used to identify this transaction uniquely.

4. B retrieves A's public key from the authority in the same manner as A retrieved B's public key.
5. At this point, public keys have been securely delivered to A and B, and they may begin their protected exchange. However, two additional steps are desirable:

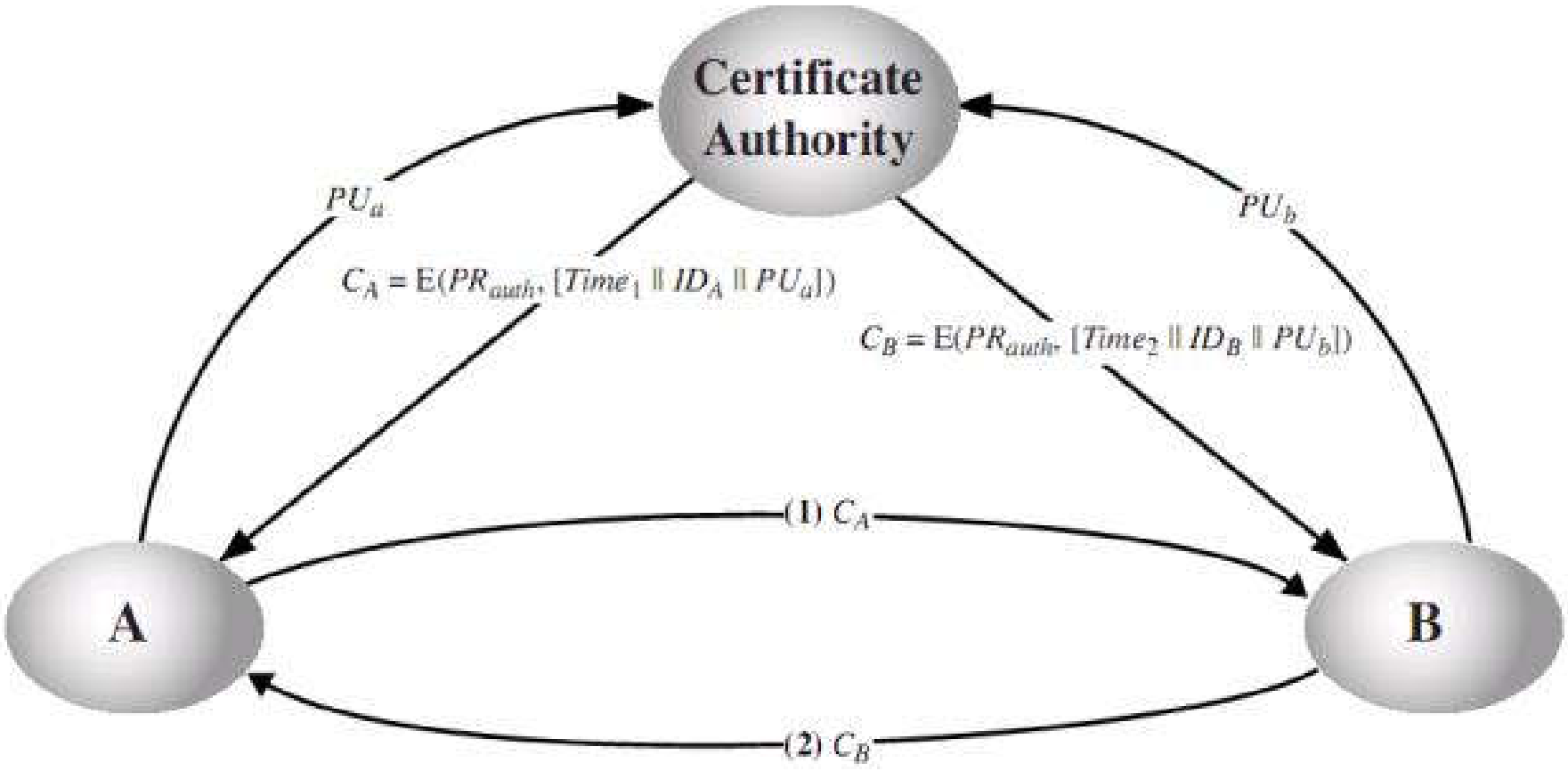
6. B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B (N_2) Because only B could have decrypted message (3), the presence of N_1 in message (6) assures A that the correspondent is B.

7. A returns N_2, encrypted using B's public key, to assure B that its correspondent is A.

Public Key Certificate

- The public-key authority could be a bottleneck in the system.
  - must appeal to the authority for the key of every other user
- certificates allow key exchange without real-time access to public-key authority
- a certificate binds **identity** to **public key**
- with all contents **signed** by a trusted Public-Key or Certificate Authority (CA)
  - Certifies the identity
  - Only the CA can make the certificates

Public Key Certificate



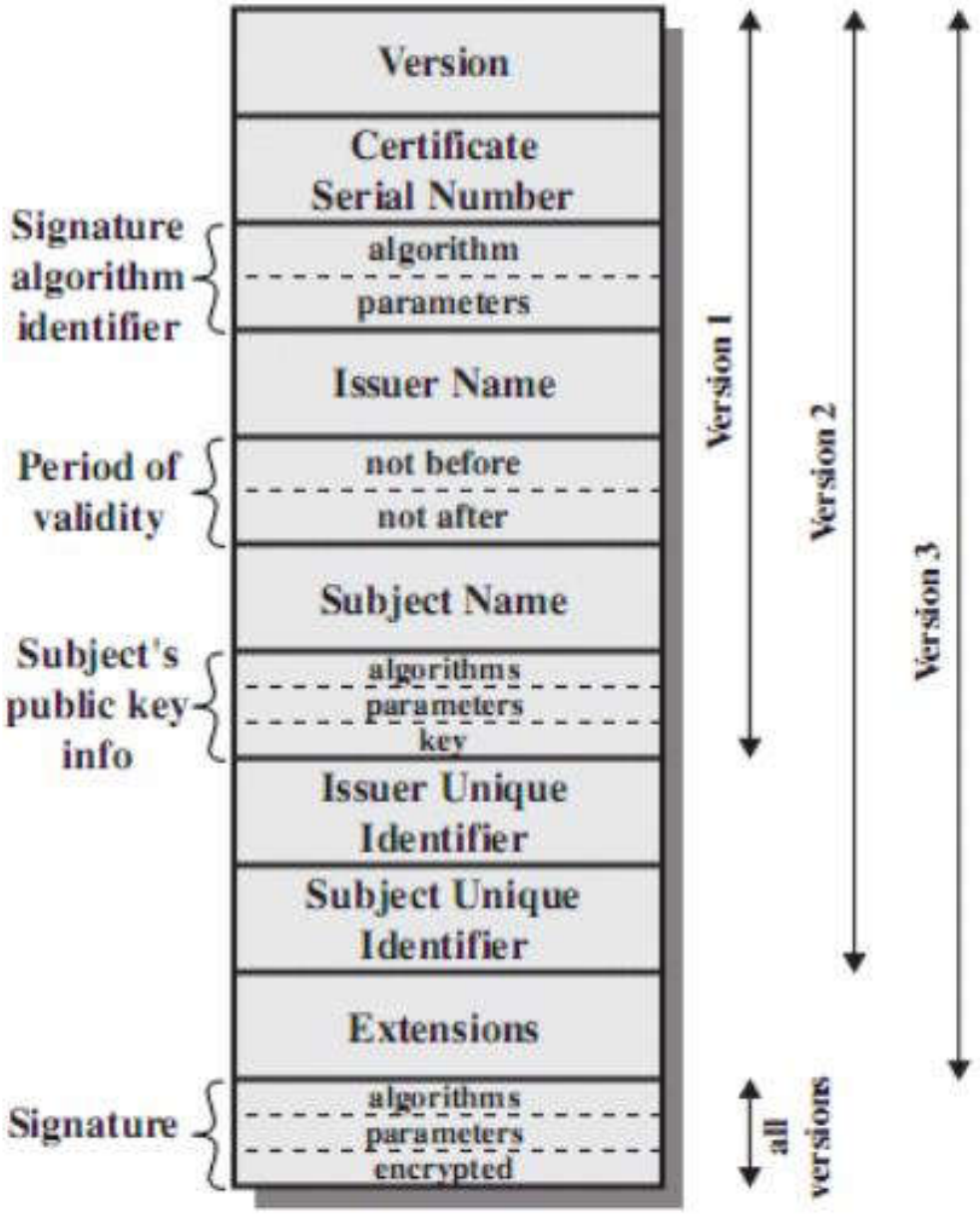
X.509 Certificate

- In cryptography, X.509 is an International Telecommunication Union (ITU) standard defining the format of public key certificates.

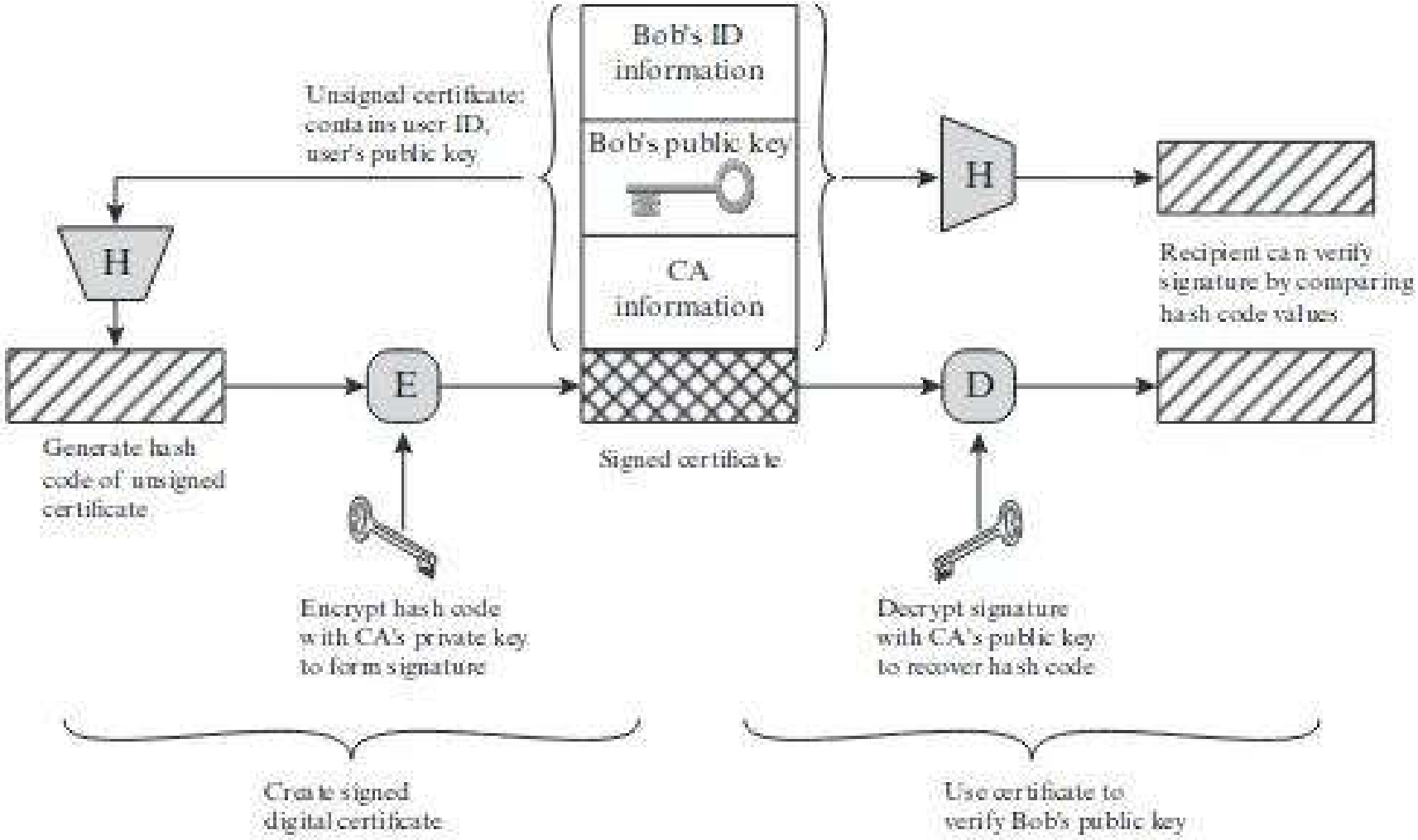
- An X.509 certificate binds an identity to a public key using a digital signature.

- A certificate contains an identity (a hostname, or an organization, or an individual) and a public key (RSA, DSA) and is either signed by a certificate authority or is self-signed.

X.509 Certificate



X.509 Certificate



Diffie–Hellman Key Exchange

- Diffie–Hellman Key Exchange (DHKE) is a cryptographic method to securely exchange cryptographic keys over a public (insecure) channel in a way that overheard communication does not reveal the keys.

- DHKE was one of the first public-key protocols, which allows two parties to exchange data securely.

- The Diffie–Hellman (DH) method is anonymous key agreement scheme: it allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel.

Key Exchange by Mixing Colors

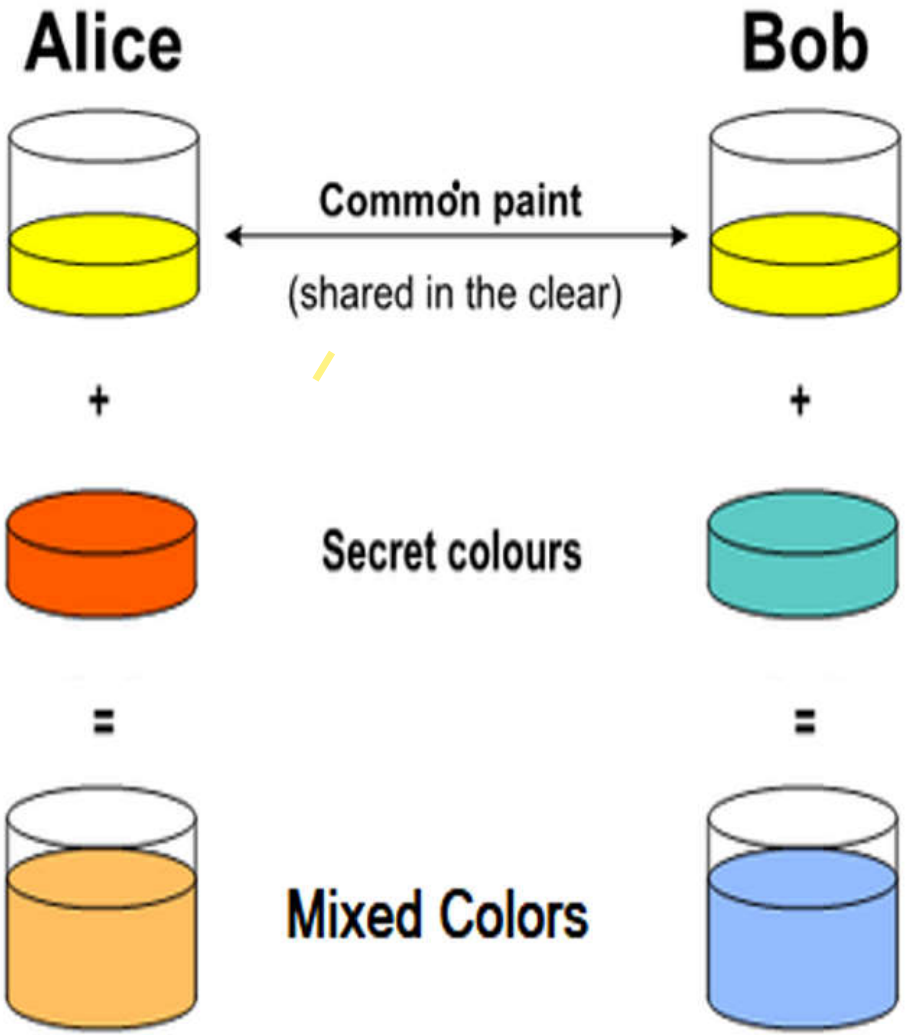
- The Diffie–Hellman Key Exchange protocol is very similar to the concept of "key exchanging by mixing colors", which has a good visual representation, which simplifies its understanding.

Key Exchange by Mixing Colors

This is the color exchange **scenario**, step by step:

- **Alice** and **Bob**, agree on an arbitrary **starting (shared) color** that does not need to be kept secret (e.g. *yellow*).
- **Alice** and **Bob** separately select a **secret color** that they keep to themselves (e.g. *red* and *sea green*).
- Finally **Alice** and **Bob mix** their secret color together with their mutually shared color. The obtained mixed colors area ready for public exchange (in our case *orange* and *light sky blue*).

Key Exchange by Mixing Colors

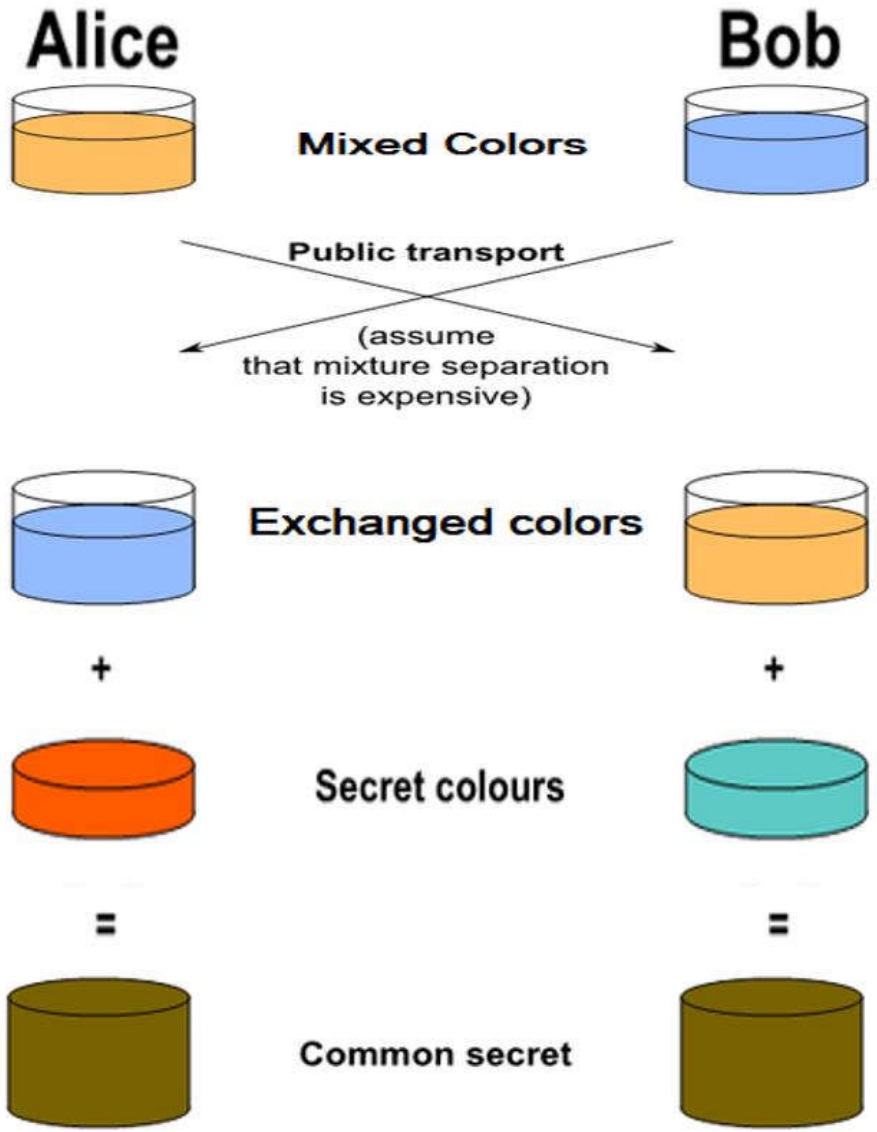


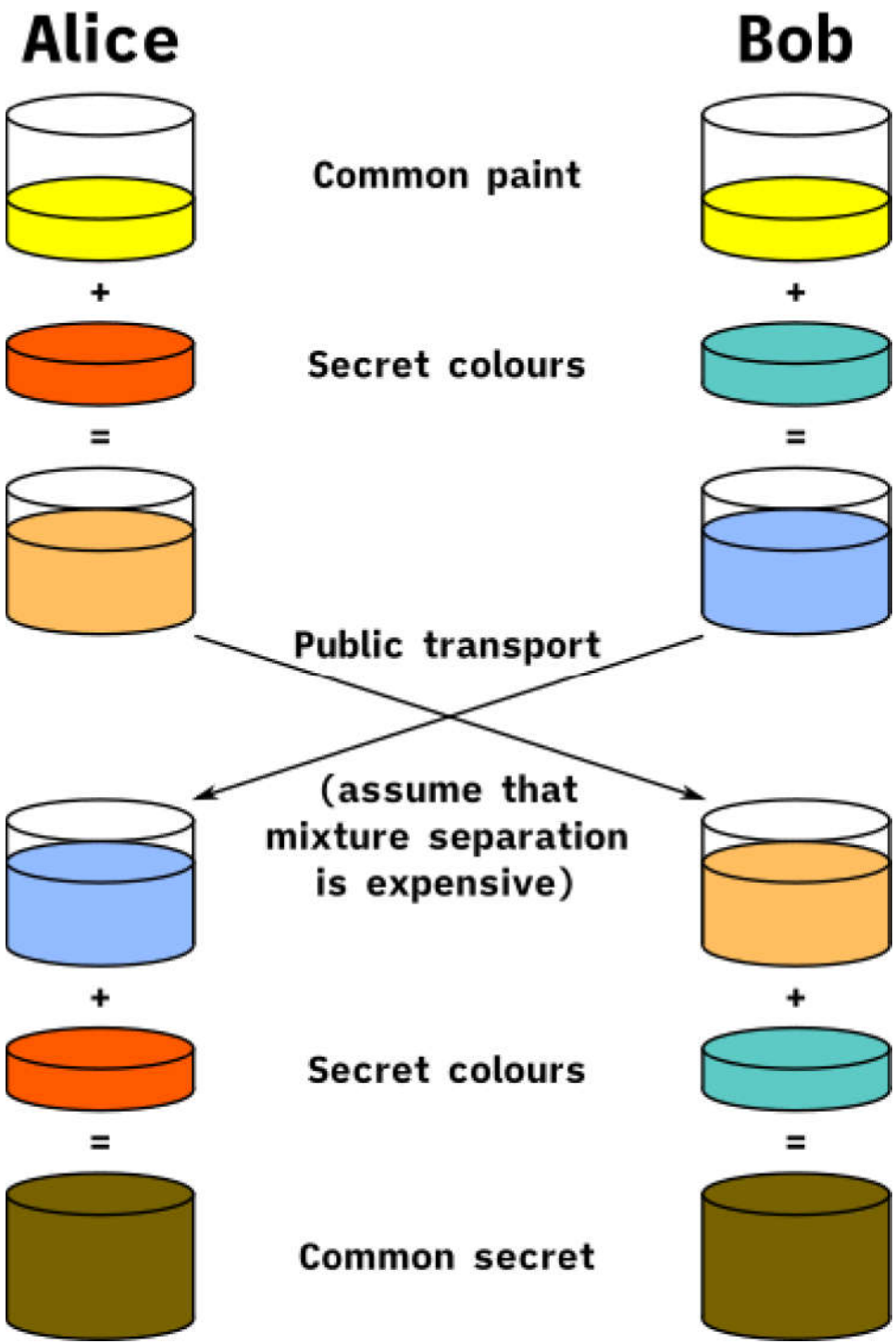
Key Exchange by Mixing Colors

The next steps in the color exchanging scenario are as follows:

- **Alice** and **Bob** publicly **exchange** their two **mixed colors**.
  - We assume that there is no efficient way to extract (separate) the secret color from the mixed color, so third parties who know the mixed colors cannot reveal the secret colors.
- Finally, **Alice** and **Bob** mix together the color they received from the partner with their own secret color.
  - The result is the **final color mixture** (*yellow-brown*) which is identical to the partner's color mixture.
  - It is the **securely exchanged shared key**.

Key Exchange by Mixing Colors




Alice
Bob
Common paint
+
+
Secret colours
=
=
Public transport
(assume that
mixture separation
is expensive)
+
+
Secret colours
=
=
Common secret

Diffie–Hellman Key Exchange

- The Diffie-Hellman Key Exchange protocol is based on similar concept, but uses discrete logarithms and modular exponentiations instead of color mixing.

- Diffie–Hellman key exchange establishes a shared secret between two parties that can be used for secret communication for exchanging data over a public network.

Diffie–Hellman Key Exchange

- first public-key type scheme proposed
  - For key distribution only
- by Diffie & Hellman in 1976 along with the exposition of public key concepts
  - note: now know that James Ellis (UK CESG) secretly proposed the concept in 1970
- is a practical method for public exchange of a secret key
- used in a number of commercial products

Diffie–Hellman Key Exchange

- a public-key distribution scheme
  - cannot be used to exchange an arbitrary message
  - rather it can establish a common key
  - known only to the two participants
- value of key depends on the participants (and their private and public key information)
- based on exponentiation in a finite (Galois) field (modulo a prime or a polynomial) - easy
- security relies on the difficulty of computing discrete logarithms (similar to factoring) – hard

Primitive root modulo n

$$\mathbb{Z}_n^* = \{a \in \mathbb{N} : 1 \leq a < n, \gcd(a, n) = 1\}.$$

$\mathbb{Z}_n^*$ has $\phi(n)$ elements, where ϕ is Euler's totient function.

A **primitive root** mod n is an element $g \in \mathbb{Z}_n^*$ whose powers generate all of $\mathbb{Z}_n^*$.

2 is a primitive root mod 5

- $2^0 = 1,\ 1 \pmod 5 = 1,$
- $2^1 = 2,\ 2 \pmod 5 = 2,$
- $2^3 = 8,\ 8 \pmod 5 = 3,$
- $2^2 = 4,\ 4 \pmod 5 = 4,$

4 is a primitive root mod 5?

- $4^0 = 1,\ 1 \pmod 5 = 1$
- $4^1 = 4,\ 4 \pmod 5 = 4$
- $4^2 = 16,\ 16 \pmod 5 = 1$
- $4^3 = 64,\ 64 \pmod 5 = 4$

Diffie–Hellman Key Exchange

Let P is a prime number, and G is the primitive root of P.

Steps	User1	User2
1.	**P, G** => available public keys.	**P, G** => available public keys.
2.	**a** is selected as a private key.	**b** is selected as a private key.
3.	Eq. to generate key: $x=G^a \bmod P$	Eq. to generate key: $y=G^b \bmod P$
4.	After exchanging keys, user1 receives key y.	After exchanging keys, user2 receives key x.
5.	User1 generates a secret key by using the received key y: $k_a=y^a \bmod P$	User2 generates a secret key by using the received key x: $k_b=x^b \bmod P$

Diffie–Hellman Key Exchange

1. User1 and User2 get public keys P = 33 and G = 8.
2. User1 selects a as a private key, i.e., 3, and User2 selects b as a private key, i.e., 2.
3. User1 calculate the public value:

   $x=(8^3 \bmod 33)=512 \bmod 33=17$
4. User2 calculate the public value:

   $y=(8^2 \bmod 33)=64 \bmod 33= 31$
5. User1 and User2 exchange public keys, i.e., 17 and 31.
6. User1 receives public key y = 31 and User2 receives public key x = 17.
7. User1 and User2 calculate symmetric keys:

   User1: $k_a=y_a \bmod P=31_3 \bmod 33=29791 \bmod 33=25$

   User2: $k_b=x_b \bmod P=17_2 \bmod 33=289 \bmod 33=25$
8. 25 is the shared secret.

Diffie–Hellman Key Exchange

Let P =7, G=3

a= 11, b=4

x = 3^11 mod 7 = 5 y=3^4 mod 7 = 4

Ka = 4^11 mod 7 = 2 Kb = 5^4 mod 7 = 2

Public Key Infrastructure (PKI)

- Public key infrastructures (PKIs) have been proposed as a workaround for the problem of identity authentication.

- In their most usual implementation, each user applies to a "certificate authority" (CA), trusted by all parties, for a digital certificate which serves for other users as a non-tamperable authentication of identity.

- These arrangements are best thought of as electronic notary endorsements that "this public key belongs to this user"

Public Key Infrastructure (PKI)

- The primary purpose of a PKI is to manage digital certificates.
- They are a powerful security tool that supports numerous operations.
- At a very high-level, the purpose of a PKI is to allow organizations to use encryption for their various security needs. It allows phishing-resistant authentication to applications or Wi-Fi.
- It is a complete infrastructure that contains many incredibly complex components.

Components of Public Key Infrastructure (PKI)

- Certificate Authority (CA)
  - Root Certificate Authority
  - Intermediate Certificate Authority
- Certificate Templates
- Certificate Lifecycle Management
- Method of Certificate Revocation
  - Certificate Revocation List (CRL)
  - Online Certificate Status Protocol (OCSP)

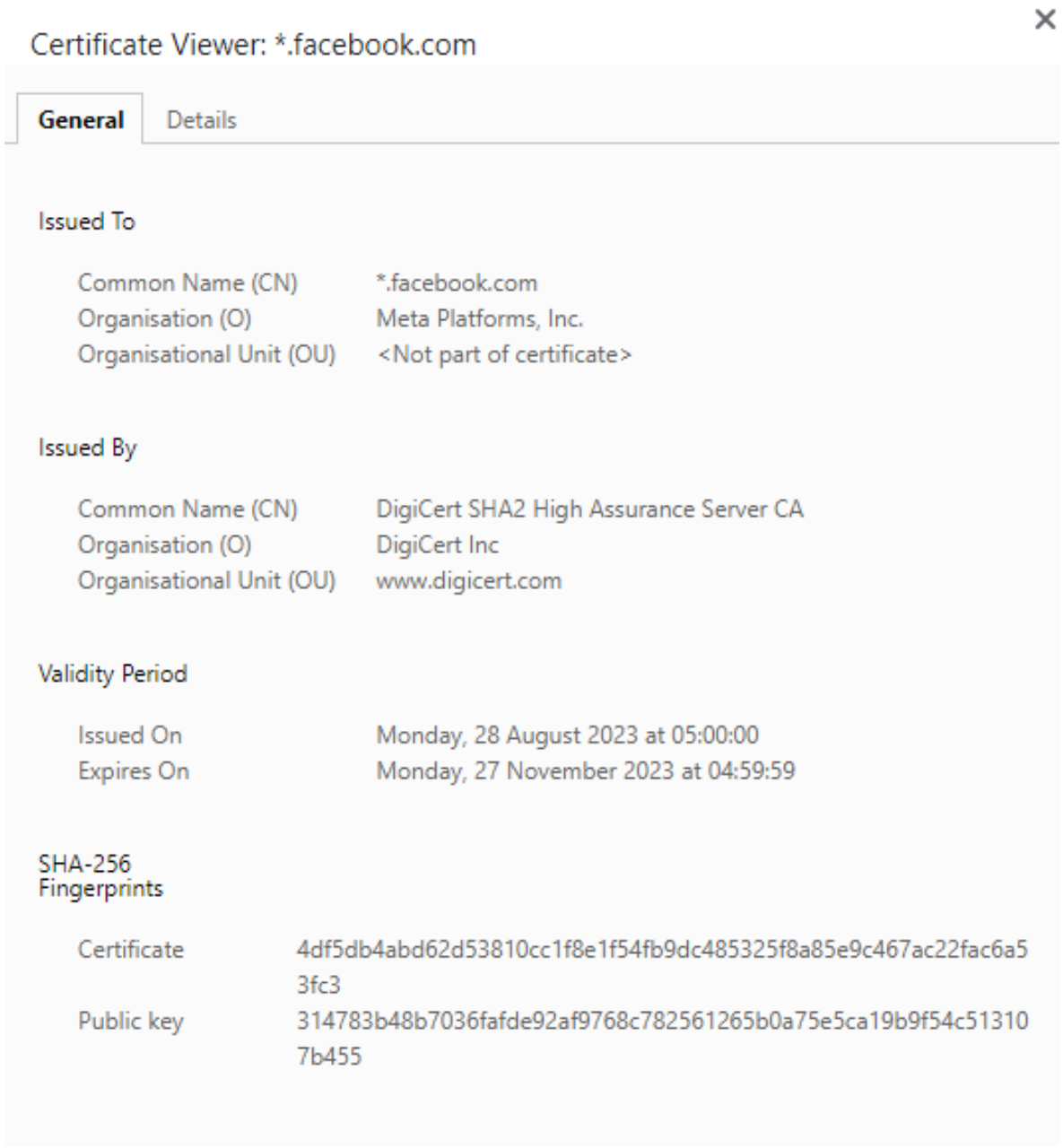
Certificate Authority (CA)

- A CA issues digital certificates to be used to confirm that the subject imprinted on the certificate is the owner of the public key.
- In a PKI system, the client generates a public-private key pair.
- The public key and information to be imprinted on the certificate are sent to the CA.
- The CA then creates a digital certificate consisting of the user's public key and certificate attributes.
- The certificate is signed by the CA with its private key.

Certificate Authority (CA)

- Certificate Authorities are all around us. Click on the lock near the URL of your web browsers, and you can see the Certificate Authority that is used to create encrypted web traffic between the web server and your web browsers.

Certificate Authority (CA)



Certificate Authority (CA)

- A Root CA is a trusted CA that is entitled to verify the identity of a person and signs the certificate that is distributed to a user.

- An Intermediate CA is also a trusted CA and is used as a chain between the root CA and the client certificate that the user enrolls for.

Certificate Template

- Certificate templates are used to determine how a certificate is structured, and what it will be used for.

Basic

Name: DEFAULT CERTIFICATE TEMPL

Subject: CN=${/auth/displayName:/dev

Display Description:

Validity Period: 1y

Override Validity Period: mm/dd/yyyy

Key Size: 2048

Signature Algorithm: SHA-256

Certificate Extensions

SAN

Other Name: ${/auth/upn:/device/identity}

RFC822: ${/auth/email:/device/identity}

DNS: ${/device/computerIdentity:/d

URI:

Extended Key Usage

Use Certificate For:

- Server Authentication
- Client Authentication
- Code Signing
- Email Protection
- Smart Card Logon

Microsoft

Certificate Template Name / Certificate Template OID

Certificate Lifecycle Management

- A certificate management system is a very important part of a PKI. Lifecycle management encompasses all phases of the certificate lifecycle, which can be broken down into the following phases.
- **Certificate Enrollment:** An entity submits a request for a certificate to the Certificate Authority (CA). An entity can be a person, a device, or even just a few lines of code.
- **Certificate Issuance:** The CA needs to validate the identity of the applicant, which is typically done through credentials or by trusting another CA that has already validated the applicant.

Certificate Lifecycle Management

- **Certificate Validation:** Every time the certificate is used to authenticate, the RADIUS (Remote Authentication Dial-In User Service) server checks with the CA to confirm that the certificate is still valid and hasn't expired or been revoked.
- **Certificate Revocation:** Certificates contain an expiration date that's specified when they are first issued, usually for a duration of several years. When that date is reached, the certificate will automatically be considered invalid for any authentication attempt.

Certificate Lifecycle Management

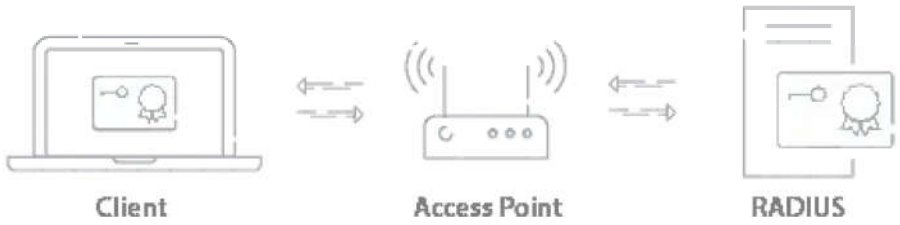
- **Certificate Renewal:** Instead of automatically being shunted to a CRL (Certificate Revocation List), some CA's have settings that renew certificates upon expiration date, though typically they re-verify identity. At this time, you can choose whether or not to generate a new key pair – effectively making it a totally new certificate.

Methods of Certificate Revocation

- **Certificate Revocation List (CRL):**
- A CRL is a list of the serial numbers of certificates revoked before their expiration date. Certificate revocation lists are downloaded and cached into a service authenticating certificates, allowing the service to efficiently check for revoked certificates.
- **Online Certificate Status Protocol (OCSP):** OCSP is a longstanding protocol that is mostly used by web servers. Everytime the authentication process happens. A request is made to the OCSP server to check to see if the certificate requesting authentication is still valid.

PKI Uses

- **Wifi Authentication:**
- Certificate-based Wi-Fi authentication uses the EAP-TLS protocol. The TLS stands for Transport Layer Security, and is generally used as a term to describe any certificate-based authentication. At it's core, the authentication process is the same as any asymmetric authentication, where the client presents it's public key, and the authenticator (the RADIUS Server) verifies it with a private key.

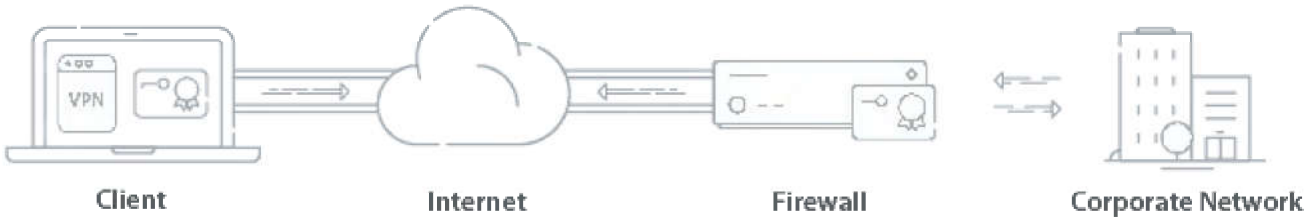


PKI Uses

- **Web Application Authentication:**
- Similar to Wi-Fi authentication, a user connecting to a web application will have their identity confirmed by the web application server. Since the certificate is signed by the trusted CA, they are able to gain access to the application.

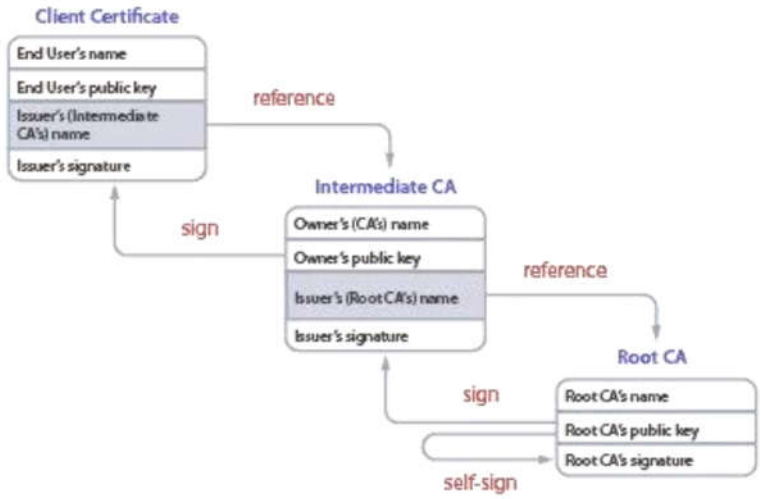
PKI Uses

- Email Security:
- Encrypting emails with certificates utilizes the S/MIME (Secure/Multipurpose Internet Mail Extensions) protocol. Both the receiver and sender are required to have a certificate signed by the CA to establish trust between the users. S



Certificate Trust Chain

- This multi-leveled hierarchy of trust is called a certificate chain. You can trace the chain from the client's certificate all the way back to a single root CA, and every chain ends with a person (or company) from which all the trust is ultimately derived.



PKI Algorithms

- **AES 256 Certificate:**
- The AES 256 certificate is an algorithm and the current encryption standard. The previous standard was AES 128. AES 256 keeps track of vulnerabilities and when the encryption has been breached, a higher standard of encryption will be implemented. The higher the standard encryption, the better cryptic the public/private key pair is. An AES 256 certificate is a long length key that causes brute force attacks virtually impossible.

PKI Algorithms

- Diffie Hellman:
- Diffie Hellman, also known as exponential key exchange, is a method of encryption that uses numbers raised to specific powers that produce decryption keys on the basis of components that are never directly shared, making it overwhelming for potential threats to penetrate. The algorithm creates a mathematically complex encryption that is shared between two parties over a secret communication over a public network so that they can allow an exchange of a private encryption key.

PKI Algorithms

- RSA Key Exchange:
- RSA is much like the Diffie Hellman algorithm and factors large integers that are the product of two large prime numbers. RSA key exchange uses public and private keys, while the public key can be shared with everyone, the private key must be kept secret. However, in RSA cryptography either of the public or private key can be used to encrypt a message while the other is used to decrypt.