Digital Certificate

What do we need?

- We need a way to ‘bind’ a public key to an identity!

This requires:

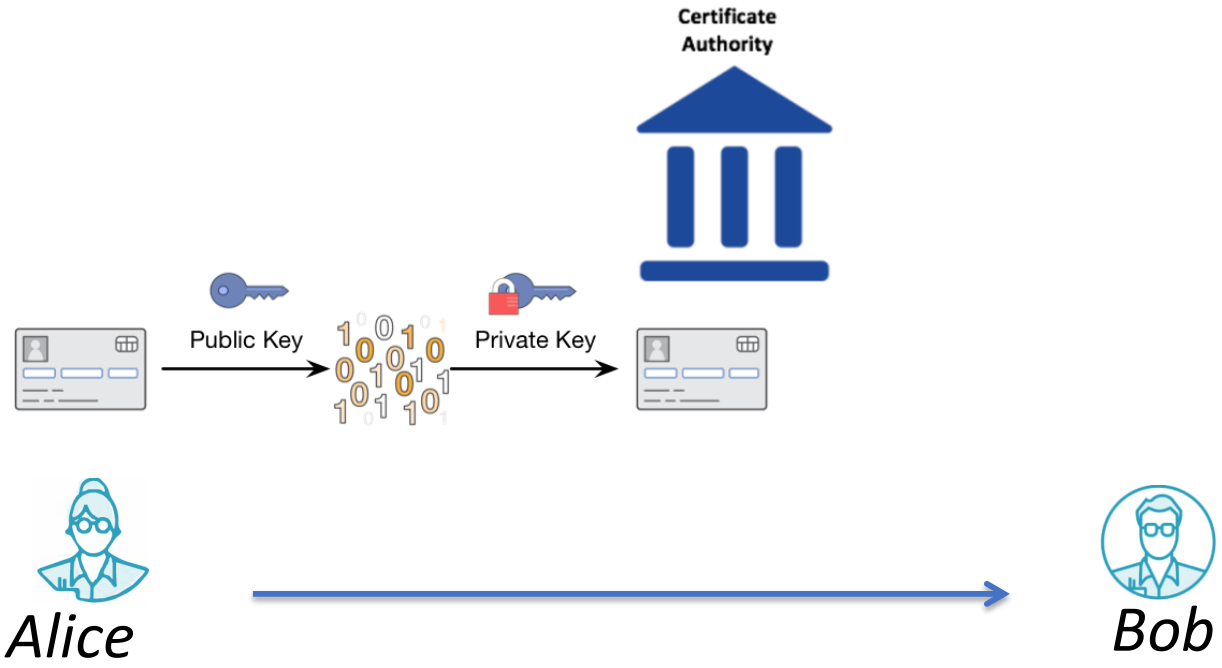
1. A set of mechanisms, format and infrastructure to “manage digital identities”

   => Digital Certificates and Public Key Infrastructure (PKI)

2. A crypto tool to authentic  e certificate?

   => Digital Signature

The only solution: a TRUSTED third party!



The binding Alice with PK=876543 is 'guaranteed' by the Certificate Authority (CA)

Public Key Certificate

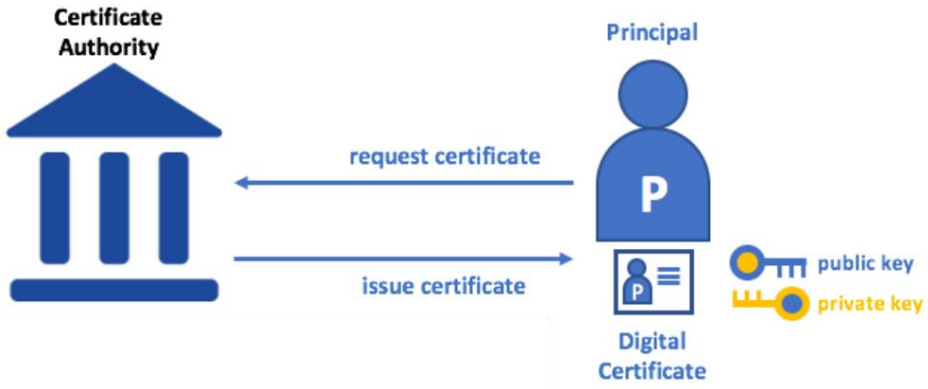
- A public key certificate is a data structure that binds a **public key** (and therefore the related private key) to the identity of the legitimate owner → $CERT_{ID}$:{ID, Pub_{ID}}
- The binding between {ID, Pub_{ID}} is granted by a trusted certification authority that signs $CERT_{ID}$
- Provided that we have the CA's public key, we can verify the CA signature and therefor verify the public key authenticity

Example:

CA issues a certificate for Alice $CERT_{alice}$

$CERT_{alice}$ contains:

1) Alice name, Pub_{alice}
2) CA identity CA_{id}
3) CA signature of $CERT_{alice}$



NOTE: Once I have the authentic Pub_{alice}, I STILL need to verify that the party I'm communicating with is actually Alice

This is the actual AUTHENTICATION procedure:

- I trust CA and I have CA's public key
- Verify CA signature $CERT_{aclice}$ → OK!
- Pub_{alice} is authentic
- I can encrypt a message for Alice

Certificates - Recap

- In order to authenticate public keys (and thus, prevent the Man in the Middle attack), all public keys are digitally signed by a central trusted authority.
- Such a construction is called *certificate*

certificate = public key + ID(user) + digital signature over public key and ID

- Acertificate for the key k_{pub} of user Alice is:

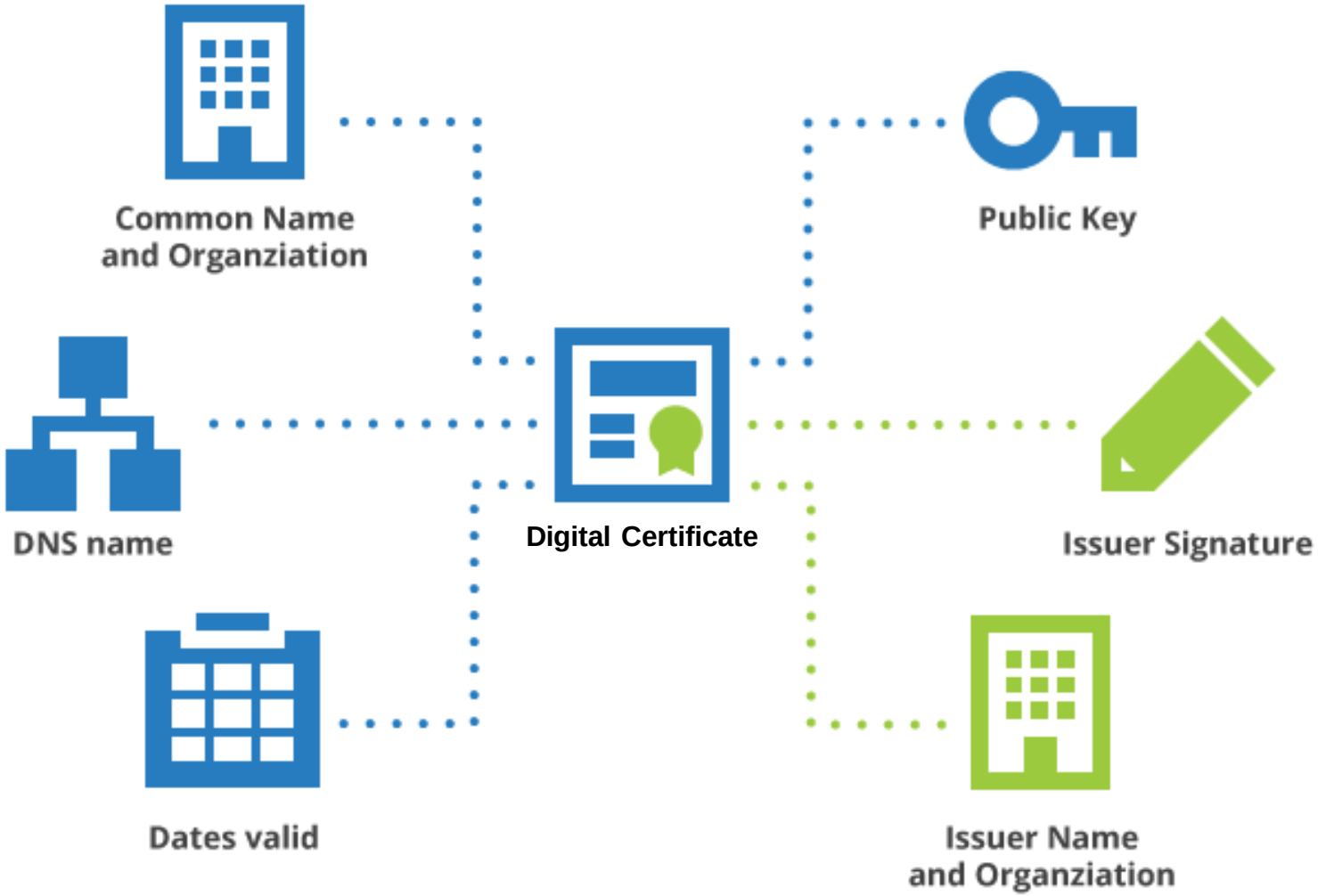
$$\textbf{Cert(Alice)} = (k_{pub}, \textit{ID(Alice)}, sig_{K_{CA}}(k_{pub}, \text{ID(Alice)}))$$

- Certificates bind the identity of user to her public key
- The trusted authority that issues the certificate is referred to as ***certificate authority (CA)***
- "Issuing certificates" means in particular that the CA computes the signature $sig_{K_{CA}}(k_{pub})$ using its (super secret!) private key k_{CA}
- The party who receives a certificate, e.g., Bob, verifies Alice's public key using the public key of the CA

Public Key Infrastructure (PKI)

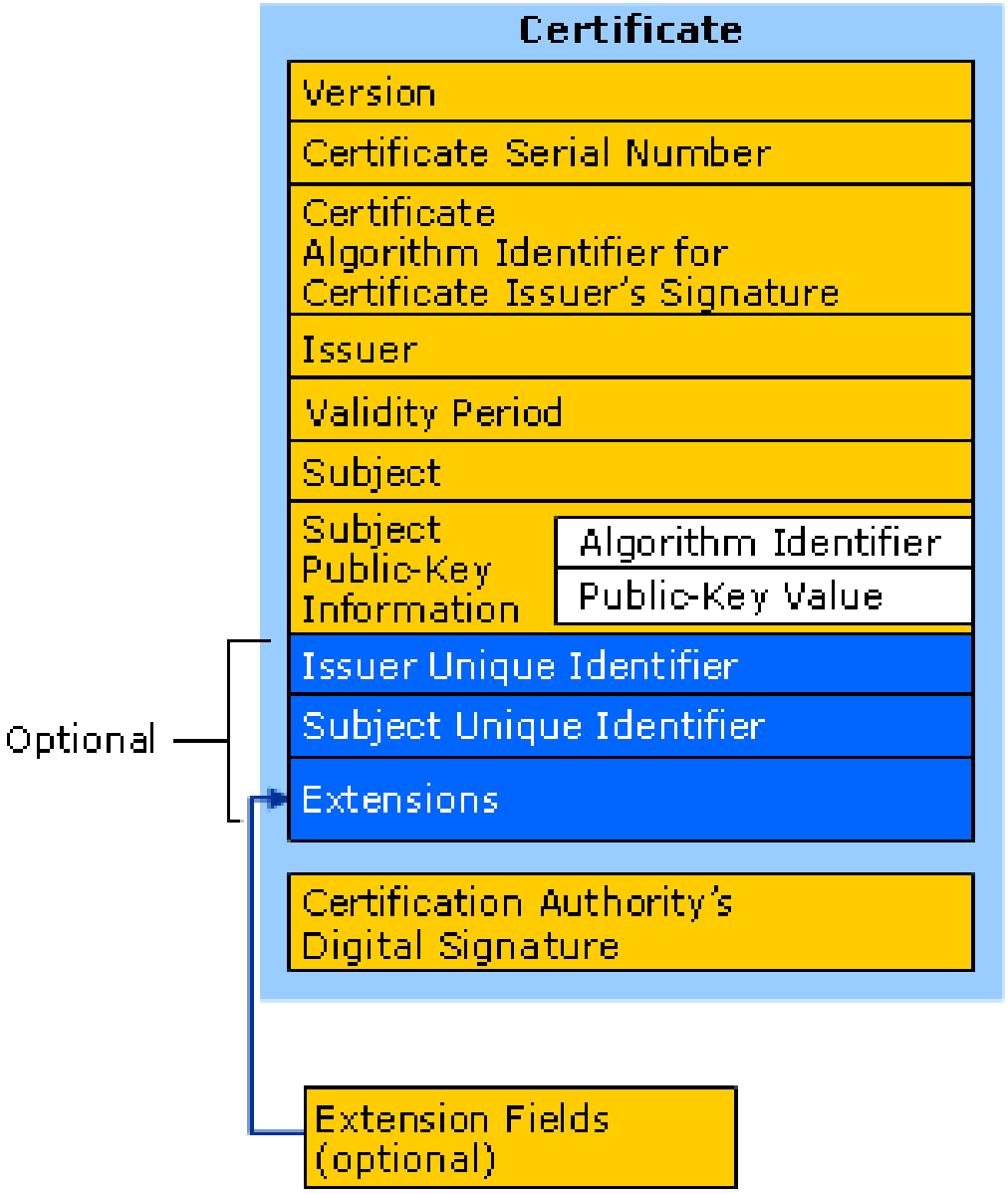
- A PKI specifies protocols, policies, and technical mechanisms needed to support exchange of public keys
- A PKI architecture requires:
  - Standard format for certificates
  - Relation among CAs, and with end users
  - Policies for issuing and REVOKING (!!) certificates
  - *Directory services*
- Typical certificate format: X.509

Anatomy of a Certificate

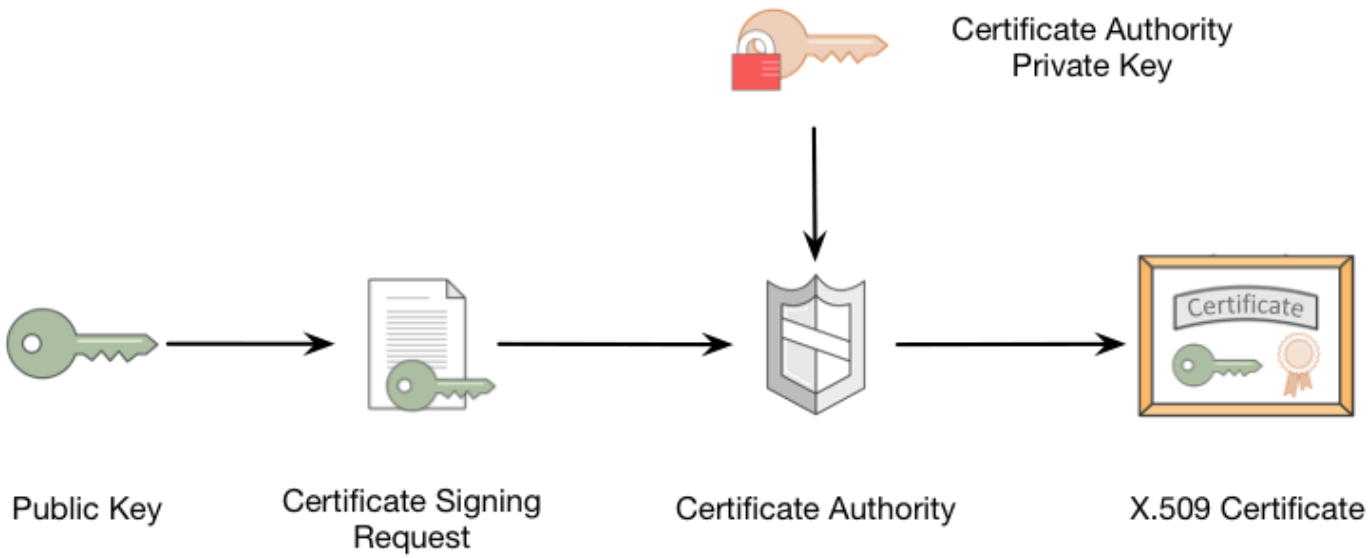


X.509 Certificate

- X509 is a popular DC standard. The main fields of such a certificate are shown to the right.
- CA Signature at the bottom is computed over all other fields in the certifcate (after hashing them).



Request a Digital Certificate



- An X.509 certificate is a document used to prove ownership of a public key.
- To make a new X.509 certificate you need to create a Certificate Signing Request (CSR) and sent it to a Certificate Authority (CA).
  - CSR is a digital document that contains your public key and other identifying information
- Once your identity has been verified (e.g., ownership of the domain mentioned in CSR), the CA creates a certificate and signs it with a private key.
  - Anyone can now validate your certificate by checking its digital signature with the CA's public key

Trust Chains

Root Certificate:
- Self-Signed, well known
- Has Root Certificate public key
- Signed by Root Certificate private key

Certificate 1:
- Has own public key
- Signed by Root Certificate private key
- Use Root Certificate public key to prove authenticity

Certificate 2:
- Has own public key
- Signed by Certificate 1 private key
- Use Certificate 1 public key to prove authenticity

- The chain of trust allows anyone to check the authenticity of any certificate by examining it all the way to a well-known, trusted root certificate.
- In Windows use **CertMgr** to see the installed Root Certificates

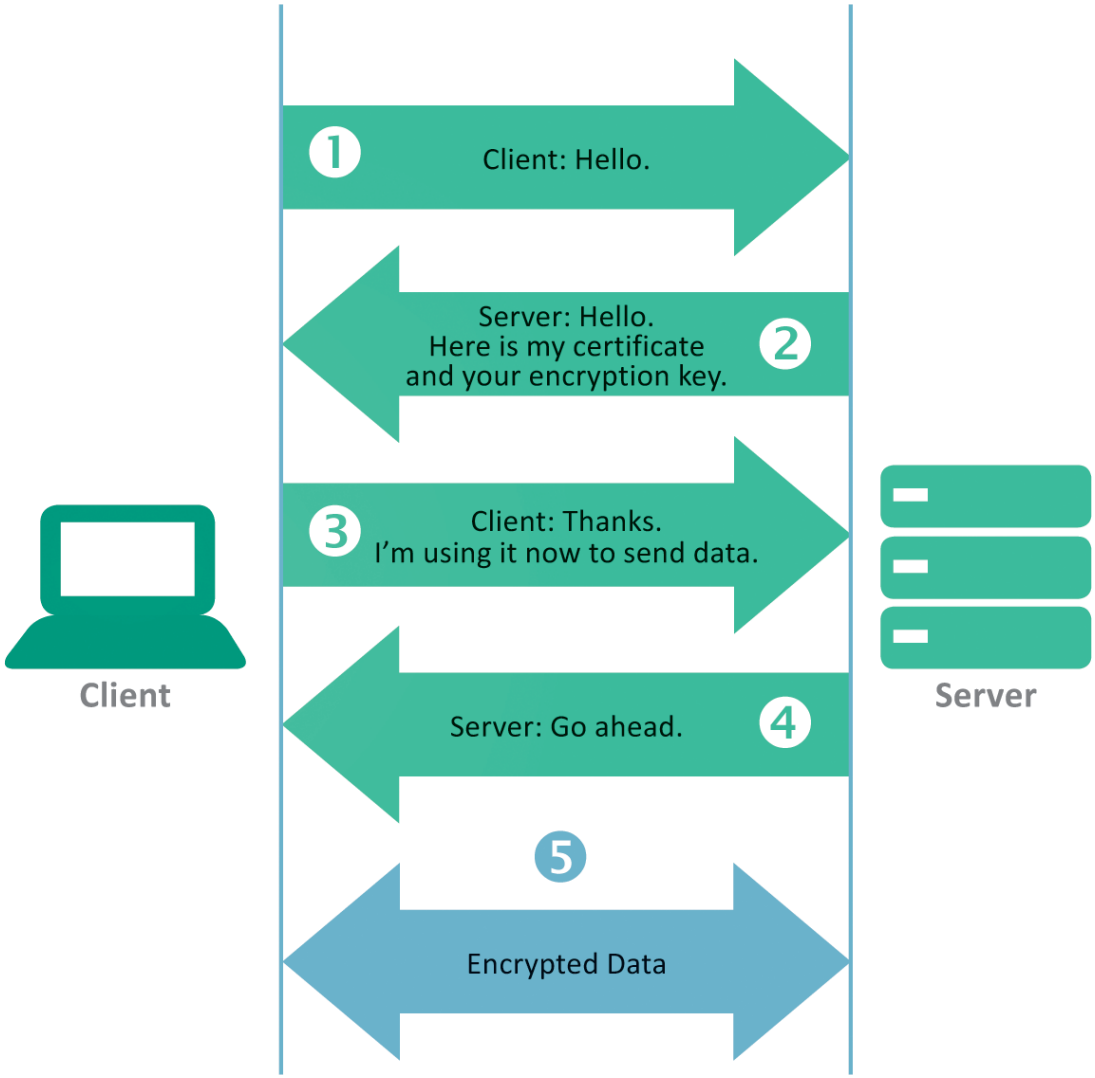
Certificate Revocation

- A PKI MUST include mechanisms for handling compromised certificates!
  - Examples:
    - Public keys whose private keys have been disclosed
    - Public keys whose private keys have been lost
    - Public keys not anymore used
- Two major (and coexisting) approaches
  - Validity period for a certificate
  - Explicit revoocation
- Classical analogy: a credit card

Revocation Approach: *Certificate Revocation List* (CRL)

- Every CA must regularly publish a list of certificates the CA has revoked
- List format: must include
  - Issuer, Last update date, Next update date
  - List of serial numbers revoked, along with revocation date
  - CA digital signature
- In principle, you SHOULD verify that the certificate of the site you are accessing has NOT been revoked
  - Overhead:
    - Online access, or offline periodic download of CRL
  - In practice… frequently overlooked by non critical applications
    - But remember, whenever you are involved in a security critical application!

Digital Certificate in HTTPS



Resources

- Digital Certificate

https://en.wikipedia.org/wiki/Public_key_certificate

- Free Certificate

https://letsencrypt.org/