COMP6224
Secure Communication part 1 – Encryption, Digital Signature and Certificates

Dr Federico Lombardi
f.lombardi@soton.ac.uk

cybersecurity centre
Academic Centre of Excellence

CyberSecuritySoton.org [w]

@CybSecSoton [fb & tw]

This week

- Recap of basic cryptographic primitives

- Key Distribution problem
  - Diffie-Hellman protocol
  - Digital Certificates
  - Public Key Infrastructures (PKI)

- Security protocols
  - Kerberos
  - TLS/SSL

Learning Outcomes

At the end of this lecture you should be able to:

- Understand the issue of distributing cryptographic keys
- List and explain the elements of an X.509 certificate
- Present an overview of public key infrastructure concepts
- Understand how to use Kerberos for authentication
- Understand how TLS/SSL protocol works

Cryptography Primitives

- Hash functions:
  - A hash function h maps a piece of information **P** to a fixed-length value **x = h(P)** called hash value or digest of **P**
  - **Question:** What property does it ensure?

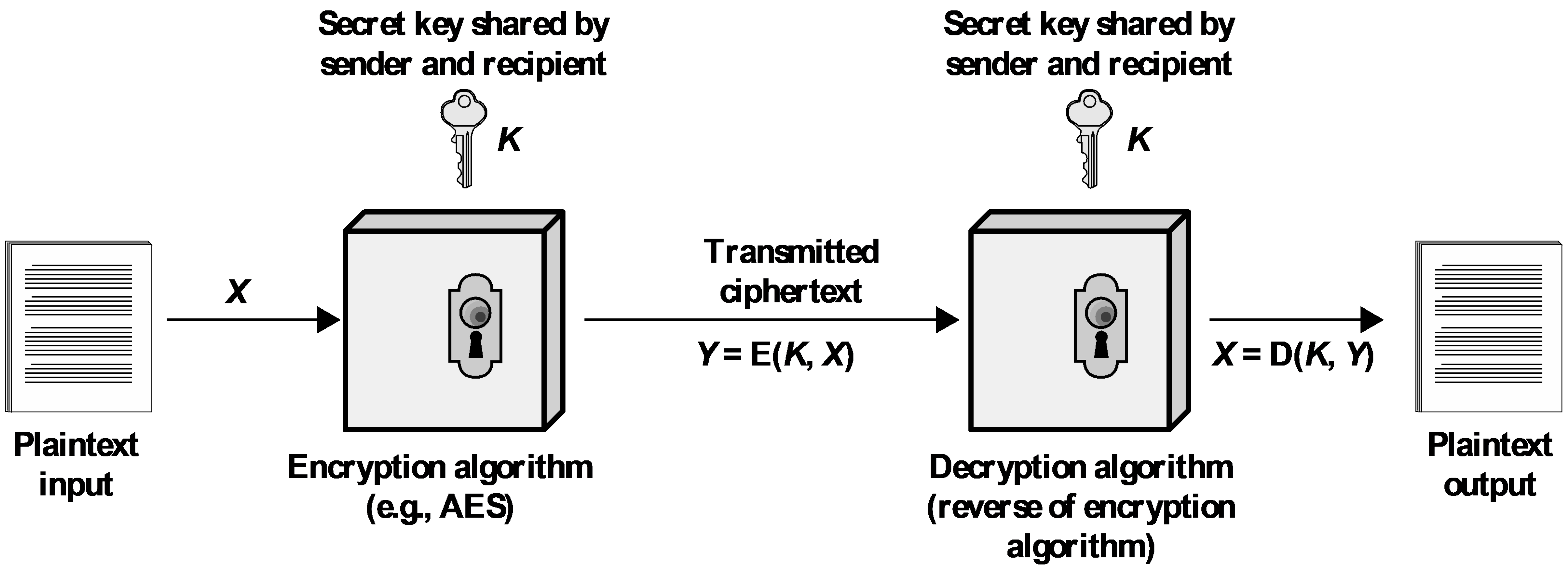
Cryptography Primitives

- Hash functions:
  - A hash function h maps a piece of information **P** to a fixed-length value **x = h(P)** called hash value or digest of **P**
  - It ensures **integrity** of x
  - Examples: SHA-256, SHA-3

GCHQ Academic Centre of Excellence EPSRC
cybersecurity southampton

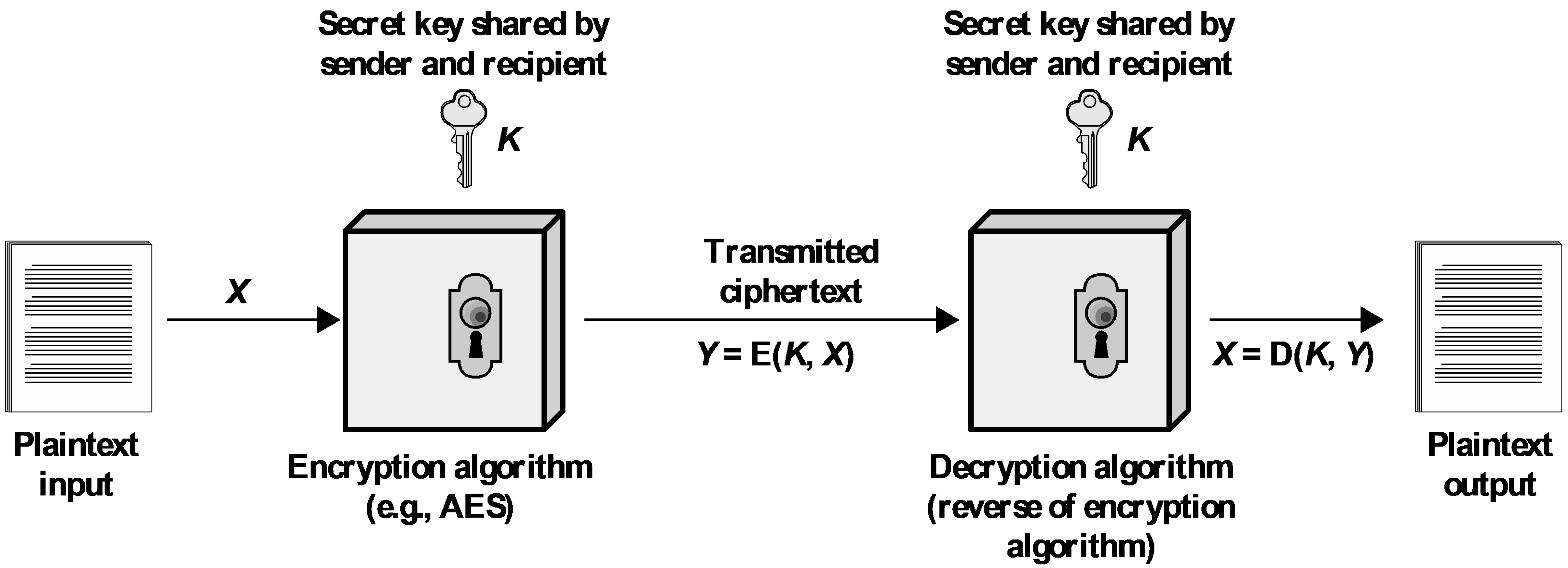
Symmetric encryption

- The same key is used to encrypt and decrypt a piece of information x
- **Question:** What property does it ensure?



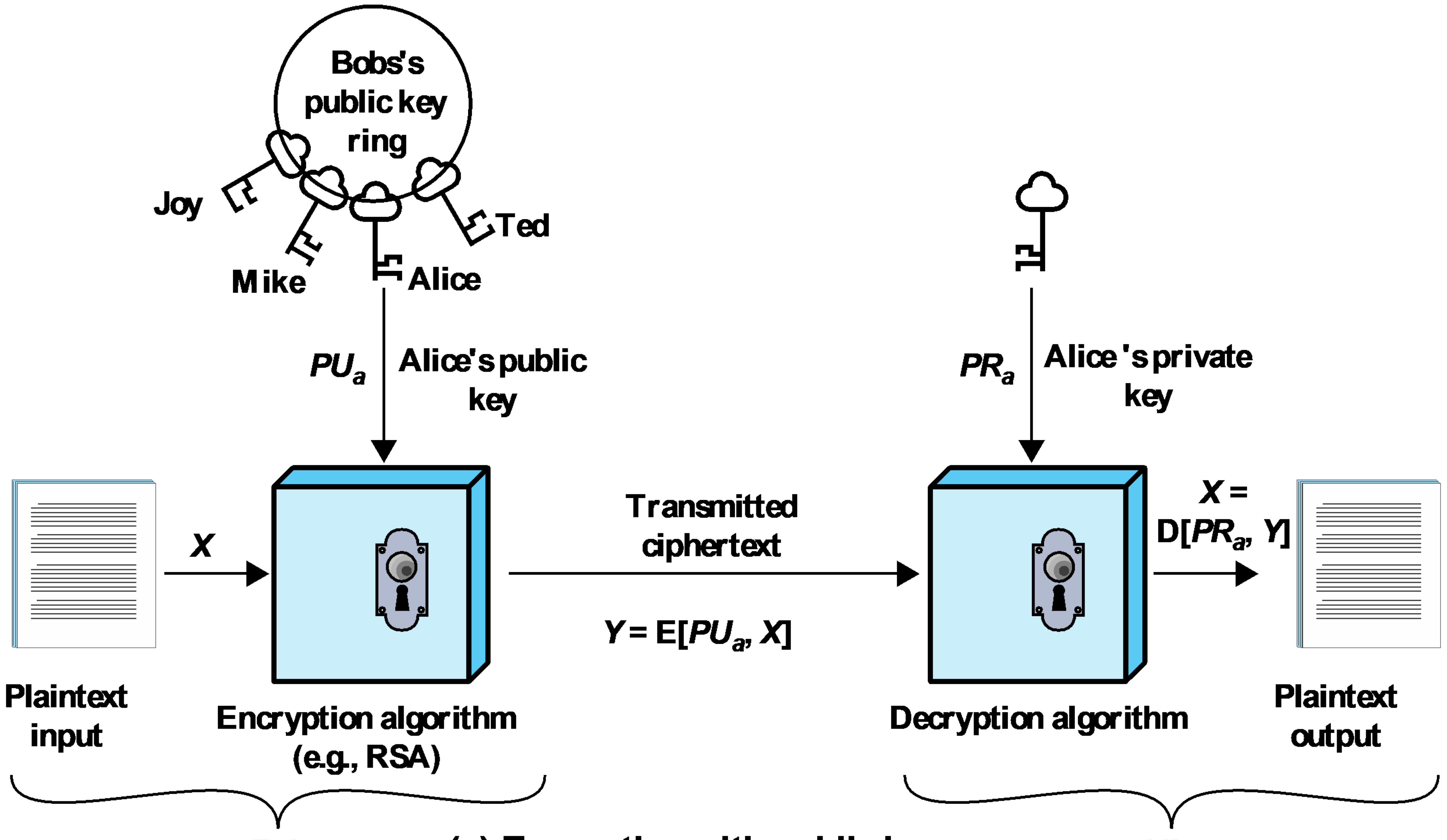
Symmetric encryption

- The same key is used to encrypt and decrypt a piece of information x
- It ensure **confidentiality** of x
- Examples: Advanced Encryption Standard



Public key encryption

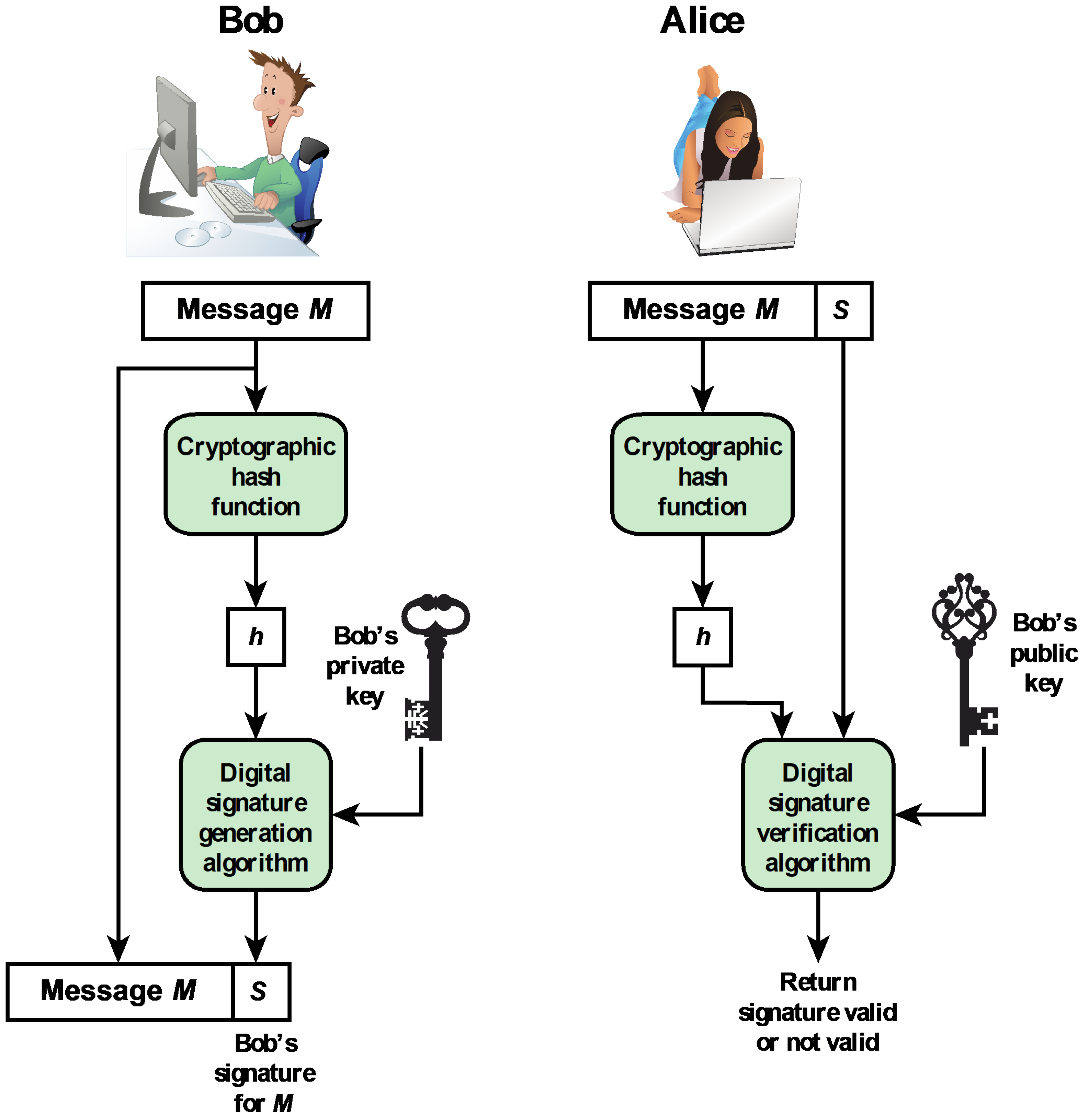
- Sender encrypts a piece of information x with the public key of the recipient
- The recipient decrypts with its private key
- Main applications: symmetric key distribution, and digital signatures
- Examples: RSA, DSA



(a) Encryption with public key

Digital signature

- It guarantees **integrity** of x, **source authentication**, and **non-repudiation**



(a) Bob signs a message (b) Alice verifies the signature

The key distribution problem

- Symmetric key cryptography requires shared, secret keys between each pair of communicating parties

- How are all these keys shared in the first place?
  - Public-key cryptography
  - Diffie-Hellman protocol

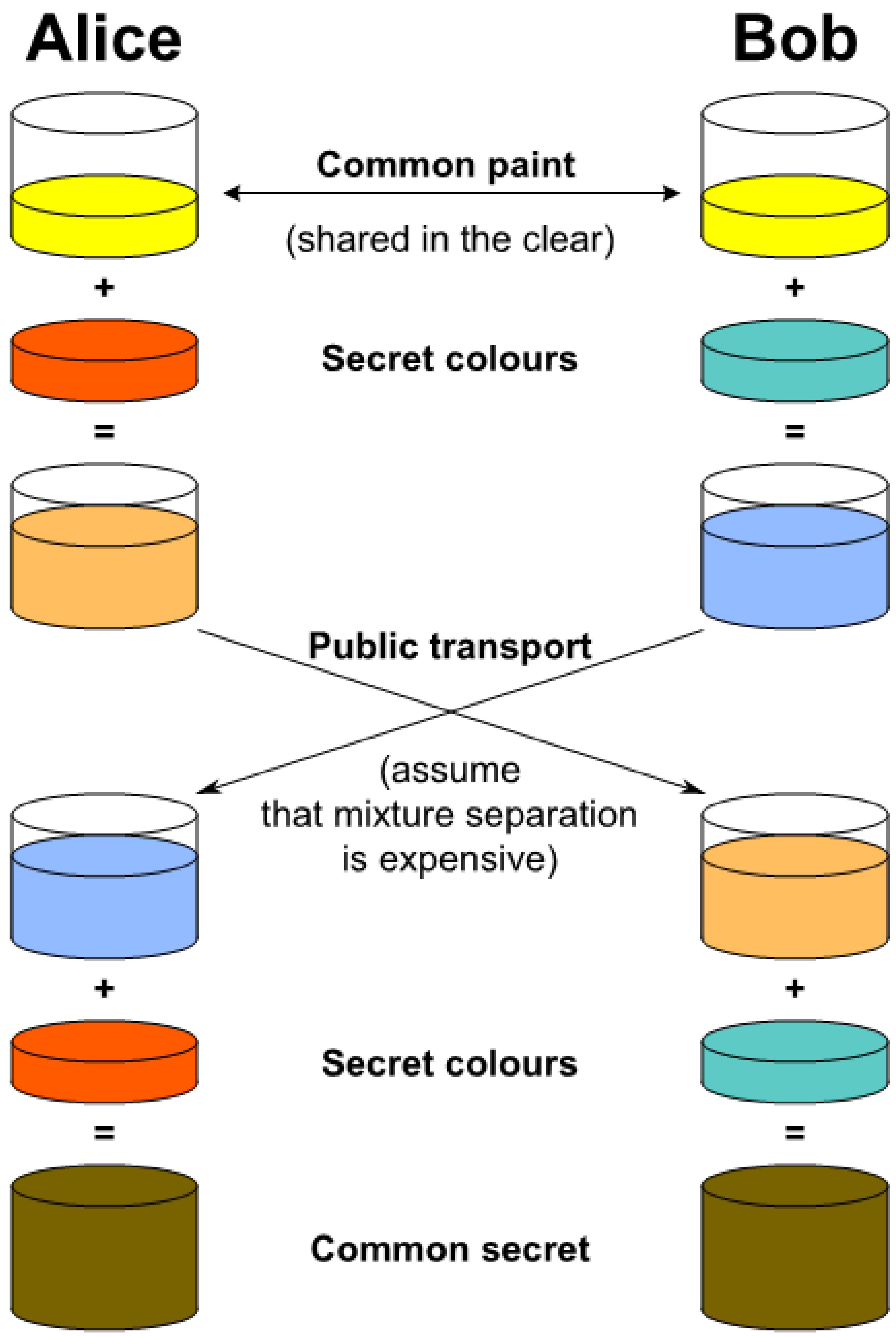
The Diffie-Hellman key exchange protocol

- Purpose is to enable two users to securely exchange a key that can then be used for subsequent symmetric encryption of messages

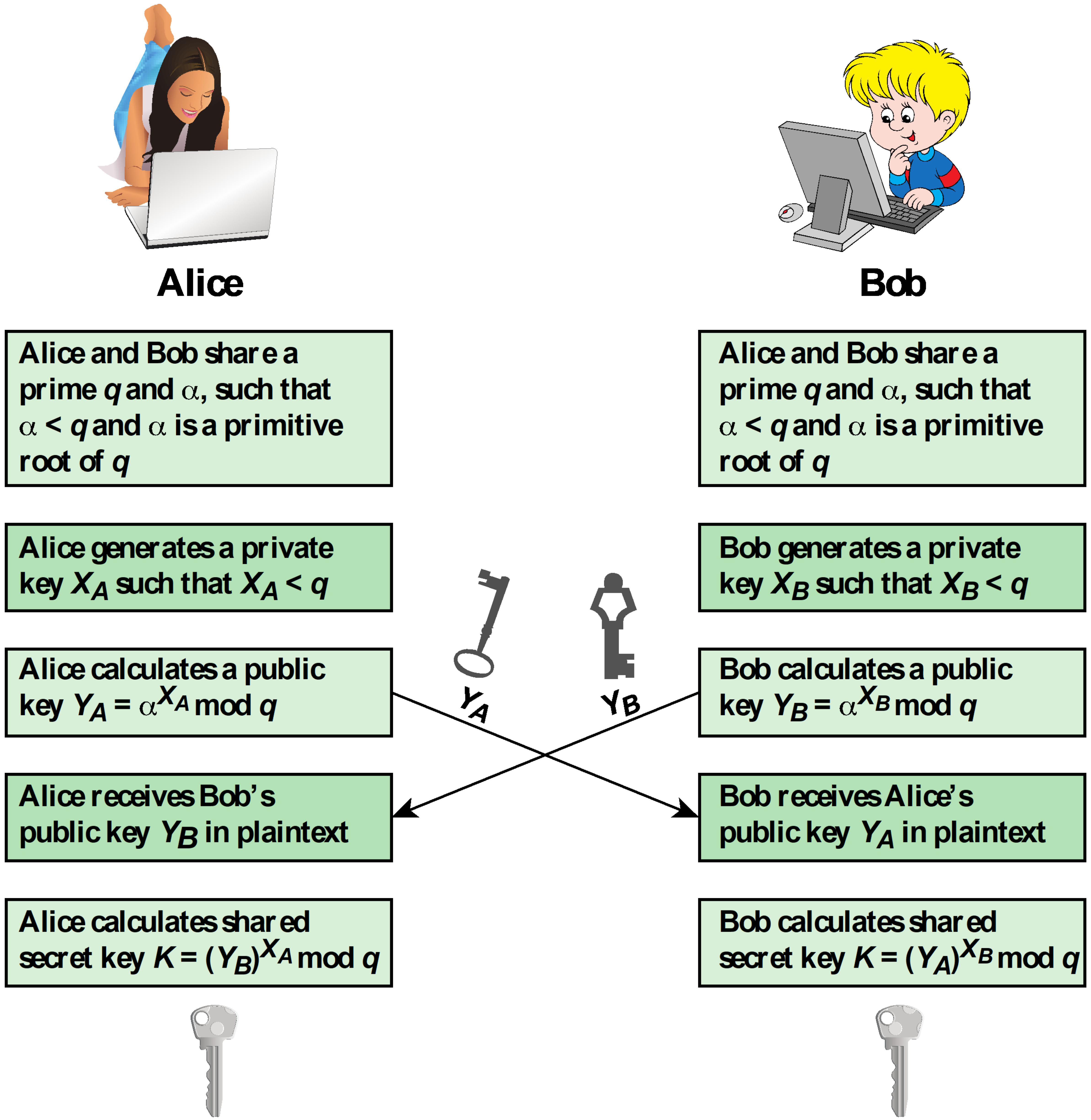
- The algorithm itself is limited to the exchange of secret values

- Its effectiveness depends on the difficulty of computing **discrete logarithms**

The Diffie-Hellman key exchange protocol



The Diffie-Hellman protocol



A modular arithmetic refresher

- A **prime number** is a number q that has as divisors 1 and q itself
  - Examples: 3, 11, 13, 349…
- **c mod q** denotes arithmetic modulo q
  - The result of c mod q is the remainder of the division of c by q
  - Examples: $29 \bmod 13 = 3 \quad 29 = 3 + 2\times 13 \quad 13 \bmod 13 = 0 \quad 13 = 0 + 1\times 13$
- The **primitive root** of α prime number q is a number whose powers modulo q 1 generates the integers from 1 to q-1

$$\alpha \bmod q,\ \alpha^2 \bmod q,\ \ldots,\ \alpha^{p-1} \bmod q \quad 0 \le i \le q-1$$

Example: If q = 13 a primitive root of 13 is 2

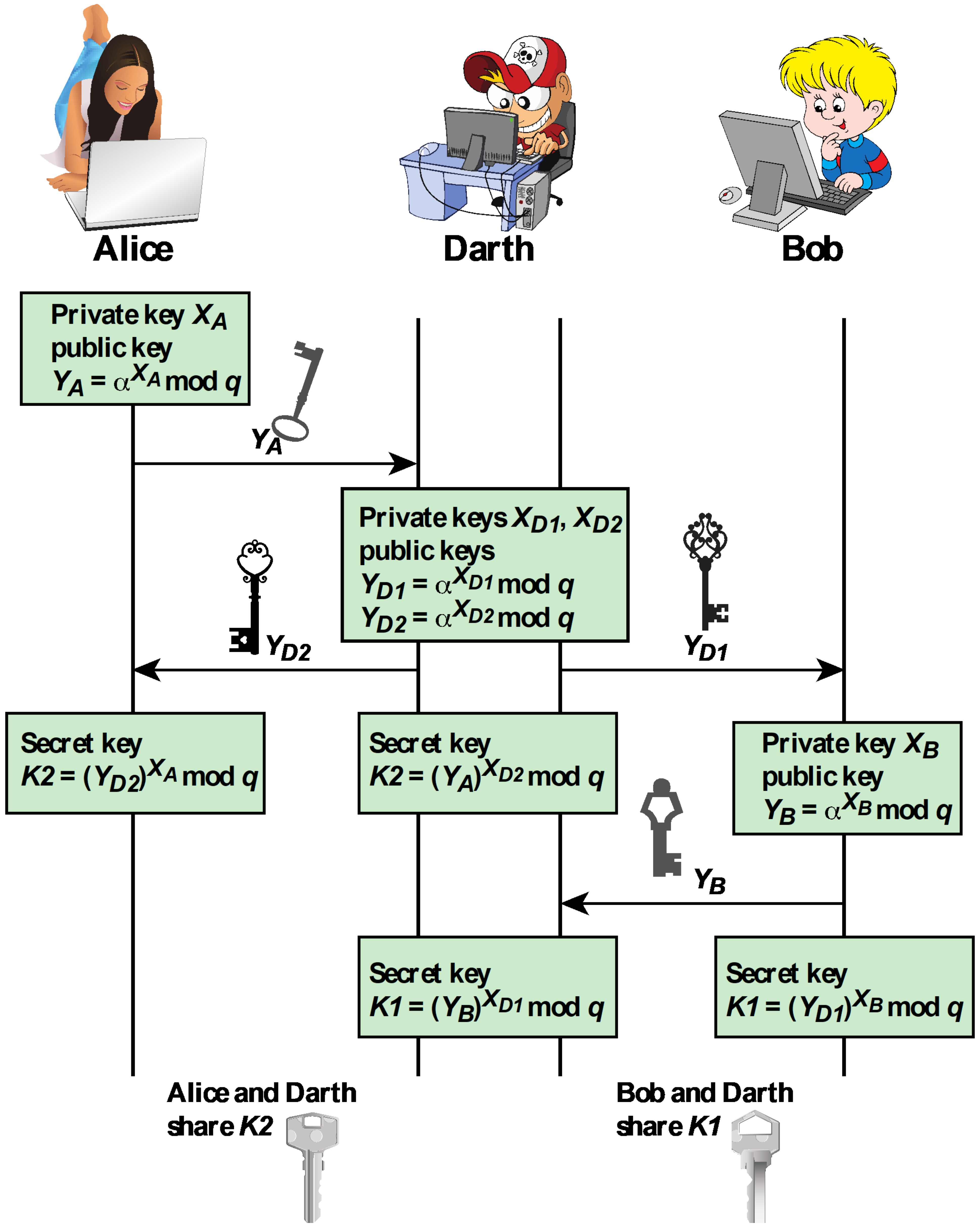
- Given an integer b, α primitive root a, and a prime number q, we define i as the **discrete logarithm** of b for the base a and module q

$$b \cong \alpha^i \bmod p \quad 0 \le i \le p-1$$

An Example

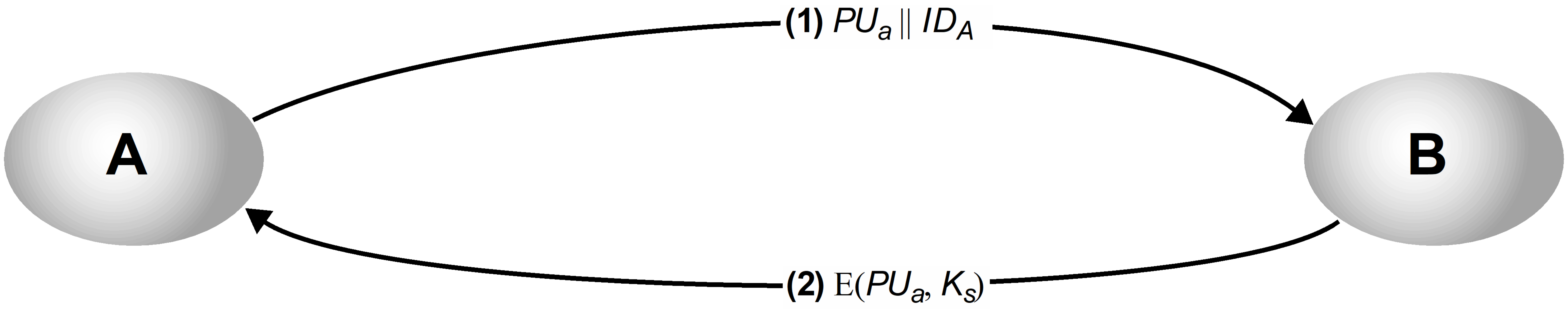
1. $q = 353$ and a primitive root of 353 is $\alpha = 3$
2. Alice selects $X_A = 97$ (private key)
3. Bob selects $X_B = 233$ (private key)
4. A computes $Y_A = 3^{97} \bmod 353 = 40$ (public key)
5. B computes $Y_B = 3^{233} \bmod 353 = 248$ (public key)
6. A and B share their public keys
7. A computes $K = (Y_B)^{XA} \bmod 353 = 248^{97} \bmod 353 = 160$
8. A computes $K = (Y_A)^{XB} \bmod 353 = 40^{233} \bmod 353 = 160$

Man in the Middle Attack

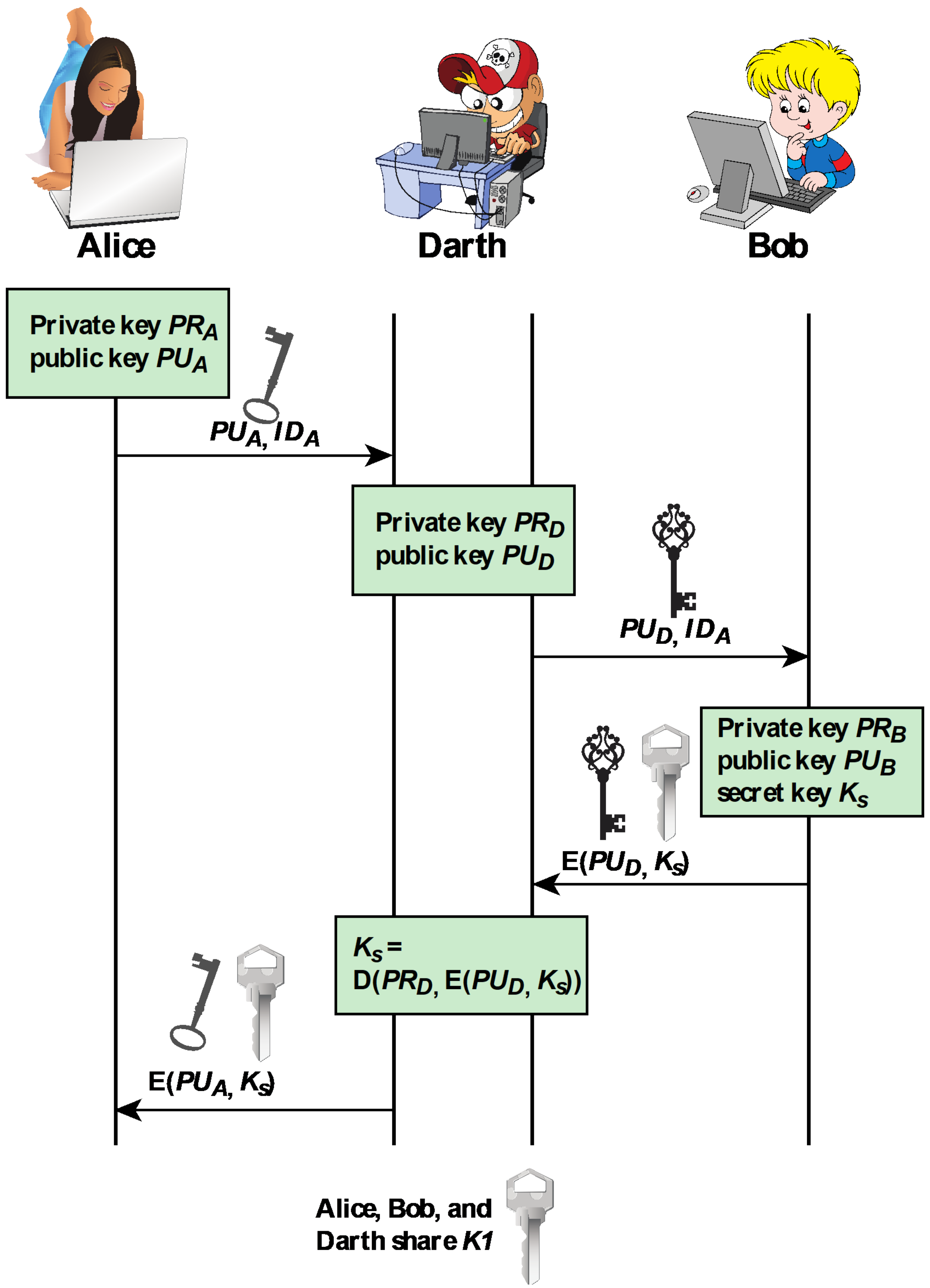


Key Distribution with Public Key Encryption

- PU = public key
- ID = identity
- Ks = secret key, sent encrypted from B to A with A's PU (public key)



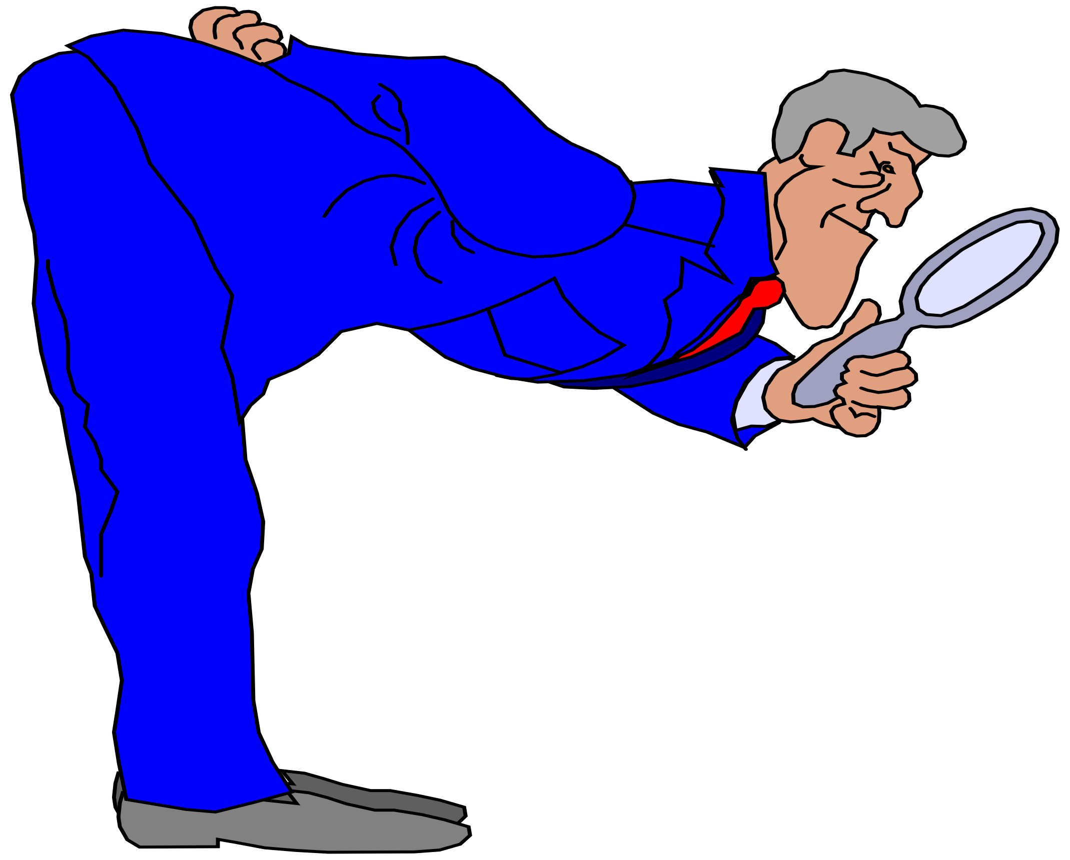
Main in the Middle Attack



The Critical Question

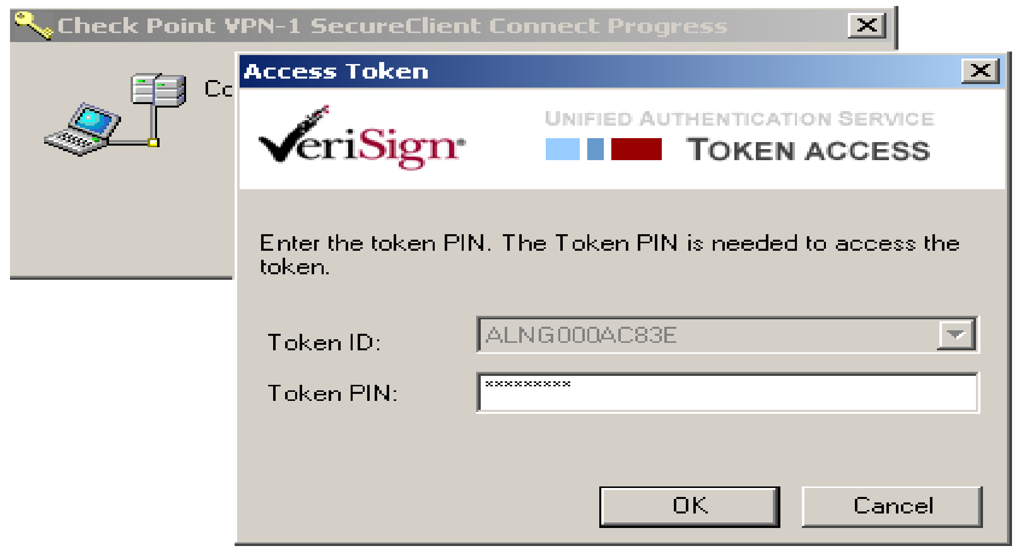
- How can the **recipient** know with certainty the **sender's public key**?
  (to validate a digital signature)

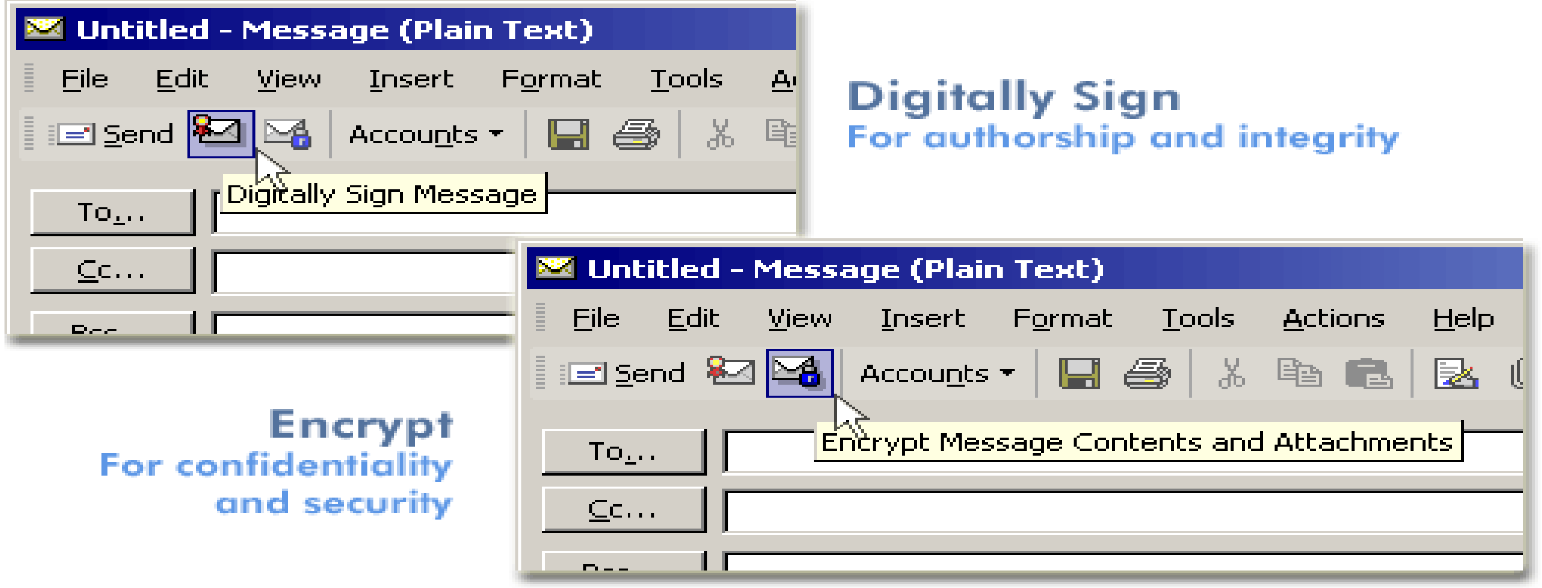
- How can the **sender** know with certainty the **recipient's public key**?
  (to send an encrypted message)

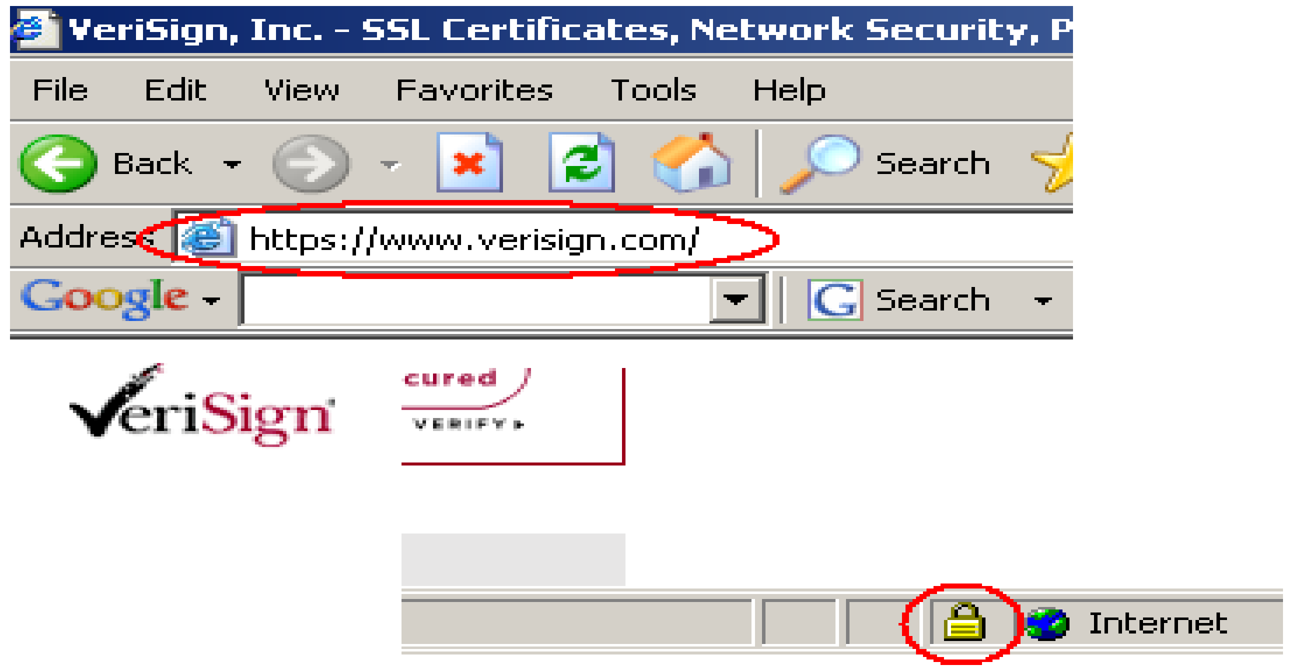


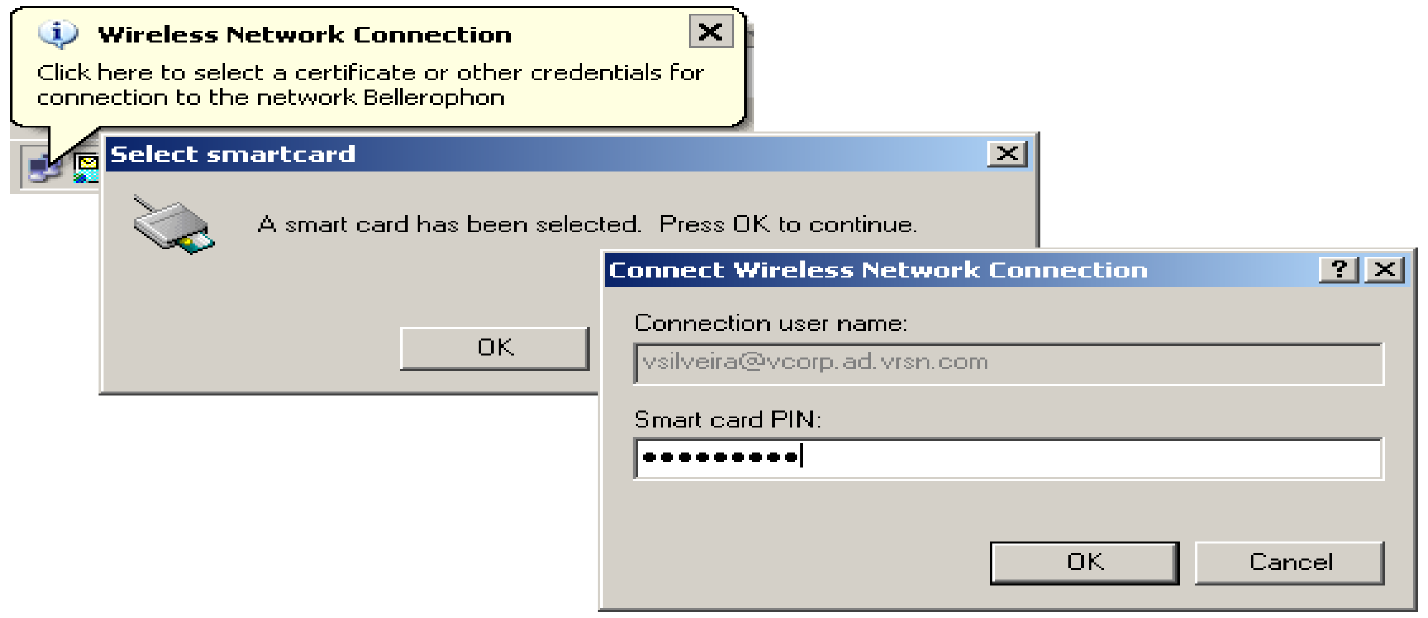
Digital Certificates

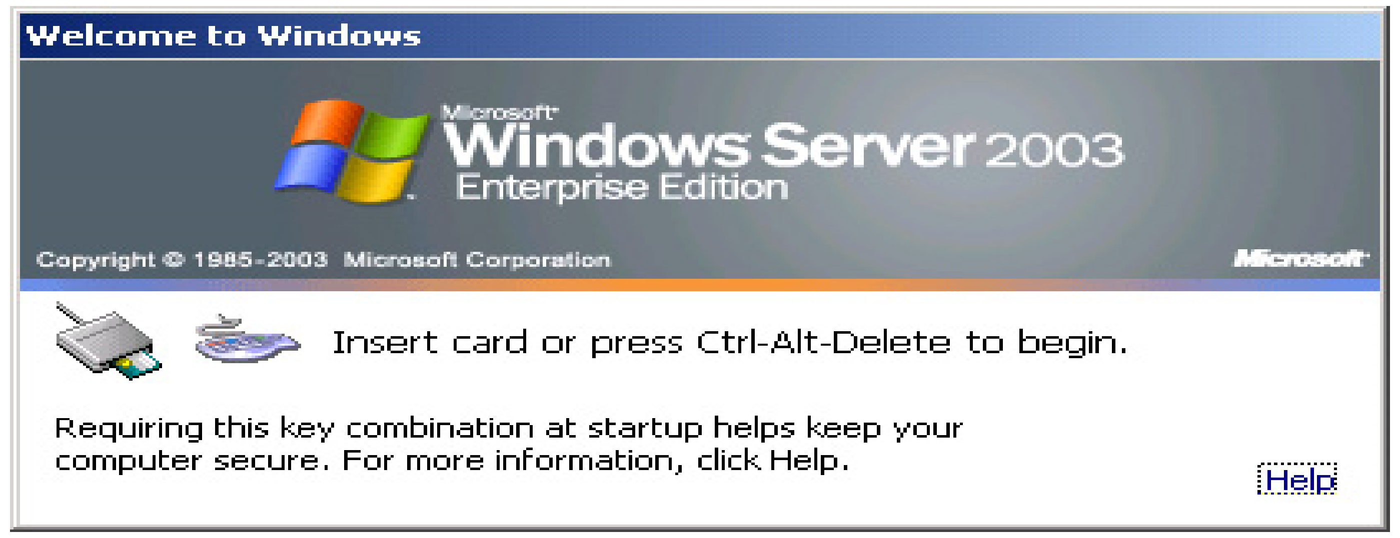
- Secure e-mail
- Virtual Private Networks
- Wireless (Wi-Fi)
- Web Servers (SSL/TLS)
- Network Authentication
- Code Signing











Digital Certificate

UNIVERSITY OF Southampton

- Binds a user / company identity to his public key
- Standard: X.509

Certificate Extensions

Key Usage:	digitalSignature dataEncipherment keyCertSign nonRepudiation keyAgreement encipherOnly keyEncipherment cRLSign decipherOny
Extended Key Usage:	serverAuth codeSigning timeStamping clientAuth emailProtection OCSPSigning
Certificate Policies:	URL of CPS and Policy notice text
Subject Alternative Name:	rfc822name, IP Address, DNS Name
CRL Distribution Point:	URL of the Certificate Revocation List

Version: v3	
Serial No: 001b6f945h75	
Algorithms: MD5 RSA	
Subject DN: John Doe	Issuer DN: State of Kansas
Validity period: from 11-03-2005 to 11-05-2005	
Public key: 30 81 89 02 81 81 00 ba 6e e5 9a 74 f5 e7 af a9 8a 9c de a8 e5 53 1b 73 c7 f7 8a 13 f3 44 91 09 dc 91 12 b7 1b b2 cf 09 f7 4b 13 7d …	

Mandatory fields

Field	**Meaning**
Version	Which version of X.509
Serial Number	Unique for each certificate
Signature	Algorithm used to sign the certificate
Issuer	x.500 name of the CA
Validity	Dates on which the validity starts and terminates
Subject	x.500 name of certificate holder
Public Key	Public key of the subject

UNIVERSITY OF Southampton

Optional fields

Field	**Meaning**
Issuer Unique Identifier	CA Identifier
Subject Unique Identifier	Subject Identifier
Extensions	Each extension includes an identifier, a criticality flag, and extension value
Key Usage	Security Services that can be implemented using the key e.g data encryption
Subject Alternative Name	Other names for the subject
CRL Distribution Point	Pointer to the CRL related to the certificate

What is PKI

- Public Key Infrastructure (PKI)

- The set of **hardware**, **software**, **people**, **processes**, and **policies**

- together, using the technology of asymmetric cryptography, facilitate the creation of a **verifiable association between a public key** and the identity of the **holder of the corresponding private key** (the private component of that pair)

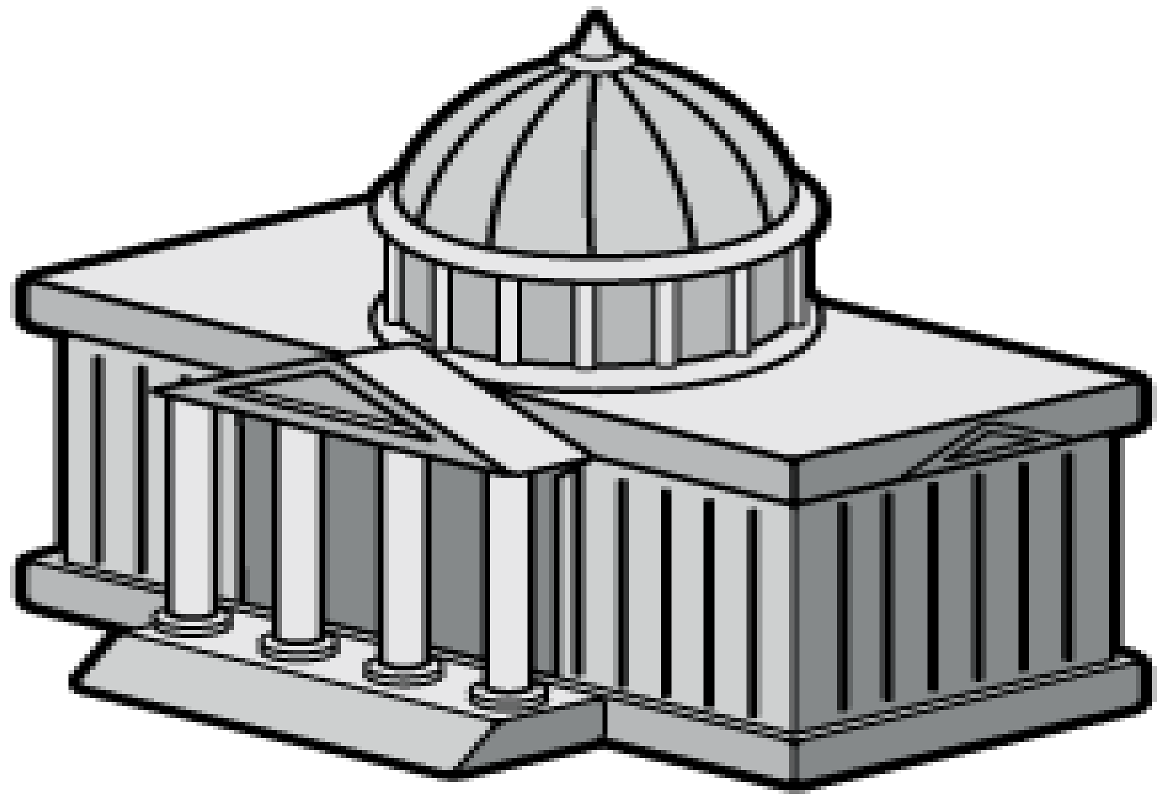
- for uses such as **authenticating** the identity of a specific entity, ensuring the **integrity** of information, providing support for **nonrepudiation**, and establishing **encrypted** communications

PKI Key Players

- Certification Authorities (CA)
- Registration Authorities (RA)
- PKI Repositories
- PKI Users

Certification Authority (CA)

- Responsible for issuing, revoking, and distributing public key certificates
- Often a trusted-third party organization. Examples:
  - VeriSign
  - DigiCert
  - Comodo

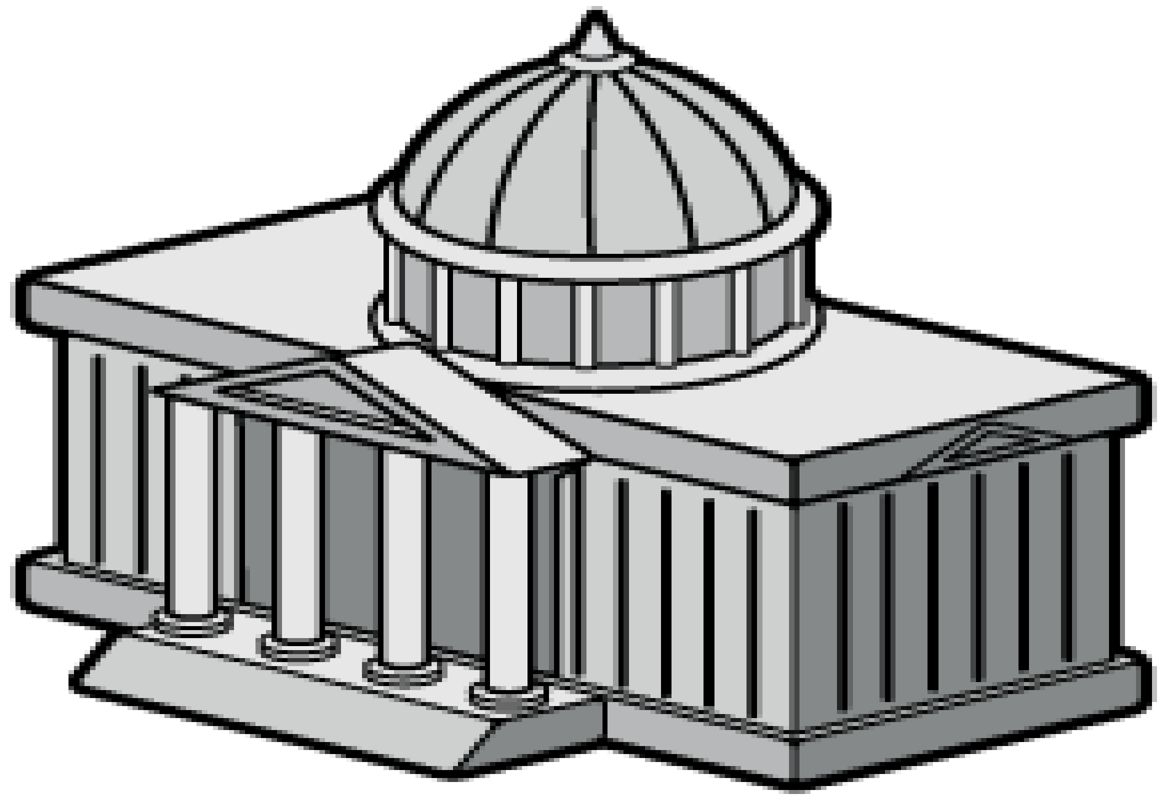


- Important to protect CA private key

- **QUESTION:** ***Who signs the certificates?***

Certification Authority (CA)

- Responsible for issuing, revoking, and distributing public key certificates
- Often a trusted-third party organization. Examples:
  - VeriSign
  - DigiCert
  - Comodo



- Important to protect CA private key
- Certificates are signed with **CA private key**
  - Thus, everybody can check the authenticity of the certificates
  - By using the CA public key (as for checking a digital signature)

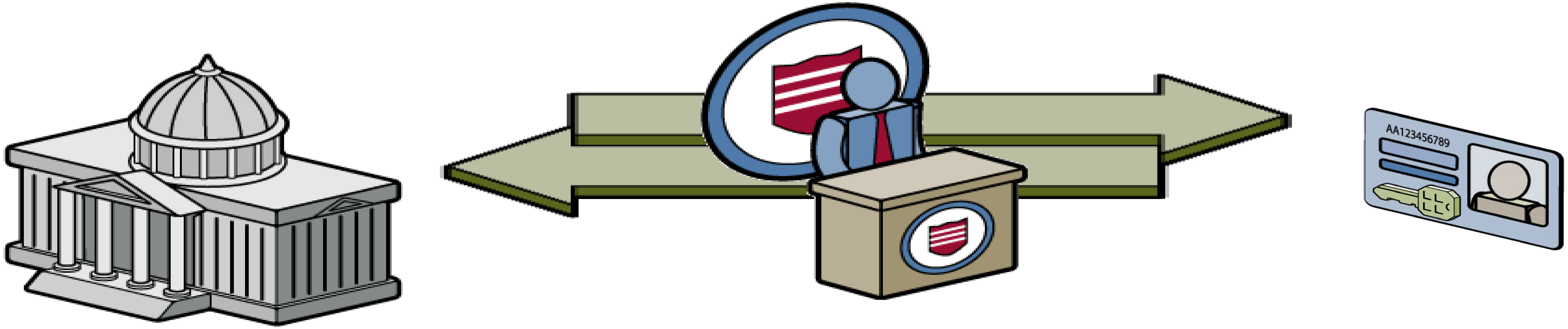
PKI Repositories

- Means of storing and distributing x.509 certificates and certificate revocation lists (CRLs) and managing updates to certificates

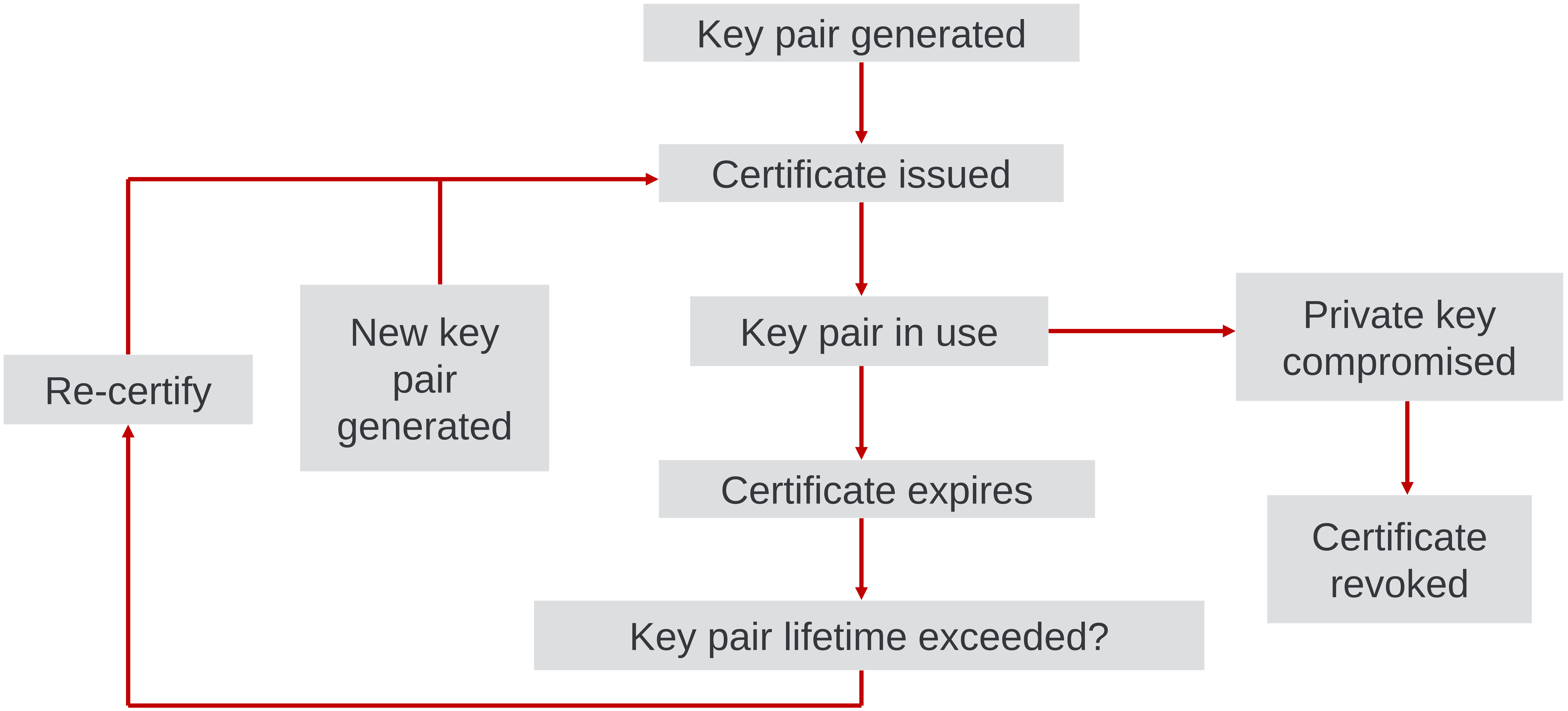
- Allow relying parties to retrieve x.509 certificates and CRLs

Registration Authority (RA)

- Performs functions for CA but does not issue certificates directly
- Verifies certificate contents for the CA before issuance of a certificate
- Known to the CA by RA name and public key



Certificate Life Cycle



Certificate Issuance

1. RA verifies subject information

2. Generate Public – Private Key Pair
    a. Generated by the subject
    b. Generated by the CA

3. CA issues the certificate

Relying party wants to verify a signature

1. Fetch certificate
2. Fetch certificate revocation list (CRL)
3. Check certificate against CRL
   - Is the certificate still valid or is revoked?
4. Check signature using certificate

Certificate Revocation

- Reasons to revoke a certificate
  - Compromised Private Key
  - Expiration
  - Human Resources Reason
  - Company changes name, physical address, DNS

Certificate Revocation Lists

- It is a list of **certificates** which are **no longer valid**
- **Published** regularly by the CA in the **PKI repository**
- But also sent to any relying party who has subscribed to it
- Standard format: x509 certification revocation list
- Problems
  - Not issued frequently enough to be effective against an attacker
  - Expensive to distribute
  - Vulnerable to simple DOS attacks

Mandatory fields

Field	Meaning
Version	Which version of X.509
Signature	Algorithm used to sign the list
This Update	Issue date of CRL
Next Update	Next Issue date of CRL
Revoked Certificates	List of Revoked Certificates

Attacks to PKI: Comodo Case

- How the attack was conducted
  - The attack was conducted on 15 March 2011
  - The attacker **compromised** an **RA** user **account**
  - The attacker used the account to issue 9 certificates for 7 different domains including
    - *mail.google.com*
    - *www.google.com*
    - *login.yahoo.com*
    - *login.skype.com*

- Possible Consequences
  - The attacker could have used the certificates to craft fake web pages

- Comodo Response
  - Certificates were immediately revoked
  - Principal browsers and domain owners were notified about the attack
  - The RA account was suspended

- **Comodo is still in business!**

Attacks to PKI: DigiNotar Case

- How the attack was conducted
  - **DigiNotar** network breached on 17th June 2011
  - Attacker gained control of **all CA servers** on 29th June 2011
  - First rogue certificate created on 10th July 2011
  - 124 rogue certificates created on 18th July 2011
  - Other 124 rogue certificates created on 20th July 2011

Attacks to PKI: DigiNotar Case

- Consequences
  - 531 rouge certificates were created
  - 1 rouge certificate for google.com domain was used to conduct a large scale Man-In-The-Middle attack against 300,000 gmail users located in Iran

- DigiNotar Response
  - Certificates were revoked

- **DigiNotar is bankcrupted!**

Recommended Readings

- Internet X.509 Public Key Infrastructure: Certificate Path Building
  - http://tools.ietf.org/html/rfc4158
- Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile
  - http://tools.ietf.org/html/rfc5280
- Black Tulip – Report of the investigation into DigiNotar Certificate Authority Breach