CSC429 – Computer Security

LECTURE 4
MODERN CRYPTOGRAPHY –
ASYMMETRIC CRYPTOGRAPHY

Mohammed H. Almeshekah, PhD
meshekah@ksu.edu.sa

Quiz 1

- From a security perspective, rather than an efficiency perspective, which of the following statements about the block size of a block cipher is most accurate?
  1. The bigger the block size the better.
  2. The block size should neither be too small nor too large.
  3. The block size should neither be too small nor too large, and should be a multiple of 8.
  4. The block size is unimportant.

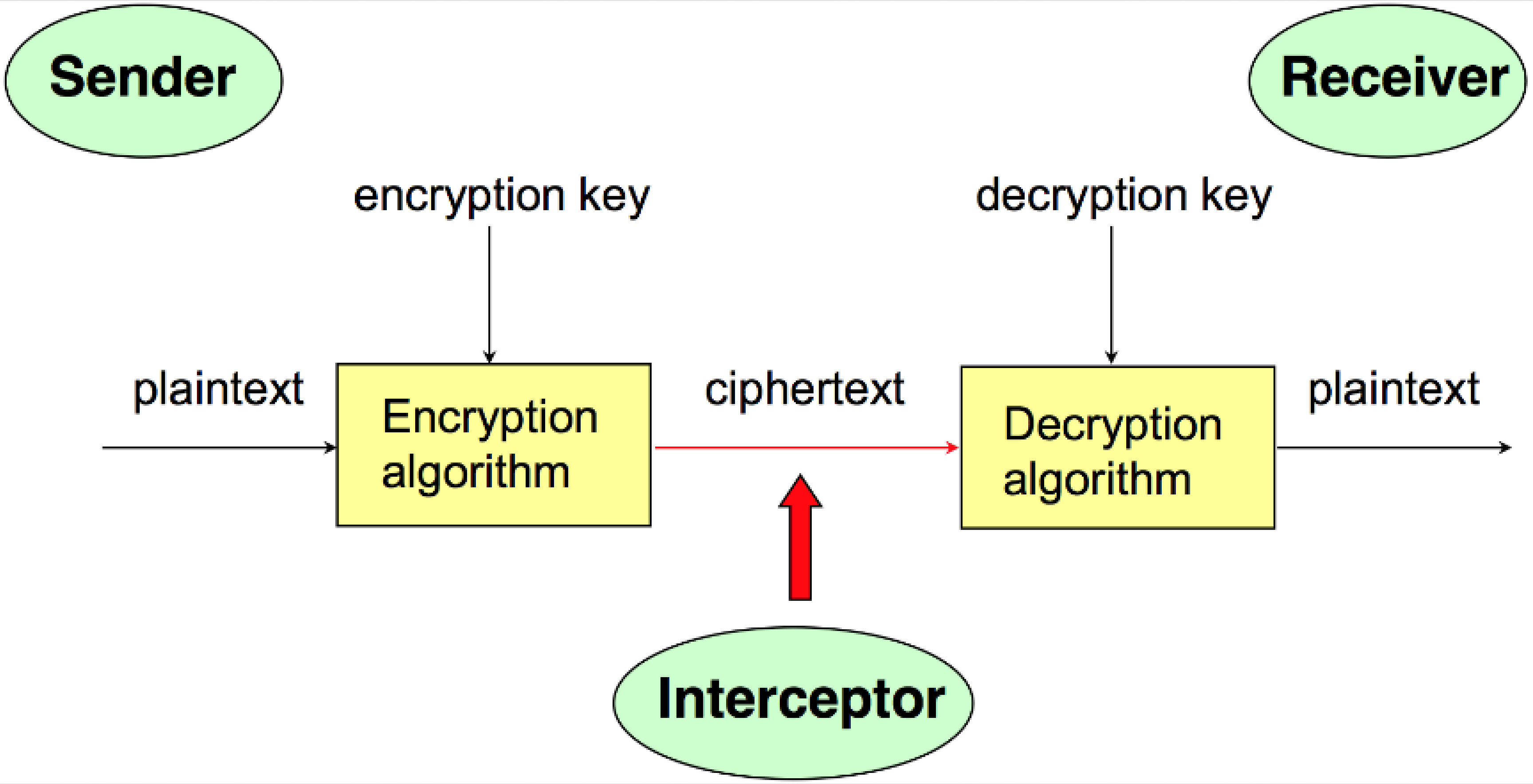
Quiz 2

- Which of the following is most accurate?
  1. Key management for stream ciphers is **easier than** for block ciphers, because the plaintext is not actually encrypted directly with the key
  2. Key management for block ciphers is **less critical when using CBC mode**, since the security of the ciphertext depends on the preceding ciphertext as well as the key
  3. Key management for stream ciphers **is more difficult** than for block ciphers because the key needs to kept synchronized at each end of the communication link
  4. Key management is **roughly of the same level** of difficulty for stream ciphers and block ciphers

Modern Cryptography

Asymmetric Cryptography

Symmetric vs. Asymmetric



Public-Key Encryption

- Each party has a PAIR (K, K^{-1}) of keys:
  - K is the **public** key, and used for encryption
  - K^{-1} is the **private** key, and used for decryption
  - Satisfies $\mathbf{D}_{K^{-1}}[\mathbf{E}_K[M]] = M$
- Knowing the public-key K, it is computationally infeasible to compute the private key K^{-1}
- The public-key K may be made publicly available, e.g., in a publicly available directory
  - Many can encrypt, only one can decrypt.

Public-Key Schemes

- Almost all public-key encryption algorithms use either number theory and modular arithmetic, or elliptic curves
- RSA
  - based on the hardness of factoring large numbers.
- El Gamal
  - Based on the hardness of solving discrete logarithm.

RSA Scheme

- Invented in **1978** by Ron **R**ivest, Adi **S**hamir and Leonard **A**dleman.

- **Key generation:**
  - Select 2 large prime numbers of about the same size, p and q.
  - Compute $n = pq$, and $\Phi(n) = (q-1)(p-1)$
  - Select e, $[1<e<\Phi(n)]$, s.t. $[\gcd(e, \Phi(n)) = 1]$.
    - Typically e=3 or e=65537
  - Compute d, $[1<d<\Phi(n)]$ s.t. $[ed \equiv 1 \bmod \Phi(n)]$.
    - Knowing $\Phi(n)$, d easy to compute.

- Keys:
  - **Public key: (e, n)**
  - **Private key: d**

RSA Encryption and Decryption

- **Encryption**
  - Given a message M, use public key (e, n) and,
  - Compute $C = M^e \bmod n$.

- **Decryption**
  - Given a ciphertext C, use private key (d)
  - Compute $C^d \bmod n = (M^e \bmod n)^d \bmod n = M^{ed} \bmod n = M$

- Security:
  - From n, difficult to figure out p,q
  - From (n,e), difficult to figure d.
  - From (n,e) and C, difficult to figure out M s.t. $C = M^e$

RSA Security

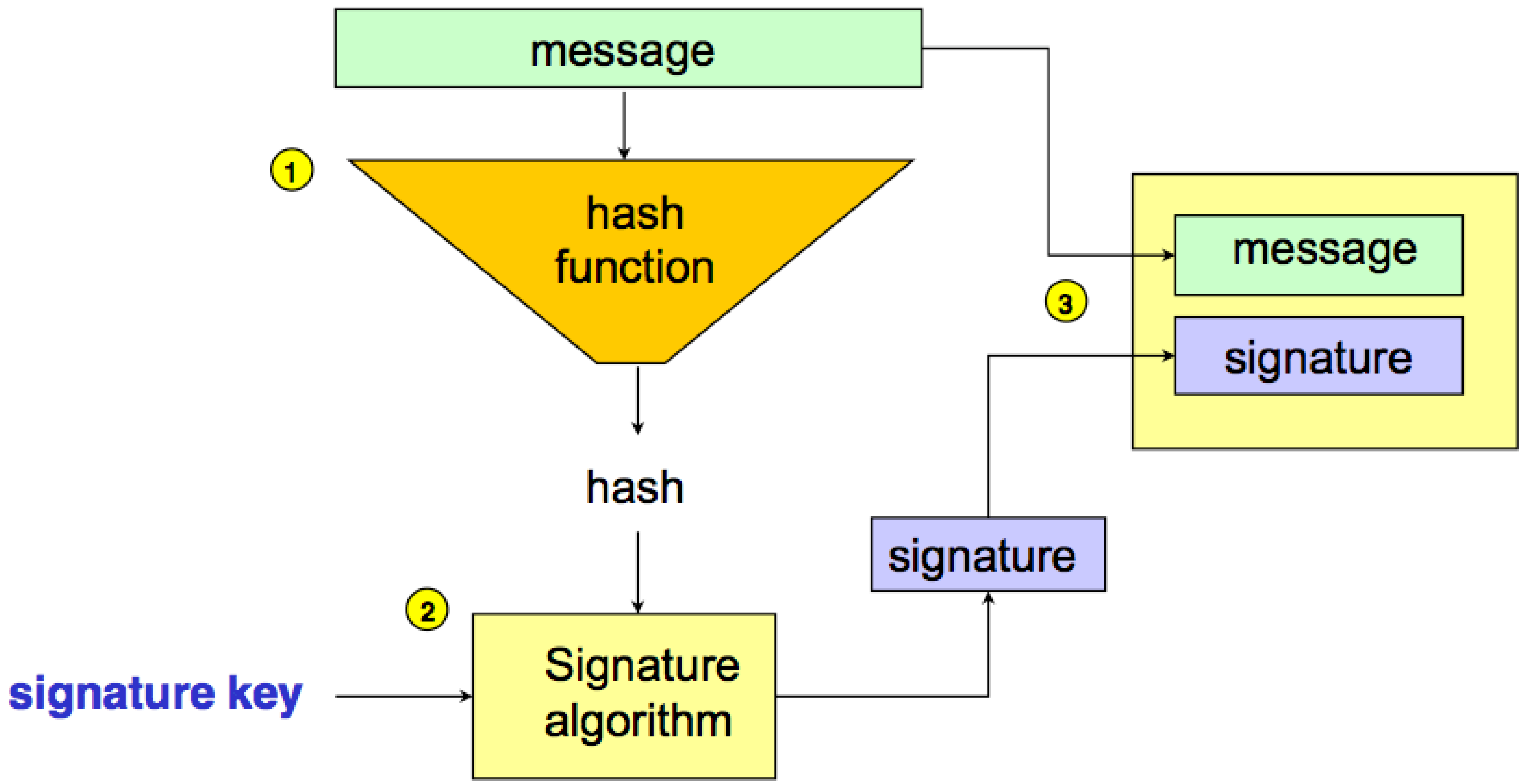
- The length of n=pq reflects the strength
  - 700-bit n factored in 2007
  - 768 bit factored in 2009
- 1024-bit for minimal level of security today
  - likely to be breakable in near future
  - Minimal 2048-bits recommended for current usage
  - NIST suggests 15360-bit RSA keys are equivalent in strength to 256-bit symmetric cipher.
- This is textbook RSA:
  - Not secure for real-life applications.
  - It is important to implement RSA according to standards:
    - PKCS#1v2 / RSA-OAEP.

Non-Repudiation – Digital Signatures

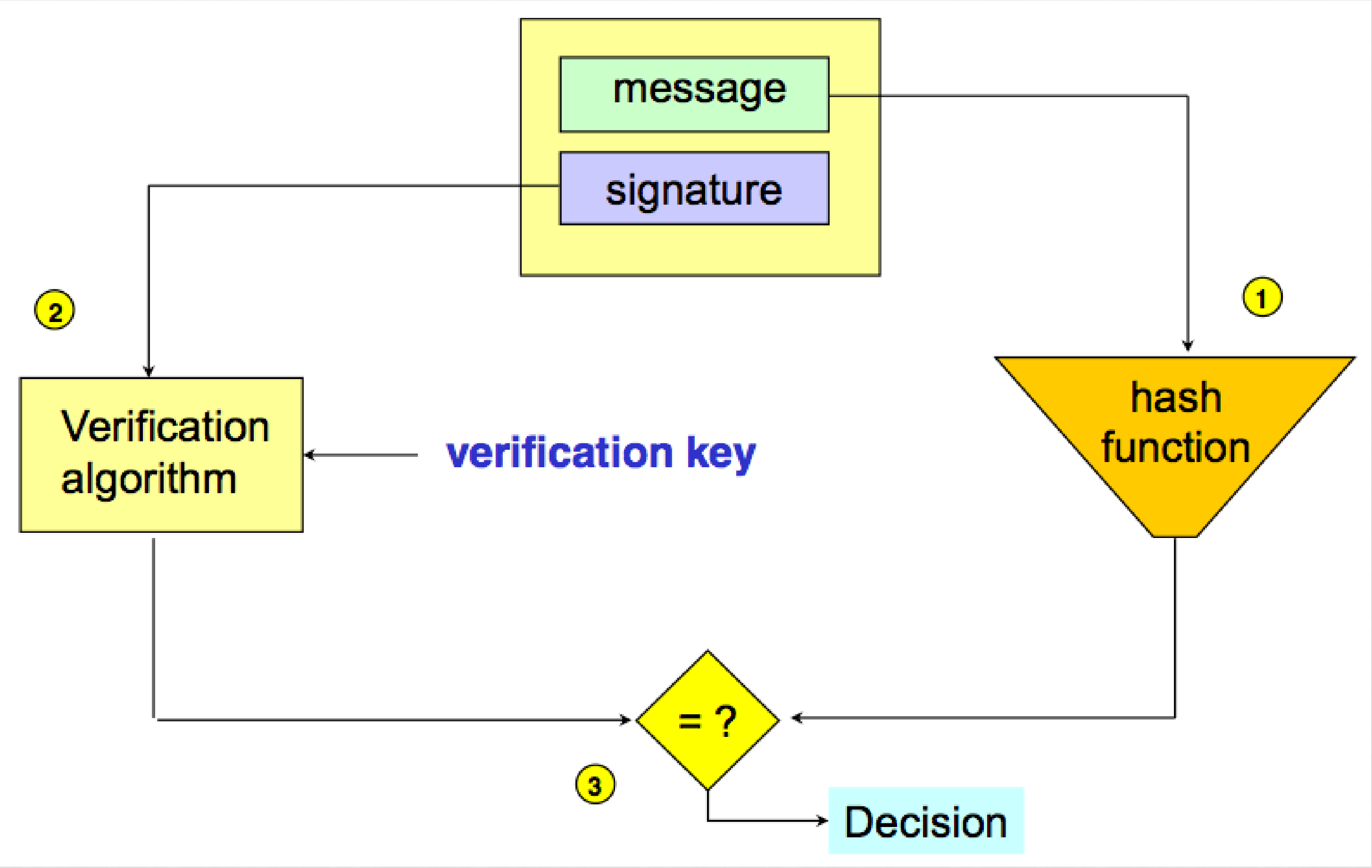
- Imagine the physical world situation of signing a contract.
  - Can cryptography provide the same service?

- Does MAC provide non-repudiation?

Digital Signature – Signing



Digital Signature – Verification



RSA Signature Scheme

- Key generation is the same as before.

- **Signing message M**
  - Use signing (private) key (d)
  - Compute $S = M^d \bmod n$

- **Verifying signature S**
  - Use verification (public) key (e, n)
  - Compute $S^e \bmod n = (M^d \bmod n)^e \bmod n = M$

- Note: in practice, a hash of the message is signed and not the message itself.

Summary of Cryptography

	Symmetric Algorithms	**Asymmetric Algorithms**
Confidentiality	Stream Ciphers Block Ciphers Encryption Modes	RSA ElGamal
Integrity	Message Authentication Code (MAC)	Digital Signatures

Cryptographic Keys

Establishment & Management

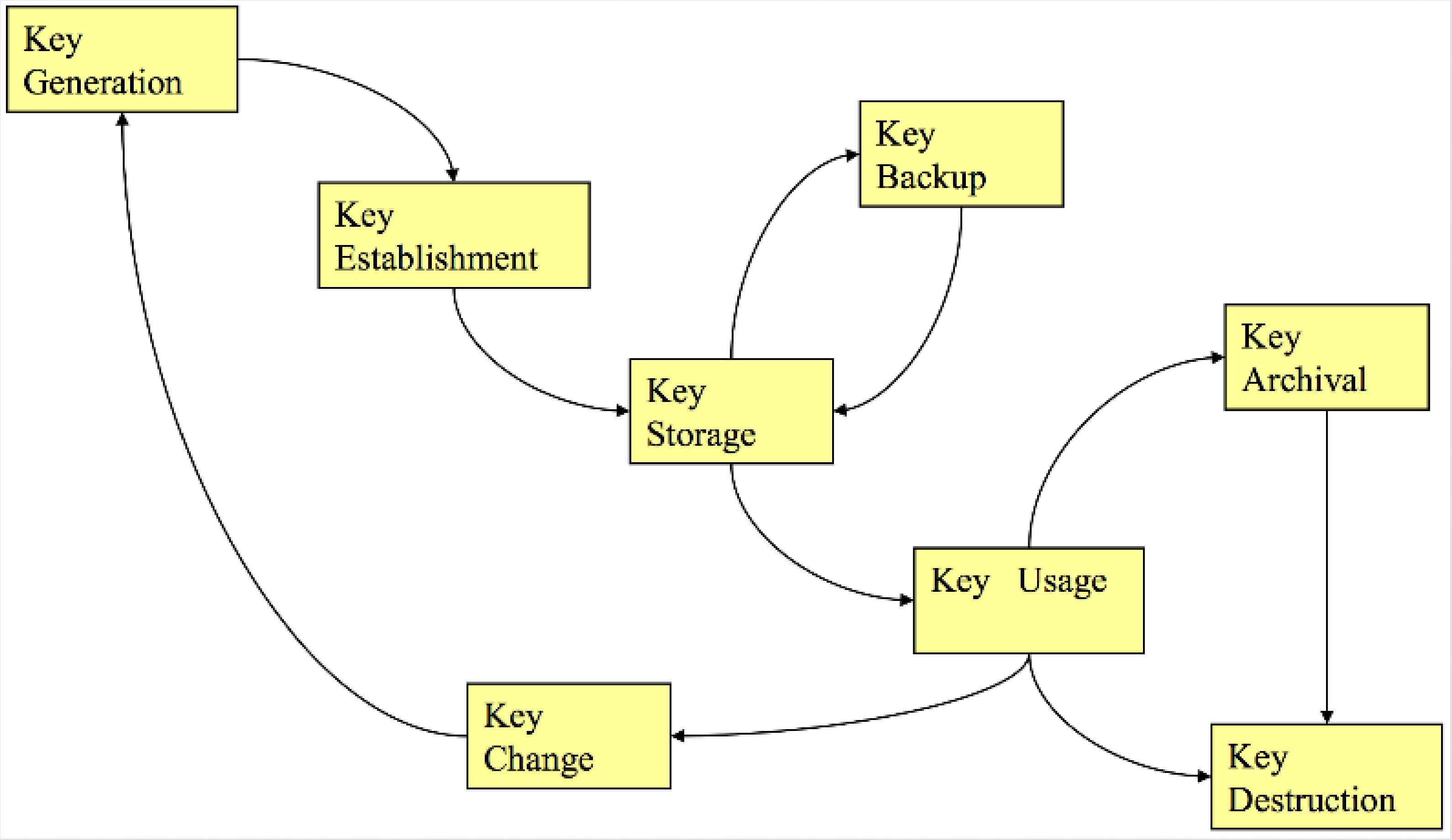
Terminology

- Entity authentication
  - The assurance that a given entity is involved and currently active in a communication session
- Mutual Entity Authentication
  - Entity authentication for both parties.
- Key Agreement Protocol
  - Is a key establishment protocol that takes place directly between the entities who will share the key.
- Key Distribution Protocol
  - Is a key establishment protocol where the key is established with the help of a trusted third party (who normally generates the key).

Key Management Requirement

- The main requirement for the management of keys used with symmetric algorithms is that the keys remain secret.
- The main requirements for the management of keys used with public key algorithms are that the private key remains secret and the public key is authentic.
- There is no single right answer!
  - There are a number of standards which can be helpful .

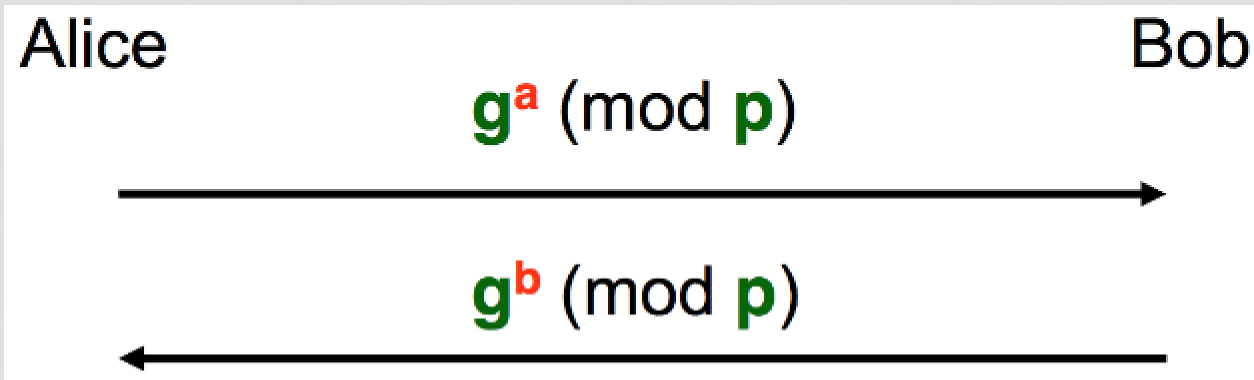
Life-time of Cryptographic Keys



Diffie-Hellman Key Agreement

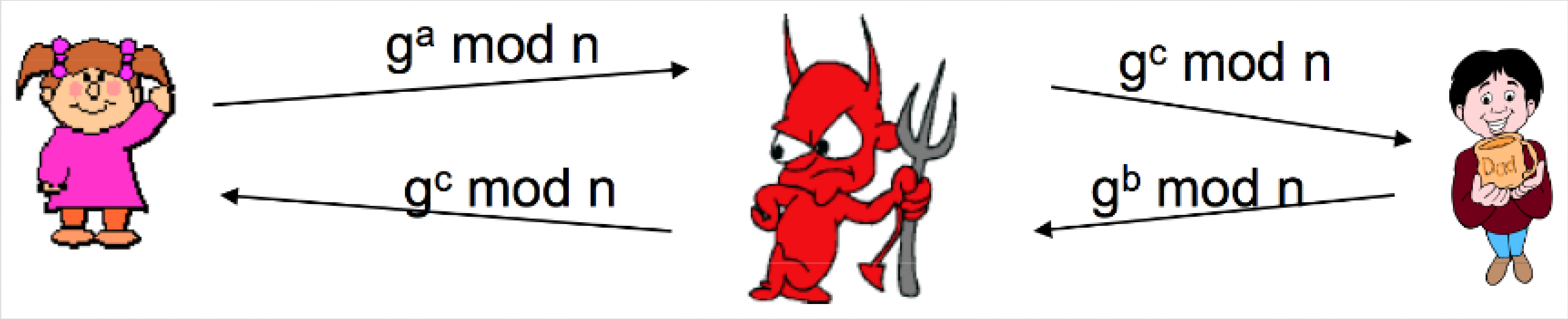
- The Diffie–Hellman (DH) key agreement protocol was first defined in their seminal paper in 1976.
- DH key agreement is a protocol for exchanging public (i.e. non- secret) information to obtain a shared secret.
- DH key agreement has the following important properties:
  - The resulting shared secret cannot be computed by either of the parties without the cooperation of the other.
  - A third party observing all the messages transmitted during DH key exchange cannot deduce the resulting shared secret at the end of the protocol.

DHKE



1. Alice generates a private random value a, calculates $g^a \pmod{p}$ and sends it to Bob. Meanwhile Bob generates a private random value b, calculates $g^b \pmod{p}$ and sends it to Alice.
2. Alice takes g^b and her private random value a to compute $(g^b)^a = g^{ab} \pmod{p}$.
3. Bob takes g^a and his private random value b to compute $(g^a)^b = g^{ab} \pmod{p}$.
4. Alice and Bob adopt $g^{ab} \pmod{p}$ as the shared secret.

Man-in-the-Middle Attack - DHKE



- Station-to-station protocol addresses the MITM attack.

Key Agreement – Symmetric Algorithms

- For a group of N parties, every pair needs to share a different key.
  - What is the total number of keys?

- Solution:
  - Need a key distribution protocol.
  - Uses a central authority, a.k.a., Trusted Third Party (TTP)
  - Every party shares a key with a central server.

Needham-Schroeder Protocol

- Parties:
  - Users A and B.
  - Trusted server T
- Setup:
  - A and T share K_{AT},
  - B and T share K_{BT}
- Goals:
  - Mutual entity authentication between A and B
  - key establishment
- Messages:
  - $A \rightarrow T$: A, B, N_A (1)
  - $A \leftarrow T$: $E[K_{AT}]$ (N_A, B, k, $E[K_{BT}](k,A)$) (2)
  - $A \rightarrow B$: $E[K_{BT}]$ (k, A) (3)
  - $A \leftarrow B$: $E[k]$ (N_B) (4)
  - $A \rightarrow B$: $E[k]$ (N_B-1) (5)

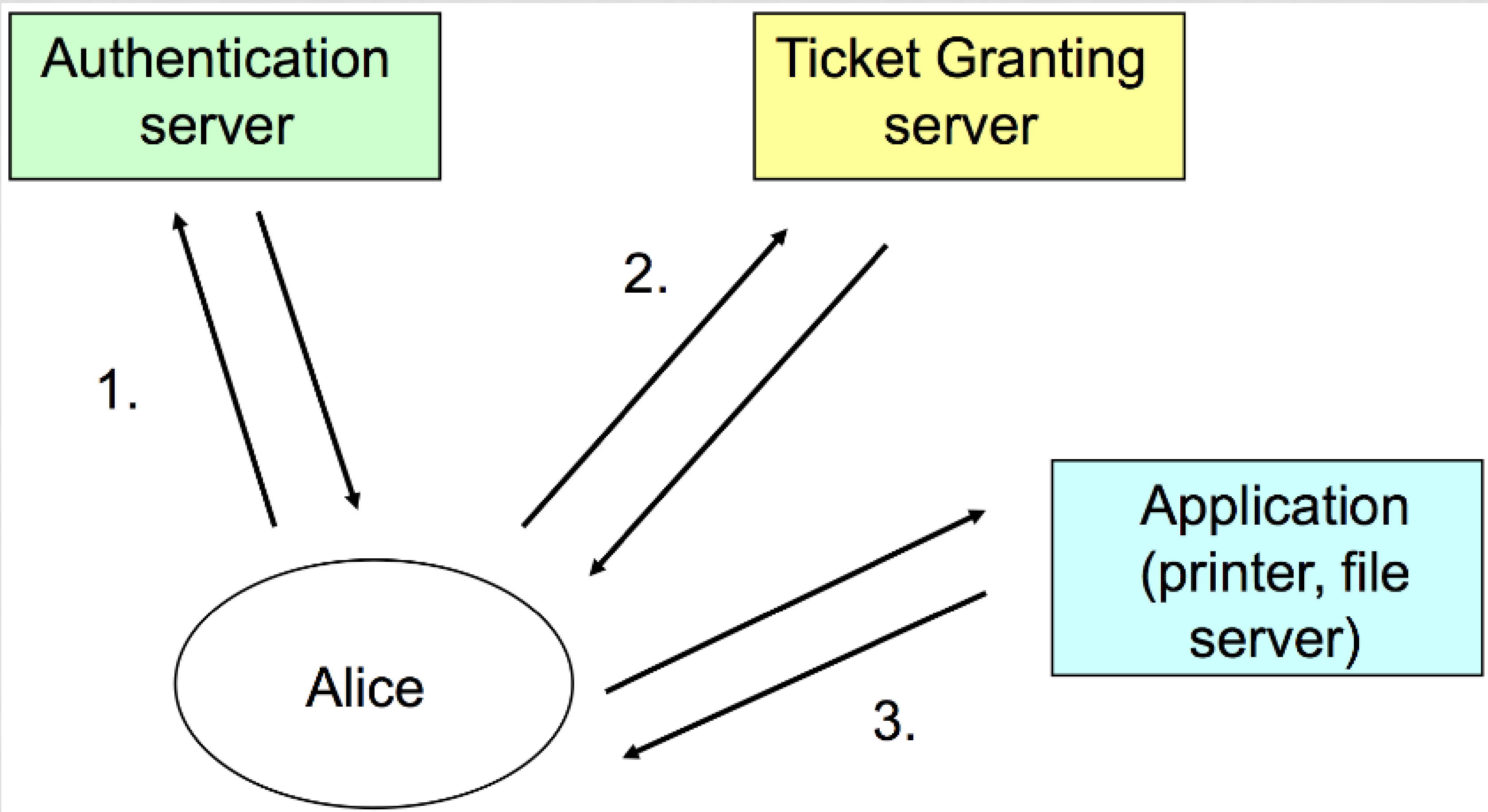
Kerberos

- Implement the idea of Needham-Schroeder protocol.
- Provides authentication and secure communication
- Developed at MIT:
  - http://web.mit.edu/kerberos/www
- Used in many systems, e.g., Windows 2000 and later as default authentication protocol.

Kerberos – Overview

- One issue of Needham-Schroeder
  - Needs the key each time a client talks with a service
- Principle:
  - Alice uses her password to sign on once a day

Kerberos Protocol



Kerberos Protocol – 2

1. Alice gets a "daily key" K_A from the authentication server
   - Based on Alice's long term secret (password)
   - K_A is stored on Alice's machine and deleted at the end of the day
2. Alice uses K_A to get application key K from the ticket granting server.
3. Alice establishes a secure link with the application using K.

Kerberos Drawbacks

- Single point of failure:
  - requires online Trusted Third Party: Kerberos server.

- Useful primarily inside an organization
  - Does it scale to Internet?

Certificate Authority (CA)

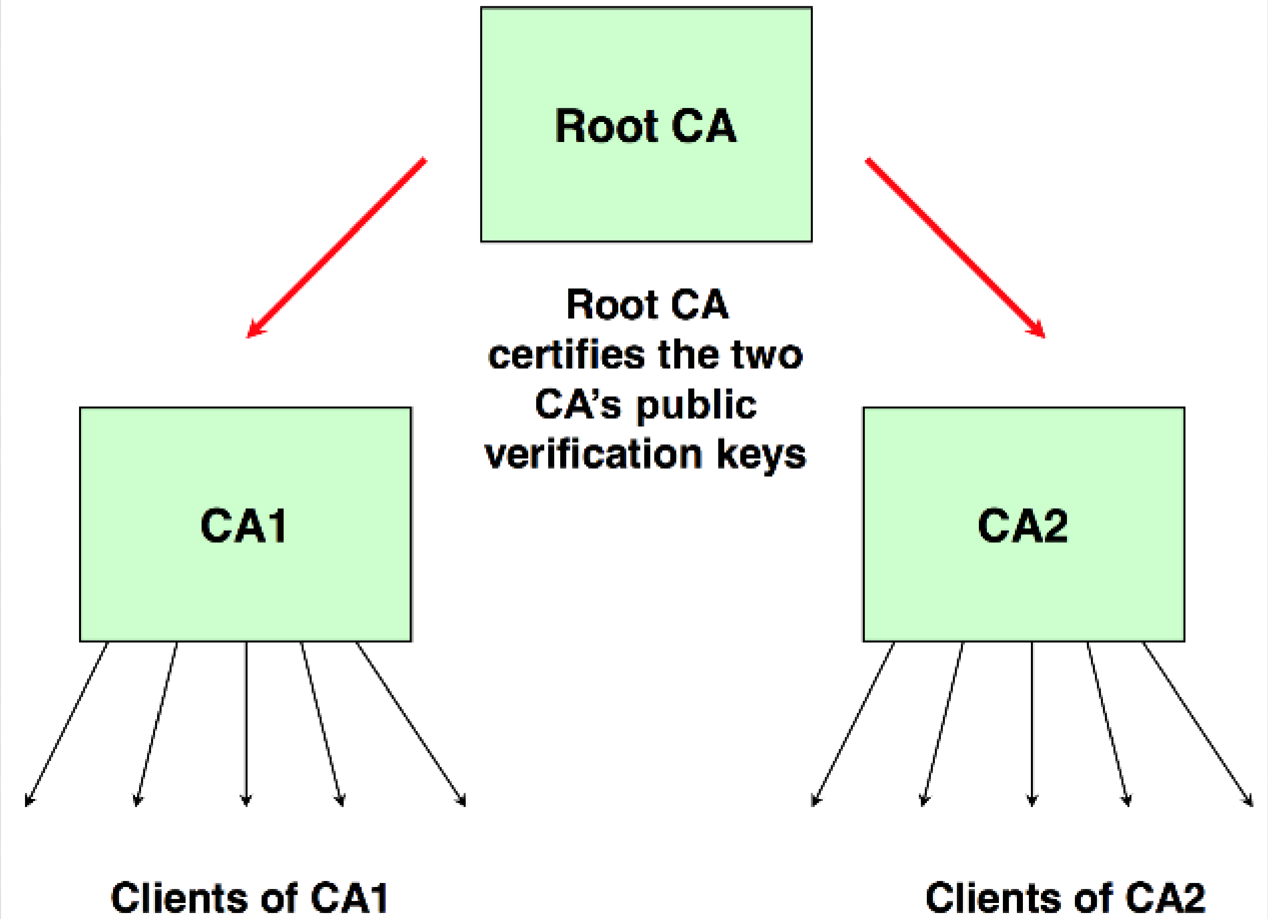
- A CA is a trusted third party, whose main purpose is to certify public keys generated by users of the system.

- A certificate is a data structure containing information about the:
  - owner of the key,
  - algorithm details,
  - key validity dates and,
  - the public key in question.

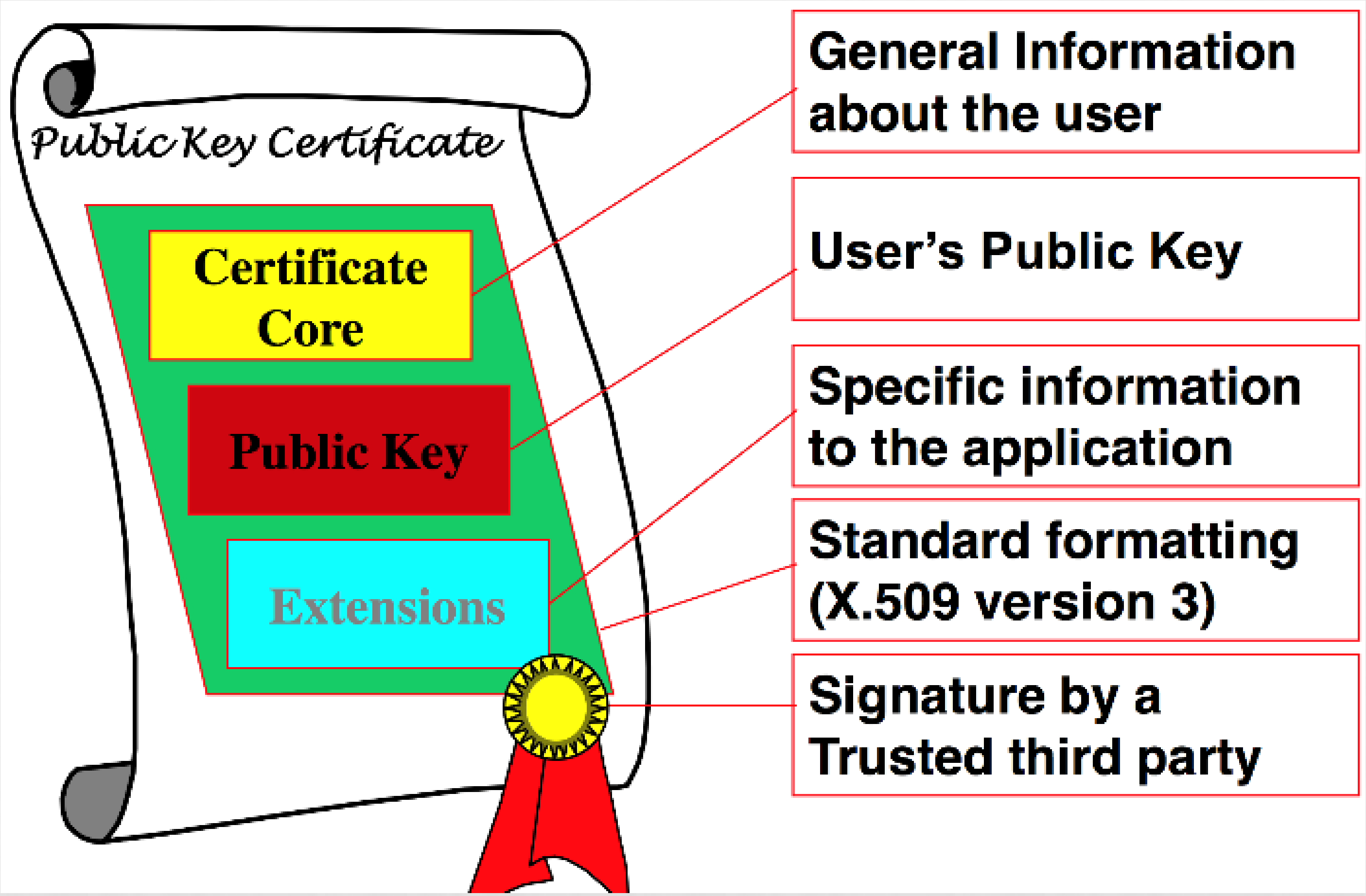
- The data is hashed and then signed using the CA private key.
  - Certificates can be validated by any party with access to the CA public key.

- The most common certificate format is defined in the X.509 standard (version 3).

Certificate Hierarchies



Digital Certificate



Certificate Revocation

- We must consider how to handle certificates that need to be "withdrawn" before their expiry date.s

1. Certificate Revocation List (or CRLs) – A lists of certificates that have been revoked.
   - CRLs need to be maintained carefully, with clear indications of how often they are updated.
   - CRLs need to be signed by the CA and be made available to users as easily as possible.
2. Online Certificate Status Protocol (OCSP)
   - An online database containing the status of certificates issued by the CA.

Next Lecture

- Software Security

- Readings for next lecture:
  - Smashing The Stack For Fun And Profit (can be found in the course resources page)