

Digital Signatures Cryptographic Key Management and Distribution

Prepared by Dr. Ayman Khallel

IT373

Information Security

Lecture 9



Course content

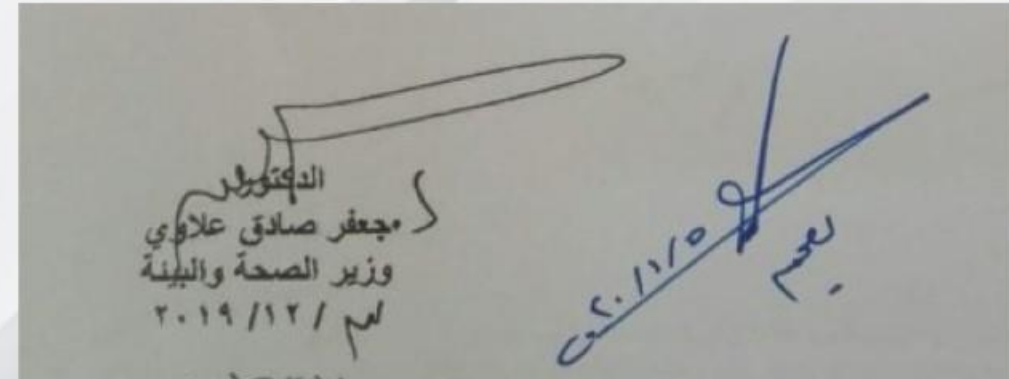
1. Overview of security concepts and definitions
2. Symmetric cryptography; modern encryption algorithms (DES, AES)
3. Modern data encryption standards
4. Block cipher cryptanalysis and usage
5. Modern stream ciphers
6. Asymmetric cryptography; public key cryptography,
7. Cryptographic Hash Functions
8. Message authentication Code
9. Digital signature and authentication protocols
10. Electronic mail security, IP security, web security.

Agenda

- Introduction to Digital Signatures
- Requirements of Digital Signatures
- Direct Digital Signatures
- Elliptic Curve Digital Signature Algorithm (ECDSA)
- RSA-PSS (Probabilistic Signature Scheme)
- Key Management and Distribution
- Key Distribution Techniques
- X.509 Certificates
- Summary and Q&A

Digital Signature

- A **digital signature** is a mathematical scheme for verifying the authenticity of digital messages or documents.
- A valid digital signature on a message gives a recipient confidence that the message came from a sender known to the recipient.
- Digital signatures are a standard element of most cryptographic protocol suites
- commonly used for software distribution, financial transactions, contract management software, and in other cases where it is important to detect forgery or tampering.



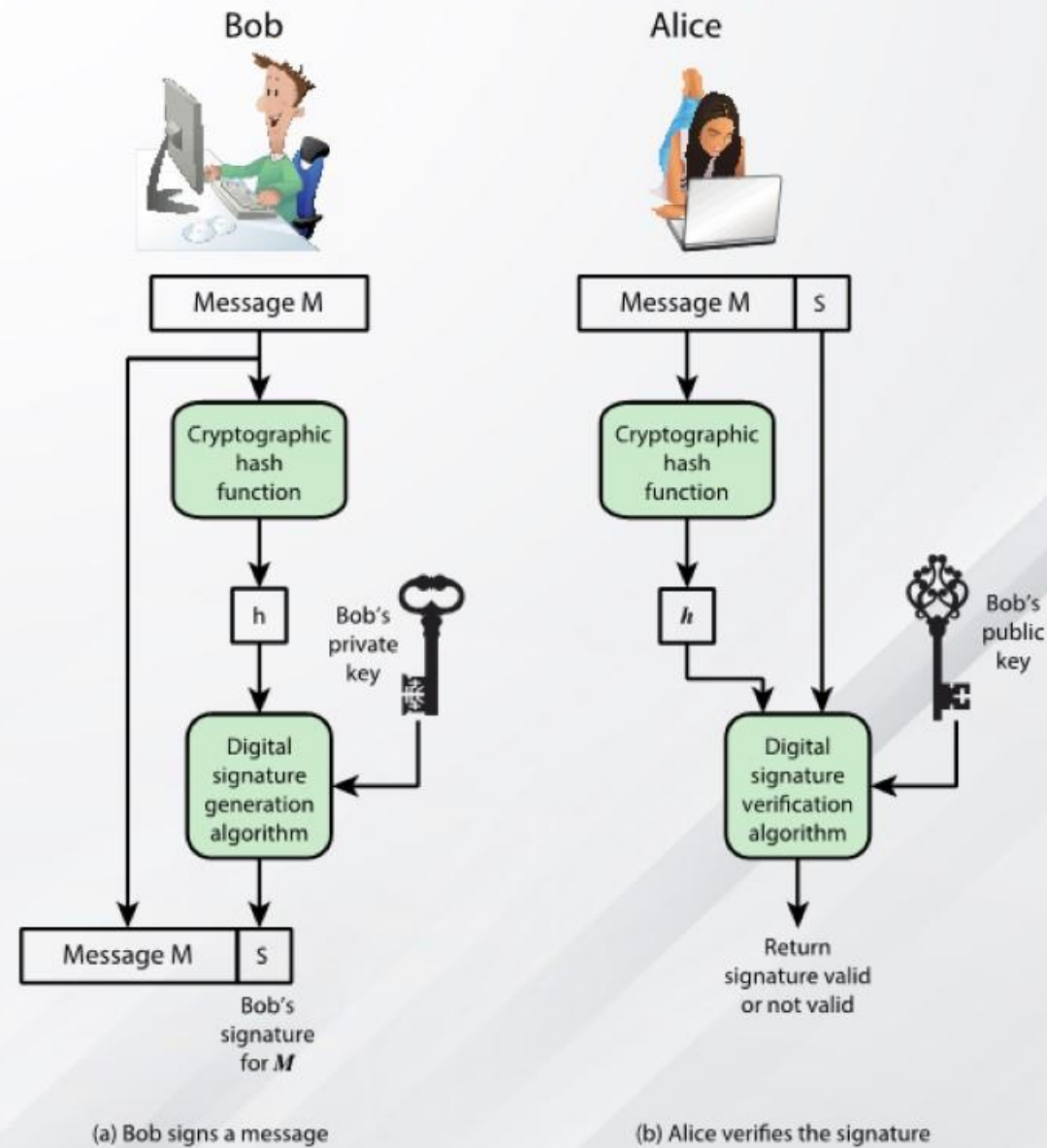
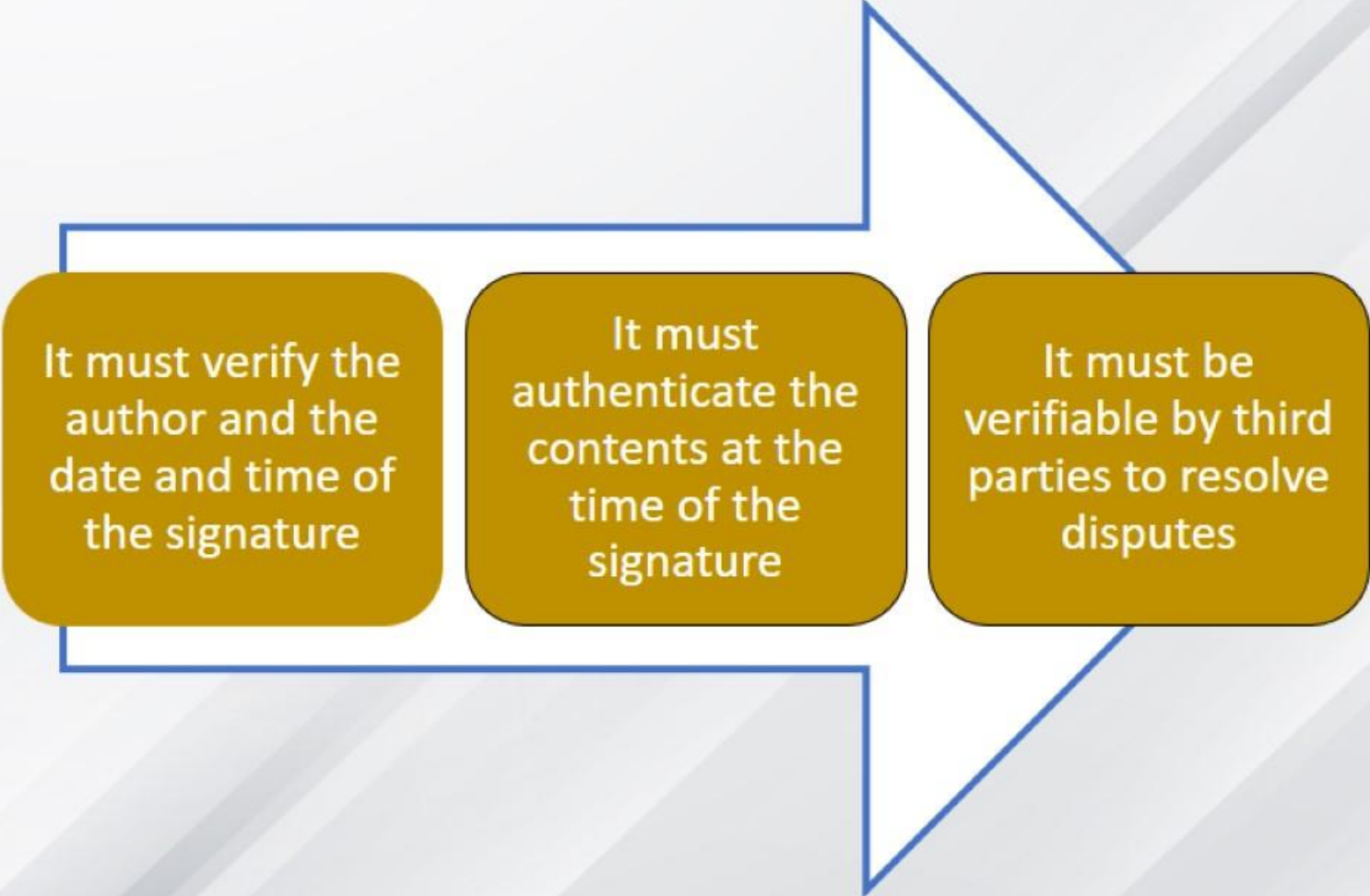


Figure 13.1 Simplified Depiction of Essential Elements of Digital Signature Process



Digital Signature Properties

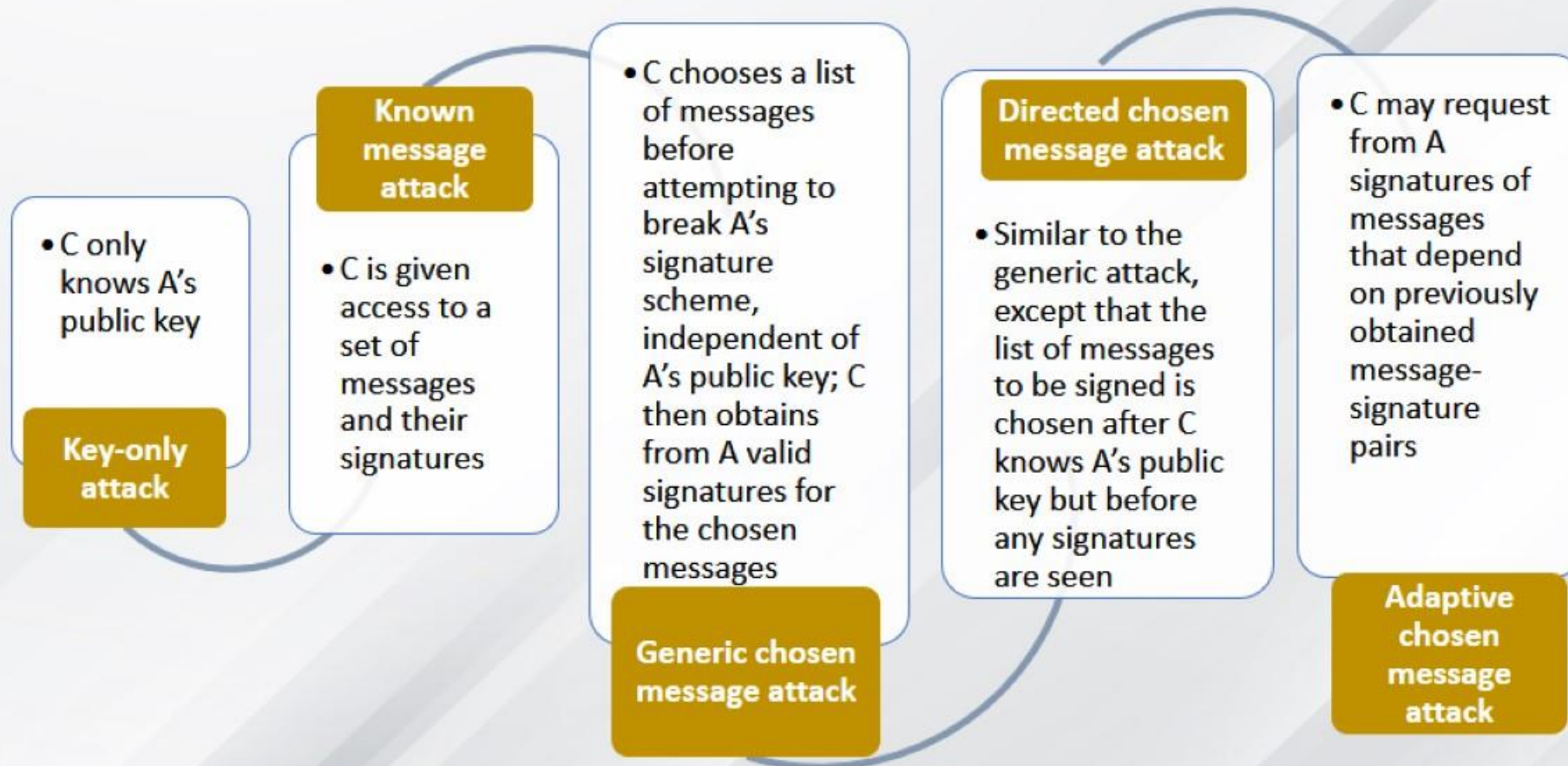


It must verify the author and the date and time of the signature

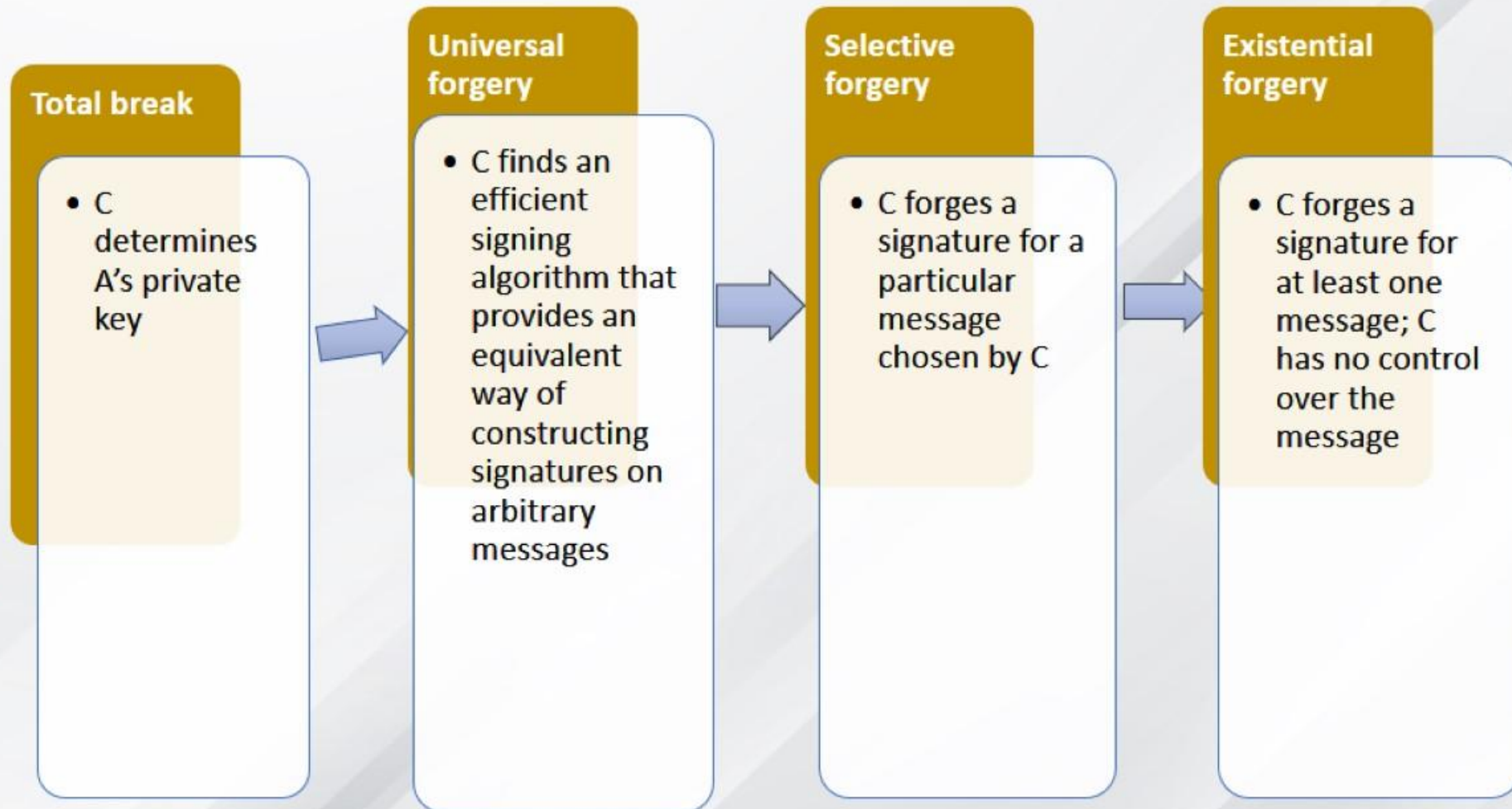
It must authenticate the contents at the time of the signature

It must be verifiable by third parties to resolve disputes

Attacks



Forgeries



Digital Signature Requirements

- The signature must be a bit pattern that depends on the message being signed
- The signature must use some information unique to the sender to prevent both forgery and denial
- It must be relatively easy to produce the digital signature
- It must be relatively easy to recognize and verify the digital signature
- It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message
- It must be practical to retain a copy of the digital signature in storage



Direct Digital Signature

Refers to a digital signature scheme that involves only the communicating parties

It is assumed that the destination knows the public key of the source

Confidentiality can be provided by encrypting the entire message plus signature with a shared secret key

It is important to perform the signature function first and then an outer confidentiality function

In case of dispute some third party must view the message and its signature

The validity of the scheme depends on the security of the sender's private key

If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and that someone else forged his or her signature

One way to thwart or at least weaken this ploy is to require every signed message to include a timestamp and to require prompt reporting of compromised keys to a central authority



Elliptic Curve Digital Signature Algorithm (ECDSA)

- A cryptographic algorithm used for digital signatures, providing data integrity and authentication.
- **Key Advantage:**
 - Offers high security with smaller key sizes compared to other algorithms like RSA.
 - Efficient in terms of computation and storage.



ECDSA Key Concepts

- **Elliptic Curve Cryptography (ECC):**
 - Based on the mathematics of elliptic curves over finite fields.
 - Uses equations like $y^2 = x^3 + ax + b$.
- **Digital Signature Process:**
 - **Key Generation:**
 - Private Key: Randomly chosen.
 - Public Key: Derived from the private key using an elliptic curve multiplication.
 - **Signing:**
 - Message is hashed (e.g., SHA-256).
 - Signature (r,s) is generated using the private key, hash, and random number.
 - **Verification:**
 - Verifies r and s against the public key and hash to confirm authenticity.



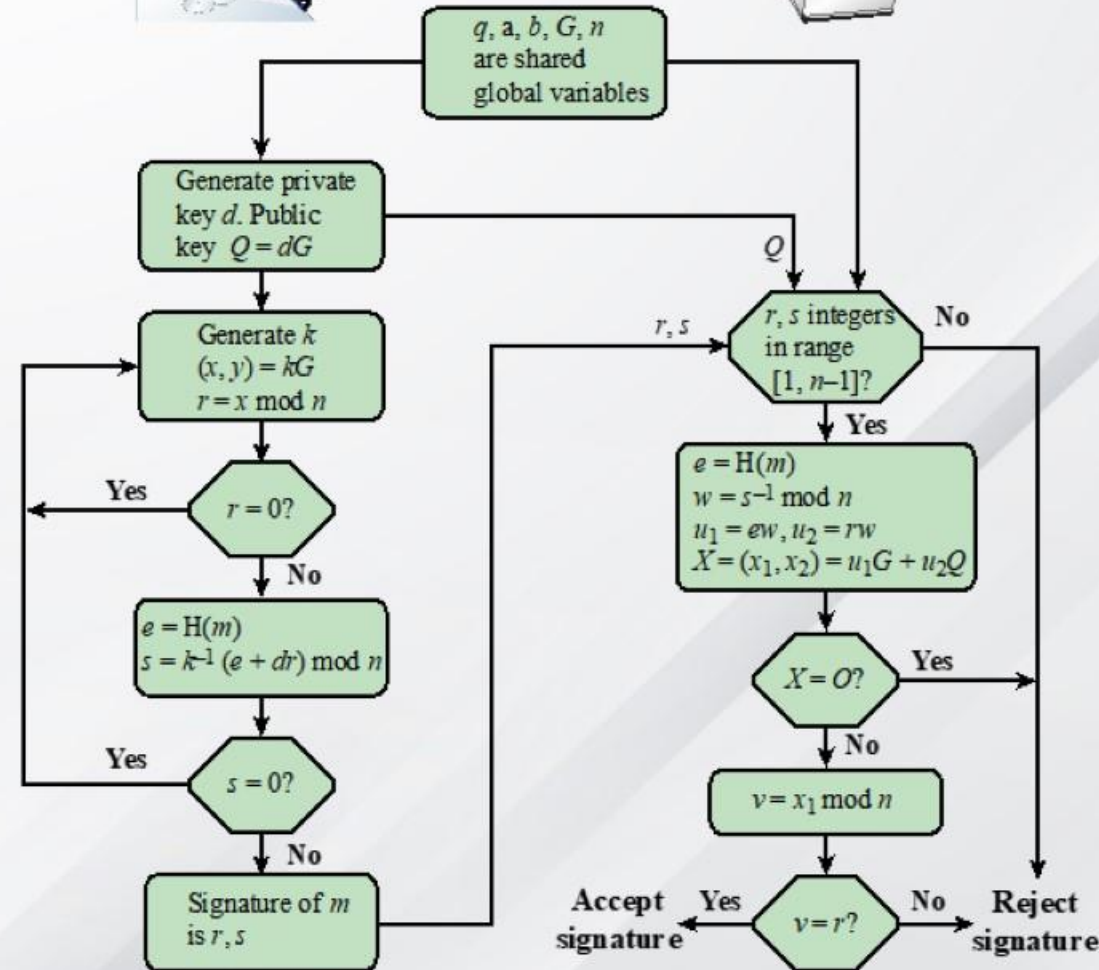


Figure 13.5 ECDSA Signing and Verifying

RSA-PSS

- RSA Probabilistic Signature Scheme
- Included in the 2009 version of FIPS 186
- Latest of the RSA schemes and the one that RSA Laboratories recommends as the most secure of the RSA schemes
- For all schemes developed prior to PSS it has not been possible to develop a mathematical proof that the signature scheme is as secure as the underlying RSA encryption/decryption primitive
- This approach, unlike the other RSA-based schemes, introduces a randomization process that enables the security of the method to be shown to be closely related to the security of the RSA algorithm itself



Mask Generation Function (MGF)

- Acts as a **pseudorandom function** in RSA-PSS, used to generate a mask.
- **Key Features**
 - Based on **secure cryptographic hash functions** (e.g., SHA-1).
 - Generates **variable-length hash** using a fixed-length cryptographic hash function.
- **Parameters:**
 - **Hash:** Cryptographic hash function with output length L (in octets).
 - **Input (X):**
 - Octet string to be masked.
 - **MaskLen:** Desired length (in octets) of the generated mask.
 - **Output:**
 - A masked **octet string** of length `maskLen`.



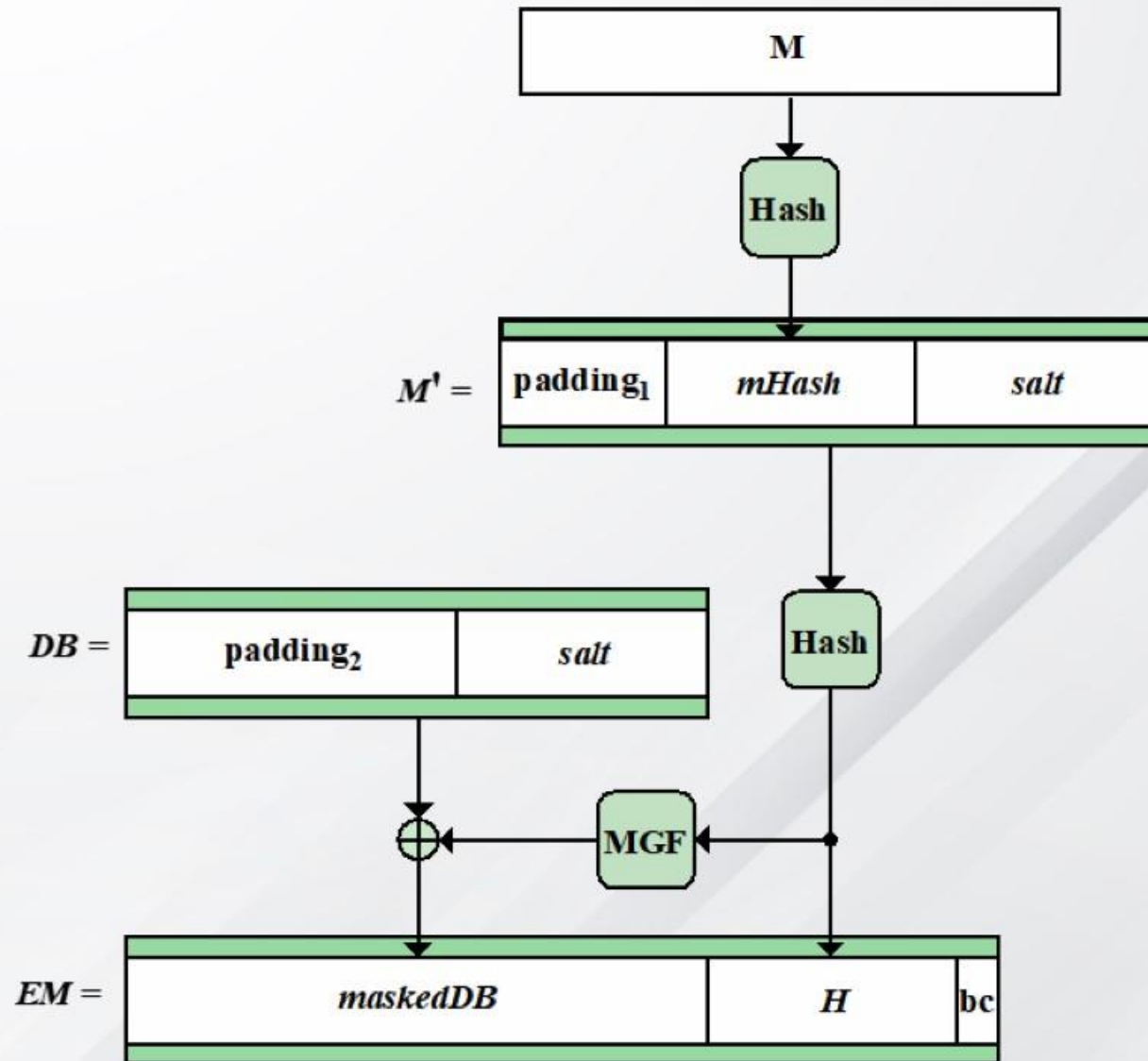


Figure 13.6 RSA-PSS Encoding

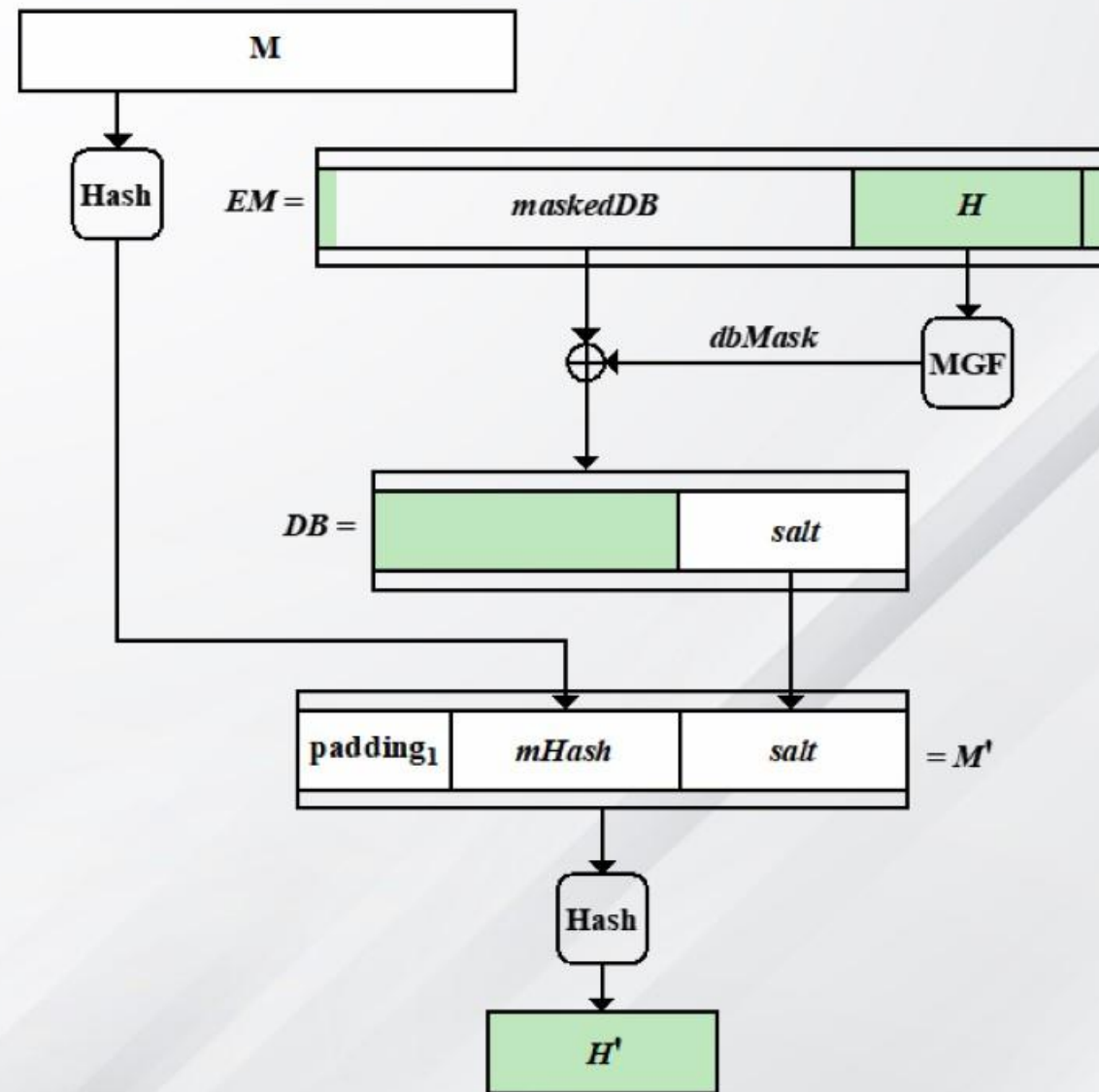


Figure 13.7 RSA-PSS EM Verification



Key Management and Distribution



Cryptographic Key Management

- The secure use of cryptographic key algorithms depends on the protection of the cryptographic keys
- *Cryptographic key management* is the process of administering or managing cryptographic keys for a cryptographic system
 - It involves the generation, creation, protection, storage, exchange, replacement, and use of keys and enables selective restriction for certain keys
- In addition to access restriction, key management also involves the monitoring and recording of each key's access, use, and context
- A key management system will also include key servers, user procedures, and protocols
- The security of the cryptosystem is dependent upon successful key management



Key Distribution Technique

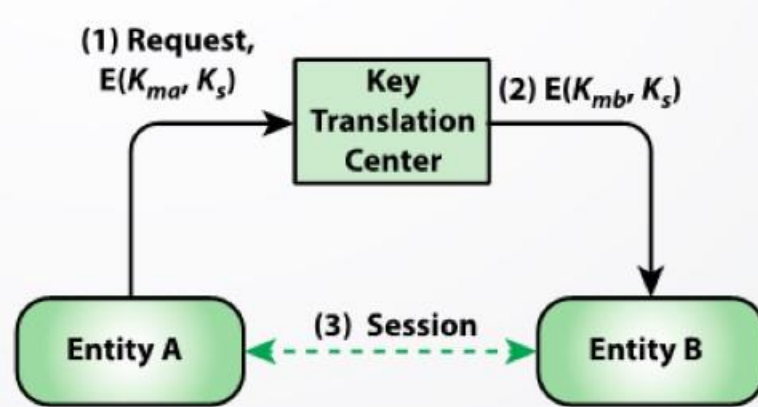
- Term that refers to the means of delivering a key to two parties who wish to exchange data without allowing others to see the key
- For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others
- Frequent key changes are desirable to limit the amount of data compromised if an attacker learns the key

Symmetric Key Distribution

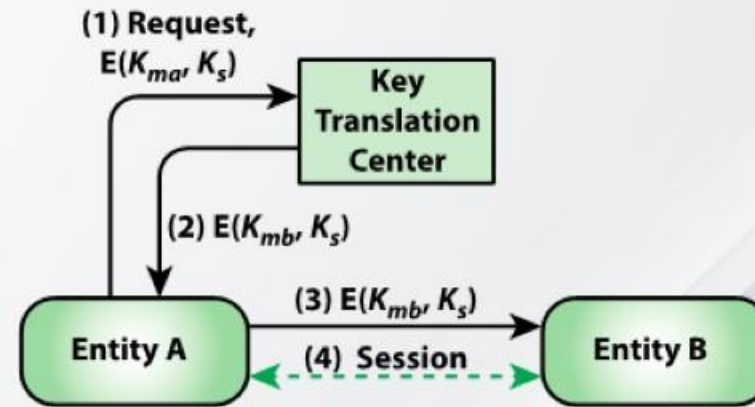
Given parties A and B, key distribution can be achieved in a number of ways:

- A can select a key and physically deliver it to B
- A third party can select the key and physically deliver it to A and B
- If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key
- If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B

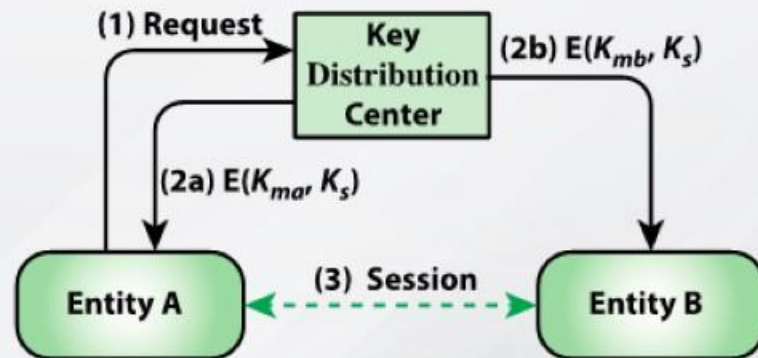




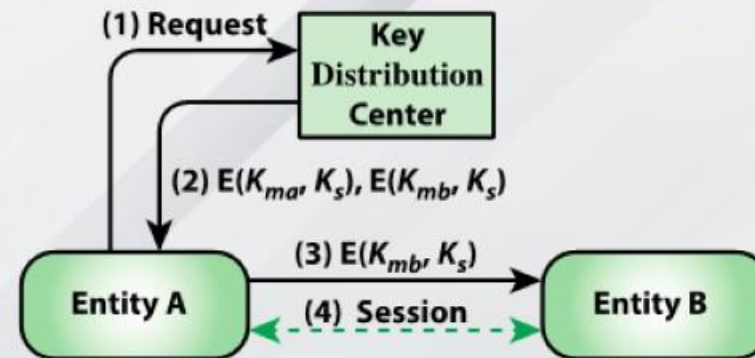
(a) Key Translation



(b) Key Translation with Key Forwarding



(c) Key Distribution



(d) Key Distribution with Key Forwarding

Figure 15.1 Key Distribution Between Two Communicating Entities



Key Hierarchy in Protocols

- **Purpose:** Secure distribution of symmetric keys (e.g., IEEE 802.11i, IPsec).
- **Key Hierarchy:**
 - **Lower-Level Keys:**
 - Used frequently and changed often to thwart attacks.
 - **Higher-Level Keys:**
 - Used infrequently and resistant to cryptanalysis.
 - Encrypt newly created lower-level keys for exchange.
- **Ephemeral Key:**
 - Used only once or for a very short duration

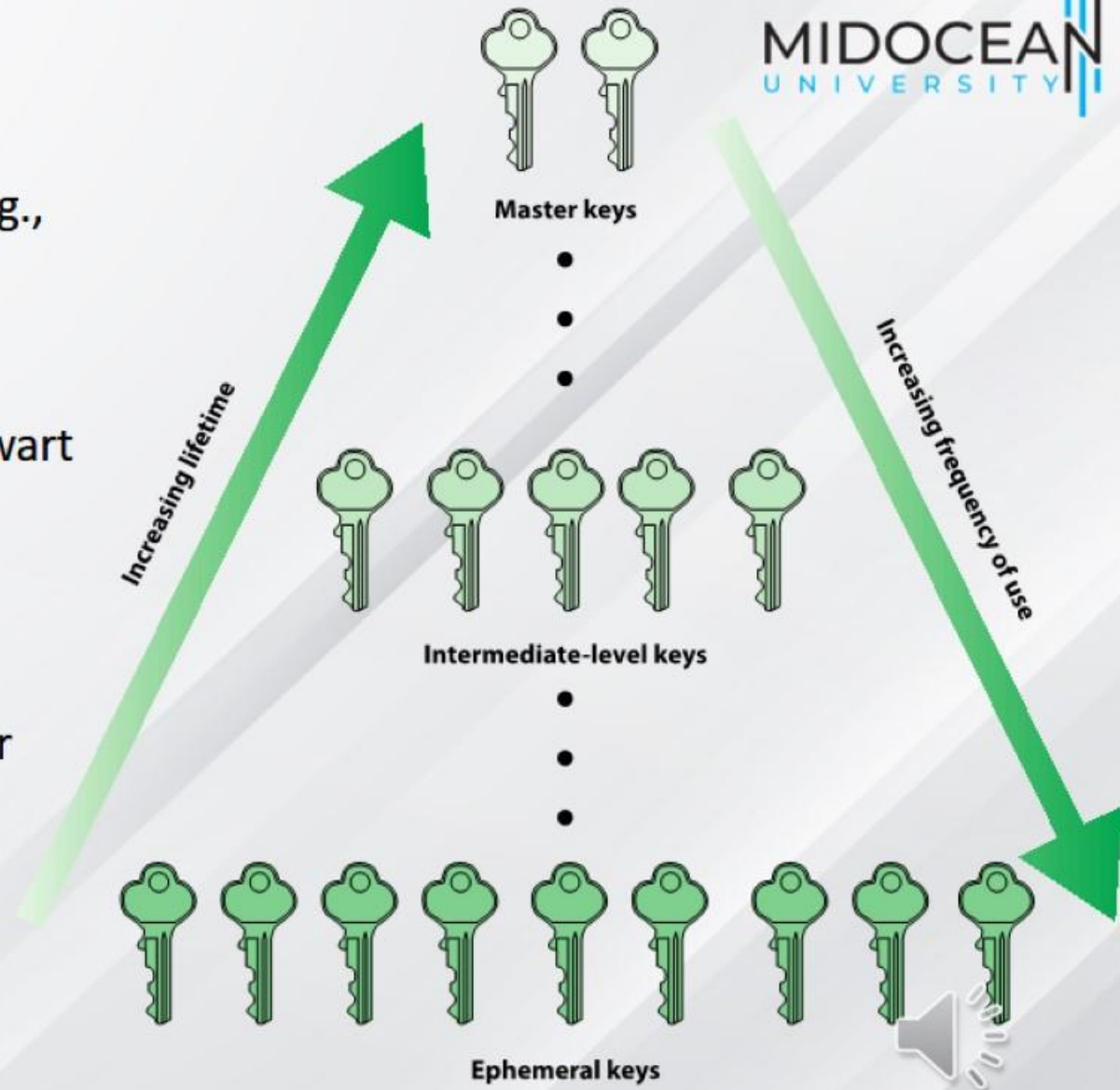


Figure 15.2 Symmetric Key Hierarchy

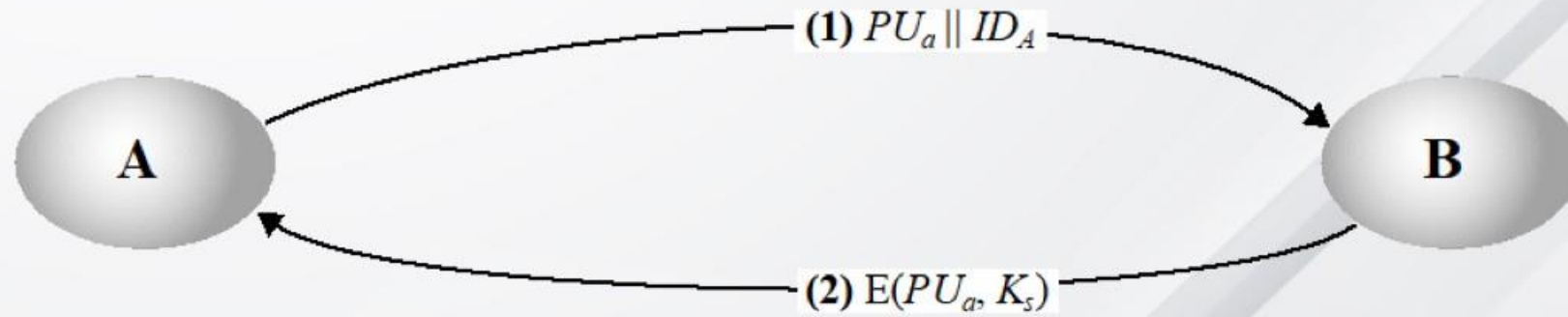


Figure 15.3 Simple Use of Public-Key Encryption to Establish a Session Key



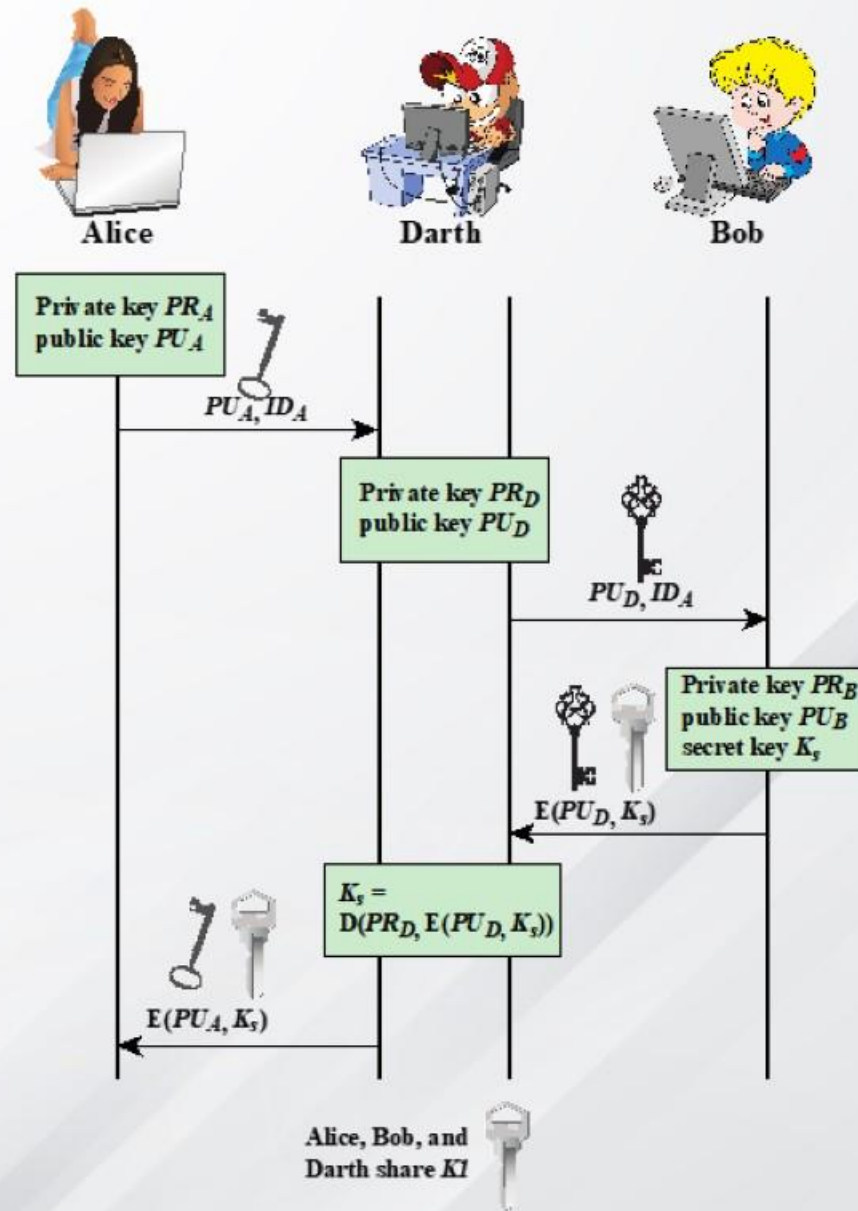


Figure 15.4 Another Man-in-the-Middle Attack

DISTRIBUTION OF PUBLIC KEYS



Public announcement

- Public keys are openly shared (e.g., RSA) for communication.
 - Any participant can broadcast their public key to others or the community.
- **Weakness:** Vulnerability to **forgery**.
 - A malicious actor can impersonate user **A** by sending a forged public key.
 - Consequences:
 - **Decrypts messages** intended for user **A**.
 - Misuses forged keys for **authentication** (Figure 9.3).
- **Solution:** User **A** must detect and alert others about forgeries.



Figure 15.6 Uncontrolled Public Key Distribution

Publicly available directory

- Managed by a **trusted authority** or organization **Key Elements:**

1. Directory contains {name, public key} for each participant.
2. **Registration:** Participants securely register their keys
3. **Key Replacement:** Replace keys after extensive use or if compromised.
4. **Electronic Access:** Secure, authenticated communication ensures safe access to the directory.

- Advantages:**

- More secure than individual public key announcements.

- Vulnerabilities:**

- **Private Key Compromise:** An attacker with the authority's private key could distribute fake public keys.
- **Tampering:** Records in the directory could be manipulated to facilitate impersonation or eavesdropping.

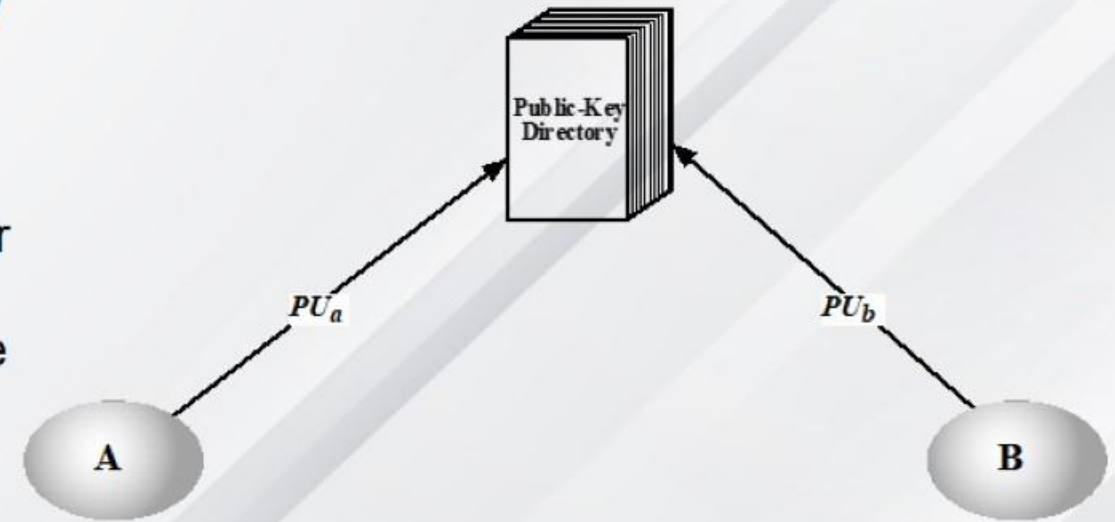


Figure 15.7 Public Key Publication

Public-Key Authority

Enhanced Directory Control:

- Central authority maintains a **dynamic directory** of public keys.
- Participants securely know the authority's **public key** (authority holds the private key).
- **Process Overview:**
 - Seven messages exchanged for initial public key distribution.
 - Steps ensure **authenticated key sharing** between participants.
 - **Optimization Through Caching:**
 - Participants **save public keys** for future use, reducing the need for repeated key exchanges.
 - Periodic updates ensure keys remain **current and secure**.
- **Benefit:** Tighter control minimizes risks from tampered or counterfeit keys.

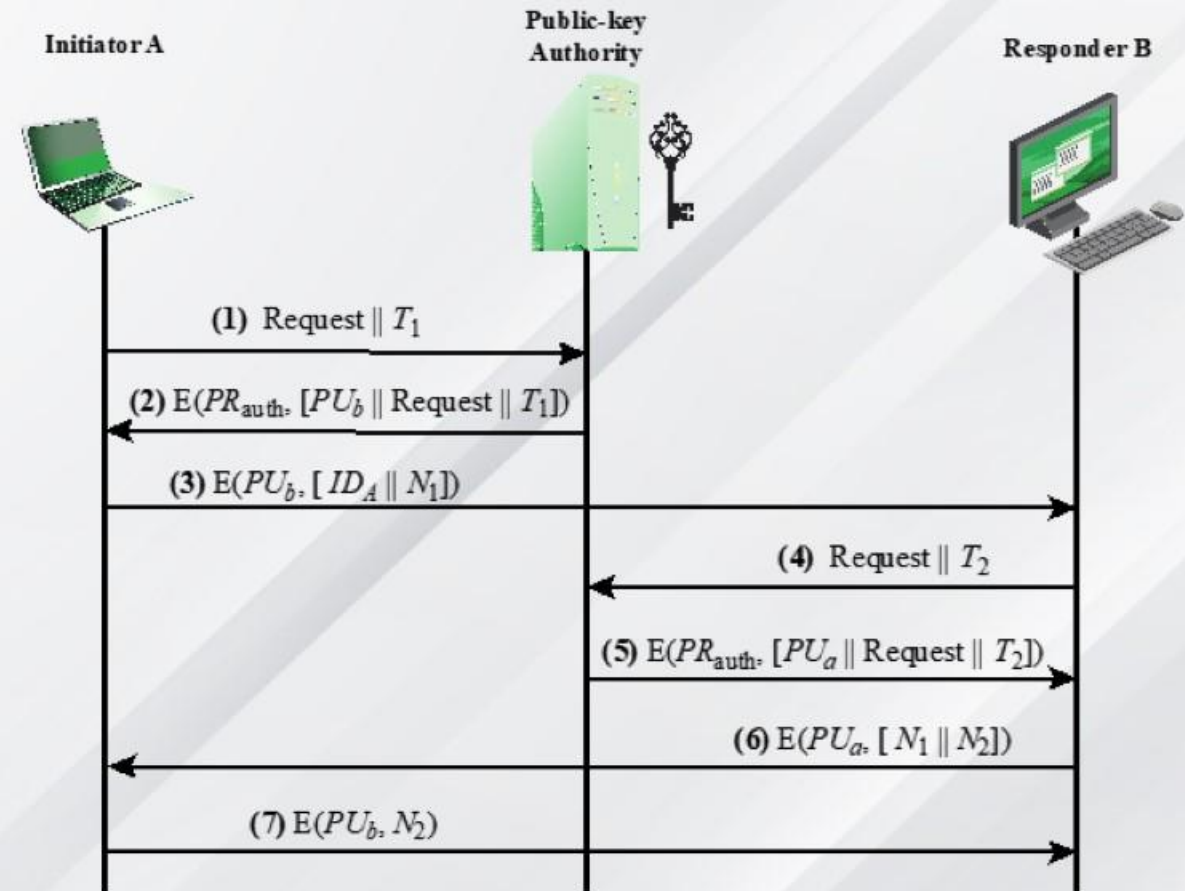


Figure 15.8 Public-Key Distribution Scenario

Public-Key Certificates

- Drawbacks of Public-Key Authorities:
 - Bottleneck: Every key request requires contacting the authority.
 - Vulnerability: Directory of public keys can be tampered with.
- Certificates: Allow users to exchange keys without contacting the public-key authority.
- Certificates are as reliable as direct keys from the authority.
- A certificate consists of:
 - Public key
 - Identifier of the key owner
 - Signed block by a trusted third party (certificate authority).

How Certificates Work

- Participants apply for a certificate with their public key.
- The certificate authority provides a signed certificate.
- The recipient verifies the certificate by decrypting it with the authority's public key.
- The certificate includes:
 - The public key.
 - The identifier of the key owner.
 - A timestamp.

Example Certificate Encryption

- The certificate format:
 - $CA = E(PR_{auth}, [T || ID_a || PU_a])$
- Decryption by recipient:
 - $D(PU_{auth}, CA) = (T || ID_a || PU_a)$
- The certificate's timestamp ensures its validity.

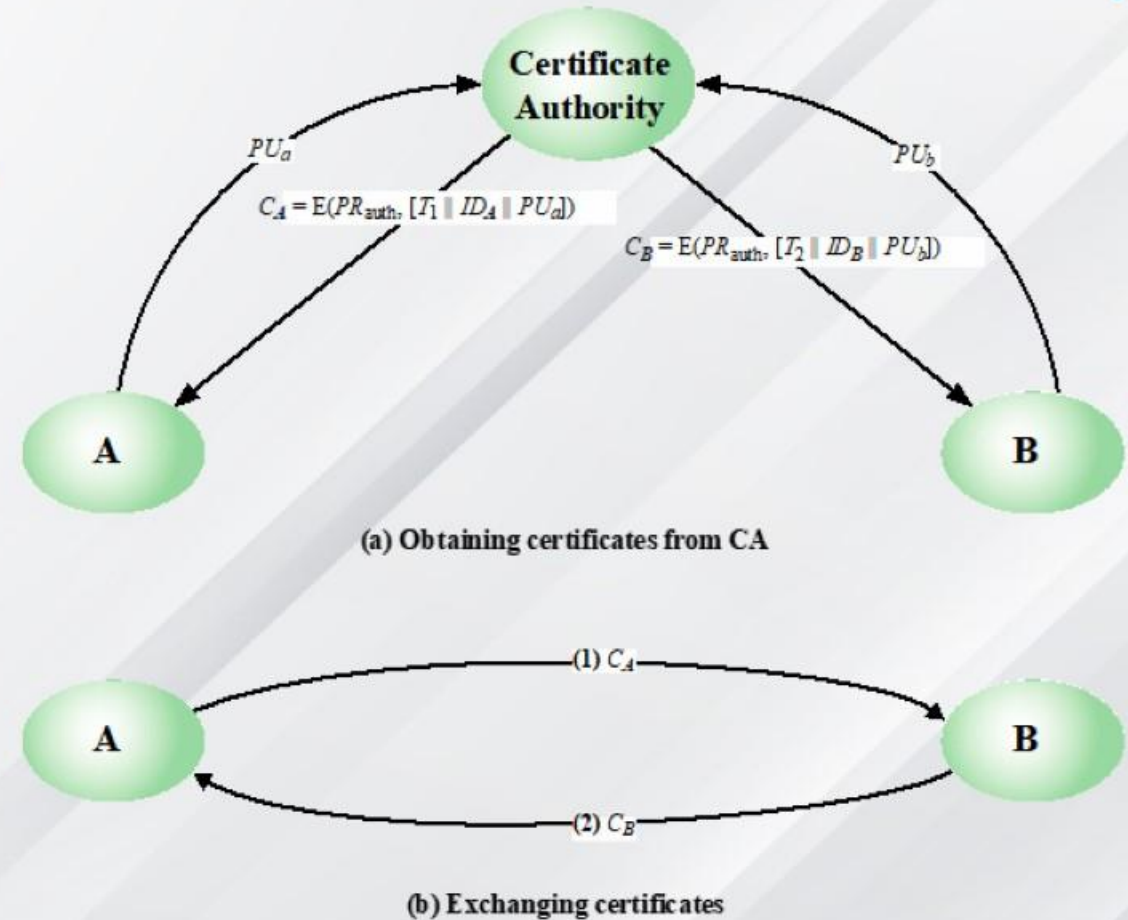


Figure 15.9 Exchange of Public-Key Certificates

Public-Key Certificates

Certificate Authorities

- **Trusted third parties:** Governments or financial institutions, trusted by the user community.
- The certificate verifies the **public key** of the user.
- The user can present their certificate to others for key exchange.

Private Key Compromise Scenario:

- Scenario: A's private key is compromised.
- A generates a new private/public key pair and requests a **new certificate**.
- **Risk:** Adversary replays old certificate to B, and B encrypts messages using compromised public key.
- Similar to losing a **credit card**; until all participants are aware, there is risk.



Public-Key Certificates

- Certificate Expiration and Timestamping
 - Timestamp acts like an expiration date.
 - If a certificate is too old, it is assumed to be expired.
 - The timestamp ensures the validity of certificates.
- Certificate Scheme Requirements
 1. Any participant can read the certificate to determine the name and public key of the owner.
 2. Any participant can verify that the certificate is valid and not counterfeit.
 3. Only the certificate authority can create and update certificates.



X.509 Certificates

- Part of the X.500 series of recommendations that define a directory service
 - The directory is, in effect, a server or distributed set of servers that maintains a database of information about users
- X.509 defines a framework for the provision of authentication services by the X.500 directory to its users
 - Was initially issued in 1988 with the latest revision in 2016
 - Based on the use of public-key cryptography and digital signatures
 - Does not dictate the use of a specific algorithm but recommends RSA
 - Does not dictate a specific hash algorithm
- Each certificate contains the public key of a user and is signed with the private key of a trusted certification authority
- X.509 defines alternative authentication protocols based on the use of public-key certificates



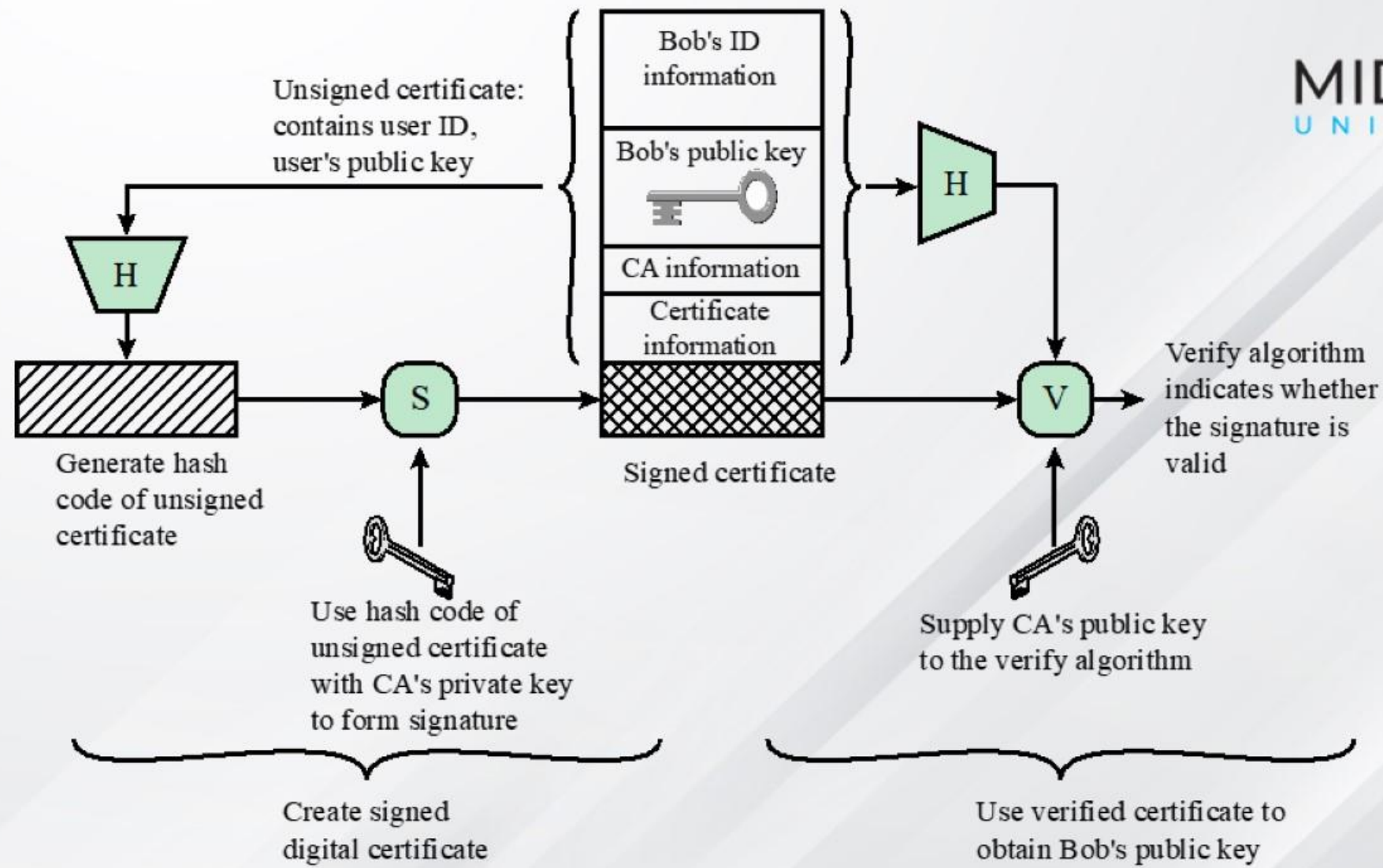


Figure 15.10 Public-Key Certificate Use

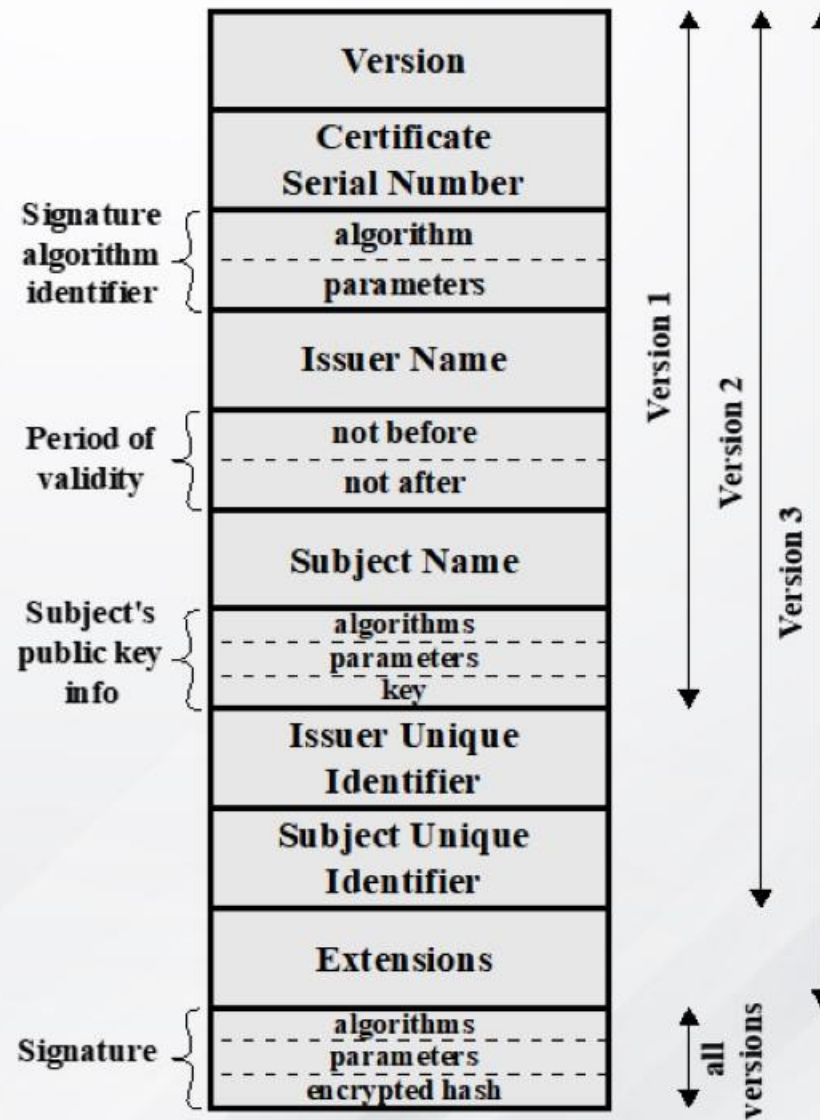


Certificates

Created by a
trusted
Certification
Authority (CA)
and have the
following
elements:

- Version
- Serial number
- Signature algorithm identifier
- Issuer name
- Period of validity
- Subject name
- Subject's public-key information
- Issuer unique identifier
- Subject unique identifier
- Extensions
- Signature





(a) X.509 Certificate



Summary

- Digital Signatures: Ensure integrity, authenticity, and non-repudiation.
- Requirements: Message dependence, sender-specific info, ease of use, and forgery resistance.
- Direct Digital Signatures: Sender-created signatures without intermediaries.
- ECDSA: Secure, efficient, and based on elliptic curve cryptography.
- RSA-PSS: Enhances RSA with randomization and Mask Generation Function (MGF).
- Key Management: Secure processes for key generation, storage, and exchange.
- Key Distribution: Symmetric challenges; certificates simplify public-key sharing.
- X.509 Certificates: Trusted framework for authentication via certified public keys.





Thank You

