UNIT - 5

DIGITAL SIGNATURE

DIGITAL SIGNATURE

- A digital signature is a specific type of electronic signature (e-signature)

- It is based on public-key cryptography to support identity authentication and provide data and transaction integrity.

- It uses public key and private key for this mechanism.

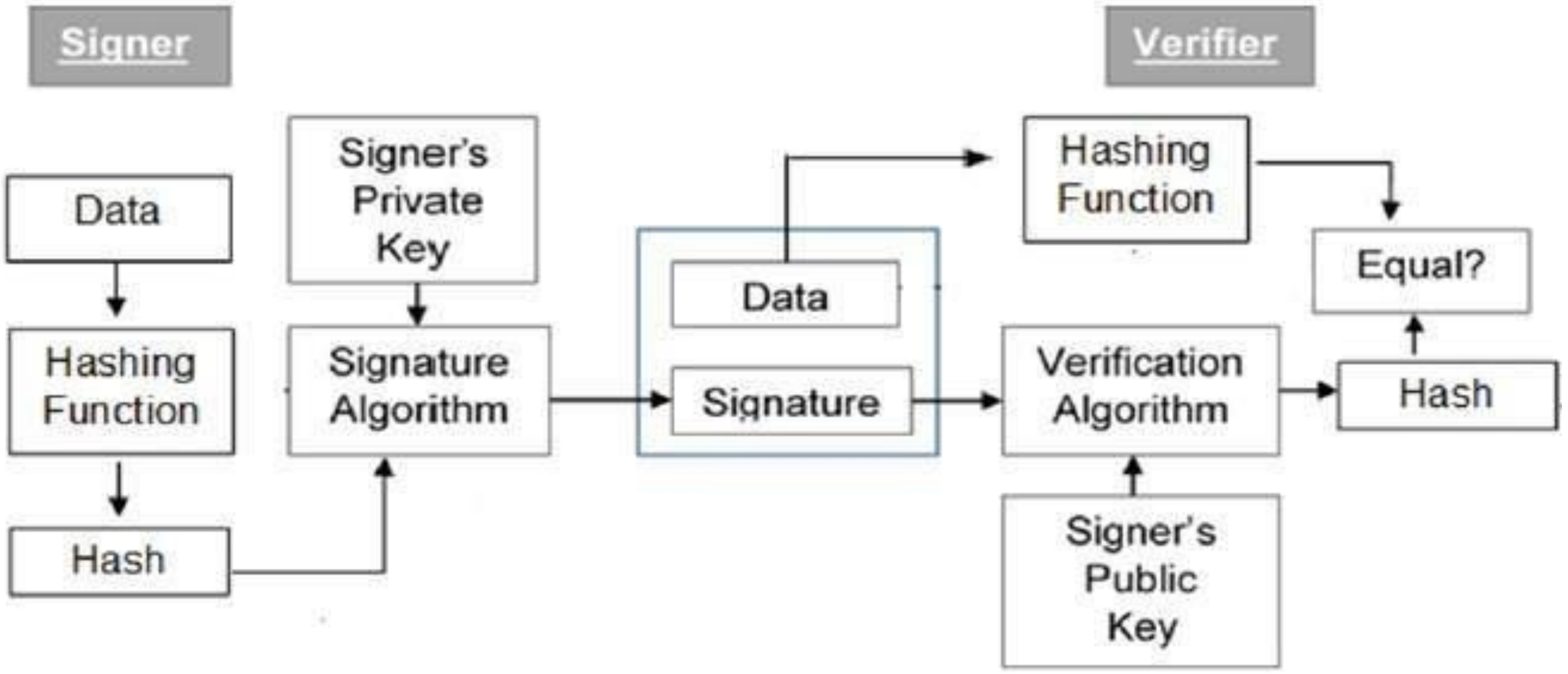
- A digital signature is a mathematical technique used to validate the authenticity and integrity of a digital document, message or software.

PURPOSE OF DIGITAL SIGNATURE

- The digital signature confirms the integrity of the message.

- This signature ensures that the information originated from the signer and was not altered, which proves the identity of the organization that created the digital signature.

- Any change made to the signed data invalidates the whole signature.


Signer
Verifier
Data
Signer's Private Key
Hashing Function
Signature Algorithm
Hash
Data
Signature
Hashing Function
Equal?
Verification Algorithm
Hash
Signer's Public Key

DIGITAL SIGNATURE ALGORITHM

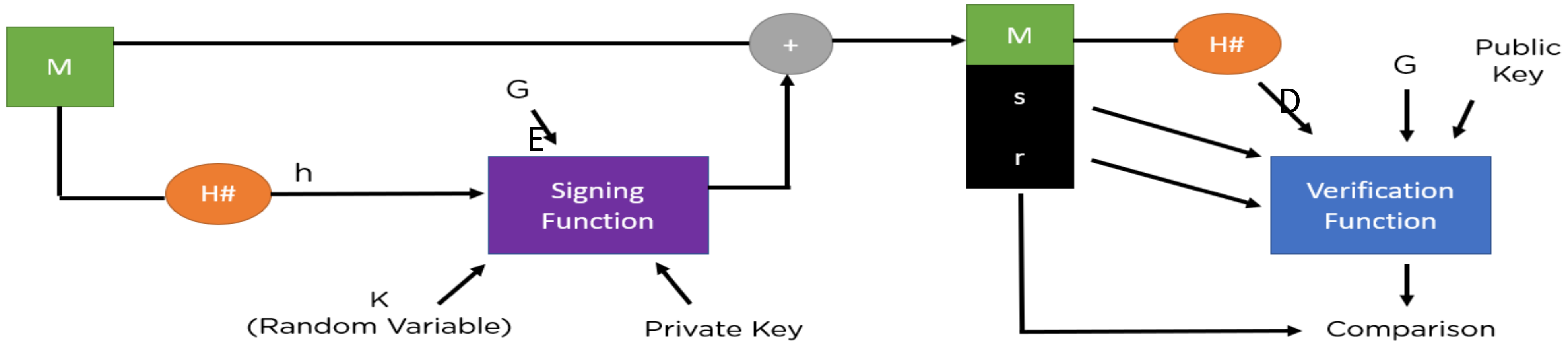
DSA Algorithm provides three benefits:

- **Message Authentication**: You can verify the origin of the sender using the right key combination.

- **Integrity of message**: You cannot tamper with the message since it will prevent the bundle from being decrypted altogether.

- **Non-repudiation**: The sender cannot claim they never sent the message if verifies the signature.

Block diagram of Digital Signature Algorithm:



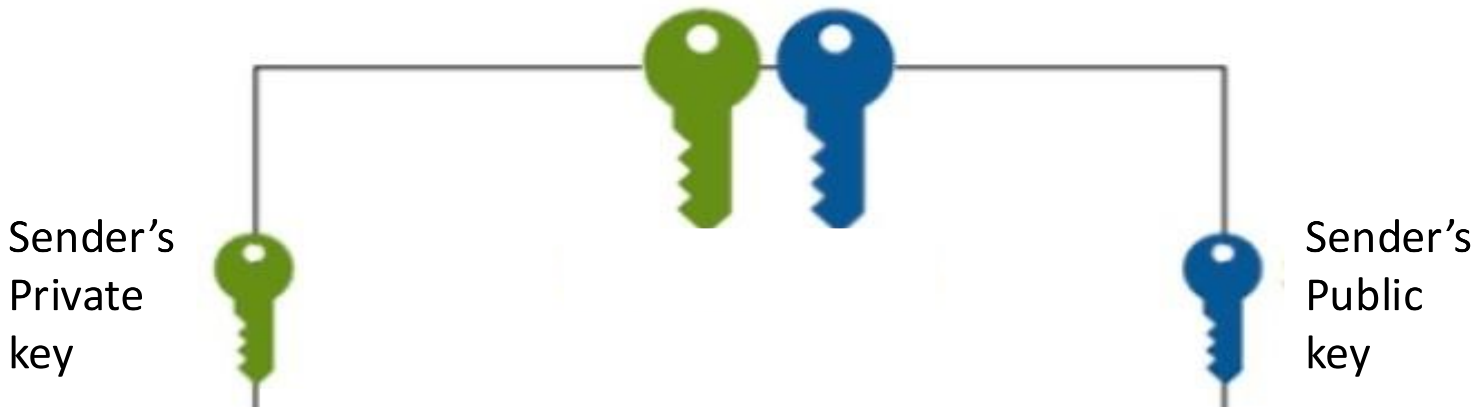
M - Plaintext
H - Hash function
h - Hash digest
S- signature
r- first part of signature

'+' - Bundle both plaintext and message digest
E - Encryption
D - Decryption

Steps In Digital Signature Algorithm :

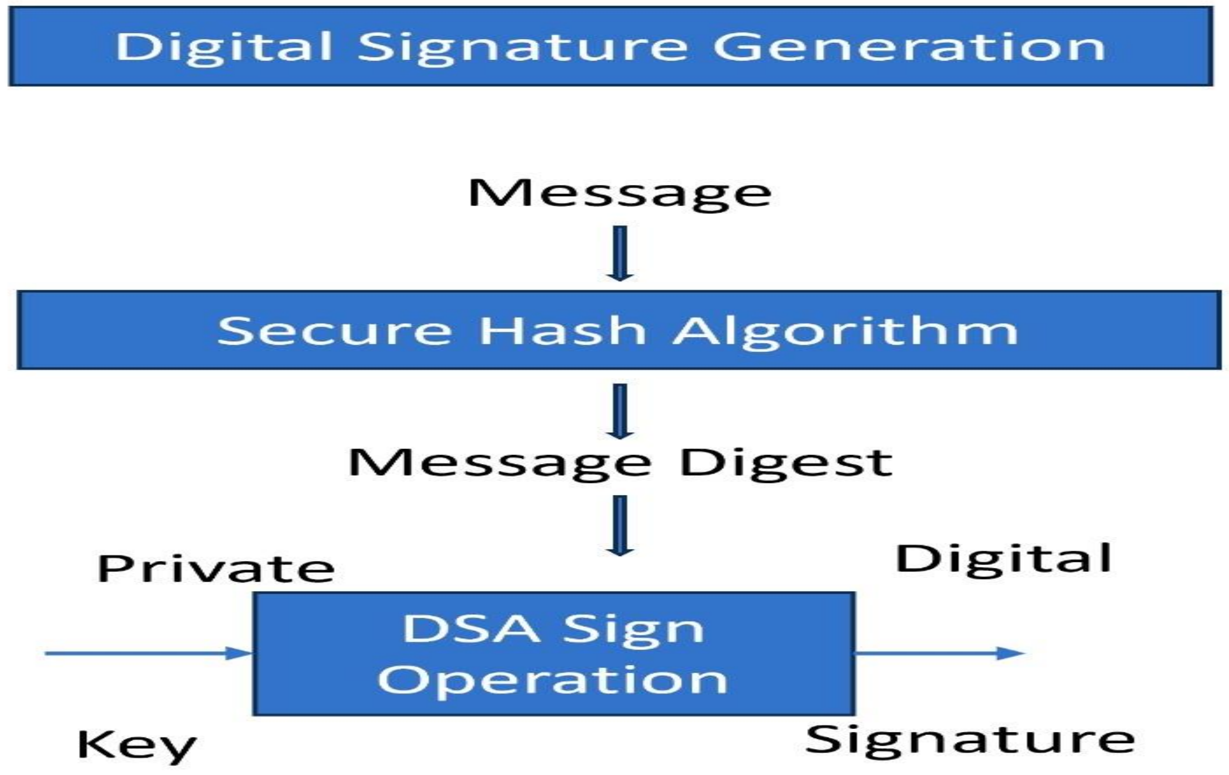
Key Generation:

- In the initial stage of the DSA process is key generation . Here pair of keys are generated- sender’s private key , sender’s public key
- Sender maintains confidentiality of the private key, whereas the public key is openly shared.



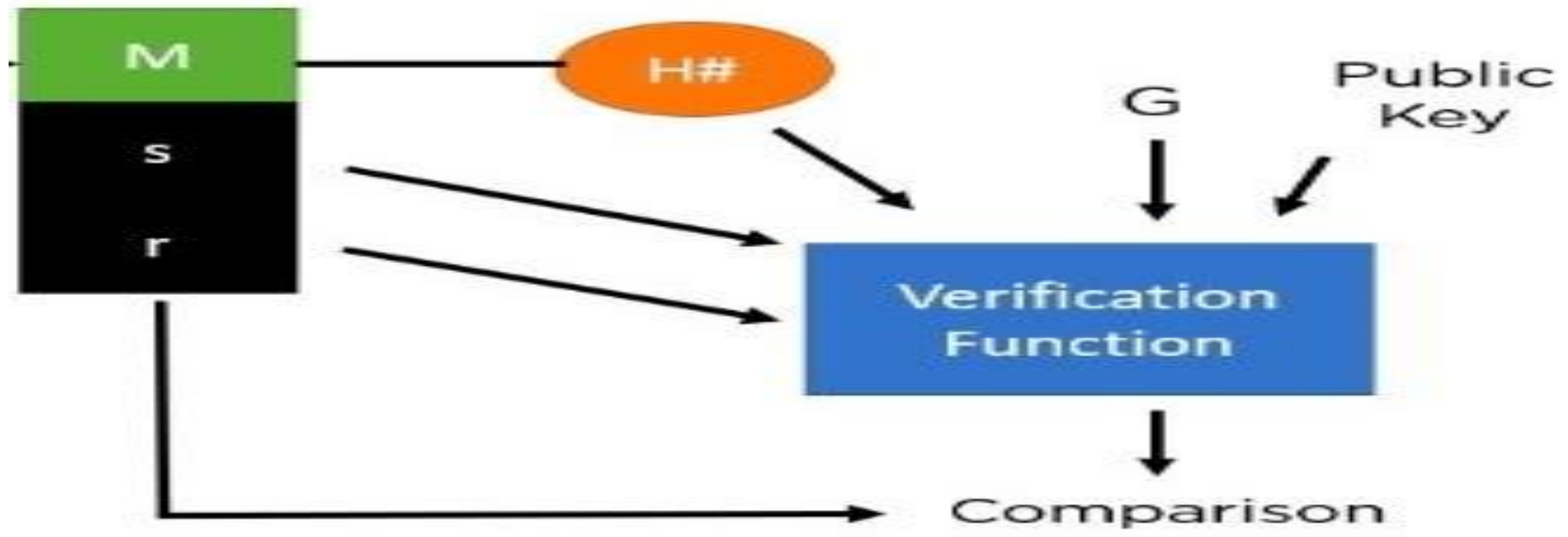
Signature Generation:

Initially the message undergoes a secure has algorithm and produces a hash value/message digest. Then message digest is encrypted using sender private key to generate a Digital signature for the message



Signature Verification:

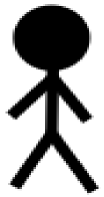
- Here the receiver uses the sender's public key to verify that the signature is valid and accepts if the comparision matches

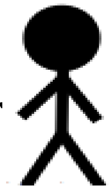


SCHNORR DIGITAL SIGNATURE

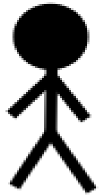
Schnorr Digital Signature

- The Schnorr signature scheme is based on discrete logarithms.
- The Schnorr scheme minimizes the message-dependent amount of computation required to generate a signature.
- The main work for signature generation does not depend on the message and can be done during the idle time of the processor.
  - Step 1 : Key Generation
  - Step 2 : Signature Generation
  - Step 3 : Verification Algorithm

Sender 

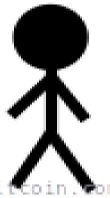
Receiver 

$$\begin{tikzcd}[column sep=small] ed & + & k & = & s \arrow[d] & \text{signature} \arrow[l, bend right] \\ edG & + & kG & = & sG \end{tikzcd}$$

Sender 

$$\overset{e}{5} \times \overset{d}{4} + \overset{k}{7} = \overset{s}{27}$$

$$G=3$$

Receiver 

$$\overset{e}{5} \times \overset{dG}{12} + \overset{kG}{21} = \overset{sG}{81}$$

e = Encryption
d = Data
k = Key
G = Generator point
s = Signature

Public Key Generation

$$P_1 = d_1 \cdot G \quad , \quad P_2 = d_2 \cdot G$$

$$P_1 + P_2 = (d_1 \cdot G) + (d_2 \cdot G)$$

$$P_1 + P_2 = (d_1 + d_2) \cdot G$$

$$P_{1+2} = d_{1+2} \cdot G$$

Signature Generation

$$s_1 = k_1 + ed_1 \quad , \quad s_2 = k_2 + ed_2$$

$$s_1 + s_2 = (k_1 + ed_1) + (k_2 + ed_2)$$

$$s_1 + s_2 = k_1 + k_2 + ed_1 + ed_2$$

$$s_1 + s_2 = (k_1 + k_2) + e(d_1 + d_2)$$

$$s_{1+2} = k_{1+2} + ed_{1+2}$$

Elliptic curve digital signature algorithm

Elliptic-curve cryptography (ECC) Algorithm

- Elliptic-curve cryptography (ECC) is type of public-key cryptography based on the algebraic structure of elliptic curves over finite fields.
- ECC requires smaller keys than to non-EC cryptography (i.e. RSA) to provide equivalent security, and is therefore preferred when higher efficiency or stronger security (via larger keys) is required.
- ECC is used for key agreement, digital signatures, pseudo-random generators and other tasks.

ECC algorithms

ECC provides several different groups of algorithms based on their use cases, defined over an elliptic curve over finities fields.

- **Digital Signatures:**

ECC provides algorithms for digital signatures like the elliptic curve digital signature algorithm **Elliptic Curve Digital Signature Algorithm(ECDSA)** and **Edwards-curve Digital Signature Algorithm(EdDSA)**.

- **Encryption:**

ECC provides algorithms for encrypting messages, namely **Elliptic Curve Integrated Encryption Scheme (ECIES)** and **EC-based ElGamal Elliptic Curve Cryptography (EEECC)**.

- **Key agreement:**

ECC provides algorithms like **Elliptic-curve Diffie–Hellman (ECDH)** and **Fully Hashed Menezes-Qu-Vanstone (FHMQV)** for key agreement.

Elliptic Curve Digital Signature Algorithm (ECDSA)

- The Elliptic Curve Digital Signature Algorithm is a Digital Signature Algorithm (DSA) that uses elliptic curve cryptography keys.
- It is a very efficient equation that is based on cryptography with public keys.
- ECDSA is utilized in many security systems, is popular in encrypted messaging apps, and is the foundation of Bitcoin security (with Bitcoin "addresses" serving as public keys).
- Elliptic Curve Digital Signature Algorithms (ECDSA) have recently received significant attention, particularly from standards developers, as alternatives to existing standard cryptosystems such as integer factorization cryptosystems and discrete logarithm problem cryptosystems.

Digital Signature of ECDSA

- A digital signature is an electronic equivalent of a handwritten signature that allows a receiver to persuade a third party that the message was indeed sent by the sender.
- Handwritten signatures are substantially less secure than digital signatures.
- A digital signature cannot be forged in any way.
- Every part of the digital message is affected by the signature key.

Key generation

The process of public-private key generation in ECDSA as follows:

Private key: The private key is a randomly selected number ***np*** such that *np* is in the interval 1 to *no*- 1, where *no* is the order of the subgroup of the elliptic curve points, generated by the generator point *G*.

Public key: The public key is given as *P=npG*, where *np* is the private key selected randomly above, *G* is the generator point of the elliptic curve, and *P* is the public key.

Signature verification

The signature verification algorithm takes the message and the signature r,s as input, and returns a boolean value representing whether the signature is verified. The signature verification algorithm works as follows:

1.Message hash:

We calculate the hash *h*h of the message m using the same hash function that was we used during the signature generation, as follows: **$h=hash(m)$**

2.Modular inverse:

We calculate the modular inverse of the signature, as follows: **$s_{inverse}=s-1(mod\ n)$**

3.Random point:

We recalculate the random point R' as in the signature generation process, where P is the public key of the sender, as follows: **$R'=(h\times s_{inverse})\times G+(r\times s_{inverse})\times P$**

4.x-**coordinate:**

We get the x-coordinate of the recalculated random point, as follows:

r’=R’.x

5.Verify:

We verify the result by matching the recently calculated r’ with the r that came as part of the signature, as follows:

r’==r

RSA-PSS

Digital Signature Algorithm

INTRODUCTION

- ✓ RSA Probabilistic Signature Scheme (RSA-PSS), which is the latest of the RSA schemes
- ✓ RSA-based schemes are widely deployed in many applications, including financial applications
- ✓ The one that RSA Laboratories recommends as the most secure of the RSA schemes.
- ✓ The PSS approach was first proposed by Bellare and Rogaway
- ✓ That scheme is secure as RSA encryption/ decryption.

RSA-PSS
Digital Signature Algorithm

#Process of RSA-PSS:

- Mask Generation Function (MGF)
- The Signing Operation
- Signature verification

Mask Generation Function (MGF)

- ✓Mask Generation Function (MGF) is used as a building fixed length output
- ✓Mask generation function (MGF) used as a building block
- ✓MGFs are typically based on a secure cryptographic hash function such as SHA-1.
- ✓An MGF based on a hash function is intended to be a cryptographically secure way of generating a message

  digest

Signing Operation

- ✓MESSAGE ENCODING PROCESS
- ✓The first stage in generating an RSA-PSS signature of a message M is to generate from M a fixed-length message digest, called an encoded message (EM).

Signature Verification

- ✓DECRYPTION
- ✓EM VERIFICATION(encoded message)

Parameters ;

- ✓**emLen**

  length of EM in octets = [emBits/8]

- ✓**Padding1**

  hexadecimal string 00 00 00 00 00 00 00 00

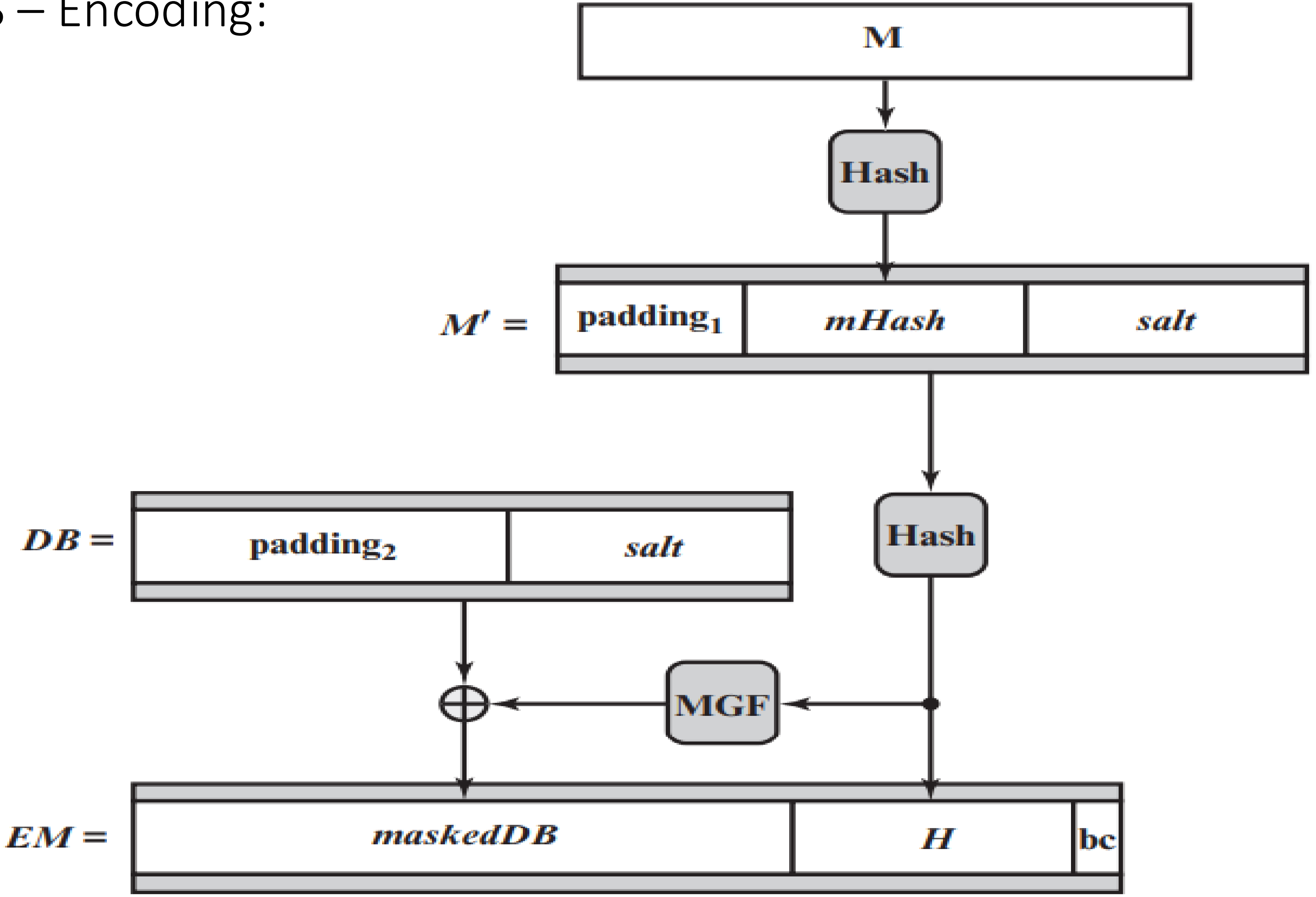
- ✓**Padding2**

  hexadecimal string of 00 octets with a length

- ✓**Salt**

  A pseudorandom number

PSS – Encoding:



SYMMETRIC KEY DISTRIBUTION USING SYMMETRIC ENCRYPTION

SYMMETRIC KEY DISTRIBUTION USING SYMMETRIC ENCRYPTION

1. **Concept**
2. **A Key Distribution Scenario**
3. **Hierarchical Key Control**
4. **Session Key Lifetime**
5. **A Transparent Key Control Scheme**
6. **Decentralized Key Control**
7. **Controlling Key Usage**

Concept

Symmetric Encryption:

Same key needs to be shared with sender and receiver

Key Distribution:

1. A can select a key and physically deliver it to B.
2. A third party can select the key and physically deliver it to A and B.
3. 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.
4. 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.

Link Encryption

Options 1 and 2 call for manual delivery of a key.link encryption device is going to be exchanging data only with its partner on the other end of the link.

End-to-end Encryption

Over a network, manual delivery is awkward. In a distributed system, any given host or terminal may need to engage in exchanges with many other hosts and terminals over time. End-to-end Encryption requires many keys.

Number of required keys is[N(N - 1)]/2.

The use of a key distribution center is based on the use of a hierarchy of keys.

Session Keys: Communication between end systems is encrypted using a temporary key, often referred to as a session key.

Master keys: For each end system or user, there is a unique master key that it shares with the key distribution center.

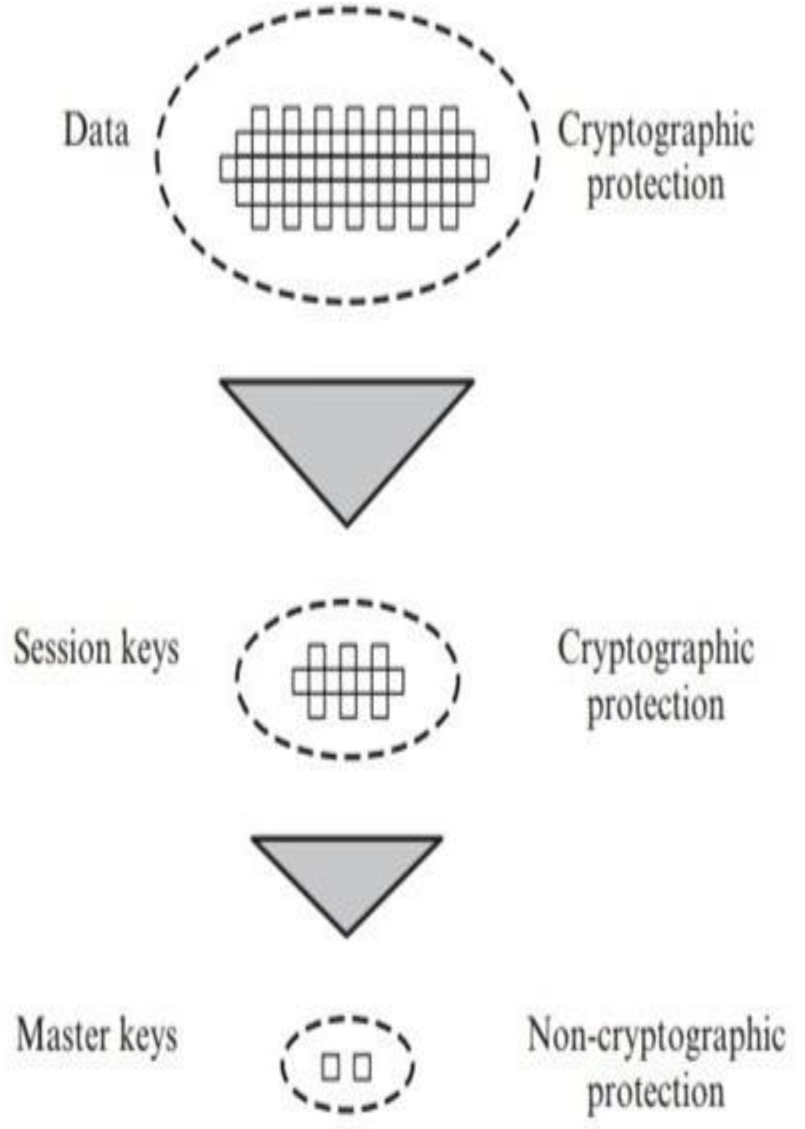


Figure 14.2 The Use of a Key Hierarchy

A Key Distribution Scenario

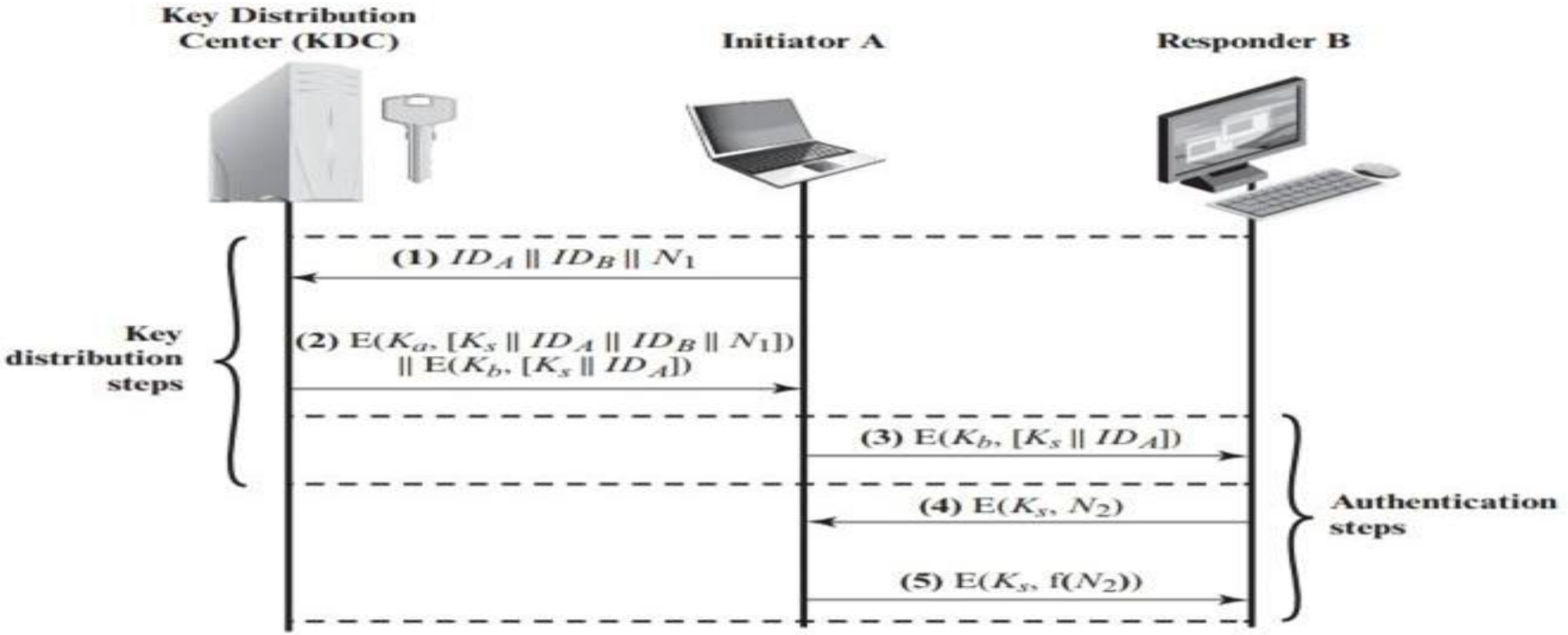

Key Distribution Center (KDC)
Initiator A
Responder B
(1) $ID_A \| ID_B \| N_1$
Key distribution steps
(2) $E(K_a, [K_s \| ID_A \| ID_B \| N_1]) \| E(K_b, [K_s \| ID_A])$
(3) $E(K_b, [K_s \| ID_A])$
(4) $E(K_s, N_2)$
(5) $E(K_s, f(N_2))$
Authentication steps

Figure 14.3 Key Distribution Scenario

A has a master key Ka, B shares the master key Kb with the KDC.

1. A issues a request to the KDC for a session key to protect a logical connection to B. The message includes the identity of A and B and a unique identifier, N1, for this transaction, which we refer to as a nonce.

2. The KDC responds with a message encrypted using Ka. Thus, A is the only one who can successfully read the message.Two items intended for A:

■ The one-time session key, Ks, to be used for the session

■ The original request message, including the nonce, to enable A to match this response with the appropriate request

Two items intended for B: These last two items are encrypted with Kb .

- ■ The one-time session key, Ks, to be used for the session
- ■ An identifier of A (e.g., its network address), IDA

3. A forwards to B the information that originated at the KDC for B, namely, E(Kb,[Ks } IDA]).Because this information is encrypted with Kb, it is protected from eavesdropping. B now knows the session key (Ks), knows that the other party is A (from IDA).

4. Using the newly minted session key for encryption, B sends a nonce, N2, to A.

5. Also, using Ks, A responds with f(N2), where f is a function that performs some transformation on N2 (e.g., adding one).

SYMMETRIC KEY DISTRIBUTION USING ASYMMETRIC ENCRYPTION

Simple Secret Key Distribution:

1. A generates a public/private key pair {PUa, PRa} and transmits a message to B consisting of PUa and an identifier of A, IDA.
2. B generates a secret key, Ks, and transmits it to A, which is encrypted with A's public key.
3. A computes D(PRa, E(PUa, Ks)) to recover the secret key. Because only A can decrypt the message, only A and B will know the identity of Ks.
4. A discards PUa and PRa and B discards PUa.

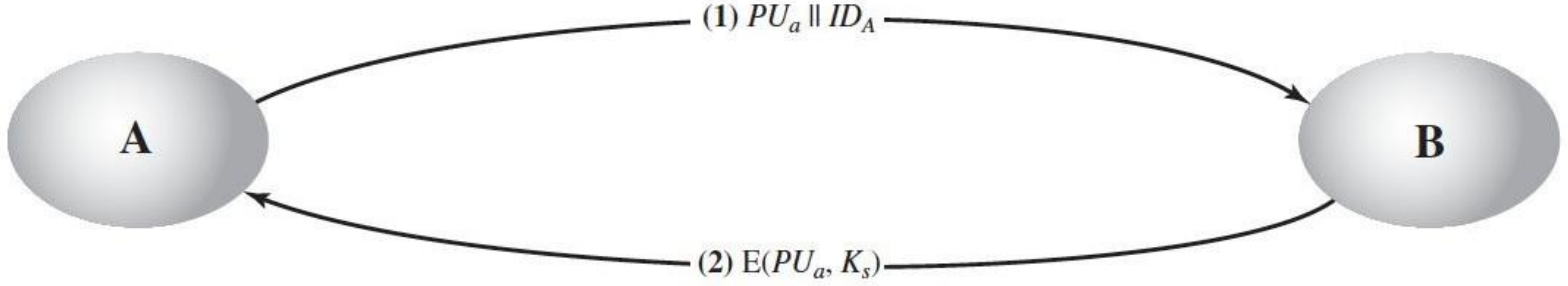


Figure 14.7 Simple Use of Public-Key Encryption to Establish a Session Key

In the prsent case, if an adversary, D, has control of the intervening communication channel, then D can compromise the communication in the following fashion without being detected (Figure 14.8).

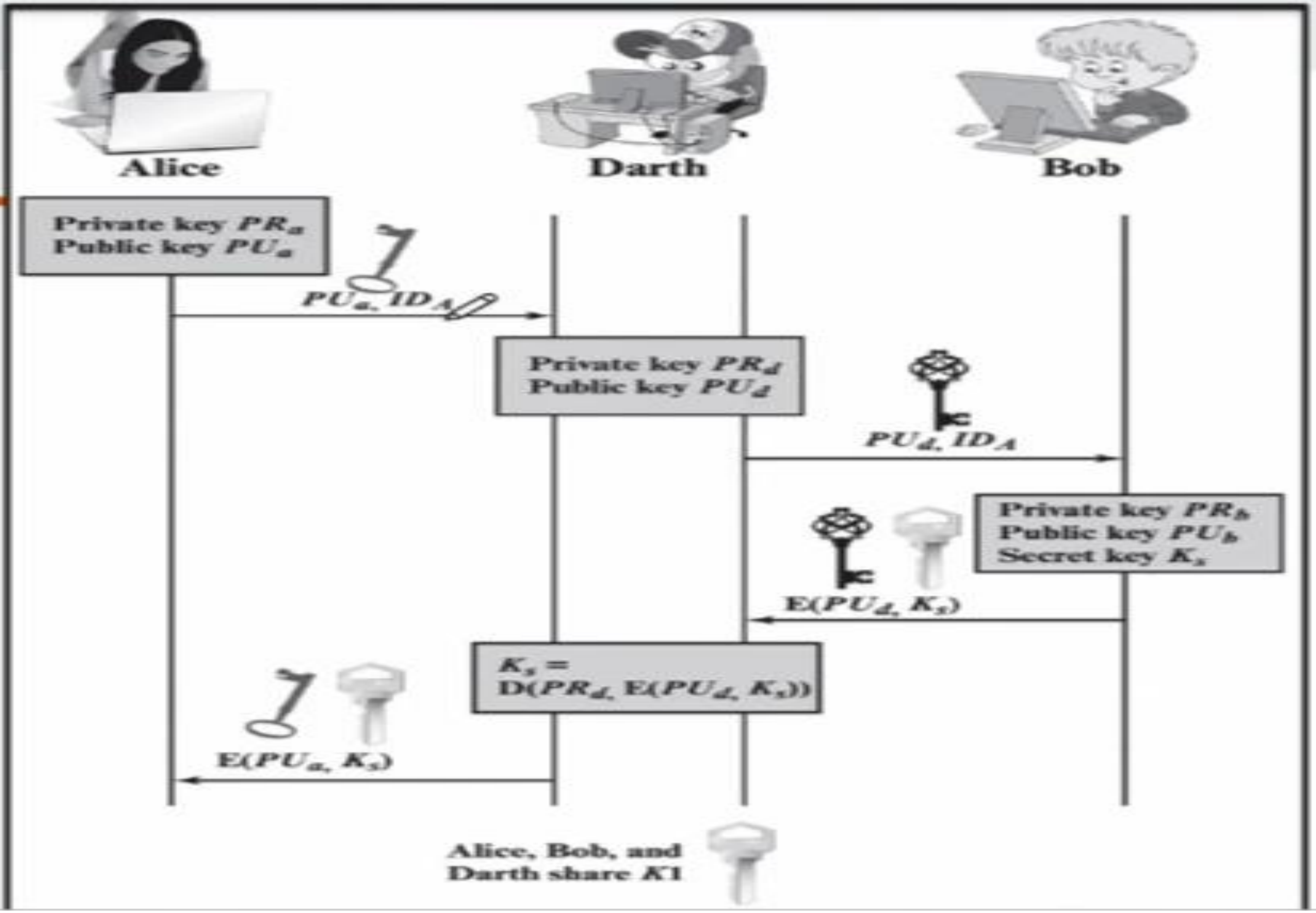
1. A generates a public/private key pair {PUa, PRa} and transmits a message intended for B consisting of PUa and an identifier of A, IDA.

2. D intercepts the message, creates its own public/private key pair {PUd, PRd} and transmits PUd } IDA to B.

3. B generates a secret key, Ks, and transmits E(PUd, Ks).

4. D intercepts the message and learns Ks by computing D(PRd, E(PUd, Ks)).

5. D transmits E(PUa, Ks) to A.


Alice
Darth
Bob
Private key PR_a
Public key PU_a
PU_a, ID_A
Private key PR_d
Public key PU_d
PU_d, ID_A
Private key PR_b
Public key PU_b
Secret key K_s
$E(PU_d, K_s)$
$K_s = D(PR_d, E(PU_d, K_s))$
$E(PU_a, K_s)$
Alice, Bob, and Darth share $K1$

Secret Key Distribution with Confidentiality and Authentication

- We begin at a point when it is assumed that A and B have exchanged public keys by one of the schemes described subsequently in this chapter. Then the following steps occur.

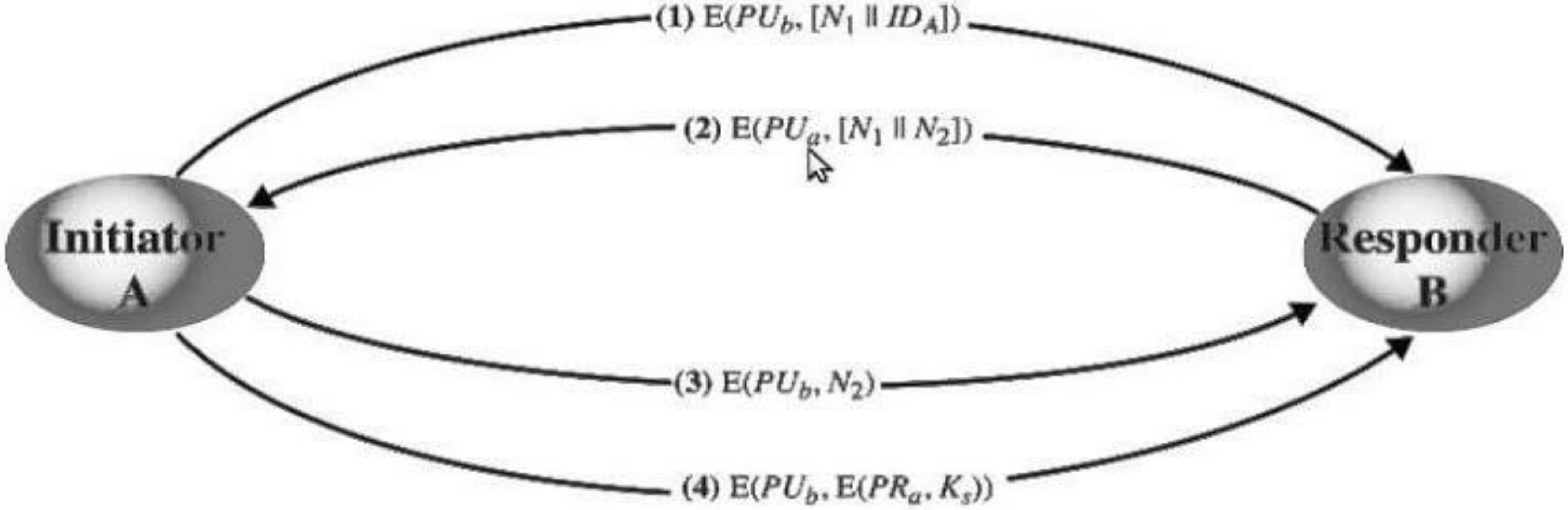
1. A uses B’s public key to encrypt a message to B containing an identifier of A(IDA) and a nonce (N1), which is used to identify this transaction uniquely.

2. B sends a message to A encrypted with PUa and containing A’s nonce (N1) as well as a new nonce generated by B (N2). Because only B could have decrypted message (1), the presence of N1 in message (2) assures A that the correspondent is B.

3. A returns N2, encrypted using B’s public key, to assure B that its correspondent is A.

4. A selects a secret key Ks and sends M = E(PUb, E(PRa, Ks)) to B. Encryption of this message with B’s public key ensures that only B can read it; encryption with A’s private key ensures that only A could have sent it.

5. B computes D(PUa, D(PRb, M)) to recover the secret key. The result is that this scheme ensures both confidentiality and authentication in the exchange of a secret key


(1) E(PU_b, [$N_1 \| ID_A$])
(2) E(PU_a, [$N_1 \| N_2$])
Initiator A
Responder B
(3) E(PU_b, N_2)
(4) E(PU_b, E(PR_a, K_s))

Public-Key Distribution of Secret Keys

DISTRIBUTION OF PUBLIC KEYS

PUBLIC KEYS

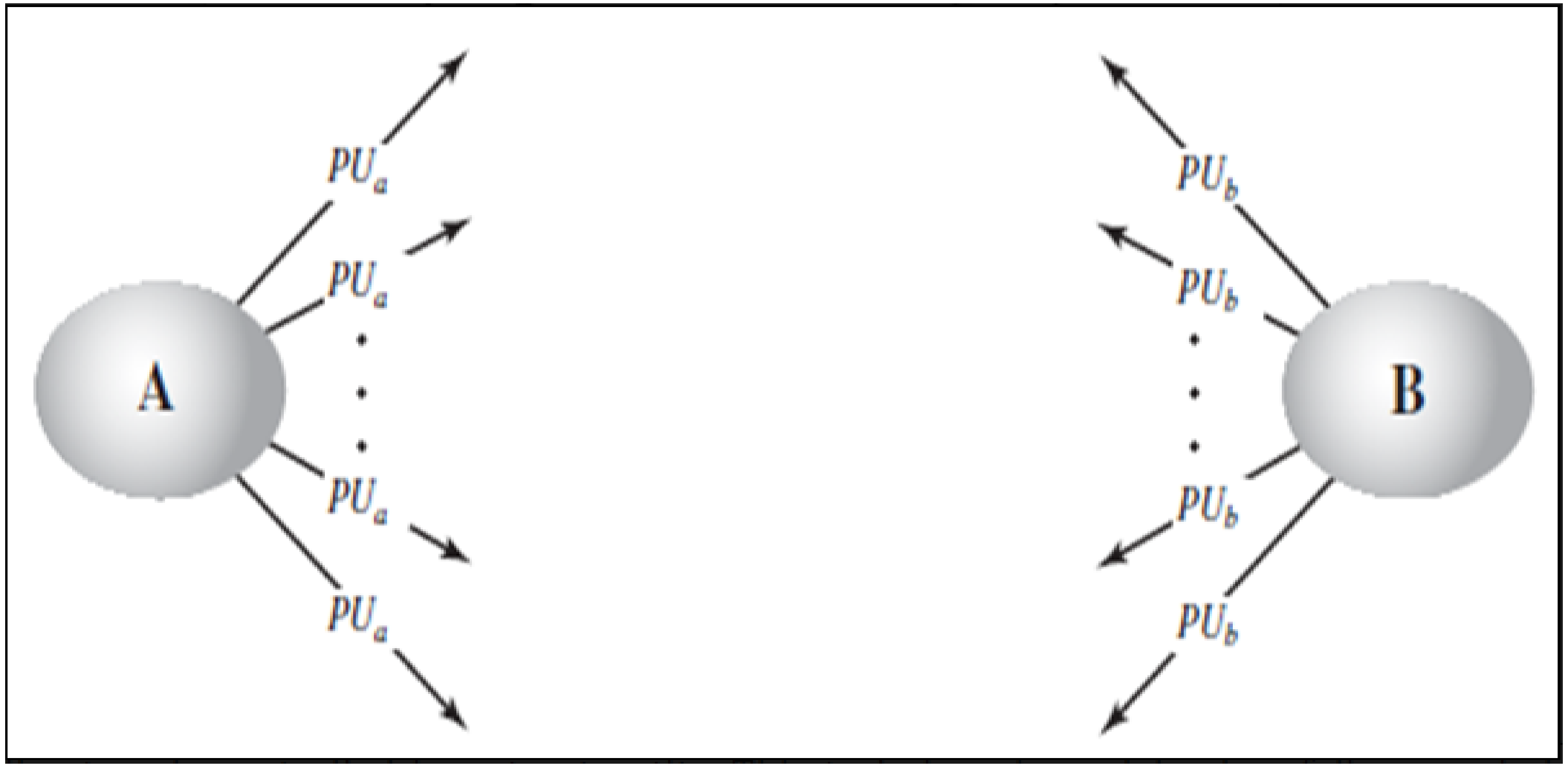
- This is freely distributed and available to anyone. It's used for encryption and verifying digital signatures. When someone wants to send an encrypted message to the owner of the public key, they encrypt the message using this public key. The public key can also be used to verify signatures created by the corresponding private key.

There are four methods of public key distribution:

- Public announcement,
- Publicly Available Directory,
- Public Key Authority and
- Public Key Certificates.

PUBLIC ANNOUNCEMENT

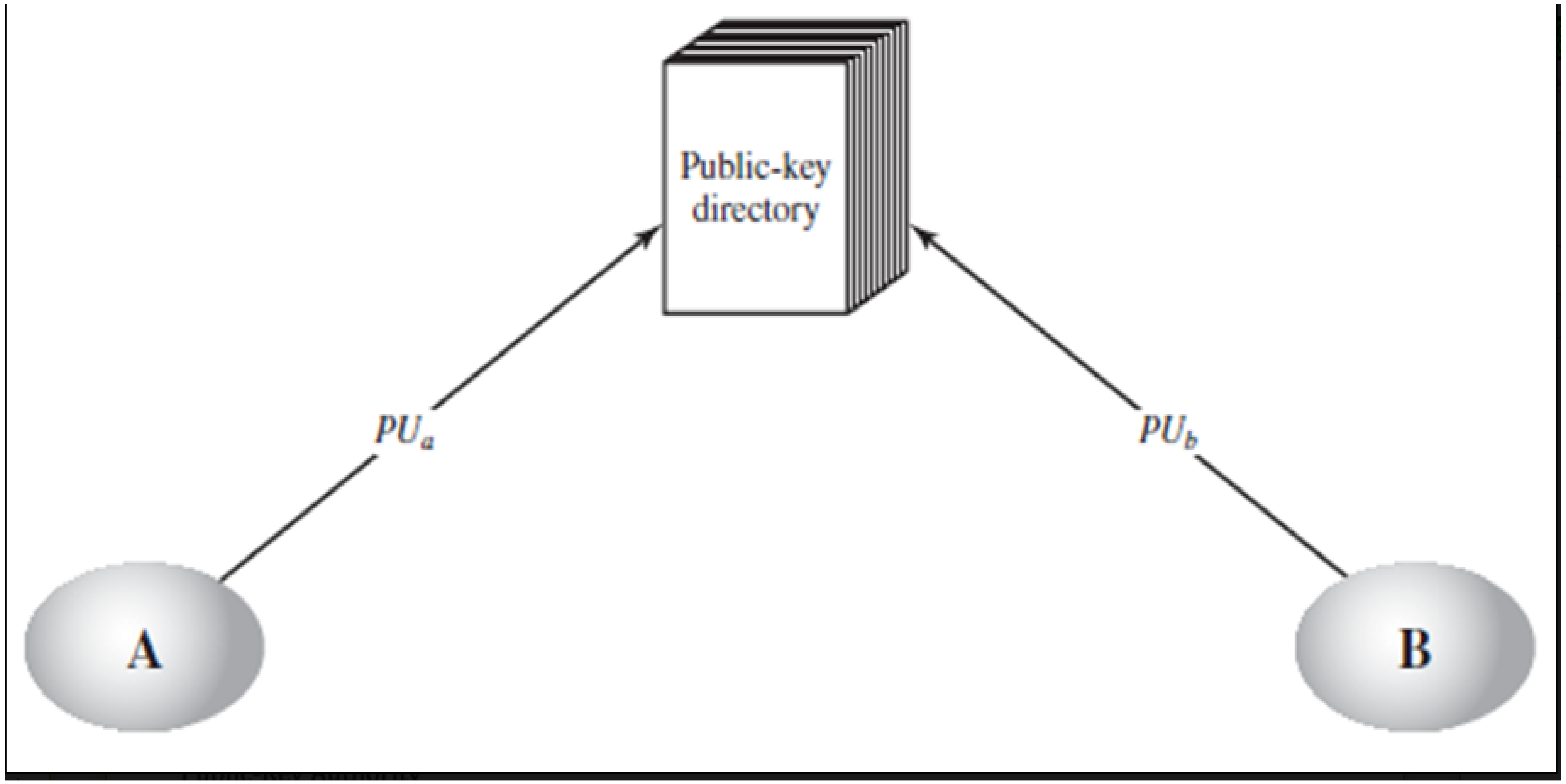
- In a public key cryptography, such as RSA, any user can send his/her key to any other user or broadcast it to the group.
- This type of approach is having a biggest drawback. Any user can pretend to be a user A and send a public to another user or broadcast it. Until user A has got this thing and alerts to other user, a pretender is able to read all encrypted message of other users.



Uncontrolled public key distribution

PUBLICLY AVAILABLE

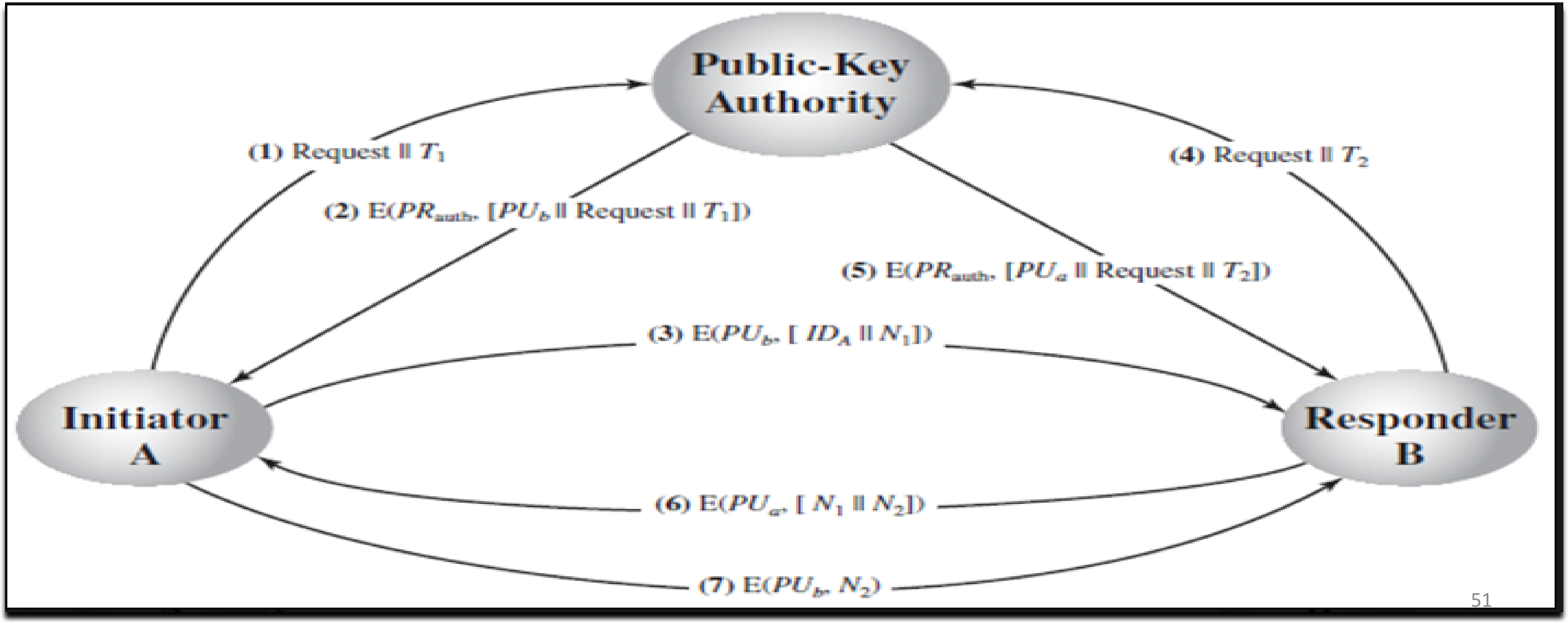
- A dynamic publicly available directory is used to achieve the security.
- Used to Maintenance and distribution of public directory is controlled by a trust entity.
- A trusted entity maintains a directory for each user as <name, public key>.
- Each user has to register a public key with the directory.
- A user can replace the existing key with a new one at any time for any particular reason.
- It is more secure than public announcement but still having some weakness. A hacker can obtain the private key of directory or temper with the information kept by directory.



Public Key Publication

PUBLIC KEY AUTHORITY

- It gives stronger security. As shown in figure a central authority keeps a dynamic directory of public keys of all users. Additionally, each user knows the public key of authority.

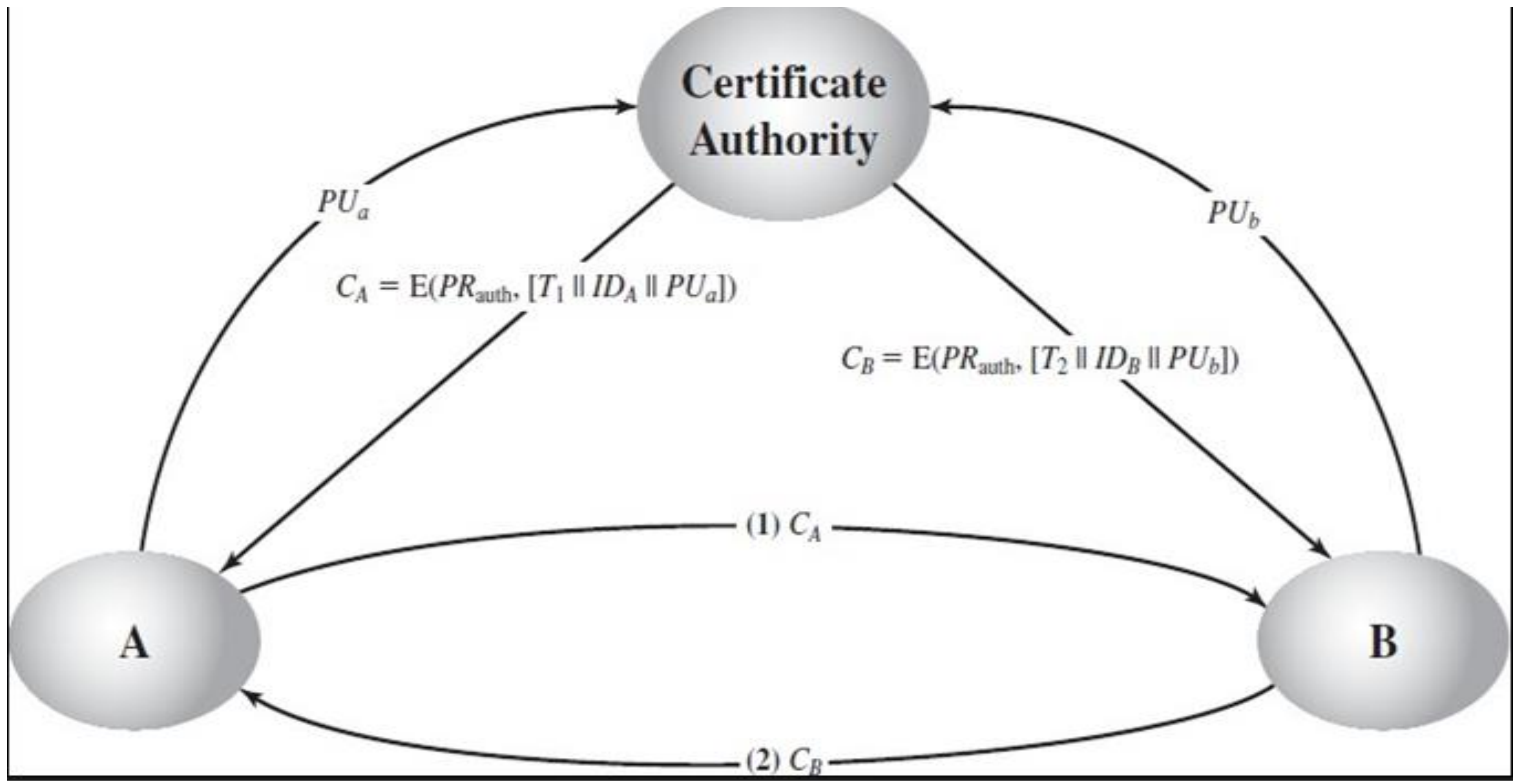

Public-Key Authority
(1) Request ‖ T_1
(2) E(PR_{auth}, [PU_b ‖ Request ‖ T_1])
(4) Request ‖ T_2
(5) E(PR_{auth}, [PU_a ‖ Request ‖ T_2])
(3) E(PU_b, [ID_A ‖ N_1])
Initiator A
Responder B
(6) E(PU_a, [N_1 ‖ N_2])
(7) E(PU_b, N_2)

- **Step – 1:** A sends a time stamped message to the public-key authority containing a request for the current public key of B.
- **Step – 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 used 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 given so A can determine that this is not an old message from the authority containing a key other than B’s current public key.

- **Step – 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.
- **Step – 4 & 5:** B retrieves A’s public key from the authority in the same manner as A retrieved B’s public key.
- **Step – 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 in message (6) assures A that the correspondent is B.
- **Step – 7:** A returns N_2, which is encrypted using B’s public key, to assure B that its correspondent is A.

PUBLIC KEY CERTIFICATES

- The directory of names and public keys maintained by the authority is vulnerable to tampering.
- An alternative approach, first suggested by Kohn Felder, is to use certificates.
- In essence, a certificate consists of a public key, an identifier of the key owner, and the whole block signed by a trusted third party. Typically, the third party is a certificate authority, such as a government agency or a financial institution that is trusted by the user community.
- A user can present his or her public key to the authority in a secure manner and obtain a certificate. The user can then publish the certificate Anyone needing this user’s public key can obtain the certificate and verify that it is valid by way of the attached trusted signature.
- A participant can also convey its key information to another by transmitting its certificate. Other participants can verify that the certificate was created by the authority. Below diagram shows the distribution of public keys using public key certificates.


Certificate Authority
PU_a
PU_b
$C_A = E(PR_{auth}, [T_1 \| ID_A \| PU_a])$
$C_B = E(PR_{auth}, [T_2 \| ID_B \| PU_b])$
(1) C_A
(2) C_B
A
B

Exchange of Public key Certificates

We can place the following requirements on this scheme:

- **Step – 1:** Any participant can read a certificate to determine the name and public key of the certificate's owner.
- **Step – 2:** Any participant can verify that the certificate originated from the certificate authority and is not counterfeit.
- **Step – 3:** Only the certificate authority can create and update certificates.
- **Step – 4:** Any participant can verify the certificate.

X.509 CERTIFICATE

X.509 CERTIFICATE INTRODUCTION :

- X.509 certificates are digital documents that are used to verify the identity of individuals, organizations, or devices over the internet.
- X.509 certificate acts like a digital identity card that enables secure communication and transaction between two parties
- The X.509 certificate is defined by the International Telecommunication Union's Telecommunication Standardization Sector (ITU-T).
- The first X.509 certificates were issued in 1988 as part of the ITU-T and the X.500 directory services standard. The current version, version 9, was defined in October 2019.
- The X.509 standard is based on Abstract Syntax Notation One, an interface description language.
- The use of X.509 certificates ensures that the communication is encrypted and authenticated, thereby providing a high level of security for online transactions.

•The public key is used to encrypt messages, and the digital signature is used to verify that the message was sent by the holder of the private key associated with the public key.

How do get an x.509 Certificate :

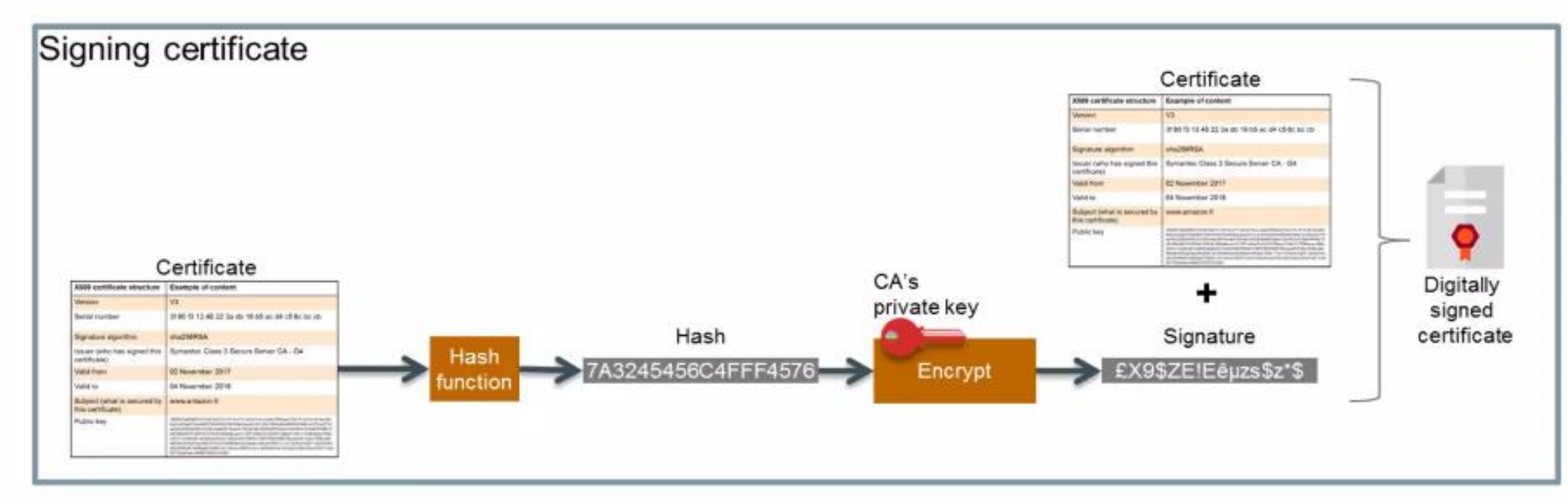
- Generate a public-private key pair
- Create a certificate request
- Submit the certificate request
- Verify your identity
- Receive the X.509 Certificate
- Install the certificate

X509 certificate structure	Example of content
Version	V3
Serial number	3f 80 f3 13 48 22 3a db 19 b5 ac d4 c5 6c bc cb
Signature algorithm	sha256RSA
Issuer (who has signed this certificate)	Symantec Class 3 Secure Server CA - G4
Valid from	02 November 2017
Valid to	04 November 2018
Subject (what is secured by this certificate)	www.amazon.fr
Public key	3082010a0282010100c5dd31c5414cd717a32d14ccca4a7ff40ead12d17cc613c453ac26c befcd22ae670ea69df1604405d5056dfae3eeb231c3d4106eb844b6fa500d81e7d5ba2316 aa76c0282b403533c9b5dfab9679ceb47620dcb622890df650a4c53b4fb5c559a94999b75 e8186ed94753292e7032de380a8acee131221a9ee2c5246578aee154b1c7396bbec494a cb57c1d4eb4fe1d4d2ba5e042140dd526f768bfcf109769620f8782eeb2fcf3a5c76f6ca0b 96209c625a25ea36d5197a53f3fd60bc02e0aab129a3818fa117a77e5fadc4d211fa26546c dfbd560be91e6f8da07b8661e214e5a18061ccb1c445d85ced70d3d2cb94c392e53817cd9 f8772baf4e4c4968f70203010001

Certification Authority(CA):

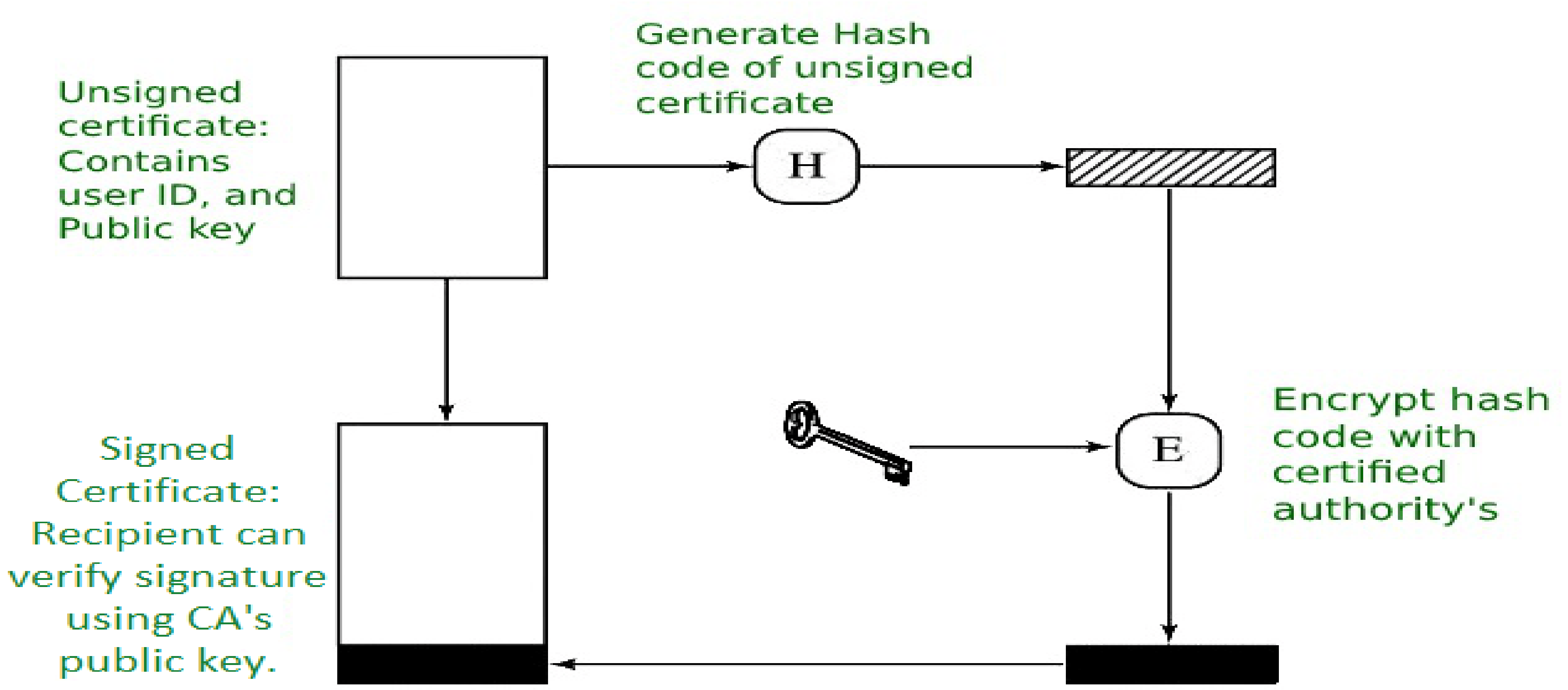
Trusted third party issuing certificates

- *Certificate Authority(CA) issues certificates*
- *Certificate is signed with the private key of the CA*
- *CA's private key must be very private, it is the basis of all trust for issued certificates*



X509 certificate structure	Example of content
Version	V3
Serial number	31 80 13 13 48 22 3a db 19 b5 ac d4 c5 8c bc cb
Signature algorithm	sha256RSA
Issuer (who has signed this certificate)	Symantec Class 3 Secure Server CA - G4
Valid from	02 November 2017
Valid to	04 November 2018
Subject (what is secured by this certificate)	www.amazon.fr
Public key	[illegible]

Block diagram :


Generate Hash code of unsigned certificate
Unsigned certificate: Contains user ID, and Public key
H
E
Encrypt hash code with certified authority's
Signed Certificate: Recipient can verify signature using CA's public key.

How does an X.509 Certificate work?

- **Certificate Verification**: When the certificate holder attempts to establish a secure communication or transaction, the recipient verifies the certificate's authenticity by checking the digital signature of the CA and the validity period of the certificate.

- **Key Exchange**: Once the certificate is verified, the recipient encrypts a session key using the certificate holder's public key and sends it back to the certificate holder.

- **Secure Communication**: The certificate holder uses their private key to decrypt the session key and establishes a secure communication session using the session key.

- **Version number:** It defines the X.509 version that concerns the certificate.
- **Serial number:** It is the unique number that the certified authority issues
- **Signature Algorithm Identifier:** This is the algorithm that is used for signing the certificate.
- **Issuer name:** Tells about the X.500 name of the certified authority which signed and created the certificate.
- **Period of Validity:** It defines the period for which the certificate is valid.
- **Subject Name:** Tells about the name of the user to whom this certificate has been issued
- **Extension block:** This field contains additional standard information.
- **Signature:** This field contains the hash code of all other fields which is encrypted by the certified authority private key.

Applications of X.509 certificates:

- **SSL/TSL Encryption : X**.509 certificates establish secure communication between a web browser and a web server. When you visit a website with "https" in the URL, the website uses SSL/TLS encryption, and X.509 certificates are used to authenticate the website and encrypt the communication between the browser and the server.

- **Code Signing**: X.509 certificates are used to digitally sign software code, ensuring that the code is from a trusted source and has not been tampered with.

- **Email Security**: X.509 certificates are used to provide secure email communication by encrypting email messages and verifying the identity of the sender.

- **VPN Authentication**: X.509 certificates are used to authenticate users in virtual private network (VPN) connections.

- **Document Signing**: X.509 certificates are used to sign electronic documents, providing assurance that the document is from a trusted source and has not been modified since it was signed.

- **User Authentication**: X.509 certificates are used to authenticate users in various applications, including online banking, e-commerce, and other online services.

PUBLIC KEY INFRASTRUCTURE

WHAT IS PUBLIC KEY INFRASTRUCTURE(PKI)?

- Public Key Infrastructure (PKI) is a system used in cryptography to manage the distribution and verification of digital certificates, which are used to verify the authenticity of entities such as people, devices, or websites.
- Public key infrastructure protects and authenticates communications between servers and users, such as between your website (hosted on your web server) and your clients the user trying to connect through their browser.
- It can also be used for secure communications within an organization to ensure that the messages are only visible to the sender and recipient, and they have not been tampered with in transit.

WHAT IS PKI USED FOR?

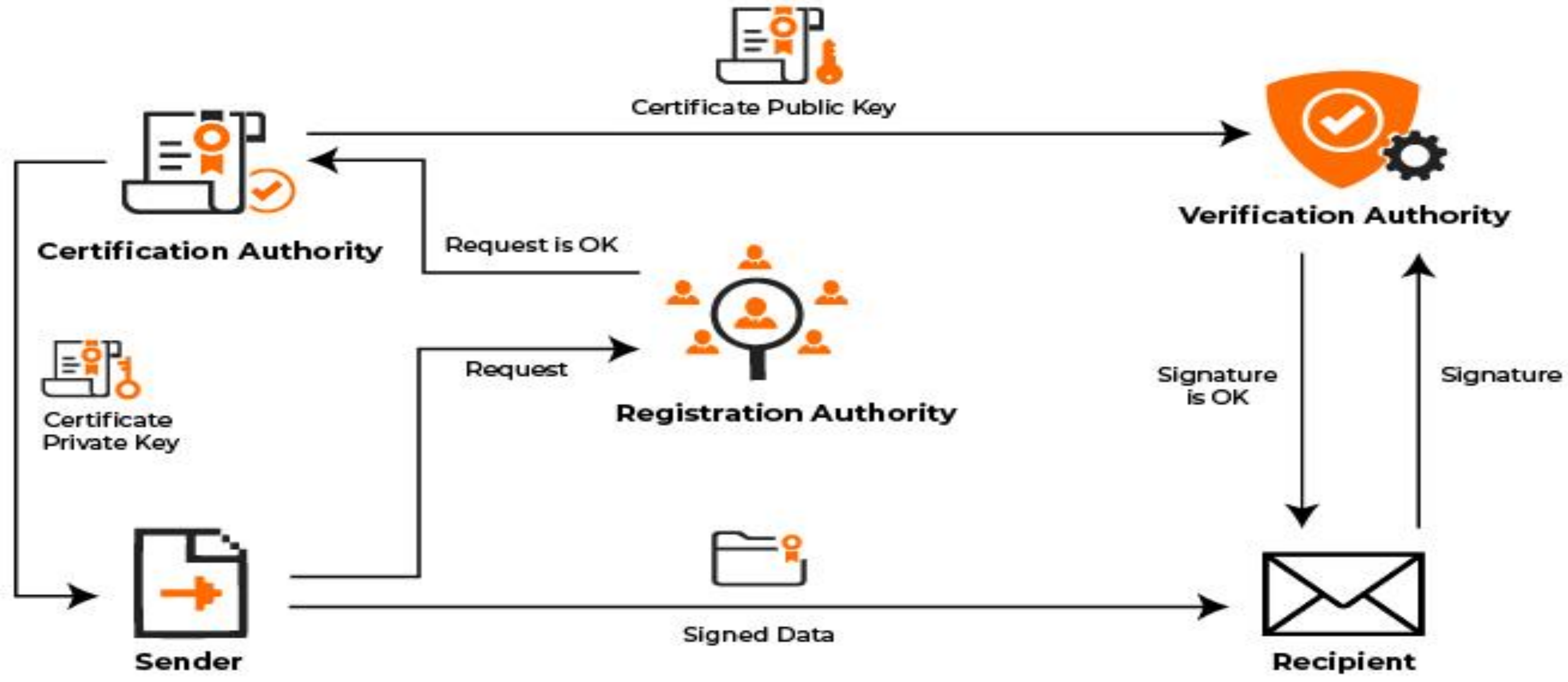
- PKI is responsible for making online interactions more secure, and it does this by:
    1. Establishing the identity of end points on a network
    2. Encrypting the flow of data via the network's communication channels
- It does this by using private keys and public keys for encryption and decryption respectively, which are facilitated in turn by digital certificates.

The workings of PKI

Components of a Public Key Infrastructure

- Public Key Infrastructure involve the participation of some or all of the below entities:
  - Public and Private Keys
  - Public Key Certificates
  - Certificate Repository
  - Registration Authority (RA)
  - Certificate Authority (CA)

PUBLIC KEY INFRASTRUCTURE(PKI)



- **Public and Private Keys:**
  - The single most important component(s) of PKI, public and private keys are used to encrypt and decrypt the information transmitted over the web, ensuring that the sending and receiving party are the only ones privy to that information. Public key information is available openly online, but can only be effectively leveraged when the receiving party has an approved private key in order to decrypt a message.
- **Public Key Certificates:**
  - Electronically signed documents that verify ownership of a public key. They are as important as keys, as they act as proof that a key-holder is legitimate. They are issued by Certificate Authorities.

- **Certificate Repository:**

  An electronic, searchable storage facility for signed certificates with public keys that have been generated. It consists of important certificate information, such as certificate validity details, revocation lists, and root certificates. They are often equipped with LDAP (Lightweight Directory Access Protocol), an online directory service where entries are classified and indexed.

- **Registration Authority (RA):**

  Assists the PKI cycle by verifying that the body requesting a certificate is legitimate. Once the verification is complete, it carries out the request by allowing the request to reach the CA, who uses a certificate server to execute it.

- **Certificate Authority (CA):**

  A trusted body which enables organizations to get themselves verified as public key holders. It does this by verifying a requesting organization, and generating an electronic document called a digital certificate which also holds the public key. It then signs the certificate with its own private key, which acts as a seal of approval that it is trusted by a Certificate Authority.