---

1. Basics

> **Network Security** → Practices, technologies, policies, and procedures to protect the integrity, confidentiality, and availability of computer networks and data. It involves safeguarding the network infrastructure from unauthorized access, misuse, destruction, modification, or disclosure.

> **Cybersecurity** → Protecting systems, data, devices, and software from cyber threats. Network Security is a subset of cybersecurity.

Information Security	Network Security
Broad - Protecting all information and the systems that handle it.	Specific - protecting network infrastructure and data in transit.
Confidentiality, Integrity, and Availability (CIA).	Protecting data during transmission and network devices (routers, switches, servers).

Information Security	Network Security
Overall protection of information from unauthorized access, use, disclosure, disruption, modification, or destruction.	Preventing unauthorized access, misuse, or attacks on the network and data moving across it.
The overarching goal; network security is a component of it.	A subset of cybersecurity and a key component of information security.

1.1. The CIA Triad

These are the 3 main objectives of Computer Security.

1. **Confidentiality** → Assures that private or confidential information is not made available or disclosed to *unauthorized individuals*. **Privacy** is also included which is that people must have control over what information related to them may be collected, stored, and disclosed, and by whom.
2. **Integrity** → Assures that information and programs are changed only in a *specified and authorized manner*. This also includes **System Integrity** (a system performs its intended function in an *unimpaired manner*, free from deliberate or inadvertent unauthorized manipulation).
3. **Availability** → Assures that systems work promptly and service is not denied to *authorized users*.

Other Possible Objectives → *Authenticity* (Verifying that users are who they say they are) and *Accountability* (Actions can be traced uniquely to an entity).

1.2. Security Attacks

> **Security Attack** → Any action that *compromises the security of information* owned by an organization.
> **Security Mechanism** → A process made to *detect, prevent, or recover* from a security attack.
> **Security Service** → A service that *improves the security* of data processing systems and information transfers.

⚠ Passive Attacks

Attempts to learn or make use of information from the system but do not affect system resources. They are difficult to identify and prevent because the attacker is not changing anything. **Types:**

- *Release of message contents* → Eavesdropping on or monitoring transmissions.

- *Traffic analysis* → Monitoring transmissions to obtain information about the traffic flow (source, destination, frequency, length).

Active Attacks

Attempts to change/alter system resources or affect their operation. They involve some modification of the data stream or the creation of a false stream. **Types**:

- *Masquerade* → Takes place when one entity pretends to be a different entity. Usually includes another form of active attack.
- *Replay* → Involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect.
- *Modification of messages* → Some portion of a legitimate message is altered, or messages are delayed or reordered to produce an unauthorized effect.
- *Denial of service* → Prevents or inhibits the normal use or management of communications facilities.

Phishing Attack

A type of online fraud and cyberattack where attackers "impersonate" as other organizations or individuals. It is done by many methods like fake emails, messages or websites, which fool the user into performing an action. Users end up revealing sensitive information (passwords, credit card numbers etc.) by clicking on fake "links" which can also infect their system with viruses.

1.3. Security Mechanisms

1. **Authentication** → Makes sure that a communication is authentic and verifying the identities of involved users.
2. **Access Control** → Limit the access to host systems and applications.
3. **Data Confidentiality** → Protecting transmitted data from passive attacks.
4. **Data Integrity** → Makes sure that the information is changed only in an authorized manner.
5. **Nonrepudiation** → Prevents either sender or receiver from denying a transmitted message.
6. **Availability** → Makes sure that systems work properly and service is not denied to users.

2. Malicious Software

> **Malware** → Software intentionally designed to harm computer systems by causing undesired actions. It spreads through email attachments, infected floppy disks, downloading/exchanging corrupted files, and embedded computer games.

Main categories of Malware are:

1. **Virus** → Self-replicating code that attaches to other programs, spreading malicious code by modifying them. It propagates itself and carries a payload for both replication and covert tasks. Viruses can compromise data availability, integrity, and confidentiality (e.g., by destroying, changing, or altering data), impact system performance by occupying disk space/memory and consuming CPU time.
    - Viruses can reside in boot sectors, memory, disk (applications/data), libraries, compilers, debuggers, and even anti-virus programs. Effective Viruses are hard to detect and destroy, spread widely, are easy to create and re-infect, and ideally platform-independent.
    - Virus Detection involves checksums using cryptographic hash functions. Virus Identification relies on the fact that each virus is a unique program with unique object code that inserts itself in a deterministic manner, providing a signature based on its object code pattern and insertion location.
    - **Types of Viruses**:
        1. *Boot Sector Virus* → It infects boot sector of systems and stays there. Activates when the machine boots.
        2. *File Virus* → It infects files of a program. Activates when that program is run.
        3. *Email Virus* → It is spread by infected attachments (macro viruses) triggered upon opening or even viewing (with scripting features).
2. **Worm** → A stand-alone program that self-propagates through a network without infecting host programs. Attacks by worms can involve deleting files, communicating information back to the attacker (e.g., passwords), and disrupting normal system operation, leading to DoS attacks due to re-infecting systems. Worms may also carry viruses.
    - *Examples*: 1988 Morris Internet Worm, 2004 MyDoom/Sasser and 2003 SQL Slammer.
3. **Trojan Horse** → A program appearing legitimate but with hidden malicious functions performed upon execution. It can propagate viruses/worms, install backdoors, or destroy data.
    - **Types of Trojan Horses**:
        1. *Remote-Access Trojan* → Lets the hacker completely control your computer from somewhere else.

    2. *Data-Sending Trojan* → Sends your information back to the hacker (like keyloggers that record everything you type).
    3. *Destructive Trojan* → Only meant to delete and ruin your files and is often hard for virus scanners to find.
    4. *Denial-of-service (DoS) attack Trojan* → Uses the power of many infected computers to attack another computer, overwhelming it so it stops working.
4. **Logic Bomb** → Code embedded inside in a program that activates when some conditions are met (like the presence/absence of file, date/time, user). When triggered, typically damages the system (modify/delete files/disks).
5. **Trapdoor** → A secret entry point bypassing normal security (authentication), potentially for troubleshooting but exploitable by attackers. They are hard to block and require careful software management.
6. **Spyware** → Programs that secretly look at the files on your computer and send this information to a specific place. They're used to watch how people use software, but they can also be used to plan attacks. They can steal your personal information like account numbers and passwords, watch what you do online, and even install other bad software on your computer.
    - *Examples*: CoolWebSearch, Gator, 180search Assistant, ISTbar/AUpdate and Transponder.
7. **Hoax** → Fake warnings about viruses that are spreading. They often come as chain emails that seem important, so people forward them to everyone they know. Sending lots of these messages can clog up the internet and slow things down. These fake warnings try to scare people and spread false rumors.

Phases of a Virus

Viruses go through these 4 phases:

1. **Dormant** (Waiting for trigger)
2. **Triggering** (By an Event to execute code)
3. **Execution** (Running code)
4. **Propagation** (Spreading to programs/disks)

3. Firewalls

> A software or hardware device that monitors and controls data entering and leaving a network. It acts as a security barrier between a trusted internal network and an untrusted external one, such as the Internet. By applying predefined rules, it decides whether to allow, block, encrypt, or log network traffic. A firewall also serves as a single point where security policies can be enforced, audits conducted, and additional functions like Network Address Translation (NAT) and traffic logging managed.

Limitations of a Firewall → It cannot protect against:

1. Attacks bypassing it like trusted organizations and trusted services (SSL/SSH).
2. Internal threats (malicious employees).
3. Viruses and Trojans from reaching the machine via email, file sharing, or direct download nor their transfer.

Firewall Characteristics → These are the design goals:

1. All network traffic (inbound and outbound) must pass through the firewall.
2. Only authorized traffic, as defined by the local security policy, will be allowed to pass.
3. The firewall itself must be immune to penetration (achieved through trusted system use).

Techniques to Enforce Security by Firewall

Firewalls use several techniques to control access:

1. **Service Control** (*Which Internet services can be accessed?*): Determines which Internet services (inbound or outbound) are allowed, filtering based on IP address, protocol, or port number, or using proxy software.
2. **Direction Control** (*Which direction, a particular service requests can be accessed?*): Determines the direction in which service requests can be initiated and flow through the firewall.
3. **User Control** (*Which user is attempting to access what service?*): Controls access to services based on the user attempting to access them, typically for internal users but can be applied to external incoming traffic.
4. **Behavior Control** (*How a particular service is being used?*): Controls how particular services are used (like filtering spam email).

Positives of a Firewall	Negatives of a Firewall
User authentication (Require user login, allowing control and tracking).	**Traffic bottlenecks** (Forcing all traffic through can cause congestion).
Auditing and logging (Can keep and analyze activity information).	**Single point of failure** (Incorrect configuration or unavailability can block all traffic).
Anti-Spoofing (Detecting when the source of network traffic is being "spoofed").	**Increased management responsibilities** (Adds complexity to network management and troubleshooting).

Hardware vs Software Firewall

- **Hardware Firewalls** → Protect an entire network, implemented on the router level, usually more expensive, harder to configure.
- **Software Firewalls** → Protect a single computer, usually less expensive, easier to configure.

3.1. Firewall Types

Based on how the firewall works, we can divide them in these categories:

Type	Layer	Pros	Cons	How It Works
Packet Filtering (Stateless)	3, 4	Fast, simple, low cost	No context, hard for complex rules	Inspects each incoming and outgoing packet individually. Decisions are based on predefined rules involving IP address, port number, protocol, and network interface. It does not keep track of previous packets, so each one is evaluated on its own.
Stateful Inspection	3, 4	More secure, still fast	Harder to configure, no user authentication	Tracks the state of active connections (e.g., TCP handshakes). Maintains a state table of outbound requests and allows inbound traffic only if it matches an expected response in that table. This blocks unsolicited or spoofed traffic.

Type	Layer	Pros	Cons	How It Works
Application-Level Gateway (Proxy)	7	High control and visibility	Slower, separate proxy needed per service	Acts as an intermediary for specific applications like web or email. The user connects to the proxy, which then makes the request on the user's behalf. The proxy checks, filters, and may modify traffic before forwarding. It fully understands and enforces protocol rules.
Circuit-Level Gateway	4, 5	More secure than packet filters	Limited to session-level, no deep inspection	Monitors TCP or UDP session establishment. Once a session is validated, it relays traffic without inspecting packet contents. It maintains a virtual circuit for each allowed session and is commonly used with SOCKS proxies.

Bastion Host

A specially secured computer that provides access to certain services, like proxies or gateways. It runs only essential functions, uses a hardened operating system, and includes extra authentication to resist attacks. It's designed to enforce security policies and may connect to multiple networks while acting as a trusted access point.

4. Cryptography

> The use of mathematical algorithms to transform data into a form that is not understandable and then recover that data (depends on the algorithm).

These algorithms are divided based on the:

1. Performed Operation ⟶ Substitution (replace) vs Transposition (shuffle).
2. Way of Processing ⟶ Stream (one bit at a time) vs Block (entire chunks at a time).
3. Keys for Encryption/Decryption ⟶ Symmetric (same key) vs Asymmetric (different keys).

Cryptanalysis ⟶ Process of attempting to discover the plaintext or key.

> ⓘ **Computationally Secure Cipher**
>
> A cipher is considered **computationally secure**, If the ciphertext generated by it meets one or both of these criteria:
>
> - The *cost* of breaking the cipher exceeds the value of the encrypted information.
> - The *time* required to break the cipher exceeds the useful lifetime of the information.

A **brute-force approach** of trying every possible key is often considered:

- `56-bit` key (used by DES) takes about 10 hours to crack.
- `128-bit` key (used by AES) take over 10^{18} years to crack.

This difference suggests that a 128-bit key algorithm is unbreakable by brute force.

4.1. Ciphers

1. **Substitution Ciphers** ⟶ Mapping individual letters of a plaintext alphabet to a particular letter of the ciphertext alphabet.
   - *Atbash Cipher* ⟶ The simplest and earliest cipher.
   - *Caesar Cipher* ⟶ More secure than Atbash but still easy to break (e.g., shift of 3).
   - *Vigenère Cipher* ⟶ A polyalphabetic cipher that uses a keyword repeated to form a keystream. Encryption and decryption use a "tabula recta" based on plaintext/ciphertext and keystream letters.

2. **Transposition Ciphers** ⟶ Shuffles the letters in the plaintext around to make the ciphertext. Examples:
    - *Rail Fence Cipher* ⟶ Writes message on alternate lines (zigzag) and reads off each line. Key is the number of rows. Encryption and Decryption processes described.
    - *Columnar Cipher* ⟶ Encryption involves writing plaintext in a grid based on a keyword, then reading columns based on the alphabetical order of the keyword. Decryption reconstructs the grid.

Block Ciphers (not important) ⟶ Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB) , Counter (CTR)

4.2. Symmetric Encryption

> Uses the same key for both encryption and decryption. It is also known as conventional, secret-key, or single-key encryption. It Involves an encryption algorithm, a secret key shared by the sender and receiver, and the plaintext and ciphertext.
> **Examples** ⟶ *AES*, *DES* and *Triple DES* algorithms.

Link Encryption	End-to-End Encryption
Encryption happens at each communication link (between every switch/router).	Encryption is done by the sender and decrypted only by the final receiver.
Each link must decrypt and re-encrypt the data, exposing it briefly at each hop.	Data remains encrypted throughout the journey, only decrypted at the endpoint.
Encrypts the entire packet (headers + payload), hiding routing info from outsiders.	Only the message content is encrypted; headers stay visible for routing.
Requires many keys—each pair of nodes must share a unique key.	Requires only the sender and receiver to share a key.
Operates at lower OSI layers (1 or 2).	Operates at higher OSI layers (3, 4, 6, or 7).
Helps protect against **traffic monitoring between points** (e.g., routers, switches).	Helps protect **message content** and authenticate the sender.

- **Traffic analysis** observes who is communicating, when, and how often—useful for surveillance or business intelligence. Even with link encryption, total traffic volume and timing can still be seen. **Traffic padding** (sending fake data to hide real patterns) can reduce visibility of traffic flows but increases bandwidth usage.

- **Best practice** → Combine **link encryption** (to hide headers and traffic patterns between nodes) with **end-to-end encryption** (to protect content and ensure authenticity).

4.3. Asymmetric Encryption

> Uses different keys for encryption and decryption. It is also known public-key cryptography. It involves an encryption algorithm, a secret unique key that is not shared (private key), a secret key that is shared across the network (public key), and the plaintext and ciphertext. Depending on the application, the sender uses either their private key, the recipient's public key, or both to perform cryptographic functions.
> **Examples** → *RSA*, *Diffie-Hellman* and *ECC* algorithms.

5. Key Distribution

> **Key Distribution Scheme** → The means of exchanging keys between two parties. These keys are used for conventional encryption, and frequent key exchanges are desirable to limit the amount of compromised data. The strength of a cryptographic system relies on the key distribution mechanism.

5.1. Symmetric Key Distribution

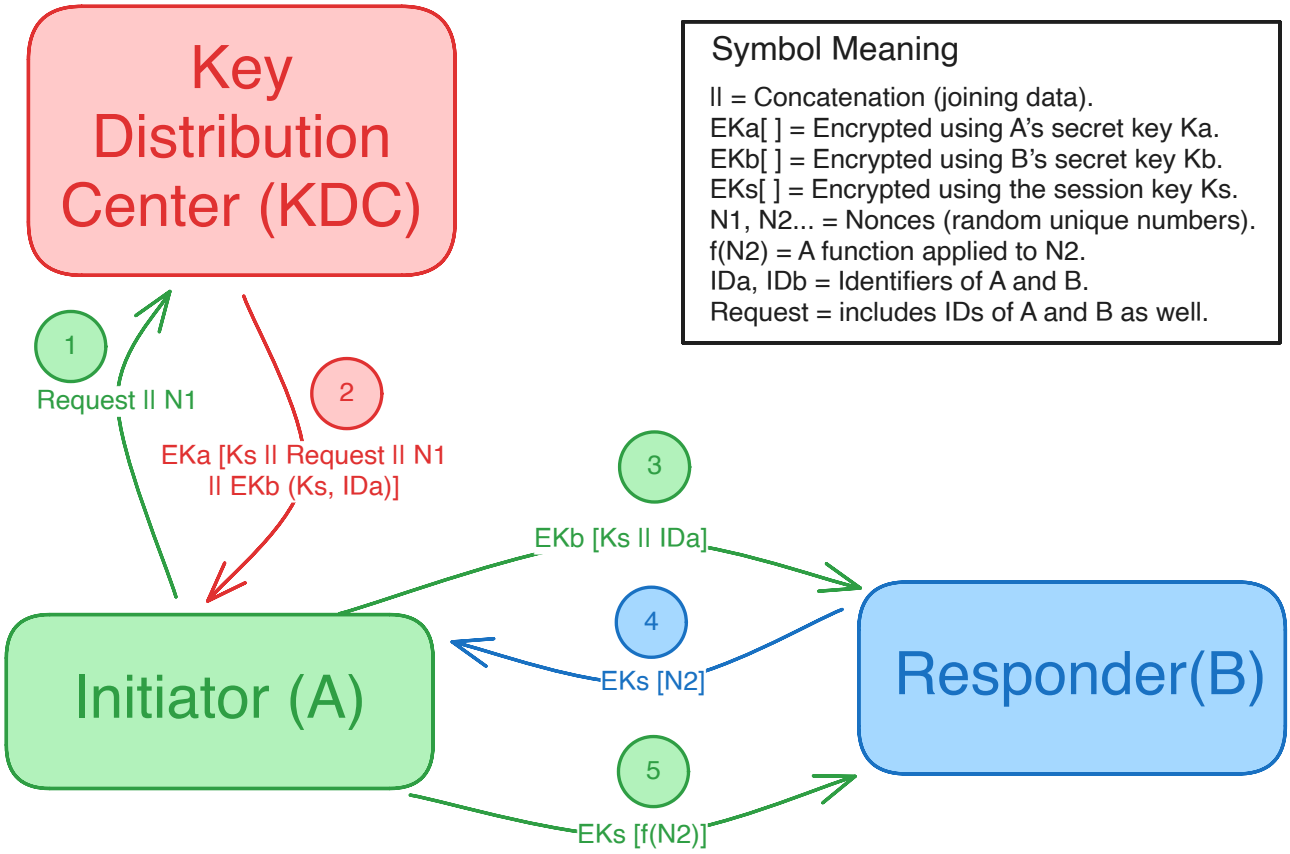
Symmetric schemes require both parties to share a common secret key. The challenge lies in securely distributing this key, and often, secure systems fail due to a break in the key distribution scheme.

Alternatives to Key Distribution

1. A can select a key and physically deliver it to B.
2. A third party can select and deliver the key to A and B.
3. If A and B have communicated previously, they can use a previous key to encrypt a new key.
4. If A and B have secure communications with a third party C, C can relay the key between A and B.

Key Distribution Scenario

Purpose: To securely distribute a session key (`KS`) between two parties (A and B) using a trusted third party — the KDC — and prevent attacks like replay attacks or impersonation.



What's Happening

1. **A → KDC:** A (Alice) sends a request to the Key Distribution Center (KDC) to communicate with B (Bob), including a nonce `N1` (a random value for freshness) and their identities.
2. **KDC → A:** The KDC responds with a session key `Ks`:
   - Encrypted with A's secret key `Ka`: includes `Ks`, the original request, and `N1`.
   - Also includes a **ticket for B**: `Ks` and A's ID, encrypted with B's secret key `Kb`.
3. **A → B:** A stores `Ks` and sends the ticket (encrypted with `Kb`) to B.
4. **B → A:** B decrypts the ticket to retrieve `Ks`, then sends a new nonce `N2` encrypted with `Ks` to verify that A knows the session key.
5. **A → B:** A processes `N2` (e.g., by incrementing it) and sends the result back encrypted with `Ks`, proving identity and that the message is fresh (not a replay).

5.2. Public-Key Management

> Public-key encryption involves two main aspects: the distribution of public keys and the use of public-key encryption to distribute secret keys.

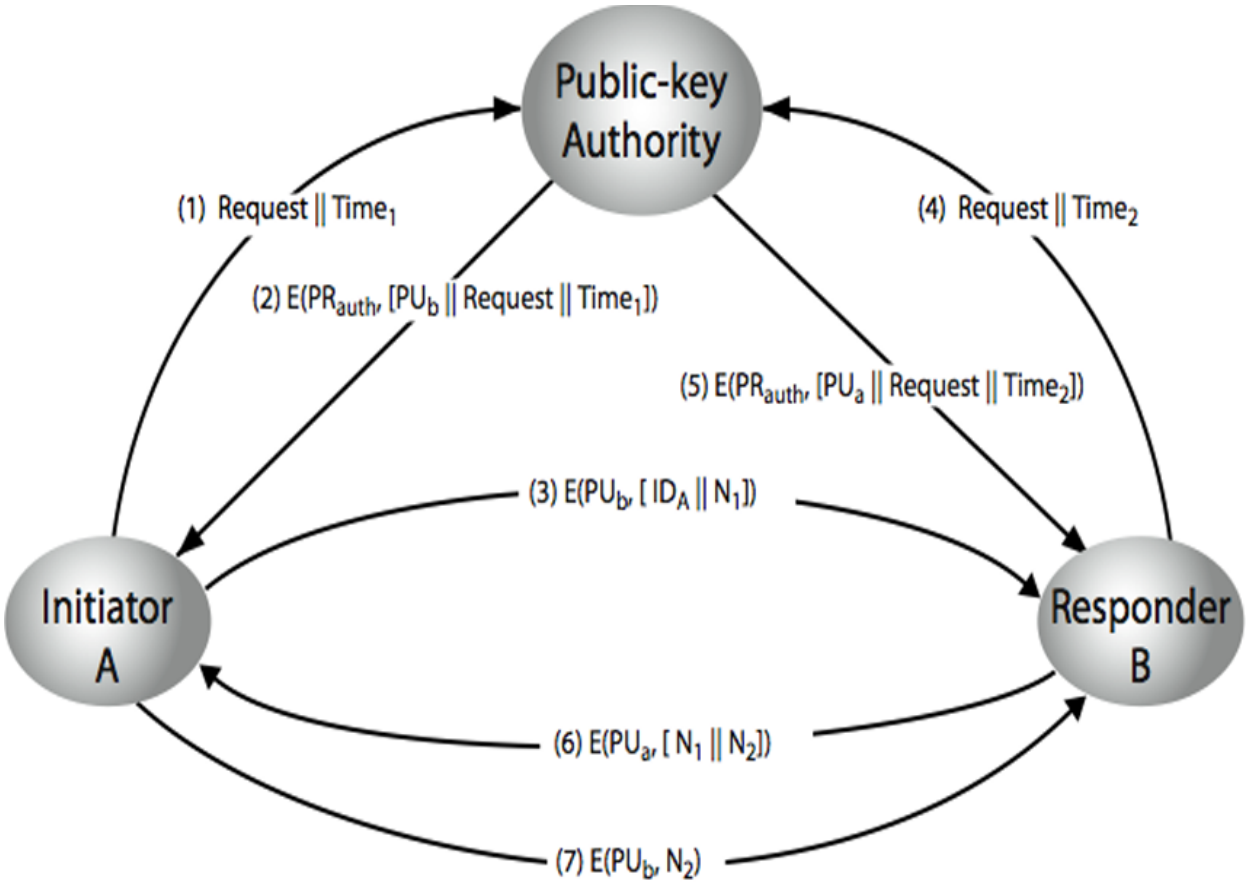
5.2.1. Distribution of Public Keys

Public keys can be distributed through:

1. **Public Announcement** → Users distribute their public keys to recipients or broadcast them to the community. The major weakness of this method is forgery,

as anyone can create a key claiming to be someone else and broadcast it, allowing them to masquerade as the claimed user until the forgery is discovered.

2. **Publicly Available Directory** → Greater security can be achieved by registering keys with a trusted public directory. It contains {name, public-key} entries, participants register securely with the directory and can replace their key at any time, the directory is periodically published and accessed electronically. However, this method is still vulnerable to tampering or forgery.
3. **Public-Key Authority** → A public-key authority improves security by tightening control over the distribution of keys from the directory. It has the properties of a directory and requires users to know the public key for the directory. Users interact with the directory to securely obtain any desired public key. This method requires real-time access to the directory when keys are needed.



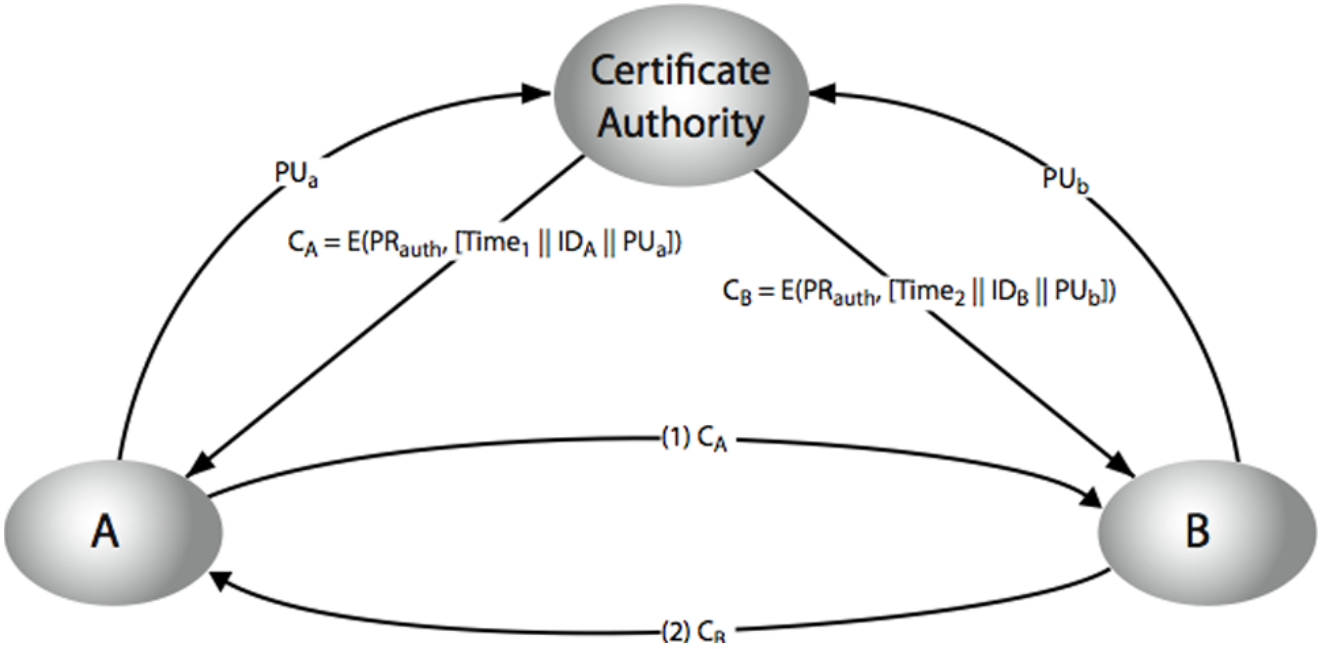
4. **Public-Key Certificates** → Certificates enable key exchange without requiring real-time access to a public-key authority. A certificate binds an identity to a public key, often including additional information like the period of validity and rights of use.

How to obtain a Certificate?

A **certificate** is a digital document that contains a public key and identity information, and it's digitally signed by a trusted Certificate Authority (CA). This signature ensures the certificate's authenticity and integrity. Anyone can verify a

certificate using the public key of the CA. A certificate for user A, signed by CA, is written as **CA⟨A⟩**.

To get a certificate, a user simply requests it from a CA. The CA verifies the user's identity and then issues the signed certificate. Only the CA can create or modify certificates, and because they're digitally signed, they cannot be forged.



X.509 Certificates

The X.509 standard is used to format public-key certificates for network security like SSL and IP security. These certificates use public-key cryptography, often with RSA, and are issued by a trusted Certification Authority (CA). They include key details such as version, serial number, issuer and subject names, validity period, public key info, and a digital signature from the CA, ensuring secure and trusted communication.

× Certificate Revocation

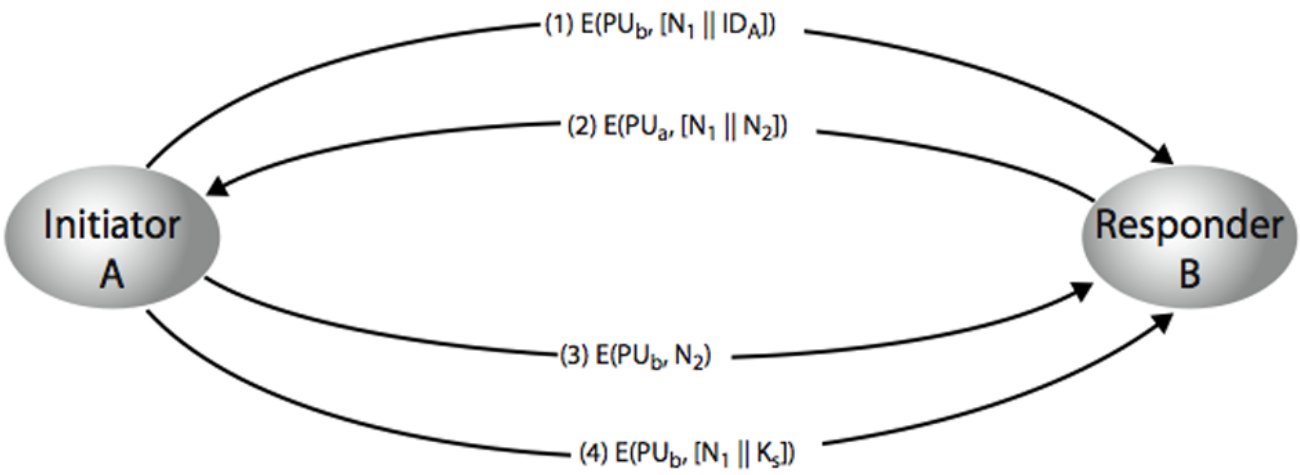
Certificates have a period of validity but may need to be revoked before expiry due to reasons such as:

1. The user's private key being compromised.
2. The user no longer being certified by the CA.
3. The CA's certificate being compromised.

CAs maintain a list of revoked certificates called the Certificate Revocation List (CRL), and users should check certificates against the CA's CRL.

5.2.2. Public-Key Distribution of Secret Keys

> Public-key cryptography can be used to obtain public keys for secrecy or authentication. However, because public-key algorithms are slow, private-key encryption is typically preferred to protect message contents, necessitating the use of a session key.



Hybrid Key Distribution

Combines the use of a private-key KDC, which shares a secret master key with each user and distributes session keys using the master key, with public-key cryptography, which is used to distribute the master keys, especially useful for widely distributed users.

5.3. Key Renewal

Keys should be renewed regularly because having too many encrypted messages with the same key can make attacks easier. How often keys are renewed depends on the algorithm, data volume, and time, as breaking keys takes time. Regular renewal limits damage if a key is compromised without detection. Protocols like SSL/TLS support secret key renewal.

The **key life-cycle** includes three main stages: generating the key, securely storing and using it, and securely destroying it so it can't be recovered. **Key Types**:

1. **Session Key** → Used once per session to encrypt data, then destroyed afterward.
2. **Permanent Key** → Used to securely distribute session keys between users.

Session keys should have limited lifetimes for better security. For connection-oriented protocols, the key lasts the whole connection; for connectionless, it lasts a set time.

Using a master key instead of session keys is risky, as a stolen master key compromises security. Different session keys serve different purposes:

- *Data encrypting key*: for general communication across the network.
- *PIN-encrypting key*: for PINs used in electronic funds transfers.
- *File encrypting key*: for encrypting files stored in a publicly accessible location.

Key Distribution Entities

- *Key Distribution Center* (KDC) issues one-time session keys to authorized users.
- *Front-end Processor* performs end-to-end encryption and gets session keys from the KDC for its host.

Large networks use a hierarchy of KDCs instead of a single one. Local KDCs handle users in their domain and communicate with other KDCs through a global KDC. These KDCs must trust each other for secure key sharing.

5.4. Digital Signature

It is like an electronic fingerprint (identity card) for digital data, using the sender's private key and asymmetric encryption. It attaches proof of origin and integrity to digital information. **RSA** is the most popular algorithm for digital signatures. It allows anyone with the sender's public key to verify:

1. Who sent it (authentication).
2. That the data hasn't been changed (integrity).
3. That the sender can't deny sending it (nonrepudiation).

How secure communication works

We have two clients, **A** and **B**. A wants to send a secure and authenticated message to B.

1. **Encrypt the Message** → A generates a random AES key (K) and encrypts the message with this key. Result: `Encrypted_Message`.
2. **Encrypt the AES Key** → A encrypts the AES Key (K) using B's public RSA key. Result: `Encrypted_AES_Key`.
3. **Create a Digital Signature (Optional but Important)** → A hashes the original message using a hash function like SHA-256. A encrypts the hash with A's private RSA key. Result: `Digital_Signature`.
4. **Send to B** → A sends `Encrypted_AES_Key`, `Encrypted_Message` and `Digital_Signature` to B.

What B Does:

1. **Decrypt AES Key** → B uses his private RSA key to decrypt `Encrypted_AES_Key` to get AES key (K).
2. **Decrypt the Message** → B uses AES key (K) to decrypt `Encrypted_Message`.
3. **Verify the Signature (if provided)** → B hashes the decrypted message. B uses A's public RSA key to decrypt the `Digital_Signature` and gets the original hash. If the hashes match → ✅ message is authentic and not tampered with.

THE END