

NETWOCK  
SECURITY

SECURITY

NETWOCK SECURITY



# Network Security

## Lecture 1

A/Prof. Fuchun Guo

# 1.1 Subject Introduction: About Me

Fuchun's Homepage

Home | 《Security Reduction》 | Wollongong | 卧报官网 |



Fuchun Guo

School of Computing and Information Technology  
Faculty of Engineering and Information Sciences  
University of Wollongong NSW 2522  
 Click Email Button  
Office: 3.202 Phone: +61 2 4221 4402



#### Book:

1. Fuchun Guo, Willy Susilo, Xiaofeng Chen, Peng Jiang, Jianchang Lai, Zhen Zhao. 《数字签名密史:从急需到有趣》 (Cryptologic Research History of Digital Signatures: From 1976 to 2020), 北京理工大学出版社 (2022)
2. Fuchun Guo, Willy Susilo, and Yi Mu. [Introduction to Security Reduction](#) Springer 2018.

#### ePrint:

3. Fuchun Guo, Willy Susilo, Xiaofeng Chen, Peng Jiang, Jianchang Lai, Zhen Zhao. ["Research Philosophy of Modern Cryptography"](#), ePrint:2023/715 (2023)

Dr Fuchun Guo

Associate Professor

ARC DECRA Fellow, ARC Future Fellow  
Institute of Cybersecurity and **Cryptology**  
School of Computing and Information Technology

Faculty of Engineering and Information Sciences

University of Wollongong NSW 2522

Office: 3.202

Phone: +61 2 4221 4402

Email: [fuchun@uow.edu.au](mailto:fuchun@uow.edu.au)

Web: [www.uow.edu.au/~fuchun](http://www.uow.edu.au/~fuchun)

# 1.1 Subject Introduction: Emails (1/2)

When sending emails to me:

- CC tutor if it is about assignment marking
- Subject (No blank)
- Introduce who you are (name&number)
- Briefly describe the background before your question (help me understand what you really want from me).
- Your question.....
- Start With: Hi/Hello/Dear Fuchun

# 1.1 Subject Introduction: Emails (2/2)

If you would like to ask a question related to knowledge:

- Clearly present your question or why you have this question.
- Try the best to give your answers and thinking.
- Correcting your thinking by me >> Delivering Knowledge

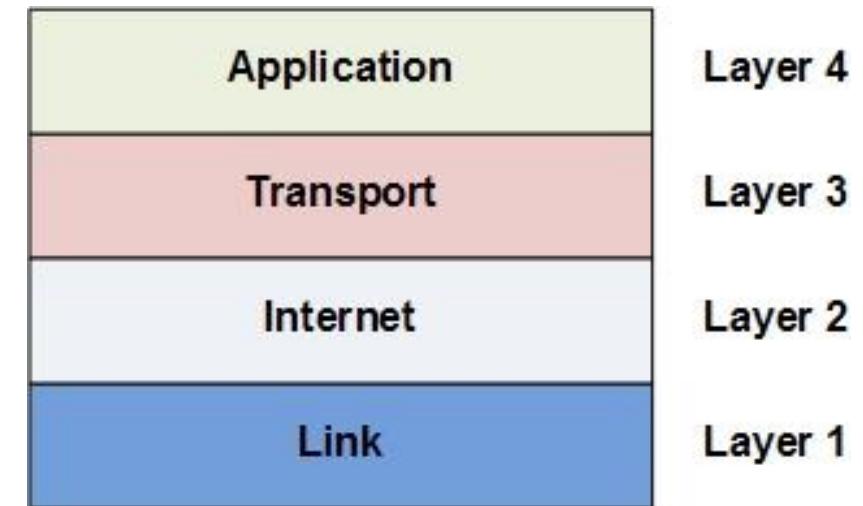
# 1.1 Subject Introduction: Class

- Start 5mins later than schedule
- Finish 5mins earlier than schedule
- **10mins** break after 50 mins

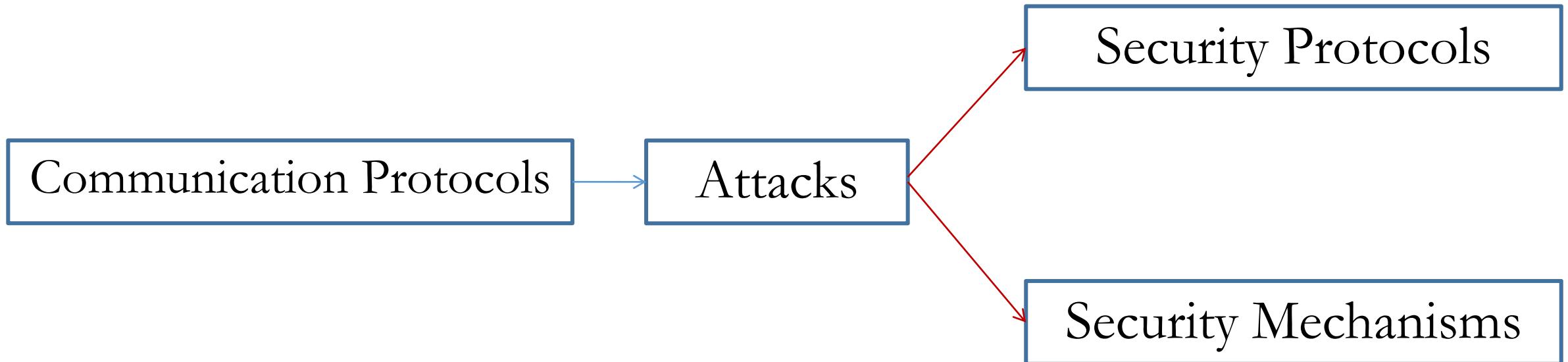
$$5 + 50 + 10 + 50 + 10 + 50 + 5 = 3 \text{ Hours}$$

# 1.1 Subject Introduction: Topics (1/3)

- Lecture 1: Subject Introduction and Basics of Network Security
- Lecture 2: Basics of Cryptography
- Lecture 3: Application Layer Security
- Lecture 4: Transport Layer Security
- Lecture 5: Internet Layer Security
- Lecture 6: Link Layer Security
- Lecture 7: Security Mechanisms and Firewalls



# 1.1 Subject Introduction: Topics (2/3)



Fine if you cannot distinguish them.

# 1.1 Subject Introduction: Topics (3/3)



## Labs

1. Setup
2. ARP Attacks
3. Sniffing\_Spoofing
4. ICMP Attacks
5. TCP Attacks
6. Firewalls
7. PKI
8. TLS

# 1.1 Subject Introduction: Aims of This Subject

- Understand network **vulnerabilities** and network-based attacks
- Apply a range network security **technologies** for securing networks
- Use appropriate security **standards** and network security tools to enhance security of a distributed system
- **Evaluate**, compare, and recommend network security applications and systems

# 1.1 Subject Introduction: Required Knowledge

- Basic cryptography (will be introduced briefly)
- Basic computer network knowledge (will be introduced )
- Programming: C, or Java or python (**not needed for AorE**)
- Mathematical calculations (**Very Basic and Not this in E**)

# 1.1 Subject Introduction: References (Just)

- William Stallings, **Cryptography and Network Security**, 7<sup>th</sup> edition, Pearson, 2016 ([more related to security protocols from cryptography](#))
- C. Kaufman, R. Perlman, and M. Speciner, **Network Security: PRIVATE communication in a PUBLIC world**, 2nd edition, Prentice Hall, 2002.
- William Stallings, **Network Security Essentials**, 6<sup>th</sup> edition, Pearson, 2016
- Colin Boyd, Anish Mathuria, Douglas Stebila, **Protocols for Authentication and Key Establishment**, 2nd Edition, Springer, 2020 ([very theoritical](#))
- Fuchun: [ChatGPT](#)
- Fuchun: [The Cyber Security Body of Knowledge](#) (<https://www.cybok.org/>)

# 1.1 Subject Introduction: Philosophy (1/2)

- Eventually, you will forget what you have learned, but this is fine and normal.
- The most important is: You have seen how the human tried to solve problems related to network and it security.
- The **methodology** will become your muscle memory in the future and benefit you.

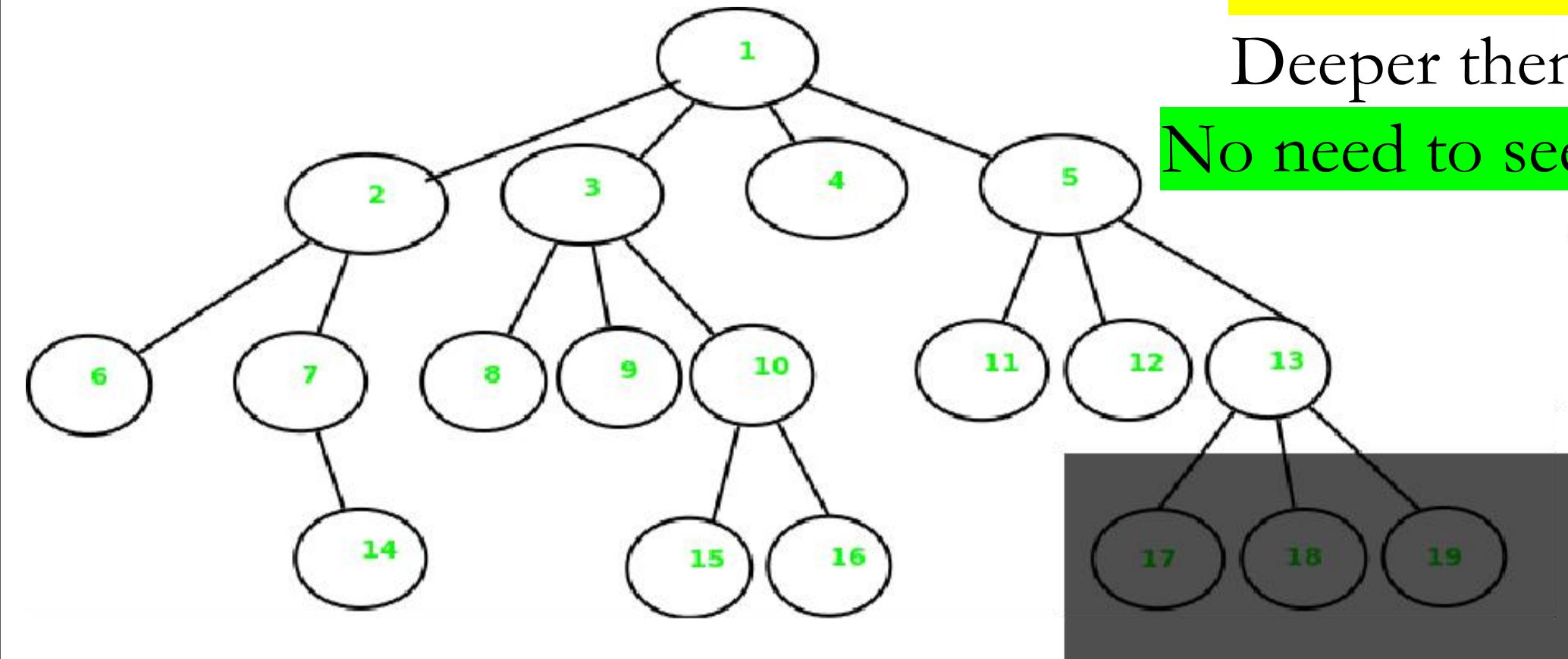
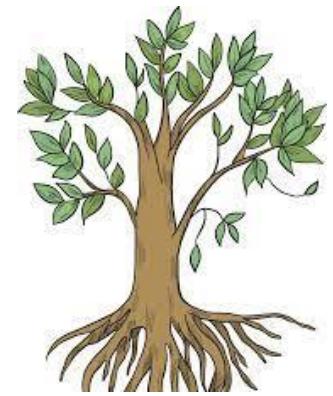


# 1.1 Subject Introduction: Philosophy (2/2)

Tree of Knowledge

Deeper then harder

No need to see all leaves



For example: At node 13, we will briefly introduce all its child nodes and stop.

# 1.1 Subject Introduction: Assessment (Individual 1/2)

Assignment 1, protocol design & analysis	15%	Due: <b>See Moodle Site</b>
Assignment 2, protocol design & analysis	15%	Due: <b>See Moodle Site</b>
Final Exam Multiple-choice questions protocol analysis	70%	Exam Period

At least 40% (28/70) in the final exam, otherwise **TF will** be given.

# 1.1 Subject Introduction: Assessment (Individual 2/2)

- Assignments must be submitted via Moodle. It is the student's responsibility to check the assignments they have submitted.
- Penalties apply to all late work, except if student **academic consideration (AC)** has been granted. Pls submit AC via system and also email me.
- Late submissions will attract a penalty of 5% of the assessment mark per day including weekends. No more late submissions after 4 days.
- Plagiarism --> Zero mark. (**can use GPT but need to meet criterias**)

## 1.2 Concept Clarification

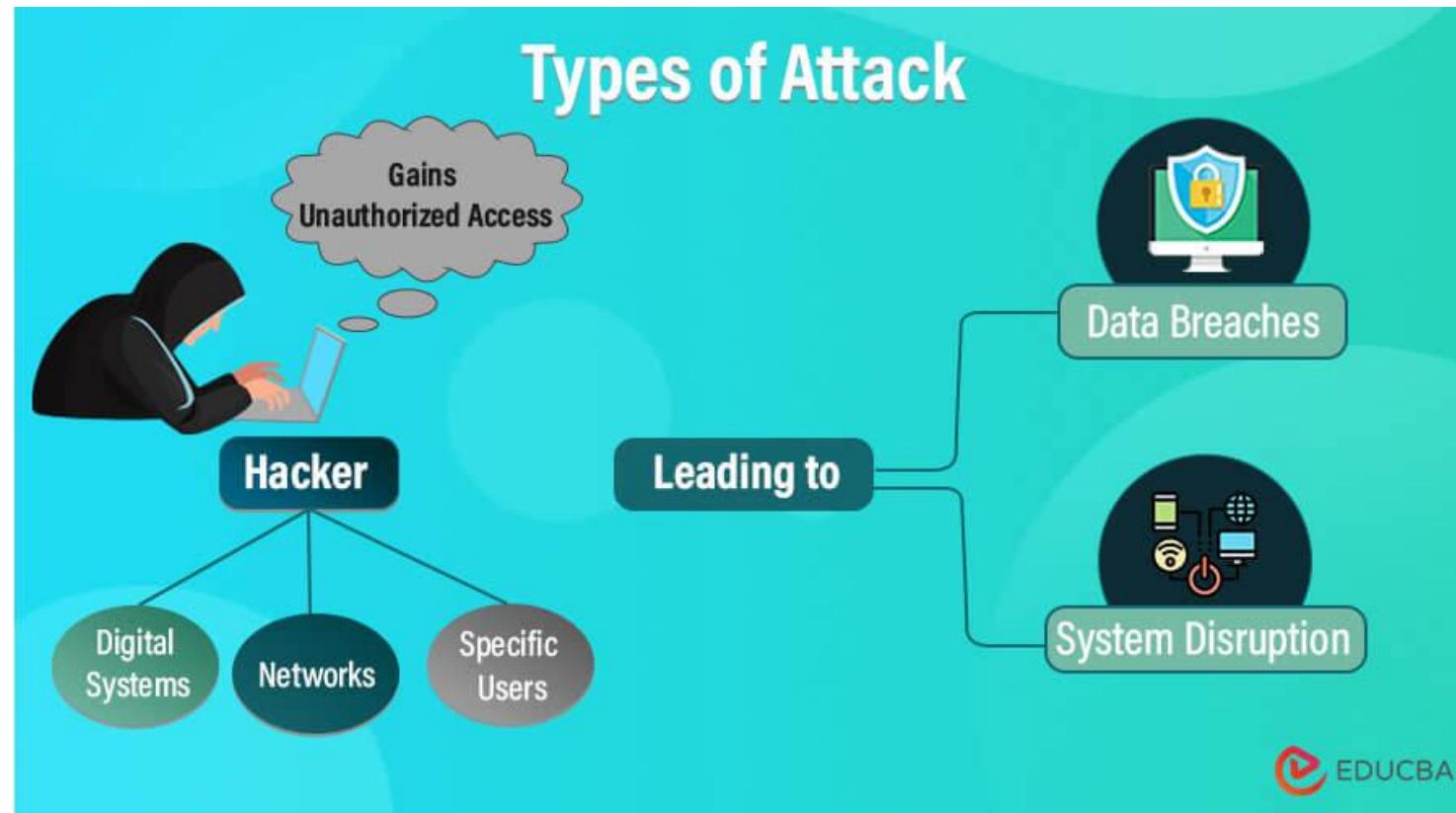
## 1.2 Concept Clarification: Role of Network and Protocol

The primary role of a network is to facilitate communication between devices. Networks enable the transfer of data, resources, and services across multiple devices, allowing them to share information and communicate efficiently using protocols.

- **Data Communication:** Networks allow devices, such as computers, servers, and mobile devices, to exchange data through protocols like TCP/IP.
- **Resource Sharing:** Networks facilitate sharing of resources (e.g., printers, storage, and applications) across multiple users and devices.
- **Collaboration and Services:** They enable collaborative tasks, remote access, and services like email, video conferencing, and file sharing.
- **Support for Applications:** Networks provide the infrastructure for various applications such as web services, databases, and cloud computing.

## 1.2 Concept Clarification: Network Attacks

Network attacks exploit **vulnerabilities in protocols**, allowing attackers to abuse them and compromise the security of communications.



## 1.2 Concept Clarification: Three Key Concepts and Two Modes

- **Protocols:** protocols for communications. Sending messages from one device to another device.
- **Security Protocol:** protocols for communications and security is a fundamental requirement when the protocol was designed.
- **Security Mechanism:** Methods using policies, rules, access control, and monitoring for securing network communication protocols.

Two modes:

- Protocol + security protocol/security mechanism
- Security protocol (replacing protocol)

## 1.2 Concept Clarification: Cyber Security VS Network Security

- **Cyber security** is a subset of information security which refers to a set of techniques and methodologies used to **protect integrity of networks, devices, programs, and data** from damage, attack, or unauthorized access. In simple terms, cyber security is the practice of protecting internet-connected systems and networks from digital attacks.
- **Network security**, on the other hand, is the act of protecting files and directories in a network of computers **against misuse, hacking, and unauthorized access to the system**. Network security is a **subset of cyber security** which protects the integrity of your network and network-accessible resources from unauthorized access.

## 1.3 OSI and TCP/IP

Communication is complicated. We split the communications into several layers (phases, or steps)

# 1.3 OSI and TCP/IP: What are they (1/3)

## OSI Reference Model

7 Application

6 Presentation

5 Session

4 Transport

3 Network

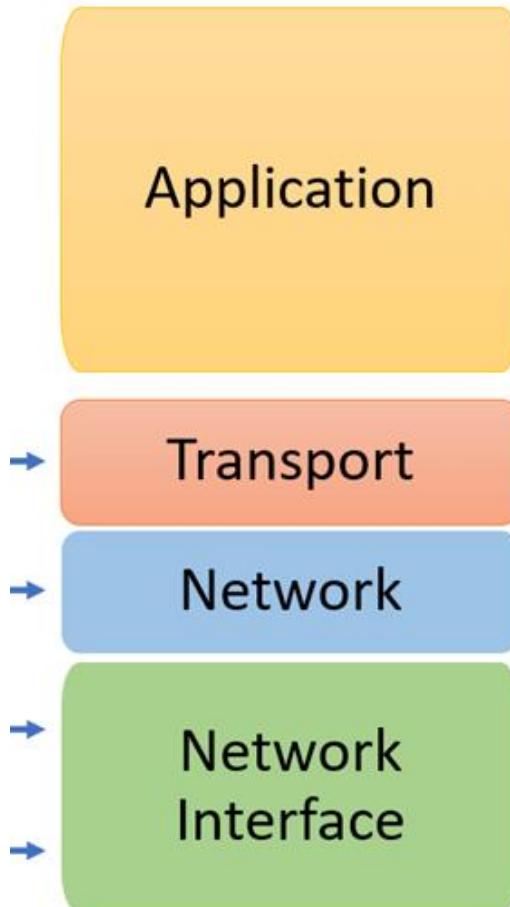
2 Data Link

1 Physical

- Released in 1983 by ISO.
- The **OSI** is an abbreviation of Open Systems Interconnection.
- It is a standard framework for designing various protocols used in computer networks
- The OSI model divides tasks into layers, allowing each layer to be addressed independently, or nearly so, to simplify problem-solving and improve network design

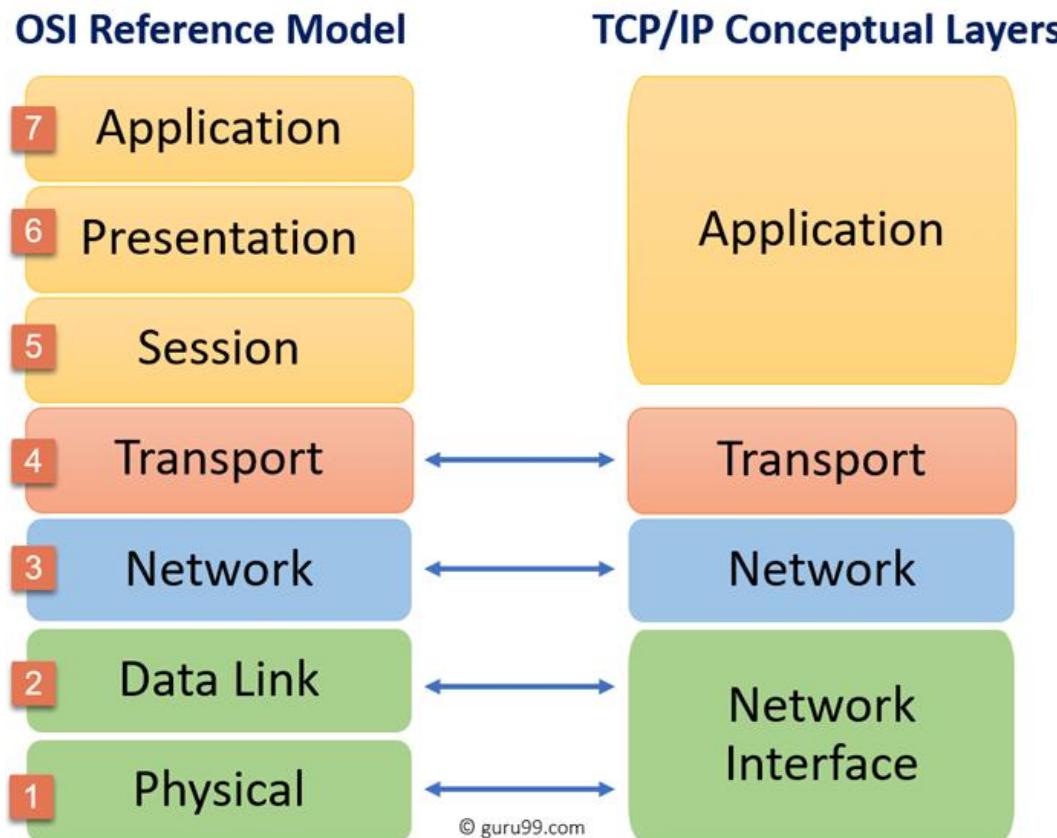
# 1.3 OSI and TCP/IP: What are they (2/3)

## TCP/IP Conceptual Layers



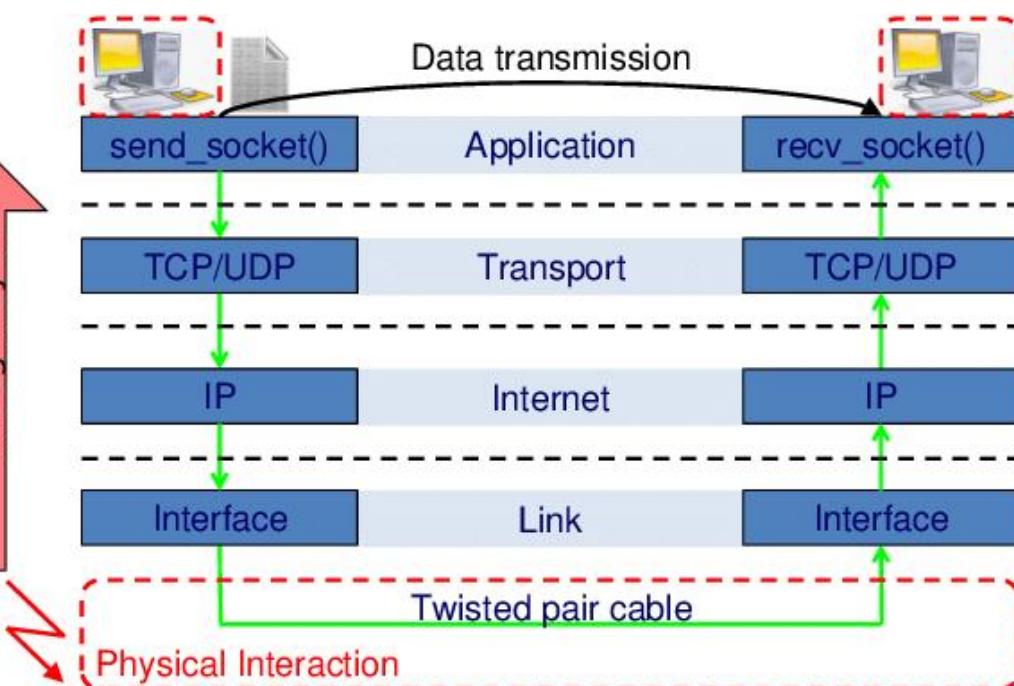
- **TCP/IP**, or Transmission Control Protocol Internet Protocol, is the foundational suite of communication protocols used for the internet and modern networks.
- Developed in the 1970s and formalized in the 1980s.
- TCP/IP standardizes how data is transmitted across networks by splitting communication tasks into layers, allowing for reliable, efficient, and scalable data exchange between systems.

# 1.3 OSI and TCP/IP: What are they (3/3)



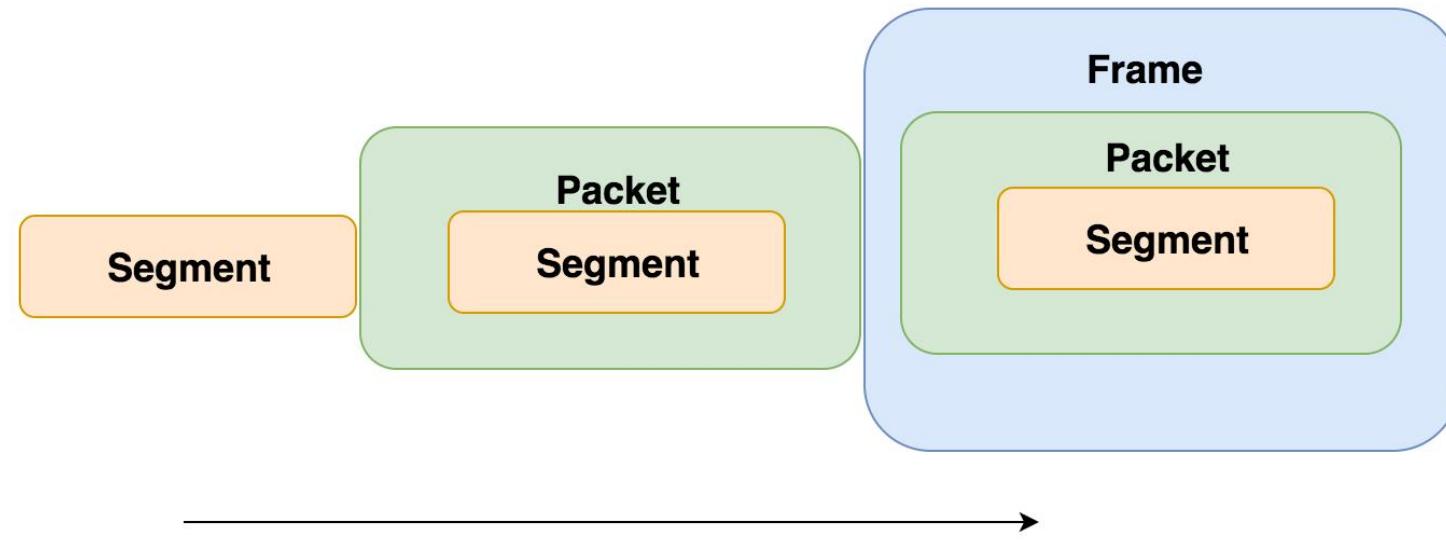
- The OSI model has seven layers providing a theoretical and more detailed approach to understanding how different protocols interact. It focuses on separating network functions into distinct layers, each with a specific role.
- The TCP/IP model is designed specifically for real-world internet communications. It is more practical, with protocols like TCP and IP already in use when the model was developed. The TCP/IP model combines some layers (like OSI's Presentation and Session into its Application layer), making it more straightforward but less descriptive compared to OSI.

# 1.3 OSI and TCP/IP: Communications and Protocols (1/3)



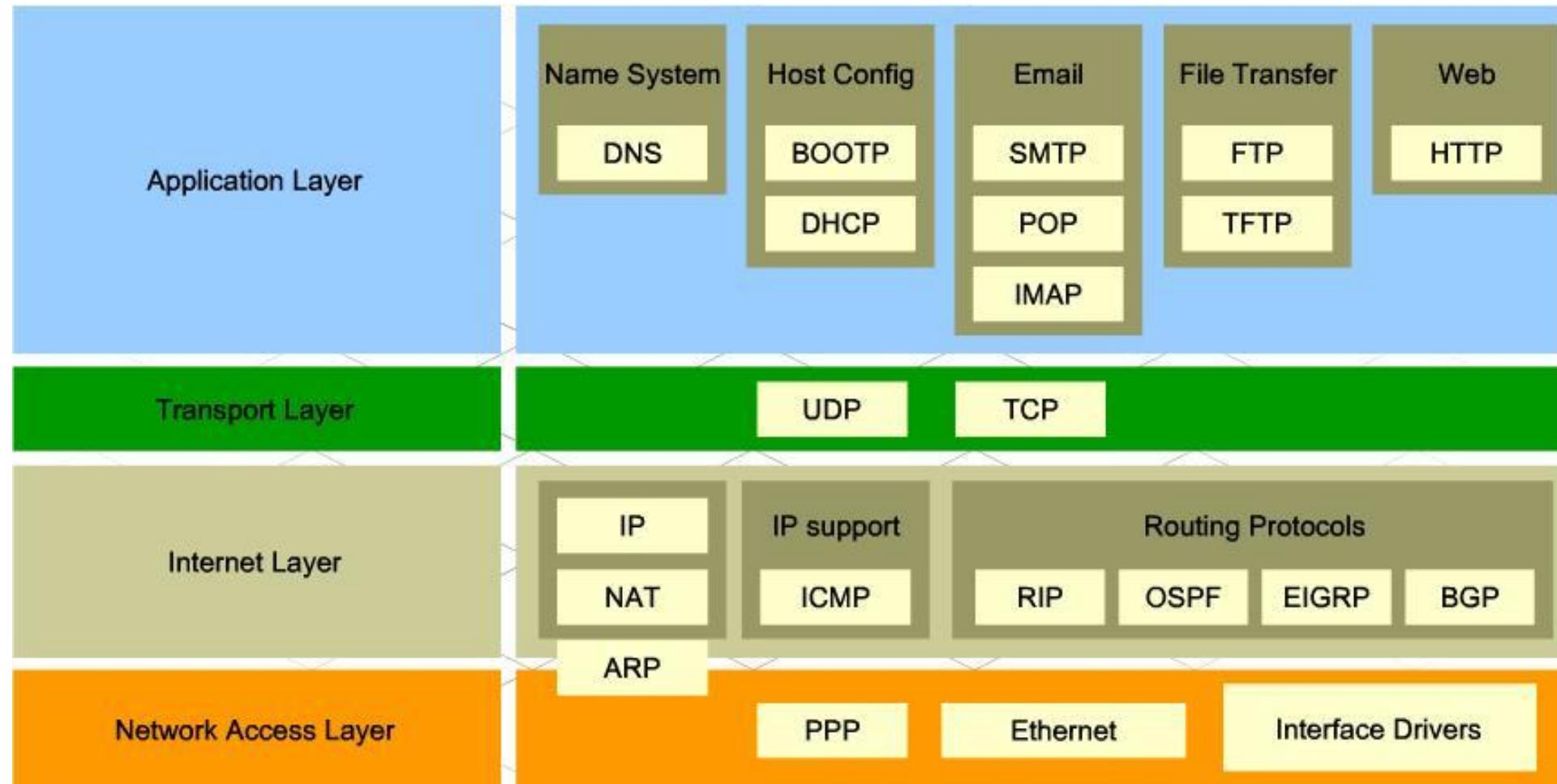
- **Application Layer Protocols:** Protocols like HTTP, FTP, or SMTP enable applications on A and B to communicate, specifying how the data should be formatted and handled at the user level.
- **Transport Layer Protocols:** TCP or UDP manage the reliability of the data transfer, ensuring data is correctly sent from A to B, handling error detection, retransmission, and flow control.
- **Internet Layer Protocols:** IP assigns addresses and routes the data packets between devices A and B across different networks.
- **Link Layer Protocols:** Protocols like Ethernet or Wi-Fi define how data is physically transmitted over the network medium between the devices.

# 1.3 OSI and TCP/IP: Communications and Protocols (2/3)

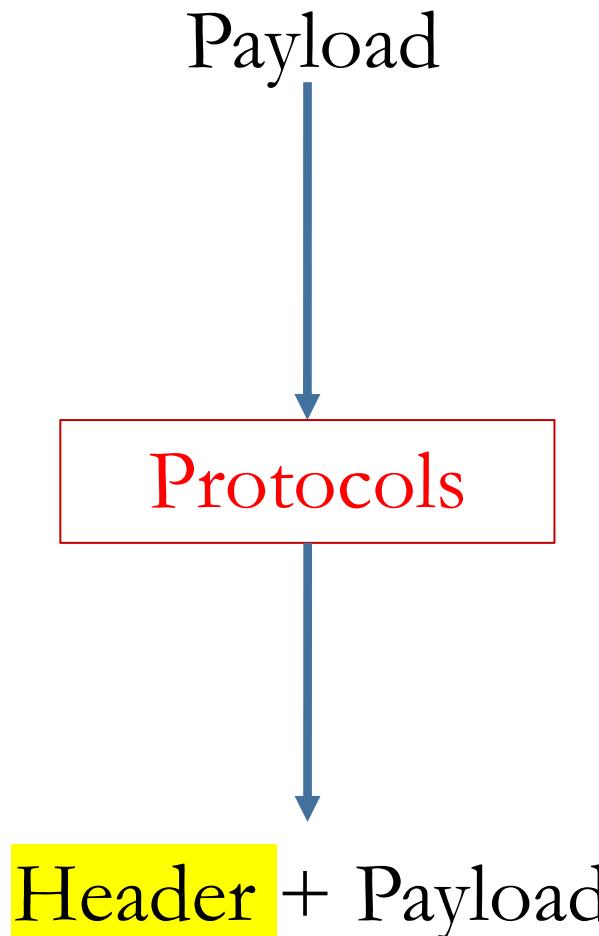


- **Segment (Transport Layer)**: If the transport protocol is TCP, the unit of data sent from TCP to network layer is called **Segment**.
- **Datagram (Internet/Network Layer)**: If the network protocol is IP, the unit of data is called **Datagram** (packet). At transport layer, if protocol is UDP, we use datagram there as well. Hence, we differentiate them as UDP Datagram, IP Datagram.
- **Frame (Data Link Layer)**: the protocol data unit at the data link layer

# 1.3 OSI and TCP/IP: Communications and Protocols (3/3)

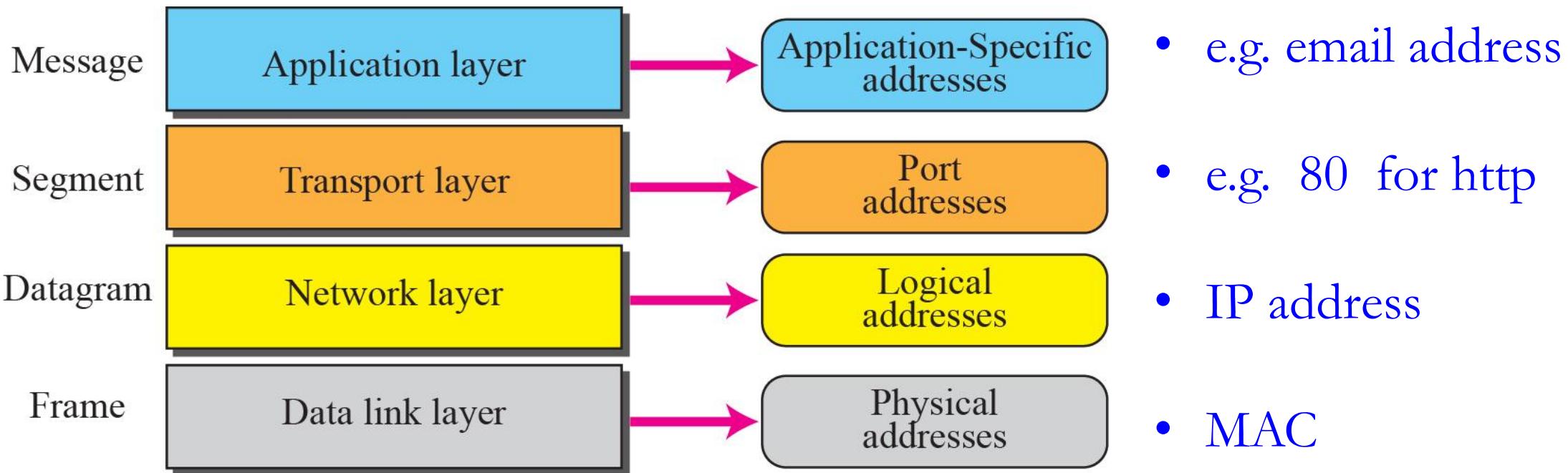


# 1.3 OSI and TCP/IP: How Protocols Work (1/2)



- The payload is the actual data being transmitted from top layer to bottom layer.
- The header is a structured section at the beginning of a packet that contains metadata about the packet (**generated by protocols**). This metadata provides critical information for routing, delivering, and processing the data.
- Some protocol could generate footer.

# 1.3 OSI and TCP/IP: How Protocols Work (2/2)



- An application address, such as an email address, identifies a specific user within an application
- A port number specifies a specific process on a device to facilitate communication between applications.
- An IP address identifies a device on a different (non-local) network at the Internet layer
- A MAC address uniquely identifies a network interface on the local network at the link layer

# 1.3 OSI and TCP/IP: HTTP Header

HTTP Protocol: POST request

POST /login HTTP/1.1

Host: www.example.com

Content-Type: application/x-www-form-urlencoded

Content-Length: 29

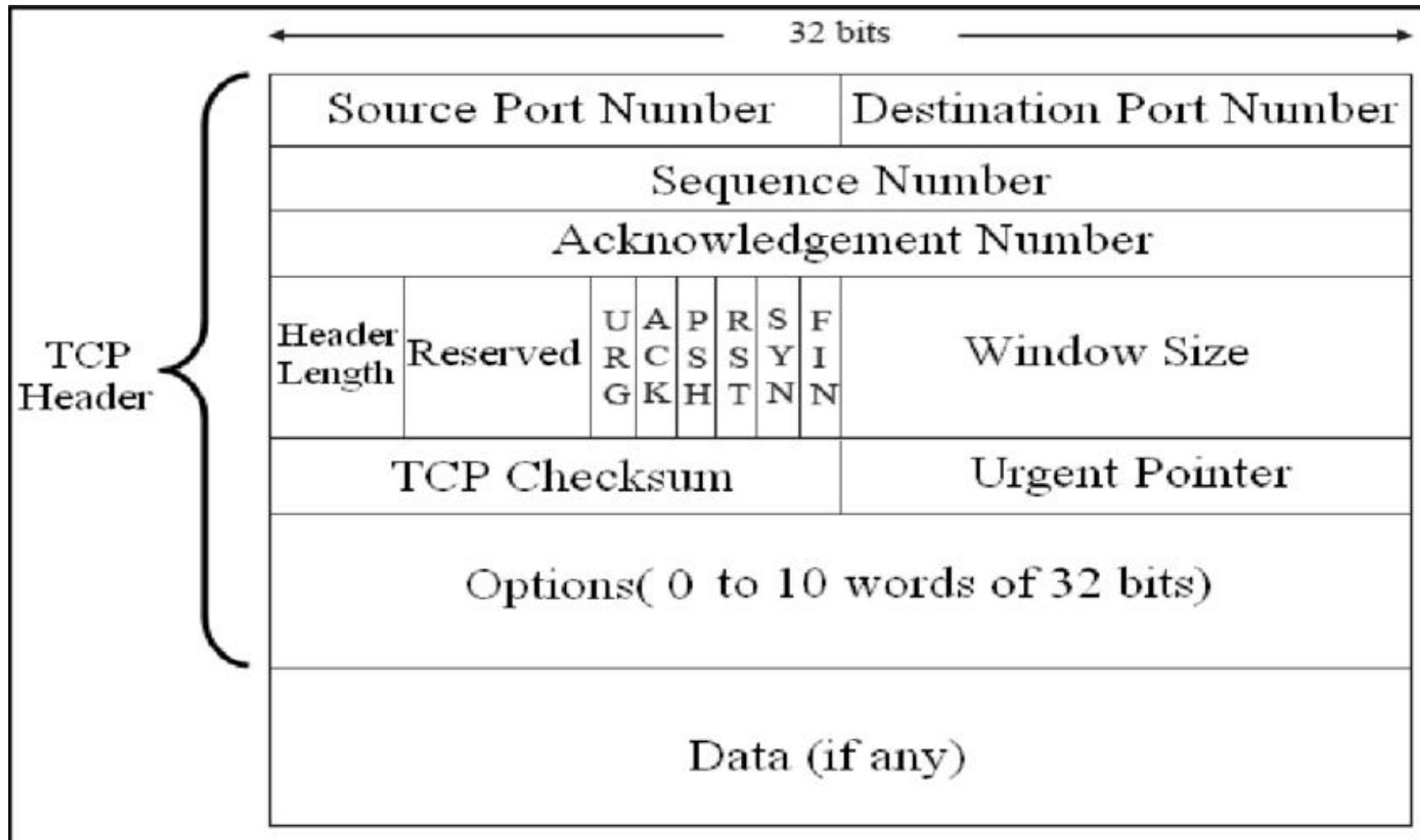
username=johnguo&password=123456

# 1.3 OSI and TCP/IP: TCP Header (1/2)

The **TCP header** has a variable size, but the minimum header size is 20 bytes (160 bits). It can be larger if optional fields are used. Fields:

- Source Port: 16 bits.
- Destination Port: 16 bits.
- Sequence Number: 32 bits.
- Acknowledgment Number: 32 bits.
- Data Offset (Header Length): 4 bits (specifies the size of the TCP header).
- Flags: 9 bits (URG, ACK, PSH, RST, SYN, FIN, etc.).
- Window Size: 16 bits.
- Checksum: 16 bits.
- Urgent Pointer: 16 bits.

# 1.3 OSI and TCP/IP: TCP Header (2/2)

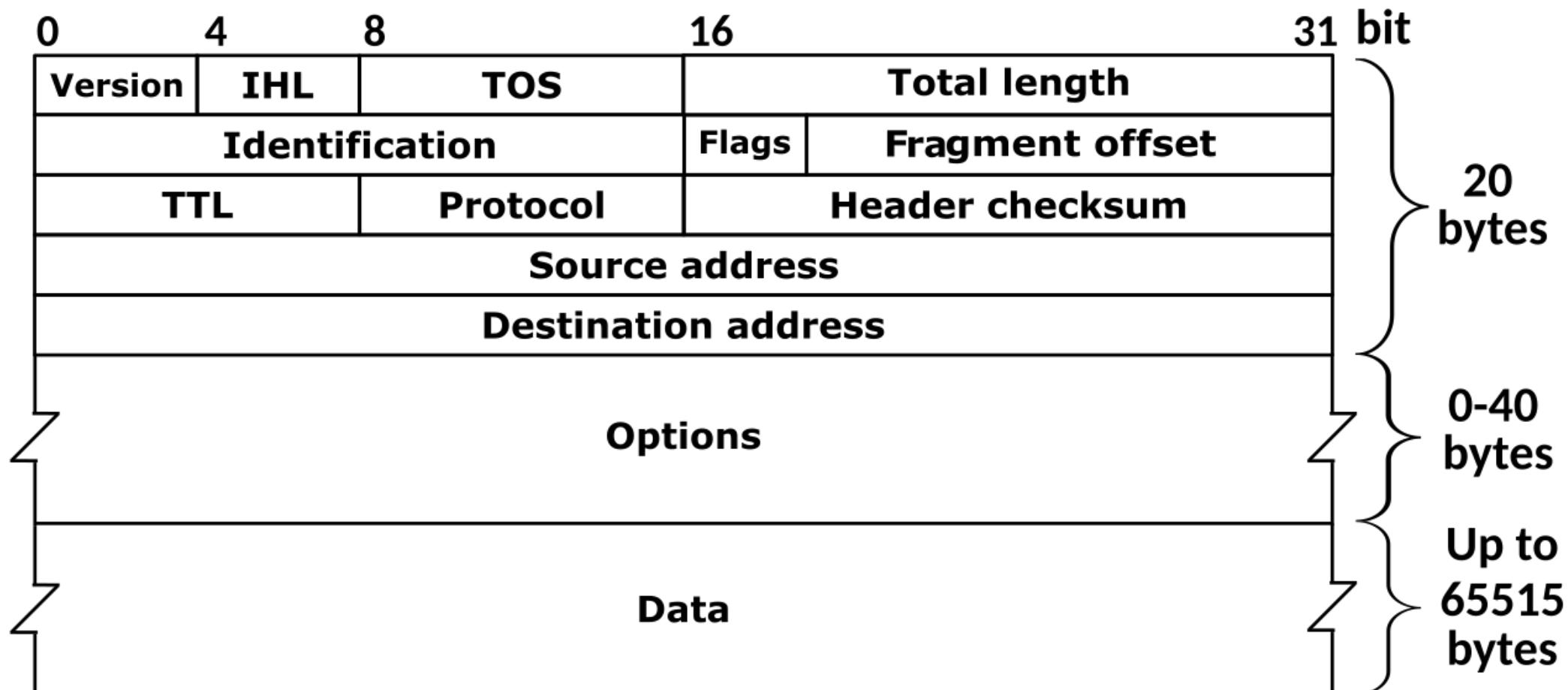


# 1.3 OSI and TCP/IP: IPv4 Header (1/2)

The **IPv4 header** has a fixed minimum size but can vary if options are used. The minimum IPv4 header size is 20 bytes. Size: 20 bytes = 160 bits (without options).

- Version: 4 bits.
- Header Length: 4 bits.
- Type of Service (ToS): 8 bits.
- Total Length: 16 bits.
- Identification: 16 bits.
- Flags: 3 bits.
- Fragment Offset: 13 bits.
- Time to Live (TTL): 8 bits.
- Protocol: 8 bits.
- Header Checksum: 16 bits.
- Source IP Address: 32 bits.
- Destination IP Address: 32 bits.

# 1.3 OSI and TCP/IP: IPv4 Header (2/2)



# 1.3 OSI and TCP/IP: Ethernet Header (1/2)

The **Ethernet header** is a fixed size.

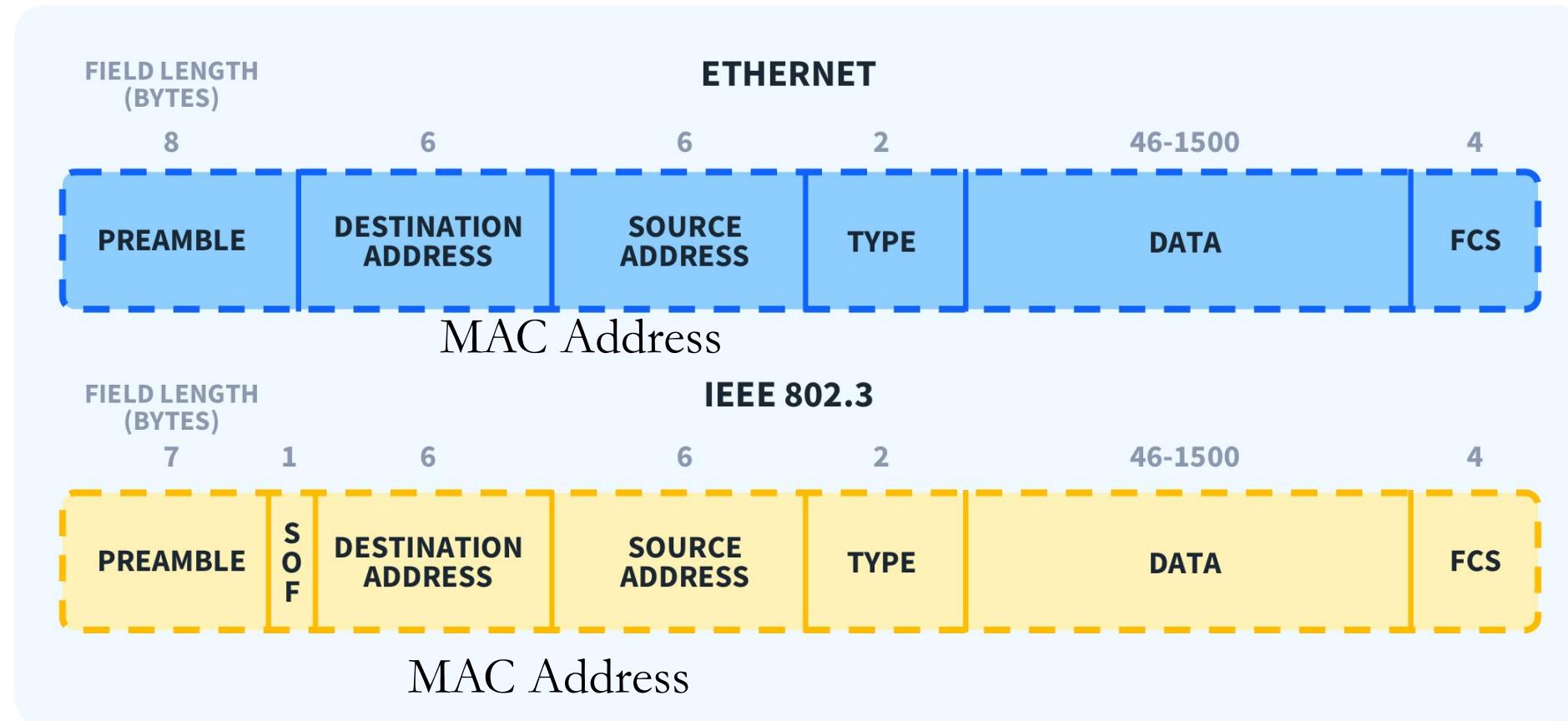
Size: 14 bytes = 112 bits.

Fields:

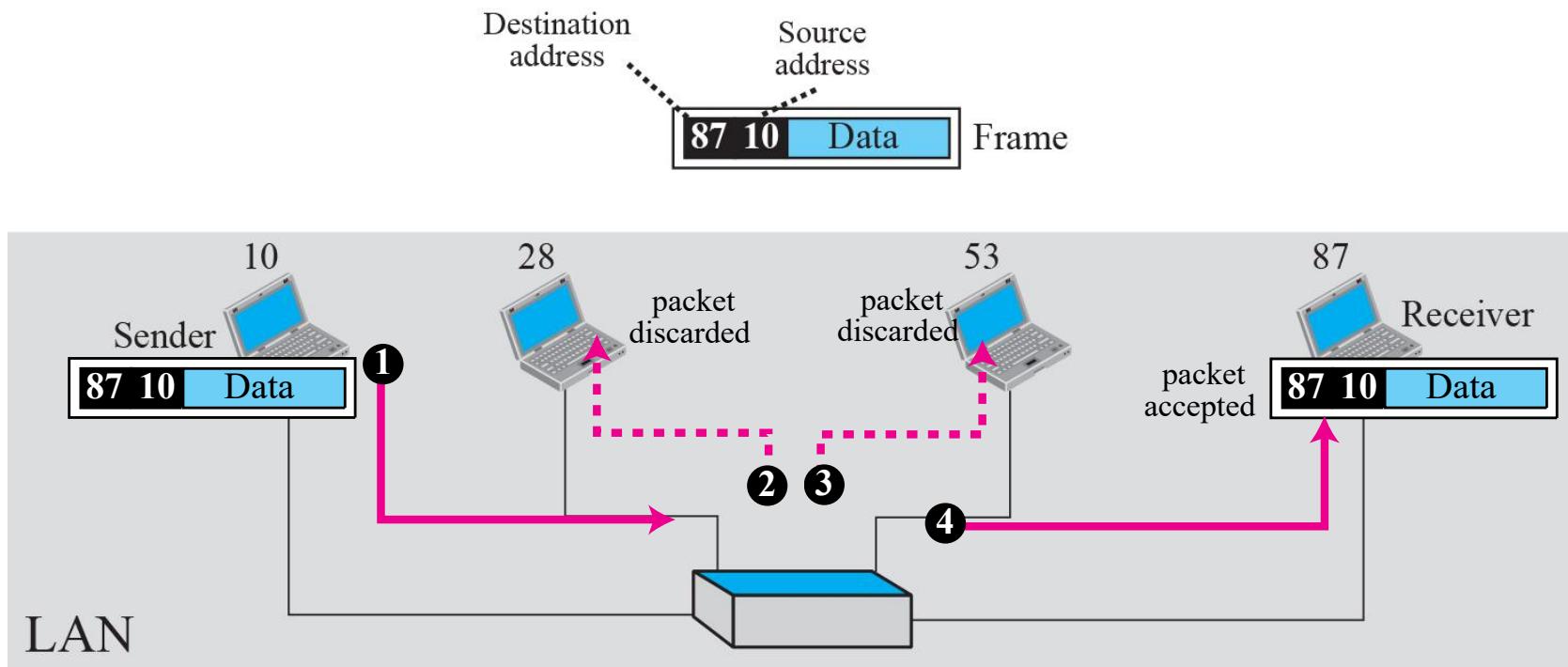
- Preamble (7 bytes, 56 bits)
- Start Frame Delimiter (SFD) (1 byte, 8 bits)
- Destination MAC Address: 6 bytes = 48 bits.
- Source MAC Address: 6 bytes = 48 bits.
- EtherType: 2 bytes = 16 bits.

Thus, the Ethernet header (excluding preamble and SFD) is 14 bytes (112 bits).

# 1.3 OSI and TCP/IP: Ethernet Header (2/2)

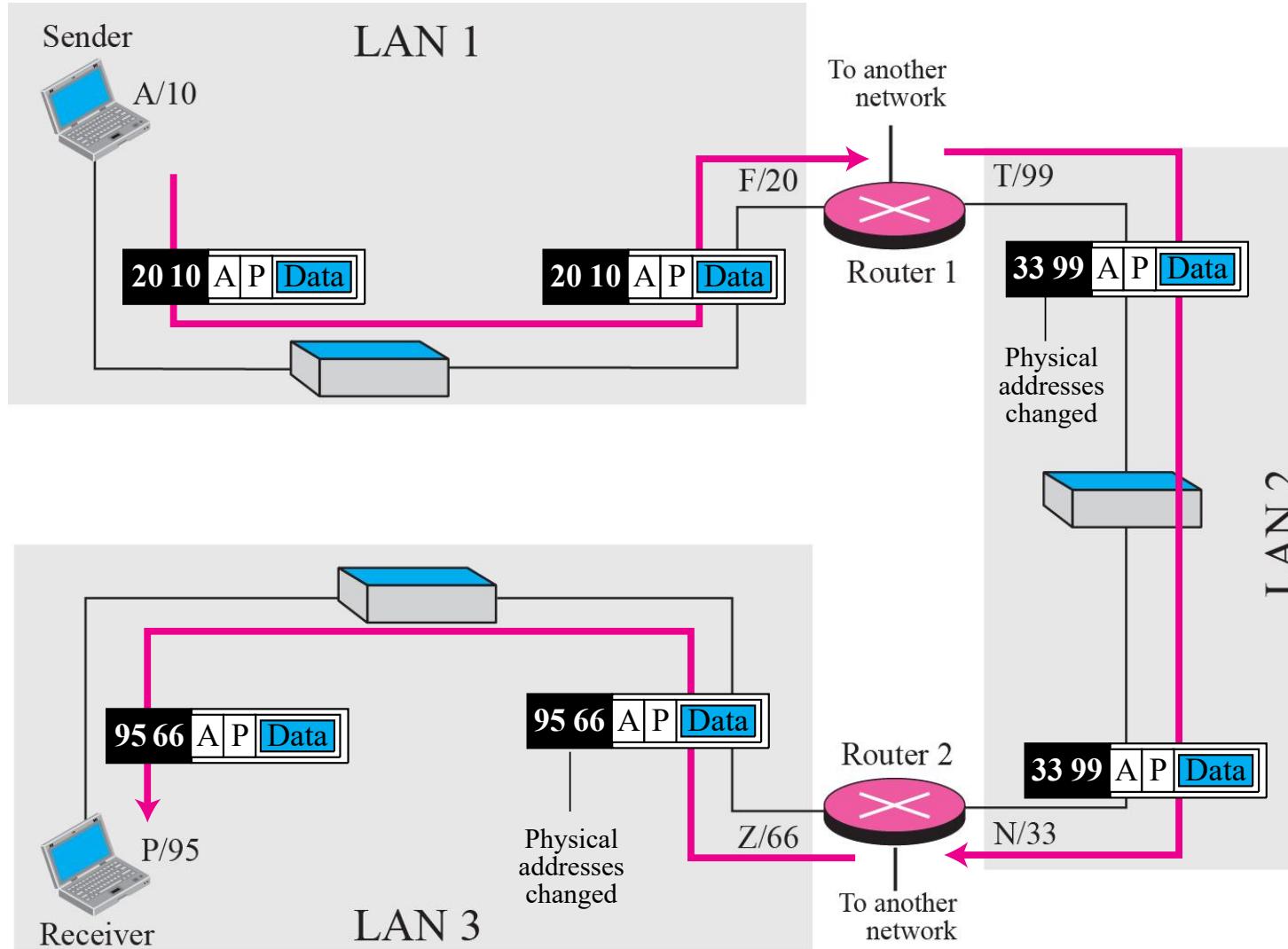


# 1.3 OSI and TCP/IP: Communication Example (1/3)



A node with physical (MAC) address 10 sends a **frame** to a node with physical address 87.

# 1.3 OSI and TCP/IP: Communication Example (2/3)

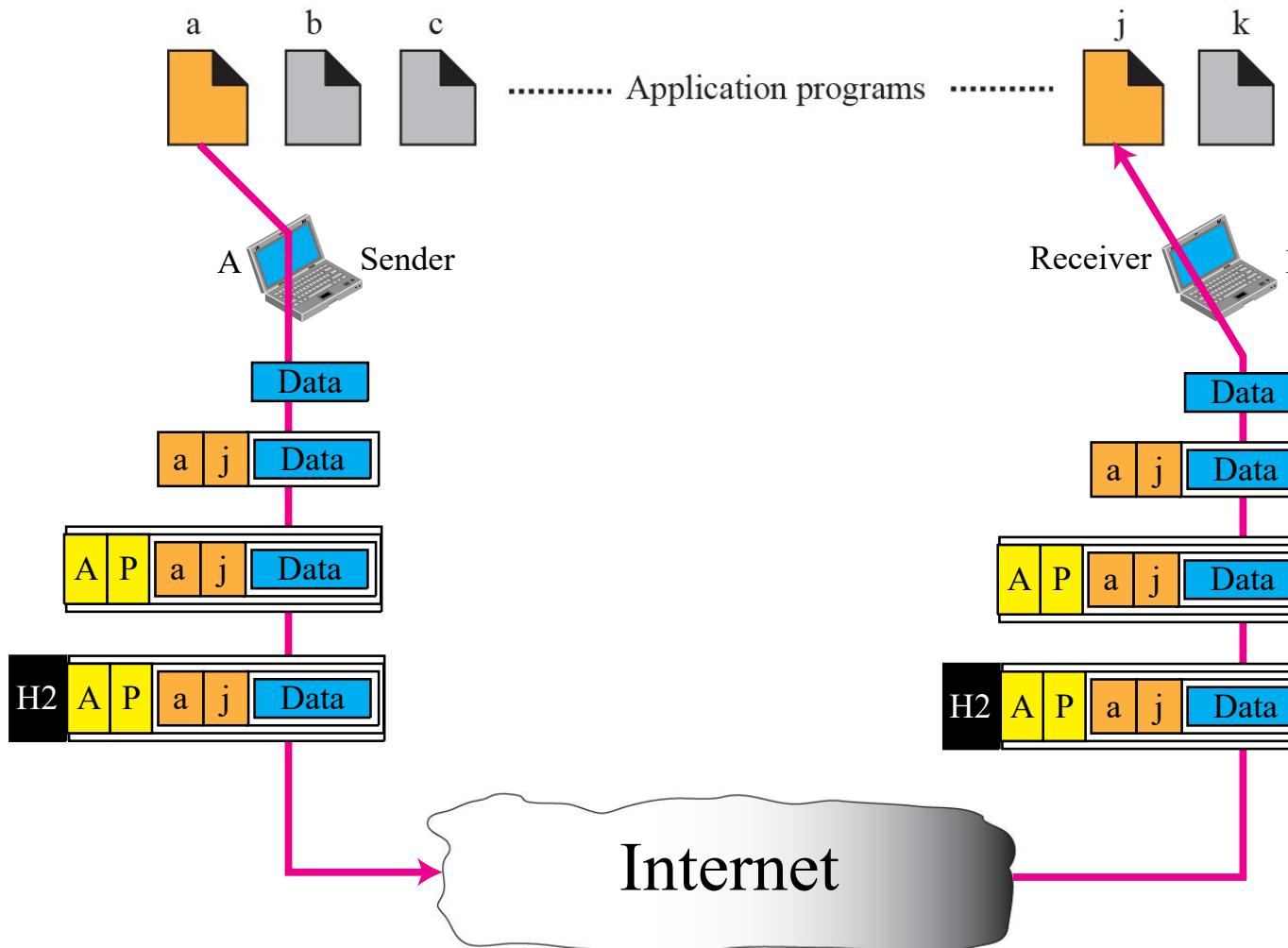


WAN: Wide-Area Network

The computer with logical address (IP) **A** and physical address 10 needs to send a packet to the computer with logical address **P** and physical address 95.

Note: There should have other devices whose MAC are 20, 99, 33, 66. But they are not shown here.

# 1.3 OSI and TCP/IP: Communication Example (3/3)



- The sending computer is running three processes at this time with port addresses **a, b, and c**.
- The receiving computer is running two processes at this time with port addresses **j and k**.

## 1.4 Use and Abuse

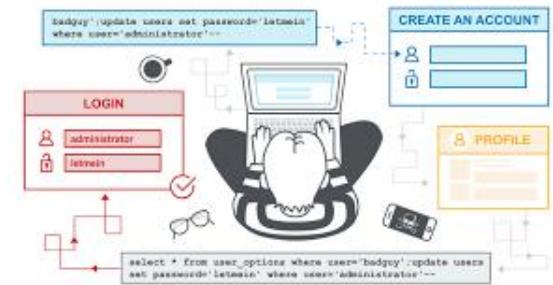
Protocols are designed for being used but could be abused

# 1.4 Use and Abuse: Case Studies (1/5)

## 1. Input Forms & SQL Injection

Use: Web applications provide input fields (such as login forms, search bars, or comment sections) where users can easily input data.

Abuse: Attackers exploit this feature by injecting malicious SQL queries (SQL injection) into these forms to gain unauthorized access to databases, retrieve sensitive information, or manipulate data.



# 1.4 Use and Abuse: Case Studies (2/5)

## 2. SMTP

Use: SMTP (Simple Mail Transfer Protocol) is used to send emails across the internet, allowing users to communicate effortlessly through email services. It enables the transfer of **ANY** emails between servers and ensures that messages reach their intended recipients.

Abuse: Attackers exploit the protocol's lack of built-in authentication by sending emails with forged sender addresses. This enables attackers to impersonate trusted sources, leading to security threats like phishing attacks

# 1.4 Use and Abuse: Case Studies (3/5)

## 3. TCP

Use: TCP (Transmission Control Protocol) ensures reliable and ordered data transmission between devices through a **three-way handshake**, using the SYN flag to establish a connection. This process makes TCP ideal for applications like web browsing, email, and file transfers, as it guarantees data integrity and proper communication flow.

Abuse: TCP can be abused through SYN flooding attacks. In this attack, adversaries exploit the initial connection process by sending a large number of SYN (synchronize) requests to a server **but never completing the three-way handshake**. This leaves the server with many half-open connections, consuming its resources and potentially causing a denial of service (DoS), preventing legitimate users from accessing the server or services.

# 1.4 Use and Abuse: Case Studies (4/5)

## 4. ICMP (Internet Control Message Protocol)

Use: ICMP (Internet Control Message Protocol) is a network layer protocol used primarily for diagnostic purposes, such as checking connectivity between devices using tools like **ping** or traceroute. It helps users detect network issues by sending echo requests and receiving echo replies, ensuring smooth communication between devices.

Abuse: ICMP can be abused through attacks like ICMP flooding or Smurf attacks. The attacker sends a large number of ICMP echo requests (pings) to a target, overwhelming it and causing a denial of service (DoS). In a Smurf attack, the attacker sends ICMP requests to a network's **broadcast address**, spoofing the victim's IP address. The network's devices then send echo replies to the victim, overwhelming it with responses and disrupting its normal operations.

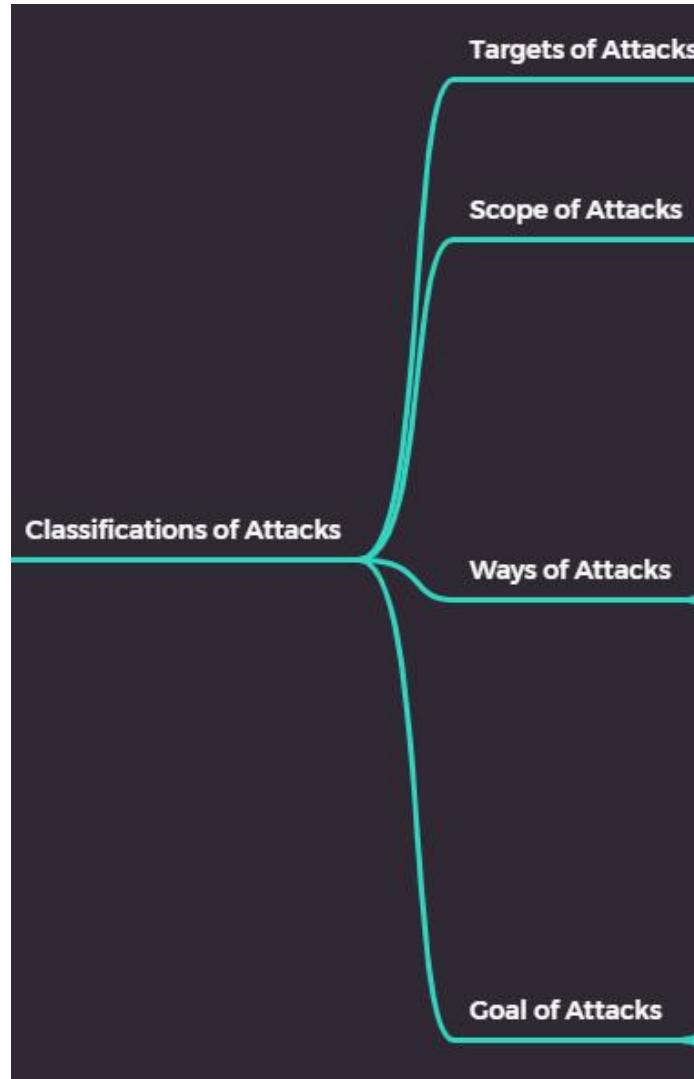
# 1.4 Use and Abuse: Case Studies (5/5)

## 5. ARP Spoofing

Use: ARP (Address Resolution Protocol) is a fundamental protocol in local networks that allows devices to automatically map IP addresses to MAC addresses. This seamless address resolution enables devices to communicate efficiently within the local network **without requiring manual configuration.**

Abuse: ARP can be abused through ARP spoofing. Attackers exploit the protocol's lack of authentication by sending false ARP messages, associating their MAC address with another device's IP address. This enables attackers to intercept, modify, or block traffic, leading to security threats like man-in-the-middle attacks and denial of service (DoS).

# 1.4 Use and Abuse: Classification of Attacks (1/5)



Which entity to be attacked

How the attacker interacts with the communication

What techniques or methods that adversaries use to exploit vulnerabilities

What specific objectives that attackers aim to achieve

## 1.4 Use and Abuse: Classification of Attacks (2/5)

The "target of attacks" refers to the specific entities that attackers aim to compromise, exploit, or disrupt in their malicious activities. In the context of host and user, here's how they can be defined:

- **Host Targets:** Hosts refer to devices such as servers, computers, or network equipment that provide resources or services within a network.
- **User Targets:** Users refer to individuals who interact with systems and services, typically involving human elements that can be manipulated or deceived.

## 1.4 Use and Abuse: Classification of Attacks (2/5)

The “scope of attacks” in network security generally falls into two broad categories: passive attacks and active attacks. These two types differ in how the attacker interacts with the communication process.

1. **Passive Attacks:** Passive attacks involve monitoring or eavesdropping on communications without altering them. The goal is usually to gather information without being detected.
2. **Active Attacks:** Active attacks involve direct interaction with communications, often aiming to alter, disrupt, or manipulate data. These attacks can damage the integrity, availability, or authenticity of the network.

# 1.4 Use and Abuse: Classification of Attacks (4/5)

The "way of attacks" refers to the various techniques or methods that adversaries use to exploit vulnerabilities and compromise security.

- Flooding Attacks: Overwhelm a network or server with excessive traffic, causing a Denial of Service (DoS) or Distributed Denial of Service (DDoS).
- Spoofing Attacks: Impersonate a legitimate entity by forging data, such as IP addresses, MAC addresses, or email headers, to gain unauthorized access or redirect communication.
- Sniffing Attacks: Intercept network traffic to capture sensitive information such as passwords, credit card numbers, or unencrypted data. Eavesdropping attack?
- Injection Attacks: Inject malicious data or commands into a vulnerable system, often through user input fields, to manipulate the system or database.

## 1.4 Use and Abuse: Classification of Attacks (4'/5)

- Man-in-the-Middle (MitM) Attacks: Intercept and alter communications between two parties without their knowledge, potentially stealing or modifying data.
- Replay Attacks: Capture and retransmit valid data transmissions to trick the system into performing unauthorized actions, such as reusing authentication credentials.
- Social Engineering Attacks: Exploit human psychology to trick users into divulging sensitive information or performing unsafe actions.
- Phishing Attacks: A common form of social engineering, where attackers masquerade as legitimate entities (e.g., emails from banks) to steal personal data like usernames and passwords.

# 1.4 Use and Abuse: Classification of Attacks (5/5)

The “goal of attacks” refers to the specific objectives that attackers aim to achieve when exploiting vulnerabilities. These goals align with the core principles of information security, often targeting one or more of the following areas:

- Confidentiality Attacks: To access or steal sensitive information without authorization. The focus is on compromising the privacy of data.
- Integrity Attacks: To alter, corrupt, or destroy data, making it inaccurate or misleading. These attacks aim to compromise the trustworthiness of information.
- Availability Attacks: To disrupt the normal functioning of a system, making services, data, or resources unavailable to legitimate users.
- Authentication Attacks: To bypass or manipulate the authentication process, gaining unauthorized access to a system or user account.

## 1.5 How to Secure Network Protocols

Protocols are designed for being used but could be abused

# 1.5 How to Secure Network Protocols: 1/5

Question: How to Secure Network Protocol A?

1. Design a security protocol  $A^*$  to replace A. (run  $A^*$ )
2. Design a security protocol B to protect A. (run  $A + B$ )
3. Design a security mechanism C to protect A. (run A under C)

# 1.5 How to Secure Network Protocols: 2/5



## Security Protocols

- must use **cryptography** for confidentiality and integrity
- could use security mechanisms (when the protocols are complicated)

# 1.5 How to Secure Network Protocols: 3/5



Security Mechanisms: Do **X** if **Y** happens

- Security mechanisms are monitoring something.
- Y is the trigger condition (under monitoring).
- X is the detailed action.

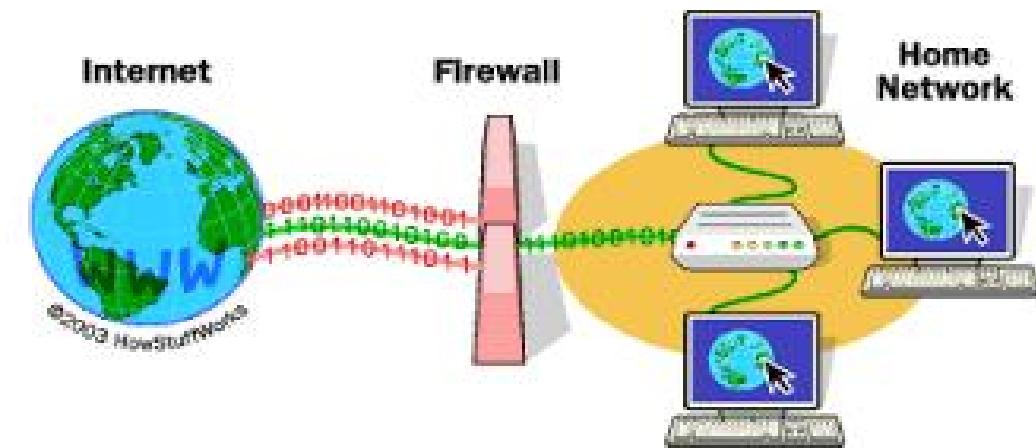
# 1.5 How to Secure Network Protocols: 4/5



Do X if Y happens. Examples

- Delay verification for 1 min if access to an account has failed more than 3 times.
- Drop the ip packet if the source IP address is NOT 225.30.283.34

# 1.5 How to Secure Network Protocols: 5/5



Do X if Y happens. **Firewalls**

A firewall is a security system that monitors and controls incoming and outgoing network traffic based on predefined security rules. It acts as a barrier between a trusted internal network (such as a company's local network) and untrusted external networks (like the internet), filtering data to prevent unauthorized access or responses.

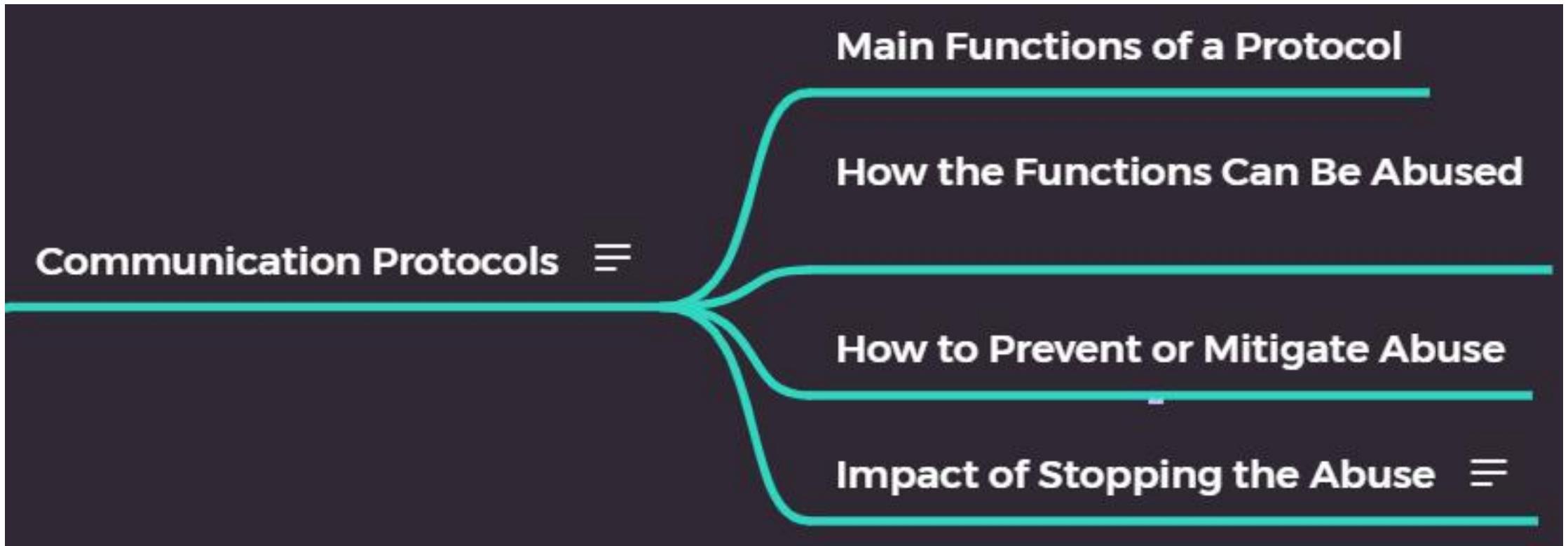
## **1.6 What to be Introduced**

The way of knowledge delivery in this subject

# 1.6 What to be Introduced: Three Categories (1/4)

1. (Communications ) Protocols
2. Security Protocols (from Cryptography)
3. Security Mechanism (via non-cryptography)

# 1.6 What to be Introduced: Three Categories (2/4)



# 1.6 What to be Introduced: Three Categories (3/4)



# 1.6 What to be Introduced: Three Categories (4/4)



# 1.6 What to be Introduced: List of Protocols In this Subject

Layers	Protocols	Security Protocols
Application Layer	HTTP, Email Protocol	S/MIME, SSH
Transport Layer	TCP, UDP	TLS, QUIC
Internet Layer	IPv4, IPv6, ICMP, Routing	IPSec
Link Layer	Ethernet (IEEE 802.3)	Wi-Fi Security Protocol

# **END OF L1**

NETWOCK  
SECURITY

SECURITY

NETWOCK SECURITY



# Network Security

## Lecture 2

A/Prof. Fuchun Guo

# Outline

- Overview
- Confidentiality
- Integrity
- Authenticity
- Key Establishment Protocols
- From Theory to Real World

## 2.1 Overview

# 2.1 Overview: Brief History (1/20)

1976 Modern Cryptography

1960 Computer Networks

1965 Computational Complexity Theory

1949 Shannon's Work

1946 Digital Computers

1883 Cryptography Principle

1936 Turing Machine

Classical Cryptography

# 2.1 Overview: Brief History (2/20)

3rd March

Dear George,

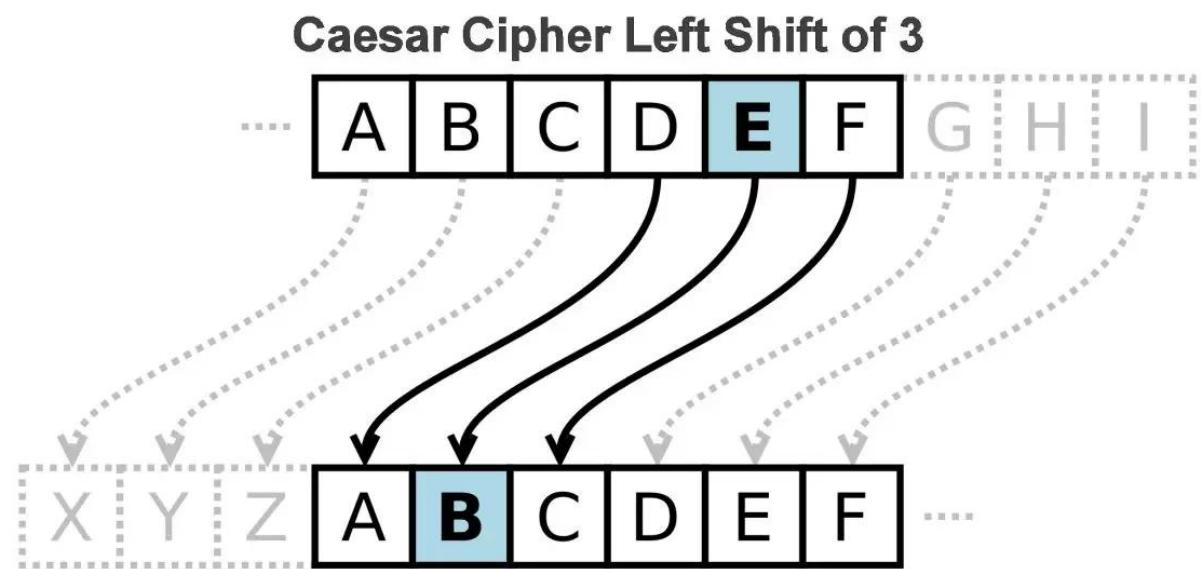
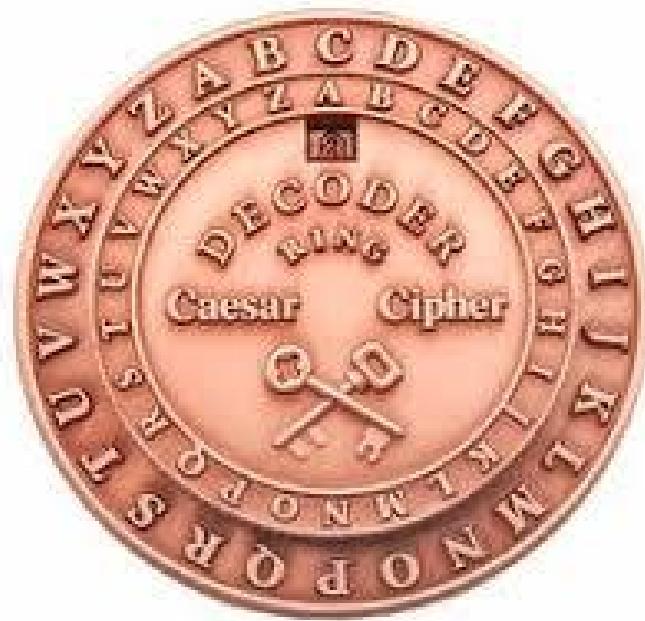
Greetings to all at Oxford. Many thanks for your letter and for the Summer examination package. All Entry Forms and Fees Forms should be ready for final despatch to the Syndicate by Friday 20th or at the very latest, I'm told, by the 21st. Admin has improved here, though there's room for improvement still; just give us all two or three more years and we'll really show you! Please don't let these wretched 16+ proposals destroy your basic O and A pattern. Certainly this sort of change, if implemented immediately, would bring chaos.

Sincerely yours,

Steganography also known as secret/covered writing, aims to hide secret messages in public ones. It focused on hiding the presence of secret information, or communication channel.

## 2.1 Overview: Brief History (3/20)

Cryptography (before 1976), the communication channel is public. It focused on transforming cleartext (plaintext) to ciphertext



# 2.1 Overview: Brief History (4/20)

Cryptography (before 1883)

Cryptography = Encryption + Decryption

Encryption= Encryption Algorithm

Decryption=Decryption Algorithm



Cryptography (1883-1976)

Cryptography = Encryption + Decryption

Encryption= Encryption Algorithm + **Secret Key**

Decryption=Decryption Algorithm + **Secret Key**

1. The system should be, if not theoretically unbreakable, unbreakable in practice.
2. The design of a system should not require secrecy, and compromise of the system should not inconvenience the correspondents ([Kerckhoff's principle](#)).
3. The key should be memorable without notes and should be easily changeable.
4. The cryptograms should be transmittable by telegraph.
5. The apparatus or documents should be portable and operable by a s
6. The system should be easy, neither requiring knowledge of a long list

# 2.1 Overview: Brief History (5/20)

Main techniques

- Shift cipher
- Substitution cipher
- Transposition cipher

Modern symmetric-key ciphers also use these techniques, but in a much more complex way.



## 2.1 Overview: Brief History (6/20)

1. Shift Cipher (Caesar Cipher): The Shift Cipher is a type of substitution cipher where each letter in the plaintext is shifted by a fixed number of positions in the alphabet.

**Example:** A shift by 3 would turn 'A' into 'D', 'B' into 'E', and so on.

**Key Difference:** It's a specific, consistent shift across all letters, making it a simple and predictable form of substitution.

2. Substitution Cipher: In a general substitution cipher, each letter in the plaintext is replaced by another letter or symbol, according to some predefined mapping. The mapping can vary for each letter.

**Example:** 'A' could be replaced with 'X', 'B' with 'K', etc. The substitution doesn't have to follow an ordered or shifted pattern.

**Key Difference:** Unlike the Shift Cipher, the substitution pattern can be arbitrary, providing more variation and complexity.

3. Transposition Cipher: In a transposition cipher, the positions of the letters are shuffled rather than replaced. The actual letters stay the same, but their order is changed according to a specific system.

**Example:** If the plaintext is "HELLO", the letters could be rearranged to "OLEHL".

**Key Difference:** Transposition ciphers change the position of the letters but keep the original letters intact, unlike substitution ciphers where the letters themselves are replaced.

# 2.1 Overview: Brief History (7/20)

## Claude Shannon

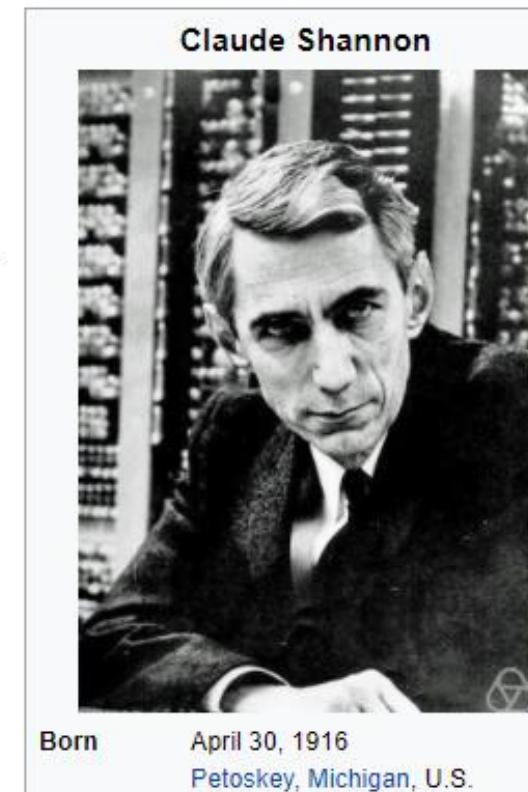
From Wikipedia, the free encyclopedia  
(Redirected from Claude E. Shannon)

**Claude Elwood Shannon** (April 30, 1916 – February 24, 2001) was an American mathematician, electrical engineer, and cryptographer known as "the father of information theory".<sup>[1][2]</sup> Shannon founded information theory with a landmark paper, "A Mathematical Theory of Communication", which he published in 1948.

He also founded digital circuit design theory in 1937, when—as a 21-year-old master's degree student at the Massachusetts Institute of Technology (MIT)—he wrote his thesis demonstrating that electrical applications of Boolean algebra could construct any logical numerical relationship.<sup>[3]</sup> Shannon contributed to the field of cryptanalysis for national defense during World War II, including his fundamental work on codebreaking and secure telecommunications.

### Contents [hide]

- 1 Biography
  - 1.1 Childhood
  - 1.2 Logic circuits
  - 1.3 Wartime research



1949 Shannon's Work  
(Contributions)

# 2.1 Overview: Brief History (8/20)

## Claude Shannon

From Wikipedia, the free encyclopedia  
(Redirected from Claude E. Shannon)

**Claude Elwood Shannon** (April 30, 1916 – February 24, 2001) was an American mathematician, electrical engineer, and cryptographer known as "the father of information theory".<sup>[1][2]</sup> Shannon founded information theory with a landmark paper, "A Mathematical Theory of Communication", which he published in 1948.

He also founded digital circuit design theory in 1937, when—as a 21-year-old master's degree student at the Massachusetts Institute of Technology (MIT)—he wrote his thesis demonstrating that electrical applications of Boolean algebra could construct any logical numerical relationship.<sup>[3]</sup> Shannon contributed to the field of cryptanalysis for national defense during World War II, including his fundamental work on codebreaking and secure telecommunications.

Contents [hide]  
1 Biography  
1.1 Childhood  
1.2 Logic circuits  
1.3 Wartime research



## 1949 Shannon's Work (Contributions)

- Shannon joined Bell Labs to work on fire-control systems and cryptography during World War II
- At the close of the war, he prepared a classified report for Bell Telephone Labs entitled "["A Mathematical Theory of Cryptography"](#)"
- In 1948, he published the paper "["A Mathematical Theory of Communication"](#)" (Information Theory)
- In 1949, a declassified version of paper was published in 1949 as "["Communication Theory of Secrecy Systems"](#)" in the Bell System Technical Journal. ([Study Cryptography with Mathematics](#))

# 2.1 Overview: Brief History (9/20)

Claude Shannon

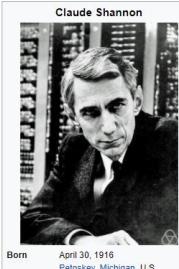
From Wikipedia, the free encyclopedia  
(Redirected from Claude E. Shannon)

**Claude Elwood Shannon** (April 30, 1916 – February 24, 2001) was an American mathematician, electrical engineer, and cryptographer known as "the father of information theory".<sup>[1][2]</sup> Shannon founded information theory with a landmark paper, "A Mathematical Theory of Communication", which he published in 1948.

He also founded digital circuit design theory in 1937, when—as a 21-year-old master's degree student at the Massachusetts Institute of Technology (MIT)—he wrote his thesis demonstrating that electrical applications of Boolean algebra could construct any logical numerical relationship.<sup>[3]</sup> Shannon contributed to the field of cryptanalysis for national defense during World War II, including his fundamental work on codebreaking and secure telecommunications.

Contents [hide]

1 Biography  
  1.1 Childhood  
  1.2 Logic circuits  
1.3 Wartime research



- What kinds of cryptosystem can be broken.  
*(Didn't consider the time cost. Hence called Theory research)*
- What kinds of cryptosystem **cannot** be broken.  
**(One-Time Pad)**

## 1949 Shannon's Work (Contributions)

One-Time Pad:  $K \oplus M$

- Secret key is as long as messages to be encrypted
- Each secret key will be used once only
- Choose secret key randomly.

# 2.1 Overview: Brief History (10/20)

1976 Modern Cryptography

1960 Computer Networks

1965 Computational Complexity Theory

1949 Shannon's Work

1946 Digital Computers

1883 Cryptography Principle

1936 Turing Machine

Classical Cryptography

# 2.1 Overview: Brief History (11/20)



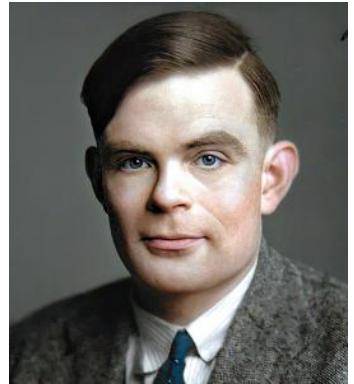
David Hilbert (1912)

- one of the most influential mathematicians of the 19th and early 20th centuries.
- He proposed 32 open problems in mathematics in 1900
- In 1928, Hilbert Asked: Is Mathematics **Complete**, is it **Consistent**, and is it **Decidable**?  
Decision Problem: Ent-schei-dungs-problem (German)

Born	23 January 1862 <a href="#">Königsberg</a> or <a href="#">Wehlau, Prussia</a>
Died	14 February 1943 (aged 81) <a href="#">Göttingen</a> , <a href="#">Nazi Germany</a>

Can any problem be decidable with True or false?  
Can any mathematic theorem be proved to be true or false?

# 2.1 Overview: Brief History (12/20)



Alan Turing

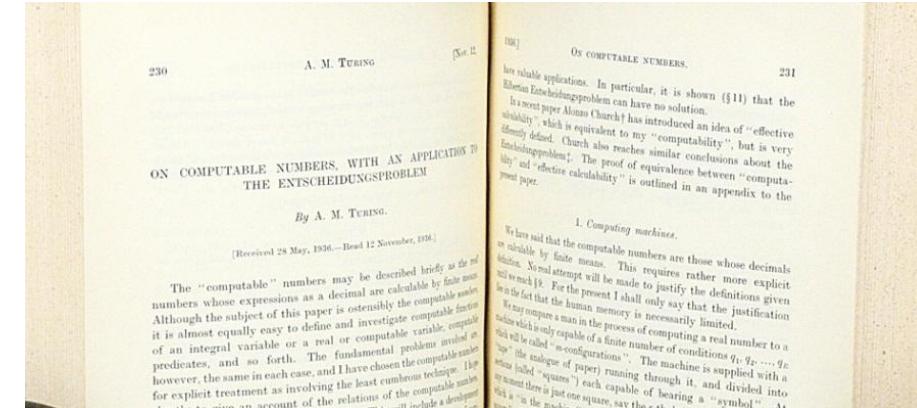
In 1936, Turing published his paper:

"On Computable Numbers, with an Application to the Entscheidungsproblem"

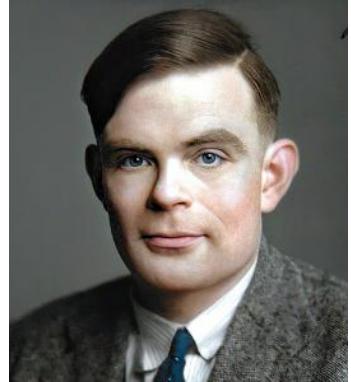


Some problems must be undecidable!

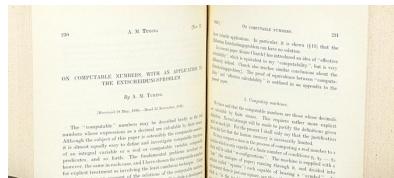
Turing Machine



# 2.1 Overview: Brief History (12+/20)

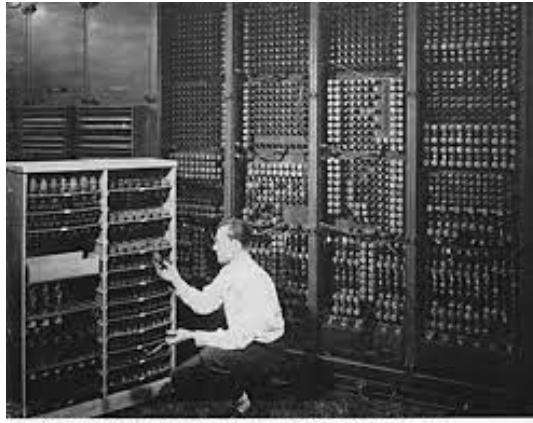


Alan Turing



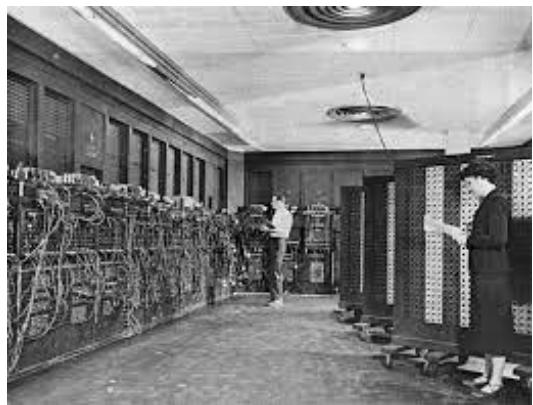
- Turing machine is not a computer but a computation (mathematical) model, which is to formally **define computing ability of machines**.
- This computation model can capture all computations by humans.
- If a computing device meets the computation model, it can perform computations as smart as humans. (theory only)
- A digital computer must provide computing ability captured by Turing machine. (**We have computers 10 years later, 1940s**)
- A turning machine was later applied in studying **computational complexity theory**.

# 2.1 Overview: Brief History (13/20)



Replacing a bad tube mount checking among ENIAC's 15,000 possibilities.

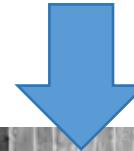
1.Computers  
1940s



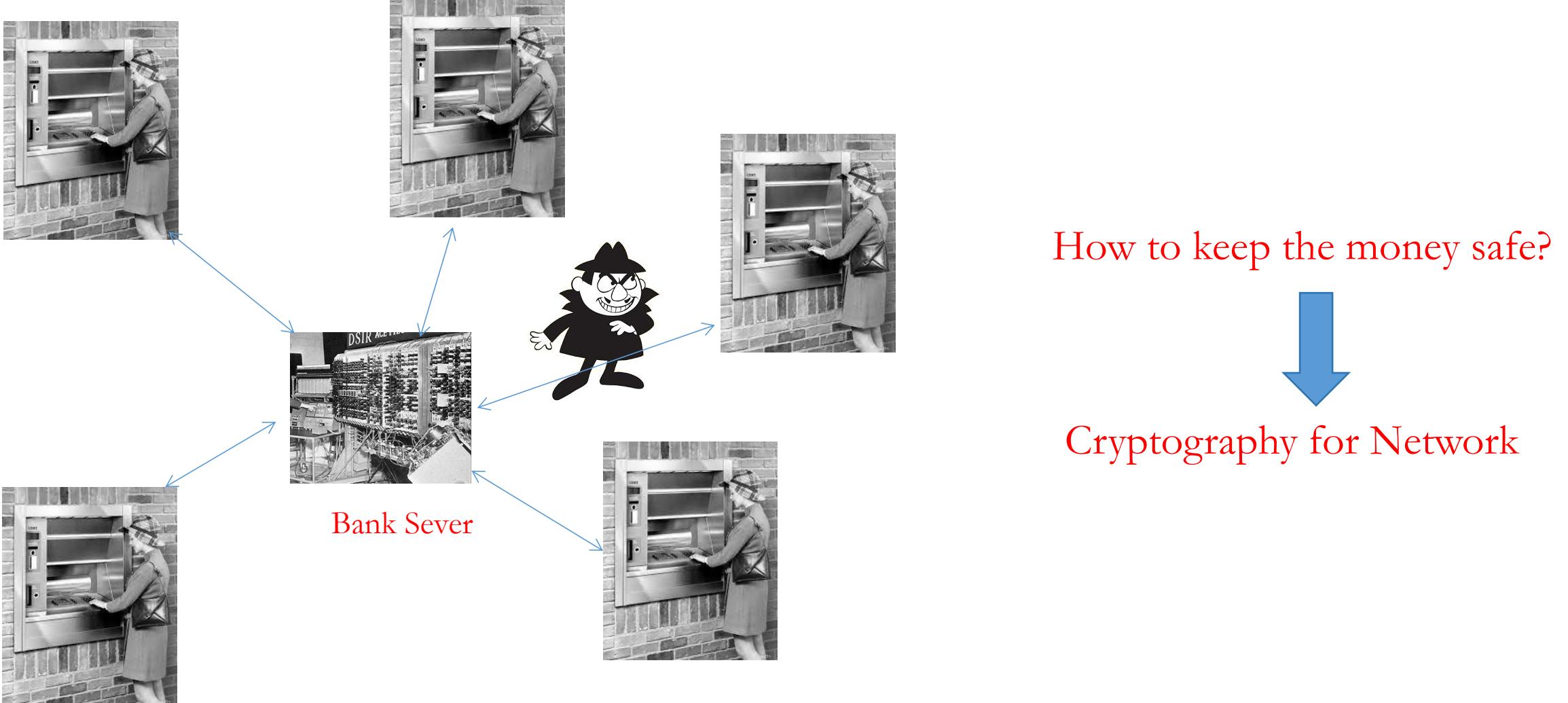
2.Networks  
1960s



3. ATM



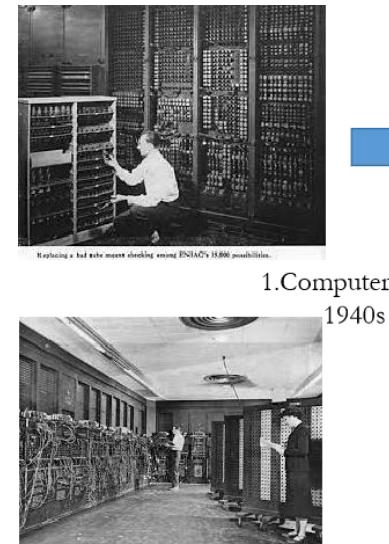
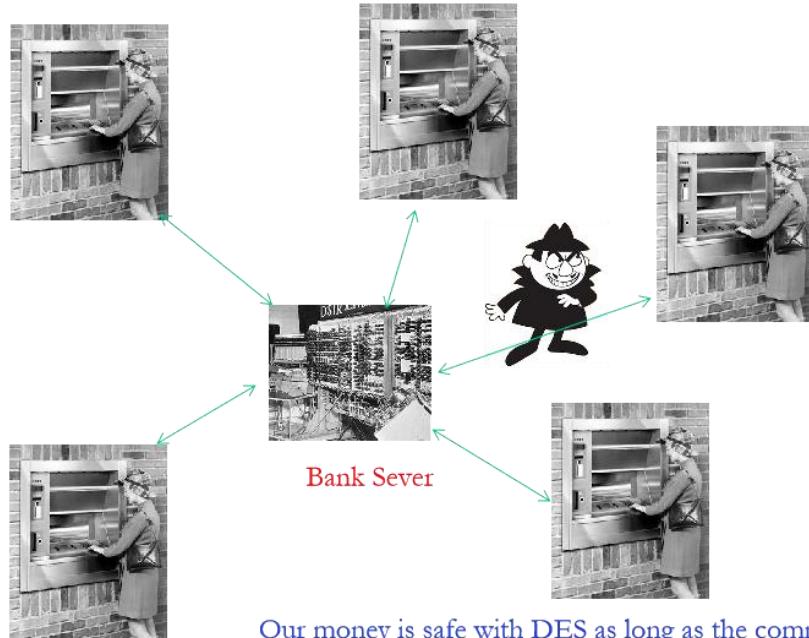
# 2.1 Overview: Brief History (14/20)



## 2.1 Overview: Brief History (15/20)

- Computer Networks need protection with cryptography (encryption and decryption)
- Computers run encryption and decryption
- Need to design cryptography for business applications
- Cryptography should be strong against very powerful computers
- The first cryptography called Modern cryptography is DES in 1970s

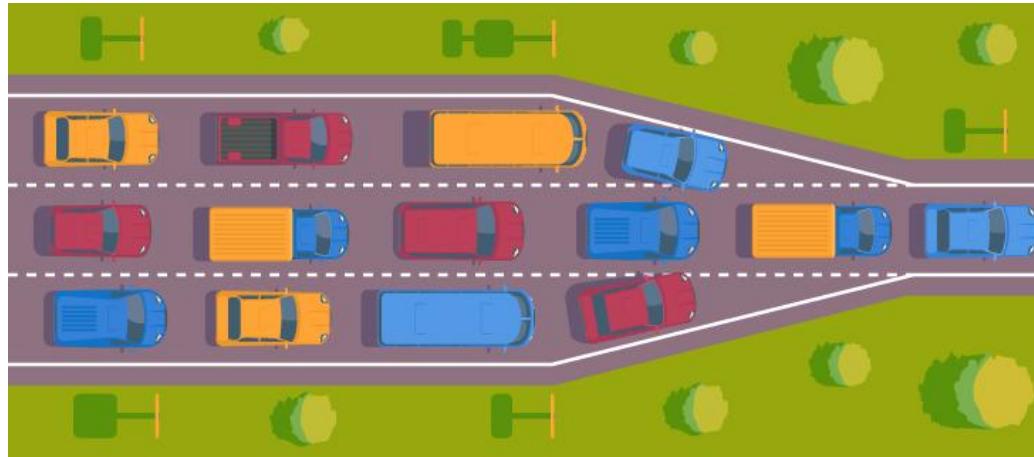
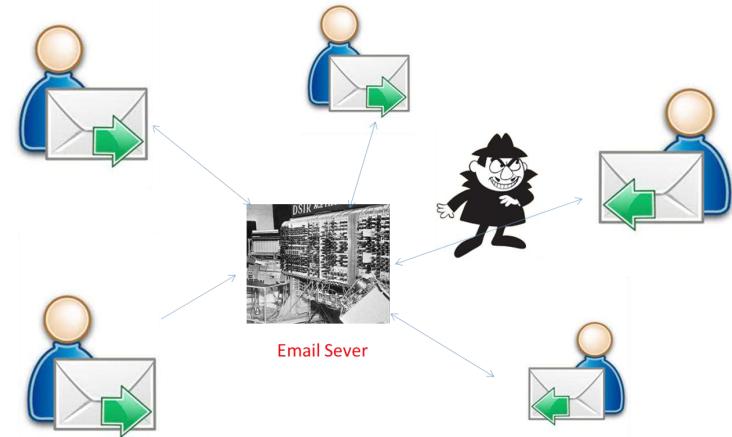
# 2.1 Overview: Brief History (16/20)



2.Networks  
1960s



# 2.1 Overview: Brief History (17/20)



- DES requires a shared key between client and server, or a shared key must be securely distributed to clients and server.
- It is insecure to send this key directly over the internet. How to solve the problem of key distribution?

# 2.1 Overview: Brief History (18/20)

1976 Modern Cryptography

1949 Shannon's Work

1883 Cryptography Principle

Classical Cryptography

1960 Computer Networks

1946 Digital Computers

1936 Turing Machine

1965 Computational Complexity Theory

Complexity theory identifies some mathematical problems that are computationally difficult for computers to solve. It provides a foundation for cryptography by ensuring that breaking cryptographic systems is as hard as solving these difficult problems.

# 2.1 Overview: Brief History (19/20)

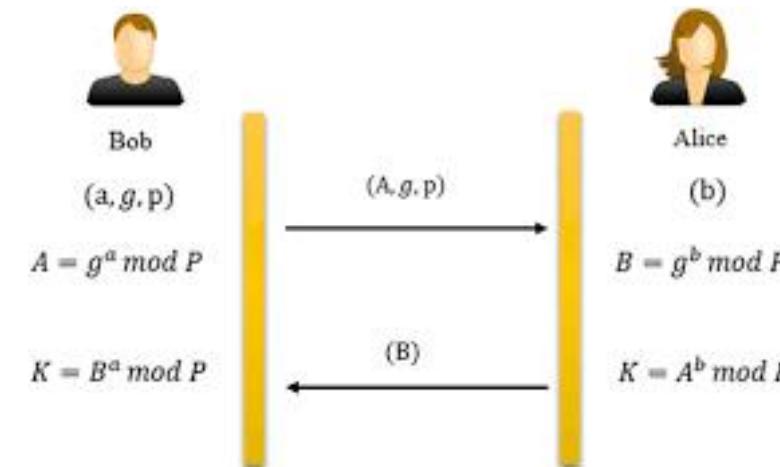
## 1976 Modern Cryptography

### 1976 Public-Key Cryptography

New Directions in Cryptography  
W. Diffie & M.E. Hellman



"We stand today on the brink of a  
revolution in cryptography"



1949 Shannon's Work

1960 Computer Networks

1965 Computational Complexity Theory

# 2.1 Overview: Brief History (20/20)

Changes after 1976:

- Symmetric K → Asymmetric (pk, sk)

Do the communication parties have a shared secret key or not?

- Confidentiality → Confidentiality or Integrity or Authenticity

Which security goal should be cared?

- Message → Data or identity

What to be protected?

The answers to these three questions decide the selection of primitive!

## 2.2 Confidentiality

## 2.2 Confidentiality: Symmetric-Key Encryption (1/4)

Syntax:

- $\text{KeyGen}(\lambda) \rightarrow K$ : This algorithm generates a secret key  $K$ .
- $\text{Enc}(K, M) \rightarrow CT$ : This algorithm encrypts message  $M$  using  $K$  into  $CT$ .
- $\text{Dec}(K, CT) \rightarrow M$ : This algorithm decrypts  $CT$  using  $K$  to get  $M$ .

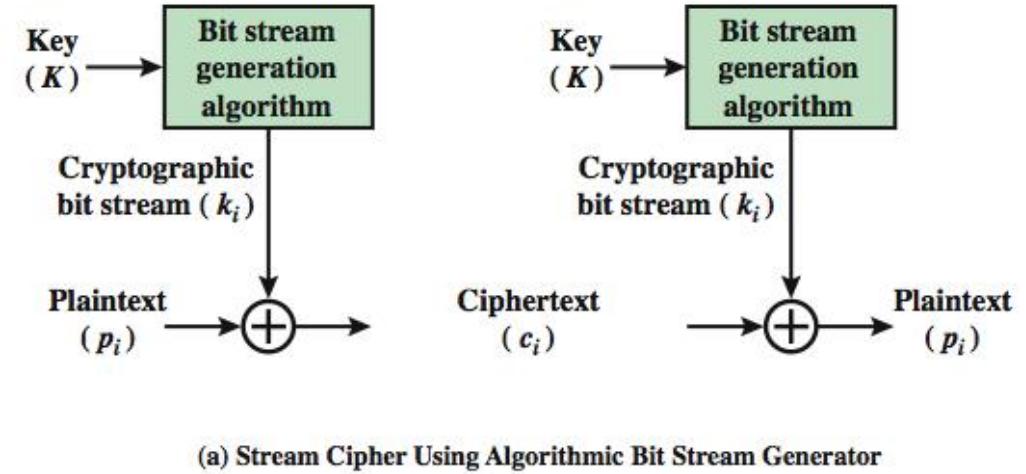
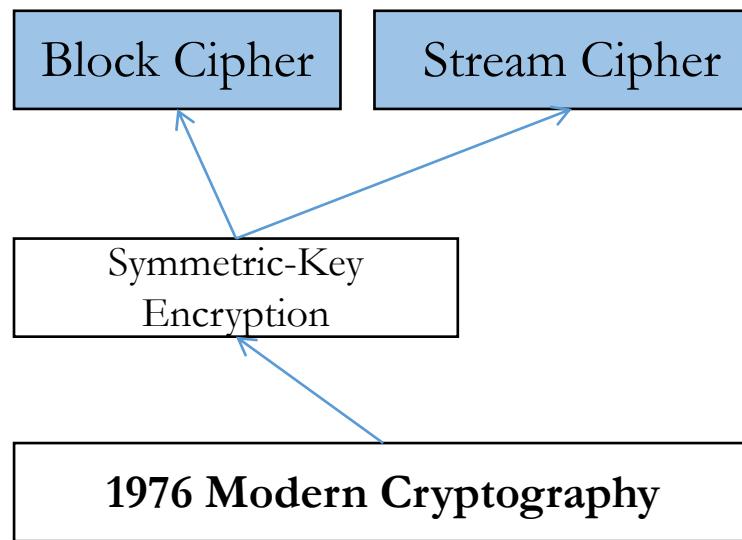
Alice and Bob have a shared secret key  $K$ . Alice encrypts  $M$  using  $K$  to get  $CT$ , which is sent to Bob. Bob can then decrypt  $CT$  with the secret key  $K$  to get  $M$ .

## 2.2 Confidentiality: Symmetric-Key Encryption (2/4)

When designing an SKE scheme.....

Scheme:  
AES , DES

Scheme: RC4.



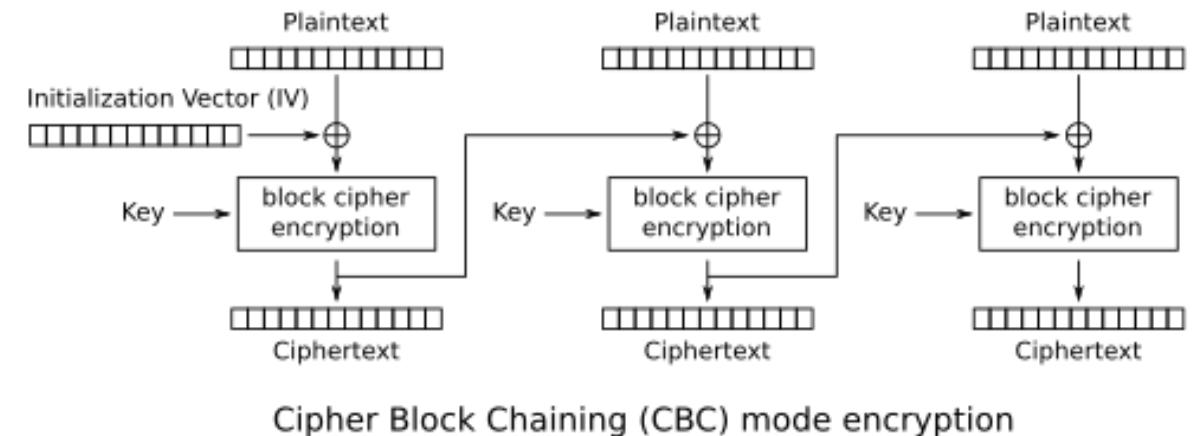
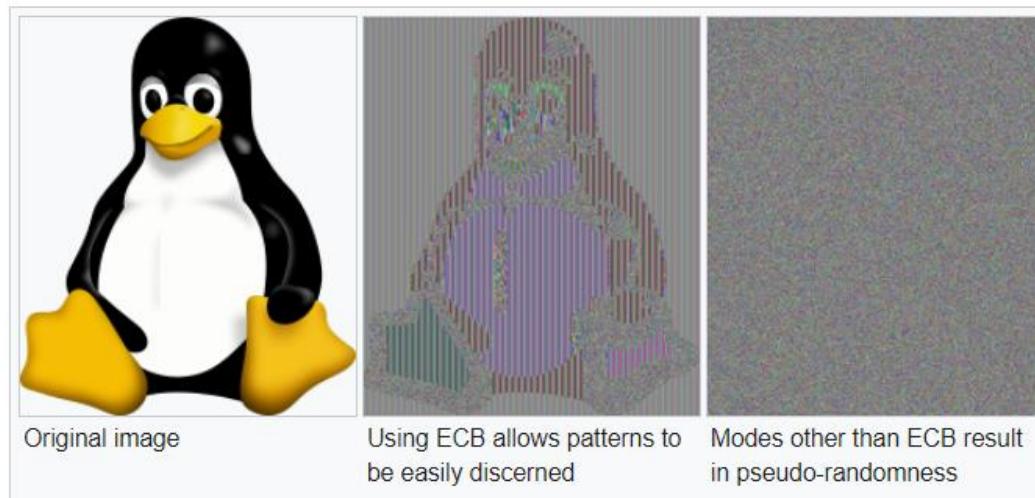
## 2.2 Confidentiality: Symmetric-Key Encryption (3/4)

Block Cipher:

- **Block Length:** This is the size of the data block that the cipher encrypts at one time. Common block sizes are 64 bits or 128 bits. Larger block sizes provide better security by reducing the chances of patterns in the plaintext appearing in the ciphertext.
- **Key Space:** This refers to the range of possible keys that can be used in the cipher. A larger key space means more possible keys, making brute-force attacks harder. For example, a 128-bit key space means there are  $2^{128}$  possible keys.
- **Modes of Operation:** (when encrypting long messages using the same K) These define how the block cipher processes multiple blocks of data, ensuring security beyond single blocks. CBC (Cipher Block Chaining) and GCM (Galois/Counter Mode). Each mode has different ways of handling block dependencies and randomization to enhance security.

## 2.2 Confidentiality: Symmetric-Key Encryption (4/4)

- **Modes of Operation:** These define how the block cipher processes multiple blocks of data, ensuring security beyond single blocks. CBC (Cipher Block Chaining) and GCM (Galois/Counter Mode). Each mode has different ways of handling block dependencies and randomization to enhance security.



## 2.2 Confidentiality: Public-Key Encryption (1/2)

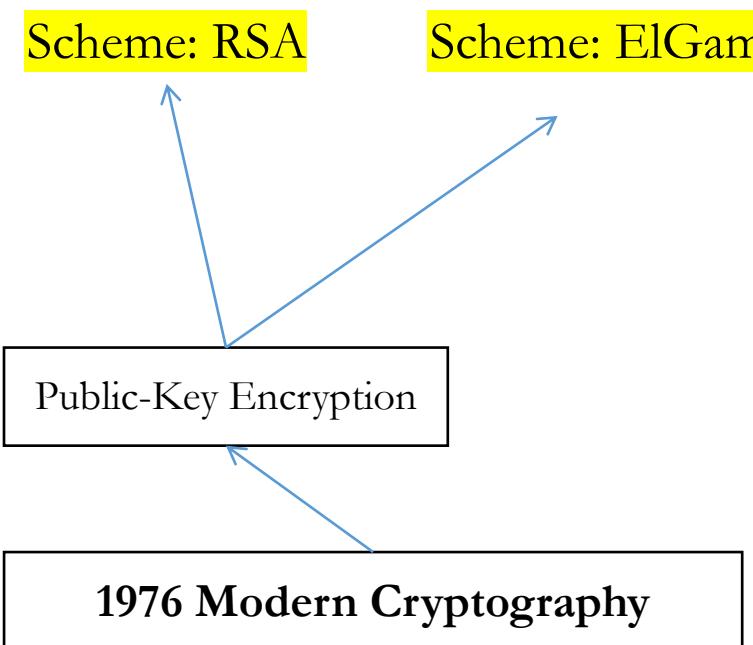
Syntax:

- $\text{KeyGen}(\lambda) \rightarrow (\text{pk}, \text{sk})$ : This algorithm generates a key pair  $(\text{pk}, \text{sk})$ . Here,  $\text{pk}$  is the public key and  $\text{sk}$  is the secret key.
- $\text{Enc}(\text{pk}, M) \rightarrow CT$ : This algorithm encrypts message  $M$  using  $\text{pk}$  into  $CT$ .
- $\text{Dec}(\text{sk}, CT) \rightarrow M$ : This algorithm decrypts  $CT$  using  $\text{sk}$  to get  $M$ .

Alice encrypts  $M$  using  $\text{pk}$  (belonging to Bob) to get  $CT$ , which is sent to Bob. Bob can then decrypt  $CT$  with the secret key  $\text{sk}$  without the need of sharing a secret key with Alice. Therefore, Bob does not need to know Alice before.

## 2.2 Confidentiality: Public-Key Encryption (2/2)

When designing a PKE scheme.....



- Public key encryption (PKE) schemes, like RSA, also have a block length that limits the size of the message they can encrypt directly, which is based on the key size.
- However, this block length is less critical because PKE is typically combined with symmetric-key encryption in what's called **hybrid encryption**.
- In hybrid encryption, the PKE is used to encrypt a symmetric key  $K$ , while the actual data is encrypted by the symmetric-key encryption using  $K$ , which can efficiently encrypt large blocks of data.

## 2.3 Integrity

## 2.3 Integrity: Hash Function (1/3)

Syntax:

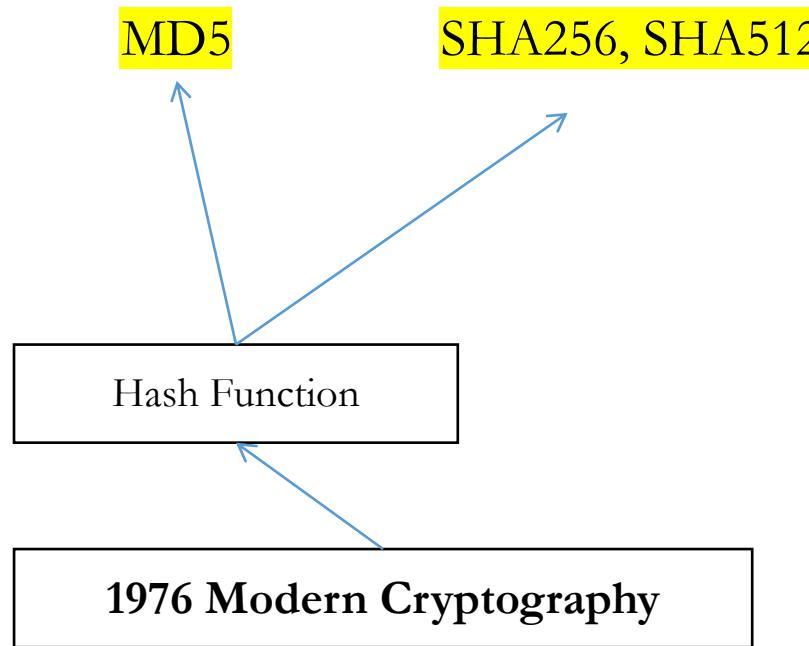
$$H: \{0, 1\}^* \rightarrow \{0, 1\}^n, n \text{ is a security parameter}$$

**Application:** Let  $x$  be some message.  $H(x)$  is called the message digest.  $x$  can be of arbitrary length while  $h(x)$  has a fixed length. (160 bits, 256 bits, 512 bits). **We want to use  $H(x)$  to represent  $x$ .**

## 2.3 Integrity: Hash Function (2/3)

$H: \{0, 1\}^* \rightarrow \{0, 1\}^n$

- It is **easy** to compute  $H(x)$  for a given  $x$ .



- It is **hard** to compute  $x$  from  $H(x)$ .
- It is **hard** to find different  $x_1$  and  $x_2$  such that  $H(x_1)=H(x_2)$ .

## 2.3 Integrity: Hash Function (3/3)

Syntax:

$$H: \{0, 1\}^* \rightarrow \{0, 1\}^n, n \text{ is a security parameter}$$

**Application:** Alice uploads a 1G size software S to a cloud, calculates H(S), and tells Bob the value H(S) (via a secure approach) which is

e3b0c-44298-fc1c1-49afb-f4c89-96fb9-2427a-e41e4-649b9-34ca4-95991-b7852-b855.

After downloading the software S from the cloud, Bob can check whether the software has been altered or not.

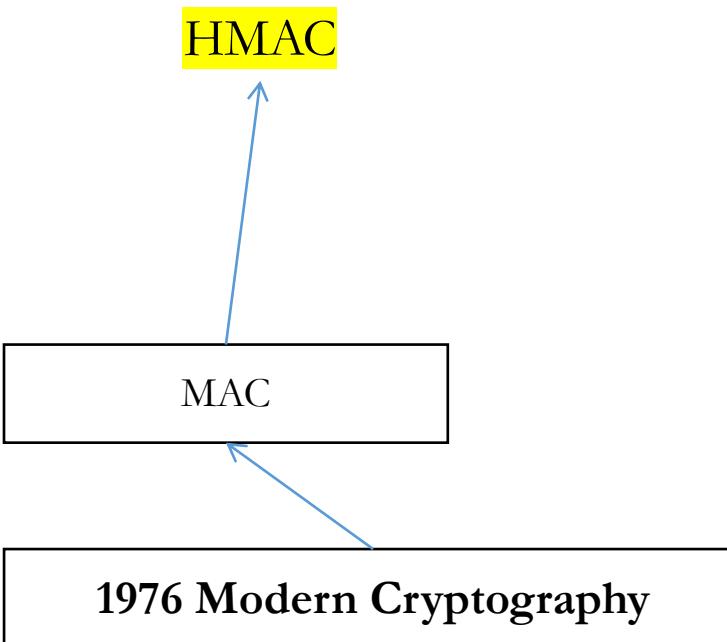
## 2.3 Integrity: Message-Authentication Code (1/3)

Syntax:

- $\text{KeyGen}(\lambda)$ : This algorithm generates a symmetric key  $K$ .
- $\text{MAC}(M, K)$ : This algorithm generates a tag  $T$ .

Application: Alice and Bob has a shared secret key  $K$ . Alice sends  $M$  and  $T$  to Bob. After receiving them, Bob uses  $M$  and  $K$  to compute  $T'$  and compare  $T=T'?$  If they are equal, it means that  $M$  is indeed sent from Alice.

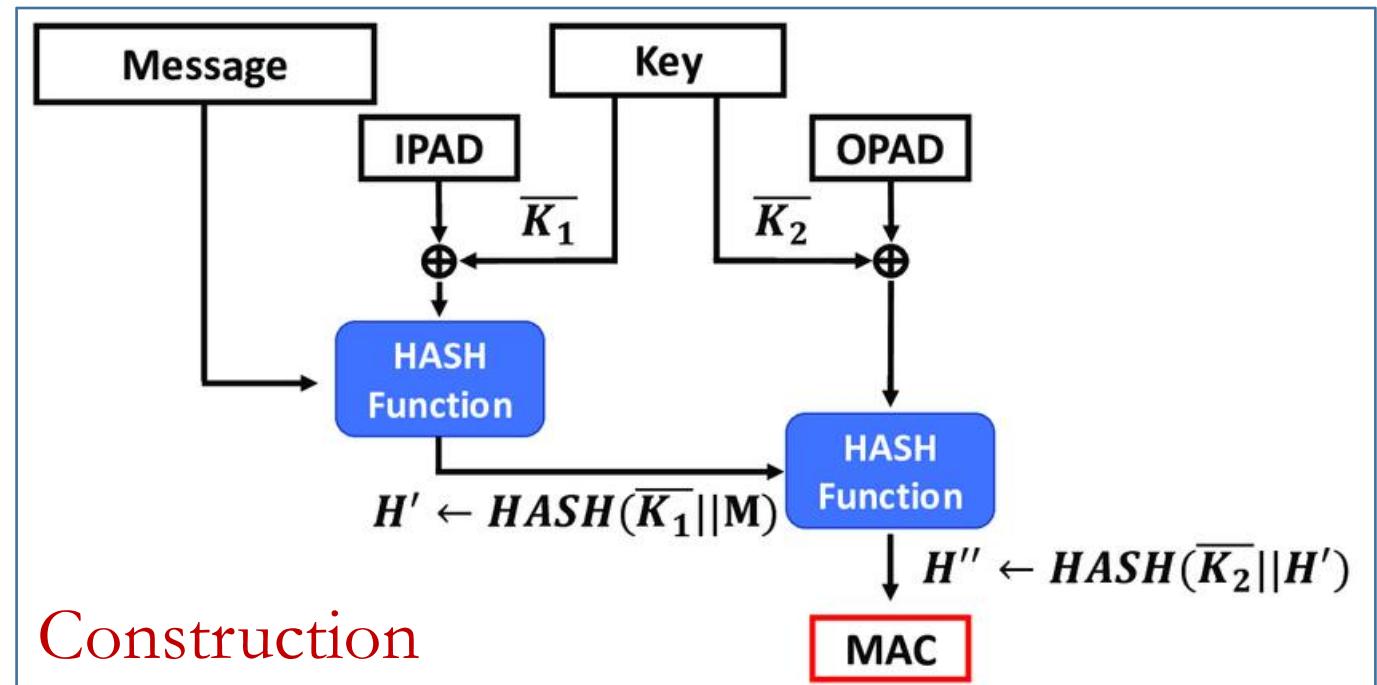
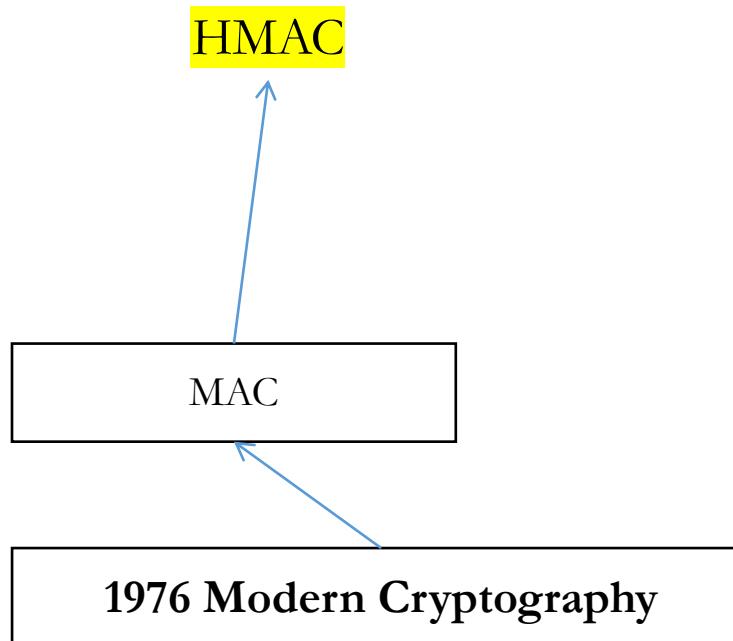
## 2.3 Integrity: Message-Authentication Code (2/3)



- Preventing unauthorised modification of data with a shared secret key. No one can compute the tag for  $M^*$  if  $sk$  is unknown.
- Different from error detection code, which is for unintentional modification of data (e.g., due to noise)
- Both involve a checksum
  - MAC needs a secret key
  - Error detection code does not use a secret key

## 2.3 Integrity: Message-Authentication Code (3/3)

HMAC: Use a hash function to construct MAC  
 $H(M \parallel K)$  where H is a hash function



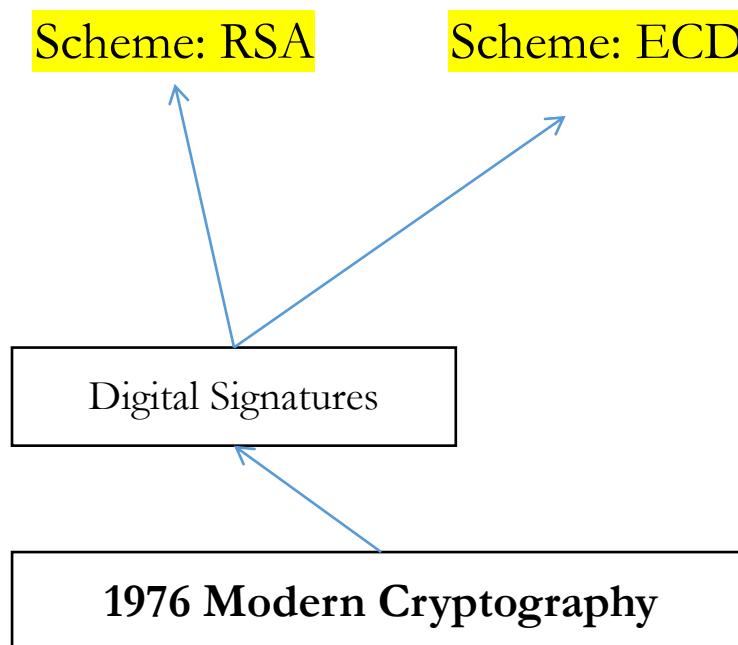
## 2.3 Integrity: Digital Signatures (1/2)

Syntax:

- $\text{KeyGen}(\lambda)$ : This algorithm generates a key pair  $(\text{pk}, \text{sk})$
- $\text{Sign}(M, \text{sk})$ : This algorithm generates a signature  $S$  on  $M$  using  $\text{sk}$ .
- $\text{Verify}(m, S, \text{pk})$ : This algorithm verifies the validity of  $S$  on  $M$  using  $\text{pk}$ .

Application: Bob knows that  $\text{pk}$  belongs to Alice. Alice sends  $M$  and  $S$  to Bob. After receiving them, Bob uses  $\text{pk}$  to verify whether or not  $S$  is a valid signature on  $M$ . If yes, it means that  $M$  is indeed sent from Alice.

## 2.3 Integrity: Digital Signatures (2/2)



- In general, a digital signature scheme can only sign messages **from a small space (256 bits)**.
- A hash function is applied to sign any arbitrary messages.
- Sign  $H(M)$  instead of  $M$ , namely hash-then-sign mechanism.

## 2.4 Authenticity

protocol, hard to use syntax and algorithms to present it.

## 2.4 Authenticity: Authentication

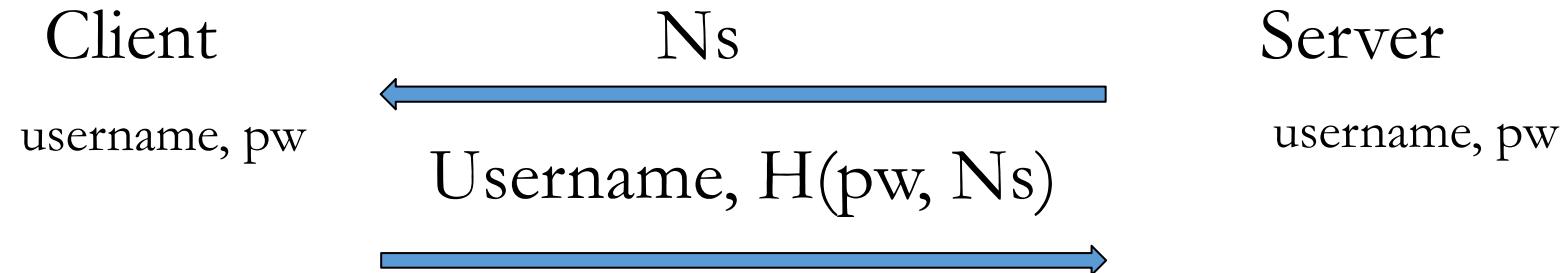
If Bob wants to authenticate whether  $m$  is from Alice:

- They can use MAC (as long as they have a shared secret key)
- They can use digital signatures (as long as Bob knows that  $pk$  belongs to Alice)

Note: Cryptography always needs some assumptions to achieve something.

## 2.4 Authenticity: Identification (1/4)

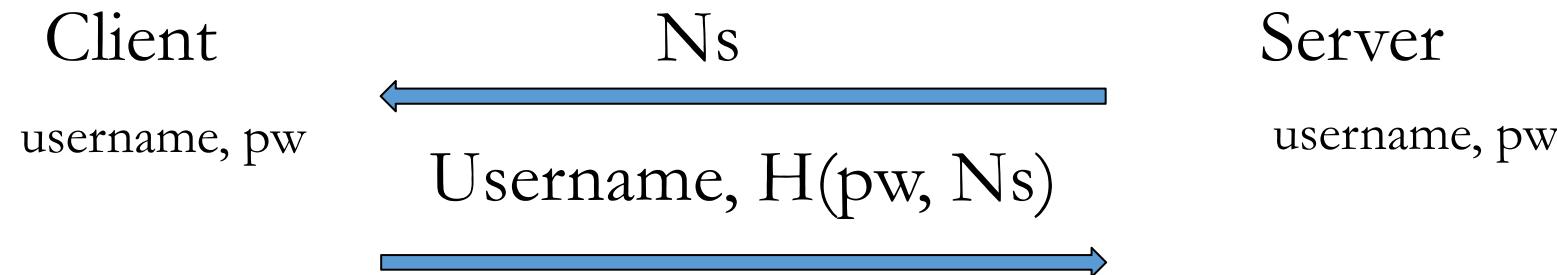
Motivation: A wants to prove who he/she is to B.



- The server sends a **nonce**  $N_s$  to the client as a challenge.
  - (A nonce is a unique, arbitrary number that can only be **used once** in a communication or authentication process. It is often used to prevent replay attacks, ensuring that each interaction is fresh and not reused by an attacker.)
- The client gives a **fresh** response based on pw and  $N_s$  in each session.

## 2.4 Authenticity: Identification (2/4)

Motivation: A wants to prove who he/she is to B.

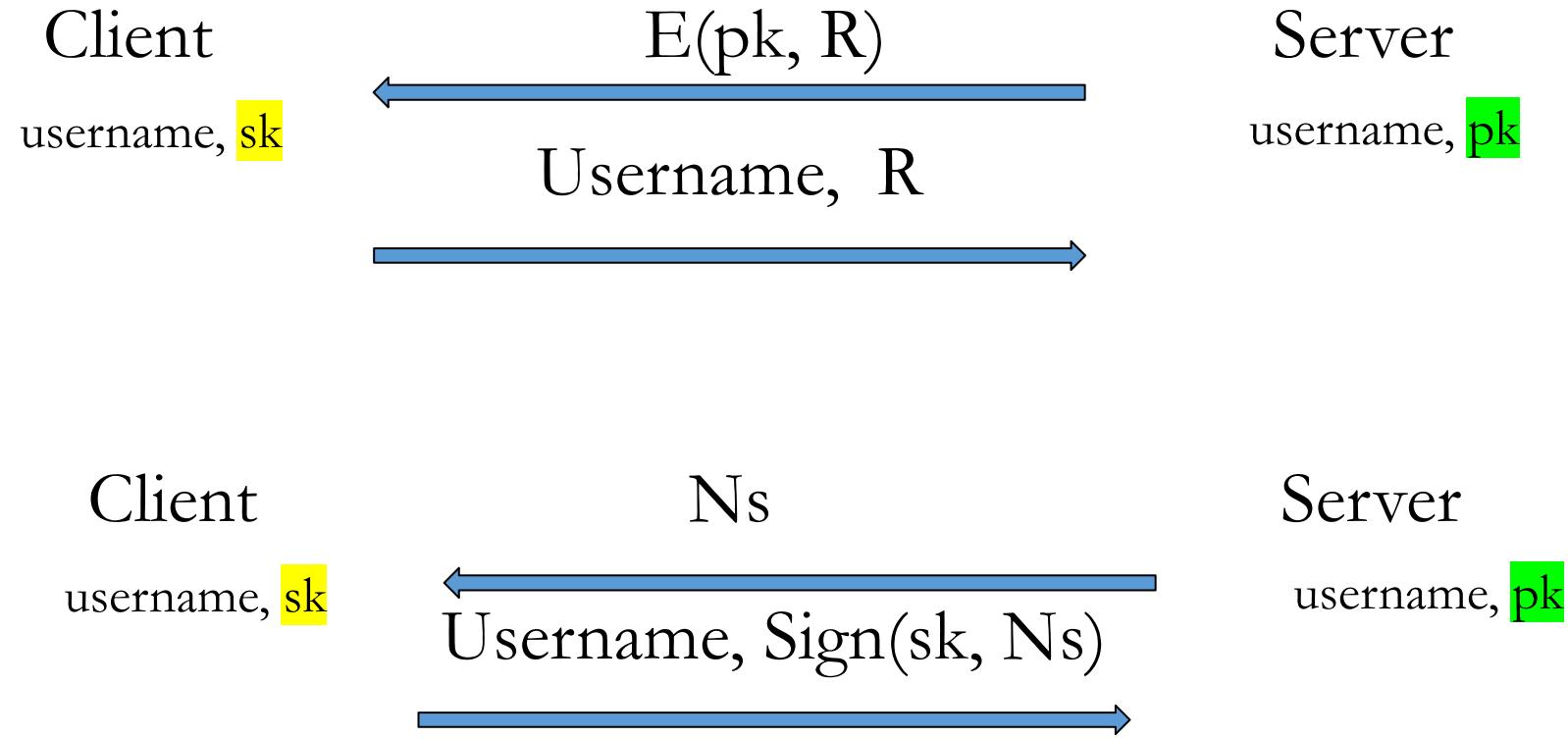


Nonce:

- A long enough random number (hard to generate two same numbers).
- A timestamp such as Unix Time (seconds since January 1, 1970).
  - The Unix time for January 1, 2025 is 1735689600

## 2.4 Authenticity: Identification (3/4)

Motivation: A wants to prove who he/she is to B.

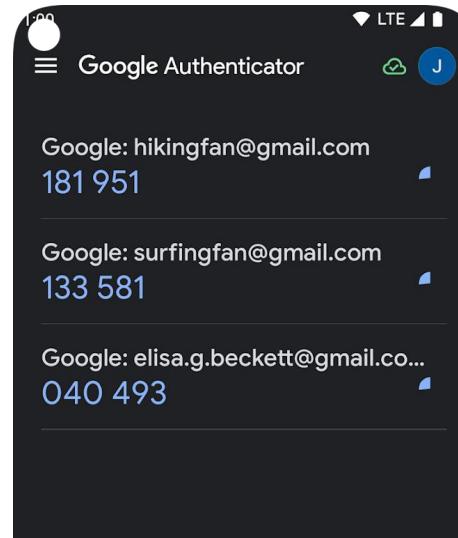


## 2.4 Authenticity: Identification (4/4)

Identification might be called: Authentication → Two-Factor Authentication



Without keypad, H(K, time)



Google Authenticator



With keypad, H(K, R)

- One factor is pw and the second factor is another secret related to a device.
- A unique and random key is installed in each device. Each device also have a **Serial Number**.
- Client is registered with (ID, pw, SN). The producer (server) knows the random key for each SN.

## 2.4 Authenticity: Summary

No matter it is an authentication or identification between A and B. It requires:

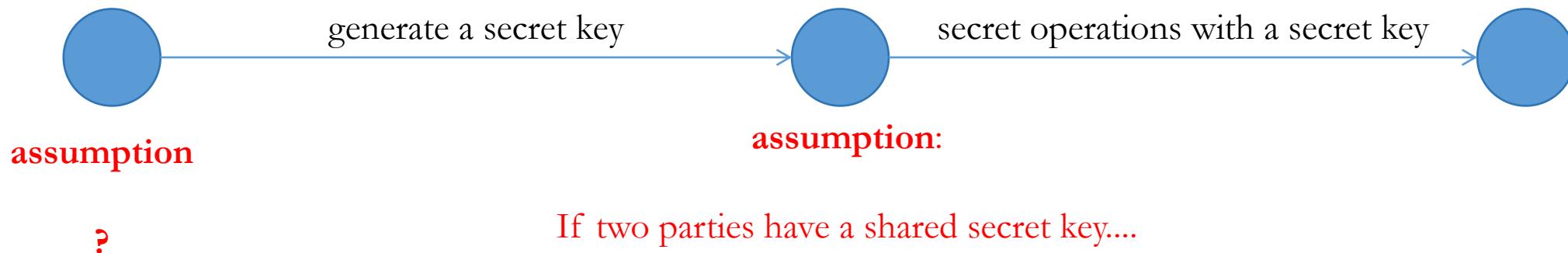
- They have a shared secret (K or pw), or
- Verifiers know the public key of publisher/prover.

Note: This is another assumption (pre-condition) for having authenticity.

Question:

Why network protocols designers didn't consider authenticity from the beginning?

## 2.5 Key Establishment



## 2.5 Key Establishment: Definition

Motivation: To generate a shared secret key.

Key Establishment is a process or protocol whereby a **shared** secret becomes available to two or more parties, for subsequent cryptographic use.

The shared secret key is a **good key** if:

- The key is random and fresh (**key freshness**);
- The key is known only to A and B (and any mutually trusted parties) (**confidentiality**)

## 2.5 Key Establishment: Two Categories (1/2)

There are two main categories of key establishment:

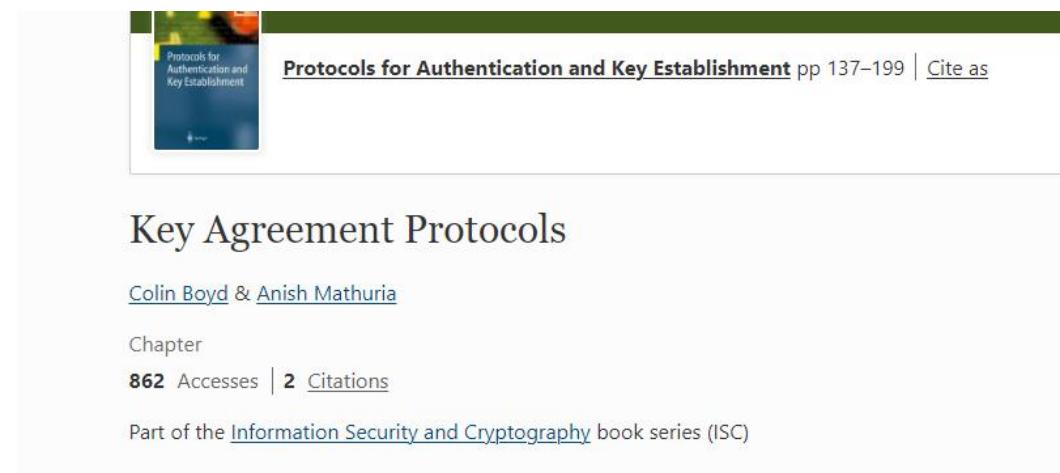
- Key **transport**, where one party generates and securely transfers the key to the other parties.
- Key **agreement/exchange**, where the parties all obtain a shared secret, which is itself a function of input from all parties.

$$\text{Key Establishment} = \text{Key Transport} + \text{Key Agreement}$$

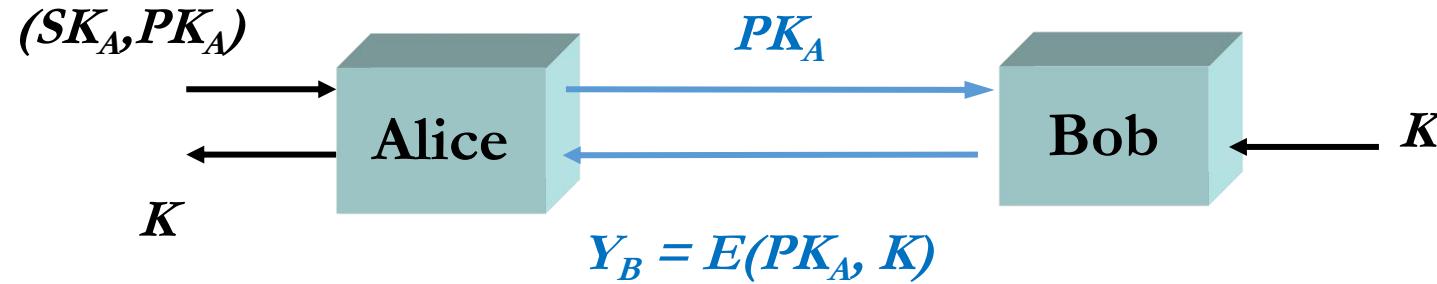
## 2.5 Key Establishment: Two Categories (2/2)

Key Establishment= Key Transport + Key Agreement

- Key agreement has become much more popular than key transport in recent years. There is an intuitive feeling (from academic community) that key agreement is '**'fairer'** than key transport, and Can result in higher **quality random** keys than key transport.



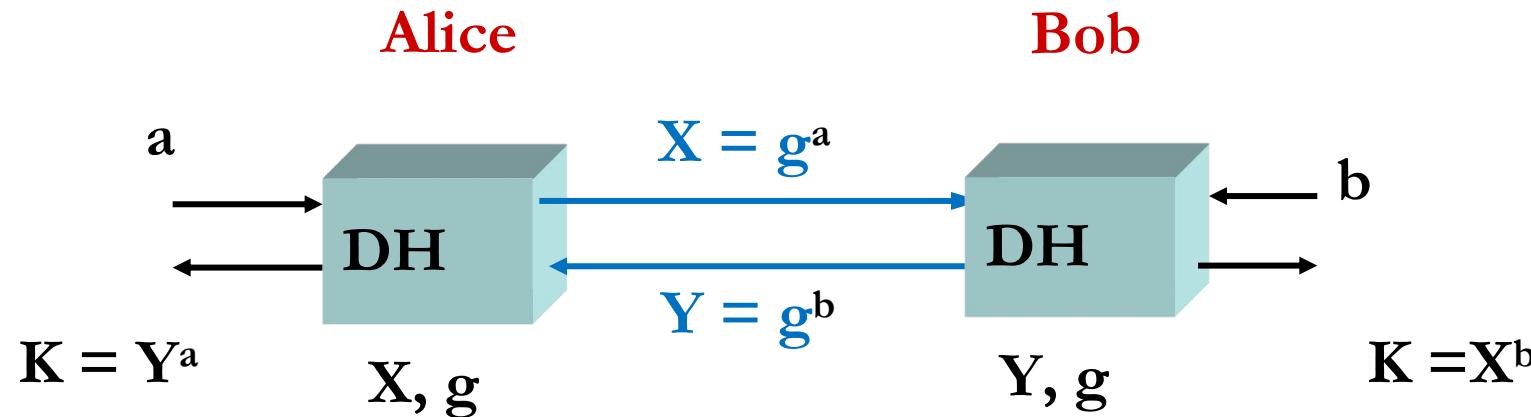
## 2.5 Key Establishment: Key Transport



- Alice sends her public key  $PK_A$  to Bob.
- Bob chooses a random secret key  $K$ , encrypts it using  $PK_A$ , and sends the ciphertext to Alice.
- Alice decrypts the ciphertext to get  $K$ .

Now, Alice and Bob have the shared secret key  $K$ .

## 2.5 Key Establishment: Diffie-Hellman Key Agreement (1/4)

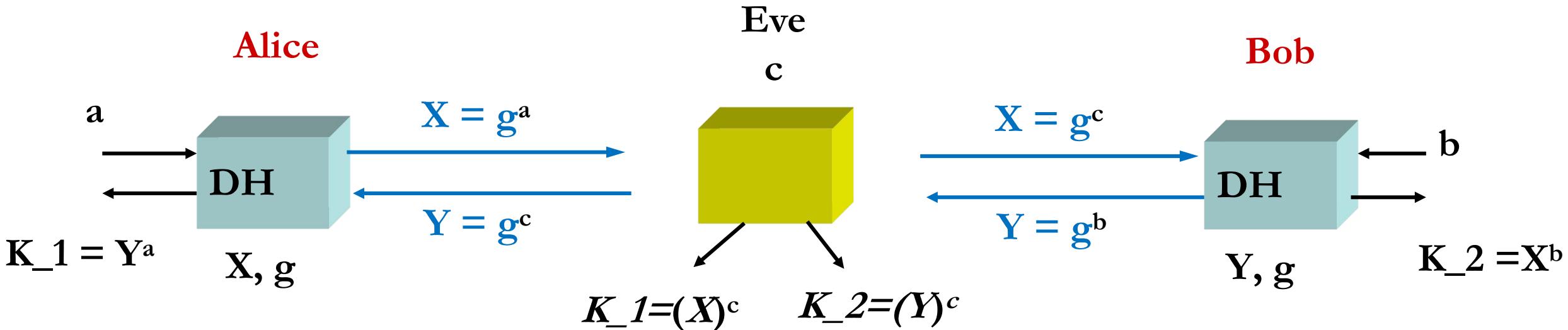


Cyclic Group: G is the group, g is a generator,  $g^x$  ( $g^x$ ) must be inside G, and  $g^{p+x} = g^x$ .

- Alice chooses a random integer  $a$ , computes  $X = g^a$ , and sends X to Bob.
- Bob chooses a random integer  $b$ , computes  $Y = g^b$ , and sends Y to Alice.
- Alice computes  $K = Y^a = g^{ab}$ , while Bob computes  $K = X^b = g^{ab}$ .
- Others cannot compute  $x$  from  $g^x$  (this is a hard problem)

Now, Alice and Bob have the shared secret key K.

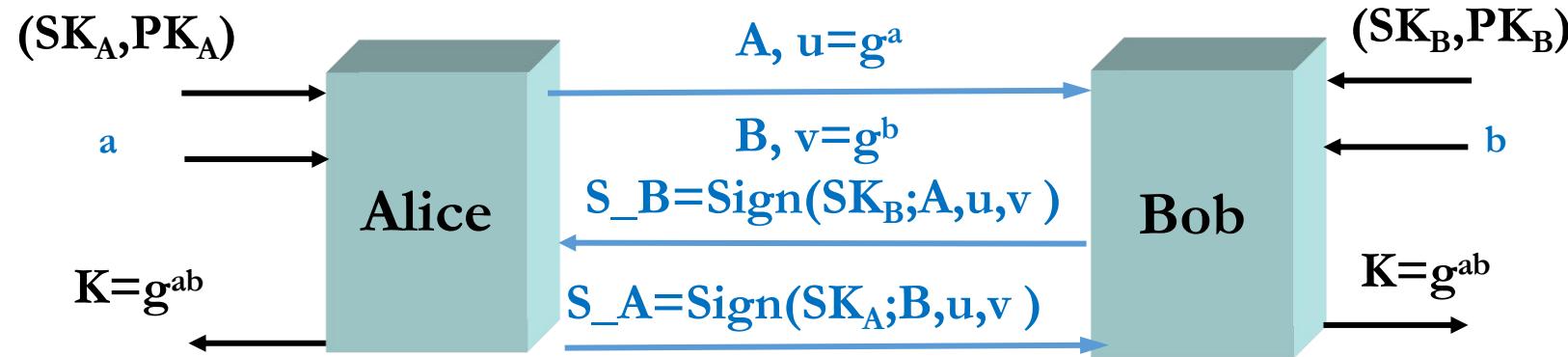
## 2.5 Key Establishment: Diffie-Hellman Key Agreement (2/4)



### Man-in-the-Middle Attack

- Alice chooses a random integer  $a$ , computes  $X = g^a$ , and sends  $X$  to Bob. While  $X$  is replaced with  $g^c$  by Eve, which is sent to Bob.
- Bob chooses a random integer  $b$ , computes  $Y = g^b$ , and sends  $Y$  to Alice. While  $Y$  is replaced with  $g^a$  by Eve, which is sent to Alice.
- Alice computes  $K_1 = Y^a = g^{ac}$ , while Bob computes  $K_2 = X^b = g^{bc}$ .
- When Alice encrypts  $M$  using  $K_1$ , this can be decrypted and re-encrypted using  $K_2$ .

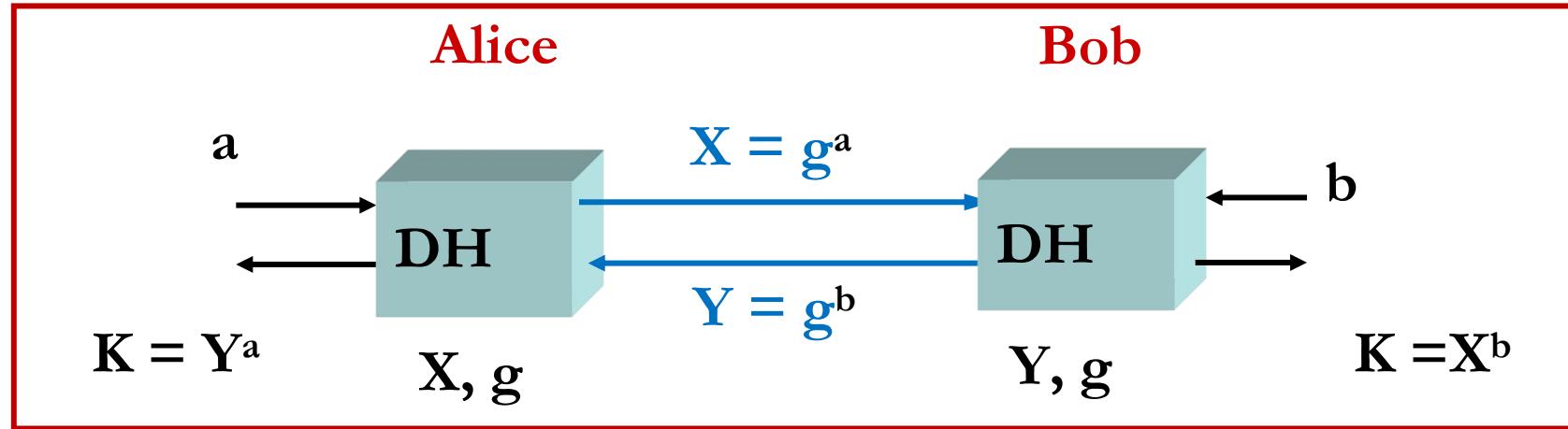
## 2.5 Key Establishment: Diffie-Hellman Key Agreement (3/4)



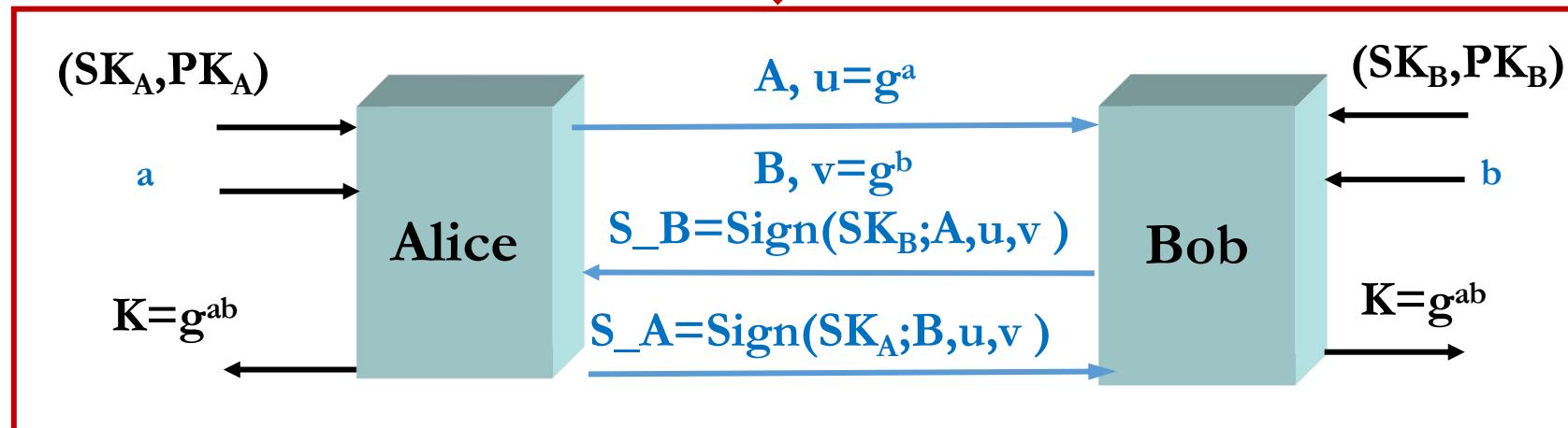
- 1:  $A \rightarrow B: A, u$
- 2:  $B \rightarrow A: B, v, \text{Sig}_B(A, u, v),$
- 3:  $A \rightarrow B: \text{Sig}_A(B, u, v)$

In this revised protocol, signatures are applied to verify all received data.

## 2.5 Key Establishment: Diffie-Hellman Key Agreement (4/4)



price to pay?



## 2.6 From Theory to Real World

## 2.6 From Theory to Real World: 1/9



FROM

**Theory** (Syntax and Construction)

TO

**Real world** (Implementation and Programming)

## 2.6 From Theory to Real World: 2/9

### 1. Confusing Descriptions

Theory: sign m using sk (secret key)

Real World: encrypt m using sk (not very popular but it is there)

## 2.6 From Theory to Real World: 3/9

### 1. Confusing Descriptions

encryption key = K or pk

signing key = sk

secret key = K

private key = sk

shared key = K

session key = ?

Note: you might need to guess based on the context.

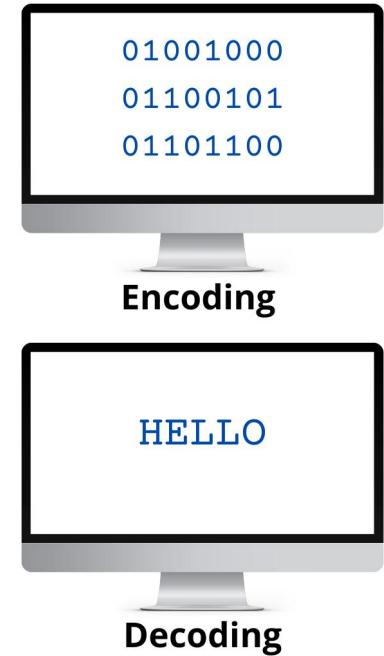
# 2.6 From Theory to Real World: 4/9

## 2. Encoding and Padding

Theory: encrypt  $m$  using  $K$  (simple and compact)

Real World:

- $m$  must be encoded into bytes before operation
- $m$  must be padded into such as  $512*k$  bits for an integer  $k$



## 2.6 From Theory to Real World: 5/9

### 3. Algorithm Parameters

Theory: encrypt  $m$  using  $K$  to get  $CT = \text{Enc}(K, m)$

AES: encrypt  $m$  using  $K$  with parameters to get  $CT = \text{Enc}(K, m)$

- Block size= 128, 256, 512?
- Mode= CBC or GCM ?

# 2.6 From Theory to Real World: 6/9

## 3. Algorithm Parameters

Theory: encrypt  $m$  using  $pk$

RSA:  $pk = (n, e)$ , which is composed of multiple elements

$n=0xb74118f199fd688f010479ad3f2332a4f2f0fc85b59c7521fb5b220f62f7332f66a7eb10822b218d00b2698b510e010d01dbf9d5e2c5c30b7b2f92c7b5ed07e275e6cd6898ed2ba86017e63abf233a20d2d1339e7deca47894915b1d872f48f970cadb373b75f2b4a515d06f59593151bb0d4b699cc4b5944d7fa11c26f3258bc94efdf01$

# 2.6 From Theory to Real World: 7/9

## 4. Assumption and Implementation

Theory: choose a random  $x$  and compute  $g^x$ .

Real World:

- How to choose random numbers/strings?
- How to implement the building block (cyclic groups)?
- Compatable: the chosen  $x$  can be used to compute  $g^x$ .

# 2.6 From Theory to Real World: 8/9

## 5. Syntax and Syntax

Theory:  $\text{Enc}(\text{pk}, \text{m})$  is to encrypt  $\text{m}$  using  $\text{pk}$  with input  $(\text{pk}, \text{m})$

Real World: (Python as the example)

```
from Crypto.Cipher import PKCS1_OAEP
from Crypto.PublicKey import RSA

message = "Secret message".encode() # Convert the message to bytes
recipient_key = RSA.import_key(public_key) # Load the recipient's public key
cipher_rsa = PKCS1_OAEP.new(recipient_key) # Create cipher object using public key
ciphertext = cipher_rsa.encrypt(message) # Encrypt the message
```

# 2.6 From Theory to Real World: 9/9

## 5. Syntax and Syntax

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad, unpad
key=get_random_bytes(16)
iv=get_random_bytes(16)
cipher=AES.new(key,AES.MODE_CBC,iv)
message="Fuchun's class is very boring".encode()
padded_message=pad(message,AES.block_size)
ciphertext=cipher.encrypt(padded_message)
```

# **END OF L2**

NETWOCK  
SECURITY

SECURITY

NETWOCK SECURITY



# Network Security

## Lecture 3

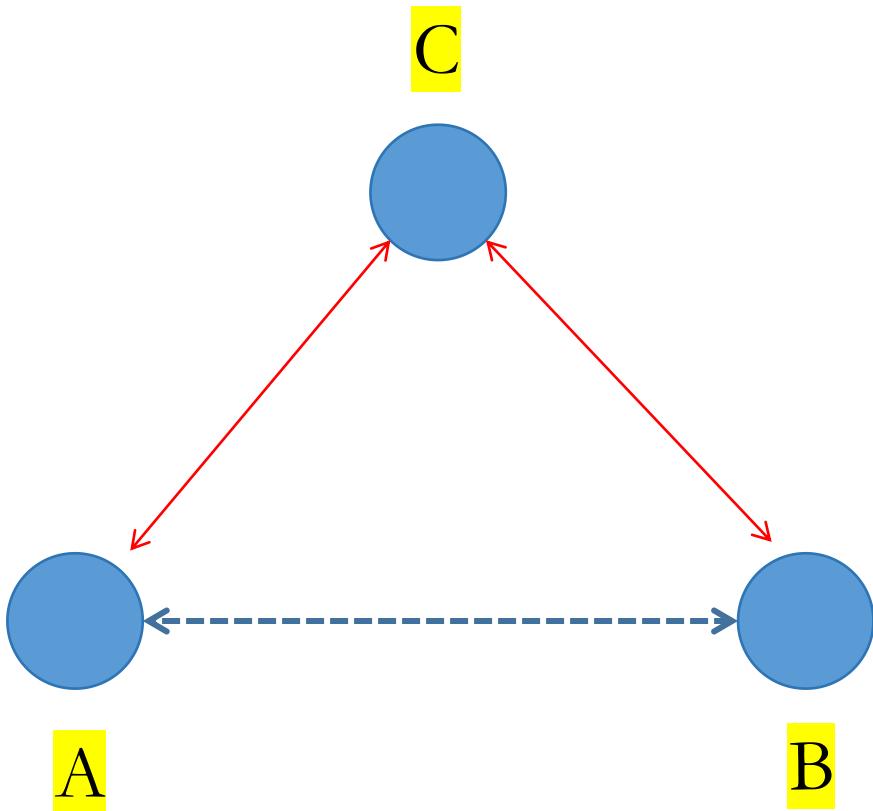
A/Prof. Fuchun Guo

# Outline

- ❑ Centralised ID Authentication
- ❑ Centralised Key Distribution
- ❑ Centralised pk Authentication
- ❑ Email Security Protocols
- ❑ SSH Protocol

## **3.1 Centralized ID Authentication**

## 3.1 Centralized ID Authentication: Motivation (1/2)



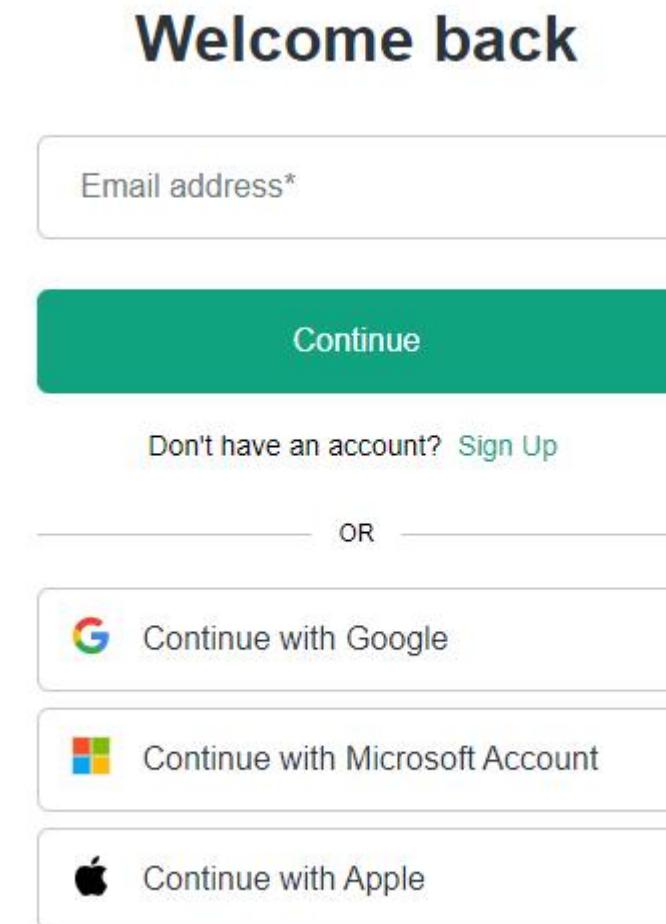
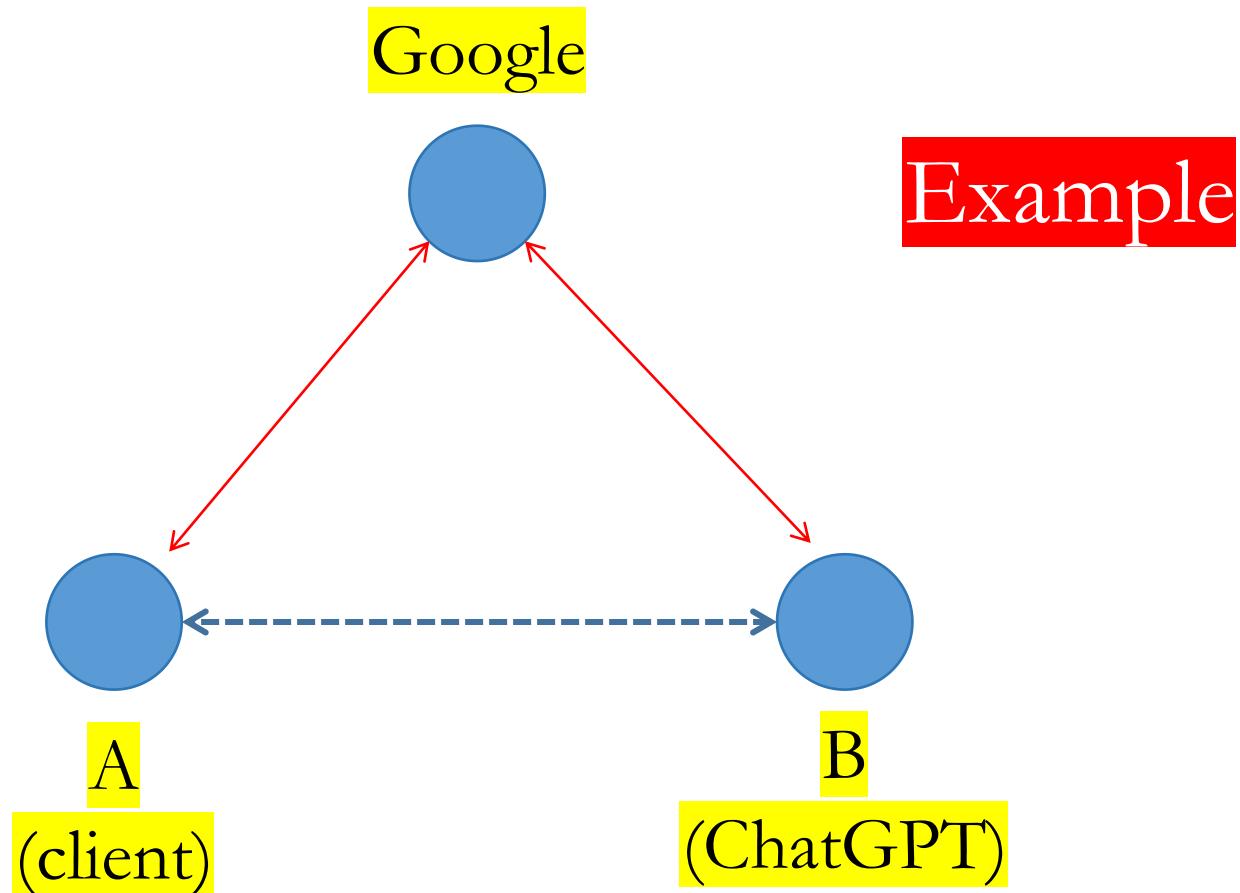
### Assumptions:

- C is the trusted third party by A and B.
- A and C have a shared secret key  $K_A$ .
- B and C have a shared secret key  $K_B$ .

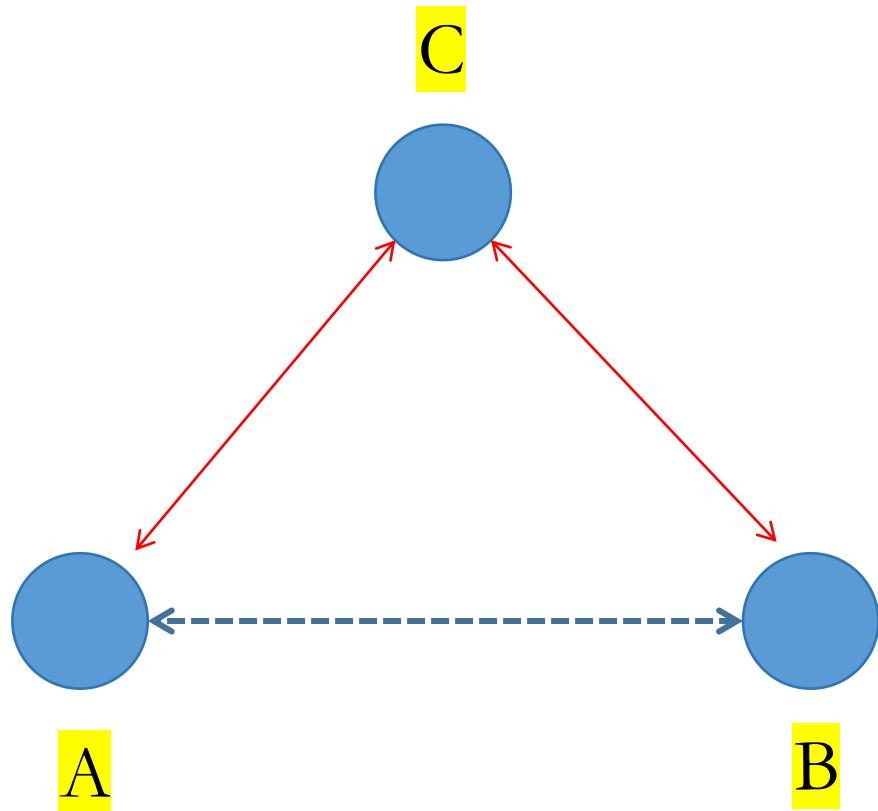
### Motivation:

A wants to prove to B who he/she is

# 3.1 Centralized ID Authentication: Motivation (2/2)



### 3.1 Centralized ID Authentication: Overview

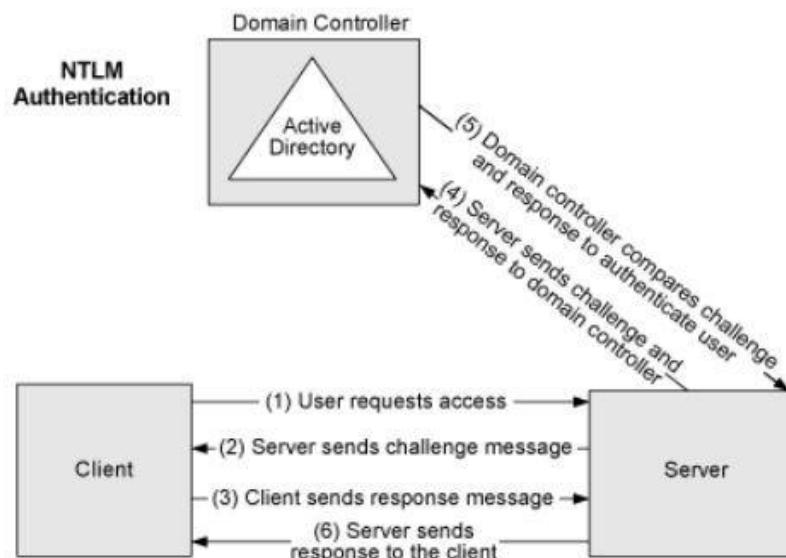


- ❖ **A→C (Authentication Server):** Pls give me something as the proof to show to B that I am the user A.
- ❖ **B:** Verify A directly with that proof.

- ❖ A: Generate something for B
- ❖ **B→C(Authentication Server):** Pls help me check the proof claiming that the one wants to talk to me is the user A.

Key Question: Who contacts C?

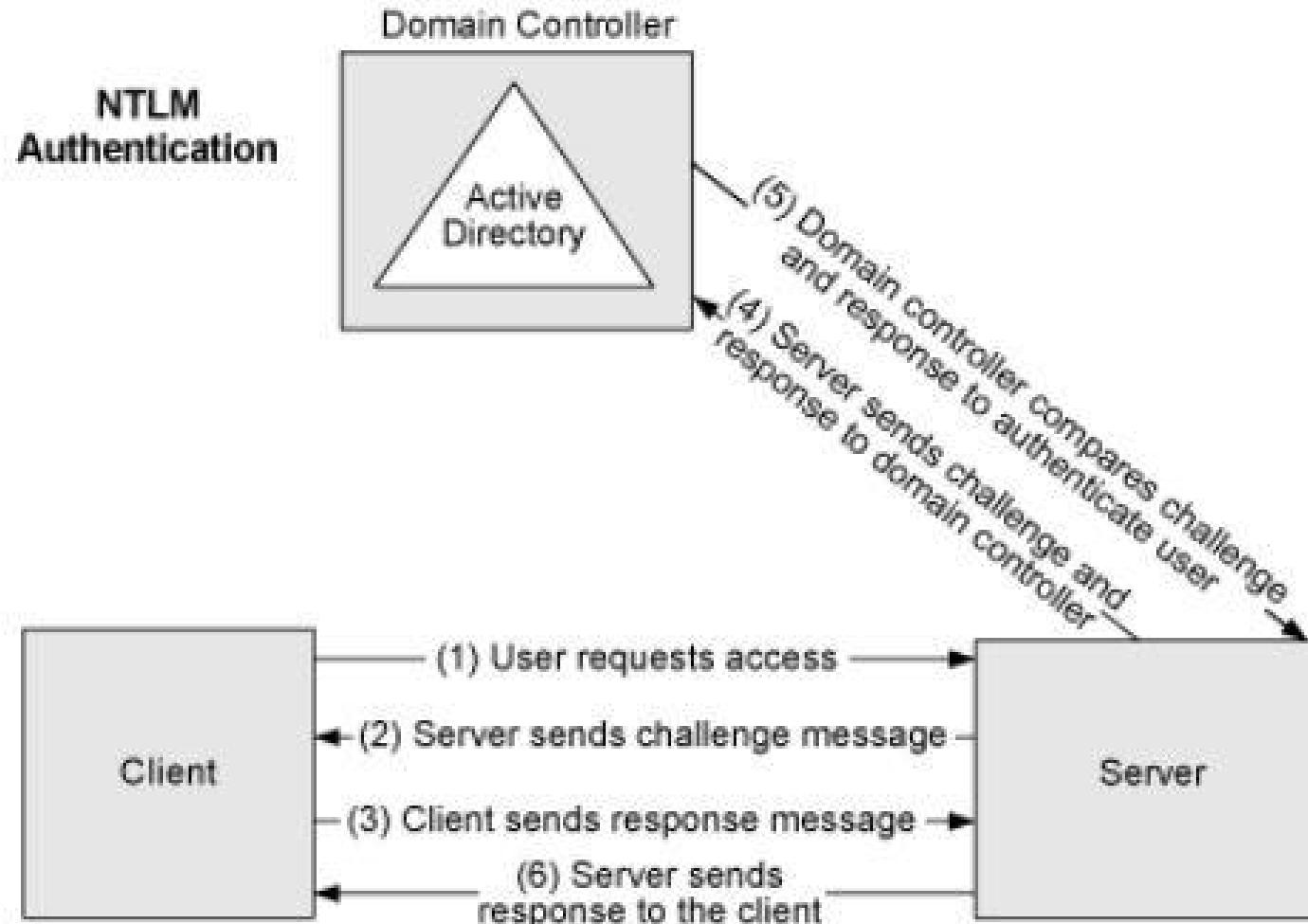
# 3.1 Centralized ID Authentication: NTLM (1/6)



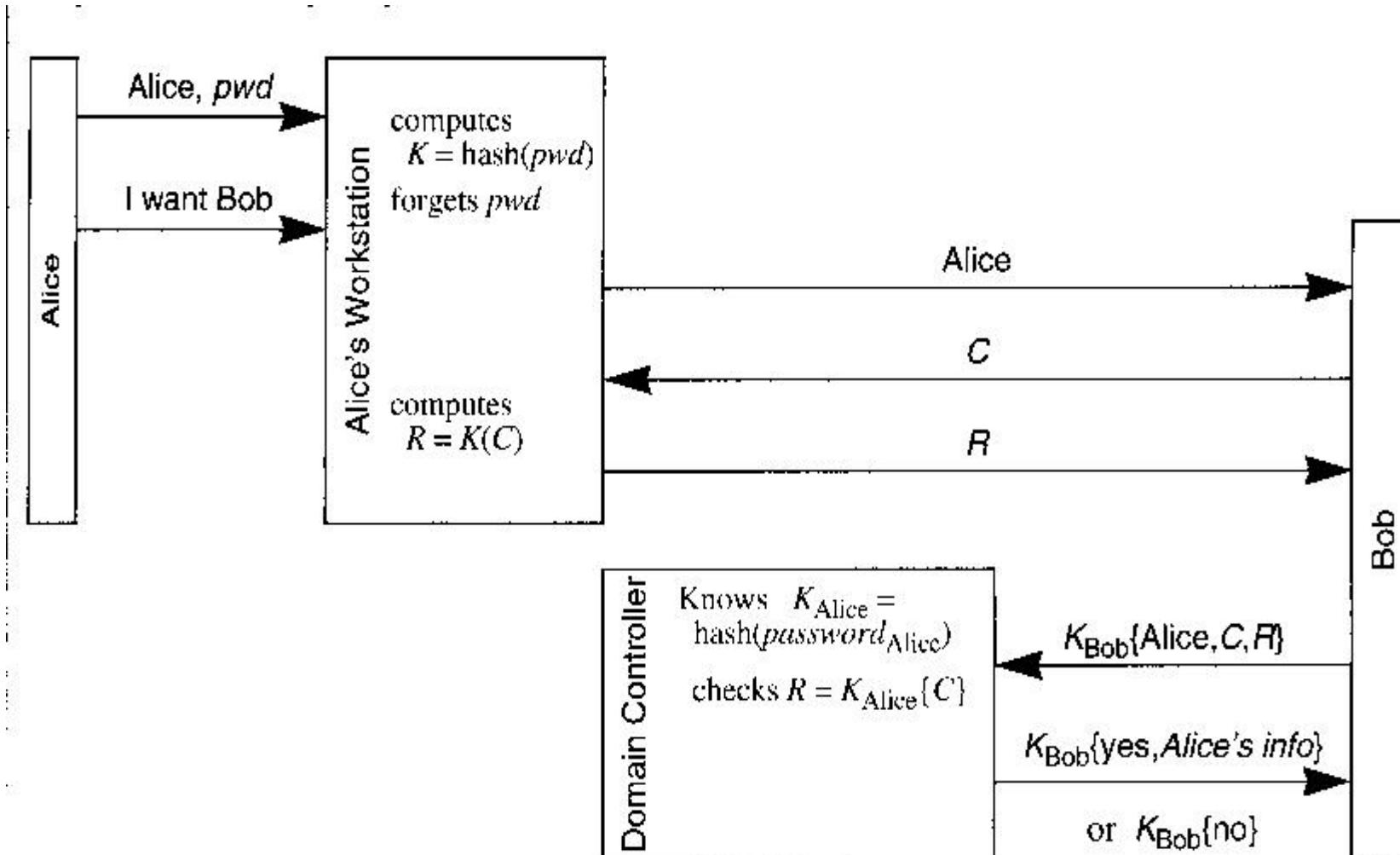
- NTLM (NT LAN Manager) is a suite of Microsoft security protocols used for authentication, integrity, and confidentiality in Windows environments.
- It relies on a **challenge-response mechanism** where users' passwords are hashed and exchanged in a secure way to authenticate clients without transmitting plaintext passwords.

# 3.1 Centralized ID Authentication: NTLM (2/6)

## Authentication Server (TTP)

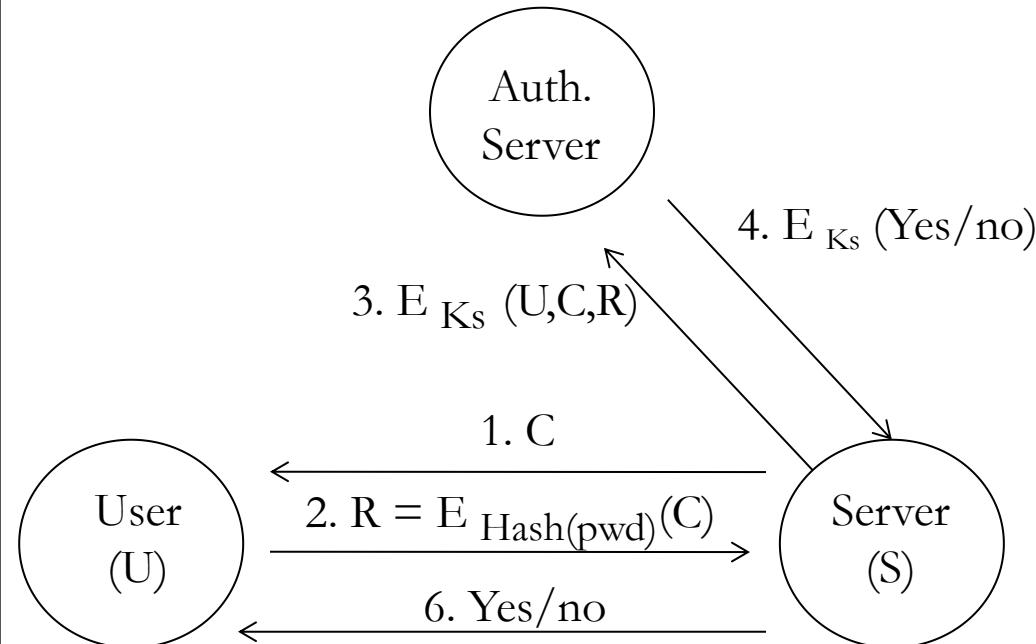


## 3.1 Centralized ID Authentication: NTLM (3/6)



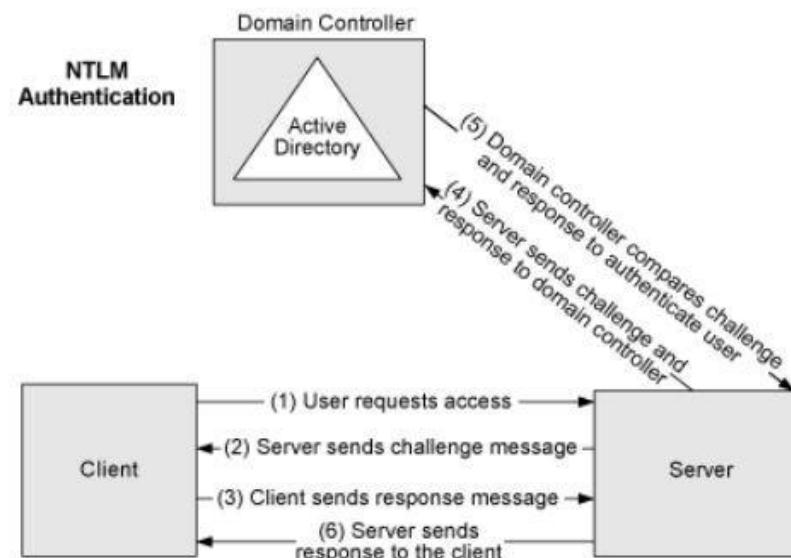
Authentication Server

## 3.1 Centralized ID Authentication: NTLM (4/6)



- Auth. Server has hashed pwds of each user, denoted by  $[U, \text{Hash}(\text{pwd})]$
- When user wants to login in as U, Server sends a challenge – **nonce C**
- User sends a response R – encrypted C with hashed pwd
- $(U, C, R)$  is forwarded to Auth. Server (encrypted with shared key between S and Auth. Server )
- Auth. Server can verify and inform results.

# 3.1 Centralized ID Authentication: NTLM (5/6)



Disadvantage:

1. Online server (common price to pay)
2. Using outdated cryptographic algorithms such as MD5 (found to be not very secure)
3. Password brute-force attacks.  $H(\text{pwd})$

Note: Despite outdated, NTLM is still widely deployed on systems. A major reason is to maintain compatibility with older systems.

# 3.1 Centralized ID Authentication: NTLM (6/6)

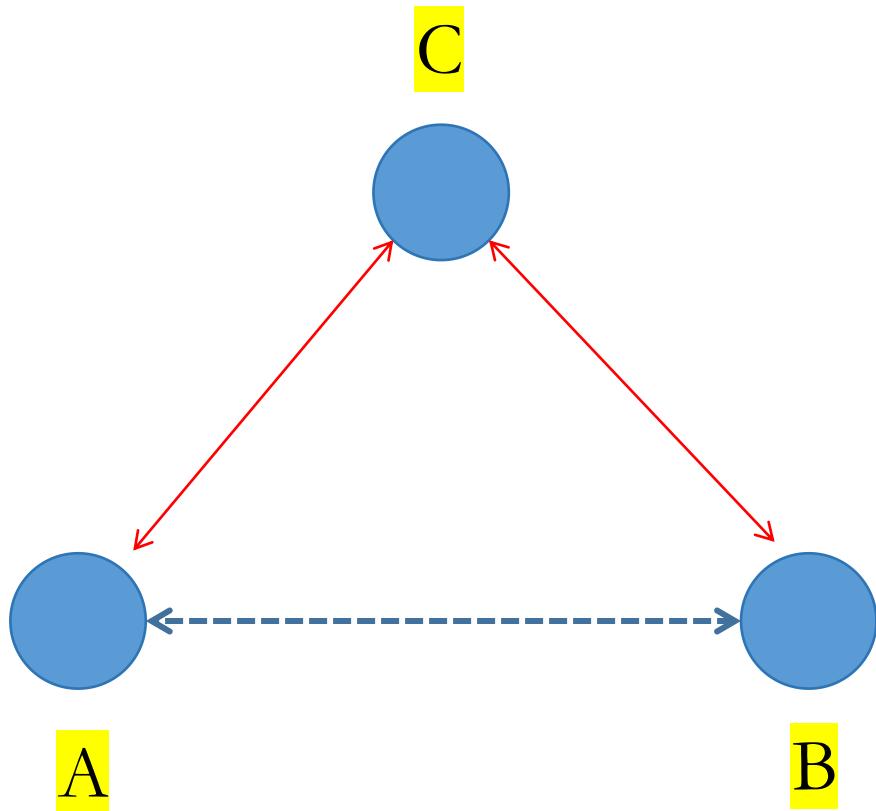
```
# Input:  
# - nt lm _hash: The target NTLM hash to break  
# - password_list: A list of candidate passwords  
# - nt lm _hash _function: Function that generates an NTLM hash  
  
# Loop through each password in the password list  
for password in password_list:  
    # Generate NTLM hash for the current password  
    candidate_hash = nt lm _hash _function(password)  
  
    # Compare the candidate hash with the target NTLM hash  
    if candidate_hash == nt lm _hash:  
        # If a match is found, print the correct password and exit  
        print("Password found: " + password)  
        break  
    else:  
        # If no match is found after all attempts, print failure message  
        print("Password not found")
```

Password brute-force attacks.

**Solution:** Add a random salt or nonce for each pwd.

## 3.2 Centralized Key Distribution

## 3.2 Centralized Key Distribution: Motivation



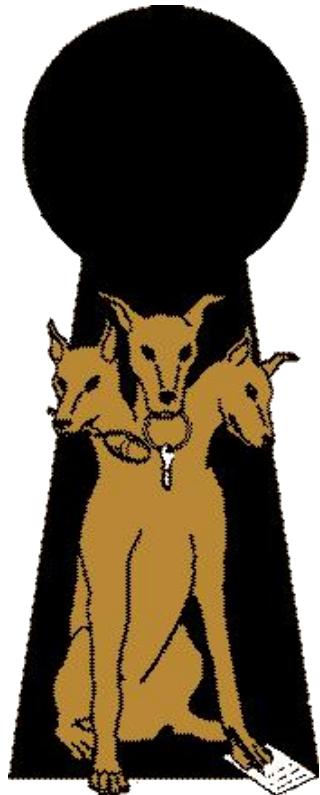
### Assumptions:

- C is the trusted third party by A and B.
- A and C have a shared secret key  $K_A$ .
- B and C have a shared secret key  $K_B$ .

### Motivation:

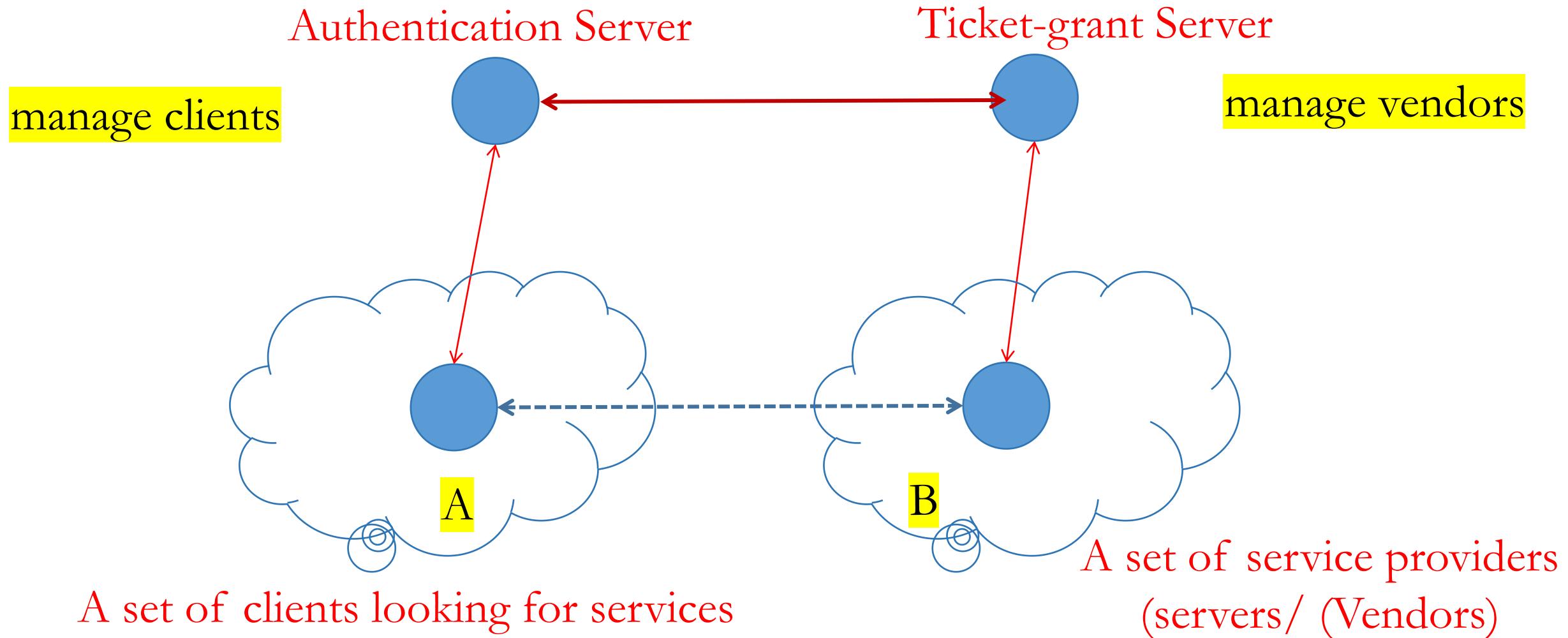
A and B want to create a shared secret key

## 3.2 Centralized Key Distribution: Kerberos (1/10)

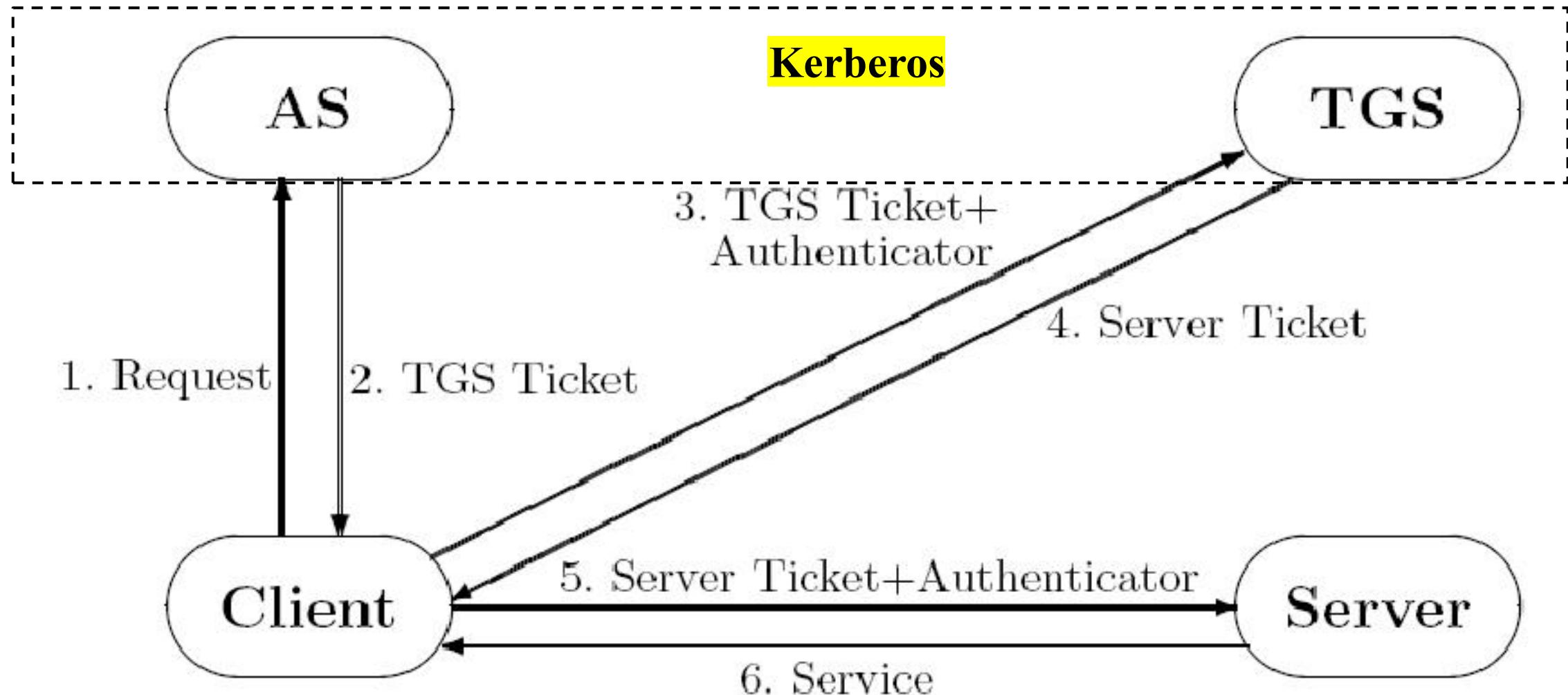


- Kerberos is a network authentication protocol developed in the late 1980s by MIT as part of Project Athena. It was designed to provide secure authentication and manage key distribution for users and services over a non-secure network, such as the internet.
- Unlike older protocols, Kerberos uses strong cryptography not only to authenticate users but also to securely distribute session keys, allowing encrypted communication between clients and services. This prevents eavesdropping, replay attacks, and ensures secure data exchange throughout the session.

## 3.2 Centralized Key Distribution: Kerberos (2/10)



## 3.2 Centralized Key Distribution: Kerberos (3/10)



## 3.2 Centralized Key Distribution: Kerberos (4/10)

1: C → AS: ID<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>1</sub>

2: AS → C: E<sub>K<sub>c</sub></sub>[K<sub>c,tgs</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>, Ticket<sub>tgs</sub>],

where Ticket<sub>tgs</sub> = E<sub>K<sub>tgs</sub></sub>[K<sub>c,tgs</sub>, ID<sub>C</sub>, AD<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>]

3: C → TGS: ID<sub>V</sub>, Ticket<sub>tgs</sub>, Authenticator<sub>C</sub>, where Authenticator<sub>C</sub> = E<sub>K<sub>c,tgs</sub></sub>[ID<sub>C</sub>, AD<sub>C</sub>, TS<sub>3</sub>]

4: TGS → C: E<sub>K<sub>c,tgs</sub></sub>[K<sub>C,V</sub>, ID<sub>V</sub>, TS<sub>4</sub>, Lifetime<sub>4</sub>, Ticket<sub>V</sub>],

where Ticket<sub>V</sub> = E<sub>K<sub>V</sub></sub>[K<sub>c,v</sub>, ID<sub>C</sub>, AD<sub>C</sub>, ID<sub>V</sub>, TS<sub>4</sub>, Lifetime<sub>4</sub>]

5: C → V: Ticket<sub>V</sub>, Authenticator<sub>C</sub>, where Authenticator<sub>C</sub> = E<sub>K<sub>c,v</sub></sub>[ID<sub>C</sub>, AD<sub>C</sub>, TS<sub>5</sub>]

6: V → C: E<sub>K<sub>c,v</sub></sub>[TS<sub>5</sub> + 1]

## 3.2 Centralized Key Distribution: Kerberos (5/10)

1: C → AS: ID<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>1</sub>

2: AS → C:

Once the user is authenticated to the Client (C), the Client sends the authentication server a request on the behalf of the user:

3: C → TGS:

- ID<sub>C</sub> - to inform AS of the user
- ID<sub>tgs</sub> - to inform AS of the Ticket Granting Service required.  
(There may be multiple TGS's.)
- TS<sub>1</sub> is a time-stamp for security against replay attack

5: C → V:

6: V → C:

## 3.2 Centralized Key Distribution: Kerberos (6/10)

1: C → AS: ID<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>1</sub>

2: AS → C: E<sub>K<sub>c</sub></sub>[K<sub>c,tgs</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>, Ticket<sub>tgs</sub>],

where Ticket<sub>tgs</sub> = E<sub>K<sub>tgs</sub></sub>[K<sub>c,tgs</sub>, ID<sub>C</sub>, AD<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>]

- K<sub>c</sub> is the shared secret key between C and AS

3: C → TGS: • K<sub>c,tgs</sub> is the key to be used by C and TGS

4: TGS → C: • ID<sub>tgs</sub> is to confirm the TGS that C wants to communicate.

- The gap between TS<sub>1</sub> and TS<sub>2</sub> tells C it is valid or not (replay)

- TS<sub>2</sub> and Lifetime<sub>2</sub> indicate the time validity of this ticket

- The Ticket<sub>tgs</sub> **cannot be opened by C**

5: C → V:

6: V → C:

## 3.2 Centralized Key Distribution: Kerberos (7/10)

1: C → AS: ID<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>1</sub>

2: AS → C: E<sub>K<sub>c</sub></sub>[K<sub>c,tgs</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>, Ticket<sub>tgs</sub>],

where Ticket<sub>tgs</sub> = E<sub>K<sub>tgs</sub></sub>[K<sub>c,tgs</sub>, ID<sub>C</sub>, AD<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>]

3: C → TGS: ID<sub>V</sub>, Ticket<sub>tgs</sub>, Authenticator<sub>C</sub>, where Authenticator<sub>C</sub> = E<sub>K<sub>c,tgs</sub></sub>[ID<sub>C</sub>, AD<sub>C</sub>, TS<sub>3</sub>]

4: TGS → C:

- The Ticket<sub>tgs</sub> is sent to TGS
- The Authenticator<sub>C</sub> protected with the shared key to tell TGS who the client is, the address, and the time-stamp of this request.
- The TGS can use the shared key with AS (K<sub>tgs</sub>) to open the ticket to see the request information and the validity of this ticket.

## 3.2 Centralized Key Distribution: Kerberos (8/10)

1: C → AS: ID<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>1</sub>

2: AS → C: E<sub>K<sub>c</sub></sub>[K<sub>c,tgs</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>, Ticket<sub>tgs</sub>],

where Ticket<sub>tgs</sub> = E<sub>K<sub>tgs</sub></sub>[K<sub>c,tgs</sub>, ID<sub>C</sub>, AD<sub>C</sub>, ID<sub>tgs</sub>, TS<sub>2</sub>, Lifetime<sub>2</sub>]

3: C → TGS: ID<sub>V</sub>, Ticket<sub>tgs</sub>, Authenticator<sub>C</sub>, where Authenticator<sub>C</sub> = E<sub>K<sub>c,tgs</sub></sub>[ID<sub>C</sub>, AD<sub>C</sub>, TS<sub>3</sub>]

4: TGS → C: E<sub>K<sub>c,tgs</sub></sub>[K<sub>C,V</sub>, ID<sub>V</sub>, TS<sub>4</sub>, Lifetime<sub>4</sub>, Ticket<sub>V</sub>],

where Ticket<sub>V</sub> = E<sub>K<sub>V</sub></sub>[K<sub>c,v</sub>, ID<sub>C</sub>, AD<sub>C</sub>, ID<sub>V</sub>, TS<sub>4</sub>, Lifetime<sub>4</sub>]

- The process is very similar to step 2 ( AS---> C )

5: C → V:

6: V → C:

## 3.2 Centralized Key Distribution: Kerberos (9/10)

1: C → AS:

- The Ticket<sub>V</sub> is sent to V

2: AS → C:

- The vendor C can open the ticket with shared key between V and TGS to get all information including a shared key K<sub>c,v</sub> between C and V.

3: C → TGS:

- The authenticator information tells V who the client is and the time stamp (indicating a new quest)

4: TGS → C:

where  $\text{Ticket}_V = E_{K_V}[K_{c,v}, ID_C, AD_C, ID_V, TS_4, \text{Lifetime}_4]$

5: C → V:  $\text{Ticket}_V, \text{Authenticator}_C$ , where  $\text{Authenticator}_C = E_{K_{c,v}}[ID_C, AD_C, TS_5]$

6: V → C:  $E_{K_{c,v}}[TS_5 + 1]$

## 3.2 Centralized Key Distribution: Kerberos (10/10)

1: C → AS:

- The server acknowledges the message from the client.

2: AS → C:

- $TS_5 + 1$  indicates that the server is able to decrypt and get the shared key from TGS. Also, the response is based on the client's time stamp.

3: C → TGS:

4: TGS → C:

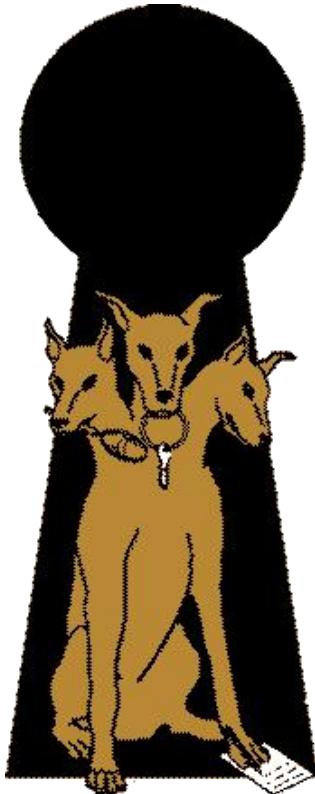
Eventually, they have a shared secret key  $K_{c,v}$  between C and V.

5: C → V: Ticket<sub>V</sub>, Authen

, where  $\text{Authenticator}_C = E_{K_{c,v}}[ID_C, AD_C, TS_5]$

6: V → C:  $E_{K_{c,v}}[TS_5 + 1]$

## 3.2 Centralized Key Distribution: Disadvantage

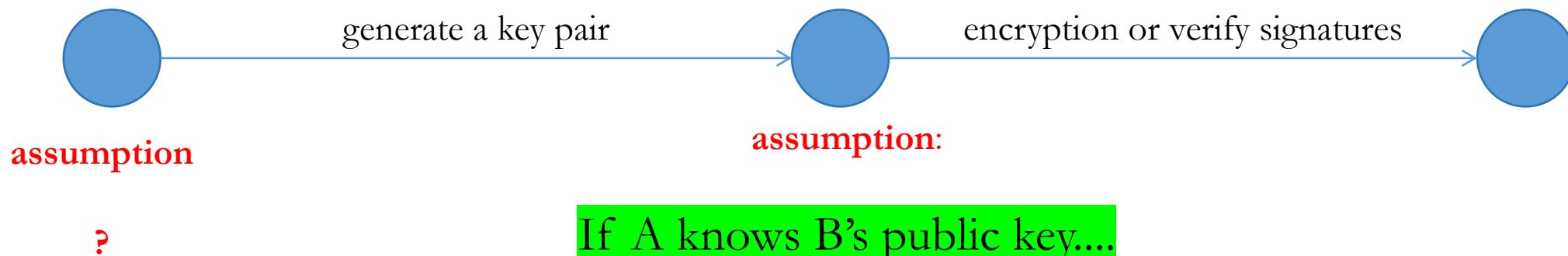


- **Single Point of Failure:** The AS+TGS is critical to the operation of Kerberos. If they become unavailable or is compromised, users cannot authenticate or access network services. This creates a single point of failure, making the network dependent on the availability and security of AS+TGS.
- **Time Synchronization Requirement:** Kerberos relies heavily on accurate time synchronization between the client, AS+TGS, and servers. Even small discrepancies can cause authentication failures, as Kerberos uses timestamps to prevent replay attacks. This requires maintaining precise system clocks across all devices.

Note: All protocols have so-called disadvantages.

# Public Key Infrastructure

## 3.3 Centralized pk Authentication



### 3.3 Centralized pk Authentication: Motivation

In Public-key encryption and digital signatures, we assume that an entity pre-knows the public key of another entity. A public key looks like:

```
30 82 01 0a 02 82 01 01 00 dd 9e e4 f6 88 b5 0d d7 a1 5f bc 25 6e cc 44 14 cf 34 e2 b5 73 09 1e e4 4b 12 52 38  
95 36 1a c6 66 ed f0 c8 03 c9 b3 43 45 4e 0d 6a 92 4b 1b eb 94 60 5b 11 b9 15 79 b1 a5 f6 fc 5d bf a4 30 59 84 02  
dd 3f 6d 21 6a 44 b7 18 1c 24 fc f5 02 2e 87 0c 20 3e c6 c5 b6 9f ad 16 1b 76 86 e9 73 9c 8d 31 60 3a a0 f0 2f da  
ad 8e f6 74 c9 81 d3 ea f7 5d ab 5d bb 05 63 b0 78 55 ed 72 13 a4 42 43 72 23 73 c0 de 33 9b 44 5c 89 a9 8a 90 d1  
99 be bc f7 21 21 5f fb 22 8a 5c 50 b9 69 7c dc 87 92 ed 79 56 ed 32 55 41 9f af 41 f6 da d4 70 88 e9 a3 41 1d 66  
9f a6 98 d2 7e 5b a9 52 38 1c 56 b4 cc 45 62 72 0c c7 f7 ef 2c 47 0a 3b 1a 7a ac e7 ae a9 a8 1e 98 43 b0 58 56 e9  
41 44 72 e6 da 67 1c d7 b6 f4 e6 b4 90 5c b5 0a 98 b3 23 0c e7 35 6d 10 14 73 0e 94 5d 7c 4e 0a 18 f4 05 20 67 9f
```

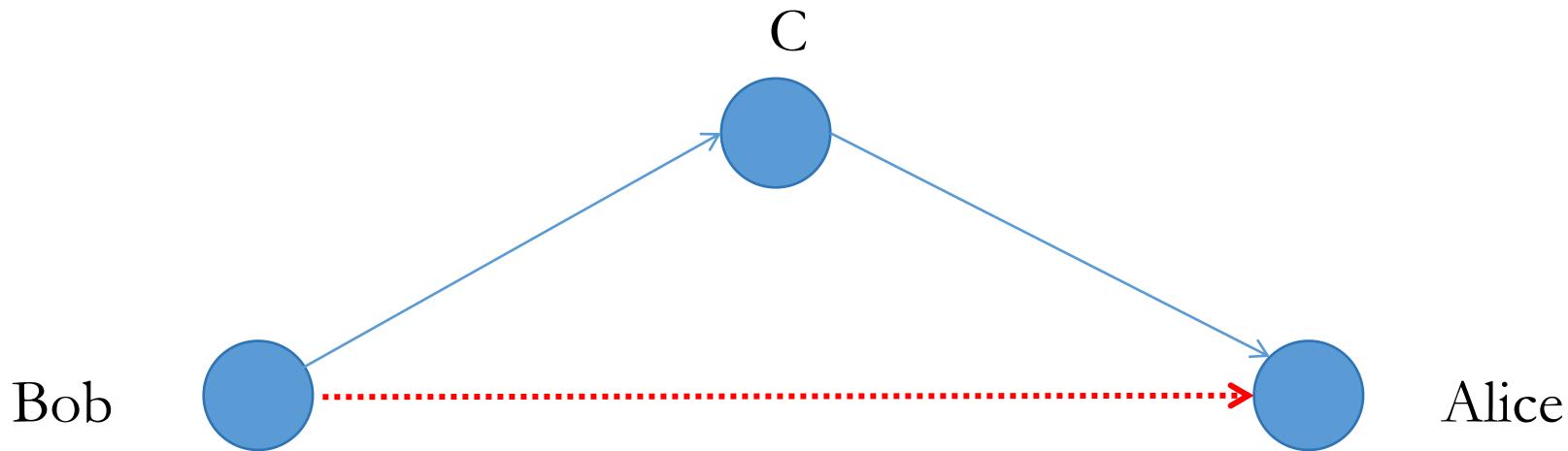
This is because **pk is computed from sk** while sk is a random string.

**Motivation:** How can users be convinced that “pk belongs to Alice”?

### 3.3 Centralized pk Authentication: PKI Overview (1/5)

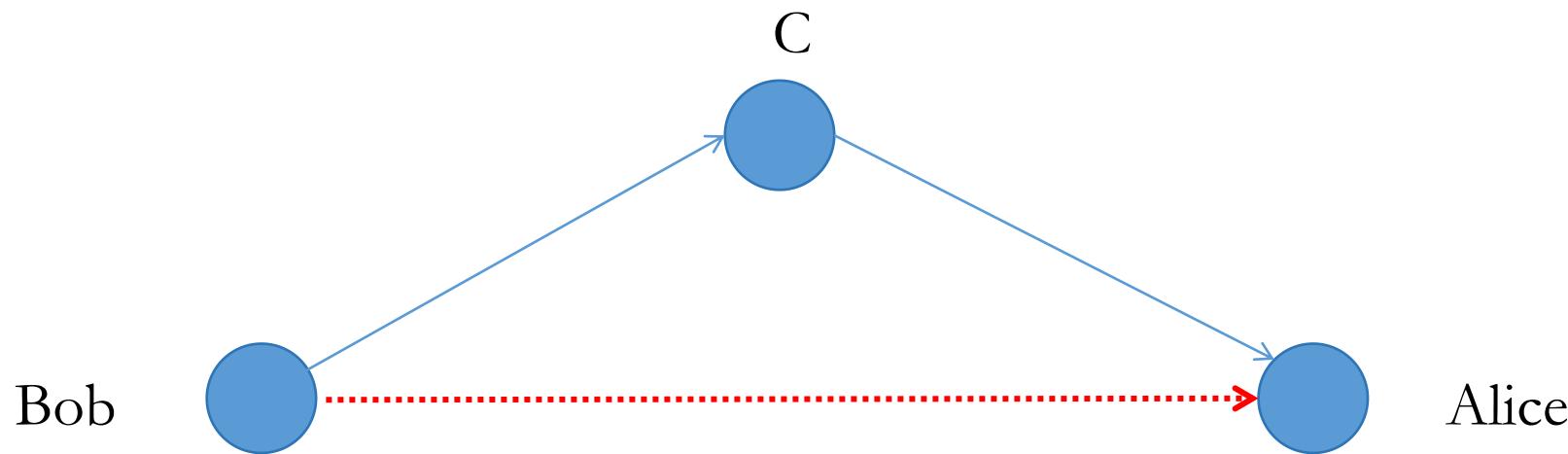
- Public-key infrastructure (PKI) is a system framework and a set of hardware, software, people, policies, and procedures needed to **create, manage, store, distribute, and revoke** digital certificates based on asymmetric cryptography.
- In particular, digital certificates are used to prove “pk belongs to Alice”
- The certificates will be issued by trusted certificate authorities (CA) and therefore PKI is one kind of centralized pk authentication.

### 3.3 Centralized pk Authentication: PKI Overview (2/5)



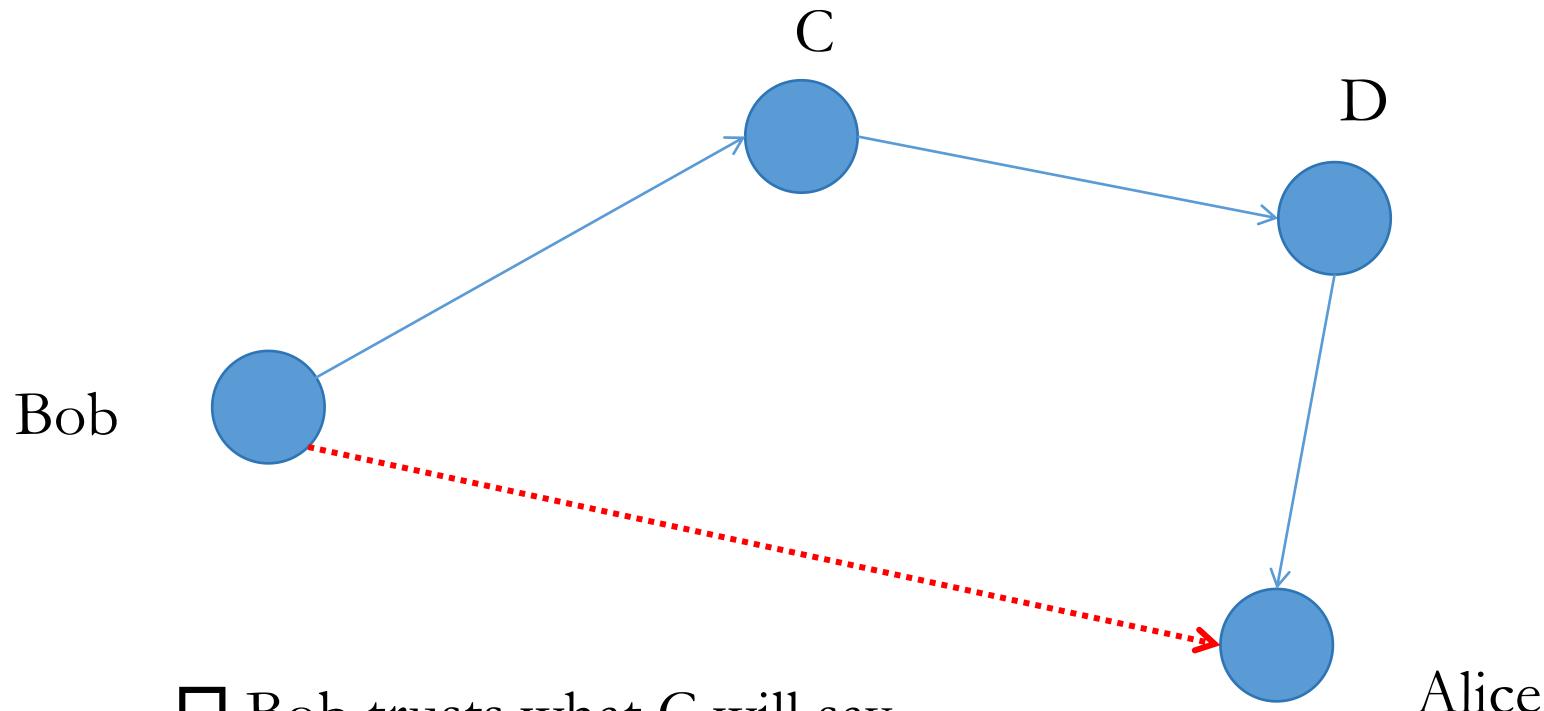
- Bob trusts what C will say.
- C says that “pk belongs to Alice”
- Then Bob believes that “pk belongs to Alice”

### 3.3 Centralized pk Authentication: PKI Overview (3/5)



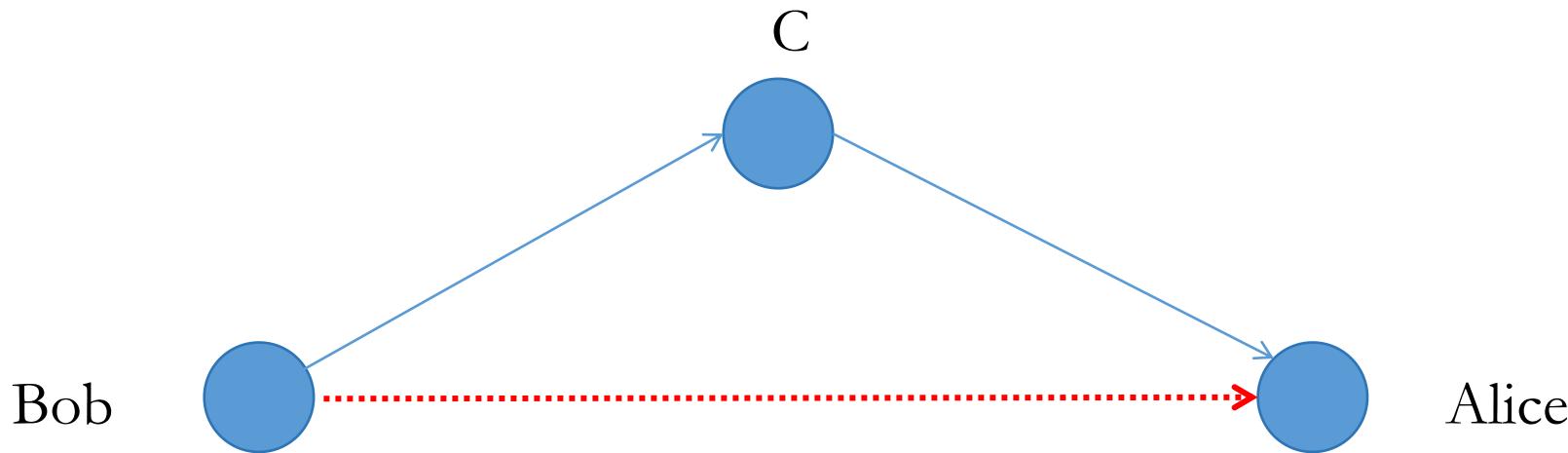
- Bob trusts what C will say and **pre-knows** that  $pk^*$  belongs to C.
- C says that  $m = "pk \text{ belongs to Alice}"$  and uses  $sk^*$  to sign on m. Then, m and the corresponding signature S is called **certificate**.
- Bob can input  $(m, S)$  and  $pk^*$  to verify that “pk belongs to Alice”

### 3.3 Centralized pk Authentication: PKI Overview (4/5)



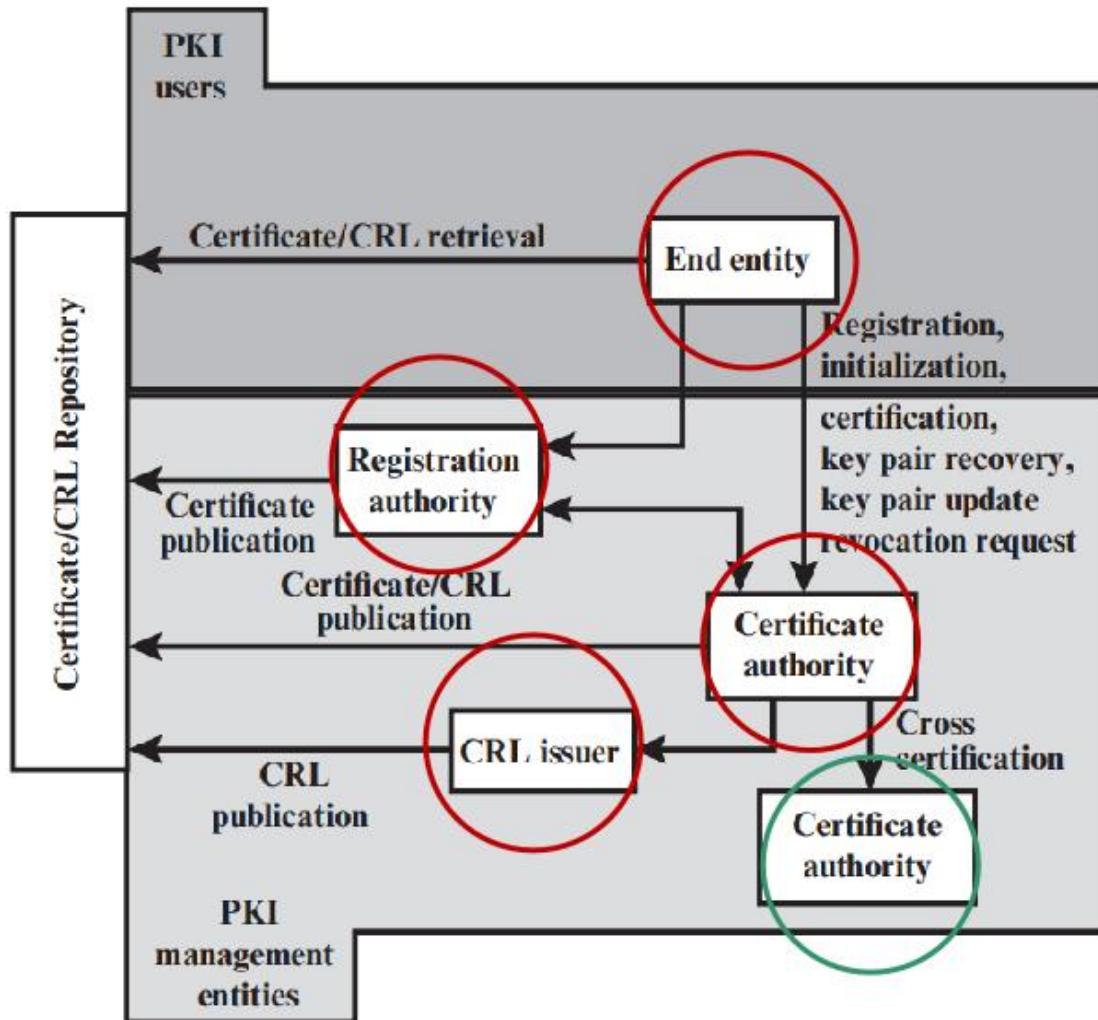
- Bob trusts what C will say.
- C trusts what D will say.
- D says that “pk belongs to Alice”
- Then Bob believes that “pk belongs to Alice”

### 3.3 Centralized pk Authentication: PKI Overview (5/5)



- **Expiracy:**  $m = \text{"pk belongs to Alice"}$   
=>  $m = \text{"pk belongs to Alice From Time X to Time Y"}$
  
- **Revocation:** Alice found that her sk was stolen before the **expiracy time Y** and need to revoke it in use.

### 3.3 Centralized pk Authentication: PKI (1/5)

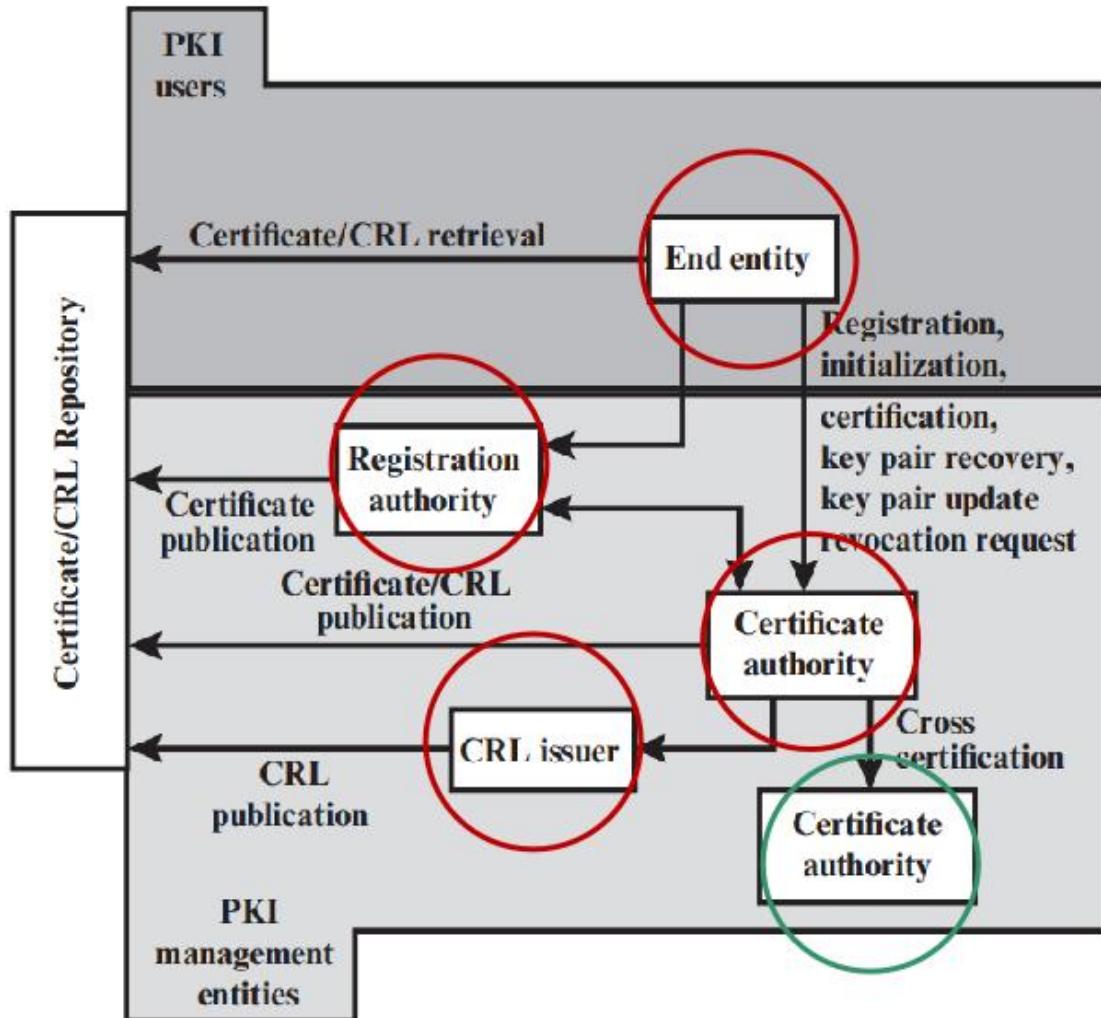


#### Certification authority (CA):

The issuer of

- certificates
- certificate revocation lists (CRLs).

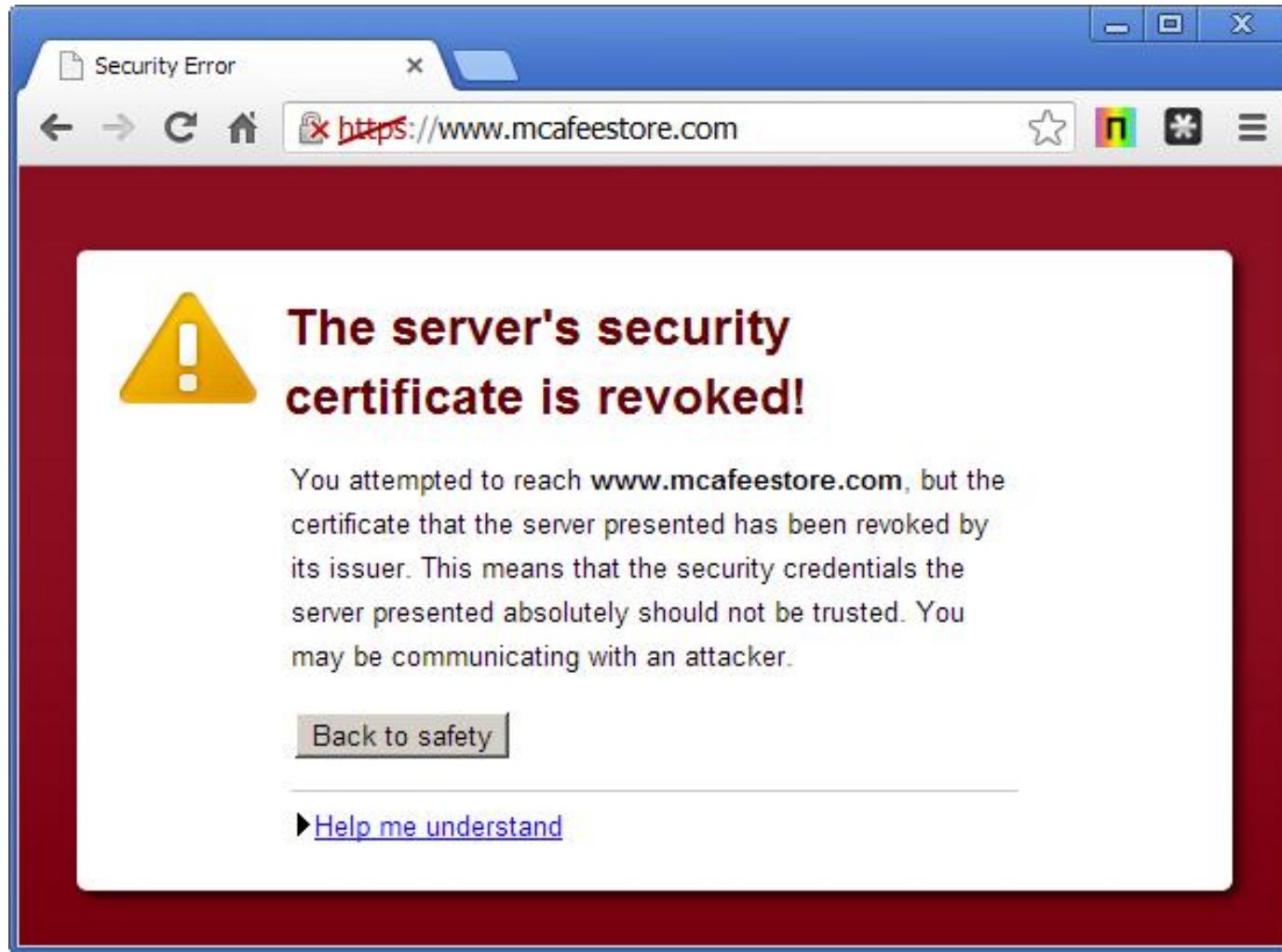
### 3.3 Centralized pk Authentication: PKI (2/5)



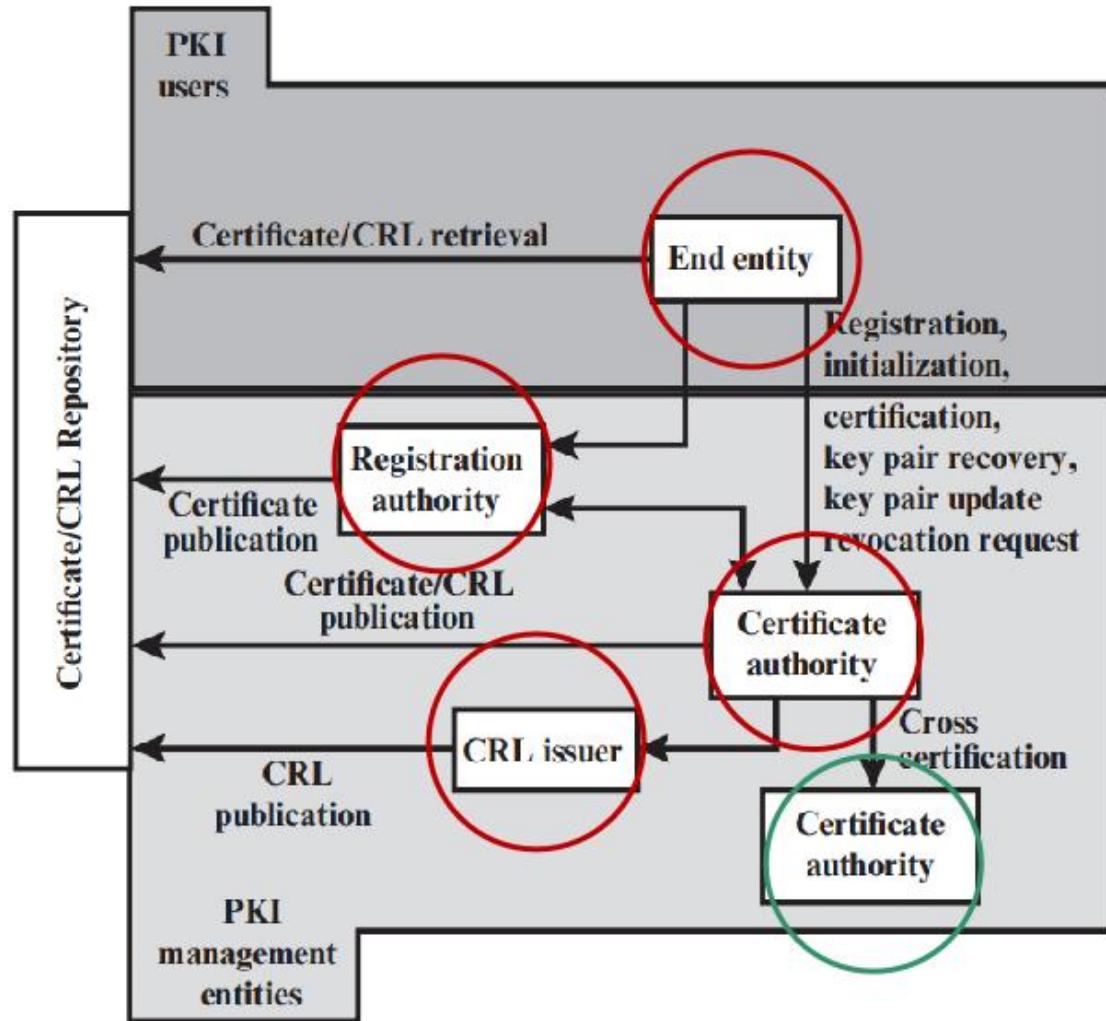
#### CRL:

- A way of telling users about revoked certificates.
- Users must ensure that they have the latest CRL.
- A CRL is a bit like the list of bad credit card numbers which used to be kept next to the tellers in supermarkets.

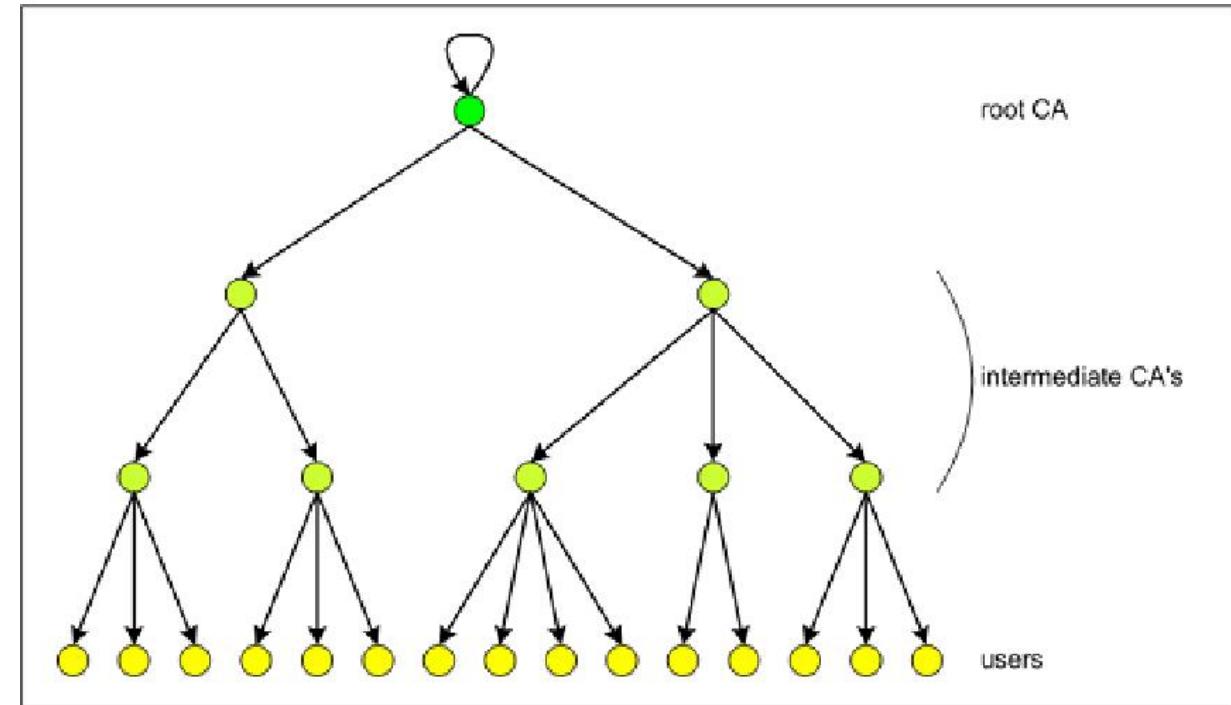
### 3.3 Centralized pk Authentication: PKI (3/5)



### 3.3 Centralized pk Authentication: PKI (4/5)

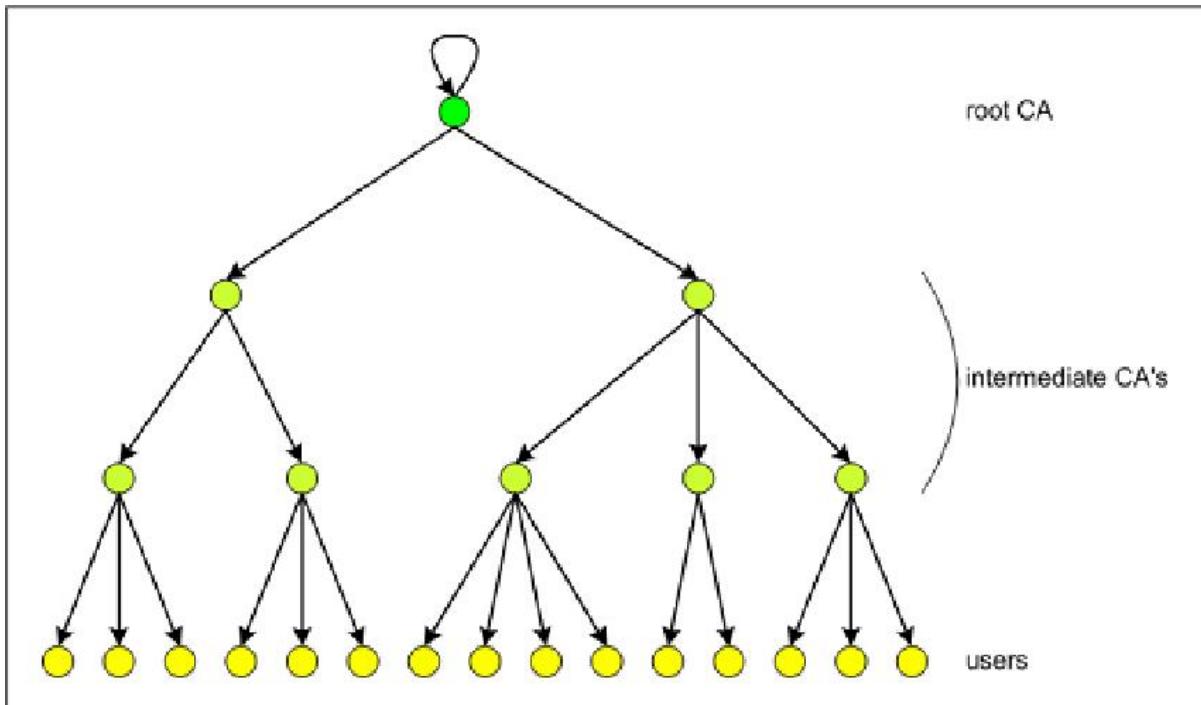


Chain of certificates:



### 3.3 Centralized pk Authentication: PKI (5/5)

Chain of certificates:



Certificate Viewer: www.uow.edu.au

General Details

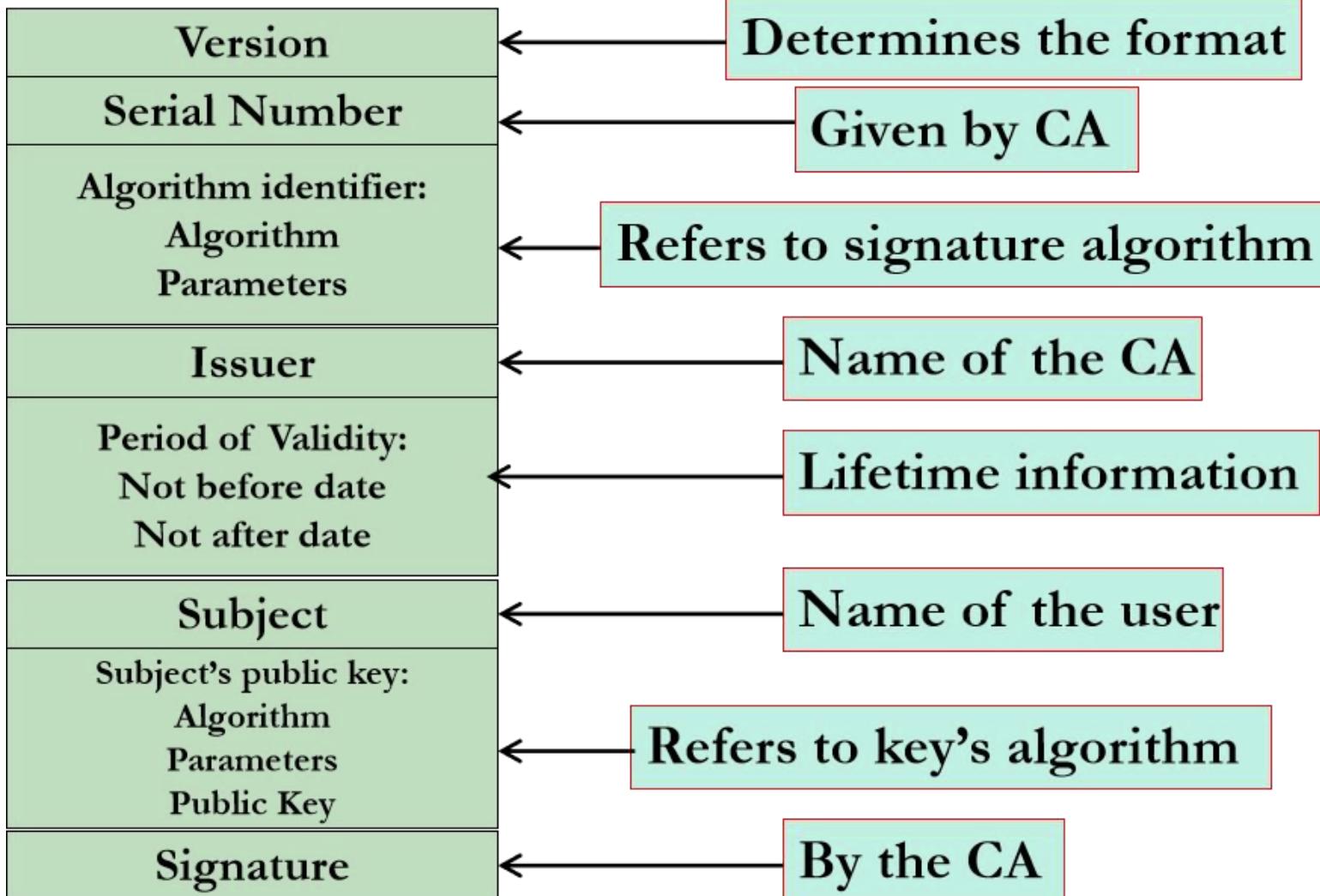
**Certificate Hierarchy**

- ▼ DigiCert Global Root G2
  - ▼ DigiCert Global G2 TLS RSA SHA256 2020 CA1
    - www.uow.edu.au

**Certificate Fields**

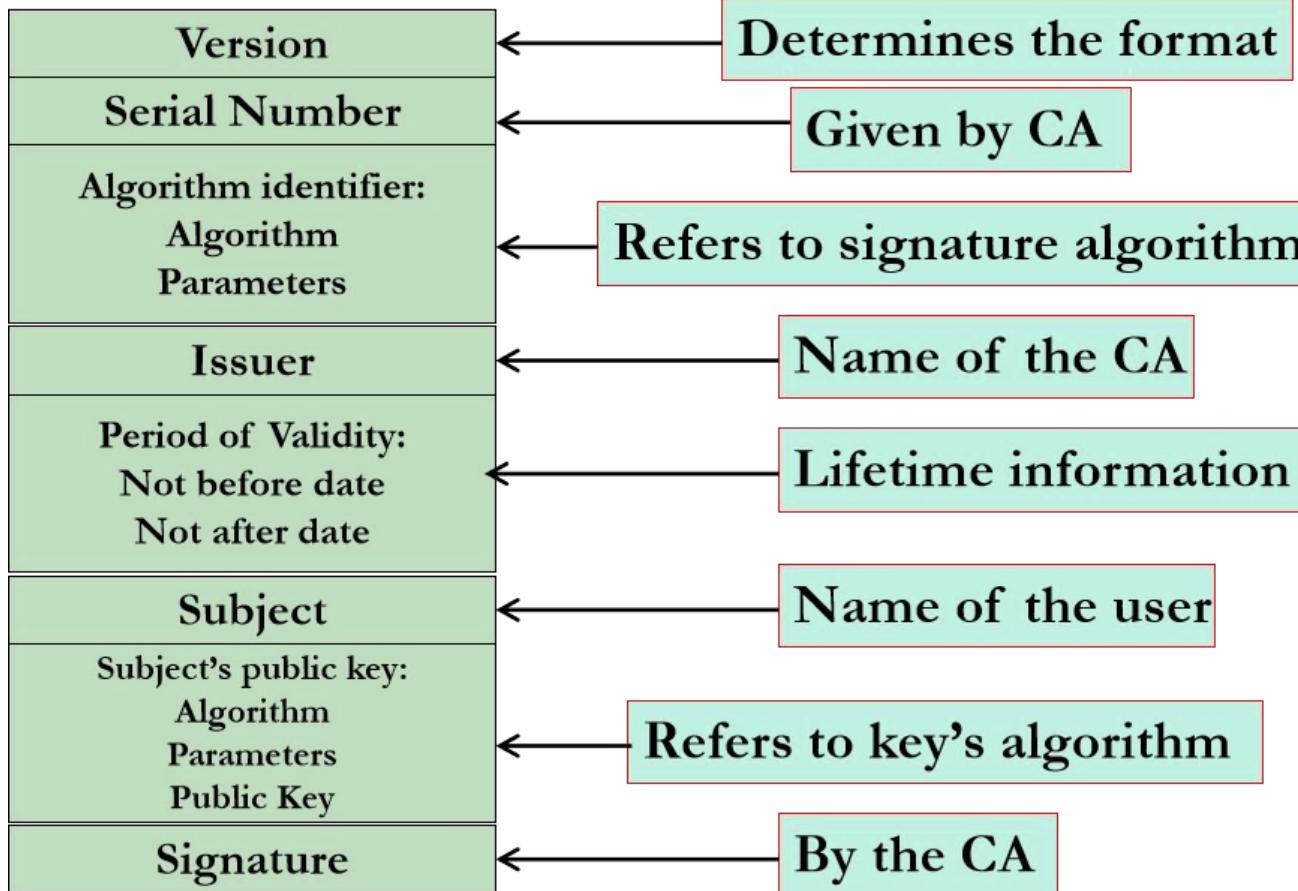
- ▼ www.uow.edu.au
  - ▼ Certificate
    - Version
    - Serial Number
    - Certificate Signature Algorithm
    - Issuer**
    - ▼ Validity
      - Not Before

### 3.3 Centralized pk Authentication: X.509 (1/3)



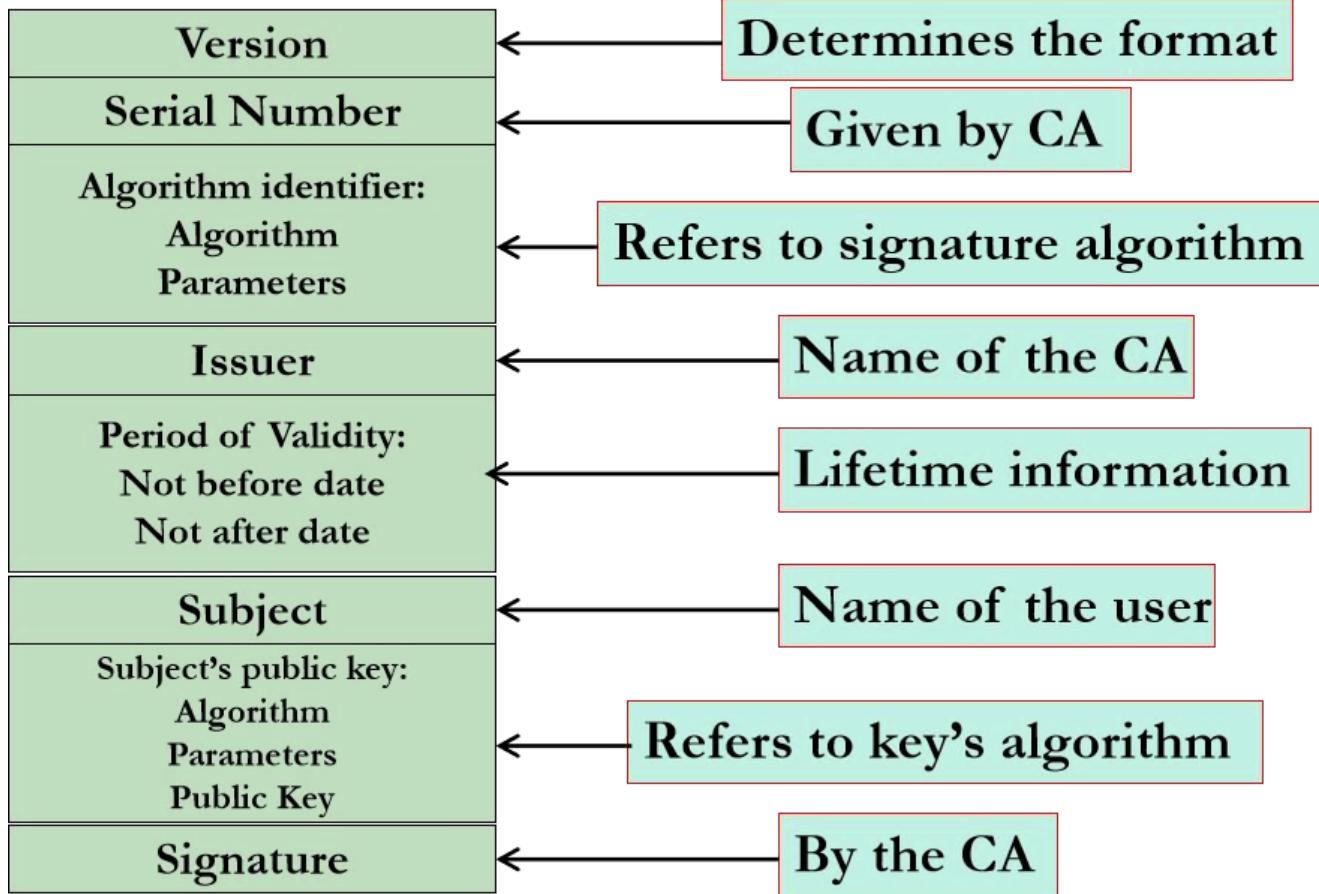
- X.509 is a widely used standard for defining the format of public key certificates.
- These certificates are used in Public Key Infrastructure (PKI) to verify the identity of entities (such as users, devices, or servers) by associating public keys with their corresponding identities.

### 3.3 Centralized pk Authentication: X.509 (2/3)



- **Serial number**: An integer value unique within the issuing CA that is associated with this certificate.
- **Signature algorithm identifier**: The algorithm used to sign the certificate together with any associated parameters.
- **Issuer**: X.500 name of the CA that created and signed this certificate.
- **Period of validity**: Consists of two dates: the first and last on which the certificate is valid.

### 3.3 Centralized pk Authentication: X.509 (3/3)



- **Subject**: The name of the user to whom this certificate refers. That is, this certificate certifies the public key of the subject (user) who holds the corresponding private key.
- **Subject's public key**: The public key of the subject, plus an identifier of the algorithm for which this key is to be used, together with any associated parameters.
- **Signature**: Covers all of the other fields of the certificate.

### 3.3 Centralized pk Authentication: Summary

- **Complexity:** Setting up and managing a PKI system can be quite complex. It involves generating, distributing, and managing certificates, which requires specialized knowledge.
- **Scalability Issues:** As the number of certificates grows, managing them becomes increasingly difficult. This includes handling certificate renewals, revocations, and ensuring all systems trust the correct certificates.
- **Certificate Revocation:** Revoking certificates can be challenging. If a certificate is compromised, it needs to be revoked promptly, and all systems must be updated to recognize the revocation.
- **Trust Chain Vulnerabilities:** The trust model relies on a chain of trust from the root certificate down to the end-entity certificates. If any certificate in this chain is compromised, it can affect the security of all certificates below it.
- **X.509 certificates** have many optional fields and extensions, making them complex to understand and implement correctly

## **3.4 Email Security Protocols**

## 3.4 Email Security Protocols: Overview (1/2)

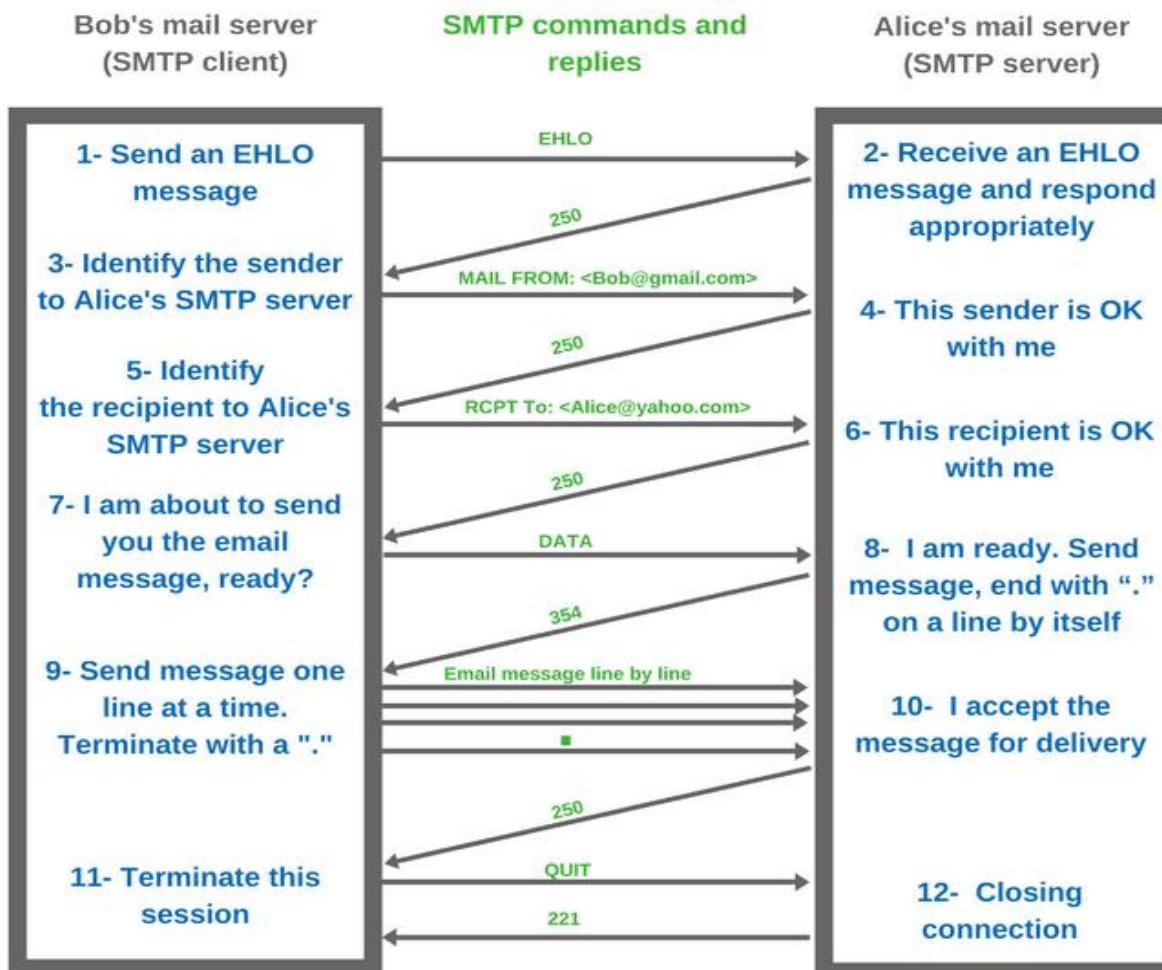
- Email is an application for exchanging messages.
- Email protocol is a set of rules defined to ensure that emails can be exchanged between various servers and email clients in a standard manner.
- This ensures that the email is universal and works for all users.



## 3.4 Email Security Protocols: Overview (2/2)

- Simple Mail Transfer Protocol (SMTP): Primarily used to send and forward emails between mail servers. It's responsible for the delivery of emails from a sender to a recipient's mail server.
- Multipurpose Internet Mail Extensions (MIME): Extends SMTP by allowing email messages to include multimedia content like images, audio, and attachments. It's not a protocol by itself but an extension that formats non-text content in email.
- Post Office Protocol (POP): Specifically designed to retrieve emails from a mail server to a local client. The most common version is POP3. Once the email is downloaded, it's typically deleted from the server, though some configurations allow messages to remain on the server.
- Internet Message Access Protocol (IMAP): Allows users to access and manage emails directly on the mail server without needing to download them. IMAP is more versatile than POP since emails are stored on the server and synchronized across multiple devices.

# 3.4 Email Security Protocols: SMTP (1/2)



```
Client: HELO example.com
Server: 250 Hello example.com, pleased to meet you

Client: MAIL FROM:<Bob@gmail.com>
Server: 250 OK

Client: RCPT TO:<Alice@yahoo.com>
Server: 250 OK

Client: DATA
Server: 354 Start mail input; end with <CRLF>.<CRLF>

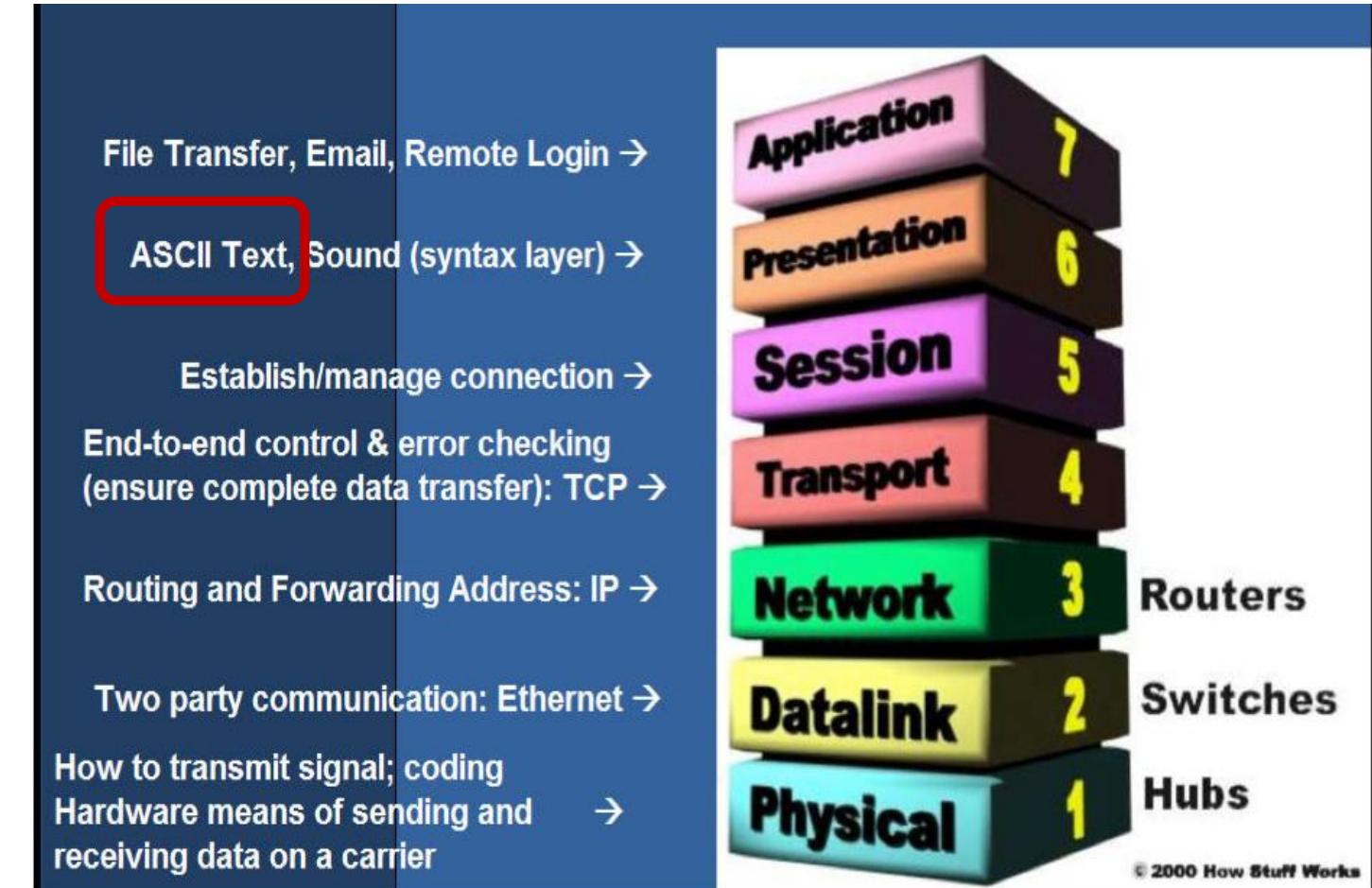
Client: [Message Body]
Client: .
Server: 250 OK, message accepted for delivery

Client: QUIT
Server: 221 Bye
```

# 3.4 Email Security Protocols: SMTP (2/2)

SMTP emails contain only ASCII characters.

0	0x41	A	16	0x51	Q	32	0x67	g	48	0x77	w
1	0x42	B	17	0x52	R	33	0x68	h	49	0x78	x
2	0x43	C	18	0x53	S	34	0x69	i	50	0x79	y
3	0x44	D	19	0x54	T	35	0x6a	j	51	0x7a	z
4	0x45	E	20	0x55	U	36	0x6b	k	52	0x30	0
5	0x46	F	21	0x56	V	37	0x6c	l	53	0x31	1
6	0x47	G	22	0x57	W	38	0x6d	m	54	0x32	2
7	0x48	H	23	0x58	X	39	0x6e	n	55	0x33	3
8	0x49	I	24	0x59	Y	40	0x6f	o	56	0x34	4
9	0x4a	J	25	0x5a	Z	41	0x70	p	57	0x35	5
10	0x4b	K	26	0x61	a	42	0x71	q	58	0x36	6
11	0x4c	L	27	0x62	b	43	0x72	r	59	0x37	7
12	0x4d	M	28	0x63	c	44	0x73	s	60	0x38	8
13	0x4e	N	29	0x64	d	45	0x74	t	61	0x39	9
14	0x4f	O	30	0x65	e	46	0x75	u	62	0x2b	+
15	0x50	P	31	0x66	f	47	0x76	v	63	0x2f	/



## 3.4 Email Security Protocols: MIME (1/3)

```
client: HELO example.com
Server: 250 Hello example.com, pleased to meet you

Client: MAIL FROM:<Bob@gmail.com>
Server: 250 OK

Client: RCPT TO:<Alice@yahoo.com>
Server: 250 OK

Client: DATA
Server: 354 Start mail input; end with <CRLF>.<CRLF>

Client: [Message Body]
Client: .
Server: 250 OK, message accepted for delivery

Client: QUIT
Server: 221 Bye
```

### Limitation of SMTP:

- **Plain Text Only:** SMTP was originally designed to handle only ASCII text, which means it couldn't support binary data (like images, videos, or attachments) or non-ASCII characters (such as special characters in different languages).
- **No Support for Attachments:** SMTP couldn't natively handle file attachments like documents, images, or other media.

## 3.4 Email Security Protocols: MIME (2/3)

```
Client: HELO example.com
Server: 250 Hello example.com, pleased to meet you

Client: MAIL FROM:<Bob@gmail.com>
Server: 250 OK

Client: RCPT TO:<Alice@yahoo.com>
Server: 250 OK

Client: DATA
Server: 354 Start mail input; end with <CRLF>.<CRLF>

Client: [Message Body]
Client: .
Server: 250 OK, message accepted for delivery

Client: QUIT
Server: 221 Bye
```

MIME allows the transfer of complex or multimedia messages by using encoding and formatting techniques, which SMTP alone cannot handle.

- **Encoding:** MIME encodes non-ASCII content (like binary files, images, audio, etc.) into 7-bit ASCII text, which SMTP can transmit. For example, binary files are often encoded using Base64 to fit SMTP's requirements.
- **Formatting:** MIME formats messages into multiple parts using boundary markers. This allows the email to include different types of content (text, images, attachments, etc.) in one message. Each part of the message can be assigned a Content-Type (e.g., text/plain, image/jpeg) to specify what kind of data it contains.

## 3.4 Email Security Protocols: MIME (3/3)

```
client: HELO example.com
Server: 250 Hello example.com, pleased to meet you

Client: MAIL FROM:<Bob@gmail.com>
Server: 250 OK

Client: RCPT TO:<Alice@yahoo.com>
Server: 250 OK

Client: DATA
Server: 354 Start mail input; end with <CRLF>.<CRLF>

Client: [Message Body]
Client: .
Server: 250 OK, message accepted for delivery

Client: QUIT
Server: 221 Bye
```



```
From: Bob@gmail.com
To: Alice@yahoo.com
Subject: Check out this image
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="boundary_string"

--boundary_string
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

Hi Alice,
Please find the attached image.

--boundary_string
Content-Type: image/jpeg; name="picture.jpg"
Content-Disposition: attachment; filename="picture.jpg"
Content-Transfer-Encoding: base64

[Base64 encoded image data]
```

## 3.4 Email Security Protocols: Security (1/2)

In general, a communication protocol is insecure because:

- **Lack of Confidentiality Protections:** Without proper encryption or confidentiality mechanisms, an attacker can intercept and read the contents of the communication.
- **Lack of Validation/Verification:** If the protocol doesn't verify the integrity or authenticity of the transmitted data, an attacker can alter or tamper with the contents. This could lead to spoofing, message manipulation, or injecting malicious data into the communication.

## 3.4 Email Security Protocols: Security (2/2)

```
client: HELO example.com
Server: 250 Hello example.com, pleased to meet you

Client: MAIL FROM:<Bob@gmail.com>
Server: 250 OK

client: RCPT TO:<Alice@yahoo.com>
Server: 250 OK

Client: DATA
Server: 354 Start mail input; end with <CRLF>.<CRLF>

client: [Message Body]
Client: .
Server: 250 OK, message accepted for delivery

Client: QUIT
Server: 221 Bye
```

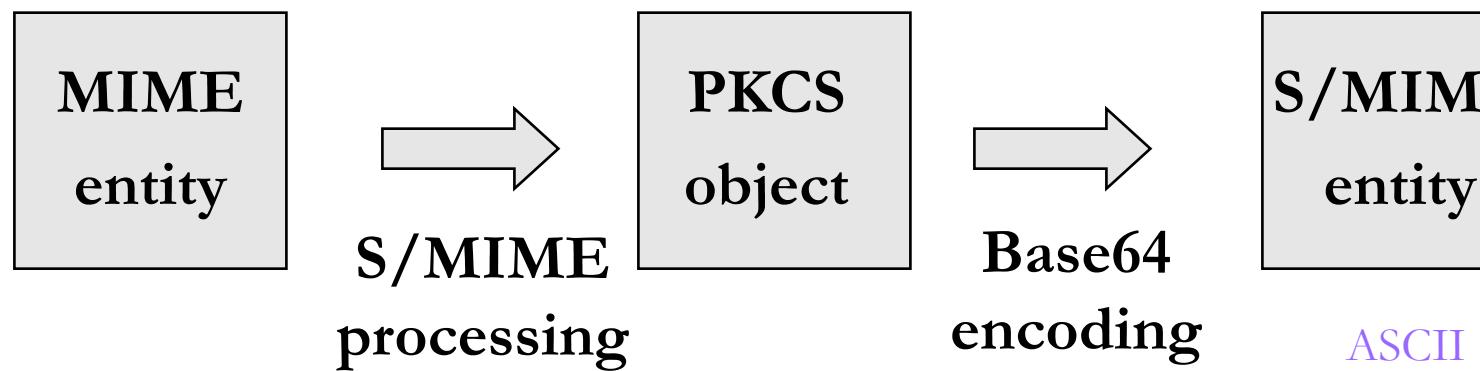
- **Lack of Confidentiality:** Emails are transmitted in plain text over open networks, making them vulnerable to interception.
- **Lack of Integrity:** Both the header and content of the email can be modified during transmission, as there are no built-in mechanisms to ensure data integrity.
- **Lack of Authentication:** The identity of the sender can be easily forged, allowing attackers to impersonate legitimate senders.

## 3.4 Email Security Protocols: S/MIME (1/5)

S/MIME provides the following functions to secure email:

- **Enveloped Data:** encrypted-only.
  - **Signed Data:** signed-only.
  - **Signed and Enveloped:** nesting of signed and encrypted entities.
- 
- Digital signatures: DSS & RSA
  - Public key encryption: ElGamal & RSA
  - Symmetric-key encryption: AES, Triple-DES, and others

## 3.4 Email Security Protocols: S/MIME (2/5)



- PKCS: Public Key Cryptography Standard.
- A PKCS object includes the original content plus all information needed for the recipient to perform security processing.

```
Client: HELO example.com
Server: 250 Hello example.com, pleased to meet you

Client: MAIL FROM:<Bob@gmail.com>
Server: 250 OK

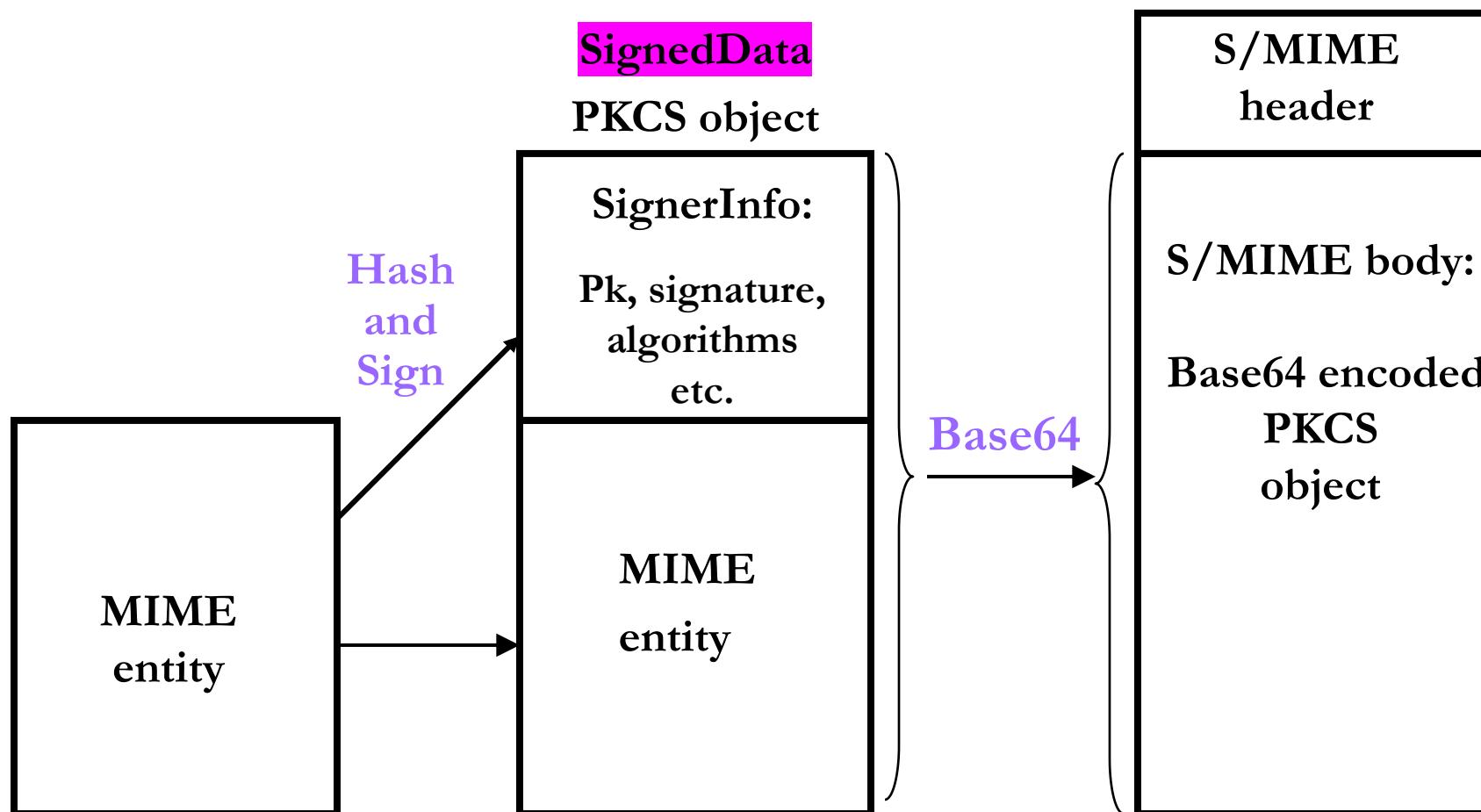
Client: RCPT TO:<Alice@yahoo.com>
Server: 250 OK

Client: DATA
Server: 354 Start mail input; end with <CRLF>.<CRLF>

Client: [Message Body]
Client: .
Server: 250 OK, message accepted for delivery

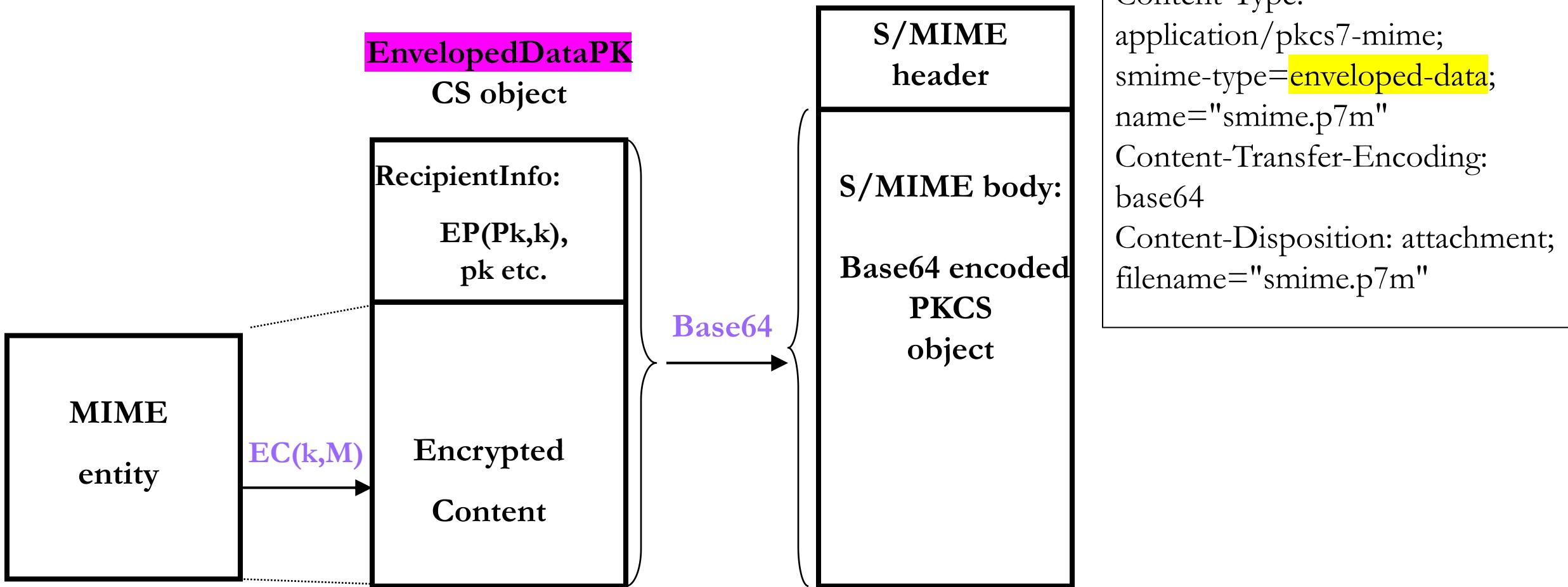
Client: QUIT
Server: 221 Bye
```

## 3.4 Email Security Protocols: S/MIME (3/5)



MIME-Version: 1.0  
Content-Type: multipart/signed;  
protocol="application/pkcs7-signature";  
micalg=sha256;  
boundary="boundary42"

## 3.4 Email Security Protocols: S/MIME (4/5)



## 3.4 Email Security Protocols: S/MIME (5/5)

### Disadvantage:

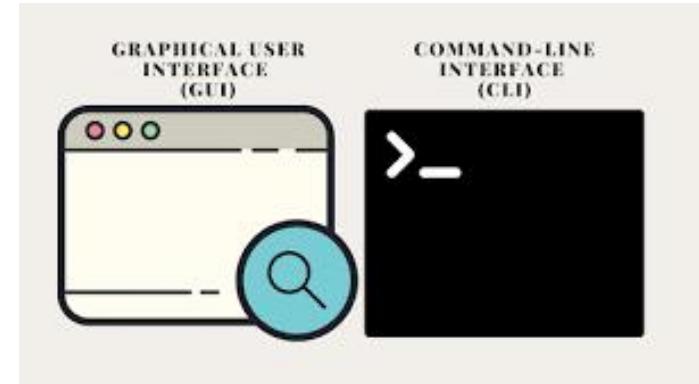
1. Complex Certificate Management: S/MIME requires digital certificates, which users need to obtain, install, and manage. This adds complexity, especially when exchanging public keys and dealing with certificate revocations.
2. Limited Interoperability: Not all email clients and platforms support S/MIME well, leading to compatibility issues. It's also harder to use on mobile devices due to certificate setup challenges.
3. Attacker Visibility: An attacker can still see the sender and recipient's email addresses during transmission via the SMTP protocol, as the envelope information is not protected by S/MIME.

## **3.5 SSH**

Secure Shell

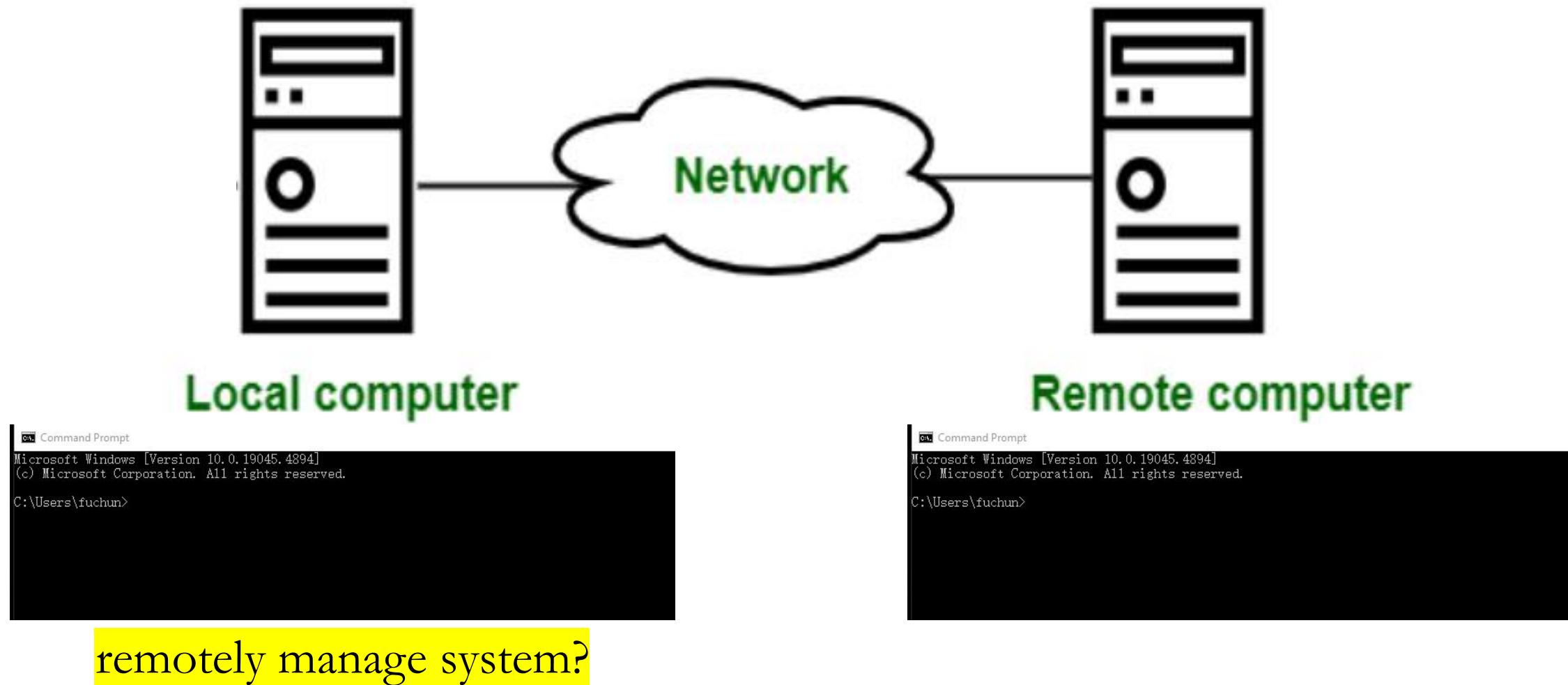
# 3.5 SSH: Bakground (1/2)

```
Command Prompt  
Microsoft Windows [Version 10.0.19045.4894]  
(c) Microsoft Corporation. All rights reserved.  
C:\Users\fuchun>
```



- A shell is a user interface that allows interaction with the operating system. It serves as a command-line interpreter to read and execute commands typed by the user.
- Command-line shells provide powerful tools for managing files, processes, and system configurations directly, allowing users to perform tasks efficiently. They are essential for remote administration of servers, especially in environments where graphical interfaces are not available or practical.

# 3.5 SSH: Bakground (2/2)



# 3.5 SSH: Overview (1/3)

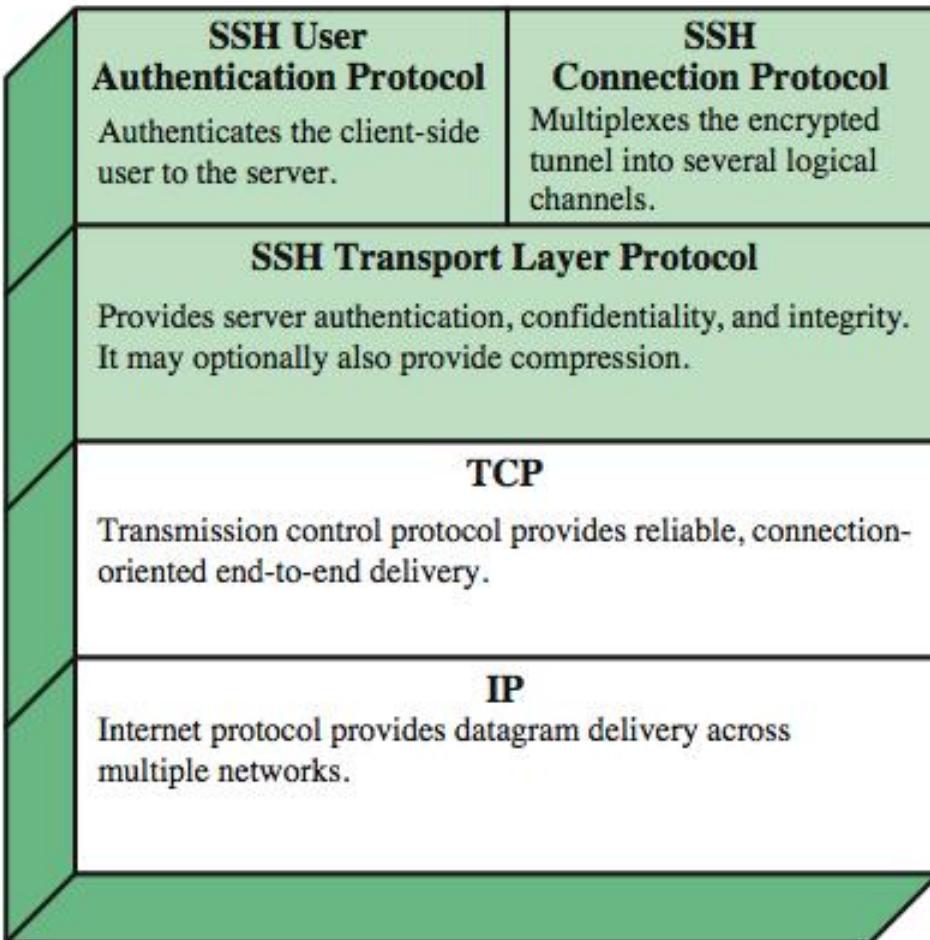
SSH (Secure Shell) is a cryptographic network protocol used to securely connect to a networked device. It ensures the confidentiality and integrity of data transmitted between two computers using cryptographic techniques.

- **Command-Line Shells:** SSH enables secure remote management of an operating system, allowing users to execute commands as if they were locally present.
- **Port Forwarding:** SSH supports encrypted connections between machines for forwarding TCP ports, providing secure data transfer even for other protocols (between local computer and remoter computer/server).

## 3.5 SSH: Overview (2/3)

- SSH (Secure Shell) and SSH-2 (Secure Shell version 2) are both protocols used to securely connect and manage remote systems, but SSH-2 is an improved version that addresses the limitations and vulnerabilities of the original SSH protocol.
- The current version in widespread use is SSH-2, which was introduced in 2006 and remains the standard for secure remote access. SSH-2 improved upon SSH-1 with better security, encryption, and flexibility, and it continues to meet the needs of modern secure communication protocols.
- So, SSH in this subject refers to SSH-2

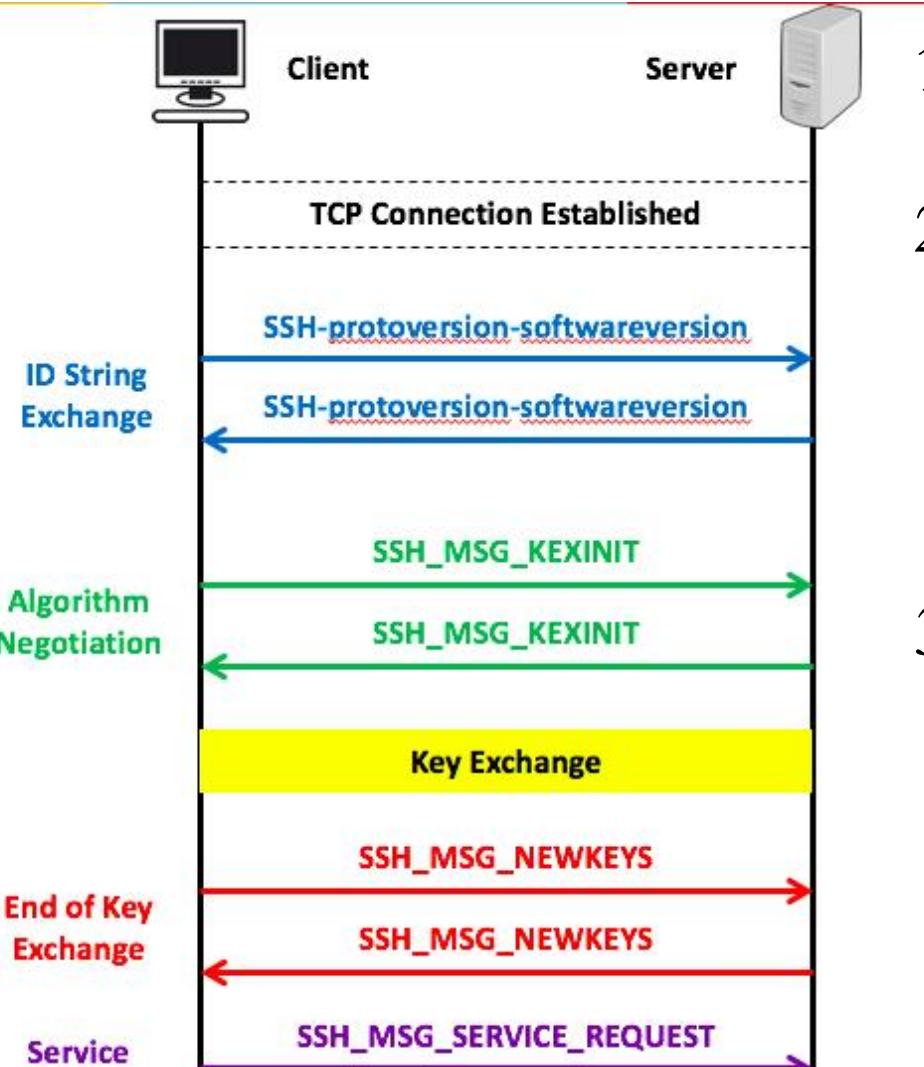
# 3.5 SSH: Overview (3/3)



**SSH adopts a three layer architecture:**

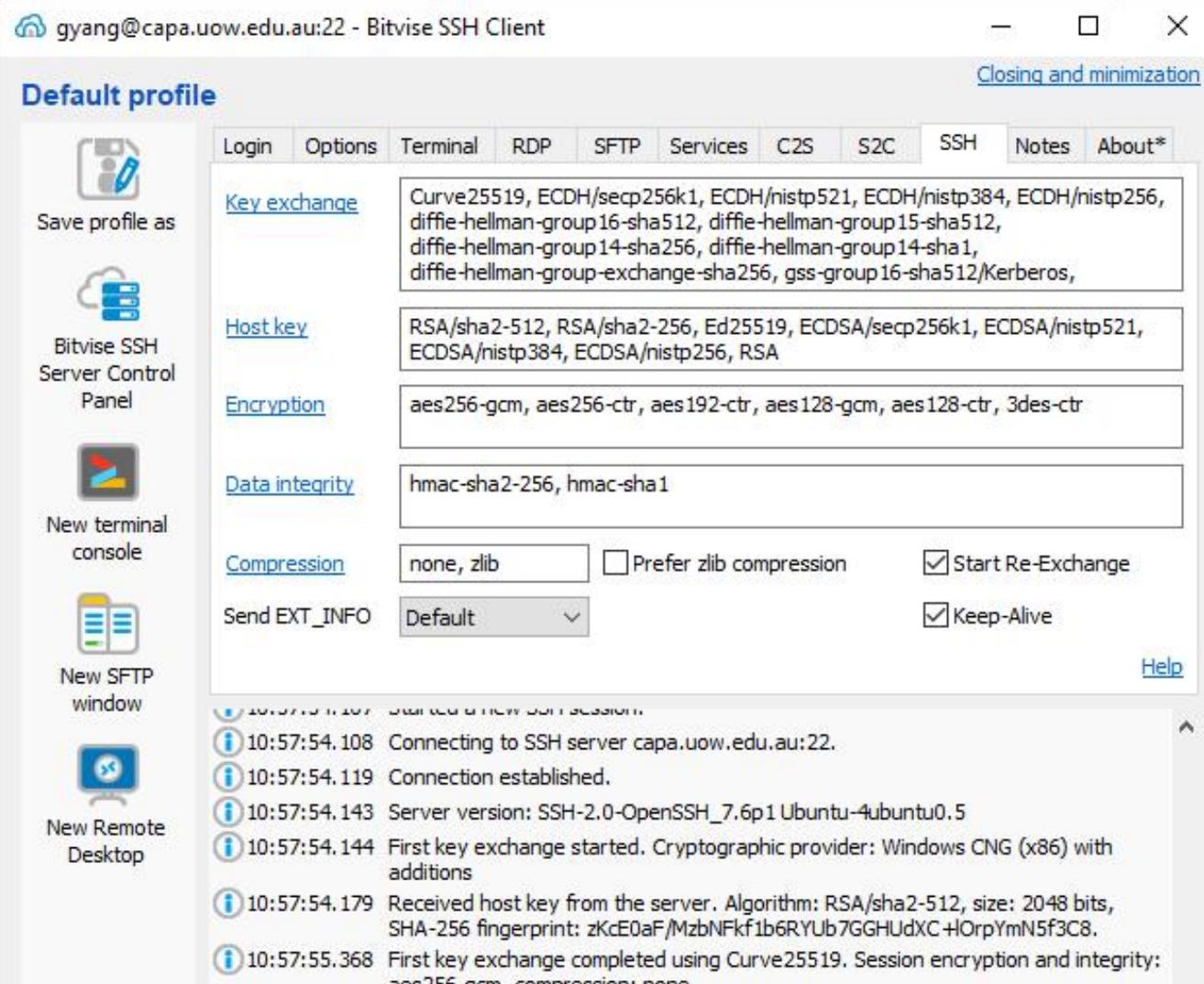
1. SSH Transport Layer Protocol.
  - Initial connection and Algorithm negotiation
  - Server authentication (is server valid?)
  - Key exchange.
  - Create a secure channel (encryption and verify)
2. SSH Authentication Protocol
  - Client authentication (who the client is) over secure transport layer channel.
3. SSH Connection Protocol
  - Supports multiple connections over a single transport layer protocol secure channel.

# 3.5 SSH: SSH Transport Layer Protocol (1/7)



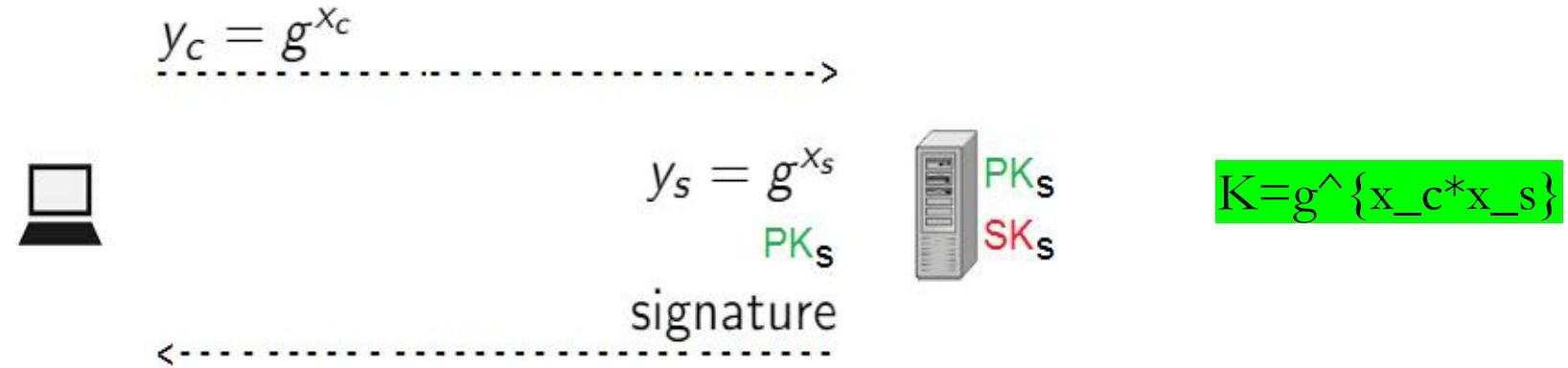
1. Client Initiation and Server Response
2. Algorithm Negotiation (KEXINIT): The server and client exchange KEXINIT messages to negotiate which algorithms will be used for key exchange, encryption, MAC, and compression.
3. Then, key exchange and server authentication

# 3.5 SSH: SSH Transport Layer Protocol (2/7)



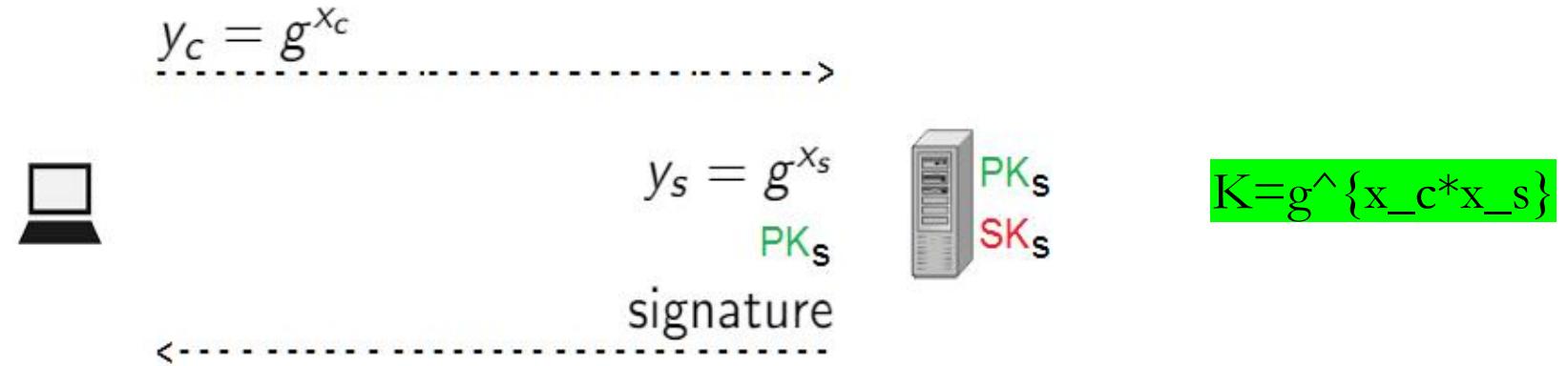
**Algorithm Negotiation (KEXINIT):**  
The server and client exchange KEXINIT messages to negotiate which algorithms will be used for key exchange, encryption, MAC, and compression.

## 3.5 SSH: SSH Transport Layer Protocol (3/7)



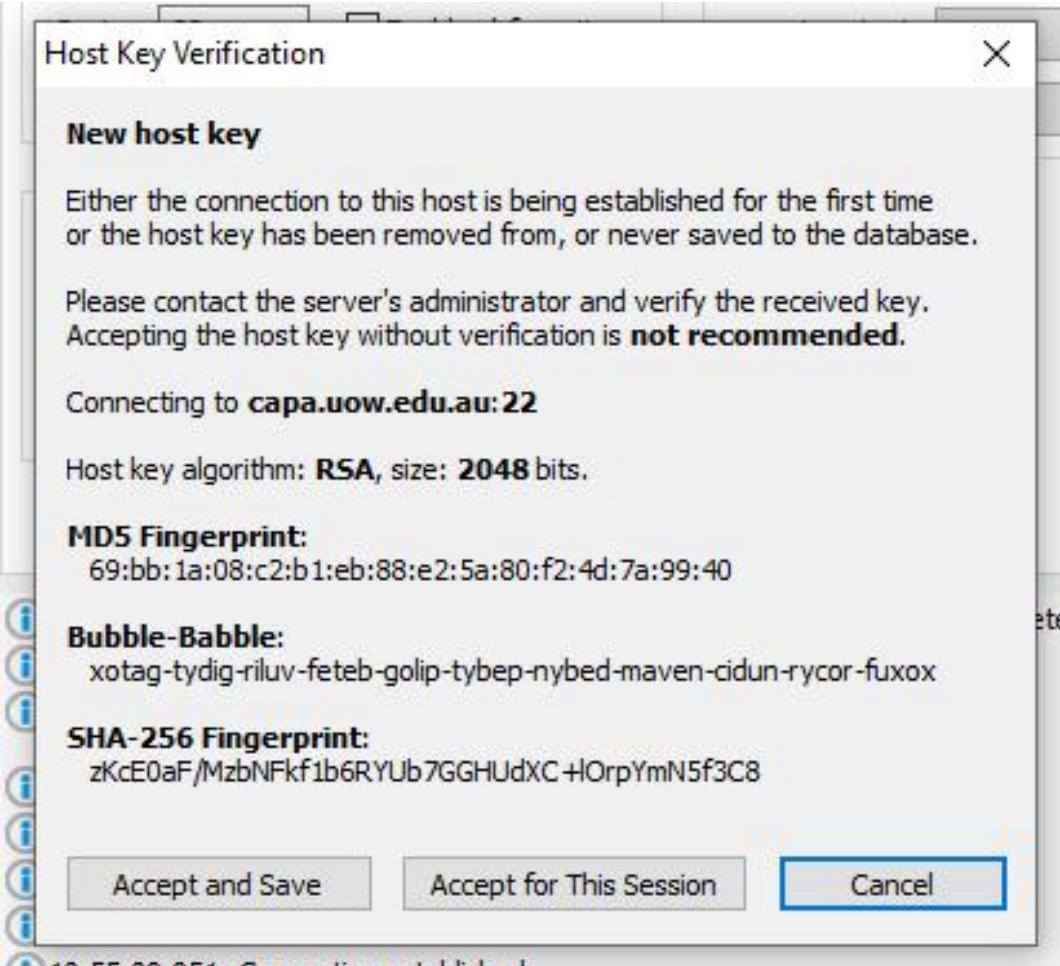
- Server computes the **exchange hash** value  
$$H = \text{hash}(id_C \parallel id_S \parallel init_C \parallel init_S \parallel PK_S \parallel y_c \parallel y_s \parallel K)$$
  - $id_S, id_C$ : Server's and Client's identification strings
  - $init_S, init_C$ : Server's and Client's Initial Messages
- Server generates the signature on the exchange hash value  $\text{signature} = \text{Sign}_{SK_S}(H)$  and sends  $(y_s, PK_S, \text{signature})$  to Client.

## 3.5 SSH: SSH Transport Layer Protocol (4/7)



- Server generates the signature on the exchange hash value  $\text{signature} = \text{Sign}_{SK_S}(H)$  and sends  $(y_s, \text{PK}_s, \text{signature})$  to Client.
- Client can compute K after receiving them from server.
- Client then can verify(authenticate) the server via  $\text{PK}_S$

## 3.5 SSH: SSH Transport Layer Protocol (5/7)



- Client then can verify(authenticate) the server via  $\text{PK}_S$ . **The verification is manual without using digital certificate.**
- After the key exchange, both Server and Client obtain two common values:
  - a **shared secret** value  $K$ , and
  - an **exchange hash** value  $H$ .
- Then, derive keys from  $(K,H)$ .

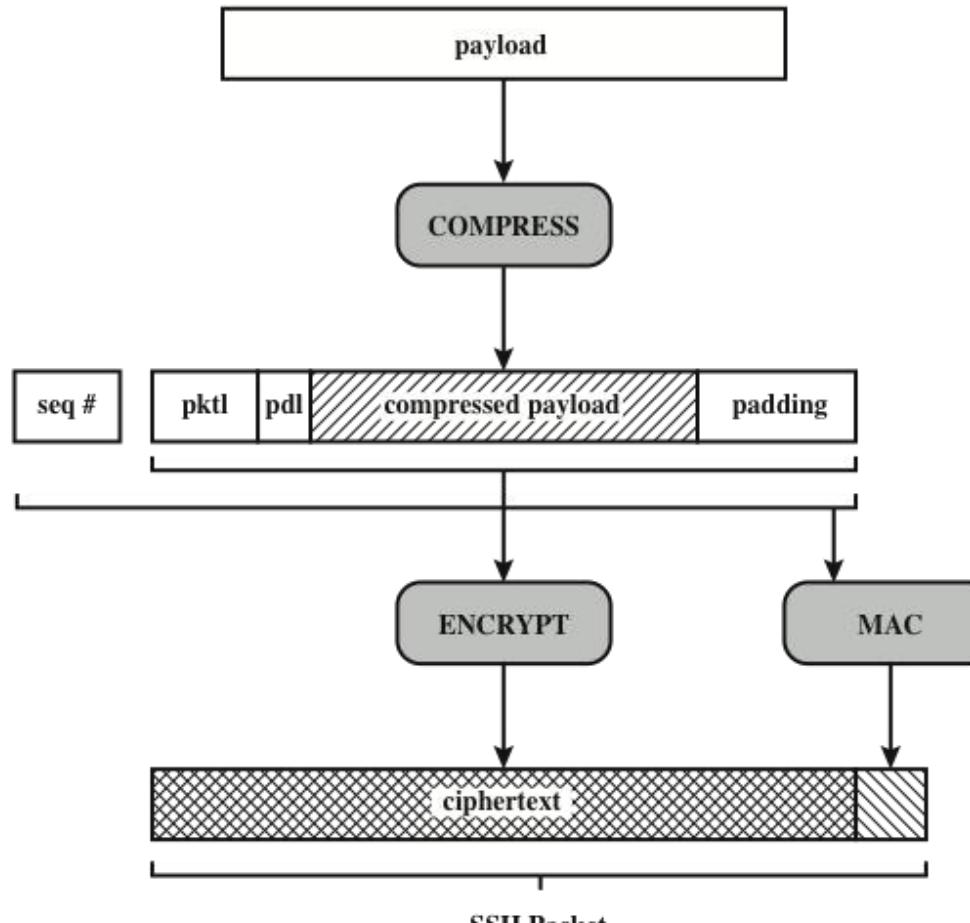
## 3.5 SSH: SSH Transport Layer Protocol (6/7)

Encryption keys must be computed as hash of the shared secret K as follows:

- Initial IV client to server:  $\text{hash}(K \mid H \mid \text{'A'} \mid \mid \text{session id})$
- Initial IV server to client:  $\text{hash}(K \mid H \mid \text{'B'} \mid \mid \text{session id})$
- Encryption key client to server:  $\text{hash}(K \mid H \mid \text{'C'} \mid \mid \text{session id})$
- Encryption key server to client:  $\text{hash}(K \mid H \mid \text{'D'} \mid \mid \text{session id})$
- MAC key client to server:  $\text{hash}(K \mid H \mid \text{'E'} \mid \mid \text{session id})$
- MAC key server to client:  $\text{hash}(K \mid H \mid \text{'F'} \mid \mid \text{session id})$

**Purpose:** ensures stronger security, protection against replay and key-compromise attacks,

# 3.5 SSH: SSH Transport Layer Protocol (7/7)



pktl = packet length  
pdl = padding length

# 3.5 SSH: SSH Authentication Protocol

Authenticates client to server (who the client is) via secure channel

- password
- public key (digital signature)

Client: Service Request: ssh-userauth

Server: Service Accept: ssh-userauth

Client: Auth Request: Password, user 'alice', password 'X'

Server: Auth Success: Welcome, alice!

Client: Service Request: ssh-userauth

Server: Service Accept: ssh-userauth

Client: Auth Request: Password, user 'alice', pk

Server: Auth Challenge: R

Client: Auth Response: Signature on R using sk

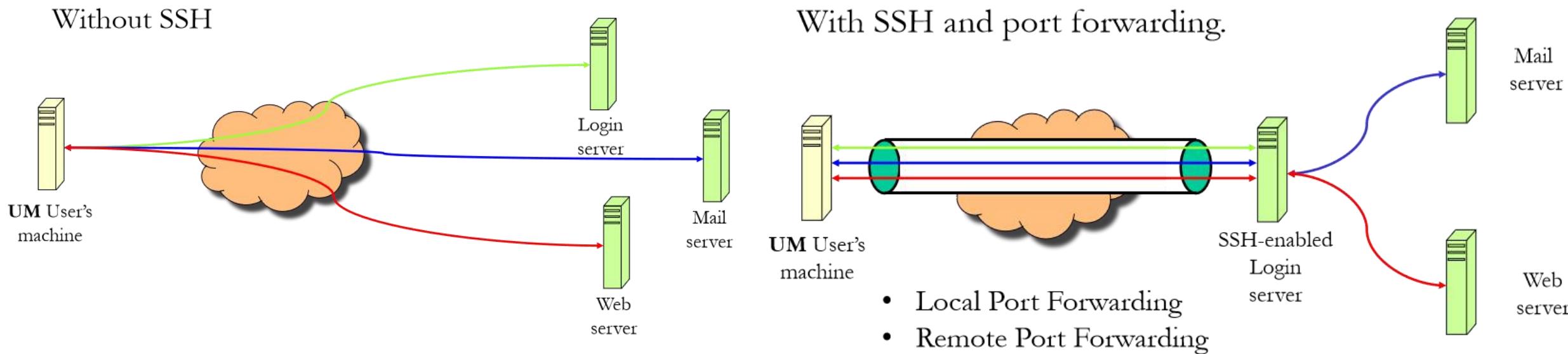
Server: Auth Success: Welcome, alice!

## 3.5 SSH: SSH Connection Protocol

- After the secure connection is established, the client can open multiple channels, each representing a different type of service (e.g., remote shell, file transfer).
- Each channel operates independently within the same SSH connection and has its own unique ID (to distinguish different service like port number).
- Three main types of channels:
  - **Session Channel:** Typically used for running remote commands or opening a terminal shell.
  - **Direct-TCP/IP Channel:** For port forwarding, where SSH tunnels connections from a local port to a remote system.
  - **Forwarded-TCP/IP Channel:** Used for reverse port forwarding, allowing a server to forward connections back to the client.

## 3.5 SSH: Port Forwarding (1/4)

Port forwarding in SSH is a technique that allows **secure tunneling** of network traffic from one machine to another over an encrypted SSH connection. It is often used to bypass firewalls, securely access remote services, or redirect traffic.



## 3.5 SSH: Port Forwarding (2/4)

How it works:

- The client establishes an SSH connection to the remote server.
- The client selects a local port  $x$  and configures SSH to accept incoming traffic on this port, forwarding it to port  $y$  on the remote application server.
- The client instructs the SSH server to create a connection to the destination at port  $y$  on the application server.
- Any data sent to the local port  $x$  is forwarded to the SSH server through the established SSH session.
- The SSH server decrypts this data and transmits the plaintext to port  $y$  of the application server.

## 3.5 SSH: Port Forwarding (3/4)

### Applications of SSH and Port Forwarding

- Accessing Blocked Servers: If firewalls have blocked all connections to server A, except through SSH, we can leverage SSH with port forwarding to access the web server running on server A.
- Securing Communications: By using SSH with port forwarding, we can secure the communications of various applications, effectively creating a Virtual Private Network (VPN). This allows for encrypted data transmission between the client and the server, ensuring privacy and security.

## 3.5 SSH: Port Forwarding (4/4)

In the context of a Virtual Private Network (VPN), "virtual" refers to the creation of a simulated or logical connection that does not rely on a physical infrastructure.

- Simulated Environment: The term "virtual" indicates that the connection is established through software rather than physical cables or hardware. This allows devices to connect securely over the internet as if they were on the same local network.
- Separation from Physical Location: A VPN allows users to connect to remote networks or services from anywhere, creating the illusion of being in a different physical location. Users can appear to be accessing the internet from the VPN server's location, rather than their actual geographical location.
- Secure Tunneling: The "virtual" aspect also implies that data is transmitted through a secure tunnel over the internet, keeping it isolated from other internet traffic. This virtual tunnel encrypts the data, ensuring privacy and security.

## 3.5 SSH: Summary of Limitations

- **Limited Accessibility:** Secure Command-Line Shells using SSH are primarily designed for administrators, making them less user-friendly for the general public.
- **Complexity of Setup:** Configuring SSH with port forwarding can be complex, and misconfigurations may result in security vulnerabilities or connectivity issues. This complexity makes it less accessible as VPN solutions.

# **END OF L3**

NETWOCK  
SECURITY

SECURITY

NETWOCK SECURITY



# Network Security

## Lecture 4

A/Prof. Fuchun Guo

# Outline

4.1 TCP/UDP Protocols

4.2 TLS Security Protocol

4.3 DTLS Security Protocol

4.4 QUIC Security Protocol

## 4.1 TCP/UDP

The protocol where two applications can talk to each other

# 4.1 TCP/UDP: Story and Motivations (1/4)



Imagine two countries, Country A and Country B, separated by a single path that only allows one postman to deliver parcels at a time. Each parcel cannot be larger than 1x1x1 meter. In Country A, there is a person named Alice, who wants to send items to Bob, who lives in Country B.

- How can the parcel be delivered correctly between any two persons in Country A and B, respectively?
- What happens if the item is too big and exceeds the parcel size limitation?

# 4.1 TCP/UDP: Story and Motivations (2/4)

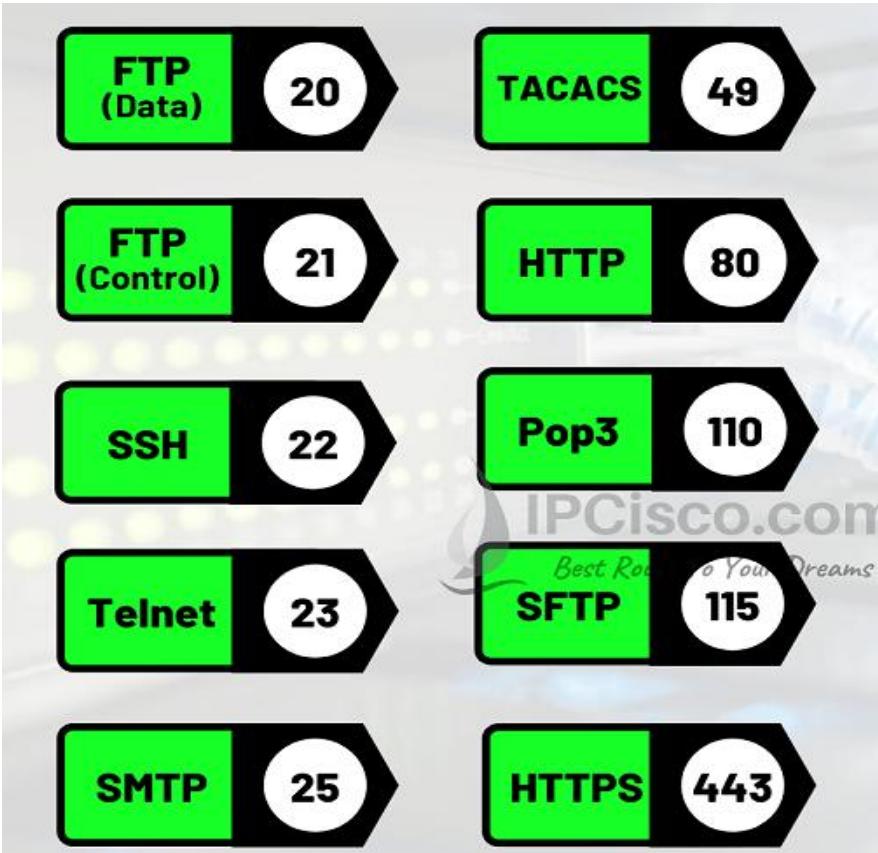


Imagine two computers, Computer A and Computer B, connected by a single data cable that only allows one data packet to be sent at a time. Each data packet cannot be larger than 1K bytes, representing the maximum size allowed for each transmission.

On Computer A, there are multiple applications, just like people wanting to send messages. One application wants to send data to a corresponding application on Computer B. Now, two key questions arise:

- How can the data packet be delivered correctly between any two specific applications on the respective computers?
- What happens if the data packet is too big and exceeds the size limitation?

# 4.1 TCP/UDP: Story and Motivations (3/4)



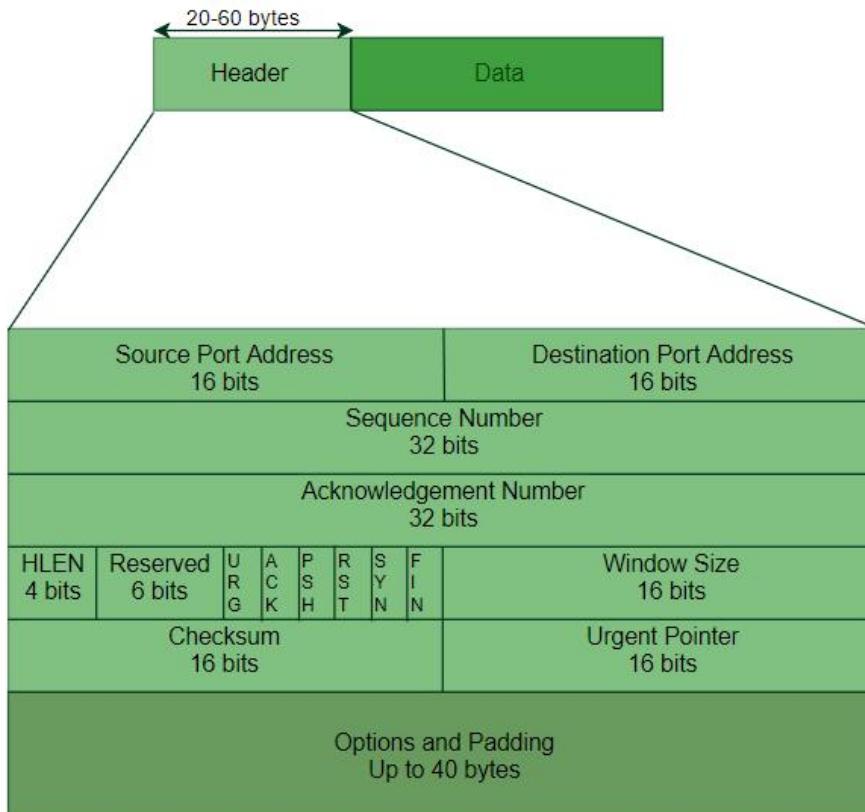
(Fuchun: No need to remember these numbers)

The solution to delivering data packets correctly between specific applications on two computers lies in using **port numbers**. Each computer has multiple applications running, and each application is assigned a unique port number, which acts as an address within the computer.

When data is sent:

1. The data packet includes the destination computer and the port number of the specific application.
2. Upon reaching the destination computer, the system looks at the port number in the packet to determine which application should receive the data.

# 4.1 TCP/UDP: Story and Motivations (4/4)



What happens if the data packet is too big and exceeds the size limitation?

- In network communication, if a data packet exceeds the maximum size allowed by the network (known as the Maximum Transmission Unit or MTU), it is broken down into smaller chunks called **segments**. Each segment is sent individually, and the receiving computer **reassembles** these segments into the original data. This process ensures that even large amounts of data can be sent efficiently over the network, despite size limitations.

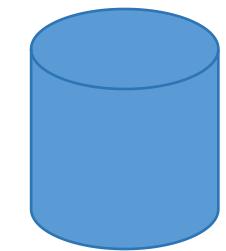
# 4.1 TCP/UDP: TCP (1/6)

TCP (Transmission Control Protocol) is a reliable, connection-oriented protocol used for data transmission over networks. It ensures that data is delivered accurately and in the correct order between two devices.

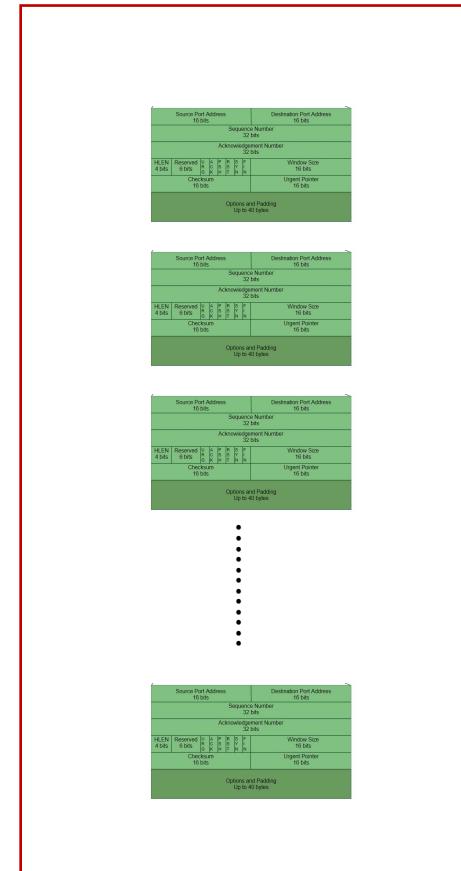
Key features of TCP include:

- **Reliable Delivery**: TCP ensures that data is received without errors, and if any packet is lost, it retransmits the missing packets.
- **Orderly Delivery**: Packets are sent and reassembled in the same order they were transmitted.
- **Connection Establishment**: Before data transmission, TCP establishes a connection through a process called the three-way handshake.
- **Flow Control**: TCP regulates the rate of data transmission to avoid overwhelming the network.

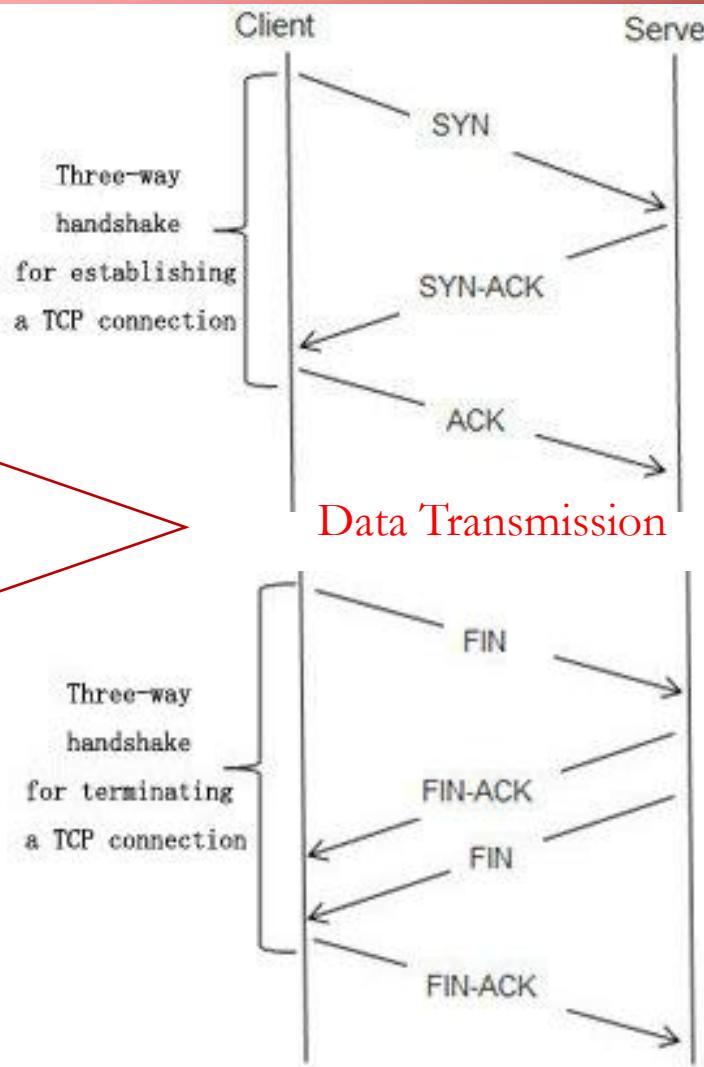
# 4.1 TCP/UDP: TCP (2/6)



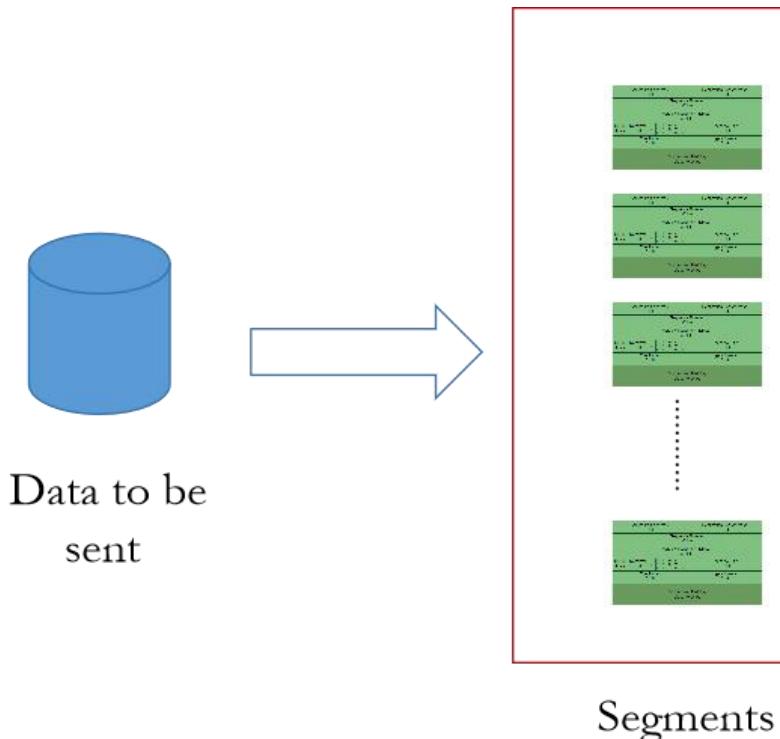
Data to be sent



Segments



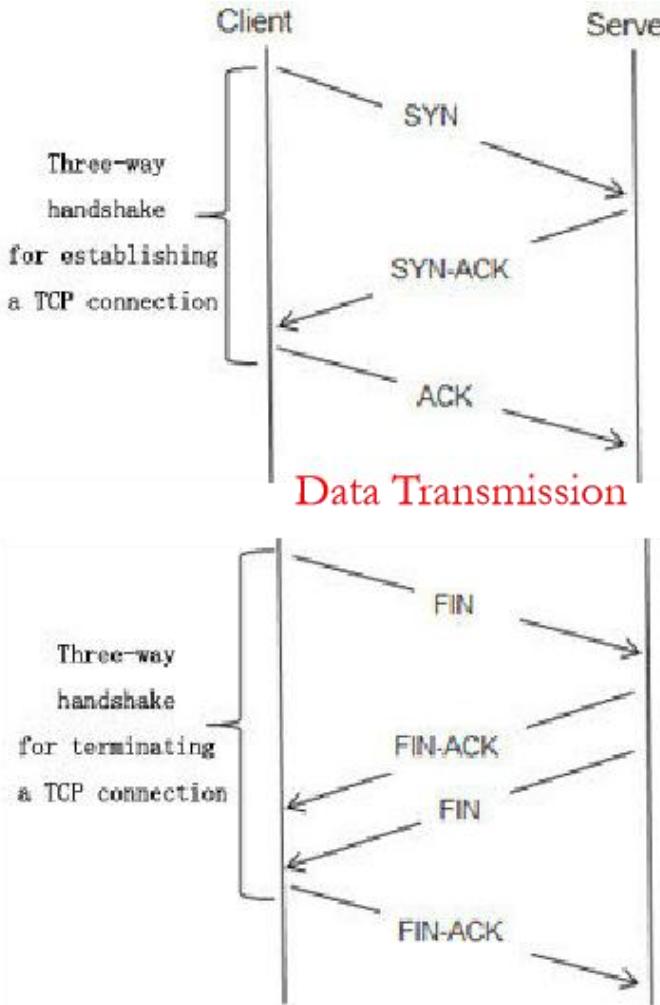
# 4.1 TCP/UDP: TCP (3/6)



When data is broken down into segments, it's essential to consider

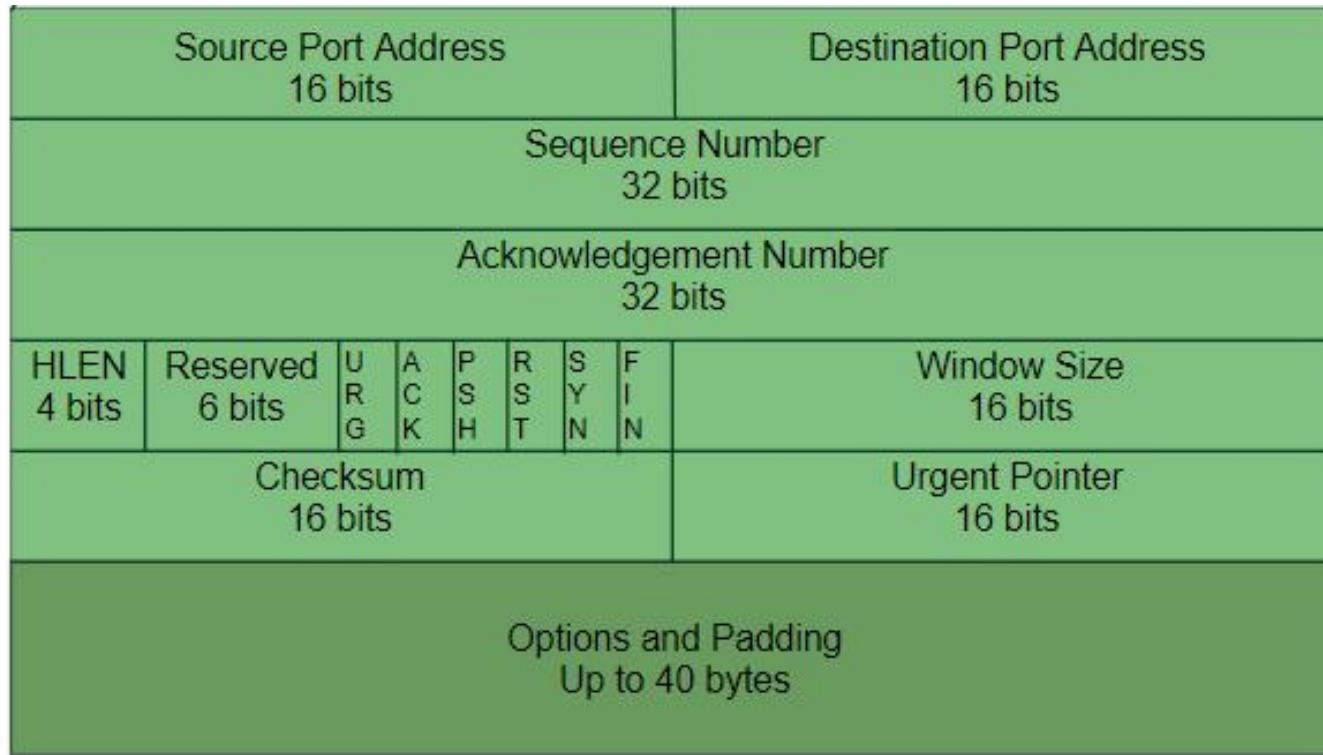
- how the receiver can reassemble the segments correctly, and
- how to handle any potential data errors.

# 4.1 TCP/UDP: TCP (4/6)



- 1. Handshake for Establishing a Connection:** The handshake process establishes a reliable connection between the sender and receiver before any data transmission occurs. This ensures that both parties are ready for communication and agree on the parameters of the session
- 2. Data Flow:** This component manages the reliable transfer of data between the sender and receiver after the connection is established. It ensures that the data is sent, received, and acknowledged in an orderly manner.
- 3. Termination:** This component cleanly ends the TCP connection between the sender and receiver, ensuring that all data has been transmitted and acknowledged before closing the connection.

# 4.1 TCP/UDP: TCP (5/6)



**Sequence Number (32 bits):** This field indicates the sequence number of the first byte of data in this segment (or the index of segment).

**Acknowledgment Number (32 bits):** This field indicates the next expected byte from the sender. It tells the sender that all bytes up to (but not including) this number have been successfully received by the receiver.

**Flags (6 bits):** Includes control flags such as SYN, ACK, FIN, etc. Each flag serves a specific communication purpose in the communication process

# 4.1 TCP/UDP: TCP (6/6)

Key features of TCP include:

- **Reliable Delivery:** It uses mechanisms like checksums to detect errors in the transmitted segments and uses acknowledgments (ACKs) to confirm the successful receipt of segments.
- **Orderly Delivery:** Each segment is assigned a sequence number, allowing the receiver to order the segments properly.
- **Connection Establishment:** It uses a three-way handshake process (SYN, SYN-ACK, ACK) before data transmission begins. This process ensures that both the sender and receiver are ready for communication and that they can synchronize their sequence numbers.
- **Flow Control:** It employs flow control mechanisms (specifically the sliding window protocol, called window size in header) to manage the rate of data transmission .

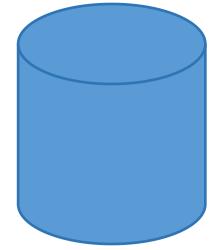
# 4.1 TCP/UDP: UDP (1/4)

UDP (User Datagram Protocol) is a simple, connectionless protocol used for **fast** data transmission over networks. It prioritizes speed and efficiency, making it ideal for real-time applications where delays are more critical than occasional data loss, such as live streaming and online gaming.

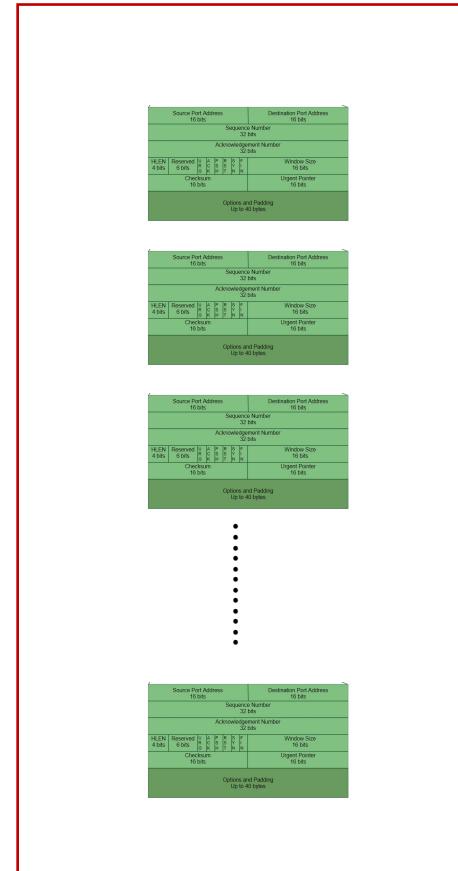
Key benefits of UDP include:

- **Speed and Efficiency:** UDP provides low latency and fast data transmission. This makes it ideal for real-time applications like online gaming and video streaming.
- **Broadcast and Multicast Support:** UDP enables efficient broadcasting and multicasting, allowing a single packet to be sent to multiple recipients simultaneously. This is beneficial for applications that need to deliver data to many users at once, such as live streaming or online conferences.

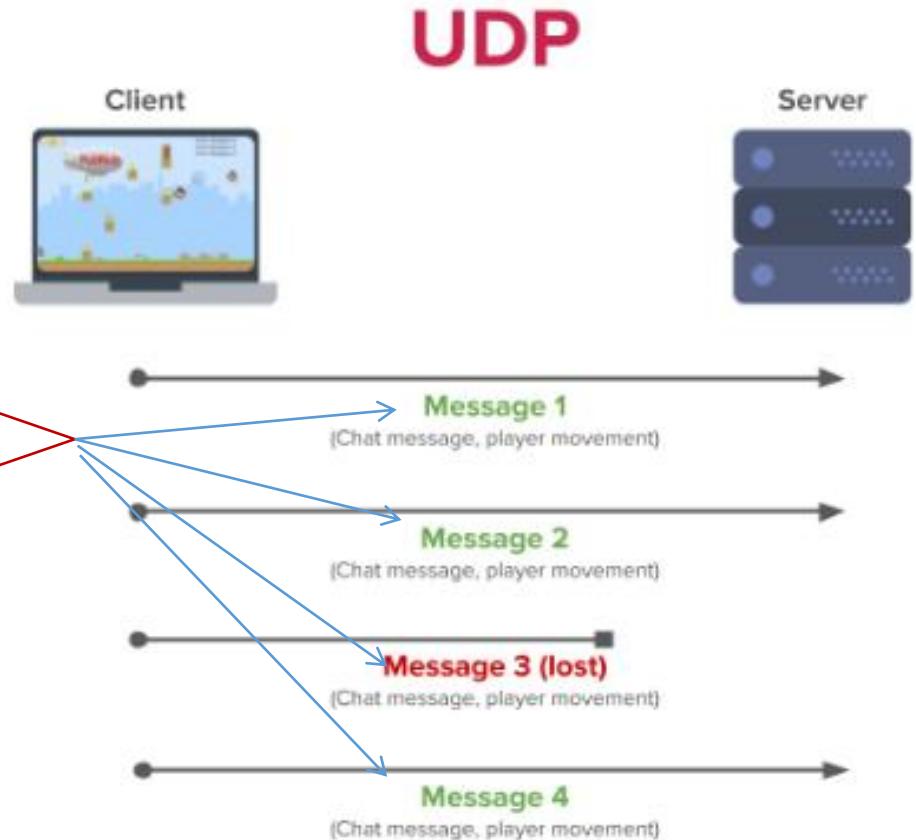
# 4.1 TCP/UDP: UDP (2/4)



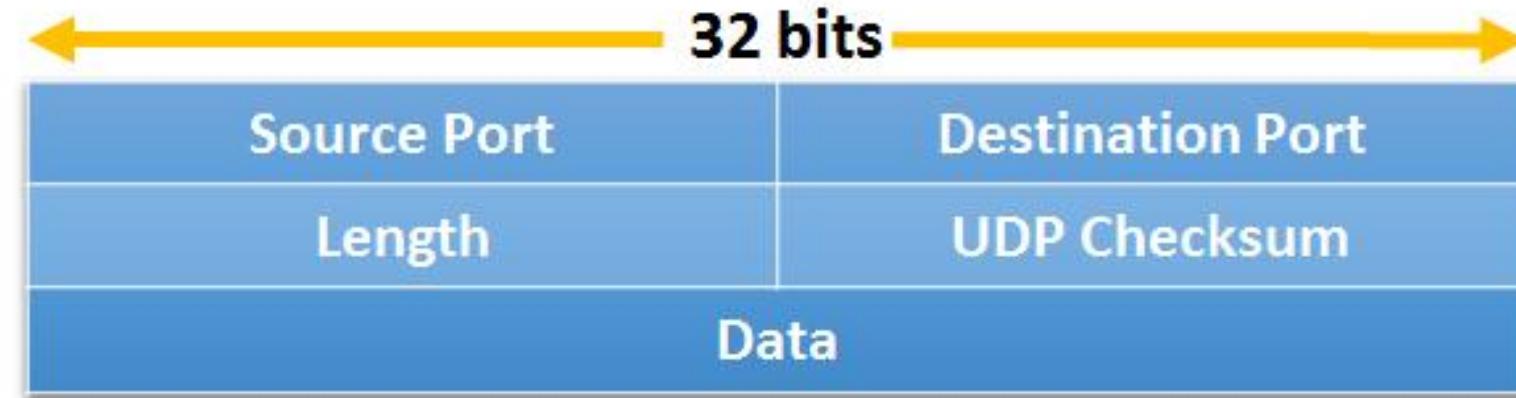
Data to be sent



Segments



## 4.1 TCP/UDP: UDP (3/4)



The UDP segment structure is simple compared to TCP. It consists of four fields in its header, and then payload:

- Source Port (16 bits): The port number of the sender, identifying the sending application.
- Destination Port (16 bits): The port number of the receiver, identifying the receiving application.
- Length (16 bits): The total length of the UDP segment, including the header and data.
- Checksum (16 bits): Used for **error-checking** (NO recovery) the header and data, ensuring data integrity.

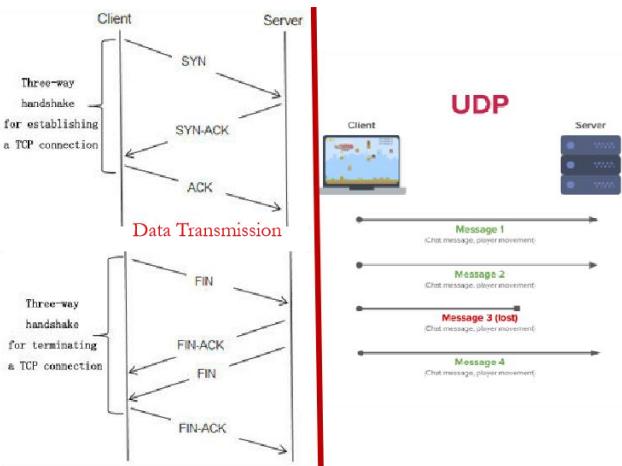
After the header comes the data (payload), which contains the actual information being sent. UDP's simplicity makes it faster and more efficient but less reliable than TCP.

# 4.1 TCP/UDP: UDP (4/4)

Key benefits of UDP:

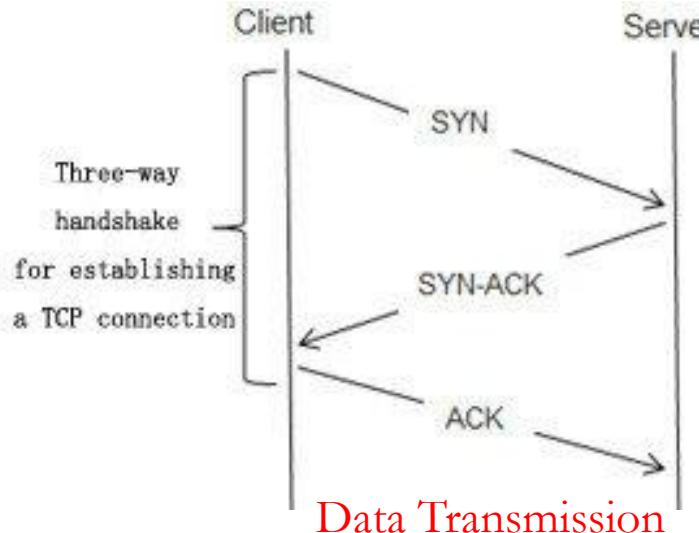
- **Speed and Efficiency**: UDP provides low latency and fast data transmission by eliminating overhead associated with connection establishment, acknowledgments, and retransmissions. This makes it ideal for real-time applications like online gaming and video streaming.
- **Broadcast and Multicast Support**: UDP enables efficient broadcasting and multicasting because it lacks the need for connection establishment, ACKs, and sequence numbers. Its simple segment structure, with minimal header overhead, allows a single packet to be sent to multiple recipients without the complexity of managing individual connections or ensuring reliable delivery, making it ideal for use cases like live streaming or online conferencing where speed and scalability are prioritized over reliability.

# 4.1 TCP/UDP: Comparison



- TCP (Transmission Control Protocol) is connection-oriented and ensures reliable data delivery. It uses mechanisms like error-checking, acknowledgments (ACKs), sequence numbers, and flow control to guarantee that data is transmitted accurately and in order. However, these features introduce additional overhead, making TCP slower but reliable, which is ideal for applications like file transfers and web browsing.
- UDP (User Datagram Protocol) is connectionless and focuses on fast, lightweight communication. It does not provide reliability, ordering, or error-checking **beyond a basic checksum**(**no re-transmission due to error**), making it faster but less reliable than TCP. UDP is well-suited for applications like streaming, gaming, or any situation where speed is more important than data integrity.

# 4.1 TCP/UDP: TCP SYN Flooding (1/3)

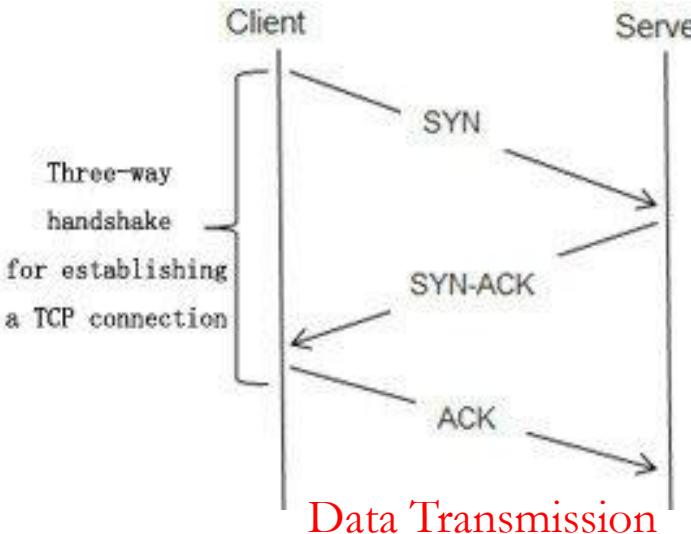


In a normal TCP connection, after a client sends a SYN packet to initiate a connection with a server:

- The server responds with a SYN-ACK packet, indicating it's ready to establish the connection.
- The server then allocates resources and waits for the client's ACK response, which never arrives.

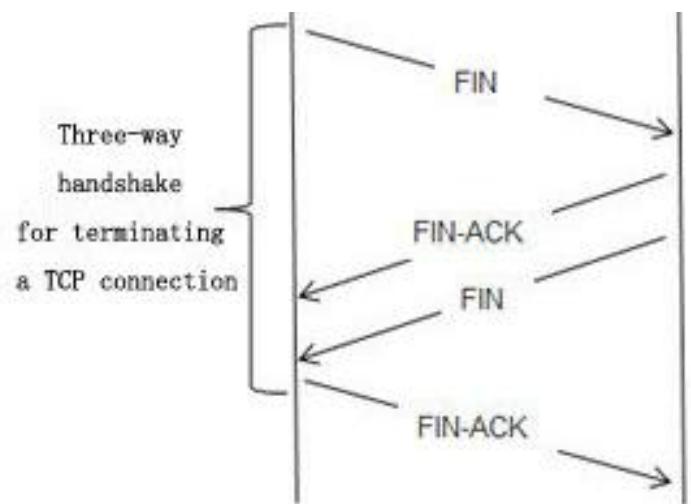
Waiting for clients could result in vulnerability

# 4.1 TCP/UDP: TCP SYN Flooding (2/3)



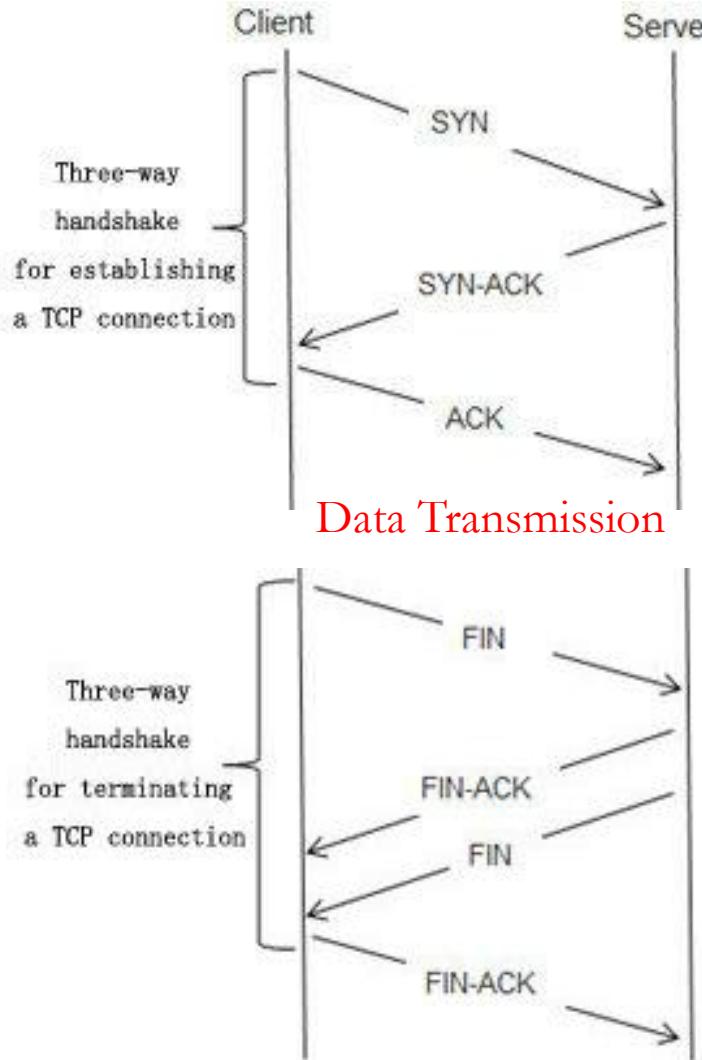
A SYN Flood attack is a type of Denial of Service (DoS) attack that targets the TCP handshake process. Here's a brief explanation:

In a normal TCP connection, the client sends a SYN (synchronize) request to the server to initiate a connection. The server responds with a SYN-ACK (synchronize-acknowledge), and the client replies with an ACK (acknowledge), completing the three-way handshake.



In a SYN Flood attack, the attacker sends **many** SYN requests to the server but never completes the handshake by sending the final ACK. The server allocates resources and waits for the ACK that never comes, resulting in a large number of half-open connections. As the server's resources are consumed, it becomes overwhelmed and unable to handle legitimate connections, leading to service disruption.

# 4.1 TCP/UDP: TCP SYN Flooding (3/3)



To mitigate a SYN Flood attack, we can use:

- **SYN Cookies:** This technique involves the server not allocating resources immediately upon receiving a SYN request. Instead, the server encodes information (namely cookie) in the SYN-ACK packet, and resources are only allocated if the client responds with the correct ACK. This avoids resource exhaustion from half-open connections.
- **TCP Filtering:** Implementing filters that drop suspicious or malformed SYN packets can help prevent excessive SYN requests from being processed.

## 4.1 TCP/UDP: UDP Flooding (Amplification Attack) (1/3)

UDP Flooding (Amplification Attack) is a type of Distributed Denial of Service (DDoS) attack that exploits the stateless nature of the User Datagram Protocol (UDP) and often amplifies the attack using a feature like UDP-based services that respond with larger data than the request sent.

1. **Attacker Sends Small UDP Packets:** The attacker sends a large volume of small UDP packets to various servers. These packets typically request data from services like DNS, NTP, or other UDP-based services that reply with larger responses.
2. **Spoofing the Source IP:** The attacker forges (spoofs) the source IP address of the UDP packets to make it appear as though they are coming from the victim's IP address.
3. **Amplified Response:** The server responds to the spoofed IP with a much larger data payload than the original request. Since UDP is connectionless, the server has no way to verify the legitimacy of the source IP address.
4. **Flooding the Victim:** The victim (not the server)'s network gets overloaded with massive amounts of response data, effectively clogging the network and degrading or taking down services.

## 4.1 TCP/UDP: UDP Flooding (Amplification Attack) (2/3)

Example:

- Memcached DDoS Attack (2018): This attack set records in amplification, peaking at 1.7 Tbps of traffic. It leveraged Memcached, a database caching system, to perform an amplification attack. By sending small requests (15 bytes) to vulnerable Memcached servers, attackers could trigger responses as large as 50,000 times the size of the original request, swamping the victim with an overwhelming flood of data.
- Overwhelmed Resources: The victim's network bandwidth, CPU, and memory get overwhelmed by the large amount of incoming traffic.

## 4.1 TCP/UDP: UDP Flooding (Amplification Attack) (3/3)

Third-Party Server:

1. **Rate Limiting and Traffic Shaping:** Third-party servers can implement rate limits on responses to reduce the amount of traffic they can send out.
2. **Response Filtering:** Using response filtering techniques can prevent servers from sending large responses unless the request is verified.
3. **Disable Unnecessary UDP Services:** Third-party servers should disable UDP services that are not essential to avoid them being used for amplification.

Victim:

1. **Limit UDP-based Communication:** The victim's network can reduce its reliance on UDP-based services, as these are most vulnerable to amplification attacks.
2. **Implement Firewalls and Intrusion Prevention Systems:** Victims can configure firewalls to detect abnormal traffic patterns that could indicate an amplification attack. Blocking traffic from known vulnerable third-party servers can reduce the effect of the attack.

## 4.2 TLS

The protocol that encrypts talks before TCP between applications in different devices.

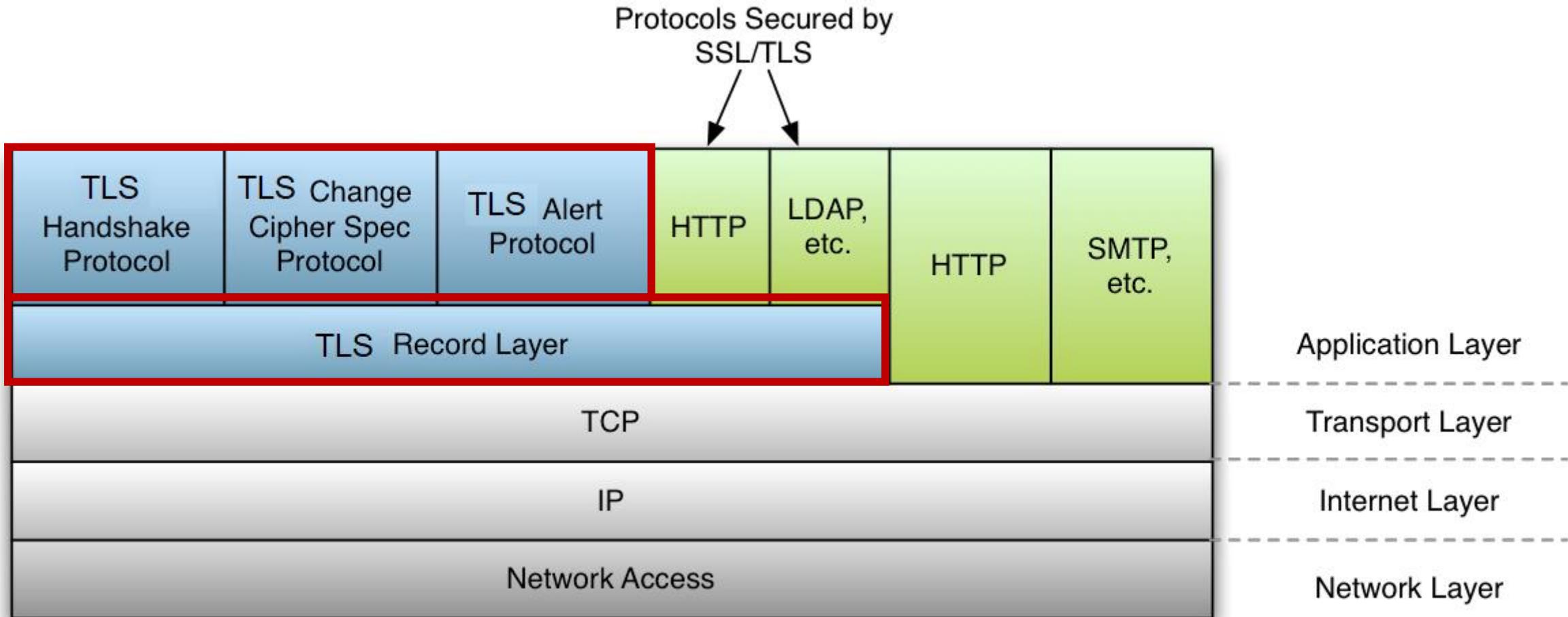
## 4.2 TLS: Background (1/2)

- **TCP Payload in Plaintext:** The payload in TCP packets consists of the actual data being communicated by applications. Without additional protection, this data is transmitted in plaintext, making it vulnerable to interception or tampering by malicious adversary.
- **TLS for Payload Protection:** The TLS (Transmission Control Protocol) protocol secures the communication by encrypting the payload of TCP packets, ensuring confidentiality, integrity, and authenticity. This prevents eavesdroppers from reading or altering the transmitted data.

## 4.2 TLS: Background (2/2)

1. **Origin of SSL:** SSL (Secure Sockets Layer) was developed by Netscape in 1994 to secure internet communications, providing confidentiality, integrity, and authentication.
2. **SSL Versions:** The first version, SSL 1.0, was never released due to security flaws. SSL 2.0 was released in 1995, followed by SSL 3.0 in 1996, which addressed many vulnerabilities.
3. **Transition to TLS:** In 1999, the Internet Engineering Task Force (IETF) standardized SSL 3.0 as TLS (Transport Layer Security) version 1.0, incorporating improvements and updates.
4. **Subsequent Versions:** TLS 1.1 was introduced in 2006, addressing weaknesses in TLS 1.0. In 2008, TLS 1.2 was released introducing support for stronger encryption algorithms and improved security mechanisms.
5. **TLS 1.3:** The latest version, TLS 1.3, was finalized in 2018, significantly simplifying the handshake process, reducing latency, and improving security by removing outdated cryptographic algorithms.

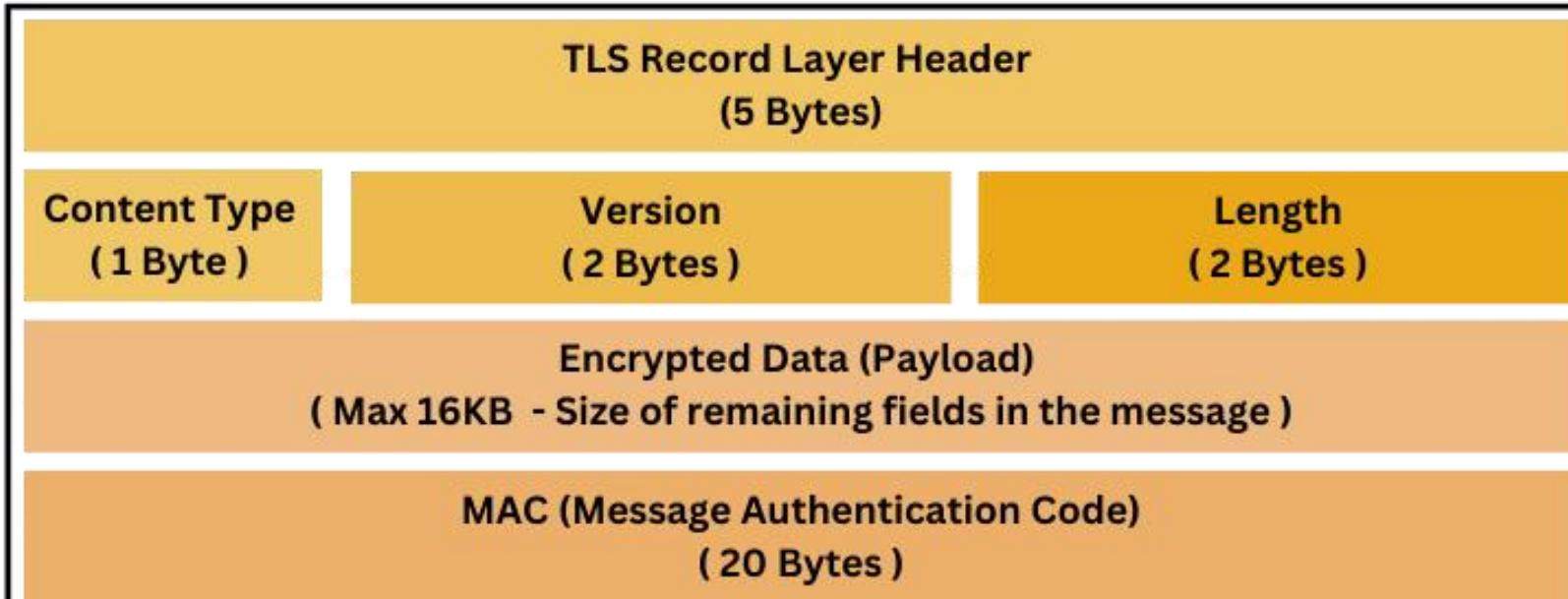
# 4.2 TLS: Architecture (1/2)



## 4.2 TLS: Architecture (2/2)

1. **Handshake Protocol:** It establishes a secure connection between the client and server by negotiating cryptographic algorithms, authenticating the parties, and exchanging session keys.
2. **Record Protocol:** It is responsible for providing confidentiality and integrity through encryption and message authentication, and ensuring that data is transmitted securely over the established TLS connection.
3. **Alert Protocol:** It communicates error and warning messages, indicating issues that arise during the handshake or data transfer phases and facilitating graceful termination of the connection when necessary.
4. **Change Cipher Spec Protocol:** It is a part of the TLS handshake process that signals the transition to a new cipher suite. When activated, it informs another party that subsequent messages will be protected using the negotiated encryption and authentication algorithms.

## 4.2 TLS: Record Protocol (1/5)

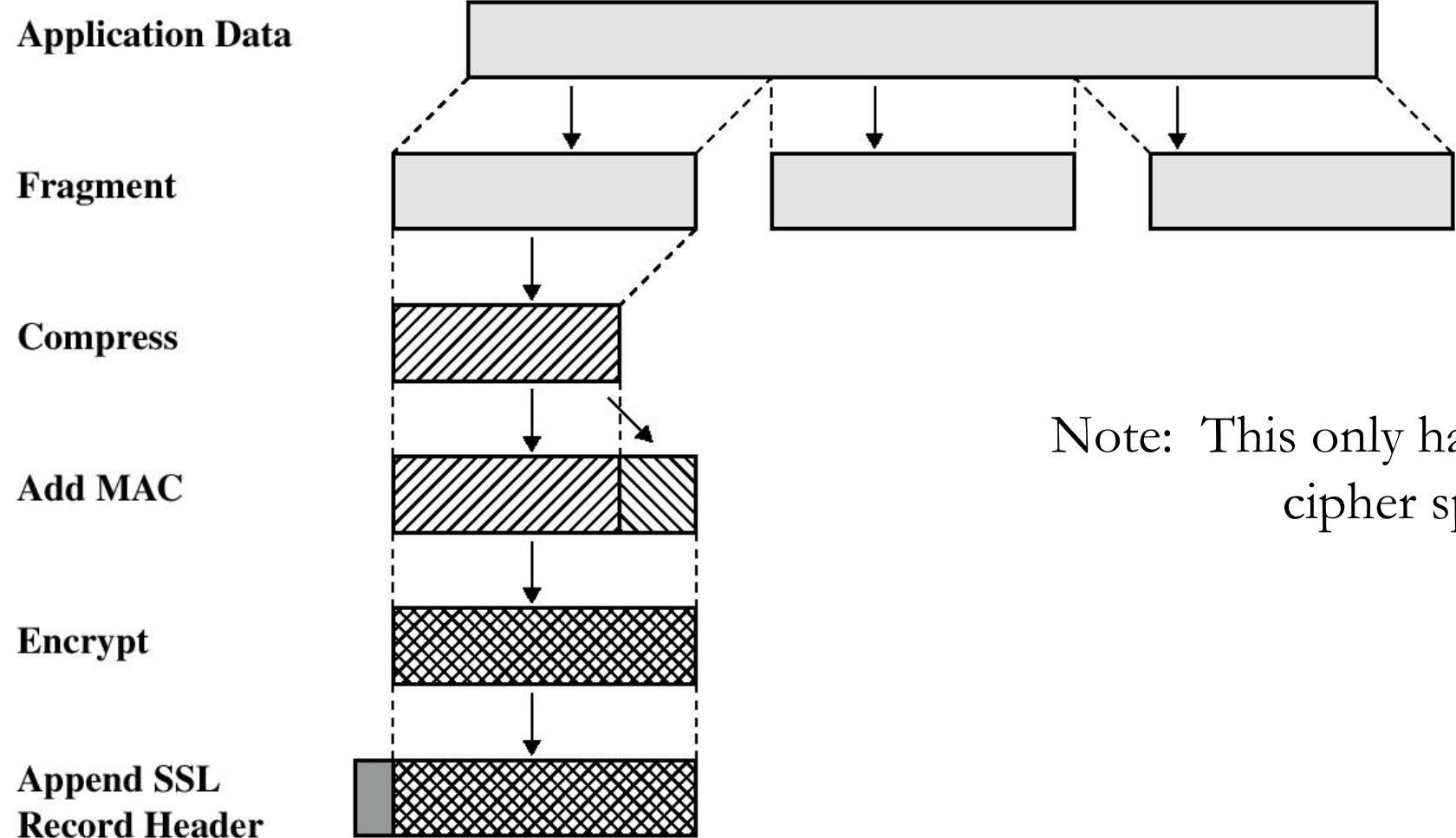


Note: plaintext before handshake protocol and change cipher spec protocol

**Record Protocol Header:** Each TLS record begins with a header that contains :

- Content Type: This type of payload protocol (e.g., Handshake, Application Data, Alert).
- Version: This specifies the version of the TLS protocol being used.
- Length: This field indicates the length of the data in the record.

## 4.2 TLS: Record Protocol (2/5)

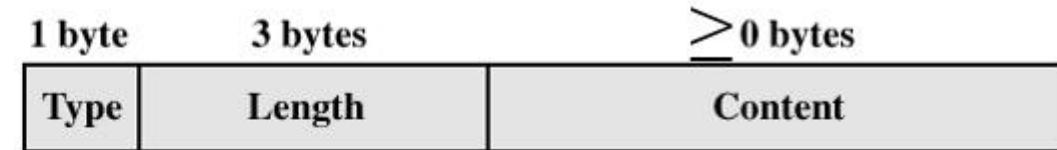


Note: This only happens after the change cipher spec protocol

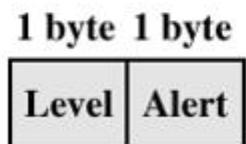
## 4.2 TLS: Record Protocol (3/5)



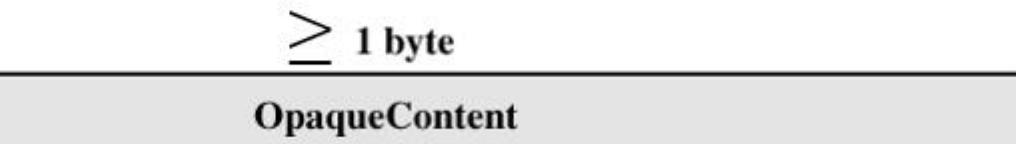
(a) Change Cipher Spec Protocol



(c) Handshake Protocol



(b) Alert Protocol

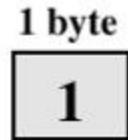


(d) Other Upper-Layer Protocol (e.g., HTTP)

Application Data



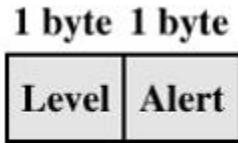
## 4.2 TLS: Record Protocol (4/5)



(a) Change Cipher Spec Protocol

- Consists of a single message with the value 1.
- The change cipher spec message is sent by both the client and server to notify the receiving party that subsequent records will be protected under the newly negotiated CipherSpec and keys.

## 4.2 TLS: Record Protocol (5/5)



(b) Alert Protocol

Fatal

- unexpected\_message
- bad\_record\_MAC
- decompression\_failure
- handshake\_failure
- illegal\_parameter
- ...

Fatal: An error that cannot be recovered from, leading to the immediate termination.

Warning

- bad\_certificate
- unsupported\_certificate
- certificate\_revoked
- certificate\_expired
- certificate\_unknown
- ...

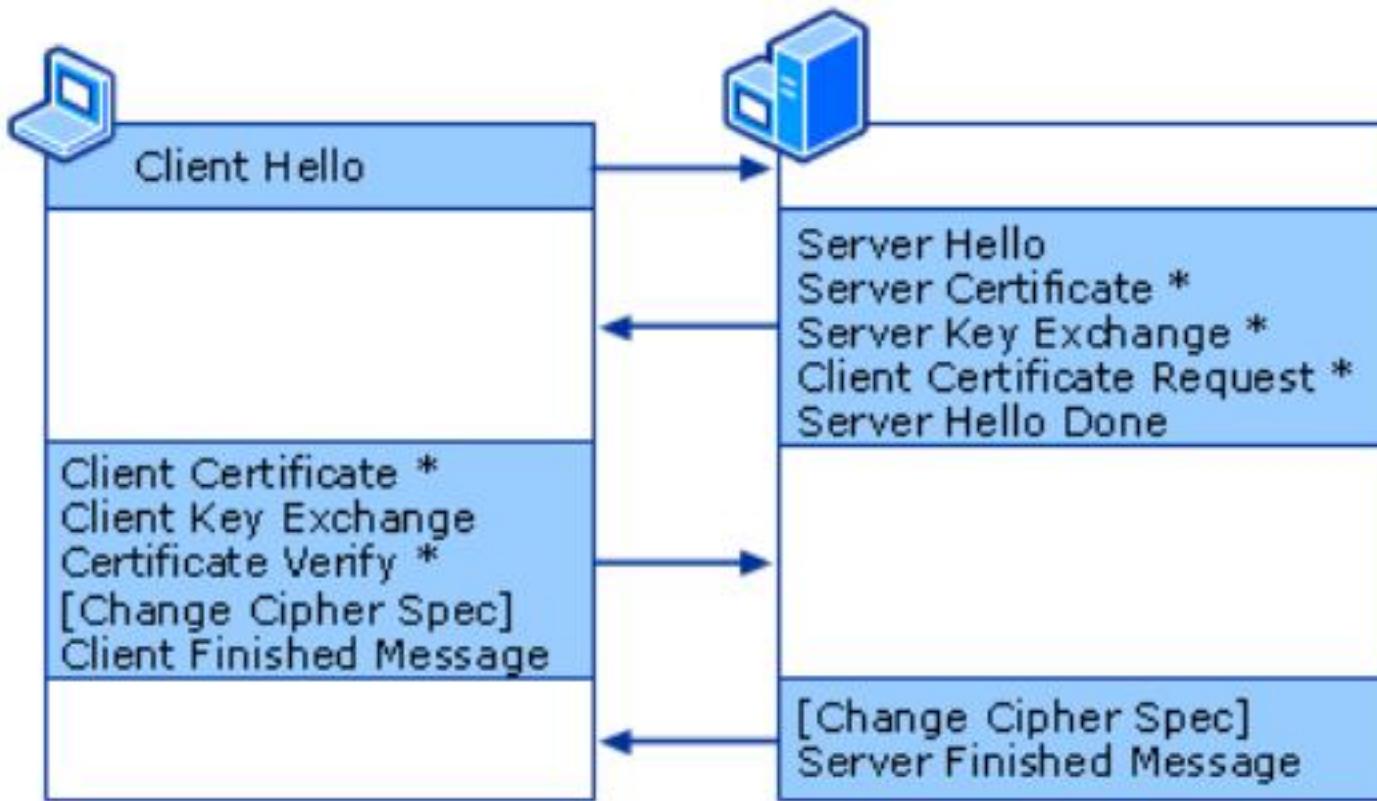
Warning: This indicates a non-critical issue, and the connection may continue.

## 4.2 TLS: Handshake Protocol (1/11)

As the most complex part of TLS, the Handshake protocol allows the server and the client to finish the following tasks:

1. Agree on a version of TLS to be used;
2. Authenticate each other (server authenticates client, optional);
3. Negotiate a set of cryptographic algorithms
4. Negotiate shared keys to be used in the TLS record protocol.

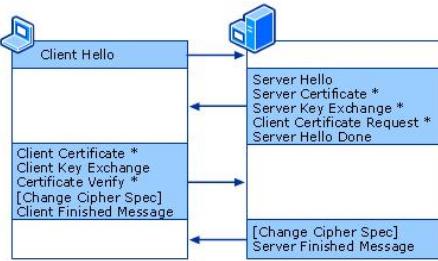
## 4.2 TLS: Handshake Protocol (2/11)



Version 1.2

1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. CertificateRequest
6. ServerHelloDone
7. ClientCertificate
8. ClientKeyExchange
9. CertificateVerify
10. Finished

# 4.2 TLS: Handshake Protocol (3/11)

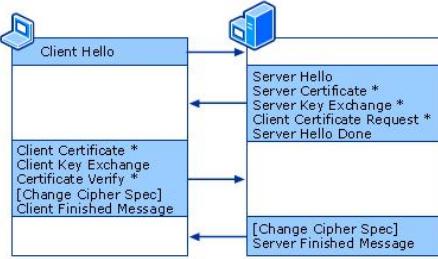


Version 1.2

1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. CertificateRequest
6. ServerHelloDone
7. ClientCertificate
8. ClientKeyExchange
9. CertificateVerify
10. Finished

1. ClientHello: Contain protocol version, client nonce as well as the client's list of preferred ciphersuites
2. ServerHello: Contain chosen protocol version, server nonce as well as the chosen ciphersuite

# 4.2 TLS: Handshake Protocol (4/11)



Version 1.2

1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. CertificateRequest
6. ServerHelloDone
7. ClientCertificate
8. ClientKeyExchange
9. CertificateVerify
10. Finished

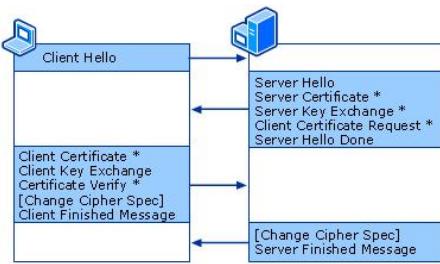
## 3. Certificate

- Required when server authentication is needed
- X.509 certificate for one of the following type (depending on the key exchange method)
  - RSA Encryption Key: pk for encryption
  - Diffie-Hellman Public Key: for secure key exchange
  - .....

The methods of generating shared key are different!

Philosophy: Don't put all eggs in one basket!

# 4.2 TLS: Handshake Protocol (5/11)

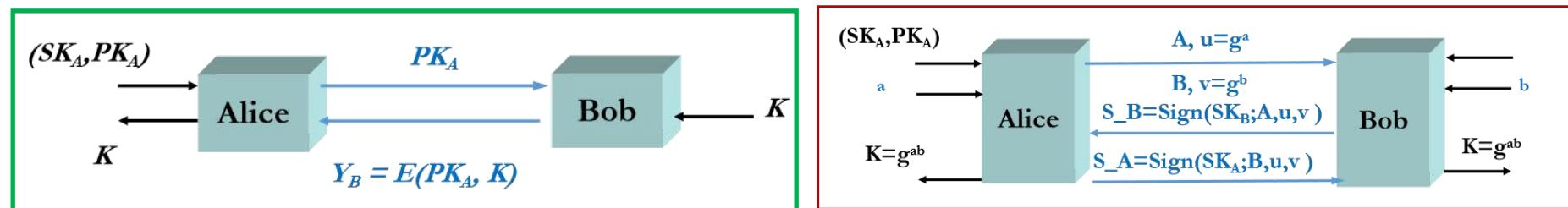


Version 1.2

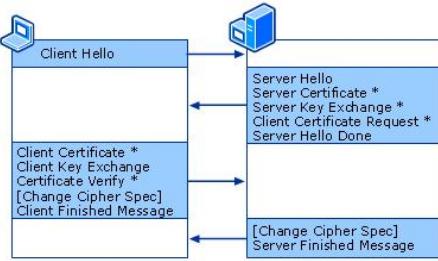
1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. CertificateRequest
6. ServerHelloDone
7. ClientCertificate
8. ClientKeyExchange
9. CertificateVerify
10. Finished

4. Server Key Exchange: send something for key exchange from server when needed

- Required when Diffie-Hellman key establishment is used
- Not required when RSA key transport is used as the key exchange method (because client will generate a key and encrypt with RSA key)



# 4.2 TLS: Handshake Protocol (6/11)



Version 1.2

1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. CertificateRequest
6. ServerHelloDone
7. ClientCertificate
8. ClientKeyExchange
9. CertificateVerify
10. Finished

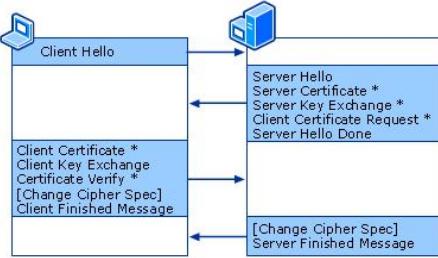
## 5. CertificateRequest

A sign of message including algorithms that the server requests and support for client-side authentication.

## 6. ServerHelloDone

A sign indicates that the server has finished sending all of its handshake messages.

# 4.2 TLS: Handshake Protocol (7/11)



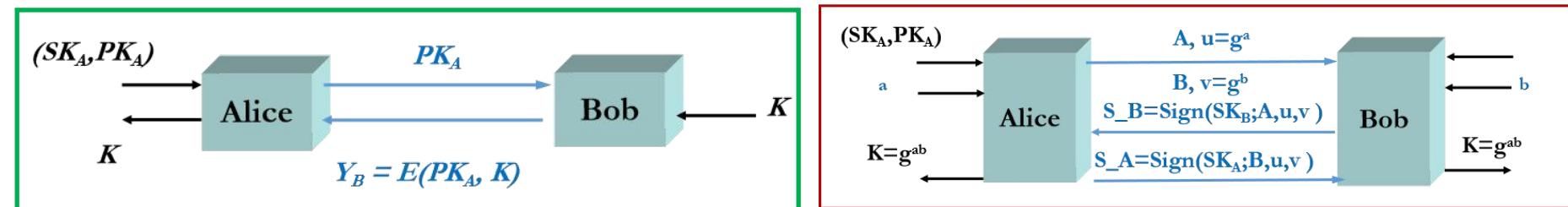
Version 1.2

1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. CertificateRequest
6. ServerHelloDone
7. ClientCertificate
8. ClientKeyExchange
9. CertificateVerify
10. Finished

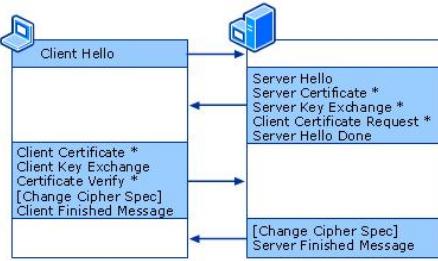
7. ClientCertificate: send client's certificate

8. ClientKeyExchange

- sends this message depending on algorithms for generating shared keys.



# 4.2 TLS: Handshake Protocol (8/11)



Version 1.2

1. ClientHello
2. ServerHello
3. Certificate
4. ServerKeyExchange
5. CertificateRequest
6. ServerHelloDone
7. ClientCertificate
8. ClientKeyExchange
9. CertificateVerify
10. Finished

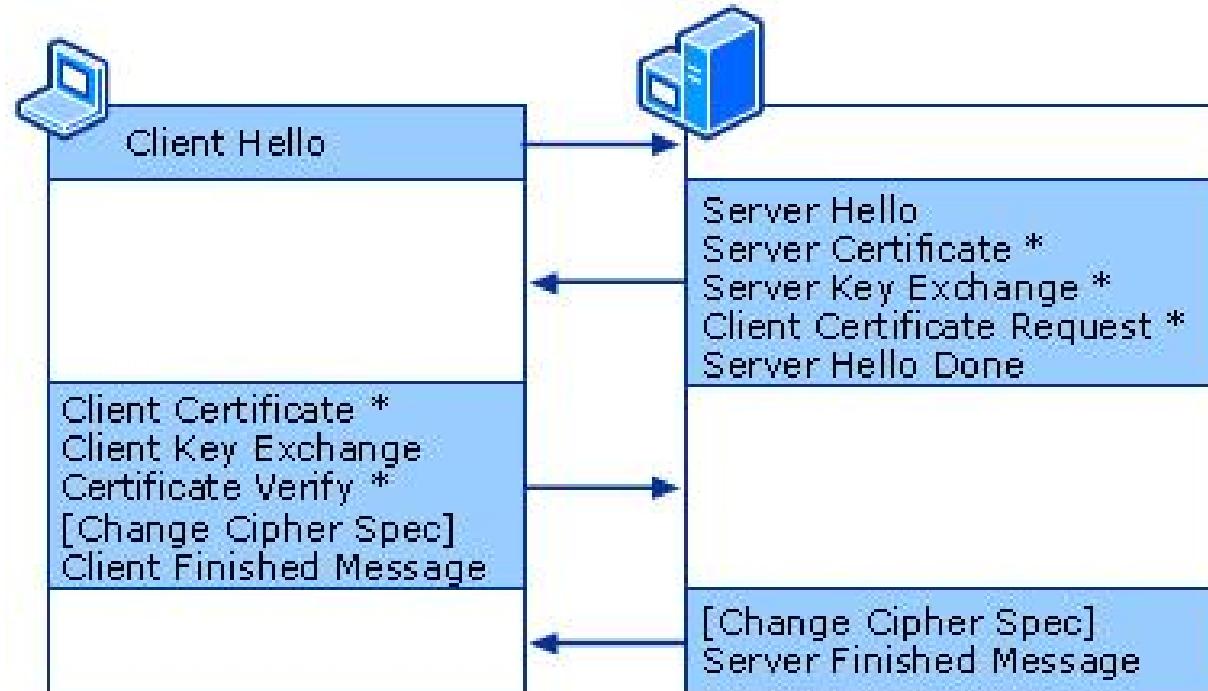
## 9. CertificateVerify

- Required only when client has a certificate for RSA/DSS signature and wants to be authenticated.
- Client signs all the handshake messages it has sent and received previously

## 10. Finished

This message is sent by both the client and server to indicate the end of the handshake process, ensuring all previous messages were sent and received correctly.

## 4.2 TLS: Handshake Protocol (9/11)



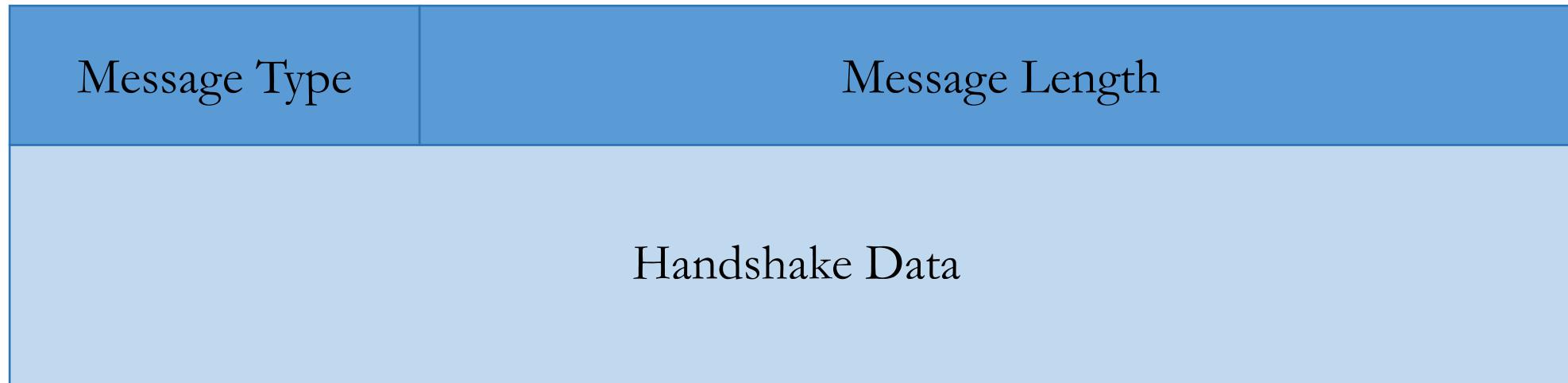
The Finished message in TLS is sent after the encryption has been established, meaning it is encrypted using the negotiated cipher suite. After both the client and server exchange the ChangeCipherSpec message, they both switch to the new encryption settings, and the Finished message is the first message encrypted using those settings.

Change Cipher Spec protocol is called before the handshake protocol (Finished ) is completed

## 4.2 TLS: Handshake Protocol (10/11)

**Handshake Protocol Header:** Each message within the Handshake Protocol also has a header:

- Message Type (1 byte): This identifies the specific handshake message type
  - (e.g., Client Hello, Server Hello, Certificate).
- Message Length: (3 bytes) This indicates the length of the message following the header.



## 4.2 TLS: Handshake Protocol (11/11)

- The key derived by an TLS key exchange method through the handshake protocol is called pre-master secret (48 bytes), which can be
  - a secret encrypted with server's RSA public key or
  - a value derived by the DH key exchange technique.
- pre-master secret → master secret → shared keys
  - master secret =  $\text{PRF}(\text{pre\_master\_secret}, \text{'master secret'}, \text{client\_random} \mid\mid \text{server\_random})$
  - shared keys =  $\text{PRF}(\text{master\_secret}, \text{'key expansion'}, \text{server\_random} \mid\mid \text{client\_random})$

Note: PRF is a pseudo-random function. Using separate encryption keys for communication between two parties not only enhances security but also provides flexibility and compliance with protocols.

## 4.2 TLS: Cipher Suite (1/2)

A cipher suite in TLS is a set of cryptographic algorithms that define how secure communication is established between a client and a server. Each cipher suite specifies the following components:

- Key Exchange Algorithm: This algorithm is used to securely exchange keys between the client and server. It establishes a shared secret that will be used for encryption.
- Authentication Algorithm: This determines how the parties authenticate each other.
- Symmetric Encryption Algorithm: This algorithm is used to encrypt the data once the secure connection has been established.
- Message Authentication Code (MAC) Algorithm: This is used to ensure the integrity and authenticity of the data transmitted over the connection. It verifies that the message has not been altered during transmission.

## 4.2 TLS: Cipher Suite (2/2)



Two Examples:

- Key Exchange: **TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256**
- Key Transport: **TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA**

Note: The key transport merges key generation and authentication together!

## 4.2 TLS: HTTPS

**HTTPS = HTTP + TLS**

- use https:// rather than http://
- use port 443 rather than 80
- encrypts URL, document contents, form data, cookies, HTTP headers
- HTTP basic user authentication can be used!

Note: The adversary can still see the domain name from TCP protocol even one IP is hosting multiple domains.

## 4.2 TLS: VPN

Bob is a programmer who has developed an app that enables communication between apps on different devices. The first version of the app transmits data in plaintext. For version 2, Bob wants to implement secure communication and is considering either SSH or TLS. What are the key differences he would face when designing version 2 using SSH or TLS?

**Answers:** Both solutions will require partial rewriting of the app. SSH typically involves more manual client-side configuration, such as setting up port forwarding. TLS, on the other hand, integrates more seamlessly with applications, offering a more automated solution for secure communication with minimal client-side configuration (often through built-in libraries).

## 4.2 TLS: Summary

### Advantage:

- Strong cryptography protections (C.I.A)
- Widely supported in many applications

### Disadvantages:

- Performance Overhead: The encryption and decryption process can introduce latency and computational overhead, particularly in resource-constrained environments.
- Certificate Management and Requirement: The need for trusted certificates
- Complex Configuration: If programmers are designing TLS based secure apps, they need to know how TLS works inside apps and implement them.

## 4.3 DTLS

protocols similar to TLS but working on UDP

## 4.3 DTLS: Background (1/2)

UDP protocols are more preferable than TCP protocols in the applications where low latency and efficiency are prioritized over reliability and ordered delivery.

- Real-Time Communications: Video conferencing require minimal delay for real-time interaction. UDP's lack of handshakes and retransmissions minimizes latency, making it suitable for these applications.
- Live Streaming: Streaming media like live broadcasts or sports events benefit from UDP, as minor packet loss is less noticeable to viewers than delays caused by retransmissions in TCP.
- Online Gaming: Many multiplayer online games use UDP to ensure fast updates of player positions and actions, where real-time responsiveness is more important than guaranteed delivery.

## 4.3 DTLS: Background (2/2)

TLS security protocol over TCP protocols is successful having the below key reasons:

- **Reliable Transmission:** Ensures no packet loss during communication.
- **Ordered Transmission:** Guarantees that all packets are received in the correct sequence.

These properties are critical for TLS functionality. If either is compromised, TLS will encounter errors. Therefore, it is not feasible to directly adapt TLS for UDP without significant modifications.

# 4.3 DTLS: Protocol (1/3)

Designing a variant of TLS over UDP must account for:

- Unreliable Nature of UDP: DTLS must handle packet loss, reordering, and duplication inherent in UDP. Mechanisms are implemented within DTLS to ensure a successful handshake under these conditions.
- Replay and Out-of-Order Protection: DTLS includes safeguards against packet replay attacks and the possibility of out-of-order packet arrivals.
- Insecurity of UDP: DTLS addresses potential security vulnerabilities associated with the open nature of UDP.
- Real-Time Constraints: DTLS must optimize for efficiency, especially when critical messages, such as those in the handshake protocol, require retransmission.

## 4.3 DTLS: Protocol (2/3)



**HelloVerifyRequest (Cookie):**

To prevent DoS attacks, the server responds with a "HelloVerifyRequest", including a cookie. The server remains stateless at this point, asking the client to send the cookie back in a subsequent message. TLS does not need this because it runs over TCP, which already establishes a connection before the handshake.

## 4.3 DTLS: Protocol (3/3)



Without considering retransmissions, reordering, and packet loss, the cookie is the key differentiator in the handshake protocol itself. The rest of the handshake steps are very similar between TLS and DTLS.

## 4.4 QUIC

advanced DTLS protocols

## 4.4 QUIC: Background (1/2)

- TCP is reliable but slow.
- Combining TLS with TCP is secure but further increases overhead.
- UDP is fast but lacks reliability and security.
- DTLS aims to create secure transmission over UDP.

	Reliable	Fast	Secure
<b>TCP Protocol</b>	YES	NO	NO
<b>UDP Protocol</b>	NO	YES	NO
<b>TLS Protocol</b>	YES	NO	YES
<b>DTLS Protocol</b>	NO	YES	YES

## 4.4 QUIC: Background (2/2)

- QUIC (Quick UDP Internet Connections) was originally developed by Google in 2012 as an experimental transport protocol. The goal was to improve web performance by reducing latency and enhancing the user experience for applications that rely on secure and fast connections.
- In 2016, Google began working with the Internet Engineering Task Force (IETF) to standardize QUIC. In May 2020, the IETF officially published RFC 9000, which standardized QUIC as an IETF protocol. This marked a significant milestone, as QUIC became an official transport layer protocol that could be widely adopted across the internet.

# 4.4 QUIC: Protocol (1/6)

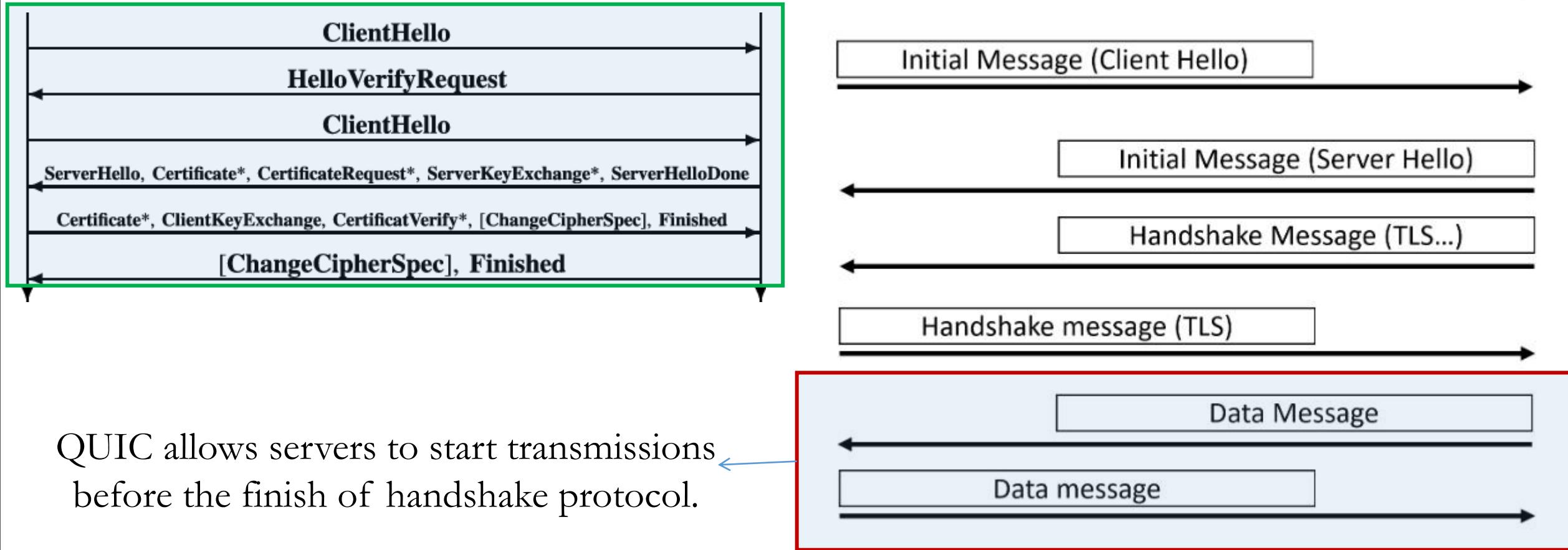
QUIC: Similar to TLS:

- Handshake Protocol: Establishes secure communication parameters and negotiates encryption algorithms.
- Change CipherSpec Protocol: Signals that the subsequent data will be encrypted using the negotiated keys.
- Alert Protocol: Handles error reporting and connection closure.
- Application Data (Record) Protocol: Transmits encrypted application data once the secure connection is established.

# 4.4 QUIC: Protocol (2/6)

DTLS

QUIC

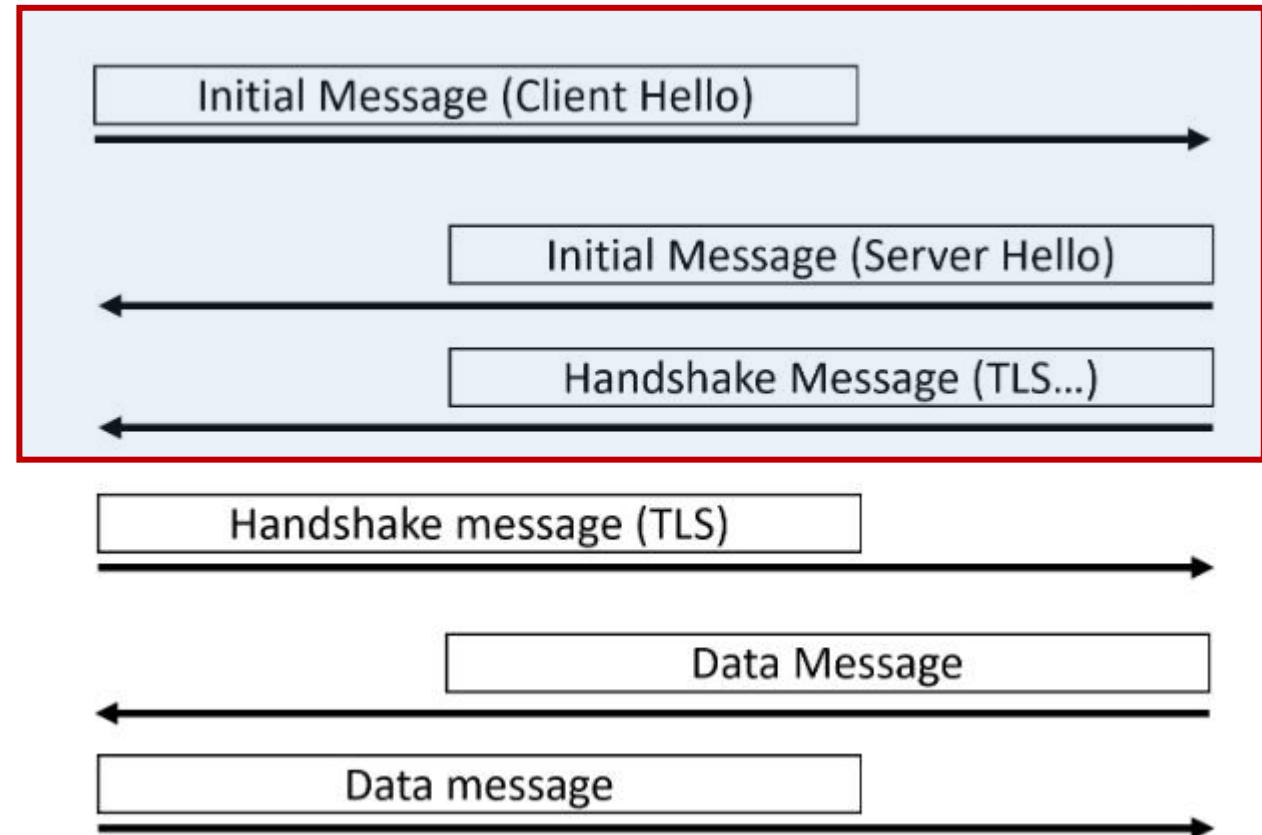


QUIC allows servers to start transmissions  
before the finish of handshake protocol.

## 4.4 QUIC: Protocol (3/6)

- In QUIC, the ClientHello message includes key exchange parameters, which allows the server to respond with a ServerHello and any additional handshake information in the same exchange.
- This setup eliminates the need for separate messages to establish encryption parameters, which is a common step in traditional handshakes.

QUIC

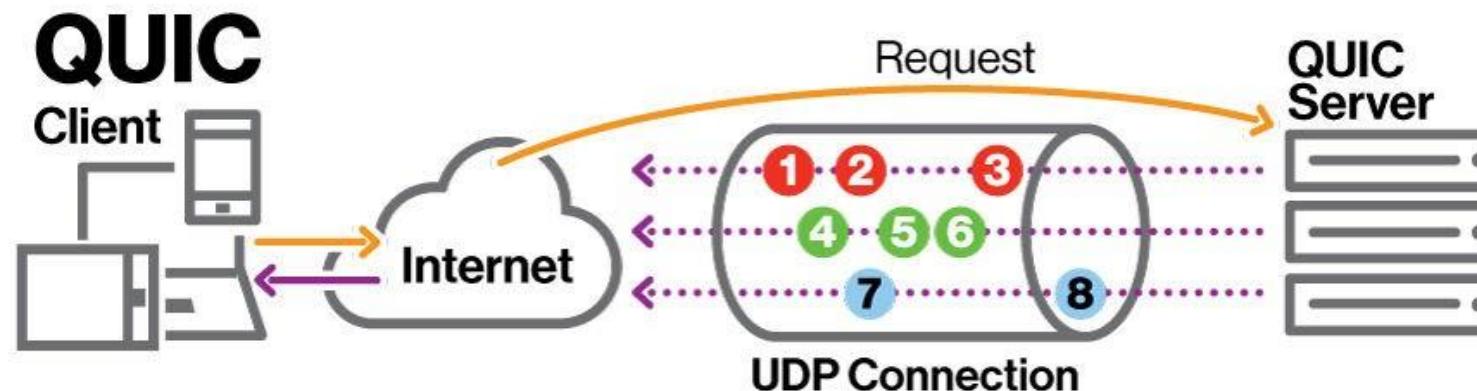
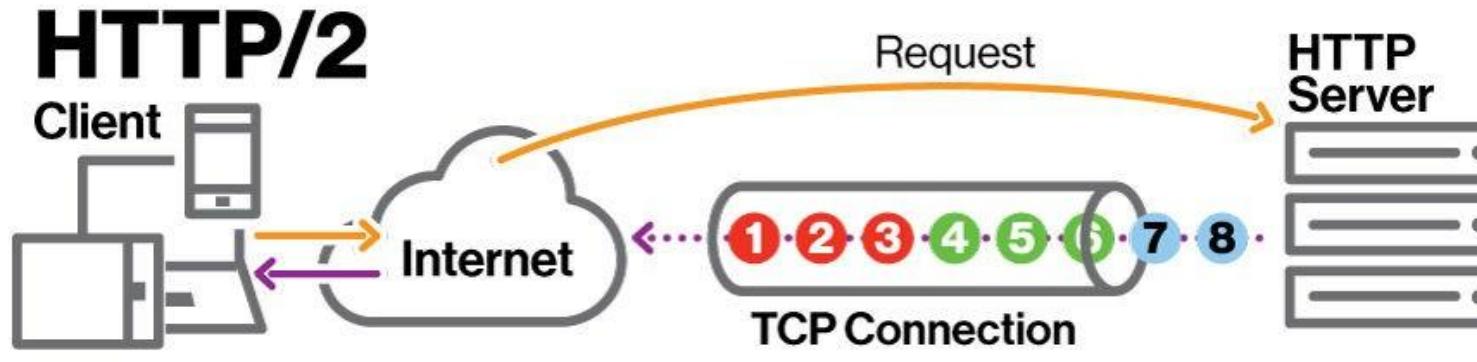


## 4.4 QUIC: Protocol (4/6)

Reliable& Fast (Together):

- Like TCP, QUIC **uses ACKs** to ensure that data is reliably delivered. Each packet sent by QUIC is acknowledged by the receiver, and if a packet is lost, QUIC retransmits it.
- QUIC tracks acknowledgments for **each packet independently**, allowing it to retransmit lost packets without affecting other streams of data.
  - TCP Head-of-Line Blocking: In TCP, when a packet is lost, the entire data stream must wait for that packet to be retransmitted before continuing. This is known as head-of-line blocking, and it can slow down communication.
  - QUIC's Solution: QUIC supports **multiplexing multiple independent streams** within a single connection. If a packet in one stream is lost, it does not block the other streams, allowing the connection to continue without unnecessary delays.

## 4.4 QUIC: Protocol (5/6)



multiplexing multiple independent streams within a single connection. If a packet in one stream is lost, it does not block the other streams.

## 4.4 QUIC: Protocol (6/6)

Congestion Control Flexibility in QUIC:

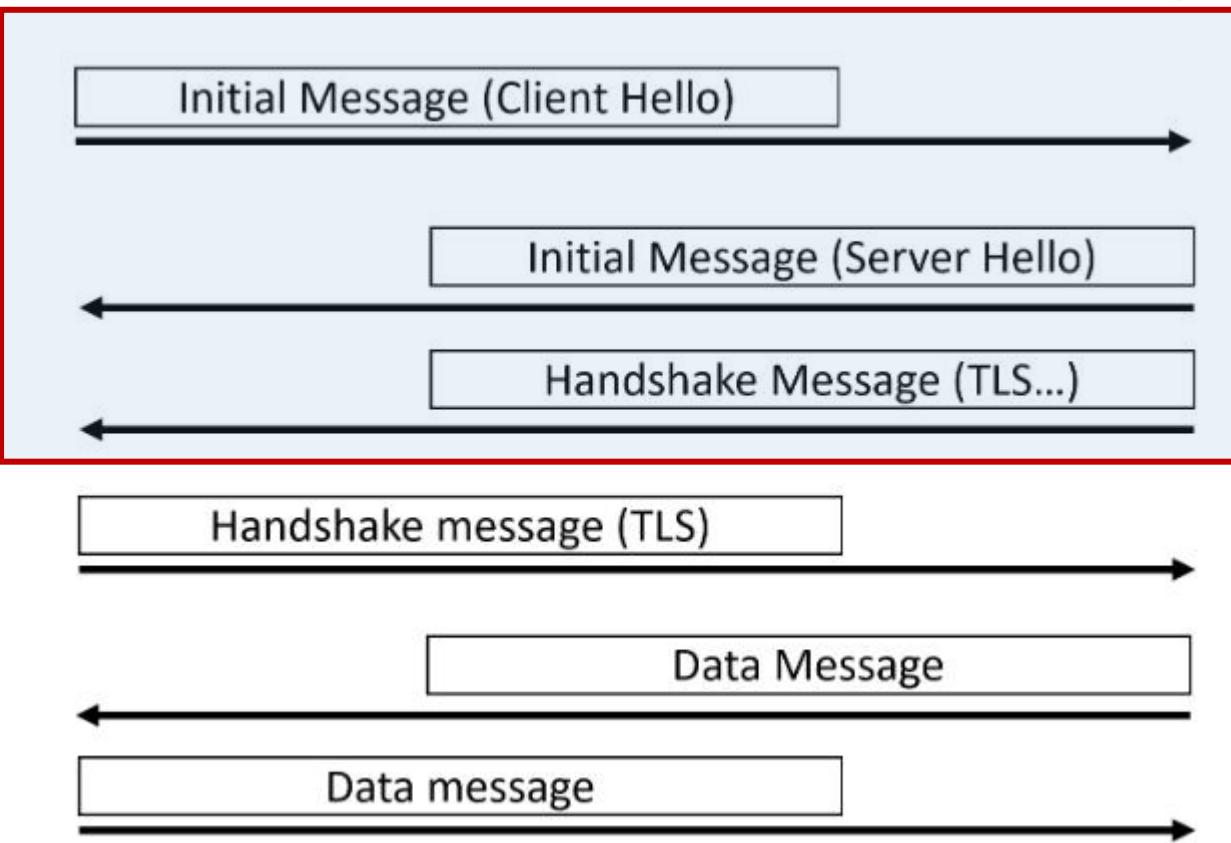
- **TCP-like Congestion Control:** QUIC supports TCP-style congestion control mechanisms. This provides a balance between efficient data transfer and avoiding network congestion.
- **UDP-Like Speed:** Unlike TCP, UDP does not have built-in congestion control, which allows it to send data continuously without worrying about network congestion, leading to lower latency in real-time applications like video or gaming. While QUIC is built on top of UDP, it can bypass TCP's strict congestion control rules to achieve faster data transfer, especially in networks with high bandwidth and low congestion.

QUIC inherits the reliability and congestion control of TCP, while offering improvements such as lower latency, and built-in security.

## 4.4 QUIC: Summary (1/2)

	Reliable	Fast	Secure
<b>TCP Protocol</b>	YES	NO	NO
<b>UDP Protocol</b>	NO	YES	NO
<b>TLS Protocol</b>	YES	NO	YES
<b>DTLS Protocol</b>	NO	YES	YES
<b>QUIC Protocol</b>	YES	YES	YES

## 4.4 QUIC: Summary (2/2)

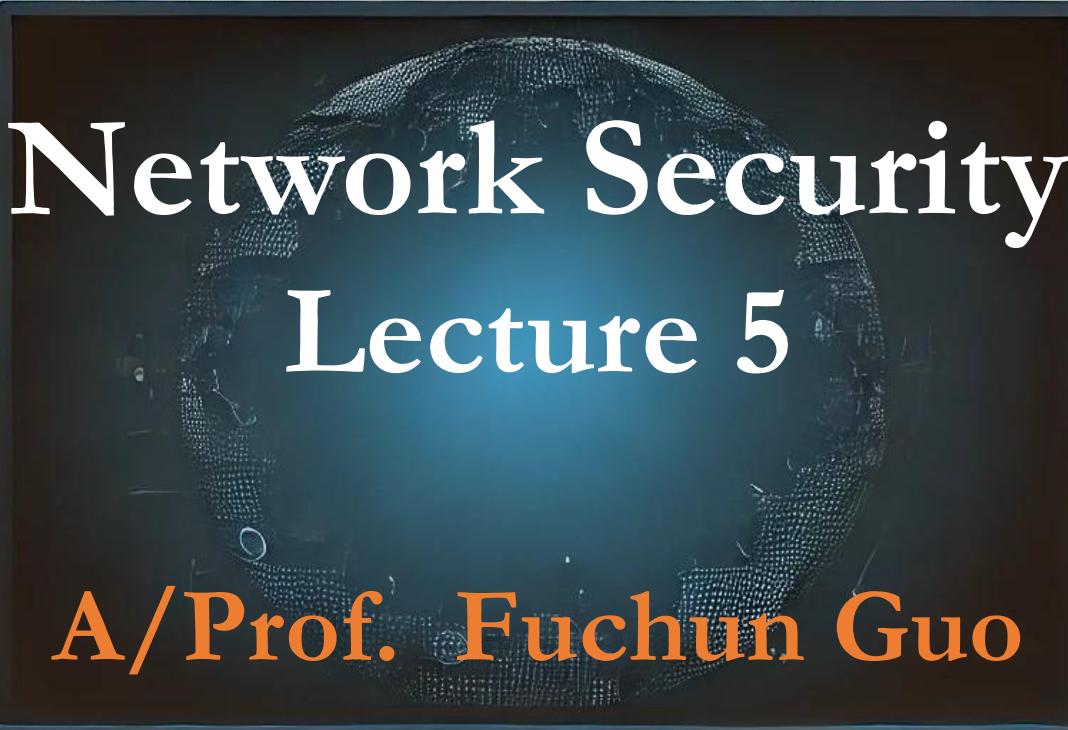


In QUIC, the ClientHello message is the first step in establishing a connection and does not inherently authenticate the client's identity. This means that the server receives a ClientHello without knowing whether the request is coming from a legitimate client or a malicious actor.

**Disadvantage:** Due to more resource in server hello. This lack of authentication can lead to scenarios where an attacker sends multiple ClientHello messages to the server (DoS )

# **END OF L4**

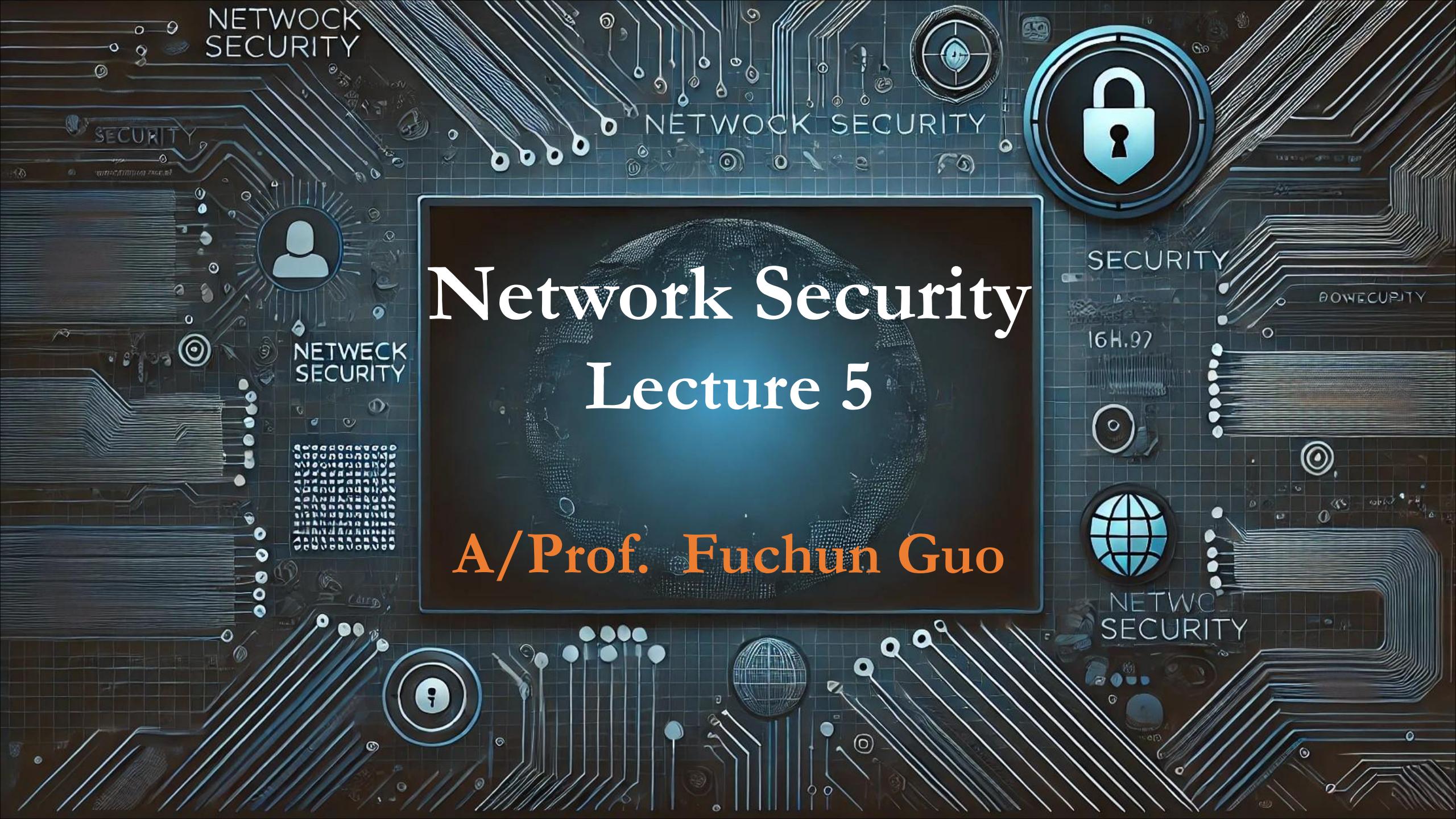
tradeoff is everywhere



# Network Security

## Lecture 5

A/Prof. Fuchun Guo



# Outline

- Internet Protocols
- Supporting Protocols
- IPsec Protocol

IPsec Tunnel

Active	<input checked="" type="radio"/> Enable <input type="radio"/> Disable
Tunnel Name	OfficeA
Type	Net-to-Net Virtual Private Network
Interface	<input checked="" type="radio"/> WAN1 <input type="radio"/> WAN2
Local Network	192.168.20.1
Local Netmask	255.255.255.0 /24
Remote Host/IP Address	109.40.78.156
Remote Network	192.168.30.1
Remote Netmask	255.255.255.0 /24

Detection

Dead Peer Detection	<input checked="" type="checkbox"/>
Time Interval	30 Seconds
Timeout	150 Seconds
Action	Restart

Authentication

Preshare Key	planet
--------------	--------

IKE Setting

Phase 1	<input checked="" type="radio"/> v1 <input type="radio"/> v2
IKE	<input checked="" type="radio"/> Main <input type="radio"/> Aggressive
Connection Type	AES (128 bit)
ISAKMP	SHA1
DH Group	2 (1024)
IKE SA Lifetime	3 hours
Phase 2	
ESP	AES (128 bit)
ESP Keylife	SHA1
Perfect Forward Secrecy (PFS)	1 hours
	<input checked="" type="radio"/> Yes <input type="radio"/> No

## 5.1 Internet Protocols

TCP or UDP have generated segments but now how to send them to another device on the internet?

# 5.1 Internet Protocols: Background (1/4)

- When a person was born, they were given a  
name
- When this person enrolls to a university, they were given a  
student number



**Student number helps manage records efficiently, reduce errors, and facilitate quick retrieval of information.**

# 5.1 Internet Protocols: Background (2/4)

- When a device is manufactured for network communications, it is assigned a

MAC address



- When this device connects to the broader internet, it is assigned an

IP address

The IP address helps facilitate communication across networks, enabling efficient routing of data, reducing the potential for errors in addressing, and ensuring that information reaches the correct destination quickly.

**Structure in management**

# 5.1 Internet Protocols: Background (3/4)

Wiki:

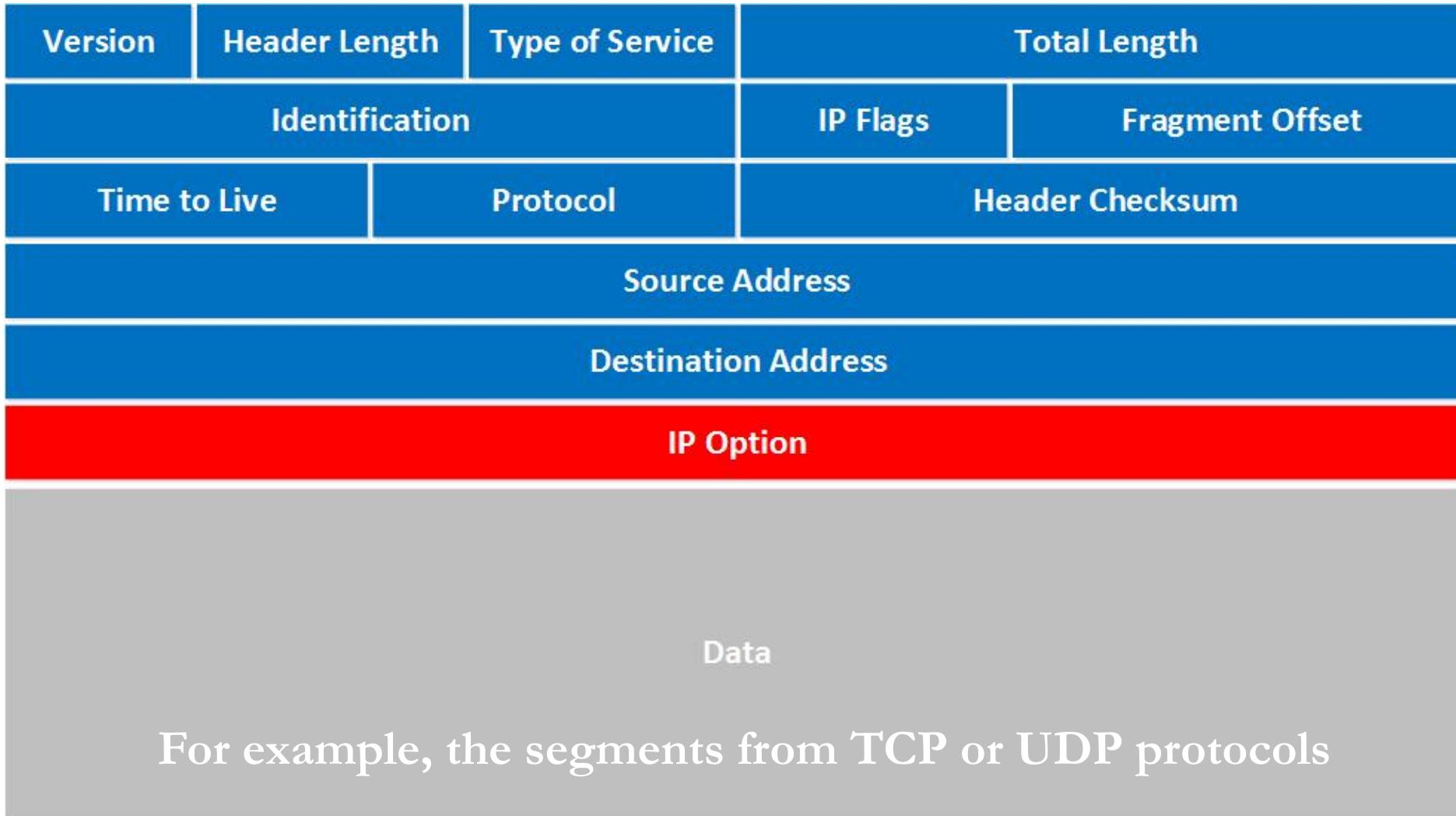
- An Internet Protocol address (IP address) is a numerical label such as 192.168.32.170 that is assigned to a device connected to a computer network that uses the **Internet Protocol** for communication. (Transmit data or payload from higher-layer protocols like UDP or TCP)
- IP addresses serve two main functions:
  - network interface identification, and
  - location addressing.



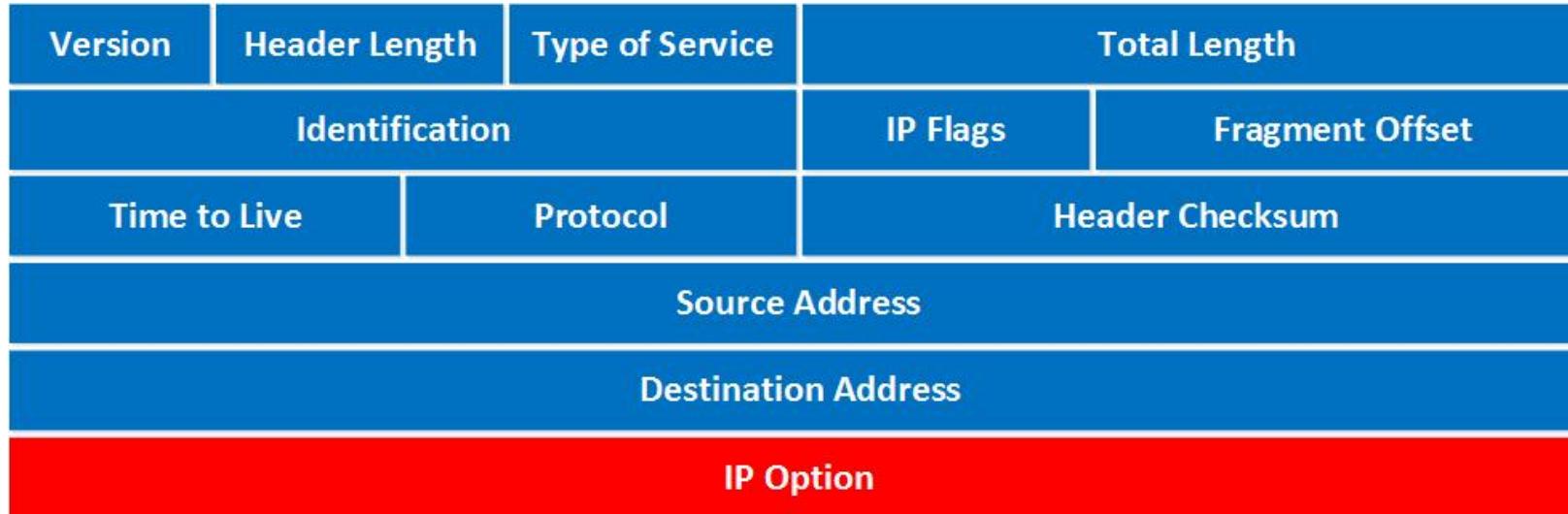
# 5.1 Internet Protocols: Background (4/4)

- IPv4 (Internet Protocol version 4) is the fourth version of the Internet Protocol (IP) and one of the core protocols that govern the way data is transmitted over the internet. It was introduced in the early 1980s and is still widely used today. IPv4 addresses are 32-bit numbers: 203.210.191.112 in decimal.
- IPv6 (Internet Protocol version 6) is the sixth version of the Internet Protocol, designed to replace IPv4 and address the limitations posed by its 32-bit address space. IPv6 was introduced in the late 1990s to provide a significantly larger address space, using 128-bit addresses formatted in hexadecimal and separated by colons (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334). This expanded address space accommodates the growing number of devices on the internet

# 5.1 Internet Protocols: IPv4 (1/7)

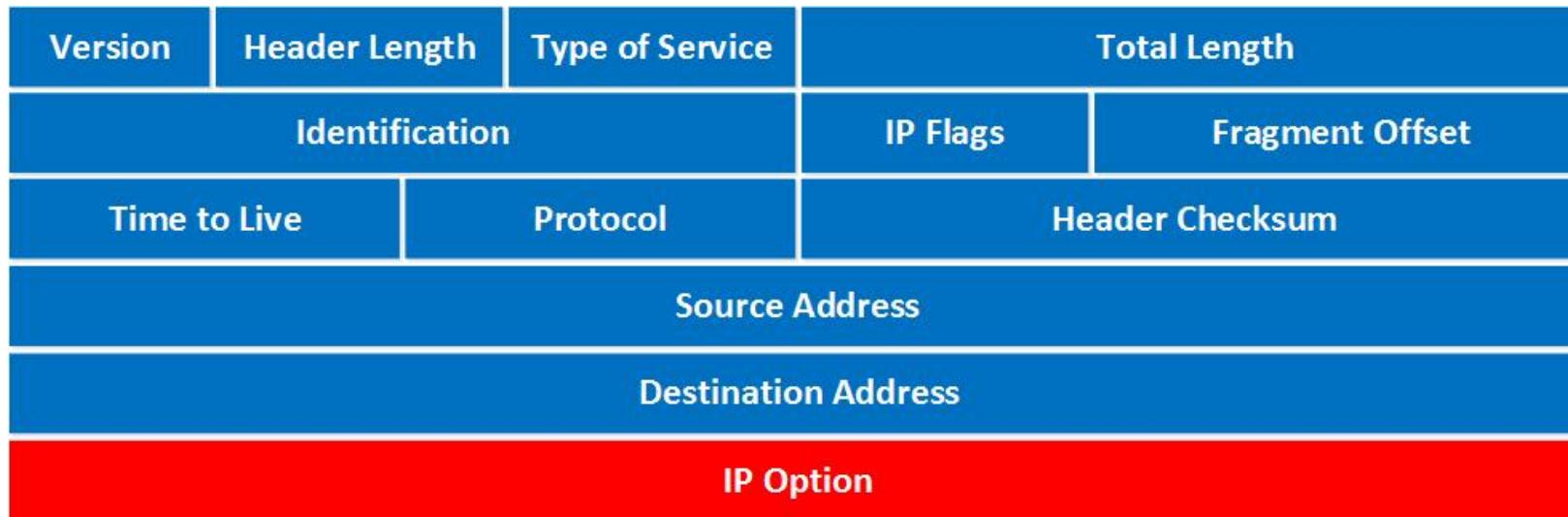


# 5.1 Internet Protocols: IPv4 (2/7)



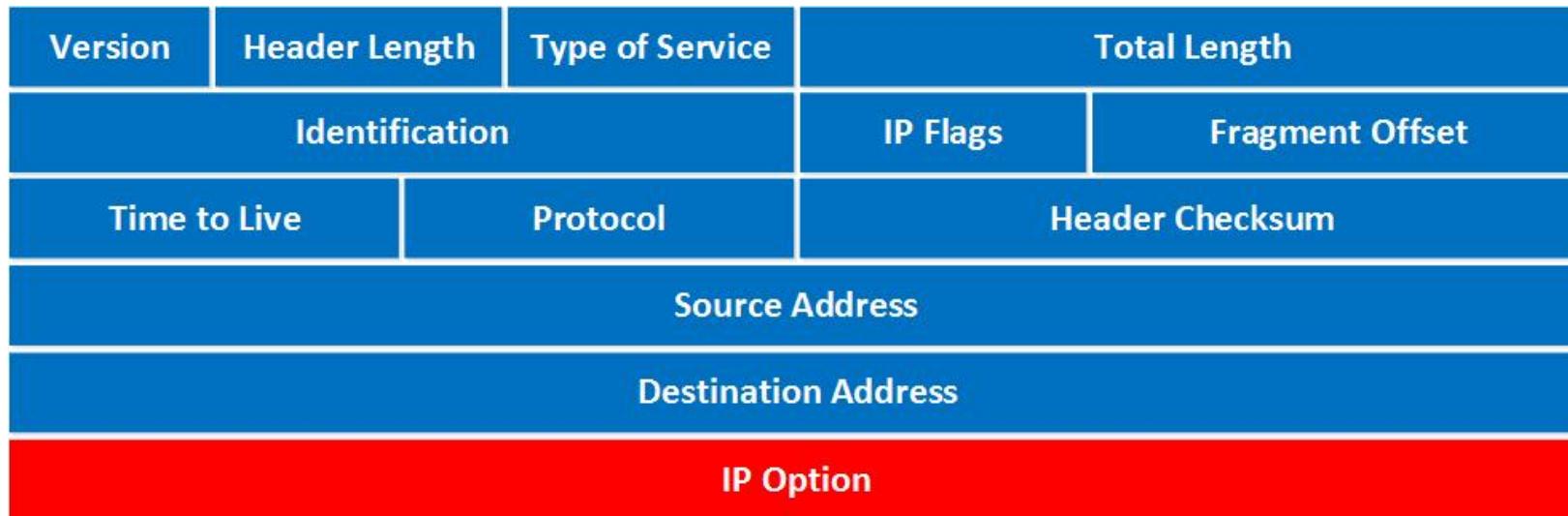
- Version (4 bits): Indicates the version of the IP protocol. For IPv4, this field always has a value of 4.
- Header Length (4 bits): Specifies the length of the header in 32-bit words (number of 32-bits). The minimum value is 5, indicating a header length of 20 bytes. If options are included, this value will be greater than 5.
- Total Length (16 bits): Specifies the entire packet size, including both the header and data, in bytes. The maximum length is 65,535 bytes.
- Identification (16 bits): This field is used for uniquely identifying the fragments of an original IP packet (same data, same ID) when fragmentation occurs. It helps reassemble the fragments at the destination.

# 5.1 Internet Protocols: IPv4 (3/7)



- **Fragment Offset** (13 bits): Specifies the position of a fragment in the original packet. This is used during reassembly to correctly order fragments.
- **Time to Live** (8 bits) limits the lifespan of a packet by decrementing with each hop; when it reaches zero, the packet is discarded to prevent infinite looping in the network. From 64 or 128.
- **Protocol** (8 bits): Indicates the transport layer protocol used in the data portion of the packet, such as TCP (value 6) or UDP (value 17). This allows the receiving device to know how to process the payload.
- **Header Checksum** (16 bits): A checksum value used for error-checking the header. It ensures that the header has not been corrupted during transmission.

# 5.1 Internet Protocols: IPv4 (4/7)



- Source Address (32 bits): The IP address of the originating device sending the packet.
- Destination Address (32 bits): The IP address of the intended recipient device.
- IP Options (variable length): This field is optional and may include additional control information (e.g., security options, timestamp, etc.). Multiple of 32 bits in length

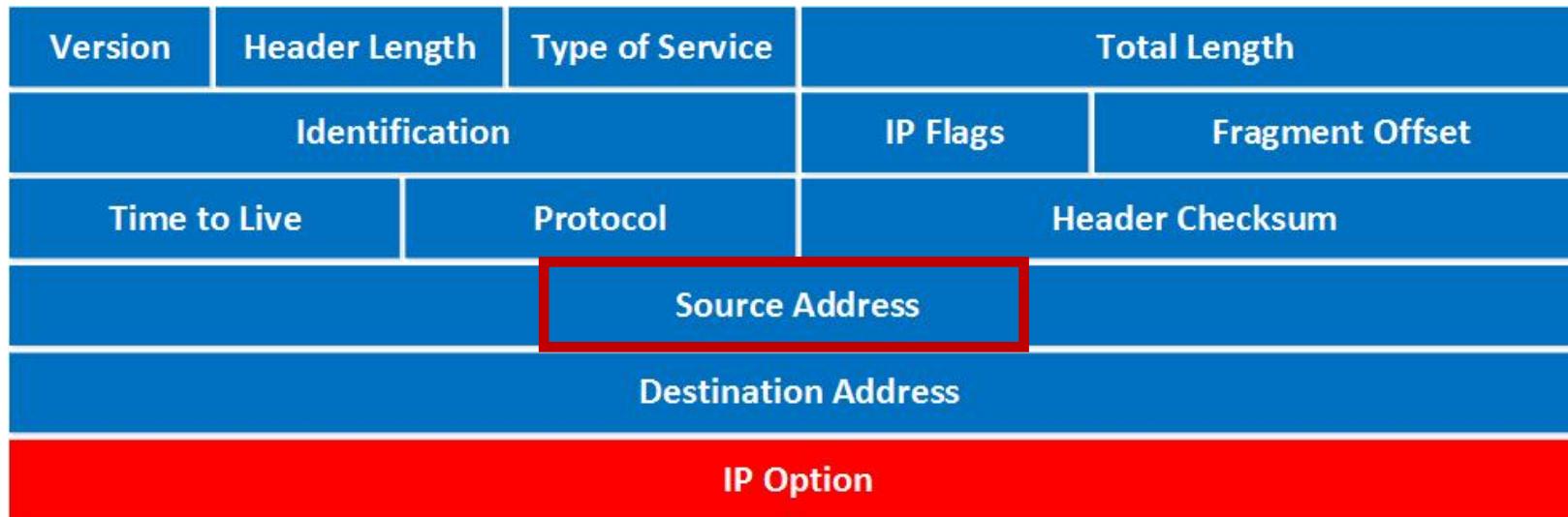
# 5.1 Internet Protocols: IPv4 (5/7)

Vulnerability Background:

- If Bob annoys Alice by calling her hundreds of times a day, Alice can simply block Bob's phone number using her smartphone's block feature.
- However, if Bob knows how to alter his caller ID, he can continue calling Alice from what appear to be different numbers. This forces Alice to answer, as the calls could be important, making it much harder for her to avoid picking up Bob's phone call.



# 5.1 Internet Protocols: IPv4 (6/7)



Address Spoofing: No authentication on source IP address

- Description: Attackers can forge the source IP address in packets, making it appear as though the packet originates from a trusted source.
- Impact: This can lead to various attacks, including Denial of Service (DoS) attacks where the attacker impersonates a legitimate user or device.

# 5.1 Internet Protocols: IPv4 (7/7)

- Step 1. Initiating the DOS Attack: The attacker crafts packets with a forged source IP address, which can be any valid IP address, including that of a legitimate user or a trusted service. The attacker then sends a high volume of these packets to the target server or network. The forged address can be a single IP address or a range of addresses.
- Step 2. Flooding the Target: Because the packets appear to come from different sources, the target server tries to respond to the requests it receives. If the forged IP addresses belong to real devices, the server will attempt to send replies to those devices, which may lead to confusion or even unintentional amplification of the attack.
- Step 3. Overwhelming Resources: The target server becomes overwhelmed with traffic as it tries to handle incoming requests and send responses. This can exhaust the server's bandwidth, CPU, or memory resources. As a result, legitimate users may experience significant delays or complete denial of access to services provided by the target.

# 5.1 Internet Protocols: IPv6 (1/4)

## IPv4

**Address Size:**  
32-bit number

**Address Format:**  
Dotted Decimal Notation:  
192.168.1.1

**Prefix Notation:**  
255.255.255.0  
/24

**Number of addresses:**  
 $2^{32} = 4,294,967,296$

## IPv6

**Address Size:**  
128-bit number

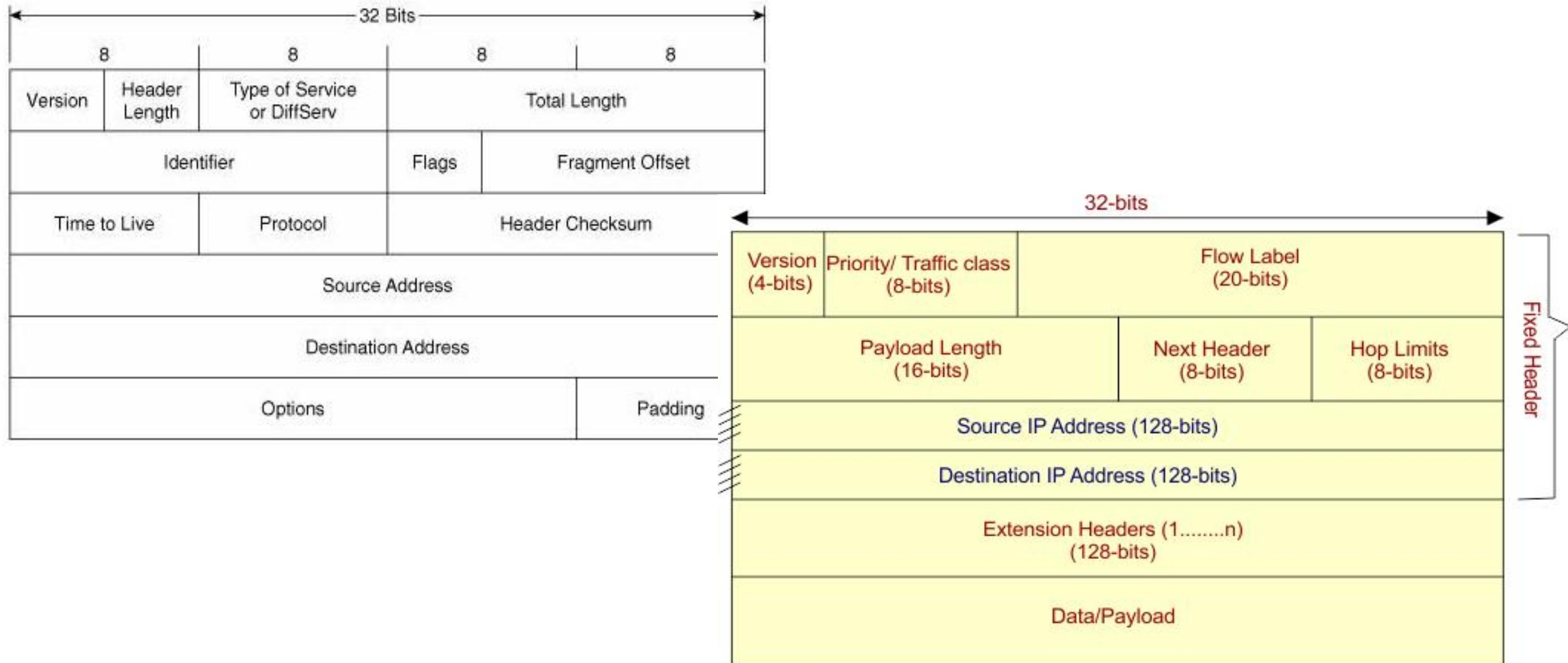
**Address Format:**  
Hexadecimal Notation:  
fe80::94db:946e:8d4e:129e

**Prefix Notation:**  
/64

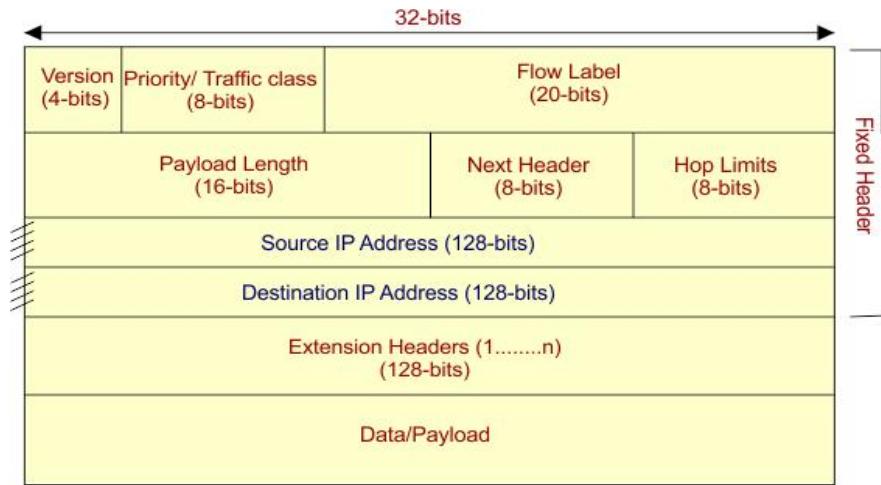
**Number of addresses:**  
 $2^{128} =$   
340,282,366,920,938,463,463,374,607,  
431,768,211,456

- The main motivation for IPv6 was due to the small number of IPv4 addresses. It became a pressing issue as the internet expanded globally to support the larger number of internet-connected devices, especially with the growth of mobile devices, IoT, and more users worldwide.
- IPv6 also provides manageable and future-proof solutions to meet the demands of the evolving internet.

# 5.1 Internet Protocols: IPv6 (2/4)



# 5.1 Internet Protocols: IPv6 (3/4)



- Simplified Header Structure: IPv6 reduces the number of fields in the base header from 14 to 8, resulting in faster processing and streamlined routing.
- Removal of Unnecessary Fields Using Extension Headers: IPv6 removes fields like fragmentation and options, which are now handled only when needed through extension headers, reducing the default header size and router workload.
- Expanded Address Fields: IPv6 uses 128-bit addresses instead of 32-bit, vastly increasing the number of available IP addresses to accommodate the growing number of devices.

# 5.1 Internet Protocols: IPv6 (4/4)

## Challenge of adopting IPv6

- Compatibility and Transition Challenges: IPv4 and IPv6 are not directly compatible, requiring special mechanisms (like dual-stack or NAT64) for communication. This complicates transitions for organizations with established IPv4 infrastructure, creating resistance to upgrading.
- Cost and Infrastructure Upgrades: Transitioning to IPv6 needs significant investments in compatible hardware, software, and network configurations. The costs and training required for network administrators add to the burden, especially for large organizations and ISPs.
- Limited Immediate Incentive: Many organizations rely on existing IPv4 workarounds, like Network Address Translation (NAT), which reduces the need for IPv6.

**Its advantages are mostly infrastructural and don't provide immediate, noticeable improvements for many users.**

## 5.2 Supporting Protocols

Suppose A and B know each party's IP address, they can run internet protocols for communications. But how to achieve “suppose”?

## 5.2 Supporting Protocols: Overview

Supporting protocols of IPv4:

- Addressing Protocols: These protocols handle the assignment and management of IP addresses. Examples include the Address Resolution Protocol (ARP), which **maps** IP addresses to MAC addresses, and the Dynamic Host Configuration Protocol (DHCP), which **dynamically** assigns IP addresses to devices on a network.
- Diagnostic Protocols: These protocols are used for **testing and troubleshooting** network connectivity and performance. Common examples include the Internet Control Message Protocol (ICMP), which is used for error messages and diagnostics (like the ping command), and the Traceroute protocol, which helps identify the path packets take through the network.
- Routing Protocols: These protocols facilitate the routing of packets between different networks (managed by different gateway). They determine the **best paths** for data to travel. Examples include Routing Information Protocol (RIP) and Border Gateway Protocol (BGP).

## 5.2 Supporting Protocols: ARP

To be introduced in the next topic!

## 5.2 Supporting Protocols: DHCP (1/4)

- A gateway is a networking device that serves as a point of access to the internet, allowing a set of devices to connect and communicate with external networks.
- It acts as an intermediary between local networks and the internet, facilitating data transfer and enabling devices to send and receive information outside their own network.



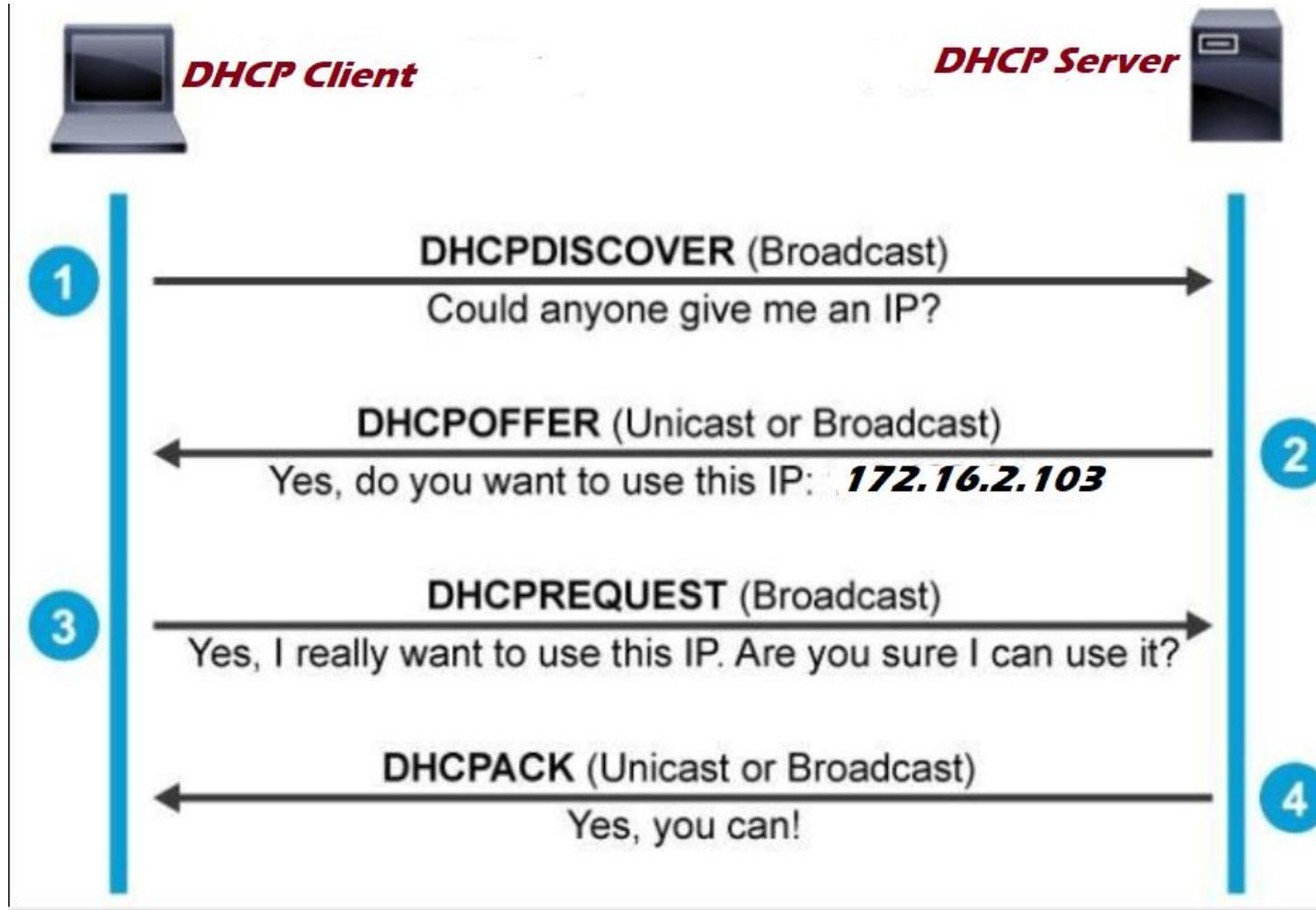
## 5.2 Supporting Protocols: DHCP (2/4)

DHCP Protocol Description:

- When a device connects to the network, it sends a DHCP Discover message to find available DHCP servers.
- The DHCP server responds with a DHCP Offer, which includes an available IP address and other configuration information.
- The client then sends a DHCP Request message to accept the offer, and the server sends a DHCP Acknowledgment to confirm the assignment.

**Benefits:** This dynamic allocation helps manage IP address space efficiently, as devices can receive different IP addresses each time they connect, and unused addresses can be reassigned to other devices.

## 5.2 Supporting Protocols: DHCP (3/4)



## 5.2 Supporting Protocols: DHCP (4/4)

**Vulnerability:** DHCP spoofing is a type of network attack where an unauthorized (or rogue) DHCP server is set up by an attacker on a network. This rogue server can respond to DHCP requests from clients instead of the legitimate DHCP server.

- Client Request: When a device (client) connects to a network, it sends a DHCP Discover message to locate available DHCP servers.
- Rogue Server Response: The attacker's rogue DHCP server responds with a DHCP Offer that includes a malicious or incorrect IP address and network configuration settings (such as the default gateway and DNS server).
- Client Acceptance: The client accepts the offer and configures itself with the settings provided by the rogue server, believing it to be legitimate.

**Potential Consequences:** Denial of Service (DoS), Network Interception

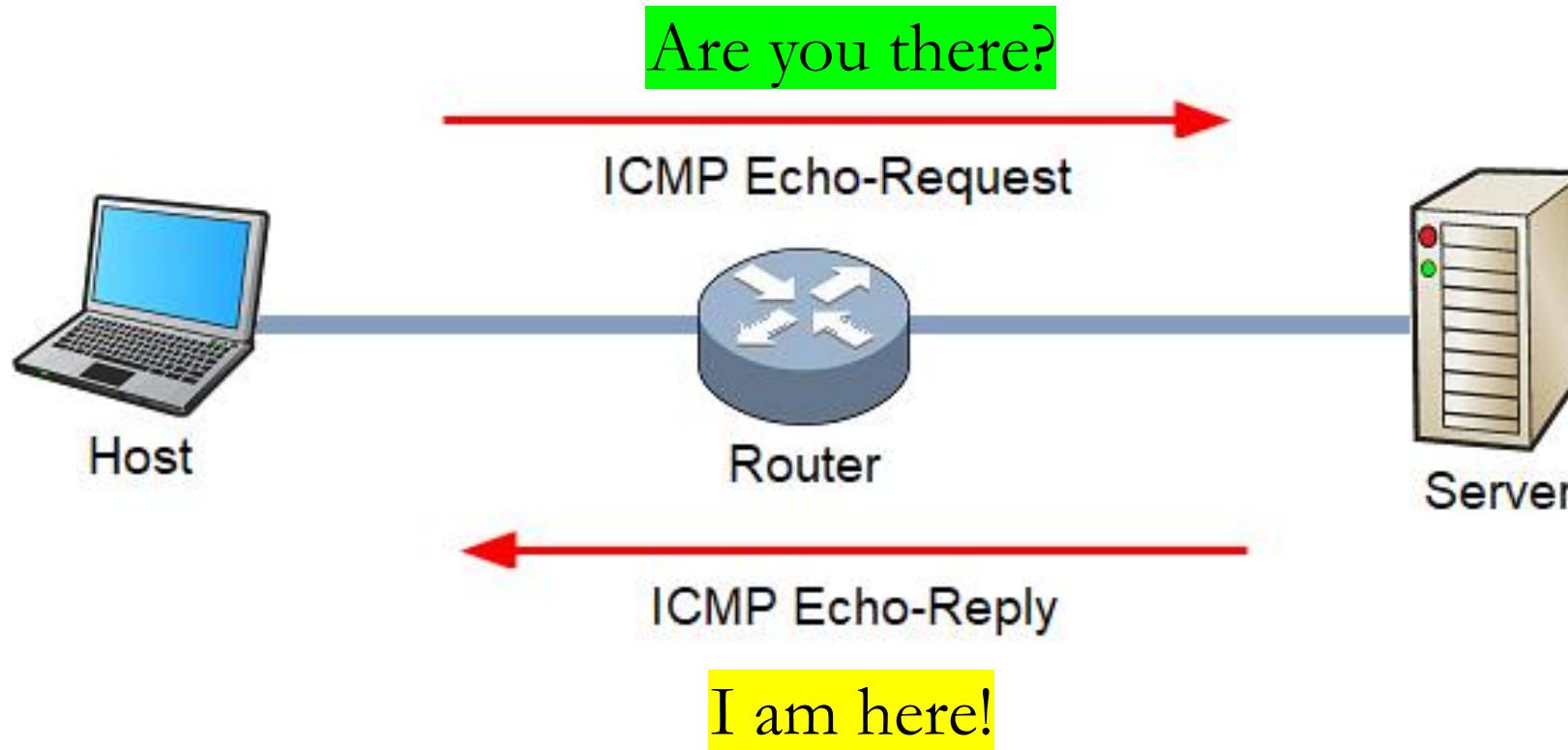
## 5.2 Supporting Protocols: ICMP (1/4)

- ICMP (Internet Control Message Protocol) is a core protocol of the Internet Protocol Suite, primarily used for sending error messages and operational information. It operates at the network layer and is essential for network diagnostics and management.
- Key Functions of ICMP
  - Error Reporting: ICMP communicates issues like unreachable destinations, time exceeded for packet delivery, and other network errors.
  - Network Diagnostics: It provides tools for network troubleshooting and performance monitoring, allowing administrators to assess network health.

## 5.2 Supporting Protocols: ICMP (2/4)

- One of the most well-known functions of ICMP is the ping utility, which is used to test the reachability of a host on a network.
- How Ping Works: When a user issues a ping command, the device sends an ICMP Echo Request message to the target IP address. If the target device is reachable and operational, it responds with an ICMP Echo Reply message.
- Purpose of Ping:
  - Reachability Testing: Ping checks whether a particular IP address is active and responsive.
  - Round-Trip Time Measurement: It measures the time it takes for the Echo Request to reach the target and for the Echo Reply to return, providing an indication of network latency.

## 5.2 Supporting Protocols: ICMP (3/4)



The ping protocol requires the server to unconditionally answer messages --> attack!

## 5.2 Supporting Protocols: ICMP (4/4)

### ICMP Flooding (Denial-of-Service Attack):

- Description: Attackers send a high volume of ICMP Echo Request (ping) packets to overwhelm a target system's resources, leading to degraded performance or complete unresponsiveness.
- Impact: This can cause significant network disruption, making services unavailable and impacting overall network stability.

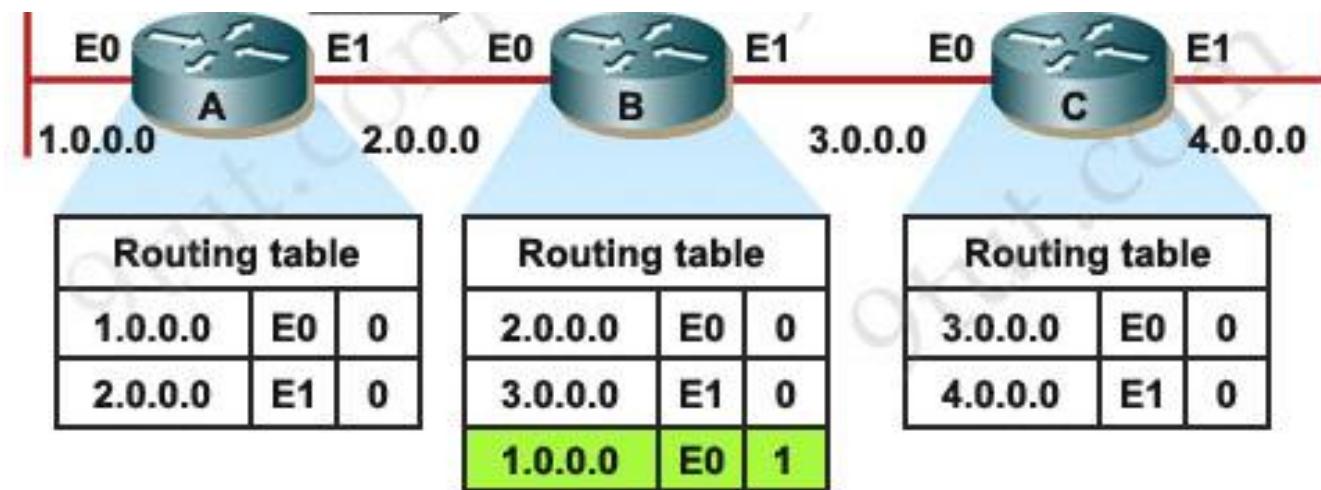
### Ping of Death:

- Description: Exploits vulnerabilities in a system's handling of oversized ICMP packets, which can cause (old systems) crashes (Lack of Size Checking).
- Impact: This can lead to system crashes or reboots, affecting the availability and reliability of network services.

## 5.2 Supporting Protocols: RIP (1/4)

RIP (Routing Information Protocol): The RIP is a dynamic routing protocol that allows routers to

- exchange information about their connected networks and
- find optimal paths to reach different destinations.



## 5.2 Supporting Protocols: RIP (2/4)

Router A<==> B <==> C are in a network, each connected to different subnetworks.

- Router A advertises its routing table to Router B, which contains a route to its connected network with a hop count (unit of distance) of 1.
- Router B receives this update, adds 1 to the hop count, and updates its table if this is a better route.
- Router B then advertises its updated table to neighbours Router C and Router A. Router A may receive updates about networks it doesn't directly connect to.
- If any route is lost (e.g., Router B loses connection with Router D for example), Router B sends a triggered update marking the route as unreachable (with a hop count of 16), allowing other routers to update their tables promptly.

## 5.2 Supporting Protocols: RIP (3/4)

RIP is vulnerable to several attacks due to its simplicity and lack of robust authentication:

### 1. RIP Spoofing *Attack* (Route Injection)

How it works: In a RIP spoofing attack, an attacker sends fake RIP update messages to a router, claiming to have a route to certain networks with a low hop count. This misleads the router into using the attacker's route to reach those destinations.

Impact: The router **redirects traffic through the attacker**, enabling a Man-in-the-Middle (MitM) attack where the attacker can intercept, modify, or discard the traffic. This attack can result in data breaches, loss of confidentiality, and potential service disruptions.

## 5.2 Supporting Protocols: RIP (4/4)

### 2. Route Poisoning *Attack*

How it works: In route poisoning, the attacker sends RIP messages with a hop count of 16 (indicating that a route is unreachable) for specific networks. This tricks routers into marking legitimate routes as invalid.

Impact: This leads to denial of service (DoS) on affected networks, as routers will remove these routes from their tables, blocking legitimate traffic from reaching its intended destination.

## 5.2 Supporting Protocols: BGP (1/3)

BGP is another routing protocol.

- BGP (Border Gateway Protocol): Designed for large-scale inter-network routing, especially between Autonomous Systems (AS) on the internet. It is primarily used by ISPs and large organizations to manage routes across different administrative domains.
- RIP (Routing Information Protocol): Created for smaller, internal networks (Local Area Networks, or LANs) such as within one AS by one ISP. RIP is simple and used in smaller networks where complex routing policies and scalability aren't a priority.

## 5.2 Supporting Protocols: BGP (2/3)

Similarities (BGP VS RIP) in the Process

Route Advertisement: Like other protocols, BGP routers advertise their routing tables to neighboring routers.

Table Updates: Each router evaluates received routes, updates its routing table based on the received information, and propagates this information to its neighbors.

Handling Route Changes: When routes become unreachable, BGP routers notify their peers to ensure all routers maintain accurate and up-to-date routing information.

**Key Difference**: Path-Vector Mechanism in BGP. This mechanism allows it to function effectively in larger, more complex networks, particularly for inter-domain routing on the internet.

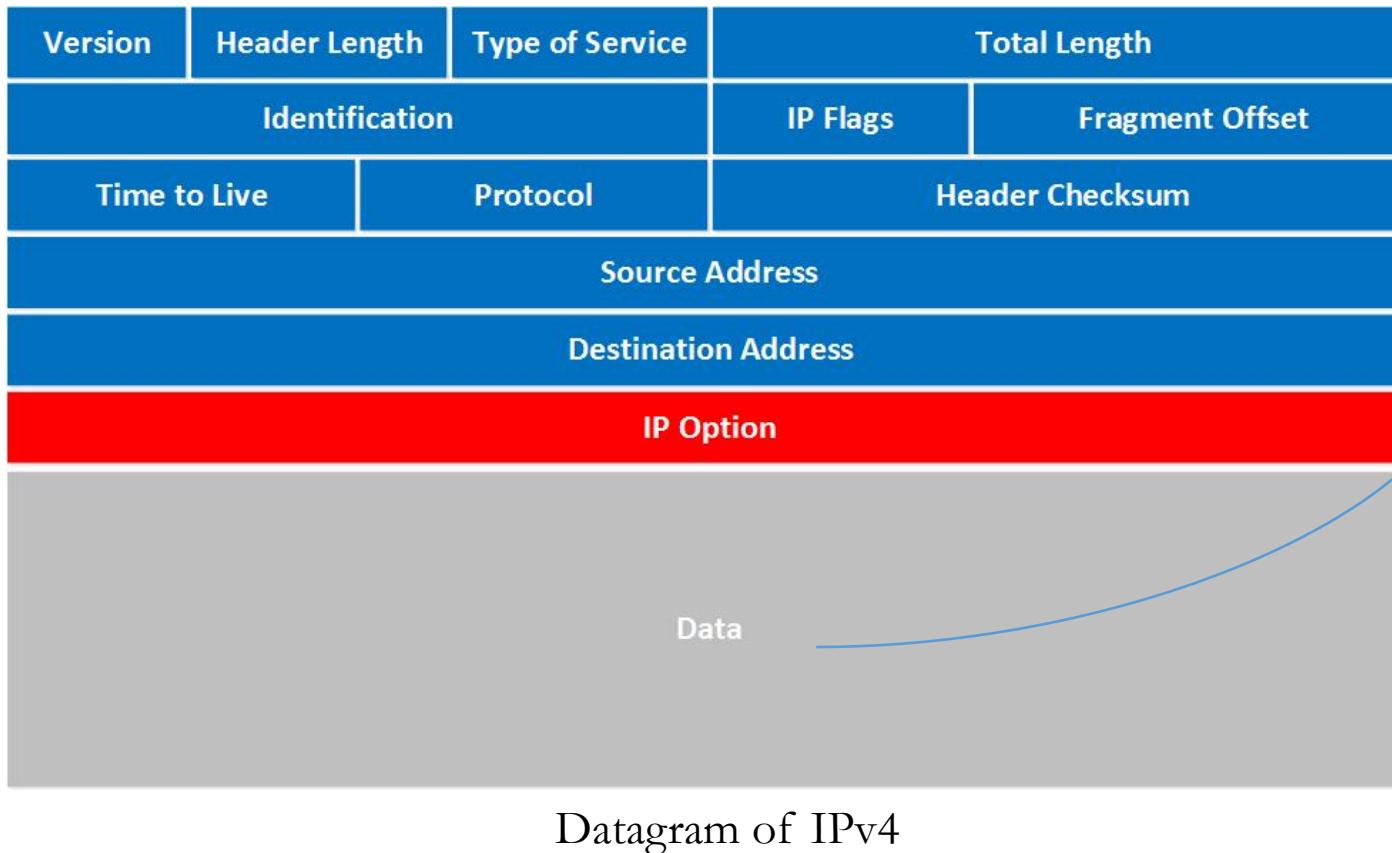
## 5.2 Supporting Protocols: BGP (3/3)

- Session Hijacking specifically refers to the unauthorized takeover of a session between a user and a server from a different network. This takeover can happen during a MitM attack when an attacker captures session identifiers (like cookies or tokens) while intercepting the communication between the user and the server.
- Session hijacking can occur through BGP vulnerabilities because BGP operates on a trust-based model, allowing routers to accept route updates without strict authentication. An attacker (ISP for example) can hijack routes by advertising fake paths for IP prefixes they don't own, redirecting traffic through their own router. This enables the attacker to intercept sensitive information, such as session tokens, which can then be used to take over legitimate user sessions.

## 5.3 IPsec

How to secure internet protocols?

# 5.3 IPsec: Background (1/7)



**Data (Payload)**

- could be saw
- could be modified

## 5.3 IPsec: Background (2/7)

- IPsec (Internet Protocol Security) is a suite of protocols designed to secure Internet Protocol (IP) communications by providing authentication, integrity, and confidentiality at the network layer.
- The development of IPsec began in the early 1990s and was standardized in 1998. IPsec gained widespread adoption in the late 1990s and early 2000s, becoming a fundamental technology for Virtual Private Networks (VPNs) and secure site-to-site communications.
- For VPN, IPsec's ability to secure all applications with a single configuration is a significant advantage over TLS, which requires separate setups for each application.

## 5.3 IPsec: Background (3/7)

Two communication modes by IPsec:

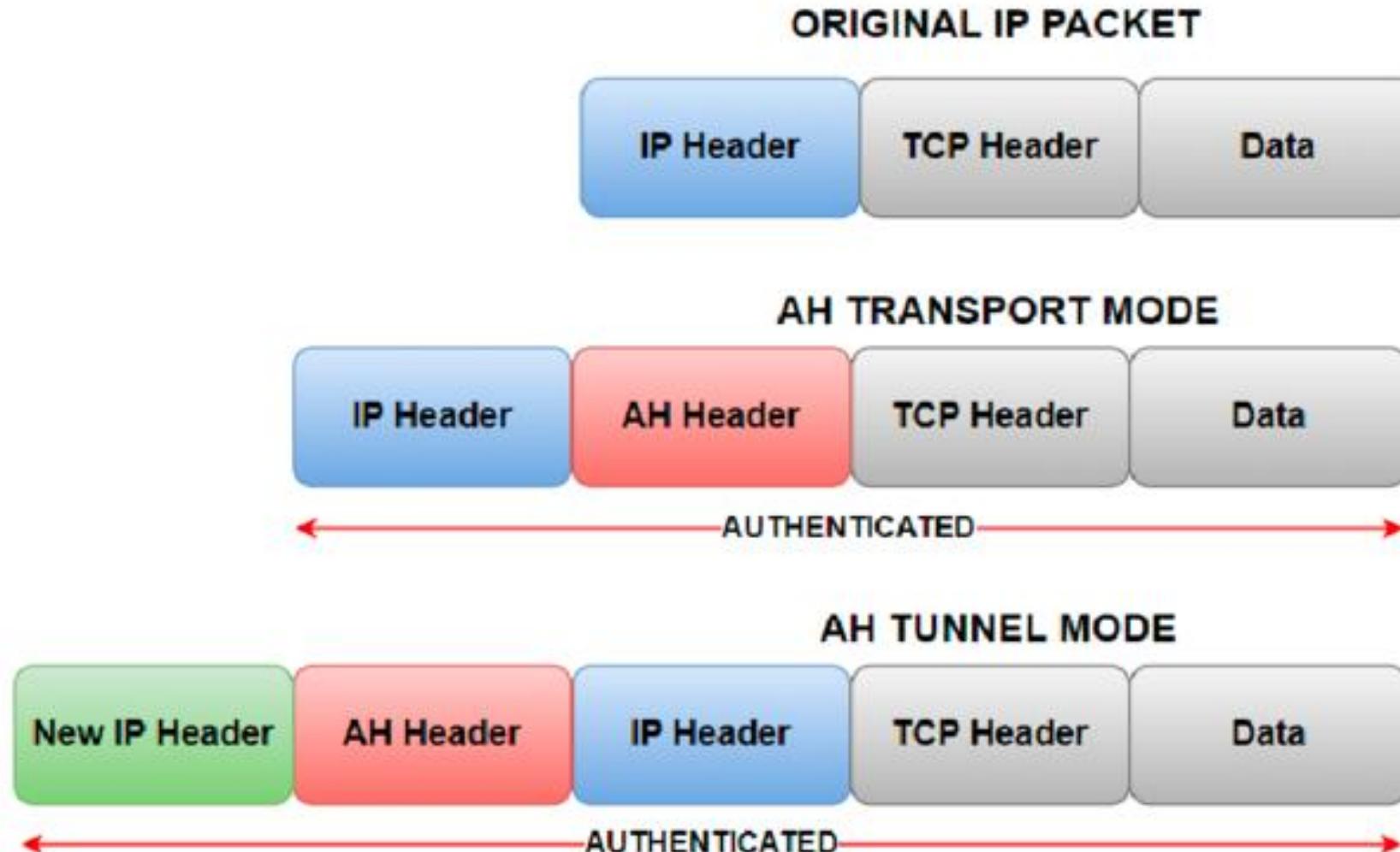
- **Transport Mode.** In this mode, only the payload (data) of the original IP packet is encrypted or authenticated. The original IP header remains intact. This mode is often used for end-to-end communication between two hosts, where both endpoints support IPsec. (A sends to B)
- **Tunnel Mode.** In this mode, the entire original IP packet (including both the header and the payload) is encapsulated within a new IP packet. A new IP header is added for routing. This mode is commonly used for Virtual Private Networks (VPNs), especially in site-to-site configurations, where two networks are securely connected over an untrusted network like the Internet. (A sends to C that is forwarded to B)

## 5.3 IPsec: Background (4/7)

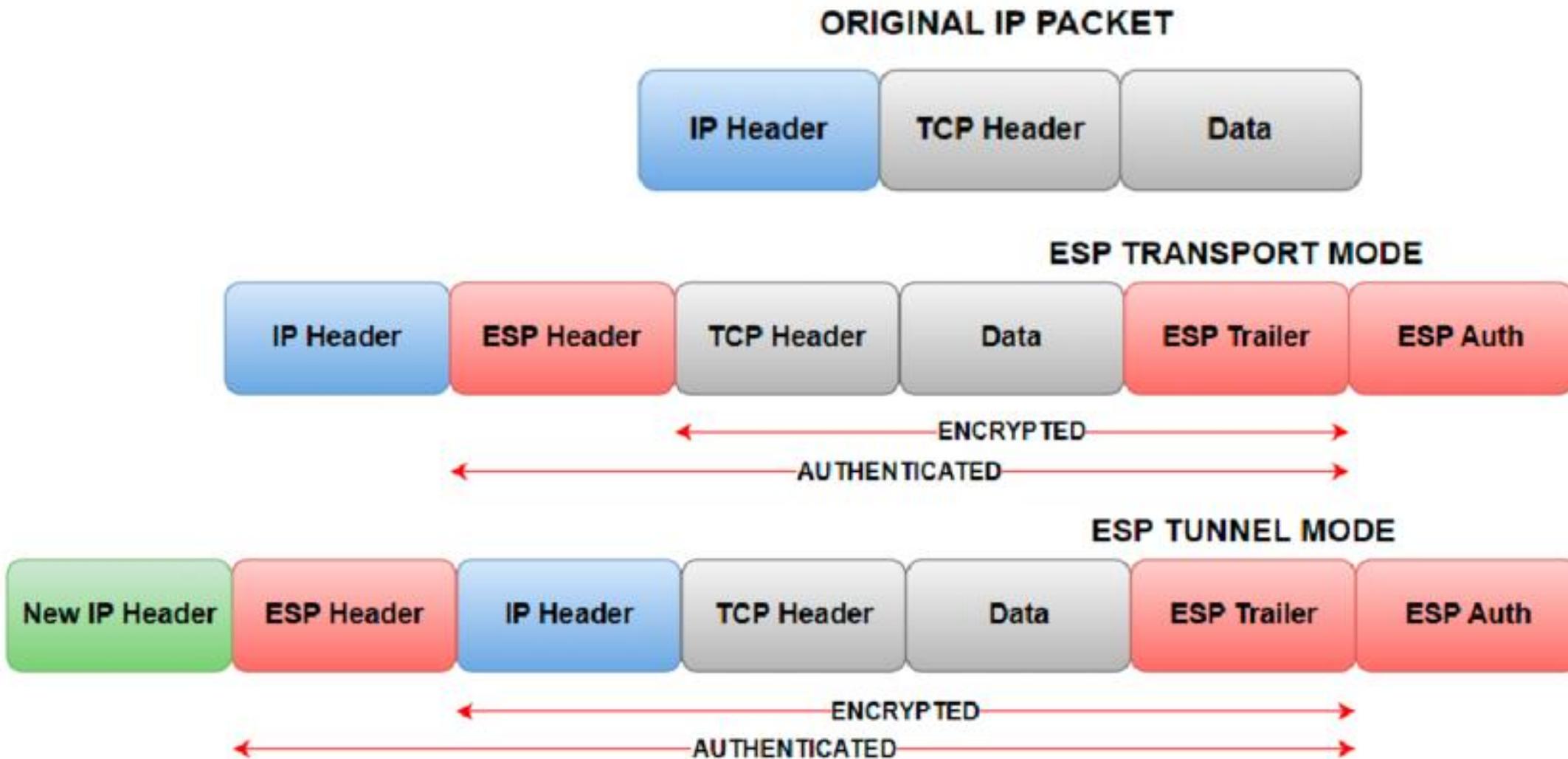
Two different security protections from IPsec:

- **Authentication Header (AH):** This component provides integrity and authentication for the data, ensuring the packet hasn't been tampered with and verifying the sender's identity. AH does not provide encryption, so it's useful for scenarios where data confidentiality isn't required. integrity
- **Encapsulating Security Payload (ESP):** ESP offers encryption for data confidentiality, along with optional integrity and authentication. ESP is typically used for scenarios that require secure, private communication, as it protects both the payload and offers optional tamper-proofing. confidentiality+ integrity

## 5.3 IPsec: Background (5/7)



## 5.3 IPsec: Background (6/7)

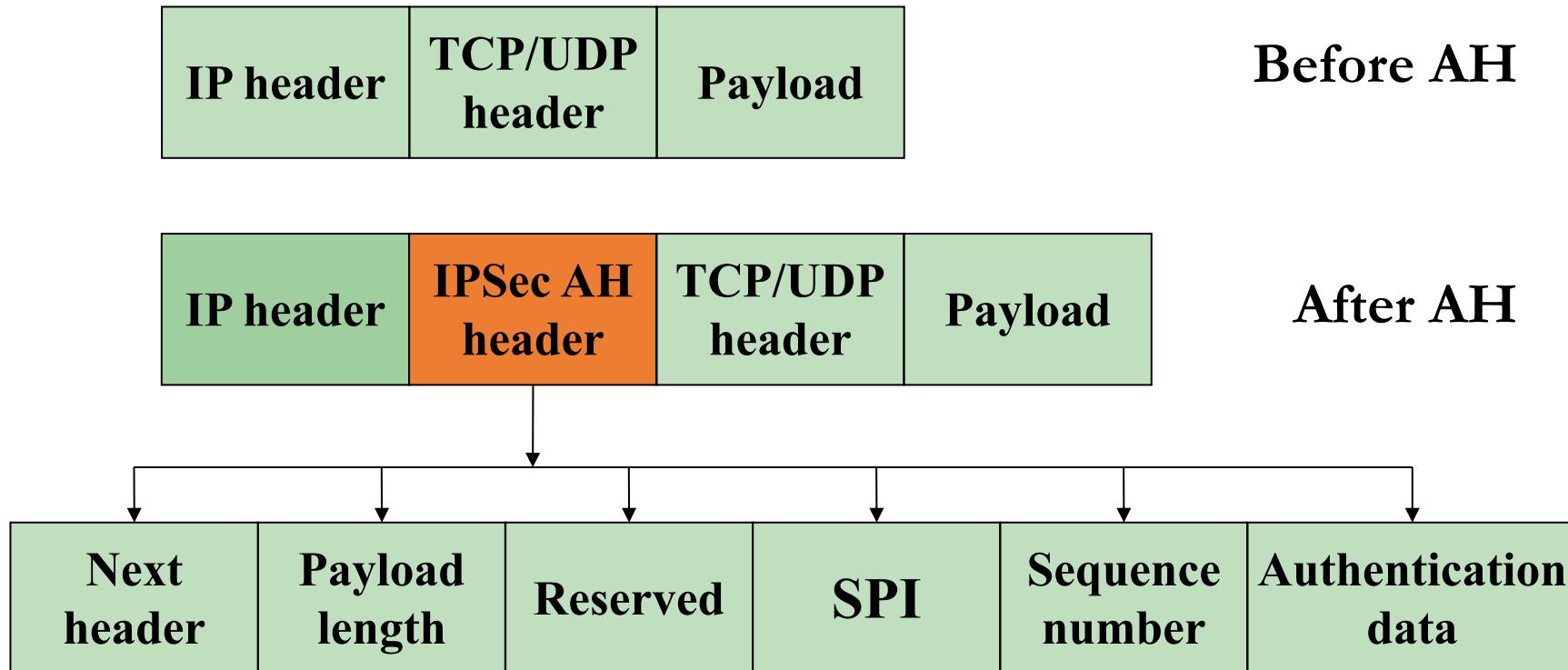


## 5.3 IPsec: Background (7/7)

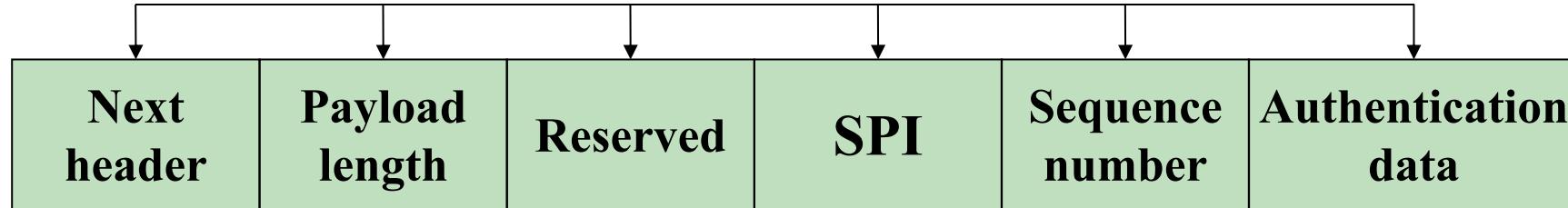
Two main components in IPsec:

- Internet Key Exchange (IKE): IKE is a protocol for establishing Security Associations and managing keys. It automates the process of negotiating and setting up SAs, which include choosing encryption methods, exchanging keys securely, and refreshing keys periodically to maintain security.
- Security Associations (SA): The SA is a set of policies and keys established between two IPSec endpoints to define how traffic will be handled. Each SA is unidirectional and determines the specific security protocols (AH or ESP), encryption and authentication methods, and keys to be used for that connection.

# 5.3 IPsec: AH&Transport (1/3)

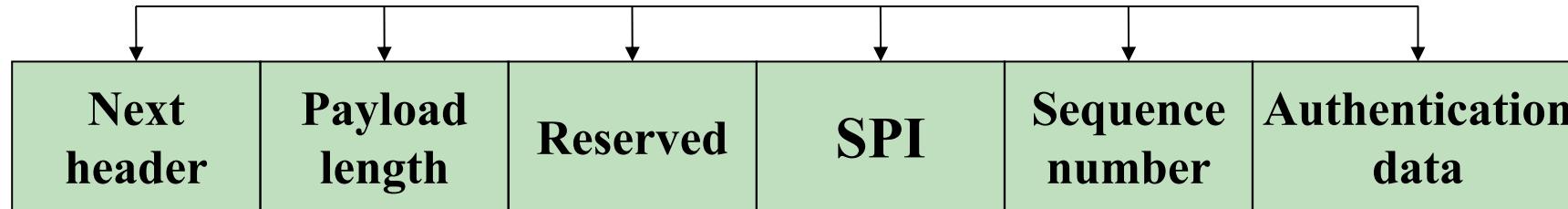


## 5.3 IPsec: AH&Transport (2/3)



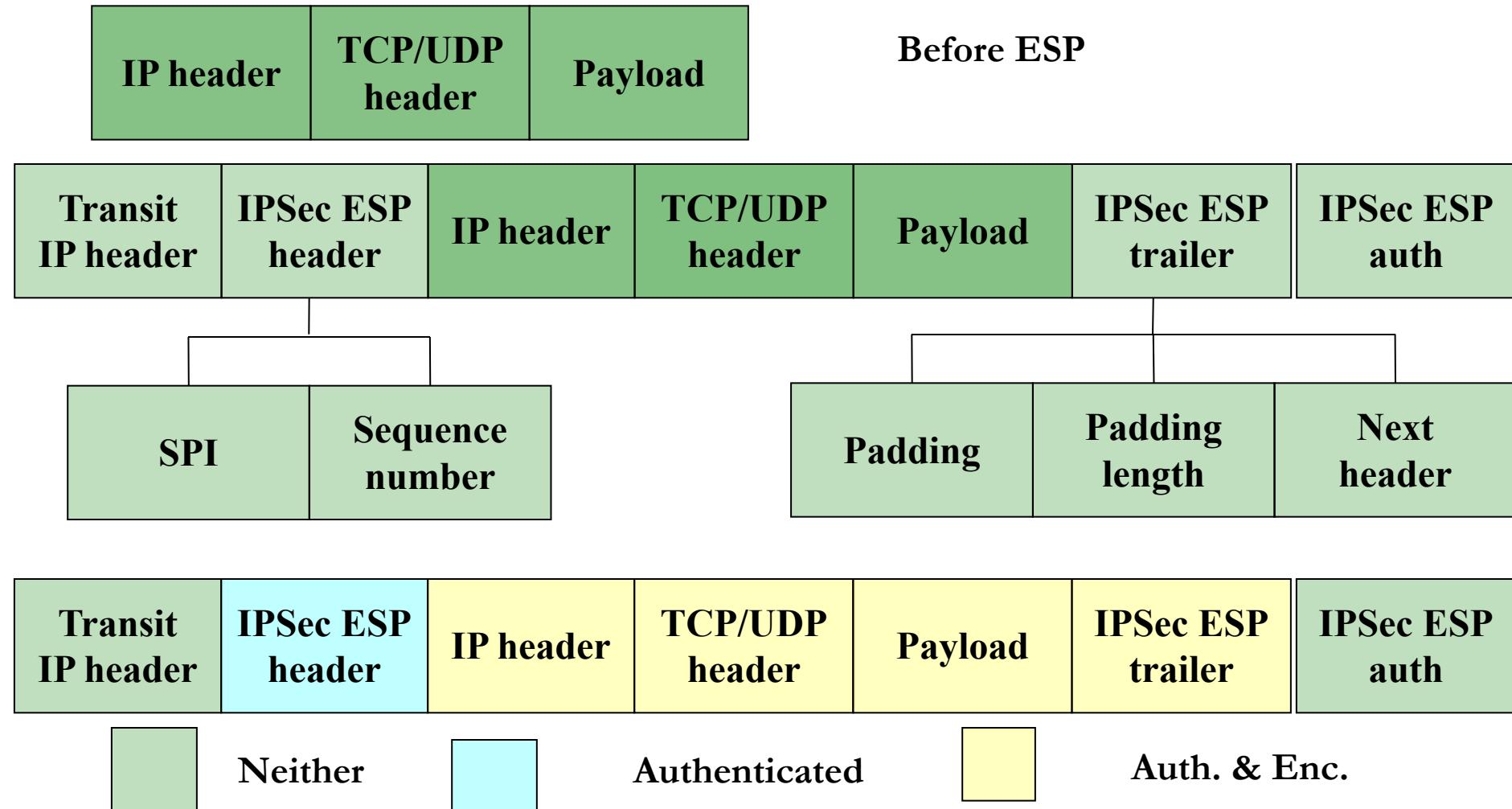
1. Next Header (8 bits): Indicates the type of payload protocol following the AH header (e.g., TCP, UDP). It points to the protocol type of the encapsulated data.
2. Payload Length (8 bits): Specifies the length of the AH header in 32-bit words (including the fields in the header). It helps in identifying where the AH header ends and the payload begins.
3. Reserved (16 bits): This field is reserved for future use and is always set to zero.

## 5.3 IPsec: AH&Transport (3/3)

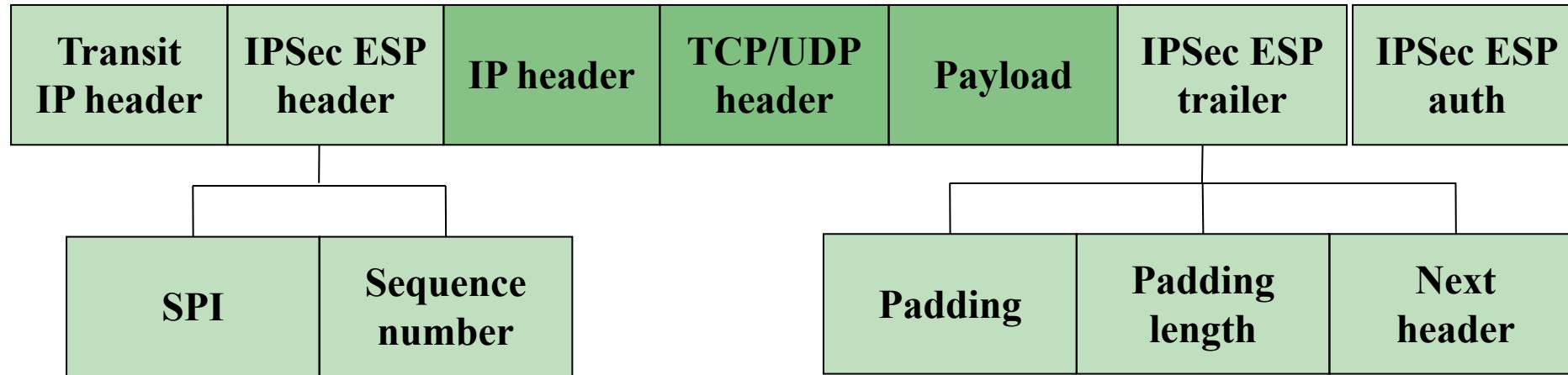


4. Security Parameters Index (SPI) (32 bits): A unique identifier that points to the Security Association (SA) being used. It helps the receiving party know which SA and related keys to use for authentication.
5. Sequence Number (32 bits): A counter that increases with each transmitted packet. It provides replay protection by ensuring that old packets cannot be resent or reordered by attackers.
6. Authentication Data (variable length): This field is used by the receiver to verify the authenticity and integrity of the packet. (part of IP header +AH header + Payload)

# 5.3 IPsec: ESP&Tunnel (1/3)

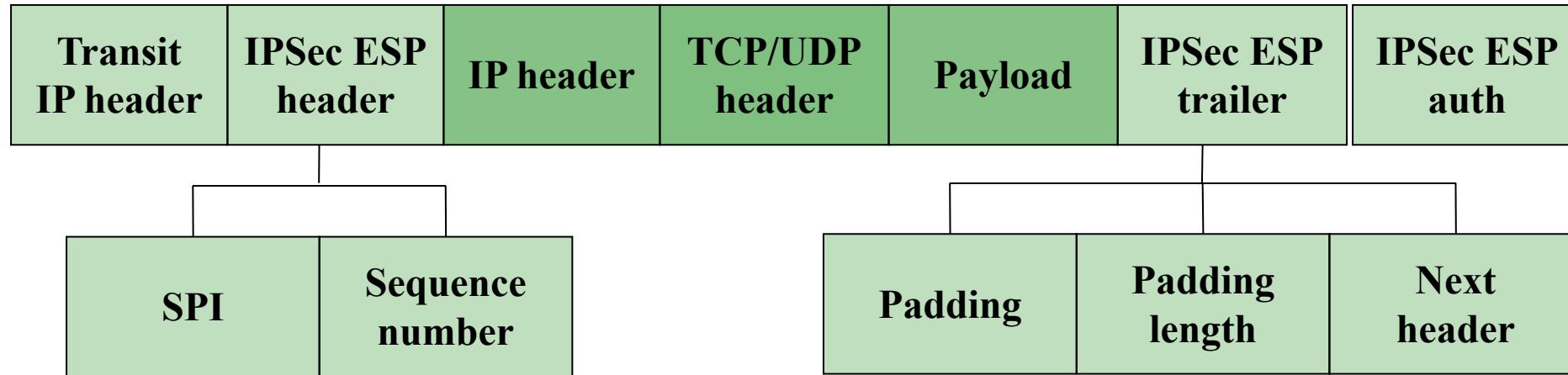


## 5.3 IPsec: ESP&Tunnel (2/3)



- Security Parameters Index (SPI) (32 bits): Identifies the Security Association (SA) used for this ESP packet. This value points to the specific cryptographic keys and algorithms agreed upon for securing the data.
- Sequence Number (32 bits): A counter that increments with each packet sent under a particular SA. It provides replay protection by ensuring that packets can't be reused or reordered by attackers.

## 5.3 IPsec: ESP&Tunnel (3/3)

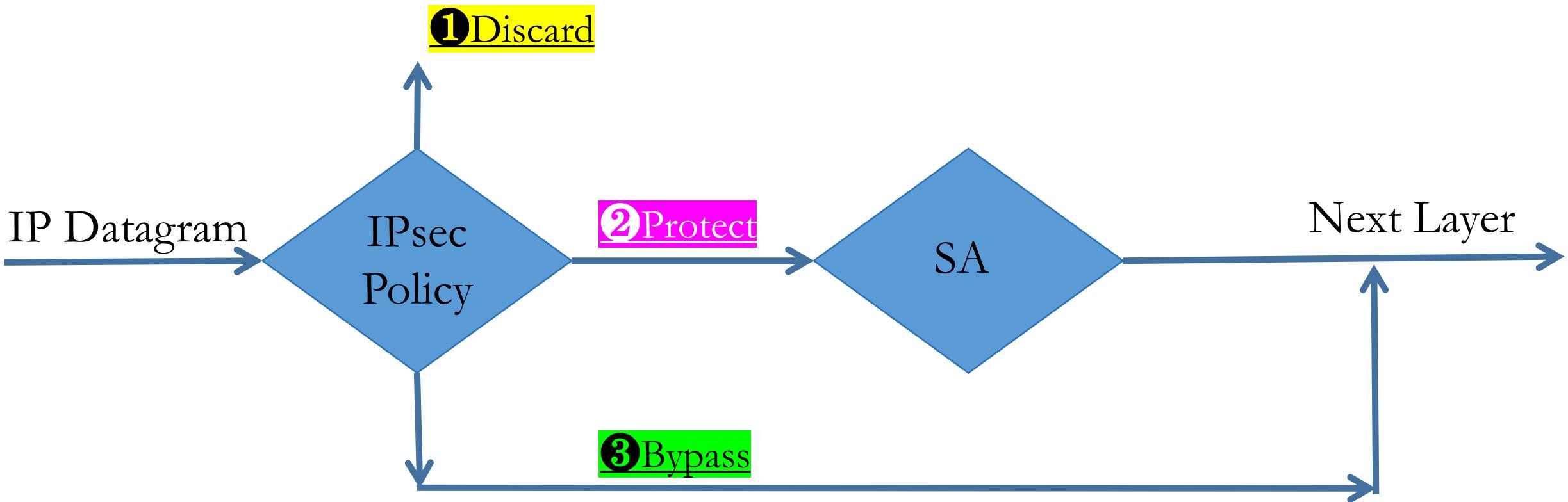


- Padding and Pad Length (variable length): Padding is used to align the data to a multiple of the encryption algorithm's block size. The Pad Length field specifies the number of padding bytes added.
- Next Header (8 bits): Indicates the type of protocol within the payload data, such as TCP, UDP, or ICMP.
- Authentication Data (variable length, optional): It covers all immutable fields.

## 5.3 IPsec: SA (1/4)

- IPsec follows a policy-based approach to enforce the local security decisions of a system.
- On input Requirements from administrator, it generates IPsec Policy.
  - Which traffic requires IPsec protection (e.g., IP addresses, or ports).
  - Which IPsec mode to use (Transport or Tunnel).
  - Whether to use ESP or AH for encryption or authentication.
- Essentially, policies are like filters and action plans that decide which packets should trigger IPsec and what type of IPsec protection is required for those packets.

## 5.3 IPsec: SA (2/4)

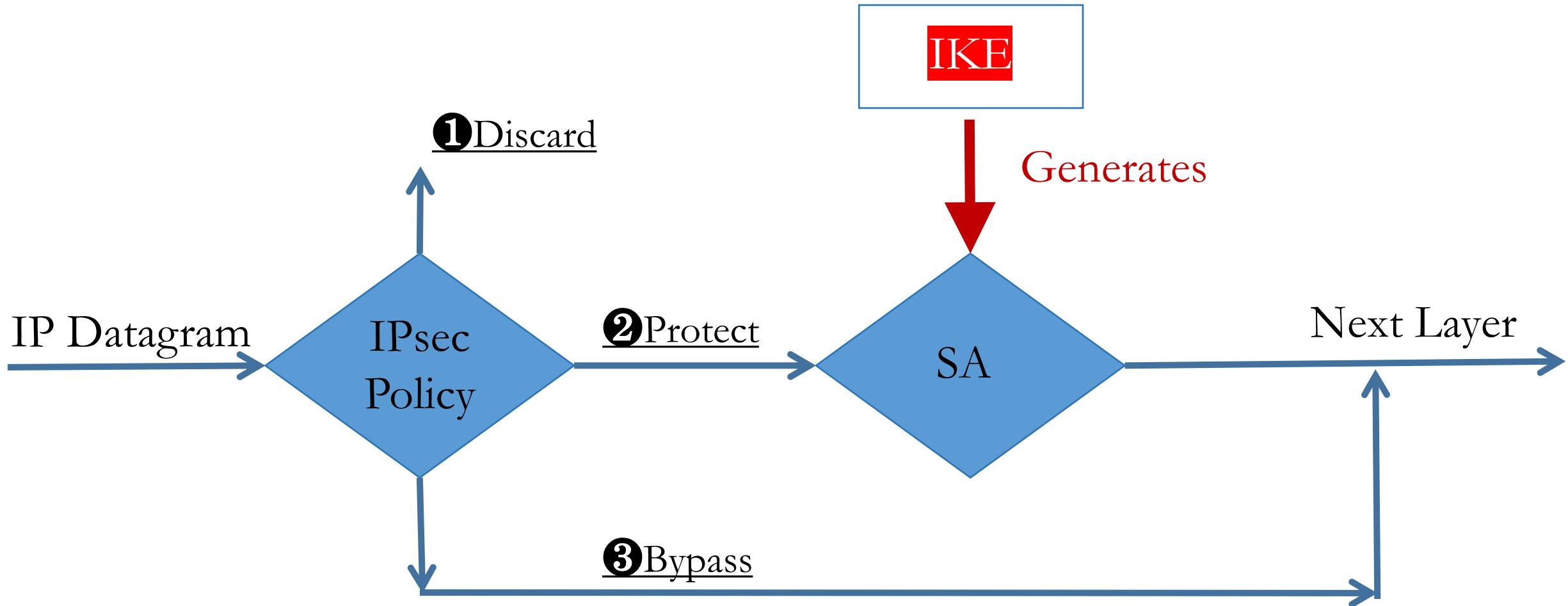


- Discard: Blocks traffic that doesn't meet policy requirements, deleting it.
- Protect: Secures matching traffic using encryption, authentication, or integrity checks as specified.
- Bypass: Allows trusted traffic to pass without IPsec protection, optimizing performance.

## 5.3 IPsec: SA (3/4)

- A Security Association (SA) in IPsec is a set of parameters and keys that define how traffic will be secured for a specific data flow, both for outgoing traffic and incoming traffic under the Protect decision.
- Components:
  - SPI (Security Parameters Index): A unique identifier for the SA, helping distinguish it among multiple SAs.
  - Cryptographic Algorithms: Specifies the encryption (e.g., AES) and authentication (e.g., SHA-256) algorithms to be used.
  - Keys: Contains encryption and authentication keys generated through protocols like IKE (Internet Key Exchange).
  - Lifetime: Determines how long the SA remains valid before rekeying or re-negotiating.

## 5.3 IPsec: SA (4/4)



## 5.3 IPsec: IKE (1/5)

- Internet Key Exchange (IKE) protocol is responsible for establishing and negotiating the parameters for SAs between two endpoints, including selecting cryptographic algorithms, generating shared keys, and setting lifetimes.
- Phase 1(Secure channel): IKE first establishes a secure communication channel between the two endpoints by authenticating them and setting up an initial secure-channel SA for IKE itself. This phase ensures that both sides are legitimate and agree on the base security settings.
- Phase 2(Negotiation): Using the secure IKE channel from Phase 1, IKE negotiates IPsec SAs for actual data protection. During this phase, IKE defines the cryptographic parameters and keys used by IPsec to secure data flow according to the agreed Protect policies.

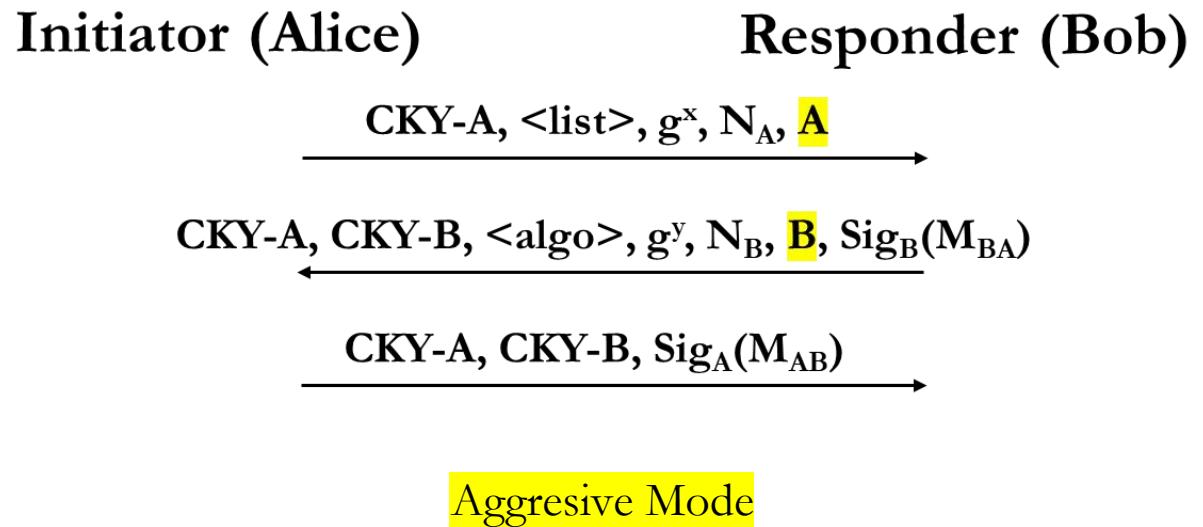
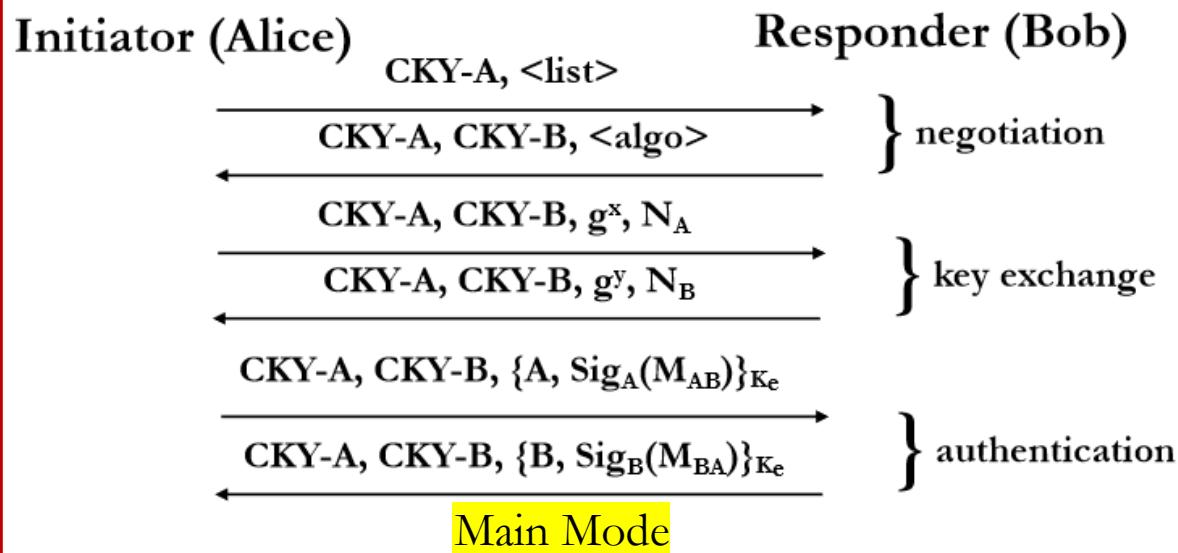
## 5.3 IPsec: IKE (2/5)

Phase 1(Secure channel): IKE first establishes a secure communication channel between the two endpoints by authenticating them and setting up an initial secure-channel SA for IKE itself. This phase ensures that both sides are legitimate and agree on the base security settings.

- There are two types of establishment in the phase-1, called *modes*:
  - Aggressive mode: mutual authentication and session key establishment in three messages.
  - Main mode: uses six messages and has additional functionality such as the ability to hide endpoint identifiers from eavesdroppers.

# 5.3 IPsec: IKE (3/5)

## Phase 1

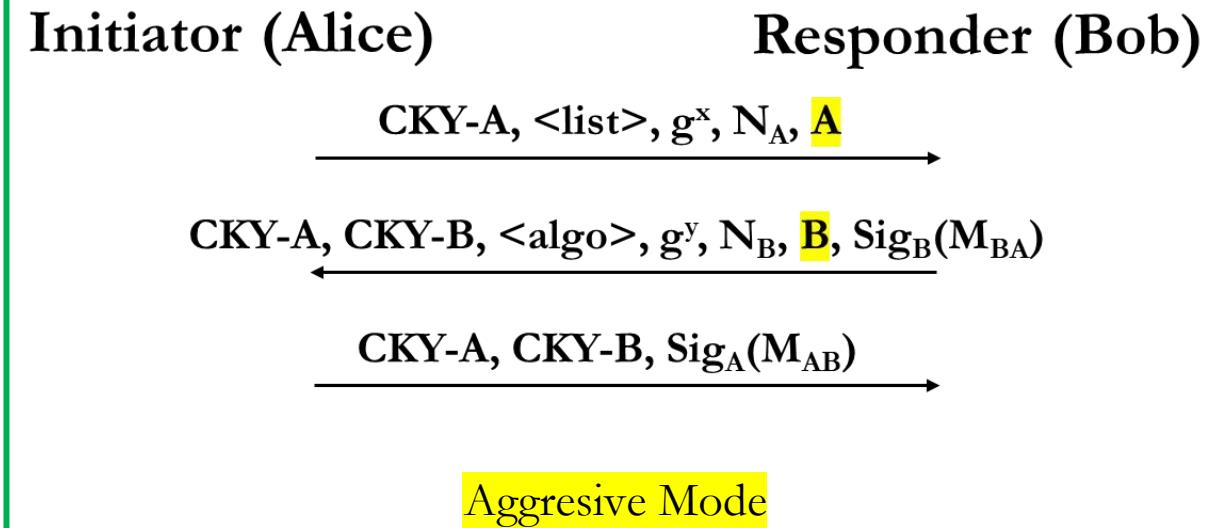
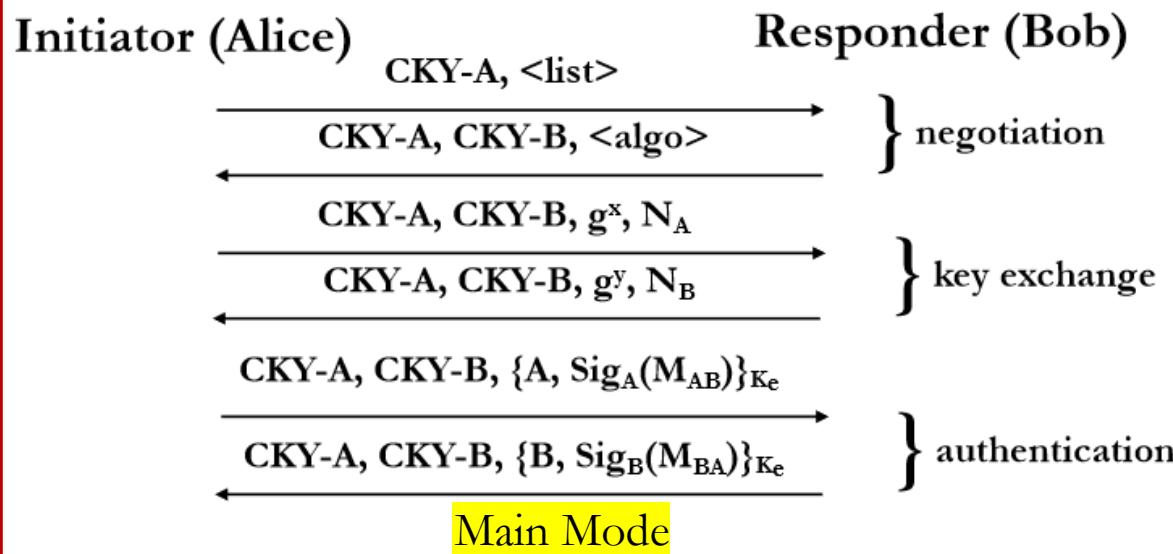


- CKY: cookie
- KM: derived from ( $N_A$  |  $N_B$ ,  $g^{xy}$ )
- Ke: derived from KM
- $M_{AB}$ :  $\text{MAC}_{\text{KM}}(g^x \mid g^y \mid \text{CKY-A} \mid \text{CKY-B} \mid <\text{list}> \mid A)$
- $M_{BA}$ :  $\text{MAC}_{\text{KM}}(g^y \mid g^x \mid \text{CKY-B} \mid \text{CKY-A} \mid <\text{list}> \mid B)$

- Only three message flows
- No identity protection

# 5.3 IPsec: IKE (4/5)

## Phase 1



IPsec can use **pre-shared keys**, where both parties involved in the communication share a secret key in advance. This key is then used for authentication during the IPsec negotiation process. (equivalent to  $sk\_A$  and  $sk\_B$  in signature)

## 5.3 IPsec: IKE (5/5)

### Phase 2

- Message 1 (Initiator → Responder): Proposal, Nonce, and optional DH Key.
- Message 2 (Responder → Initiator): Confirmed Proposal, Nonce, and optional DH Key.
- Message 3 (Initiator → Responder): Final confirmation with a Hash Payload.

- Phase 2 is communicated in secure channel with the help of algorithms and key in phase 1.
- Proposal =Encryption algorithm (e.g., AES or 3DES), Integrity algorithm (e.g., HMAC-SHA1 or HMAC-MD5), and Lifetime of the SA (how long the SA is valid)
- Optional DH Key: The secret key for encryption and integrity can be generated based on this DH or the secret key from phase 1. Aim to achieve stronger security, Perfect Forward Secrecy.
- Hash= a MAC of all communicated message

# **END OF L5**

NETWOCK  
SECURITY

SECURITY

NETWOCK SECURITY



# Network Security

## Lecture 6

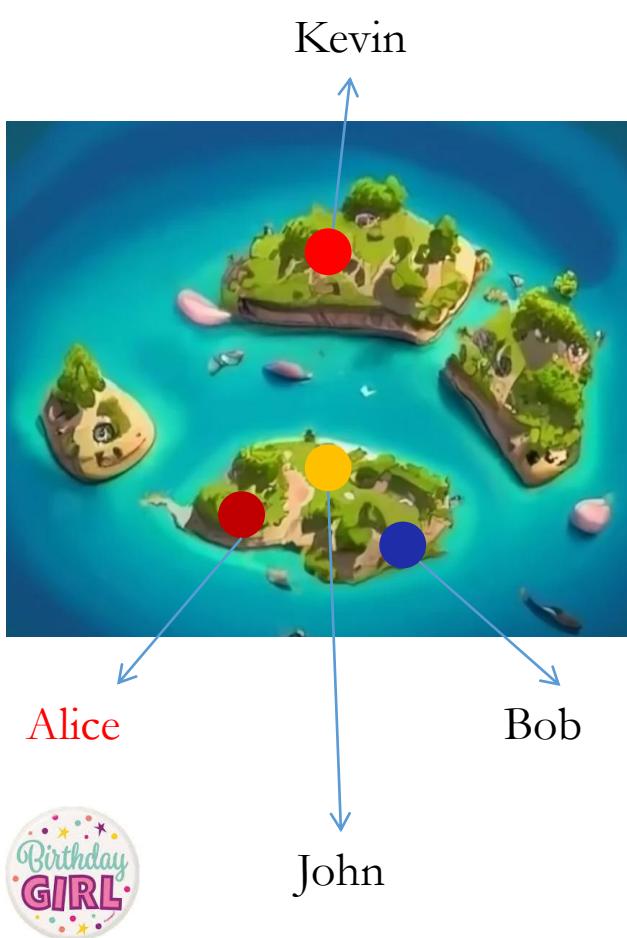
A/Prof. Fuchun Guo

# Outline

- 6.1 Birthday Party (and solutions)
- 6.2 Wired Protocols
- 6.3 Wireless Protocols

# 6.1 Story

# 6.1 Story: Birthday Party (1/5)



- Imagine there's an island with 100 residents, including John. On this island, people don't have phones or other communication devices—except for John, who has the only phone, making him the go-to person for reaching anyone beyond the island.
- One day, Alice decides to invite Bob, another resident on the island, to her upcoming birthday party. Since they live on the same island, Alice simply walks over to Bob's house and tells him directly.
- But later, Alice also wants to invite her friend Kevin, who lives on a distant island. Because Kevin is on another island, Alice can't reach him directly. So, Alice goes to John and asks him to relay the invitation to Kevin. John, with his phone, can connect to the other island and send Alice's message to Kevin.

# 6.1 Story: Birthday Party (2/5)

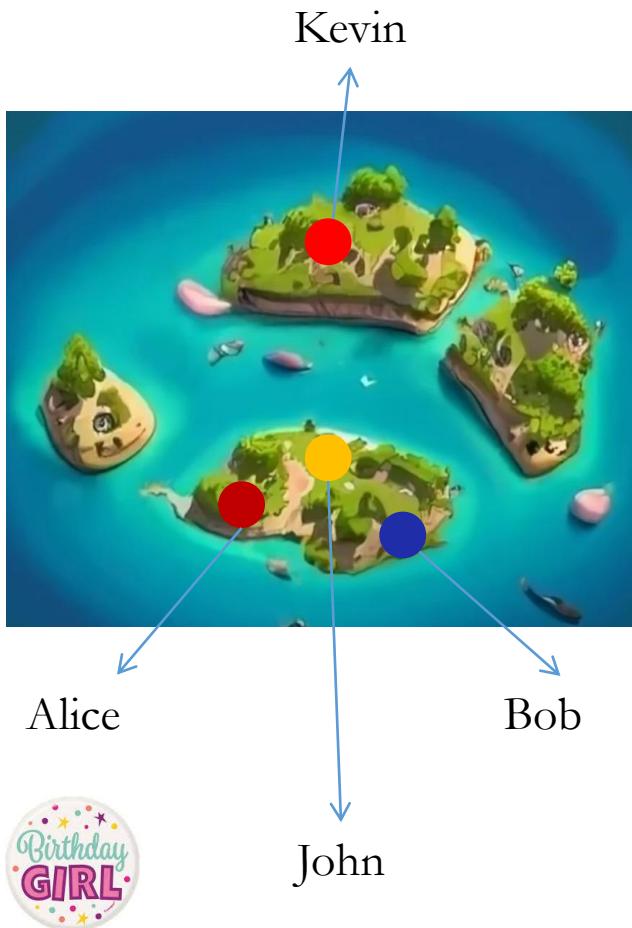


In this story:

- Alice's walk to Bob's house represents communication within a local area (local network).
- John, with his phone, represents a gateway, connecting different islands (or networks) so people can communicate across distances.

Note: Kevin's island also has another person having a phone.

# 6.1 Story: Birthday Party (3/5)

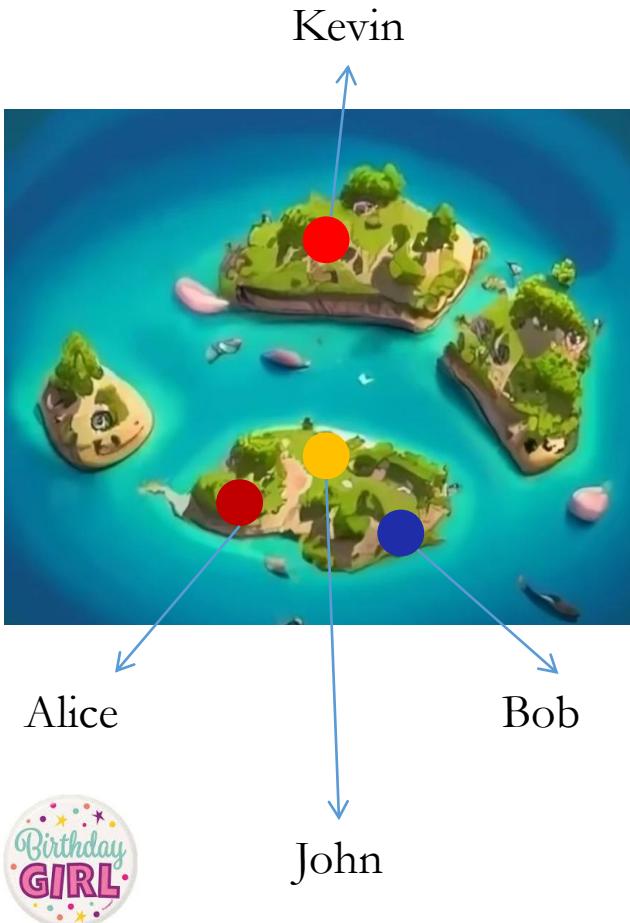


MAC Address:

Unique “Names” for Direct Communication on the Same Network:

On the island, each resident has a unique name. When Alice wants to talk to Bob, she doesn't need to rely on anyone else; she can simply go to Bob's house (or shout to all residents in this island Bob, can you come to my birthday party?). This is similar to how MAC addresses work within a local network—each device has a unique MAC address, so devices can communicate directly within the same network by identifying each other by these “names”. Each device knows whether they are the message receiver or not.

# 6.1 Story: Birthday Party (4/5)

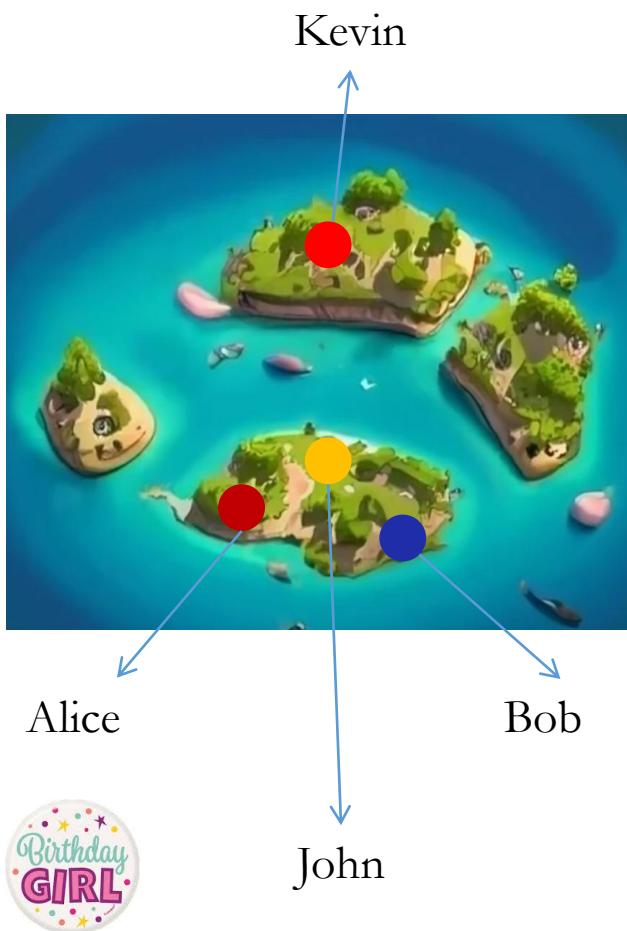


IP Address:

Addresses Beyond the Local Network:

When Alice wants to invite Kevin on the other island, she can't just use his name (or MAC address ) because he's not on her island. Instead, she needs a way to connect beyond her local area, and John helps her with this. John knows how to reach people on different islands as long as the address is clear and ensures the message gets to the right destination. Similarly, IP addresses are like global addresses, allowing devices to be identified across different networks, with routers (like John) forwarding data between networks.

# 6.1 Story: Birthday Party (5/5)



MAC for Local, IP for Global:

- If Alice wants to invite her local friends, she can simply shout to everyone:

(Name) XXX, would u like to come to my party?

(Single-Hop or End-to-End communication using MAC address)

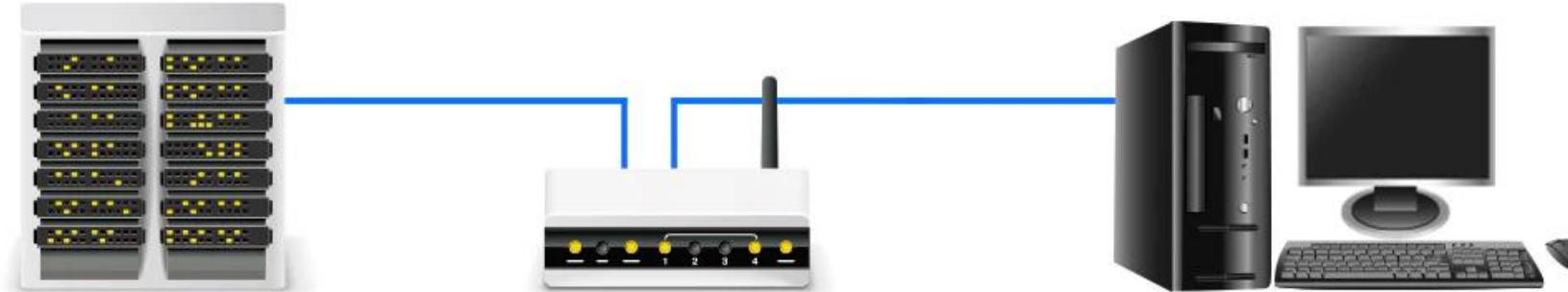
- If Alice wants to invite her friends from other islands, she can send this message to John:

(Island address)YYY: would u like to come to my party?

(Multi-Hop communication using IP address)

# 6.1 Story: Types of Communications

Wired Communication

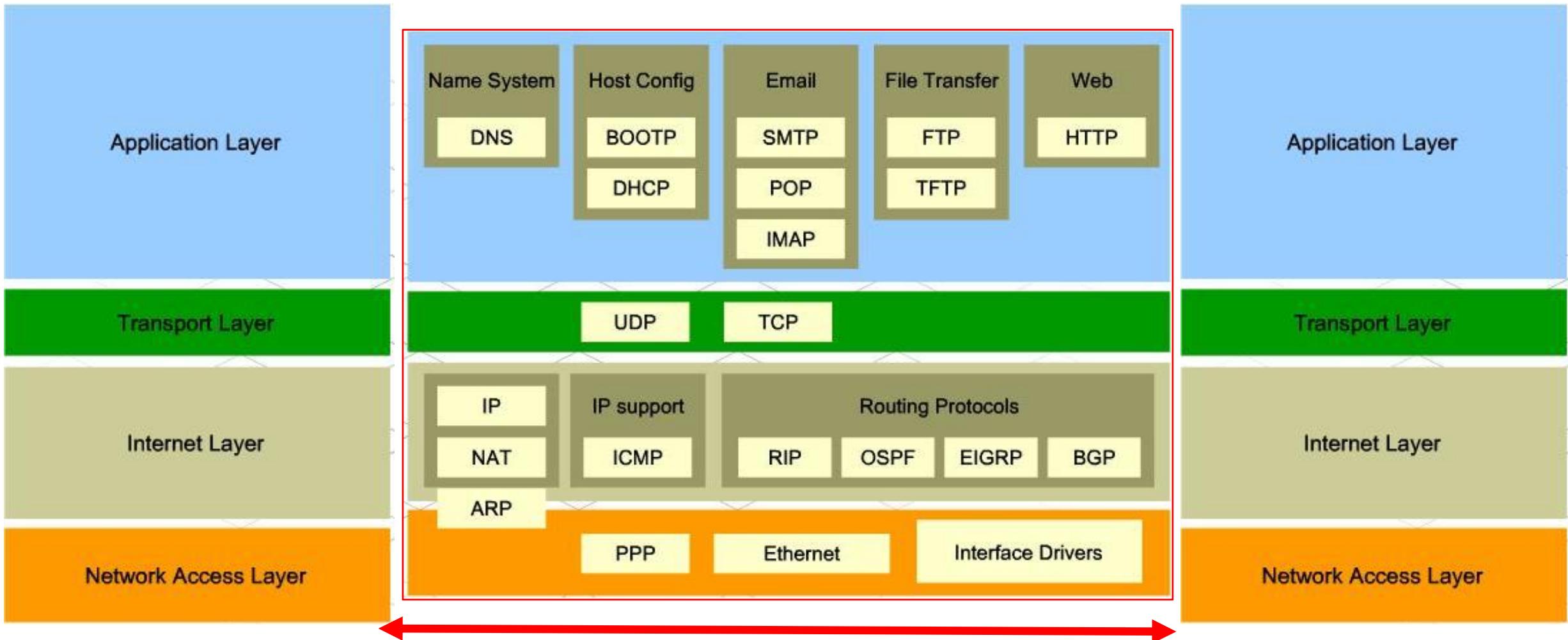


Wireless Communication



## 6.2 Wired Protocols

# 6.2 Wired Protocols: Overview (1/3)



## 6.2 Wired Protocols: Overview (2/3)

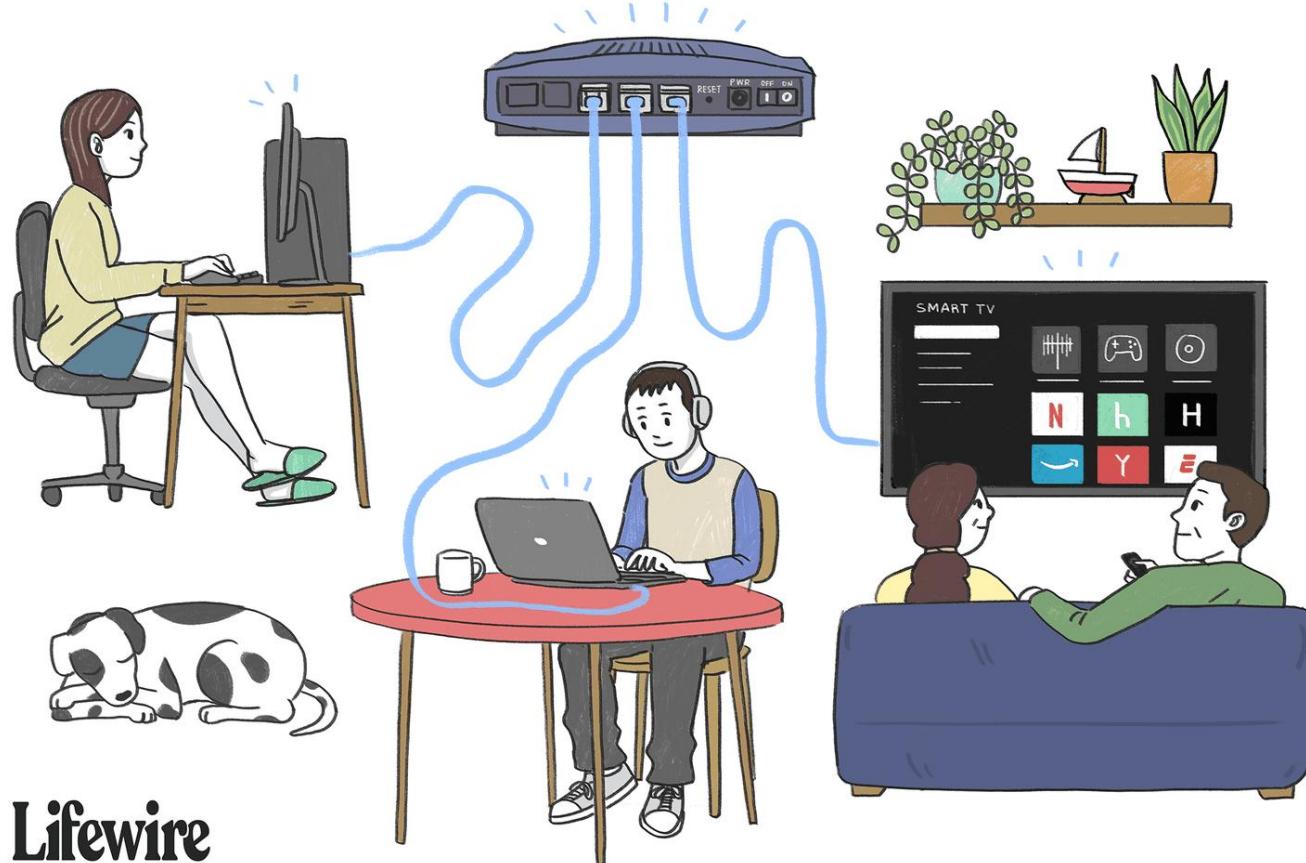
The top three types of link-layer protocols for wired communications

- Data Link Control and Framing Protocols: These are foundational to wired communication, as they define how data is packaged, addressed, and transmitted across a physical link to another device.
- Physical Addressing and Mapping Protocols: Protocols like ARP (Address Resolution Protocol) are crucial for linking network-layer IP addresses to MAC addresses within a local network.
- Error Detection and Correction Protocols: These protocols play an important role in ensuring the reliability of data transmission.

## 6.2 Wired Protocols: Overview (3/3)

- Ethernet Protocol: Ethernet is the fundamental protocol for wired local area networks (LANs). It defines how data is framed, addressed, and transmitted over physical connections, using MAC addresses to deliver packets to the correct devices within a network.
- ARP (Address Resolution Protocol): ARP is responsible for mapping IP addresses to MAC addresses on a local network. When a device needs to communicate with another device within the same network, it uses ARP to find the corresponding MAC address for a known IP address.
- LLDP (Link Layer Discovery Protocol): LLDP is a network discovery protocol that enables devices to share information about themselves and their neighbors. It allows devices, like switches and routers, to advertise their identity, capabilities, and connection details.

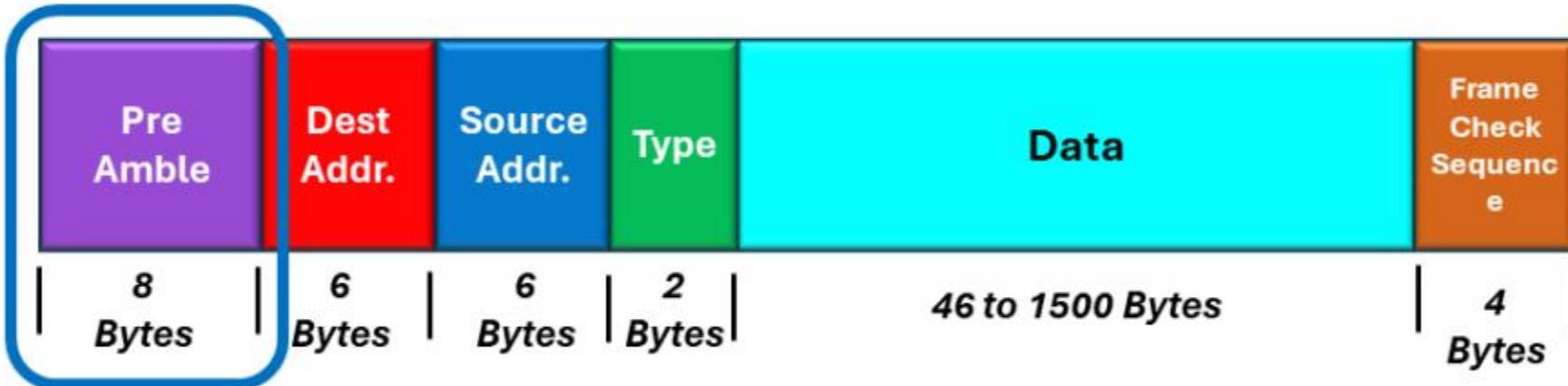
# 6.2 Wired Protocols: Ethernet (1/4)



Lifewire

- Ethernet is the fundamental protocol for wired local area networks (LANs). It defines how data is framed, addressed, and transmitted over physical connections, using MAC addresses to deliver packets to the correct devices within a network.

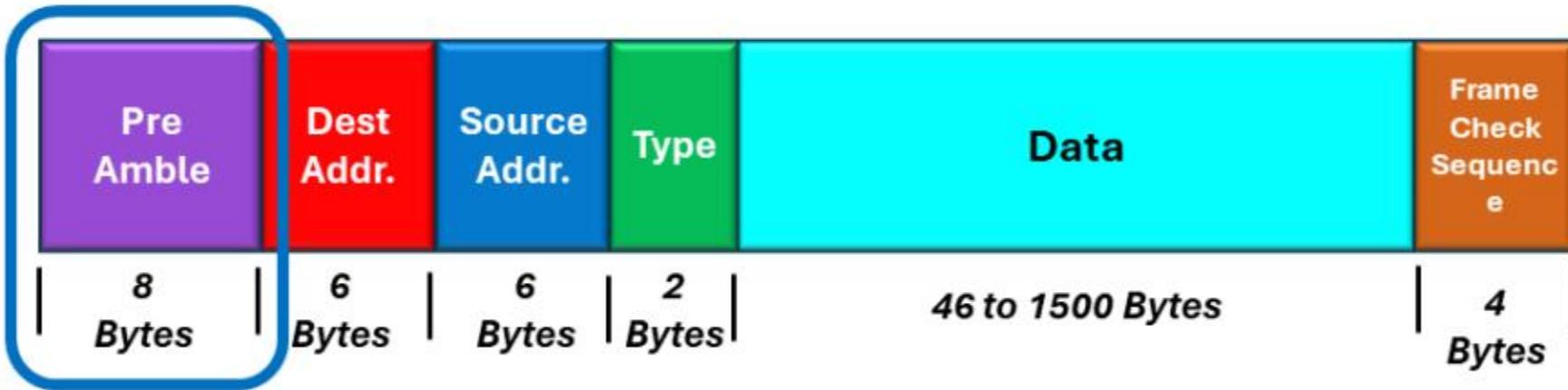
# 6.2 Wired Protocols: Ethernet (2/4)



IEEE 802.3: Standardized in 1983, establishing rules for Ethernet's physical and data link layers.

- Preamble: Signs for synchronizing receiver clocks to make sure it can receive the frame. All values in preamble field of all frames are identical.
- Dest Address: Receiver's MAC address (48 bits, 12 hexadecimal digits)
- Source Address: Sender's MAC address (such as 00:1A:2B:3C:4D:5E)
  - First 24 bits: Device Manufacturers (Cisco Systems, Inc)
  - Last 24 bits: Device Identifier

## 6.2 Wired Protocols: Ethernet (3/4)



- Type: Indicates payload type (which protocol from network layer)
- Data: Contains the actual data (with optional padding)
- Check Sequence: Error-checking using CRC. If incorrect, delete the frame. Will not send it to above layer.

## 6.2 Wired Protocols: Ethernet (4/4)

The MAC address in Ethernet frames is **forgeable** due to the lack of authentication. For instance, if device A is using MAC address X to communicate with router R, an adversary could forge their device's MAC address to match X when connecting to router R. This situation presents two potential issues:

- Router Accepts Identical MAC Addresses: If the router allows two devices to have the same MAC address, any data sent to device A will also be forwarded to the adversary. This could lead to the interception of sensitive information.
- Router Rejects Identical MAC Addresses: If the router detects a conflict and rejects the connection of devices with identical MAC addresses, device A will also be disconnected from the router. This results in a loss of connectivity for device A (DOS attack).

## 6.2 Wired Protocols: ARP (1/5)

- The Address Resolution Protocol (ARP) is a fundamental network protocol used to map IP addresses to MAC (Media Access Control) addresses within a local area network (LAN).
- When a device wants to communicate with another device using an IP address, it must first determine the corresponding MAC address of that device.

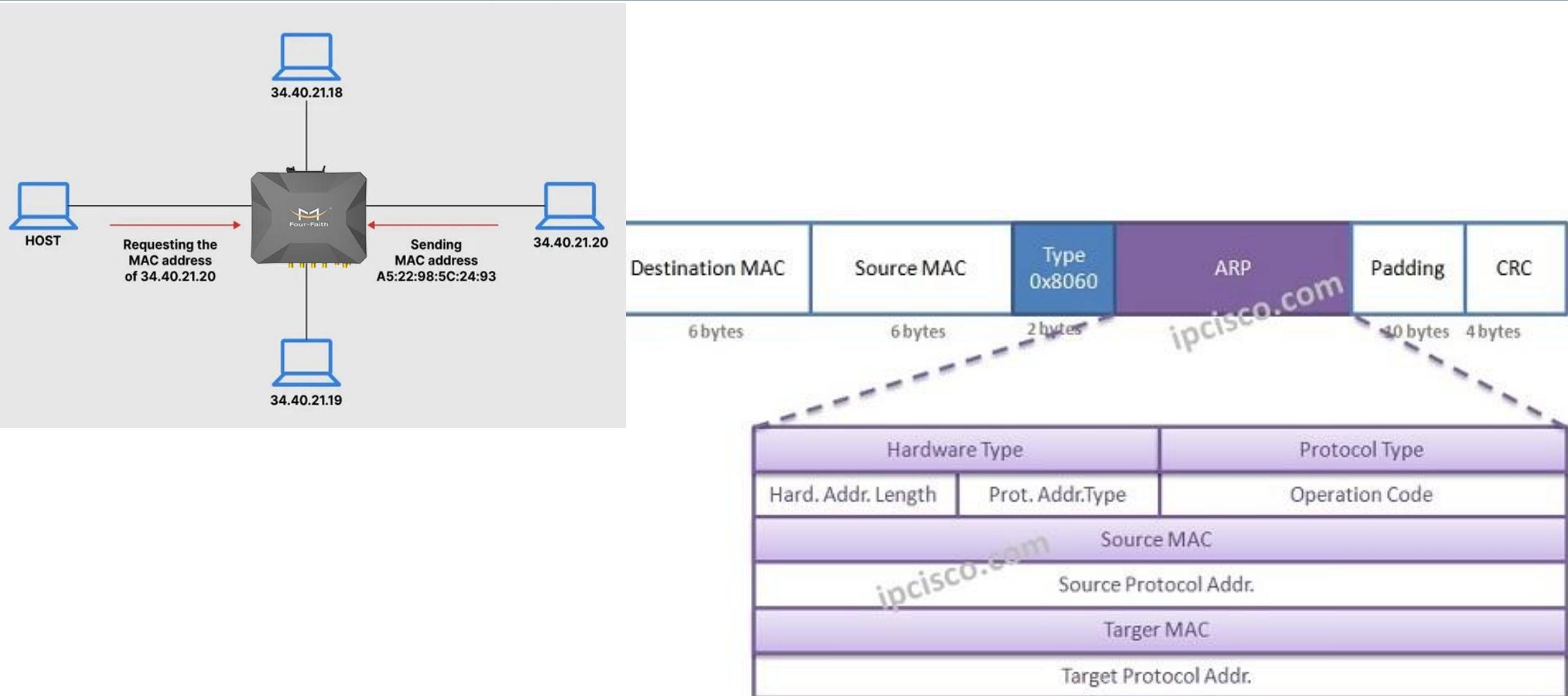
## 6.2 Wired Protocols: ARP (2/5)

### How ARP Works (High-Level):

- ARP Request: When a device (let's call it Device A) wants to send a packet to another device (Device B) on the same local network, Device A will check its ARP cache (a table that stores IP-to-MAC address mappings) to see if it already knows the MAC address of Device B. If it doesn't, Device A sends an ARP request broadcast to all devices on the LAN. The request contains Device B's IP address and asks:

"Who has this IP address? Please send me your MAC address."
- ARP Response: All devices on the network receive the ARP request, but only Device B, which recognizes its own IP address, will respond. Device B sends back an ARP reply directly to Device A, containing its MAC address.
- Updating ARP Cache: Upon receiving the ARP reply, Device A updates its ARP cache with the new mapping of Device B's IP address to its MAC address. Now, Device A can send frames directly to Device B using its MAC address.

# 6.2 Wired Protocols: ARP (3/5)



## 6.2 Wired Protocols: ARP (4/5)

- When a device (let's call it Device A) wants to communicate with an IP address that is outside its LAN, it must send the data through a default gateway.
- ARP protocol uses broadcast MAC address. It is a special address used in Ethernet networks to send a packet to all devices on a local network segment simultaneously. It allows for communication with all devices without needing to specify each device's MAC address individually. The broadcast MAC address is represented as FF:FF:FF:FF:FF:FF.
- Broadcast MAC address plays a crucial role in processes like ARP and service discovery while also presenting challenges related to network performance and security.

## 6.2 Wired Protocols: ARP (5/5)

### ARP Spoofing/Poisoning:

- When device A asks the MAC address of a specific IP addree and an adversary sends falsified ARP messages claiming that their MAC address corresponds to the IP address of a legitimate device, then the legitimate device A's traffic will be redirected to the attacker.
- Impact 1: This enables the attacker to intercept and manipulate the data, leading to potential data breaches or unauthorized access.
- Impact 2: Legitimate device A's traffic will be sent to the attacker instead of the intended recipient, leading to potential data loss and service disruption.

## 6.2 Wired Protocols: LLDP (1/5)

- The Link Layer Discovery Protocol (LLDP) is a standardized network protocol used for discovering and advertising information about directly connected devices on a local area network (LAN).
- LLDP is primarily used to facilitate network management and monitoring by allowing devices to advertise their identity, capabilities, and neighbors. This helps network administrators understand the topology of their network.

## 6.2 Wired Protocols: LLDP (2/5)

### How It Works (High-Level):

- Each LLDP-enabled device periodically sends out (advertise) LLDP Data Units (LLDPDU) as multicast packets. These packets contain various information, including the device's MAC address, system name, port identifier, and the type of device (e.g., switch, router).
- When a device receives an LLDPDU from another device, it updates its LLDP database with the information received, allowing it to learn (discover) about its neighboring devices.

## 6.2 Wired Protocols: LLDP (3/5)

### LLDP Spoofing:

- How it Works: An adversary can configure their device to send false LLDP Data Units (LLDPDUs) claiming to be another legitimate device on the network. By broadcasting a forged identity, they can mislead network devices into believing they are interacting with a trusted device.
- Implications: This can result in the adversary gaining unauthorized access to sensitive data, altering traffic flows, or even launching man-in-the-middle attacks by positioning themselves between legitimate communication channels.

## 6.2 Wired Protocols: LLDP (4/5)

### Denial of Service (DoS):

- How it Works: An adversary could flood the network with excessive LLDP traffic by sending numerous LLDPDUs. This could overwhelm the processing capacity of legitimate devices, causing them to become slow or unresponsive.
- Implications: Such a denial of service can disrupt network operations and make it difficult for legitimate devices to communicate with each other, thereby affecting productivity and service availability.

## 6.3 Wireless Protocols

Wireless communication involves the transfer of information between two or more devices through electromagnetic waves, eliminating the need for physical connections like wires or cables.

# 6.3 Wireless Protocols: Background (1/6)

- Wired Communication Protocol (1983): The wired communication protocol was standardized in 1983. This foundational protocol established a framework for reliable data transmission over wired networks and set the stage for further advancements in network technology.
- Wireless Communication Protocol (1997): The wireless communication protocol, known as IEEE 802.11, was standardized in 1997. This groundbreaking development marked the beginning of wireless networking, enabling devices to communicate without physical connections. The standard has evolved significantly ensuring it meets the growing demands of users and applications in an increasingly connected world.

# 6.3 Wireless Protocols: Background (2/6)



- In Wireless network, each mobile device client sends all of its communications to a network device called an access point (AP).
- The AP acts as an Ethernet bridge and forwards the communications to the appropriate network, such as a wired LAN or another Wireless Network.

# 6.3 Wireless Protocols: Background (3/6)

Security requirements are **more critical** in wireless communications for several reasons:

- Inherent Vulnerability: Wireless communications transmit data over radio waves, making signals accessible to anyone within range. This open medium exposes data to eavesdropping and unauthorized interception, increasing the need for robust security measures to protect sensitive information.
- Lack of Physical Barriers: Unlike wired networks, where physical access to cables is required to intercept data, wireless networks do not have the same physical limitations. This ease of access means that malicious actors can exploit vulnerabilities from a distance, necessitating stronger security protocols to mitigate risks.

# 6.3 Wireless Protocols: Background (4/6)

## 1. Eavesdropping

Description: Eavesdropping involves intercepting data packets transmitted over the wireless medium. Since wireless signals travel through the air, attackers can use simple tools to capture these signals without needing physical access to the network.

Impact: Attackers can gain access to sensitive information such as passwords, personal data, and confidential communications.

## 2. Man-in-the-Middle (MitM) Attacks

Description: In MitM attacks, an attacker secretly intercepts and relays messages between two parties who believe they are directly communicating with each other. This is easier in wireless settings because attackers can position themselves between devices without needing physical access.

Impact: Attackers can modify the communication, steal credentials, or inject malicious content into the data stream.

# 6.3 Wireless Protocols: Background (5/6)

## 3. Rogue Access Points

Description: Attackers can set up rogue access points that appear legitimate but are actually controlled by the attacker. Unsuspecting users may connect to these rogue points, thinking they are connecting to a trusted network.

Impact: This allows attackers to capture credentials and sensitive information from connected devices.

## 4. Denial of Service (DoS) Attacks

Description: In wireless networks, attackers can easily disrupt service by flooding the network with traffic or sending de-authentication packets to disconnect legitimate users. This can be done from a distance, making it easier than in wired networks.

Impact: Users may be unable to access the network, leading to service interruptions.

# 6.3 Wireless Protocols: Background (6/6)

## 5. Session Hijacking

Description: Attackers can take control of a user's session by stealing session tokens or cookies over the air. This is more feasible in wireless environments due to the ease of intercepting data packets.

Impact: Attackers can impersonate users and gain unauthorized access to accounts or services.

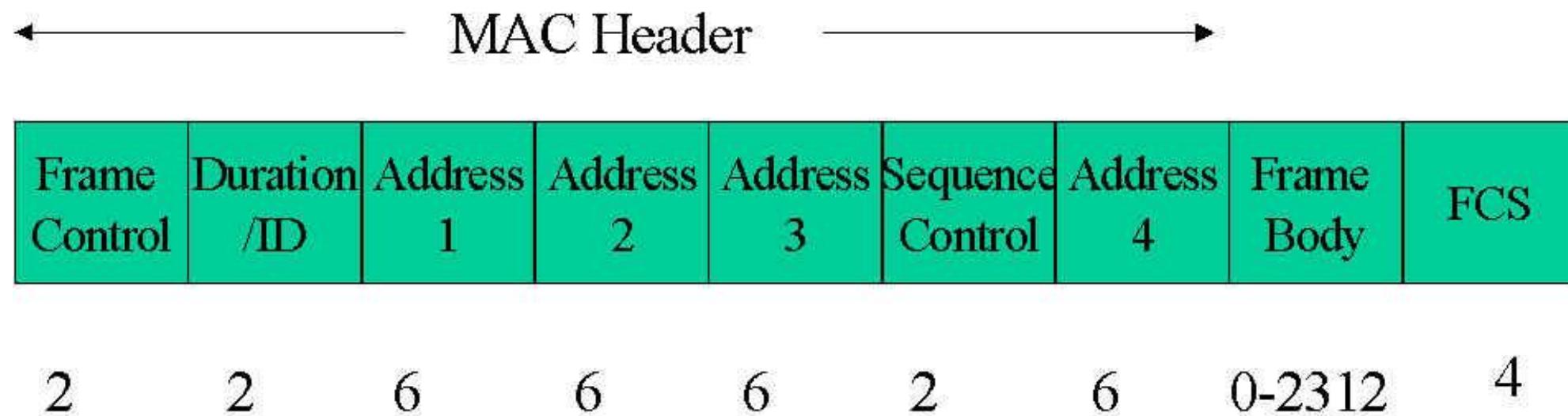
## 6. Jamming

Description: Jamming involves broadcasting signals that disrupt the communication channels of a wireless network. This can effectively block legitimate communication.

Impact: Users cannot connect to the network, resulting in denial of service.

# 6.3 Wireless Protocols: Frame (1/5)

A Wi-Fi frame, also known as an 802.11 frame, is the unit of communication in Wi-Fi networks. It contains all the data needed to transfer information between devices over a wireless network, including addressing, control information, and the payload. Wi-Fi frames are more complex than Ethernet frames because they support wireless-specific functions like security.



# 6.3 Wireless Protocols: Frame (2/5)

Frame Control	Duration /ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	Frame Body	FCS
---------------	--------------	-----------	-----------	-----------	------------------	-----------	------------	-----

1. Frame Control (2 bytes): This field contains subfields that specify the frame type (management, control, or data) . To make sure the receiver know the purpose of this frame. ACK is applied here to gurantee that the transmission is reliable similar to TCP protocol.
2. Duration/ID (2 bytes): This field indicates how long the frame will occupy the wireless medium, which allows other devices to know when to wait before transmitting.

## 6.3 Wireless Protocols: Frame (3/5)

Frame Control	Duration /ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	Frame Body	FCS
---------------	--------------	-----------	-----------	-----------	------------------	-----------	------------	-----

3. Addresses (6 bytes each)

Address 1 (Receiver Address): Specifies the MAC address of the destination device.

Address 2 (Transmitter Address): Specifies the MAC address of the device sending the frame.

Address 3 (BSSID): Represents the MAC address of the **wireless access point (router)** or base station, used to identify the specific network.

Address 4 (Optional): This is present only in certain frame types, like frames in wireless distribution systems (WDS). It is used when packets travel between access points.

## 6.3 Wireless Protocols: Frame (4/5)

Frame Control	Duration /ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	Frame Body	FCS
---------------	--------------	-----------	-----------	-----------	------------------	-----------	------------	-----

4. Sequence Control (2 bytes): This field is used to reassemble fragmented frames by providing a sequence number, which helps the receiver reassemble frames in the correct order.

- Sequence Number: This part remains the same for all fragments of a single packet. Each packet (before fragmentation) is assigned a unique sequence number.
- Fragment Number: Each fragment within a sequence has a different fragment number, starting from 0 and increasing by 1 for each subsequent fragment of the same packet. This lets the receiver reassemble the fragments correctly in sequence

# 6.3 Wireless Protocols: Frame (5/5)

Frame Control	Duration /ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	Frame Body	FCS
---------------	--------------	-----------	-----------	-----------	------------------	-----------	------------	-----

## 5. Frame Body (0-2312 bytes)

Contains the actual data payload for data frames or information specific to management or control frames. This section may be encrypted data if security is enabled.

## 6. FCS (Frame Check Sequence) (4 bytes)

This field holds a CRC (Cyclic Redundancy Check) value to ensure data integrity. The receiver checks this value to verify that the frame wasn't corrupted during transmission. If the FCS doesn't match the calculated value, the frame is discarded.

# 6.3 Wireless Protocols: Thinking

Who can connect to AP?

Using K

1997

WEP

Using K or register (ID, pw)

2003

TKIP

Using K-based DH  
or register (ID, pw)

2004

AES

2018

AES

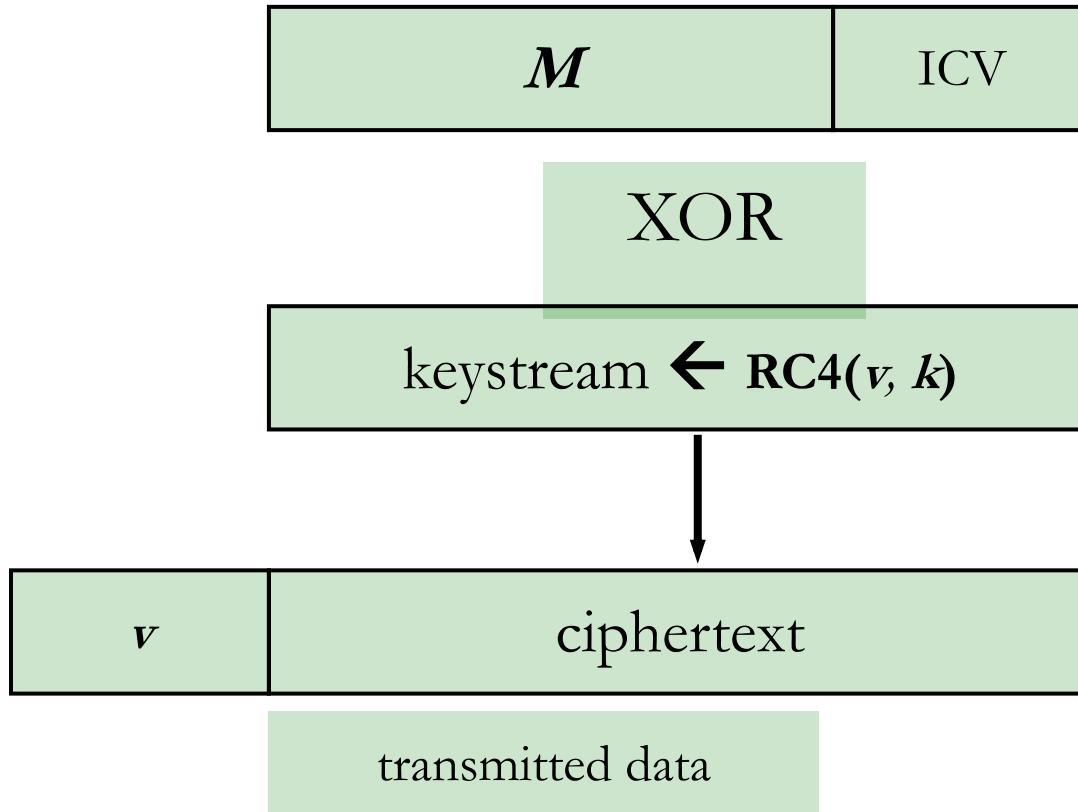
How to secure communications?

# 6.3 Wireless Protocols: WEP (1/6)

- Wired Equivalent Privacy (WEP) is an early security protocol (1997) designed to protect wireless networks by encrypting data transmitted over Wi-Fi.
- It provides authentication and a level of security (confidentiality & integrity) comparable to wired networks by encrypting data with the RC4 stream cipher and using a shared key for all devices on the network.
- However, WEP has significant weaknesses, particularly in how it manages encryption keys and its use of a short initialization vector (IV), making it vulnerable to attacks. These vulnerabilities allow attackers to crack the encryption within minutes, exposing data to interception and manipulation.

## 6.3 Wireless Protocols: WEP (2/6)

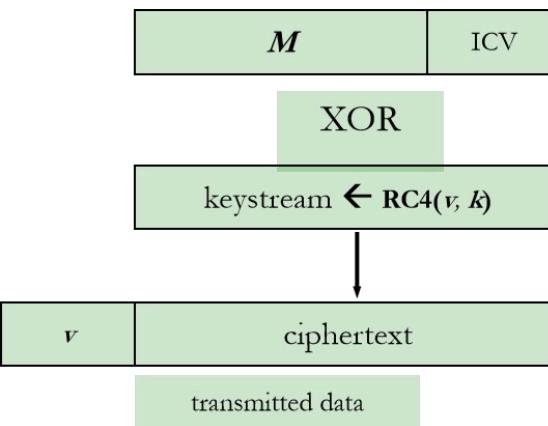
Let  $M$  be the message,  $v$  the current IV and  $k$  the secret shared key:



To protect against unauthorised data, an integrity algorithm CRC-32 operates on the plaintext to produce the ICV. This is a **(non-cryptographic)** 32-bit checksum, or integrity check value (ICV).

RC4 algorithm will generate a long-enough keystream to XOR bitstrings  $M // ICV$

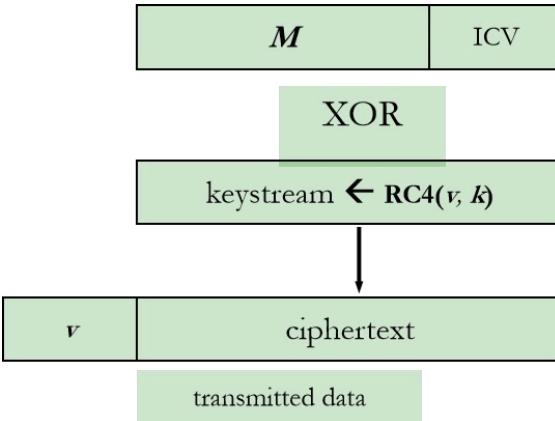
# 6.3 Wireless Protocols: WEP (3/6)



1. The 40-bit secret key is concatenated with an 24-bit initialisation vector (IV), resulting in a key with an overall length of 64-bits.
2. The resulting key is put into the pseudo-random number generator (PRNG), i.e., the stream cipher RC4.
3. The PRNG (RC4) outputs a pseudo-random key sequence based on the input key.
4. The resulting sequence is used to encrypt the data ( $M$  and ICV) by doing a bitwise XOR.

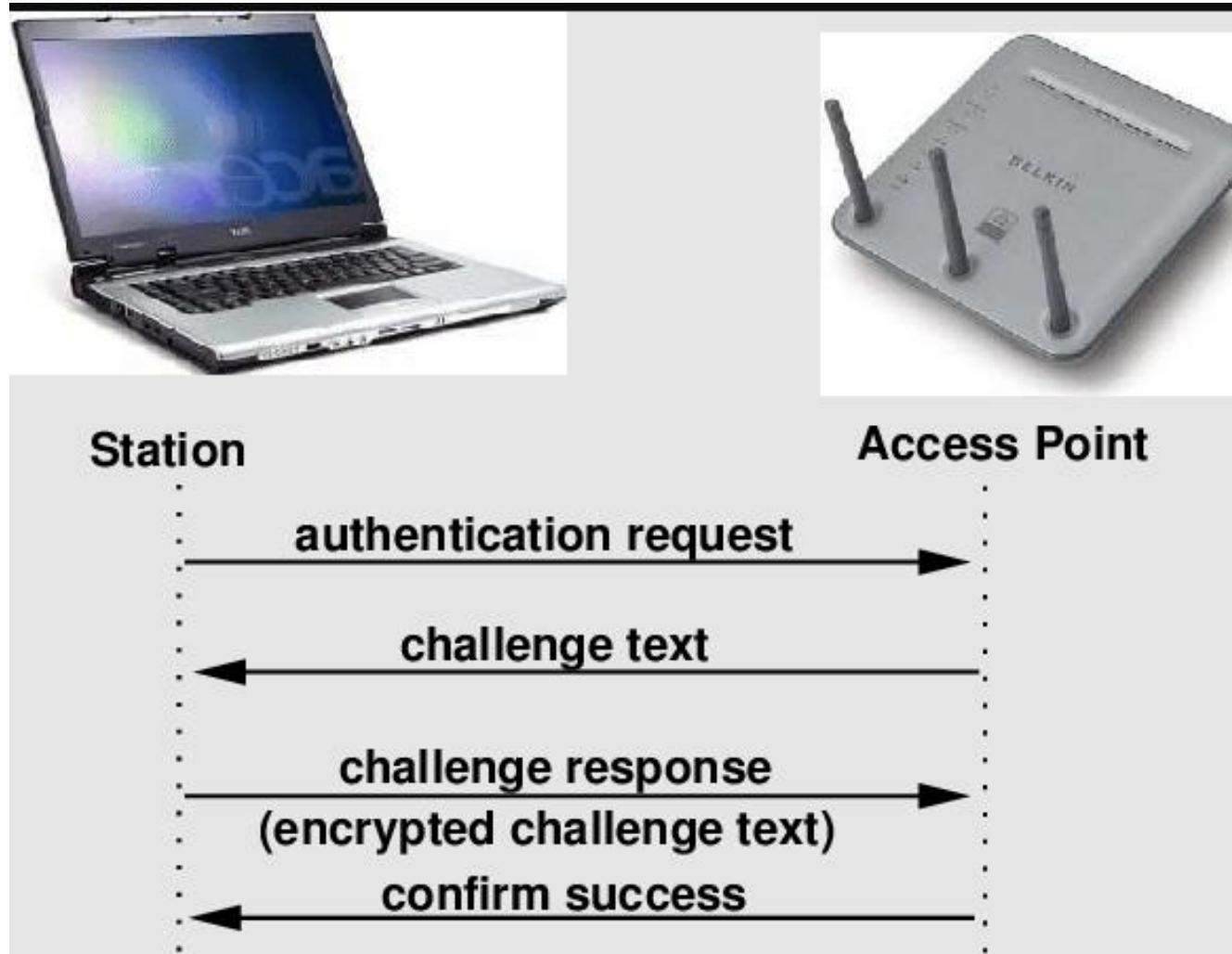
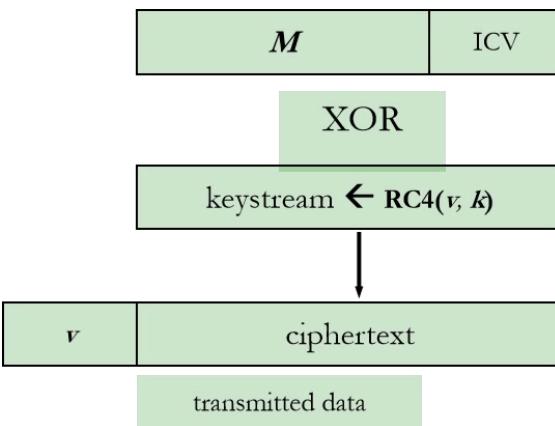
## 6.3 Wireless Protocols: WEP (4/6)

1. The IV of the incoming message is used to generate the key sequence necessary to decrypt the incoming message.



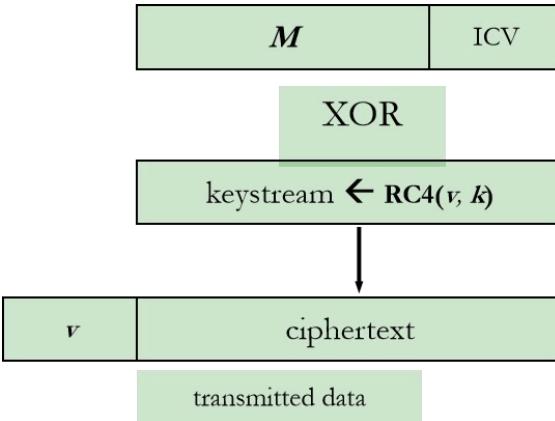
2. The ciphertext, combined with the proper key sequence, yields the original plaintext and ICV.
3. The decryption is verified by performing the integrity check algorithm on the recovered plaintext and comparing the output ICV to the ICV transmitted with the message.
4. If the ICV is not equal to the ICV received, the message has an error and returns the error.

# 6.3 Wireless Protocols: WEP (5/6)



# 6.3 Wireless Protocols: WEP (6/6)

- **Key management:** the same shared secret key is used for both authentication and encryption.



- **Integrity:** It is possible to modify some bits in a message so that the resulting message still passes the ICV test.
- **Confidentiality:** key size is too short, Key stream reuse (IV is too short).

## 6.3 Wireless Protocols: TKIP

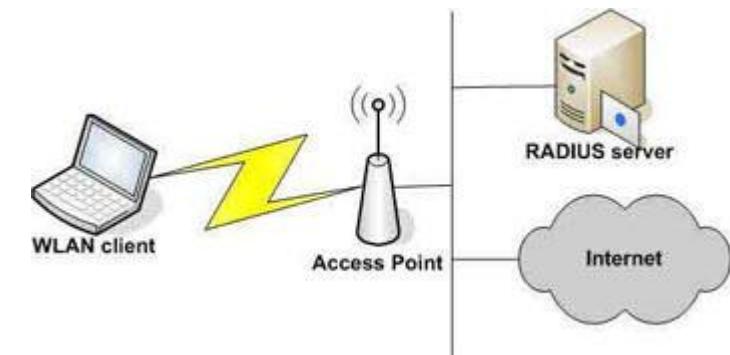
- Temporal Key Integrity Protocol (TKIP) is a security protocol introduced to enhance the security of wireless networks, primarily as part of the WPA (Wi-Fi Protected Access) standard. TKIP was developed as a temporary solution to address the vulnerabilities of WEP (Wired Equivalent Privacy) while maintaining compatibility with existing hardware.
- TKIP allows to be implemented on older wireless devices without requiring significant hardware upgrades or replacements. This approach enabled users to enhance their network security while continuing to use their current equipment.

## 6.3 Wireless Protocols: Authentication in WEP

- In WEP, authentication is achieved through a shared secret key, which is the same for all devices connected to the access point (AP). When a device wants to join the network, it must present the shared key to authenticate itself, allowing it to encrypt and decrypt the data transmitted over the network.
- However, Because all devices use the same key, managing that key becomes problematic. If the key needs to be changed (for instance, if a device is compromised), it must be updated on all devices connected to the network, which can be cumbersome and prone to error.

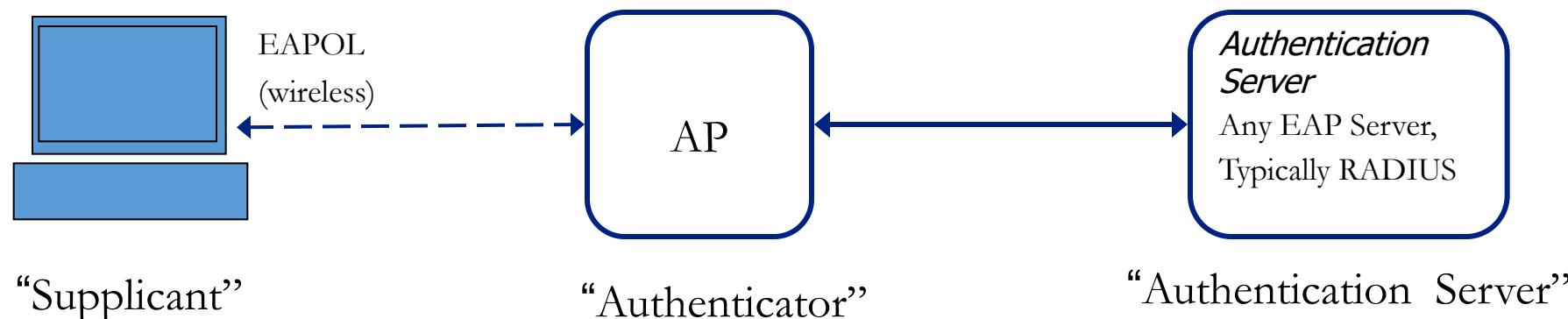
## 6.3 Wireless Protocols: IEEE 802.1X (1/9)

- IEEE 802.1X (2003) is a network access control standard that provides a framework for authenticating devices connecting to a network, typically over Wi-Fi.
- It uses a protocol framework called EAP (Extensible Authentication Protocol) to perform secure authentication among
  - a client (known as a "supplicant"),
  - an authenticator (such as access point), and
  - an authentication server (usually a RADIUS server).



# 6.3 Wireless Protocols: IEEE 802.1X (2/9)

- IEEE 802.1X setup:
  - Supplicant authenticates via Authenticator to central Authentication Server.
  - Authentication Server confirms Supplicants credentials.
  - Authentication Server directs Authenticator to allow the Supplicant access to services after the successful authentication.



## 6.3 Wireless Protocols: IEEE 802.1X (3/9)

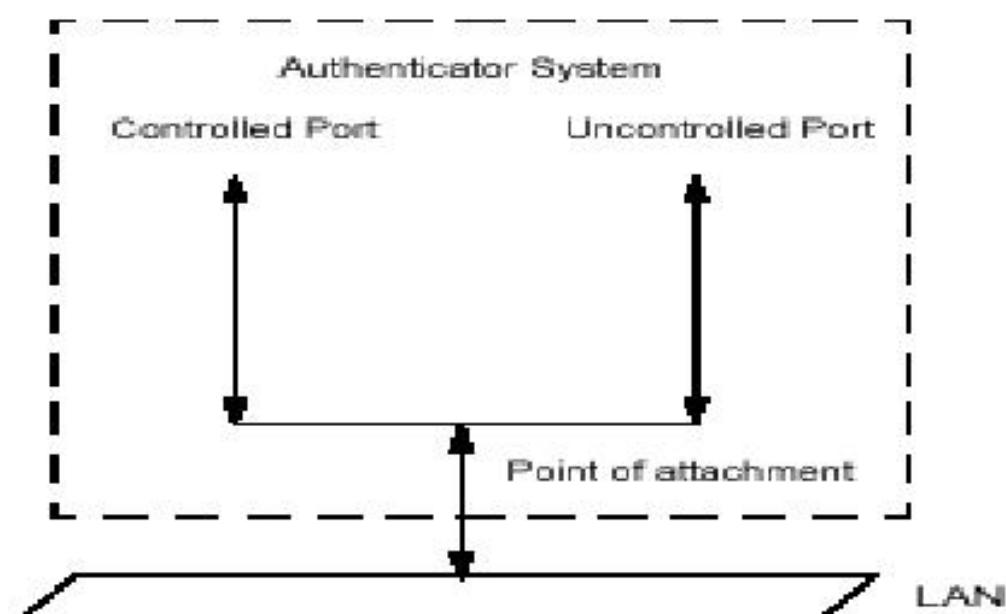
- Port-based access control is a network security mechanism that restricts network access at the point of physical or logical connection, such as an Ethernet or Wi-Fi access port.
- Defined in the IEEE 802.1X standard, it primarily uses the concepts of "ports" to control which devices are allowed access to the network.

- **Controlled Port :**

accepts packets from authenticated devices.

- **Uncontrolled Port :**

only passes 802.1X packets.



## 6.3 Wireless Protocols: IEEE 802.1X (4/9)

EAP (Extensible Authentication Protocol), as used in 802.1X, is a authentication framework nad provides several advantages over WEP's simple shared-key authentication, particularly in areas like user management and key management:

- Individual User Authentication: EAP allows each user to authenticate individually, often with unique credentials (passwords), rather than relying on a single, static network key shared by all devices as in WEP. This approach improves user management, making it easier to track and control access on a per-user basis, which is essential for auditing and accountability.
- Dynamic Key Management: EAP supports the dynamic generation and distribution of session keys. After successful authentication, unique encryption keys are generated for each session, providing stronger security than WEP's fixed shared key. This dynamic keying mechanism means that each user has their own encryption key, reducing the risk of a compromised key affecting the entire network.

## 6.3 Wireless Protocols: IEEE 802.1X (5/9)

PEAP: Password-Based Authentication (via TLS tunnel)

- Process: Client and Authentication Server first establish a secure TLS tunnel. Inside the tunnel, the client sends username/password (using a challenge-response method) for authentication.
- Security: Encrypts the entire authentication process using TLS to protect passwords from exposure.

EAP-TLS: Mutual Certificate-Based Authentication

- Process: Client and Server both use X.509 certificates for mutual authentication. TLS handshake establishes a secure communication channel.
- Security: High security, as it requires certificates on both the client and server sides.

# 6.3 Wireless Protocols: IEEE 802.1X (6/9)

How it (PEAP) works:

- On detection of a new supplicant, the port on the authenticator is enabled and set to the "unauthorized" state. In this state, only 802.1X traffic is allowed; other traffic is dropped.
- Through 802.1X traffic, the client (supplicant) and the Authentication Server (AS) establish a secure TLS tunnel via running TLS protocols. In this step, the server is authenticated by the client using the server's certificate (issued by a trusted Certificate Authority). The TLS tunnel is established to encrypt all further communication between the client and the server. In this step, both know a shared key.

## 6.3 Wireless Protocols: IEEE 802.1X (7/9)

How it (PEAP) works:

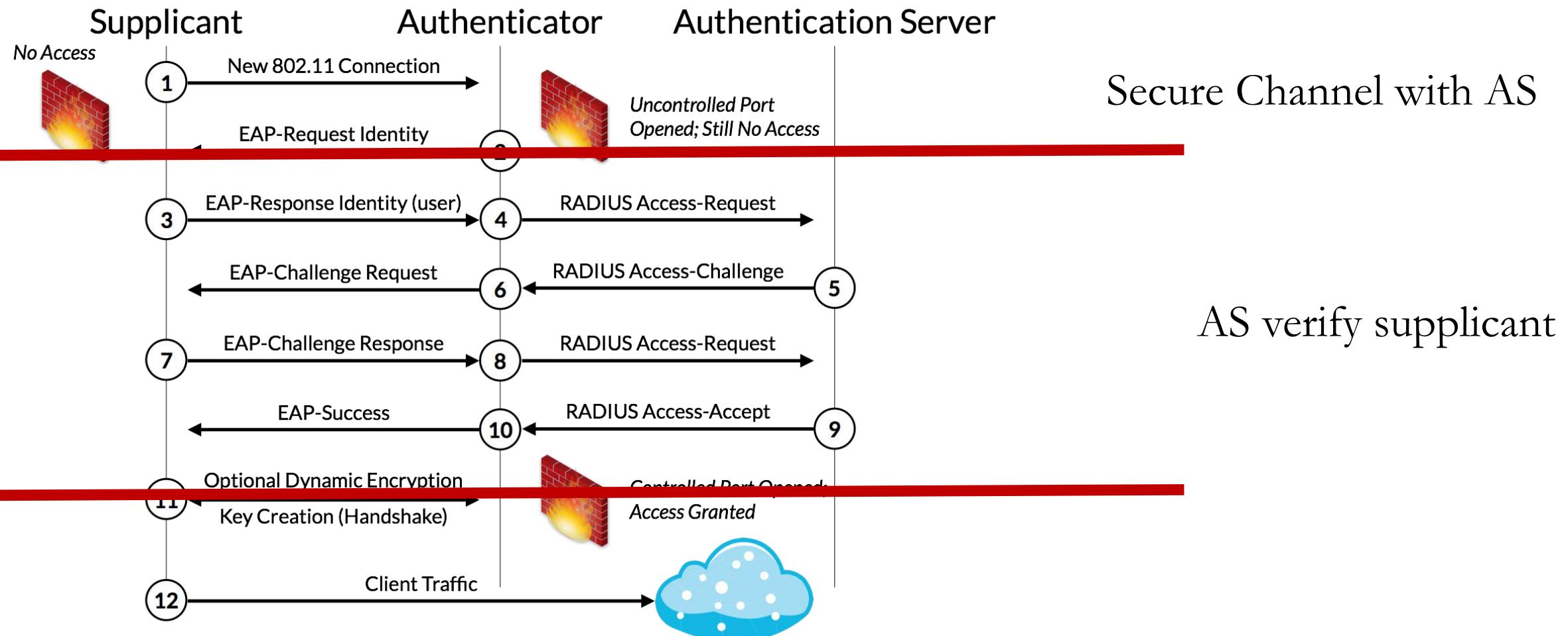
- To initiate authentication on client, the authenticator (AP) will periodically transmit EAP-Request Identity frames to a special Layer 2 address on the local network segment. The supplicant listens on this address, and on receipt of the EAP-Request Identity frame it responds with an EAP-Response Identity frame containing an identifier for the supplicant such as a User ID.
- The authenticator then encapsulates this Identity response in a RADIUS Access-Request packet and forwards it on to the authentication server.

## 6.3 Wireless Protocols: IEEE 802.1X (8/9)

- The authentication server sends a reply (encapsulated in a RADIUS Access-Challenge packet) to the authenticator, containing an EAP Request specifying the EAP authentication Method. The authenticator encapsulates the EAP Request in an EAPOL frame and transmits it to the supplicant.
- If the authentication server and supplicant agree on an EAP Method, EAP Requests and Responses are sent between the supplicant and the authentication server (translated by the authenticator) until the authentication server responds with either an EAP-Success message, or an EAP-Failure message.
- If authentication is successful, the authenticator sets the port to the "authorized" state and normal traffic is allowed, if it is unsuccessful the port remains in the "unauthorized" state.

# 6.3 Wireless Protocols: IEEE 802.1X (9/9)

EAP and RADIUS messages in 802.1X authentication session.



## 6.3 Wireless Protocols: PSK (1/2)

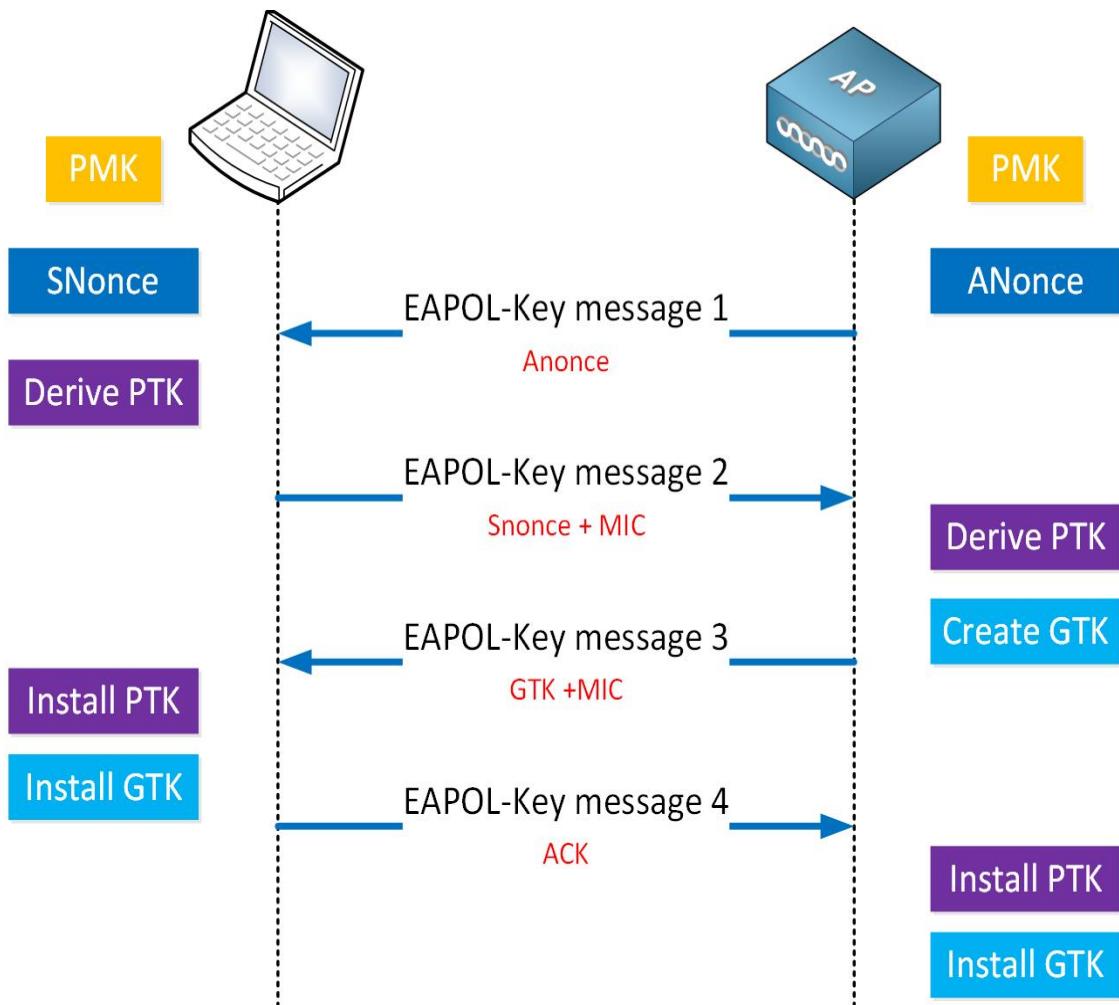
PSK (pre-shared key) mechanism is a simpler authentication method than EAP.

- Shared Key: A common secret key (password) is shared between the user and the network (e.g., a Wi-Fi password).
- Key Derivation: The PSK and nonces are used to derive session keys that are used for encrypting communication between the device and the access point.
- No Server Authentication: PSK does not require a dedicated authentication server and is typically used in home or small office networks.

### How Does PSK Work?

1. The user enters a shared password to authenticate with the access point.
2. Both the client and the access point generate a session key from the password (PSK).
3. This session key is used to encrypt all subsequent communication between the client and the access point.

# 6.3 Wireless Protocols: PSK (2/2)



MIC = Message Integrity Check

PMK = shared secret key

PTK =  $H(\text{PMK}, \text{Snonce}, \text{Anonce}, \text{MAC1}, \text{MAC2})$

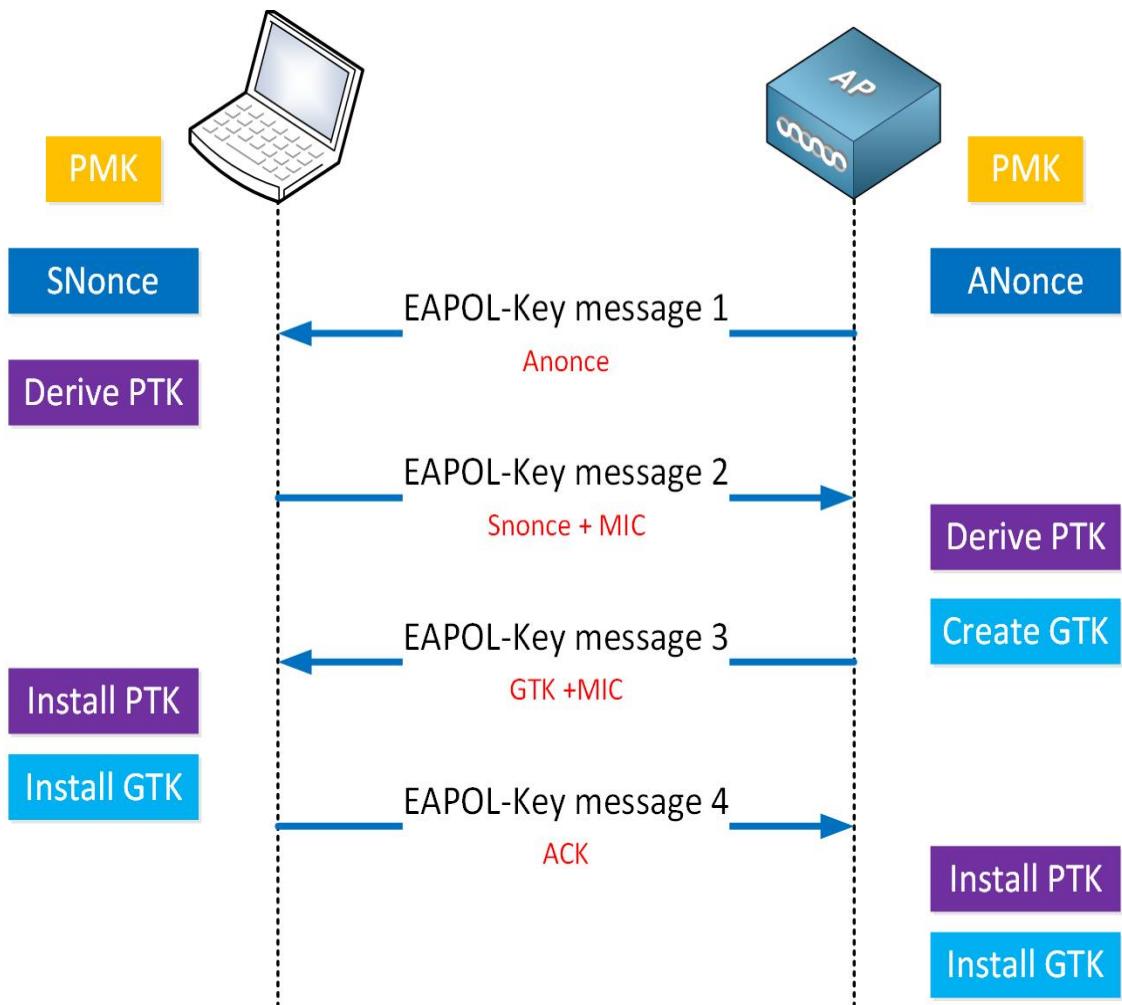
GTK = Generated by the AP, then encrypted with the KEK from the PTK and sent to the client

# 6.3 Wireless Protocols: Comparisons

Feature	WEP	WPA	WPA2
Encryption Algorithm	RC4	RC4-based TKIP	AES
Key	Static shared key	Dynamic key	Dynamic key
Authentication Method	PSK	EAP or PSK	EAP or PSK
Security	Insecure	Better than insecure	Strong

Dynamic = computed from static shared key + nonce

# 6.3 Wireless Protocols: WPA3 (1/3)



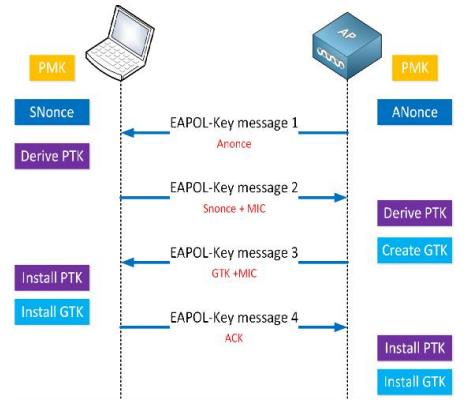
MIC = Message Integrity Check

PMK = from Diffie-Hellman Key Exchange protocol

PTK =  $H(\text{PMK}, \text{PW}, \text{Snonce}, \text{Anonce}, \text{MAC1}, \text{MAC2})$

GTK = Generated by the AP, then encrypted with the KEK from the PTK and sent to the client

# 6.3 Wireless Protocols: WPA3 (2/3)



MIC = Message Integrity Check)

PMK = from Diffie-Hellman Key Exchange protocol

PTK =  $H(\text{PMK}, \text{PW}, \text{Snonce}, \text{Anonce}, \text{MAC1}, \text{MAC2})$

GTK = Generated by the AP, then encrypted with the KEK from the PTK and sent to the client

- Forward security (or forward secrecy) is a security property that ensures that even if long-term keys (such as a pre-shared key) are compromised in the future, past session keys and communications (recorded by the adversary) remain secure.
- Even if an adversary obtains the pre-shared key on Day 2, they would still be unable to decrypt the recorded communications from Day 1. This is because each session's encryption relies on a unique, temporary session key that is separate from the long-term pre-shared key

# 6.3 Wireless Protocols: WPA3 (3/3)

Feature	WEP	WPA	WPA2	WPA3
Encryption Algorithm	RC4	RC4-based TKIP	AES	AES
Key	Static shared key	Dynamic key	Dynamic key	Dynamic key
Authentication Method	PSK	EAP or PSK	EAP or PSK	EAP or PSK
Security	Insecure	Better than insecure	Strong	Stronger

Note: the authentication methods are slightly different.

# **END OF L6**

NETWOCK  
SECURITY

SECURITY

NETWOCK SECURITY



# Network Security

## Lecture 7

A/Prof. Fuchun Guo

# Outline

- P.M.T: Policy, Mechanism, Tools
- Security Policy
- Security Mechanism
- Firewalls
- Instruction Detection and Prevention
- Subject Revision and Final Exam

**7.1 P.M.T.**

# 7.1 P.M.T: Overview (1/2)

- Security Policy: It outlines an organization's high-level security objectives, rules, and expectations, such as "block unauthorized access to the internal network."
- Security Mechanisms: These are the abstract processes or functions designed to implement the policy. For example, a mechanism might specify: "If a data packet originates from an untrusted IP address (rule), then block the packet (action)."
- Security Tools: These are the practical implementations of security mechanisms. Tools like firewalls and intrusion detection systems applying the rules and actions defined.

# 7.1 P.M.T: Overview (2/2)

## From Policy to Mechanism:

- When tailoring a security policy for specific application scenarios, additional requirements may arise. These special requirements, combined with the policy, can result in various mechanisms designed to meet the specific needs. This flexibility allows organizations to address diverse security challenges effectively.

## From Mechanism to Tool:

- When mechanisms are implemented into tools or applications, the implementation may be incomplete or slightly altered to accommodate practical constraints or system limitations. Such adaptations may negatively (insecure) affect the tool's performance, scope, or alignment with the original policy objectives.

## 7.2 Security Policy

## 7.2 Security Policy: Two Categories

**Category 1:** Data and Access Policies: Focus on protecting sensitive data and controlling access to network resources by defining rules for data security, authentication, and authorization.

**Category 2:** Operation and Incident Policies: Emphasize maintaining secure operations, detecting and responding to threats, and ensuring systems recover effectively from security incidents.

We care unauthorized access, misuse, or other malicious activities.

## 7.2 Security Policy: Data and Access Policies

1. Data Protection Policy: Outlines how sensitive data (e.g., personal, financial, or business-critical data) is stored, transmitted, and protected against unauthorized access or leaks.
2. Access Control Policy: Defines rules for granting, monitoring, and revoking user access to the network and resources.
3. Password Management Policy: Enforces password complexity, expiration, and storage requirements to ensure secure authentication practices.
4. Remote Access Policy: Governs secure access to the network from external locations, including VPN usage, device requirements, and user authentication standards.

## 7.2 Security Policy: Operation and Incident Policies

1. Network Security Monitoring Policy: Requires continuous monitoring of network traffic to detect anomalies, threats, and unauthorized access in real time.
2. Incident Response Policy: Specifies procedures for identifying, reporting, responding to, and recovering from security incidents such as breaches, malware infections, or DDoS attacks.
3. Firewall Management Policy: Details the configuration, updating, and auditing of firewalls to control and filter incoming and outgoing network traffic effectively.
4. Patch Management Policy: Establishes procedures for regularly updating and patching software, hardware, and network devices to protect against known vulnerabilities.

## 7.2 Security Policy: Top 3 Keywords

- **Access Control:** Who can access resources and under what conditions.
- **Data:** the integrity and confidentiality of sensitive information.
- **Threats:** potential vulnerabilities and risks to the network.

## 7.2 Security Policy: This topic mainly cares

- Access Control Policy: Defines rules for granting, monitoring, and revoking user access to the network and resources.
- Network Security Monitoring Policy: Requires continuous monitoring of network traffic to detect anomalies, threats, and unauthorized access in real time.

### KEY COMPONENTS OF NETWORK ACCESS CONTROL



## 7.3 Security Mechanism

## 7.3 Security Mechanism: Definition (1/3)

A security mechanism is a set of functions designed to fulfil security policies. It applies (methods) rules and actions (purposes).

If X (rules) are met, then do Y (actions)

Example (Firewall Rule) 1: If a data packet is from an unknown IP address (rule), then block/drop the packet (action). This protects the network from potentially harmful or unauthorized sources.

Example (Failed Login Attempts) 2: If a user fails to log in after three attempts (rule), then lock the account for 5 mins (action). This helps prevent brute-force attacks by limiting repeated access attempts.

## 7.3 Security Mechanism: Definition (2/3)

A security mechanism is a set of tools designed to fulfil security policies . It applies (methods) rules and actions (purposes).

If X (rules) are met, then do Y (actions)

- Rule: define the conditions specifying 'when' actions are taken.
- Action: representing 'what' and 'how' responses or operations will be taken.

## 7.3 Security Mechanism: Definition (3/3)

- Cryptography is a **specialized field of security mechanism** focused on securing data by transforming it so that only authorized parties can understand or use it. It uses mathematical algorithms to provide confidentiality (e.g., encryption), integrity (e.g., hashing), and authenticity (e.g., digital signatures).
- Security mechanisms are broader tools. It may include cryptographic methods but also involve non-cryptographic techniques (firewalls, intrusion detection systems, intrusion prevention, and more) aiming to protect the system as a whole including availability.

## 7.3 Security Mechanism: Data Protection Policy

### 1. Encryption Mechanisms:

- AES (Advanced Encryption Standard): A widely used symmetric encryption algorithm for securing sensitive data at rest and in transit.
- RSA (Rivest-Shamir-Adleman): A public-key encryption system that secures data during transmission and ensures safe key exchanges.

### 2. Data Masking:

Data masking is used to protect sensitive data ensuring that unauthorized users cannot access the original information. At the same time, it keeps applications or processes like testing, development, or training to operate without exposing sensitive data.

## 7.3 Security Mechanism: Access Control Policy

1. **Role-Based Access Control (RBAC)**: Grants user access based on their role within an organization, ensuring users can only access resources necessary for their job.
2. **Attribute-Based Access Control (ABAC)**: Provides access based on user attributes (e.g., department, clearance level) and environmental conditions (e.g., time of day, location).
3. **Network Access Control (NAC)**: Regulates network access by verifying user identity and device compliance against security policies, ensuring only authorized and secure entities can connect to the network.

## 7.3 Security Mechanism: Password Management Policy

1. **Password Hashing Mechanisms:** Hash passwords securely before storage, protecting them against direct retrieval.

2. **Multi-Factor Authentication (MFA):** Requires users to provide two or more verification methods (e.g., password + smartphone OTP) to ensure secure authentication.



## 7.3 Security Mechanism: Remote Access Policy

1. **Virtual Private Network (VPN)**: Encrypts network traffic between remote users and the corporate network, ensuring data privacy and secure communication.
2. **Zero Trust Network Access (ZTNA)**: Involves continuously authenticating and verifying each access request individually, regardless of the user's location, network, or device. It ensures access is granted only to specific resources based on user identity, device health, and security policies, enforcing the principle of "never trust, always verify."

## 7.3 Security Mechanism: Network Security Monitoring Policy

1. Traffic Filtering **Mechanism:** inspects incoming and outgoing network packets based on predefined rules and policies. It controls which traffic is allowed or blocked, ensuring that only authorized data can enter or leave the network. This mechanism acts as a gatekeeper, preventing unauthorized access and protecting network resources from threats such as malicious payloads, viruses, or unauthorized connections.
2. **Intrusion Detection Mechanism:** continuously monitors network traffic for signs of suspicious or malicious activity, such as unauthorized access attempts, attacks, or abnormal traffic patterns. It analyzes traffic in real-time to detect known attack signatures or anomalous behaviors, providing alerts to network administrators when potential security threats are identified.

## 7.3 Security Mechanism: Incident Response Policy

1. **Security Information and Event Management** (SIEM): It collects and analyzes log data from various sources within the network to identify patterns that may indicate a security incident.



## 7.3 Security Mechanism: Firewall Management Policy

1. Next-Generation Firewalls (NGFW): Advanced firewalls that provide deep packet inspection, application filtering, and threat prevention features.

Deep: Incorporating multiple layers of security features

Note: This policy is not created for network security but firewall management.

## 7.3 Security Mechanism: Patch Management Policy

1. **Patch Management Software:** Automates the deployment of patches to operating systems, applications, and devices to address known vulnerabilities.
2. **Vulnerability Scanners:** Identifies outdated software, missing patches, and misconfigurations to ensure timely correction.



## 7.4 Firewalls

Security mechanisms inside firewalls

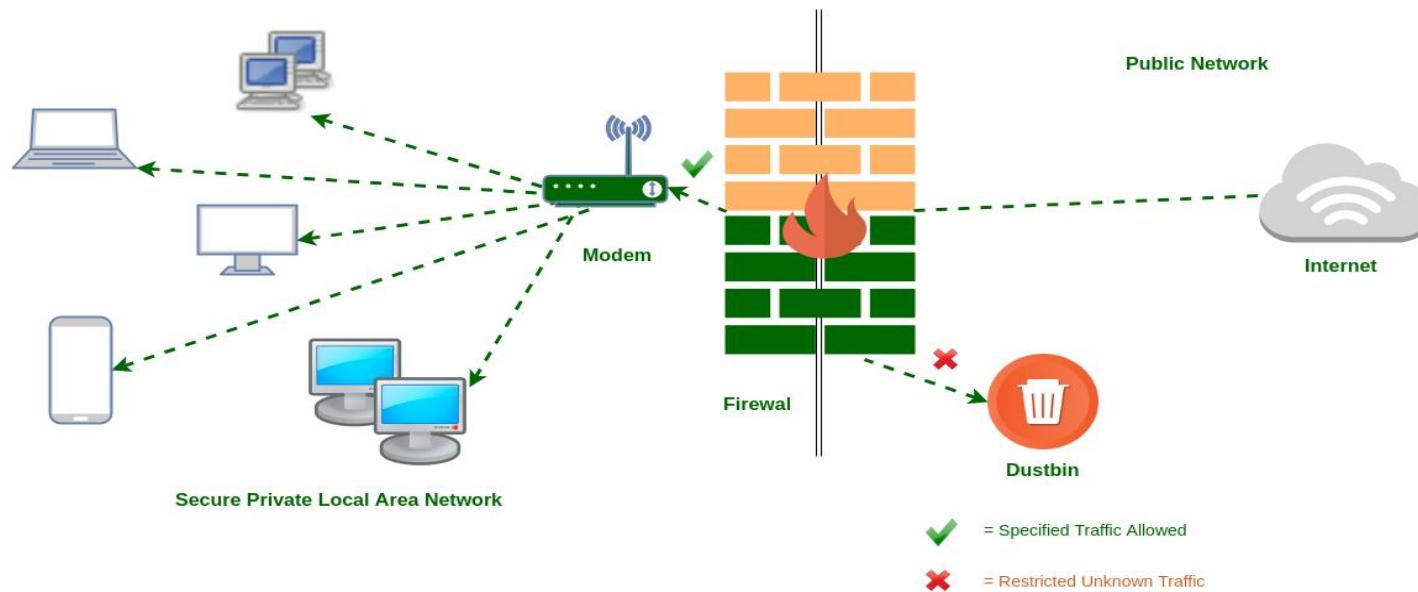
## 7.4 Firewalls: Overview (1/3)

- Firewalls are tools that implement various security mechanisms designed to protect networks by controlling traffic based on predefined rules. These mechanisms typically belong to Access Control Policies and Network Security Monitoring Policies, depending on their specific function.

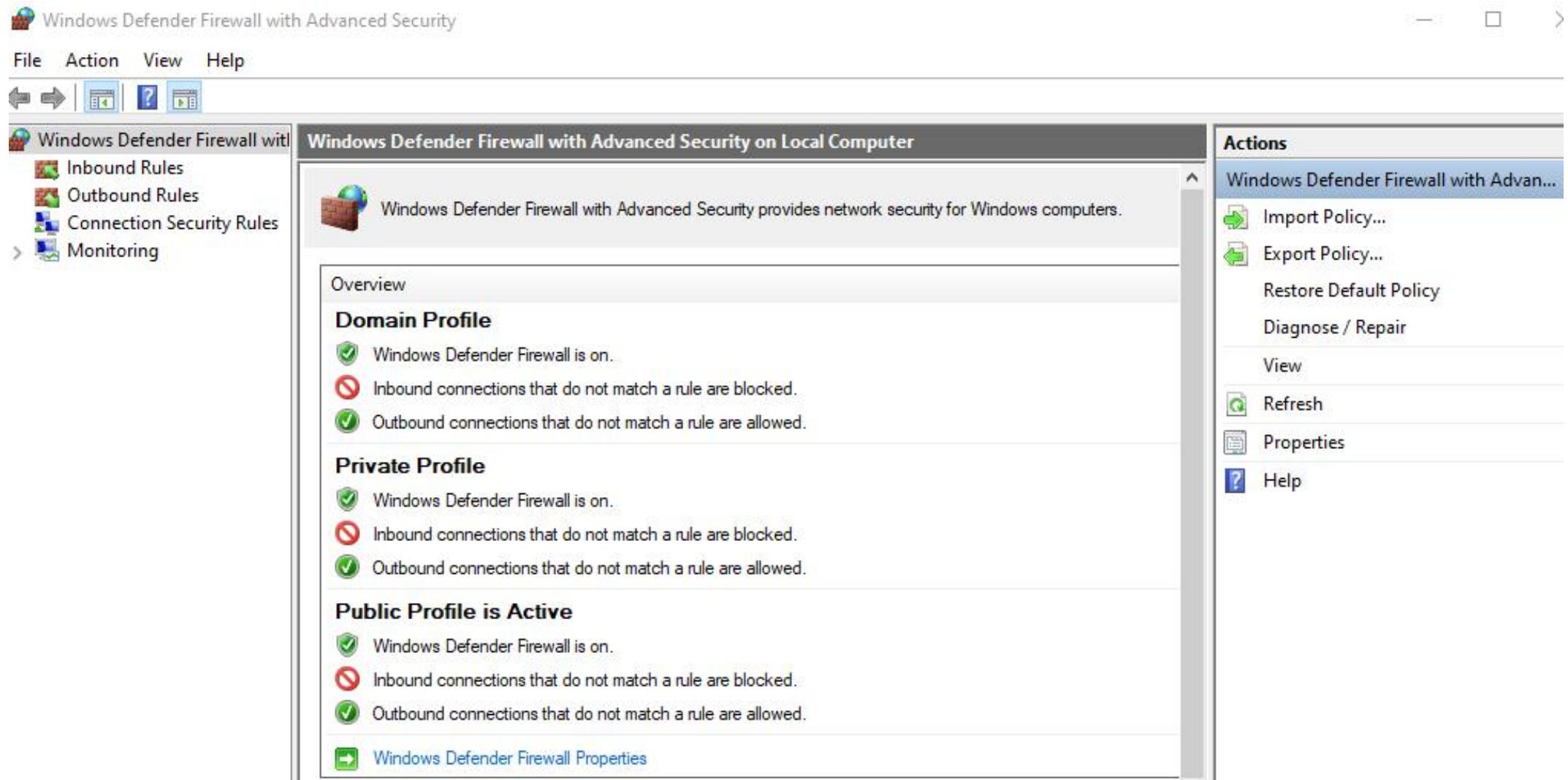


# 7.4 Firewalls: Overview (2/3)

- **Packet Filtering** is the core function of a firewall, where the firewall inspects incoming and outgoing packets based on predefined security rules. It checks information such as IP addresses, ports, and protocols to determine if the packet should be allowed or blocked.



# 7.4 Firewalls: Overview (3/3)



## 7.4 Firewalls: Two Types

Feature	1. Network Firewall	2. Host-Based Firewall
Scope	Protects a network or subnet	Protects a single device (host)
Placement	At the network perimeter or gateway	On the host operating system
Traffic Type	Filters traffic between networks	Filters traffic to/from the host
Administration	Centrally managed	Managed per device

## 7.4 Firewalls: Filtering Criteria (1/4)

- **IP addresses:** Source and destination.

Example: Only devices with IPs in 192.168.1.0/24 are permitted access to internal resources.

- **Port numbers:** Applications or services.

Example: Allow web traffic (HTTP/HTTPS) using port number 443 only while blocking others (80).

- **Protocol types:** TCP, UDP, ICMP.

Example: Deny UDP traffic to limit streaming services on a corporate network.

## 7.4 Firewalls: Filtering Criteria (2/4)

Actions include:

- Allow: Let the packet through.
- Deny/Block: Reject or discard the packet.
- Limit: Restrict the number of packets under the same type.

## 7.4 Firewalls: Filtering Criteria (3/4)

Firewall Rule for (allow or deny): These are syntaxes in high-level representations

[ACTION] [PROTOCOL] [SOURCE IP/SUBNET] [SOURCE PORT] -> [DESTINATION IP/SUBNET] [DESTINATION PORT]

"Inbound" means traffic is entering the host. Destination IP matches the local IP.

INBOUND ALLOW TCP ANY ANY -> 192.168.1.10 3389

"Outbound" means traffic is leaving the host. Source IP matches the local IP.

OUTBOUND BLOCK UDP 192.168.1.10 ANY -> ANY 53

Tips: private IP addresses is a strong indicator of which device or network is local

## 7.4 Firewalls: Filtering Criteria (4/4)

Firewall Rule for (limit):

LIMIT <Protocol> <Source IP> -> <Destination IP> <Port> <Flags> rate <Rate>

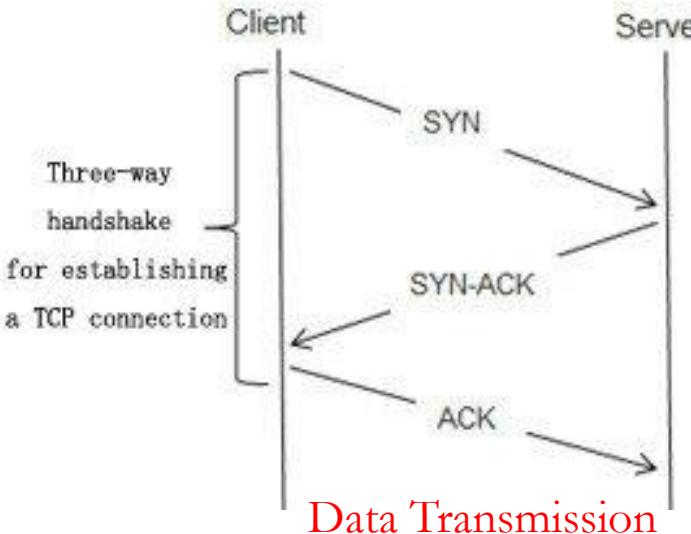
Flags: Specify flag inside the source of protocol

Rate: how many packets per second (or minute) are allowed.

LIMIT TCP ANY -> 192.168.1.1 80 SYN rate 200/s

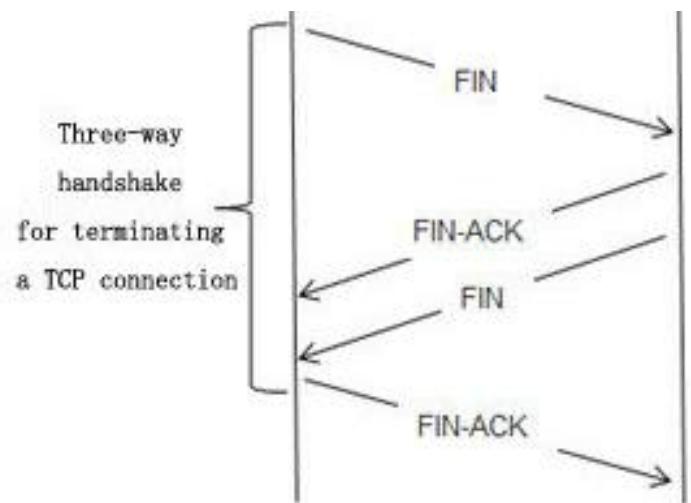
Allow up to 200 packets per second from any source to a web server (port 80)

# 7.4 Firewalls: Applications (1/9)



A SYN Flood attack is a type of Denial of Service (DoS) attack that targets the TCP handshake process. Here's a brief explanation:

In a normal TCP connection, the client sends a SYN (synchronize) request to the server to initiate a connection. The server responds with a SYN-ACK (synchronize-acknowledge), and the client replies with an ACK (acknowledge), completing the three-way handshake.



In a SYN Flood attack, the attacker sends many SYN requests to the server but never completes the handshake by sending the final ACK. The server allocates resources and waits for the ACK that never comes, resulting in a large number of half-open connections. As the server's resources are consumed, it becomes overwhelmed and unable to handle legitimate connections, leading to service disruption.

## 7.4 Firewalls: Applications (2/9)

Firewall Rule against TCP SYN Flood attack:

- Rejecting SYN Packets After a Threshold  
**LIMIT** TCP ANY -> <Host IP> 80 SYN rate 50/s
- Denying SYN Packets from Known Bad Sources  
**DENY** TCP <Malicious IP Range> -> <Host IP> 80 SYN

## 7.4 Firewalls: Applications (3/9)

UDP Flooding (Amplification Attack) is a type of Distributed Denial of Service (DDoS) attack that exploits the stateless nature of the User Datagram Protocol (UDP) and often amplifies the attack using a feature like UDP-based services that respond with larger data than the request sent.

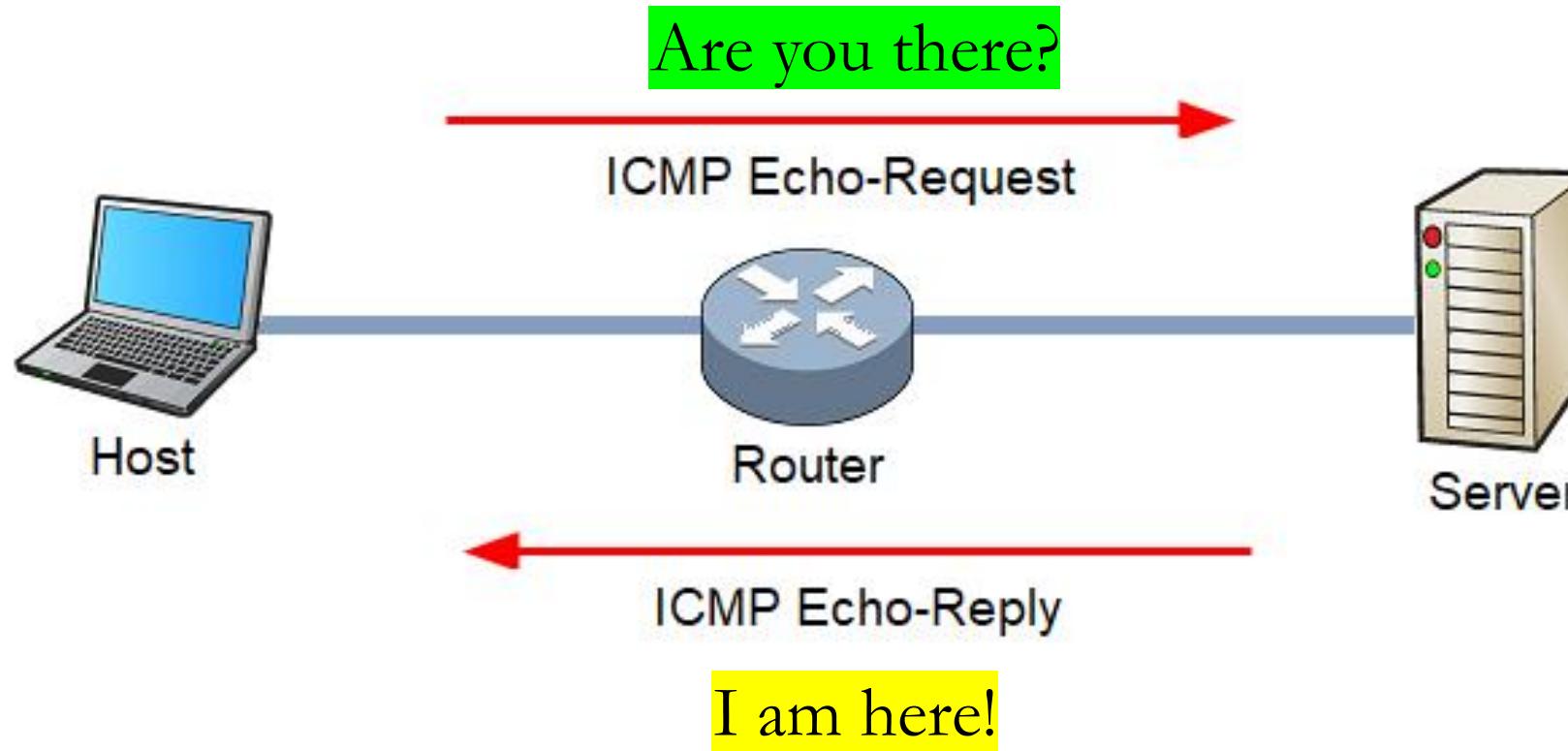
1. **Attacker Sends Small UDP Packets:** The attacker sends a large volume of small UDP packets to various servers. These packets typically request data from services like DNS, NTP, or other UDP-based services that reply with larger responses.
2. **Spoofing the Source IP:** The attacker forges (spoofs) the source IP address of the UDP packets to make it appear as though they are coming from the victim's IP address.
3. **Amplified Response:** The server responds to the spoofed IP with a much larger data payload than the original request. Since UDP is connectionless, the server has no way to verify the legitimacy of the source IP address.
4. **Flooding the Victim:** The victim (not the server)'s network gets overloaded with massive amounts of response data, effectively clogging the network and degrading or taking down services.

## 7.4 Firewalls: Applications (4/9)

Firewall Rule against UDP Flooding (Amplification Attack) :

- Limit incoming UDP traffic to port 53 (DNS):  
LIMIT UDP ANY -> 192.168.1.100 53 rate 10/s
- Drop unnecessary UDP services (e.g., port 123 for NTP):  
DROP UDP ANY -> 192.168.1.100 123
- Block large UDP responses to prevent amplification:  
DENY UDP 192.168.1.100 -> ANY 53 size > 512  
(extra functionality support)

## 7.4 Firewalls: Applications (5/9)



The ping protocol requires the server to unconditionally answer messages --> attack!

## 7.4 Firewalls: Applications (6/9)

### ICMP Flooding (Denial-of-Service Attack):

- Description: Attackers send a high volume of ICMP Echo Request (ping) packets to overwhelm a target system's resources, leading to degraded performance or complete unresponsiveness.
- Impact: This can cause significant network disruption, making services unavailable and impacting overall network stability.

### Ping of Death:

- Description: Exploits vulnerabilities in a system's handling of oversized ICMP packets, which can cause (old systems) crashes (Lack of Size Checking).
- Impact: This can lead to system crashes or reboots, affecting the availability and reliability of network services.

## 7.4 Firewalls: Applications (7/9)

Firewall Rule against ICMP Flooding:

- Limit the number of ICMP echo requests (ping requests) to 1 per second for traffic directed to the IP address  
**LIMIT ICMP ANY -> 192.168.1.100 echo-request rate 1/s**
- Deny all ICMP echo requests (ping requests) directed to the IP address  
**Deny ICMP ANY -> 192.168.1.100 echo-request**
- Limit the size to mitigate Ping of Death

**LIMIT ICMP ANY -> ANY echo-request length 1000-65535**

# 7.4 Firewalls: Applications (8/9)

Firewalls primarily operate at the following two layers

## 1. Internet Layer

Protocols: IP (Internet Protocol), ICMP (Internet Control Message Protocol)

Function: Filters traffic based on IP addresses and network protocols.

## 2. Transport Layer

Protocols: TCP (Transmission Control Protocol), UDP (User Datagram Protocol)

Function: Filters traffic based on port numbers and transport layer protocols.

## 7.4 Firewalls: Applications (9/9)

- A **stateless** firewall, also known as a packet-filtering firewall, operates by inspecting each packet individually without considering the context of the traffic flow. It filters packets based on predefined rules that consider only the packet's source and destination IP addresses, ports, and protocol types.
- Stateless firewalls do not track the state of connections, making them unable to distinguish between legitimate and illegitimate packets that are part of the same session. In a TCP connection, a stateless firewall cannot track the three-way handshake (SYN, SYN-ACK, ACK) process. It treats each packet independently, which can lead to issues in properly managing the connection state and filtering traffic accurately.

## 7.4 Firewalls: Softwares (not important)

- Windows Defender Firewall:

Platform: Windows

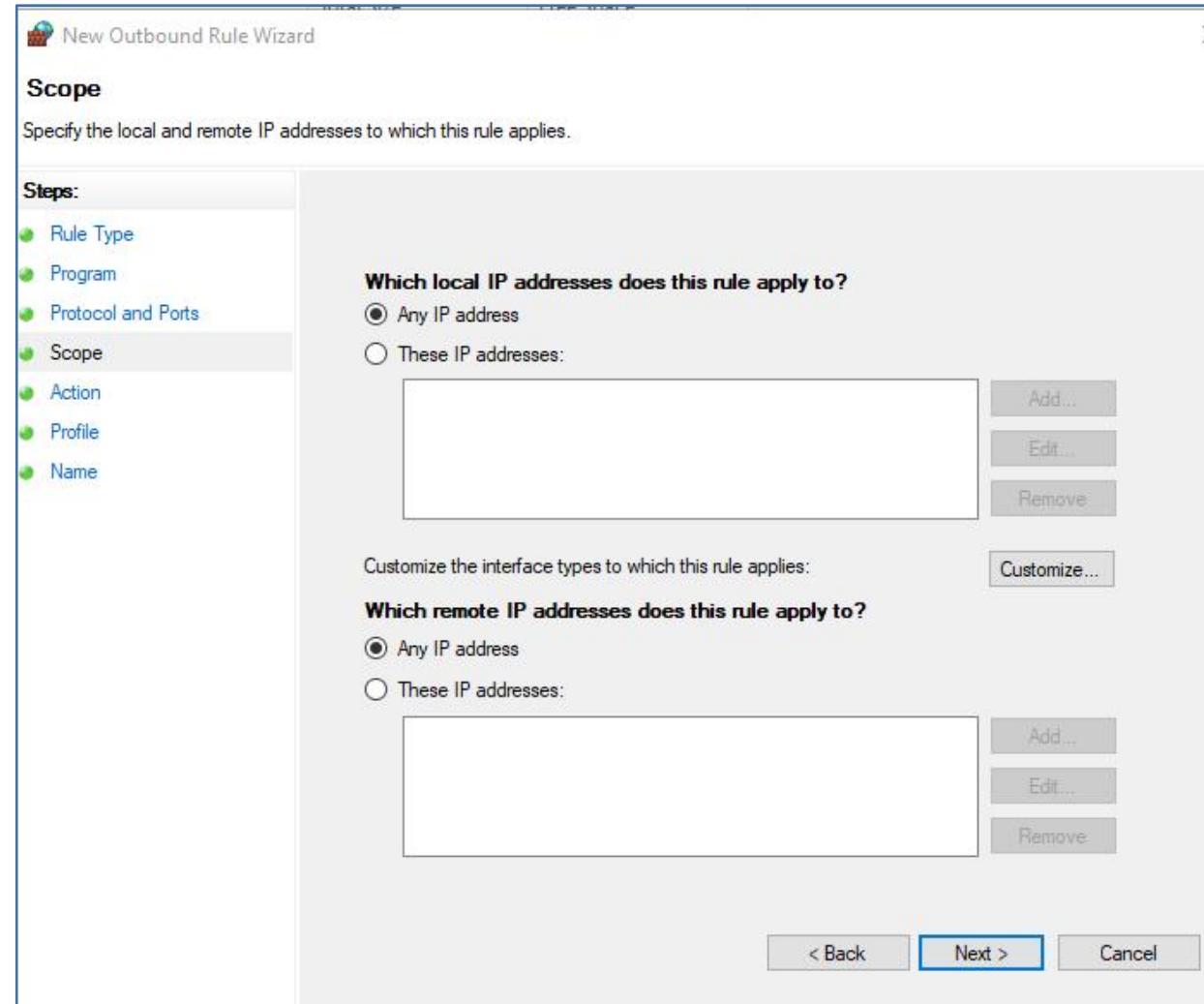
Features: Integrated with Windows OS, easy to use, supports inbound and outbound rules, profiles for different network types (Domain, Private, Public).

- iptables/nftables:

Platform: Linux

Features: Highly flexible and powerful, supports detailed packet filtering, NAT, and rate limiting. nftables is the modern replacement for iptables.

# 7.4 Firewalls: Windows Defender Firewall



## 7.5 IDP

# Intrusion Detection and Prevention

## 7.5 IDP: Detection

An IDS is designed to monitor network traffic or system activity for suspicious behavior, policy violations, or unauthorized access. It functions as a passive monitoring tool, alerting administrators when potential threats are detected but not actively taking measures to stop them. IDS can be categorized into two main types:

- Network-based IDS (NIDS): Monitors network traffic for malicious activities.
- Host-based IDS (HIDS): Monitors activities on individual devices for anomalies.

## 7.5 IDP: Prevention

- An IPS extends IDS functionality by not only detecting potential threats but also actively blocking malicious activities. It operates in-line with network traffic, enabling it to prevent harmful actions by dropping packets, resetting connections, or blocking access based on pre-defined rules.
- IPS also has the firewall functions. However, an IPS does not rely on firewall rules. It uses its own methods (pattern) to identify and block threats, even if those threats bypass the firewall. Together, they provide layered security, but they have distinct roles.

# 7.5 IDP: Against SQL Injection Attack

Description: An SQL injection attack occurs when an attacker exploits a vulnerability in a web application by inserting malicious SQL queries into input fields (e.g., login forms). This allows the attacker to manipulate the backend database, potentially gaining unauthorized access to sensitive data.

## How IDP Stops It:

- **Detection:** The IDP system can detect malicious SQL queries embedded in network traffic, looking for known patterns or suspicious strings (e.g., OR 1=1) **Namely, based on payload.**
- **Prevention:** When an attack is detected, the IDP can block the query or immediately stop the connection (**more than drop packets**) from being processed, preventing the attacker from executing the harmful SQL command.

Example: An attacker tries to inject malicious SQL code into a website's login form to bypass authentication. The IDP system detects the abnormal query structure in the traffic and blocks it before it reaches the backend database.

# **Subject Revision**

# Subject Revision: Two Components

- Network Protocols for Devices Communications
- Security Mechanisms for Improving Network Protocols



# About Final Exam

# About Final Exam

# Thanks for Taking this Subject

