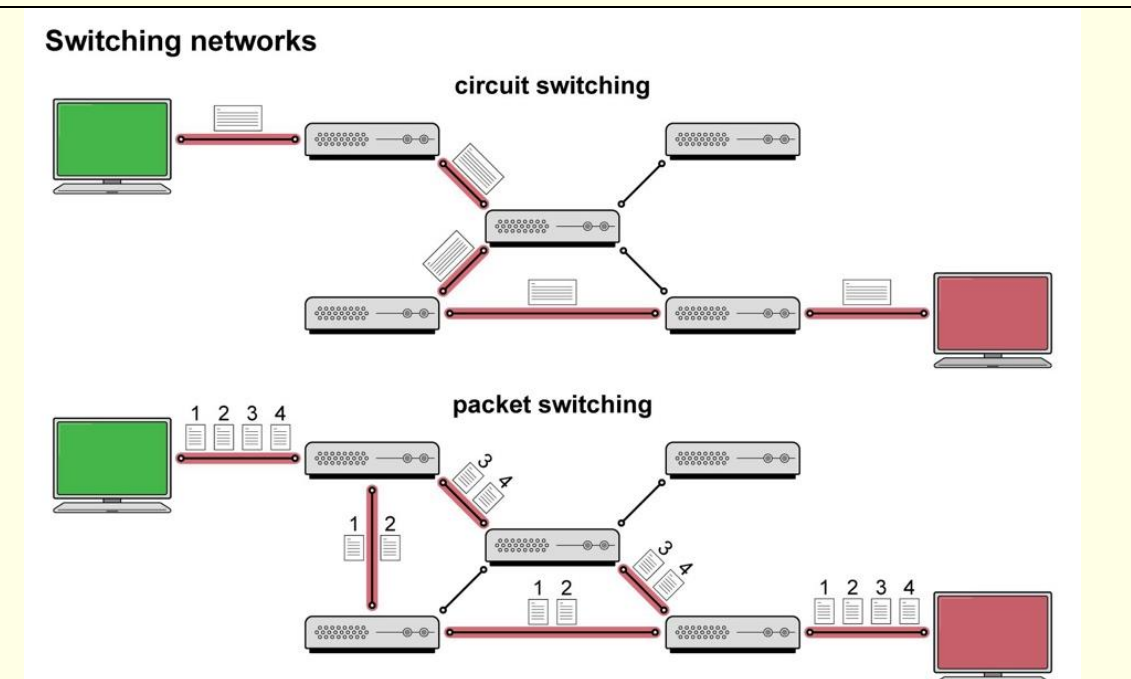


Benefits and Drawbacks of networks

Benefits of a network	Benefit to the Company and employees	Drawbacks
Users on a network can easily share resources such as printers and access to the internet.	A company doesn't have to buy so much equipment	<p>1. Higher Cost – Networks are more expensive to set up than standalone computers because they require network cards, cables, a server, and a Network Operating System.</p> <p>2. Maintenance and Support – Networks are complex and need specialists to set up and maintain them, increasing costs and requiring ongoing management.</p> <p>3. Security Risks – Since multiple machines are connected, hackers can target the network to access sensitive data, making security a bigger concern than with standalone computers.</p> <p>4. Network Failures – If the server goes down in a client-server network, all connected machines may stop working. A single faulty machine or cable failure can also disrupt the entire network.</p>
Users on a network can share files	Team might be working on a project and they can all access and work on the same files easily,	
New software need only be added once and distributed to all other PCs automatically.	Saves time and resources.	
Users can retrieve and work on files from any machine on a network.	If one machine is being used or breaks down, you simply move to a different machine!	
When data files are backed up they only need to be backed-up once, centrally, at the server.	If the files on the network are compromised or a user accidentally deletes a file, it can be restored easiliy from the back up.	
Networks can be managed centrally.	Network manager is able through the network software to control who can access the network, when they can access it, and what files and software and hardware they are allowed to use. Carry out an audit trail of each user.	
Security can be centrally managed by the network manager.	They can add patches centrally, ensure virus patterns are up-to-date centrally and so on.	

Packet transmission

	Packet Switching	Circuit Switching
Definition	Data is broken into packets and transmitted independently .	Data is transmitted in a continuous stream, using a dedicated communication path
Efficiency	Very efficient use of network as each channel only used for short time	Lower efficiency as a dedicated path is maintained even when no data is being transmitted.
Reliability and error detection	If there is an error then only a small, identifiable part of the data is affected. This can be retransmitted easily	Less flexible in handling network failures as the dedicated path, once broken, needs to be re-established.
Interception	Transmission is safer from interception because it is impossible to intercept all the packets as they use different routes	Transmission is less safe from interception because it is using one dedicated data stream.
Use Cases	Best for data that can tolerate some delay, such as emails and web pages.	Ideal for real-time services, like voice calls or video conferencing that require low latency.



- Each packet is typically composed of a header, payload (actual data), and a footer (or trailer)
 - **Source IP address:** identifies the sender of the packet
 - Destination IP address: identifies the intended recipient of the packet
 - **Sequence Number:** helps in reassembling the packets back into the original message at the receiving end
 - **Protocol:** identifies the transport protocol (TCP, UDP, etc.)
 - **Packet Length:** indicates the size of the packet
 - **Checksum:** a value used for error-checking

Steps of Sending a Packet

1. **Data is Split** – The sender's computer breaks data into small **packets**.
2. **Header is Added** – Each packet gets a **source & destination address**, **order number** and **error-checking info**.
3. **Packets are Sent** – They travel through routers, possibly taking different routes.
4. **Packets Arrive** – The receiver **reassembles** them in order.
5. **Error Check & Acknowledge** – The receiver verifies data and requests any missing packets.

Tips For the exam

- **Avoid** talking about the **speed of data transmission** in an answer to a question on packet or circuit switching.
- This will not get you a mark in the exam and, in some questions, is explicitly stated as not worthy of a mark.
- It is better to talk about **higher bit rates** or **bandwidth (the number of bits sent per second)** or **the efficiency of the transmission**

Data Transmission and Error Detection

A **parity bit** is an extra bit added to a group of data bits to help detect errors during transmission. It is a simple form of **error detection** used in digital communication systems.

How Parity Bits Work:

1. **Even Parity:** The parity bit is set so that the total number of 1s in the byte (including the parity bit) is even.
2. **Odd Parity:** The parity bit is set so that the total number of 1s in the byte (including the parity bit) is odd.

Example:

- Suppose we have a **7-bit data**: 1011001
- Using **even parity**, we count the number of 1s (which is 4). Since it's already even, the parity bit is **0**.
- The transmitted byte becomes: **10110010**
- If an error occurs during transmission (e.g., a bit flips), the receiver counts the 1s and checks if the parity is still even. If not, an error is detected.

Limitations:

- Parity bits can detect **single-bit errors** but cannot correct them.
- They fail to detect errors when two bits flip (even number of errors).

Parity checking is a simple but effective way to detect errors in data transmission systems like serial communication, RAM storage, and networking protocols.

Checksum and Sending Bytes in Data Transmission

A **checksum** is an error-detection method used to ensure that data sent over a network or storage system is received correctly. It works by calculating a **sum** of all the bytes before transmission and sending this sum (checksum) along with the data. The receiver then performs the same calculation and checks if the result matches the sent checksum. If they match, the data is correct; if not, an error has occurred.

How Checksum Works with Sending Bytes

Step 1: Sender Computes the Checksum

The sender takes all the data bytes and adds them together.

Given Bytes (8-bit each):

01101101 (109 in decimal)

10000001 (129 in decimal)

10001000 (136 in decimal)

$$109 + 129 + 136 = 374$$

To calculate the checksum:

Add the bytes and ignore the carry to keep the checksum as 8 bits

(1)01110110

The sender sends:

01101101 10000001 10001000 01110110 (checksum)

The checksum value is then sent along with the data.

Step 2: Data and Checksum are Sent

- The data bytes and the checksum are transmitted to the receiver.

Step 3: Receiver Verifies the Checksum

- The same sum is done at the receiving end and the results compared

Why Use Checksum?

- + Detects errors in data transmission.
- + Simple and efficient for detecting single-bit and some multi-bit errors.
- + Cannot **correct** errors—only detects them.

Checksum is widely used in **networking (TCP/IP), file transfers, and storage systems** to ensure reliable data communication.

Serial vs Parallel Transmission

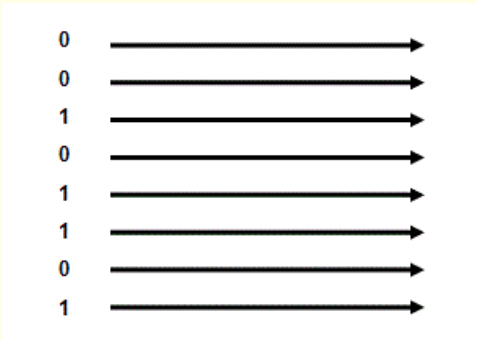
Serial Transmission – Sends data **one bit at a time** using **one wire**.

Example: Sending **10111010** as **1 → 0 → 1 → 1 → 1 → 0 → 1 → 0**

Slower but reliable for long distances.

Parallel Transmission – Sends **multiple bits at the same time** using **multiple wires**.

- **Example:** Sending **10111010** all at once through **8 wires**.
- **Faster** but can cause **errors over long distances** (over 10m).
- Used for **printers & scanners** that need fast data transfer but are close to the computer.




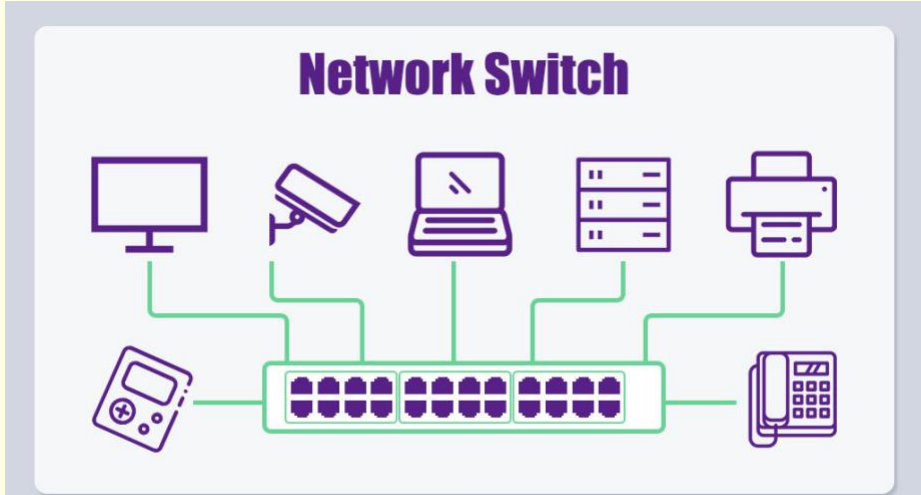



Simplex, Half-Duplex, and Full-Duplex Transmission

Basis for Comparison	Simplex	Half Duplex	Full Duplex
Direction of Communication	Unidirectional	Two-directional, one at a time	Two-directional, simultaneously
Send / Receive	The sender can only send data	The sender can send and receive data, but one a time	The sender can send and receive data simultaneously
Performance	Worst performing mode of transmission	Better than Simplex	Best performing mode of transmission
Example	Keyboard and monitor	Walkie-talkie	Telephone

Networking Hardware

Router	Cable/ Ethernet	Gateway
--------	-----------------	---------

<ul style="list-style-type: none"> • Connect networks together • Assign IP address to devices • Examines data packets and forwards them  <p>Firewall</p> <p>Filters traffic coming in and out of a network</p>	<ul style="list-style-type: none"> • Carries digital data from one device/NIC to the next • Connects wired devices to the network 	<ul style="list-style-type: none"> • Connects different types of network • Translates protocols from one network to another <p>Bridge</p> <p>Provides a link between (local area) networks</p>
<p>Wireless Access Point / WAP</p> <ul style="list-style-type: none"> • Allows wireless devices to communicate with each other • Sends and receives radio waves • Examines data packets and forwards them  <p>Repeater</p> <ul style="list-style-type: none"> • Receives a signal and retransmits it 	<p>Switch</p> <ul style="list-style-type: none"> • Connects multiple wired devices to the network • Receives data and forwards it to the intended recipient • Examines data packets and forwards them • Routes based on MAC addresses 	<p>Network Interface Card / NIC e.g.</p> <ul style="list-style-type: none"> • Gives each device a MAC address / unique ID • Allows a computer system to interface with a network 

Protocols and TCP/IP stack

Remember this definition

A protocol is a **set of rules** and standards that define how data is **transmitted and communicated** between **devices** in a network.

Why?

Setting standards, rules that all manufacturers of hardware and software will follow, are important for a number of reasons:

- Standards describe **accurately** and unambiguously how information is transmitted.
- A manufacturer's products will work successfully with other manufacturer's products if they all follow the same standards.
- By defining a set of standards, you are providing a framework within which all manufacturers can design new, successful products.
- Standards break down complex ideas into smaller, methodical, easier-to-understand components.

Successful communication includes:

- What **baudrate** will be used - **It determines how fast data is sent and received.**
- What error checking will be used - **Checksums** or **Parity Checking**
- Whether software or hardware 'handshaking' is to be used - Handshaking is the process of **establishing communication** between devices
- What character set is to be used. (**ASCII** or **UNICODE**)
- How many bits will be used for data.
- How many control bits will be used to control data transfer (**Start Bit** – Signals the beginning of a data packet or a **Stop Bit** – Marks the end of the data packet)

A **logical protocol** refers to a set of rules and standards that define how data is transmitted, processed, and understood within a network at a logical level, without focusing on the physical hardware or infrastructure. For example: **IP, TCP, HTTP.**

Physical protocols define how data is actually *transmitted* through the physical medium Deals with the **hardware** and **physical signals**. For example: **Ethernet and Wi-Fi.**

HTTP (Hypertext Transfer Protocol) – Sends web pages over the internet at the application layer.

• **HTTPS (Secure HTTP)** – Encrypts HTTP data using TLS/SSL for secure web browsing.

• **FTP (File Transfer Protocol)** – Transfers files between a client and server; FTPS adds security.

• **TCP (Transmission Control Protocol)** – Ensures reliable, ordered data transmission.

• **UDP (User Datagram Protocol)** – A faster alternative to TCP. It focuses on ensuring that data is transmitted efficiently, making it ideal for applications like video streaming, online gaming, and VoIP calls, where speed is more important than ensuring every single packet is received correctly

• **IP (Internet Protocol)** – Handles addressing and routing of data packets over networks.

• **SMTP (Simple Mail Transfer Protocol)** – Sends emails between clients and servers.

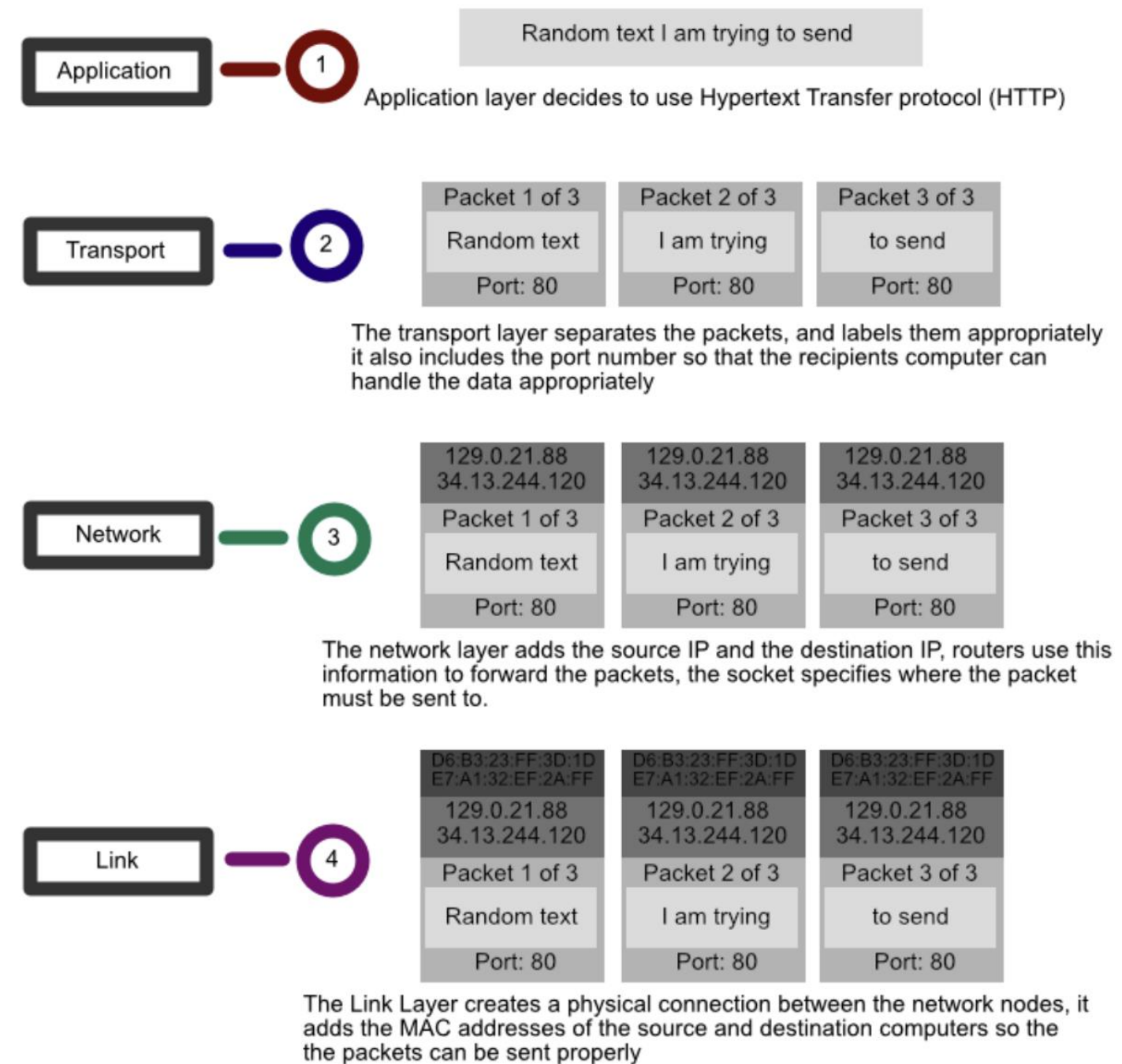
• **POP3 (Post Office Protocol v3)** – Downloads emails from a server to a local device.

• **IMAP (Internet Message Access Protocol)** – Allows email access and sync across multiple devices.

TCP/IP (Transmission Control Protocol / Internet Protocol) is a set of networking protocols that work together to send data across networks.

It is divided into **four layers**:

Application Layer	The top layer that determines the correct protocol based on the application (e.g., HTTP for web browsing, FTP for file transfers).
Transport Layer	<ul style="list-style-type: none"> • Uses TCP to establish a connection between sender and receiver. • Splits data into packets, labels them with numbers and port details. • Requests retransmission if packets are lost.
Network Layer	<ul style="list-style-type: none"> • Adds IP addresses of the sender and recipient to ensure correct delivery. • Routers use these IPs to forward packets. • The combination of IP address + port number is called a socket address.
Link Layer	<ul style="list-style-type: none"> • Handles the physical connection between devices. • Adds MAC addresses to identify the sender and recipient's network devices. • If the recipient is on a different network, the packet goes to the router's MAC address first.



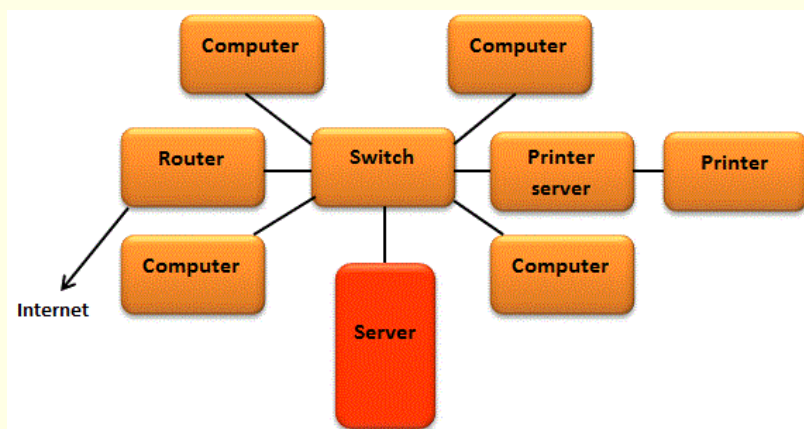
Client-server and Peer-to-Peer

Client-server	Peer-to-Peer
<p>One computer acts as a server (file server).</p> <p>Server:</p> <ul style="list-style-type: none"> • Manages network security. 	<ul style="list-style-type: none"> • No server—all computers have equal status (peers). • Each computer can:

- Stores user login details and access rights.
- Controls access to software, files, and hardware (e.g., printers).
- Has large storage (hard disk) and high RAM for multiple tasks.

Clients:

- Are the computers used by authorised users.
- Users log in to access files and applications.
- Users can usually log in from any client on the network.
- Access rights determine what a user can do (e.g., teachers vs. students).



Pros

- **Centralized Management:** Easier to manage security, updates, and resources from a single server.
- Centralized security and access control.
- Servers handle complex tasks, allowing clients to perform more efficiently.

Cons

- If the server goes down, all clients are affected.
- High hardware and maintenance costs for servers.
- Requires specialist staff
- Requires a stable network for communication; slow networks reduce performance.

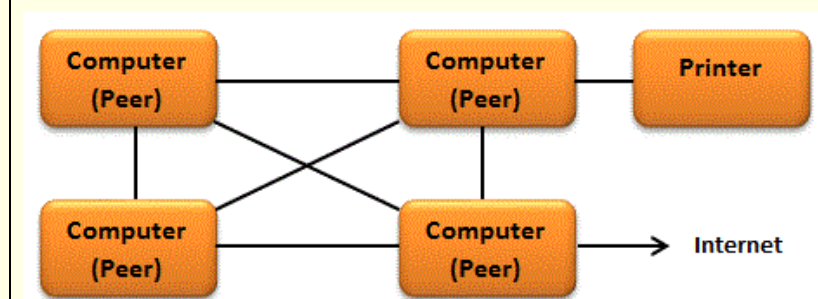
- Share files with others.
- Send print jobs to another computer with a printer.

Pros

- Simple to set up—best for small networks (10 computers or fewer).
- No need for a dedicated server.

Cons

- Less security—each computer must manage its own security.
- Difficult backups—files must be backed up individually on each computer.
- Lacks advanced features found in client-server networks.



Network Security

Issue	Description	Prevention
Malware (Viruses, Worms, Trojans)	Malicious software that can damage or steal data.	Install and update antivirus software, avoid suspicious downloads.

Phishing Attacks	Fraudulent emails or websites trick users into revealing sensitive information.	Educate users, use email filters, verify sources before clicking links.
Denial of Service (DoS) & Distributed DoS (DDoS)	Overloading a network or server to disrupt services.	Use firewalls, rate limiting, and DDoS protection tools.
Man-in-the-Middle (MITM) Attacks	Attackers intercept communication to steal or alter data.	Use encryption (SSL/TLS), VPNs, and secure authentication methods.
Unauthorized Access	Hackers gaining access to sensitive systems or data.	Implement strong passwords, multi-factor authentication (MFA), and access control.
Data Breaches	Unauthorized data access leading to leaks or theft.	Encrypt sensitive data, regularly update security policies, and monitor access logs.
Ransomware Attacks	Hackers encrypt data and demand payment for release.	Maintain regular backups, update software, and avoid suspicious email attachments.
Weak Passwords	Easily guessed passwords lead to unauthorized access.	Enforce strong password policies and MFA.
Wi-Fi Security Threats	Unsecured Wi-Fi networks allow unauthorized access.	Use WPA3 encryption, strong passwords, and hide SSID if necessary.
Social Engineering	Manipulating people to gain access to confidential information.	Conduct security awareness training and verify unknown requests.
Outdated Software & Patches	Unpatched vulnerabilities can be exploited by attackers.	Regularly update and patch all software and hardware.

Keeping Data Secure

The **Data Protection Act 1998** requires organisations to keep data safe. Security measures include **firewalls, proxy servers, encryption, and authentication.**

Firewall: A firewall **blocks unauthorised access** to a system. It allows only approved users to access certain data while keeping others out.

Proxy Server: A **proxy server** sits between users and the main server. It checks user permissions before granting access to data, adding an extra layer of security.

How it Works:

- 1. A user requests data.
- 2. The **firewall** checks if they have permission.
- 3. If valid, the **proxy server** retrieves the data and sends it back.
- 4. Users cannot access the main server directly.

VPN

Public Wi-Fi is risky, so encrypt your communications for safety.

Best solution: Use a VPN (Virtual Private Network).

- ✓ Encrypts all your internet activity.
- ✓ Hides your IP address (useful for accessing geo-restricted content).
- ✓ Costs just a few pounds per month.

How it works:

- 1. Install VPN software.
- 2. Connect to a public network.
- 3. VPN verifies your identity and encrypts data.

No system is 100% secure!

- ✓ Check VPN terms—some log data.
- ✓ Employees at VPN companies could still pose risks.

DNS Resolution Process:

The **Domain Name System (DNS)** translates human-readable website addresses (URLs) into numerical **IP addresses**, which computers use to communicate over the internet. The process involves multiple steps:

Step	Action
------	--------

1. User enters URL	The browser sends a request to resolve the domain name to an IP address.
2. DNS Resolver Checks Cache	If the IP is found in the local cache, it's returned immediately. Otherwise, the request continues.
3. Query to TLD Name Server	If the resolver lacks the IP, it queries the TLD Name Server (e.g., .com, .org) .
4. Query to Authoritative Name Server	If necessary, the request is forwarded to the Authoritative Name Server holding the actual IP.
5. IP Address Returned	The authoritative server responds with the correct IP address.
6. Browser Connects to Website	The browser uses the IP to establish a connection and load the web page.
7. Error Handling	If no match is found, an error is returned (e.g., "Site Not Found").

