

Q1) What is Cyber Security?

Cyber Security is the practice of protecting computer systems, networks, applications, and data from unauthorized access, attacks, damage, or theft.

Its main goal is to ensure that information remains **safe, accurate, and accessible only to authorized users.**

Cyber security is built on three core principles known as the **CIA Triad:**

1. Confidentiality

Confidentiality ensures that sensitive information is accessed **only by authorized individuals.**

How it is achieved:

- Passwords and PINs
- Encryption
- Access control and authentication (OTP, biometrics)

Real-World Examples:

Banking

- Your bank account details, PIN, and transaction history are encrypted.
- Only you (using login credentials and OTP) can view or transfer money.
- If hackers steal login credentials, confidentiality is compromised.

2. Integrity

Integrity ensures that data remains **accurate, complete, and unaltered** unless modified by authorized users.

How it is achieved:

- Hashing
- Digital signatures
- Audit logs

Real-World Examples:

Banking

- If you transfer ₹5,000, the amount should not change during the transaction.

- Any unauthorized change in balance or transaction records violates integrity.

3. Availability

Availability ensures that systems, services, and data are **accessible when needed** by authorized users.

How it is achieved:

- Backup systems
- Redundancy
- Protection against DDoS attacks

Real-World Examples:

Banking

- Online banking and ATMs should be available 24/7.
- If servers go down due to a cyberattack, customers cannot access their money.

Q2) Types of Cyber Attackers

- Cyber attackers are individuals or groups that attempt to gain unauthorized access to systems, networks, or data. Their **skills, motivations, and targets** differ widely.

1. Script Kiddies

Who they are:

- Inexperienced attackers with **limited technical knowledge**
- Use **pre-written scripts, tools, or exploits** created by others

Real-World Example:

- A beginner using tools like **LOIC** or **Metasploit modules** to attack a website without understanding how they work.

2. Insider Threats

Who they are:

- Employees, contractors, or partners with **legitimate access**
- Can be **malicious or accidental**

Real-World Example:

- An employee copying sensitive data to a USB drive and selling it.
- Accidentally exposing data by misconfiguring cloud storage.

3. Hacktivists

Who they are:

- Attackers driven by **political, social, or ideological beliefs**

Real-World Example:

- Groups like **Anonymous** targeting government or corporate websites.

4. Nation-State Actors

Who they are:

- Highly skilled attackers **sponsored by governments**
- Operate with advanced tools and long-term strategies

Real-World Example:

- **Stuxnet** attack on Iranian nuclear facilities.
- APT groups targeting defense or energy sectors.

Q3) What is an Attack Surface?

- An **attack surface** is the **sum of all possible points** where an attacker can try to enter, exploit, or extract data from a system.
The larger the attack surface, the **higher the risk of cyber attacks**.

1. Web Applications

Description:

Web applications are software programs accessed through a browser (e.g., banking portals, e-commerce sites).

Common Vulnerabilities:

- SQL Injection
- Cross-Site Scripting (XSS)

- Cross-Site Request Forgery (CSRF)
- Broken authentication

Real-World Examples:

- Online banking websites
- Shopping websites like Amazon

2. Mobile Applications

Description:

Apps installed on smartphones that interact with backend servers.

Common Vulnerabilities:

- Insecure data storage
- Weak authentication
- Hard-coded API keys
- Insecure communication (no HTTPS)

Real-World Examples:

- UPI apps
- Social media apps

3. APIs (Application Programming Interfaces)

Description:

APIs allow different software systems to communicate with each other.

Common Vulnerabilities:

- Broken Object Level Authorization (BOLA)
- Lack of rate limiting
- Improper authentication
- Exposed API endpoints

Real-World Examples:

- Payment gateway APIs

- Social media login APIs

4. Networks

Description:

Networks include internal and external communication channels such as LAN, WAN, and the internet.

Common Vulnerabilities:

- Open ports
- Weak firewall rules
- Unpatched routers and switches
- Man-in-the-Middle (MITM) attacks

Real-World Examples:

- Corporate office networks
- Public Wi-Fi

5. Cloud Infrastructure

Description:

Cloud infrastructure includes servers, storage, databases, and services hosted on platforms like AWS, Azure, or GCP.

Common Vulnerabilities:

- Misconfigured storage (public S3 buckets)
- Weak IAM policies
- Exposed management consoles
- Insecure APIs

Real-World Examples:

- Cloud-hosted websites
- SaaS applications

Q4) OWASP Top 10: 2025 – Most Critical Web Application Risks

- The **OWASP Top 10** is a globally recognized list of the **most serious and common security issues in web and modern software applications**. It's updated by analyzing real-world vulnerabilities and expert input.

A01: Broken Access Control

- **What it is:**
Flaws that let attackers bypass authorization and perform actions they shouldn't be able to — like viewing or editing another user's data.

A02: Security Misconfiguration

- **What it is:**
Incorrect or unsafe configurations in apps, servers, or cloud services.

A03: Software Supply Chain Failures

- **What it is:**
Problems originating from **external dependencies** — libraries, tools, CI/CD pipelines, build systems, or third-party code — that get integrated into your software.

A04: Cryptographic Failures

- **What it is:**
Weak or missing encryption, poor key management, or outdated cryptographic methods

A05: Injection

- **What it is:**
Occurs when untrusted input is executed as code or commands — e.g., SQL, OS, template, or expression injections.

A06: Insecure Design

- **What it is:**
Weak **architectural decisions** or missing security planning before coding starts.

A07: Authentication Failures

- **What it is:**
Problems in login, session handling, or credential mechanisms.

A08: Software or Data Integrity Failures

What it is:

Failures to verify that **software, updates, or data haven't been tampered with**.

A09: Logging & Alerting Failures

What it is:

Inadequate logging and lack of alerts for critical security events.

A10: Mishandling of Exceptional Conditions (New in 2025)

What it is:

Improper handling of errors, logic bugs, and *abnormal states* in code.

Q5) Mapping Daily-Used Applications to Attack Surfaces

Modern applications interact with multiple components such as **web apps, mobile apps, APIs, networks, and cloud infrastructure**. Each component exposes an **attack surface** that attackers can target.

1. Email Applications (Gmail, Outlook, Yahoo Mail)

Possible Attack Surfaces:

- **Web Application** – Browser-based email access
- **Mobile Application** – Email apps on smartphones
- **APIs** – Mail sync and third-party integrations
- **Network** – Internet, public Wi-Fi
- **Cloud Infrastructure** – Email servers and storage

Common Attacks:

- Phishing emails
- Credential theft
- Session hijacking
- Malware attachments

Example:

- Clicking a phishing link that steals login credentials via a fake login page.

2. WhatsApp / Messaging Applications

Possible Attack Surfaces:

- **Mobile Application** – Installed app on phone
- **APIs** – Message delivery, media upload
- **Network** – Mobile data, Wi-Fi
- **Cloud Infrastructure** – Backup storage (Google Drive/iCloud)

Common Attacks:

- Account takeover via OTP interception
- Malicious links and scam messages
- Exploiting app vulnerabilities

Example:

- Attacker tricks user into sharing OTP and hijacks the WhatsApp account.

3. Banking Applications (Mobile Banking / Net Banking)

Possible Attack Surfaces:

- **Mobile Application** – Banking app
- **Web Application** – Internet banking portal
- **APIs** – Payment gateways, UPI, fund transfer APIs
- **Network** – Public Wi-Fi, insecure networks
- **Cloud Infrastructure** – Core banking backend

Common Attacks:

- Phishing and fake banking apps
- Man-in-the-Middle (MITM) attacks
- Malware stealing credentials

Example:

- User installs a fake banking app that captures login credentials and OTPs.

Q6) Data Flow: User → Application → Server → Database

Data flow explains **how information travels** through a system when a user performs an action (login, message, payment, etc.).

1. User

Role:

- The **user** is the starting point of data flow.
- Interacts with the system using a **browser or mobile app**.

Examples:

- Entering email and password
- Sending a WhatsApp message
- Making an online payment

Data Generated:

- Login credentials
- Messages
- Transaction details

2. Application (Frontend)

Role:

- The **application interface** collects user input.
- Performs **basic validation** before sending data.

Examples:

- Web app (HTML, CSS, JavaScript)
- Mobile app (Android / iOS)

Actions:

- Checks required fields
- Encrypts data using HTTPS
- Sends request to backend via API

3. Server (Backend)

Role:

- The **server processes requests** and enforces business logic.

Examples:

- Authentication server
- Application server (Java, Python, Node.js)

Actions:

- Validates user credentials
- Applies authorization rules
- Logs activities
- Communicates with database

4. Database

Role:

- **Stores and retrieves data securely.**

Examples:

- User credentials
- Messages
- Transaction records

Actions:

- Executes queries (SELECT, INSERT, UPDATE)
- Returns results to server
- Ensures data integrity

Reverse Flow (Response Back to User)

1. Database sends requested data to the **server**
2. Server processes and formats the response
3. Application displays results to the **user**

Q7) Identify where attacks can happen during this flow

Attack Points in the Data Flow

Cyber attacks can occur at **every stage** of the data flow if proper security controls are missing.

1. User Level

Where attacks happen:

- User device (laptop, mobile)
- User behavior

Common Attacks:

- Phishing attacks
- Malware / keyloggers

- Social engineering
- Credential theft

Example:

- User enters credentials on a fake banking website.

Impact:

- Account compromise
- Unauthorized access

2. Application (Frontend) Level

Where attacks happen:

- Web or mobile app interface
- Client-side code

Common Attacks:

- Cross-Site Scripting (XSS)
- Fake or tampered mobile apps
- Client-side validation bypass

Example:

- Injecting malicious JavaScript into a vulnerable comment field.

Impact:

- Session hijacking
- Data theft

3. Network Level (In Transit)

Where attacks happen:

- Internet connections
- Public Wi-Fi networks

Common Attacks:

- Man-in-the-Middle (MITM)
- Packet sniffing
- DNS spoofing

Example:

- Attacker intercepts data on public Wi-Fi without HTTPS.

Impact:

- Credential exposure
- Data manipulation

4. Server (Backend) Level

Where attacks happen:

- Application servers
- Authentication & authorization logic

Common Attacks:

- Broken access control
- Injection attacks (SQL/Command)
- Authentication bypass

Example:

- Attacker accesses admin APIs without proper authorization.

Impact:

- Privilege escalation
- Full system compromise

5. API Level

Where attacks happen:

- Exposed backend APIs

Common Attacks:

- Broken Object Level Authorization (BOLA)
- Excessive data exposure
- Lack of rate limiting

Example:

- Changing user ID in API request to access another user's data.

Impact:

- Mass data leakage
- Account takeover

6. Database Level

Where attacks happen:

- Database queries
- Database access controls

Common Attacks:

- SQL Injection
- Unauthorized database access
- Data exfiltration

Example:

- Attacker dumps entire user database using SQL injection.

Impact:

- Data breach
- Loss of integrity

7. Logging, Monitoring & Error Handling

Where attacks happen:

- Poor logging and alerting
- Verbose error messages

Common Attacks:

- Stealth attacks going unnoticed
- Information disclosure via error messages

Example:

- Error message reveals database structure.

Impact:

- Delayed detection

- Easier exploitation

Q8) Summarize the entire understanding in your own words for clarity

Cybersecurity is all about protecting data and systems so that information stays **private (confidentiality)**, **accurate (integrity)**, and **available when needed (availability)**. These three principles, known as the **CIA Triad**, form the foundation of all security practices we see in daily life—whether we are using banking apps, email, or social media. Different types of attackers target systems in different ways. **Script kiddies** use ready-made tools with little understanding, **insiders** misuse or accidentally expose systems they already have access to, **hacktivists** attack for ideological reasons, and **nation-state actors** carry out highly sophisticated attacks for espionage or warfare. Knowing who the attacker might be helps in understanding the level of risk. Modern applications expose many **attack surfaces**, such as **web applications, mobile apps, APIs, networks, and cloud infrastructure**. Everyday apps like email, WhatsApp, and banking apps are not just a single system—they rely on multiple components, each of which can be attacked if not secured properly. Data typically flows from the **user → application → server → database and back**. At each stage, there is a chance for attacks to occur. For example, users can be tricked through phishing, applications can suffer from vulnerabilities like XSS, networks can be attacked using Man-in-the-Middle techniques, servers can be exploited through broken access control or injection attacks, APIs can leak data due to poor authorization, and databases can be compromised if inputs are not handled securely. The **OWASP Top 10 (latest)** highlights the most dangerous and common vulnerabilities that attackers exploit, such as broken access control, insecure design, injection flaws, and poor logging. These issues are dangerous because they can lead to **data breaches, account takeovers, financial loss, and service outages**. In simple terms, cybersecurity requires a **layered approach**—securing users, applications, networks, servers, and data together. Even if one layer fails, other controls should prevent a full compromise. Understanding how data flows and where attacks can happen makes it easier to design, defend, and monitor systems effectively.