**Chapter: Introductory class**

1. CIA Triad

**Confidentiality**

**Definition**: Ensuring that sensitive data is only accessible to those authorized to see it.

* ✅ **Goal**: Prevent unauthorized access.
* 🔒 **Methods**: Encryption, access controls, authentication, and classification of data.

**🧠 Example**:  
A company's HR database contains employee salaries. Access is restricted so only HR personnel can view them. If someone outside HR tries to access it and fails because of access controls, confidentiality is preserved.

**Integrity**

**Definition**: Ensuring that data is accurate, consistent, and hasn’t been tampered with.

* ✅ **Goal**: Maintain trustworthy data.
* 🔒 **Methods**: Hashing, digital signatures, checksums, and version control.

**🧠 Example**:  
An online banking transaction must not be altered during transfer. If Alice sends $100 to Bob, integrity ensures the amount isn’t changed to $1000 by a malicious actor. A cryptographic hash validates that the message hasn’t been changed.

**Availability**

**Definition**: Ensuring that systems and data are available to authorized users when needed.

* ✅ **Goal**: Minimize downtime.
* 🔒 **Methods**: Redundancy, backups, disaster recovery, and DDoS protection.

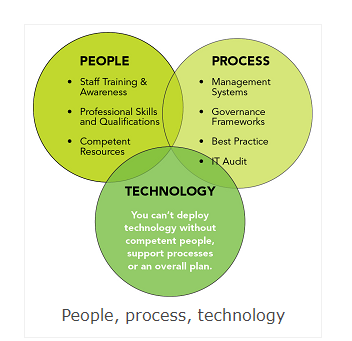
**🧠 Example**:  
An e-commerce site going down on Black Friday would cost millions. To ensure availability, the company uses load balancers, redundant servers, and cloud scaling to handle traffic and prevent outages.

**💡 Real-World Scenario: Online Banking System**

| **CIA Principle** | **Applied Example in Banking** |
| --- | --- |
| Confidentiality | Users must log in with 2FA to view account info. |
| Integrity | Transaction logs are hashed to detect tampering. |
| Availability | Redundant servers ensure 24/7 banking access. |

1. What is Cyber Security? Write about Three Pillars of Cyber Security.

Cyber security is the practice of defending computers, servers, mobile devices, electronic systems, networks, and data from malicious attacks.



1. What types of components are considering some (ICT Indices) organization to published the ranking?

|  |  |  |
| --- | --- | --- |
| Organization | Components | Ranking |
| Networked Readiness Index (NRI) | 1. Environment for ICT (market, political and regulatory, infrastructural environment), 2. The Stakeholders (individuals, businesses and government) to use ICT, and 3. The usage of ICT. | In 2024, Bangladesh is in 91th position. |
| ICT Development Index (IDI) | 1. ICT readiness (Infrastructure, access) 2. ICT Use (Intensity) 3. ICT capability (Skills) | In 2023, Bangladesh score is 61.1 |
| E-Government Development Index (EGDI) | 1. Online Service 2. Technological infrastructure 3. Human Capital | In 2024, Bangladesh is 111th position). |
| Global Cybersecurity Index (GCI) | 1. Legal measures 2. Technical measures 3. Organizational measures 4. Capacity development measures 5. Cooperation measures | In 2024, Bangladesh is 53th position). |
| National Cyber Security Index (NCSI) | 1. Legislation in force – Legal acts, regulations, orders, etc., 2. Established units – Existing organizations, departments, etc., 3. Cooperation formats – Committees, working groups, etc. and, 4. Outcomes – Policies, exercises, technologies, websites, programs, etc. | In 2024, Bangladesh s is 36th position. |

1. Dimensions of cybersecurity: Capacity maturity Model (CMM)
   1. Developing Cyber Security Policy & Strategy
   2. Encouraging responsible Cyber Security culture within society
   3. Building Cyber Security knowledge & capacity
   4. Creating effective legal & regulatory frameworks
   5. Controlling risks through standard & technologies

**Chapter: Introduction to Secure Software Development:**

1. Overview of secure software development principles

Secure software development involves embedding security throughout the Software Development Life Cycle (SDLC). This proactive approach ensures that security is not an afterthought but a core component of software design and implementation. Key principles include:

* **Security Requirement Engineering (SRE)**: Incorporating security requirements from the planning phase.
* **Secure Design Patterns**: Applying best practices like input validation, encryption, and proper authentication during the design phase.
* **Secure Coding Practices**: Reviewing code with a focus on security during development.
* **Dynamic Application Security Testing (DAST)**: Testing running applications to identify vulnerabilities.
* **Secure Deployment**: Ensuring that configurations are safe and secure at the deployment stage.
* **Continuous Monitoring and Maintenance**: Regular patching, auditing, and incident response planning post-deployment.

“Integrating security into the SDLC is not optional—it is a necessity in modern software development.”

1. Importance of integrating security into the software development life cycle (SDLC).

The slides highlight several reasons why security should be integrated into every phase of the SDLC:

1. **Early Detection and Mitigation of Vulnerabilities**
   * Fixing security issues early is far cheaper than post-deployment fixes.
   * Helps prevent critical flaws from reaching production.
2. **Reduced Cost of Fixes**
   * Addressing issues early avoids costly remediation, lawsuits, or data breach penalties.
   * Supported by industry reports (e.g., IBM’s Cost of a Data Breach).
3. **Enhanced Regulatory Compliance**
   * Helps meet strict industry regulations (e.g., GDPR, HIPAA, PCI-DSS).
   * Reduces the risk of non-compliance penalties.
4. **Protection Against Emerging Threats**
   * Involves proactive threat modeling and risk assessment.
   * Enables anticipation and mitigation of attack vectors.
5. **Improved Software Quality and Reliability**
   * Secure coding reduces bugs and performance issues.
   * Enhances system stability and resilience.
6. **Increased Customer Trust and Business Reputation**
   * Demonstrates commitment to user data protection.
   * Builds credibility and attracts more users.
7. How to integrate the security into the SDLCs?

­­­ To effectively integrate security into each phase of the SDLC, consider the following key practices:

1. Requirement Analysis (Planning Phase)

 Use Security Requirement Engineering (SRE) frameworks.

2. Design Phase

 Implement secure design patterns (e.g., input validation, encryption, authentication).

3. Development Phase

 Implement code reviews with security focus.

4. Testing Phase

 Perform Dynamic Application Security Testing (DAST) to detect vulnerabilities in running applications.

5. Deployment Phase

 Ensure secure configurations before deploying applications.

6. Maintenance and Monitoring Phase.

 Implement patch management and incident response plans.

 Conduct periodic security audits and compliance checks.

1. Describe Waterfall, Agile, DevOps model.

**Waterfall**

A linear and sequential model where each phase must be completed before moving to the next.

Phases:

1. Requirements Gathering

2. System Design

3. Implementation (Coding)

4. Testing

5. Deployment

6. Maintenance

Pros:

✔️ Simple and easy to understand

✔️ Well-structured with clear milestones

Cons:

❌ No flexibility for changes after the initial phase

❌ Late testing phase increases risk of defects

Best Used for:

 Small, well-defined projects Regulated industries (e.g., healthcare, banking)

**Agile**

A flexible and adaptive model where development is incremental and iterative, emphasizing customer collaboration.

Key Agile Frameworks:

 Scrum – Uses sprints (2-4-week development cycles)

 Kanban – Focuses on continuous workflow without predefined iterations

 Extreme Programming (XP) – Focuses on customer satisfaction, frequent releases, and

coding best practices

Pros:

✔️ Highly flexible to changes

✔️ Faster delivery with continuous feedback

Cons:

❌ Requires experienced developers

❌ Difficult to estimate cost and time

Best Used for:

 Start-ups, SaaS, mobile applications

 Projects with frequently changing requirements

**DevOps**

A modern SDLC approach that integrates development (Dev) and operations (Ops) for continuous integration and deployment (CI/CD).

Key Components:

 Continuous Integration (CI) – Automated code integration

 Continuous Deployment (CD) – Frequent and automated releases

 Infrastructure as Code (IaC) – Automated infrastructure management

Pros:

✔️ Faster software releases

✔️ Improved collaboration between teams

Cons:

❌ Requires strong automation and DevOps expertise

❌ Complex implementation in traditional organizations

Best Used for:

 Cloud applications, micro services, enterprise

**Chapter: Security Requirements Engineering:** Identifying and defining security requirements, integrating security requirements with functional requirements.

**Chapter: Threat Modelling:**

1. Techniques for identifying and assessing potential security threats.

The slides outline a **structured process** for identifying and assessing threats as part of **threat modelling**. The main techniques include:

#### 🔎 **1. Threat Identification Techniques**

Used to systematically discover possible threats that could impact the system:

* **STRIDE** (Developed by Microsoft):  
  Categorizes threats into six common types:
  + **S**poofing – Impersonation of a user/system
  + **T**ampering – Unauthorized data modification
  + **R**epudiation – Denial of performing an action
  + **I**nformation Disclosure – Unauthorized data access
  + **D**enial of Service – Disrupting system availability
  + **E**levation of Privilege – Gaining unauthorized access
* **Attack Trees**:  
  A visual technique where threats are structured like a tree, with the goal as the root and different ways to achieve that goal as branches. Helps analyze how attacks might be carried out.

#### 🧩 **2. Risk Assessment Techniques**

After identifying threats, assess them based on:

* **Likelihood** – How probable is the threat?
* **Impact** – What damage would it cause if it occurred?
* **Prioritization** – Rank threats based on risk to focus on high-impact/high-probability threats first.

#### 🧠 **3. System Modelling**

Used to visualize the system and its attack surface:

* **Data Flow Diagrams (DFDs)** – Represent how data flows through a system, showing inputs/outputs, processes, storage, and trust boundaries.
* **Architecture Diagrams** – Depict the structure of the system and components, which helps identify where security controls are required.

#### 🧰 **4. Threat Modelling Methodologies**

Different methodologies provide frameworks for assessing threats:

* **PASTA (Process for Attack Simulation and Threat Analysis)** – Focuses on aligning technical threats with business impact.
* **Trike** – Risk-based approach emphasizing attacker perspective.
* **VAST (Visual, Agile, Simple Threat Modelling)** – Designed for large-scale, agile environments.

#### 🛡️ **5. Validation and Review**

As part of the process:

* Continuously **validate** the threat model against the actual system.
* **Review** periodically to adapt to system changes and new emerging threats.

1. Creating threat models to analyse and prioritize risks.

We have to consider the following steps to creating threat models to analyse and prioritize risks.

**Step 1: Define the Scope**

* + Identify the system or application to be modeled.
  + Define the boundaries of the analysis, including what is in scope and out of scope.

**Step 2: Identify Assets**

* + Determine the critical assets that need protection, such as sensitive data, user credentials, or system functionality.

**Step 3: Create a System Model**

* + Develop a visual representation of the system, including its components, data flows, and trust boundaries.

Common modeling techniques include data flow diagrams (DFDs) and architecture diagrams.

**Step 4: Identify Threats**

* + Use threat identification techniques (e.g., STRIDE, attack trees) to enumerate potential threats to the system.
  + Consider both external and internal threats.

**Step 5: Assess Risks**

* + Evaluate the likelihood and impact of each identified threat.
  + Prioritize threats based on their risk level.

**Step 6: Mitigate Threats**

* + Develop strategies to address the identified threats, such as implementing security controls, redesigning components, or adding monitoring.

**Step 7: Validate and Review**

* + Validate the threat model to ensure it accurately represents the system and its risks.
  + Regularly review and update the threat model as the system evolves

1. What is Threat & Threat Modelling? Describe the Key concepts of threat modelling.

**Threat:** In Cyber Security threat is malicious activity to gain un authorized access.

**Threat Modelling:** Threat modelling is the structured approach to identifying, prioritizing and mitigating the security threats to the systems, applications and networks.

**Key Concepts of Threat Modelling:**

* + - Assets
    - Threat
    - Vulnerabilities
    - Attack surface
    - Mitigating

**Secure Architecture Design:** Principles of designing secure software architecture, Security patterns and best practices for system design,

**Secure Coding Practices:** Writing secure code and common programming vulnerabilities, Code reviews and static code analysis for security.