E-learning recap

- Risks, Vulnerabilities and Threats
- Attack types
- Security controls

Learning Objectives

1. Understand the fundamental principles of the CIA triad and its application in cyber security.
2. Recognise risks, vulnerabilities, and threats to ensure robust security for data products.
3. Explain security controls and quantify and evaluate the impact of security breaches
4. Identify and mitigate common cyber threats

Confidentiality approaches

- **Access Controls: Restricting data access to authorised users**
- **Authentication Protocols: Verifying user identities**
- **Encryption: Scrambling data to protect it**

  We will learn about symmetric and assymetric encryption. They are cryptographic methods (cryptography is a branch of applied maths dealing with keeping secrets!)

Tutor Guidance: Introduce encryption, one of the main ways of achieving confidentiality

Tutor Notes:
Encryption is a process used to protect data by converting it into an unreadable format using cryptographic techniques. It ensures **confidentiality**, making it difficult for unauthorised parties to access sensitive information. There are **two primary types of encryption: symmetric and asymmetric**.

1. Symmetric Encryption
Symmetric encryption uses **a single key** for both encryption and decryption. This means that the same secret key is used to scramble the data and later used to return it to its original form. Because only one key is involved, symmetric encryption is generally **faster** and requires less computational power.

How it Works:
1. The sender encrypts the plaintext using a secret key.
2. The encrypted data (ciphertext) is sent to the recipient.
3. The recipient decrypts the ciphertext using the same secret key.

Example Algorithms:
- **AES (Advanced Encryption Standard)** – Widely used in secure communications.
- **DES (Data Encryption Standard)** – An older encryption standard, now mostly replaced.
- **Blowfish** – A strong alternative for secure encryption.

Use Cases:
- Encrypting files and databases.
- Securing communications within a closed system (e.g., VPN tunnels).
- Payment processing systems (e.g., card transactions).

Key Challenge: Key Distribution Problem – The secret key must be securely shared between parties, making it vulnerable to interception.

2. Asymmetric Encryption
Asymmetric encryption, also known as **public-key cryptography**, uses **two mathematically linked keys**: a **public key** for encryption and a **private key** for decryption. This eliminates the need for sharing a single key.

How it Works:
1. The sender encrypts the message using the recipient's **public key**.
2. The recipient decrypts it using their **private key**.
3. Since the private key is never shared, security is enhanced.

Example Algorithms:
- **RSA (Rivest-Shamir-Adleman)** – A widely used standard for secure data transmission.
- **ECC (Elliptic Curve Cryptography)** – Offers strong security with shorter key lengths.
- **DSA (Digital Signature Algorithm)** – Used for digital signatures and authentication.

Use Cases:
- Secure email communication (PGP encryption).
- Digital certificates and SSL/TLS encryption for websites.
- Secure online transactions and authentication.

Key Challenge: Slower Performance **–** Asymmetric encryption is computationally intensive, making it less efficient for encrypting large amounts of data.

21 – Cryptography

21.2 – Confidentiality
21.2.2 – Symmetric Encryption

21 – Cryptography

21.2 – Confidentiality
21.2.2 – Symmetric Encryption

21 – Cryptography

21.2 – Confidentiality
21.2.3 – Asymmetric Encryption

21 – Cryptography

21.2 – Confidentiality
21.2.3 – Asymmetric Encryption

21 – Cryptography

21.2 – Confidentiality
21.2.4 – Asymmetric Encryption - Confidentiality

21 – Cryptography

21.2 – Confidentiality

21.2.5 – Asymmetric Encryption - Authentication

Step-by-Step Process for Authenticating a Message

1. **Alice Creates a Digital Signature and Sends a Message**
    1. Alice takes the original message and applies a **cryptographic hash function** (e.g., SHA-256).
    2. This produces a unique **message hash** (a fixed-length representation of the message).
    3. Alice then **encrypts this hash using her private key**, creating a **digital signature**.
    4. She sends **both the original message and the digital signature** to Bob.
2. **Bob Receives the Message and Extracts the Digital Signature**
    1. Bob extracts both the **original message** and the **digital signature** sent by Alice.
3. **Bob Computes the Hash of the Received Message**
    1. Bob applies the **same cryptographic hash function (e.g., SHA-256)** to the received message.
    2. This generates the **computed hash**.
4. **Bob Decrypts the Digital Signature Using Alice's Public Key**
    1. Since Alice signed the message using her **private key**, Bob can **decrypt the signature using Alice's public key**.
    2. This gives Bob the **original hash** that Alice computed before sending the message (let's call this the **decrypted hash**).
5. **Bob Compares the Decrypted Hash and Computed Hash**
    1. If **both hashes match**, the message is **authentic and untampered**.
    2. If the hashes **do not match**, the message has been **modified in transit or is from an imposter**.

Additional notes:

A **ciphered hash** (or **hashed message signature**) is a fundamental concept in asymmetric encryption, particularly in **digital signatures** and **message authentication**.

What is a Hash?
A **hash function** converts data into a fixed-length string (hash value) that uniquely represents the original data. Hash functions are **one-way**, meaning the original data **cannot be reversed** from the hash.

Example of a hash function:

SHA-256("Hello") →
185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969

Classic class discussion on asymmetric encryption.

Description:

The diagram illustrates how **asymmetric encryption** and **digital signatures** work together to ensure **confidentiality, authenticity, and integrity** in secure communication. Here's a breakdown of the process:

1. Key Generation and Sharing

- **Alice and Bob each have a pair of cryptographic keys:**
    - **Private key (stored locally):** This is kept secret and never shared.
    - **Public key (published on a key server):** This is available to anyone for encryption and verification.

2. Encryption and Signing by Alice

- Alice wants to send a **secure** message to Bob.
- She **encrypts the message using Bob's public key**:
    - Since only Bob has the corresponding **private key**, only he can decrypt the message.
    - This ensures **confidentiality** because even if someone intercepts the message, they cannot decrypt it.
- Alice also **signs the message with her private key**:
    - This creates a **digital signature**, which allows Bob to verify that Alice was the sender.
    - This ensures **authenticity** and **integrity** because the signature is unique to Alice and any tampering would make it invalid.

3. Message Transmission

- The encrypted and signed message travels across the network.
- **Eve (an attacker)** intercepts the message but cannot read it because:
    - She does not have Bob’s private key to decrypt it.
    - She cannot forge Alice’s signature without Alice’s private key.

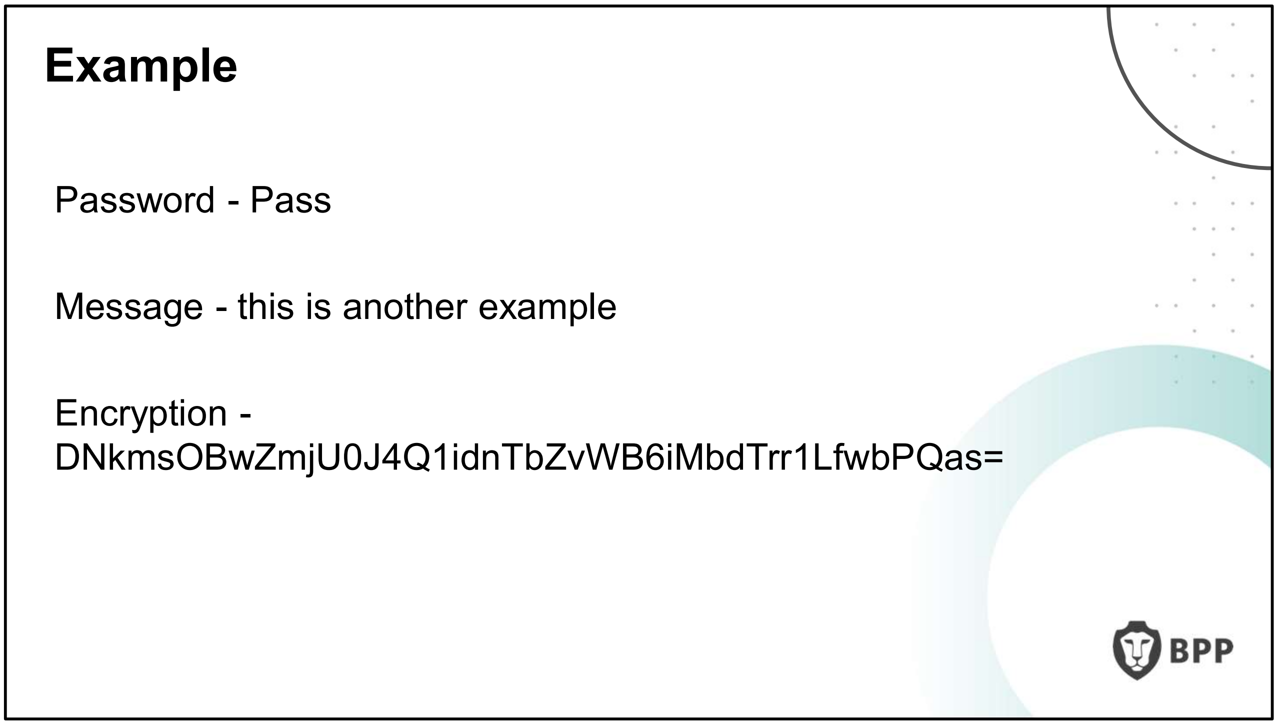
4. Decryption and Verification by Bob

- Upon receiving the message, Bob:
    - **Decrypts the message using his private key** (since it was encrypted with his public key).
    - **Verifies the digital signature using Alice's public key**:
        - If the decrypted hash matches the computed hash of the received message, it confirms that Alice signed the message and it has not been altered.

21 – Cryptography

21.2 – Confidentiality

21.2.10 – Lab - Encrypting and Decrypting Data Using OpenSSL



Example

Password - Pass

Message - this is another example

Encryption -
DNkmsOBwZmjU0J4Q1idnTbZvWB6iMbdTrr1LfwbPQas=

Tutor Guidance: Explain how Public Key Management works with a relatable example

Tutor Notes:

The process of establishing secure communication using **asymmetric encryption and digital certificates** is similar to how **identity verification works in real life**, such as applying for and using a **driver's license**.

1. Exchanging Public Keys (Alice Applies for a License)
- When two computers (hosts) want to communicate securely, they **exchange their public keys**, just like Alice **applies for a driver's license** to prove her identity.
- However, just sharing a public key is **not enough –** it needs to be verified by a trusted authority to confirm it actually belongs to the entity it claims to represent.

2. Trusted Third-Party Validation (Government Issues a License)
- On the Internet, **trusted third parties** (called Certificate Authorities or CAs) validate the authenticity of these public keys, similar to how the **government verifies Alice's identity** before issuing her a driver's license.
- The CA issues a **digital certificate**, which acts like a driver's license for a website or service, confirming its authenticity.

3. Accepting Trusted Credentials (Using a License to Cash a Check)
- Once Alice has her license, businesses and banks **trust her identity** without needing to verify it again, just by checking the license.
- Similarly, in cybersecurity, once a website or entity has a **valid digital certificate**, users and computers that trust the issuing Certificate Authority **accept the certificate automatically**.

4. Public Key Infrastructure (PKI) – The System Behind It
- PKI is the **entire system** that creates, manages, distributes, and revokes digital certificates, just like the **government's system** that manages driver's licenses.
- It includes **Certificate Authorities (CAs)**, **registration processes**, and **validation checks**.

5. Certificate Authority (CA) – The Issuer of Digital Trust
- A CA is like the **government agency** that issues Alice's driver's license.
- It ensures that a public key belongs to a legitimate entity, like a **website** or **an individual**.
- Users trust websites with **SSL/TLS certificates** because their identities are backed by a CA, just as a bank trusts Alice's license because it was issued by the government.

Summary
- Just as Alice **proves her identity to get a driver's license**, websites and services **prove their identity to a Certificate Authority to get a digital certificate**.
- Once Alice has her license, businesses **trust it without verifying it again**, just as users trust a website's **certificate issued by a CA**.
- PKI is the **system that manages and enforces** digital certificate trust, just like a government agency **manages the driver's license system**.

21 – Cryptography

21.4 – Authorities and the PKI Trust System

21.4.2 – The Public Key Infrastructure

This image illustrates the **Public Key Infrastructure (PKI) process** and how **digital certificates** are issued, exchanged, an

Step-by-Step Breakdown of the PKI Process:

1. **Issuing a PKI Certificate (Step 1)**
    1. Bob **requests a digital certificate** from a **Certificate Authority (CA)**.
    2. The **CA verifies Bob's identity** and then **issues a PKI certificate** that ties Bob's **public key** to his verified identity.
    3. The issued certificate is stored in a **certificate database**, ensuring its authenticity.
2. **Exchanging the PKI Certificate (Step 2)**
    1. Bob wants to communicate securely with Alice.
    2. Instead of sending his **raw public key**, he sends his **PKI certificate**, which includes his public key and is signed by the **trusted CA**.
    3. This ensures Alice can verify his identity.
3. **Verifying the PKI Certificate (Step 3)**
    1. Alice **receives Bob's PKI certificate** and needs to verify that it's legitimate.
    2. She checks the certificate's authenticity by:
        1. **Using the CA's public key** to verify that the certificate was **indeed issued by a trusted CA**.
        2. Checking the **certificate database** to ensure it hasn't been revoked or tampered with.
    3. If the certificate is valid, Alice **trusts Bob's identity** and can establish a secure communication channel.

21 – Cryptography

21.4 – Authorities and the PKI Trust System

21.4.3 – The PKI Authorities System

Method 1: Viewing SSL Certificates for a Website

If you want to check a website's SSL/TLS certificate:

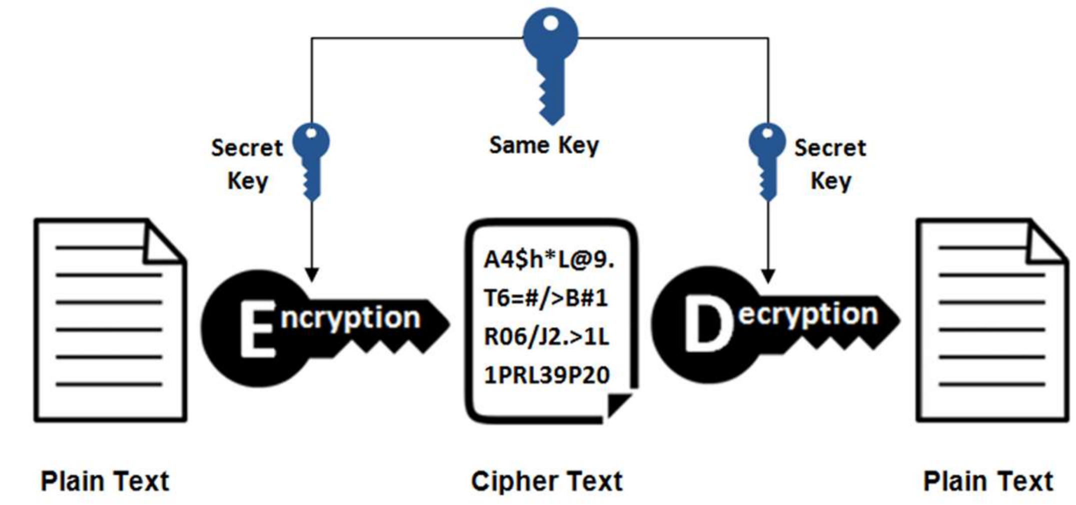
1. **Open Google Chrome** and navigate to the website.
2. **Click on the padlock icon** 🔒 in the address bar (to the left of the URL).
3. Select **"Connection is secure"** (if available).
4. Click **"Certificate is valid"** to open the certificate details.
5. Here, you can view:
    1. **Issuer (CA)**
    2. **Expiration Date**
    3. **Certificate Chain**
    4. **Encryption Details (Public Key, Signature Algorithm, etc.)**

Method 2: Viewing Installed Certificates on Your Computer

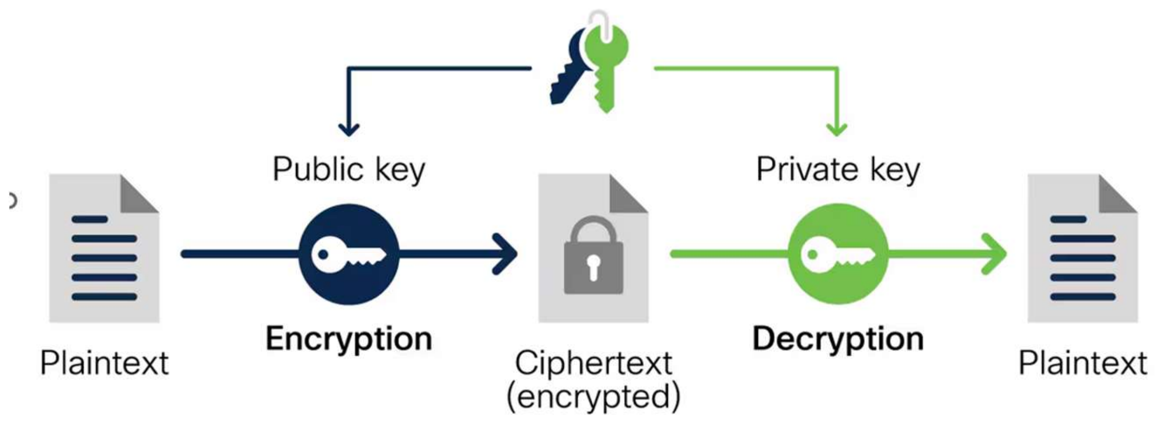
If you want to check your **own system's certificates** (trusted root CAs, personal certificates, etc.):

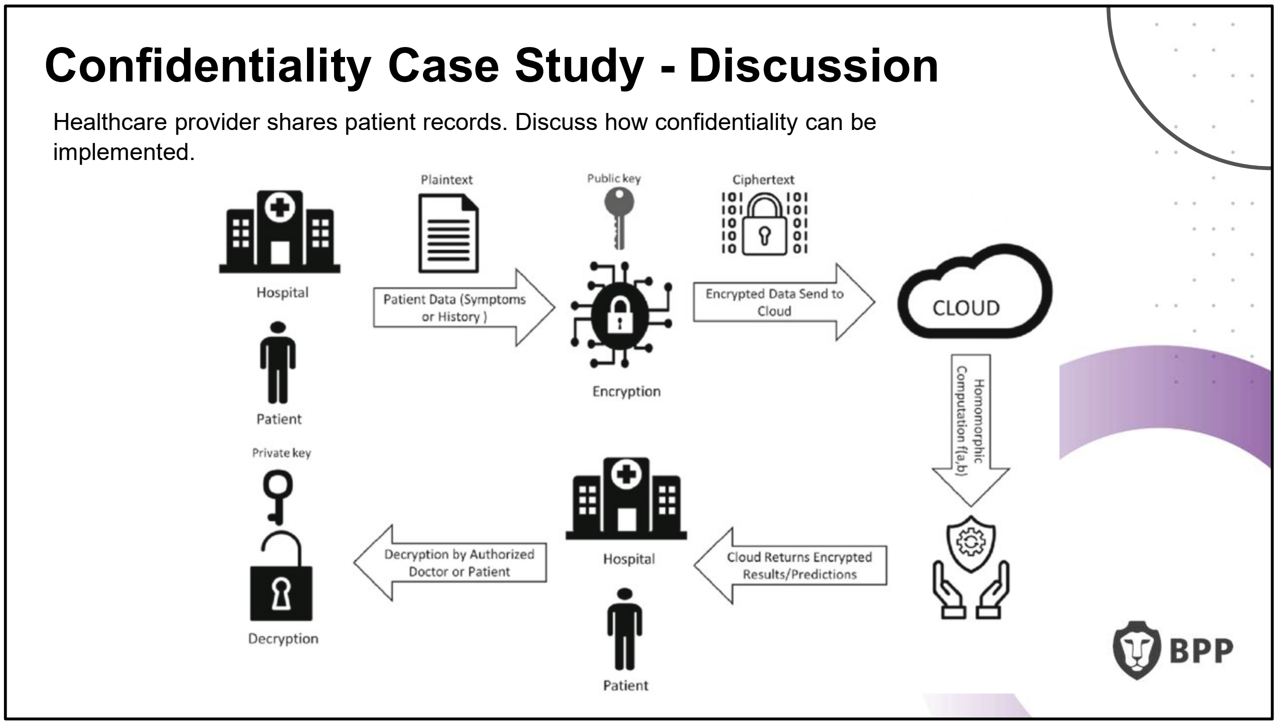
1. Open **Google Chrome**.
2. Click the **three-dot menu** (⋮) in the top-right corner.
3. Go to **Settings**.
4. Scroll down and select **Privacy and security**.
5. Click **Security**.
6. Scroll down and click **Manage certificates**.
7. This will open the **Windows Certificate Manager** (if on Windows) or the **macOS Keychain Access** (if on Mac).
8. Here, you can browse different certificate categories:
    1. **Personal Certificates** (Your own certificates)
    2. **Trusted Root Certification Authorities** (Certificates from trusted CAs)
    3. **Intermediate Certification Authorities**
    4. **Other People / Untrusted Certificates**
9. You can **export, import, or delete** certificates as needed.

Summary: symmetric encryption



Summary: asymmetric encryption





Confidentiality Case Study - Discussion

Healthcare provider shares patient records. Discuss how confidentiality can be implemented.

Hospital
Patient
Plaintext
Patient Data (Symptoms or History)
Public key
Encryption
Ciphertext
Encrypted Data Send to Cloud
CLOUD
Homomorphic Computation f(a,b)
Cloud Returns Encrypted Results/Predictions
Hospital
Patient
Decryption by Authorized Doctor or Patient
Private key
Decryption
BPP

Secure Cloud Processing Using Homomorphic Encryption

This diagram illustrates a **secure healthcare data processing system** using **homomorphic encryption** to protect patient confidentiality while utilizing cloud computing for analysis.

Step-by-Step Breakdown of the Process:

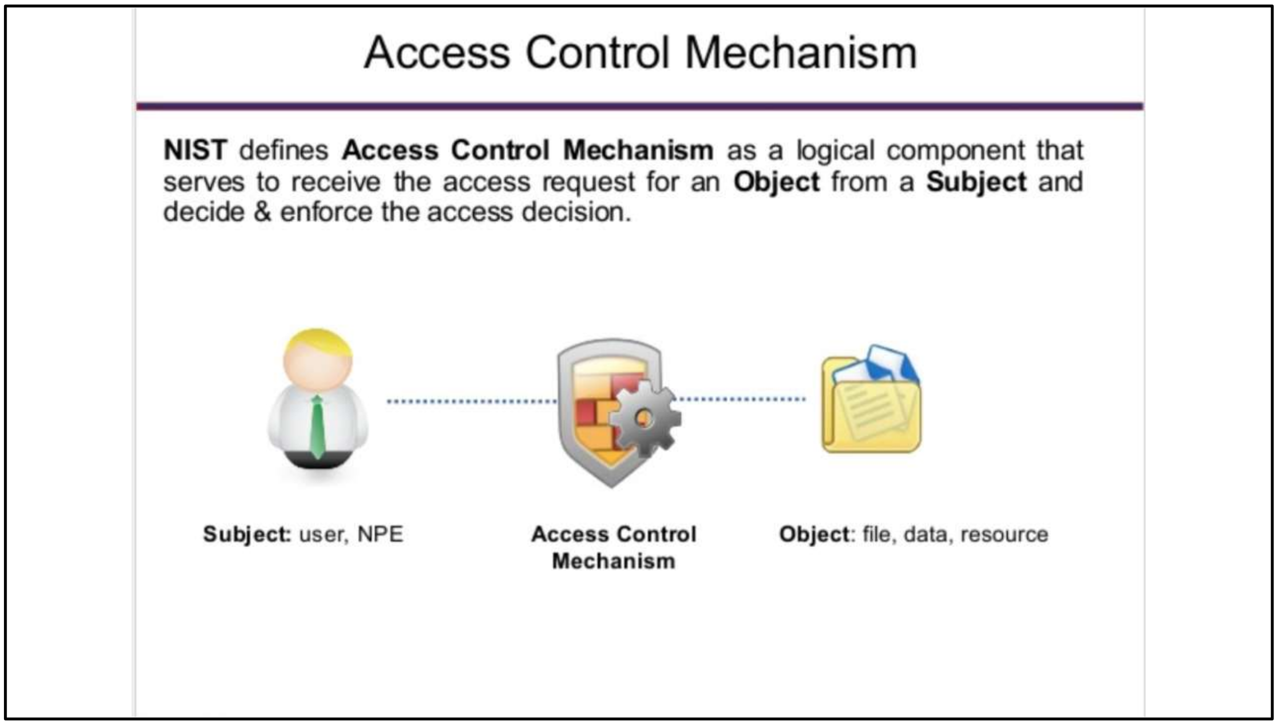
1. **Patient Data Collection (Plaintext)**
    1. A hospital collects patient data (such as symptoms or medical history).
    2. This data is in **plaintext** (readable form) initially.
2. **Encryption with Public Key**
    1. The patient’s data is encrypted using a **public key** before being sent to the cloud.
    2. This ensures **data confidentiality** as only authorized parties with the corresponding **private key** can decrypt it.
3. **Data Sent to the Cloud (Ciphertext)**
    1. The encrypted data (ciphertext) is sent to a cloud server.
    2. The cloud **never has access to the original plaintext data**, ensuring privacy.
4. **Homomorphic Computation on Encrypted Data**
    1. The cloud **performs computations on the encrypted data without decrypting it**.
    2. This process is called **homomorphic encryption**, which allows operations like medical predictions, analytics, or AI-based diagnosis **directly on encrypted data**.
5. **Encrypted Results Sent Back**

    1. The cloud returns the **processed encrypted results** to the hospital.

1. **Decryption by Authorized Users**
    1. Only an **authorized doctor or the patient** (who has the private key) can **decrypt the processed results**.
    2. This ensures that **sensitive medical information remains confidential throughout the process**.

How Confidentiality is Implemented:

1. **Encryption Before Transmission**
    1. Patient data is **encrypted at the source (hospital)** using a public key before being sent to the cloud.
    2. This prevents unauthorized access during transmission.
2. **Homomorphic Encryption for Secure Processing**
    1. Since the cloud **only receives encrypted data**, it **cannot see or access** the original information.
    2. Computations are performed **without decryption**, preserving data privacy.
3. **Private Key Control for Decryption**
    1. Only **authorized individuals (doctors or patients)** possess the **private key** required for decryption.
    2. Even if a hacker or unauthorized user gains access to the cloud, they **cannot decrypt the data**.
4. **Minimized Data Exposure**
    1. At no point is sensitive data exposed **in plaintext** outside of the hospital or authorized access points.
    2. This reduces the risk of **data breaches, leaks, or insider threats**.



Access Control Mechanism

NIST defines **Access Control Mechanism** as a logical component that serves to receive the access request for an **Object** from a **Subject** and decide & enforce the access decision.

Subject: user, NPE
Access Control Mechanism
Object: file, data, resource

Tutor Guidance: Define what an Access Control Mechanism is
Tutor Notes:

This image illustrates the **Access Control Mechanism** as defined by **NIST (National Institute of Standards and Technology)**. It depicts the relationship between a **subject**, an **object**, and the access control mechanism that enforces security policies.

Step-by-Step Breakdown of the Access Control Process

1. **Subject (User or NPE)**
    1. The **subject** represents an **entity that initiates a request** to access a resource.
    2. This can be:
        1. A **user** (e.g., an employee logging into a system).
        2. A **Non-Person Entity (NPE)** (e.g., an automated process or software).

2. **Access Control Mechanism**
    1. This is the **security control system** that evaluates whether the subject has permission to access the requested resource.
    2. It enforces **access policies** based on:
        1. User roles, authentication credentials, security policies, and predefined rules.

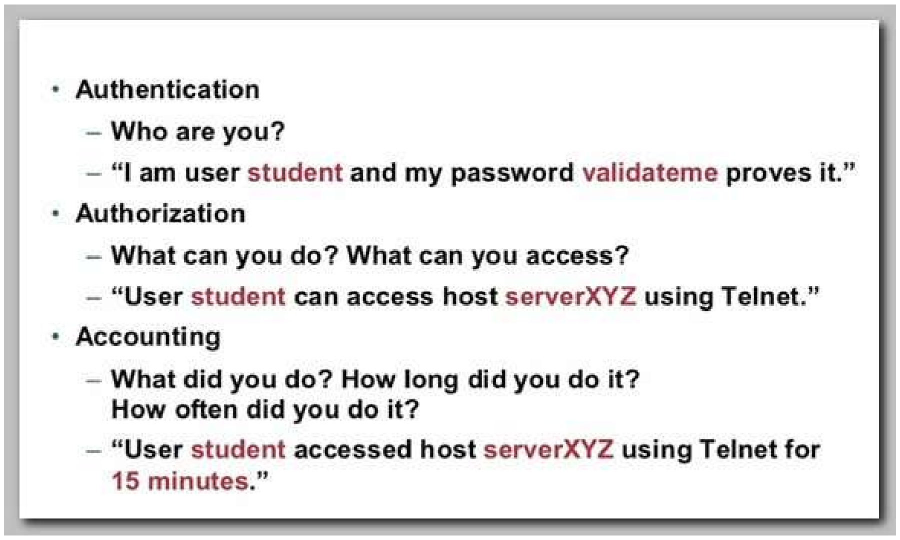
3. Examples include:
    1. **Role-Based Access Control (RBAC)** – Access is granted based on job roles.
    2. **Mandatory Access Control (MAC)** – Access is strictly controlled based on classification levels.
    3. **Discretionary Access Control (DAC)** – Users have the flexibility to control access to resources they own.

1. **Object (File, Data, or Resource)**
    1. The **object** represents the **target resource** that the subject wants to access.
    2. This could be:
        1. A **file** (e.g., a confidential report).
        2. **Data** (e.g., customer records).
        3. A **system resource** (e.g., a network database).

How the Access Control Mechanism Works
- The **subject (user or process)** requests access to an **object (data, file, or resource)**.
- The **Access Control Mechanism** evaluates the request based on:
    - User credentials.
    - Access control policies.
    - Security rules.
- If **approved**, access is **granted**.
- If **denied**, access is **blocked** to prevent unauthorized access.

Real-World Examples of Access Control Mechanisms
- **Corporate Networks:** Employees are assigned roles (e.g., "HR Manager") and can only access HR records but not IT security logs.
- **Cloud Storage (Google Drive, Dropbox):** A user can share a file with **read-only, edit, or full access** permissions.
- **Healthcare Systems (HIPAA Compliance):** Doctors can access patient records, but administrative staff cannot view medical history.

Key Takeaways
- **Subjects** (users or processes) request access to **objects** (data, files, or resources).
- The **Access Control Mechanism** evaluates the request and **enforces security policies**.
- Different access control models (RBAC, MAC, DAC) determine how permissions are granted.
- This ensures **data security, confidentiality, and integrity**.



- Authentication
  - Who are you?
  - "I am user student and my password validateme proves it."
- Authorization
  - What can you do? What can you access?
  - "User student can access host serverXYZ using Telnet."
- Accounting
  - What did you do? How long did you do it? How often did you do it?
  - "User student accessed host serverXYZ using Telnet for 15 minutes."

The image describes the **AAA Security Model**, which consists of **Authentication, Authorization, and Accounting (AAA)**. This model is fundamental to cybersecurity and access control.

1. **Authentication: Proving Identity**
   1. **What is it?**
      Authentication is the process of verifying that a user or entity is who they claim to be.
   2. **Example Explanation:**
      Imagine a person entering a secured building. They must present their **ID badge** or enter a **password** to prove their identity before being allowed in.
   3. **In the image's context:**
      1. The user provides their username (student) and password (validateme) to log in.
      2. If the credentials match stored records, access is **granted**.

2. **Authorization: Granting Permissions**
   1. **What is it?**
      Authorization determines **what actions a user can perform** or **what resources**

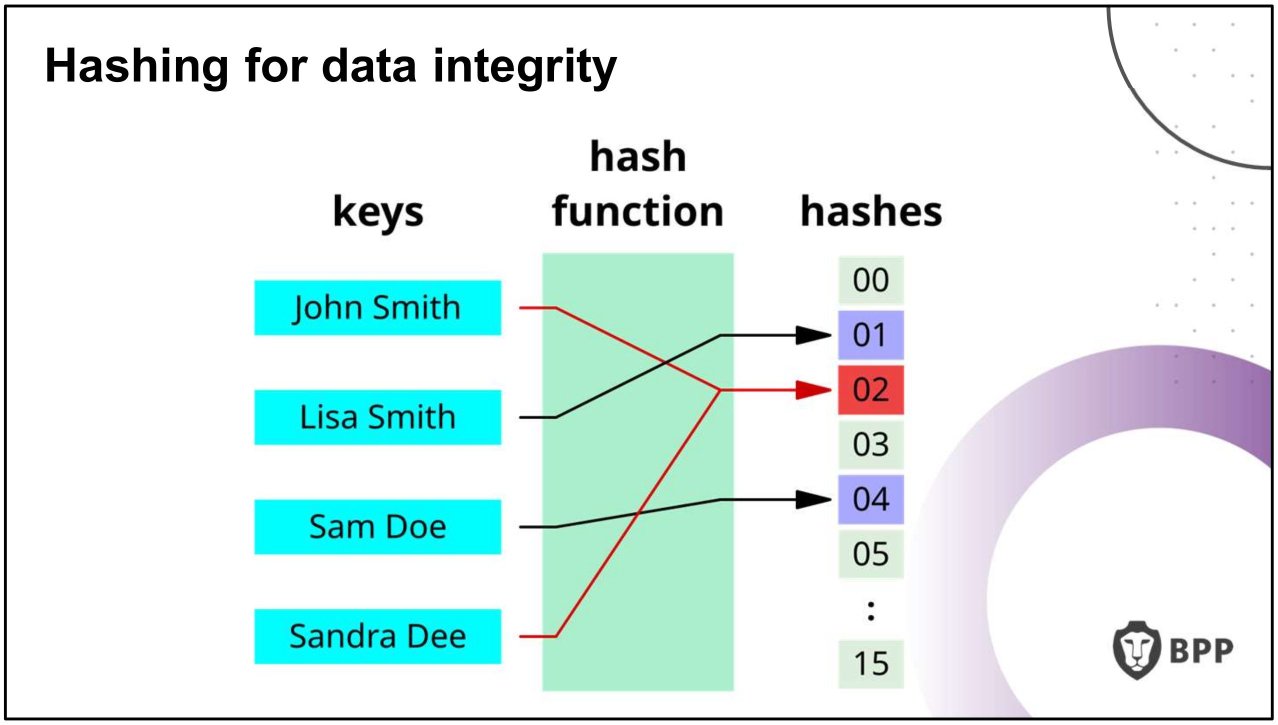
    they can access once authenticated.

  2. **Example Explanation:**
     After proving their identity, an employee in a company may have access to **certain floors** but not **executive offices**.
  3. **In the image's context:**
     1. The system checks whether student has permission to use **Telnet** to connect to serverXYZ.
     2. If authorized, access is **granted**.

1. **Accounting: Tracking Activity**
   1. **What is it?**
      Accounting (also known as **Auditing or Logging**) keeps track of **what users do, how long they do it, and how often they access resources**.
   2. **Example Explanation:**
      A security camera logs when an employee enters and exits the office.
   3. **In the image's context:**
      1. The system logs that student accessed serverXYZ using **Telnet** for **15 minutes**.
      2. These logs can help with **auditing, security monitoring, and detecting suspicious behavior**.

Key Takeaways

- **Authentication answers**: "Who are you?"
- **Authorization answers**: "What can you do?"
- **Accounting answers**: "What did you do?"



Hashing for data integrity

The image represents a **hash function** mapping different inputs (names) to fixed numerical values, resembling how **hashing ensures data integrity** in cybersecurity.

Key Integrity Aspects in the Image:

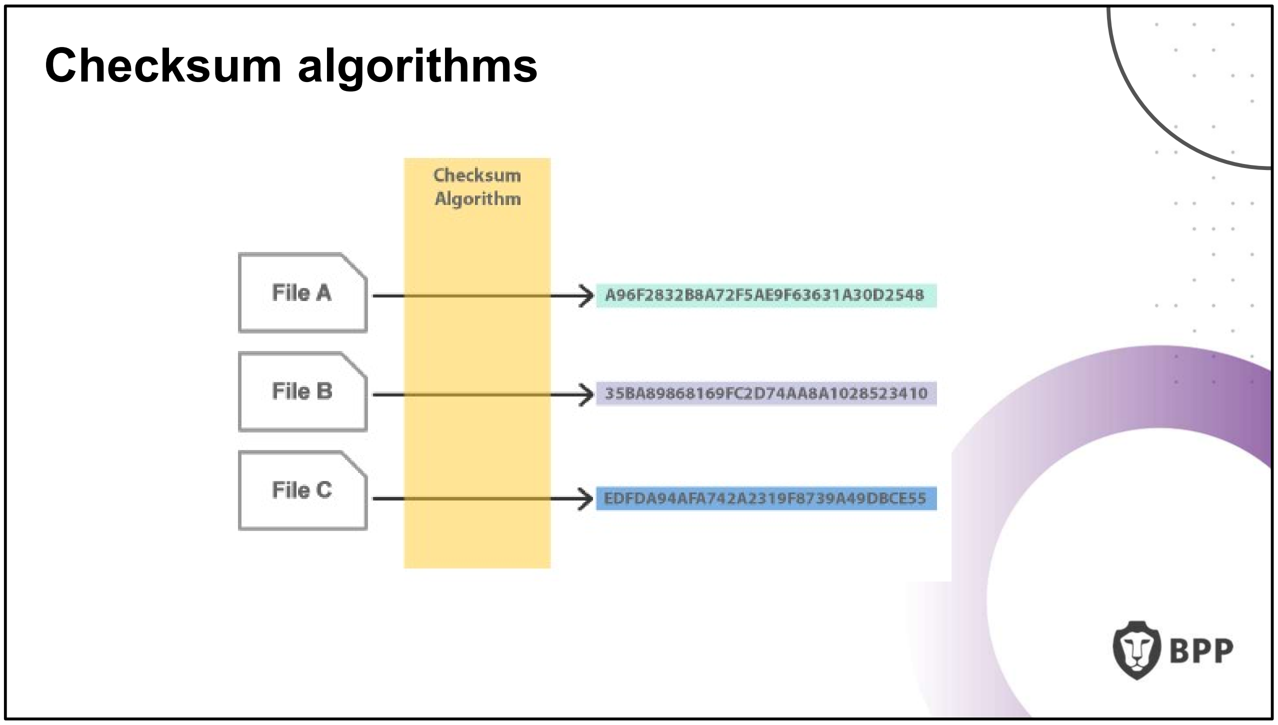
1. **Hashing Process for Integrity Verification**
    1. The image demonstrates how different names are processed through a hash function, producing **fixed-size hash values**.
    2. Hashing is a **core integrity mechanism**, ensuring that **data has not been altered** by generating a unique fingerprint.
2. **Collision Detection (Integrity Violation)**
    1. The red arrow and overlap at index **02** indicate a **hash collision**, where two different inputs (e.g., "John Smith" and "Sandra Dee") map to the same hash value.
    2. This implies a potential **integrity issue** since **different inputs should produce unique hashes** to ensure accurate verification.
    3. Cryptographic hash functions, such as **SHA-256**, minimize these collisions, reinforcing data integrity.
3. **Immutable Data Mapping**
    1. The hash function ensures that a given input (e.g., "Lisa Smith") **always maps to**

        the same hash value.
    2. This consistency is critical for integrity, as it helps detect even minor **modifications in data**.

1. **Use in Integrity Checks (Data Verification)**
    1. In cybersecurity, **hashing is used to detect unauthorized changes** in:
        1. **Digital signatures** (ensuring a document hasn't been altered).
        2. **File integrity checks** (comparing hash values to detect tampering).
        3. **Password storage** (verifying credentials without storing plaintext passwords).

Conclusion

This image suggests **data integrity through hashing**, illustrating how input values are mapped to fixed outputs. If the mapping is altered (e.g., due to an attack or corruption), the hash value changes, alerting to a potential **integrity breach**. In **real-world cybersecurity**, strong cryptographic hash functions (e.g., SHA-256) help maintain data reliability by preventing unauthorized modifications.



Checksum algorithms

Checksum Algorithm
File A
File B
File C
A96F2832B8A72F5AE9F63631A30D2548
35BA89868169FC2D74AA8A1028523410
EDFDA94AFA742A2319F8739A49DBCE55
BPP

A **checksum algorithm** is used to verify **data integrity** by generating unique checksum values (hashes) for different files.

Step-by-Step Breakdown of the Process

1. **Input Files (File A, File B, File C)**
    1. Each file contains data that needs to be validated for integrity.
    2. These could be software downloads, stored documents, or transmitted files.
2. **Checksum Algorithm Processing**
    1. Each file is processed through a **checksum algorithm**, which computes a unique hash value based on the file's contents.
    2. The checksum acts like a **fingerprint**—if even a small change occurs in the file, the resulting checksum will be **completely different**.
3. **Generated Checksums**
    1. Each file produces a unique checksum (hash) as output:
        1. **File A →** A96F2832B8A72F5AE9F63631A30D2548
        2. **File B →** 35BA89868169FC2D74AA8A1028523410
        3. **File C →** EDFDA94AFA742A2319F8739A49DBCE55

2. If the files are **modified, corrupted, or tampered with**, their checksum values will change, signaling **integrity issues**.

How This Ensures Data Integrity

- **File Verification:** Before and after transmission, the checksum can be compared to detect alterations.
- **Tamper Detection:** If a hacker modifies a file, its checksum will not match the original, exposing unauthorized changes.
- **Software & Data Integrity:** When downloading software, websites often provide checksum values (e.g., **SHA-256, MD5**) so users can verify the file's authenticity.

Real-World Applications

1.Software Downloads: Websites like **Ubuntu Linux** provide an official checksum. Users can verify the downloaded file using a hashing tool.
2.Data Transmission (Cloud & Networking): Cloud storage providers use checksums to detect corrupted files during upload/download.
3.Cybersecurity & Forensics: Cryptographic hashes (SHA-256, SHA-512) ensure digital evidence is **not tampered with**.

Key Takeaways

- **Checksums are unique values** that verify file integrity.
- **Even a minor file change produces a completely different checksum.**
- **Cryptographic hashing algorithms (SHA-256, SHA-512) are used to prevent forgery and maintain security.**
- **Common checksum algorithms include MD5, SHA-1, and SHA-256.**

21 – Cryptography

21.1 – Integrity and Authenticity
21.1.2 – Cryptographic Hash Functions

21 – Cryptography

21.1 – Integrity and Authenticity
21.1.4 – MD5 and SHA

21 – Cryptography

21.1 – Integrity and Authenticity
21.1.5 – Origin Authentication

21 – Cryptography

21.2 – Confidentiality

21.2.10 – Lab - Encrypting and Decrypting Data Using OpenSSL

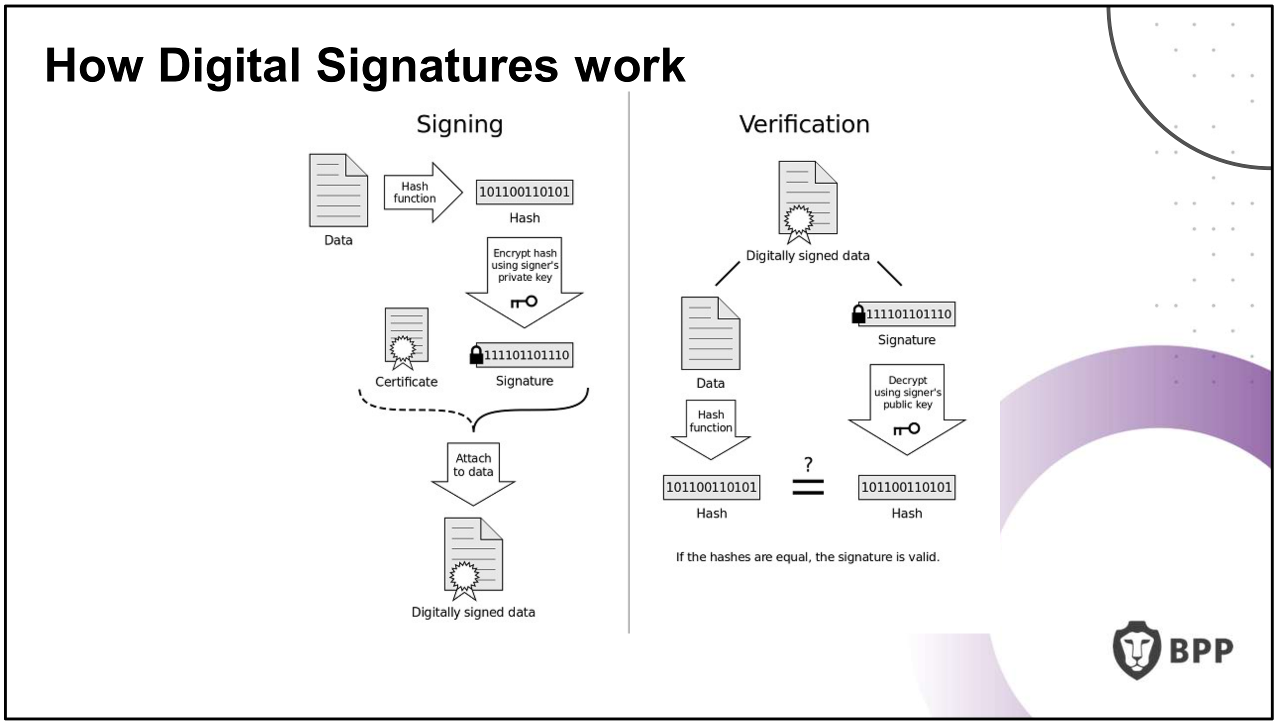
21 – Cryptography

21.3 – Public Key Cryptography

21.3.1 – Using Digital Signatures

21 – Cryptography

21.3 – Public Key Cryptography
21.3.3 – Digital Signatures For Digital Certificates



How Digital Signatures work

Digital Signatures (Signing & Verification)

There are two main processes:

1. **Signing (Creating a Digital Signature)**
2. **Verification (Checking the Signature's Authenticity)**

Digital signatures are a crucial part of **cybersecurity** and **data integrity**, ensuring that documents and messages are authentic and have not been altered.

Signing Process

The **signing** process ensures that data is **authentic and untampered** by using a cryptographic hash function and a private key.

1. **Data Preparation**
   1. A document, message, or file (referred to as **data**) is prepared for signing.
2. **Hashing the Data**
   1. A **hash function** (e.g., SHA-256) is applied to the data, producing a unique **hash value** (fingerprint of the data).

    2. Example: The document generates the hash **101100110101**.
1. **Encrypting the Hash with a Private Key**
    1. The hash is **encrypted using the sender's private key**, generating a **digital signature**.
    2. This ensures that only the sender (who owns the private key) could have signed the data.
2. **Attaching the Signature to the Data**
    1. The **digital signature** is attached to the data, forming a **digitally signed document**.
    2. A **certificate** may also be included to validate the sender's identity.

Verification Process

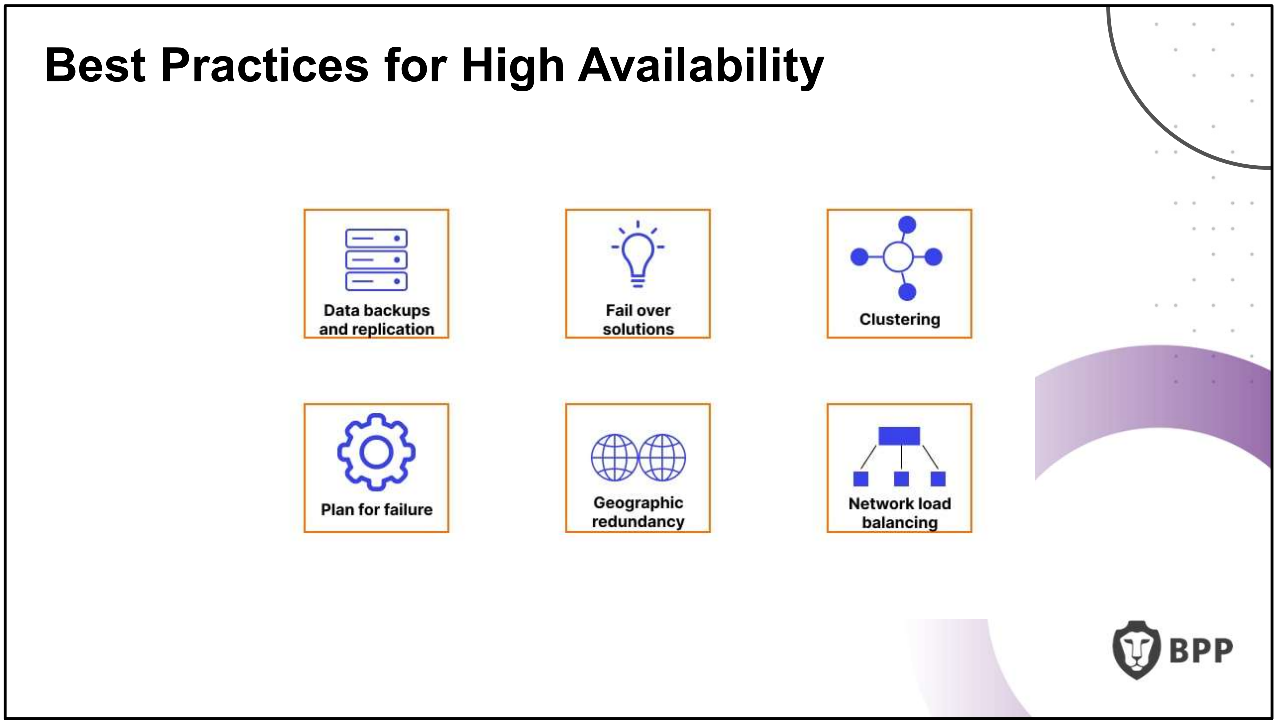
The **verification** process ensures that the received document or message **has not been altered** and was **signed by a trusted sender**.

1. **Extracting the Signature and Data**
    1. The recipient receives the **digitally signed data** (document + signature).
2. **Recomputing the Hash from the Data**
    1. The recipient applies the **same hash function** to the received data.
    2. This generates a **new hash value** (e.g., **101100110101**).
3. **Decrypting the Signature with the Public Key**
    1. The recipient **decrypts the digital signature** using the sender's **public key**.
    2. Since the digital signature was created using the sender's private key, **only the sender's public key** can decrypt it.
    3. The result is the **original hash value**.
4. **Comparing Hashes**
    1. The recipient compares:
        1. The **hash they computed** from the data.
        2. The **hash extracted from the decrypted signature**.
    2. If the two hashes **match**, the signature is **valid**, meaning:
        1. The document has **not been altered**.
        2. The sender's identity is **authentic**.
    3. If the hashes **do not match**, the document has **been tampered with or the sender is not who they claim to be**.

Key Takeaways
- **Digital signatures ensure authenticity, integrity, and non-repudiation**.
- A **private key** is used to create the signature, and a **public key** is used for verification.
- Hashing ensures that **even a small change in data results in a completely different signature**.
- If the hashes **do not match**, it means the data **has been altered** or the signature is **forged**.

Real-World Applications of Digital Signatures

- **Email Security (PGP, S/MIME):** Emails are signed digitally to prove they are from a trusted sender.
- **Software Distribution (Code Signing):** Software updates are digitally signed to prevent malicious modifications.
- **Blockchain & Cryptocurrencies:** Transactions use digital signatures to ensure they are valid and authorized.
- **Document Signing (Adobe PDF, DocuSign, etc.):** Legally binding contracts use digital signatures for verification.



Best Practices for High Availability

Data backups and replication

Fail over solutions

Clustering

Plan for failure

Geographic redundancy

Network load balancing

Tutor Guidance: Elaborate on the best practices for high availability
Tutor Notes:

It is essential to ensure that systems, networks, and applications remain operational with minimal downtime. Below is a more detailed explanation of each concept:

1. Data Backups and Replication

What it means: Regularly backing up and replicating data ensures that in the event of system failures, data loss, or cyberattacks, information can be restored without significant downtime.

Best practices:
- Use **automated backup schedules** (daily, weekly, etc.).
- Implement **redundant storage solutions** (e.g., RAID, cloud-based storage).
- Use **real-time data replication** to keep copies up to date.

Example: A company stores its critical data in **both an on-premises server and a cloud backup**. If the local server fails, the data can be retrieved from the cloud.

2. Failover Solutions

What it means: A failover solution is a **redundancy mechanism** that automatically switches to a backup system if the primary system fails.

Best practices:
- Implement **hot standby servers** that take over immediately when the primary fails.
- Use **automatic failover systems** for databases, networks, and applications.
- Test failover mechanisms regularly to ensure reliability.

Example: A **primary database server crashes**, but a **secondary standby server** automatically takes over with no downtime.

3. Clustering

What it means: Clustering refers to linking multiple servers or systems together to function as a **single unit** to improve availability and performance.

Best practices:
- Implement **load-balanced clusters** where traffic is evenly distributed.
- Use **high-availability clusters (HA Clusters)** where if one server fails, another takes over.
- Deploy **database clustering** to distribute database workloads across multiple servers.

Example: An **e-commerce website** uses a cluster of web servers. If one server goes down, the traffic is redistributed to other servers to maintain availability.

4. Plan for Failure

What it means: Organizations should assume failures **will** happen and plan strategies to **mitigate risks** before they occur.

Best practices:
- Identify **single points of failure (SPOF)** and implement redundancies.
- Maintain **incident response plans** for IT outages.
- Conduct **disaster recovery (DR) drills** regularly.

Example: A company **simulates a cyberattack** to test how well its IT team responds and how quickly systems can recover.

5. Geographic Redundancy

What it means: Storing **critical infrastructure in multiple geographical locations** ensures availability even if a disaster affects one region.

Best practices:

- Use **geo-replicated cloud storage**.
- Deploy **data centers in multiple locations**.
- Implement **DNS failover**, so traffic is redirected to another region if one fails.

Example: A **financial institution** has **data centers in New York, Texas, and California**. If one data center is hit by a power outage, services continue to run from the other locations.

6. Network Load Balancing

What it means: Network load balancing distributes incoming traffic **evenly** across multiple servers to **prevent overload** and **increase performance**.

Best practices:

- Use **hardware or software load balancers** to distribute network requests.
- Implement **round-robin DNS** to balance traffic across multiple servers.
- Monitor **server performance metrics** to optimize load balancing strategies.

Example: A **video streaming service** uses a **load balancer** to distribute traffic between multiple servers, preventing overload during peak streaming hours.

Key Takeaways

- **High availability** ensures that services and applications are **always accessible** with minimal downtime.
- **Combining multiple strategies** (e.g., failover + clustering + geo-redundancy) provides the **best resilience**.
- **Automation** and **regular testing** are essential to ensuring these mechanisms work when failures occur.

Feedback with example answers and feedback for each stage of the activity:

Let's analyse each of the three **cybersecurity scenarios—Phishing Attack, Accidental Data Overwrite, and DDoS Attack**

1. Phishing Attack – Confidentiality

Why?

- Phishing attacks **trick users into revealing sensitive information** (e.g., credentials, financial data, or classified information).
- Once credentials are compromised, attackers can access **confidential data** such as emails, customer records, and internal documents.

Threat Mitigation Strategies

- **User Credential Theft:**
    - **Multi-Factor Authentication (MFA)** to prevent unauthorized logins even if credentials are compromised.
    - **Employee Security Awareness Training** to help staff recognize phishing attempts.
- **Email-Based Phishing**
    - **AI-powered Email Filtering** to detect and block suspicious messages.
    - **Domain-based Message Authentication (DMARC, DKIM, SPF)** to prevent email spoofing.
- **Malicious Links & Attachments**
    - **Web Filtering & DNS Security** to block access to phishing sites.
    - **Endpoint Protection Software** to detect and prevent malware payload execution.

Key Takeaway:
Phishing primarily impacts **Confidentiality** by exposing sensitive data. The best defense includes **employee training, strong authentication, and automated phishing detection**.

2. Accidental Data Overwrite – Integrity

Why?

- Data integrity is compromised when critical files or databases are **accidentally modified or deleted** by authorized users.
- This could lead to **incorrect records, corrupted transactions, or operational failures**.

Threat Mitigation Strategies

- **Human Error (Accidental Deletion/Modification)**
    - **Role-Based Access Control (RBAC)** to limit permissions.
    - **Version Control Systems** to track and revert changes.
- **File Corruption or Overwriting**
    - **Regular Data Backups (Incremental & Full)** for recovery.
    - **Data Loss Prevention (DLP) Tools** to monitor and restrict risky actions.

- **Database Tampering Risks**
    - **Database Transaction Logs & Change Audits** to track all modifications.
    - **Immutable Backups** to prevent unauthorized modifications.

Key Takeaway:
Accidental data overwrite **impacts Integrity**, meaning data consistency and accuracy must be maintained. Preventive measures include **access control, backup strategies, and change auditing**.

3. DDoS (Distributed Denial-of-Service) Attack – Availability

Why?
- A **DDoS attack overwhelms a system or network** with excessive traffic, making services **unavailable** to legitimate users.
- Businesses relying on uptime (e.g., e-commerce, online banking) **lose revenue and customer trust** when systems are down.

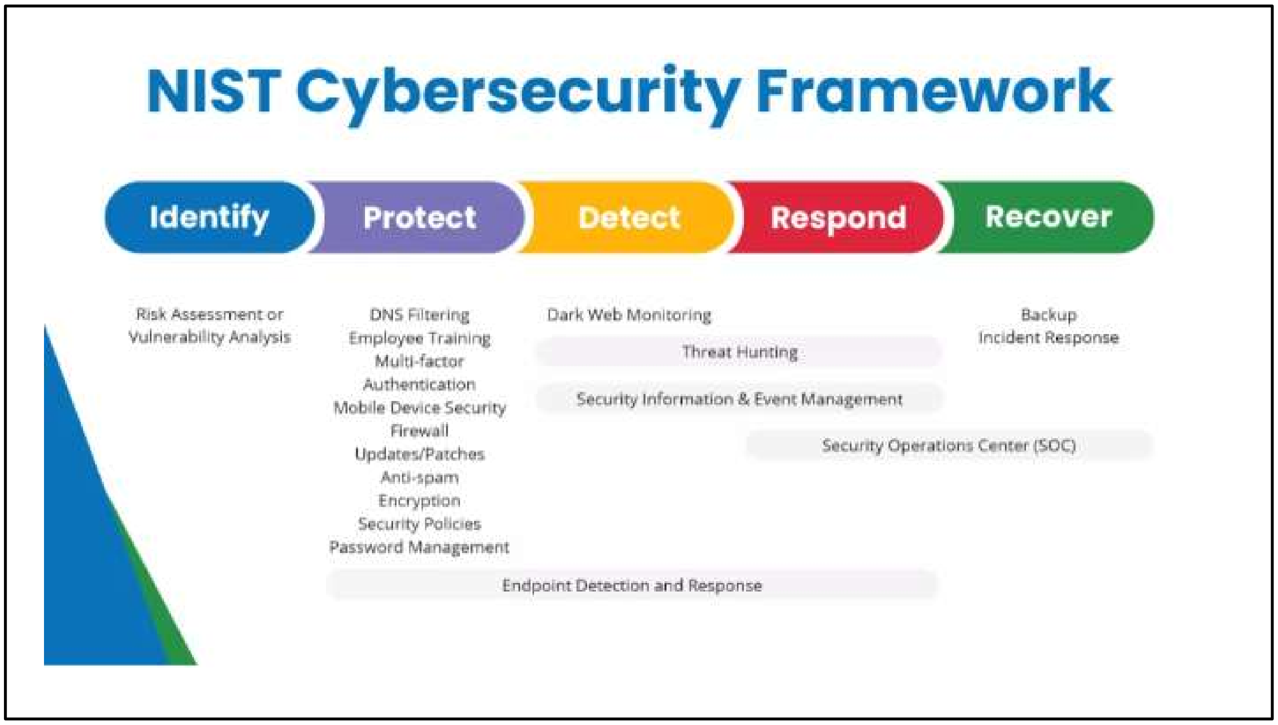
Threat Mitigation Strategies

- **Traffic Overload (Volumetric Attacks)**
    - **Cloud-based DDoS Protection Services** (e.g., Cloudflare, AWS Shield) to absorb excess traffic.
    - **Rate Limiting & Traffic Filtering** to block excessive requests.
- **Exploiting Network Vulnerabilities**
    - **Web Application Firewalls (WAFs)** to filter malicious traffic.
    - **Intrusion Detection & Prevention Systems (IDPS)** to monitor and block DDoS patterns.
- **Targeting Critical Infrastructure**
    - **Redundant Network Design & Load Balancing** to distribute traffic across multiple servers.
    - **Geographically Distributed Content Delivery Networks (CDNs)** to reduce load on a single location.

Key Takeaway:
DDoS attacks impact **Availability**, preventing users from accessing services. The best countermeasures involve **traffic filtering, load balancing, and scalable network security solutions**.

Final Summary

Phishing Attack: Confidentiality: Employee training, MFA, email filtering, web security
Accidental Data Overwrite: Integrity: Access control, versioning, backups, database logging
DDoS Attack: Availability: DDoS mitigation tools, traffic filtering, load balancing



Tutor Guidance: Explain the NIST Cybersecurity Framework
Tutor Notes:

The **NIST Cybersecurity Framework (CSF)**, which is a set of best practices, standards, and guidelines for improving an organization's cybersecurity posture. The framework consists of **five core functions**: **Identify, Protect, Detect, Respond, and Recover**, each of which plays a critical role in cybersecurity risk management.

1. Identify 🔵
Develop an understanding of cybersecurity risks to assets, systems, and data.

Key Activities:
- **Risk Assessment & Vulnerability Analysis:**
    - Identifying security risks by assessing vulnerabilities in networks, software, and infrastructure.
    - Common tools: **Nmap, Nessus, OpenVAS**.

Why it matters?
- Helps organizations **prioritize resources** to protect their most valuable assets.
- Reduces the likelihood of security breaches by **proactively identifying weaknesses**.

2. Protect 🟣
Implement security measures to safeguard assets from cybersecurity threats.

Key Activities:
- **DNS Filtering:** Prevents access to malicious websites by blocking known harmful domains.
- **Employee Training:** Educates staff on cybersecurity threats (e.g., phishing, social engineering).
- **Multi-Factor Authentication (MFA):** Adds an extra layer of security beyond just passwords.
- **Mobile Device Security:** Protects smartphones, tablets, and laptops from unauthorized access.
- **Firewalls & Encryption:** Ensures secure communication and data protection.
- **Security Policies & Password Managers:** Implements structured policies to enforce best security practices.

Why it matters?
- **Minimizes human error**, which is one of the leading causes of security breaches.
- **Strengthens defenses** against malware, phishing, and brute-force attacks.

3. Detect 🟡
Continuously monitor systems to quickly identify cybersecurity incidents.

Key Activities:
- **Dark Web Monitoring:** Tracks leaked credentials and sensitive data on the dark web.
- **Threat Hunting:** Actively searches for threats inside an organization's network.
- **Security Information & Event Management (SIEM):**
  - Centralized system that collects, analyzes, and reports on security data.
  - Common SIEM tools: **Splunk, IBM QRadar, Microsoft Sentinel**.
- **Endpoint Detection & Response (EDR):** Monitors devices for suspicious behavior.

Why it matters?
- Enables **early detection** of cyber threats before they escalate.
- Reduces the time between an attack and the organization’s response.

4. Respond 🔴
Take immediate action to contain and mitigate cybersecurity incidents.

Key Activities:
- **Security Operations Center (SOC):**
  - A dedicated team that **monitors, analyzes, and responds to threats** in real-time.
- **Incident Response Plan:**
  - Defines procedures to **contain, eliminate, and recover from cyberattacks**.
  - Key steps: **Identify, Contain, Eradicate, Recover, Learn**.

- **Threat Hunting & SIEM Integration:**
    - Uses collected security data to **track and remove active threats**.

Why it matters?

- A **fast and effective response** minimizes the impact of cybersecurity breaches.
- Prevents attackers from **gaining deeper access** to systems and networks.

5. Recover 🟢

Restore operations and minimize long-term impact after a cybersecurity incident.

Key Activities:

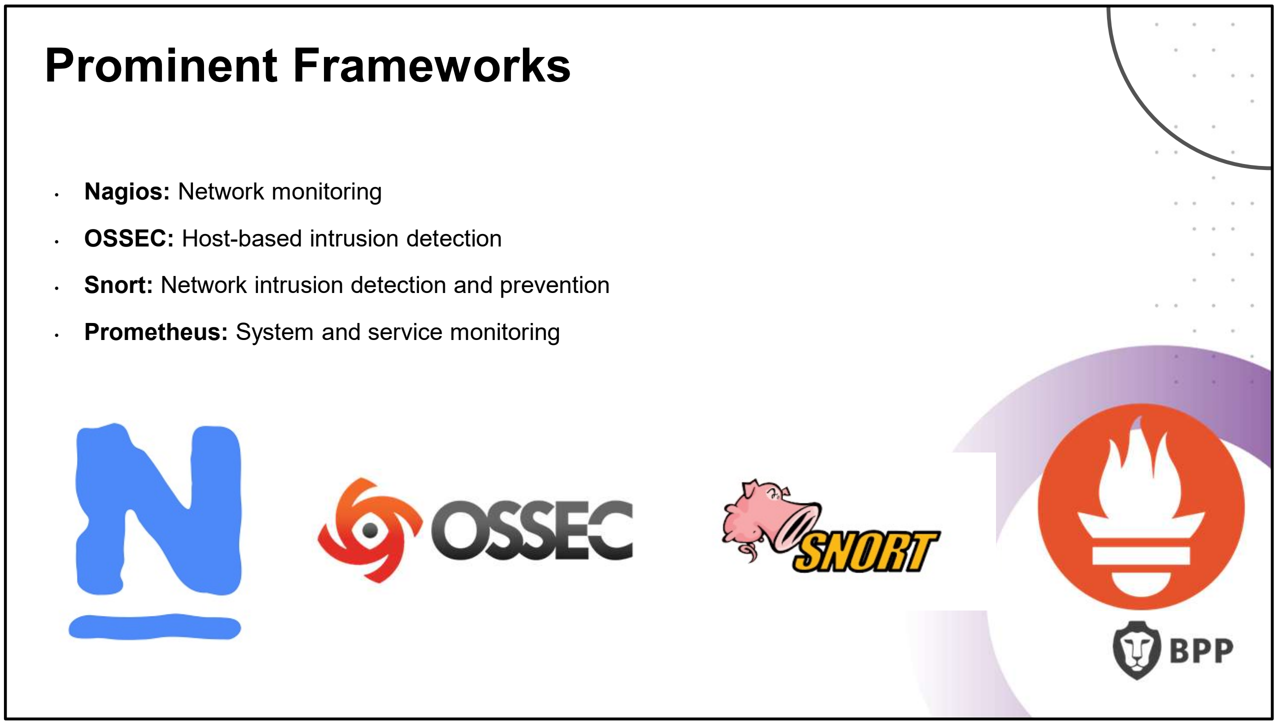
- **Backup & Disaster Recovery:**
    - Regular backups ensure that **data can be restored** after ransomware attacks or system failures.
- **Incident Response Review:**
    - Conducts a **post-mortem analysis** to learn from incidents and improve security defenses.
- **System Patching & Security Updates:**
    - Ensures vulnerabilities exploited in an attack are fixed.

Why it matters?

- Helps organizations **quickly resume business operations** after an attack.
- Ensures **continuous improvement** by analyzing what went wrong.

Final Thoughts

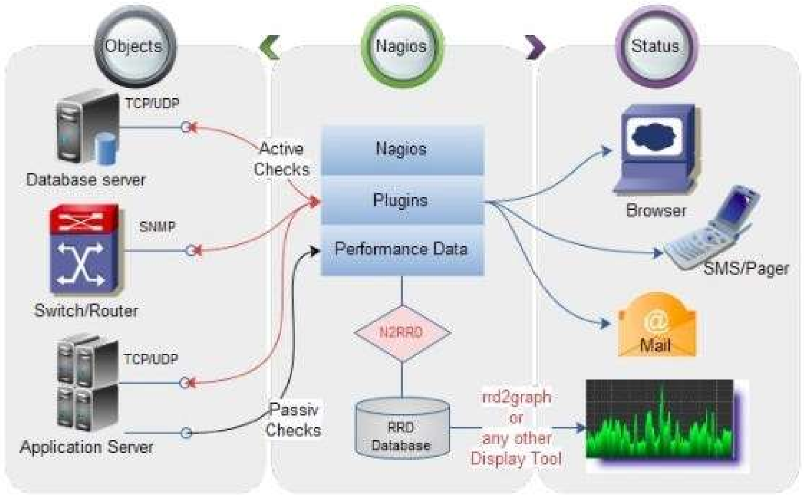
- The **NIST Cybersecurity Framework (CSF)** is widely used by organizations, including **governments, corporations, and financial institutions**, to **enhance cybersecurity**.
- The framework is **flexible** and can be adapted to different industries and risk levels.
- Implementing **all five phases** ensures **strong cybersecurity resilience**.



Prominent Frameworks

- **Nagios:** Network monitoring
- **OSSEC:** Host-based intrusion detection
- **Snort:** Network intrusion detection and prevention
- **Prometheus:** System and service monitoring

Architecture of Nagios



Explanation of Nagios Architecture

Nagios monitoring is widely used for **network, server, and application monitoring**. The architecture consists of three main sections: **Objects (Monitored Devices), Nagios (Monitoring System), and Status (Alerts & Reporting)**.

1. Objects (Monitored Devices)
These are the **network and system components** that Nagios monitors.

- **Database Server:** Monitored using **TCP/UDP** connections to check availability and performance.
- **Switch/Router:** Monitored via **SNMP (Simple Network Management Protocol)** to check network traffic and device health.
- **Application Server:** Also monitored over **TCP/UDP**, ensuring that applications are running properly.

Two Types of Monitoring Checks

1. **Active Checks (Red Arrows)**
   - Nagios **directly queries** devices at regular intervals to collect data.
   - Example: Nagios pings a server or checks if a website is accessible.
2. **Passive Checks (Black Arrows)**

- Devices send performance data **without being actively polled**.
- Example: A server automatically reports CPU usage to Nagios.

2. Nagios Core (Monitoring & Processing)

This is the **central monitoring engine** of Nagios, responsible for:

- **Nagios Core:** The main monitoring framework.
- **Plugins:**
  - Small scripts or programs that execute monitoring commands.
  - Examples: **Check HTTP, Check Ping, Check Disk Usage**.
- **Performance Data Processing:**
  - Collects and stores monitoring data in an **RRD (Round Robin Database)**.
  - Enables **historical trend analysis**.

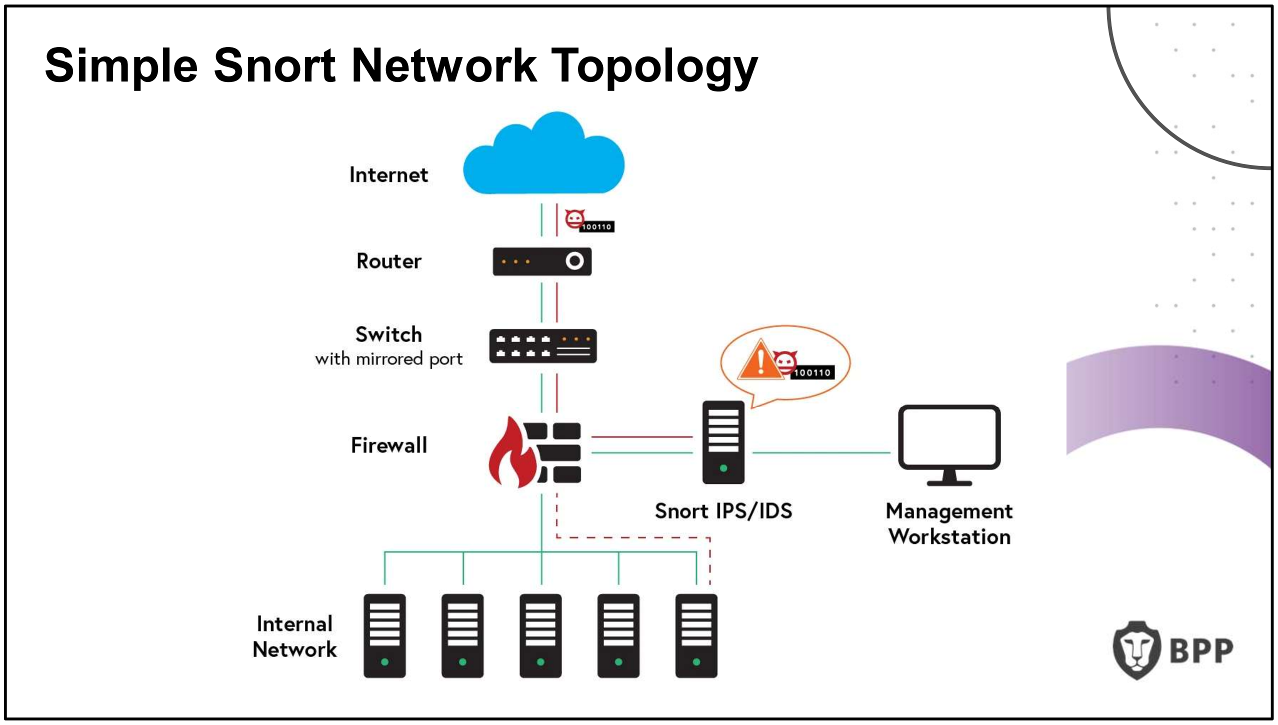
3. Status (Alerts & Visualization)

Nagios provides **real-time monitoring updates** through different alerting mechanisms:

- **Browser/Web Interface:**
  - Administrators can view system status, logs, and performance graphs.
- **Email Alerts:**
  - Notifications are sent if a system or service fails.
- **SMS/Pager Alerts:**
  - Critical failures can trigger text message alerts for quick action.
- **Graphical Reporting:**
  - Tools like **RRDtool (rrd2graph)** generate **graphs and dashboards** for trend analysis.

Key Takeaways

- **Nagios provides real-time monitoring** of IT infrastructure.
- **Active and passive checks ensure system health** and minimize downtime.
- **Alerts notify administrators** of failures to enable **quick resolution**.
- **Performance data storage** allows for **trend analysis and forecasting**.



Snort is a an open-source **Intrusion Detection System (IDS) and Intrusion Prevention System (IPS)**. This image represents a **network security monitoring setup** using Snort and how it integrates into a network to **detect and prevent malicious activities**.

1. **Key Components in the Network Setup**

Internet

- The **external network**, where potential cyber threats originate.
- Malicious traffic (indicated by the **red warning symbol**) can come from attackers, malware, or automated bots.

Router

- Directs traffic between the **Internet** and the **internal network**.
- Traffic is **monitored** for anomalies before passing through the **firewall**.

Switch with Mirrored Port

- The switch contains a **mirrored port** (SPAN Port) to **duplicate network traffic**.
- This allows Snort IDS/IPS to **inspect packets** without affecting network flow.

Firewall

- Acts as a **barrier** between the **internal network** and external threats.
- **Filters** traffic based on predefined security rules (e.g., blocking unauthorized access).

Snort IDS/IPS
- **Intrusion Detection System (IDS):**
    - Monitors traffic **passively** and raises alerts if suspicious activity is detected.
- **Intrusion Prevention System (IPS):**
    - Actively **blocks or drops** malicious traffic before it reaches the internal network.
- **Traffic Flow:**
    - The **switch sends mirrored traffic to Snort** for analysis.
    - If Snort detects a threat, it **generates alerts and can block** malicious connections.

Management Workstation
- Used by security teams to **analyze Snort alerts and logs**.
- Admins can **configure rules, investigate incidents, and respond** to security threats.

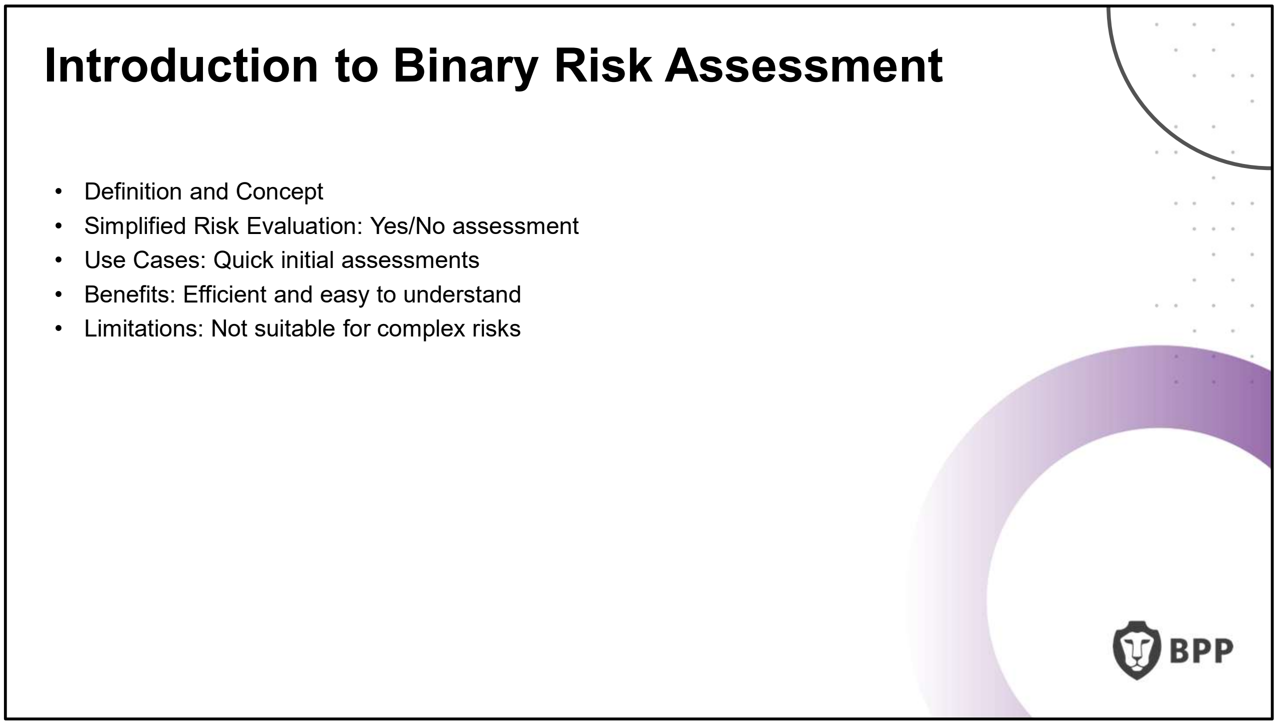
Internal Network (Servers)
- The **protected environment**, consisting of company resources, databases, and internal applications.
- If threats bypass the firewall, **Snort IDS/IPS helps mitigate risks** before they compromise the network.

Key Functions of Snort in this Setup
- **Packet Inspection:** Analyzes network traffic for malicious patterns using signature-based and anomaly-based detection.
- **Alert Generation:** Notifies security teams when potential threats (e.g., malware, DDoS, SQL injection) are detected.
- **Traffic Logging:** Captures and stores network traffic data for forensic analysis.
- **Blocking Malicious Activity:** In IPS mode, Snort can drop harmful packets before they reach internal servers.

3. Security Benefits of This Setup
- **Early Threat Detection** – Identifies cyberattacks before they cause damage.
- **Prevention & Response** – IPS functionality blocks known attack patterns.
- **Network Visibility** – Monitors all inbound/outbound traffic.
- **Centralized Security Management** – Security admins can **review alerts, update rules, and investigate threats** from a management console.

Summary
This architecture **combines multiple security layers** (firewall, Snort IDS/IPS, network monitoring) to provide **comprehensive protection** against cyber threats. By using **mirrored traffic**, Snort can efficiently **detect, analyze, and mitigate** attacks before they reach critical internal systems.
Would you like an example **Snort configuration file** to demonstrate how rules are

implemented? 🚀



Introduction to Binary Risk Assessment

- Definition and Concept
- Simplified Risk Evaluation: Yes/No assessment
- Use Cases: Quick initial assessments
- Benefits: Efficient and easy to understand
- Limitations: Not suitable for complex risks

Definition and Concept
Binary Risk Assessment is a **simplified risk evaluation approach** where risks are categorized into **two possible outcomes**: **Yes/No, Pass/Fail, Safe/Unsafe, or Acceptable/Unacceptable**. Instead of using complex risk matrices or probability scales, this method offers a **quick and straightforward way** to determine whether a risk is present and whether action is needed.

This approach is particularly useful when organizations need a **rapid, high-level evaluation** of risks before conducting a more detailed analysis.

Simplified Risk Evaluation: Yes/No Assessment
Binary Risk Assessment relies on a **Yes/No framework**, making it easy to:
- **Determine if a risk exists** (Yes = Present, No = Not Present).
- **Decide whether a control is effective** (Yes = Sufficient, No = Insufficient).
- **Assess compliance with policies** (Yes = Meets Requirements, No = Fails Requirements).

Example of Yes/No Binary Risk Evaluation:
Access Control: Are multi-factor authentication (MFA) and strong passwords enforced?
Data Encryption: Is sensitive customer data encrypted in storage and transit?
Backup Strategy: Are backups stored securely and tested regularly?
Network Security: Is a firewall properly configured and active?

A **"No"** response in any of these answers highlights areas requiring **immediate attention**.

Use Cases: Quick Initial Assessments
Binary Risk Assessments are useful in scenarios where **fast decision-making** is required, such as:

1. **Preliminary Security Audits:** Quickly checking if security policies are followed.
2. **Compliance Checks:** Determining whether an organization meets regulatory or internal security requirements.
3. **Incident Response Readiness:** Assessing if fundamental protections are in place before a cyber threat escalates.
4. **Vendor Risk Assessments:** Rapidly evaluating third-party security risks before deeper due diligence.
5. **Physical Security Checks:** Ensuring doors, surveillance, and access controls are properly functioning.

Example: A company onboarding a new cloud provider may use a binary checklist to confirm if **data encryption** and **access controls** meet internal security policies before engaging in further evaluations.

Benefits: Efficient and Easy to Understand

- **Fast Decision-Making:** Provides immediate risk insights without lengthy analysis.
- **Clear & Simple:** Easy to communicate to both technical and non-technical stakeholders.
- **Minimal Training Required:** Can be conducted by teams without deep cybersecurity expertise.
- **Standardized Assessment:** Ensures consistency in risk evaluation across different departments or vendors.

Example: A **CISO** (Chief Information Security Officer) can quickly determine whether **basic cybersecurity controls are in place** across multiple departments without needing extensive risk reports.

Limitations: Not Suitable for Complex Risks

- **Lacks Granularity:** Cannot measure risk severity or likelihood.
- **No Contextual Analysis:** Fails to account for **risk impact** on different systems.
- **Potential for False Confidence:** A "Yes" answer does not always mean full security compliance.
- **Not Effective for Dynamic Threats:** Binary risk assessment cannot track **evolving cyber threats** that require deeper risk modeling.

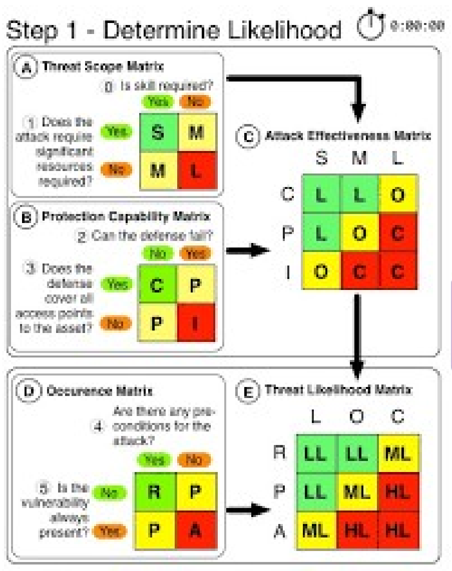
Example: A **Binary Risk Assessment** might mark a **website security** risk as "No Risk" if a firewall is present, **without considering** whether it is **misconfigured or outdated**—potentially leading to a false sense of security.

Summary
Binary Risk Assessment is **a useful tool for quick, high-level evaluations**, especially in time-sensitive situations or as an **initial filtering mechanism** before a deeper risk analysis. However, for **complex cybersecurity challenges** (e.g., evaluating the likelihood and impact of cyber threats), more **detailed risk assessment frameworks** like **NIST, ISO 27005, or FAIR (Factor Analysis of Information Risk)** should be applied.

Activity

Your tutor will guide you through Binary Risk Analysis:

https://binary.protect.io/


Step 1 - Determine Likelihood
0:00:00
A Threat Scope Matrix
0 Is skill required?
Yes No
1 Does the attack require significant resources required?
Yes No
S M
M L
B Protection Capability Matrix
2 Can the defense fail?
No Yes
3 Does the defense cover all access points to the asset?
Yes No
C P
P I
C Attack Effectiveness Matrix
S M L
C L L O
P L O C
I O C C
D Occurence Matrix
4 Are there any pre-conditions for the attack?
Yes No
5 Is the vulnerability always present?
No Yes
R P
P A
E Threat Likelihood Matrix
L O C
R LL LL ML
P LL ML HL
A ML HL HL

BPP

Determining Threat Likelihood
This diagram represents **determining likelihood** of a **Cybersecurity Threat**. It follows a **structured evaluation approach** to analyse how likely a threat is to occur. The process consists of five key matrices (**A-E**), each playing a role in defining the **likelihood of an attack**.

Step-by-Step Breakdown

A. Threat Scope Matrix

- **Purpose:** Determines the skill and resources required to execute an attack.
- **Key Questions:**
  - **Is skill required for the attack?**
    - If **Yes**, the attack is classified as **Small (S) or Medium (M) risk**.
    - If **No**, meaning anyone can execute it, it's classified as **Medium (M) or Large (L) risk**.
  - **Does the attack require significant resources?**
    - If **Yes**, the attack may be harder to execute (lower likelihood).
    - If **No**, the attack is easier to perform (higher likelihood).
- 🔹 **Output:** This matrix categorizes threats based on skill and resources, providing an

initial severity assessment.

B. Protection Capability Matrix

- **Purpose:** Evaluates how strong the current defenses are.
- **Key Questions:** 3. **Does the defense cover all access points?**
    - **Yes → Strong protection (C: Controlled, P: Protected).**
    - **No → Weaker security (P: Partial, I: Insufficient).**
    - **Can the defense fail?**
        - **Yes → Increased vulnerability to attack.**
        - **No → Stronger security posture.**
- **Output:** This helps **identify gaps in protection** and assess whether attackers can bypass defenses.

C. Attack Effectiveness Matrix

- **Purpose:** Combines data from **Threat Scope (A) and Protection Capability (B)** to measure how effective an attack might be.
- **How it Works:**
    - If an attack is **easy to execute (A)** and **defenses are weak (B)**, the attack effectiveness increases.
    - Categories: **Low (L), Occasional (O), Critical (C).**
- **Output:** This helps assess the **impact an attack may have if it occurs**.

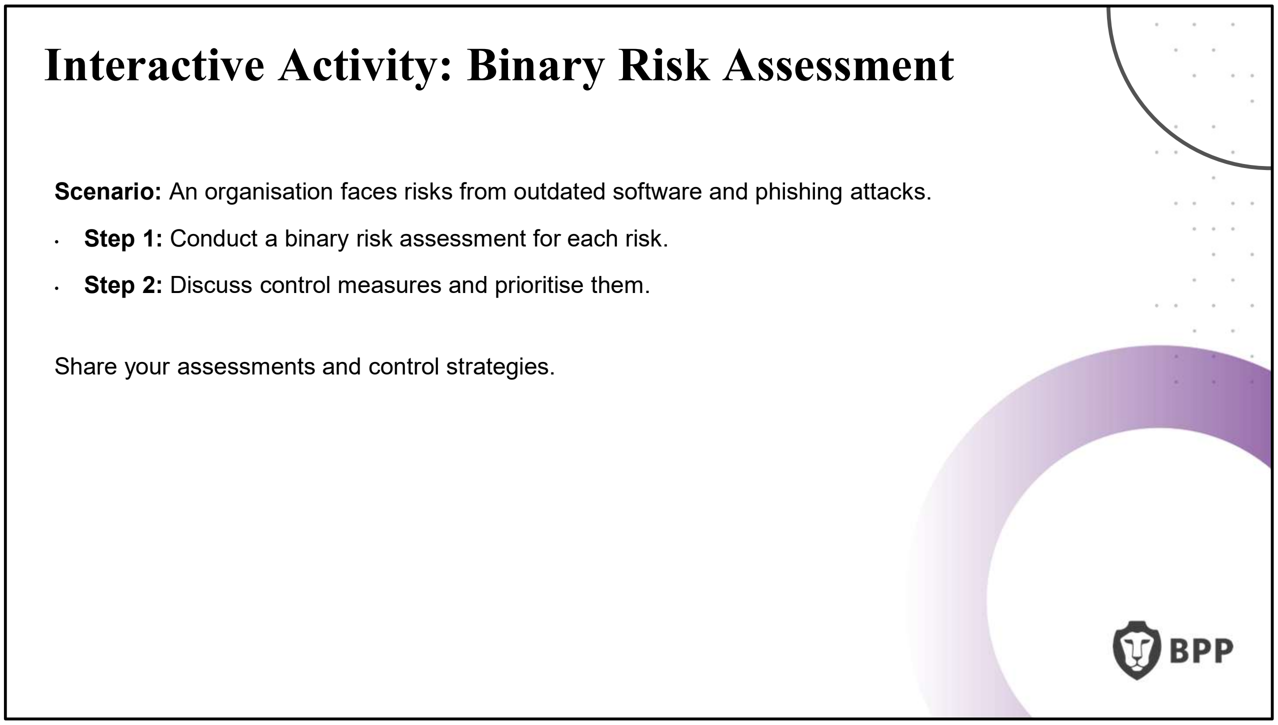
D. Occurrence Matrix
- **Purpose:** Assesses how often a vulnerability is present in the system.
- **Key Questions:** 5. **Are there any preconditions for the attack?**
    - **Yes → Reduced likelihood.**
    - **No → Higher likelihood.**
    - **Is the vulnerability always present?**
        - **Yes → Higher risk (A: Always vulnerable).**
        - **No → Lower risk (R: Rare, P: Partial exposure).**
- **Output:** Determines if the system **always remains vulnerable or if the risk is occasional**.

E. Threat Likelihood Matrix
- **Purpose:** Combines all **previous matrices (A-D)** to calculate the **final likelihood of an attack**.
- **Legend:**
    - **LL = Low Likelihood**
    - **ML = Medium Likelihood**
    - **HL = High Likelihood**
- **How it Works:**
    - **Lower risks in previous matrices = Lower likelihood.**
    - **Higher risks = Increased likelihood.**

- **Final Output:** A clear **risk likelihood rating** (Low, Medium, or High) to help **prioritize cybersecurity measures**.

Key Takeaways

- **Structured Threat Assessment:** Helps security teams analyze attacks step by step.
- **Identifies Weak Points:** Highlights where defenses may fail.
- **Guides Risk Mitigation:** Final likelihood rating helps **prioritize security fixes**.



Interactive Activity: Binary Risk Assessment

Scenario: An organisation faces risks from outdated software and phishing attacks.

- **Step 1:** Conduct a binary risk assessment for each risk.
- **Step 2:** Discuss control measures and prioritise them.

Share your assessments and control strategies.

Scenario: An organisation faces risks due to **outdated software** (which can be exploited by attackers) and **phishing attacks** (which can lead to credential theft and unauthorised access).

Step 1: Binary Risk Assessment
Each risk is assessed using a **Yes/No (Pass/Fail) approach** to determine if adequate controls exist.
Example questions:

- **Outdated Software:**
  - Are all systems and applications regularly updated and patched? **High**
  - Are legacy systems isolated or protected with compensating controls? **High**
- **Phishing Attacks**
  - Do employees receive regular phishing awareness training? **High**
  - Is Multi-Factor Authentication (MFA) enabled for critical accounts? **Medium**
  - Is an email security solution in place to filter phishing emails? **High**

Assessment Outcome:
- **Outdated Software** → **High Risk** due to missing updates.
- **Phishing Attacks** → **High Risk** due to lack of training and filtering, but MFA reduces

impact slightly.

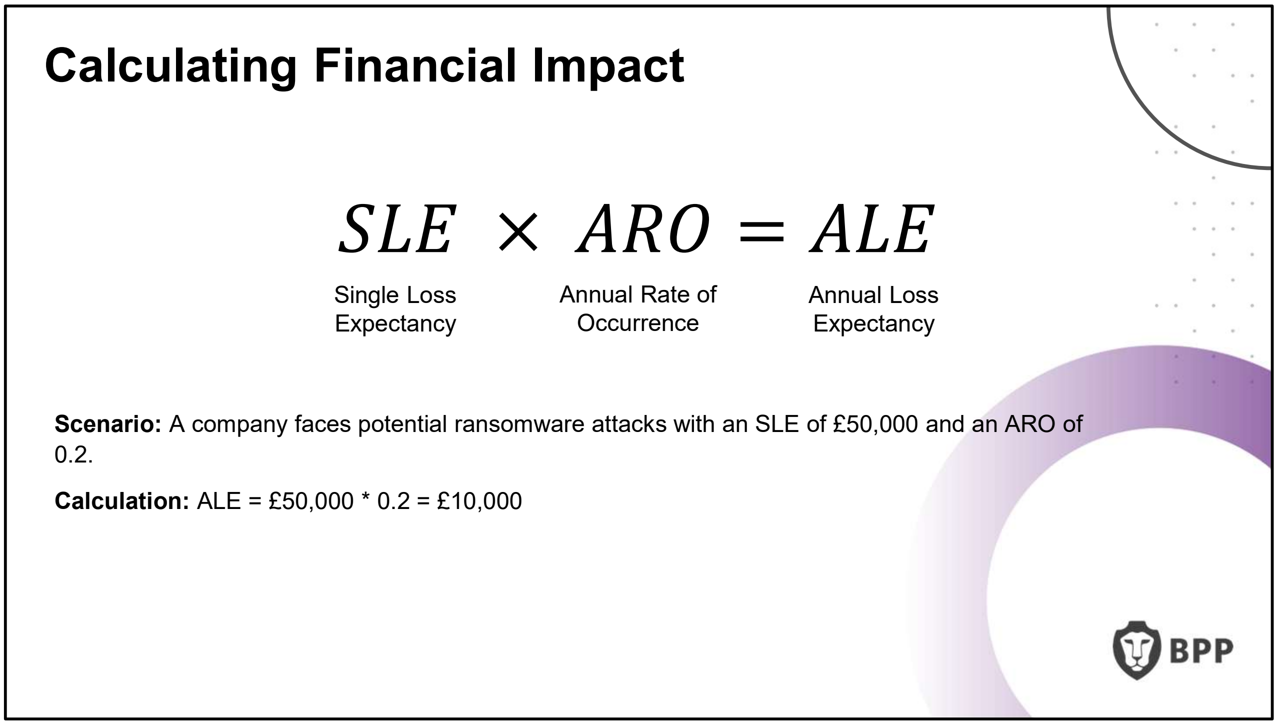
Step 2: Control Measures and Prioritization
To reduce risks effectively, we prioritize controls based on **urgency and impact**.

- **Priority Patch Management Policy** Implement **automated updates** and apply security patches for OS, applications, and firmware.
- **Vulnerability Scanning** Regularly scan for unpatched vulnerabilities and remediate risks.
- **Phishing Awareness Training** Conduct **quarterly phishing simulations** and train employees to recognize email threats.
- **Email Security (Spam Filtering & DMARC)** Deploy **email filtering tools** to block phishing attempts and **DMARC/DKIM/SPF** for email authentication.
- **Legacy System Hardening** If legacy systems cannot be updated, **isolate them**, use **firewall rules**, and restrict internet access.
- **Multi-Factor Authentication (MFA) Expansion** Extend MFA usage beyond critical accounts to all **email, VPN, and internal systems**.
- **Endpoint Detection & Response (EDR)** Deploy an **EDR solution** to detect exploitation of outdated software vulnerabilities.

Prioritization Strategy

1. **Immediate Fixes (High Priority 🔴)**
    1. **Patch management & vulnerability scanning** to close software gaps.
    2. **Email filtering & phishing protection** to stop phishing attempts before they reach users.
2. **Medium-Term Measures (Medium Priority 🟠)**
    1. **Employee phishing training** to reduce human error.
    2. **MFA Expansion** to add another security layer against phishing.
3. **Long-Term Solutions (Low Priority 🟡)**
    1. **Implement EDR & advanced monitoring** to detect sophisticated attacks.

Conclusion
By **immediately patching software** and **improving email security**, the organization **closes critical attack vectors**. Over time, **security training and MFA expansion** will further reduce risks.



Calculating Financial Impact

$$SLE \times ARO = ALE$$

Single Loss Expectancy

Annual Rate of Occurrence

Annual Loss Expectancy

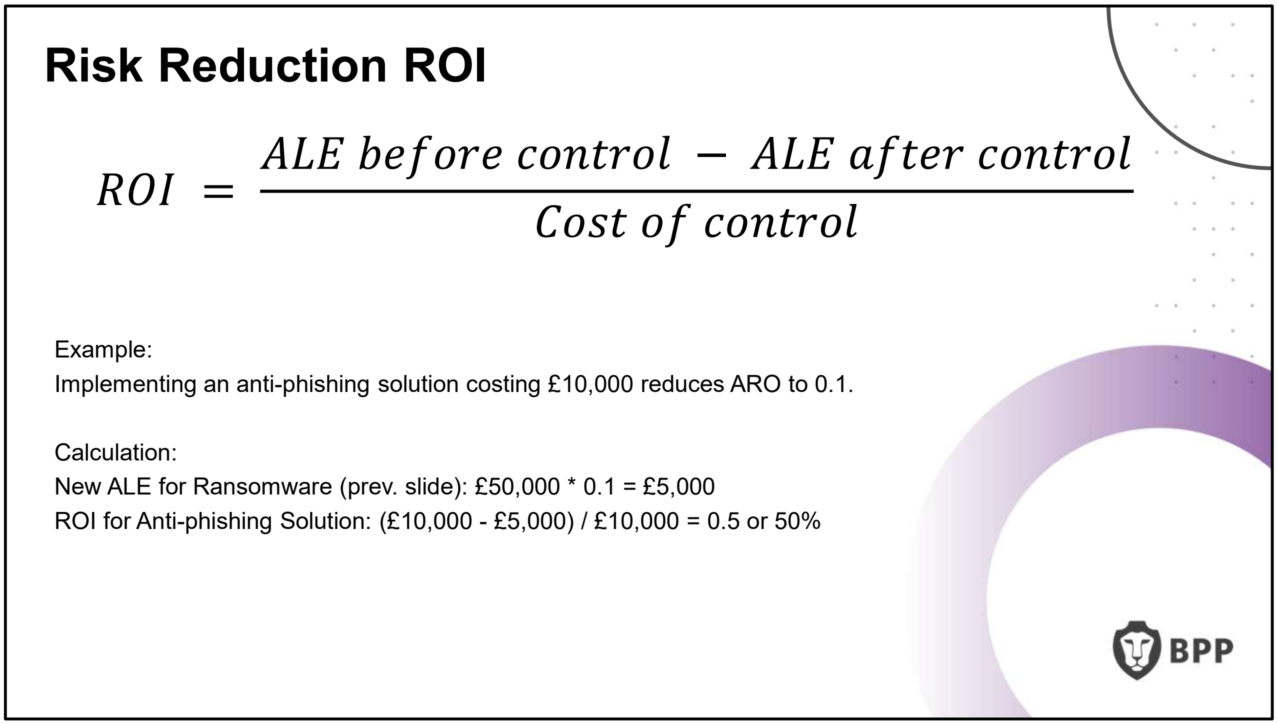
Scenario: A company faces potential ransomware attacks with an SLE of £50,000 and an ARO of 0.2.

Calculation: ALE = £50,000 * 0.2 = £10,000

SLE: Cost of a single incident

ARO: Expected frequency of an incident per year

ALE (Annual Loss Expectancy) = SLE * ARO



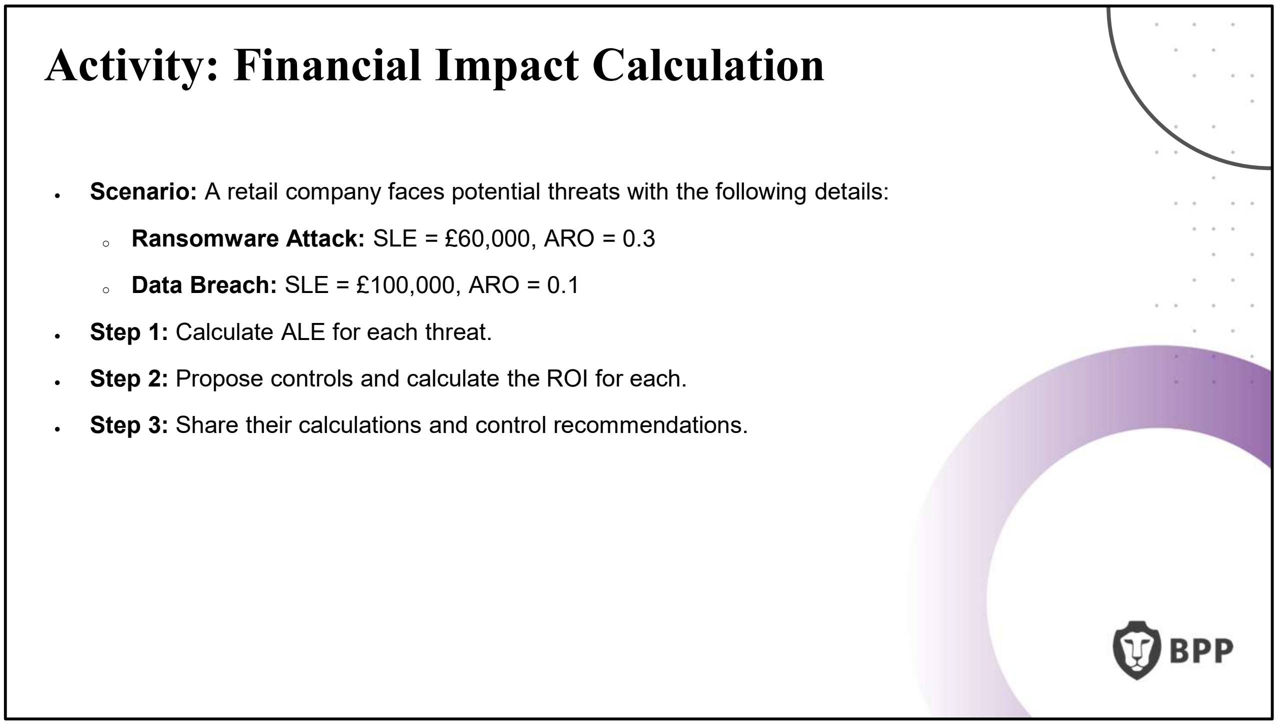
Risk Reduction ROI

$$ROI = \frac{ALE\ before\ control - ALE\ after\ control}{Cost\ of\ control}$$

Example:
Implementing an anti-phishing solution costing £10,000 reduces ARO to 0.1.

Calculation:
New ALE for Ransomware (prev. slide): £50,000 * 0.1 = £5,000
ROI for Anti-phishing Solution: (£10,000 - £5,000) / £10,000 = 0.5 or 50%

BPP



Activity: Financial Impact Calculation

- **Scenario:** A retail company faces potential threats with the following details:
  - **Ransomware Attack:** SLE = £60,000, ARO = 0.3
  - **Data Breach:** SLE = £100,000, ARO = 0.1
- **Step 1:** Calculate ALE for each threat.
- **Step 2:** Propose controls and calculate the ROI for each.
- **Step 3:** Share their calculations and control recommendations.

Scenario: A retail company faces threats from **Ransomware Attacks** and **Data Breaches**. Below are **recommended security controls** to mitigate these threats.

1. Ransomware Attack

A **ransomware attack** involves **malicious software that encrypts company data** and demands a ransom for decryption. Retail companies are **prime targets** due to their vast customer databases and reliance on online transactions.

- **Regular Data Backups:** Implement **frequent automated backups** (cloud & offline). Ensure **backups are immutable** and stored separately from the network.
- **Patch Management & Vulnerability Scanning:** Regularly update **operating systems, POS systems, and third-party applications** to close vulnerabilities.
- **Endpoint Detection & Response (EDR) :** Deploy **AI-based EDR tools** to detect and block ransomware before execution.
- **Least Privilege Access (Zero Trust Model) :** Restrict admin access. Employees should only have **minimum necessary permissions** to perform their tasks.
- **Email & Phishing Protection:** Use **email filtering solutions** (DMARC, SPF, DKIM) to block phishing emails that deliver ransomware.
- **Security Awareness Training:** Train employees on **recognizing phishing emails and suspicious attachments**. Conduct regular **ransomware simulation drills**.
- **Application Whitelisting:** Block unauthorized software execution. Allow only **approved**

applications to run on company devices.

- **Network Segmentation:** Separate **critical systems (POS, inventory, customer data)** from **general user access networks**. Limit **lateral movement** within the network.
- **Multi-Factor Authentication (MFA) :** Require **MFA for admin accounts, remote access, and cloud applications**.
- **Incident Response Plan for Ransomware:** Develop a **ransomware response plan** that includes **containment, recovery, and legal compliance steps**.

Key Focus: Backup strategy, email filtering, EDR, and network segmentation are crucial to stopping ransomware before it spreads.

Data Breach – Controls & Mitigation

A **data breach** occurs when **customer or company-sensitive information is leaked, stolen, or accessed without authorization**. This can lead to **financial losses, reputational damage, and legal consequences**.

- **Data Encryption (At Rest & In Transit):** Encrypt sensitive data **(customer information, payment details, employee records)** using **AES-256 and TLS encryption**.
- **Data Loss Prevention (DLP):** Deploy **DLP solutions** to prevent **unauthorized access or sharing of customer data** via email, USB, or cloud storage.
- **Access Control & Role-Based Access (RBAC):** Implement **least privilege access (Zero Trust)** to restrict sensitive data access based on **job roles**.
- **Network Monitoring & Intrusion Detection (IDS/IPS):** Use **Intrusion Detection/Prevention Systems (IDS/IPS)** to monitor traffic for signs of **unauthorized access**.
- **Regular Penetration Testing:** Conduct **quarterly penetration tests** to identify security weaknesses before attackers do.
- **Customer Data Masking & Tokenization:** Mask **stored customer payment data** and use **tokenization** to reduce exposure of plaintext sensitive information.
- **Third-Party Vendor Security Reviews:** Regularly audit **third-party services (payment processors, cloud storage providers)** for compliance with **security best practices**.
- **Security Information & Event Management (SIEM):** Deploy **SIEM tools (Splunk, IBM QRadar, or Microsoft Sentinel)** to analyze security logs for suspicious activity.
- **GDPR & PCI-DSS Compliance:** Ensure compliance with **privacy regulations** (GDPR, CCPA, PCI-DSS) by enforcing **data protection policies**.
- **Data Retention Policy:** Minimize **storage of sensitive customer data**. Implement **automatic deletion** of outdated records.

Key Focus: Data encryption, DLP, network monitoring, and compliance measures reduce the likelihood of a data breach.

Post-Webinar Quiz Questions

What does the CIA Triad stand for?
What are the core functions of the NIST Cybersecurity Framework?
How is ALE (Annual Loss Expectancy) calculated?

Useful Tools for Cybersecurity

SIEM Systems
IDS/IPS Tools
Encryption Software

Additional Resources for Further Reading

NIST Cybersecurity Framework
ISO 27001 Standards
Books and Articles on Cybersecurity

Courses and Certifications

Certified Information Systems Security Professional (CISSP)
Certified Ethical Hacker (CEH)

Group activity

See the worksheet (Word document) file