NETWORK SECURITY AND CRYPTOGRAPHY

UNIT 04

IPSec:

IPSec is a set of communication rules or protocols used to establish secure network connections. Internet Protocol (IP) is the common standard that controls how data is transmitted across the internet. IPSec enhances the protocol's security by introducing encryption and authentication. For example, it encrypts data at the source and then decrypts it at the destination. It also verifies the source of the data.

How does IPsec work:

IPsec works through five key steps:

1. **Host Recognition**: The system identifies packets needing protection, termed "interesting traffic," and applies relevant IPsec policies for encryption and authentication.

2. **Negotiation** (**IKE Phase** 1): The hosts negotiate security policies and authenticate each other to establish a secure channel. This can occur via:

- **Main Mode**: A secure tunnel is created for negotiation, making it more secure.
- **Aggressive Mode**: Faster setup without extensive negotiation.

3. **IPsec Circuit Setup** (**IKE Phase** 2): The hosts set up an IPsec circuit over the secure channel, agreeing on encryption algorithms and exchanging keys and cryptographic nonces, which are random numbers used to authenticate sessions.

4. **IPsec Transmission**: Data is exchanged through the established secure tunnel using the pre-defined security associations for encryption and decryption.

5. **Termination**: The IPsec tunnel is closed after a set amount of data or time, with the hosts disposing of the keys used during transmission.

Uses of IP Security:

IPsec can be used to do the following things:

- To encrypt application layer data.
- To provide security for routers sending routing data across the public internet.
- To provide authentication without encryption, like to authenticate that the data originates from a known sender.
- To protect network data by setting up circuits using IPsec tunneling in which all data being sent between the two endpoints is encrypted, as with a Virtual Private Network(VPN) connection.

IPsec components:

The components of IPSec are listed below:

1. **Authentication Header (AH)**: Provides authentication and integrity for IP packets but does not encrypt the data.
2. **Encapsulating Security Payload (ESP)**: Offers both encryption for confidentiality and authentication for integrity and authenticity.
3. **Internet Key Exchange (IKE)**: Used to negotiate security associations and establish keys for IPsec protocols. It has two phases: IKE Phase 1 and IKE Phase 2.
4. **IKEv2**: An updated version of IKE that improves efficiency and security, supporting mobility and multihoming.
5. **IPsec Policy Framework**: Defines how policies are set up and enforced for securing traffic.

These components work together to ensure secure communication over IP networks.

What is Email Security?

Email security refers to the steps where we protect the email messages and the information that they contain from unauthorized access, and damage. It involves ensuring the confidentiality, integrity, and availability of email messages, as well as safeguarding against phishing attacks, spam, viruses, and another form of malware. It can be achieved through a combination of technical and non-technical measures.

Some standard technical measures include the encryption of email messages to protect their contents, the use of digital signatures to verify the authenticity of the sender, and email filtering systems to block unwanted emails and malware, and the non-technical measures may include training employees on how to recognize and respond to phishing attacks and other email security threats, establishing policies and procedures for email use and management, and conducting regular security audits to identify and address vulnerabilities.

Why is email security important?

- **Protection against Cyberattacks:** Email is a top goal for cybercriminals. Malware, phishing attacks, and other threats often arrive via email. In fact, 94% of malware is delivered through email channels1. By implementing robust email security measures, organizations can defend against these threats.
- **Reducing Risk:** Cybersecurity incidents can have devastating consequences, including financial losses, operational disruptions, and damage to an organization's reputation. Effective email security helps protect your brand, reputation, and bottom line.
- **Compliance:** Email security ensures compliance with data protection laws like GDPR and HIPAA. By safeguarding sensitive information, organizations avoid legal fines and other intangible costs associated with cyberattacks.
- **Productivity Enhancement:** With email security in place, disruptions caused by threats like phishing emails are minimized. This allows organizations to focus more on business growth and less on handling security incidents.

Benefits of Email security:

- **Shielding against Phishing and Spoofing**: Email security acts like a digital bodyguard, detecting and stopping phishing and spoofing attacks that can lead to breaches and malware.
- **Locking Down Data**: Email encryption protects sensitive information like credit card and employee details, preventing leaks and costly data breaches.

- **Whispers Only**: Secure encryption ensures only the intended recipients receive messages, keeping confidential content private.
- **Spotting the Bad Apples**: It's like a powerful spam filter, identifying malicious and spam emails, so you don't fall for scams.
- **Top-Secret Protection**: Protects intellectual property, financial records, and classified info from hackers and cybercriminals.
- **Real-Time Guardian**: Provides real-time protection against threats like zero-day exploits, stopping malware and spam before they reach you.
- **Locking Up Identity Theft**: Encryption prevents attackers from stealing login credentials or personal data, guarding against identity theft.

Steps should be taken to Secure Email:

- **Choose a secure password:** Password must be at least 12 characters long, and contains uppercase and lowercase letters, digits, and special characters.
- **Two-factor authentication:** Activate the two-factor authentication, which adds an additional layer of security to your email account by requiring a code in addition to your password.
- **Use encryption:** It encrypts your email messages so that only the intended receiver can decipher them. Email encryption can be done by using the programs like PGP or S/MIME.
- Keep your software up to date. Ensure that the most recent security updates are installed on your operating system and email client.
- **Beware of phishing scams:** Hackers try to steal your personal information by pretending as someone else in phishing scams. Be careful of emails that request private information or have suspicious links because these are the resources of the phishing attack.
- **Choose a trustworthy email service provider:** Search for a service provider that protects your data using encryption and other security measures.
- **Use a VPN:** Using a VPN can help protect our email by encrypting our internet connection and disguising our IP address, making it more difficult for hackers to intercept our emails.
- **Upgrade Your Application Regularly:** People now frequently access their email accounts through apps, although these tools are not perfect and can be taken advantage of by hackers. A cybercriminal might use a vulnerability, for example, to hack accounts and steal data or send spam mail. Because of this, it's important to update your programs frequently.

Email Security Standards:

Some popular email security standards are:

1. PEM (Privacy-Enhanced Mail)
2. PGP (Pretty Good Privacy)
3. TLS (Transport Layer Security)
4. SPF (Sender Policy Framework)

Let's explain them:

1. PEM (Privacy-Enhanced Mail)

Privacy Enhanced Mail (PEM) is an email security standard to provide secure electronic mail communication over the internet. Security of email messages has become extremely important nowadays. In order to deal with the security issues of emails the internet architecture board has adopted it.

The PEM mainly provides the following services:

Confidentiality:
Confidentiality refers to the act of preventing unauthorized access to the information hence protecting it. The confidentiality is obtained in PEM by encrypting the messages by using various standard algorithms such as **Data Encryption Standard (DES)**. DES in cipher block chaining mode is being currently used by PEM.

Integrity:
Data integrity refers to the consistency of data throughout its life cycle. PEM ensures that the message has not been altered in transit. This is obtained by using a unique concept called as **message digest**.

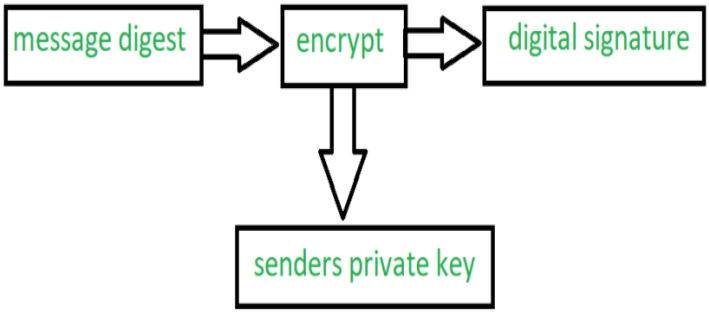
Working of PEM :

The PEM works basically in 4 main steps.

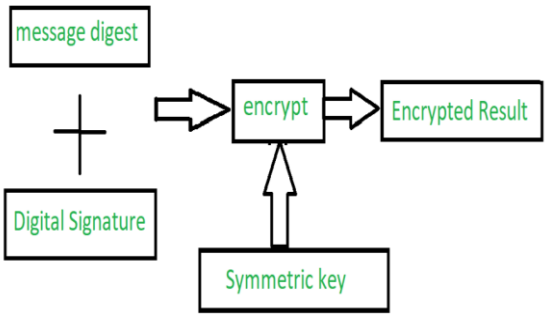
1. **Canonical Conversion:**
   This step involves the conversion of the message into a standard format that is independent of the computer architecture and the operating system of the sender and the receiver. If the sender and receiver has different computer architecture or operating system. It may lead to generation of different

message digest due to difference in their interpretation because of syntactical difference from one operating system to another.

2. **Digital Signature:**
   In this step, the digital signature is generated by encrypting the message digest of an email message with the sender's private key.



3. **Encryption**
   The encrypted message is generated by encrypting the original message and digital signature together along with the symmetric key as shown in the figure below. This step is very crucial in order to obtain the confidentiality.



4. **Base-64 Encoding:**
   This is the last step where the binary output is transformed into character output. The binary output which is 24 bits is divided into 4 equal sets and mapped with the 8 bit character output generating a decimal code. Now PEM uses a separate map table and each number from the code generated is

mapped with its corresponding value from the mapping table and binary equivalent corresponding to the 8 bit ASCII of the character is written.

Authentication:

Authentication refers to the process of verifying the identity of a user, device, or entity in a system. It ensures that the entity attempting to gain access to a resource is indeed who or what it claims to be. Authentication is a fundamental aspect of secure communication and access control in cryptographic systems, ensuring that only authorized users or systems can access sensitive data or perform certain actions.

Importance of Authentication in Cryptography:

- **Security**: Authentication is critical to ensuring that only authorized users can access sensitive systems and data, helping to prevent unauthorized access, data breaches, and fraud.
- **Integrity**: It helps ensure that the data or message received comes from a legitimate source and has not been tampered with during transmission.
- **Non-repudiation**: In some cases (e.g., with digital signatures), authentication provides proof that a specific entity performed an action, such as signing a document or sending a message, preventing them from denying their involvement later.

User authentication types:

1. Password-based authentication
2. Two-factor/multifactor authentication
3. Biometric authentication
4. Token-based authentication
5. Certificate-based authentication

Let's explain them in detail:

1. Password-based authentication:

Password based authentication is also known as **knowledge-based authentication**. Password-based authentication is a security mechanism that requires the user to enter their credentials; username and password, in order to confirm their identity. Once credentials are entered, they are compared against the stored credentials in the system's database, and the user is only granted access if the credentials match. It is a fundamental method used across various systems to restrict access to only authorized users.

Passwords are a knowledge factor i.e. something only the user knows.

> **Note:** Passwords are a sequence of characters (letters, numerals, special characters) that are used to authenticate a user's identity and grant access to a system, application, or device. They are typically created by the user and kept confidential.

How password authentication works:

Password-based authentication is intuitive for users: they enter the right credentials and they're granted access to a page or service. On the back end, however, there are a few more technical steps to authentication than users see on the login page.

Most password-based authentication systems follow a process in which:

1. The user creates an account by providing a unique identifier such as email, username, or phone number.
2. The user is prompted to create a password, which usually must meet certain complexity requirements.
3. The set of credentials is stored in the system's database, usually in an encrypted form to protect against data breaches.
4. When a user tries to log in, the authentication system checks their submitted credentials against those stored in its database.
5. If they match, the user is granted access.
6. If they don't match, the user will be denied entry and may be prompted to reenter their information or reset their password in case they forgot it.

2. Two-factor/multifactor authentication:

Two-factor authentication (2FA) is a security method that requires users to provide **two different types of verification** to prove their identity. These two factors typically come from two of the following categories:

1. **Something you know** (e.g., a password or PIN)
2. **Something you have** (e.g., a smartphone, security token, or smart card)
3. **Something you are** (e.g., biometric data like a fingerprint, facial recognition, or retina scan)

The idea behind 2FA is that even if one factor (like your password) is compromised, the second factor (like a one-time code sent to your phone) provides an extra layer of security.

Two-factor authentication adds an additional layer of security to the authentication process by making it harder for attackers to gain access to a person's devices or online accounts because, even if the victim's password is hacked, a password alone is not enough to pass the authentication check.

Two-factor authentication has long been used to control access to sensitive systems and data. Online service providers are increasingly using 2FA to protect their users' credentials from being used by hackers who stole a password database or used phishing campaigns to obtain user passwords.

Multi-Factor Authentication (MFA):

Multi-factor authentication (MFA) extends the concept of 2FA by requiring two or more verification factors. It uses a combination of factors to provide an even higher level of security. MFA doesn't limit you to just two factors; it can involve more than two, typically adding layers for critical systems.

3. Biometric authentication:

Biometric authentication refers to a cybersecurity process that verifies a user's identity using their unique biological traits such as fingerprints, voices, retinas, and facial features. Biometric authentication systems store this information in order to verify a user's identity when that user tries to access their account. Biometric authentication has much security, unlike other normal ones that utilize passwords or passcodes that can be easily guessed.

Types of Biometric Authentication:

1. **Facial recognition:** These systems use a person's unique facial features to identify them. It's used in a variety of places such as smartphones, credit card payments, and law enforcement.
2. **Fingerprint:** Fingerprint, which is unique to you, is used to verify the identity of a user. Fingerprint authentication can secure devices like phones and laptops, as well as cars and buildings. Fingerprint recognition is currently the most widely used biometric method in the world.
3. **Eye Recognition:** Eye recognition uses the unique pattern of someone's iris or retina to identify them. Eye recognition has very high accuracy; however, it isn't as popular due to its requirement for special infrared cameras and lighting. It is mostly found in places with high levels of security where extra precautions are needed.
4. **Voice Recognition:** Voice recognition uses the tone, pitch, and frequencies that are unique to an individual to authenticate them. Voice recognition is mainly used for authenticating purposes when dialing customer service numbers such as those offered by banks.

4. Token-based authentication:

A Token is a computer-generated code that acts as a digitally encoded signature of a user. They are used to authenticate the identity of a user to access any website or application network. It acts as a digital key that grants access to resources or services without requiring users to repeatedly enter their credentials (like username and password).

A token is classified into two types: **A Physical token** and **a Web token.**

Tokens make it difficult for attackers to gain access to user accounts. Attackers would need physical access to the token and know the user's credentials to infiltrate the account.

Employees must be trusted to keep track of their tokens, or they may be locked out of accounts. Because users are locked out if they forget or lose the token, companies must plan for a reenrollment process.

How token-based authentication works:

1. **Login Request:** The user submits login credentials (like username, password, or biometrics) via a website or app.
2. **Verification:** The authentication server verifies the credentials. If valid, it generates a token (usually in JWT format) with a header, payload (user info), and signature. This token is sent back to the user.
3. **Token Validation:** The user receives the token code and enters it into the resource server to grant access to the network. The access token has a validity of 30-60 seconds and if the user fails to apply it can request the Refresh token from the authentication server. There’s a limit on the number of attempts a user can make to get access. This prevents brute force attacks that are based on trial and error methods.
4. **Token Storage:** Once the resource server validated the token and grants access to the user, it stores the token in a database for the session time you define. The session time is different for every website or app. For example, Bank applications have the shortest session time of about a few minutes only.

5. Certificate-based authentication:

Certificate-based authentication involves the use of **digital certificates**, which are electronic papers, to confirm your identity. Mostly, this digital certificate verifies your identification by proving you are the owner of a private key, much as an electronic passport does.

A digital certificate is like an electronic passport used to prove your identity by confirming your ownership of a private key. Digital certificates contain:

- Identification data
- Public key information
- A digital signature derived from the private key of the certificate authority (CA) verified with their public key

How Does Certificate-Based Authentication Work?

Certificate-based authentication systems use certificates and **single sign-on** (**SSO**) to identify a person, machine, or device. The electronic passport is used to prove your identity. Authentication is achieved through the exchange of public keys, private keys, and certificate authorities (CAs).

Every public key has a corresponding unique private key. The associated private key is kept a secret, even while public keys are released. The only way to decode data encrypted with the public key is to have the matching private key. This provides increased security throughout the authentication process since every private key is unique to the person or device.

Maintenance certificates must be digitally signed by a third party (the CA) who vouches for your validity. The full login process is handled in your browser and the website you are dealing with

The process is generally as follows:

1. A user makes a request to access a protected resource.

2. The server presents its certificate to the browser, and the browser validates the public certificate.
3. An authentication request is made from the server for the user to authenticate themselves.
4. While the user is being authenticated, the browser presents the user's certificate to the server for validation.
5. The server authenticates the user's identity and allows access to the network.

Benefits of Certificate-Based Authentication:

- **Reduces insecure password practices:** Eliminates the need for shared logins or written passwords.
- **Enhanced security:** More secure than token- or SMS-based multi-factor authentication, especially with Trusted Platform Modules (TPM).
- **User-friendly:** Easy to use for end customers, as most systems support it out of the box with minimal user action.
- **Extensible to external users:** External partners like contractors and vendors can easily receive certificates for network access.

Drawbacks of Certificate-Based Authentication:

- **High cost:** Expensive to set up a digital infrastructure, making it less accessible for small businesses.
- **Adoption:** Limited use reduces its impact on improving overall online security.
- **Maintenance:** Requires continuous upkeep, including issuing, renewing, and revoking certificates.
- **Limited assurance:** Domain-validated certificates offer basic identity verification, which may be insufficient for high-security applications and can leave room for attacks.

What is Kerberos?

Kerberos is a network authentication protocol designed to provide secure authentication for users and services over an untrusted network, such as the internet.

It uses secret-key cryptography and a trusted third party, known as the Key Distribution Center (KDC), to provide mutual authentication between clients and servers, ensuring that neither party can impersonate the other. Kerberos operates by issuing time-limited tickets that grant access to resources, without transmitting passwords over the network, thereby mitigating the risk of eavesdropping and replay attacks.

Initially developed by the **Massachusetts Institute of Technology** (**MIT**), it's now a default authorization technology in Microsoft Windows and is also implemented in other operating systems like Apple OS, FreeBSD, UNIX, and Linux.

In Kerberos, Passwords are never sent over the network, reducing the risk of password theft. Kerberos uses tickets and session keys to authenticate users and services without repeatedly transmitting sensitive information.

The elements of Kerberos are:

- **Client**: The user or device that wants to access a service. Or we can say, a server or client that Kerberos can assign tickets to.
- **Authentication server (AS):** Server that authorizes the principal and connects it to the ticket- granting server.
- **Ticket-granting server (TGS):** Provides tickets.
- **Key distribution center (KDC):** A server that provides the initial ticket and handles TGS requests. Often it runs both AS and TGS services.

Working of Kerberos:

Step 1: Authentication Request

- The client (user) sends a request to the **Authentication Server (AS)** for authentication.

- The request typically includes the client's ID (e.g., username). The client doesn't send their password directly.

Step 2: Authentication Response (Ticket-Granting Ticket - TGT)

- The **AS** checks if the user exists in its database and retrieves the user's secret key, which is derived from the user's password.
- If valid, the **AS** responds by sending back two things:
    1. **Ticket-Granting Ticket (TGT)**: Encrypted using the KDC's secret key (not accessible to the client).
    2. **Session Key**: A session key encrypted with the user's secret key (derived from their password).
- The TGT contains the client's ID, timestamp, and a validity period, but it's encrypted with the KDC's key, so the client can't tamper with it.

Step 3: Decrypting the Session Key

- The client uses their password to decrypt the session key received from the AS. If successful, it proves that the client knows their password (authentication), but the password was never sent over the network.
- The client can now use this session key to interact securely with the Ticket-Granting Server (TGS).

Step 4: Requesting Access to a Service

- The client sends the **TGT** (received from the AS) to the **Ticket-Granting Server (TGS)**, along with a request for access to a specific service (e.g., file server).
- The TGT proves that the client has been authenticated.

Step 5: Service Ticket Issuance

- The **TGS** decrypts the TGT using its own key to validate the client's identity.
- If valid, the TGS issues a **Service Ticket** for the requested service. This Service Ticket is encrypted using the service's secret key (not accessible to the client).
- The TGS also sends another session key (called the **service session key**) encrypted with the client's session key (from the AS).

Step 6: Accessing the Service

- The client sends the **Service Ticket** to the requested service along with an authenticator (which includes a timestamp, encrypted with the service session key).
- The service decrypts the Service Ticket using its own secret key and verifies the timestamp in the authenticator.
- If everything is valid, the client is granted access to the service.

Step 7: Mutual Authentication (Optional)

- The service can optionally authenticate itself back to the client by sending a response encrypted with the service session key, completing mutual authentication.
- This assures the client that they are indeed communicating with the legitimate service, not an imposter.

How Kerberos works (in short):

1. **Client** requests authentication from the **AS**.
2. **AS** provides a **TGT** and session key.
3. **Client** uses the TGT to request access to a service from the **TGS**.
4. **TGS** issues a **Service Ticket**.
5. **Client** presents the **Service Ticket** to the service for access.

Key Distribution Center:

A Key Distribution Center (KDC): is a centralized server in a cryptographic network that is responsible for securely **generating**, **managing**, and **distributing** symmetric cryptographic keys to clients and services. It authenticates users and grants them access to network resources by issuing **session keys** and **tickets**, which are used to establish secure communication channels. KDC is a critical component in authentication protocols like Kerberos, where it facilitates secure key exchange and minimizes the need for users to directly share secret keys.

The KDC will use cryptographic techniques to authenticate requesting users, lookup their permissions, and grant them a ticket permitting access. The user can then present the ticket to the target resource/system, which verifies it and grants the user access.

Functions of a KDC:

1. **Authentication**: It authenticates users or systems requesting access to a network service.
2. **Key Generation**: The KDC generates symmetric session keys used for secure communication between parties.
3. **Key Distribution**: Once a user is authenticated, the KDC provides the session key securely to both parties, allowing them to communicate securely.
4. **Ticket Granting**: In protocols like Kerberos, the KDC issues a *Ticket Granting Ticket (TGT)*, which the user can use to request access to services within the network.

Components of a KDC:

- **Authentication Server (AS):** This part verifies the identity of the user and provides an initial ticket I,e Ticket Granting Tickets (TGT).
- **Ticket Granting Server (TGS):** This part issues service tickets based on the TGT, allowing access to specific network resources.

How KDC works:

1. User Authentication (AS Exchange)

- **Client Request (Step 1)**: The user (client) initiates communication by sending an authentication request to the **Authentication Server (AS)**. This request typically includes the user's identity (username).
- **AS Response (Step 2)**:
    - The **AS** checks the user's credentials (e.g., verifies a password or another secret).
    - If the credentials are correct, the AS generates a **Ticket-Granting Ticket (TGT)** and a session key.
    - The TGT is encrypted using the KDC's secret key, while the session key is encrypted using the user's password hash.
    - The AS sends both the encrypted TGT and session key back to the client.

2. Obtaining a Service Ticket (TGS Exchange)

- **Client Request (Step 3)**:
    - The client decrypts the session key using its password and stores the TGT for future use.
    - When the client wants to access a network service, it sends the **TGT** to the **Ticket Granting Server (TGS)** along with the service request.
- **TGS Response (Step 4)**:
    - The **TGS** decrypts the TGT using its own key to verify its validity.
    - If valid, the TGS generates a **service ticket** and a new session key for communication between the client and the requested service.
    - The service ticket is encrypted using the secret key of the requested service, while the session key is encrypted using the client's session key from the TGT.
    - The client receives both the service ticket and session key.

3. Accessing the Service (Client/Server Exchange)

- **Client Request (Step 5)**:
    - The client sends the **service ticket** and session key to the desired service (e.g., a file server or database).
- **Service Response (Step 6)**:

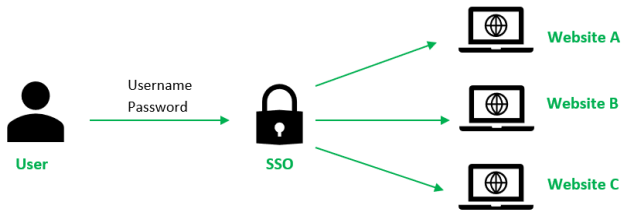
- The service decrypts the ticket using its own secret key and verifies the client.
- Once verified, the service allows the client access, and they can communicate securely using the shared session key.

Single sign on:

Single Sign-On (SSO) in cryptography is an authentication process that allows users to access multiple independent systems or applications with a single set of login credentials, such as a username and password. Instead of requiring separate logins for each system, SSO provides a seamless authentication experience by managing user credentials securely across different platforms.

For example: logging in to your Google account once will allow you to access Google applications such as Google Docs, Gmail, and Google Drive.

SSO works by sharing authentication information between different systems via secure tokens and establishing trust between them. This improves user convenience by reducing the need for multiple logins and enhances security by centralizing authentication and reducing password-related risks.



How does SSO Login work?

- The user enters login credentials on the website and the website checks to see if the user has already been authenticated by SSO solution. If so, the SSO solution would give the user access to the website. Otherwise, it presents the user with the SSO solution for login.
- The user enters a username and password on the SSO solution.
- The user's login credentials are sent to the SSO solution.

- The SSO solution seeks authentication from the **identity provider**, such as an **Active Directory**, to verify the user's identity. Once the user's identity is verified, the identity provider sends a verification to the SSO solution.
- The authentication information is passed from the SSO solution to the website where the user will be granted access to the website.
- Upon successful login with SSO, the website passes authentication data in the form of tokens as a form of verification that the user is authenticated as the user navigates to a different application or web page.

Advantages of Single Sign-On (SSO):

- **Improved User Experience**: Users only need to remember one set of login credentials, which reduces login fatigue and enhances convenience by eliminating the need for multiple logins.
- **Increased Productivity**: SSO reduces the time users spend logging into different systems, allowing them to focus more on tasks rather than repeatedly entering credentials.
- **Enhanced Security**: Centralized authentication allows for stronger security policies (e.g., enforcing multi-factor authentication).It reduces password reuse across systems, which is a common security risk. Fewer credentials means less attack surface for hackers to exploit.

Disadvantages of Single Sign-On (SSO):

- **Single Point of Failure**: If the SSO system or Identity Provider (IdP) is compromised or goes down, users may be locked out of all connected systems.
- **Initial Setup Complexity**: Implementing SSO can be technically complex and may require significant effort to integrate with existing systems and ensure compatibility with various applications.
- **Potential Security Risks**: If a user's credentials are compromised, the attacker could gain access to all connected systems.
- **Dependence on the Identity Provider**: Organizations relying on third-party IdPs like Google or Microsoft are dependent on those services being reliable and secure.

- **Logout Issues**: Single Logout (SLO) is not always implemented properly, meaning logging out from one service may not log users out from all connected services.

These notes are prepared by **Suhail Abass Hurrah** (S.P College Srinagar, batch 2019)
For any queries, you can email me at Hurrahsuhail1@gmail.com