Authentication and Authorization:

Authentication:
In distributed systems, authentication is the process of verifying the identity of nodes or users across a network. It ensures that entities trying to access resources or communicate with each other are who they claim to be. Common methods include cryptographic keys, digital certificates, and tokens.

Authorization:
Authorization in distributed systems governs the permissions and actions granted to authenticated entities. It defines what each entity is allowed to do within the system. Access control lists (ACLs), role-based access control (RBAC), and distributed policies play a crucial role in authorization.

Key Differences:

- **Scope:** Authentication verifies the identity of nodes or users, while authorization determines their access rights within a distributed system.
- **Sequence:** Authentication precedes authorization in the authentication and authorization process.
- **Focus:** Authentication confirms identity, while authorization specifies what actions or resources are permitted.
- **Methods:** Authentication relies on secure credentials, cryptographic keys, and certificates, while authorization employs policies and access control mechanisms.
- **Outcome:** Authentication establishes identity trust, while authorization enforces access control.

OAuth 2.0 (Open Authorization):

OAuth 2.0 enables secure access to user data without exposing credentials. It is commonly used for third-party application access to user data on platforms like Google and Facebook.

How it Works:

- Client Registration: Applications (clients) register with the authorization server.
- User Authentication: Users log in and grant permissions to the client.
- Authorization Grant: The client receives an authorization grant.
- Token Request: The client exchanges the grant for an access token.
- Access Token: The access token is used to access protected resources on the resource server.

JWT (JSON Web Tokens):

JWT is a compact, self-contained means of securely transmitting information between parties as a JSON object. It is often used for authentication and data exchange in web applications and APIs. For example, a user logs in, receives a JWT, and uses it to access protected resources or services.

How it Works:

- Header: Contains metadata about the token, such as the signing algorithm.
- Payload: Contains claims about the entity and additional data.
- Signature: Ensures the token is not tampered with.

SSO (Single Sign-On):

SSO is an authentication process that allows users to access multiple applications with a single set of credentials. It streamlines user access to multiple applications within an organization. For example, an employee logs in once and can access email, cloud storage, and other tools without repeated logins.

How it Works:

- User Authentication: Users log in once to the identity provider (IdP).
- Token Generation: Upon successful authentication, the IdP generates tokens.
- Token Validation: Tokens are validated by service providers (SPs) when users access their applications.
- Access Granted: Users gain access to SPs without re-entering credentials.