인증 전략

1. Basic Authentication

- 작동 방식:
  - 클라이언트가 보호된 URL에 접근하면, 서버는 `Authorization` 헤더에 포함된 자격 증명 을 확인한다.
  - 자격 증명은 Base64로 인코딩된 문자열로 전달되며, TLS/HTTPS와 함께 사용해야 안전하다.
  - 서버는 자격 증명이 올바르면 200 (OK), 그렇지 않으면 401 (Unauthorization) 응답을 보낸다.
- 장단점:
  - 간단하지만, TLS 없이 사용하면 매우 취약하다.

2. Session Based Authentication

- 작동 방식:
  - 사용자가 로그인하면 서버는 세션을 생성하고, 세션 ID를 클라이언트에게 전달한다.
  - 클라이언트는 이후 모든 요청에 세션 ID를 포함시켜 보낸다.
  - 서버는 세션 ID를 통해 사용자를 식별하고, 요청을 처리한다.
- 장단점:
  - Stateful 방식으로 서버는 세션을 메모리에 유지해야 한다. 로그아웃시 세션이 삭제된다.

3. Token Based Authentication

- 작동 방식:
  - 클라이언트가 로그인하면 서버는 토큰을 생성하여 클라이언트에게 전달한다.
  - 클라이언트는 이후 요청시 이 토큰을 사용하여 인증을 받습니다.

    - 토큰은 서버에서 저장하지 않고, 자체적으로 데이터를 포함하며, 유효성을 확인할 수 있다.
- 장단점:
    - Stateless 방식으로 서버에 세션을 저장하지 않아 확장성이 좋다. 그러나 토큰이 유출되면 보안 문제가 발생할 수 있다.

4. JWT (JSON Web Token) Authentication

- 작동 방식:
    - JWT는 헤더, 페이로드, 서명으로 구성된 세 부분으로 나뉜다.
    - 서버가 클라이언트의 자격 증명을 검증한 후, 비밀 키로 JWT를 생성하여 클라이언트에게 전달한다.
    - 클라이언트는 이후 요청 시 JWT를 사용하여 인증을 받는다.
    - 서버는 JWT를 검증하여 요청을 처리한다.
- 장단점:
    - 토큰에 포함된 데이터를 누구나 볼수 있지만, 서명된 데이터를 조작하는 것은 불가능하다. 자체 포함된 데이터로 인해 서버의 부하가 줄어든다.

5. OAuth (Open Authorization)

- 작동 방식:
    - OAuth는 사용자가 자신의 자격 증명을 제공하지 않고도 제 3자 애플리케이션이 자원에 접근할 수 있도록 한다.
    - 주로 OAuth 2.0 을 사용하며, 4가지 주요 인증 플로우 (Authorization Code Grant, Client Credential Grant, Password Grant) 가 있다.
    - 유저는 인증 서버와 상호작용하여 인증을 받고, 그 후 클라이언트는 토큰을 통해 보호된 리소스에 접근한다.
- 장단점:
    - 매우 유연하며, 특히 제 3자 서비스와의 통합에 유용하다. 그러나 설정과 구현이 복잡해 질수 있다.

6. SSO (Single Sign-On)

- 작동 방식:

    - 사용자가 한번의 로그인으로 여러 애플리케이션이나 서비스에 접근할 수 있도록 하는 인증방식이다.
    - 사용자는 ID 공급자를 통해 인증을 받으며, 이 인증 정보를 다른 서비스 제공자와 공유한다.
    - SAML 프로토콜을 주로 사용하여 IDP와 SP 간의 인증 정보를 교환한다.
- 장단점:
    - 사용자 편의성이 높고 기업환경에서 많이 사용된다. 그러나 SSO 시스템이 손상되면 모든 서비스가 위험해질 수 있다.

7. SAML (Security Assertion Markup Language)

- 작동 방식:
    - SAML은 XML 기반의 프로토콜로, 사용자 인증 정보를 안전하게 주고받기 위해 사용된다.
    - IDP와 SP 간에 인증 정보를 교환하여 사용자를 인증한다.
    - SAML Assertion이라는 XML 문서에 인증 정보가 포함되어 있으며, SP는 이를 검증하여 리소스 접근을 허용한다.
- 장단점:
    - 보안성이 높고 대규모 조직에서 자주 사용된다. 설정이 복잡하고 XML 기반이기 때문에 처리 속도가 느릴 수 있다.