인증(Authentication)?

- 사용자인지 아닌지를 확인. 주로 ID/PW 인증이 많이 사용되나 생체인식 등의 인증도 사용
- 서비스를 호출하는 서비스 클라이언트가 자신이라는 것을 어떻게 알 수 있을까?
- 사원증 발급

인가(Authorization)?

- 사용자가 권한이 있는지 없는지 확인하여 해당 행위를 제한하거나 가능하게 함
- 마이크로서비스를 호출하는 서비스 클라이언트가 수행하려는 작업을 수행할 자격이 있는지 어떻게 알 수 있을까?
- 사원증 찍고 게이트 통과

Rest API 인증 방법 4가지

1. HTTP Basic Authentication

- 사용자 이름과 암호 **Base64 인코딩**을 하여 서버로 전송
- 요청할 때마다 사용자 이름과 암호를 전송해야 함
- 중간자 공격(Man-in-the-middle attack)에 취약함

2. Cookies and Session

- 세션은 **키를 사용자 아이디에 매핑**함
- 세션이 존재하는 경우 키가 유효한 것으로 판단함
- 매 요청마다 쿠키에서 키를 보내 인증을 하여 세션이 유효한지 판단함
- REST API의 기본 원칙인 Stateless에 위배됨

3. API Keys

- 로그인 후 API 키를 전달받아서 **매 요청시마다 API키를 전달**하는 방법
- API 키가 유출될 시 보안문제가 생길 수 있음
- 주기적으로 키를 업데이트 하더라도, 서버와 클라이언트가 잘 맞지 않으면 서비스 장애로 이어짐
- Cloud 상징인 Auto Scaling 등으로 서버의 수가 변하고, IP가 유동적이라면 사실상 관리가 안됨

Rest API 인증 방법 4가지

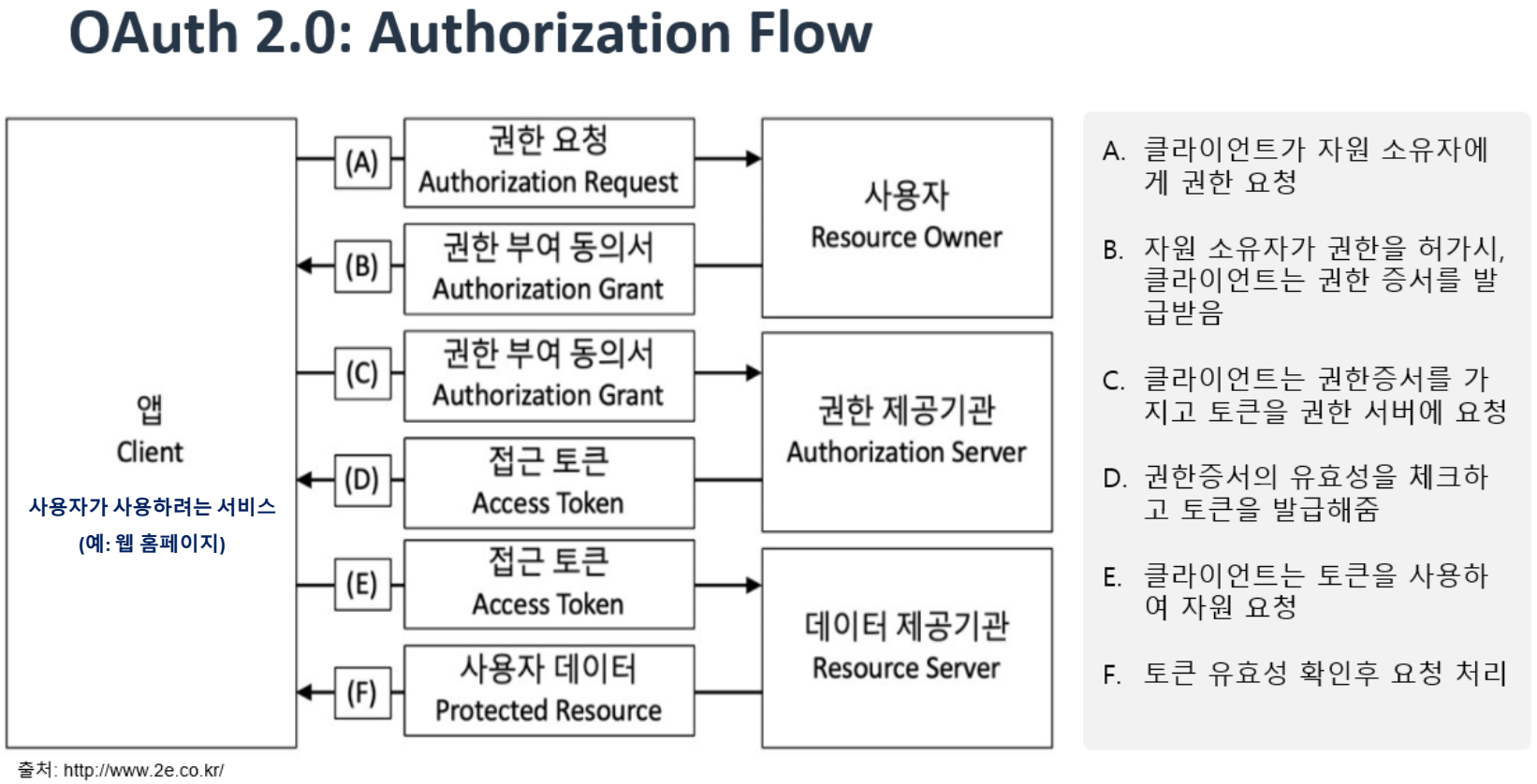
4. OAuth 2.0 (Open Authorization) (Token in HTTP Header)

- API Key의 단점을 보완하여 요청하는 서버와 요청 받는 서버 정도로 단순한 구성이 아닌,
  요청하는 서버/요청 받는 서버/인증하는 서버 등으로 세분화가 이뤄짐 ***(중립적인 위치에서 인증을 전담하는 구조)***
- OAuth 1.0 보다 인증 절차를 간소화 하고 다양한 인증 방식을 지원
- OAuth 1.0 은 디지털 서명 기반이었지만, **OAuth2.0의 암호화는 https에 맡김**으로써
  복잡한 디지털 서명에 관한 로직을 요구하지 않기때문에 구현 자체가 개발자입장에서 쉬워 짐

Oauth 2.0 (Open Authorization) ?

웹, 모바일 어플리케이션에서 ***타사의 API를 사용하고 싶을 때 권한 획득을 위한 프로토콜***

예를 들면, 어떤 Site를 들어갔을 때 "Google 계정으로 로그인", "카카오톡 계정으로 로그인"을 말할 수 있음.
Oauth 2.0을 이용하여 특정 서비스에 대한 회원가입을 거치지 않고 기존에 사용하던 서비스들의 계정으로 로그인을 진행하는 방법



출처: http://www.2e.co.kr/

* Authorization Server : 사용자의 동의를 받아서 권한을 부여하는 서버 (보통 Resource Server 의 하위 도메인)

* Resource Server : 서비스에 자신의 API를 제공하는 타사 서비스 (예: 구글,페이스북 등)