7주차 발표_민서연

API
: 보안과 OAuth 2.0

01 API 보안

의미!

API 보안은 개발자가 API를 설계하고 구현할 때 고려해야 하는 측면 중 하나
공격으로부터 API를 보호하는 프로세스

01 API 보안

필요한 이유

API는 공격자들의 대상이 되기 쉬운 환경에 놓여져 있다.
광범위한 접근이 혀용될 뿐만 아니라 웹 공격을 위한 요소들로 이루어져 있고
구조와 정보를 알아보기 쉽기 때문.

URI, 메소드, 헤더 및 파라미터 등

01 API 보안

취약점

BOLA (Broken Object Level Authentication)

BOLA 인증의 결함은 공격자가 데이터를 허락 없이 본다거나 수정하고 파괴하는 등의 결과로 이어지거나 전체 계정 해킹으로 이어질 수 있음.
API 공격에서 40%를 차지하고 있다.
API 동작에 대한 식별을 기존 보안 제어가 하지 못하게 마비 시킴.

01 API 보안

취약점

사용자 인증 실패

인증서 순환 기간, 긴 암호, 열악한 암호 위행과 취약한 암호 복잡성등 여러가지 요인이 존재함.
사용자 인증이 손상되였을 때, 공격자는 자격 증명 스터핑 및 무차별 대입 공격을 통해 애플리케이션 액세스 권한을 얻게 됨.

01 API 보안

취약점

과도한 데이터 노출

발생 가능성이 높은 보안 문제 중 하나로 과도한 데이터 노출임.
효율성을 위해 API에 필요한 것보다 더 많은 데이터를 공유하지만 이는 공격자가 데이터를 사용해 API에서 중요 정보를 추출할 수 있게 만든다.

01 API 보안

취약점

리소스 부족 및 속도 제한

API에는 클라이언트나 사용자가 요청할 수있는 리소스 수가 항상 제한 되는 것은 아니기 때문에, 데이터 가져오기를 담당하는 API에 대한 무차별 및 열거 공격에 노출.
공격자는 자격 증명 크래킹 및 토큰 크래킹을 포함해 제한이 없는 API에 대해 자동화된 공격을 가할 수 있음.

01 API 보안

취약점

보안 구성 오류

불완전한 구성, 잘못 구성된 HTTP 헤더, 너무 자세한 오류 메시지, 개방형 클라우드 스토리지 등 API의 취약점을 악용할 수 있도록 허술하게 설계된 보안 솔루션.
공격자는 이를 통해 API 구성 요소를 알아본 후 그에 맞는 공격을 감행할 수 있음.

01 API 보안

고려사항

인증 (Authentication)

API를 호출하는 사용자 또는 애플리케이션을 식별하는 프로세스.
API키, 토큰, 사용자 이름 및 비밀번호 등의 방법을 사용.

인가 (Authorization)

인증된 사용자가 특정 작업이나 리소스에 대한 액세스 권한을 가지고 있는지 확인.
사용자 역할, 권한 부여 및 RBAC(Role-Based Access Control)을 활용.

01 API 보안

고려사항

암호화 (Encryption)

데이터 전송 중에 암호화를 사용하여 중간에서의 데이터 탈취 방지함.

API 토큰 관리

효과적인 토큰 관리를 통해 무효화된 토큰의 사용을 방비하고 보안을 강화함.

02 OAuth 2.0

인증을 위한 개발형 표준 프로토콜!
이 프로토콜에서는 Third-Party 프로그램에게 리소스 유지를 대신해서
리소스 서버에서 제공하는 자원에 대한 접근 권한을 위임하는 방식 제공.

인증 및 권한 부여를 위한 개방형 표준 프로토콜로, 많은 웹 및 모바일 애플리케이션에서
사용됨.

대규모 회사에서도 사용함!




02 OAuth 2.0

핵심 컨셉

클라이언트

사용자의 데이터에 액세스하려는 애플리케이션 또는 서비스

리소스 소유자

자신의 데이터에 대한 액세스를 허용하는 사용자

02 OAuth 2.0

핵심 컨셉

인증 서버

사용자의 동의를 받아 클라이언트에게 액세스 토큰을 발급하는 서버

리소스 서버

클라이언트가 액세스하려는 데이터를 보유하고 있는 서버

액세스 토큰

클라이언트가 리소스에 접근할 때 사용되는 토큰.
유효기간이 있으며, 필요할 때 갱신 가능함.