스프링 부트 Security와 JWT를 통한 React 기반 로그인 시스템 설계

이영찬[1], 김민성[1], 유현우[1], 김민재[1], 민홍[2]
[1]가천대학교 AI·소프트웨어학부 학부생
[2]가천대학교 AI·소프트웨어학부 교수
{lych0918, kkw9430, youhyunwoo, minjae1134, hmin}@gachon.ac.kr

React-based login system design using Spring Boot Security and JWT

Youngchan Lee[1], Minsung Kim[1], Hyunwoo You[1], MinJae Kim[1], Hong Min[2]
[1]School of Computing, Gachon University
[2]School of Computing, Gachon University

요 약

스프링 부트는 개발 및 실행 환경 설정이 간편하기 때문에 백엔드 개발에 활용되는 프레임워크이고 React는 프론트엔드 개발에 활용되는 프레임워크이다. 본 논문에서는 스프링 부트와 React를 사용하는 웹 응용에서 로그인 시스템 구축 시 JWT를 활용하는 방법과 구조에 관해 설명하였다.

1. 서론

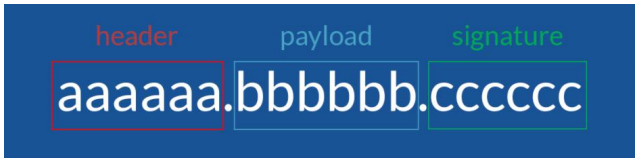
스프링 부트(Spring Boot)는 스프링 프레임워크 모듈 중의 하나로 RAD(Rapid Application Development)를 지원한다[1]. RAD는 웹 기반 대형 시스템 개발을 위한 환경 설정을 쉽게 해주는 장점이 있다. 이렇게 최소한의 환경 설정만으로 스피링 기반의 응용 프로그램을 개발하고 실행할 수 있어 많은 백엔드 개발자가 스프링 부트를 활용하고 있다. React는 JavaScript를 사용하는 오픈 소스기만의 프론트엔드 개발도구로 Facebook을 개발하는 데 사용된 프레임워크이다[2]. 본 논문에서는 스프링 부트를 백엔드 프레임워크로 React를 프론트엔드 기법으로 활용 시 적용할 수 있는 로그인 시스템에 대한 설계 및 Jason Web Token(이하 JWT)의 구조를 설명한다.

2. 로그인 시스템에서의 JWT 활용

JWT는 사용자 인증을 위해 널리 사용되는 토큰 기반 인증 시스템이다. JWT는 사용자의 세션을 서버 측에서 관리하는 것이 아닌 클라이언트가 자신의 필요 정보를 가지고 서버에 전송하는 방식으로 인증한다. JWT는 사용자의 인증 정보를 Jason 형식으로 인코딩하여 토큰을 생성하고 이 토큰을 클라이언트에게 발급한다. 클라이언트는 이 토큰을 HTTP 헤더에 포함시키고 서버에 전송하고 서버는 이를 받아 필터 체인 기반인 Spring Security을 통해 해당 토큰의 유효성을 판단하고 클라이언트의 요청에 응답한다. JWT의 핵심 강점은 토큰을 Header, Payload, Signature로 세 등분하는 방식으로 디지털 서명하여 데이터의 무결성과 신뢰성을 보장하는 것이다. 또한 토큰 자체가 필요한 모든 정보를 포함하고 있어 인증 서버에 대한 의존성이 낮다. 이는 서버의 부하를 감소시키고 네트워크 트래픽을 최적화하는데 기여한다. 이러한 JWT의 장점은 분산 시스템이나 마이크로 소프트 아키텍처에서 큰 이점으로 작용한다. 분산 시스템에서의 인증 처리는 종종 복잡하고 리소스 집약적인 작업이다. 각각의 서비스나 애플리케이션에서 별도의 인증 로직을 처리할 경우, 사용자 정보를 여러 시스템에 걸쳐 동기화하고 관리해야 하며 이는 네트워크 지연과 부하 증가로 이어질 수 있다. 이를 JWT를 이용하여 해결할 수 있다. 모든 인증 정보를 클라이언트가 관리하기 때문에 각 분산 시스템이 인증 정보를 저장할 필요가 없고 받은 JWT를 Spring Security와 같은 유효성 판별 로직만 있으면 되기 때문에 각각의 시스템이 독립적으로 유효성을 검증하고 사용자를 인증할 수 있다. 또한 이러한 장점은 분산 시스템의 확장성에도 크게 기여한다. 분산 시스템이 확장됨에 따라 새로운 서비스나 애플리케이션이 추가되어도 JWT는 일관된 방식으로 서비

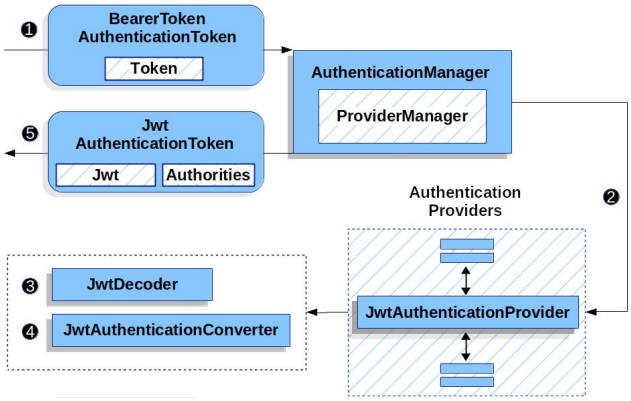
스 간의 인증을 처리하여 시스템의 유연성을 증가시키고 다양한 서비스가 통합되는 환경에서의 보안 요구를 충족시킬 수 있다. 이처럼 JWT는 분산 시스템에서 인증의 복잡성을 감소시키고 보안성을 유지하며 시스템 간 효율적 데이터 통신을 가능하게 하는 중요한 역할을 가진다.

그림 1은 JWT를 이루는 세 가지 요소로 Header는 암호화 알고리즘을 저장하고 Payload는 실질적 정보를 저장하고 Signature는 Header와 Payload를 묶어 JWT Secret Key를 사용하여 암호화해 변조를 막는다. 이때 Signature의 암호화 정보를 바탕으로 Payload나 Header의 조작 여부를 판단하여 보안성을 높인다.



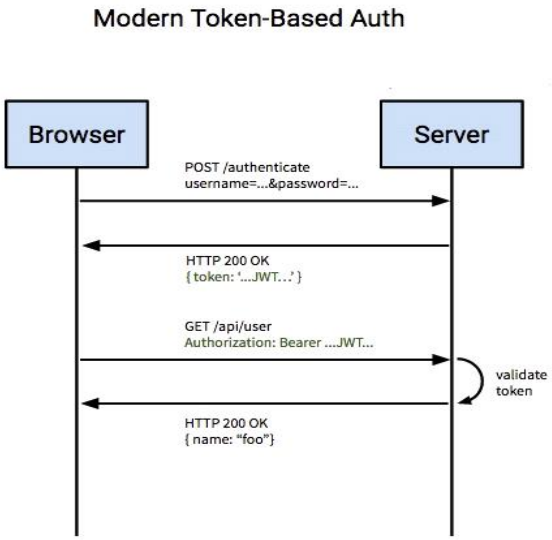
(그림 1) JWT 구조

그림 2는 JWT 인증 과정을 Spring Security에 접목하여 간략화한 것으로 JWT의 decoding과 converter로 유효성과 사용자 권한을 판별할 수 있다.



(그림 2) JWT 필터 인증 과정

그림 3은 JWT 기반의 클라이언트-서버 통신 과정을 간략하게 나타낸 것으로 사용자 인증이 완료될 시 서버 측에서 JWT 토큰을 HTTP Body에 담아 클라이언트에게 전달하고 클라이언트는 받은 JWT를 요청하는 API의 HTTP Header에 담아 서버에 리소스를 요청한다. 이때 Authorization Bearer 타입으로 넣어 IP, PW가 아닌 토큰 자체에 정보를 담기 때문에 무결성이 보장된다.



(그림 3) 클라이언트 서버 통신 과정

3. 결론

웹은 백엔드와 프론트엔드로 구분되며 다양한 프레임워크가 개발되고 있다. 본 논문에서는 스프링 부트를 백엔드로하고 React를 프론트엔드로 구성한 프레임워크에서 JWT를 활용한 로그인 시스템의 구조를 분석하고 인증 과정을 설명하였다.

참고문헌

[1] M. Mythily, A. A. Raj, and I. T. Joseph, "An Analysis of the Significance of Spring Boot in The Market" The 5th International Conference on Inventive Computation Technologies, Lalitpur, Nepal, 2022, pp.1277-1281.

[2] P. Rawat and A. N. Mahajan, "ReactJS: A Modern Web Development Framework", International Journal of Innovative Science and Research Technology, Vol.5, No.11, pp.698-702, 2020.