Presigned URL 기반 사용자 인증 과정 분석

한하영§, 서민재§, 김주령§, 이시연§, 방인규▪†, 김태훈▪§

§국립한밭대학교 컴퓨터공학과, †국립한밭대학교 지능미디어공학과

{20222517, 20201738, 20222021, 20242032}@edu.hanbat.ac.kr, {ikbang, thkim}@hanbat.ac.kr

Analysis of User Authentication Process based on Presigned URL

Hayeong Han§, Minjae Seo§, Juryeong Kim§, Siyeon Lee§, Inkyu Bang■†, Taehoon Kim■§

§Department of Computer Engineering, Hanbat National University

†Department of Intelligence Media Engineering, Hanbat National University

요 약

최근 웹 기반 가상환경의 수요 증가에 따라, 인증 절차의 보안성과 실시간성 확보는 필수 요소로 대두되고 있다. 본 논문은 Presigned URL을 활용해 사용자가 웹 브라우저에서 Kubernetes Pod에 안전하게 접속할 수 있는 인증 시스템을 제안한다. 서버는 인증된 사용자에게 일정 시간 동안 유효한 서명된 URL을 발급하며, 클라이언트는 이를 통해 WebSocket 기반 웹 터미널에 접속한다. 실험을 통해 제안한 방식이 무결성과 실시간성을 충족하며 안정적으로 동작함을 확인하였다.

Ⅰ. 서론

클라우드 네이티브 환경이 확산되면서, 사용자가 웹 브라우저를 통해 가상머신(VM)이나 컨테이너에 직접 접속할 수 있는 시스템이 주목받고 있다. 특히 교육, 테스트베드, DevOps 환경처럼 실시간성과 보안성이 모두 요구되는 상황에서는 효율적인 접근 제어가 필수적이다. 그러나 기존 인증 방식은 요청마다 인증 서버와의 통신이 필요하기 때문에 성능 저하와 서버 부하를 유발하며, 분산 환경에서는 보안 취약점으로 이어질 수 있다.

이를 해결하기 위해 본 연구에서는 Presigned URL을 기반으로 한 실시간 Kubernetes Pod 접속 인증 시스템을 제안한다.

제안된 시스템은 Presigned URL을 단순한 리소스 다운로드가 아닌 WebSocket 기반의 실시간 웹 터미널 인증 방식으로 확장하였다. 이 구조에서 사용자 인증과 권한 검증은 백엔드 서버에서 전담하며, 프론트엔드는 인증 과정에 개입하지 않아 클라이언트 측 보안 위협을 최소화한다. 가상머신 서버는 URL의 서명을 검증해 실시간으로 터미널 세션을 허용하거나 차단하며, 타임스탬프 기반 만료 기능을 통해 재사용 공격도 방지한다.

본 논문에서는 이와 같은 구조의 구현 방식과 실험 환경을 소개하고, Presigned URL 기반 인증의 유효성과 효과를 검증한 결과를 제시한다.

Ⅱ. 용어

Presigned URL은 일정 시간 동안 유효한 서명된 URL로, 서버가 사전 인증을 수행한 후 생성되며, 클라이언트는 별도의 인증 정보 없이 해당 URL을 통해 제한된 리소스에 안전하게 접근할 수 있다. 이 URL은 서버의 비밀 키로 생성된 HMAC 기반 서명을 포함하고 있어 위조가 불가능하며, 유효시간 내에서만 사용이 가능하다.

Ⅲ. 시스템 구현

1. 시스템 준비

본 시스템은 로컬 개발 환경에서 구축되었으며, 실험은 동일한 운영체제 하의 세 구성 요소에서 수행되었다. 각 구성 요소는 React, Spring Boot, Minikube를 기반으로 구동되며, 표 1은 실험 환경에 사용된 하드웨어 및 소프트웨어 사양을 시각적으로 나타낸 것이다.

표 1. 실험 환경 구성

구분	Frontend	Backend	Kubernentes
CPU	Apple M3 (8-core CPU)	Apple M2 (8-core CPU)	Apple M3 (8-core CPU)
RAM	16GB	16GB	16GB
OS	macOS Sequoia 15.4.1	macOS Sequoia 15.4.1	macOS Sequoia 15.4.1
Software	React 19.0.0	Spring Boot 3.4.2	Minikube (Parallels driver)

2. 시스템 구조

Presigned URL을 단순 리소스 다운로드가 아닌 실시간 가상머신 접속 인증 방식으로 확장하였다. 전체 시스템은 프론트엔드, 백엔드, 가상머신 서버로 구성되며, 각 요소는 보안성과 실시간성을 고려하여 독립적인 역할을 수행한다.

사용자가 웹 인터페이스에서 접속을 요청하면, 프론트엔드는 해당 요청을 백엔드에 전달한다. 백엔드는 JWT 기반 사용자 인증

■ Corresponding Authors: Inkyu Bang (ikbang@hanbat.ac.kr), Taehoon Kim (thkim@hanbat.ac.kr)

후, Kubernetes API를 통해 사용자의 Pod 상태를 확인하고 필요 시 새로운 Pod를 동적으로 생성한다. 이후 생성된 Pod의 접근경로와 정보를 기반으로 유효시간과 서명된 파라미터를 포함한 Presigned WebSocket URL을 생성한다. 프론트엔드는 이 URL을 받아 가상머신 서버에 WebSocket 연결을 시도하고, 가상머신 서버는 URL의 서명을 검증하여 유효한 경우에만 접속을 허용한다.

또한 Presigned URL은 타임스탬프 기반 만료 기능을 통해 재사용 공격(replay attack)을 방지하며, URL 내부 파라미터를 통해 특정 세션 또는 Pod에 대한 정밀한 접근 제어가 가능하다. 이러한 구조는 인증 서버의 부하를 줄이면서도 보안성과 확장성을 동시에 만족시킨다. 그림 2는 시스템이 AWS VPC 환경 내에서 React 프론트엔드, Spring Boot 백엔드, Redis, MySQL, Kubernetes 등을 배치한 아키텍처를 보여준다. 그림 3은 사용자의 접속 요청 시 Presigned URL이 발급되는 과정을 흐름도로 나타낸 것이다. 사용자 인증, 사용자 존재 여부, 컨테이너 상태 확인 단계를 거쳐 URL이 성공적으로 생성되며, 조건 불충족 시 예외 처리 경로로 분기된다.

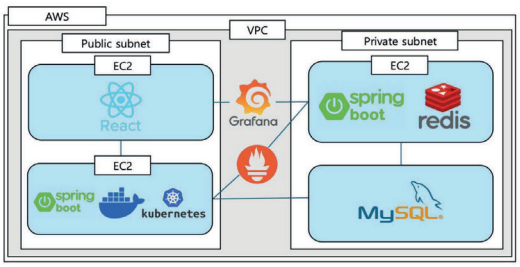

그림 1. 시스템 구조

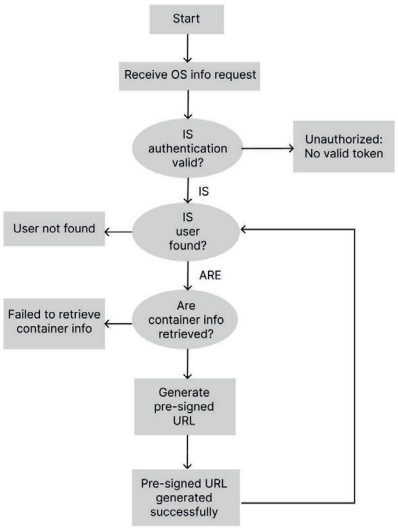

그림 2. 시스템 흐름

Ⅳ. 테스트 및 결과

본 시스템은 256비트 길이의 랜덤 키를 생성한 후, 이를 Base64로 인코딩하여 발급 서버와 검증 서버 간의 대칭키로 공유한다. 발급 서버는 해당 키를 Base64 디코딩한 뒤 HMAC 서명 키로 변환해 저장하며, 이후 사용자 식별 정보(userId), Pod 메타데이터, 유효기간 5분을 포함한 Presigned URL을 생성하고 서명하여 사용자에게 발급한다.

검증 서버는 전달받은 URL에서 정보를 추출하고, 동일한 대칭키로 서명의 무결성과 유효성을 검증한다. 이 검증 과정은 JWT 서명 방식과 유사하며, 인증이 유효할 경우만 WebSocket 접속이 허용된다. 인증에 성공하면 그림 4와 같이 웹쉘 인터페이스가 정상적으로 출력되며, 실패 시에는 HTTP STATUS 403 오류와 함께 접속이 차단된다.

실험에서 Pod 내부 자원 접근에 용이하게 Apache 웹 서버를 이용한 웹쉘 환경을 구성하여 작동하는 Pod를 생성하였다. 해당 Pod에 접근하기 위해서는 Presigned URL이 반드시 요구되며, 서명된 메타데이터와 유효시간 기반으로 접근이 통제된다.

그 결과, Presigned URL 기반 인증은 Kubernetes 환경에서 안전하고 유효한 접근 제어 수단으로 효과적으로 작동함을 확인할 수 있었다.

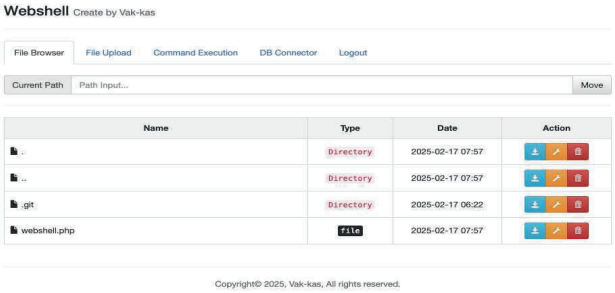

그림 3. Presigned URL 검증 성공 후 Pod 접속

Ⅴ. 결론

본 논문에서는 Presigned URL을 활용하여 사용자의 실시간 Kubernetes Pod 접속을 안전하게 제어하는 인증 시스템을 설계하고 구현하였다. 제안한 방식은 서버 측에서 서명된 URL을 발급하고, 클라이언트는 이를 기반으로 별도의 인증 절차 없이도 제한된 리소스에 접근할 수 있도록 구성된다. 실험을 통해 Presigned URL 기반 인증이 무결성과 보안성을 갖춘 실시간 인증 구조로서 효과적으로 동작함을 확인하였다.

특히, 타임스탬프 기반 만료 기능과 서명 검증 메커니즘은 재사용 공격을 방지하고, WebSocket 기반 웹 터미널 환경에서도 안정적인 인증 절차를 가능하게 했다.

향후에는 본 구조를 다중 사용자 환경이나 자동 확장 클러스터에서도 적용 가능하도록 개선하고, 다양한 리소스 유형(PVC, Service 등)에 대한 세밀한 권한 제어 기능을 추가함으로써, 클라우드 기반 가상 환경 전반에서의 실용성을 더욱 높일 수 있을 것으로 기대된다.

ACKNOWLEDGMENT

본 연구는 국립한밭대학교 공학교육혁신센터 “창의융합형공학인재양성지원사업”, 2025년 과학기술정통신부 및 정보통신기획평가원의 SW중심대학사업의 지원으로 수행되었음(2022-0-01068)

참 고 문 헌

[1] AWS Documentation
[2] D. Hardt and M. Jones, “JSON Web Token (JWT) Profile for OAuth 2.0 Access Tokens,” IETF RFC 9068, May 2021.