제로 트러스트 기반 디바이스 인증 체계의 설계 및 구현에 관한 연구

유용상[§], 장예나[§], 방인규▪[†], 김태훈▪[§]

[§]국립한밭대학교 컴퓨터공학과, [†]국립한밭대학교 지능미디어공학과
{20211884, 20222006}@edu.hanbat.ac.kr, {ikbang, thkim}@hanbat.ac.kr

A Study on the Design and Implementation of Zero Trust-Based Device Authentication Architecture

Yongsang Yu[§], Yena Jang[§], Inkyu Bang▪[†], Taehoon Kim▪[§]

[§]Department of Computer Engineering, Hanbat National University
[†]Department of Intelligence Media Engineering, Hanbat National University

요 약

본 논문은 기존 식별자 기반 디바이스 인증의 스푸핑 취약성, 분실·도난 기기 문제, 확장성 부족 등을 개선하기 위해 제로 트러스트(Zero Trust) 보안 모델을 적용한 새로운 인증 체계를 설계·구현하였다. 제안된 시스템은 브라우저에서 생성된 고유 식별자(dev_id), 서버 서명 토큰(devtk), 그리고 TOTP 기반 다중 인증을 결합해 사용자와 디바이스를 동시에 검증한다. 승인 메일 기반 관리자 승인 절차를 통해 미등록 기기의 접근을 제어하며, User-Agent와 IP 정보를 해시 처리해 세션 환경의 무결성을 보조적으로 검증하였다. 본 구조는 물리 식별자 수집 없이 프라이버시를 보장하며, 경량 환경에서도 안전하고 확장 가능한 제로 트러스트 인증을 실현하였다.

Ⅰ. 서 론

디바이스 인증(Device Authentication)은 네트워크 보안의 기초이나, 전통적인 식별자 기반(identifier-based) 방식은 스푸핑 및 분실·도난 기기 등 다양한 취약점을 가진다 [2]. 이러한 문제를 해결하기 위해 최근에는 네트워크 경계 자체를 신뢰하지 않는 제로 트러스트(Zero Trust) 보안 패러다임이 주목받고 있다 [1]. 본 연구는 NIST SP 800-207에서 정의한 제로 트러스트 모델 [1]을 디바이스 인증 구조에 적용하여, 지속 검증(Continuous Verification)과 최소 권한 부여(Least Privilege)를 실현하는 새로운 인증 체계를 제안한다.

Ⅱ. 시스템 설계

본 연구에서 구현한 제로 트러스트 기반 디바이스 인증 시스템은 브라우저에 생성·저장된 장치 식별자(dev_id) 와 서버 측 서명 토큰(devtk), 그리고 TOTP 기반 사용자 인증을 결합한 구조로 설계되었다. 설계 원칙은 제로 트러스트의 핵심인 지속 검증(continuous verification), 최소 권한(least privilege), 그리고 운영상 프라이버시 준수이다. 특히 본 시스템은 운영기관의 정책·법적 제약을 고려하여 물리적 MAC 주소 등 저수준 식별자를 수집하지 않도록 설계되었다. 시스템의 주요 흐름은 다음과 같다.

1) 첫째, 클라이언트는 최초 접속 시 localStorage에 임의의 UUID(dev_id)를 생성하고 저장한다. 로그인 요청 시 이 dev_id를 서버로 함께 전송한다.
2) 둘째, 서버는 해당 dev_id가 사용자 계정에 등록되어 있는지 확인한다. 등록되지 않은 경우 서버는 승인 메일을 사용자에게 발송하며, 메일 내 승인 링크(/auth/device/approve?...)를 통해 사용자가 직접 해당 기기를 등록하도록 유도한다.
3) 셋째, 사용자가 같은 브라우저에서 승인 링크를 열면 서버는 make_device_token(uid, did) 함수를 통해 dev_id에 대한 서명 토큰(devtk)을 생성하고, 이를 쿠키로 설정한다. 서버는 동시에 devices 컬렉션에 user_id, device_id, ua_hash, ip_hash를 기록하거나 갱신한다. 이때 User-Agent와 IP 정보는 원본을 저장하지 않고 해시 형태로 보관하여 프라이버시를 보호한다.
4) 넷째, 이후 동일 브라우저는 devtk 쿠키를 전송함으로써 신뢰된 디바이스로 인식되며, OTP(TOTP) 기반의 2단계 인증 과정을 수행한 후 최종 접근 권한을 얻는다.

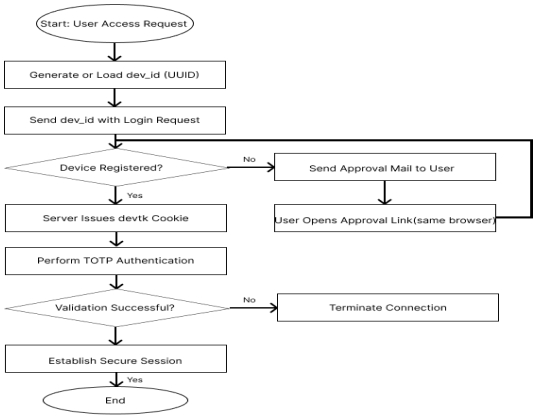

그림 1 시스템 시나리오 순서도

본 설계는 개인정보 비식별화를 전제로 한 브라우저 단위의 신뢰 모델을 구현한다는 점에서 프라이버시와 보안의 균형을 갖추었다. 브라우저 내부에 저장된 UUID와 서버 서명 토큰의 조합으로 개별 사용자의 접속 환경을 고유하게 식별할 수 있으며, 이를 통해 사용자는 불필요한 재인증 절차 없이 안정적인 접근이 가능하다.

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

서버는 ua_hash와 ip_hash의 변화를 통해 세션의 이상 여부를 감시할 수 있고, 관리자는 승인 및 철회 절차를 통해 기기 신뢰 상태를 동적으로 관리할 수 있다. 그러나 이러한 구조는 다음과 같은 한계가 존재한다. 브라우저의 로컬스토리지 초기화나 변경 시 dev_id가 삭제되어 재승인이 필요하며, 쿠키(devtk) 탈취나 재사용 공격에 대비한 철회·모니터링 메커니즘이 필수적이다. 또한 User-Agent나 IP 정보는 변동 가능한 요소이므로 이를 절대적 기준으로 신뢰하기 어렵다. 향후에는 행동 기반 이상탐지나 FIDO/WebAuthn과 같은 하드웨어 기반 인증을 병행하여 신뢰 수준을 높이는 방향의 보완이 필요하다.

표 1 실험 환경 구성 요약

구분	항목	내용
Hardware	CPU	AMD Ryzen 5 7500F (6-Core, 3.70 GHz)
	RAM	32GB
Software	OS	Windows 11 Pro (24H2, Build 26100.3476)
	Frontend	PyQt 5.15.10 (GUI Client)
	Backend	Flask 3.0.3 (Python 3.11)
	Database	MongoDB 7.0
	OTP Library	PyOTP 2.9.0
	Web Framework	Flask + Socket 통신
Runtime Environment	IDE	Visual Studio Code 1.93
	Network	Localhost
	Browser	Chrome 129.0

Ⅲ. 시스템 구현

본 시스템은 웹 클라이언트(브라우저)와 서버(백엔드)로 구성되며, 클라이언트에서 생성된 고유 식별자(dev_id)와 서버가 발급하는 서명 토큰(devtk), 그리고 TOTP 기반 사용자 인증 절차를 중심으로 동작한다. 서버는 Flask 프레임워크(Python 기반)로 구현되었으며, 클라이언트는 JavaScript 환경에서 실행된다. 클라이언트는 최초 접속 시 localStorage에 dev_id(UUID)를 생성·저장하고, 로그인 시 이를 서버로 전송한다. 서버는 dev_id를 확인하여 미등록 기기일 경우 승인 메일을 발송하며, 사용자가 메일의 승인 링크를 같은 브라우저에서 열면 devtk 쿠키가 발급된다. 이후 브라우저는 devtk를 통해 신뢰된 기기로 인식되며, 사용자는 TOTP 기반 2단계 인증을 수행하여 최종 접근 권한을 획득한다. 그림 2의 상단은 로그인 요청 시 서버 응답 메시지를, 하단은 승인 절차 이후 TOTP 코드를 입력하여 인증하는 과정을 보여준다. 서버는 Flask 백엔드에서 인증 요청을 처리하며, 클라이언트는 PyQt GUI를 통해 OTP 입력창을 표시한다. 서버는 승인 처리 시 devtk 쿠키를 생성하고, 동시에 devices 컬렉션에 user_id, device_id, ua_hash, ip_hash를 저장·갱신한다. User-Agent와 IP 정보는 원본을 저장하지 않고 해시 형태로만 관리하여 개인정보 노출을 방지하였다. 모든 요청에서 서버는 devtk의 서명과 만료 상태를 검증하고, 유효할 경우에만 OTP 인증 절차로 진입할 수 있도록 하였다. 쿠키에는 HttpOnly, Secure, SameSite=Strict 속성을 적용해 탈취 위험을 최소화하였다. devtk는 짧은 유효기간 내에서 정기적으로 재발급되며, 관리자는 웹 인터페이스를 통해 등록된 디바이스의 신뢰를 철회할 수 있다. ua_hash나 ip_hash의 급격한

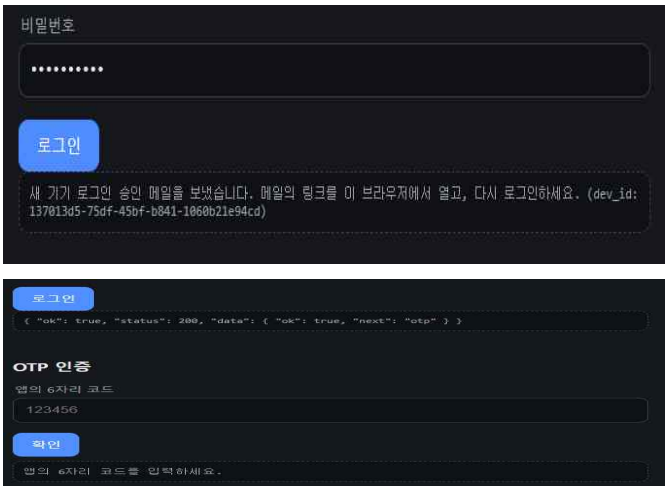

그림 2 기기 미인증 로그인(상)과 기기 인증 후 TOTP 로그인(하)

변동이 감지되면 서버는 추가 인증을 요구하도록 설계하였다. 브라우저 변경이나 로컬스토리지 초기화 시에는 재승인 절차를 통해 신뢰 관계를 복구한다. 본 시스템은 dev_id와 devtk 쿠키 조합을 통해 개인정보 수집을 최소화하면서 실질적인 제로 트러스트 기반의 디바이스 인증을 구현하였다. 다만, 브라우저 데이터 초기화나 쿠키 탈취 상황에서는 재승인 절차가 필요하고, User-Agent 및 IP 정보의 변조 가능성도 존재한다. 향후에는 토큰 철회·블랙리스트 관리, 승인 링크의 일회성 검증, 그리고 FIDO(WebAuthn) 기반 하드웨어 인증을 결합하여 보안성과 확장성을 강화할 예정이다.

Ⅳ. 결론

본 연구는 기존 식별자 기반 디바이스 인증의 스푸핑 취약성, 분실·도난 기기 문제, 확장성 부족, 지속 검증 부재 등의 한계를 개선하기 위해 제로 트러스트(Zero Trust) 보안 모델을 적용한 새로운 인증 체계를 설계·구현하였다. 제안된 시스템은 브라우저에서 생성된 고유 식별자(dev_id), 서버의 서명 토큰(devtk), 그리고 TOTP 기반 다중 인증을 결합하여 사용자와 디바이스를 동시에 검증한다. 또한 승인 메일 절차를 통해 미등록 기기의 접근을 제어하고, ua_hash 및 ip_hash를 활용해 세션 환경의 무결성을 보조적으로 검증하였다. 향후 연구에서는 관리자 승인 절차의 자동화, 토큰 블랙리스트 및 일회성 승인 링크 관리, 그리고 블록체인 기반 기기 등록 이력 관리와 인공지능 이상행위 탐지를 결합하여 시스템의 보안성과 효율성을 한층 강화할 예정이다.

ACKNOWLEDGMENT

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

참 고 문 헌

[1] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, “Zero Trust Architec ture,” NIST Special Publication 800-207, National Institute of Standards and Technology, 2020.

[2] P. Frustaci, P. Pace, G. Aloi, and G. Fortino, “Evaluating critical security issues of the IoT world: Present and future challenges,” IEEE Internet of Things Journal, vol. 5, no. 4, pp. 2483-2495, Aug. 2018.