부산대학교 정보컴퓨터공학부

마이크로 컨트롤러에서의 안전한 로봇 어플리케이션 수행을 위한 원격 증명 기술 개발

2025 전기 졸업과제 착수보고서

202055506 강승민
202055523 김의준
202055543 박재선

2025-5-15

지도 교수 권동현 (인)

목차

Ⅰ. 과제 배경 및 목표

1. 과제 배경

ROS 2는 로봇 소프트웨어의 분산화 및 표준화를 이끄는 핵심 프레임워크로, 자율주행차, 드론, 서비스 로봇 등 다양한 분야에 폭넓게 채택되고 있다. 최근에는 자원 제약이 있는 마이크로컨트롤러(MCU) 기반의 임베디드 시스템에서도 ROS 2 생태계를 확장할 수 있도록 micro-ROS가 개발되었으며 이를 통해 초소형 센서 및 액추에이터 노드까지 ROS 기반 통합 제어망에 포함되는 추세다.

그러나 이러한 시스템 확장은 보안 위협의 공격 표면 확대로 이어진다. 공격자는 변조된 펌웨어를 업로드하거나, 런타임 파라미터를 비정상적으로 변경하거나, 네트워크 세션 키를 탈취함으로써 시스템 동작을 오염시킬 수 있다[1].

특히 micro-ROS는 단일 바이너리 구조와 자원 제약으로 인해 프로세스 격리 및 런타임 무결성 검증이 어렵기 때문에 보안 취약점이 발생할 가능성이 높다. 이러한 취약성은 단일 노드에 그치지 않고, 해당 임베디드 장치뿐 아니라 이를 제어하거나 통신하는 상위 시스템 전체에 보안 위협을 확산시킬 수 있다.

이러한 문제를 해결하기 위한 핵심 기술로 Remote Attestation(RA, 원격 무결성 검증) 기법이 주목받고 있다. RA는 부팅 시점 또는 런타임 중 신뢰 앵커(TrustZone-M, SRAM-PUF 등)를 활용해 장치의 소프트웨어 무결성을 증명하고 그 결과를 기반으로 DDS 통신 세션의 생성 여부나 권한 부여를 결정할 수 있도록 한다.

이를 구현하기 위한 선행 연구로 기존의 DDS handshake 과정에 TPM 2.0 기반의 무결성 증명을 통합하여 검증 실패 시 통신을 차단하는 DDS Security+ 연구[2], 그리고 TEE(TrustZone·SGX) 내부에서 ROS 2 노드의 실행 코드 및 상태를 해시하여 보고하는 TEE-ROS RA 연구[3]가 존재한다.

본 과제는 이러한 구조적 접근을 통합하여 자원 제약 MCU 환경에서도 안전한 실행 환경을 보장할 수 있도록 펌웨어 무결성을 원격에서 증명하고 이를 기반으로 DDS 통신 권한을 통제하는 보안 기술을 개발하고자 한다.

2. 과제 목표

본 과제는 임베디드 시스템 환경에서 ROS 2 기반의 분산 로봇 및 마이크로 컨트롤러 노드의 소프트웨어 무결성을 원격지에서 검증할 수 있는 기술을 구현하는 것을 핵심 목표로 한다. 이를 위해 micro-ROS 통신 구조를 활용하고 하드웨어 신뢰 앵커(TEE)를 기반으로 하는 2단계 원격 증명 체계를 설계한다.

2.1. 부팅 시점에서의 무결성 검증

첫 번째 목표는 임베디드 디바이스가 시스템 부팅 직후 자체 펌웨어의 무결성을 검증 가능한 방식으로 증명하고 그 결과를 micro-ROS 클라이언트가 DDS Agent 에 전달함으로써 네트워크 통신 권한을 통제할 수 있는 구조를 구현하는 것이다. 이를 통해 무결성 검증을 통과하지 못한 노드는 ROS 2 DDS 네트워크에 참여하지 못하도록 차단할 수 있다.

이 구조에서는 STM32L552 마이크로컨트롤러의 TF-M(Trusted Firmware-M) 기능과 내장된 HUK(Hardware Unique Key)를 하드웨어 Root of Trust 로 활용한다. 시스템 부팅 과정에서 Boot ROM 이 부트로더, TF-M, Non-Secure Application 전체의 해시 값을 계산한 후 해당 값을 Secure World(TZ-M)로 전달한다. 이후 Secure World 는 이 값을 기반으로 remote attestation 토큰을 생성하고 HUK 에서 파생된 디바이스 서명 키로 서명한 뒤 DDS Agent 에게 전송한다.

DDS Agent 는 이 토큰 내의 해시 정보와 디바이스 ID, nonce 등을 검증하고 장치가 사전에 정의된 신뢰 가능한 소프트웨어 구성을 사용하고 있는지를 확인한다. 검증에 실패할 경우 해당 노드는 DDS 네트워크에 참여할 수 없으며 전체 시스템에 접근을 차단당한다.

이 방식은 기존의 인증서(PKI) 기반 인증 구조를 보완하여 "실제 실행 중인 코드의 무결성"을 하드웨어 기반으로 증명하는 조건부 인증 체계를 제공한다. 특히 부트 체인(부트로더, RTOS, 애플리케이션)의 어느 한 부분이라도 변조되었을 경우에도 Secure World 수준에서 검출 가능하며 DDS Agent 측에서 이를 기반으로 ROS 2 그래프 상의 신뢰된 노드만 참여하도록 통제할 수 있는 구조를 실현한다.

2.2. 시스템 동작 중의 무결성 검증

두 번째 목표는 임베디드 기기에서 실행 중인 애플리케이션 자체의 런타임 무결성을 주기적으로 검증함으로써, 코드가 정상적으로 실행되고 있는지 실시간으로 신뢰성을 확보하는 구조를 구현하는 것이다. 이때의 검증 대상은 애플리케이션 코드 그 자체와 그 실행 상태에 해당한다.

이를 위해 STM32L552 와 같은 TF-M 지원 MCU 에서는 Secure World 에 구축된 경량 보안 모듈이 주기적으로 함수 호출과 같은 실행흐름이나 주요 센서의 값과 같은 파라미터 변경내용 등의 런타임 정보를 수집한다. 수집된 정보는 실시간으로 해시 체인에 누적되며 일정 시간 간격(예: 100ms~500ms)으로 해시 결과를 포함한 Rolling Attestation Token 을 생성하고, 이를 micro-ROS 통신 경로를 통해 상위 Agent 에 전송한다.

상위 Agent 는 이 토큰을 검증하여 시스템이 의도한 제어 흐름과 파라미터 상태를 유지하고 있는지 혹은 비정상적인 행위(예: ROP 공격, 파라미터 변조, 지연 공격 등)가 발생했는지를 실시간으로 판단한다. 이상이 탐지되면 해당 노드의 ROS 2 통신을 차단하여 시스템 전체의 무결성을 확보하고 해당 노드는 재부팅, 펌웨어 재배포 등의 대응이 가능하다.

이 구조에서 가장 중요한 점은 이 검증이 애플리케이션이 정상적으로 동작하는 중간에도 정기적으로 반복 수행되어야 하며 MCU 자원과 실시간성 요구사항을 동시에 고려해야 한다는 점이다. 따라서 해시 연산 경량화, 이벤트 선택 최소화, 토큰 생성 시점 지연 처리, Secure World 와의 최소 교신 설계 등 최적화된 구현 방식이 검증 구조의 핵심이다.

Ⅱ. 대상 문제 및 요구조건 분석

1. 대상 문제 분석

1.1. 단일 노드의 변조로 인한 전체 시스템의 붕괴 가능성

micro-ROS는 MCU에서 실행되는 RTOS 기반 시스템으로 애플리케이션 코드와 RTOS 커널, micro-ROS 클라이언트 스택이 하나의 단일 펌웨어 바이너리로 통합되어 빌드 된다. 이 바이너리를 공격자가 임의로 수정하여 ROS를 이용한 통신을 시도하여 Agent나 다른 노드에 접근이 가능하면 Agent와 Agent에 있는 다른 노드까지 영향을 받아 시스템 전체에 치명적인 손상을 줄 수 있다.

1.2. 런타임 행위의 변조 가능성

펌웨어 해시가 동일하더라도 공격자는 Stack overflow나 함수 포인터 덮기를 통해 제어 흐름을 탈취하는 ROP 공격과 같은 공격이 가능하다. 뿐만 아니라 파라미터 조작과 같은 방법을 통해 안전계수, PID, 속도 제한 등 설정을 변경하여 추가적인 문제를 발생시킬 가능성이 존재한다.

1.3. 제한된 자원 내에서의 무결성 검증 시스템 구현 필요

micro-ROS가 적용되는 MCU 플랫폼은 대체로 256 KB 이하의 Flash와 수십 KB의 RAM을 가지며 MMU(Memory Management Unit)를 지원하지 않기 때문에 기존 리눅스 기반 ROS 2처럼 TEE 또는 컨테이너 기반 격리된 인증 메커니즘을 직접 적용하기 어렵다.

1.4. 중간자 공격의 가능성

XRCE-DDS는 경량화를 위해 1바이트 Session ID + 2바이트 Stream ID로 구성된다. 인증서 기반 TLS 통신을 사용하지 않기 때문에 공격자가 micro-ROS 클라이언트와 Agent 사이의 네트워크를 가로채 통신 내용을 복제, 조작 또는 재전송할 수 있다.

또한 XRCE-DDS의 단순한 Session/Stream 구조와 낮은 난수성으로 인해 세션 하이재킹이나 replay 공격이 쉽다는 문제도 존재한다.

2. 요구조건 분석

2.1. 단일 노드의 변조방지를 위한 펌웨어 무결성 검증

임베디드 디바이스는 단일 펌웨어 이미지로 모든 ROS 노드와 RTOS 커널이 통합되어 배포되기 때문에, 펌웨어 자체의 무결성을 검증하는 것이 전체 시스템 신뢰성의 출발점이다. 이를 위해 부팅 시점에 Secure Boot와 해시 기반 Remote Attestation을 활용하여 펌웨어 변조 여부를 외부에서 검증할 수 있는 구조가 필요하다.

2.2. 런타임에서의 실행 환경 무결성 검증

정적 해시만으로는 런타임 공격(예: ROP, 파라미터 조작)을 방지할 수 없다. 따라서 시스템이 동작 중일 때의 제어 흐름, 파라미터 상태, 타이밍 행태 등을 주기적으로 모니터링하고 이를 증명 가능한 형식으로 요약해 전송하는 런타임 무결성 검증 메커니즘이 요구된다.

2.3. 임베디드 환경에 적절한 도구 사용

micro-ROS가 적용되는 MCU는 자원이 제한적이며 외부 TPM 등 고가 장치를 장착하기 어렵다. 따라서 TF-M, SRAM-PUF와 같이 MCU 내부에 이미 존재하는 보안 기능을 활용하고 경량 토큰 형식(CBOR/COSE)과 적은 오버헤드의 해시/서명 알고리즘을 사용하는 방식이 적절하다.

2.4. 중간자 공격(MITM) 예방을 위한 암호화 적용

XRCE-DDS는 기본적으로 인증과 암호화 기능이 없기 때문에 Session ID 탈취 및 MITM(Man-in-the-Middle) 공격에 취약하다. 이를 방지하기 위해 원격 증명에 성공한 노드에만 통신 세션 키를 발급하고 해당 키를 암호화하여 통신에 사용하는 구조를 적용해야 한다. RA 기반 세션 바인딩을 통해 키 유출 위험도 최소화할 수 있다.

Ⅲ. 현실적 제약 사항 분석 결과 및 대책

1. 현실적 제약 사항 분석

1.1. 임베디드 기기의 자원 제약 (Flash/RAM/연산 능력 부족)

micro-ROS가 주로 실행되는 MCU는 일반적으로 256 KB 이하의 Flash, 수십 KB 수준의 RAM, 100~200 MHz급의 저전력 코어를 탑재하고 있으며 전력 소비 또한 엄격히 제한된다. 이러한 환경에서는 복잡도가 높은 무결성 검증 로직이나 대용량 토큰의 반복 전송이 시스템의 실시간성을 저해할 가능성이 크다. 따라서 보안 기능 구현 시 경량 알고리즘 및 최소한의 연산 오버헤드를 고려한 설계가 필수적이다.

1.2. 임베디드 기기 자체의 취약한 보안성

임베디드 디바이스는 비용, 면적, 전력 소모의 제약으로 인해 TPM, MMU, Secure Storage와 같은 보안 하드웨어 기능을 갖추지 않은 경우가 많다. 이로 인해 디바이스 내부의 소프트웨어 및 통신 정보는 물리적 접근이나 펌웨어 변조에 취약하다. 특히 이러한 노드가 micro-ROS를 통해 ROS 2 네트워크에 연결될 경우 공격자는 펌웨어를 조작하여 무단 퍼블리셔로 위장하거나, 위조된 센서 데이터를 ROS 그래프에 삽입하는 스푸핑 공격을 수행할 수 있다. 이는 단일 노드에 그치지 않고 시스템 전체의 신뢰성에 악영향을 줄 수 있다.

2. 대책

2.1. 임베디드 기기의 자원 제약에 맞는 설계 구상

임베디드 MCU의 Flash, RAM, 연산 성능 등의 자원 제약을 고려하여 내장된 하드웨어 보안 기능을 적극 활용함으로써 연산 속도를 향상시키고 무결성 검증 과정의 오버헤드를 최소화하는 경량 보안 설계를 적용한다. 또한 검증 주기의 간격을 최적화하거나 이벤트 로그의 처리 방식을 간소화함으로써 실시간 제어 성능을 유지하면서도 보안 요구사항을 충족할 수 있도록 설계한다.

2.2. 임베디드 기기의 보안성 부족에 대응하는 설계 구상

외부 보안 칩 없이도 MCU 단독으로 신뢰 기반의 무결성 증명을 수행할 수 있도록 Trusted Firmware-M 기반의 Secure Boot 및 TF-M Attestation 구조를 채택한다. 이를 통해 MCU 내부를 Secure/Non-secure 영역으로 분리하고 Secure World에서 부팅 시점에 펌웨어, RTOS, ROS 구성 정보에 대한 해시 값을 생성하여 서명한 후 외부에 증명 토큰으로 제공하는 방식을 적용한다.

IV. 추진 체계 및 일정

개발 순서는 다음과 같은 과정으로 예상한다.

1. Cold-Boot 펌웨어 무결성 검사
2. Hot-Boot 펌웨어 무결성 검사
3. Runtime Control Flow (call/return 관련 루프/경로 화이트리스트 기반 검증) 무결성 검사

앞선 과제를 끝내고 시간이 충분하다면 데이터 섹션(캘리브레이션 값이나 상수, 센서 데이터 등) 무결성 검사 코드까지 작성

구분	5월			6월				7월				8월					9월		
	3	4	5	1	2	3	4	1	2	3	4	1	2	3	4	5	1	2	3
사전조사 및 주제 고도화																			
펌웨어 무결성 검증 임베디드 코드 작성																			
펌웨어 무결성 검증 Agent 코드 작성																			
런타임 환경 무결성 검증 임베디드 코드 작성																			
런타임 환경 무결성 검증 Agent 코드 작성																			
최종 확인 및 보고서 작성																			

V. 구성원 역할 분담

이름	세부 역할 분담
강승민	- Micro-ROS 활용 - Zephyr RTOS 활용 - Attestation token 생성 코드 작성 - 관련 선행연구 조사 - 공격방식 조사 및 보호기법 조사 - ROS2 기반 코드 micro-ROS로 포팅 - 보고서 작성
김의준	- Micro-ROS 활용 - Zephyr RTOS 활용 - 무결성 검증 서버 시스템 - 관련 선행연구 조사 - 공격방식 조사 및 보호기법 조사 - ROS2 기반 코드 micro-ROS로 포팅 - 보고서 작성
박재선	- Micro-ROS 활용 - Zephyr RTOS 활용 - TZ-M을 사용한 무결성 확보 코드 작성 - 관련 선행연구 조사 - 공격방식 조사 및 보호기법 조사 - ROS2 기반 코드 micro-ROS로 포팅 - 보고서 작성

VI. 참고 문헌

[1] Jaewoong Heo, Lee Yeji, & Hyo Jin Jo (2023). ROS2 공격 기술 동향 분석. REVIEW OF KIISC, 33(4), 57-63.

[2] WAGNER, Paul Georg; BIRNSTILL, Pascal; BEYERER, Jürgen. DDS Security+: Enhancing the Data Distribution Service With TPM-based Remote Attestation. In: Proceedings of the 19th International Conference on Availability, Reliability and Security. 2024. p. 1-11.

[3] WANG, Qian; LEE, Brian; QIAO, Yuansong. Support Remote Attestation for Decentralized Robot Operating System (ROS) using Trusted Execution Environment. In: 2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). IEEE, 2024. p. 693-695.