

## Security Architecture Design and Performance Analysis in Heterogeneous ROS2-microROS Communication Environments

Seung-Hwan Ju\*, Hee-Suk Seo\*\*

\*Researcher, Technology Planning Team, AUTOCRYPT Co., Ltd., Seoul, Korea

\*\*Professor, Dept. of Computer Science & Engineering, KoreaTech, Cheonan, Korea

### [Abstract]

As autonomous driving systems adopt more distributed architectures, communication between ROS2 nodes and resource-constrained microROS devices is becoming increasingly important. However, microROS uses XRCE-DDS, which does not natively provide DDS-Security, leaving a security gap in heterogeneous ROS2-microROS communication. This paper designs a security architecture that applies TLS and VPN-based protection to address this limitation and evaluates its performance on a Raspberry Pi 5-based distributed testbed. The evaluation metrics included latency, throughput, CPU usage, and memory usage. The results show that TLS and VPN introduce moderate latency overhead while throughput and resource usage remain stable. These findings indicate that layered security can be a practical approach for protecting ROS2-microROS communication.

▶ **Key words:** ROS2, microROS, Heterogeneous Communication, Security Architecture, TLS, VPN

### [요 약]

자율주행 시스템이 분산형 구조로 발전함에 따라 ROS2 노드와 자원 제약이 있는 microROS 장치 간 통신의 중요성이 증가하고 있다. 그러나 microROS는 XRCE-DDS를 사용하며, 이는 DDS-Security를 기본적으로 제공하지 않기 때문에 ROS2-microROS 이중 통신 환경에서 보안 공백이 발생할 수 있다. 본 논문은 이러한 한계를 보완하기 위해 TLS 및 VPN 기반 보호 방식을 적용한 보안 아키텍처를 설계하고, 성능을 평가했다. 실험은 ROS2, SROS2, ROS2-microROS, TLS 기반 SROS2-microROS, VPN 기반 SROS2-microROS의 시나리오를 대상으로 수행하였으며, 지연시간, 처리량, CPU 사용률, 메모리 사용률을 비교하였다. TLS와 VPN은 일정 수준의 지연시간 오버헤드를 발생시키지만 처리량과 자원 사용률은 안정적으로 유지되는 것으로 나타났다. 이를 통해 계층형 보안 방식이 ROS2-microROS 통신 보호를 위한 실용적인 접근 방식이 될 수 있음을 확인했다.

▶ **주제어:** ROS2, microROS, 이중 통신, 보안 아키텍처, TLS, VPN

• First Author: Seung-Hwan Ju, Corresponding Author: Hee-Suk Seo

\*Seung-Hwan Ju (jooseunghwan@gmail.com), Technology Planning Team, AUTOCRYPT Co., Ltd.

\*\*Hee-Suk Seo (histone@koreatech.ac.kr), Dept. of Computer Science & Engineering, KoreaTech

• Received: 2026. 06. 25, Revised: 2026. 07. 27, Accepted: 2026. 08. 19.

## I. Introduction

자율주행 및 로봇 시스템은 고성능 연산 장치와 자원 제약이 있는 임베디드 장치가 함께 동작하는 분산형 구조로 발전하고 있다. 인지, 판단, 경로 계획과 같은 기능은 고성능 컴퓨팅 환경에서 수행되는 반면, 센서 데이터 수집이나 액추에이터 제어와 같은 기능은 마이크로컨트롤러 기반 장치에서 수행되는 경우가 많다. 이러한 구조에서는 ROS2 기반 노드와 microROS 기반 장치 간의 안정적인 이종 통신이 중요한 요구사항이 된다[1][2].

자율주행 및 로봇 시스템에서 ROS 기반 모듈 간 통신은 단순한 데이터 교환 수단이 아니라, 시스템 동작을 결정하는 핵심 경로이다. 센서 노드가 생성한 객체 인식 결과, 위치 정보, 상태 정보는 판단 및 제어 노드로 전달되며, 제어 노드는 이를 기반으로 주행 명령이나 액추에이터 제어 명령을 생성한다. 따라서 모듈 간 통신 데이터가 위변조되거나 비인가 노드가 통신에 참여할 경우, 잘못된 환경 인식, 비정상적인 제어 명령, 시스템 오동작으로 이어질 수 있다. 특히 분산형 구조에서는 여러 노드가 동일한 네트워크를 공유하므로, 노드 인증, 메시지 무결성, 통신 기밀성을 확보하는 것이 중요하다.

ROS2는 DDS(Data Distribution Service)를 기반으로 publish/subscribe 통신 구조를 제공하며, DDS-Security를 통해 인증, 접근제어, 암호화 기능을 적용할 수 있다. [3] 그러나 microROS는 자원 제약 환경을 고려하여 XRCE-DDS 기반의 Client-Agent 구조를 사용하므로, ROS2에서 제공하는 DDS-Security 기능을 동일한 방식으로 적용하기 어렵다. [2][4] 이로 인해 ROS2-microROS 이종 통신 환경에서는 통신 구간에 따라 보안 적용 수준이 달라질 수 있으며, microROS 연동 구간에는 별도의 보호 방식이 필요하다. [5]

이러한 한계를 보완하기 위해 TLS 또는 VPN 기반의 보안 방식을 적용할 수 있다. TLS는 전송 계층에서 특정 통신 채널을 보호할 수 있으며, VPN은 네트워크 계층에서 장치 간 통신 경로 전체를 보호할 수 있다. 다만 보안 기능을 추가하면 암호화, 세션 관리, 터널링 등으로 인해 지연시간이나 시스템 자원 사용량이 증가할 수 있다. 따라서 ROS2-microROS 환경에서 이러한 보안 방식이 실제로 적용 가능한지 확인하기 위해서는 성능 영향에 대한 정량적 분석이 필요하다.

본 논문은 ROS2-microROS 이종 통신 환경에서 적용 가능한 보안 아키텍처를 설계하고, TLS 및 VPN 기반 보호 방식의 성능 영향을 분석한다. 이를 위해 표준 ROS2

통신, SROS2 기반 보안 통신, ROS2-microROS 이종 통신, TLS 기반 SROS2-microROS 통신, VPN 기반 SROS2-microROS 통신의 5개 시나리오를 구성한다. 각 시나리오는 동일한 라즈베리파이 5 기반 분산 환경에서 수행하였으며, 지연시간, 처리량, CPU 사용률, 메모리 사용률을 기준으로 비교한다.

기존 연구에서는 ROS2의 DDS-Security 적용 구조, SROS2 기반 보안 도구, ROS2 보안 기능의 성능 영향, 그리고 자원 제약 노드와 ROS2 간 보안 통신 방식이 각각 검토되어 왔다. 그러나 대부분의 연구는 ROS2 단독 환경의 보안 기능 분석 또는 microROS 보안 통신 구현에 초점을 두고 있으며, ROS2, SROS2, ROS2-microROS, TLS 기반 보호, VPN 기반 보호를 동일한 테스트베드에서 비교한 연구는 제한적이다. 본 논문은 이러한 점을 고려하여 ROS2-microROS 이종 통신 환경에서 적용 가능한 보안 구조를 제안하고, 통신 구조와 보안 적용 방식에 따른 성능 차이를 동일 조건에서 비교한다.

본 논문의 구성은 다음과 같다. II. Preliminaries에서는 ROS2, microROS, DDS-Security, SROS2, TLS 및 VPN과 관련된 배경기술을 설명한다. III. The Proposed Scheme에서는 ROS2-microROS 이종 통신 환경에서 적용 가능한 계층형 보안 아키텍처를 제안한다. IV. Experimental Design에서는 실험 환경, 통신 시나리오, 성능 평가 방법을 제시한다. V. Experimental Results and Analysis에서는 각 시나리오의 지연시간, 처리량, CPU 사용률, 메모리 사용률을 비교하고 결과를 분석한다. 마지막으로 VI. Conclusions에서는 연구 결과를 요약하고 향후 연구 방향을 제시한다.

## II. Preliminaries

본 장에서는 ROS2-microROS 이종 통신 환경에서 보안 아키텍처를 설계하기 위해 필요한 주요 기술 요소를 간략히 설명한다. 본 논문에서 중요한 배경은 ROS2와 microROS의 통신 구조 차이, ROS2 보안 기능의 적용 범위, 그리고 microROS 연동 구간을 보호하기 위한 TLS 및 VPN 방식이다.

### 1. ROS2 and microROS

ROS2는 DDS(Data Distribution Service)를 기반으로 하는 분산형 로봇 미들웨어이다. ROS2 노드는 publish/subscribe 방식으로 데이터를 교환하며, QoS 정

책을 통해 메시지 전달 특성을 제어할 수 있다. 이러한 구조는 다수의 노드가 상호 통신하는 로봇 및 자율주행 시스템에 적합하다. [3]

microROS는 ROS2의 기능을 마이크로컨트롤러와 같은 자원 제약 장치에서 사용할 수 있도록 경량화된 프레임워크이다. microROS는 일반 ROS2 노드처럼 DDS 네트워크에 직접 참여하기보다, XRCE-DDS 기반의 Client-Agent 구조를 통해 ROS2 네트워크와 연동된다. [2][4] 이 구조는 경량 장치를 ROS2 시스템에 연결할 수 있게 하지만, ROS2와 동일한 보안 기능을 그대로 적용하기 어렵다는 한계를 가진다.

## 2. Security Limitations of ROS2 and microROS

ROS2는 DDS-Security를 기반으로 인증, 접근제어, 암호화 기능을 제공할 수 있다. SROS2는 이러한 DDS-Security 기능을 ROS2 환경에서 사용할 수 있도록 지원하는 도구이며, 노드 간 통신을 보호하기 위한 보안 정책을 구성할 수 있다. [6]

그러나 microROS는 XRCE-DDS를 사용하므로 DDS-Security 기반의 보안 기능을 동일한 방식으로 적용하기 어렵다. 특히 microROS Client와 microROS Agent 사이의 통신 구간은 ROS2 노드 간 통신과 구조가 다르기 때문에, ROS2에서 제공하는 보안 정책이 그대로 확장되지 않을 수 있다. 따라서 ROS2-microROS 이종 통신 환경에서는 microROS 연동 구간을 보호하기 위한 별도의 보안 설계가 필요하다.

## 3. TLS- and VPN-Based Protection Methods

TLS는 전송 계층에서 통신 채널을 보호하는 방식이다. TLS를 적용하면 통신 주체 인증, 데이터 암호화, 무결성 검증을 통해 특정 통신 경로를 보호할 수 있다. 따라서 microROS 연동 구간에 TLS를 적용하면 DDS-Security가 직접 적용되지 않는 구간의 보안성을 보완할 수 있다.

VPN은 네트워크 계층에서 장치 간 통신 경로 전체를 암호화된 터널로 보호하는 방식이다. VPN을 사용하면 ROS2 노드와 microROS 연동 장치가 동일한 보호 네트워크 안에서 통신할 수 있으며, 애플리케이션 구조 변경을 최소화할 수 있다. 본 논문에서는 TLS와 VPN을 microROS 보안 한계를 보완하기 위한 2개 대안으로 설정하고, 각각의 성능 영향을 비교한다.

## 4. Related Work and Research Positioning

ROS2 보안과 관련된 기존 연구는 주로 DDS-Security와 SROS2를 중심으로 이루어졌다. ROS2는 DDS를 통신 기반으로 사용하므로 DDS-Security를 통해 인증, 접근제어, 암호화 기능을 적용할 수 있으며, SROS2는 이러한 보안 기능을 ROS2 환경에서 사용할 수 있도록 지원한다. 기존 연구에서는 SROS2를 이용한 보안 정책 구성과 ROS2 그래프 보호 방법이 제안하며, ROS2 보안 기능의 사용성과 적용 가능성을 검토한다.

ROS2 보안의 성능 영향 측면에서, 보안 기능은 통신 기밀성과 무결성을 향상시킬 수 있지만 암호화와 인증 과정에서 지연시간 및 처리량에 영향을 줄 수 있다. 따라서 로봇 및 자율주행 시스템에서는 보안 기능의 적용 여부뿐 아니라, 보안 기능이 실시간성 및 통신 성능에 미치는 영향을 함께 고려해야 한다.

microROS와 관련된 연구는 자원 제약 장치에서 ROS2 기능을 활용하기 위한 실행 구조, 실시간성, 경량 통신에 초점을 두고 있다. microROS는 XRCE-DDS 기반의 Client-Agent 구조를 통해 ROS2 네트워크와 연동되지만, ROS2의 DDS-Security 기능이 microROS 구간까지 동일하게 적용되기는 어렵다.

본 논문은 이러한 기존 연구와 달리 ROS2-microROS 이종 통신 환경에서 보안 적용 방식별 성능 차이를 동일한 테스트베드에서 비교한다. 구체적으로 표준 ROS2 통신, SROS2 기반 보안 통신, ROS2-microROS 이종 통신, TLS 기반 SROS2-microROS 통신, VPN 기반 SROS2-microROS 통신을 함께 구성하고, 지연시간, 처리량, CPU 사용률, 메모리 사용률을 비교한다. 이를 통해 microROS의 보안 한계를 보완하기 위한 TLS 및 VPN 기반 보호 방식의 적용 가능성을 실험적으로 확인한다.

본 논문의 구성은 다음과 같다. II. Preliminaries에서는 ROS2, microROS, DDS-Security, SROS2, TLS 및 VPN과 관련된 배경기술을 설명한다. III. The Proposed Scheme에서는 ROS2-microROS 이종 통신 환경에서 적용 가능한 계층형 보안 아키텍처를 제안한다. IV. Experimental Design에서는 실험 환경, 통신 시나리오, 성능 평가 방법을 제시한다. V. Experimental Results and Analysis에서는 각 시나리오의 지연시간, 처리량, CPU 사용률, 메모리 사용률을 비교하고 결과를 분석한다. 마지막으로 VI. Conclusions에서는 연구 결과를 요약하고 향후 연구 방향을 제시한다.

### III. The Proposed Scheme

본 장에서는 ROS2-microROS 이종 통신 환경에서 적용 가능한 보안 아키텍처를 제안한다. 제안 아키텍처는 ROS2 구간에는 SROS2를 적용하고, microROS 연동 구간에는 TLS 또는 VPN 기반 보호 방식을 추가하는 계층형 구조를 가진다.

#### 1. Architecture Design

ROS2-microROS 이종 통신 환경에서는 ROS2 노드와 microROS 장치가 서로 다른 통신 구조를 기반으로 동작한다. ROS2는 DDS 기반 통신을 사용하므로 DDS-Security를 적용할 수 있지만, microROS는 XRCE-DDS 기반의 Client-Agent 구조를 사용하므로 동일한 보안 기능을 그대로 적용하기 어렵다. 따라서 이종 통신 환경에서는 ROS2 내부 구간과 microROS 연동 구간을 구분하여 보안 기능을 설계할 필요가 있다.

본 논문에서 제안하는 아키텍처는 기존 ROS2 및 microROS 통신 구조의 변경을 최소화하면서, 보안 기능이 부족한 구간을 별도의 보호 계층으로 보완하는 것을 기본 방향으로 한다. ROS2 노드 간 통신에는 SROS2를 적용하고, microROS Client와 Agent 사이 또는 관련 네트워크 경로에는 TLS 또는 VPN을 적용한다. 이를 통해 microROS의 DDS-Security 미지원 문제를 완화하고, 이종 통신 구간에서 메시지의 기밀성과 무결성을 보호하고자 한다.

#### 2. Proposed Architecture Structure

제안 아키텍처는 ROS2 보안 구간, microROS 연동 구간, 추가 보호 계층으로 구성된다. ROS2 보안 구간에서는 SROS2를 통해 DDS-Security 기반의 인증, 접근제어, 암호화 기능을 적용한다. 이 구간은 ROS2 노드 간 통신을 대상으로 하며, 기존 ROS2 보안 체계를 활용할 수 있다.

microROS 연동 구간은 microROS Client와 microROS Agent 사이의 통신을 포함한다. 이 구간은 XRCE-DDS 기반으로 동작하므로, ROS2 DDS-Security 기능이 직접 적용되기 어렵다. 따라서 제안 아키텍처에서는 이 구간을 보안 적용 경계로 보고, TLS 또는 VPN 기반의 추가 보호 계층을 배치한다.

TLS 기반 구조에서는 특정 통신 채널을 전송 계층에서 보호한다. 반면 VPN 기반 구조에서는 ROS2 노드와 microROS 연동 장치가 포함된 네트워크 경로 전체를 암호화된 터널로 보호한다. 두 방식은 적용 계층과 보호 범

위가 다르지만, 모두 microROS 연동 구간의 보안 한계를 보완하기 위한 대안으로 사용된다.

Fig 1은 제안 아키텍처의 개념적 구조를 나타낸다. ROS2 노드 간 통신은 SROS2 기반으로 보호되며, microROS와 연동되는 통신 구간은 TLS 또는 VPN을 통해 추가로 보호된다. 이 구조에서 microROS Agent는 ROS2 DDS 영역과 microROS XRCE-DDS 영역이 만나는 지점이며, 보안 적용 경계를 설정하는 핵심 구성요소로 볼 수 있다.

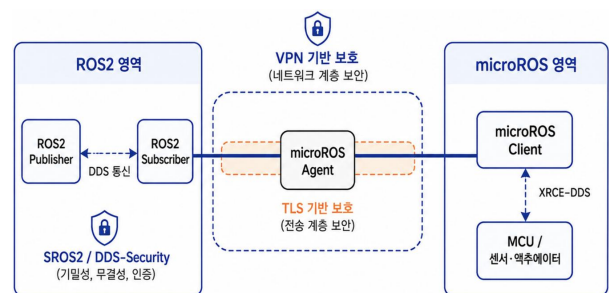


Fig. 1. Proposed Security Architecture for Heterogeneous ROS2-microROS Communication Environments

#### 3. TLS and VPN-Based Security Methods

TLS 기반 보안 방식은 microROS 연동 구간의 통신 채널을 보호하는 데 적합하다. TLS는 통신 주체 인증, 데이터 암호화, 무결성 검증을 제공하므로, XRCE-DDS 자체에서 DDS-Security를 제공하지 않더라도 전송 구간의 보안성을 보완할 수 있다. 이 방식은 특정 통신 경로를 명확하게 보호할 수 있다는 장점이 있으나, 인증서 검증과 암호화 과정에서 추가적인 지연시간이 발생할 수 있다.

VPN 기반 보안 방식은 장치 간 네트워크 경로 전체를 보호하는 데 적합하다. VPN을 적용하면 ROS2 노드와 microROS 연동 장치가 암호화된 터널 내부에서 통신하게 되므로, 애플리케이션 구조를 크게 변경하지 않고 네트워크 수준의 보호를 적용할 수 있다. 다만 터널링과 패킷 처리 과정으로 인해 통신 지연시간이 증가할 수 있다.

본 논문에서는 TLS와 VPN을 서로 다른 계층에서 적용 가능한 보안 대안으로 설정한다. TLS는 통신 채널 중심의 보호 방식이고, VPN은 네트워크 경로 중심의 보호 방식이다. 따라서 두 방식의 성능을 동일한 실험 환경에서 비교함으로써, ROS2-microROS 이종 통신 환경에서 보안성과 성능을 함께 고려한 적용 가능성을 확인한다.

본 논문은 이러한 보안 적용 경계를 보완하기 위해 microROS 연동 구간에 추가적인 보호 계층을 적용하는 방식을 고려한다. 구체적으로는 전송 계층에서 TLS를 적

용하는 방식과, 네트워크 계층에서 VPN을 적용하는 방식을 비교 대상으로 설정한다. 두 방식은 microROS의 DDS-Security 미지원 문제를 직접 수정하는 방식은 아니지만, 통신 채널 또는 네트워크 경로를 보호함으로써 ROS2-microROS 이종 통신 환경에서 발생할 수 있는 보안 공백을 완화할 수 있다.

## IV. Experimental design

본 장에서는 제안 보안 아키텍처의 성능 영향을 분석하기 위한 실험 설계를 설명한다. 실험은 동일한 하드웨어와 네트워크 환경에서 통신 구조와 보안 적용 방식만 변경하여 수행하였다. 이를 통해 ROS2-microROS 이종 통신 구조와 TLS 및 VPN 기반 보호 방식이 지연시간, 처리량, CPU 사용률, 메모리 사용률에 미치는 영향을 비교한다.

### 1. Experimental Environment

실험 환경은 2대의 라즈베리파이 5를 동일한 무선 네트워크에 연결한 분산 구조로 구성하였다. 1대는 Publisher 역할을 수행하고, 다른 1대는 Subscriber 역할을 수행하도록 설정하였다. 네트워크는 Wi-Fi 6 기반 공유기를 사용하였으며, 운영체제와 ROS2 배포판은 모든 시나리오에서 동일하게 유지하였다.

Table 1. Experimental Environment

| Category           | Description              |
|--------------------|--------------------------|
| Computing Device   | Raspberry Pi 5 8GB 2ea   |
| Network Device     | TP-Link Archer AX17      |
| Network Mode       | Wi-Fi 6 AX1500           |
| Operating System   | Ubuntu 24.04.2 LTS ARM64 |
| ROS2 Distribution  | ROS2 Jazzy               |
| DDS Implementation | Fast DDS                 |
| TLS Implementation | OpenSSL                  |
| VPN Implementation | OpenVPN                  |
| QoS Policy         | BEST_EFFORT              |

### 2. Experimental Scenarios

실험은 5개 통신 시나리오로 구성하였다. 표준 ROS2 통신은 기준 성능을 확인하기 위한 시나리오이며, SROS2 통신은 ROS2 보안 기능 적용에 따른 성능 변화를 확인하기 위한 시나리오이다. ROS2-microROS 통신은 이종 통신 구조 자체의 영향을 확인하기 위한 시나리오이고, TLS 및 VPN 기반 시나리오는 제안 아키텍처의 보호 방식에 따른 성능 영향을 분석하기 위한 것이다.

Table 2. Communication Scenarios

| No.       | Communication Structure | Security Mechanism |
|-----------|-------------------------|--------------------|
| Scenario1 | ROS2 → ROS2             | None               |
| Scenario2 | SROS2 → SROS2           | DDS-Security       |
| Scenario3 | ROS2 → microROS         | None               |
| Scenario4 | SROS2 → microROS        | TLS                |
| Scenario5 | SROS2 → microROS        | VPN                |

이 시나리오는 각각 ROS2 기본 통신 성능 측정, ROS2 보안 통신 오버헤드 측정, 이종 통신 구조의 성능 측정, 전송 계층 보호 방식의 성능 측정, 네트워크 계층 보호 방식의 성능 측정의 목적으로 수행한다.

이를 통해 지연시간, 처리량, CPU 사용률, 메모리 사용률을 평가지표로 활용한다.

Table 3. Experimental Parameters

| Category              | Set Values            |
|-----------------------|-----------------------|
| Message Size (bytes)  | 128, 1,280, 12,800    |
| Message Frequency     | 1 Hz, 10 Hz           |
| Number of Repetitions | 10 for each condition |
| Stabilization Time    | 5 seconds             |

지연시간은 Publisher가 메시지를 전송한 시점부터 Subscriber가 메시지를 수신한 시점까지의 단방향 시간을 기준으로 측정하였다. 메시지에는 송신 시점의 타임스탬프를 포함하였으며, Subscriber 측에서 수신 시점과의 차이를 계산하여 밀리초 단위로 변환하였다.

처리량은 네트워크 인터페이스의 송수신 데이터양을 기준으로 Mbps 단위로 산출하였다. CPU 사용률과 메모리 사용률은 실험 중 시스템 모니터링 값을 주기적으로 수집하여 평균값을 계산하였다. 이러한 방식으로 각 시나리오별 통신 성능과 시스템 자원 사용량을 비교하였다.

### 3. Performance Evaluation Method

성능 평가는 메시지 크기와 메시지 발행 주기를 변경하면서 수행하였다. 메시지 크기는 128 bytes, 1280 bytes, 12800 bytes로 설정하였고, 메시지 발행 주기는 1 Hz와 10 Hz로 설정하였다. 각 조건은 10회 반복 측정하였으며, 측정 시작 전 5초의 안정화 시간을 두었다.

## V. Experimental Results and Analysis

본 장에서는 5개 통신 시나리오의 성능 측정 결과를 분석한다. 결과는 지연시간, 처리량, CPU 사용률, 메모리 사용

를 기준으로 정리하였다. 특히 보안 기능 적용에 따른 영향은 지연시간에서 가장 뚜렷하게 나타났으며, 처리량과 시스템 자원 사용률은 시나리오 간 큰 차이를 보이지 않았다.

## 1. Latency Analysis

Table 4는 시나리오별 평균 지연시간을 나타낸다. 표준 ROS2 통신의 평균 지연시간은 7.89 ms로 가장 낮게 측정되었다. ROS2-microROS 이중 통신은 8.25 ms로 나타났으며, 표준 ROS2 통신 대비 약 4.6% 증가하였다. 이는 microROS Agent를 통한 XRCE-DDS 연동 구조가 추가되더라도, 기본적인 이중 통신 자체의 지연시간 증가는 제한적임을 보여준다.

Table 4. Latency by Communication Scenario

| Scenario              | Average Latency [ms] | Increase over ROS2 |
|-----------------------|----------------------|--------------------|
| ROS2 → ROS2           | 7.89                 | -                  |
| ROS2 → microROS       | 8.25                 | 4.6%               |
| SROS2 → SROS2         | 10.24                | 29.8%              |
| SROS2 → microROS(TLS) | 10.37                | 31.5%              |
| SROS2 → microROS(VPN) | 10.94                | 38.7%              |

보안 기능을 적용한 시나리오에서는 지연시간 증가가 확인되었다. SROS2 통신은 평균 10.24 ms로, 표준 ROS2 통신보다 약 29.8% 증가하였다. TLS 기반 SROS2-microROS 통신은 10.37 ms로 SROS2 통신과 유사한 수준을 보였으며, VPN 기반 SROS2-microROS 통신은 10.94 ms로 가장 높은 지연시간을 나타냈다.

이 결과는 보안 기능의 성능 영향이 주로 지연시간 측면에서 발생함을 의미한다. 특히 VPN 방식은 네트워크 경로 전체를 보호할 수 있다는 장점이 있지만, 터널링과 패킷 처리 과정으로 인해 TLS 방식보다 상대적으로 높은 지연시간을 보였다.

Table 5는 메시지 크기와 발행 주기에 따른 평균 지연시간을 나타낸다. 전체적으로 메시지 크기가 증가할수록 지연시간도 증가하는 경향을 보였다. 또한 일부 조건에서는

1 Hz의 평균 지연시간이 10 Hz보다 높게 나타났다. 본 실험은 각 조건에서 제한된 메시지 수를 기준으로 평균 지연시간을 산출하였기 때문에, 낮은 발행 주기 조건에서는 개별 측정값의 변동이 평균값에 더 크게 반영될 수 있다. 따라서 본 논문에서는 1 Hz와 10 Hz 간의 절대적 우열보다는 동일 조건 내에서 시나리오별 상대적 지연시간 차이를 중심으로 결과를 해석한다.

## 2. Throughput and Resource Usage Analysis

처리량은 모든 시나리오에서 유사한 수준으로 나타났다. 평균 수신 처리량은 약 0.047-0.048 Mbps 범위에 있었으며, TLS 또는 VPN 적용 여부에 따른 뚜렷한 감소는 확인되지 않았다. 이는 본 실험 조건에서 보안 기능이 데이터 전송률보다는 지연시간에 더 큰 영향을 주었음을 의미한다.

CPU 사용률과 메모리 사용률도 시나리오 간 큰 차이를 보이지 않았다. CPU 사용률은 모든 시나리오에서 1% 미만으로 측정되었고, 메모리 사용률은 모든 시나리오에서 약 18% 수준으로 유사하게 나타났다.

다만 본 실험의 메모리 사용률은 Raspberry Pi 5 8GB 시스템의 전체 메모리를 기준으로 산출한 비율이므로, TLS 또는 VPN 적용으로 인해 통신 프로세스가 추가로 사용한 메모리의 절대량을 정밀하게 나타내지는 못한다.

Table 6. Throughput and Resource Usage by Communication Scenario

| Scenario  | Average Received Throughput [Mbps] | Average CPU Usage [%] | Average Memory Usage [%] |
|-----------|------------------------------------|-----------------------|--------------------------|
| Scenario1 | 0.047                              | 0.91                  | 18.02                    |
| Scenario2 | 0.048                              | 0.66                  | 17.97                    |
| Scenario3 | 0.047                              | 0.76                  | 17.99                    |
| Scenario4 | 0.047                              | 0.77                  | 17.99                    |
| Scenario5 | 0.047                              | 0.72                  | 18.03                    |

따라서 본 결과는 시나리오 간 메모리 사용률에 큰 변화가 관찰되지 않았다는 수준에서 해석하는 것이 적절하다.

Table 5. Average Latency by Message Size and Publishing Rate [ms]

| Scenario              | 128B / 1Hz | 128B / 10Hz | 1280B / 1Hz | 1280B / 10Hz | 12800B / 1Hz | 12800B / 10Hz |
|-----------------------|------------|-------------|-------------|--------------|--------------|---------------|
| ROS2 → ROS2           | 7.58       | 6.88        | 8.05        | 7.09         | 9.38         | 8.36          |
| ROS2 → microROS       | 7.88       | 7.27        | 8.38        | 7.57         | 9.70         | 8.70          |
| SROS2 → SROS2         | 11.59      | 7.92        | 11.54       | 8.18         | 12.91        | 9.31          |
| SROS2 → microROS(TLS) | 11.76      | 8.26        | 11.51       | 8.41         | 12.71        | 9.56          |
| SROS2 → microROS(VPN) | 12.18      | 8.52        | 12.41       | 8.79         | 13.84        | 9.89          |

### 3. Overall Analysis

실험 결과를 종합하면, ROS2-microROS 이중 통신 구조 자체는 표준 ROS2 통신과 비교하여 큰 성능 저하를 발생시키지 않았다. 평균 지연시간은 7.89 ms에서 8.25 ms로 증가하였으며, 증가율은 약 4.6% 수준이었다. 따라서 microROS Agent를 통한 이중 통신 구조는 본 실험 환경에서 실용적인 통신 성능을 제공하는 것으로 볼 수 있다.

반면 SROS2, TLS, VPN과 같은 보안 기능을 적용한 경우에는 지연시간 증가가 확인되었다. TLS 기반 방식은 SROS2 통신과 유사한 수준의 지연시간을 보였고, VPN 기반 방식은 가장 높은 지연시간을 나타냈다. 그러나 처리량, CPU 사용률, 메모리 사용률은 모든 시나리오에서 유사하게 유지되었다.

기존 연구들은 ROS2의 DDS-Security 적용 구조, SROS2 기반 보안 도구, ROS2 보안 기능의 성능 영향, 또는 microROS와 같은 자원 제약 노드의 보안 통신 방식에 각각 초점을 두었다. 반면 본 연구는 표준 ROS2 통신, SROS2 기반 보안 통신, ROS2-microROS 이중 통신, TLS 기반 보호 통신, VPN 기반 보호 통신을 동일한 테스트베드에서 비교하였다. 특히 기존 연구가 특정 보안 방식의 적용 가능성이나 개별 성능 특성에 집중한 것과 달리, 본 연구는 이중 통신 환경에서 전송 계층 보호 방식과 네트워크 계층 보호 방식을 함께 비교하고, 지연시간뿐 아니라 처리량, CPU 사용률, 메모리 사용률을 함께 분석하였다는 점에서 차별성을 가진다. 실험 결과 TLS 및 VPN 기반 보호 방식은 지연시간 증가를 유발하였으나, 처리량과 시스템 자원 사용률은 큰 차이를 보이지 않았다. 이는 제안 구조가 microROS의 보안 한계를 보완하면서도 제한된 테스트베드 환경에서 과도한 자원 부담을 발생시키지 않는다는 점을 보여준다.

따라서 ROS2-microROS 이중 통신 환경에서 보안 기능을 적용할 때는 보호 범위와 지연시간 요구사항을 함께 고려해야 한다. 특정 통신 채널 중심의 보호가 필요한 경우에는 TLS 방식이 적합할 수 있으며, 네트워크 경로 전체를 보호하고 애플리케이션 변경을 줄이고자 하는 경우에는 VPN 방식이 유용할 수 있다. 다만 지연시간이 중요한 시스템에서는 VPN 적용에 따른 추가 오버헤드를 사전에 검토할 필요가 있다.

### VI. Conclusions

본 논문은 ROS2-microROS 이중 통신 환경에서 발생할 수 있는 보안 적용 한계를 보완하기 위해 TLS 및 VPN 기반 보호 방식을 적용한 보안 아키텍처를 제안하고, 이에 따른 성능 영향을 실험적으로 분석하였다. ROS2는 DDS-Security를 통해 보안 기능을 적용할 수 있지만, microROS는 XRCE-DDS 기반의 Client-Agent 구조를 사용하므로 동일한 방식의 보안 기능을 직접 적용하기 어렵다. 이에 본 논문에서는 ROS2 구간에는 SROS2를 적용하고, microROS 연동 구간에는 TLS 또는 VPN 기반 보호 계층을 적용하는 구조를 제안하였다.

실험 결과, ROS2-microROS 이중 통신은 표준 ROS2 통신과 비교하여 지연시간 증가가 크지 않은 것으로 나타났다. 반면 SROS2, TLS, VPN 기반 보안 시나리오에서는 지연시간이 증가하였으며, 특히 VPN 기반 방식에서 가장 높은 지연시간이 측정되었다. 그러나 처리량, CPU 사용률, 메모리 사용률은 모든 시나리오에서 유사한 수준을 유지하였다. 이는 본 실험 환경에서 보안 기능 적용의 주요 영향이 시스템 자원 사용량보다는 지연시간 측면에서 나타났음을 의미한다.

TLS 기반 방식은 특정 통신 채널을 보호하는 데 적합하며, VPN 기반 방식은 네트워크 경로 전체를 보호하고 애플리케이션 구조 변경을 줄이는 데 장점이 있다. 따라서 ROS2-microROS 이중 통신 환경에서는 보호 범위와 지연시간 요구사항을 고려하여 TLS와 VPN을 선택적으로 적용할 필요가 있다.

다만 본 연구는 라즈베리파이 5 기반의 제한된 분산 테스트베드와 단일 무선 네트워크 환경에서 수행되었으므로, 실제 차량 또는 로봇 시스템의 모든 운용 조건을 반영하지는 못하였다. 특히 라즈베리파이 5 8GB는 일반적인 MCU 기반 microROS 장치보다 높은 연산 성능과 메모리 용량을 가지므로, 본 실험에서 측정된 CPU 및 메모리 사용률이 실제 자원 제약 장치에서의 암복호화 부하와 메모리 사용 특성을 직접적으로 대표한다고 보기는 어렵다. 실제 시스템에서는 다중 노드 간 동시 통신, 무선 간섭, 패킷 손실, 네트워크 혼잡, 다양한 QoS 정책, 실제 마이크로컨트롤러의 연산 성능과 메모리 제약이 함께 고려되어야 한다. 따라서 본 연구의 결과는 ROS2-microROS 이중 통신 환경에서 TLS 및 VPN 기반 보안 방식을 적용하기 위한 기초 성능 자료이자 제한된 테스트베드에서의 상대 비교 결과로 해석하는 것이 적절하다. 향후 연구에서는 실제 MCU 기반 microROS 장치, 차량 또는 로봇 네트워크 환

경, 다중 노드 조건, 실제 공격 시나리오를 포함한 보안성 검증을 수행하여 제안 아키텍처의 적용 가능성을 보다 구체적으로 평가할 필요가 있다.

## ACKNOWLEDGEMENT

This paper was supported by the Education and Research Promotion Program of KOREATECH in 2026.

## REFERENCES

- [1] J. Staschulat, I. Lütkebohle, and R. Lange, "The rclc Executor: Domain-specific deterministic scheduling mechanisms for ROS 2," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems Workshops, 2020.
- [2] J. Staschulat, R. Lange, and I. Lütkebohle, "Budget-based real-time Executor for micro-ROS," arXiv preprint arXiv:2105.05590, 2021.
- [3] ROS 2 Design, "ROS 2 DDS-Security Integration," [https://design.ros2.org/articles/ros2\\_dds\\_security.html](https://design.ros2.org/articles/ros2_dds_security.html)
- [4] EProsima, "Micro XRCE-DDS and micro-ROS included in ROS 2," 2021.
- [5] G. Spilere Nandi et al., "Secure integration of extremely resource-constrained nodes on distributed ROS2 applications," Open Research Europe, 2023.
- [6] V. Mayoral-Vilches et al., "SROS2: Usable Cyber Security Tools for ROS 2," 2022.
- [7] G. Deng, Y. Liu, S. Xu, T. Zhang, and Y. Xu, "On the (In)Security of Secure ROS2," Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2022.

## Authors



Seung-Hwan Ju received the B.S., M.S., and Ph.D. degrees in Computer Science and Engineering in 2009, 2011, and 2017, respectively at Koreatech. He has worked in the fields of cybersecurity research, security

testing and evaluation. His research interests include automotive cybersecurity, IoT security, cybersecurity testing and evaluation, public key infrastructure (PKI), and cybersecurity regulations and standards.



Hee-Suk Seo received the M.S. and Ph.D. degrees in Electrical, Electronic and Computer Engineering from Sungkyunkwan University, Korea, in 2002 and 2005, respectively. He joined the faculty of the School of Computer

Science and Engineering at Korea University of Technology and Education (KOREATECH), Cheonan, Korea, in 2005, where he is currently a Professor. His research interests include network security, cybersecurity modeling and simulation, IoT security, AI security, and security technologies for connected and distributed systems.