Check for updates

Journal of KIIT. Vol. 22, No. 2, pp. 93-100, Feb. 28, 2024. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2024.22.2.93

ROS2 DDS 미들웨어의 통신 오버헤드 실험 분석

이화성*[1], 이현우*[2], 허선동*[3], 박무성*[4], 유찬곤*[5]

Experimental Analysis of ROS2 DDS Middleware's Communication Overhead

Hwaseong Lee*[1], Hyunwoo Lee*[2], Seondong Heo*[3], Moosung Park*[4], and Changon Yoo*[5]

이 논문은 정부의 재원으로 국방과학연구소의 지원을 받아 수행된 연구임

요 약

ROS(Robot Operating System)는 로봇 소프트웨어 개발에 사용되는 일종의 프레임워크로서 무인화 시대를 맞이하여 다수의 로봇에서 현재 사용되고 있다. 2010년 ROS1.0이 발표 된 후 보안과 실시간성을 개선하기 위해서 DDS(Data Distribution Service) 미들웨어 기반의 ROS2가 발표되었다. ROS2는 DDS 기반으로 Publish-Subscribe(발행-구독) 메커니즘을 통해 통신하고 평문/암호통신도 가능하며 DDS 벤더간 통신도 가능하다. 이에 기존의 ROS2 성능 분석 논문이 발표되고 있고 주로 벤더간 통신 테스트, 평문/암호통신에 따른 지연시간/처리량, QoS 옵션별 오버헤드 등이 대부분이다. 본 논문에서는 ROS2의 암호통신이 그룹키 기반의 통신이 아니라 Pairwise 키 기반으로 수행된다는 점을 고려하여 Subscriber 수에 따른 통신시간을 실험하였다. 실험 결과 통신방법(평문/암호) 뿐 아니라 Subscriber 수에 따라 통신시간이 폭발적으로 증대됨을 확인하였다.

Abstract

ROS is a type of framework used in robot software development and is currently being used in many robots in the unmanned era. ROS2 based on DDS middleware was announced to improve security and real-time. ROS2 basically uses DDS, communicating through the Publish-Subscribe mechanism, and encrypted communication is possible, and communication between DDS vendors is also possible. Accordingly, ROS2 performance analysis papers have been published, and most of them are about communication tests between vendors, latency/throughput according to plain text/encrypted communication, and latency by QoS options. In this paper, we experimented with the communication time according to the number of subscribers by considering the characteristics of ROS2, which performs encrypted communication with a parise key, and confirmed that the communication time increases depending on the number of subscribers as well as plaintext/encrypted communication.

Keywords

ROS2, DDS, DDS-Security, communication overhead, subscriber

* 국방과학연구소 연구원(*[1] 교신저자)
- ORCID[1]: https://orcid.org/0009-0007-9302-4456
- ORCID[2]: https://orcid.org/0009-0004-4254-7298
- ORCID[3]: https://orcid.org/0000-0002-7050-7208
- ORCID[4]: https://orcid.org/0000-0003-0760-5072
- ORCID[5]: https://orcid.org/0000-0003-1878-2162

· Received: Nov. 09, 2023, Revised: Jan. 18, 2024, Accepted: Jan. 21, 2024
· Corresponding Author: Hwaseong Lee
Agenmcy for Defense Development, Korea
Tel.: +82-2-3400-2646, Email: mail.hslee@gmail.com

Ⅰ. 서 론

로봇 기반 시스템은 최근 산업 전반에 수요 증가로 급속도로 발전하고 있다. ROS(Robot Operating System)는 로봇 소프트웨어 개발에 사용되는 일종의 분산 객체 프레임워크로서 다수의 로봇에서 사용되고 있다[1][2]. ROS는 2010년 ROS1.0이 발표되었고 보안성과 실시간성 개선을 위해 DDS(Data Distribution Service)를 미들웨어로 채택한 ROS2가 발표되었다[3]. ROS2에서는 ROS1과 달리 보안 측면이 강화 되었다. 기본적으로 마스터 노드를 통해서 Publisher/Subscriber(발행자/구독자)를 관리하던 ROS1과 달리 DDS의 Publish/Subscribe(발행/구독) 기능을 통해서 통신한다. 그리고 키 생성 메커니즘이 추가되어서 운용자의 선택에 따라 평문/암호통신이 가능하다. 또한 DDS의 QoS 옵션 선택도 가능해졌다.

ROS2가 발표됨에 따라 성능분석 연구도 발표되었다. 2016년 Y. Maruyama et al. 연구진은 ROS1(원격, 로컬), ROS2(원격, 로컬), ROS1/ROS2 브릿지 등 다양한 ROS 환경에서 QoS 옵션을 적용하여 패킷 크기에 따른 지연시간(Latency)을 실험하였다[4]. 2018년 J. Kim et al. 연구진은 DDS-Security를 비활성화(Disable)와 활성화(Enable) 설정을 통해서 평문통신과 암호통신에 따르는 지연시간을 실험하였다[5]. 2022년 M. Aartsen et al. 연구진은 이종간 통신 테스트를 통해서 DDS간 호환성 성공 여부 테스트 및 지연시간을 측정하였다[6]. 위 연구는 ROS2의 보안 기능이 강화됨에 따라 보안기능 활성화, QoS 옵션별, 이종간 통신테스트를 진행한 연구이다. 하지만 ROS2의 암호키 방식인 Pairwise키를 고려하여 Subscriber 수가 통신시간에 미치는 영향은 분석되지 않았다.

UGV(Unmanned Ground Vehicle)는 ROS2를 사용하여 구동되는 대표적인 자율주행 무인기기이다. UGV에서는 주변 장애물 및 위치인식을 위해 센서(IMU와 LiDAR 등)로부터 다양한 아날로그 정보를 스캔하여 ROS2의 토픽(Topic) 형태로 주기적으로 발생한다. 이 토픽은 가공을 거쳐 목적에 따라 PointCloud 형태나 LaserScan 형태의 토픽으로 재발행된다. 즉, 자율주행을 위해서 외부 센싱 데이터는 ROS2 환경에서 지속적으로 재생산 되거나 복수개의 노드로 유통된다. 따라서 암호통신을 고려한 자율주행 알고리즘을 설계할 때, ROS2의 특성을 고려한 오버헤드 실험이 필요하다. 본 논문에서는 다수의 노드가 Pairwise 키로 암호통신을 한다는 ROS의 특징을 고려하여 ROS2 Foxy 환경에서 Subscriber 수에 따른 평문통신/암호통신의 네트워크 통신시간 실험한다.

논문의 구성은 다음과 같다. 2장에서는 ROS2/DDS/SROS2에 대한 배경 지식을 설명하고 3장에서는 실험 구성, 실험 지표 및 실험 방법을 설명한다. 4장에서는 실험 결과를 분석하고 5장에서 본 실험의 결론을 정리한다.

Ⅱ. 배경 지식

2.1 ROS2

ROS는 구글의 핵심 개발자였던 Scott Hassan이 설립한 Willow Garage에서 처음 만들어졌다. 이후 로봇 상용화 문제 해결에 필요한 보안, 안정성, 실시간성, 개발 용이성, 기술 지원, 하드웨어 연동 등의 이슈 사항을 보완하여 Open Robotics는 ROS2를 개발했다. ROS는 ROS1과 ROS2간에 아키텍쳐가 상이하여 통신 방식과 보안 측면에서 큰 차이점을 보인다. 먼저 ROS1은 노드 외 마스터가 존재하고 마스터는 노드를 관리하는 역할을 담당한다. 이와 달리 ROS2에서는 마스터가 존재하지 않고 DDS를 미들웨어로 사용한다. ROS2는 구조는 그림 1과 같고 다음과 같은 특징이 있다[7]-[9].

• 플랫폼: ROS2는 Linux만 지원하던 ROS1과 달리 Linux, Windows, macOS를 모두 지원한다. ROS2는 DDS 보안 사양을 기반으로 보안 메커니즘을 구축하였다. SROS2(Secure ROS2)는 ROS2 Userland Code에서 선언된 보안 구성을 DDS 보안 플러그인으로 전달한다. 대표적인 예로는 인증서 생성, 권한 파일 생성, 벤더 설정, DDS-Security 활성화/비활성화 설정이 있다[10]. 즉, ROS2는 SROS2를 활용하여 인증/접근제어/암호화 기능을 지원하고 ROS1에 비해 보안 기능을 강화하였다.

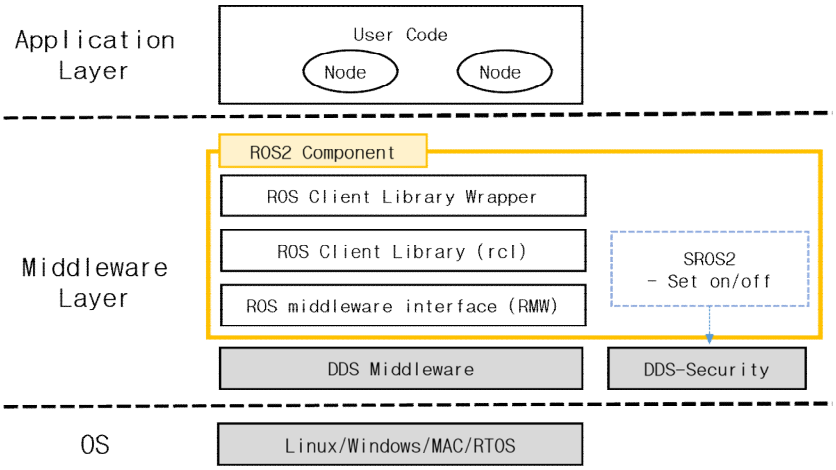

그림 1. ROS2 구조
Fig. 1. ROS2 structure

• 실시간성: ROS2는 실시간 작업을 지원한다. 이를 위해 선별된 하드웨어 사용, 실시간 OS 사용, DDS의 RTPS(Real Time Publish Subscribe Wire Protocol)와 같은 통신 프로토콜 사용, 실시간 적용 코드 사용을 전제로 하고 있다.

• 보안: ROS2는 ROS1에서 문제가 되었던 보안을 크게 개선하였다. ROS1은 보안보다는 로봇 개발에 사용되는 다양한 하드웨어 및 소프트웨어와 작동하는 유연성에 더 초점을 맞춰 개발되었기 때문에 보안 이슈는 개발 우선순위에 두지 못했다. 예를 들면, ROS1은 TCPROS 프로토콜 구조상 노드를 관리하는 ROS 마스터의 IP와 포트만 노출되면 모든 시스템을 다운시킬 수 있을 정도로 보안 입장에서 문제가 많았다[11]. ROS2는 DDS(Data Distribution Service)를 도입하여 DDS 보안 사양을 ROS2에 적용했다. ROS 커뮤니티에서는 SROS2를 개발하여 인증서 생성 등 보안 관련 툴킷을 배포하고 있다[10].

• 통신: ROS1에서는 자체 개발한 TCPROS와 같은 통신 라이브러리를 사용하고 있던 반면 ROS2는 RTPS를 지원하는 통신 미들웨어 DDS를 사용하고 있다. DDS는 노드 간의 자동 감지 기능을 지원하고 있기 때문에 ROS1에서 각 노드들의 정보를 관리하던 ROS 마스터가 없어도 여러 DDS 기반 호스트 간에 통신이 가능하다. 또한, 노드 간의 통신을 조정하는 QoS 옵션을 설정할 수 있다.

• QoS: ROS2에서는 QoS 옵션을 통해 원하는 기능을 선택적으로 사용할 수 있다. 이는 DDS의 QoS를 도입하였기 때문으로, Publisher와 Subscriber를 선언할 때, QoS를 매개변수형태로 지정하는 방식이 사용된다. 상황에 따라서 데이터의 신뢰성이 중요할 수도 있고 전송 속도가 중요할 수도 있기 때문에 QoS 옵션을 통해 선별적 구현이 가능하다.

- Reliability 옵션: 데이터 통신의 신뢰성 레벨(재전송 여부)을 결정하며 두 가지 속성을 설정할 수 있다.

(i) Reliable 속성: DataWriter History에 있는 모든 샘플들이 DataReader에 전달되는 것을 보장한다.

(ii) BEST Effort: 통신 시 손실된 데이터 샘플을 재전송 하지 않고, 전달된 데이터의 순서는 유지한다.

2.2 DDS

DDS는 OMG에서 정의한 Publish-Subscribe 방식의 실시간 데이터 분배 서비스 표준이고 Publish-Subscribe 방식은 ROS1의 Topic 통신과 매우 유사하다. OMG는 분산 객체 컴퓨팅 영역의 표준을 제정하는 국제 비영리 컨소시엄으로 영향력 있는 각종 세계 표준을 정의하고 있다. OMG DDS는 통신 프로토콜이 아닌 기능적 표준으로 TCP/UDP 같은 통신 프로토콜 위에서 정의되는 객체이다[12].

DDS는 산업 표준으로 다양한 벤더들에서 제공되고 있다. ROS2 설치 시 자동으로 Fast DDS가 사용되지만, 표 1과 같이 다양한 DDS 벤더들이 존재하고 있다. 다양한 DDS 벤더들에 맞추기 위해 ROS2는 RMW(ROS MiddleWare Interface)라는 패키지들을 제공한다.

표 1. DDS 벤더별 제품 및 RMW 정보
Table 1. DDS Vender's product and RMW implemetation

Product name	Licence	RMW implementation
eProsima fast DDS	Apache 2	rmw_fastrtps_cpp
Eclipse cyclone DDS	Eclipse public license v2.0	rmw_cyclonedds_cpp
RTI connext	Commercial, research	rmw_connext_cpp
GurumNetworks GurumDDS	Commercial	rmp_gurumdds_cpp

2.3 SROS2

SROS2는 ROS2 보안을 위한 도구들의 집합이다. CA 인증서를 자동 생성하고 x.509에 기반한 Public/Private 키쌍을 생성한다. Keypairs, Permissions Files(권한파일) 등이 담기는 Keystore 생성한다. DDS는 DDS-Security 기능적 보안 표준(인증,인가,암호,로깅,태깅)은 있으나 디폴트 설치 시 보안이 활성화 되어 있지 않다. 대신, 사용자는 이들을 플러그인 형태로 추가하여 사용하게 되며, 주로 Vendor 단에서 패치 형태로 제공한다. ROS2에서는 DDS-Security의 5가지 보안 기능(인증/접근제어/암호화/태깅/로깅) 중 아래 3가지 보안을 채택했다.

• 인증(Authentication): x.509 인증서 기반으로 같은 네트워크를 사용하고, 특정 도메인 내에 존재하는 Participant(참여자) 사이의 신원을 검사한다.

• 인가(Access control): Participant의 동작을 제한한다. 예를 들면, 특정 Participant가 특정 Topic 발행/구독을 가능/제한하는 형태이다.

• 암호(Cryptography): 암복호화, 사인, 해쉬, 전자서명과 같은 암호화 기법을 사용한다. 즉, Topic은 암복호화가 가능하고, 인증과 인가에서 암호 알고리즘을 사용한다.

III. 실험 환경

3.1 실험 구성

본 논문에서는 VMware로 ROS2 가상환경을 구성하여 실험한다. 1개의 IP에서 다수개의 노드를 생성하였고 실험 환경 구성에 사용된 머신의 세부 정보는 표 2와 같다.

표 2. 실험 환경
Table 2. Experimental environments

Type	Machine information
Host OS	Windows 10
Guest OS	Ubuntu 20.04 LTS
Memory	4G
CPU	Intel Core(TM) i5-8265u
ROS version	ROS2 Foxy
DDS	eProsima Fast DDS

다수의 노드간 신뢰할 수 있는 평문/암호화 통신을 위해서 아래의 단계를 실험 전에 진행한다.

• QoS 옵션 설정: ROS2는 DDS를 사용하기 때문에 QoS 옵션을 설정할 수 있다. 신뢰할 수 있는 Topic 유통을 위해서 그림 2와 같이 Publisher의 Reliability를 Reliable로 지정한다.

```
my_profile = QoSProfile(
    reliability=QoSReliabilityPolicy.RELIABLE,
    history=QoSHistoryPolicy.KEEP_LAST,
    depth=10,
    durability=QoSDurabilityPolicy.VOLATILE
)

self.string_publisher = self.create_publisher(
    String,
    'qos_test_topic',
    my_profile)
```

그림 2. QoS reliability 옵션 설정
Fig. 2. QoS option(reliability) setting

• 인증서 생성: ROS2는 PKI 기반 인증서를 생성한다. Keystore 폴더를 생성하여 아래와 같이 노드별 인증서를 Keystore 폴더내에 생성한다. 그림 3은 Talker(Publisher에 해당됨)와 Listener(Subscriber에 해당됨)만 존재할 경우의 예시이고 본 실험에서는 노드별로 인증서를 개별적으로 생성한다.

그림 3. 인증서 생성
Fig. 3. Certification creation

인증서를 생성하면 Keystore 내에 그림 4와 같이 CA의 인증서 뿐 아니라 Talker/Listener의 인증서가 생성되고 권한파일(Permission)도 생생됨을 확인할 수 있다.

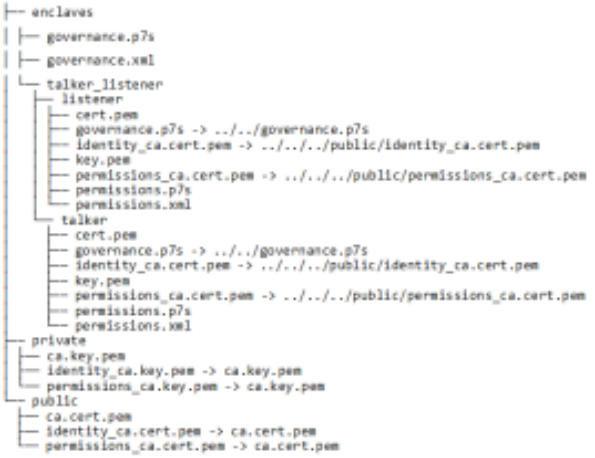

그림 4. Keystore 내 파일구조
Fig. 4. File structure in Keystore

ROS2는 DDS 미들웨어 상에서 동작하고 DDS는 DDS-security 표준을 따른다. 하지만 ROS2를 설치하면 기본 세팅은 DDS-Security 비활성화 상태이다. 따라서 평문/암호통신 실험을 위해서 DDS-Security를 선택적으로 활성화/비활성화 설정을 할 필요가 있다. 본 실험에서는 Fast DDS를 사용한다. 그림 5는 DDS-Security 설정과 DDS 벤더 설정 방법이다.

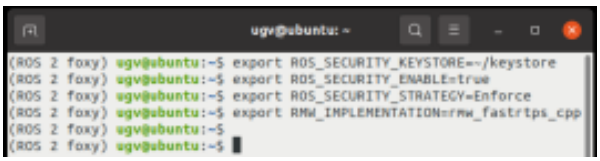

그림 5. DDS-Security 설정 (Enable 및 DDS 벤더)
Fig. 5. DDS-Security setting (Enable & DDS ventor)

3.2 실험 지표 및 시나리오

본 실험에서는 통신 효율성을 측정하기 위해서 통신시간을 계산한다. Publisher는 1K의 임의의 메시지를 0.2초 간격으로 생성하고 생성시간(T1)을 메시지에 추가한 후 즉시 Topic을 발행(전송)한다. Subscriber는 Topic을 구독(수신)하면 수신시간(T2)을 생성하여 통신시간(T2-T1)과 기타 정보(예: 수신바이트)를 계산하여 로그에 저장한다. Subscriber의 로그파일은 통신시간, Topic의 메시지 사이즈 등을 포함시켜서 Topic 메시지의 통신시간 뿐 아니라 메시지 무결성을 로그 레벨에서 확인이 가능하다. 그림 6은 터미널을 통해서 Publisher가 발행한 임의의 Topic 메시지의 전송 바이트(bw 옵션) 및 전송 주기(hz 옵션)를 확인한 화면이다. 아래 터미널 정보를 통해서 Publisher가 1K 크기의 Topic을 0.2초 간격으로 정상적으로 발행함을 확인할 수 있다. 본 실험에서는 0.2초 간격으로 총 100개의 Topic을 발행하였고 이를 총 10번씩 2차례 진행하여 통신시간의 평균값을 계산한다(표 3 참고).

표 3. 실험 결과 (통신시간)
Table 3. Experimental result (communication time)

Sub #	Communication time (s)			
	The first round		The second round	
	Plain	Cipher	Plain	Cipher
2	0.002385	0.002272	0.002982	0.002825
4	0.003616	0.004147	0.003927	0.004317
6	0.00536	0.007539	0.004909	0.006149
8	0.007677	0.010427	0.00694	0.009155
10	0.011636	0.027384	0.012081	0.026138

사용된 노드 수와 인증서 발행은 다음과 같다. 총 8개의 노드가 1K 크기의 Topic을 하나씩 발행(총 8개 Publisher)하고 최대 10개의 노드가 모든 Topic을 구독(최대 10개의 Subscriber)하도록 실험을 진행한다. 암호통신을 위해서 각 노드는 개별적인 인증서를 발행하여 Pairwise 키를 생성하여 통신한다. ROS2 환경에서 평문통신과 암호통신의 비교도 중요하지만 Subscriber 수에 따라 통신시간 차이를 비교하기 위해서 Publisher는 8개로 고정하고 Topic 크기로 1K로 고정한다. 대신 Subscriber의 수는 2개에서 최대 10개까지 변화를 주어 실험을 진행한다. 실제로 자율주행시 유통되는 토픽은 작게는 바이트단위에서 크게는 KB 단위로 1초에도 몇십개씩 지속적으로 발생하고 처리 노드들로 순차적으로 전달된다. 본 논문에서는 오버헤드에 영향을 끼치는 요소를 노드 수에 따른 암복호화 통신시간에 제한하기 위해서 토픽의 볼륨과 주기는 고정으로 제한시켰다.

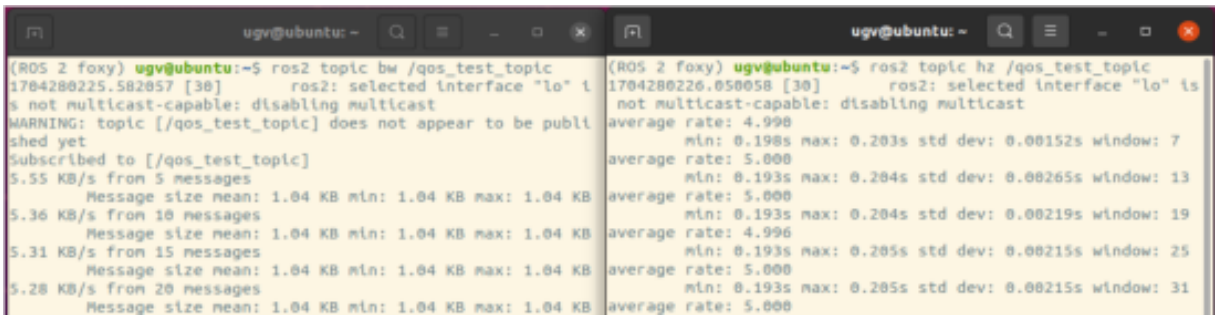

그림 6. Topic 전송 바이트 및 주기 확인
Fig. 6. Bytes and frequency for transmitted topic

Ⅳ. 실험 결과 및 분석

본 실험 결과는 Subscriber 수에 따른 통신시간은 그림 7과 같다. Publisher는 8개이고 Subscriber는 2, 4, 6, 8, 10개로 실험을 진행했다. 실험 결과 동일 Topic에 대해서 Subscriber 수가 많아지면 통신시간이 급격히 지연됨을 확인할 수 있다. 특히 Subscriber 수가 10개를 넘어설 경우 기하급수로 증가됨을 알 수 있다. 이는 ROS2에서 암호키를 노드 간 Pairwise 키로 생성하기 때문이다(본 실험에서는 ECDH로 노드 간 Pairwise 키를 생성했음). 따라서 평문에서는 Publisher가 멀티캐스트를 수행하는 반면 암호통신에서는 Pairwise 키 특성상 1:1 유니캐스트를 수행해야 한다. 이는 아래의 간단한 RTPS 실험을 통해서도 알 수 있다.

그림 7. Subscriber 수에 따른 통신시간
Fig. 7. Communication time according to the number of subscribers

RTPS는 네트워크 내, 참여자 정보를 유지하고, 네트워크 참여자에 대한 정보를 기반으로 자동 검색을 해준다. 또한, QoS 중 Reliable 통신에서는 DataWriter가 DATA정보 뿐 아니라 HEARTBEAT와 GAP을 통해 DataWriter의 HistoryCache 정보를 같이 전달하고 DataReader는 ACKNACK을 통해 누락된 정보를 전달한다[13].

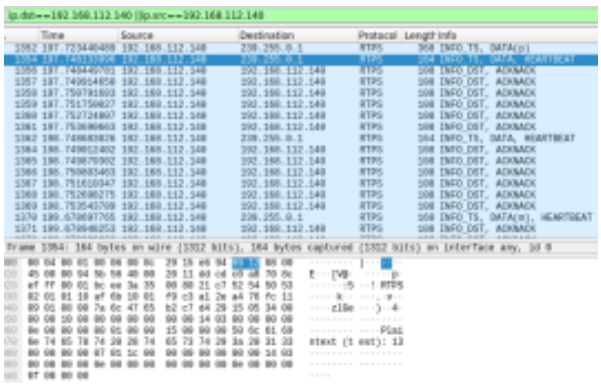

그림 8. 평문통신 시 RTPS 패킷
Fig. 8. RTPS packets in plaintext communication

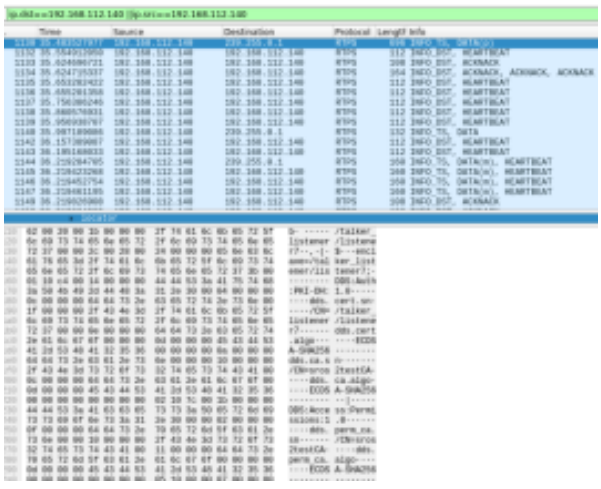

그림 9. 암호통신 시 RTPS 패킷
Fig. 9. RTPS packets in ciphertext communication

그림 8과 그림 9는 1개의 Publisher가 평문통신과 암호통신(Pairwise 키 생성)으로 카운트 정보를

Topic으로 발행하고 6개의 Subscriber가 이를 구독할 때 유통되는 패킷이다. 평문통신의 경우 하나의 HEARTBEAT와 6개의 ACKNACK을 전송하는 반면 암호통신에서는 다수의 HEARTBEAT를 전송함을 알 수 있다. 이는 Topic이 Pairwise 키로 암호화되었으므로 1:1 유니캐스트로 전송해야 되기 때문이다.

V. 결 론

무인화 시대에 ROS2 기반 자율주행기능은 필수불가결한 기능이고 이를 위해서 외부 물리공간의 센싱 정보로부터 주변 장애물 인식 및 위치 인식이 기본적으로 이뤄진다. 이 때, 아날로그 정보는 토픽 형태로 변환되어 내부네트워크 전반에 재발행되거나 유통된다. 본 논문은 토픽의 발행/구독의 빈도수가 매우 높은 ROS2 환경에서 지연없는 통신을 설계하기 위해서 Subscriber의 수에 따른 통신시간에 대해서 확인해보았다. 기존 연구에서는 평문통신과 암호통신에 따라 지연시간 혹은 QoS 옵션과 DDS 벤더간 통신에 대해 연구되어 왔지만 ROS2의 암호 특성을 고려하여 Subscriber 수가 통신시간에 미치는 영향도 확인이 필요하다. 본 실험을 통해서 Subscriber 수가 10개 이상으로 늘어날수록 통신시간은 급격히 증가됨을 확인하였다. ROS2의 플랫폼 특성상 향후 무인무기체계에서도 활용될 것으로 예상된다. 따라서 미래 무인 무기체계의 운영체계 구축 시 기본적인 미션을 지연 없이 수행하기 위해서는 Subscriber 수를 고려해서 설계해야 한다. 향후 UGV 암호통신 기반 자율주행 개발 시 본 실험 결과를 기초로 노드를 구성하여 설계를 진행할 예정이다.

References

[1] B. Dieber, B. Breiling, S. Taurer, and S. Kacianka, "Security for the robot operating system", Robotics and Autonomous Systems, Vol. 98, pp. 192-203, Dec. 2017. https://doi.org/10.1016/j.robot.2017.09.017.

[2] A. Botta, S. Rotbei, S. Zinno, and G. Ventre, "Cyber Security of Robots: a Comprehensive Survey", Intelligent Systems with Applications, Vol. 18, May 2023. https://doi.org/10.1016/j.iswa.2023.200237.

[3] H. Lee, C. Han, M. Park, and C. Yoo, "A Survey of Vulnerability Analysis based on ROS", In Proc. of KIIT Conference, JeJu, Korea, pp. 261-263, Jun. 2023.

[4] Y. Maruyama, S. Kato, and T. Azumi, "Exploring the Performance of ROS2", In Proc. of the 13th International Conference on Embedded Software, New York, United States, No. 5, pp. 1-10, Oct. 2016. https://doi.org/10.1145/2968478.2968502.

[5] J. Kim, J. M. Smereka, C. Cheung, S. Nepal, and M. Grobler, "Security and Performance Considerations in ROS 2: A Balancing Act", arXiv preprint arXiv:1809.09566, Sep. 2018. https://doi.org/10.48550/arXiv.1809.09566.

[6] M. Artsen, K. Banga, K. Talko, and D. Touw, "Analyzing Interoperability and Security Overhead of ROS2 DDS Middleware", 2022 30th Mediterranean Conference on Control and Automation (MED), Vouliagmeni, Greece, pp. 976-981, Jun. 2022. https://doi.org/10.1109/MED54222.2022.9837282.

[7] H. Lee, C. Han, M. Park, and C. Yoo, "The Survey for ROS Concept and Its Security Trend", ADD, Feb. 2023

[8] B. Dieber, B. Breiling, S. Taurer, and S. Kacianka, "Security for the Robot Operating System", Robotics and Autonomous Systems, Vol. 98, pp. 192-203, Dec. 2017. https://doi.org/10.1016/j.robot.2017.09.017.

[9] https://rb2023ros.github.io/kr [accessed: Jan. 04, 2024]

[10] https://github.com/ros2/sros2 [accessed: Jan. 04, 2024]

[11] N. Goerke, D. Timmermann, and I. Baumgart, "Who Controls Your Robot? An Evaluation of ROS Security Mechanisms", 2021 International Conference on Automation, Robotics and Applications, Prague, Czech Republic, pp. 60-66, Feb. 2021. https://doi.org/10.1109/ICARA51699.2021.9376468.

[12] J. J. Rengers, "DDS in a Zero Trust Cloud Native Environment in the Naval Domain", THALES, version 1.0.0 16-11-2022, 2022.

[13] O. M. G. Specification, "The real-time publish-subscribe protocol (RTPS) DDS interoperability wire protocol specification", Object Management Group ptc-18-09-03, 2018.

저자소개

이 화 성 (Hwaseong Lee)

2006년 2월 : 고려대학교 정보보호학과(공학석사)
2013년 2월 : 고려대학교 정보보호학과(공학박사)
2013년 9월 ~ 현재 : 국방과학연구소 선임연구원
관심분야 : ROS2, DDS-Security, 사이버 공격

이 현 우 (Hyunwoo Lee)

2019년 2월 : 고려대학교 사이버국방학과(공학사)
2023년 8월 : 고려대학교 정보보안학과(공학석사)
2023년 8월 ~ 현재 : 국방과학연구소 현역연구원
관심분야 : IVN, 사이버전

허 선 동 (Seondong Heo)

2011년 2월 : 한국과학기술원 전산학과(공학석사)
2017년 2월 : 한국과학기술원 전산학과(공학박사)
2017년 2월 ~ 현재: 국방과학연구소 선임연구원
관심분야 : 침입탐지 및 대응

박 무 성 (Moosung Park)

1990년 2월 : 서강대학교 전자계산학과(전자계산 석사)
2005년 2월 : 광운대학교 전자공학과(전자공학 박사수료)
1990년 1월 ~ 현재 : 국방과학연구소 수석연구원
관심분야 : 사이버보안, 사이버훈련, IoT 보안, Robot 보안

유 찬 곤 (Changon Yoo)

1997년 2월 : 충남대학교 전산학과(이학사)
2003년 8월 : South Dakata State University Computer Science(공학석사)
2003년 8월 ~ 현재 : 국방과학연구소 책임연구원
관심분야 : MUM-T 보안