머신러닝 기법을 활용한 ROS 2 의 콜백 실행 시간 분석 및 이상 탐지

김경환 [1], 강정환 [2], 권동현 [3]
[1] 부산대학교 정보컴퓨터공학부 학부생
[2] 부산대학교 정보융합공학과 박사과정
[3] 부산대학교 정보컴퓨터공학부 교수 (교신저자)

{kyounghwankim, jeonghwan, kwondh}@pusan.ac.kr

Anomaly Detection on Callback Duration in ROS 2 with Machine Learning

Kyoung-Hwan Kim[1], Jeong-Hwan Kang[2], Dong-Hyun Kwon[1]
[1]School of Computer Science and Engineering, Pusan National University
[2]Dept. of Information Convergence Engineering, Pusan National University

요 약

ROS 2 는 노드라는 프로세스로 구성되고, 실행기에 의해 실행되는 대부분의 동작은 노드와 연결된 콜백 함수로 정의되어 있다. 따라서, 특정 노드에 연결된 콜백 함수가 정상적으로 동작하지 않을 경우 애플리케이션 전체에 영향을 줄 수 있다. 이에 본 연구에서는 ROS 2 의 실행 정보를 추적하는 도구인 ros2_tracing 을 활용하여 ROS 2 애플리케이션 내에서 동작하는 콜백 함수의 실행 시간을 분석하고, 머신러닝 기법을 사용해 이를 탐지하는 방법을 제안한다.

1. 서론

ROS(Robot Operating System)가 산업 현장과 연구 기관에서 널리 사용됨에 따라, Robot Operating System 환경에서 이상 징후를 탐지하기 위한 연구는 꾸준히 지속되어 왔다. ROS 2 의 미들웨어 계층에서 발생하는 네트워크 패킷을 가로챈 뒤 이를 분석하여 이상 징후를 탐지하는 연구가 대표적이다[1]. 그러나 네트워크 패킷에 대한 암호화 기법을 적용했을 때 해당 연구에서 제안한 탐지 기법을 적용할 수 없게 된다.

ROS 2 애플리케이션은 노드라는 프로세스로 구성되어 있고, 노드 간의 통신은 토픽, 서비스, 액션이라는 인터페이스를 통해 수행된다. 또한, ROS 2 애플리케이션에서 실행기에 의해 실행되는 대부분의 동작은 콜백 함수로 정의되어 있다[2]. ROS 2 에서 동작하는 콜백 함수로는 구독, 타이머, 서비스 콜백 함수 등이 존재한다. 각 노드들은 하나 이상의 콜백 함수를 포함하게 되며, 이러한 콜백 함수를 통해 노드 간의 메시지 처리 및 통신을 수행한다. 많은 노드들과 콜백 함수들이 유기적으로 연결되어 로봇의 움직임을 구현해 내기 때문에 특정 노드에 연결된 콜백 함수가 정상적으로 동작하지 않을 경우 애플리케이션 전체에 영향을 줄 수 있다.

이러한 특징을 이용하여, 본 연구에서는 ROS 2 의 실행 정보를 추적하는 도구인 ros2_tracing[3]을 활용하여 ROS 2 애플리케이션 내에서 각 노드 별로 실행되는 콜백 함수의 실행 시간을 분석하였다. 이때, 공격자가 콜백 함수에 존재하는 어떠한 취약점을 악용하여 콜백 함수의 수행을 변조하는 경우, 콜백 함수가 개발자의 의도와 다른 비정상적인 동작을 수행할 것이며 공격자가 의도하는 일련의 공격을 수행하기 위해 평소와는 다른 실행 시간이 관측될 것이라고 판단하였다. 따라서, 본 연구에서는 콜백 함수의 실행 시간을 관측하여 기존에 비해 비정상적으로 긴 실행 시간을 가지는 경우, 이상 징후로 정의하고 이를 판별하기로 하였다. 이를 위해 비지도 학습 기반의 머신러닝 기법인 Isolation Forest, One-Class SVM, DBSCAN 모델을 사용하여 실험하였다.

2. 구현 및 실험

2-1. 데이터 수집

본 연구에서 수행하는 모든 실험은 Ubuntu 22.04 LTS 운영체제 및 ROS 2 Iron 배포판에서 수행되었다. 실험을 위해 서비스 요청을 위한 client 노드와, client 의 요청을 처리하는 server 노드를 포함한 ROS 2 애플리케이션을 구현했다. 이때, client 노드의 요청 메시지

a 만큼 $for\ loop$를 수행하는 동작을 server 노드의 서비스 콜백 함수로 정의했다.

해당 환경에서 약 3 분간 client 노드가 server 노드를 향해 지속적으로 서비스를 요청하는 실험을 진행하며 ros2_tracing 을 이용해 계측 지점에서의 로그를 수집했다. 이때, server 노드의 이상 징후를 주입하기 위해 $for\ loop$를 수행하는 횟수를 결정하는 a의 값을 낮은 확률로 기존의 값보다 상당히 큰 값으로 지정하여 요청한다. a의 값은 아래 식 (1)과 같이 결정된다.

$$a = \begin{cases} 10{,}000, & with\ probability\ 0.99 \\ 200{,}000, & with\ probability\ 0.01 \end{cases} \quad (1)$$

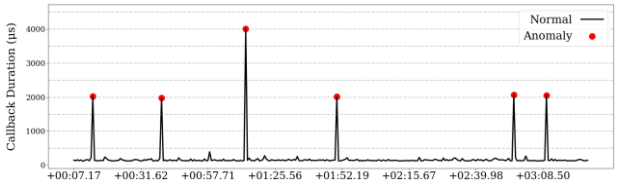
2-2. 데이터 전처리

ros2_tracing 은 LTTng[4] 기반으로 동작하기 때문에 수집된 로그를 CTF(Common Trace Format) 형식으로 저장한다. 따라서, 저장된 CTF 파일을 babeltrace2[5]를 통해 Human-Readable 한 파일로 변환해 준다.

변환된 로그를 기반으로 노드 생성과 같은 일회성 정보들을 분석하여 실행 중인 노드들의 정보와 해당 노드에 연결된 서비스, 콜백 함수 등과 같은 정보를 얻었다. 이후 server 노드에 연결된 콜백 함수의 수행 시간을 계산하였다. 콜백 함수의 수행 시간은 아래의 식 (2)에 따라 계산하였다.

$$T_{duration} = T_{callback\_end} - T_{callback\_start} \quad (2)$$

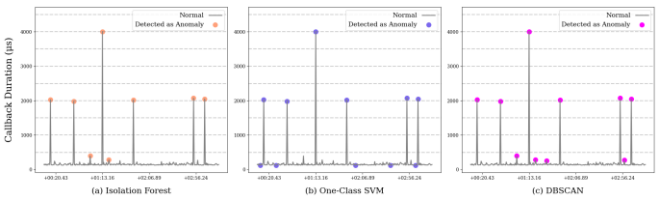
2-3. 성능 평가



(그림 1) server 노드의 콜백 함수 수행 시간 시각화

Server 노드의 콜백 함수 수행 시간을 그림 1 과 같이 시각화하였다. 정상적인 요청에 비해 비정상적인 요청을 받았을 때의 콜백 함수의 수행 시간이 확연히 높은 것을 눈으로 확인할 수 있었다.

이후 해당 데이터를 이용해 비지도 학습 기반의 이상치 탐지 알고리즘 모델인 Isolation Forest, One-Class SVM, DBSCAN 을 적용했다. 모델의 성능 평가 지표로는 정밀도, 재현율, 특이도, F1-Score 를 사용했다. 그 결과는 아래의 그림 2 와 같다.



(그림 2) 머신러닝 기법을 활용한 이상 징후 탐지 시각화

<표 1> 콜백 수행 시간 기반 이상 탐지 모델 적용 결과

Model	*Prec*	*Rec*	*Spec*	*F1*
Isolation Forest	1.0	0.993	1.0	0.997
One-Class SVM	1.0	0.983	1.0	0.991
DBSCAN	1.0	0.986	1.0	0.993

각 모델을 적용한 결과를 나타내는 표 1 에서 알 수 있듯이 대다수의 모델에서 이상치를 잘 탐지해 내는 것을 확인하였고, 성능 평가 지표를 기준으로 Isolation Forest, DBSCAN, One-Class SVM 순으로 탐지 성능이 우수하다는 것을 확인하였다.

3. 결론

본 연구에서는 ros2_tracing 을 활용해 수집된 ROS 2 애플리케이션의 로그를 전처리하고, 이를 통해 특정 노드의 콜백 수행 시간 데이터를 구축하였다. 이후 머신러닝 기법을 통해 ROS 2 에서의 콜백 함수 수행 시간 기반의 이상 탐지를 수행하였다. 실험 결과, 비정상적인 요청에 대한 콜백 함수의 실행 시간에서 확연한 차이가 발생하였으며, 이를 학습된 머신러닝 모델을 통해 탐지할 수 있었다.

향후 연구 계획으로는 ros2_tracing 을 활용해 더욱 다양한 정보를 수집하여 데이터를 구축할 것이며, 콜백 함수의 실행 시간뿐만 아니라 더욱 다양한 ROS 2 의 시스템 성분을 고려함으로써 향상된 성능과 다양한 형태의 이상 행위를 탐지하기 위한 머신러닝, 딥러닝 기반 이상 탐지 시스템으로 확장할 것이다.

사사문구

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학 ICT 연구센터사업의 연구결과로 수행되었음 (IITP-2024-RS-2023-00259967).

참고문헌

[1] Rivera, Sean, et al. "ROS-FM: fast monitoring for the robotic operating system (ROS)." *2020 25th International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2020.

[2] Bédard, Christophe, Ingo Lütkebohle, and Michel Dagenais. "ros2_tracing: Multipurpose low-overhead framework for real-time tracing of ROS 2." *IEEE Robotics and Automation Letters* 7.3 (2022): 6511-6518.

[3] Open Robotics. 2024. ROS 2 Documentation: Iron - Using Callback Groups. https://docs.ros.org/en/iron/How-To-Guides/Using-callback-groups.html. [Online; accessed 31-March-2024].

[4] Desnoyers, Mathieu, and Michel R. Dagenais. "The lttng tracer: A low impact performance and behavior monitor for gnu/linux." *OLS (Ottawa Linux Symposium)*. Vol. 2006. Citeseer, 2006.

[5] EfficiOS. 2024. Babeltrace Documentation (Babeltrace 2). https://babeltrace.org/. [Online; accessed 31-March-2024].