RoIFuzz: 강화된 로봇 보안 정책을 적용한 ROS IDL 퍼저 연구

2024. 11. 28

세종대학교 박민건, 유지현, 윤주범

세종대학교
SEJONG UNIVERSITY

Table of Contents

- 서론
- 배경지식
- **RoIFuzz**
- 실험 및 평가
- 결론

1. 서론

서론

- **연구 배경**
  - 로봇 시스템의 복잡성과 자율성 증가로 인한 보안 위협 증가
  - 선제적 보안 취약점 기술인 퍼징 기법을 로봇 보안 기술에 적용 시도 가 증가
  - 네트워크 관점의 취약점 외에 로봇의 취약점을 검사할 방법이 적음
  - ROS 기반 로봇의 보안 검사를 제안한 RoboFuzz가 제안되었으나, **상용 로봇에 초점이 있어** ROS IDL **같은 내부 패키지의 보안검사가 기초 단계에 머물러 있음**

- **제시 방향**
  - 기존 RoboFuzz의 한계점 극복을 위해 ROS IDL에 대한 퍼징 정책 강화하여 보안 안정성 확보
  - 새로운 취약점 발견을 위해 강화된 로봇 정책을 적용한 퍼징 시스템 RoIFuzz 제안

2. 배경 지식

2. 배경 지식

- **ROS2 (Robot Operating System 2)**
  - DDS 통신을 기반으로 노드 간 데이터를 교환하는 분산 로봇을 위한 메타 운영체제
  - 구성 요소
    - **Node :** 독립단위의 프로세스
    - **Message:** 정보의 기본 단위
  - 통신 방식
    - **DDS** 기반의 **RTPS Protocol**을 사용
    - 기본적으로 발행자-구독자 모델을 기반
    - **Topic, Service, Action**의 3가지 방식이 있으나, 주로 **Topic** 방식을 채택

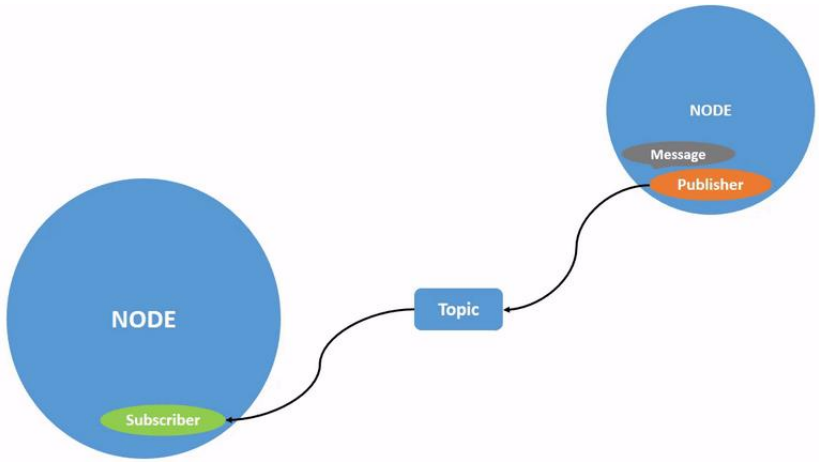

그림 1. Topic의 동작

https://docs.ros.org/en/foxy/_images/Topic-SinglePublisherandSingleSubscriber.gif

2. 배경 지식

- **ROS IDL(Interface Definition Language)**
  - ROS2 시스템에서 노드 간 통신에 사용되는 메시지, 서비스, 액션의 구조를 정의하는 언어
  - 노드 구조를 정의할 때 실질적인 자료형을 지정하고 시스템에 의해 실제 언어로 객체로 치환
  - 타입 안정성, 노드 간 통신의 일관성, 통합된 문법으로 인한 개발 편의성을 제공

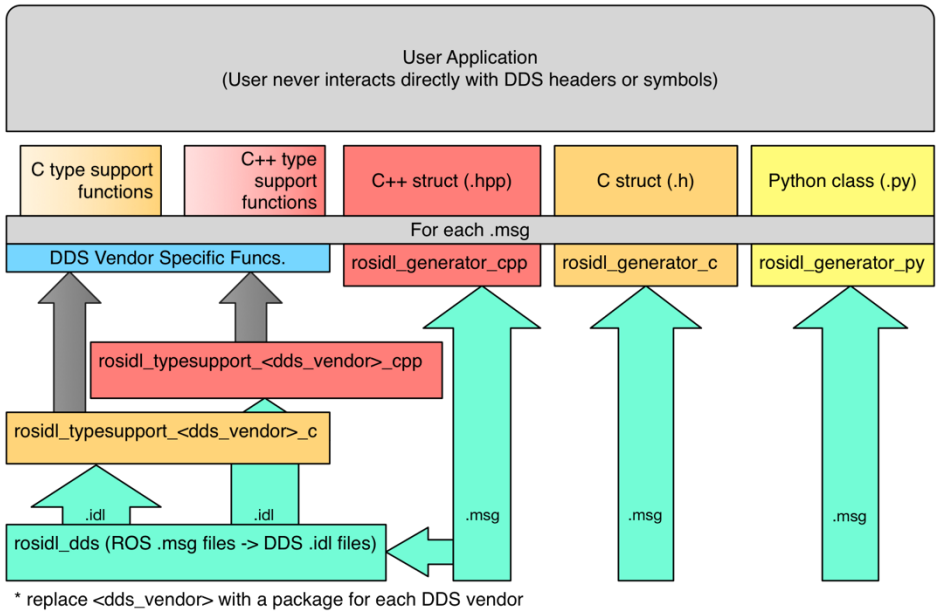

그림 2. IDL의 동작 구조

https://docs.ros.org/en/foxy/Concepts/About-Internal-Interfaces.html

2. 배경 지식

- 퍼징(Fuzzing)
  - 무작위로 생성된 데이터를 프로그램에 입력하여 취약점을 발견하는 소프트웨어 테스트 기법
  - 예상 치 못한 입력에 대한 프로그램의 반응을 확인하여 잠재적 버그나 보안 취약점을 사전 식별하여 소프트웨어의 안정성과 신뢰성을 향상
  - 전통적인 퍼징을 로봇 시스템에 적용하기에는 많은 도전과제가 존재함
  - 로봇 시스템은 사이버-물리의 특징을 가진 복잡한 시스템으로 분산 프로세스의 특징과 로봇 자체의 물리적 특성을 반영할 수 있는 퍼징 프로세스가 새롭게 필요

2. 배경 지식

- **RoboFuzz**
  - **ROS** 기반 로봇 시스템을 위한 피드백 기반 퍼징 프레임워크
  - 정확성 오라클과 **Semantic Feedback Mechanism**을 이용한 정확성 버그 탐지
  - **IDL**에 정의된 노드의 속성들을 기반으로 값을 **mutation**하며 퍼징 테스트를 할 수 있는 시스템을 제안
  - 그러나 **IDL** 오라클의 제한적인 보안 검사로 인해 **IDL**에 대한 취약점 탐지 한계

RoboFuzz. Figure 4, https://doi.org/10.1145/3540250.3549087

3. ROIFUZZ

3. RoIFuzz

- **RoIFuzz: 강화된 ROS IDL 퍼저**
  - **RoboFuzz의 IDL Oracle을 기반으로 보안 정책을 강화하여 새로운 취약점을 발견할 수 있는 RoIFuzz(Robot Interface Fuzzer) 제안**
  - **주요 목표 : ROS IDL의 취약점 탐지 오라클 개선**

실행 흐름:

1. **ROS Target 시스템을 매개변수로 System Inspector 에 지정**
2. **System Inspector에 의해 ROS Network Graph를 list로 반환**
3. **Scheduler에 의해 Target 노드와 Mutation Schedule을 지정**
4. **Message Mutator에 의해 Target Node의 테스트 토픽을 무작위로 선택해서 메시지 발행(Publish)**
5. **Oracle에 의해 Bug를 선별하여 Bug Report에 정리하여 반환**

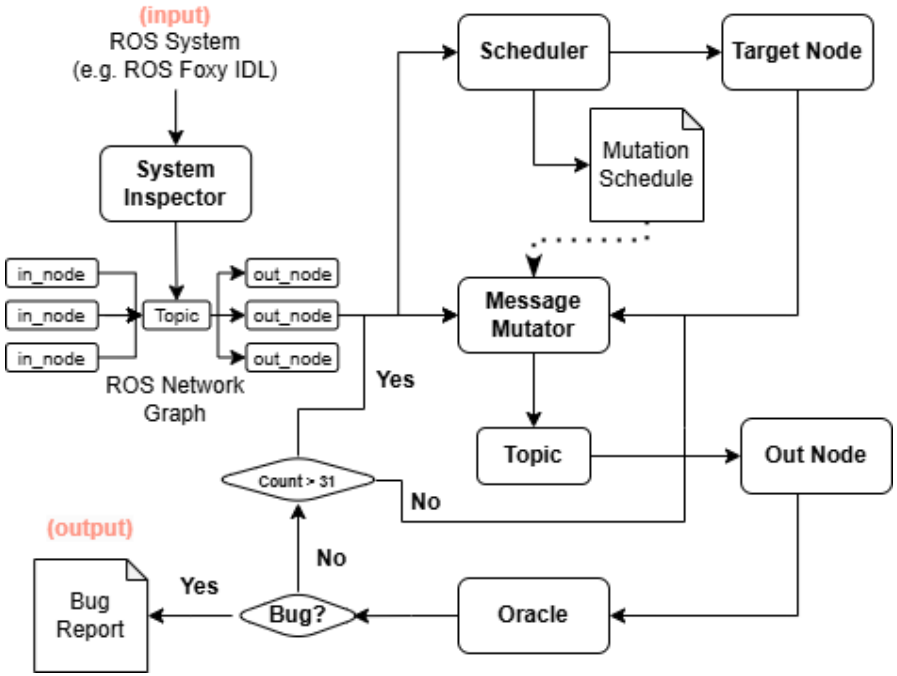

그림 3. RoIFuzz 아키텍처

3. RolFuzz

- **RoboFuzz의 ROS IDL Oracle**
  - **RoboFuzz의 노드의 메시지를 기반으로 퍼징 하는 프로세스는 동일**
  - **Msg/Uint32 같은 더미 메시지를 생성하여 각 노드의 속성의 값을 변조하여 테스트**
  - **RoboFuzz의 Oracle은 총 4가지를 불변 검사**
    - 1. 타입 일치 검사
    - 2. 최대값 검사
    - 3. 최소값 검사
    - 4. 배열 요소 타입 일치 검사

Algorithm 1 ROS IDL Correctness Oracle on RoboFuzz

1: **if** f_1에 $t \neq t'$인 t' 타입의 값이 할당됨 **then**
2: 할당 실패
3: **else if** f_1에 $\max(t_1)$보다 큰 값이 할당됨 **then**
4: 할당 실패
5: **else if** f_1에 $\min(t_1)$보다 작은 값이 할당됨 **then**
6: 할당 실패
7: **else if** f_2의 원소 중 하나에 $t_2 \neq t'$인 t' 타입의 값이 할당됨 **then**
8: 할당 실패
9: **else**
10: 할당 성공
11: **end if**

알고리즘1. RoboFuzz IDL 검사 오라클

3. RoIFuzz

- **RoIFuzz의 ROS IDL Oracle**
  - **RoIFuzz의 불변 검사를 개선**
    1. 타입 검사 정교화
       - 노드 간 타입 일치 여부 명시적 검사
       - 복잡한 사용자 정의 타입 데이터 검사 개선
    2. 경계 검사 강화
       - 특수 값 **(-inf, inf)** 검사 분리
       - 범위 단위 검사 확장(주변 검사 추가)
    3. 배열 타입 검사 개선
       - **isInstance** 함수와 **numpy**를 이용해 정확한 타입 검사
    4. 데이터 구조 유효성 검사
       - 메시지 구조가 특정 속성을 가지는지 검사 **(**일반적 속성에 대해 추가 **)**
    5. 에러 처리 및 리포팅 개선
       - 에러 중복 제거

Algorithm 2 개선된 ROS IDL Correctness Oracle

```
function CHECK(config, msg_list, state_dict, feedback_list)
    errs ← 빈 리스트
    msg ← 테스트할 메시지 선택
    msg_name ← msg의 타입 이름
    topic_name ← "/idltest_" + msg_name + "_out"
    if topic_name이 state_dict에 없음 then
        errs에 "Topic {topic_name} is lost" 추가
    else
        msg_out_list ← state_dict[topic_name]
        if msg_out_list의 길이 ≠ 1 then
            "[-] multiple messages replayed by idl target" 출력
            프로그램 종료 (-1)
        end if
        (ts, msg_out) ← msg_out_list[0]
        if msg의 타입 ≠ msg_out의 타입 then
            errs에 "Message types do not match" 추가
        else if msg 또는 msg_out에 'data' 속성이 없음 then
            errs에 "Invalid message structure: 'data' attribute missing" 추가
        else
            if msg.data의 타입이 배열 then
                if msg.data의 길이 ≠ msg_out.data의 길이 then
                    errs에 "Sent and replayed array lengths do not match" 추
가
                else
                    for i ← 0 to msg.data의 길이 - 1 do
                        if msg.data[i]의 타입 ≠ msg_out.data[i]의 타입 then
                            errs에 "Array element types do not match at index
{i}" 추가
                        else if msg.data[i] ≠ msg_out.data[i] then
                            errs에 "Array element values do not match at index
{i}" 추가
                        end if
                    end for
                end if
            else
                if msg.data ≠ msg_out.data then
                    errs에 "Sent and replayed data do not match" 추가
                end if
            end if
        end if
    end if
    중복 제거된 errs를 err-report 파일에 기록
    return errs
end function
```

알고리즘2. RoIFuzz 검사 오라클

4.실험 및 평가

4. 실험 및 평가

- 실험 환경
  - 운영체제 : **Ubuntu 20.04 64bit**
  - **ROS 버전 : ROS2 Foxy**
  - 미들웨어 : **FastDDS**
  - 프로그래밍 : **Python3**
  - 실험 대상 : **ROS IDL 시스템**
  - 실험 흐름 : **RoboFuzz Docker Container**를 기반으로 강화된 정책 적용 후 12시간 동안 취약점 탐지 테스트 진행

4. 실험 및 평가

- 실험 결과
  - 데이터 타입 처리 : 1, 2, 5
  - 배열 요소 타입 검사 누락
    - 3, 4, 6, 7, 8, 11
  - 메시지 구조 검증 : 9
  - 특수 값 검사 : 10
  - 총 11개의 취약점을 발견
  - ROS2 메시지 처리 과정에서 취약점이 발생할 수 있음
  - 배열 관련 취약점이 다수 인 것으로 보아 배열 구조가 취약

취약점 분류	RoIFuzz	RoboFuzz
데이터 타입 처리	3	2
배열 요소 타입 검사 누락	6	6
메시지 구조 검증	1	0
특수 값 검사	1	0
합계	**11**	**8**

표2. RoIFuzz와 RoboFuzz 취약점 결과

Table 1: RoIFuzz에 의해 발견된 ROS IDL 버그 목록

#	버그 설명
1	32/64비트 float 경계 미확인, 모든 float을 double로 취급
2	byte 타입 오처리, 내부적으로 문자열 리터럴로 취급
3	int 배열 요소 데이터 범위 검사 누락
4	float 배열 요소 데이터 범위 검사 누락
5	배열 요소 암시적 타입 변환으로 인한 데이터 변경
6	bool 배열 요소 타입 검사 누락
7	byte 배열 요소 타입 검사 누락
8	string 배열 요소 타입 검사 누락
9	메시지 구조 유효성 검사, 노드 생성 시 메시지 구조 검사
10	특수값 검사, -inf와 inf에 대한 특수값 검사를 추가하여 엄밀히 구분
11	BoundedDynArray 배열 요소 타입 검사 누락

RoIFuzz: Robot Interface Fuzzer

표1. RoIFuzz에 의해 발견된 ROS IDL Bug List

4. 실험 및 평가

- 실험 평가
  - **RoboFuzz**에 비해 3개의 취약점을 추가 발견하여 **RoboFuzz**의 **IDL** 퍼저를 개선함
    - 배열 요소 암시적 타입 변환으로 인한 데이터 값 변경
    - 메시지 구조 유효성 검사
    - 특수 값 검사
  - 새로운 취약점 발견을 통해 로봇 시스템의 보안 안정성 강화에 기여
  - 실험 결과 메시지 처리 과정에서 보안 검사 및 유효성에 대한 중요성과 필요성을 시사
  - **IDL** 보안 검사의 한계점 => **IDL**을 정의하는 메시지 구조가 복잡해지면 검사 수행 시간이나 정확성이 낮아진다는 어려움이 있어 향후 연구과제로 남아 있음.

5.결론

4. 실험 및 평가

- 결론
  - RoIFuzz를 통해 강화된 로봇 정책을 적용하여 기존 RoboFuzz의 불변식을 강화함
  - 실험 중 11가지 취약점을 검출하여 로봇 시스템의 보안 안정성 강화에 기여
    - 데이터 타입 관련 취약점 3개
    - 배열 요소 타입 검사 누락 관련 취약점 7개
    - 메시지 구조 검증 관련 취약점 1개
  - 실험 결과, ROS 메시지 처리 과정에서 보안 검사 및 유효성 검증이 충분히 이뤄지지 않았음을 확인
- 향후 계획
  - IDL 검사의 한계점 개선을 위한 연구 진행 예정
    - 로봇 시스템의 복잡성으로 인해 IDL 타입이 복잡해지면 처리하는데 어려움이 존재
    - 강화학습이나 LLM을 도입하여 복합 메시지 생성, 피드백 매커니즘 개선 등을 연구하고자 함

감사합니다.

Q & A