오픈소스 드론시스템 대상 **Fuzzing** 환경 개발

:**PX4** 와 **libfuzzer** 의 활용

경희대학교 컴퓨터공학과 장대희

경희대학교
KYUNG HEE UNIVERSITY

PWNJAB
@KYUNGHEE UNIVERSITY

드론시스템 개요

❖드론 활용분야

- 군사적 목적
  - ✓ 정찰, 요격, 수송 등
- 개인 활용
  - ✓ 사진/영상 촬영
  - ✓ 놀이용
- 산업활용
  - ✓ 설비 점검
  - ✓ 농작물 관리
  - ✓ 건물 보안 등

출처:탄약투발형 소형드론 시스템의 필요성 및 발전방향, 한국군사과학기술학회

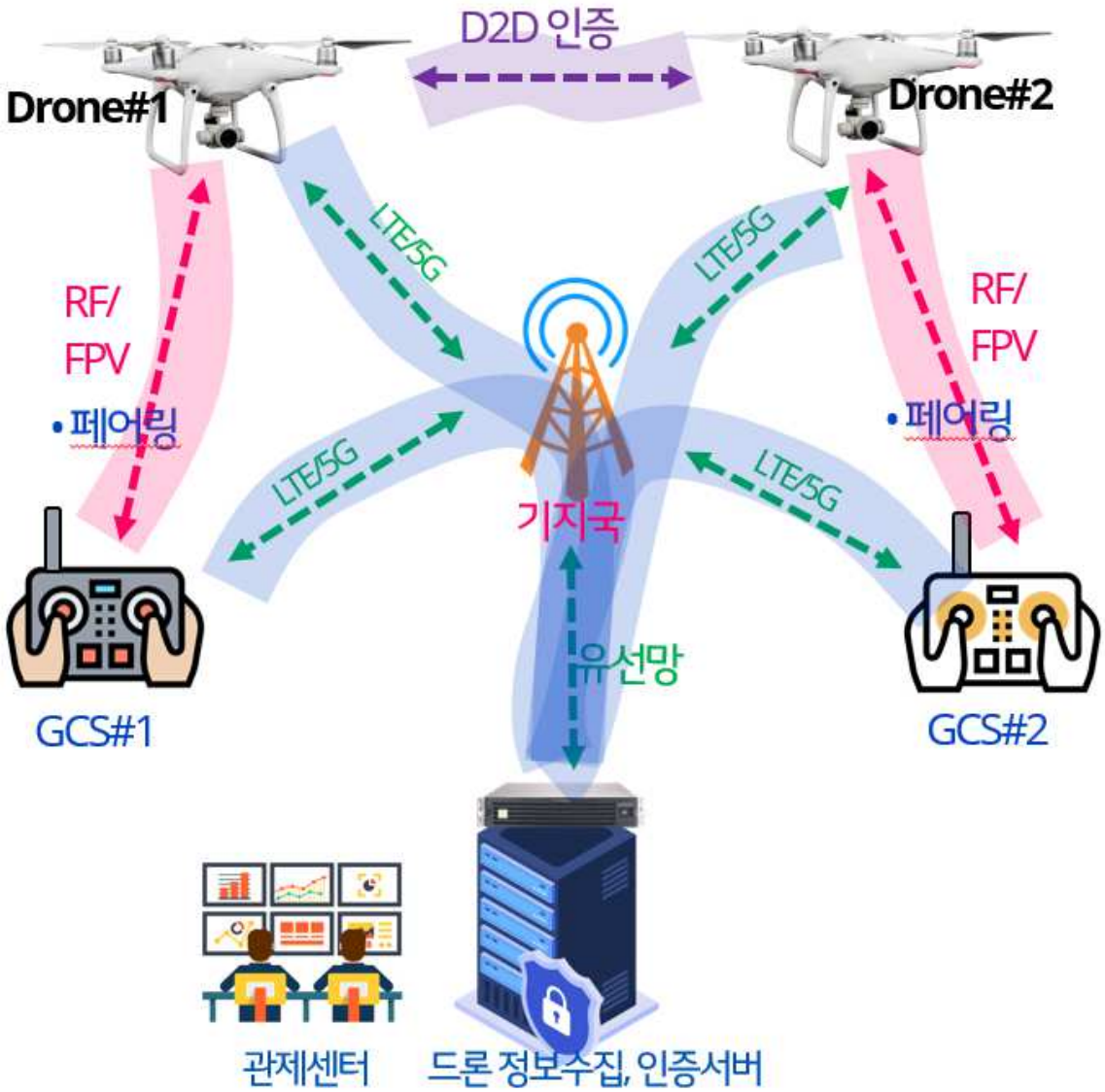
그림 5. 분야별 드론 시장

출처 : DroneDeploy, 「The Rise of Drones in Construction」, 2018

드론 시스템 구조

❖드론 구성 컴포넌트

- GCS
  - ✓ Ground Control Station
- RF
  - ✓ Radio Frequency
- FPV
  - ✓ First Person View
- D2D
  - ✓ Drone to Drone



드론 시스템과 보안

❖드론 시스템의 복잡도 증가

- 센서값 위변조
- 물리적 보안
- 다양한 Application 추가 -> 새로운 공격벡터
  - ✓ Malware?
  - ✓ Memory Hack?
  - ✓ Fuzzing?

드론의 소프트웨어 안정성

❖드론 시스템의 취약점 분류

- GPS
- 전파/신호/재밍
- 네트워크
  - ✓ Spoofing
  - ✓ MITM
  - ✓ 도청, 암호화
- 소프트웨어
  - ✓ 펌웨어 리버싱
  - ✓ 인증 실패
  - ✓ DoS 공격
  - ✓ 악성코드 삽입
- 인증/제어 시스템
- 포렌식 분석 공격 (칩 분해)
- 영상탈취
- 물리적 취약점

〔표 1〕 드론 주요 구성요소와 관련된 취약점 분류

구성요소	취약점	내용	관련연구
GPS	GPS 스푸핑	GPS 좌표 임의 변경을 통한 드론 납치	〔2〕, 〔3〕
자이로 스코프	오프셋 값 추적	자이로스코프의 오프셋 값을 사용하여 드론 추적	〔4〕
	센서 값 불안정	공진 주파수를 사용한 데이터 임의 변동	〔5〕
네트워크	개방형 Wi-Fi 취약점	개방형 Wi-Fi를 사용한 드론 납치	〔3〕, 〔6〕, 〔8〕, 〔10〕
	개방된 포트 취약점	개방된 포트를 통한 루트 접근 권한 획득	〔6〕
	퍼징 공격	데이터 무작위 주입을 통한 드론의 오작동 유도	〔11〕, 〔12〕
	DoS 공격	과도한 패킷 주입을 통한 시스템 과부하 발생	〔6〕, 〔9〕, 〔10〕, 〔11〕
	중간자 공격	위장을 통한 사용자 개인정보 데이터 탈취	〔6〕, 〔13〕
애플리케이션	하드코딩	주요 데이터 애플리케이션 내부 자체 배치	〔7〕, 〔8〕
	암호화되지 않은 데이터 전송 취약점	평문 데이터 노출을 통한 개인정보 탈취	〔9〕
인증	접근 제어	접근 인증 과정 부재로 인한 공격	〔3〕, 〔6〕, 〔7〕
	인증 해제	암호화 되어 이지 않은 부분을 사용한 인증 해제	〔3〕, 〔6〕, 〔8〕

출처: 무인이동체 드론의 취약점분석 및 대응기술 연구 동향 (정보보호학회지 30권 2호)

드론 구조 분석 (PX4 Intel Aero)

❖1개의 보드로 구성

- FC (Flight Controller)
- 소형 드론의 구조

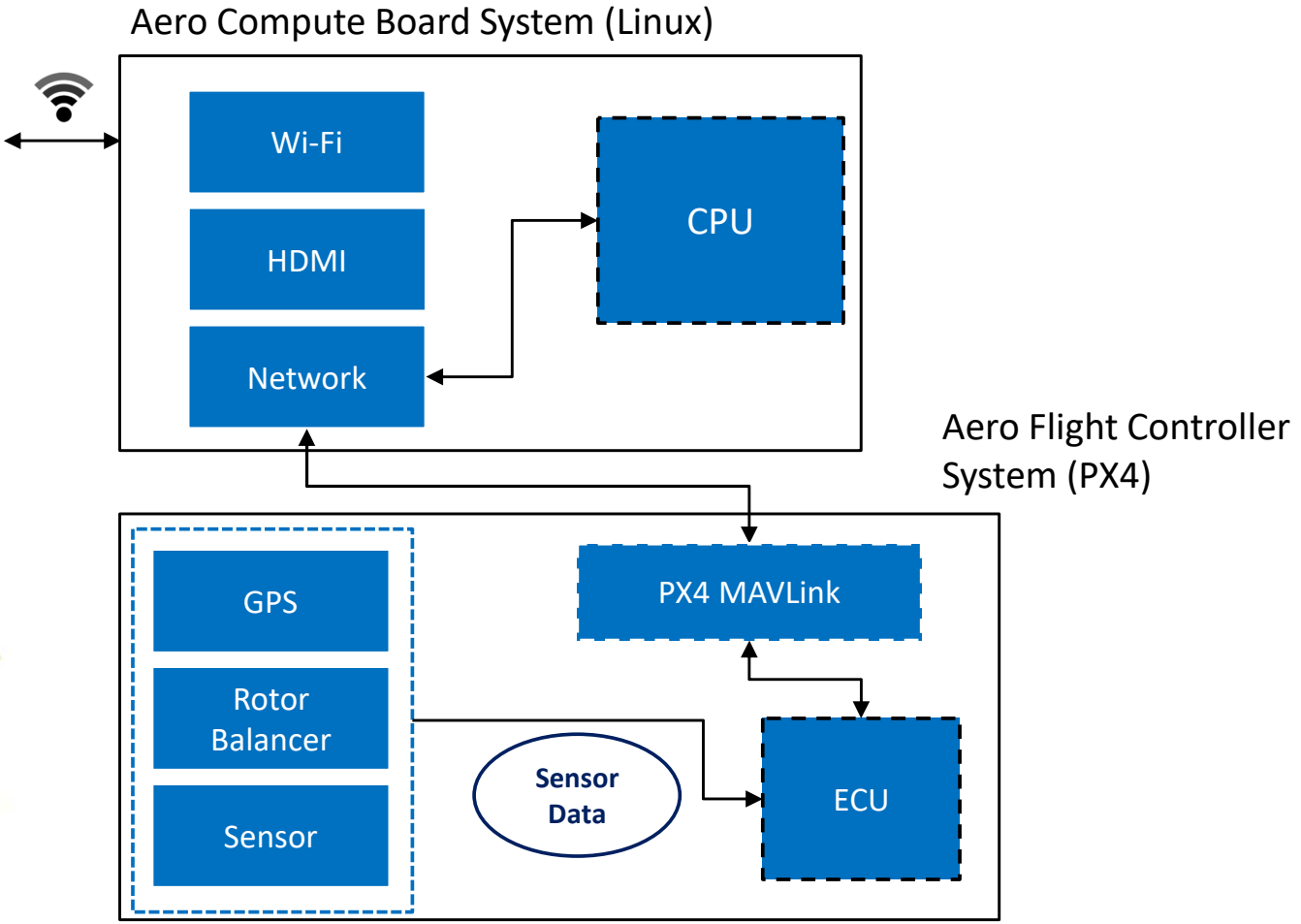
❖2가지 이상의 시스템으로 구성

- 컴퓨팅 시스템
  - ✓ 연산 능력을 위한 추가보드
- FC (Flight Controller)
  - ✓ 비행제어 시스템 (e.g., PX4)

드론 구조 분석 (Intel Aero)

Compute Board

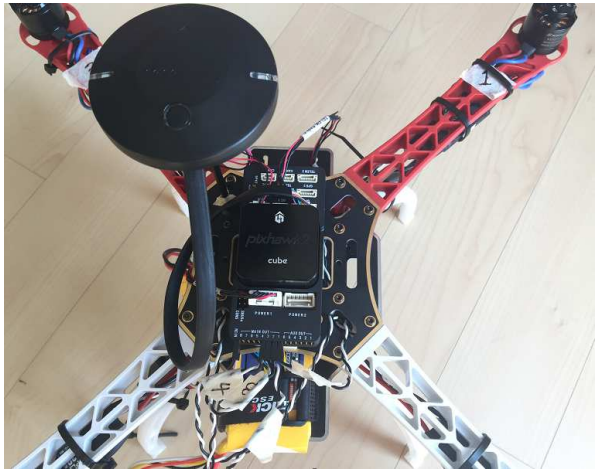
Flight Controller



PX4 와 DJI 드론

❖PX4

- 소스 코드 제공 O
- 각 기능 별 모듈화
- uORB 방식을 이용한 통신
- MAVLink 프로토콜 채택
  - ✓ GCS와 PX4 간 통신
  - ✓ UAV 시스템 및 구성 요소를 위한 경량 통신 프로토콜



❖DJI 드론

- 소스 코드 제공 X
- 통신 프로토콜: 모델 별 상이
  - ✓ Ocusync
    - ➢ 보안 기능이 내장된 독점 프로토콜
    - ➢ 공공 분야 (농업, 에너지, 미디어 등)
  - ✓ Wi-Fi 프로토콜
    - ➢ 개인 촬영용 (이미지 전송)

PX4 드론이 연구하기 좋은점

❖Open Source

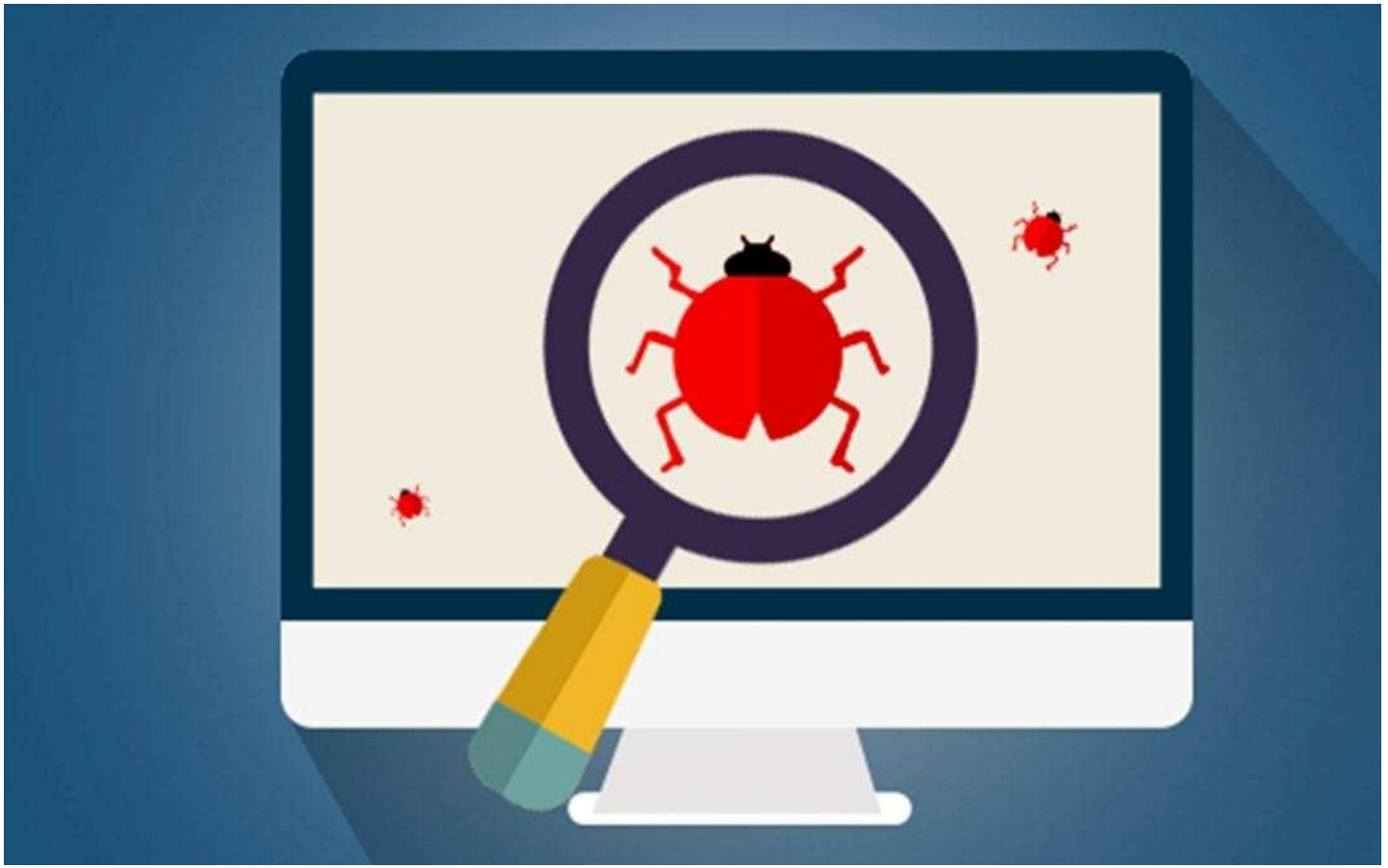
- 소스코드 분석이 가능함
- Sanitizer, Fuzzer 포팅 및 각종 실험이 가능함
- SSH/NSH 쉘을 통해 개발/테스팅/제어가 가능함

What is Fuzzing?

Bug

❖Finding "Bug"

- 소프트웨어 개발자의 실수



Fuzzing Basics

❖1. Input 생성

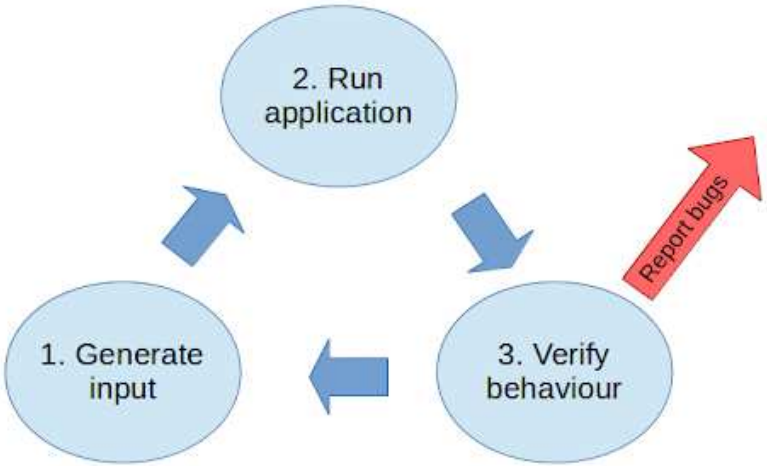
- 어떻게 하면 "좋은 Input" 을 생성할 수 있을까?
  - ✓ Dumb Fuzzing: 순수하게 Random Data 생성
  - ✓ Smart/Hybrid fuzzing: Random + @ 를 이용해서 버그 발견 확률을 증대
  - ✓ 수많은 연구가 진행중

❖2. 프로그램 실행

- 어떻게 하면 "빠르게" 실행할까?
  - ✓ 수많은 연구가 진행중

❖3. 결과 관찰

- 어떻게 하면 "Bug 를 놓치지 않을까?"
  - ✓ w/ Source Code
    - ◦ Instrumentation (Address Sanitizer)
  - ✓ w/o Source Code
    - ◦ Dynamic Binary Instrumentation (AFL-QEMU), Full-Speed Fuzzing (INT3 Trap)



Address Sanitizer

❖구글에서 개발. 퍼징을 위한 표준 도구.

❖필요성?

- int array[100];
- int a = array[100]; // error.

❖Memory 버그가 존재함에도 보통 아무일도 일어나지 않음

- 해결해야 할 문제

Address Sanitizer

❖프로그램의 모든 memory access 를 instrument

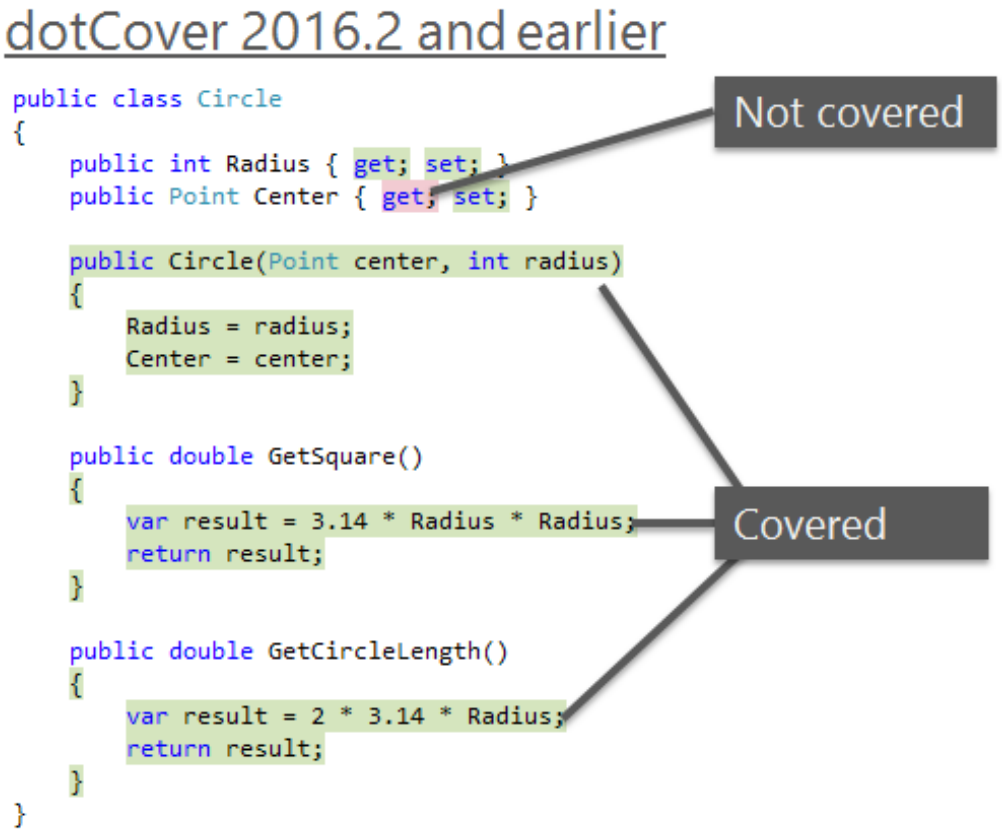
- 메모리 접근시마다 허용범위를 확인
  - ✓ RedZone
- 소스코드 필요


```
==10951==ERROR: AddressSanitizer: heap-buffer-overflow on address 0xb5301a80
READ of size 4 at 0xb5301a80 thread T0
    #0 0xb7195910 in jpc_pi_nextpcrl /home/fire/bing/afl/libraries/jasper/ja
    #1 0xb719249e in jpc_pi_next /home/fire/bing/afl/libraries/jasper/jasper
    #2 0xb719c56c in jpc_dec_decodepkts /home/fire/bing/afl/libraries/jasper/
    #3 0xb711efcf in jpc_dec_process_sod /home/fire/bing/afl/libraries/jaspe
    #4 0xb711da24 in jpc_dec_decode /home/fire/bing/afl/libraries/jasper/jas
    #5 0xb711ce62 in jpc_decode /home/fire/bing/afl/libraries/jasper/jasper-
```

Coverage?

❖프로그램의 실행흐름에 대한 측정

- Function Coverage
- Line Coverage
- Basic Block Coverage
- Edge Coverage
  - ✓ 일반적으로 많이 사용

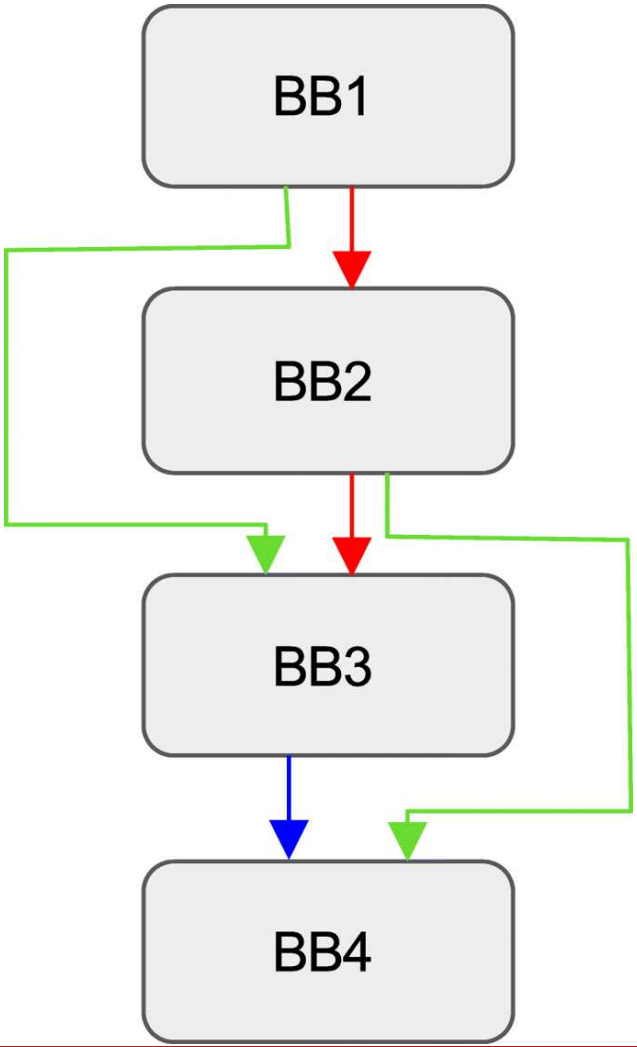


Block Coverage?

❖어셈블리 레벨에서 흐름 측정

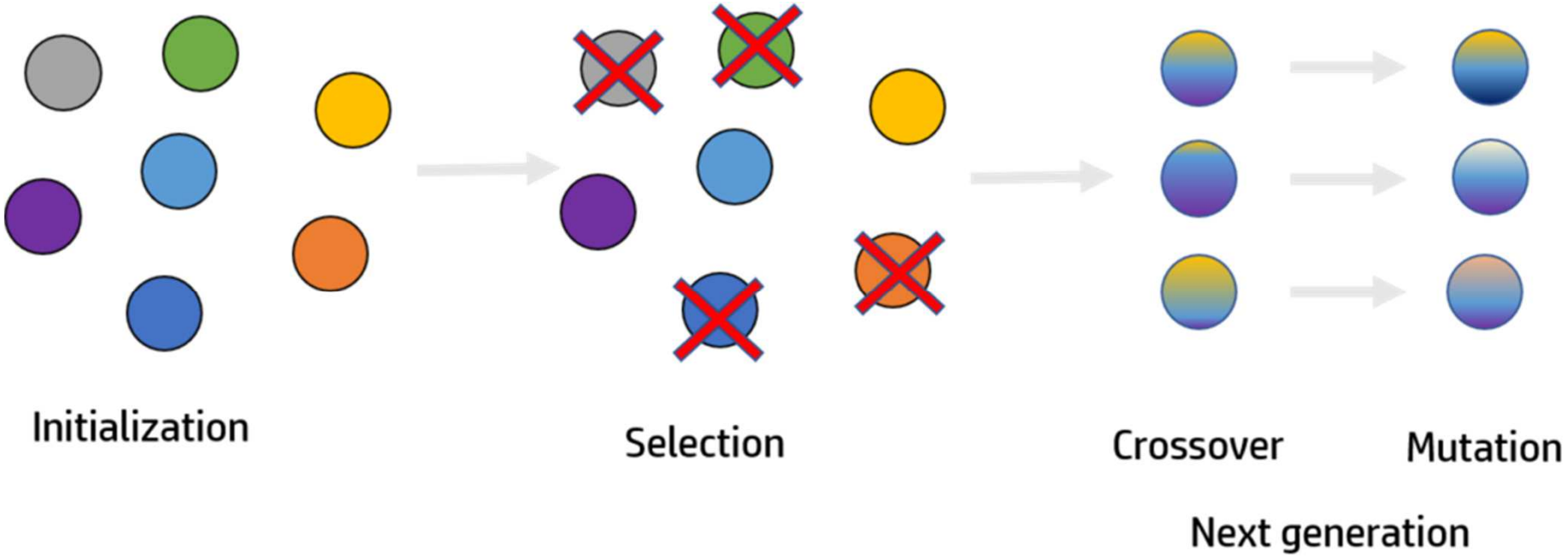
Edge Coverage?

❖프로그램 실행 측면에서 더 많은 경우의 수를 고려



유전 알고리즘

퍼징 Round (세대) 마다 새로운 Corpus Set 을 생성



유전 알고리즘

❖1세대: "H" 로 시작하는 입력들이 살아남음.

❖2세대: "I" 로 시작하는 입력들이 살아남음.

❖...

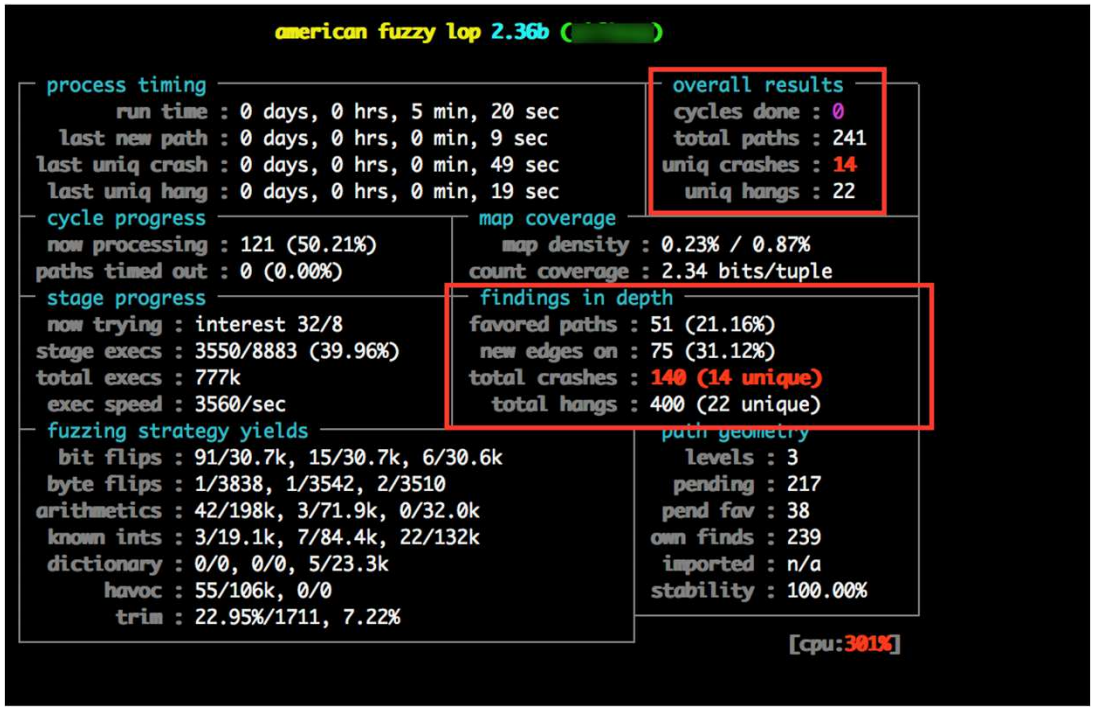
```
cat << EOF > test_fuzzer.cc
#include <stdint.h>
#include <stddef.h>
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *data, size_t size) {
  if (size > 0 && data[0] == 'H')
    if (size > 1 && data[1] == 'I')
       if (size > 2 && data[2] == '!')
       __builtin_trap();
  return 0;
}
EOF
# Build test_fuzzer.cc with asan and link against libFuzzer.
clang++ -fsanitize=address,fuzzer test_fuzzer.cc
# Run the fuzzer with no corpus.
./a.out
```

AFL (American Fuzzy Lop)

❖Google 에서 개발

❖초기의 Coverage-Feedback 기반 Fuzzer

- 2015 경부터 유명해짐
- 유전자 알고리즘 적용
- 비약적인 Fuzzing 능력 향상. Fuzzing Boom 의 시초가 됨



libFuzzer

❖Google 에서 개발, LLVM 프로젝트의 일환

❖AFL 이후 등장, 실제로 업계에서 가장 많이 사용
- 크롬 등 메이저 S/W 에서 수많은 버그들을 발견하고 patch 하는데 기여

❖In-Process Fuzzer
- 퍼징을 위해 프로그램 전체를 독립 실행하지 않음
- library 의 함수 단위로 퍼징을 수행
  - ✓ 비약적인 성능 향상, 효율성 도모
- 단점: Fuzzer Stub 의 작성단계가 필요함

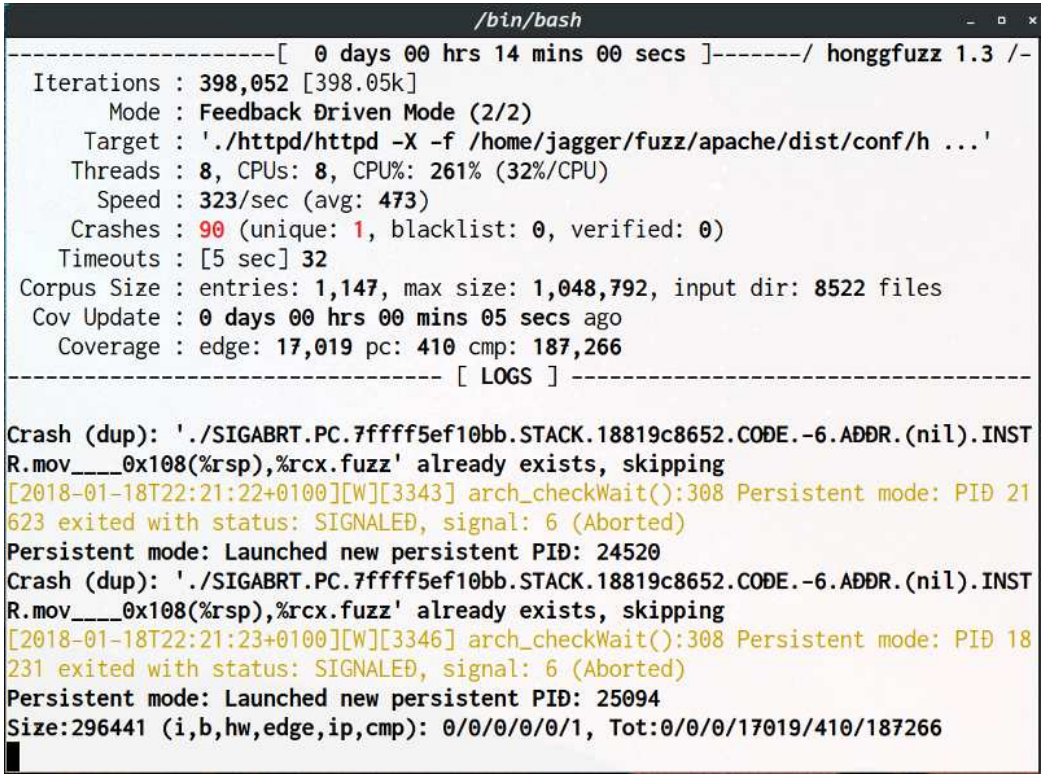
```
INFO: Seed: 1523017872
INFO: Loaded 1 modules (16 guards): [0x744e60, 0x744ea0),
INFO: -max_len is not provided, using 64
INFO: A corpus is not provided, starting from an empty corpus
#0      READ units: 1
#1      INITED cov: 3 ft: 2 corp: 1/1b exec/s: 0 rss: 24Mb
#3811   NEW    cov: 4 ft: 3 corp: 2/2b exec/s: 0 rss: 25Mb L: 1 MS: 5 ChangeBit-ChangeByte-ChangeBit-
#3827   NEW    cov: 5 ft: 4 corp: 3/4b exec/s: 0 rss: 25Mb L: 2 MS: 1 CopyPart-
#3963   NEW    cov: 6 ft: 5 corp: 4/6b exec/s: 0 rss: 25Mb L: 2 MS: 2 ShuffleBytes-ChangeBit-
#4167   NEW    cov: 7 ft: 6 corp: 5/9b exec/s: 0 rss: 25Mb L: 3 MS: 1 InsertByte-
==31511== ERROR: libFuzzer: deadly signal
...
artifact_prefix='./'; Test unit written to ./crash-b13e8756b13a00cf168300179061fb4b91fefbed
```

Honggfuzz

❖Google 에서 개발

❖Intel PT 기반 퍼저

- Intel CPU 의 특수 기능 (하드웨어적 로그수집) 을 기반으로 Coverage 측정
- 연구/실험용으로 사용, 대중적이지는 않음



libFuzzer 의 작동과정

Libfuzzer 컴파일

- Libfuzzer 제작방법
  - LLVMFuzzerTestOneInput 함수 작성
  - Libfuzzer 의 main 함수 역할

Toy example

A simple function that does something interesting if it receives the input "HI!":

```
cat << EOF > test_fuzzer.cc
#include <stdint.h>
#include <stddef.h>
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *data, size_t size) {
  if (size > 0 && data[0] == 'H')
    if (size > 1 && data[1] == 'I')
       if (size > 2 && data[2] == '!')
       __builtin_trap();
  return 0;
}
EOF
# Build test_fuzzer.cc with asan and link against libFuzzer.
clang++ -fsanitize=address,fuzzer test_fuzzer.cc
# Run the fuzzer with no corpus.
./a.out
```

Libfuzzer 컴파일

- Libfuzzer 코드 작성
  - 아래와 같은 코드를 “test.cpp” 로 작성

```
#include <stdint.h>
#include <stddef.h>
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *data, size_t size) {
  if (size > 0 && data[0] == 'H')
    if (size > 1 && data[1] == 'I')
       if (size > 2 && data[2] == '!')
       __builtin_trap();
  return 0;
}
```

Libfuzzer 컴파일

- Libfuzzer 제작방법
  - Clang 컴파일러의 기능 사용
  - -fsanitize 플래그
    - ✓ Address: Address Sanitizer
    - ✓ Fuzzer: libfuzzer

❖아래와 같이 C++ 코드 컴파일

- clang++ -fsanizite=address,fuzzer test.cpp
- 생성된 a.out 바이너리 확인

```
daehee@none:~/tmp$ clang++ -fsanitize=address,fuzzer test.cpp
daehee@none:~/tmp$ ls
a  a.c  AFLplusplus  a.out  a.txt  b.txt  chal.db  data  Docke
```

Libfuzzer 컴파일

- Libfuzzer 구동
  - ./a.out 바이너리 실행시 아래와 같은 모습 확인

```
daehee@none:~/tmp$ ./a.out
INFO: Running with entropic power schedule (0xFF, 100).
INFO: Seed: 2972326201
INFO: Loaded 1 modules   (8 inline 8-bit counters): 8 [0x55bdb981bed0, 0x55bdb981bed8),
INFO: Loaded 1 PC tables (8 PCs): 8 [0x55bdb981bed8,0x55bdb981bf58),
INFO: -max_len is not provided; libFuzzer will not generate inputs larger than 4096 bytes
INFO: A corpus is not provided, starting from an empty corpus
#2      INITED cov: 2 ft: 2 corp: 1/1b exec/s: 0 rss: 30Mb
#47     NEW    cov: 3 ft: 3 corp: 2/4b lim: 4 exec/s: 0 rss: 30Mb L: 3/3 MS: 5 ChangeByte-InsertByte-ChangeBit-ChangeByte-I
#62     NEW    cov: 4 ft: 4 corp: 3/5b lim: 4 exec/s: 0 rss: 30Mb L: 1/3 MS: 5 InsertByte-ShuffleBytes-CrossOver-CrossOver-
#83     REDUCE cov: 4 ft: 4 corp: 3/4b lim: 4 exec/s: 0 rss: 31Mb L: 2/2 MS: 1 EraseBytes-
#1073   NEW    cov: 5 ft: 5 corp: 4/13b lim: 11 exec/s: 0 rss: 31Mb L: 9/9 MS: 5 ShuffleBytes-CopyPart-CrossOver-CMP-Insert
#1094   REDUCE cov: 5 ft: 5 corp: 4/10b lim: 11 exec/s: 0 rss: 31Mb L: 6/6 MS: 1 CrossOver-
#1268   REDUCE cov: 5 ft: 5 corp: 4/9b lim: 11 exec/s: 0 rss: 31Mb L: 5/5 MS: 4 ChangeByte-ChangeBit-ChangeByte-EraseBytes-
#1313   REDUCE cov: 5 ft: 5 corp: 4/7b lim: 11 exec/s: 0 rss: 31Mb L: 3/3 MS: 5 CopyPart-CrossOver-CopyPart-ShuffleBytes-Er
#1322   REDUCE cov: 6 ft: 6 corp: 5/9b lim: 11 exec/s: 0 rss: 31Mb L: 2/3 MS: 4 ChangeBit-InsertByte-CrossOver-CrossOver-
==49770== ERROR: libFuzzer: deadly signal
    #0 0x55bdb97a6a81 in __sanitizer_print_stack_trace (/home/daehee/tmp/a.out+0xe4a81) (BuildId: 8b407bb4295723b2f90284f56
    #1 0x55bdb9719318 in fuzzer::PrintStackTrace() (/home/daehee/tmp/a.out+0x57318) (BuildId: 8b407bb4295723b2f90284f56de05
    #2 0x55bdb96fed93 in fuzzer::Fuzzer::CrashCallback() (/home/daehee/tmp/a.out+0x3cd93) (BuildId: 8b407bb4295723b2f90284f
    #3 0x7ff864a4251f  (/lib/x86_64-linux-gnu/libc.so.6+0x4251f) (BuildId: a43bfc8428df6623cd498c9c0caeb91aec9be4f9)
    #4 0x55bdb97d9f54 in LLVMFuzzerTestOneInput (/home/daehee/tmp/a.out+0x117f54) (BuildId: 8b407bb4295723b2f90284f56de05ae
    #5 0x55bdb9700323 in fuzzer::Fuzzer::ExecuteCallback(unsigned char const*, unsigned long) (/home/daehee/tmp/a.out+0x3e3
    #6 0x55bdb96ffa79 in fuzzer::Fuzzer::RunOne(unsigned char const*, unsigned long, bool, fuzzer::InputInfo*, bool, bool*)
56de05ae49b1d1cd4)
    #7 0x55bdb9701269 in fuzzer::Fuzzer::MutateAndTestOne() (/home/daehee/tmp/a.out+0x3f269) (BuildId: 8b407bb4295723b2f902
    #8 0x55bdb9701de5 in fuzzer::Fuzzer::Loop(std::vector<fuzzer::SizedFile, std::allocator<fuzzer::SizedFile> >&) (/home/d
9b1d1cd4)
    #9 0x55bdb96eff22 in fuzzer::FuzzerDriver(int*, char***, int (*)(unsigned char const*, unsigned long)) (/home/daehee/tm
)
    #10 0x55bdb9719c12 in main (/home/daehee/tmp/a.out+0x57c12) (BuildId: 8b407bb4295723b2f90284f56de05ae49b1d1cd4)
    #11 0x7ff864a29d8f  (/lib/x86_64-linux-gnu/libc.so.6+0x29d8f) (BuildId: a43bfc8428df6623cd498c9c0caeb91aec9be4f9)
    #12 0x7ff864a29e3f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x29e3f) (BuildId: a43bfc8428df6623cd498c9c0ca
    #13 0x55bdb96e4964 in _start (/home/daehee/tmp/a.out+0x22964) (BuildId: 8b407bb4295723b2f90284f56de05ae49b1d1cd4)

NOTE: libFuzzer has rudimentary signal handlers.
      Combine libFuzzer with AddressSanitizer or similar for better crash reports.
SUMMARY: libFuzzer: deadly signal
MS: 1 InsertByte-; base unit: dbee5f8c7a5da845446e75b4a5708e74428b520a
0x48,0x49,0x21,0x48,
HI!H
artifact_prefix='./'; Test unit written to ./crash-b13e8756b13a00cf168300179061fb4b91fefbed
Base64: SEkhSA==
```

Libfuzzer 컴파일

- Libfuzzer 구동 결과 해석
  - Fuzzer Iteration Count

```
#2      INITED cov: 2 ft: 2 corp: 1/1b exec/s: 0 rss: 30Mb
#47     NEW    cov: 3 ft: 3 corp: 2/4b lim: 4 exec/s: 0 rss: 30Mb L: 3/3 MS: 5 ChangeByte-InsertByte-ChangeBit
#62     NEW    cov: 4 ft: 4 corp: 3/5b lim: 4 exec/s: 0 rss: 30Mb L: 1/3 MS: 5 InsertByte-ShuffleBytes-CrossOv
#83     REDUCE cov: 4 ft: 4 corp: 3/4b lim: 4 exec/s: 0 rss: 31Mb L: 2/2 MS: 1 EraseBytes-
#1073   NEW    cov: 5 ft: 5 corp: 4/13b lim: 11 exec/s: 0 rss: 31Mb L: 9/9 MS: 5 ShuffleBytes-CopyPart-CrossOv
#1094   REDUCE cov: 5 ft: 5 corp: 4/10b lim: 11 exec/s: 0 rss: 31Mb L: 6/6 MS: 1 CrossOver-
#1268   REDUCE cov: 5 ft: 5 corp: 4/9b lim: 11 exec/s: 0 rss: 31Mb L: 5/5 MS: 4 ChangeByte-ChangeBit-ChangeByt
#1313   REDUCE cov: 5 ft: 5 corp: 4/7b lim: 11 exec/s: 0 rss: 31Mb L: 3/3 MS: 5 CopyPart-CrossOver-CopyPart-Sh
#1322   REDUCE cov: 6 ft: 6 corp: 5/9b lim: 11 exec/s: 0 rss: 31Mb L: 2/3 MS: 4 ChangeBit-InsertByte-CrossOver
```

  - 오류 signal 확인단계

```
==49770== ERROR: libFuzzer: deadly signal
    #0 0x55bdb97a6a81 in __sanitizer_print_stack_trace (/hom
    #1 0x55bdb9719318 in fuzzer::PrintStackTrace() (/home/da
    #2 0x55bdb96fed93 in fuzzer::Fuzzer::CrashCallback() (/h
    #3 0x7ff864a4251f  (/lib/x86_64-linux-gnu/libc.so.6+0x42
    #4 0x55bdb97d9f54 in LLVMFuzzerTestOneInput (/home/daehe
    #5 0x55bdb9700323 in fuzzer::Fuzzer::ExecuteCallback(uns
    #6 0x55bdb96ffa79 in fuzzer::Fuzzer::RunOne(unsigned cha
56de05ae49b1d1cd4)
    #7 0x55bdb9701269 in fuzzer::Fuzzer::MutateAndTestOne()
```

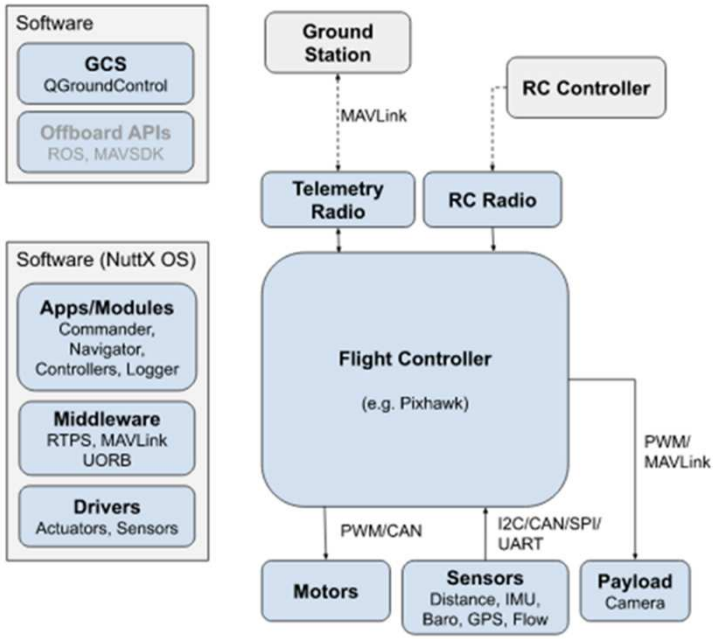
  - 결과 출력단계

```
SUMMARY: libFuzzer: deadly signal
MS: 1 InsertByte-; base unit: dbee5f8c7a5da845446e75b4a57
0x48,0x49,0x21,0x48,
HI!H
artifact_prefix='./'; Test unit written to ./crash-b13e87
Base64: SEkhSA==
```

PX4 드론 시스템

PX4 드론 구조

- PX4 Hardware
  - Flight Controller
  - Sensors
  - Motors
  - .etc
- PX4 Software
  - APP/Modules
  - Middleware
  - Drivers
  - .etc



[Flight Controller]

- 센서, GCS(지상 기지국) 등과 소통

[APP / Modules]

- PX4 Flight Stack에서 실행됨
- 주행에 필요한 Commander, Navigator, Controller, Logger 등

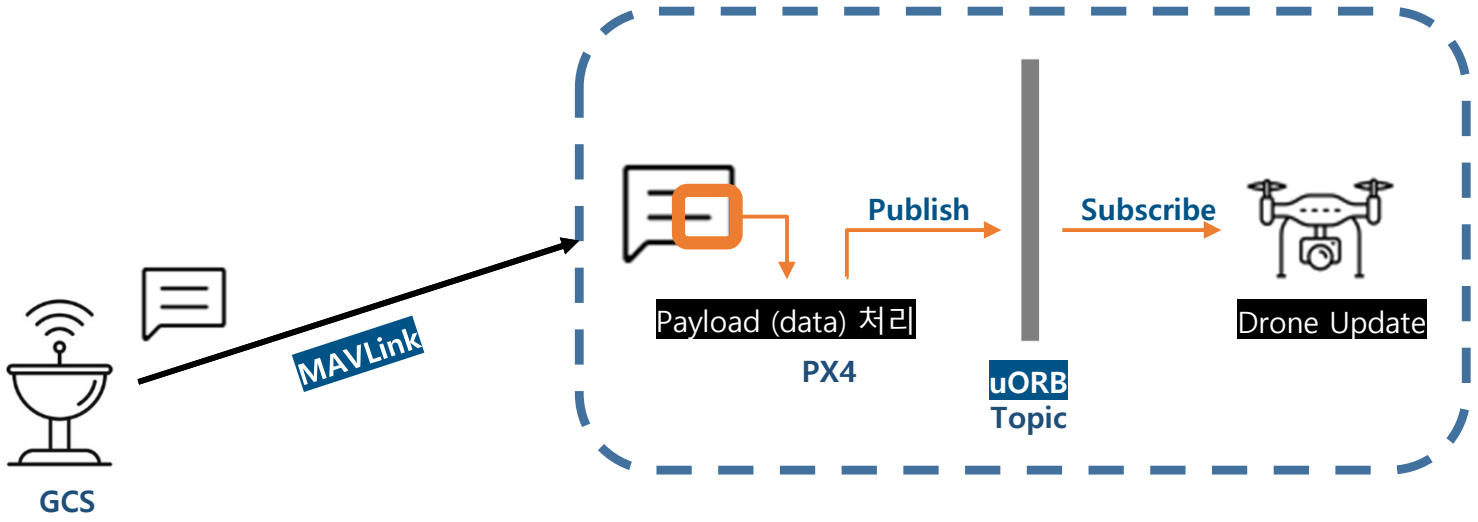
[Middleware]

- RTPS, MAVLink, uORB 등
- MAVLink : QGC와 PX4 드론 간의 통신에 사용
- uORB : PX4 드론 내부의 모듈들 간의 통신에 사용

드론 통신시스템

❖MAVLink (외부 통신 프로토콜)

- 지상 기지국(GCS)과 무인이동체 사이의 통신을 위한 경량화 메시징 프로토콜
  - 비행 모니터링/제어
  - 데이터 송·수신

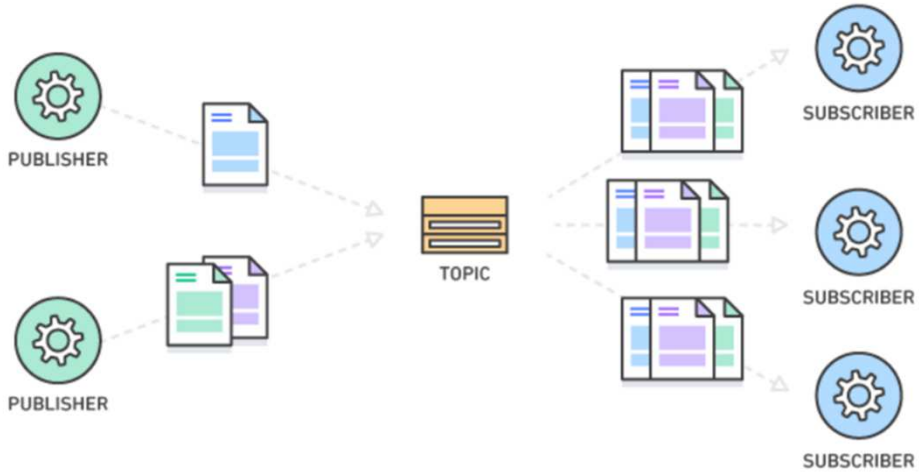


MAVLink와 uORB를 사용한 GCS & Drone 통신 과정

드론 통신시스템

❖uORB (내부 통신 프로토콜)

- PX4의 메시징 미들웨어
- 메시징 API인 Publish()와 Subscribe()를 구현



Publisher가 Topic에 Message를 publish

↓

advertise

↓

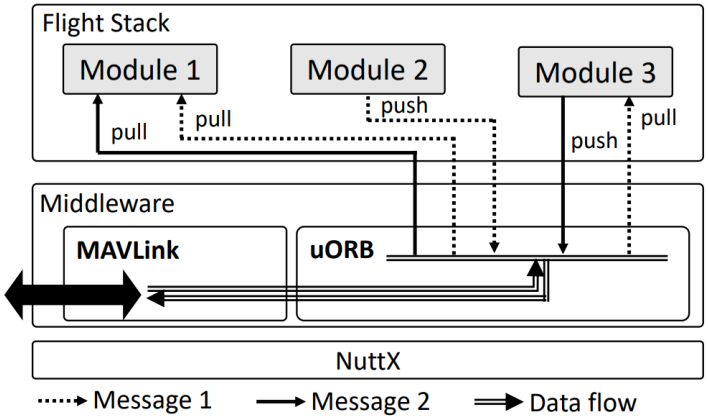
그 Topic을 subscribe 하고 있는 Subscriber들은
즉각적으로 Message를 받음

→ Queue 기반의 IPC와 유사한 개념이나,
직렬화 / 역직렬화는 지원하지 않음

드론 통신시스템

❖MAVLink와 uORB 통신

- PX4에서 내부 메시지를 처리할 때에는 uORB 프로토콜 사용
- 카메라 및 기타 하드웨어와 같은 오프 보드 통신은 MAVLink 프로토콜 활용

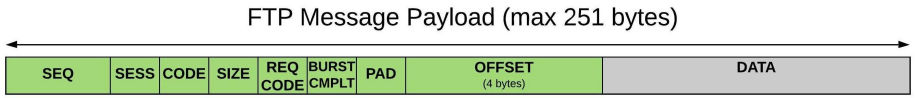


1. GCS가 PX4로 MAVLink 패킷 전송
2. PX4는 MAVLink의 페이로드만을 추출하여 uORB 의 topic에 publish
3. 업데이트된 topic 값을 subscribe하여 드론 상태 업데이트

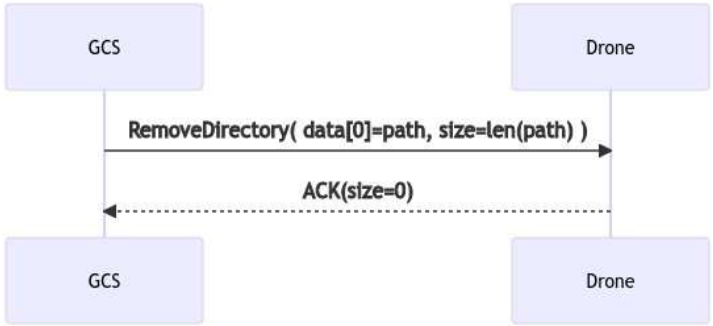
드론 통신시스템

❖상위 프로토콜들의 활용 (예: FTP)

- MAVLink를 통해 파일 전송을 지원하는 프로토콜
  - 파일 read, write, create, remove뿐만 아니라 디렉터리 listing, remove할 때 사용
  - PX4 : MAVLinkFTP 사용
    - ✓ PX4-Autopilot/src/modules/mavlink/mavlink_ftp.cpp



FTP 패킷 구조



(ex) 디렉터리 remove

드론 SW 취약점 예시

- CVE-2020-29664
  - DJI Mavic 2 취약점
  - 펌웨어 업데이트시 임의 코드가 실행 가능한 보안 취약점
  - 펌웨어 서명 과정에서 커맨드 인젝션 발생

```
int main(){
 /* ... */
                snprintf(cmd,0x100,"busybox find %s" ₩
                                "-name ₩"*.cfg.sig₩"",
                                "/data/upgrade/backup/");
                FILE *f = popen(cmd,"r");
                if (f)
                                fgets(cfg_name,0x100, f);
/* ... */
                int verify_ret = dji_verify_sig(₩
                                cfg_name,"/tmp/wm330_0000.xml",&DAT_0005c8a5);
/* ... */
}
```

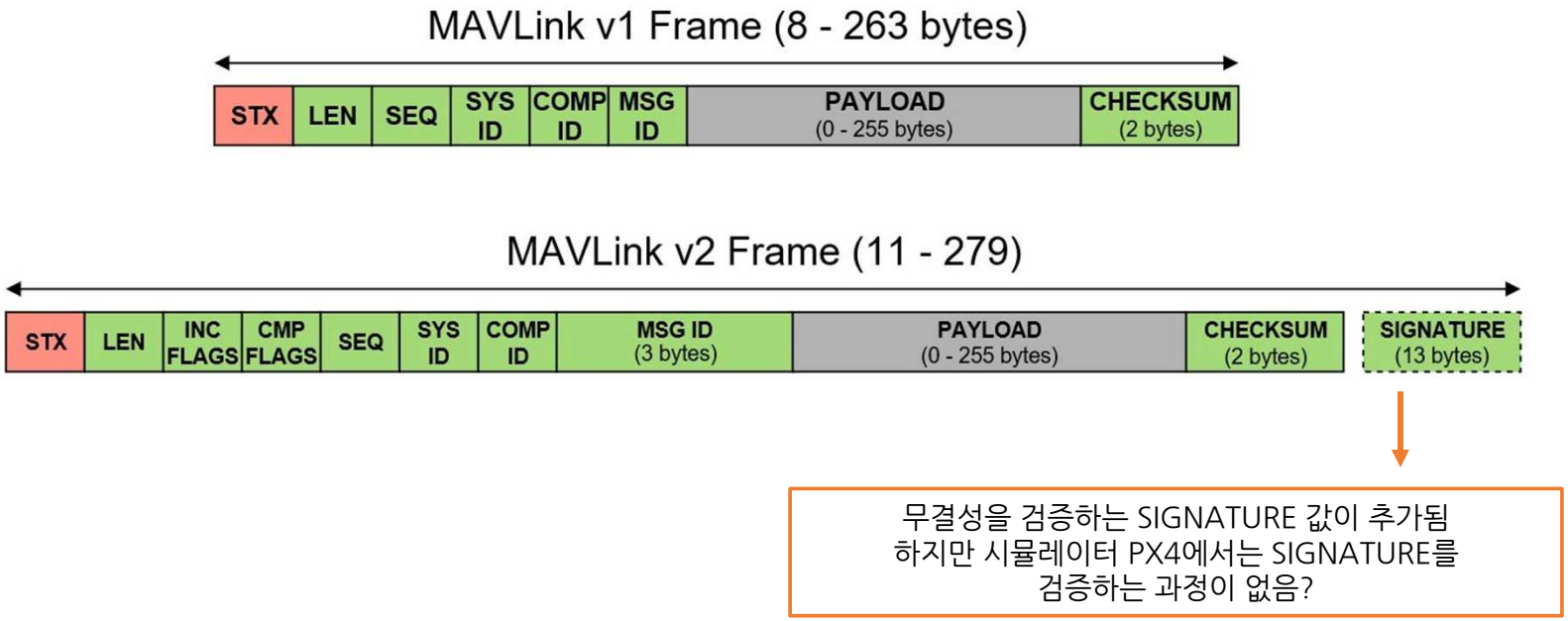
PX4 대상 libfuzzer 적용

PX4 대상 Fuzzing 적용

❖Libfuzzer 를 적용함

- PX4 는 오픈소스이므로 libfuzzer 포팅이 가능
- MAVLink, uORB, FastRTPS 등 PX4 외부/내부 통신 프로토콜 이용
  - ✓ Fuzzing 데이터 전달통로
- 통신 프로토콜 자체의 결함이 아닌, 해당 프로토콜 및 Pub/Sub IPC 를 통해서 데이터를 전달받는 데몬 (Task) 을 퍼징함
- MAVLink 를 Fuzzing 의 주 통로로 활용함
  - ✓ MAVLink 란?

MAVLink 프로토콜



Simulator 에서는 UDP 트랜스포트 이용

Libfuzzer 포팅

❖PX4 libfuzzer 적용

- PX4 빌드 과정을 변경 (cmake)
- 기본적으로 Fuzzer 빌드옵션이 존재함

```
if (CMAKE_BUILD_TYPE STREQUAL AddressSanitizer)
        message(STATUS "AddressSanitizer enabled")

        # environment variables
        #  ASAN_OPTIONS=check_initialization_order=1,detect_stack_use_after_return=1
        add_compile_options(
                -O1
                -g3

                -fsanitize=address

                -fno-omit-frame-pointer # Leave frame pointers. Allows the fast unwinder to function properly.
                -fno-common # Do not treat global variable in C as common variables (allows ASan to instrument them)
                -fno-optimize-sibling-calls # disable inlining and and tail call elimination for perfect stack trace
        )

        set(CMAKE_EXE_LINKER_FLAGS "${CMAKE_EXE_LINKER_FLAGS} -fsanitize=address" CACHE INTERNAL "" FORCE)
        set(CMAKE_SHARED_LINKER_FLAGS "${CMAKE_SHARED_LINKER_FLAGS} -fsanitize=address" CACHE INTERNAL "" FORCE)
        set(CMAKE_MODULE_LINKER_FLAGS "${CMAKE_MODULE_LINKER_FLAGS} -fsanitize=address" CACHE INTERNAL "" FORCE)

        function(sanitizer_fail_test_on_error test_name)
                set_tests_properties(${test_name} PROPERTIES FAIL_REGULAR_EXPRESSION "ERROR: AddressSanitizer")
                set_tests_properties(${test_name} PROPERTIES FAIL_REGULAR_EXPRESSION "ERROR: LeakSanitizer")
        endfunction(sanitizer_fail_test_on_error)

elseif (CMAKE_BUILD_TYPE STREQUAL MemorySanitizer)
        if ("${CMAKE_CXX_COMPILER_ID}" STREQUAL "GNU
                message(WARNING "MemorySanitizer mig
        else()
                message(STATUS "MemorySanitizer enab
        endif()

        add_compile_options(
                -O1
```

```
elseif (CMAKE_BUILD_TYPE STREQUAL FuzzTesting)
        message(STATUS "FuzzTesting enabled")

        add_compile_options(
                -O1
                -g3
                -fsanitize=fuzzer,address
                -fno-omit-frame-pointer # Leave frame pointers. Allows the fast unwinder to functi
                -fno-common # Do not treat global variable in C as common variables (allows ASan t
                -fno-optimize-sibling-calls # disable inlining and and tail call elimination for p
                -DFUZZTESTING
        )

        set(CMAKE_EXE_LINKER_FLAGS "${CMAKE_EXE_LINKER_FLAGS} -fsanitize=fuzzer,address $ENV{LIB_F
"" FORCE)
        set(CMAKE_SHARED_LINKER_FLAGS "${CMAKE_SHARED_LINKER_FLAGS} -fsanitize=fuzzer,address $ENV
ERNAL "" FORCE)
        set(CMAKE_MODULE_LINKER_FLAGS "${CMAKE_MODULE_LINKER_FLAGS} -fsanitize=fuzzer,address $ENV
ERNAL "" FORCE)
```

```
FROM ubuntu:20.04

RUN sed -i "s/http:\/\/archive.ubuntu.com/http:\/\/kr.archive.ubuntu.com/g"
/etc/apt/sources.list

# install related libraries
ARG DEBIAN_FRONTEND=noninteractive
ENV TZ=Asia/Seoul
RUN apt update && apt-get install -y build-essential clang gcc g++ bsdmainutils git cmake
ant

# at root
WORKDIR /

# setup PX4 source
RUN git clone https://github.com/PX4/PX4-Autopilot.git
RUN apt-get install -y python3 python3-pip
RUN pip3 install kconfiglib jinja2
RUN pip3 install --user empy pyros-genmsg jsonschema packaging toml numpy future

WORKDIR /PX4-Autopilot
ENV CC clang
ENV CXX clang
ENV PX4_ASAN 1
ENV PX4_FUZZ 1

RUN make px4_sitl jmavsim

ADD init.sh /
```

PX4 libfuzzer stub 코딩

```
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *data, const size_t size)
{
    initialize_fake_px4_once();

    send_mavlink(data, size);

    return 0;
}

void initialize_fake_px4_once()
{
    static bool first_time = true;

    if (!first_time) {
        return;
    }

    first_time = false;

    px4::init_once();
    px4::init(0, nullptr, "px4");

    px4_daemon::Pxh pxh;
    pxh.process_line("uorb start", true);
    pxh.process_line("param load", true);
    pxh.process_line("dataman start", true);
    pxh.process_line("load_mon start", true);
```

```
struct sockaddr_in addr {};

addr.sin_family = AF_INET;

inet_pton(AF_INET, "0.0.0.0", &(addr.sin_addr));

addr.sin_port = htons(14540);

if (bind(socket_fd, reinterpret_cast<sockaddr *>(&addr), s
    PX4_ERR("bind error: %s", strerror(errno));
    close(socket_fd);
    return;
}

mavlink_message_t message {};
uint8_t buffer[MAVLINK_MAX_PACKET_LEN] {};

for (size_t i = 0; i < size; i += sizeof(message)) {

    const size_t copy_len = std::min(sizeof(message), size
    //printf("copy_len: %zu, %zu (%zu)\n", i, copy_len, si
    memcpy(reinterpret_cast<void *>(&message), data + i, c

    const ssize_t buffer_len = mavlink_msg_to_send_buffer(

    struct sockaddr_in dest_addr {};
    dest_addr.sin_family = AF_INET;

    inet_pton(AF_INET, "127.0.0.1", &dest_addr.sin_addr.s_
    dest_addr.sin_port = htons(14556);
```

- PX4-Autoilpot/platforms/posix/src/px4/common/main_fuzztesting.cpp

- fake 초기화 작업 이후 mavlink 페이로드를 생성하여 수신 모듈에 전달
- pxh.process_line("...") 에서 초기 데몬들 실행
- Localhost:14540에 UDP로 libfuzzer 데이터를 파싱 후 패킷을 보냄

Fuzzing 패킷 포맷

Mutation 휴리스틱을 위해 Packet 구조 파악 필요

- src/modules/mavlink/mavlink_receiver.cpp
  - handle_message 함수

void MavlinkReceiver::handle_message(mavlink_message_t *msg)

_mavlink_message Struct Reference

#include <mavlink_types.h>

Public Attributes

uint16_t	checksum	
uint8_t	compid	ID of the message sender component. More...
uint8_t	len	Length of payload. More...
uint8_t	magic	sent at end of packet More...
uint8_t	msgid	ID of message in payload. More...
uint64_t	payload64 [(MAVLINK_MAX_PAYLOAD_LEN+MAVLINK_NUM_CHECKSUM_BYTES+7)/8]	
uint8_t	seq	Sequence of packet. More...
uint8_t	sysid	ID of message sender system/aircraft. More...

Fuzzing 패킷 포맷

❖Libfuzzer 포팅관련 코드분석

- Mavlink_msg_to_send_buffer
  - ✓ MAVLink 패킷을 전달하는 매크로
  - ✓ 시뮬레이터에서는 UDP 이용
  - ✓ 체크섬 계산

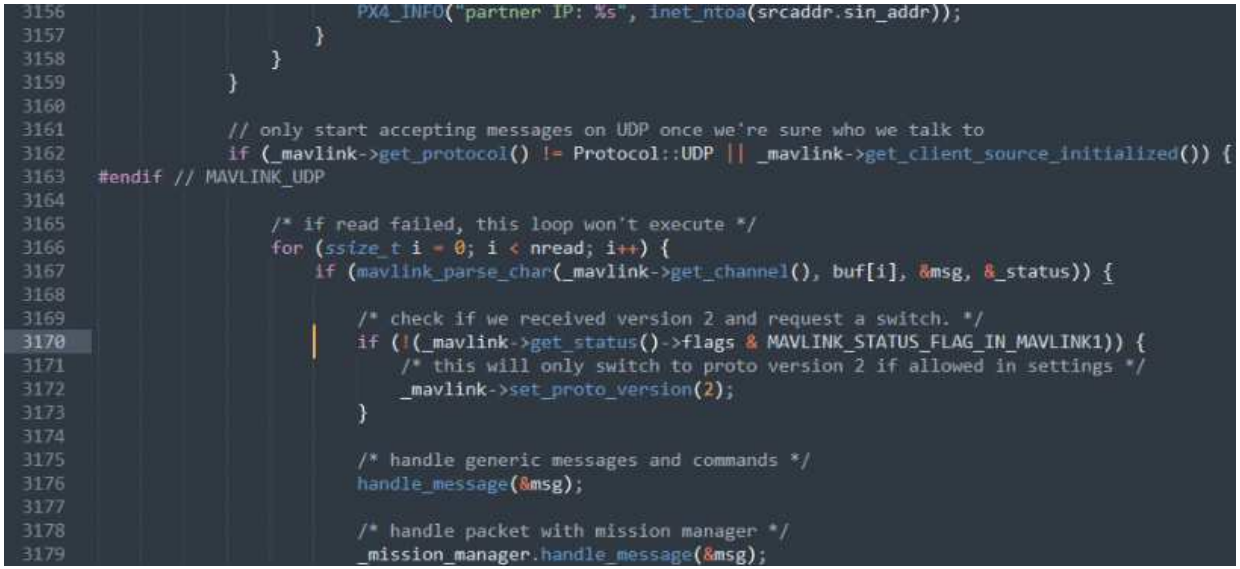
```
00363 MAVLINK_HELPER uint16_t mavlink_msg_to_send_buffer(uint8_t *buf, const mavlink_message_t *msg)
00364 {
00365         uint8_t signature_len, header_len;
00366         uint8_t *ck;
00367
00368         if (msg->magic == MAVLINK_STX_MAVLINK1) {
00369                 signature_len = 0;
00370                 header_len = MAVLINK_CORE_HEADER_MAVLINK1_LEN;
00371                 buf[0] = msg->magic;
00372                 buf[1] = msg->len;
00373                 buf[2] = msg->seq;
00374                 buf[3] = msg->sysid;
00375                 buf[4] = msg->compid;
00376                 buf[5] = msg->msgid & 0xFF;
00377                 memcpy(&buf[6], _MAV_PAYLOAD(msg), msg->len);
00378                 ck = buf + header_len + 1 + (uint16_t)msg->len;
00379         } else {
00380                 header_len = MAVLINK_CORE_HEADER_LEN;
00381                 buf[0] = msg->magic;
00382                 buf[1] = msg->len;
00383                 buf[2] = msg->incompat_flags;
00384                 buf[3] = msg->compat_flags;
00385                 buf[4] = msg->seq;
00386                 buf[5] = msg->sysid;
00387                 buf[6] = msg->compid;
00388                 buf[7] = msg->msgid & 0xFF;
00389                 buf[8] = (msg->msgid >> 8) & 0xFF;
00390                 buf[9] = (msg->msgid >> 16) & 0xFF;
00391                 memcpy(&buf[10], _MAV_PAYLOAD(msg), msg->len);
00392                 ck = buf + header_len + 1 + (uint16_t)msg->len;
00393                 signature_len = (msg->incompat_flags & MAVLINK_IFLAG_SIGNED)?MAVLINK_SIGNATURE_BLOCK_LEN:0;
00394         }
00395         ck[0] = (uint8_t)(msg->checksum & 0xFF);
00396         ck[1] = (uint8_t)(msg->checksum >> 8);
00397         if (signature_len > 0) {
00398                 memcpy(&ck[2], msg->signature, signature_len);
00399         }
00400
00401         return header_len + 1 + 2 + (uint16_t)msg->len + (uint16_t)signature_len;
00402 }
```

PX4 소스코드 분석

❖MAVLink 수신 모듈 분석

- UDP 로 전달받은 MAVLink 패킷파싱
- handle_message 함수 호출

void MavlinkReceiver::run()



```
3156                        PX4_INFO("partner IP: %s", inet_ntoa(srcaddr.sin_addr));
3157                    }
3158                }
3159            }
3160
3161            // only start accepting messages on UDP once we're sure who we talk to
3162            if (_mavlink->get_protocol() != Protocol::UDP || _mavlink->get_client_source_initialized()) {
3163    #endif // MAVLINK_UDP
3164
3165                /* if read failed, this loop won't execute */
3166                for (ssize_t i = 0; i < nread; i++) {
3167                    if (mavlink_parse_char(_mavlink->get_channel(), buf[i], &msg, &_status)) {
3168
3169                        /* check if we received version 2 and request a switch. */
3170                        if (!(_mavlink->get_status()->flags & MAVLINK_STATUS_FLAG_IN_MAVLINK1)) {
3171                            /* this will only switch to proto version 2 if allowed in settings */
3172                            _mavlink->set_proto_version(2);
3173                        }
3174
3175                        /* handle generic messages and commands */
3176                        handle_message(&msg);
3177
3178                        /* handle packet with mission manager */
3179                        _mission_manager.handle_message(&msg);
```

PX4 소스코드 분석

MAVLink 메시지 파싱의 시작: handle_message 함수

```
void
MavlinkReceiver::handle_message(mavlink_message_t *msg)
{
	switch (msg->msgid) {
	case MAVLINK_MSG_ID_COMMAND_LONG:
		handle_message_command_long(msg);
		break;

	case MAVLINK_MSG_ID_COMMAND_INT:
		handle_message_command_int(msg);
		break;
```

- msgid 별 처리 함수
- handle_message_command_long(msg) & handle_message_command_int(msg) …

PX4 소스코드 분석

handle_message 함수 이후의 흐름 -> uORB 의 Topic 에 publish

```
516
517         uint16_t message_id = (uint16_t)roundf(vehicle_command.param1);
518         result = handle_request_message_command(message_id,
519                                 vehicle_command.param2, vehicle_command.param3, vehicle_command.param4,
520                                 vehicle_command.param5, vehicle_command.param6, vehicle_command.param7)
521
522     } else if (cmd_mavlink.command == MAV_CMD_SET_CAMERA_ZOOM) {
523         struct actuator_controls_s actuator_controls = {};
524         actuator_controls.timestamp = hrt_absolute_time();
525
526         for (size_t i = 0; i < 8; i++) {
527             actuator_controls.control[i] = NAN;
528         }
529
530         switch ((int)(cmd_mavlink.param1 + 0.5f)) {
531         case vehicle_command_s::VEHICLE_CAMERA_ZOOM_TYPE_RANGE:
532             actuator_controls.control[actuator_controls_s::INDEX_CAMERA_ZOOM] = cmd_mavlink.param2
533             break;
534
535         case vehicle_command_s::VEHICLE_CAMERA_ZOOM_TYPE_STEP:
536         case vehicle_command_s::VEHICLE_CAMERA_ZOOM_TYPE_CONTINUOUS:
537         case vehicle_command_s::VEHICLE_CAMERA_ZOOM_TYPE_FOCAL_LENGTH:
538         default:
539             send_ack = false;
540         }
541
542         _actuator_controls_pubs[actuator_controls_s::GROUP_INDEX_GIMBA ].publish(actuator_controls)
543
544     } else if (cmd_mavlink.command == MAV_CMD_INJECT_FAILURE) {
545         if (_mavlink_failure_injection_enabled()) {
546             _cmd_pub.publish(vehicle_command);
547             send_ack = false;
```

- handle_message_command_both 함수에서 publish -> 다른 노드가 subscribe -> IPC 형성

- Topic 이 사전에 subscribe 가 안된경우 현재의 퍼징방식에서 커버리지 도달 불가
  -> 정적 분석의 필요성

PX4 소스코드 분석

uORB 를 통해 다양한 Task 들이 이후 데이터를 전달받음 -> Fuzzing 시 데이터 흐름파악 중요

src/modules/mavlink/mavlink_receiver.h (Publisher)

```
290         // ORB publications
291         uORB::Publication<actuator_controls_s>              _actuator_controls_pubs[4] {ORB_ID(actuator_controls_0),
292         uORB::Publication<airspeed_s>                       _airspeed_pub{ORB_ID(airspeed)};
293         uORB::Publication<battery_status_s>                 _battery_pub{ORB_ID(battery_status)};
294         uORB::Publication<camera_status_s>                  _camera_status_pub{ORB_ID(camera_status)};
295         uORB::Publication<cellular_status_s>                _cellular_status_pub{ORB_ID(cellular_status)};
296         uORB::Publication<collision_report_s>               _collision_report_pub{ORB_ID(collision_report)};
297         uORB::Publication<differential_pressure_s>          _differential_pressure_pub{ORB_ID(differential_pressure)}
298         uORB::Publication<follow_target_s>                  _follow_target_pub{ORB_ID(follow_target)};
299         uORB::Publication<gimbal_manager_set_attitude_s>    _gimbal_manager_set_attitude_pub{ORB_ID(gimbal_manager_se
300         uORB::Publication<gimbal_manager_set_manual_control_s> _gimbal_manager_set_manual_control_pub{ORB_ID(gimbal_mana
301         uORB::Publication<gimbal_device_information_s>      _gimbal_device_information_pub{ORB_ID(gimbal_device_infor
302         uORB::Publication<gimbal_device_attitude_status_s>  _gimbal_device_attitude_status_pub{ORB_ID(gimbal_device_a
303         uORB::Publication<irlock_report_s>                  _irlock_report_pub{ORB_ID(irlock_report)};
304         uORB::Publication<landing_target_pose_s>            _landing_target_pose_pub{ORB_ID(landing_target_pose)};
305         uORB::Publication<log_message_s>                    _log_message_pub{ORB_ID(log_message)};
306         uORB::Publication<mavlink_tunnel_s>                 _mavlink_tunnel_pub{ORB_ID(mavlink_tunnel)};
307         uORB::Publication<obstacle_distance_s>              _obstacle_distance_pub{ORB_ID(obstacle_distance)};
308         uORB::Publication<offboard_control_mode_s>          _offboard_control_mode_pub{ORB_ID(offboard_control_mode)}
309         uORB::Publication<onboard_computer_status_s>        _onboard_computer_status_pub{ORB_ID(onboard_computer_stat
310         uORB::Publication<generator_status_s>               _generator_status_pub{ORB_ID(generator_status)};
311         uORB::Publication<optical_flow_s>                   _flow_pub{ORB_ID(optical_flow)};
312         uORB::Publication<vehicle_attitude_s>               _attitude_pub{ORB_ID(vehicle_attitude)};
313         uORB::Publication<vehicle_attitude_setpoint_s>      _att_sp_pub{ORB_ID(vehicle_attitude_setpoint)};
314         uORB::Publication<vehicle_attitude_setpoint_s>      _mc_virtual_att_sp_pub{ORB_ID(mc_virtual_attitude_setpoin
```

PX4 소스코드 분석

orb_subscribe를 이용해 MAVLink로부터 데이터를 받는 코드들을 분석 (Subscriber)
-> Fuzzing Vector 관련 Data 흐름 파악!

```
D:\stage\PX4-Autopilot-master\PX4-Autopilot-master\src\drivers\telemetry\frsky_telemetry\frsky_telemetry.cpp:
387       set_uart_invert(uart, frsky_state == SPORT_SINGLE_WIRE_INVERT);
388
389:      /* Subscribe to topics */
390       if (!sPort_init()) {
391           PX4_ERR("could not allocate memory");
...
397       float filtered_alt = NAN;
398       float last_baro_alt = 0.f;
399:      int airdata_sub = orb_subscribe(ORB_ID(vehicle_air_data));
400
401       uint32_t lastBATV_ms = 0;
...
648       int iteration = 0;
649
650:      /* Subscribe to topics */
651       if (!frsky_init()) {
652           PX4_ERR("could not allocate memory");

D:\stage\PX4-Autopilot-master\PX4-Autopilot-master\src\drivers\telemetry\hott\messages.cpp:
 75  init_sub_messages(void)
 7   {
 7:    _battery_sub = orb_subscribe(ORB_ID(battery_status));
 7:    _gps_sub = orb_subscribe(ORB_ID(vehicle_gps_position));
 7:    _home_sub = orb_subscribe(ORB_ID(home_position));
 8:    _airdata_sub = orb_subscribe(ORB_ID(vehicle_air_data));
 8:    _airspeed_sub = orb_subscribe(ORB_ID(airspeed));
```

Libfuzzer stub 포팅

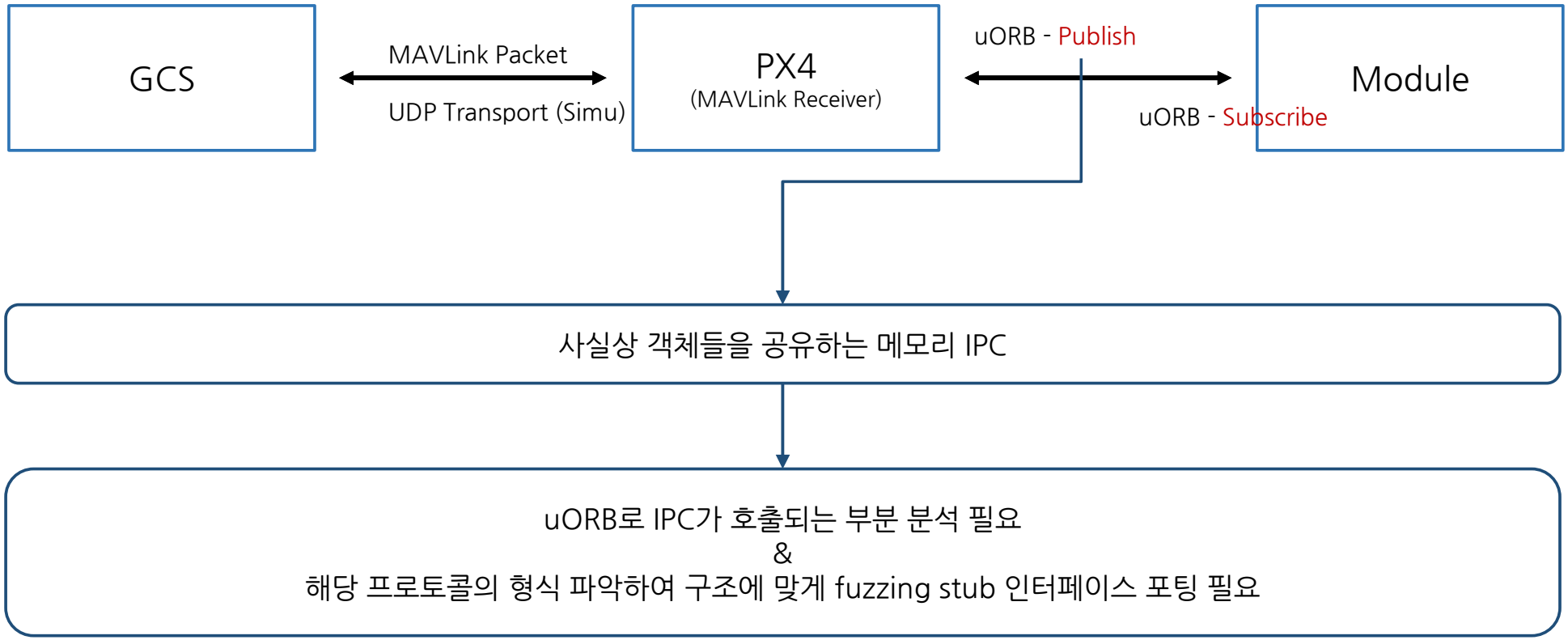
❖PX4 Fake 초기화 코드

- Libfuzzer stub 과의 연동만을 위한 최소한의 초기화
- 데이터 흐름 및 네트워크 관련 데몬 실행

```
void initialize_fake_px4_once()
{
        static bool first_time = true;

        if (!first_time) {
                return;
        }

        first_time = false;

        px4::init_once();
        px4::init(0, nullptr, "px4");

        px4_daemon::Pxh pxh;
        pxh.process_line("uorb start", true);
        pxh.process_line("param load", true);
        pxh.process_line("dataman start", true);
        pxh.process_line("load_mon start", true);
        pxh.process_line("battery_simulator start", true);
        pxh.process_line("tone_alarm start", true);
        pxh.process_line("rc_update start", true);
        pxh.process_line("sensors start", true);
        pxh.process_line("commander start", true);
        pxh.process_line("navigator start", true);
        pxh.process_line("ekf2 start", true);
        pxh.process_line("mc_att_control start", true);
        pxh.process_line("mc_pos_control start", true);
        pxh.process_line("land_detector start multicopter", true);
        pxh.process_line("logger start", true);
        pxh.process_line("mavlink start -x -o 14540 -r 4000000", true);
        pxh.process_line("mavlink boot_complete", true);
}
```

platforms/posix/src/px4/common/main_fuzztesting.cpp

PX4 Libfuzzer 포팅 방향



PX4 Fuzzing 분석결과

❖분석결과 정리

- MAVLink: 다양한 링크계층 프로토콜을 기반으로 MAVLink Receiver 모듈까지 각종 명령/데이터 에 대한 Payload 데이터를 전송 (checksum 존재)
- Receiver: 다양한 Payload 데이터들을 파싱한 후, 핸들러들을 호출
- 핸들러 함수: 각 Payload를 분석하여 알맞은 ORB Publish를 수행
- ORB Subscribe: 병렬적으로 실행되던 기존의 Application들이 ORB Topic Subscribe를 통해 Payload 데이터를 수신하여 처리

- PX4 내부에서 다양한 Application 실행이 가능함
- ORB는 네트워크 프로토콜이 아닌 공유 객체 메모리 IPC를 위한 라이브러리에 가까움
- 별도의 ORB 프로토콜 구조를 활용하는 퍼징 알고리즘 개발은 무의미한 것으로 판단됨

PX4 Application

❖PX4 어플리케이션 (Task)

기본 PX4 libfuzzer 바이너리에 포함되지 않은 Appication

init_app_map 분석 ->
uORB 로 호출 가능할 것으로 예상되는 Application 리스팅 ->
ORB Subscribe 하여 추가 fuzzing

```
pxh.process_line("controllib_test start", true);
pxh.process_line("rc_tests start", true);
pxh.process_line("uorb_tests start", true);
pxh.process_line("wqueue_test start", true);
pxh.process_line("camera_trigger start", true);
pxh.process_line("gps start", true);
pxh.process_line("pwm_out_sim start", true);
pxh.process_line("rpm_simulator start", true);
pxh.process_line("tone_alarm start", true);
pxh.process_line("airship_att_control start", true);
pxh.process_line("airspeed_selector start", true);
pxh.process_line("attitude_estimator_q start", true);
pxh.process_line("camera_feedback start", true);
pxh.process_line("commander start", true);
pxh.process_line("commander_tests start", true);
pxh.process_line("control_allocator start", true);
pxh.process_line("dataman start", true);
pxh.process_line("ekf2 start", true);
pxh.process_line("send_event start", true);
pxh.process_line("flight_mode_manager start", true);
pxh.process_line("fw_att_control start", true);
pxh.process_line("fw_autotune_attitude_control start", true);
pxh.process_line("fw_pos_control_l1 start", true);
pxh.process_line("gimbal start", true);
```

```
pxh.process_line("gyro_calibration start", true);
pxh.process_line("gyro_fft start", true);
pxh.process_line("land_detector start", true);
pxh.process_line("landing_target_estimator start", true);
pxh.process_line("load_mon start", true);
pxh.process_line("local_position_estimator start", true);
pxh.process_line("logger start", true);
pxh.process_line("mag_bias_estimator start", true);
pxh.process_line("manual_control start", true);
pxh.process_line("mavlink start", true);
pxh.process_line("mavlink_tests start", true);
pxh.process_line("mc_att_control start", true);
pxh.process_line("mc_autotune_attitude_control start", true);
pxh.process_line("mc_hover_thrust_estimator start", true);
pxh.process_line("mc_pos_control start", true);
pxh.process_line("mc_rate_control start", true);
pxh.process_line("navigator start", true);
pxh.process_line("rc_update start", true);
pxh.process_line("replay start", true);
pxh.process_line("rover_pos_control start", true);
pxh.process_line("sensors start", true);
pxh.process_line("simulator start", true);
pxh.process_line("battery_simulator start", true);
pxh.process_line("sensor_baro_sim start", true);
pxh.process_line("sensor_gps_sim start", true);
pxh.process_line("sensor_mag_sim start", true);
pxh.process_line("temperature_compensation start", true);
```

```
pxh.process_line("actuator_test start", true);
pxh.process_line("dyn start", true);
pxh.process_line("failure start", true);
pxh.process_line("led_control start", true);
pxh.process_line("mixer start", true);
pxh.process_line("motor_test start", true);
pxh.process_line("param start", true);
pxh.process_line("perf start", true);
pxh.process_line("pwm start", true);
pxh.process_line("sd_bench start", true);
pxh.process_line("shutdown start", true);
pxh.process_line("system_time start", true);
pxh.process_line("tests start", true);
pxh.process_line("hrt_test start", true);
pxh.process_line("listener start", true);
pxh.process_line("tune_control start", true);
pxh.process_line("uorb start", true);
pxh.process_line("ver start", true);
pxh.process_line("work_queue start", true);
pxh.process_line("fake_gps start", true);
pxh.process_line("fake_imu start", true);
pxh.process_line("fake_magnetometer start", true);
pxh.process_line("ex_fixedwing_control start", true);
pxh.process_line("hello start", true);
```

PX4 – libfuzzer 포팅실습

❖1단계: PX4 빌드하기

- 준비물
  - ✓ Ubuntu 20.04 ~ 등 최신 리눅스 환경
    - Ubuntu 18.04
    - Ubuntu 20.04
    - Ubuntu 22.04
    - …
  - ✓ Git 리포지토리 도구
    - apt install git
  - ✓ 빌드에 필요한 라이브러리들
    - C/C++ 관련: clang, gcc, cmake, build-essential…
    - Python 관련: python3-pip, python3, python3-dev …
  - ✓ Pip 패키지들
    - pip3 install --user empy
    - pip3 install --user jsonschema
    - pip3 install --user pyros-genmsg
    - …

PX4 – libfuzzer 포팅실습

❖2단계: PX4 다운로드

- https://github.com/PX4/PX4-Autopilot.git

```
daehee@none:~/drone$ git clone https://github.com/PX4/PX4-Autopilot.git
Cloning into 'PX4-Autopilot'...
remote: Enumerating objects: 440855, done.
remote: Counting objects: 100% (1027/1027), done.
remote: Compressing objects: 100% (661/661), done.
remote: Total 440855 (delta 602), reused 600 (delta 360), pack-reused 439828
Receiving objects: 100% (440855/440855), 208.99 MiB | 10.86 MiB/s, done.
Resolving deltas: 100% (325589/325589), done.
daehee@none:~/drone$
```

PX4 – libfuzzer 포팅실습 (기본)

❖3단계: 기본 fuzzer 빌드

- export CC=clang
- export CXX=clang
- export PX4_FUZZ=1
- make px4_sitl jmavsim
  - ✓ 버전차이관련 여러 오류 예상

```
[  4%] Building C object platforms/common/work_queue/CMakeFiles/work_queue.dir/sq_addafter.c.o
[  4%] Performing configure step for 'libmicroxrceddsclient_project'
[  4%] Linking CXX static library libmode_util.a
[  4%] Building C object platforms/common/work_queue/CMakeFiles/work_queue.dir/sq_addlast.c.o
[  4%] Building C object platforms/common/work_queue/CMakeFiles/work_queue.dir/sq_remfirst.c.o
[  4%] Built target mode_util
[  4%] Generating px4 event json file from source
[  4%] Building C object platforms/common/work_queue/CMakeFiles/work_queue.dir/work_cancel.c.o
[  5%] Building C object platforms/common/work_queue/CMakeFiles/work_queue.dir/work_lock.c.o
[  5%] Building C object platforms/common/work_queue/CMakeFiles/work_queue.dir/work_queue.c.o
[  5%] Building C object platforms/common/work_queue/CMakeFiles/work_queue.dir/work_thread.c.o
CMake Error at /home/daehee/drone/PX4-Autopilot/build/px4_sitl_default/src/modules/uxrce_dds_clien
ject-stamp/libmicroxrceddsclient_project-configure-FuzzTesting.cmake:37 (message):
  Command failed: 1

   '/usr/bin/cmake' '-GUnix Makefiles' '-C/home/daehee/drone/PX4-Autopilot/build/px4_sitl_default/
p/libmicroxrceddsclient_project-cache-FuzzTesting.cmake' '/home/daehee/drone/PX4-Autopilot/src/mod
E-DDS-Client'

  See also

    /home/daehee/drone/PX4-Autopilot/build/px4_sitl_default/src/modules/uxrce_dds_client/src/libmi
libmicroxrceddsclient_project-configure-*.log


-- stdout output is:
loading initial cache file /home/daehee/drone/PX4-Autopilot/build/px4_sitl_default/src/modules/uxr
dsclient_project-cache-FuzzTesting.cmake
-- The C compiler identification is Clang 14.0.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - failed
-- Check for working C compiler: /usr/bin/clang
-- Check for working C compiler: /usr/bin/clang - broken
-- Configuring incomplete, errors occurred!
```

```
daehee@none:~/drone/PX4-Autopilot$ make px4_sitl jmavsim
-- PX4 version: v1.14.0-beta2-196-gebe152fc22 (1.14.0)
-- Found PythonInterp: /usr/bin/python3 (found suitable version "3.10.6"
-- PX4 config file: /home/daehee/drone/PX4-Autopilot/boards/px4/sitl/def
-- PLATFORM posix
-- ROMFSROOT px4fmu_common
-- ROOTFSDIR .
-- TESTING y
-- ETHERNET y
-- PX4 config: px4_sitl_default
-- PX4 platform: posix
-- PX4 lockstep: disabled
-- The CXX compiler identification is Clang 14.0.0
-- The C compiler identification is Clang 14.0.0
-- The ASM compiler identification is Clang with GNU-like command-line
-- Found assembler: /usr/bin/clang
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/clang - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/clang - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- cmake build type: FuzzTesting
-- FuzzTesting enabled
-- Could NOT find gz-transport (missing: gz-transport_DIR)
```

PX4 – libfuzzer 포팅실습 (심화)

❖3단계: 빌드 시스템 수정 (libfuzzer + ASAN)

- PX4-Autopilot/cmake/sanitizers.cmake 업데이트
  - ✓ PX4_FUZZ 옵션에 address 컴파일러 옵션추가

```
elseif (CMAKE_BUILD_TYPE STREQUAL FuzzTesting)
        message(STATUS "FuzzTesting enabled")

        add_compile_options(
                -O1
                -g3
                -fsanitize=fuzzer,address
                -fno-omit-frame-pointer # Leave frame pointers. Allows the fast unwinder to function properly.
                -fno-common # Do not treat global variable in C as common variables (allows ASan to instrument th
                -fno-optimize-sibling-calls # disable inlining and and tail call elimination for perfect stack tr
                -DFUZZTESTING
        )

        set(CMAKE_EXE_LINKER_FLAGS "${CMAKE_EXE_LINKER_FLAGS} -fsanitize=fuzzer,address $ENV{LIB_FUZZING_ENGINE}"
"" FORCE)
        set(CMAKE_SHARED_LINKER_FLAGS "${CMAKE_SHARED_LINKER_FLAGS} -fsanitize=fuzzer,address $ENV{LIB_FUZZING_EN
ERNAL "" FORCE)
        set(CMAKE_MODULE_LINKER_FLAGS "${CMAKE_MODULE_LINKER_FLAGS} -fsanitize=fuzzer,address $ENV{LIB_FUZZING_EN
ERNAL "" FORCE)

        function(sanitizer_fail_test_on_error test_name)
                # Not sure what to do here
        endfunction(sanitizer_fail_test_on_error)
else()

        function(sanitizer_fail_test_on_error test_name)
```

PX4 – libfuzzer 포팅실습

❖4단계: PX4 소스코드 수정 (Stub 코드 작성)

- PX4-Autopilot/platforms/posix/src/px4/common/main_fuzztesting.cpp
  - ✓ MAVLINK_STX_MAVLINK1 프로토콜 사용
  - ✓ Sysid = 255 고정
  - ✓ Compid = 255 고정
  - ✓ Broadcast ID 사용

```
        pxh.process_line("list_tasks start", true);
        pxh.process_line("list_files start", true);

        PX4_WARN("Daemons all set! start mavlink now.");
        pxh.process_line("mavlink start -x -o 14540 -r 4000000", true);
        pxh.process_line("mavlink boot_complete", true);

}

mavlink_message_t message {};
void send_mavlink(const uint8_t *data, const size_t size)
{
        for (size_t i = 0; i < size; i += sizeof(message)) {
                const size_t copy_len = std::min(sizeof(message), size - i);
                memcpy(reinterpret_cast<void *>(&message), data + i, copy_len);
                message.magic = MAVLINK_STX_MAVLINK1;
                message.seq++;
                message.sysid = 255;
                message.compid = 255;
                memcpy(fuzzing_buffer, &message, sizeof(message));
        }

}
```

PX4 – libfuzzer 포팅실습

❖5단계: PX4 libfuzzer 모드 빌드

- make px4_sitl jmavsim

```
-- PX4 version: v1.13.0-beta1-170-gd04a91a3ae
-- Found PythonInterp: /usr/bin/python3 (found suitable version "3.8.10
-- PX4 config file: /PX4-Autopilot/boards/px4/sitl/default.px4board
-- PLATFORM posix
-- ROMFSROOT px4fmu_common
-- TESTING y
-- ETHERNET y
-- PX4 config: px4_sitl_default
-- PX4 platform: posix
-- PX4 lockstep: disabled
-- cmake build type: FuzzTesting
-- The CXX compiler identification is Clang 10.0.0
-- The C compiler identification is Clang 10.0.0
-- The ASM compiler identification is Clang
-- Found assembler: /usr/bin/clang
-- Check for working CXX compiler: /usr/bin/clang
-- Check for working CXX compiler: /usr/bin/clang -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Check for working C compiler: /usr/bin/clang
-- Check for working C compiler: /usr/bin/clang -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Building for code coverage
-- FuzzTesting enabled
```

PX4 – libfuzzer 포팅실습

❖6단계: PX4 libfuzzer 실행

- ASAN_OPTIONS=detect_leaks=0:coverage=1
  - ✓ Address Sanitizer 옵션을 환경변수로 libfuzzer 에 전달 (LeakSAN 비활성화)
- /PX4-Autopilot/build/px4_sitl_default/bin/px4
  - ✓ 최종 빌드된 px4-libfuzzer 바이너리
- -workers=4
  - ✓ Libfuzzer 옵션 (4코어 병렬가동)
- -detect_leaks=0
  - ✓ Libfuzzer 옵션 LeakSAN 비활성화
- /out
  - ✓ Libfuzzer 옵션 (퍼징 데이터 저장 디렉토리 경로)

```
ASAN_OPTIONS=detect_leaks=0:coverage=1 /PX4-Autopilot/build/px4_sitl_default/bin/px4 -workers=4 -detect_leaks=0 /out
```

PX4 libfuzzer 분석

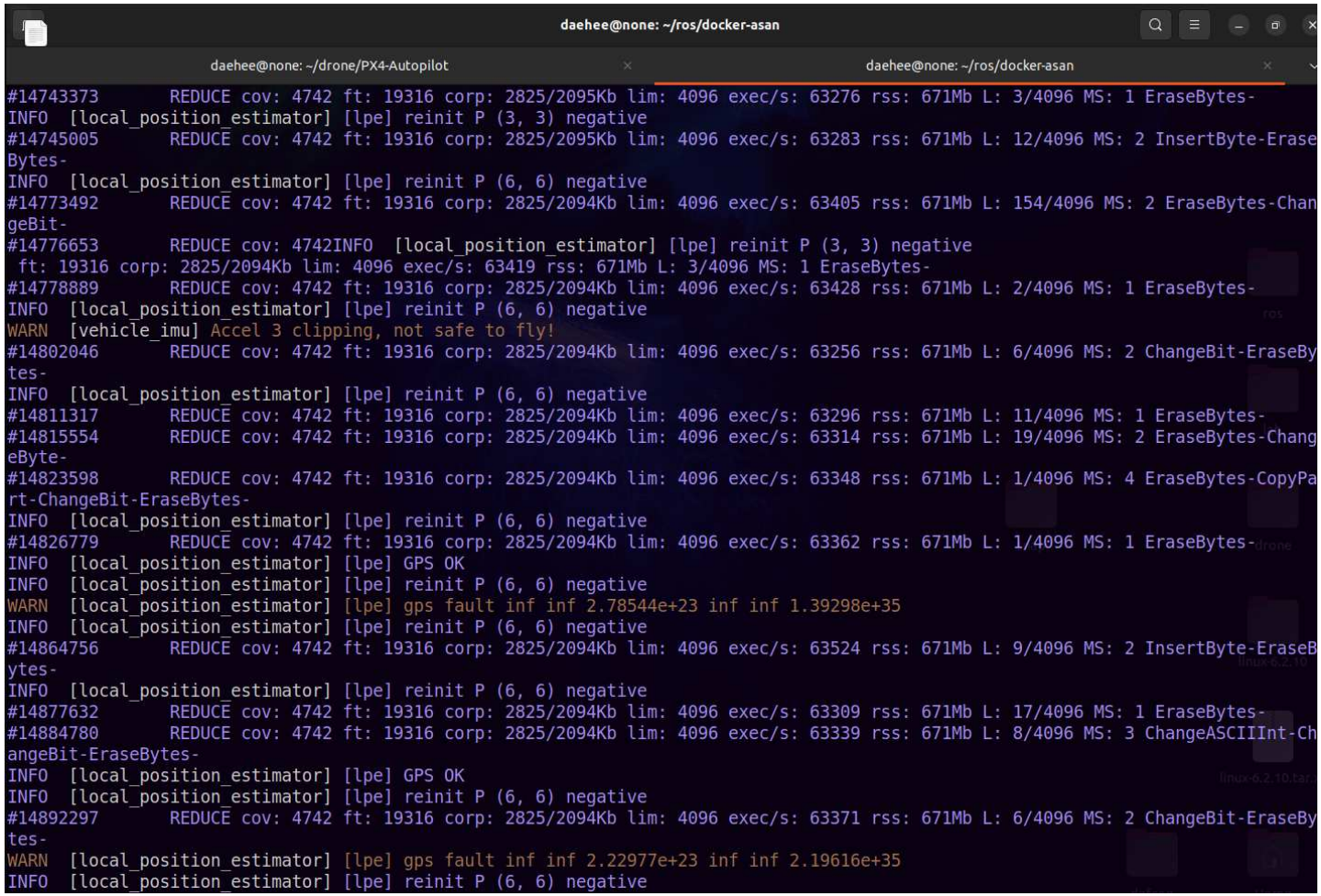
❖아래와 같이 ASAN instrumentation 적용 확인

```
pwndbg> disass send_mavlink
Dump of assembler code for function send_mavlink(unsigned char const*, unsigned long):
   0x00000000004d8070 <+0>:     push   rbp
   0x00000000004d8071 <+1>:     mov    rbp,rsp
   0x00000000004d8074 <+4>:     sub    rsp,0x330
   0x00000000004d807b <+11>:    mov    al,BYTE PTR [rip+0x62c4a3]        # 0xb04524
   0x00000000004d8081 <+17>:    add    al,0x1
   0x00000000004d8083 <+19>:    mov    BYTE PTR [rip+0x62c49b],al        # 0xb04524
   0x00000000004d8089 <+25>:    mov    rcx,rbp
   0x00000000004d808c <+28>:    mov    rdx,0xfffffffffffffd0
   0x00000000004d8093 <+35>:    mov    rdx,QWORD PTR fs:[rdx]
   0x00000000004d8097 <+39>:    cmp    rcx,rdx
   0x00000000004d809a <+42>:    mov    QWORD PTR [rbp-0x2b0],rdi
   0x00000000004d80a1 <+49>:    mov    QWORD PTR [rbp-0x2b8],rsi
   0x00000000004d80a8 <+56>:    mov    QWORD PTR [rbp-0x2c0],rcx
   0x00000000004d80af <+63>:    jae    0x4d80c7 <send_mavlink(unsigned char const*, unsigned long)+87>
   0x00000000004d80b5 <+69>:    mov    rax,0xfffffffffffffd0
   0x00000000004d80bc <+76>:    mov    rcx,QWORD PTR [rbp-0x2c0]
   0x00000000004d80c3 <+83>:    mov    QWORD PTR fs:[rax],rcx
   0x00000000004d80c7 <+87>:    xor    edx,edx
   0x00000000004d80c9 <+89>:    mov    rax,QWORD PTR [rbp-0x2b0]
   0x00000000004d80d0 <+96>:    mov    QWORD PTR [rbp-0x8],rax
   0x00000000004d80d4 <+100>:   mov    rcx,QWORD PTR [rbp-0x2b8]
   0x00000000004d80db <+107>:   mov    QWORD PTR [rbp-0x10],rcx
   0x00000000004d80df <+111>:   mov    esi,0x2
   0x00000000004d80e4 <+116>:   mov    edi,esi
   0x00000000004d80e6 <+118>:   call   0x42b830 <socket@plt>
   0x00000000004d80eb <+123>:   xor    edi,edi
   0x00000000004d80ed <+125>:   mov    DWORD PTR [rbp-0x14],eax
   0x00000000004d80f0 <+128>:   mov    eax,DWORD PTR [rbp-0x14]
   0x00000000004d80f3 <+131>:   mov    esi,eax
   0x00000000004d80f5 <+133>:   mov    DWORD PTR [rbp-0x2c4],eax
   0x00000000004d80fb <+139>:   call   0x4811a0 <__sanitizer_cov_trace_const_cmp4>
   0x00000000004d8100 <+144>:   mov    eax,DWORD PTR [rbp-0x2c4]
   0x00000000004d8106 <+150>:   cmp    eax,0x0
   0x00000000004d8109 <+153>:   jge    0x4d8150 <send_mavlink(unsigned char const*, unsigned long)+224>
```

PX4 – libfuzzer 포팅실습

❖실행결과

- Crash 발견시 퍼저를 실행한 폴더에 crash-[해시] PoC 파일이 생성됨



결론

❖PX4 open source에 libfuzzer 를 적용시 연구포인트

- 퍼징 데이터 전달 수단 분석필요
  - ✓ uORB, MAVLink 등 통신 프로토콜 흐름 파악필요

❖주요 연구 사항

- Coverage 를 넓히기 위해 PX4 내부 Task 들이 많이 동작하게 해야함
- Sanitizer 적용을 위해 빌드시스템 수정이 필요함
- 네트워크 Packet 을 통해 퍼징데이터 전달시 성능이 매우 하락함

❖한계점

- Closed Source 의 경우 적용불가
- 완전 자동화는 어려우며, 정적분석 병행이 많이 요구됨


Q
A