PX4 Autopilot에 대한 보안 취약점 분석*

이명진○ 유승근 차가민 조진성
경희대학교 컴퓨터공학과
lumy0726@khu.ac.kr, yuu3730@khu.ac.kr, gmcha0323@khu.ac.kr, chojs@khu.ac.kr

Vulnerabilities Analysis on PX4 Autopilot

Myeongjin Lee○ Seunggeun Yu Gamin Cha Jinsung Cho
Department of Computer Science and Engineering, KyungHee University

요 약

오픈소스(Open Source) 제작 드론의 경우 많은 사용자가 접근할 수 있으나 보안 취약점이 악용될 경우, 물리적 및 정보적 손실을 초래할 수 있다. 급변하는 드론 발전의 흐름에 맞추어 드론 사용의 안정성을 향상하기 위해 보안 취약점에 대한 지속적인 개선이 필요하다. 본 연구에서는 드론 및 기타 자율주행 무인기에 사용되는 PX4 Autopilot을 대상으로 CodeQL을 이용한 정적 분석을 진행하며, 알려진 취약점과 정적 분석 결과에 대한 공격 재현을 수행한다.

1. 서 론

드론은 무인기(Unmanned Aerial Vehicle, UAV)로 통용되며, 조종사가 탑승하지 아니하고 원격 조종 또는 자율 비행이 가능한 항공기를 의미한다[1]. 전 세계 드론 시장 규모는 2022년에 243억 9,000만 달러를 기록하였고 2030년까지는 약 5,045억 달러까지 성장할 것으로 예상된다[2].

드론의 확산에 따라 무단 비행, 보안구역 침해 등 문제점이 증가하고 있다. 오픈소스 제작 드론의 경우 많은 사용자의 접근이 가능하여 보안 취약점이 악용될 가능성이 크다. 따라서 급변하는 드론 사용의 안전성을 향상하기 위해 보안 취약점에 대한 지속적인 개선이 필요하다.

본 논문은 드론 제어 오픈소스 프로그램 PX4 Autopilot을 대상으로 CodeQL을 이용한 정적 분석을 진행한다. 또한 알려진 취약점과 정적 분석 결과에 대한 공격 재현을 수행한다. 이를 통해 자율 비행 무인기 프로그래밍에서의 유의점을 파악한다.

2. 연구 배경

2.1 PX4 Autopilot

PX4 Autopilot은 자율주행 무인기의 비행 제어를 지원하는 오픈 소프트웨어로 NuttX ROS2 기반 등의 컴퓨터 및 MAVSDK API와 통합이 이루어지며 기타 하드웨어에 연결하기 위해 QGroundControl과 MAVLink protocol이 적용된다.

2.2 QGroundControl

QGroundControl은 MAVLink 프로토콜으로 통신을 하는 autopilot 기반 기기를 위해 비행 제어 및 기기 설정을 제공하는 지상제어국(Ground Control System, GCS)이다. Mission planning 기능과 비행 허용 구역을 설정하기 위해 geofence와 같은 기능을 제공한다.

2.3 MAVlink

MAVlink는 지상제어국, autopilot 및 기타 하드웨어를 위한 통신 프로토콜이다. Mission protocol 등의 서브 프로토콜으로 이루어져 있다. 송수신 메세지는 XML 파일에 정의되어 있다.

2.4 uORB

uORB(Micro Object Request Broker)는 PX4 Autopilot 내부의 오브젝트 통신을 위한 publish-subscribe 기반 미들웨어이다. PX4 Autopilot는 MAVLink에 대한 의존성을 방지하기 위해 MAVLink에서 받은 패킷을 uORB 자료구조로 변환한다.

2.5 CodeQL

CodeQL은 오픈소스 정적 코드 분석 도구이다. CodeQL에서 코드는 데이터처럼 취급되며 보안 취약점, 버그 및 기타 오류들은 코드에서 추출된 데이터베이스에 대해 실행할 수 있는 쿼리로 모델링된다[3]. CodeQL을 사용한 정적분석은 분석 대상에 대한 데이터베이스 생성, 해당 데이터베이스에 쿼리 실행, 쿼리 실행 결과 출력의 세 단계로 구성된다. 출력결과에는 검색한 패턴의 소스코드 위치, 위험도 등의 메타데이터들이 포함된다.

3. 관련 연구

3.1 CVE

CVE(Common Vulnerabilities and Exposures)는 공개적으로 알려진 소프트웨어 보안 취약점이다. 보안 데이터 분석에서 NIST(Natinal Institute of Standars and Technology)의 NVD(National Vulnerability Database)에서 제공해주는 CVE 정보는 분석의 핵심정보로 사용되고 있다[4]. PX4 Autopilot와 관련하여 알려진 CVE로는 2024.04.29. 기준 10개가 존재한다[5].

해당 CVE들의 유형으로는 부실한 type checking을 이용한 전역 오버플로우, 부족한 동기화 메커니즘을 이용한 race condition, NSH(NuttX Shell) 명령어를 이용한 취약점들이 있으며, 이를 이용하여 펌웨어 정보 유출, 드론 임무의 geofence 무시, 드론의 추락 유도 등의 악용 가능성이 존재한다.

3.2 MAVLink Fuzzer 구현 및 분석

[6]에서는 MAVLink 패킷을 생성하여 전송하는 Fuzzer를 직접

* “본 연구는 과학기술정보통신부 및 정보통신기획평가원의 2024년도 SW중심대학사업의 결과로 수행되었음” (2023-0-00042)

구현하여 MAVLink 프로토콜에 대한 동적 분석을 수행하였다. 분석 결과 2개의 취약점을 발견하였고 MsgID 332인 패킷에 대한 취약점으로 CVE를 할당받았다.

3.3 LDRA 정적분석

[7]에서는 LDRA Tool을 이용하여 PX4 Autopilot를 대상으로 MISRA 코딩 규칙에 위반하는 코드를 검출하는 정적 분석을 수행하였다. PX4 Autopilot 소스코드 중 /firmware-master/src/modules/ 폴더의 하위 37개의 폴더에 위치한 C++ 파일을 대상으로 하여 54%의 코딩규칙 위반을 검출하였다. 호출 깊이와 시험 가능성에서 낮은 품질을 보였다.

4. 취약점 분석 및 공격 재현

4.1 PX4 Autopilot 코드 구조 분석

CodeQL 결과를 분석하기 앞서서 PX4 Autopilot 소스 코드의 구조에 대한 기본적인 이해가 필요하다. 따라서 본 연구에서는 빌드 및 실행 과정 등의 전반적인 분석을 수행한다.

소스 코드 계층으로 “./src/”, “./boards”, “./platforms”, “./msg”, “./ROMFS” 등의 디렉토리가 존재하며 이들의 주요 역할을 분석한다. PX4 Autopilot이 MAVLink를 어떻게 git submodule로 포함하고 빌드하여 사용하는지 확인한다. 빌드 구성 파일의 확장자는 “px4board”이며, 이와 함께 “Make”, “CMake”, “Ninja” 등으로 이어지는 빌드 절차를 분석한다. 이를 통해 SITL, HITL, 실 기기 등의 환경별로 빌드가 어떻게 이루어지는지 확인한다. PX4 실행파일 바이너리와 초기화 스크립트가 어떻게 하나의 서버 데몬과 여러 client 프로세스 간의 통신을 통해 실행되는지 분석한다. 초기화 스크립트의 핵심 동작과 이를 통한 MAVLink stream 설정과 QGC 연결 동작을 확인한다.

4.2 CodeQL을 사용한 정적 취약점 분석

본 절에서는 CodeQL을 사용한 정적 취약점 분석 결과에 대해 서술한다. Ubuntu 22.04 환경에서 CodeQL CLI를 사용하여, PX4-Autopilot과 QGroundControl 소스코드를 대상으로 분석을 진행한다. CodeQL 분석 결과를 분류하고 취약점 가능성을 확인한 뒤 실제 공격을 재현한다.

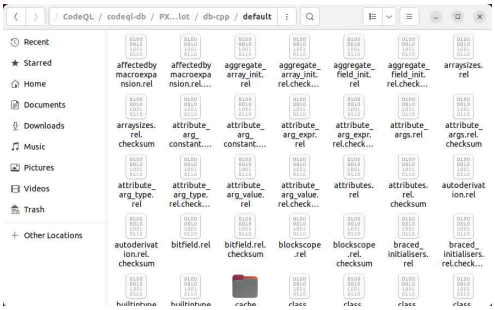
4.2.1 CodeQL 분석 시행 및 결과

먼저 각각의 Github repository에서 소스코드들을 clone한다. 그리고 CodeQL CLI를 사용하여 각각의 빌드 명령어를 인자로 넘겨주어 데이터베이스를 생성한다. 분석대상 언어는 C와 C++으로 한정한다.

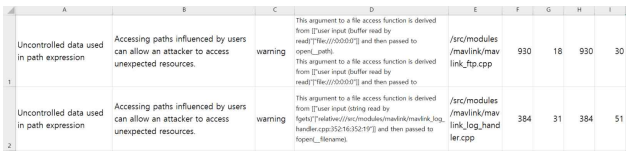
생성된 데이터베이스는 아래의 [그림 1]와 같이 각 소스코드 간의 ralational data가 추출되어 저장된다.

취약점 패턴 분석에는 CodeQL github repository의 cpp/ql/src/Security/CWE 경로에 저장되어있는 쿼리들을 사용한다. 해당 쿼리들은 CWE(Common Weakness Enumeration)에 해당하는 패턴들을 정의한 쿼리들로, C++기준 40가지의 CWE에 대한 68개의 쿼리들로 구성되어있다. 원활한 분석을 위해 전체 쿼리들을 6개로 분할하여 각각에 대한 쿼리 실행 결과를 csv 형식으로 출력시킨 뒤 각 결과를 취합하여 하나의 csv 파일로 통합한다.

분석 결과 PX4-Autopilot에서는 227곳의 보안 약점이 존재하고, QGroundControl에서는 365곳의 보안 약점이 존재한다. PX4-Autopilot의 분석 결과에서는 다양한 유형의 CWE 패턴이



[그림 1] PX4-Autopilot의 relational data



	A	B	C	D	E	F	G	H	I
1	Uncontrolled data used in path expression	Accessing paths influenced by users can allow an attacker to access unexpected resources.	warning	This argument to a file access function is derived from [[“user input (buffer read by read)”]“file:///:0:0:0:0”] and then passed to open(..,path). This argument to a file access function is derived from [[“user input (buffer read by read)”]“file:///:0:0:0:0”] and then passed to	/src/modules/mavlink/mavlink_ftp.cpp	930	18	930	30
2	Uncontrolled data used in path expression	Accessing paths influenced by users can allow an attacker to access unexpected resources.	warning	This argument to a file access function is derived from [[“user input (string read by fgets)”]“relative://src/modules/mavlink/mavlink_log_handler.cpp:352:16:352:19”] and then passed to fopen(..,filename).	/src/modules/mavlink/mavlink_log_handler.cpp	384	31	384	51

[그림 2] PX4 Autopilot CWE 쿼리 분석 결과

존재하나 QGroundControl의 경우 CWE-843에 해당하는 Type Confusion 관련 패턴만이 존재한다.

CodeQL 정적 분석 결과를 보안 약점 종류가 동일하고 원인의 위치가 같거나 비슷한 것을 합쳐 총 131개의 실질적인 보안 약점으로 분류하고, 그 종류와 위치에 따라서 아래의 [표 1]과 [표 2]와 같이 분류한다.

실질적인 보안 약점을 분류하면, 이중 86개의 약점은 PX4의 핵심 기능을 담당하는 모듈 안에 위치한다. 10개의 약점은 PX4의 입출력 드라이버 안에 위치하며, 12개의 약점은 드론의 하드웨어 및 운영체제별 동작을 담당하는 부분에 위치한다.

또한 아래의 [표 3]과 같이 소스 코드상에서 보안 약점의 실제 취약성을 추정하고 그 결과를 분류한다.

[표 1] - 보안 약점 종류 분류

보안 약점 종류	수
계	131
버퍼 길이 오버플로우	12
산술 연산 오버플로우	20
초기화 문제	82
파일 입출력 문제	10
동기화 문제	4
사용자 입력 실행 문제	3

[표 2] - 보안 약점 위치 분류

보안 약점 위치	수
계	131
‘./platforms’	12
‘./src/drivers’	10
‘./src/examples’	4
‘./src/lib’	16
‘./src/modules’	85
‘./src/systemcmds’	4

[표 3] - 보안 약점 실제 취약성 추정 결과

보안 약점 취약성 추정	수
계	131
취약성 확인	3
가능성 높음	3
가능성 낮음	6
가능성 없음	1
확인 불가, 추가분석 필요	76
미 분석	42

4.2.2 CodeQL 취약점 공격 재현

본 절에서는 CodeQL 보안 약점 결과에서 일부를 선정하여 실제 공격을 재현한다.

먼저 CodeQL 분석 결과에서 mavlink_log_handler.cpp 파일의 357번 줄에서 메모리 overflow 관련 보안 약점이 확인되었으므로 이를 SITL 환경에서 재현한다. PX4 Autopilot 로그 파일들의 목록을 관리하는 파일을 읽어오면서 파일명이 너무 긴 경우 스택 overflow를 유발할 수 있다는 결과이므로, 공격자가 PX4 Autopilot 파일 시스템에 대한 쓰기 권한을 가지고 있다는 가정하에 PX4 Autopilot에 해당 파일을 변조하는 별도의 코드를 추가하여 공격을 재현한다.

그 결과 실제로는 로그 목록을 읽어오는 과정에서 fgets 함수를 호출하면서 파일명의 최대 길이가 제한되어 스택 overflow가 발생하지 않음을 확인하였다. 다만 잘못된 파일명으로 인해 QGroundControl에서 로그 파일을 가져오는 동작에 에러가 발생함을 확인하였다.

또한 CodeQL 분석 결과에서 mavlink_log_handler.cpp 파일의 457번 줄에서 파일을 누구나 쓰기 가능한 권한으로 생성한다는 보안 약점을 확인하였다. 이는 앞서 공격을 재현한 로그 목록 파일로, 공격자가 PX4 Autopilot에서 임의의 user ID를 가지는 shell만 얻어도 취약점으로 악용될 수 있다. 따라서 이러한 가정하에 앞서 진행한 것과 같이 PX4 Autopilot에 해당 파일을 변조하는 별도의 코드를 추가하여 공격을 재현한다.

그 결과 공격자가 로그 목록의 파일명을 변조하여 PX4 실행 파일 바이너리를 QGroundControl을 통해 가져올 수 있음을 확인하였다.

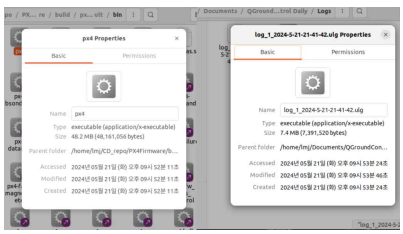
4.3 알려진 보안 취약점 공격 재현

본 절에서는 기존에 알려진 CVE 보안 취약점에 대한 공격 재현을 진행한다.

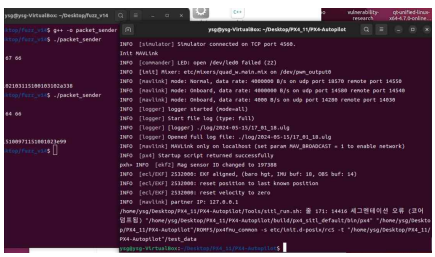
4.3.1 CVE 2021-46896 공격 재현

CVE-2021-46896은 msgID가 332인 MAVLINK 프로토콜 패킷을 수신할 때 payload의 값에 따라 힙 오버플로우가 발생하는 취약점이다. 해당 취약점을 재현하기 위해 C++로 직접 작성한 packet_sender 프로그램을 이용해 crash를 유도하는 패킷을 14580번 포트에 UDP로 전송한다. 재현 결과 PX4 Autopilot v1.11.0에서 세그멘테이션 오류와 함께 jmavsim 시뮬레이터와 px4_sitl가 비정상적으로 중단되는 것을 확인하였다. 또한

[그림 3] QGroundControl로 받아온 PX4 바이너리 일부



[그림 4] CVE 2021-46896 재현 결과



v1.14.0에서는 문제가 되는 변수의 값에 제한을 두어 더 이상 해당 취약점을 통한 공격을 허용하지 않는 것을 확인하였다.

4.3.2 CVE-2024-30800 공격 재현

CVE-2024-30800는 QGroundControl로 설정한 geofence 기능을 무시하여 발생하는 취약점이다. 해당 취약점을 재현하기 위해 먼저 waypoint(경유지)가 geofence 바깥에 있도록 mission plan을 구성하여 업로드한다. 이후 geofence를 접근하기 전에 드론을 공중에 멈추도록 하고, 기존 geofence를 넘어가던 waypoint를 포함하도록 geofence를 업데이트한다. 그 결과 PX4 Autopilot v1.14.0의 드론이 waypoint까지 비행하여 비행금지구역을 접근하는 것을 확인하였다. v1.15.0에서는 비행 전에 geofence를 벗어나는 경로를 점검하고 새로운 geofence를 업로드하지 못 하도록 하여 해당 공격을 예방하는 것을 확인하였다.

5. 결론 및 향후 연구

드론 시장이 확대됨에 따라서 드론 소프트웨어의 보안 중요성 또한 증가하고 있다. 그러나 오픈소스 드론 소프트웨어의 보안 취약점은 계속 발견되고 있으나 소프트웨어의 개발과 확대 속도와 비교하면 취약점 검사와 해결은 많이 부족한 실정이다.

따라서 본 논문에서는 PX4 Autopilot 오픈소스 드론 소프트웨어의 보안 약점을 CodeQL로 분석하여 결과를 확인하고, 추가로 알려진 CVE 취약점을 포함하여 SITL 환경에서 공격을 재현하였다. CodeQL 결과 중 일부는 전제조건이 갖추어지면 취약점을 악용할 수 있음을 확인하였다. 오픈소스 프로그램 작성 시 잠재적인 보안 약점을 더 주의해야 함을 확인하였다.

향후 연구에서는 유의미한 결과를 얻지 못했던 AFL++ 실행환경을 개선하여 동적분석을 진행하고, 정적·동적 분석결과를 바탕으로 추가적인 분석과 재현을 진행할 예정이다.

[참고 문헌]

[1] 항공안전기술원, 2021년도 국내외 드론 산업 동향 분석보고서, 2021, p.12.
[2] 정보통신산업진흥원, 드론 시장동향 보고서 2023, 2023, p.6.
[3] CodeQL overview, https://codeql.github.com/docs/codeql-overview/about-codeql/
[4] 홍성삼, 이동환, 김동화, 김용현, 김주엽, 김동욱, 안명길.(2023).텍스트 마이닝 기반의 머신러닝을 이용한 CVE-MITRE ATT&CK Technique 매핑 기법.한국정보과학회 학술발표논문집, 1181-1183.
[5] px4 CVE, https://cve.mitre.org/
[6] 경구창, 김동현, 최수빈, 이준오, 김지수, 류형호, 조진성, PX4 Autopilot의 MAVLink 모듈에 대한 취약점 분석, 한국정보보호학회 동계학술대회, 2021.
[7] 장정훈, 강유선, 이지현, 무인비행체 비행제어 Open Source 소프트웨어에 대한 정적분석 및 개선방안, 한국항공우주학회지, 49(6), 2021, pp.473-480.

[부록]

시연 동영상 URL:
https://drive.google.com/drive/folders/1NOcSTKmVoUHCInxG3EebE1d6uU6hCSbH?usp=sharing
PX4 Autopilot에 대한 CodeQL 보안 약점 분류 결과:
https://github.com/KhuCapstoneDesign/vulnerability-research/tree/master/static_analysis/codeql/PX4-Autopilot