2023 ACM MOBISYS

No More Companion Apps Hacking but One Dongle: Hub-Based Blackbox Fuzzing of IoT Firmware

2023-07-11

숭실대학교 대학원 석사과정 서영재

INDEX

01 Introduction

- 전 세계적으로 IoT 시장 규모가 커짐에 따라 IoT 디바이스에 대한 사이버 공격 또한 증가하고 있음
- 이러한 공격은 주로 IoT 펌웨어의 취약점을 악용하기 때문에, IoT 펌웨어 취약점을 발견하는 것이 중요함

취약점 찾기
- 정적 분석 : 소스코드 기반의 구조적 취약점을 분석하는 기술이며, 동적분석은 소프트웨어의 실행 기반으로 변수를 확인하며 분석하는 기법
- 동적 분석
  - Whitebox fuzzing : 테스트 대상의 내부 구조와 동작을 알고, 코드 커버리지 정보를 활용하여 효율적으로 취약점을 발견하는 기법
  - Greybox fuzzing : 일부 내부 정보를 알고, 중요한 경로나 분기 지점을 분석하여 해당 부분에 집중적으로 퍼징을 수행하여 취약점을 발견하는 기법
  - Blackbox fuzzing : 테스트 대상의 내부 구조에 대한 사전 정보 없이 외부에서 무작위 입력을 생성하여 실행하여 새로운 취약점을 발견하는 기법

01 Introduction

- 전 세계적으로 IoT 시장 규모가 커짐에 따라 IoT 디바이스에 대한 사이버 공격 또한 증가하고 있음
- 이러한 공격은 주로 IoT 펌웨어의 취약점을 악용하기 때문에, IoT 펌웨어 취약점을 발견하는 것이 중요함

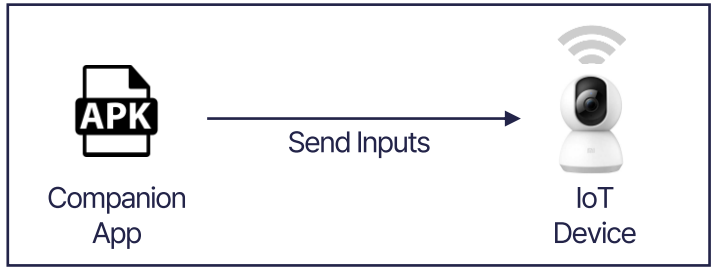
취약점 찾기
- 정적 분석
- 동적 분석
  - Whitebox fuzzing
  - Greybox fuzzing
  - Blackbox fuzzing

[Blackbox Fuzzing Example]

- IoTFuzzer 과 DIANE
  - IoT Companion App 활용하여 IoT Device로 입력을 전송함

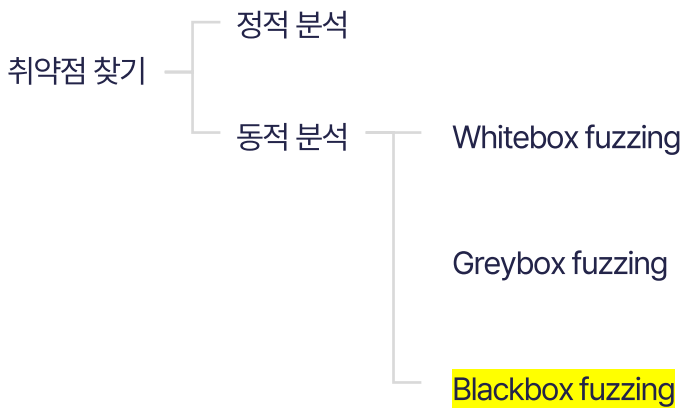
한계점

- 난독화된 Companion App을 리버스 엔지니어링하고 수정하는 데 노력이 필요함
- 정적 분석과 수작업을 결합하여 퍼징 메시지를 삽입하는데 사용할 수 있는 루틴을 식별함
  - 시간이 오래걸리고 불완전한 퍼징 메시지
- 이러한 퍼징은 Companion App 내부의 input sanitization로 인해 방해를 받음 > 특정 sanitization을 우회하여 완화(DIANE)



01 Introduction

- 전 세계적으로 IoT 시장 규모가 커짐에 따라 IoT 디바이스에 대한 사이버 공격 또한 증가하고 있음
- 이러한 공격은 주로 IoT 펌웨어의 취약점을 악용하기 때문에, IoT 펌웨어 취약점을 발견하는 것이 중요함



[Blackbox Fuzzing Example]

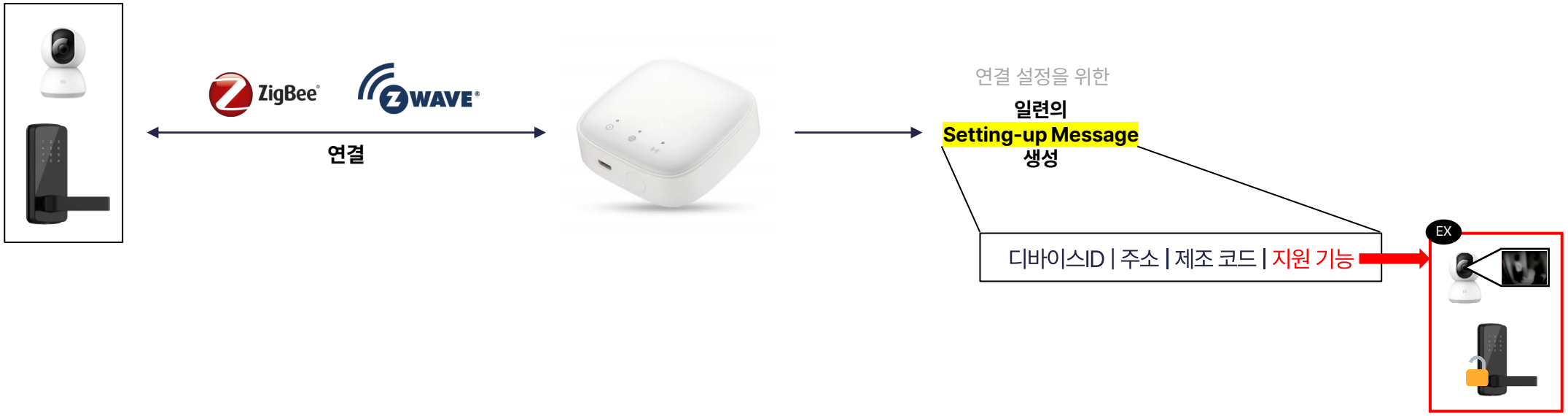
- SNIPUZZ
  - 수동으로 API 테스트 프로그램 수집하여 IoT Device 퍼징

한계점

- API 테스트 프로그램을 공개하는 vendor 거의 없음
- 공개된 테스트 스크립트가 반드시 device의 모든 기능 커버하는 것이 아님
- 수동으로 테스트 프로그램 수집
- API 테스트 프로그램에서 지원하는 디바이스 기능만 테스트할 수 있어 오탐이 발생할 수 있음
- 암호화된 메시지 처리 불가

01 Introduction

- **Insight & Main Idea**

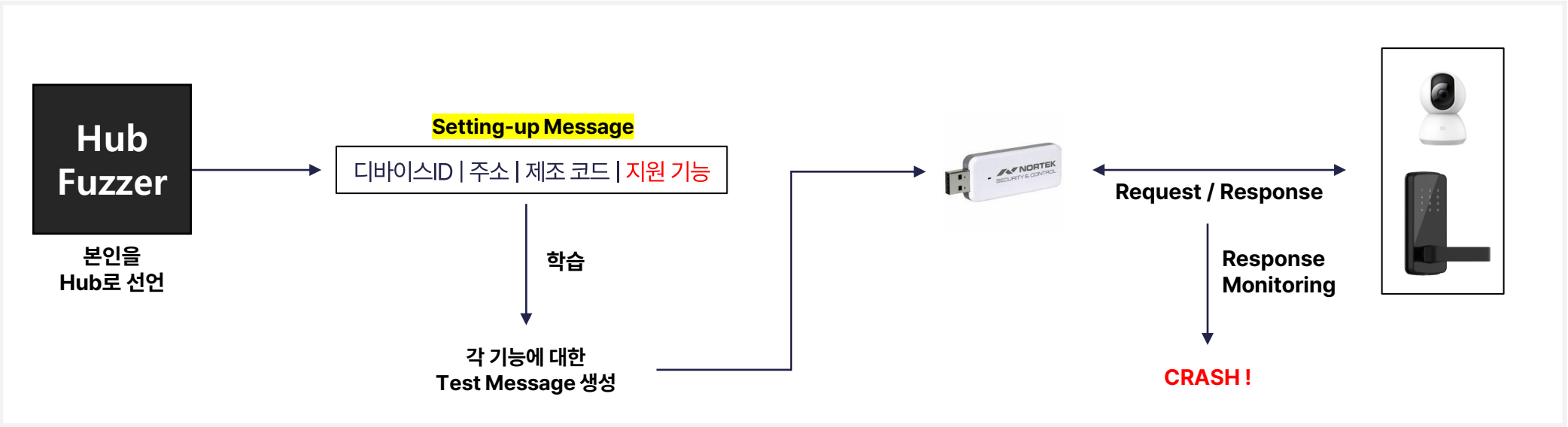


- 기존 어떠한 연구도 Setting-up Message를 IoT 퍼징에 활용하지 않음
- 본 연구는 Setting-Up Message 활용하여 기기의 기능 발견후 체계적인 **기능 지향 퍼징** 수행할 것을 제안
- **IoT Companion App을 리버스 엔지니어링하지 않고 Setting-up message 에서 기기가 지원하는 전체 기능 목록을 자동으로 도출함**

01 Introduction

- **Fuzzing Approach**
  - **기존 방식 :** 디바이스당 하나의 Companion App을 해킹하거나 수동으로 테스트 스크립트를 수집하여 퍼징 메시지 전송
  - **제안 방식 :** 균일한 허브 기반 퍼징

✓ Compainion App을 사용하지 않기 때문에, Companion App 내부의 다양한 input sanitization으로 퍼징이 방해되지 않음
✓ Device context가 테스트 결과에 영향을 미치는 경우가 많으므로 민감한 퍼징 수행
✓ 테스트 메시지 전송 전 디바이스를 특수 상태로 설정



* Device Context : 현재 퍼징 메시지가 전송되기 전의 디바이스 상태

01 Introduction

- **Contribution**
  - 새로운 허브 기반 동적 분석 아키텍처 제안하고 블랙박스 IoT 퍼징에 대한 유용성 입증
  - Setting-up message에서 IoT 장비가 지원하는 기능 도출하고 기능 지향 블랙박스 퍼징 제시
  - 프로토콜 사양에서 메시지 구조 및 명령/속성 값 범위에 대한 지식을 추출하여 IoT 장비를 테스트하는데 재사용
  - 디바이스 상태 민감 퍼징은 특정 디바이스 조건이 충족될 때만 트리거될 수 있는 취약점을 탐지하도록 설계
  - HubFuzzer을 구현하고 21개의 인기있는 IoT 디바이스에 대한 평가 수행하였으며, 4개의 CVE와 23개의 제로데이 취약점 발견

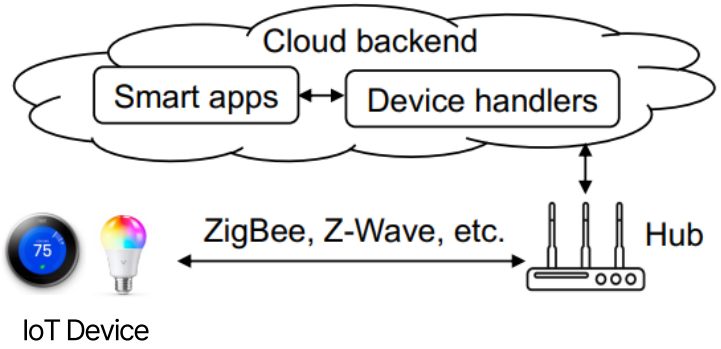
02 Related Work

- **Static Analysis-based Approaches**
  - IoT 펌웨어 이미지에 액세스함
  - 제조업체가 펌웨어를 공개하지 않는 경우가 많으므로, 접근법의 적용 가능성은 제한적임
  - 오탐률이 높은 경향이 있음

- **Dynamic Analysis-based Approaches**
  - **Emulation**
    - 확장성과 처리량 문제
  - **Symbolic Execution**
    - IoT 펌웨어의 정확한 실행을 위해서는 다양한 주변장치에 접근해야 함
    - 주변 장치의 모든 입력을 symbolic으로 간주하여 부정확성을 유발하거나 부정확한 emulation 결과를 사용함
  - **Blackbox Fuzzing**
    - IoTFuzzer과 DIANE은 리버스 엔지니어링 companion App에 의존
    - Snipuzz는 수동으로 수집한 테스트 프로그램에 의존

03 Background

- **IoT Platform Architecture**



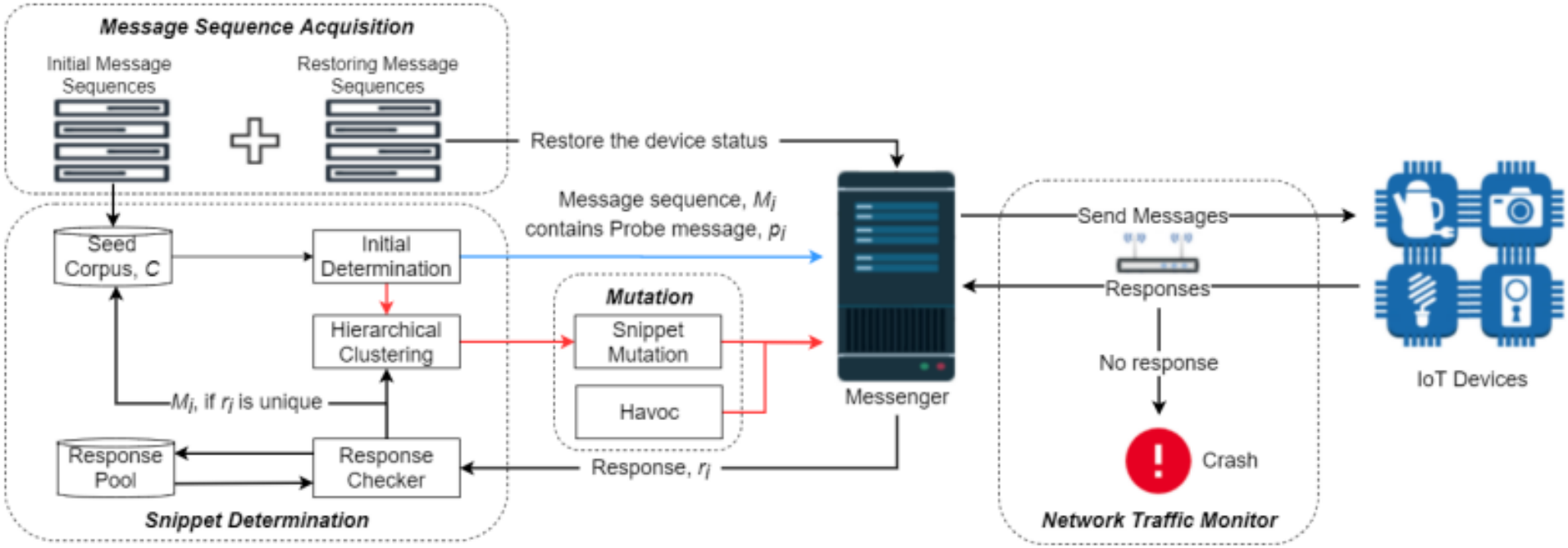
- **ZigBee and Z-Wave**

  - 홈 자동화 분야에서 가장 널리 사용되는 무선 프로토콜
  - 전력 소비가 적고 에너지 비용 절감
  - **본 연구에서는 Z 기반 장치라고 하는 Zigbee 및 Z-Wave IoT 장치에 대한 허브 기반 퍼징 접근 방식을 보여줌**

04

Limitations of the State of the Art
Overview

- **SNIPUZZ**
  - IoT 디바이스의 피드백을 사용하여 네트워크 패킷을 스니펫으로 분할하는 스니펫 기반 mutation 전략 구현



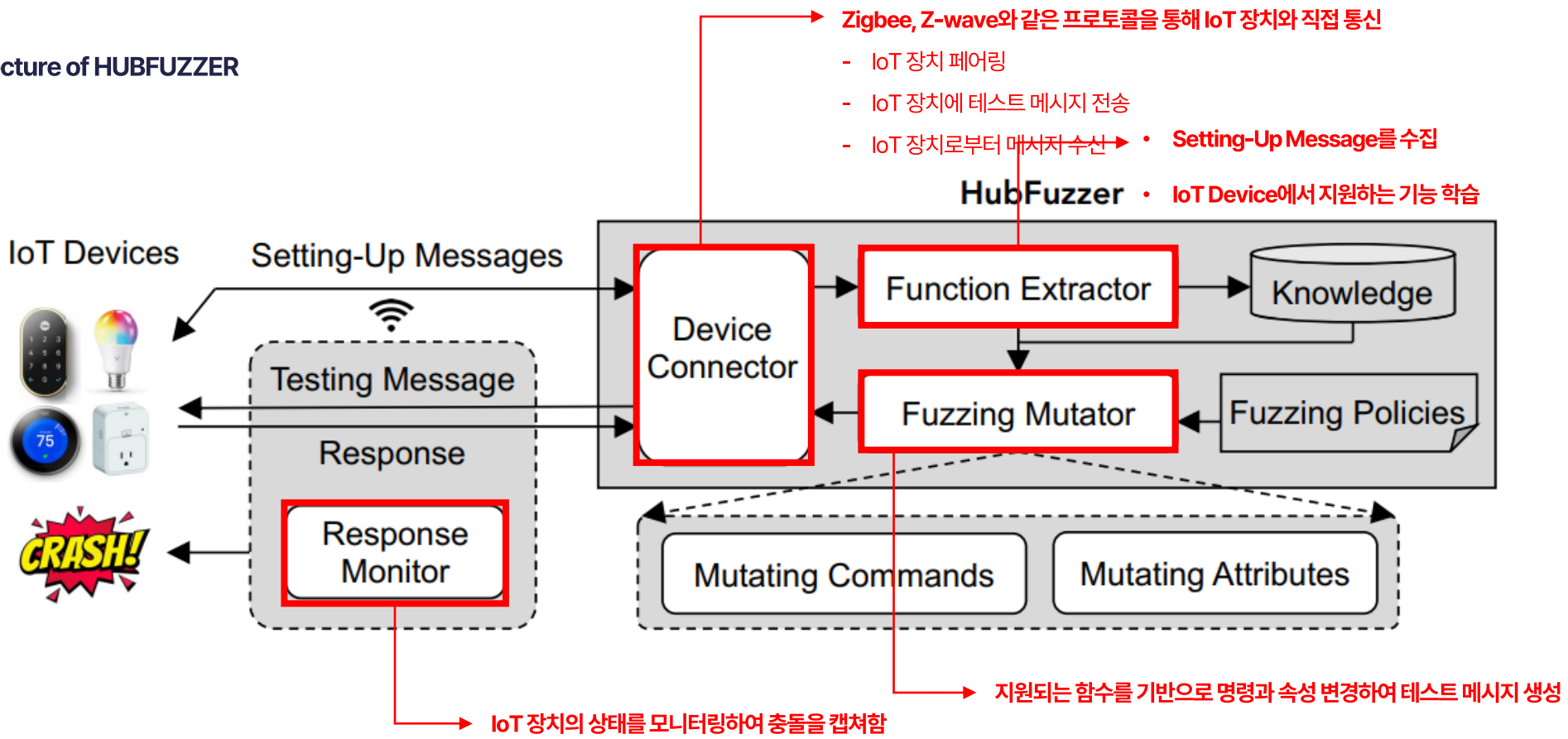
04 Overview

- **Our Goals**
  - 간편한 사용
  - 높은 기능 커버리지
  - 일반화 가능 : 복잡한 메시지와 암호화를 사용하는 디바이스에 일반화 가능
  - 상태에 민감한 퍼징

- **Insight and Idea**
  - 설정 메시지를 활용하여 IoT 디바이스가 지원하는 기능 파악 후 체계적인 **기능 지향 퍼징** 수행
  - 프로토콜 사양에서 메시지 명령/속성 값 범위에 대한 지식을 추출
  - 상태 감지 퍼징을 포함한 테스트 입력 변이 전략을 구축하는 데 사용됨

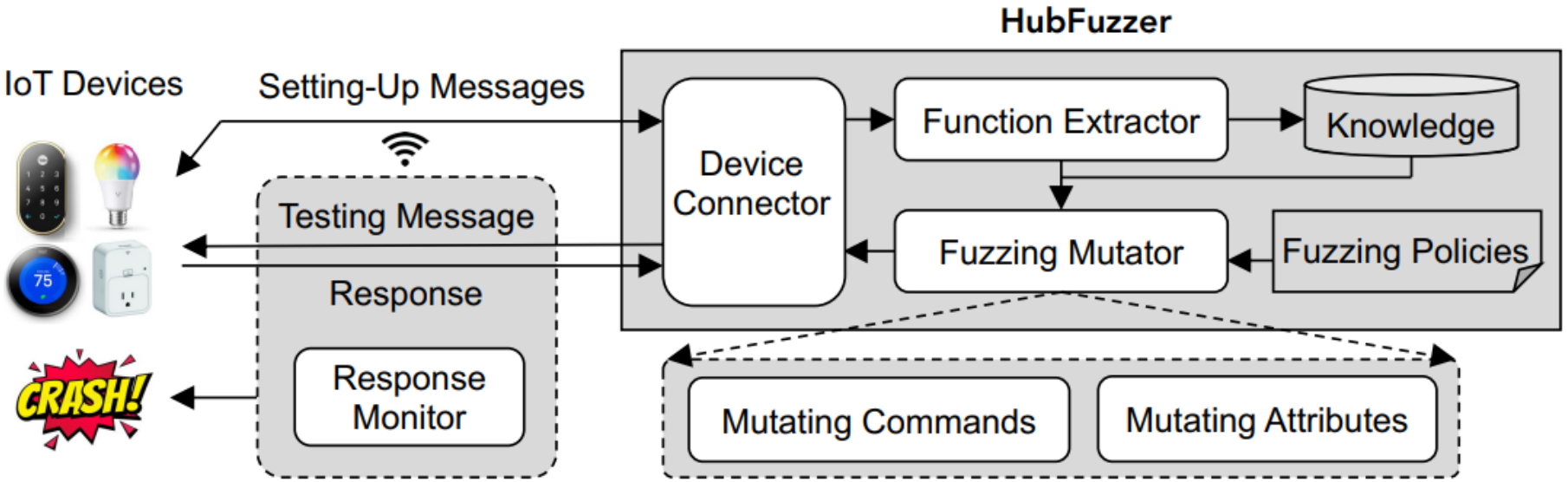
04 Overview

- **Architecture of HUBFUZZER**

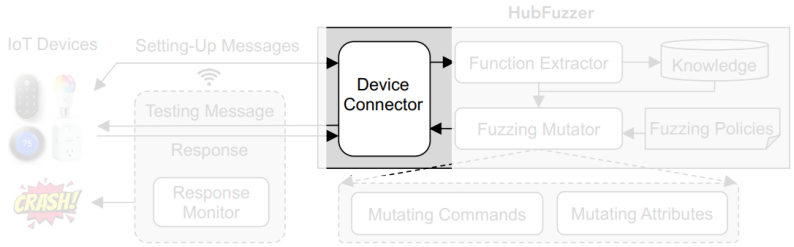


04 Overview

- **Architecture of HUBFUZZER**



05 Design of HubFuzzer



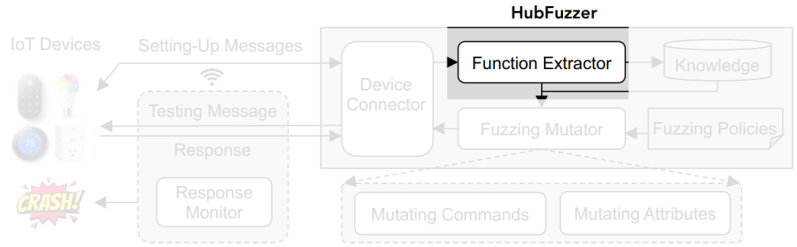
- **Device Connector**
  - Zigbee, Z-Wave와 같은 디바이스별 프로토콜을 통해 IoT 디바이스와 직접 통신
  - 여러 오픈 소스 플랫폼(홈 어시스턴트, 오픈햅, 웹씽즈 등)에서 장치 커넥터와 유사한 기능을 제공하며, 개발자가 다양한 IoT 장치를 통합하기 위한 애드온을 사용할 수 있음

  - Zigbee Protocol Stack이 포함되어 있어 Zigbee Device를 홈 어시스턴트에 직접 연결할 수 있음
  - Z-Wave JS 제공하여 Z-Wave Device와의 연결 지원

➢ 해당 Device Connector는 Zigbee 및 Z-Wave 무선 통신 기능을 위한 USB 동글로 제작됨

05

Learning Supported Functions
Design of HubFuzzer



- **Zigbee 의 클러스터**
  - ZCL에 정의된 대로 각 클러스터는 특정 기능에 해당하며 연결된 2 바이트 클러스터 식별자(CID)를 가짐
  - 예) ON/OFF 클러스터 (CID = 0x0006) 사용하면 스위치 장치를 전환할 수 있음

    Level 제어 클러스터 (CID = 0x0008) 사용하면 장치의 물리적 양의 레벨을 제어할 수 있음

  - 클러스터 : 특정 기능에 대한 인터페이스를 정의하는 명령과 속성의 모음

    속성 : 상태로 저장할 수 있는 장치의 속성

    명령 : 장치를 제어하고 속성을 조작할 수 있는 메서드

현재 클러스터 ID 저장

```
def request(device, profile, cluster, src_ep, dst_ep,
        sequence, data, expect_reply=True, use_ieee=False)
```

Listing 1: ZigBee packing procedure

Payload 저장

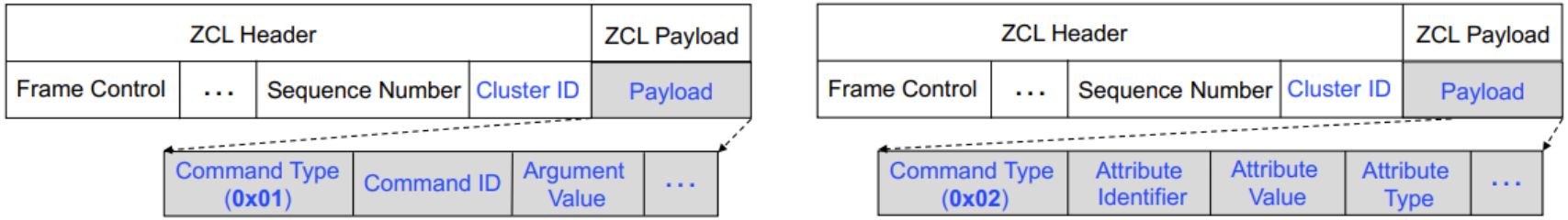
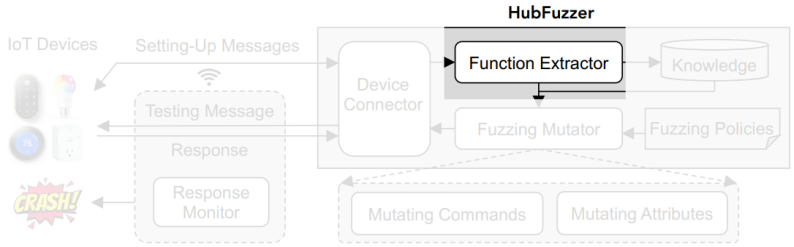


Figure 5: Format of the ZCL frame.

05

Learning Supported Functions
Design of HubFuzzer



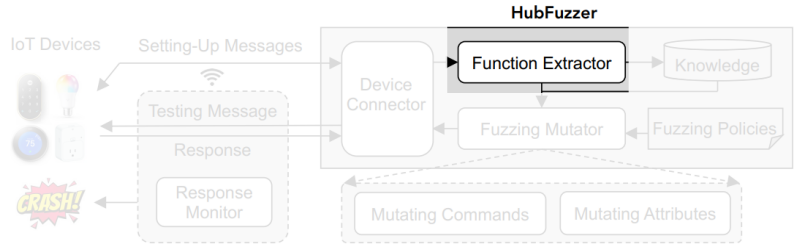
- **Z-Wave의 명령어 클래스**
  - 장치 기능을 추상화하고 관련 기능을 명령어 클래스로 그룹화함
  - 명령어 클래스는 명령과 속성의 모음으로, 명령은 특정 기능에 해당하는 장치 속성을 제어, 쿼리 및 보고하는데 사용됨

  - Set : 장치에 특정 작업(장치 상태 변경할 수 있는)을 수행하도록 장치에 전송
  - Get : 장치의 현재 상태를 요청하기 위해 장치로 전송
  - Report : 상태가 변경된 경우 현재 장치 상태를 보고하기 위해 장치에서 전송

05

Learning Supported Functions

Design of HubFuzzer



- **Our Method**

목표 : 테스트 메시지를 IoT 장비에 전송하여 장비 동작을 트리거하는 것

1) 장치가 지원하는 기능 학습
2) 장치를 제어할 수 있는 명령 결정
3) 장치에 저장된 속성 결정

Tr_zigbee.db

- attributes_cache_v7
- devices_v7
- endpoints_v7
- group_members_v7
- groups_v7
- in_clusters_v7
- neighbors_v7
- node_descriptors_v7
- out_clusters_v7
- relays_v7
- unsupported_attributes...

ieee	endpoint_id	cluster
28:2c:02:bf:ff:e9:7d:e5	1	0
28:2c:02:bf:ff:e9:7d:e5	1	1
28:2c:02:bf:ff:e9:7d:e5	1	6

가장 중요한 메시지

현재 장치가 해당 명령을 사용하여 다른 장치를 제어할 수 있음을 나타냄

기본 클러스터
- 모든 ZigBee 장치에 필수적인 클러스터
- 디바이스와 기본 속성을 포함한 21개의 속성 + reset 명령

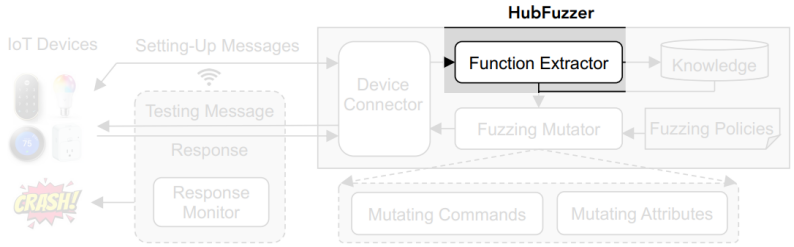
전원 구성 클러스터
- 장치의 전원에 대한 정보를 얻고 전압 알람 구성
- 배터리와 관련된 58개의 속성 정의

On/Off 클러스터
- 조명의 on/off 시간 지정과 같은 5개의 속성 + 6개의 명령

[그림 4] 스마트 스위치 장치의 Setting-up Message 예시

05

Learning Supported Functions
Design of HubFuzzer



- **퍼징을 위한 지식 추출**
  - 프로토콜 사양은 포함된 각 클러스터에 대한 자세한 설명 제공하고, 해당 프로토콜 스택 라이브러리에 반영됨
  - 명령어, 인수/속성, 각 인수의 데이터 유형 및 값 범위 등

1) 스크립트를 사용하여 라이브러리에서 관련 정보 추출
2) 데이터베이스에 저장
3) HubFuzzer에 통합

일회성 작업

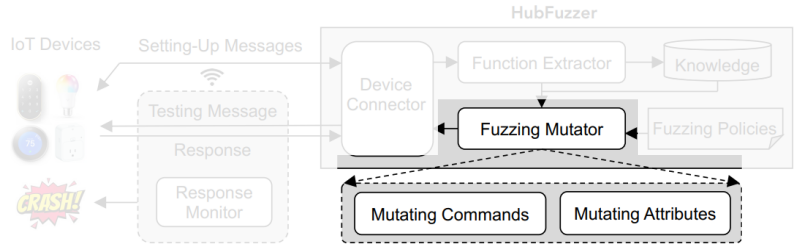
- **디바이스 퍼징**

1) 설정 메시지에서 지원되는 클러스터를 학습
2) 데이터베이스에서 지원하는 명령과 속성, 각 명령 인수의 값 범위와 데이터 유형 검색
3) 이 정보를 통해 테스트 메시지 생성 지원

05

Packing Procedure

Design of HubFuzzer



- **Packing Procedure**

1) 각 명령 요청에 의해 호출되어 IoT 장치로 전송할 메시지 생성

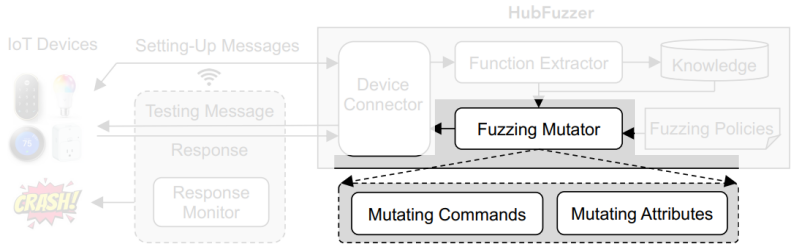
2) Packing Procedure에서 **input sanitization** 처리를 제거하고 메시지를 작성하기 위한 정보를 변경하여 테스트 메시지를 생성함

- **기존 작업**
  - 테스트 메시지를 준비하기 위한 기능을 찾아야 함
  - 각 디바이스의 컴패니언 앱에서 sanitization 처리 제거
    = 각 IoT 기기의 컴패니언 앱을 리버스 엔지니어링

- **HubFuzzer**
  - 테스트 메시지를 준비하기 위한 기능을 찾아야 함
  - sanitization 처리 제거는 한 번만 수행
    = 패킹 절차를 수정하는데 한 번만 노력

05

Fuzzing Policies
Design of HubFuzzer



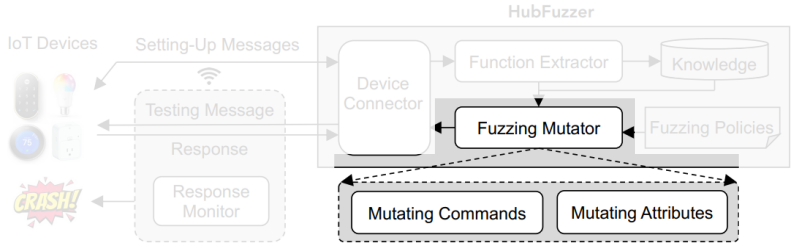
We have the following fuzzing policies.

- **Policy 1 : 인수 값 변경**
  - 고위험 유형 : 해당 유형의 인자 값이 변경될 경우 디바이스 충돌을 일으킬 가능성이 높은 유형 (예: OctetString, CharString, 정수, 실수, float, Array, Set 및 NoData)
  - 저위험 유형 : (예: 시간, 날짜)
- **Policy 2 : 인자 유형 변경**
- **Policy 3 : Agument 수 변경**
- **Policy 4 : 지원되지 않는 클러스터 및 명령어 시도**

05

State-Sensitive Fuzzing

Design of HubFuzzer



- **실험 결과**

  디바이스 상태가 충돌을 유발하는 데 상당한 영향을 미치는 것으로 나타남

  예) 특정 비율로 스마트 전구의 밝기를 높이는 테스트 메시지가 주어졌을 때 전구의 밝기가 가장 높을 때 미시지가 전구 충돌을 일으킴

  ➢ **상태 민감 퍼징 수행**

1. 디바이스 상태를 특수한 상태로 설정한 후 상태에 영향을 줄 수 있는 테스트 메시지를 보냄

2. 명령이 주어지면 먼저 수정하려는 디바이스 속성을 결정함
   - **Write-Attribute Command :** 속성이 지정되어 있으므로 쉽게 결정할 수 있음
   - **Cluster Command :** 명령에 의해 수정되는 속성 찾음

3. 속성의 값 범위 결정

4. 범위 지정 시 - 속성과 관련된 인수에 최대값과 최소값을 제공하여 두 개의 리셋 메시지 생성

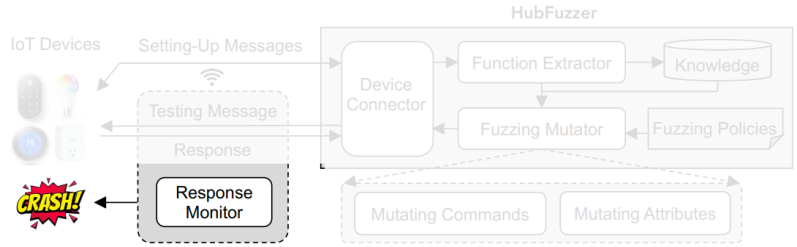
   범위 지정 X - 임의의 값을 사용하여 재설정 메시지 생성

- 기존 IoT Fuzzing 연구에서는 디바이스 상태의 영향 고려하지 않음 (IoTFuzzer, SNIPUZZ, DIANE)

  ➢ 특수한 조건이 충족될 때만 충돌을 일으키는 많은 취약점을 탐지하지 못함

05

Response Monitor
Design of HubFuzzer



- **IoT 디바이스가 메시지 m을 수신할 때의 4가지 시나리오**

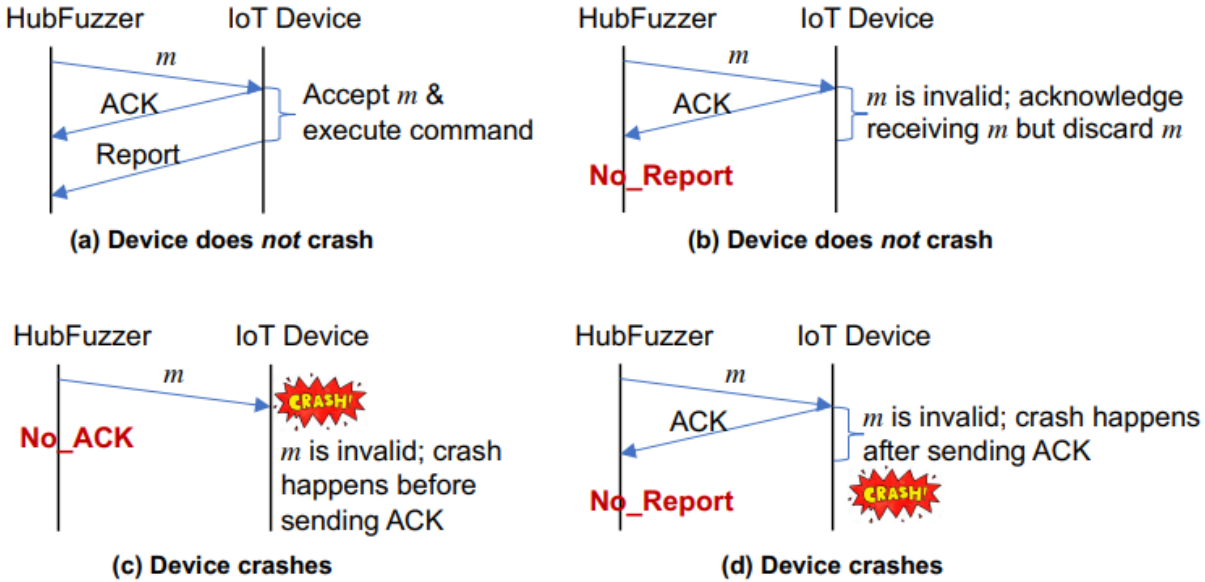


Figure 6: Different scenarios when an IoT device receives a message.

- **허브 퍼저에서 ACK 응답이 수신되지 않으면 장치 충돌이 발생한 것**
  - 네트워크 연결 상태가 좋지 않은 경우 메시지가 손실되어 ACK도 손실될 수 있으나 누락된 ACK가 충돌로 인한 것이라면 동일한 메시지를 사용하여 증상을 안정적으로 재현할 수 있음
  - 그러나 네트워크 연결 상태가 좋지 않아서 발생한 경우에는 재현할 수 없으며, 기기가 충돌로 인해 재시작해도 LED가 깜박이거나 경고음이 울리는 등의 증상이 나타나지 않음
- **ACK는 수신되었지만 Report 응답이 수신되지 않은 경우, 디바이스의 충돌 여부를 확인할 수 없음(그림 6(b) 및 (d) 참조)**
  - 다음 상태 민감도 테스트를 위해 유효한 것으로 알려진 리셋 메시지를 전송하여 이를 구분해야 함
  - Report 응답이 수신되지 않으면 디바이스가 충돌했음을 나타냄

06

Implementation
Evaluation

- **HubFuzzer은 Zigbee, Z-wave와 같은 디바이스별 프로토콜을 통해 IoT 디바이스와 직접 통신함**
- **여러 오픈소스 플랫폼(Home Assistant, ..)에서 장치 커넥터와 유사한 기능을 제공하며 개발자가 다양한 IoT 장치를 통합하기 위한 애드온을 사용할 수 있음**
- **홈 어시스턴트의 ZigBee 및 Z-Wave 애드온을 활용하여 Z 기반 장치와 연결함**
  - 장치 커넥터 : Nortek security & Control HUSBZB-1 USB 동글로 제작

Nortek security & Control HUSBZB-1 USB Dongle

06

Experimental setup
Evaluation

- **온/오프라인 시장에서 인기있는 Z 기반 IoT 기기 21종 선정**
  - (a) 디바이스 세부 정보
  - (b) 디바이스의 사진

- **Test 환경**
  - Ubuntu 20.04 PC with 4.9 GHz Intel® Core(TM) i7 CPU and 32 GB RAM 에서 실행
  - 관련없는 트래픽의 간섭을 피하기 위해 완전히 제어된 네트워크에서 Z 기반 디바이스를 구성함

- **Baseline Method**
  - SNIPUZZ

ID	디바이스 유형 Device Type	공급업체 Vendor	모델 Model	펌웨어 버전 Firmware Version	프로토콜 Protocol
1	Thermostat	Centralite	Pearl	0x04075010	ZigBee
2	Lighting	Sengled Bulb	E11-N1EAW	0x00000024	ZigBee
3	Lighting	Innr	FL 120 C	0x28002162	ZigBee
4	Lighting	Philip Bloom	Bloom	1.88.1	ZigBee
5	Lighting	Sengled Strip	E1G-G8E	0x00000024	ZigBee
6	Blind	Third Reality	3RSB015BZ	1.00.54	ZigBee
7	Plug	Lumi	ZNCZ12LM	18	ZigBee
8	Locker	Schlage	BE468GBAK CAM 619	0.21.0	ZigBee
9	Locker	Kwikset	99140-139	0x40a32a10	ZigBee
10	Sensor	Tuya	B09TDQ9GP6	v1.0.10	ZigBee
11	Switch	Tuya	B09XCP7DN1	v1.1.0	ZigBee
12	Switch	Third Reality	3RSS009B	v1.01.10	ZigBee
13	Dimmer Switch	Sengled	E1E-G7F	v0.0.9	ZigBee
14	Plug	Minoston	MP21Z	v1.0.1	Z-Wave
15	Wall Switch	Aeotec	ZW130-A	v2.3	Z-Wave
16	Sensor	Aeotec	ZW080	v3.28	Z-Wave
17	Switch	Aeotec	ZW096-A	v1.7	Z-Wave
18	Motion Sensor	Fibaro	FGMS-001	v3.4	Z-Wave
19	Thermostat	Honeywell	TH6320ZW2003	v1.3	Z-Wave
20	Locker	Kwikset	98880-004	v4.79	Z-Wave
21	Dimmer Plug	Minoston	MP22Z	v7.13.9	Z-Wave

(a) Device details

SNIPUZZ와 비교하기 위해 선택
(SNIPUZZ에서 테스트한 다른 디바이스는 WiFi기반이므로 포함하지 않았음)

(b) Photo of devices

06

Function Coverage Evaluation

- 기능 커버리지 결과

Table 1: Function Coverage Results. N/A: Z-Wave devices do not support write-attribute commands.

ID	HubFuzzer			SNIPUZZ		
	Cluster	Command	Attribute	Cluster	Command	Attribute
1	8(100%)	29(100%)	91(100%)	2(25%)	19(66.7%)	0(0%)
2	8(100%)	48(100%)	99(100%)	3(37.5%)	5(10.4%)	0(0%)
3	8(100%)	44(100%)	94(100%)	3(37.5%)	5(11.4%)	0(0%)
4	8(100%)	54(100%)	100(100%)	4(50%)	14(25.9%)	0(0%)
5	8(100%)	48(100%)	99(100%)	3(37.5%)	5(10.4%)	0(0%)
6	7(100%)	30(100%)	93(100%)	1(14.3%)	10(33.3%)	0(0%)
7	7(100%)	26(100%)	100(100%)	1(14.3%)	3(11.5%)	0(0%)
8	7(100%)	55(100%)	81(100%)	1(14.3%)	3(6%)	0(0%)
9	7(100%)	55(100%)	81(100%)	1(14.3%)	3(6%)	0(0%)
10	3(100%)	4(100%)	79(100%)	0(0%)	0(0%)	0(0%)
11	5(100%)	26(100%)	34(100%)	1(20%)	3(11.5%)	0(0%)
12	6(100%)	26 (100%)	91(100%)	1(16.7%)	3(11.5%)	0(0%)
13	5(100%)	26(100%)	42(100%)	1(20%)	3(11.5%)	0(0%)
14	14(100%)	89(100%)	N/A	1(7%)	3(3%)	N/A
15	19(100%)	117(100%)	N/A	3(16%)	6(5%)	N/A
16	16(100%)	85(100%)	N/A	3(19%)	7(8%)	N/A
17	18(100%)	89(100%)	N/A	1(5%)	3(4%)	N/A
18	20(100%)	115(100%)	N/A	0(0%)	0(0%)	N/A
19	21(100%)	113(100%)	N/A	2(9.5%)	19(16.8%)	N/A
20	15(100%)	106(100%)	N/A	1(6.7%)	3(2.8%)	N/A
21	15(100%)	96(100%)	N/A	1(6.7%)	3(3.1%)	N/A

Write-Attribute Command 지원 X

- **Cluster and Command Coverage**
  - **HubFuzzer**
    설정 메시지를 통해 장치의 지원하는 클러스터를 추출
    이를 통해 클러스터, 명령어, 속성에 대한 전체 커버리지를 달성할 수 있음
  - **SNIPUZZ**
    API 테스트 프로그램을 사용하여 테스트 메시지를 생성
    API 테스트 프로그램이 커버하지 못하는 경우에는 명령어 커버리지를 달성하기 어렵거나 불가능함
- **Attribute Coverage**(ZigBee 장치의 경우)
  - **HubFuzzer**
    쓰기 속성 명령을 통해 모든 속성을 수정하는 테스트 메시지를 생성할 수 있음
  - **SNIPUZZ**
    API 테스트 프로그램에서는 쓰기 속성 명령을 지원하지 않아 속성 수정을 테스트할 수 없음

06 Vulnerability Discovery
Evaluation

- 디바이스 Fuzz test하려면 > 수동으로 Hubfuzzer와 페어링하는 방법 밖에 없음

Table 2: Vulnerability Discovery Results. (*UV*: Unknown Vulnerability. *DoS*: Denial of Service.)

ID	HubFuzzer		SNIPUZZ	
	Vul. Type	Number	Vul. Type	Number
1	UV/DoS	2	-	0
2	UV/DoS	8	-	0
3	-	0	-	0
4	-	0	-	0
5	UV/DoS	8	-	0
6	UV/DoS	2	-	0
7	-	0	-	0
8	-	0	-	0
9	-	0	-	0
10	-	0	-	0
11	-	0	-	0
12	-	0	-	0
13	DoS	1	-	0
14	-	0	-	0
15	UV	1	-	0
16	-	0	-	0
17	-	0	-	0
18	DoS	1	-	0
19	-	0	-	0
20	-	0	-	0
21	-	0	-	0

7개의 디바이스에서 23개의 제로데이 취약점 발견

Table 3: *Some* of Discovered Vulnerabilities Details. (HD API: Hidden API, defined in Section 4.1.)

ID	HD API?	Command	Normal Mes. ⟶ Exploit (only *payload* is shown)	Required Device Initial State	Observations	CVE
1	✓	`Write_Battery_thres(uint8)`	0x02007d20 ⟶ 0x02007d42 ***Policy 2:*** Change argument type from `uint8` to `CharString`	Heating mode is set to 32 degrees Celsius	Device crashed & bricked	CVE-2023-24678
2	✓	`Move_up(uint8)`	0x01010003 ⟶ 0x01010000 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the lowest (0)	Device crashed	CVE-2022-47100
2	✓	`Move_down(uint8)`	0x01010102 ⟶ 0x01010100 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the highest (254)	Device crashed	CVE-2022-47100
5	✓	`Move_up_OnOff(uint8)`	0x01050007 ⟶ 0x01050000 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the lowest (0)	Device crashed	CVE-2022-47100
5	✓	`Move_down_OnOff(uint8)`	0x01050108 ⟶ 0x01050100 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the highest (254)	Device crashed	CVE-2022-47100
6	✗	`Down_close()`	0x0103 ⟶ 0x010303 ***Policy 3:*** Provide one or more arguments to the command	Any state	Device crashed	CVE-2023-29780
13	✓	`Set_short_poll_interval(uint16)`	0x01030009 ⟶ 0x01030000 ***Policy 1:*** Provide an invalid value 0x0000 to the argument	Any state	Device kept reporting state until battery was drained	CVE-2023-29779
15	✓	`Firmware_Update_Request_Get(uint8,uint8)`	0x7A038601 ⟶ 0x7A03ff ***Policy 3:*** Provide *only one* argument with an invalid value 0xff	Any state	Device crashed	Under review
18	✓	An invalid command in the Clock cluster	0x8101 ***Policy 4:*** Try unsupported cluster 0x81 and invalid command 0x01	Any state	Device crashed	Under review

06

Vulnerability Discovery
Evaluation

ID	HD API?	Command	Normal Mes. ⟶ Exploit (only *payload* is shown)	Required Device Initial State	Observations	CVE
1	✓	`Write_Battery_thres(uint8)`	0x02007d20 ⟶ 0x02007d42 ***Policy 2:*** Change argument type from `uint8` to `CharString`	Heating mode is set to 32 degrees Celsius	Device crashed & bricked	CVE-2023-24678

- **사례 연구 1 : Centralite Pearl Thermostat (ID = 1)**
  - UV(알려지지 않은 취약점) / DoS(서비스 거부)
  - **Write_Battery_thres(uint8)** : 배터리 부족 경보의 임계값을 설정할 수 숨겨진 명령으로, Uint8인 인수를 하나만 받음

- **취약점 트리거**
  - 데이터 유형을 CharString으로 변경하고 동시에 디바이스를 섭씨 32도로 설정하면 디바이스 충돌 발생
  - 정상 메시지에서는 0G20으로 표시되는 단위 8 데이터 유형을 CharString으로 변경하면 디바이스 충돌 발생

- **관찰 결과**
  1. 첫 번째 관찰 결과는 익스플로잇(0G02007342)이 전송되면 디바이스의 연결이 끊기고 약 1초 후에 자동으로 재연결됨
  2. 두 번째 관찰 결과는 일정 시간 내에 여러 번의 익스플로잇을 전송하면 디바이스의 연결이 끊어지고 자동으로 재연결되지 않아 공격자가 DoS공격을 수행할 수 있음
  - 이러한 상태에서는 디바이스를 재부팅하거나 수동으로 공장 초기화하거나 페어링을 수동으로 해도 네트워크에 다시 연결할 수 없어 디바이스가 완전히 사용 불가능한(bricked) 상태가 됨

06

Vulnerability Discovery
Evaluation

2	✓	Move_up(uint8)	0x01010003 ⟶ 0x01010000 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the lowest (0)	Device crashed	CVE-2022-47100
2	✓	Move_down(uint8)	0x01010102 ⟶ 0x01010100 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the highest (254)	Device crashed	CVE-2022-47100
5	✓	Move_up_OnOff(uint8)	0x01050007 ⟶ 0x01050000 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the lowest (0)	Device crashed	CVE-2022-47100
5	✓	Move_down_OnOff(uint8)	0x01050108 ⟶ 0x01050100 ***Policy 1:*** Provide an invalid value 0x00 to the argument	Brightness is set to the highest (254)	Device crashed	CVE-2022-47100

- **사례 연구 2 : Sengled Bulb E11-N1EAW (ID = 2)**
  - UV(알려지지 않은 취약점) 4개 / DoS(서비스 거부) 4개
  - **Move_up(unit8), Move_down(unit8), Move_up_OnOff(unit8), Move_down_OnOff(unit8)**
    : 특정 비율로 장치의 밝기를 켜거나 끌 수 있는 OnOff 효과의 유무와 관계없이 밝기를 증가 또는 감소시키는 명령

- **취약점 트리거**
  - 데이터 유형을 CharString으로 변경하고 동시에 디바이스를 섭씨 32도로 설정하면 디바이스 충돌 발생
  - 정상 메시지에서는 0G20으로 표시되는 단위 8 데이터 유형을 CharString으로 변경하면 디바이스 충돌 발생

- **관찰 결과**
  1. UV: 익스플로잇이 한 번 전송되면 디바이스가 한 번 충돌하고 연결이 끊긴 후 자동으로 공장 상태로 변경된 다음 1초 후에 네트워크에 다시 접속
  2. 확장 DoS: 익스플로잇이 일정 시간 내에 여러 번 전송되면 수동으로 페어링할 때까지 디바이스가 자동으로 다시 연결되지 않으므로 공격자는 확장된 DoS 공격을 수행할 수 있음

06

Vulnerability Discovery
Evaluation

6	×	Down_close()	0x0103 ⟶ 0x010303 ***Policy 3:*** Provide one or more arguments to the command	Any state	Device crashed	CVE-2023-29780

- **사례 연구 3 : Third Reality 3RSB015BZ (ID = 6)**
  - **CVE-2023-29780이 할당된 두 가지 취약점을 탐지**
  - Down_close() : 스마트 블라인드를 최대 길이까지 확장할 수 있는 명령어로, 인수를 받지 않음

- **취약점 트리거**
  - 인수를 제공하면 현재 디바이스 상태에 관계없이 생성된 테스트 메시지가 취약점을 트리거할 수 있음(Privacy 3)

- **관찰 결과**
  1. 테스트 메시지가 한 번 전송되면 기기의 연결이 끊어졌다가 자동으로 재연결
  2. 여러 번 연속해서 전송되면 약 30초 후에 연결이 끊어졌다가 재연결되어 공격자가 DoS 공격을 수행할 수 있는 것으로 관찰됨

06

Vulnerability Discovery
Evaluation

13	✓	Set_short_poll_interval(uint16)	0x01030009 ⟶ 0x01030000 ***Policy 1:*** Provide an invalid value 0x0000 to the argument	Any state	Device kept reporting state until battery was drained	CVE-2023-29779

- **사례 연구 4: Sengled E1E-G7F (ID = 13)**
  - **CVE-2023-29779가 할당된 DoS 취약점**
  - Set_short_poll_interval(uint16) : 짧은 폴링의 간격을 설정할 수 있는 명령어

- **취약점 트리거**
  - 인자에 유효하지 않은 값 0x0000을 제공

- **관찰 결과**
  1. 디바이스가 간격 없이 상태를 계속 보고하고 정상적인 메시지에 응답하지 않음
  2. 실험에서 디바이스가 상태를 계속 보고하면서 배터리가 방전됨

06

Vulnerability Discovery
Evaluation

15	✓	`Firmware_Update_Request_Get(uint8,uint8)`	0x7A038601 ⟶ 0x7A03ff ***Policy 3:*** Provide *only one* argument with an invalid value 0xff	Any state	Device crashed	Under review

- **사례 연구 5: Aeotec ZW130-A (ID = 15)**
  - **UV 취약점 발견,** 해당 취약점에 대한 CVE 요청했으며 현재 검토중
  - Firmware_Update_Request_Get (uint8, uint8) : 펌웨어 업데이트를 시작하는데 사용되는 숨겨진 명령어로, 2개의 인수 필요

- **취약점 트리거**
  - 유효하지 않은 값 0x#을 가진 인수를 하나만 제공하면 디바이스 충돌 발생

06

Vulnerability Discovery
Evaluation

18	✓	An invalid command in the Clock cluster	0x8101 ***Policy 4:*** Try unsupported cluster 0x81 and invalid command 0x01	Any state	Device crashed	Under review

- **사례 연구 6: Fibaro FGMS-001 (ID = 18)**
  - **DoS 취약점 발견,** 해당 취약점에 대한 CVE 요청했으며 현재 검토중
  - Fibaro FGMS-001 : 동작 감지, 조명 변화, 환경 온도 변화 등 다양한 동작이나 이벤트를 트리거하도록 설계되어 있음
  - Device 시계를 컨트롤러 시스템 시계와 동기화하는 것을 목표로 하는 Clock 클러스터(0x81)의 잘못된 명령과 관련

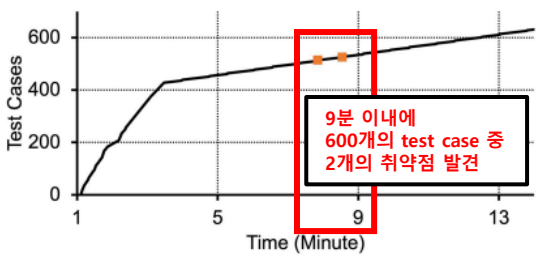
- **취약점 트리거**
  - 지원되지 않는 클러스터(0x81)와 유효하지 않은 명령(0x01)이 포함된 테스트 메시지를 보내면 해당 명령은 클러스터(0x81)에 존재하지 않으므로 유효하지 않음

- **관찰 결과**
  - 테스트 메시지를 일정 시간 내에 여러 번 전송하면(실험에서는 약 60초 동안 120번 전송) 디바이스 충돌

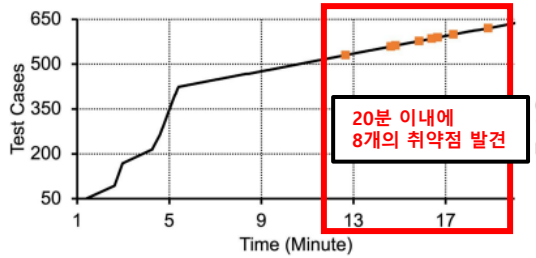
06

Efficiency Evaluation

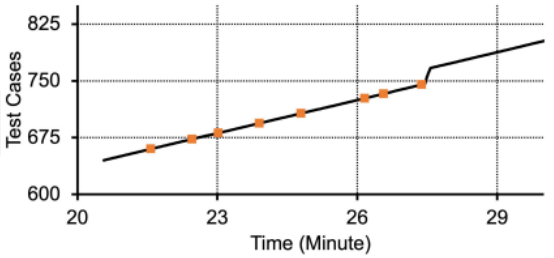
- (a)~(g) : 시간 경과에 따라 발견된 취약점과 Test 메시지 수에 따른 퍼징의 효율성 측정
- (h) : 21개의 디바이스 각각에 대한 퍼즈 테스트에 HubFuzzer가 걸린 총 시간



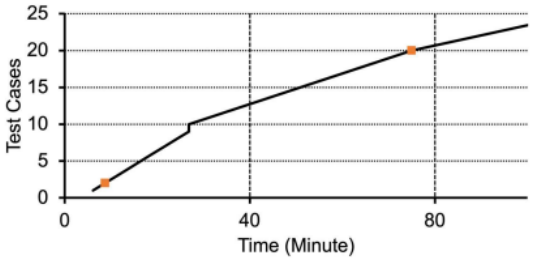
(a) Centralite Pearl Thermostat (ID = 1)



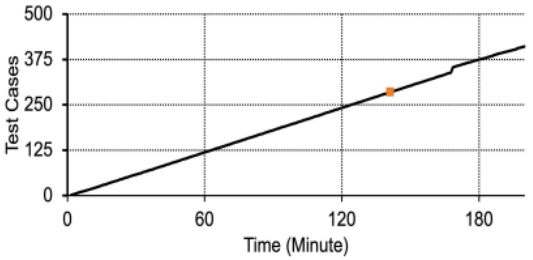
(b) Sengled Bulb E11-N1EAW (ID = 2)



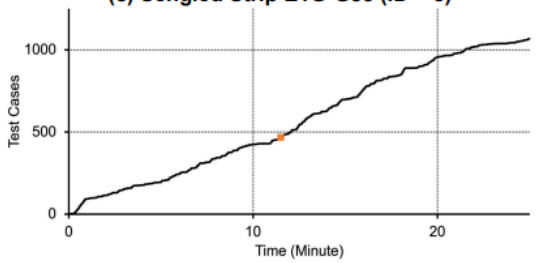
(c) Sengled Strip E1G-G85 (ID = 5)



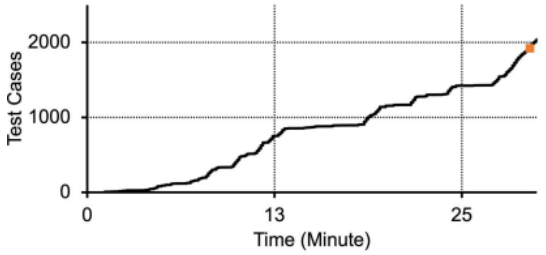
(d) Third Reality 3RSB015BZ (ID = 6)



(e) Sengled E1E-G7F (ID = 13)



(f) Aeotec ZW130-A (ID = 15)



(g) Fibaro FGMS-001 (ID = 18)

ID	Time	ID	Time	ID	Time	ID	Time	ID	Time	ID	Time	ID	Time
1	2.98 h	4	11.89 h	7	2.89 h	10	1.71 h	13	3.06 h	16	0.43 h	19	0.58 h
2	11.35 h	5	11.26 h	8	12.08 h	11	7.2 h	14	0.43 h	17	0.48 h	20	0.55 h
3	13.75 h	6	11.19 h	9	12.19 h	12	2.6 h	15	0.55 h	18	0.58 h	21	0.49 h

Zigbee 장치의 경우
가장 긴 Fuzz Test 시간 : 13.75h

(h) Total fuzzing time for all devices

Z-Wave에서 지원하는
명령이 적어
Test 시간 약 30분 소요

07 Discussion

- **Testing WiFi and Bluetooth Devices**
  - 디바이스는 홈킷과 호환되는 한 하나 이상의 로컬 제어 채널 지원하며 많은 IoT 공급업체가 홈킷을 지원
  - 홈킷 프로토콜에 따르면 홈킷 지원 장치는 홈킷 허브와 페어링할 뿐만 아니라, 장치 유형과 기능도 선언

  - **향후** : 홈키트 프레임워크 활용하여 WiFi 및 블루투스 장치를 테스트하기 위한 허브 구축할 것임

- **Testing Matter Devices**
  - Matter과 Thread는 홈 디바이스 상호 운용성과 연결성을 개선하기 위해 설계된 두 가지 업계 전반의 표준임

    Matter : 애플리케이션 계층에서의 상호 운용성을 향상시키기 위한 표준

    Thread : IoT 장치들 간의 하위 계층 통신을 위한 표준

  - **향후 :** Matter 장치 테스트에 허브 기반 퍼징 아이디어를 적용

08 Conclusion

- **기존 작업 : Companion App을 리버스 엔지니어링 하거나 테스트 스크립트를 수동으로 수집**
- **HubFuzzer : 퍼저가 스스로를 허브로 선언하여 IoT 디바이스 연결하는 허브 기반 IoT 퍼징 기법 제안**
  - Setting-up message를 활용하여 IoT Device가 지원하는 기능 발견하고 이를 이용하여 체계적인 기능 지향 퍼징 수행
  - Input sanitization에 제약을 받지 않으며, 암호화된 통신도 처리할 수 있음
- **21개의 인기 IoT 디바이스를 대상으로 평가 진행, 23개의 제로데이 취약점 발견 및 4개의 CVE 지정**

Q&A