IoTFuzzer: Discovering Memory Corruptions in IoT Through App-based Fuzzing

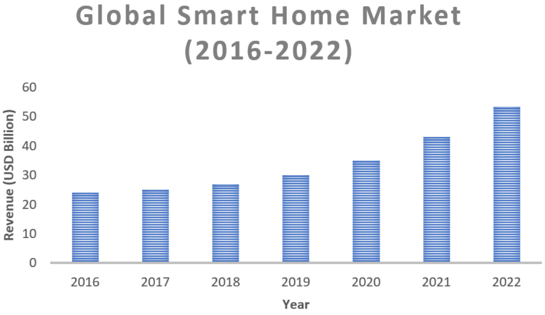
Jiongyi Chen[1], Wenrui Diao[2], Qingchuan Zhao[3], Chaoshun Zuo[3], Zhiqiang Lin[3,4], XiaoFeng Wang[5], Wing Cheong Lau[1], Menghan Sun[1], Ronghai Yang[1], Kehuan Zhang[1]

The Chinese University of Hong Kong[1], Jinan University[2], University of Texas at Dallas[3], The Ohio State University[4], Indiana University Bloomington[5]

Introduction

More and more IoT devices are entering the consumer market, forming a huge market:

- Connected "things" will reach 20.4 billion by 2020 [1]
- Global smart home market will rise to $53.45 billion by 2022



Source: Zion Research Analysis 2017

Introduction

- More than 90 independent IoT attack incidents have been reported from 2014 to 2016 [2]
- Examples: Mirai botnet, Reaper

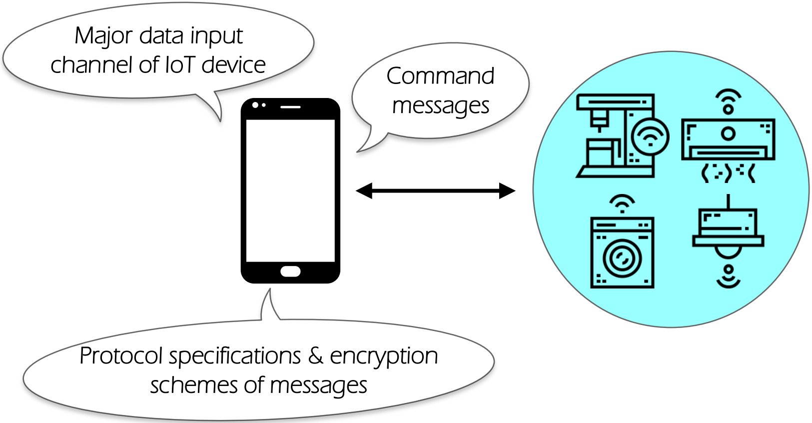
The firmware of IoT device is poorly implemented and loosely protected

Vulnerability Detection in IoT Devices

1. Firmware acquisition: vendors may not make their firmware images publicly available
2. Firmware identification and unpacking: unknown architectures, proprietary compression/encryption algorithms
3. Executable analysis:
   - Static analysis: disassembling errors, inaccurate points-to analysis, etc
   - Dynamic analysis: disabled debugging port, emulation problems for extracted program, etc

Motivation

- IoT official apps play an important role in controlling and managing IoT devices
- They contain rich information about IoT devices



IoTFuzzer

A firmware-free fuzzing framework that:

- aims at detecting memory corruptions in IoT devices
- utilizes program logic in official mobile apps of IoT to produce meaningful test messages
- fuzzes in a protocol-guided way without explicitly reverse engineering the protocol

Technical Challenges

```java
// Message construction
public final ControlResult a(...) {
...
Object localObject = new com/tplink/
    smarthome/b/e;
((e)localObject).<init>("system");
g localg = new com/tplink/smarthome/b/g;
localg.<init>("set_dev_location");
...
localg.a("longitude", localDouble);
localDouble = Double.valueOf(paramDouble1);
localg.a("latitude", localDouble);
...
return (ControlResult)localObject;
}
// Message: {"system":{"set_dev_location":{"
    longitude":10.111213141,"latitude
    ":51.617181920}}}

//Message encryption
public static byte[] a(byte[]
    paramArrayOfByte) {
    ...
    k = paramArrayOfByte[j];
    i = (byte)(i ^ k);
    paramArrayOfByte[j] = i;
    i = paramArrayOfByte[j];
    j += 1;
    ...
    return paramArrayOfByte;
}
```

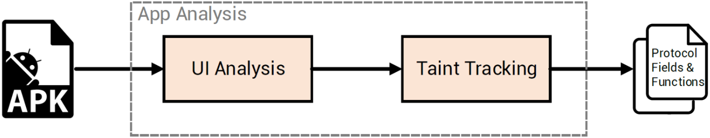
- Diverse protocols and formats (e.g., XML, JSON, key-value pairs)
- Use of homemade cryptographic functions
- Crash monitoring

Our Solutions

- Mutate protocol fields before they are constructed as a message
- Replay cryptographic functions in context
- Insert heartbeat messages

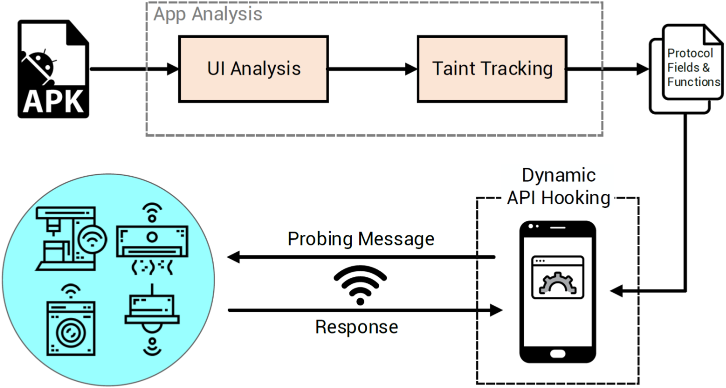
System Architecture

- Phase I: App Analysis
-



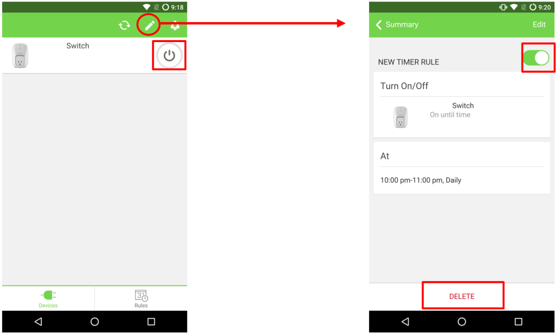
System Architecture

- Phase I: App Analysis
- Phase II: Fuzzing



Phase I: UI Analysis

- To identify networking UI elements, we construct code paths from networking APIs to UI event handlers
- To reach certain activities and trigger the network sending events, we interact with UI elements and record activity transitions.



Phase I: Taint Tracking

The goal is to identify protocol fields and the functions that the fields pass to

- Taint sources: strings, system APIs, user inputs
- Taint sinks: data uses at networking APIs and encryption functions

Taint Tracking Output Example

Example code:

```
// Message construction function
public final ControlResult a(...) {
...
Object localObject = new com/tplink/smarthome/b
    /e;
((e)localObject).<init>("system");
g localg = new com/tplink/smarthome/b/g;
localg.<init>("set_dev_location");
...
localg.a("longitude", localDouble);
localDouble = Double.valueOf(paramDouble1);
localg.a("latitude", localDouble);
...
return (ControlResult)localObject;
}
```

Taint tracking outputs:

```
com.tplink.smarthome.b.e.<init>(String)
com.tplink.smarthome.b.g.<init>(String)
com.tplink.smarthome.b.g.a(String, Object)
```

Phase II: Runtime Mutation

Hooked functions and mutated parameters in the example code:

```
com.tplink.smarthome.b.e.<init>(String)
com.tplink.smarthome.b.g.<init>(String)
com.tplink.smarthome.b.g.a(String, Object)
```

- Fuzzing scheduling: to only fuzz a subset of all fields
- Fuzzing policy:
  - Change the length of strings
  - Change the integer, double or float values
  - Change the types, or provide empty values

Phase II: Response Monitoring

- Response types:
  - Expected response
  - Unexpected response
  - No response
  - Disconnection
- Crash detection:
  - TCP-based connection: disconnection
  - UDP-based connection: inserting heartbeat messages during fuzzing to confirm the status of IoT devices

Evaluation

We selected 17 products of different categories offered by mainstream manufacturers

Device Type	Vendor	Device Model	Protocol and Format	Encryption?
IP Camera	D-Link	DCS-5010L	HTTP, K-V Pairs	No
Smart Bulb	TP-Link	LB100	UDP, JSON	Yes
	KONKE	KK-Light	UDP, String	Yes
Smart Plug	Belkin	WeMo Switch	HTTP, XML	No
	TP-Link	HS110	TCP, JSON	Yes
	D-Link	DSP-W215	HNAP, XML	No
Printer	Brother	HL-L5100DN	LPD & HTTP	No
NAS	Western Digital	My Passport Pro	HTTP, JSON	No
		My Cloud	HTTP, JSON	No
	QNAP	TS-212P	HTTP, K-V Pairs	No
IoT Hub	Philips	Hue Bridge	HTTP, JSON	No
Home Router	NETGEAR	N300	HTTP, XML	No
	Linksys	E1200	HNAP, XML	No
	Xiaomi	Xiaomi Router	HTTP, K-V Pairs	No
Story Teller	Xiaomi	C-1	UDP, JSON	Yes

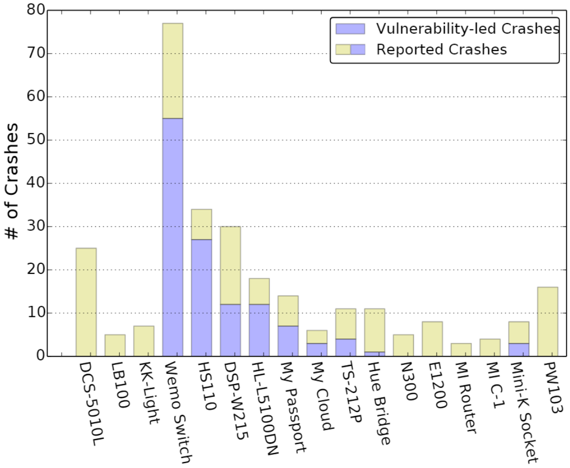
Evaluation

15 memory corruptions were discovered (including 8 zero-days)

Device	**Vulnerability Type**	**# of Issues**
Belkin WeMo (Switch)	Null Pointer Dereference	1
TP-Link HS110 (Plug)	Null Pointer Dereference	3
D-Link DSP-W215 (Plug)	Buffer Overflow (Stack-based)	4
WD My Cloud (NAS)	Buffer Overflow (Stack-based)	1
QNAP TS-212P (NAS)	Buffer Overflow (Heap-based)	2
Brother HL-L5100DN (Printer)	Unknown Crash	1
Philips Hue Bridge (Hub)	Unknown Crash	1
WD My Passport Pro (NAS)	Unknown Crash	1
POVOS PW103 (Humidifier)	Unknown Crash	1

Evaluation

Crashes reported by IoTFuzzer v.s. Vulnerability-led crash



Evaluation

Comparison with two popular fuzzers

Vulnerability	Device	IoTFuzzer	Sulley	BED
Null Dereference 1	TP-Link HS110	0.71 h (2517)	NA	NA
Null Dereference 2	TP-Link HS110	1.56 h (7068)	NA	NA
Null Dereference 3	TP-Link HS110	4.38 h (14839)	NA	NA
Null Dereference 4	Belkin WeMo	19.52 h (62424)	>24 h (309985)	>24 h (30274)
Buffer Overflow 1 (Stack-based)	D-Link DSP-W215	3.22 h (9392)	>24 h (314297)	>24 h (28131)
Buffer Overflow 2 (Stack-based)	D-Link DSP-W215	3.34 h (14696)	>24 h (314297)	>24 h (28131)
Buffer Overflow 3 (Stack-based)	D-Link DSP-W215	4.50 h (11110)	>24 h (314297)	0.87 h (1249)
Buffer Overflow 4 (Stack-based)	D-Link DSP-W215	10.85 h (42478)	>24 h (314297)	>24 h (28131)
Buffer Overflow 5 (Stack-based)	WD My Cloud	5.49 h (20323)	>24 h (333255)	>24 h (28493)
Buffer Overflow 6 (Heap-based)	QNAP TS-212P	2.95 h (10068)	>24 h (286552)	>24 h (29319)
Buffer Overflow 7 (Heap-based)	QNAP TS-212P	3.27 h (11811)	>24 h (286552)	>24 h (29319)
Crash 1	Brother HL-L5100DN	0.23 h (1021)	0.15 h (2034)	0.21 h (359)
Crash 2	Philips Hue Bridge	1.70 h (7415)	>24 h (308424)	>24 h (31810)
Crash 3	WD My Passport Pro	3.24 h (11016)	>24 h (323848)	0.28 h (453)
Crash 4	POVOS PW103	4.11 h (12832)	NA	NA

NA: not applicable for encrypted messages; >24 h: not found in 24 hours

Evaluation

Comparison with two popular fuzzers

Vulnerability	Device	IoTFuzzer	Sulley	BED
Null Dereference 1	TP-Link HS110	0.71 h (2517)	NA	NA
Null Dereference 2	TP-Link HS110	1.56 h (7068)	NA	NA
Null Dereference 3	TP-Link HS110	4.38 h (14839)	NA	NA
Null Dereference 4	Belkin WeMo	19.52 h (62424)	>24 h (309985)	>24 h (30274)
Buffer Overflow 1 (Stack-based)	D-Link DSP-W215	3.22 h (9392)	>24 h (314297)	>24 h (28131)
Buffer Overflow 2 (Stack-based)	D-Link DSP-W215	3.34 h (14696)	>24 h (314297)	>24 h (28131)
Buffer Overflow 3 (Stack-based)	D-Link DSP-W215	4.50 h (11110)	>24 h (314297)	0.87 h (1249)
Buffer Overflow 4 (Stack-based)	D-Link DSP-W215	10.85 h (42478)	>24 h (314297)	>24 h (28131)
Buffer Overflow 5 (Stack-based)	WD My Cloud	5.49 h (20323)	>24 h (333255)	>24 h (28493)
Buffer Overflow 6 (Heap-based)	QNAP TS-212P	2.95 h (10068)	>24 h (286552)	>24 h (29319)
Buffer Overflow 7 (Heap-based)	QNAP TS-212P	3.27 h (11811)	>24 h (286552)	>24 h (29319)
Crash 1	Brother HL-L5100DN	0.23 h (1021)	0.15 h (2034)	0.21 h (359)
Crash 2	Philips Hue Bridge	1.70 h (7415)	>24 h (308424)	>24 h (31810)
Crash 3	WD My Passport Pro	3.24 h (11016)	>24 h (323848)	0.28 h (453)
Crash 4	POVOS PW103	4.11 h (12832)	NA	NA

NA: not applicable for encrypted messages; >24 h: not found in 24 hours

Evaluation

Comparison with two popular fuzzers

Vulnerability	**Device**	IoTFuzzer	**Sulley**	**BED**
Null Dereference 1	TP-Link HS110	0.71 h (2517)	NA	NA
Null Dereference 2	TP-Link HS110	1.56 h (7068)	NA	NA
Null Dereference 3	TP-Link HS110	4.38 h (14839)	NA	NA
Null Dereference 4	Belkin WeMo	19.52 h (62424)	>24 h (309985)	>24 h (30274)
Buffer Overflow 1 (Stack-based)	D-Link DSP-W215	3.22 h (9392)	>24 h (314297)	>24 h (28131)
Buffer Overflow 2 (Stack-based)	D-Link DSP-W215	3.34 h (14696)	>24 h (314297)	>24 h (28131)
Buffer Overflow 3 (Stack-based)	D-Link DSP-W215	4.50 h (11110)	>24 h (314297)	0.87 h (1249)
Buffer Overflow 4 (Stack-based)	D-Link DSP-W215	10.85 h (42478)	>24 h (314297)	>24 h (28131)
Buffer Overflow 5 (Stack-based)	WD My Cloud	5.49 h (20323)	>24 h (333255)	>24 h (28493)
Buffer Overflow 6 (Heap-based)	QNAP TS-212P	2.95 h (10068)	>24 h (286552)	>24 h (29319)
Buffer Overflow 7 (Heap-based)	QNAP TS-212P	3.27 h (11811)	>24 h (286552)	>24 h (29319)
Crash 1	Brother HL-L5100DN	0.23 h (1021)	0.15 h (2034)	0.21 h (359)
Crash 2	Philips Hue Bridge	1.70 h (7415)	>24 h (308424)	>24 h (31810)
Crash 3	WD My Passport Pro	3.24 h (11016)	>24 h (323848)	0.28 h (453)
Crash 4	POVOS PW103	4.11 h (12832)	NA	NA

NA: not applicable for encrypted messages; >24 h: not found in 24 hours

Evaluation

Comparison with two popular fuzzers

Vulnerability	Device	IoTFuzzer	Sulley	BED
Null Dereference 1	TP-Link HS110	0.71 h (2517)	NA	NA
Null Dereference 2	TP-Link HS110	1.56 h (7068)	NA	NA
Null Dereference 3	TP-Link HS110	4.38 h (14839)	NA	NA
Null Dereference 4	Belkin WeMo	19.52 h (62424)	>24 h (309985)	>24 h (30274)
Buffer Overflow 1 (Stack-based)	D-Link DSP-W215	3.22 h (9392)	>24 h (314297)	>24 h (28131)
Buffer Overflow 2 (Stack-based)	D-Link DSP-W215	3.34 h (14696)	>24 h (314297)	>24 h (28131)
Buffer Overflow 3 (Stack-based)	D-Link DSP-W215	4.50 h (11110)	>24 h (314297)	0.87 h (1249)
Buffer Overflow 4 (Stack-based)	D-Link DSP-W215	10.85 h (42478)	>24 h (314297)	>24 h (28131)
Buffer Overflow 5 (Stack-based)	WD My Cloud	5.49 h (20323)	>24 h (333255)	>24 h (28493)
Buffer Overflow 6 (Heap-based)	QNAP TS-212P	2.95 h (10068)	>24 h (286552)	>24 h (29319)
Buffer Overflow 7 (Heap-based)	QNAP TS-212P	3.27 h (11811)	>24 h (286552)	>24 h (29319)
Crash 1	Brother HL-L5100DN	0.23 h (1021)	0.15 h (2034)	0.21 h (359)
Crash 2	Philips Hue Bridge	1.70 h (7415)	>24 h (308424)	>24 h (31810)
Crash 3	WD My Passport Pro	3.24 h (11016)	>24 h (323848)	0.28 h (453)
Crash 4	POVOS PW103	4.11 h (12832)	NA	NA

NA: not applicable for encrypted messages; >24 h: not found in 24 hours

Limitations and Future Work

- Device acquisition: require physical IoT devices
- Connection mode: only support local Wi-Fi connection
- Code coverage: can only fuzz app-related code in IoT devices
- Crash detection: only detect memory corruptions that cause program to crash

Summary

- We built a firmware-free fuzzing framework for IoT devices based on mobile apps
- We developed several new techniques, such as protocol-guided fuzzing without protocol specifications and in-context cryptographic and network function replay
- By conducting experiments in real environment, we identified 15 memory corruptions in 17 IoT devices with IoTFuzzer

Q & A

Thank you!

References

[1]. Gartner, “Internet of Things (IoT) Market,” https://www.gartner.com/ newsroom/id/3598917, February 2017

[2]. N. Zhang, S. Demetriou, X. Mi, W. Diao, K. Yuan, P. Zong, F. Qian, X. Wang, K. Chen, Y. Tian, C. A. Gunter, K. Zhang, P. Tague, and Y. Lin, “Understanding IoT Security Through the Data Crystal Ball: Where We Are Now and Where We Are Going to Be,” CoRR, vol. abs/1703.09809, 2017.