DIANE: Identifying Fuzzing Triggers in Apps to Generate Under-constrained Inputs for IoT Devices

Nilo Redini[+], Andrea Continella[*], Dipanjan Das[+], Giulio De Pasquale[+], Noah Spahn[+], **Aravind Machiry**[∓], Antonio Bianchi[∓], Christopher Kruegel[+], Giovanni Vigna[+]

[+]UC Santa Barbara, [*]University of Twente, [∓]Purdue University

UNIVERSITY OF TWENTE.

PURDUE UNIVERSITY

Motivation

Classic fuzzing approaches are inefficient to uncover bugs in IoT devices

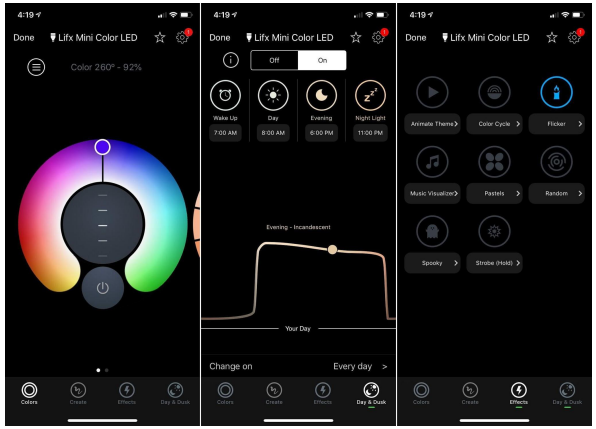
- Gray-box techniques require access to the fuzzed program
- Black-box techniques require knowledge about the data format accepted by the device (**network fuzzers**)
- Emulation is an open problem

IoTFuzzer's Idea

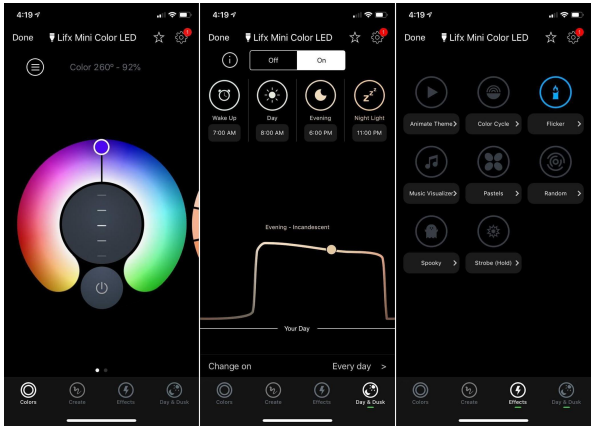
Use the companion apps to create fuzzing inputs for IoT devices [1]

[1] IOTFUZZER: Discovering Memory Corruptions in IoT Through App-based Fuzzing, NDSS 2018

IoTFuzzer's Idea



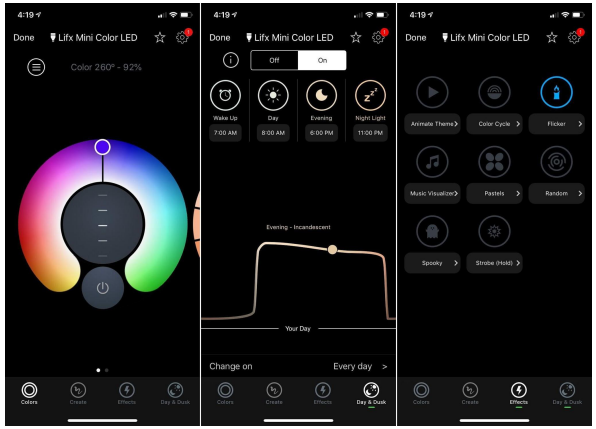
IoTFuzzer's Idea



```
public void getBrFromUI(String val) {
     // ...
     process_brightness(val);
}

public void process_brightness(String msg) {
    byte[] cnt = encode(msg);
    send_to_device(cnt);
}
```

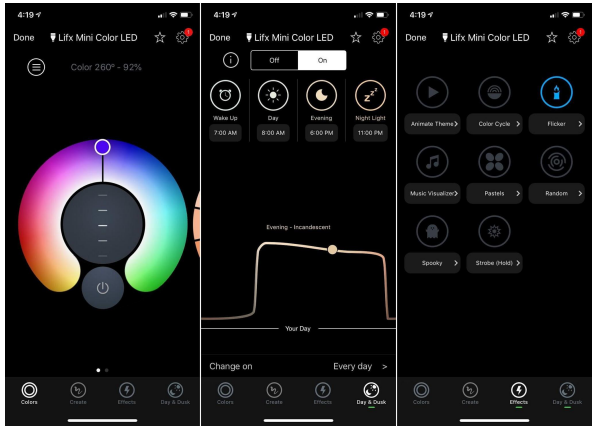
IoTFuzzer's Idea




```
public void getBrFromUI(String val) {
     // ...
     process_brightness(val);
}

public void process_brightness(String msg) {
    byte[] cnt = encode(msg);
    send_to_device(cnt);
}
```

IoTFuzzer's Idea



```
public void getBrFromUI(String val) {
      // ...
      process_brightness(val);
}

public void process_brightness(String msg) {
     byte[] cnt = encode(msg);
     send_to_device(cnt);
}
```

IoTFuzzer's Idea

IoTFuzzer

- Finds UI elements that generate network traffic
- Finds functions that retrieve data from UI
- Fuzzes functions' arguments containing UI data

```
public void getBrFromUI(String val) {
    // ...
    process_brightness(val);
}
```

IoTFuzzer's Idea

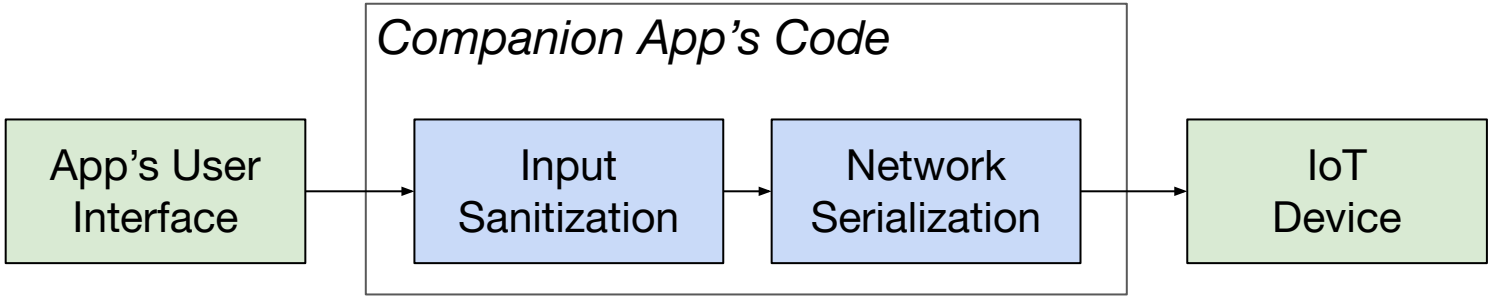
IoTFuzzer

- Finds UI elements that generate network traffic
- Finds functions that retrieve data from UI
- Fuzzes functions' arguments containing UI data

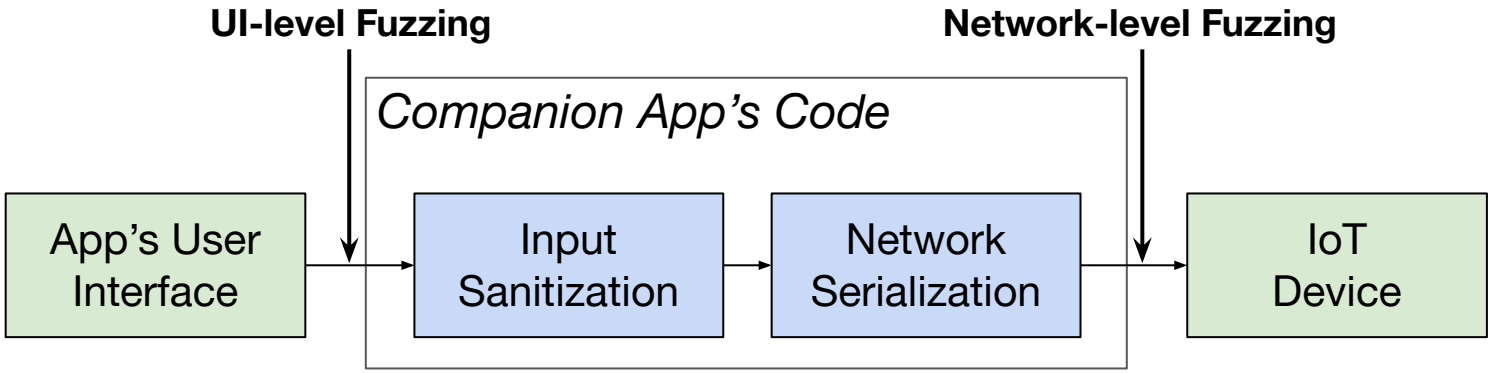
UI-level fuzzer

```
public void getBrFromUI(String val) {
    // ...
    process_brightness(val);
}
```

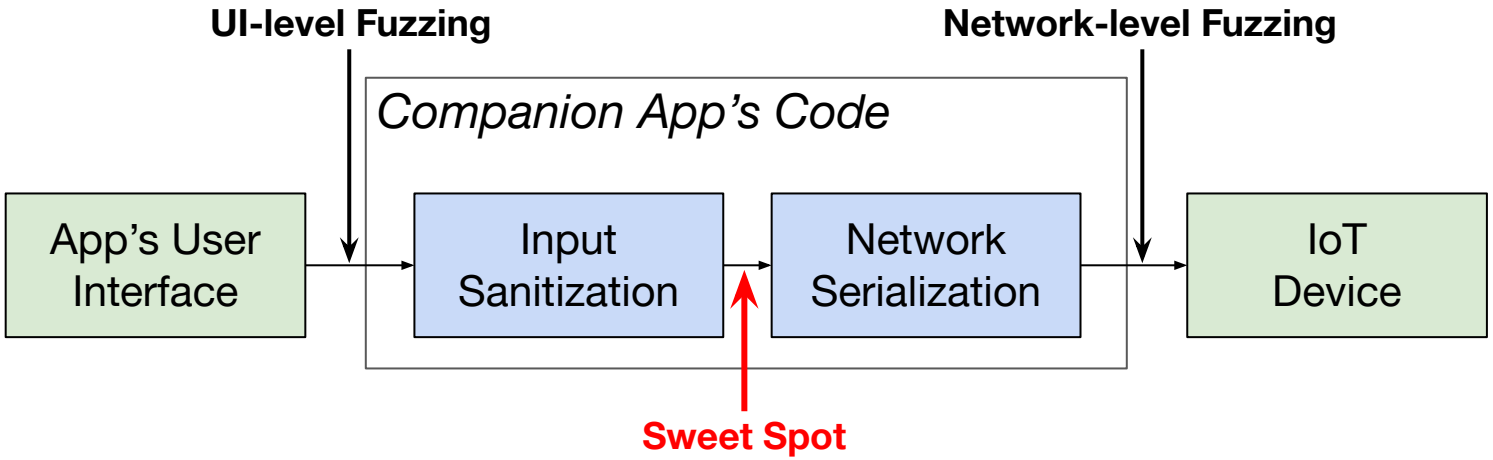
Companion Apps



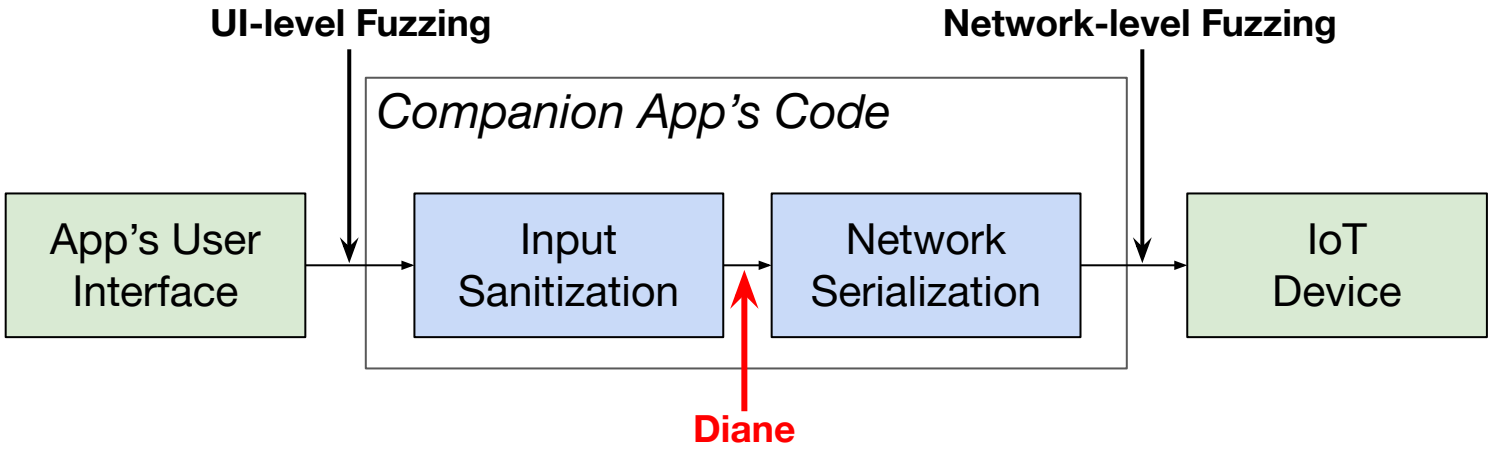
Companion Apps



Companion Apps



Companion Apps



	UI-level Fuzzing	Network-level Fuzzing	Diane
Well-formatted input	✓	✗	✓
Not constrained by input sanitization	✗	✓	✓

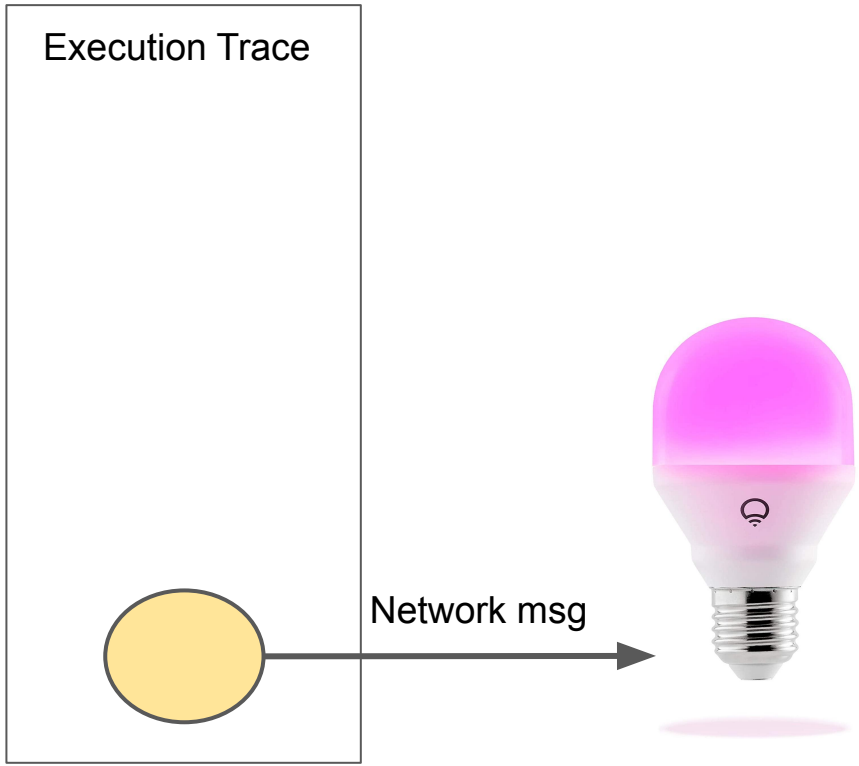
Diane

Bottom-Up approach

Diane

Bottom-Up approach

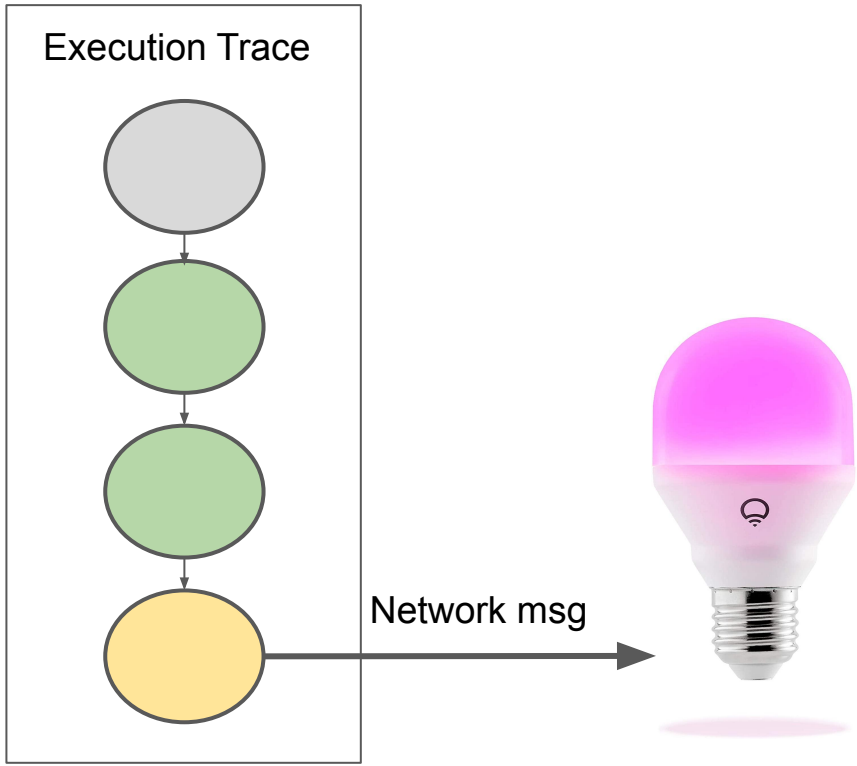
1. Find functions that *send messages* to the IoT device (**send-message**)



Diane

Bottom-Up approach

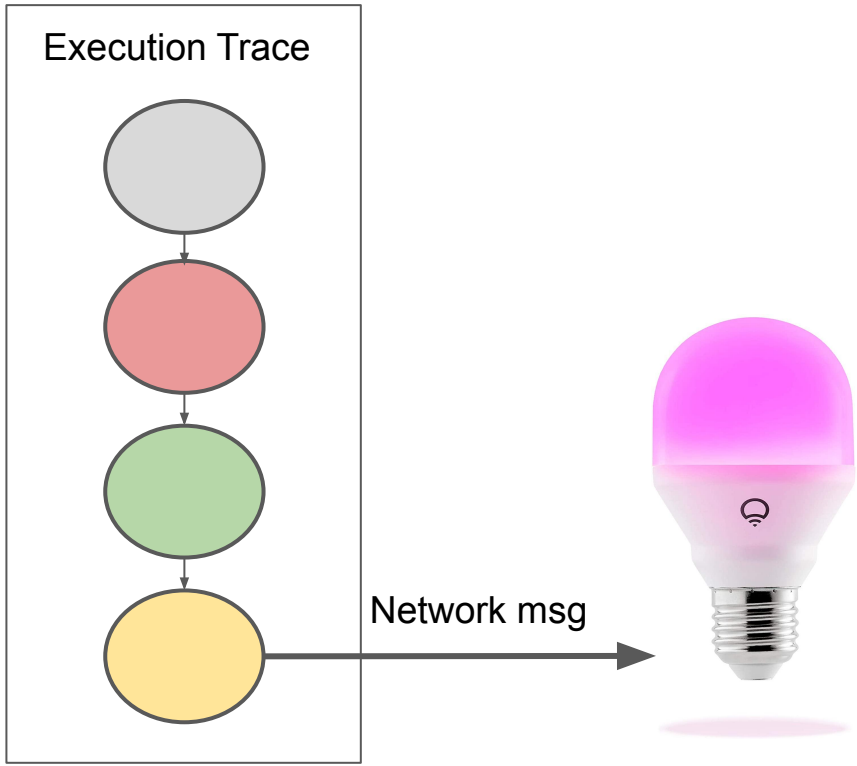
1. Find functions that *send messages* to the IoT device (**send-message**)
2. Find functions that *transform* the sent data (**data-transforming**)



Diane

Bottom-Up approach

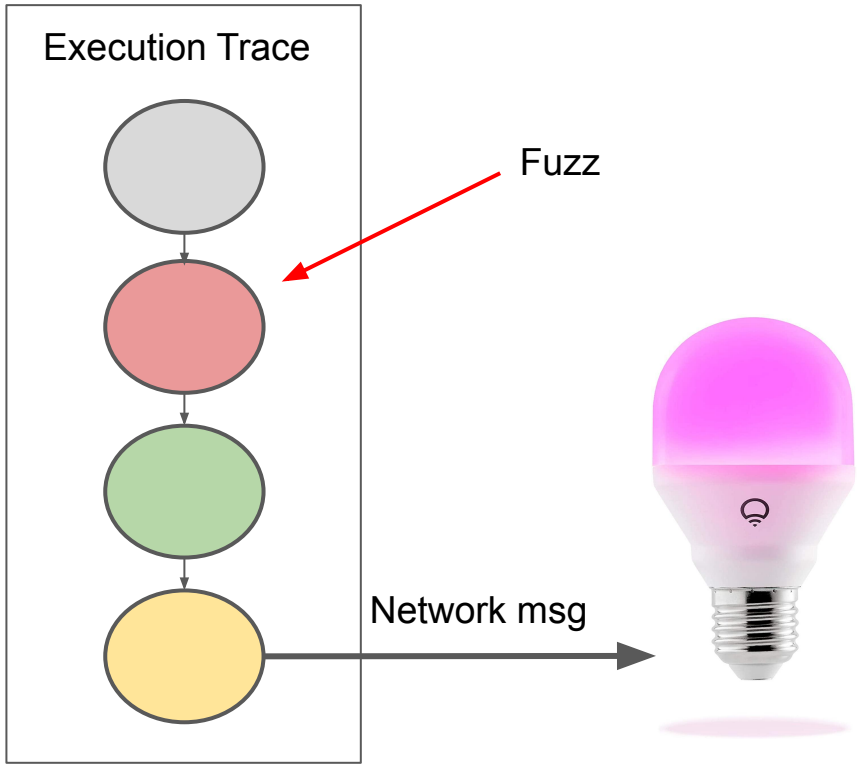
1. Find functions that *send messages* to the IoT device (**send-message**)
2. Find functions that *transform* the sent data (**data-transforming**)
3. Find the top data-transforming functions (**fuzzing triggers**)



Diane

Bottom-Up approach

1. Find functions that *send messages* to the IoT device (**send-message**)
2. Find functions that *transform* the sent data (**data-transforming**)
3. Find the top data-transforming functions (**fuzzing triggers**)
4. Fuzz fuzzing triggers and monitor device's responses



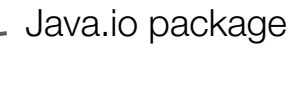
Send-Message Functions

Send-Message Functions

Intuitions

1. Border functions that sit in between the app's code and JNI/Android/Java framework

```
public void sendToDevice(byte[] en) {
     // ...
     outputStream.write(en);
}
```

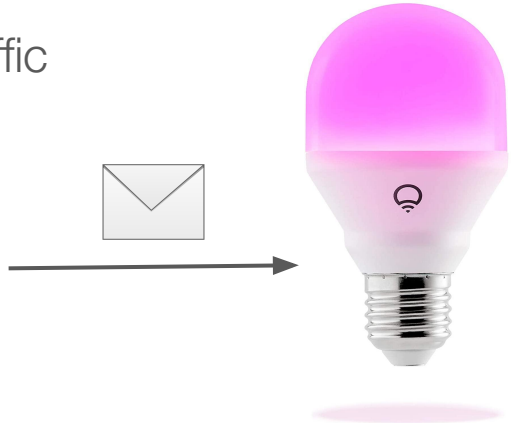


Send-Message Functions

Intuitions

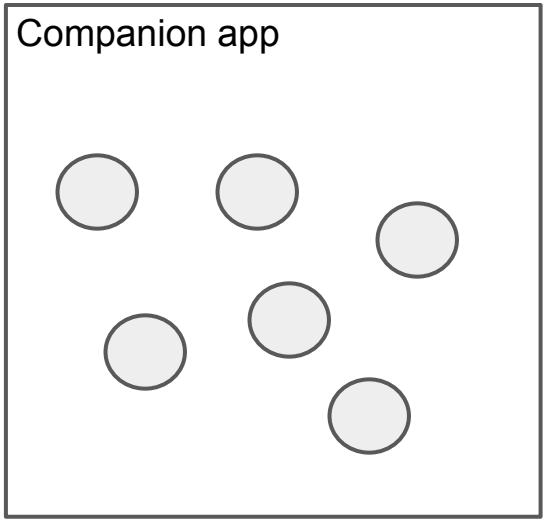
1. Border functions that sit in between the app's code and JNI, Android, or Java framework
2. Functions that, when invoked, generate network traffic

```
public void sendToDevice(byte[] en) {
     // ...
     outputStream.write(en);
}
```



Send-Message Functions

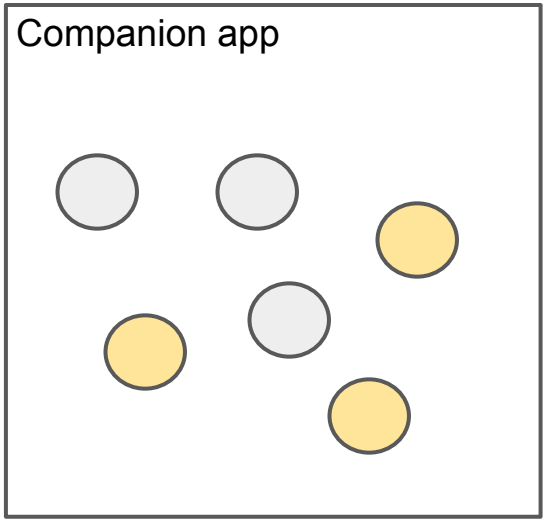
Consider the companion app



Send-Message Functions

Consider the companion app

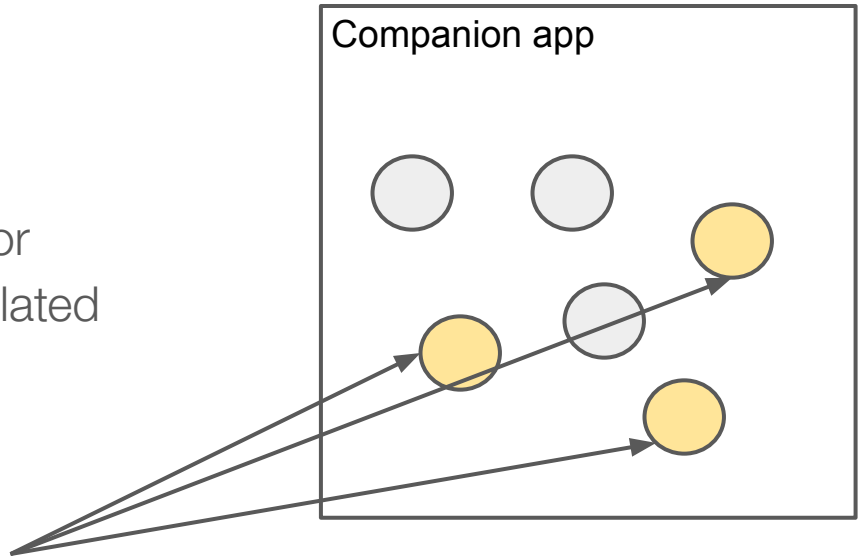
- Static analysis to find all border functions
  - Functions that invoke JNI or Java/Android's network-related functions



Send-Message Functions

Consider the companion app

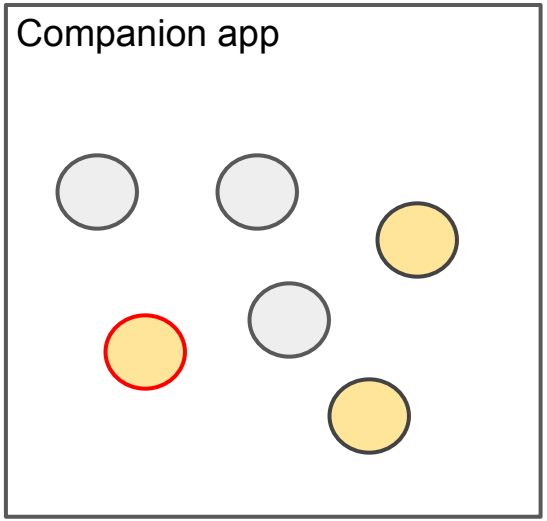
- Static analysis to find all border functions
  - Functions that invoke JNI or Java/Android's network-related functions



Send-Message Functions

Consider the companion app

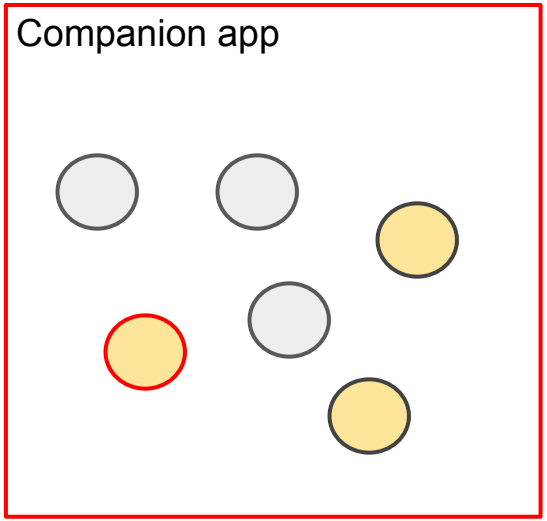
- Dynamically hook each of them



Send-Message Functions

Consider the companion app

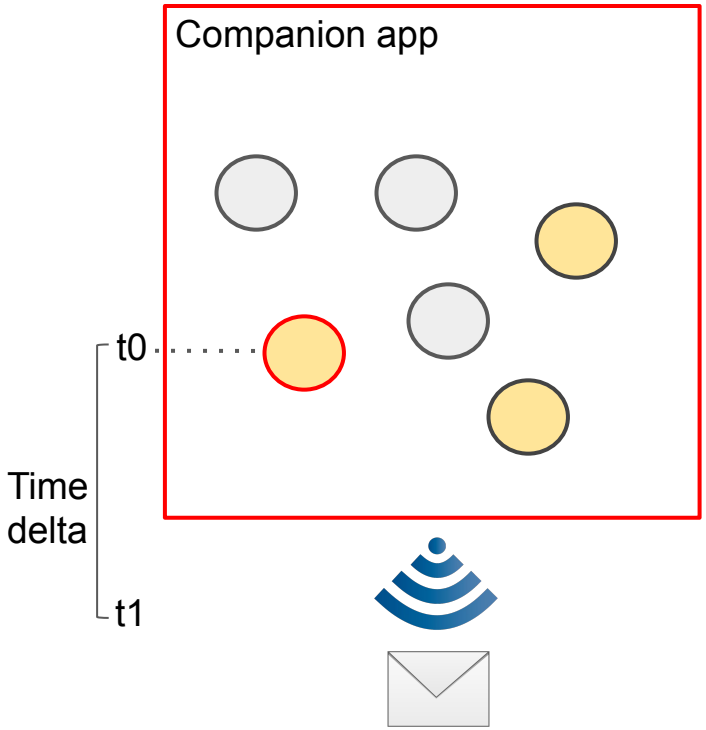
- Stimulate the app's UI
  - We ask the user to interact with the device and record the interaction



Send-Message Functions

Consider the companion app

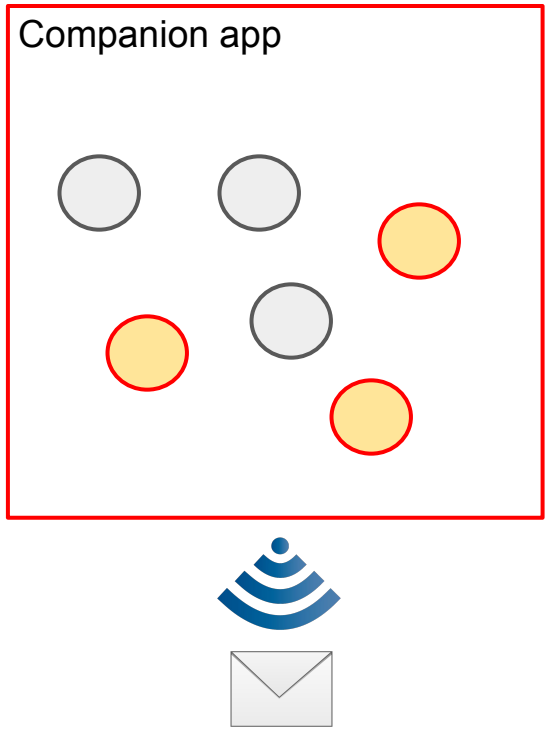
- Register the delta between function invocation and traffic generated by the app



Send-Message Functions

Consider the companion app

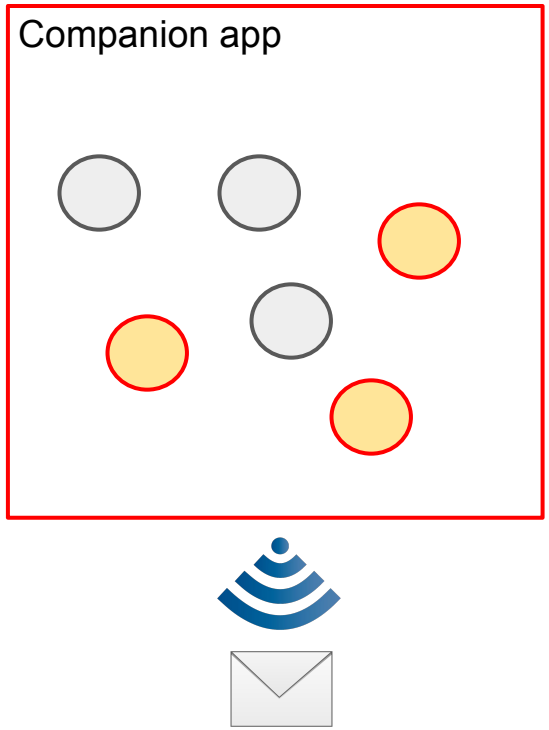
- Repeat the procedure N times for all border functions, and calculate mean, standard deviation, and mode
  - N sets to 10 in our experiments



Send-Message Functions

Consider the companion app

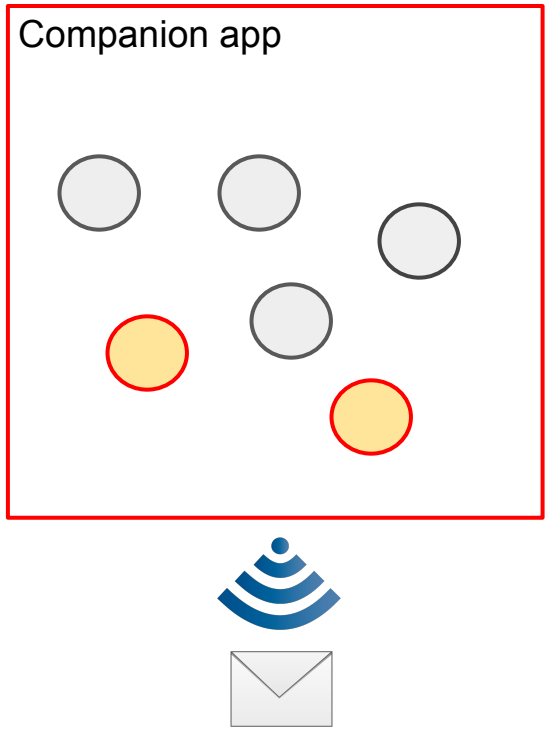
- Apply machine learning
  - K-mean clustering algorithm



Send-Message Functions

Consider the companion app

- Consider send-message functions those in the cluster with smallest means



Data-Transforming Functions

Data-Transforming Functions

Functions that transform user data in the format accepted by the IoT device

Data-Transforming Functions

Observation

- Data-transforming functions increase entropy of data (e.g., base64encode) [1]

[1] IOTFUZZER: Discovering Memory Corruptions in IoT Through App-based Fuzzing, NDSS 2018

Data-Transforming Functions

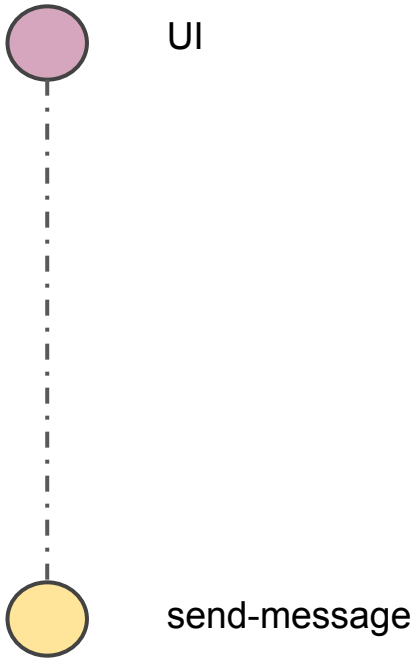
Starting from the send-message functions

send-message

Data-Transforming Functions

Starting from the send-message functions

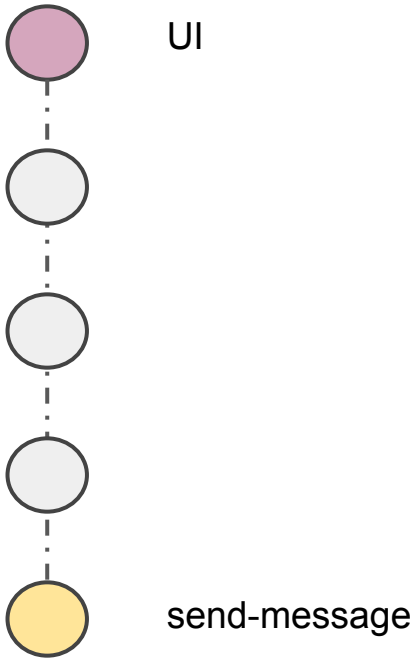
- Perform a backward slice up to the UI
  - Consider send-message function's arguments



Data-Transforming Functions

Starting from the send-message functions

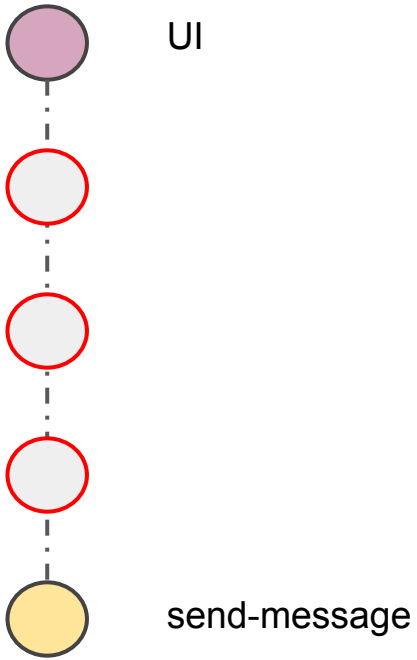
- Perform a backward slice up to the UI
- Identify traversed functions



Data-Transforming Functions

Starting from the send-message functions

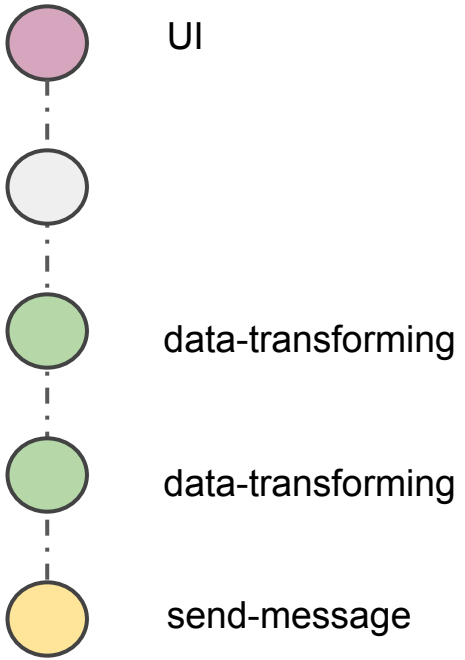
- Perform a backward slice up to the UI
- Identify traversed functions
- Dynamically hook functions and calculate the introduced Shannon entropy



Data-Transforming Functions

Starting from the send-message functions

- Perform a backward slice up to the UI
- Identify traversed functions
- Dynamically hook functions and calculate the introduced Shannon entropy
- Consider data-transforming functions those whose **introduced** entropy >= T
  - T set to 2 [3]

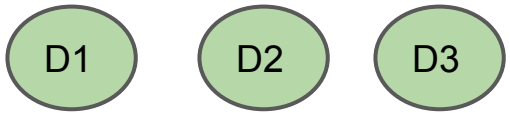


[3] Reformat: Automatic reverse engineering of encrypted messages

Fuzzing Triggers

Fuzzing Triggers

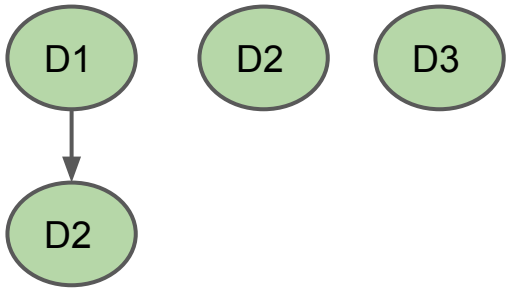
For each data-transforming function



Fuzzing Triggers

For each data-transforming function

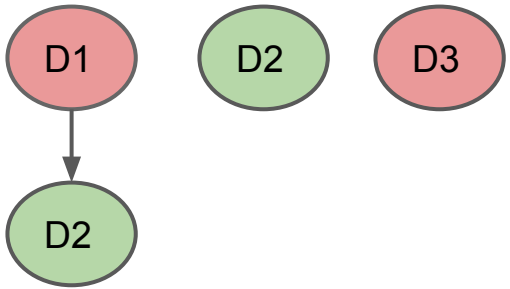
- Build the dominance tree



Fuzzing Triggers

For each data-transforming function

- Build the dominance tree
- Identify data-transforming functions that are not dominated by other data-transforming functions (**fuzzing triggers**)



Example

```
public void setDeviceName(String oname) { // UI
     String name = substring(oname, 15);
     setDeviceInternal(name);
}

public byte[] encode(String s) {
     byte[] enc;
     // encode cmd
     return enc;
}

public void setDeviceInternal(String name) {
     byte[] e = encode(name);
     return sendToDevice(e);
}

public void sendToDevice(byte[] c) { /* … */ }
```

Example

```
public void setDeviceName(String oname) { // UI
     String name = substring(oname, 15);
     setDeviceInternal(name);
}

public byte[] encode(String s) {
     byte[] enc;
     // encode cmd
     return enc;
}

public void setDeviceInternal(String name) {
     byte[] e = encode(name);
     return sendToDevice(e);
}

public void sendToDevice(byte[] c) { /* … */ }
```

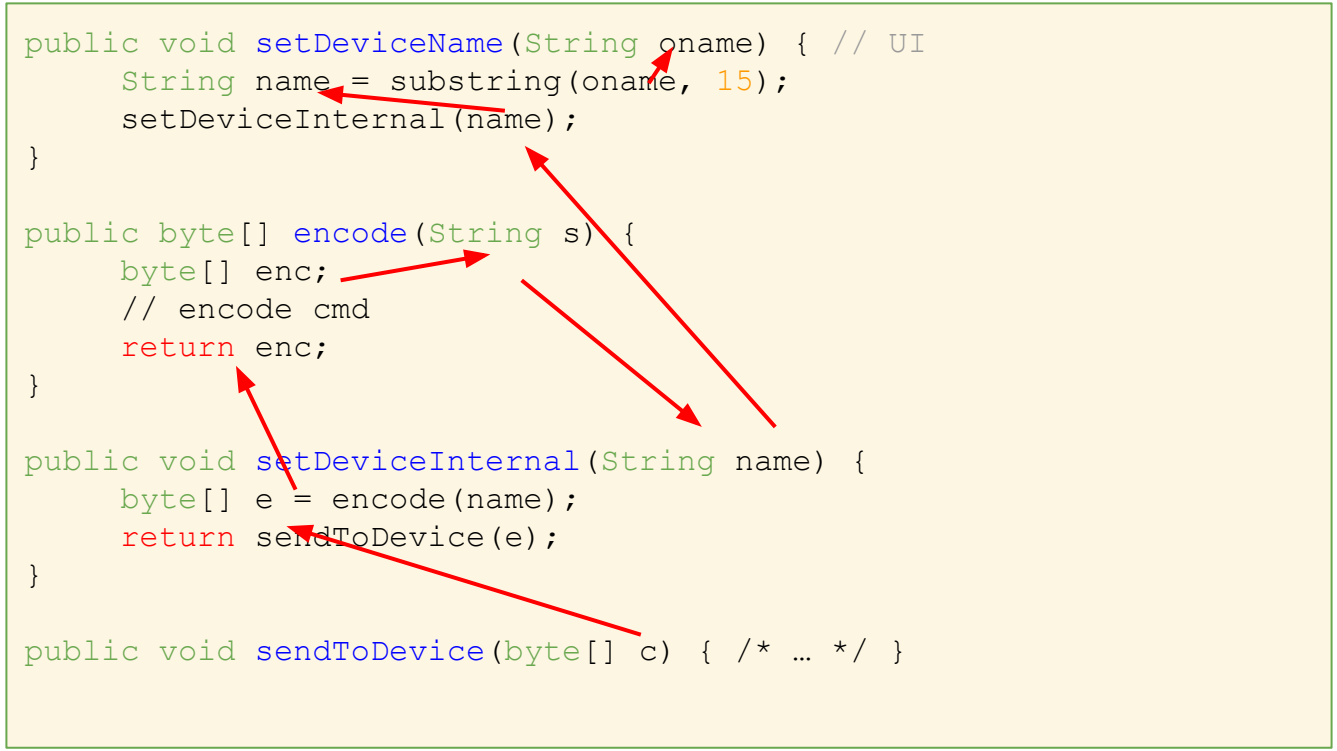
Example

```
public void setDeviceName(String oname) { // UI
     String name = substring(oname, 15);
     setDeviceInternal(name);
}

public byte[] encode(String s) {
     byte[] enc;
     // encode cmd
     return enc;
}

public void setDeviceInternal(String name) {
     byte[] e = encode(name);
     return sendToDevice(e);
}

public void sendToDevice(byte[] c) { /* … */ }
```



Example

```
public void setDeviceName(String oname) { // UI
     String name = substring(oname, 15);
     setDeviceInternal(name);
}

public byte[] encode(String s) {
     byte[] enc;
     // encode cmd
     return enc;
}

public void setDeviceInternal(String name) {
     byte[] e = encode(name);
     return sendToDevice(e);
}

public void sendToDevice(byte[] c) { /* … */ }
```

Example

```
public void setDeviceName(String oname) { // UI
     String name = substring(oname, 15);
     setDeviceInternal(name);
}

public byte[] encode(String s) {
     byte[] enc;
     // encode cmd
     return enc;
}

public void setDeviceInternal(String name) {
     byte[] e = encode(name);
     return sendToDevice(e);
}

public void sendToDevice(byte[] c) { /* … */ }
```

Entropy < T

Entropy > T

Entropy < T

Example

```
public void setDeviceName(String oname) { // UI
     String name = substring(oname, 15);
     setDeviceInternal(name);
}

public byte[] encode(String s) {
     byte[] enc;
     // encode cmd
     return enc;
}

public void setDeviceInternal(String name) {
     byte[] e = encode(name);
     return sendToDevice(e);
}

public void sendToDevice(byte[] c) { /* … */ }
```

Data-transforming function

Example

```
public void setDeviceName(String oname) { // UI
     String name = substring(oname, 15);
     setDeviceInternal(name);
}

public byte[] encode(String s) {
     byte[] enc;
     // encode cmd
     return enc;
}

public void setDeviceInternal(String name) {
     byte[] e = encode(name);
     return sendToDevice(e);
}

public void sendToDevice(byte[] c) { /* … */ }
```

Fuzzing Trigger

Fuzzing and Response Monitoring

Fuzzing

Hook the fuzzing triggers and run the companion app

- When a fuzzing trigger is invoked we mutate the set of its **input** variables
  - Fuzz both primitive variables and objects

Response Monitoring

Muench et al. showed that detecting bugs in IoT devices is a hard problem [4]

[4] What You Corrupt Is Not What You Crash: Challenges in Fuzzing Embedded Devices, NDSS 2018

Response Monitoring

Before starting a fuzzing campaign

- We register the amount of traffic generated by a clean run of the app
- Ping the device at regular intervals (**heartbeat monitoring**)

While fuzzing, we monitor the network traffic, and raise an alert if

- The device generated significant less traffic (≤ 50%)
- There is a significant delay in the device's responses (10 s)

Evaluation

Evaluation

We evaluated Diane against 11 IoT devices

- **E1**: Ability of Diane to find fuzzing triggers
- **E2**: Ability of Diane to find bugs
- **E3**: Performance against related work (both UI-level and network-level fuzzers)

Evaluation: Devices

Device ID	Type	Vendor	Model	Firmware Vers.	Android App Package Name	App Vers.	Online Account*	Setup Time [Seconds]
1	Camera	Wansview	720P X Series WiFi	00.20.01	wansview.p2pwificam.client	1.0.10	✗	219
2	Camera	Insteon	HD Wifi Camera	2.2.200	com.insteon.insteon3	1.9.8	✓	427
3	Smart Socket	TP-Link	HS110	1.2.5	com.tplink.kasa_android	2.2.0.784	✗	311
4	Camera	FOSCAM	FI9821P	1.5.3.16	com.foscam.foscam	2.1.8	✓	406
5	Camera	FOSCAM	FI9831P	1.5.3.19	com.foscam.foscam	2.1.8	✓	403
6	Smart Socket	Belkin	Wemo Smart Socket	2.0.0	com.belkin.wemoandroid	1.20	✗	211
7	Bulb	iDevices	IDEV0002	1.9.4	com.idevicesllc.connected	1.6.95	✗	274
8	Smart Socket	iDevices	IDEV0001	1.9.4	com.idevicesllc.connected	1.6.95	✗	276
9	Camera	Belkin	NetCam	Unknown	com.belkin.android.androidbelkinnetcam	2.0.4	✓	1,040
10	Bulb	LIFX	Z	2.76	com.lifx.lifx	3.9.0	✓	313
11	Smart Lock	August	August Smart Lock	1.12.6	com.august.luna	8.3.13	✓	213

Evaluation: Devices

Device ID	Type	Vendor	Model	Firmware Vers.	Android App Package Name	App Vers.	Online Account*	Setup Time [Seconds]
1	Camera	Wansview	720P X Series WiFi	00.20.01	wansview.p2pwificam.client	1.0.10	✗	219
2	Camera	Insteon	HD Wifi Camera	2.2.200	com.insteon.insteon3	1.9.8	✓	427
3	Smart Socket	TP-Link	HS110	1.2.5	com.tplink.kasa_android	2.2.0.784	✗	311
4	Camera	FOSCAM	FI9821P	1.5.3.16	com.foscam.foscam	2.1.8	✓	406
5	Camera	FOSCAM	FI9831P	1.5.3.19	com.foscam.foscam	2.1.8	✓	403
6	Smart Socket	Belkin	Wemo Smart Socket	2.0.0	com.belkin.wemoandroid	1.20	✗	211
7	Bulb	iDevices	IDEV0002	1.9.4	com.idevicesllc.connected	1.6.95	✗	274
8	Smart Socket	iDevices	IDEV0001	1.9.4	com.idevicesllc.connected	1.6.95	✗	276
9	Camera	Belkin	NetCam	Unknown	com.belkin.android.androidbelkinnetcam	2.0.4	✓	1,040
10	Bulb	LIFX	Z	2.76	com.lifx.lifx	3.9.0	✓	313
11	Smart Lock	August	August Smart Lock	1.12.6	com.august.luna	8.3.13	✓	213

E1: Fuzzing Triggers Identification

Device ID	Network Protocol	Native Code	Sanity Checks	No. Candidate sendMessage	No. sendMessage	No. Fuzzing Triggers	No. Classes	No. Functions	No. Statements
1	UDP	✓	✓	4 (1 TP, 3 FP)	1 (1 TP)	7 (6 TP, 1 FP)	4,341	31,847	409,760
2	HTTP	✓	✓	12 (8TP, 4FP)	9 (6 TP, 3 FP) ⋆	6 (6 TP)	11,870	76,558	1,180,817
3	TCP + JSON	✗	?	6 (2 TP, 4 FP)	6 (2 TP, 4 FP)	3 (2 TP, 1 FP)	16,461	107,935	1,267,785
4	UDP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
5	TCP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
6	HTTP + SOAP	✓	✗	15 (3 TP, 12 FP)	6 (2 TP, 4 FP)⋆	9 (8 TP, 1 FP)	4,169	30,462	378,733
7	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4 (3 TP, 1 NC)	8,418	52,013	813,444
8	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4(3 TP, 1 NC)	8,418	52,013	813,444
9	TCP	✓	?	6 (3 TP, 3 FP)	1 (1 TP)⋆	1 (1 TP)•	6,010	42,358	467,670
10	UDP	✓	?	9 (1 TP, 8 FP)	3 (1 TP, 2 FP)	0	5,646	33,267	457,719
11	Bluetooth	✓	✓	9 (4 TP, 5 FP)	9 (4 TP, 5 FP)	16 (14 TP, 2 FP)	22,406	108,507	1,411,798
Total		10/11	7/11	97 (30 TP, 65 FP, 2 NC)	45 (25 TP, 20 FP)	54 (47 TP, 5 FP, 2 NC)	101,457	617,472	8,431,990

• *fuzzing triggers* coincide with *sendMessage* functions.

E1: Fuzzing Triggers Identification

Device ID	Network Protocol	Native Code	Sanity Checks	No. Candidate sendMessage	No. sendMessage	No. Fuzzing Triggers	No. Classes	No. Functions	No. Statements
1	UDP	✓	✓	4 (1 TP, 3 FP)	1 (1 TP)	7 (6 TP, 1 FP)	4,341	31,847	409,760
2	HTTP	✓	✓	12 (8TP, 4FP)	9 (6 TP, 3 FP) ⋆	6 (6 TP)	11,870	76,558	1,180,817
3	TCP + JSON	✗	?	6 (2 TP, 4 FP)	6 (2 TP, 4 FP)	3 (2 TP, 1 FP)	16,461	107,935	1,267,785
4	UDP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
5	TCP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
6	HTTP + SOAP	✓	✗	15 (3 TP, 12 FP)	6 (2 TP, 4 FP)⋆	9 (8 TP, 1 FP)	4,169	30,462	378,733
7	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4 (3 TP, 1 NC)	8,418	52,013	813,444
8	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4(3 TP, 1 NC)	8,418	52,013	813,444
9	TCP	✓	?	6 (3 TP, 3 FP)	1 (1 TP)⋆	1 (1 TP)•	6,010	42,358	467,670
10	UDP	✓	?	9 (1 TP, 8 FP)	3 (1 TP, 2 FP)	0	5,646	33,267	457,719
11	Bluetooth	✓	✓	9 (4 TP, 5 FP)	9 (4 TP, 5 FP)	16 (14 TP, 2 FP)	22,406	108,507	1,411,798
Total		10/11	7/11	97 (30 TP, 65 FP, 2 NC)	45 (25 TP, 20 FP)	54 (47 TP, 5 FP, 2 NC)	101,457	617,472	8,431,990

• *fuzzing triggers* coincide with *sendMessage* functions.

E1: Fuzzing Triggers Identification

Device ID	Network Protocol	Native Code	Sanity Checks	No. Candidate sendMessage	No. sendMessage	No. Fuzzing Triggers	No. Classes	No. Functions	No. Statements
1	UDP	✓	✓	4 (1 TP, 3 FP)	1 (1 TP)	7 (6 TP, 1 FP)	4,341	31,847	409,760
2	HTTP	✓	✓	12 (8TP, 4FP)	9 (6 TP, 3 FP) ⋆	6 (6 TP)	11,870	76,558	1,180,817
3	TCP + JSON	✗	?	6 (2 TP, 4 FP)	6 (2 TP, 4 FP)	3 (2 TP, 1 FP)	16,461	107,935	1,267,785
4	UDP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
5	TCP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
6	HTTP + SOAP	✓	✗	15 (3 TP, 12 FP)	6 (2 TP, 4 FP)⋆	9 (8 TP, 1 FP)	4,169	30,462	378,733
7	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4 (3 TP, 1 NC)	8,418	52,013	813,444
8	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4(3 TP, 1 NC)	8,418	52,013	813,444
9	TCP	✓	?	6 (3 TP, 3 FP)	1 (1 TP)⋆	1 (1 TP)•	6,010	42,358	467,670
10	UDP	✓	?	9 (1 TP, 8 FP)	3 (1 TP, 2 FP)	0	5,646	33,267	457,719
11	Bluetooth	✓	✓	9 (4 TP, 5 FP)	9 (4 TP, 5 FP)	16 (14 TP, 2 FP)	22,406	108,507	1,411,798
Total		10/11	7/11	97 (30 TP, 65 FP, 2 NC)	45 (25 TP, 20 FP)	54 (47 TP, 5 FP, 2 NC)	101,457	617,472	8,431,990

• *fuzzing triggers* coincide with *sendMessage* functions.

E1: Fuzzing Triggers Identification

Device ID	Network Protocol	Native Code	Sanity Checks	No. Candidate sendMessage	No. sendMessage	No. Fuzzing Triggers	No. Classes	No. Functions	No. Statements
1	UDP	✓	✓	4 (1 TP, 3 FP)	1 (1 TP)	7 (6 TP, 1 FP)	4,341	31,847	409,760
2	HTTP	✓	✓	12 (8TP, 4FP)	9 (6 TP, 3 FP) ⋆	6 (6 TP)	11,870	76,558	1,180,817
3	TCP + JSON	✗	?	6 (2 TP, 4 FP)	6 (2 TP, 4 FP)	3 (2 TP, 1 FP)	16,461	107,935	1,267,785
4	UDP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
5	TCP	✓	✓	10 (2 TP, 7 FP, 1 NC)	2 (2 TP)	2 (2 TP) •	6,859	41,256	615,410
6	HTTP + SOAP	✓	✗	15 (3 TP, 12 FP)	6 (2 TP, 4 FP)⋆	9 (8 TP, 1 FP)	4,169	30,462	378,733
7	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4 (3 TP, 1 NC)	8,418	52,013	813,444
8	TCP	✓	✓	8 (2 TP, 6 FP)	3 (2 TP, 1 FP)	4(3 TP, 1 NC)	8,418	52,013	813,444
9	TCP	✓	?	6 (3 TP, 3 FP)	1 (1 TP)⋆	1 (1 TP)•	6,010	42,358	467,670
10	UDP	✓	?	9 (1 TP, 8 FP)	3 (1 TP, 2 FP)	0	5,646	33,267	457,719
11	Bluetooth	✓	✓	9 (4 TP, 5 FP)	9 (4 TP, 5 FP)	16 (14 TP, 2 FP)	22,406	108,507	1,411,798
Total		10/11	7/11	97 (30 TP, 65 FP, 2 NC)	45 (25 TP, 20 FP)	54 (47 TP, 5 FP, 2 NC)	101,457	617,472	8,431,990

• *fuzzing triggers* coincide with *sendMessage* functions.

E2: Vulnerability Detection

	DIANE				
Device ID	**No. Generated Alerts**	**No. Bugs**	**Zero-day**	**Vuln. Type**	**Time [hours] (No. Generated Inputs)**
1	1	1	✓	Unknown	$\leq$0.5 (60,750)
2	3	7	✓	Buff overflow	$\leq$0.5 (322)
3	1	1		Unknown	$\leq$1.2 (7,344)
4	1	0		N/A	N/A
5	1	0		N/A	N/A
6	4	1		Unknown	$\leq$10 (34,680)
7	3	0		N/A	N/A
8	3	0		N/A	N/A
9	0	0		N/A	N/A
10	1	0		N/A	N/A
11	0	1	✓	Unknown	2.2 (3,960)

E2: Vulnerability Detection

	DIANE				
Device ID	No. Generated Alerts	No. Bugs	Zero-day	Vuln. Type	Time [hours] (No. Generated Inputs)
1	1	1	✓	Unknown	$\leq$0.5 (60,750)
2	3	7	✓	Buff overflow	$\leq$0.5 (322)
3	1	1		Unknown	$\leq$1.2 (7,344)
4	1	0		N/A	N/A
5	1	0		N/A	N/A
6	4	1		Unknown	$\leq$10 (34,680)
7	3	0		N/A	N/A
8	3	0		N/A	N/A
9	0	0		N/A	N/A
10	1	0		N/A	N/A
11	0	1	✓	Unknown	2.2 (3,960)

E2: Vulnerability Detection

	DIANE				
Device ID	**No. Generated Alerts**	**No. Bugs**	**Zero-day**	**Vuln. Type**	**Time [hours] (No. Generated Inputs)**
1	1	1	✓	Unknown	$\leq$0.5 (60,750)
2	3	7	✓	Buff overflow	$\leq$0.5 (322)
3	1	1		Unknown	$\leq$1.2 (7,344)
4	1	0		N/A	N/A
5	1	0		N/A	N/A
6	4	1		Unknown	$\leq$10 (34,680)
7	3	0		N/A	N/A
8	3	0		N/A	N/A
9	0	0		N/A	N/A
10	1	0		N/A	N/A
11	0	1	✓	Unknown	2.2 (3,960)

E2: Vulnerability Detection

	DIANE				
Device ID	No. Generated Alerts	No. Bugs	Zero-day	Vuln. Type	Time [hours] (No. Generated Inputs)
1	1	1	✓	Unknown	$\leq$0.5 (60,750)
2	3	7	✓	Buff overflow	$\leq$0.5 (322)
3	1	1		Unknown	$\leq$1.2 (7,344)
4	1	0		N/A	N/A
5	1	0		N/A	N/A
6	4	1		Unknown	$\leq$10 (34,680)
7	3	0		N/A	N/A
8	3	0		N/A	N/A
9	0	0		N/A	N/A
10	1	0		N/A	N/A
11	0	1	✓	Unknown	2.2 (3,960)

E3: Vulnerability Detection

	DIANE					IoTFuzzer			Other Fuzzers			
Device ID	No. Generated Alerts	No. Bugs	Zero-day	Vuln. Type	Time [hours] (No. Generated Inputs)	No. Fuzzed Functions	No. Bugs	Time [hours]	No. Bugs BED	Sulley	uFuzz	bss
1	1	1	✓	Unknown	≤ 0.5 (60,750)	• 1	0	N/A	N/A	0	N/A	N/A
2	3	7	✓	Buff overflow	≤ 0.5 (322)	5	2	0.98	0	0	N/A	N/A
3	1	1		Unknown	≤ 1.2 (7,344)	1	1	4	0	0	N/A	N/A
4	1	0		N/A	N/A	• 1	0	N/A	N/A	0	N/A	N/A
5	1	0		N/A	N/A	• 1	0	N/A	0	0	N/A	N/A
6	4	1		Unknown	≤ 10 (34,680)	1	1	≤ 10	0	0	0	N/A
7	3	0		N/A	N/A	N/A	N/A	N/A	0	0	N/A	N/A
8	3	0		N/A	N/A	N/A	N/A	N/A	0	0	N/A	N/A
9	0	0		N/A	N/A	3	0	N/A	0	0	0	N/A
10	1	0		N/A	N/A	N/A	N/A	N/A	N/A	0	N/A	N/A
11	0	1	✓	Unknown	2.2 (3,960)	N/A	N/A	N/A	N/A	N/A	N/A	0

E3: Vulnerability Detection

	DIANE					IoTFuzzer			Other Fuzzers			
Device ID	No. Generated Alerts	No. Bugs	Zero-day	Vuln. Type	Time [hours] (No. Generated Inputs)	No. Fuzzed Functions	No. Bugs	Time [hours]	No. Bugs BED	Sulley	uFuzz	bss
1	1	1	✓	Unknown	$\leq$0.5 (60,750)	• 1	0	N/A	N/A	0	N/A	N/A
2	3	7	✓	Buff overflow	$\leq$0.5 (322)	5	2	0.98	0	0	N/A	N/A
3	1	1		Unknown	$\leq$1.2 (7,344)	1	1	4	0	0	N/A	N/A
4	1	0		N/A	N/A	• 1	0	N/A	N/A	0	N/A	N/A
5	1	0		N/A	N/A	• 1	0	N/A	0	0	N/A	N/A
6	4	1		Unknown	$\leq$10 (34,680)	1	1	$\leq$10	0	0	0	N/A
7	3	0		N/A	N/A	N/A	N/A	N/A	0	0	N/A	N/A
8	3	0		N/A	N/A	N/A	N/A	N/A	0	0	N/A	N/A
9	0	0		N/A	N/A	3	0	N/A	0	0	0	N/A
10	1	0		N/A	N/A	N/A	N/A	N/A	N/A	0	N/A	N/A
11	0	1	✓	Unknown	2.2 (3,960)	N/A	N/A	N/A	N/A	N/A	N/A	0

E3: Vulnerability Detection

	DIANE					IoTFuzzer			Other Fuzzers			
Device ID	No. Generated Alerts	No. Bugs	Zero-day	Vuln. Type	Time [hours] (No. Generated Inputs)	No. Fuzzed Functions	No. Bugs	Time [hours]	No. Bugs BED	Sulley	uFuzz	bss
1	1	1	✓	Unknown	$\leq$0.5 (60,750)	• 1	0	N/A	N/A	0	N/A	N/A
2	3	7	✓	Buff overflow	$\leq$0.5 (322)	5	2	0.98	0	0	N/A	N/A
3	1	1		Unknown	$\leq$1.2 (7,344)	1	1	4	0	0	N/A	N/A
4	1	0		N/A	N/A	• 1	0	N/A	N/A	0	N/A	N/A
5	1	0		N/A	N/A	• 1	0	N/A	0	0	N/A	N/A
6	4	1		Unknown	$\leq$10 (34,680)	1	1	$\leq$10	0	0	0	N/A
7	3	0		N/A	N/A	N/A	N/A	N/A	0	0	N/A	N/A
8	3	0		N/A	N/A	N/A	N/A	N/A	0	0	N/A	N/A
9	0	0		N/A	N/A	3	0	N/A	0	0	0	N/A
10	1	0		N/A	N/A	N/A	N/A	N/A	N/A	0	N/A	N/A
11	0	1	✓	Unknown	2.2 (3,960)	N/A	N/A	N/A	N/A	N/A	N/A	0

Evaluation

We reported our findings to the vendors

- 10 bugs confirmed
- 1 bug is still being investigated (at time of writing)

Thanks! && Questions?

Nilo Redini

nredini@cs.ucsb.edu

https://badnack.it

@badnack

Aravind Machiry

amachiry@purdue.edu

https://machiry.github.io/

@machiry_msidc