BLOOMFUZZ

Unveiling Bluetooth L2CAP Vulnerabilities via State Cluster Fuzzing with Target-Oriented State Machines

ESORICS'2024
(29th European Symposium on Research in Computer Security)

Pyeongju Ahn, Yeonseok Jang, Seunghoon Woo*, Heejo Lee*

Korea University

pingjuu@korea.ac.kr

Sep 16th, 2024

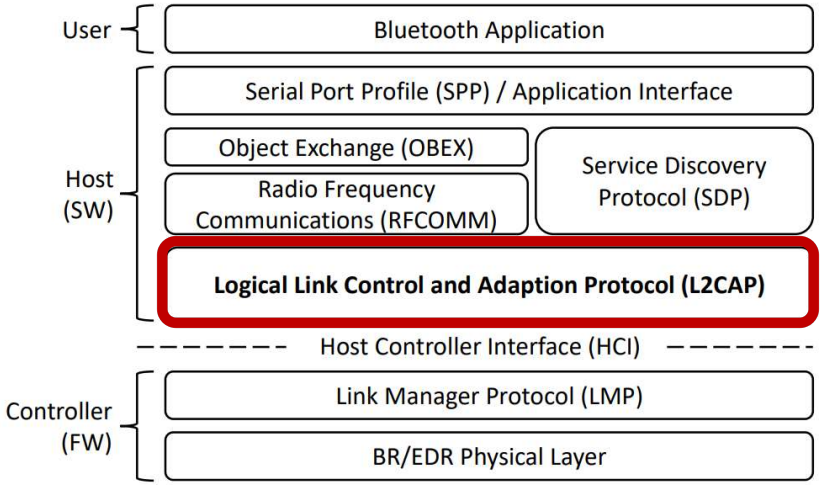
INDEX

1-1. Bluetooth Classic Stack

Bluetooth is short-range wireless technology standard

Bluetooth L2CAP

The L2CAP facilitates the transmission and reception between lower and upper-level



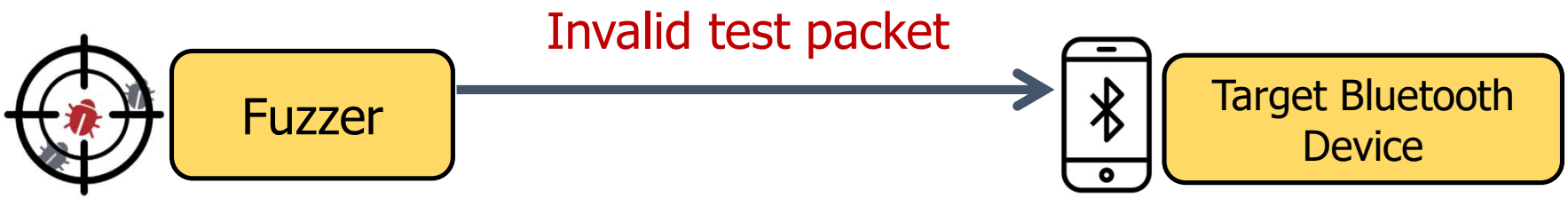
[Bluetooth BR/EDR Stack]

1-2. Stateful Fuzzing

Stateful fuzzing is effective in identifying L2CAP vulnerabilities

Stateful fuzzing

- The communication process of the L2CAP can be represented by *states*
- A fuzzing technique designed to consider these states to detect potential threats

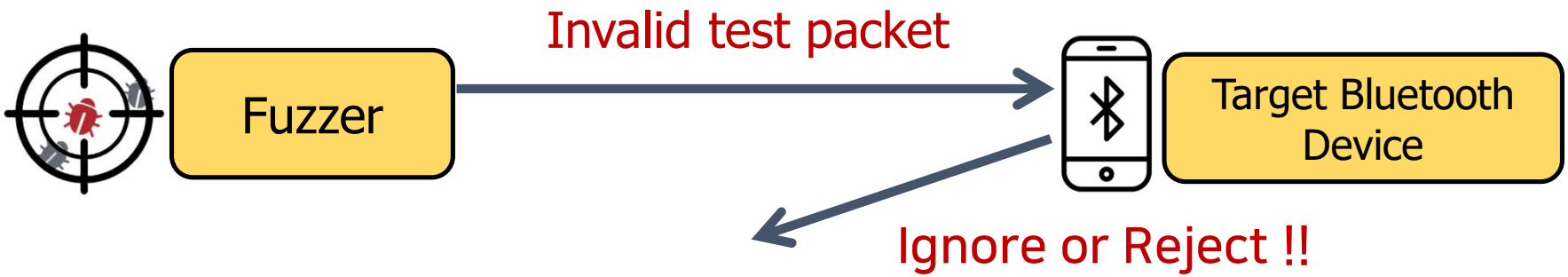


1-2. Stateful Fuzzing

Stateful fuzzing is effective in identifying L2CAP vulnerabilities

Stateful fuzzing

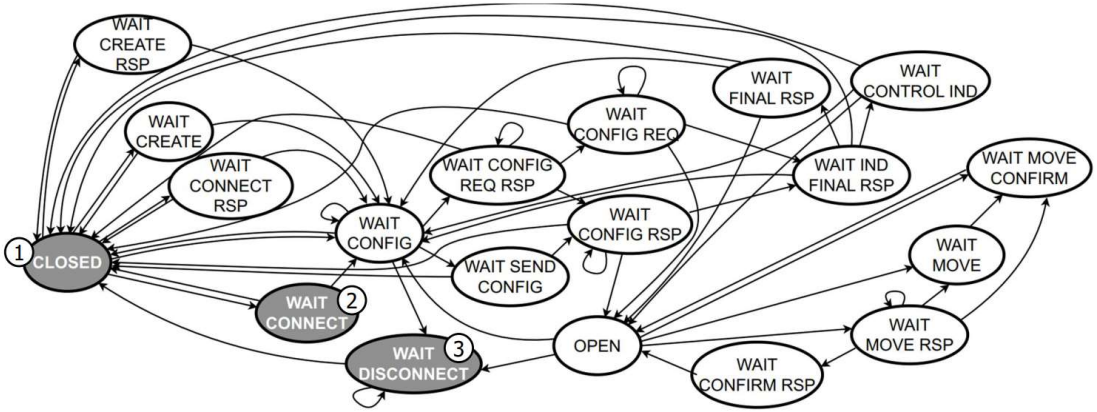
- The communication process of the L2CAP can be represented by *states*
- A fuzzing technique designed to consider these states to detect potential threats



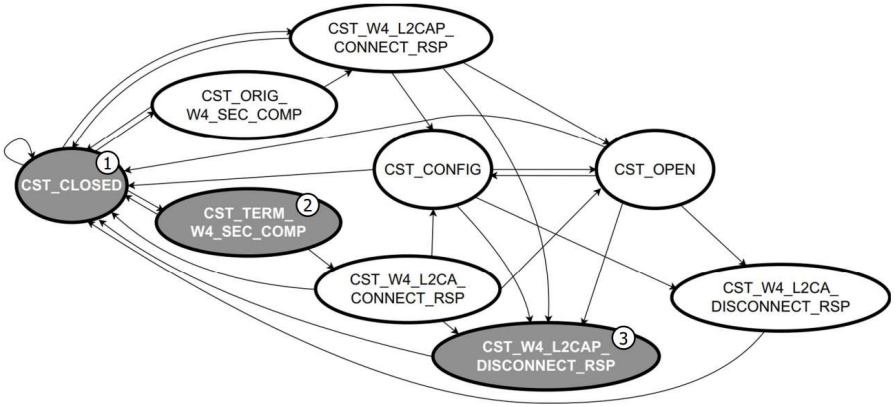
2-1. Motivation

There is a gap between specification and device

- The L2CAP state machine in the specification differs from the one in the device
- Bluetooth devices are implemented based on the specification, but are modified



[State machine of Bluetooth 5.2 specification]

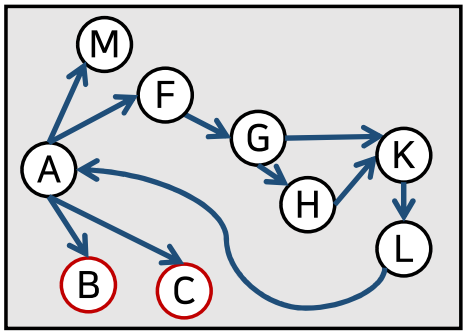


[State machine of BlueDroid v12.1.0.r19]

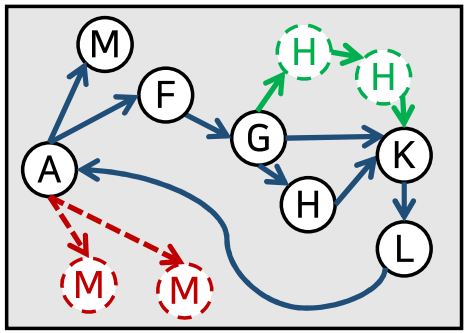
2-2. Challenge #1

Difficulty in precisely generating a state machine

- States = {Normal states, Missing states, Hidden states}
  - ✓ Normal state : (N)
  - ✓ Missing state : (M)
  - ✓ Hidden state : (H)



[Example state machine of specification]

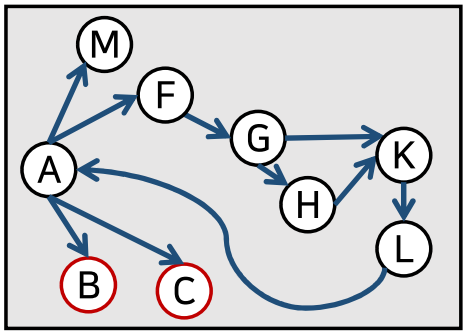


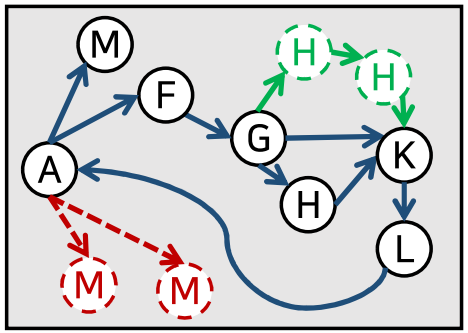
[Example state machine of device]

2-2. Challenge #1

Difficulty in precisely generating a state machine

- States = {Normal states, Missing states, Hidden states}
  - ✓ Normal state : (N)
  - ✓ Missing state : (M)
  - ✓ Hidden state : (H)





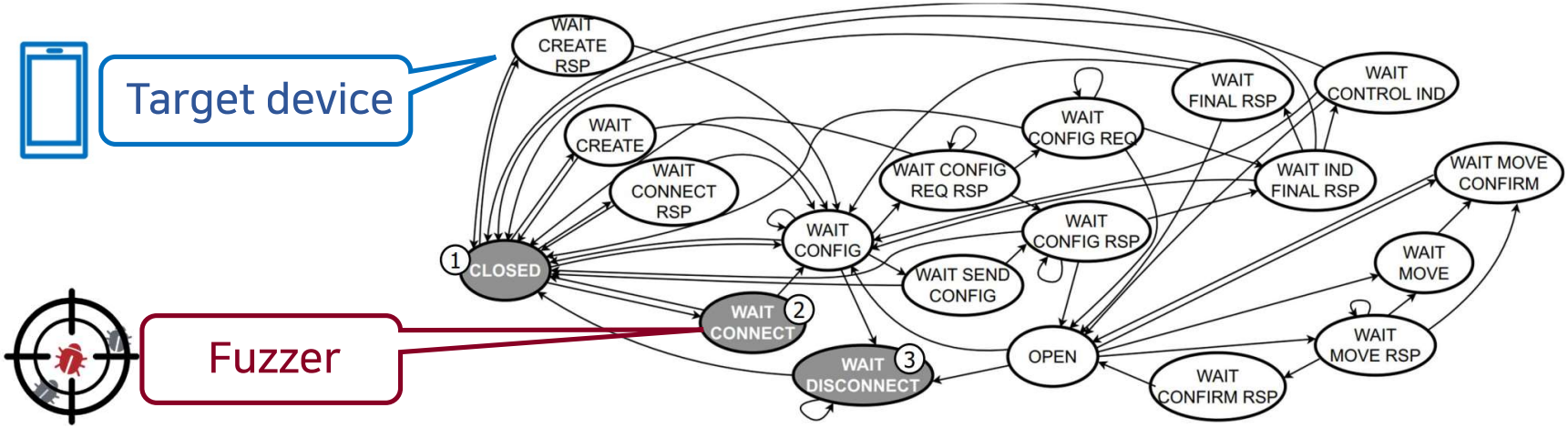
We need to precisely generate the state machine for the target device by addressing the missing and hidden states

2-2. Challenge #2

Difficulty in correctly tracking states during fuzzing process

The current state of the target device ≠ Target state of the fuzzer

➡ Test packet is likely to be ignored or rejected early



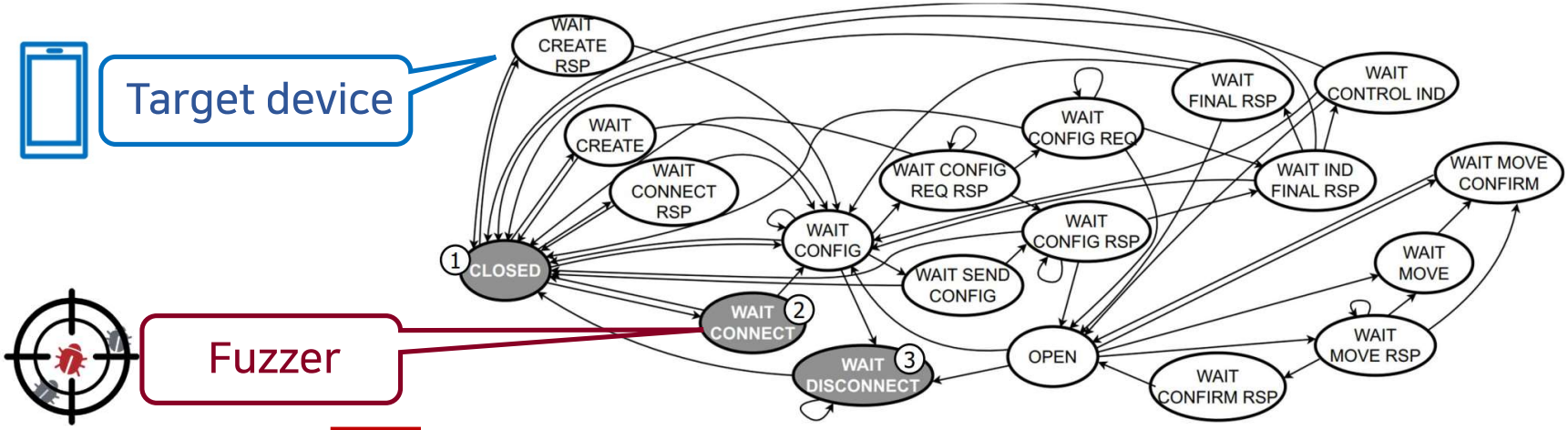
[State machine of Bluetooth 5.2 specification]

2-2. Challenge #2

Difficulty in correctly tracking states during fuzzing process

The current state of the target device ≠ Target state of the fuzzer

➡ Test packet is likely to be ignored or rejected early

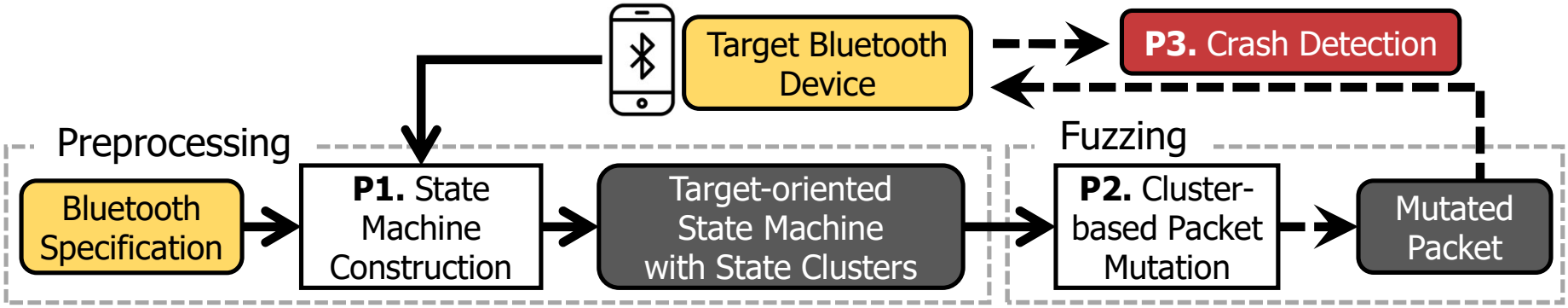


To achieve effective stateful fuzzing,

it is essential to identify the target device's current state

3-1. BLOOMFUZZ

Discovering L2CAP vulnerabilities via state cluster fuzzing with target-oriented state machines

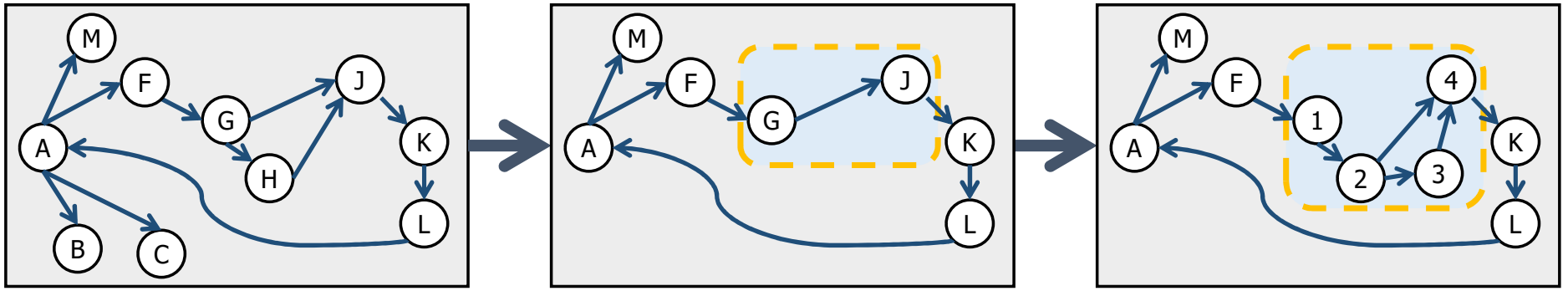


Key techniques

- State clustering
- Generating target-oriented state machine

3-1. BLOOMFUZZ

High-level workflow



1. Spec state machine generating

2. Missing states pruning

3. Hidden state addressing

BLOOMFUZZ mutate packets and performs fuzzing based on this target-oriented machine.

3-2. State clustering (P1)

A cluster is a set of one or more states with similar attributes

- Valid *L2CAP commands
- The role on the target device

*Most transitions are carried out through L2CAP commands

Cluster IDX	States	Commands	Role*
1	CLOSED	All commands	C/P
2	WAIT_CONNECT	L2CAP_Connect_Req/Rsp	P
3	WAIT_CONNECT_RSP	L2CAP_Connect_Req/Rsp	C
4	WAIT_CREATE	L2CAP_Create_Channel_Req/Rsp	P
5	WAIT_CREATE_RSP	L2CAP_Create_Channel_Req/Rsp	C
6	WAIT_CONFIG	L2CAP_Configuration_Req/Rsp	C/P
7	WAIT_SEND_CONFIG, WAIT_CONFIG_RSP, †WAIT_IND_FINAL_RSP, †WAIT_FINAL_RSP, †WAIT_CONTROL_IND	L2CAP_Configuration_Req/Rsp	P
8	WAIT_CONFIG_REQ, WAIT_CONFIG_REQ_RSP, †WAIT_IND_FINAL_RSP, †WAIT_FINAL_RSP, †WAIT_CONTROL_IND	L2CAP_Configuration_Req/Rsp	C
9	OPEN	All commands	C/P
10	WAIT_MOVE, WAIT_MOVE_CONFIRM	L2CAP_Move_Channel_Req/Rsp, L2CAP_Move_Channel_Confirmation_Req/Rsp	P
11	WAIT_CONFIRM_RSP, WAIT_MOVE_RSP	L2CAP_Move_Channel_Req/Rsp, L2CAP_Move_Channel_Confirmation_Req/Rsp	C
12	WAIT_DISCONNECT	L2CAP_Disconnection_Req/Rsp	C/P

*The role of the target device (Central or Peripheral); †States belonging to Clusters #7 and #8.

3-2. State clustering (P1)

A cluster is a set of one or more states with similar attributes

- Valid *L2CAP commands
- The role on the target device

*Most transitions are carried out through L2CAP commands

Cluster IDX	States	Commands	Role*
1	CLOSED	All commands	C/P
2	WAIT_CONNECT	L2CAP_Connect_Req/Rsp	P
3	WAIT_CONNECT_RSP	L2CAP_Connect_Req/Rsp	C
4	WAIT_CREATE	L2CAP_Create_Channel_Req/Rsp	P
5	WAIT_CREATE_RSP	L2CAP_Create_Channel_Req/Rsp	C
6	WAIT_CONFIG	L2CAP_Configuration_Req/Rsp	C/P
7	WAIT_SEND_CONFIG, WAIT_CONFIG_RSP, †WAIT_IND_FINAL_RSP, †WAIT_FINAL_RSP, †WAIT_CONTROL_IND	L2CAP_Configuration_Req/Rsp	P
8	WAIT_CONFIG_REQ, WAIT_CONFIG_REQ_RSP, †WAIT_IND_FINAL_RSP, †WAIT_FINAL_RSP, †WAIT_CONTROL_IND	L2CAP_Configuration_Req/Rsp	C
9	OPEN	All commands	C/P
10	WAIT_MOVE, WAIT_MOVE_CONFIRM	L2CAP_Move_Channel_Req/Rsp, L2CAP_Move_Channel_Confirmation_Req/Rsp	P
11	WAIT_CONFIRM_RSP, WAIT_MOVE_RSP	L2CAP_Move_Channel_Req/Rsp, L2CAP_Move_Channel_Confirmation_Req/Rsp	C
12	WAIT_DISCONNECT	L2CAP_Disconnection_Req/Rsp	C/P

*The role of the target device (Central or Peripheral); †States belonging to Clusters #7 and #8.

BLOOMFUZZ can achieve two key effects:

1. The ability to handle to hidden states
2. The efficient generation of test packets with a low probability of rejection

3-3. State machine construction (P1)

BLOMMFUZZ generates the state machine by addressing the missing and hidden states

1. Generating a specification-based state machine

   A manual analysis of the specification while considering all the states and transitions

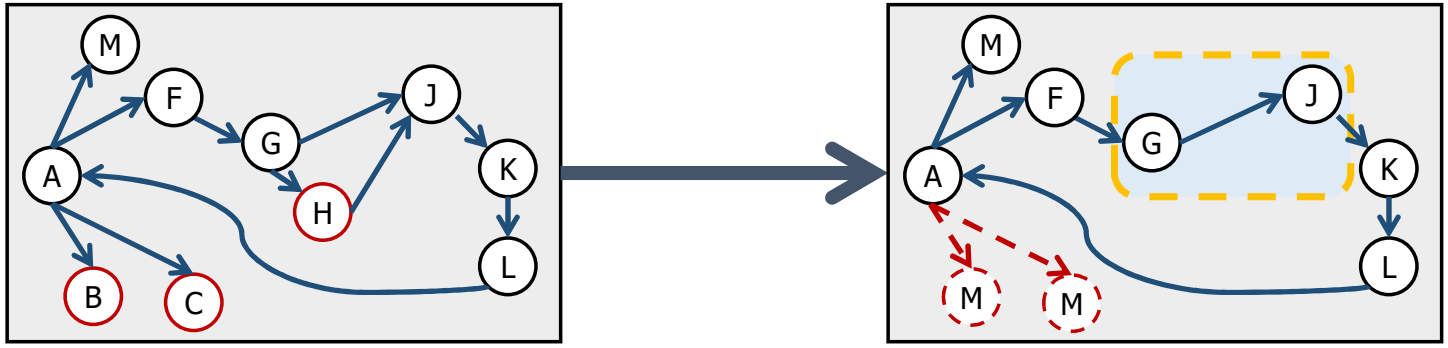
3-3. State machine construction (P1)

BLOMMFUZZ generates the state machine by addressing the missing and hidden states

2. Pruning missing states

2.1. Traversing the specification-based state machine

2.2. Verifying whether the states specified in the specification are implemented in the target device



Example of Spec state machine

State machine
with missing states pruned

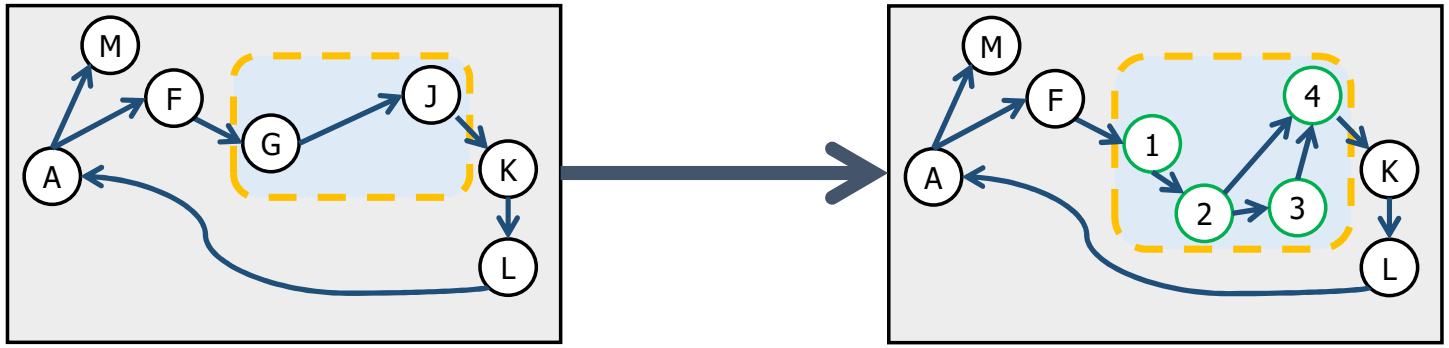
3-3. State machine construction (P1)

BLOMMFUZZ generates the state machine by addressing the missing and hidden states

3. Addressing hidden states

3.1. Record communication

3.2. Parse and transform into the state machine



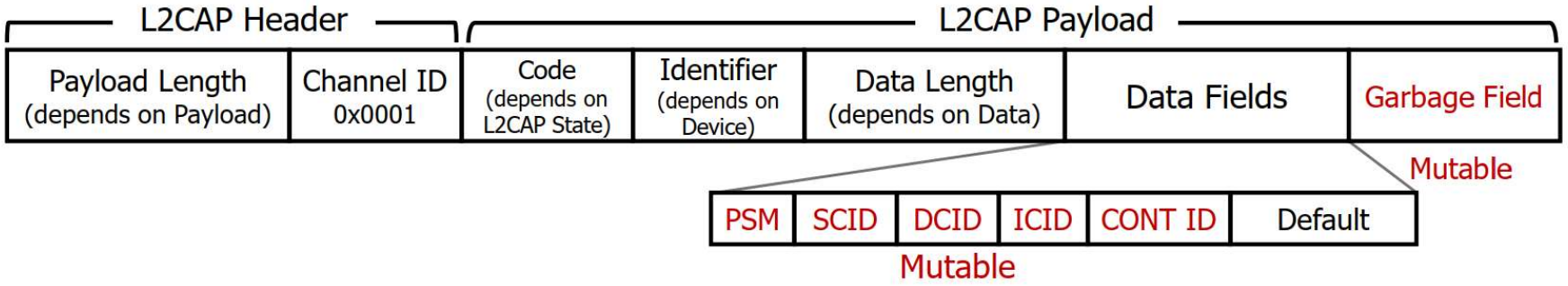
State machine
with missing states pruned

Addressing hidden states
with clusters

3-4. Cluster-based packet mutation (P2)

Field classification

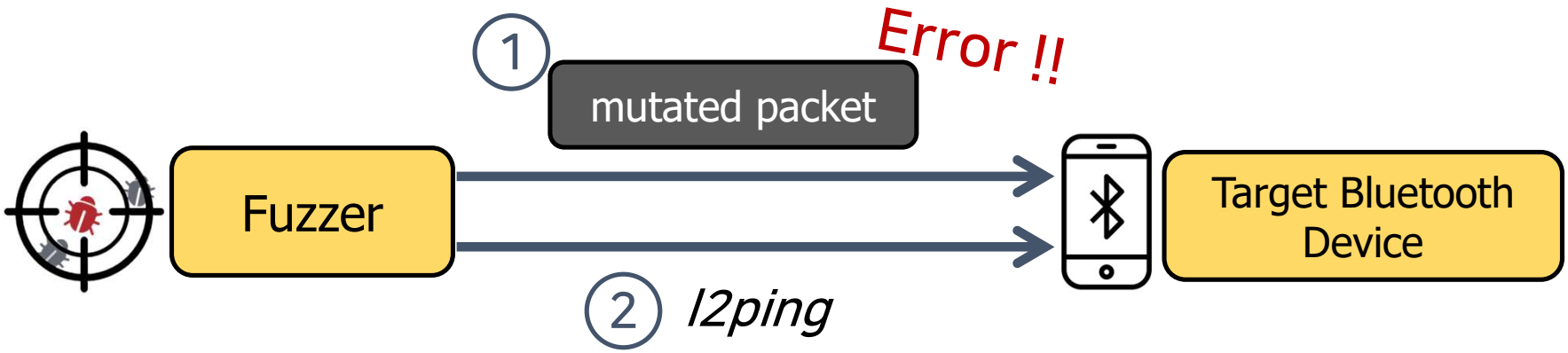
- BLOOMFUZZ generates valid packets for target cluster
- Then, BLOOMFUZZ performs mutations only in fields that do not affect the packet validity



[Mutable field selection for packet mutation]

3-5. Crash Detection (P3)

BLOOMFUZZ detects crashes by sending mutated packets to the target device



4-1. Experimental Setup

Experimental environment

- Ubuntu 20.04.LTS
- 16 GB memory, Intel Core i5-7500 CPU @ 3.30 GHz, and 64 GB SSD
- Cambridge Silicon Radio Bluetooth Classic dongle

Target devices

ID	Type	Vendor	Name	OS/FW*
D1	Laptop	LG	Gram	Windows 10
D2	Laptop	LG	Gram	Ubuntu 18.04.4
D3	Phone	Google	Pixel7	Android 14
D4	Phone	Google	Pixel3	Android 12
D5	Tablet	Samsung	Galaxy Tab S6 Lite	Android 12
D6	Earphone	Samsung	Galaxy Buds+	R175XXU0AUK1
D7	Earphone	Xiaomi	Redmi Buds 3 Pro	1.0.9.9

Comparison target

- BSS, BFuzz, L2Fuzz

4-2. Experiment on crash detection

BLOOMFUZZ discovered 56 crashes

Target	#Detected crashes in each fuzzer			
	BloomFuzz	L2Fuzz	BFuzz	BSS
D1	**17**	0	3	0
D2	**6**	0	0	0
D3	**8**	0	8	0
D4	**1**	0	0	0
D5	0	0	**12**	0
D6	**14**	4	0	0
D7	10	**26**	0	0
Total	**56**	30	23	0

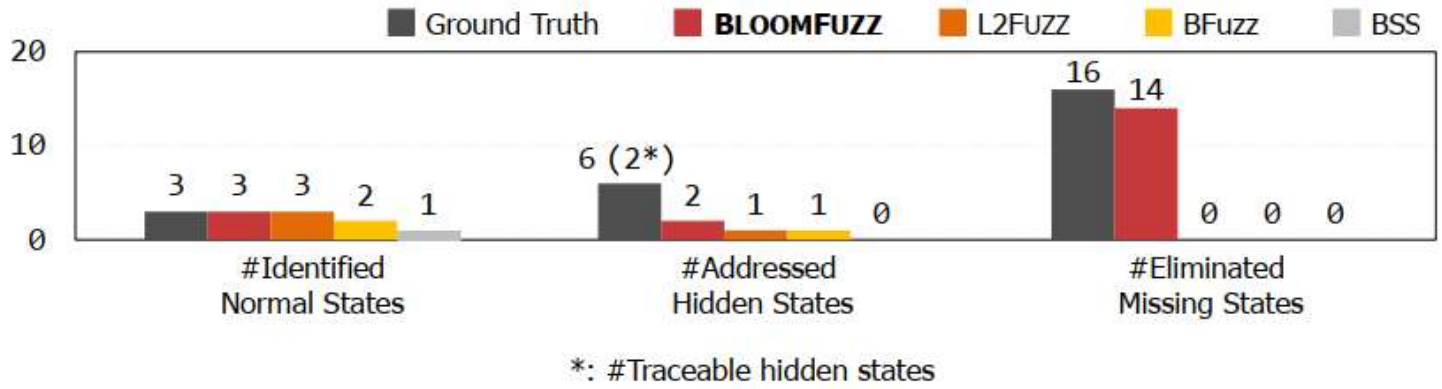
[Crash detection results of each fuzzer]

- We reported two vulnerabilities that were reproducible among the detected crashes to the respective vendors
- Only BFUZZ was able to find the crash on D5 because it mutates all fields that BLOOMFUZZ and L2FUZZ do not

4-3. Effectiveness of state machine generation

BLOOMFUZZ precisely generates a target-oriented state machine

- *Q1: How effectively are missing states pruned (A_m)?*
- *Q2: How effectively are the implemented states identified (A_i)?*



*Ground Truth: Pixel 3 running to Android 12

4-4. Efficiency of state tracking and packet mutation

- BLOOMFUZZ shows the packet acceptance (A_t) of 77% and the mutation efficiency (M_e) of 49%

Fuzzers	#Total Sent Pkts	#Rejected Pkts	#Malformed Pkts	A_t	M_e
BLOOMFUZZ	1,459,515	**341,858**	**923,468**	**77%**	**49%**
L2FUZZ	926,768	511,070	585,616	45%	28%
BFuzz	**2,002,862**	1,457,943	99,745	27%	1%
BSS	1,202,518	908,986	389,763	24%	8%

[Measurement result of packet acceptance ratio and mutation efficiency]

- The packet acceptance

$$A_t = 1 - \left(\frac{\#\text{Rejected Packets}}{\#\text{Total Sent Packets}} \right)$$

- The mutation efficiency

$$M_e = \left(\frac{\#\text{Malformed Packets}}{\#\text{Total Sent Packets}} \right) \times A_t$$

5. Conclusion

Conclusion

- BLOOMFUZZ is based on the Bluetooth v5.2 specification but can be used with any Bluetooth version, as v5.2 includes all states from each version

- Based on cluster, BLOOMFUZZ infers the state machine implemented in the target device with high accuracy and enhances fuzzing efficiency

- BLOOMFUZZ exhibited significantly higher fuzzing efficiency, reporting 2 vulnerabilities in real-world Bluetooth devices

Q&A

Thanks for your attention

- BLOOMFUZZ will be presented at the ESORICS'2024 conference
- BLOOMFUZZ source code repository is (https://github.com/pingjuu/BLOOMFUZZ)
- BLOOMFUZZ will be available at (https://iotcube.net) as a part of B2FUZZ

Contact

- PyeongJu Ahn (pingjuu@korea.ac.kr)
- Computer & Communication Security Lab (https://ccs.korea.ac.kr)