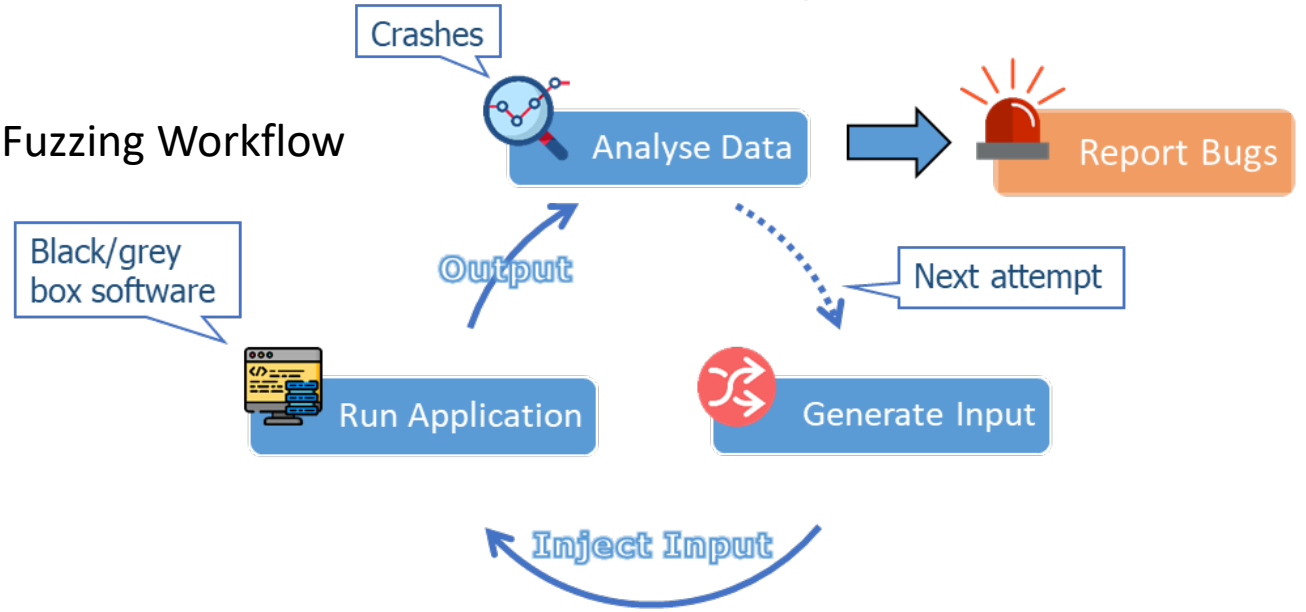
# BrakTooth: Causing Havoc on Bluetooth Link Manager via Directed Fuzzing

Matheus Eduardo Garbelini[1]
Vaibhav Bedi[1]
Sudipta Chattopadhyay[1]
Sun Sumei[2]
Ernest Kurniawan[2]

[1] ASSET Research Group, Singapore University of Technology and Design (SUTD)
[2] Institute for Infocomm Research, A*Star

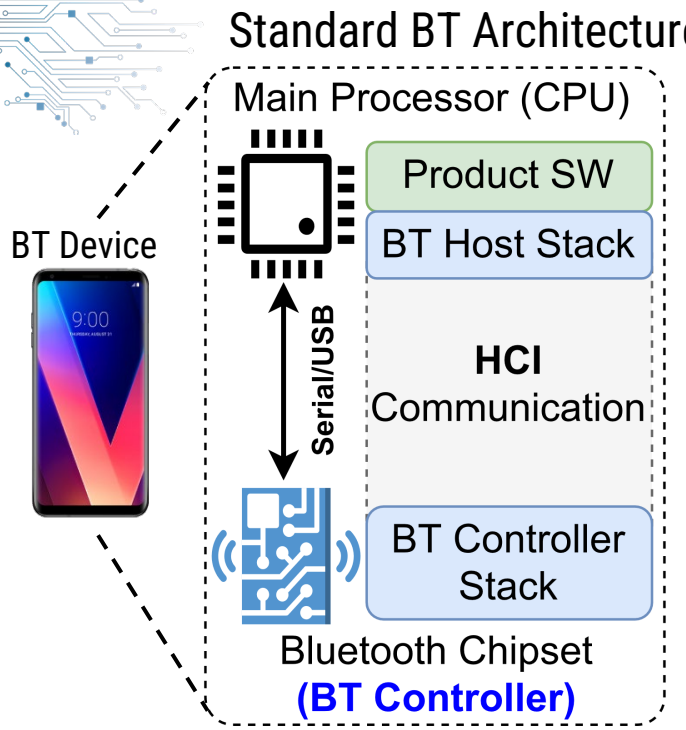

# Motivation – Wireless Fuzzing

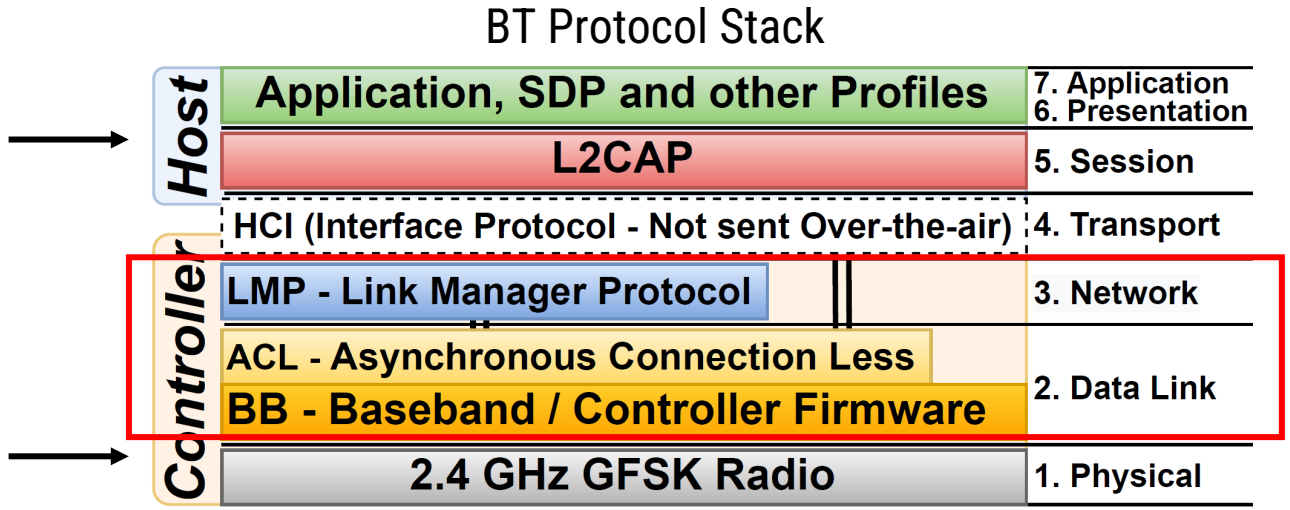
Standard Fuzzing Workflow



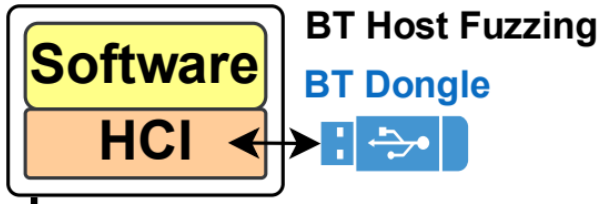

1. Higher protocol complexity usually means more bugs to find;
2. Lack of freedom to inject (fuzz) packets over-the-air;
   - ➢ Closed Source implementations;
   - ➢ Wireless timing constraints makes fuzzing more difficult;
3. Current BT fuzzing approaches either require manual generation of inputs (too complex) or generate too many invalid input (mutation).
   - ➢ Need for a more generalized wireless fuzzing approach.

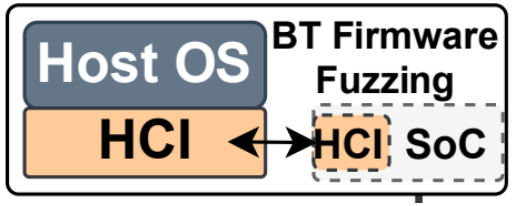
# Our Approach and Targets – Bluetooth Classic Fuzzing

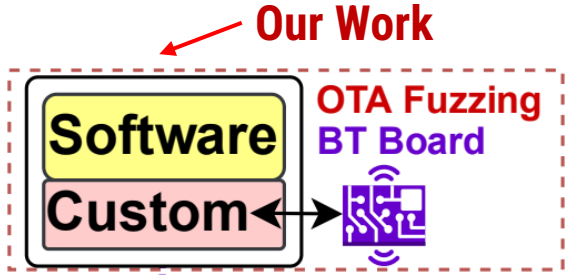



Other BT Classic Fuzzers – How we compare to state-of-art?



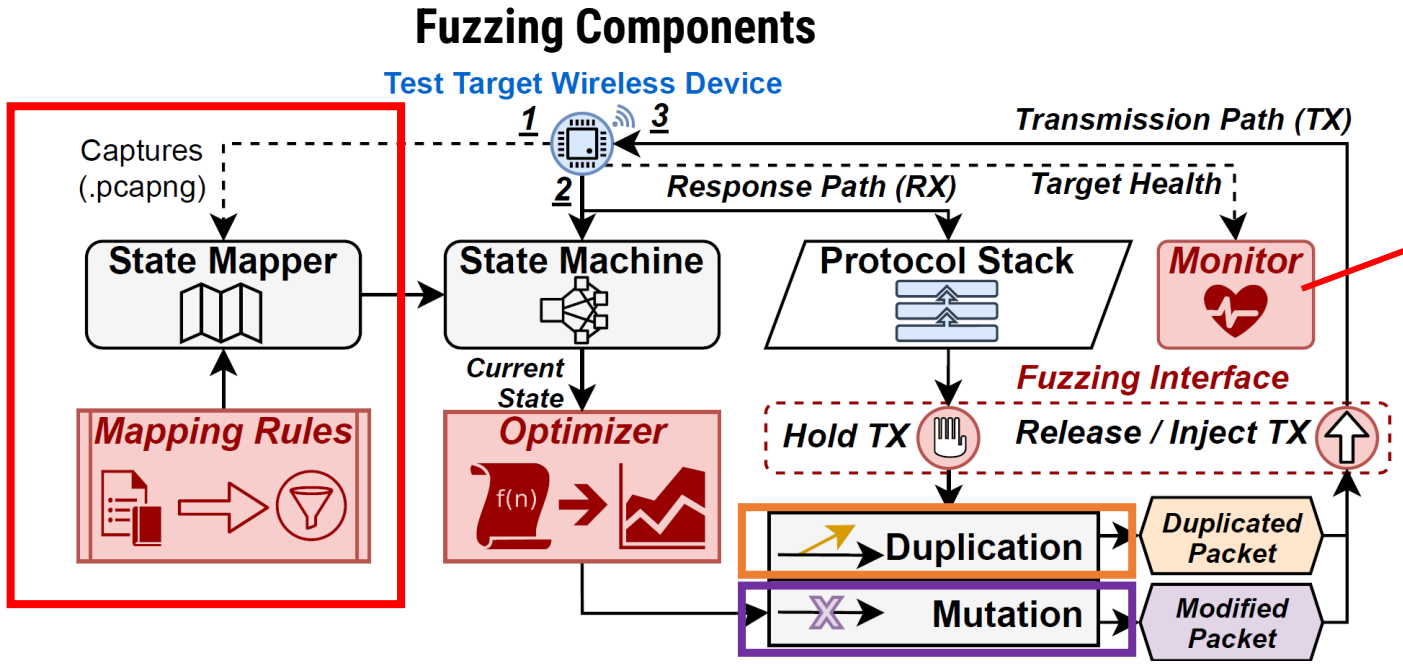

Trade-offs

+ No special BT Hardware required
- Target only BT Host Layers

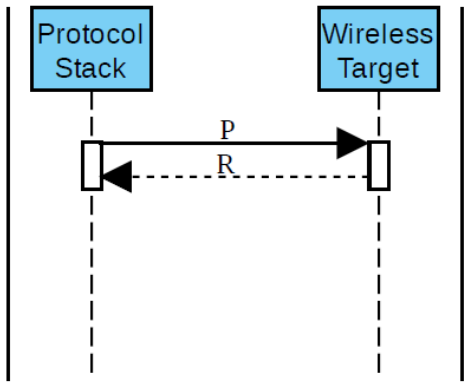
+ Fuzzing is Fast
- Specific SoC Architectures

+ Test Arbitrary BT SoCs
- Fuzzing is Slower

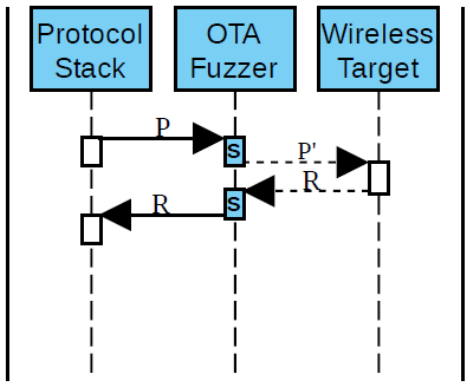
# General Over-the-Air Fuzzing Workflow



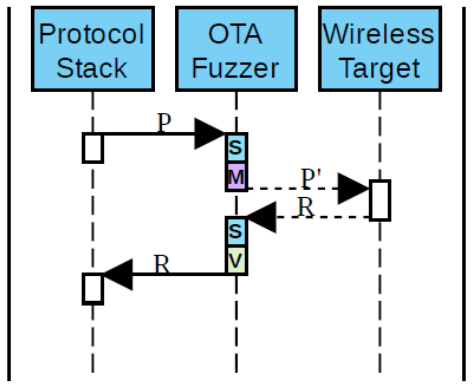

Target Monitoring is detailed on the paper



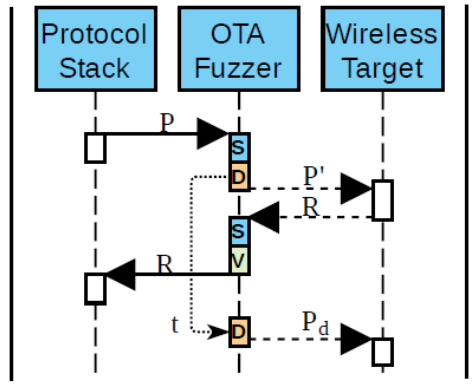

a) Normal Communication




b) State Machine Generation (P=P')



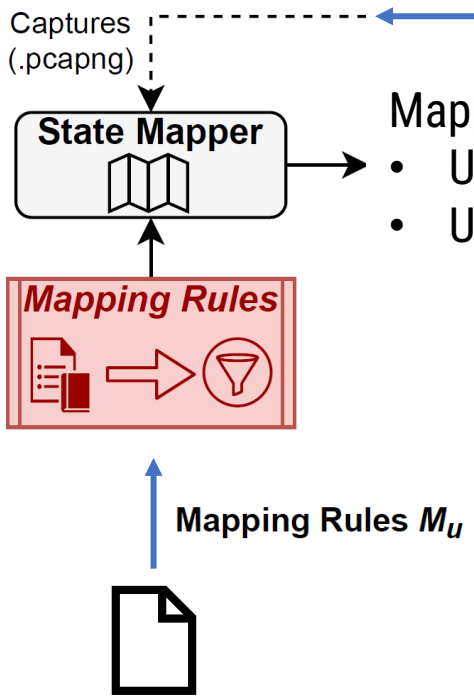

c) Fuzzing: Mutation (P≠P')




d) Fuzzing: Duplication (P=P'=$P_d$)

# 1. Fuzzing Workflow - Protocol State Mapping

Intuition: “Type” fields of packets can inform the protocol state during communication.

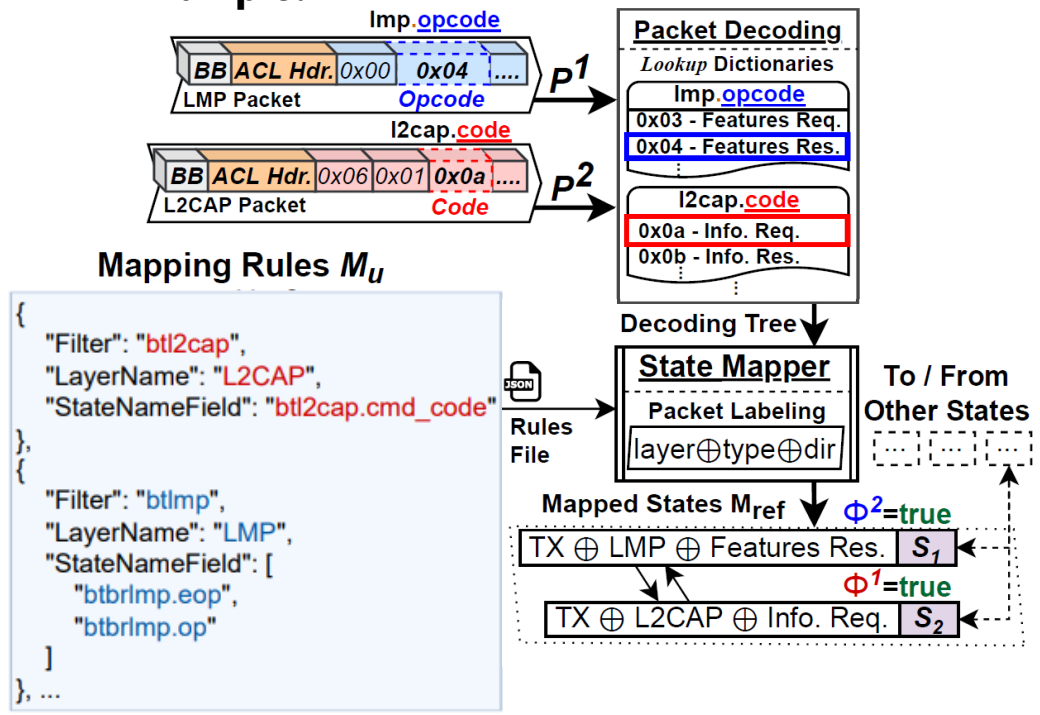



Provided by the user or received via live communication.

Mapped States

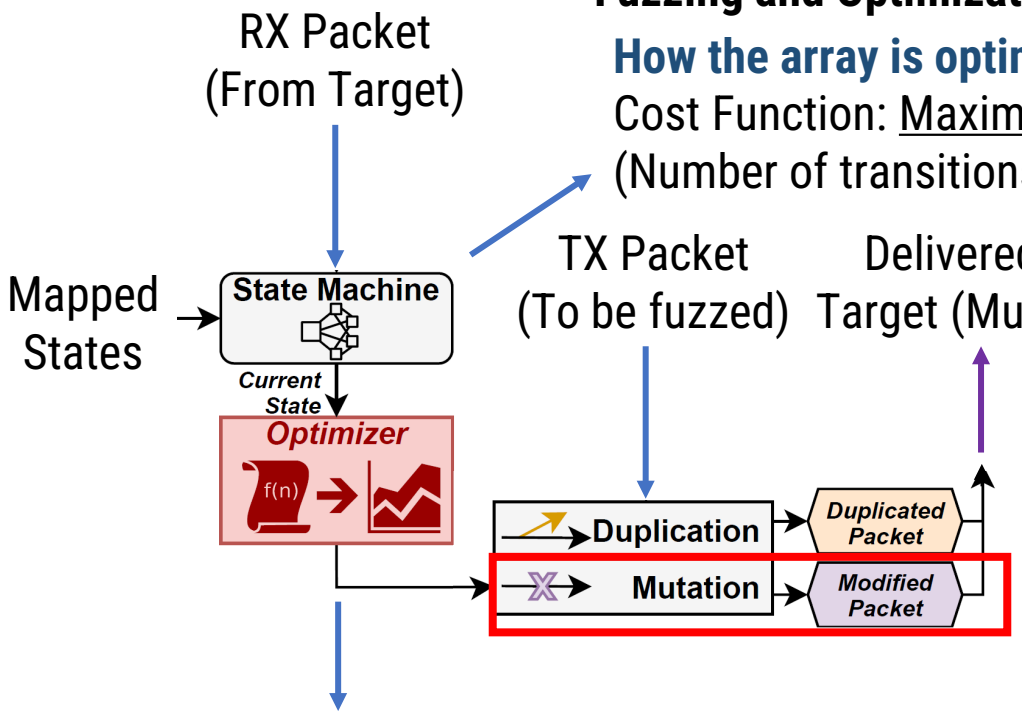
- Used for the fuzzer to know the current state during fuzzing;
- Used to detect “unexpected” responses that may indicate vulnerabilities or non-compliance

**Example:**



# 2. Fuzzing Workflow - Directed Fuzzing

Intuition: Directing the fuzzing towards fields or layers that contribute to more state transitions




**Fuzzing and Optimization**

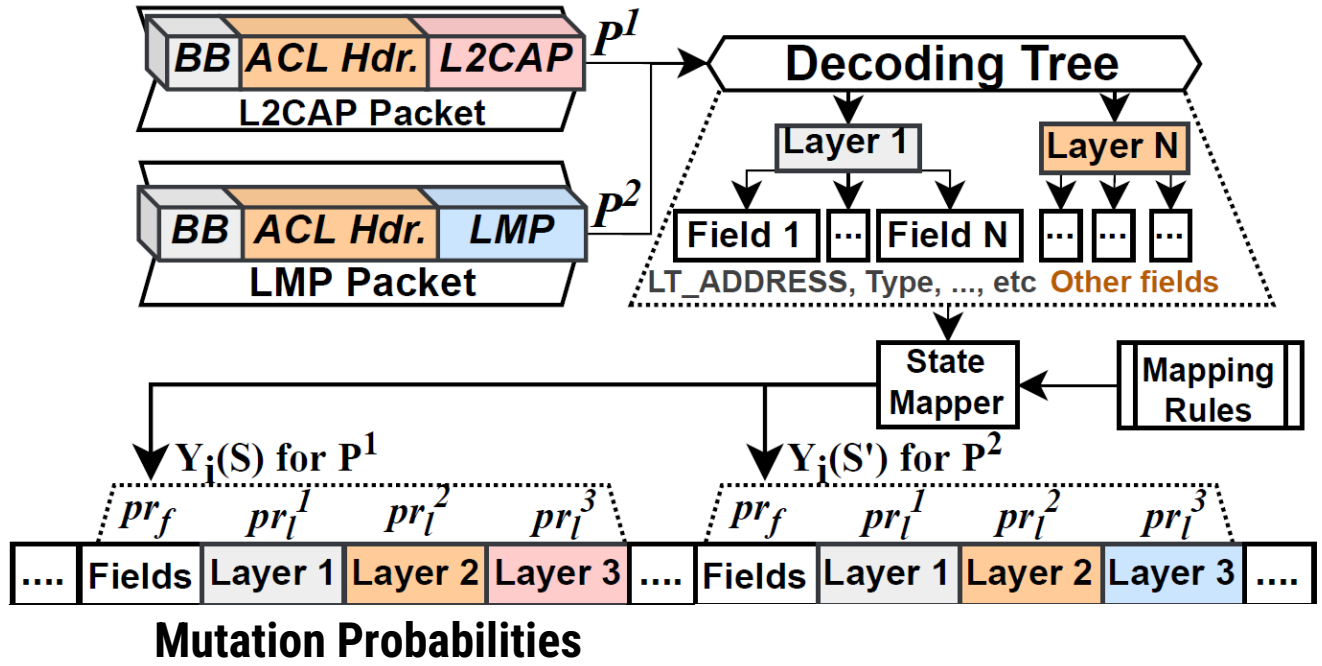
**How the array is optimized? Via Particle Swarm Optimization (PSO)**

Cost Function: Maximize Transition Coverage

(Number of transitions in state machine.)

Mutation Probabilities is an array which contains the fuzzing chances of protocol layers for each state.

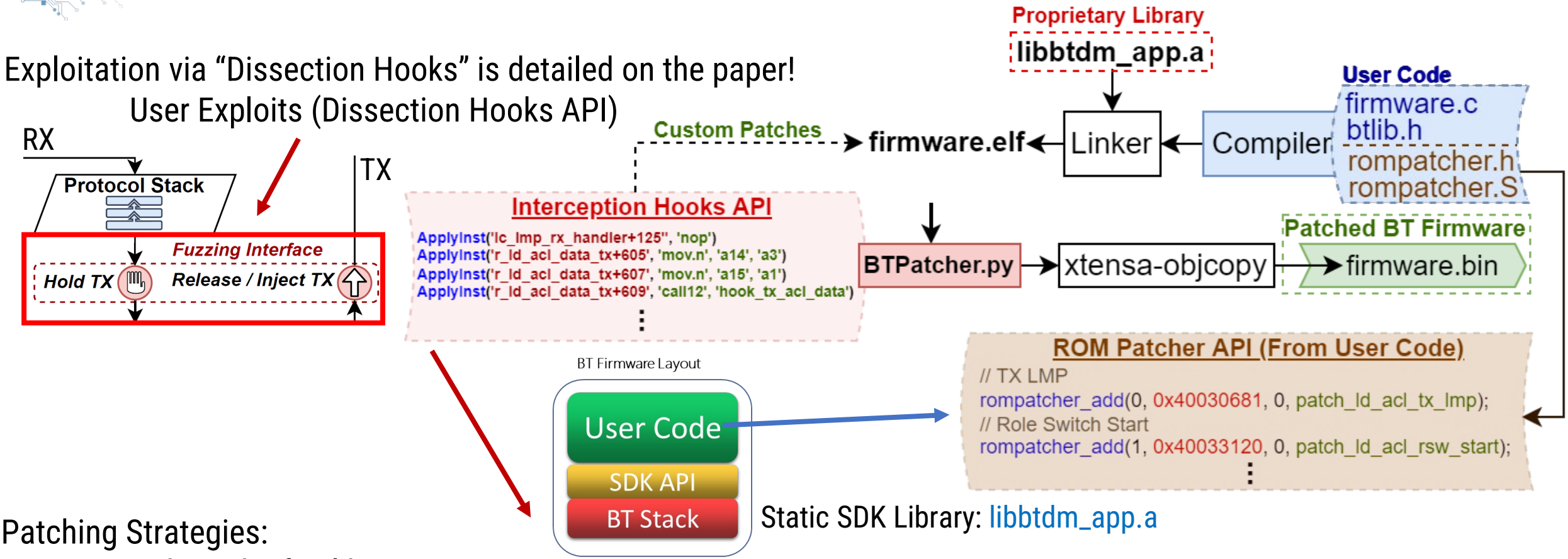
**Example:**



# 3. Fuzzing Interface - Reverse Engineering ESP32 BT Stack

Intuition: Interception! Taking over the packet control from the BT Stack

Exploitation via “Dissection Hooks” is detailed on the paper!



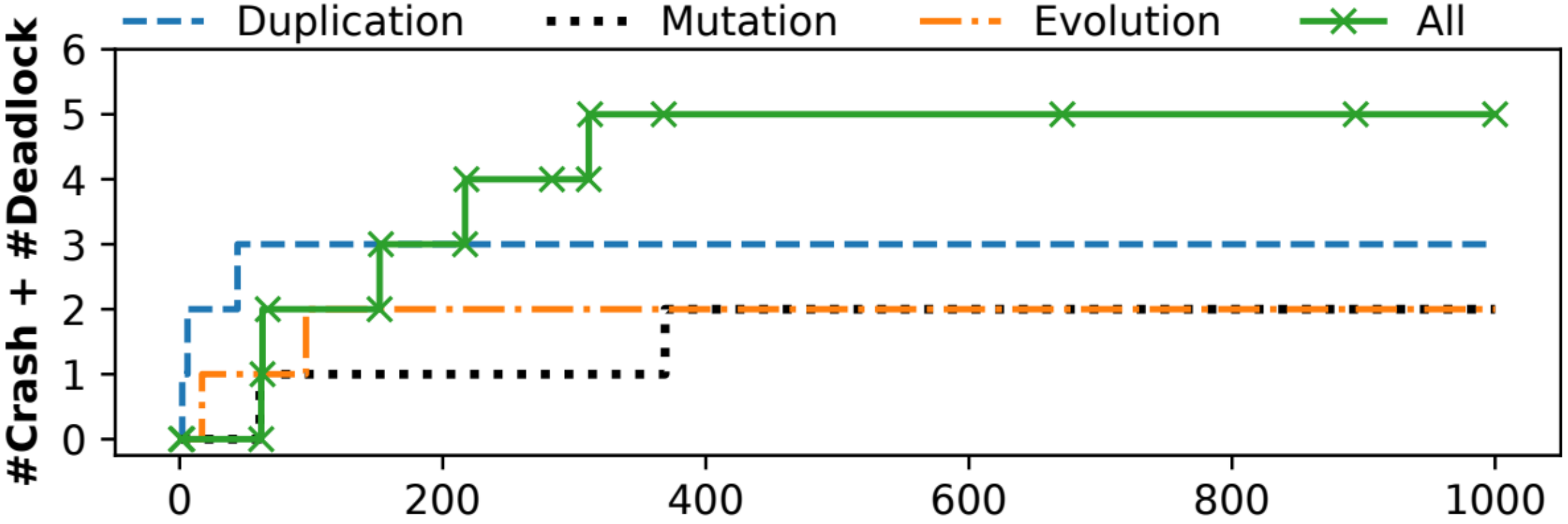

Patching Strategies:

1. Pathing the final binary;
2. Patching ROM via user code (runtime);

- Real-time Requirement: < 625us of Round Trip Time between host PC and ESP32 firmware!
  - ➢ Make use of USB High Speed (FT2232H IC)

# Evaluation – Design Choices Comparison

How different components affect the fuzzing process?




Unique crashes/deadlocks w.r.t ESP32 fuzzing iterations.

Main takeaways:

- Duplication or Mutation alone cannot find all bugs;
- Duplication + Mutation compete with each other, thus delaying the “All” variant.

# Evaluation – Summary

- Evaluated 13 BT devices from 11 vendors (Intel, Qualcomm, Cypress, etc);
  - ➢ Discovered a total of 18 unknown implementation flaws (24 CVEs);
  - ➢ Vulnerabilities classified as “Crashes” or “Deadlocks”;
  - ➢ 1 RCE on ESP32 (CVE-2021-28138);
  - ➢ 6 Bug Bounties awarded! (Intel, Espressif Systems and Xiaomi)

Summary of unknown implementation bugs and other anomalies found (Vx: Vulnerability, Ax: Non-compliance).

| Anomalies | CVE ID(s) | Device(s) |
|---|---|---|
| **V1** Feature Pages Execution | CVE-2021-28139 | ESP-WROVER-KIT |
| **V2** Invalid Public Key | CVE-2021-28138 | ESP-WROVER-KIT |
| **V3** Feature Req. Ping-Pong | CVE-2021-28137 | ESP-WROVER-KIT |
| **V4** Duplicated IOCAP | CVE-2021-28136 | ESP-WROVER-KIT |
| **V5** Feature Resp. Flooding | CVE-2021-28135<br>CVE-2021-28155<br>CVE-2021-31717 | ESP-WROVER-KIT<br>JBL TUNE500BT<br>Xiaomi MDZ-36-DB |
| **V6** LMP Auto Rate Overflow | CVE-2021-31609<br>CVE-2021-31612 | DKWT32I-A<br>BT Audio Receiver |
| **V7** LMP 2-DH1 Overflow | CVE-2021-35093 | DVK-BT900-SA |
| **V8** LMP DM1 Overflow | CVE-2021-34150 | AB32VG1 |
| **V9** Truncated LMP Accepted | CVE-2021-31613 | BT Audio Receiver<br>XY-WRBT Module |
| **V10** Invalid Setup Complete | CVE-2021-31611 | BT Audio Receiver<br>XY-WRBT Module |
| **V11** Host Conn. Flooding | CVE-2021-31785 | Xiaomi MDZ-36-DB |
| **V12** Same Host Connection | CVE-2021-31786 | Xiaomi MDZ-36-DB |
| **V13** AU Rand Flooding | CVE-2021-31610<br>CVE-2021-34149<br>CVE-2021-34146 | AB32VG1<br>CC256XCQFN-EM<br>CYW920735Q60EVB |
| **V14** Invalid Max Slot Type | CVE-2021-34145 | CYW920735Q60EVB |
| **V15** Max Slot Length Overflow | CVE-2021-34148 | CYW920735Q60EVB |
| **V16** Invalid Timing Accuracy | CVE-2021-34147<br>CVE-2021-30348<br>CVE-2021-33139 | CYW920735Q60EVB<br>Pocophone F1<br>Intel AX200 |
| **V17** Paging Scan Deadlock | CVE-2021-33155 | Intel AX200 |
| **V18** SDP Element Size Overflow | Pending | Beken BK3260N |
| **A1** Accepts Lower LMP Length | N.A | **All, except ESP32** |
| **A2** Accepts Higher LMP Length | N.A | **All tested devices** |
| **A3** Multiple Encryption Start | N.A | Xiaomi MDZ-36-DB |
| **A4** Ignore Role Switch Reject | N.A | Pocophone F1 |
| **A5** Invalid Response | N.A | Intel AX200<br>DVK-BT900-SA |
| **A6** Ignore Encryption Stop | N.A | CYW920735Q60EVB |

# Evaluation - Extensions

- Created Wi-Fi AP and BLE Host fuzzer variants;
  - ➢ Required changes: Protocol Stack, Fuzzing Interface and Mapping Rules;

- Wi-Fi and BLE Host fuzzing variants discovered other 6 unknown bugs;

Summary of unknown flaws found by fuzzing extension.

| Extension | Stack | Target | Vulnerability | CVE (New) |
|---|---|---|---|---|
| **BLE Host** | Bluekitchen | ESP32 | Null Dereference | CVE-2022-26604 |
| | | Telink TLSR8258 | Re-Advertisement DoS | CVE-2022-26602 |
| | | NXP KW41Z | – | – |
| | | TI CC2540 | – | – |
| **Wi-Fi AP** | Hostapd | ESP32 | EAP Heap Overflow | CVE-2022-26603 |
| | | | Association Deadlock | CVE-2022-26600 |
| | | ESP8266 | Association Crash | CVE-2022-26601 |
| | | Rasp. Pi 3 B | Probe Resp. Deadlock | CVE-2022-26599 |
| | | One Plus 5T | – | – |

# Conclusion

Disclosure: https://braktooth.com/



- Impact: Exposed firmware bugs and non-compliances in hundreds BT SoC models, affecting IoTs, Laptops, Smartphones and Audio products across the industry.
  - ➢ Independent testing has revealed other SoC vendors to be affected such as Mediatek, Samsung, Airoha, Apple;
  - ➢ Highlighted the need for more security-oriented Over-the-Air testing tools;

- Lower the cost for Bluetooth Classic experimentation with ESP32 (~5 USD);
- Fuzzer design can be generalized to other protocols (e.g: Wi-Fi, BLE Host);
- Requires proper monitoring configuration, otherwise crashes are missed.
  - ➢ Expected disadvantage for a OTA fuzzer.

Code Availability:

ESP32 Patching Framework: https://github.com/Matheus-Garbelini/esp32_firmware_patching_framework

Fuzzer Runtime and PoC: https://github.com/Matheus-Garbelini/braktooth_esp32_bluetooth_classic_attacks

Fuzzer Source Code (academic research only): https://src.braktooth.com/