BLuEMan: A Stateful Simulation-based Fuzzing Framework for Open-Source RTOS Bluetooth Low Energy Protocol Stacks

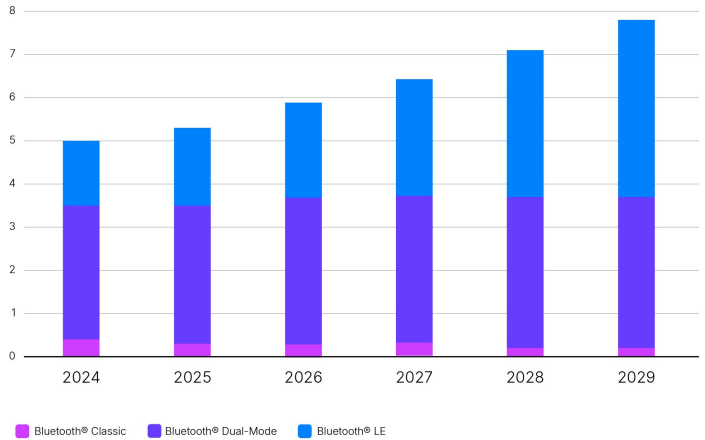
Wei-Che Kao, Yen-Chia Chen, Yu-Sheng Lin, Yu-Cheng Yang, Chi-Yu Li, Chun-Ying Huang
National Yang Ming Chiao Tung University

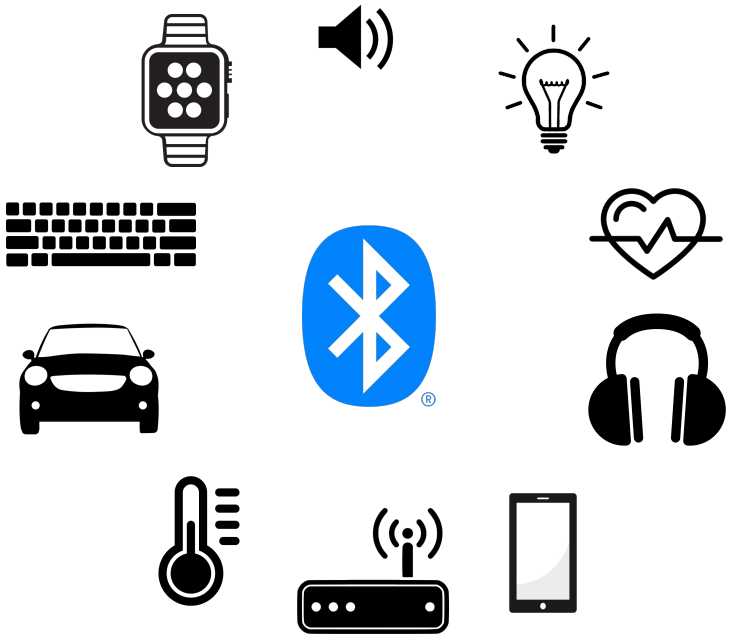

國立陽明交通大學
NATIONAL YANG MING CHIAO TUNG UNIVERSITY

Popularity of BLE

Bluetooth® enabled device shipments by controller configuration

numbers in billions

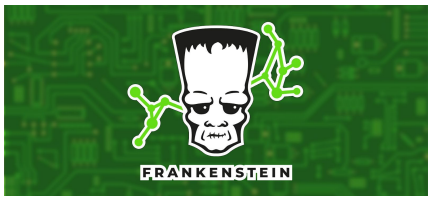




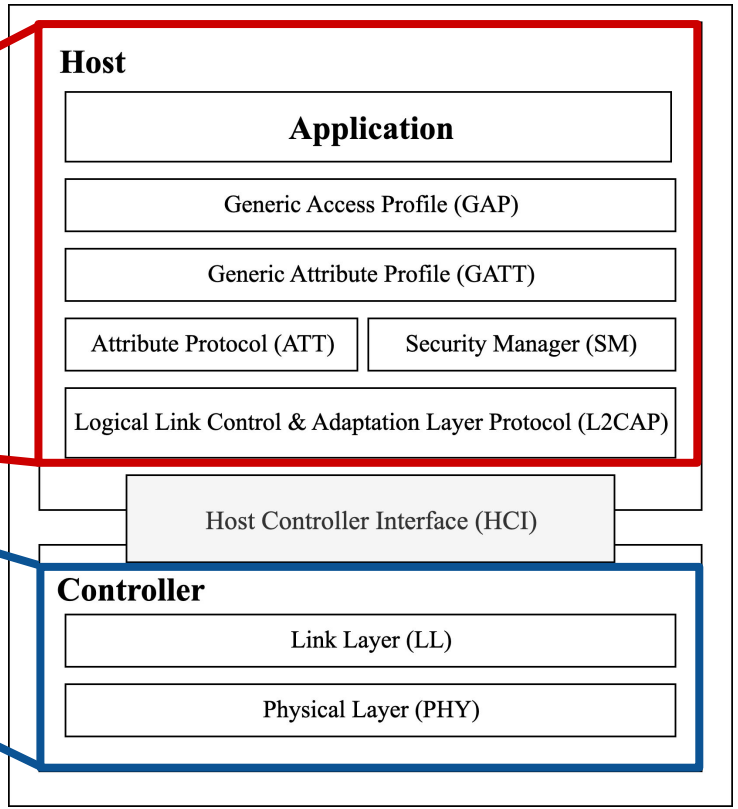
[1] Bluetooth® market update 2025

Bluetooth Vulnerabilities

- Past research has shown the impact of Bluetooth vulnerabilities
  - RCE: Gain full control of the devices
  - DoS: Device unusable or permanently disabled

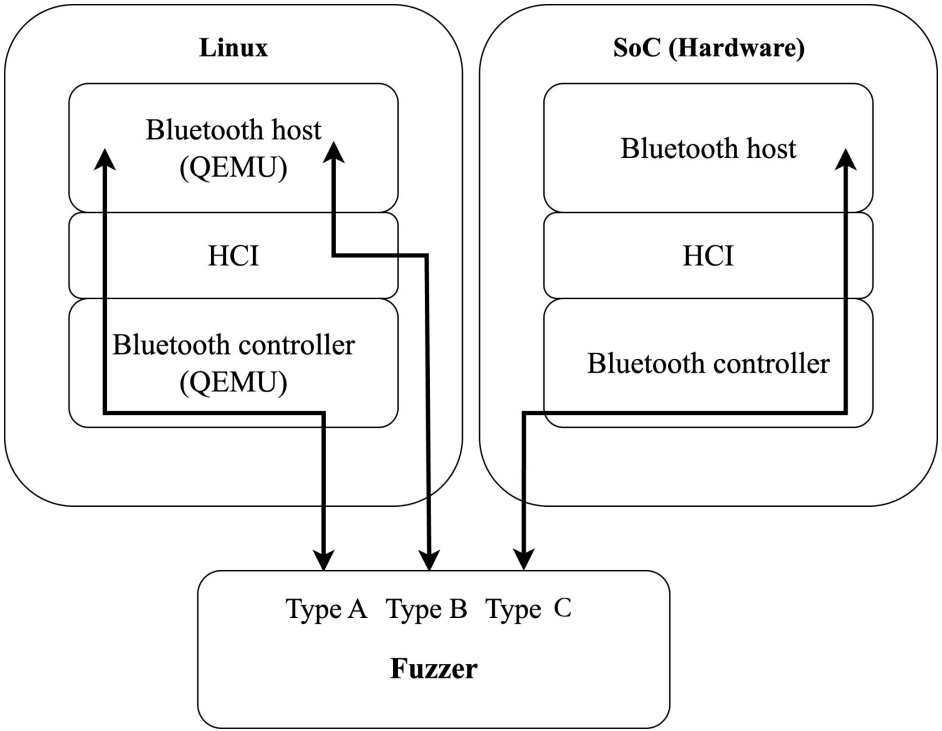


Overview of BLE Stack

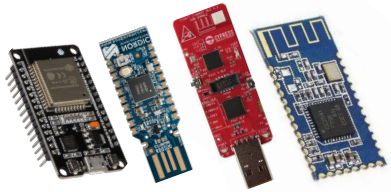


Type of BLE Fuzzer

- Type A: Emulation-based BLE full stack fuzzer
- Type B: Emulation-based BLE host fuzzer
- Type C: Hardware-based BLE full stack fuzzer



Challenges



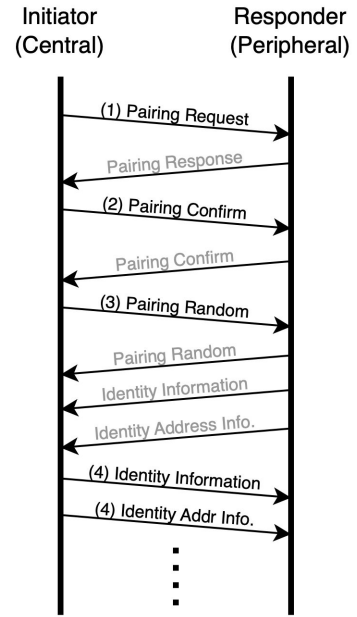
setup cost, coverage,
speed, crash verification

1. Runtime Scalability

2. High-quality Seed

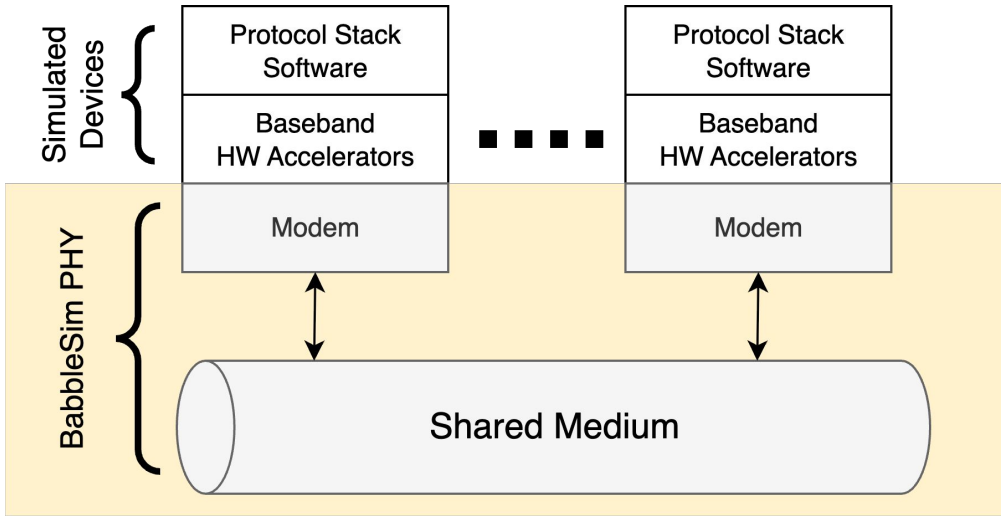
3. Complexity of BLE

Core 4.0 - 6.1/
BLE Mesh/
HRP/DIS/HTP/F
MP/HID/OTS/…



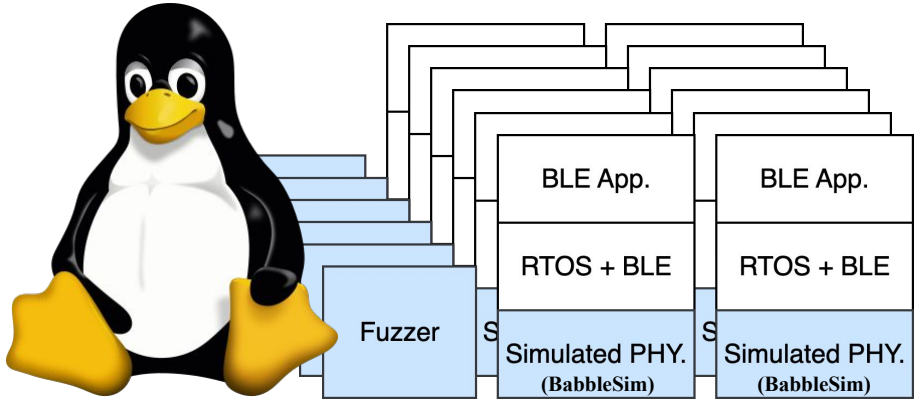
Solution - Runtime Scalability

- **Runtime Scalability**
- High-quality Seed
- Complexity of BLE



Solution - Runtime Scalability

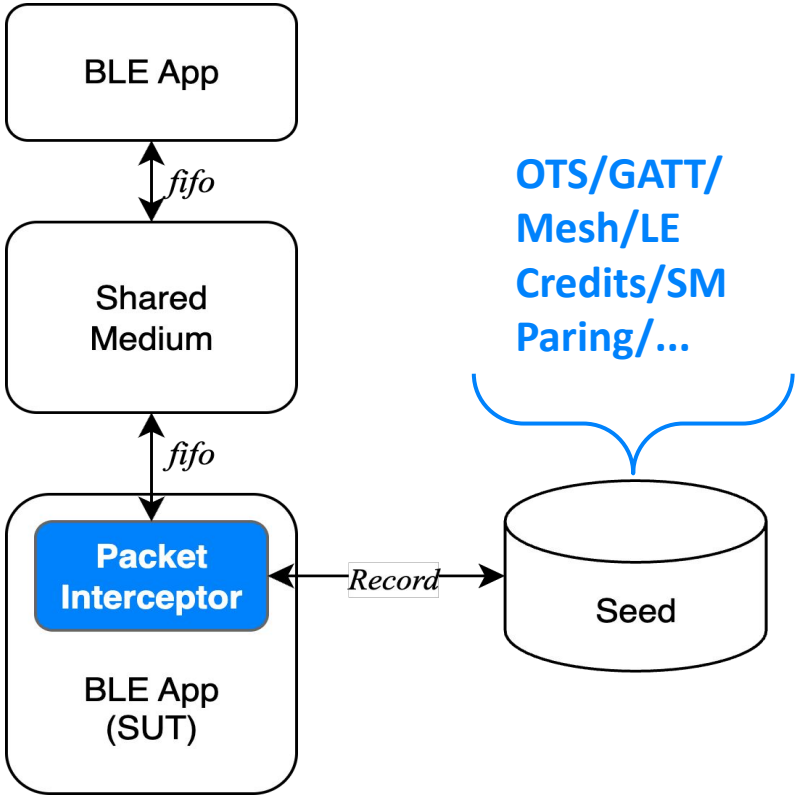
- **Runtime Scalability**
- High-quality Seed
- Complexity of BLE



Simultaneously fuzz
Different BLE implementations
Different BLE applications

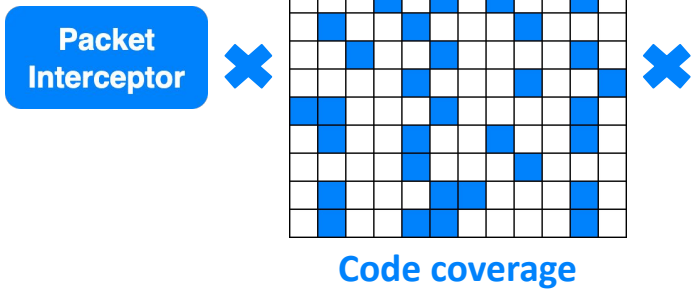
Solution - High-quality Seed

- Runtime Scalability
- **High-quality Seed**
- Complexity of BLE



Solution - Complexity of BLE

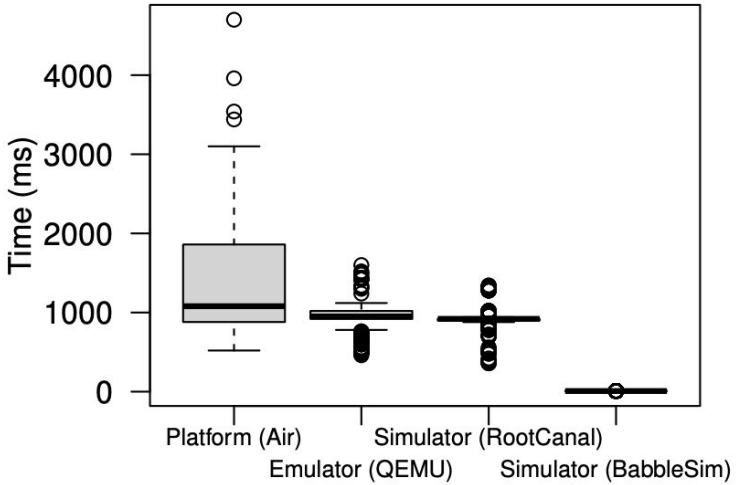
- High-quality Seed
- Runtime Scalability
- **Complexity of BLE**




```
1:  L' ← ∅    ▷ The array containing the detected protocols.
2:  for p ∈ L do
3:      if p.R(pkt) is true then
4:          p.w ← UpdateWeight(p)
5:          add p to L'
6:      end if
7:  end for
8:  n ← number of elements in L'
9:  add {∅, w_x, ∅} to L'
10: W ← ∑ p.w ∀p ∈ L'
11: m ← −1                                  ▷ Mutation index
12: pr ← draw a probability from [0, 1]
13: w_L ← 0
14: for i = 0 to (n − 1) do
15:     w_U ← w_L + L'[i].w/W
16:     if w_L ≤ pr < w_U then
17:         m ← i                           ▷ Assign mutation index
18:     end if
19:     w_L ← w_U
20: end for
21: if m > −1 then
22:     pkt' ← L'[m].M(pkt)
23: else
24:     pkt' ← pkt
25: end if
26: return pkt'
```

Evaluation

- To select a fuzzing platform, we compare BabbleSim, RootCanal, and physical devices
- BabbleSim is approximately 100 times faster than other platform



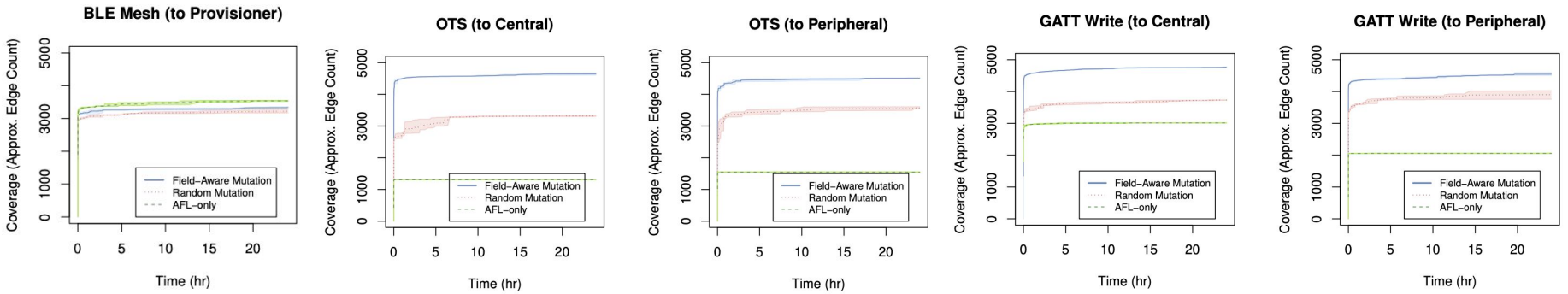
Evaluation

- We compare the packet transmission rate of the BLE fuzzing research works
- Our approach is 18.0× to 162.3× faster than existing approaches

Name	Approach	Rate (packets/min)
SweynTooth [16]	Platform-based	119
BTFuzz [23]	Simulation-based	1,073
BLuEMan (Ours)	Simulation-based	19,315

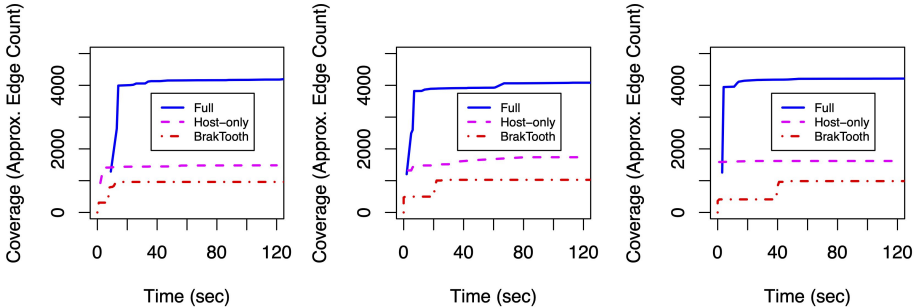
Evaluation

- We compare our stackable mutation architecture to AFL-only mutator
- Stackable mutation architecture achieves 6.14%–256.49% higher coverage than AFL-only mutator

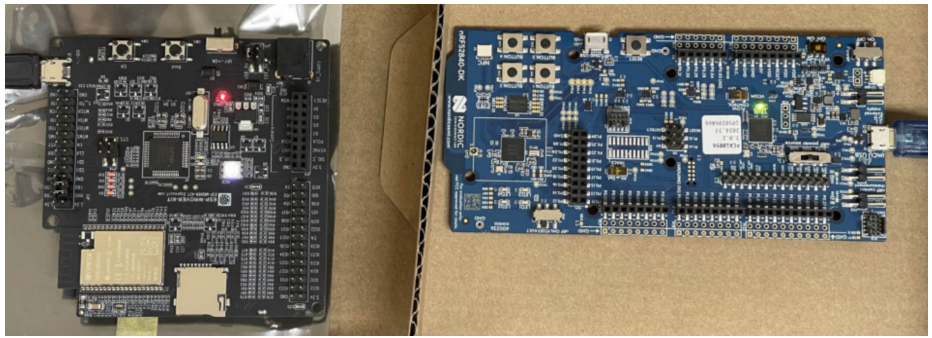


Evaluation

- We compare our work to the ESP-WROVER-KIT running BrakTooth firmware
- BLuEMan outperforms BrakTooth by 1.54x–1.69x in coverage



(a) GATT Write. (a) Heart Rate. (a) OTS.



Conclusion

- By combining an RTOS with a PHY simulator, BLuEMan provides an efficient, high fidelity and scalable platform for BLE fuzzing
- It features a novel MITM architecture for automated seed collection and a packet-driven workflow that streamlines state management
- it uncovered four new vulnerabilities spanning different layers of the BLE stack

Thank you !

Github

Zenodo

Paper