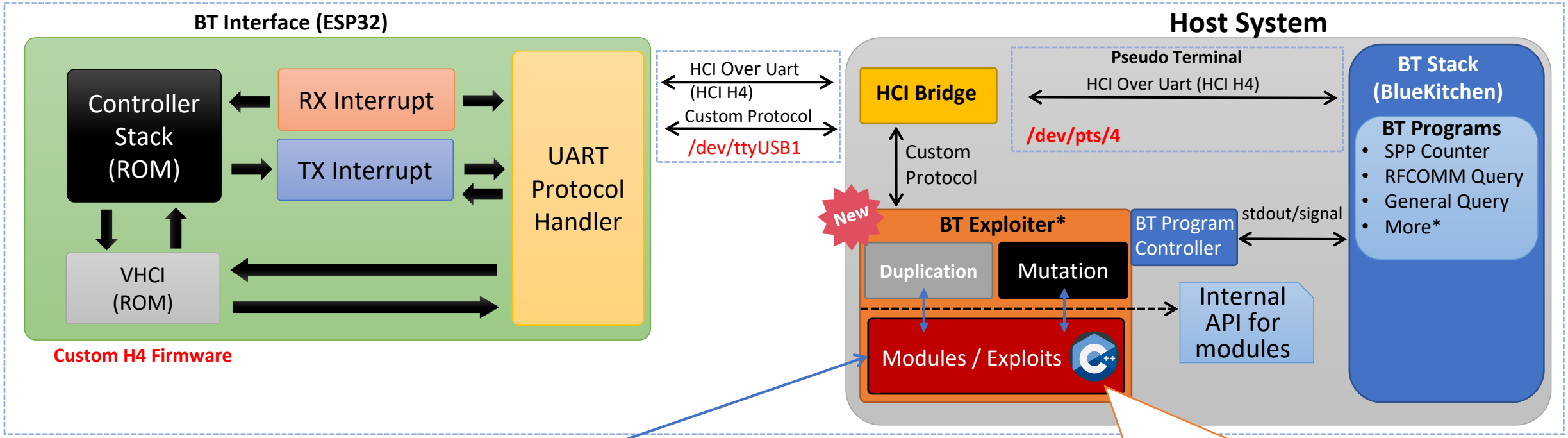
ESP32 Exploitation Module Design - Motivation

Fuzzing, Validation and State Mapper is disabled



Exploitation Module as a simplified version of the fuzzer

- The modules sources can be released to the public;
- Mutation and duplication can be triggered manually from inside the modules;
- Modules are automatically recompiled from source.

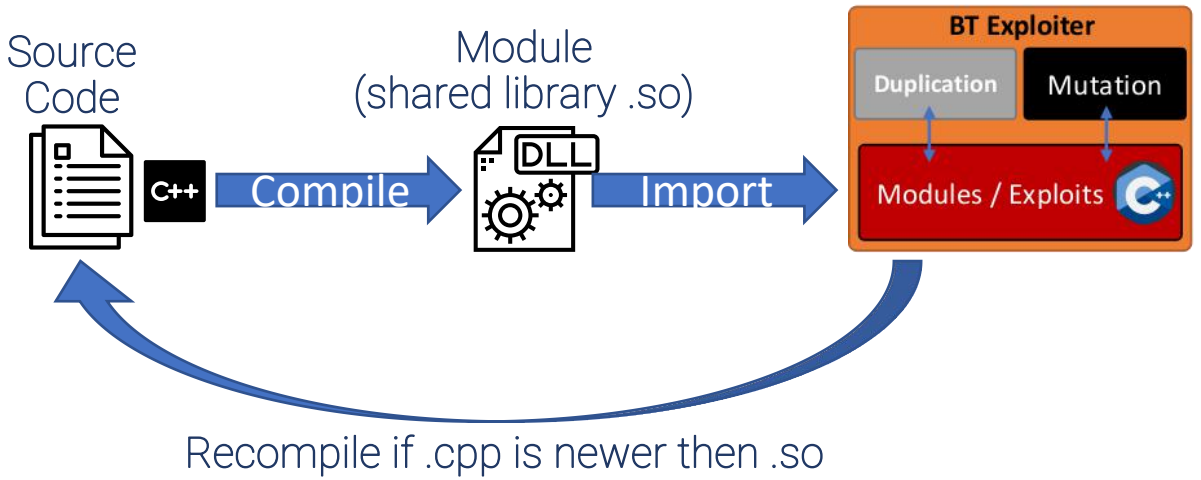
C++ modules at **modules/exploits**

- knob.so
- invalid_feature_page_execution.so
- duplicated_iocap.so
- etc, ...

ESP32 Exploitation Module Design - Module Generation

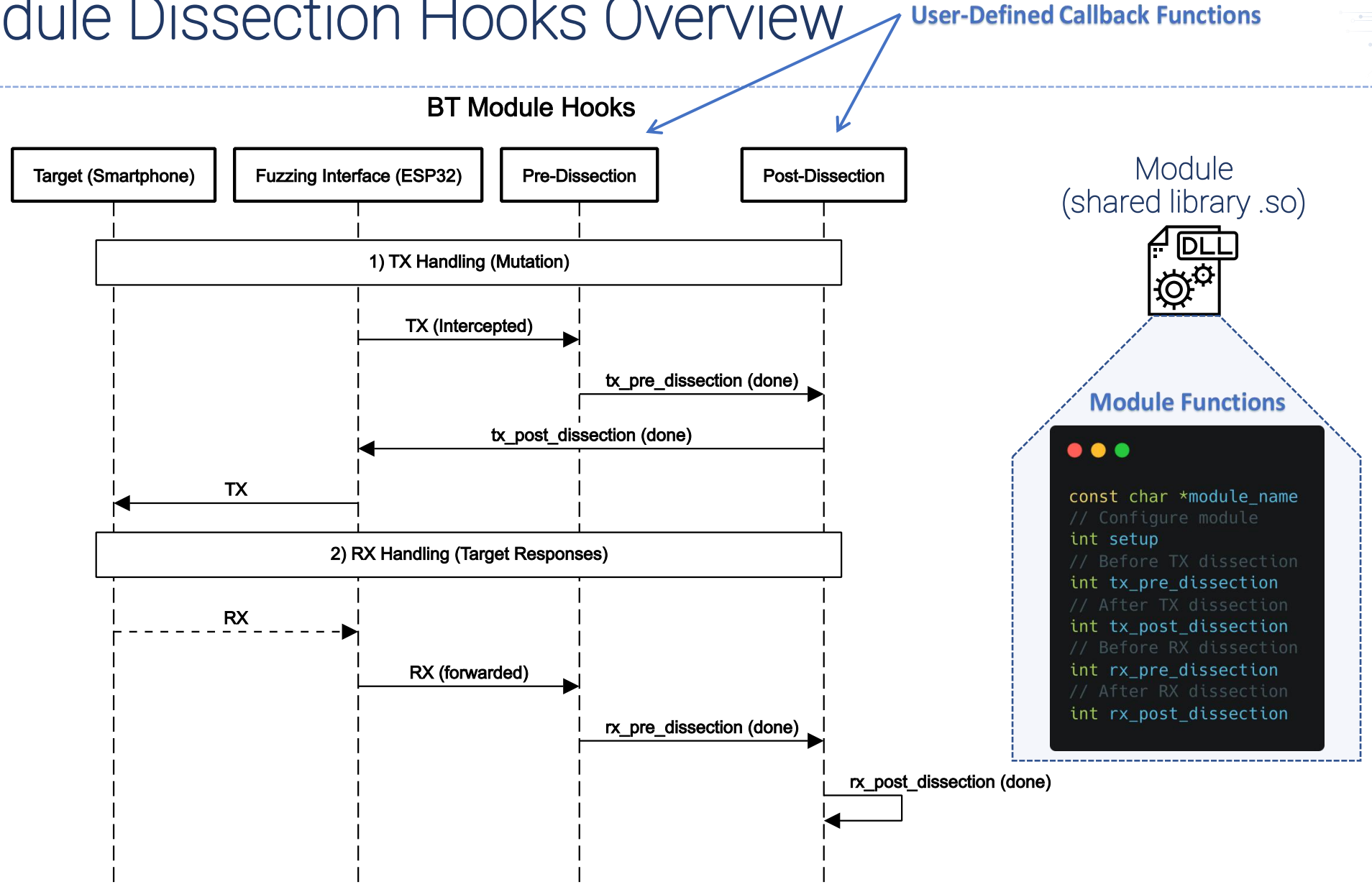
During startup:

1. Check for (.so) modules on modules/exploits folder and load them
2. Compile (.cpp) modules without corresponding (.so) module
3. Recompile outdated (.so) module from newer corresponding (.cpp) file

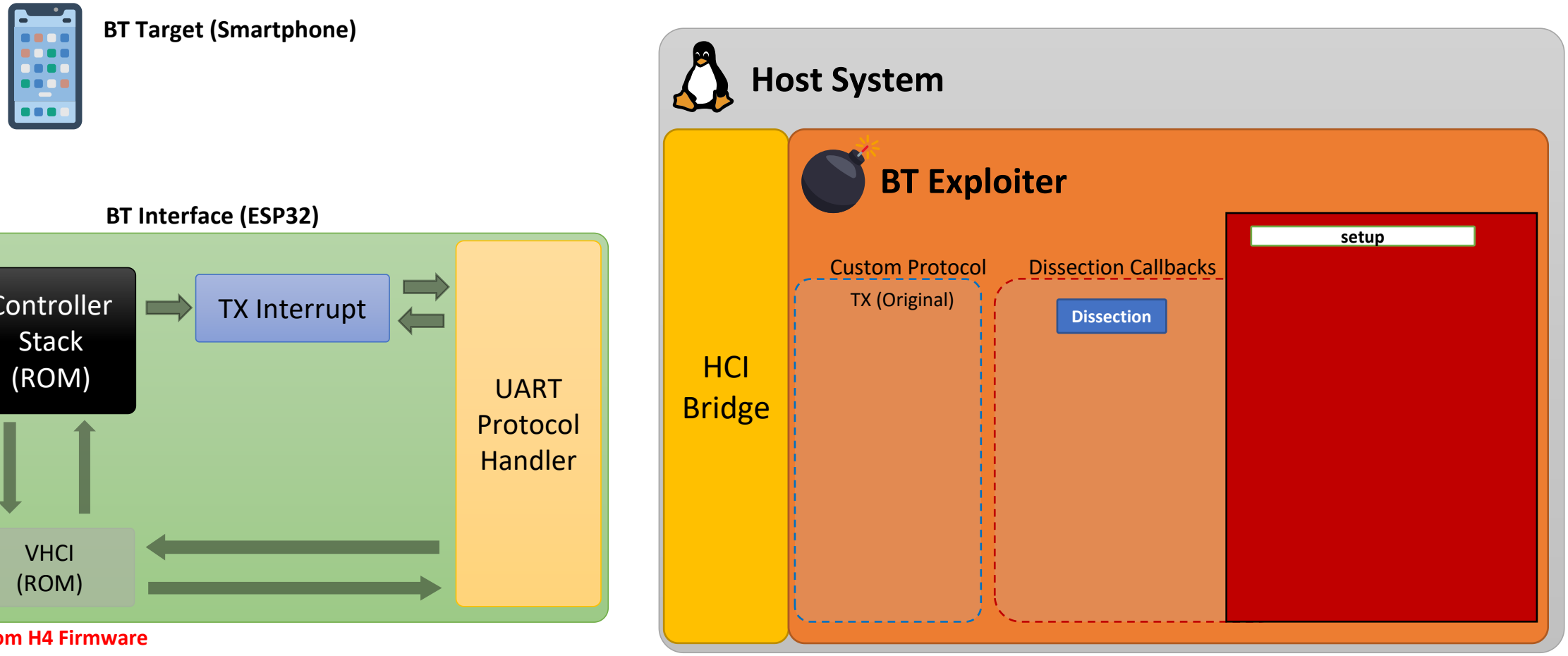


ESP32 Exploitation Module Design - Diagram

Module Dissection Hooks Overview



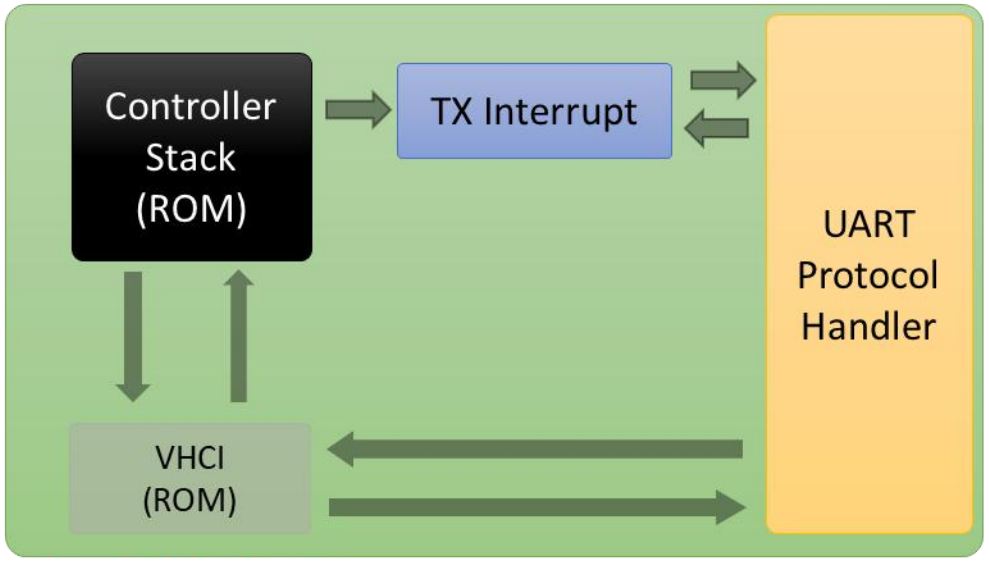
ESP32 Exploitation Module Design – How it works?
A) Dissection Events - TX Mutation and Injection (Duplication)



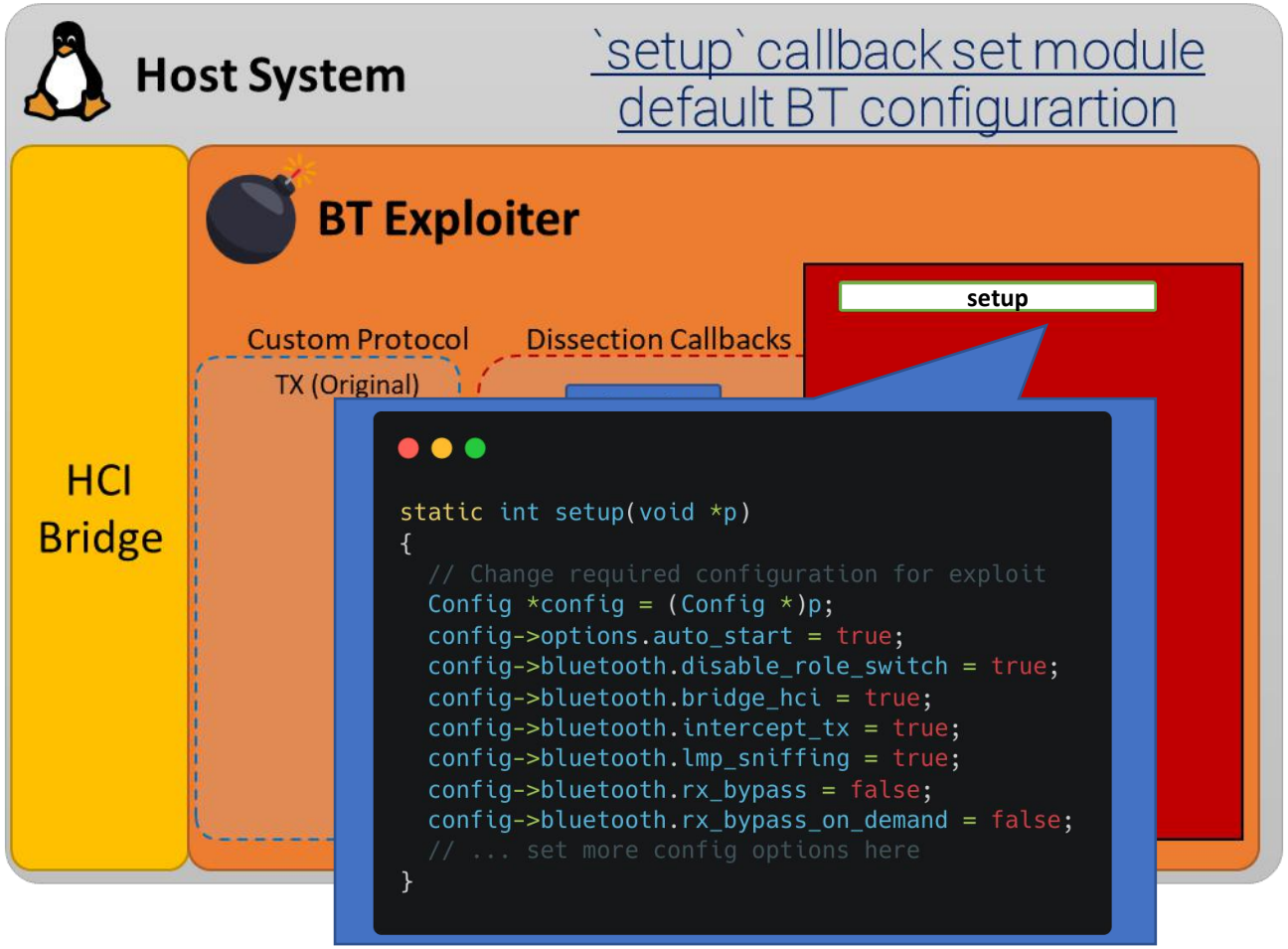
ESP32 Exploitation Module Design – How it works?
A) Dissection Events - TX Mutation and Injection (Duplication)

BT Target (Smartphone)

BT Interface (ESP32)



Custom H4 Firmware



ESP32 Exploitation Module Design – How it works?
B) Dissection Events - TX Mutation and Injection (Duplication)

BT Target (Smartphone)

BT Interface (ESP32)

Controller Stack (ROM)

TX Interrupt

UART Protocol Handler

VHCI (ROM)

Custom H4 Firmware

Host System

More callbacks!!!

HCI Bridge

BT Exploiter

Custom Protocol

TX (Original)

Dissection Callbacks

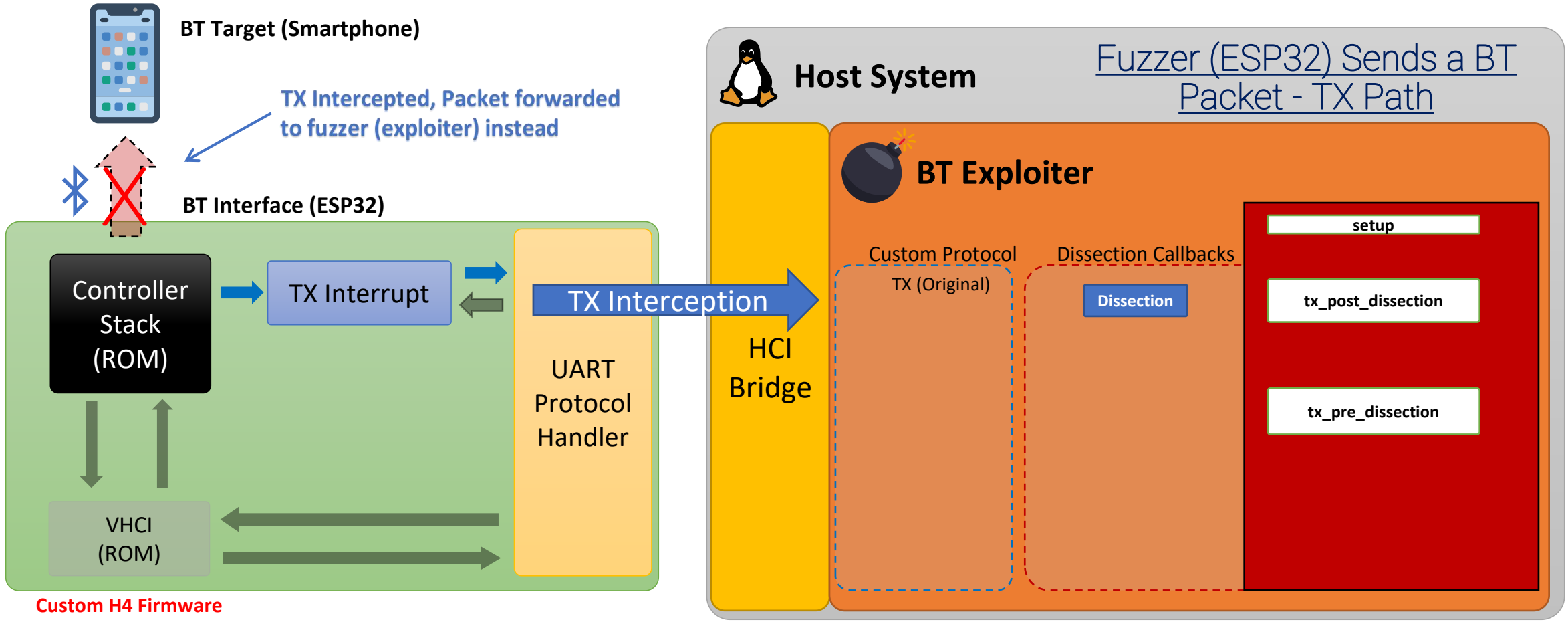
Dissection

setup

tx_post_dissection

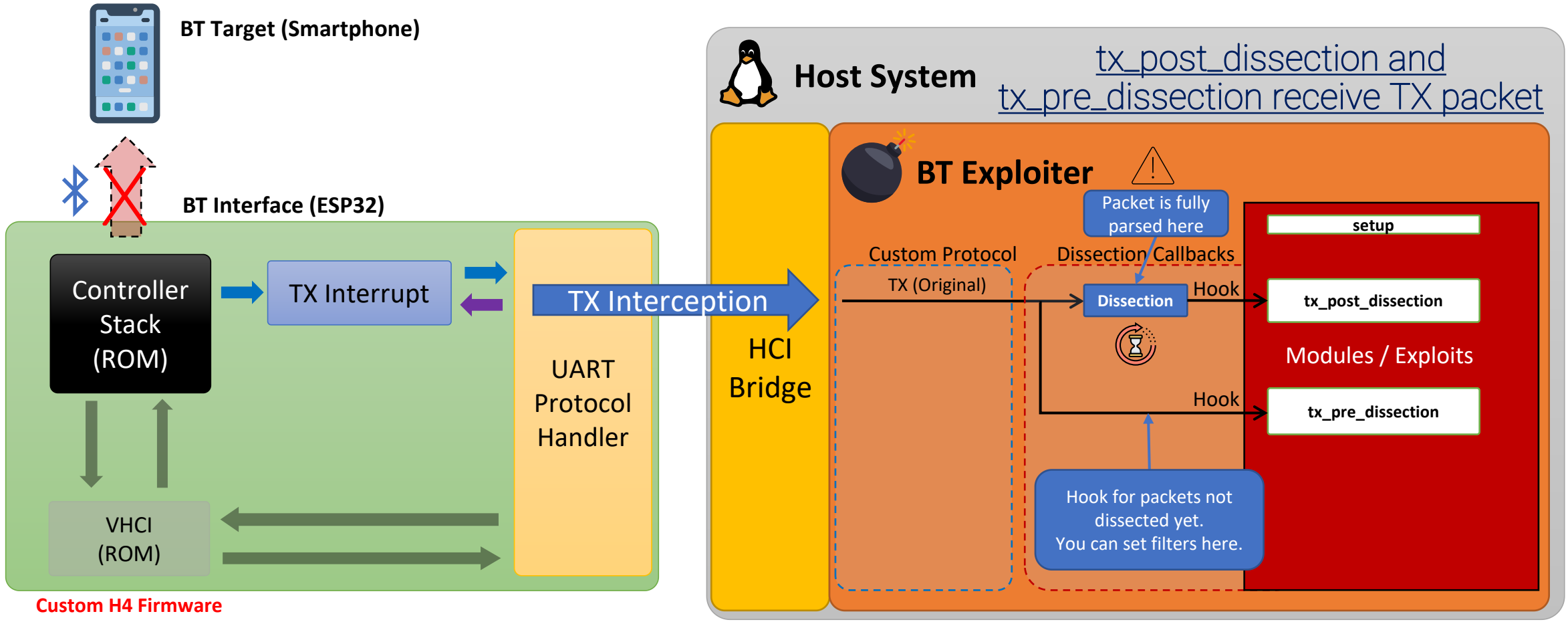
tx_pre_dissection

ESP32 Exploitation Module Design – How it works?
B) Dissection Events - TX Mutation and Injection (Duplication)



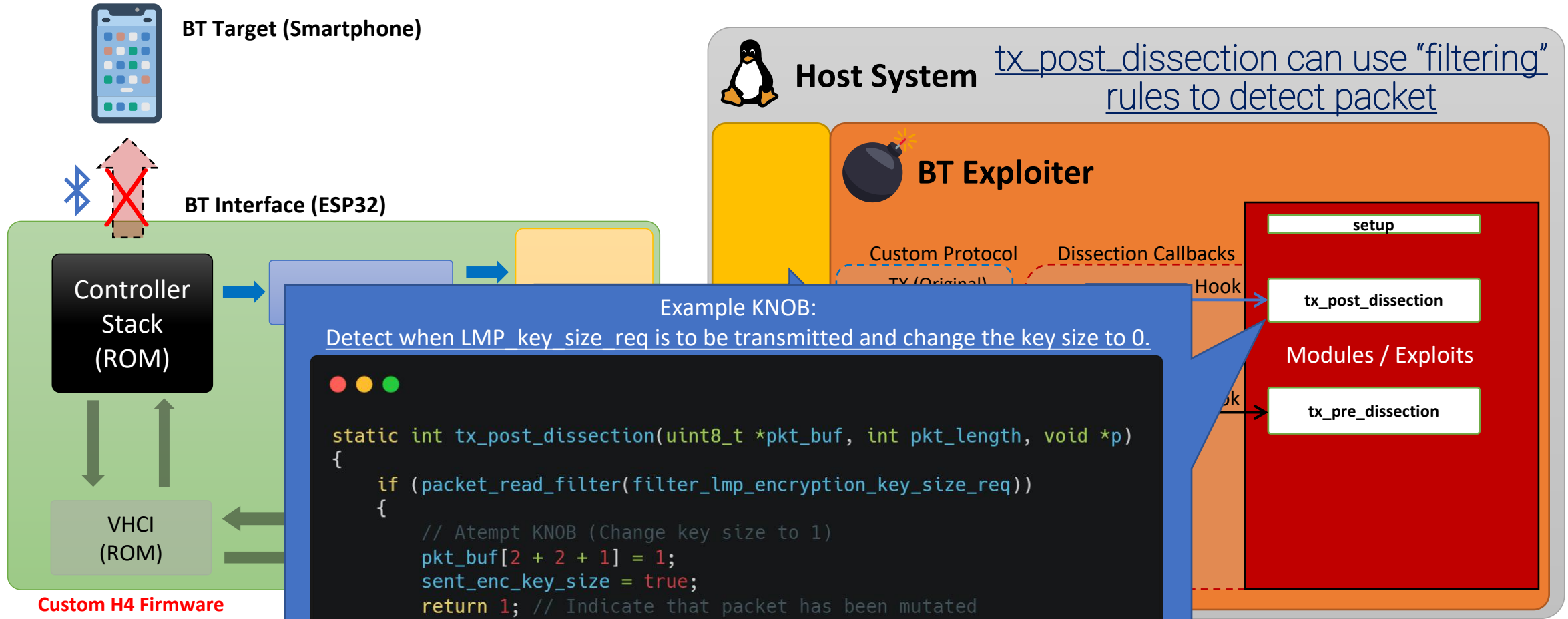
1. Controller sends a BT packet to target (Interception occurs)

ESP32 Exploitation Module Design – How it works?
C) Dissection Events - TX Mutation and Injection (Duplication)



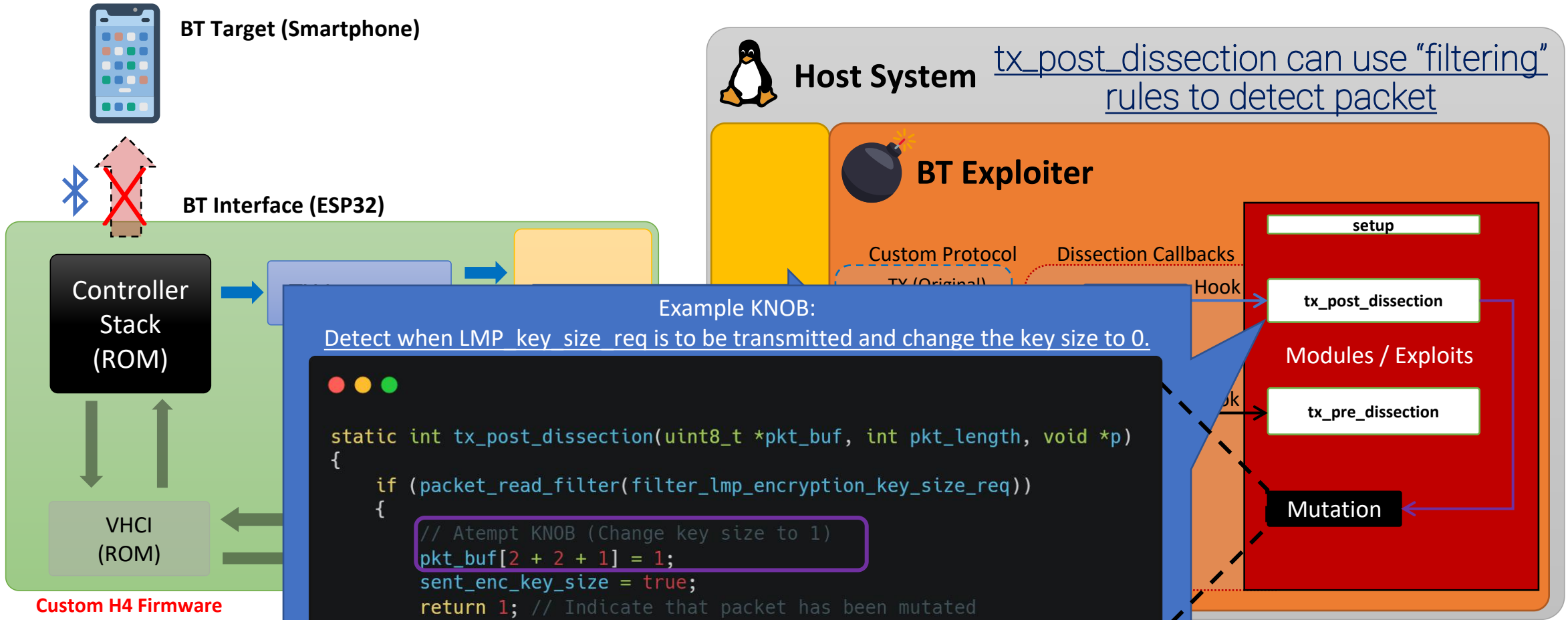
2. Module intercept TX packet on pre or post dissection hooks

ESP32 Exploitation Module Design – How it works?
C) Dissection Events - TX Mutation and Injection (Duplication)



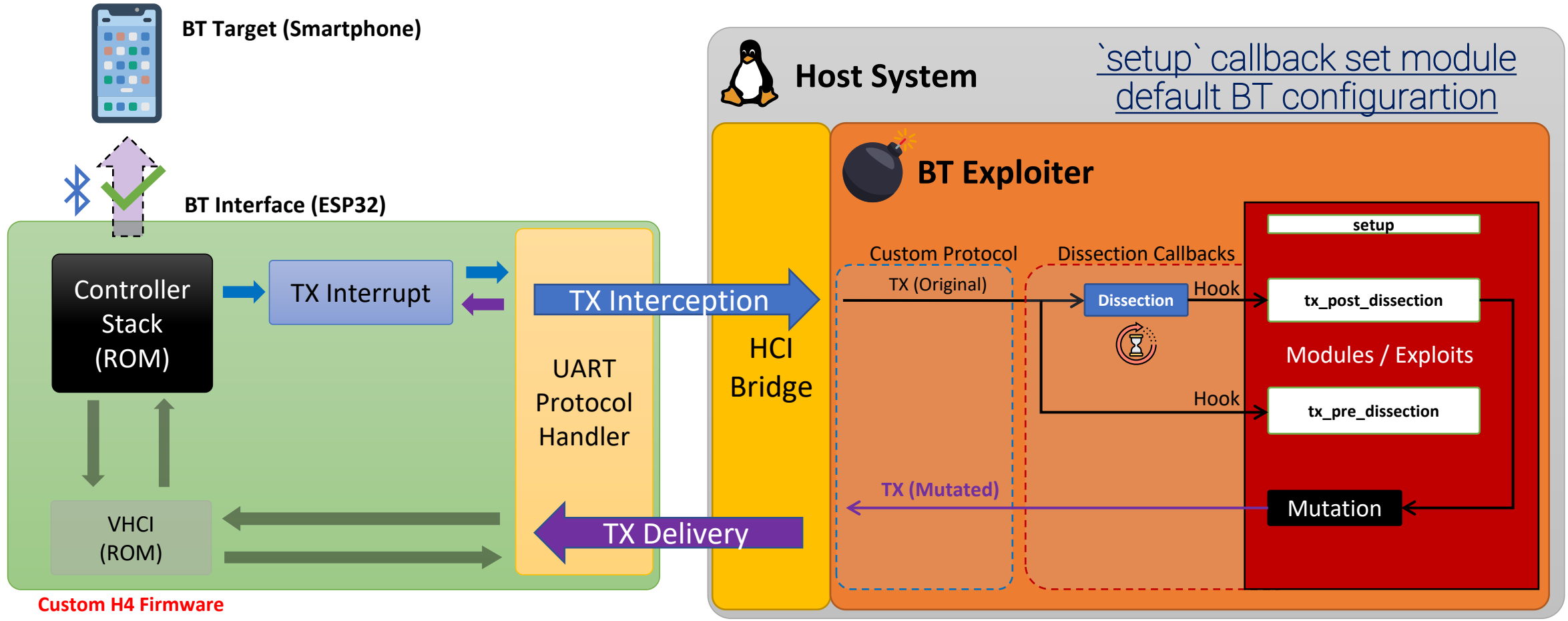
2. Modu dissection hooks

ESP32 Exploitation Module Design – How it works?
C) Dissection Events - TX Mutation and Injection (Duplication)



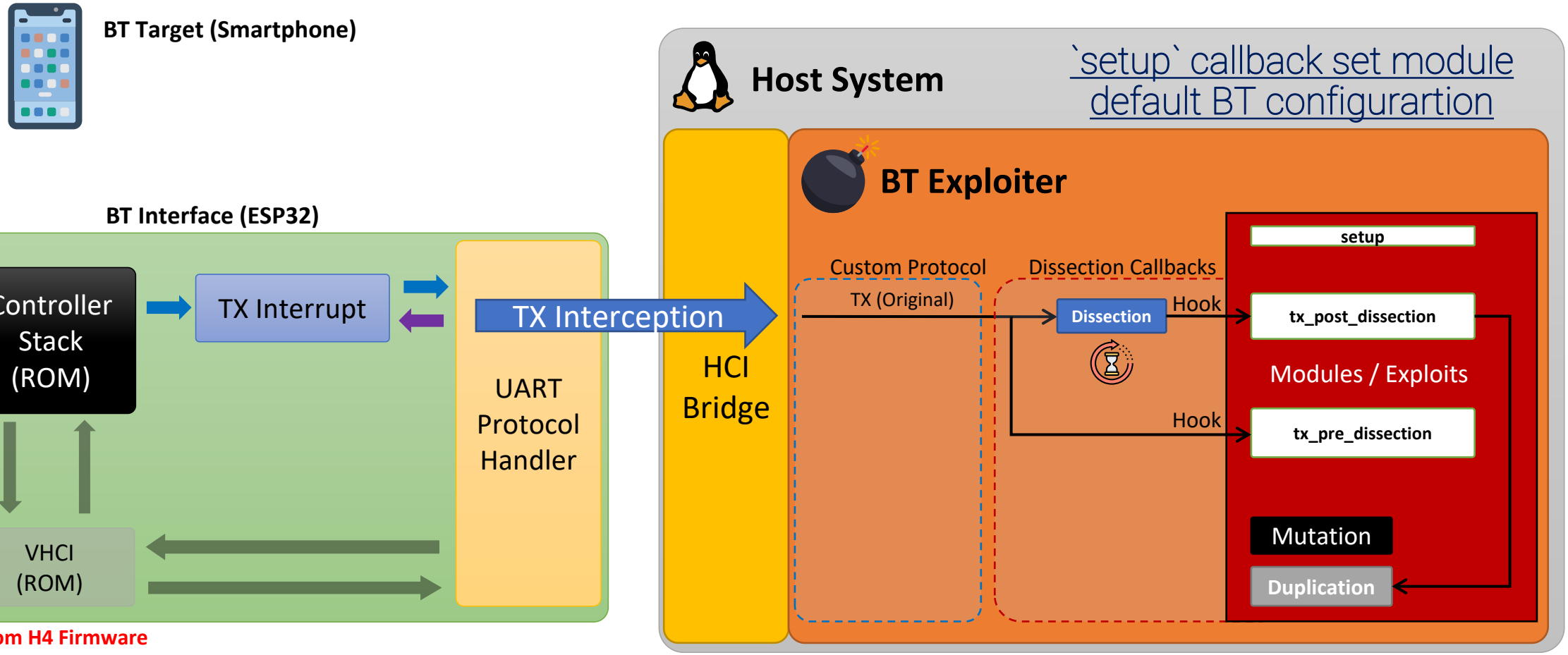
2. Modu dissection hooks

ESP32 Exploitation Module Design – How it works?
D) Dissection Events - TX Mutation and Injection (Duplication)



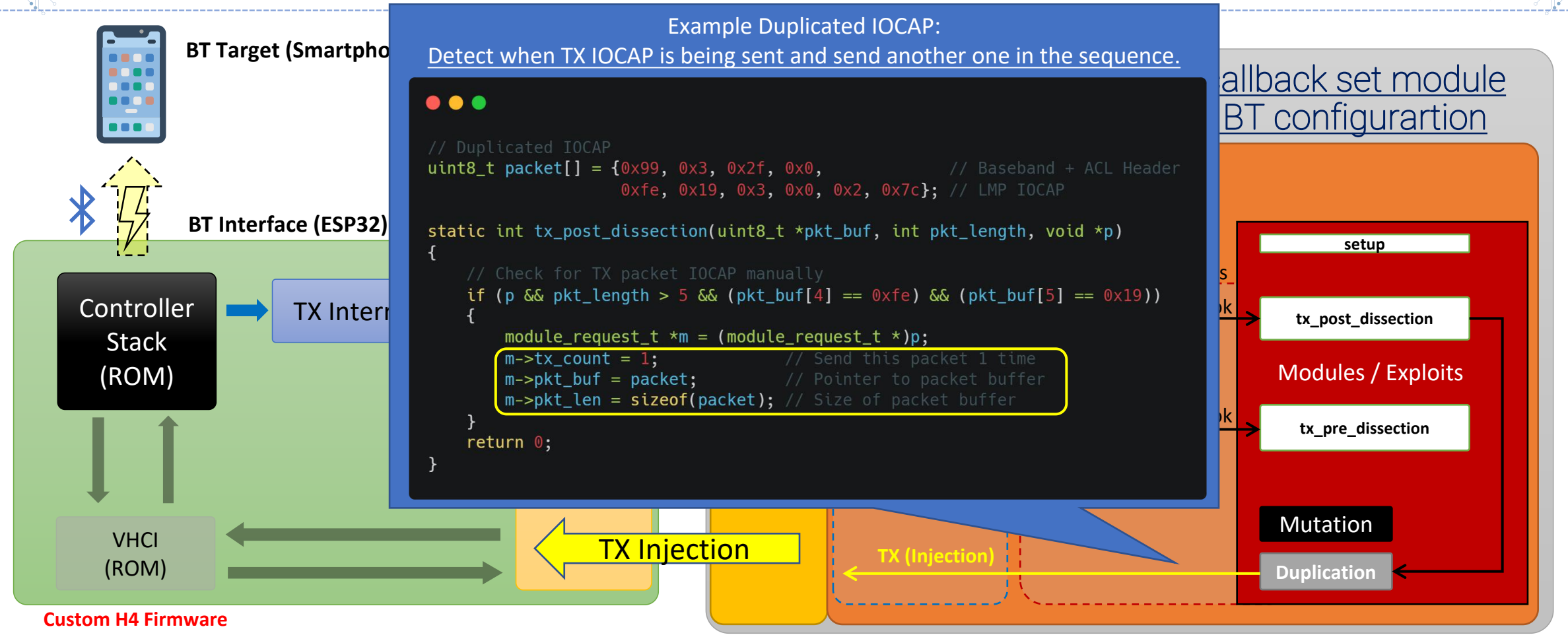
3. Module can mutate TX packet before it is sent to target

ESP32 Exploitation Module Design
E) Dissection Events - TX Mutation and Injection (Duplication)



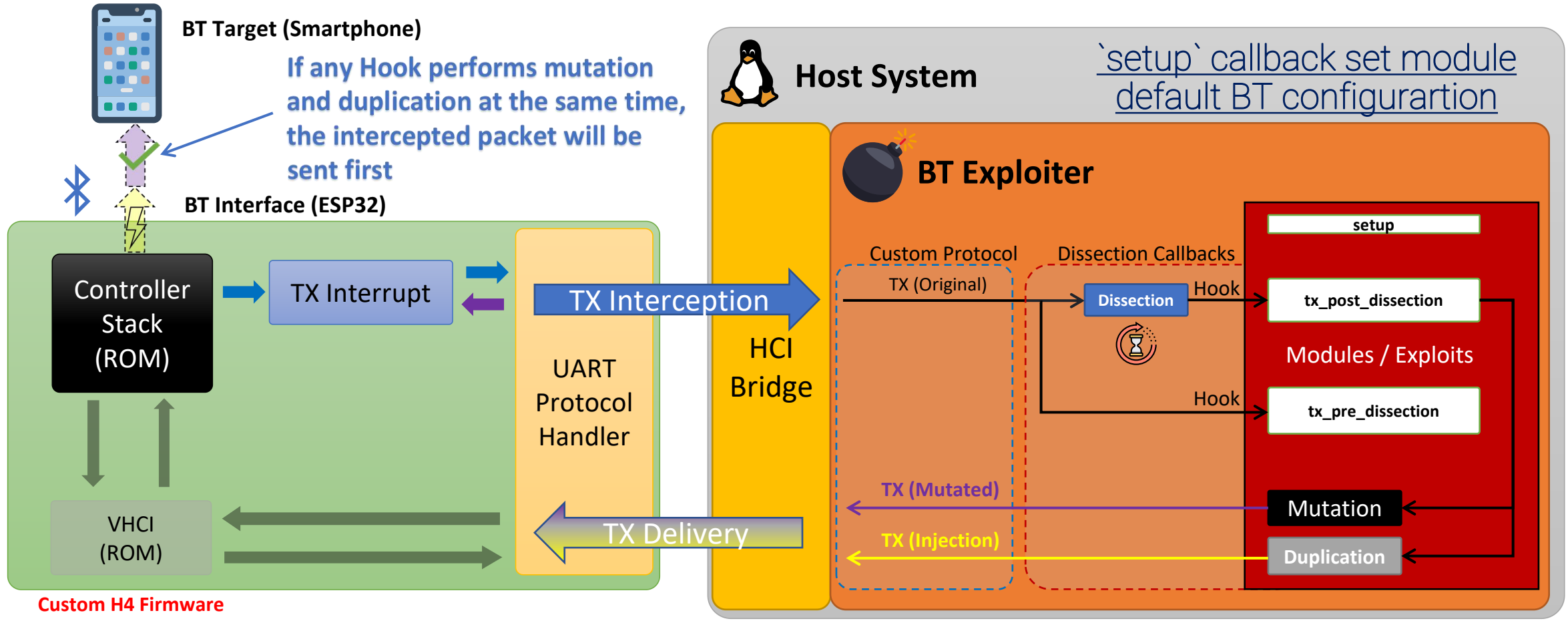
4. Module can also inject packets

ESP32 Exploitation Module Design – How it works?
E) Dissection Events - TX Mutation and Injection (Duplication)



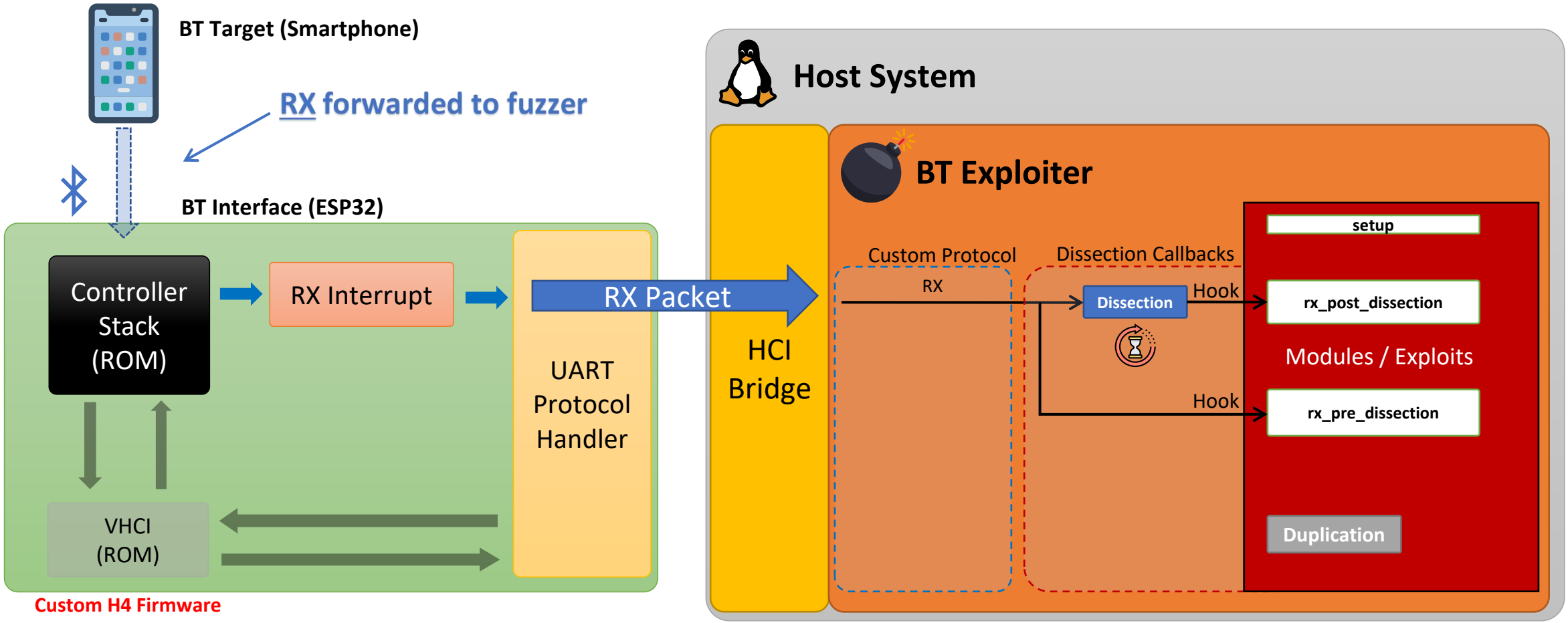
4. Module can also inject packets

ESP32 Exploitation Module Design – How it works?
F) Dissection Events - TX Mutation and Injection (Duplication)



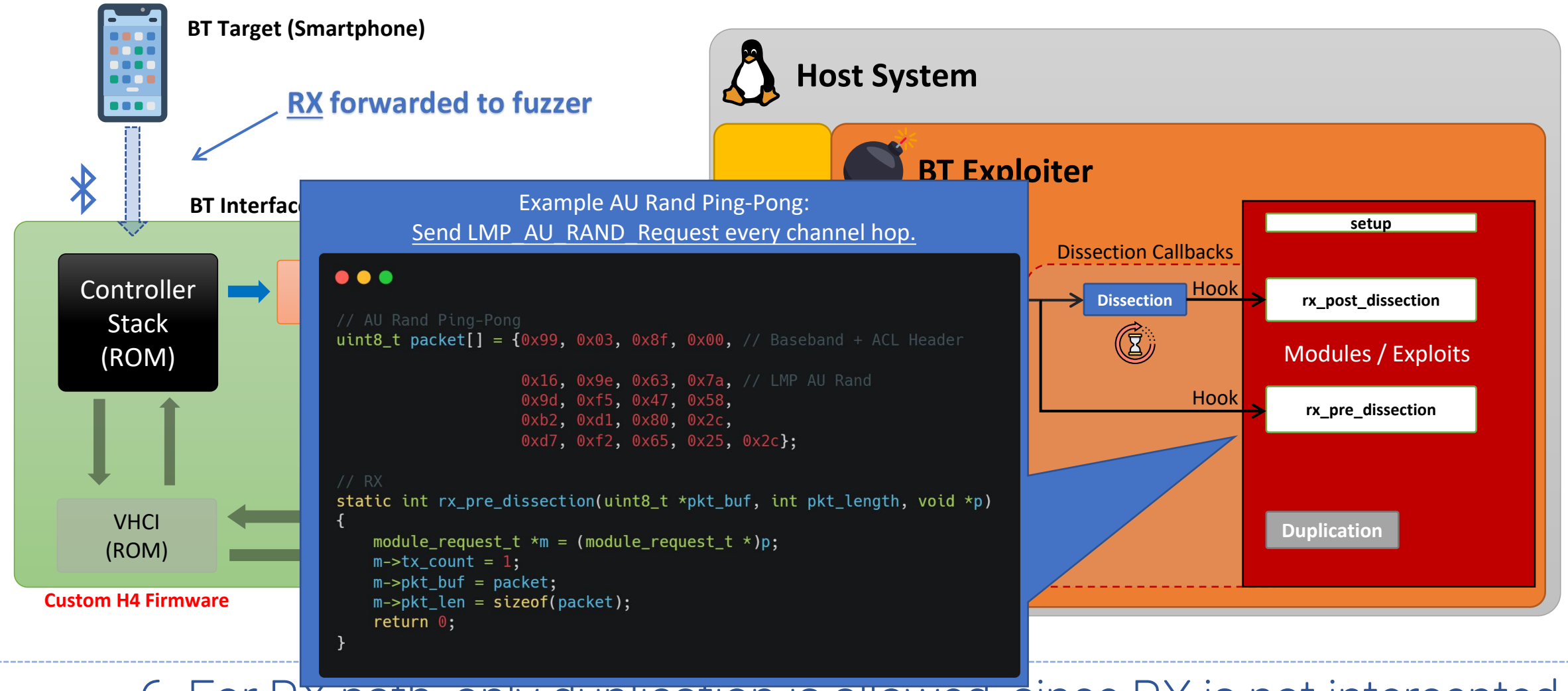
5. Module can combine both TX mutation and injection

ESP32 Exploitation Module Design – How it works?
G) Dissection Events - RX Reception and Injection (Duplication)



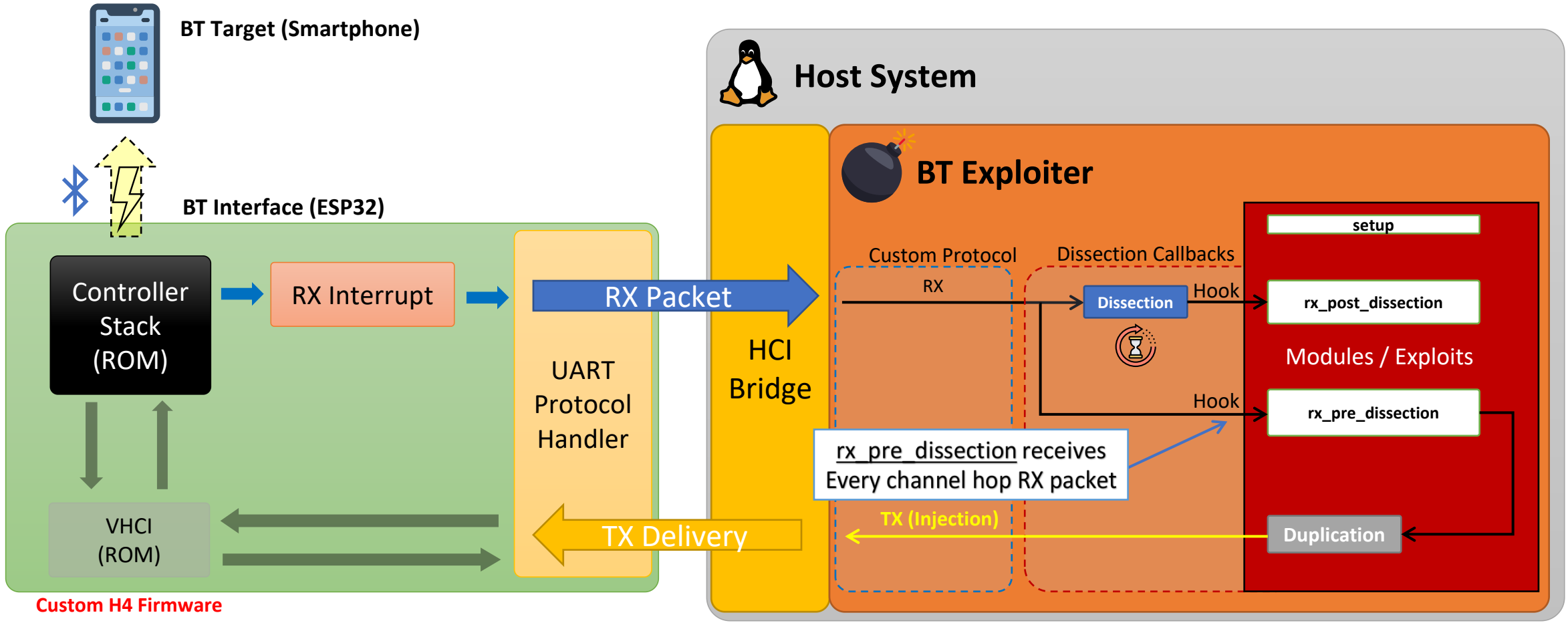
6. For RX path, only duplication is allowed, since RX is not intercepted

ESP32 Exploitation Module Design – How it works?
G) Dissection Events - RX Reception and Injection (Duplication)



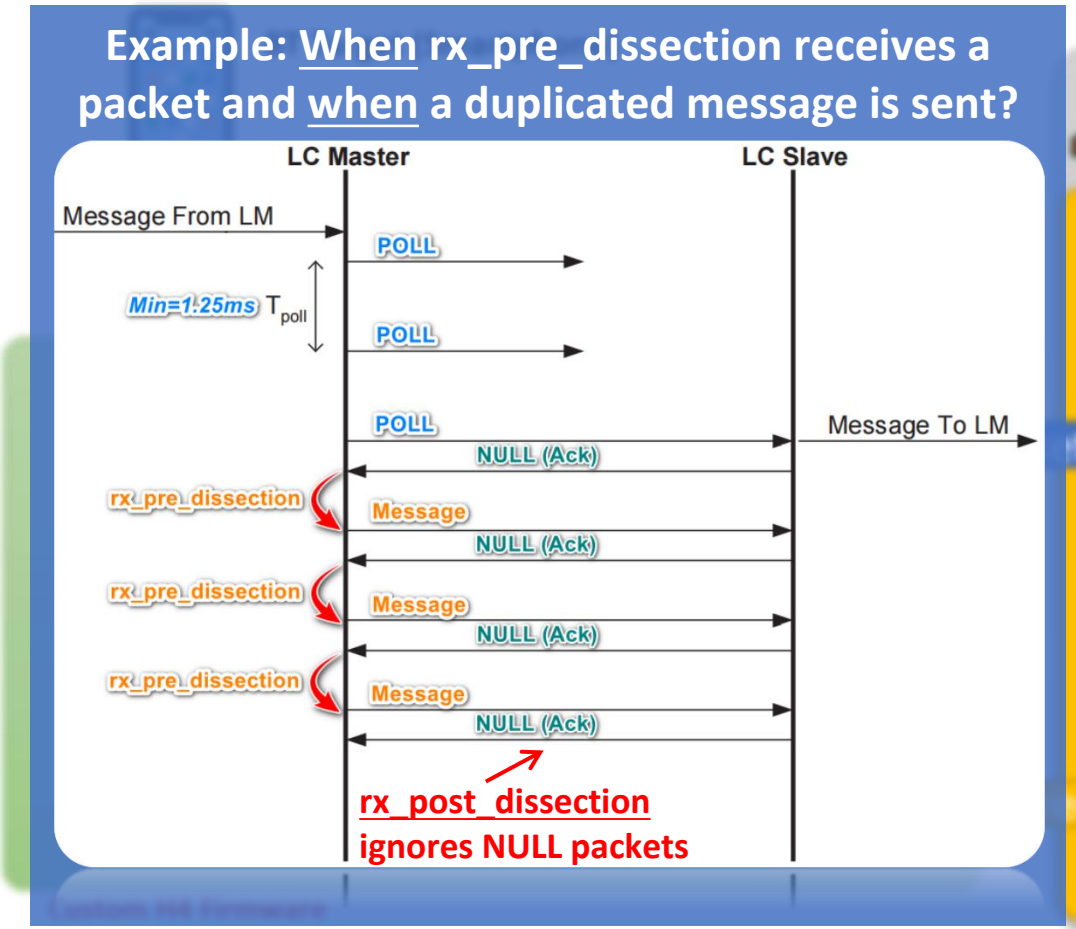
6. For RX path, only duplication is allowed, since RX is not intercepted

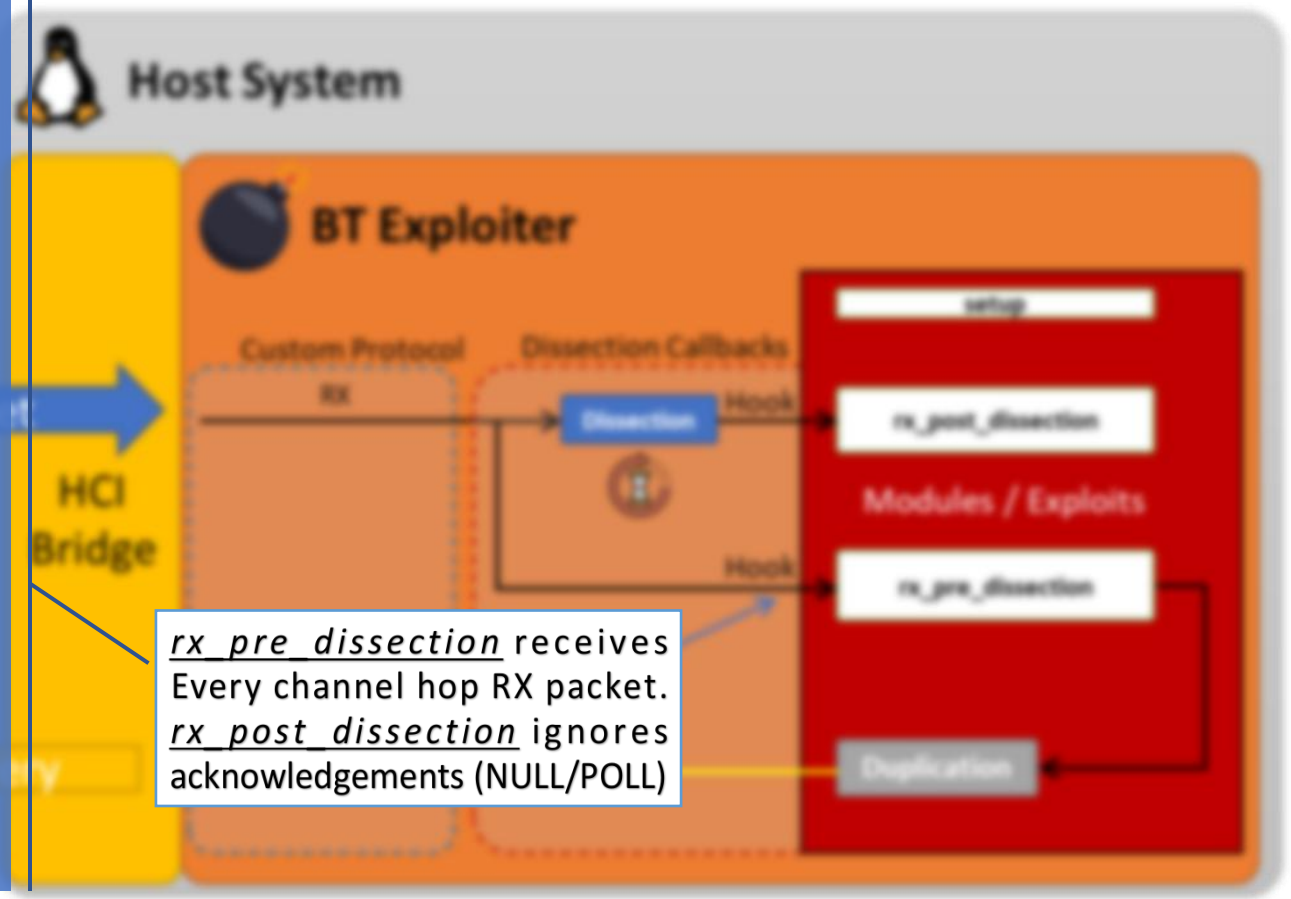
ESP32 Exploitation Module Design – How it works?
G) Dissection Events - RX Reception and Injection (Duplication)



6. For RX path, only duplication is allowed, since RX is not intercepted

ESP32 Exploitation Module Design – How it works?
G) Dissection Events - RX Reception and Injection (Duplication)

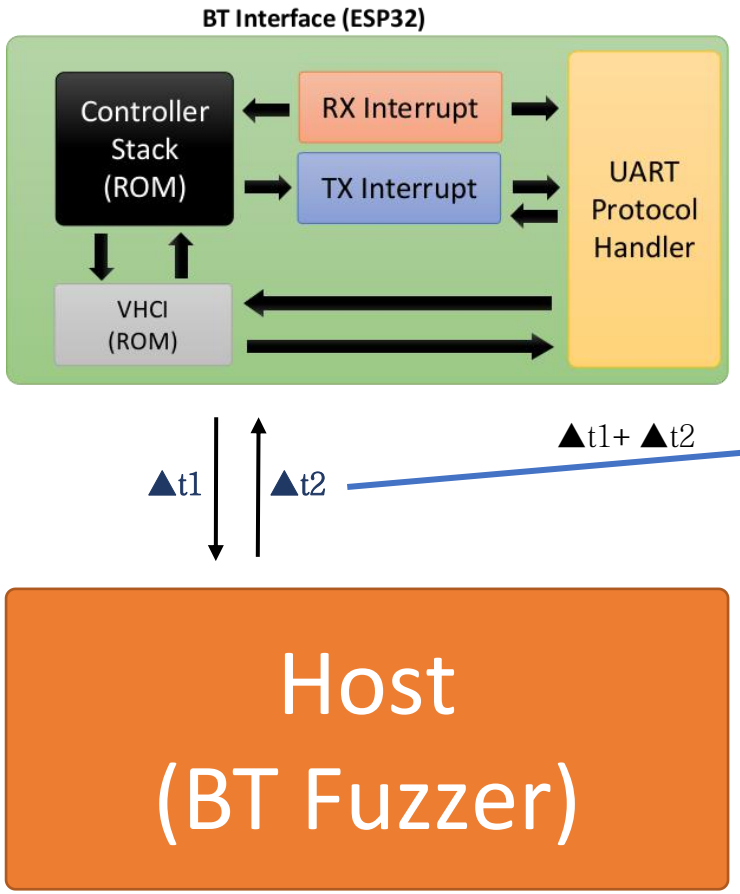


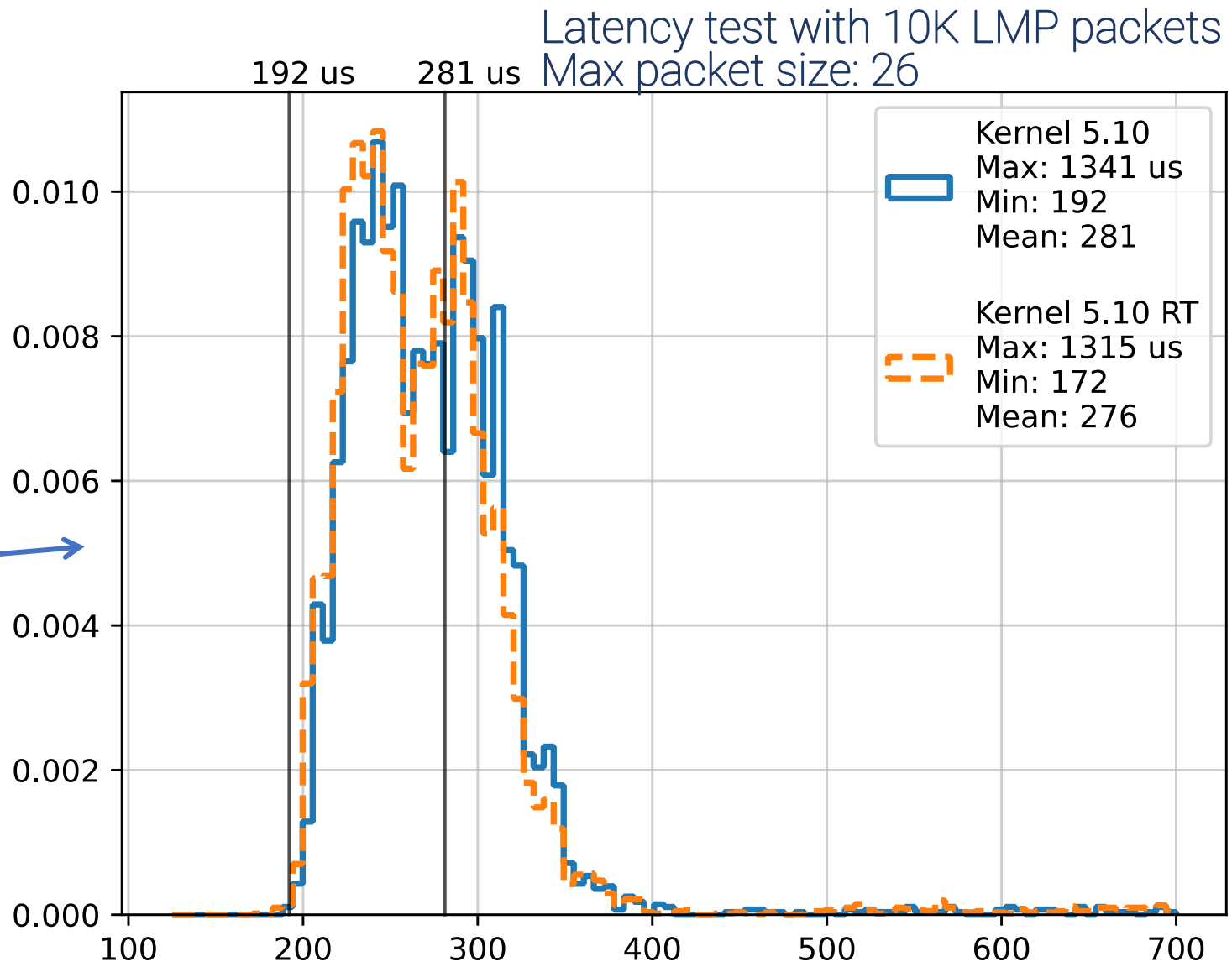


6. For RX path, only duplication is allowed, since RX is not intercepted

Interception Latency Benchmark

Test Setup





Thank you

Questions?