浙江大学网络系统安全与隐私实验室
NETWORK SYSTEM SECURITY & PRIVACY LAB

hexhive

# SyzTrust: State-aware Fuzzing on Trusted OS Designed for IoT Devices

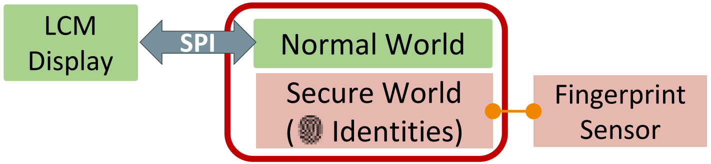
**Qinying Wang**, Boyu Chang, Shouling Ji, Yuan Tian, Xuhong Zhang, Binbin Zhao, Gaoning Pan, Chenyang Lyu, Mathias Payer, Wenhai Wang, Raheem Beyah

浙江大學 ZHEJIANG UNIVERSITY　EPFL　UCLA　Georgia Tech

# Contents

# Motivation

- Trust Execution Environments (TEEs) are essential to **securing important data and operation** in IoT devices.

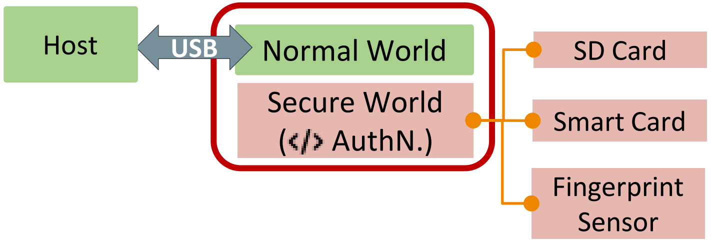



TEE in Smart Lock

Smart Lock

Drone

...



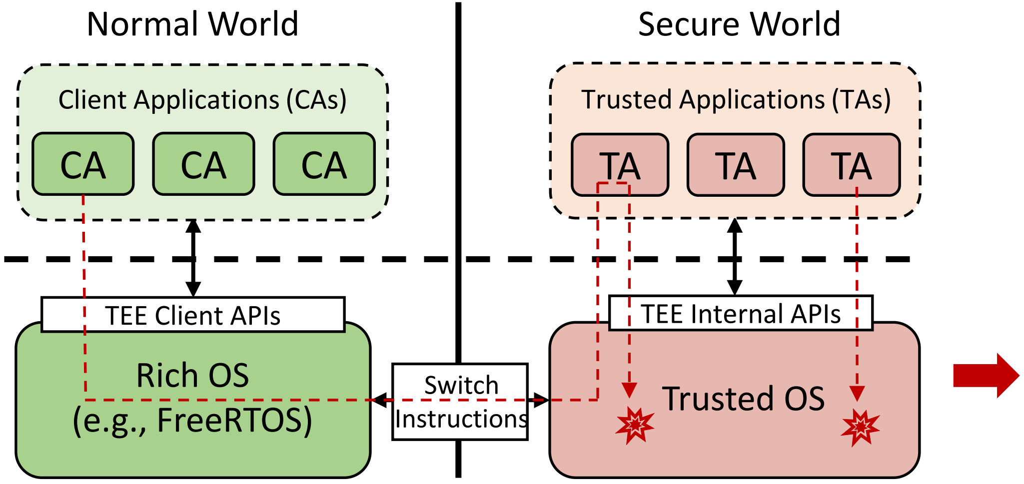

TEE in FIDO Security Key

FIDO Security Key

Smart Locker

# Motivation

- Trusted OS is the **primary** component to enable the TEE to use security techniques.
- The flaws in Trusted OS result in **sensitive data leakage** and **code execution**.




TrustZone-M based TEE

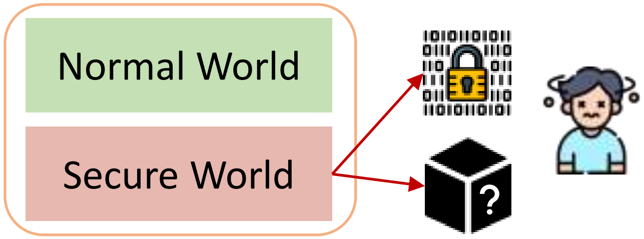
- Gaining control
- Extracting confidential data
- Causing system-wide crashes

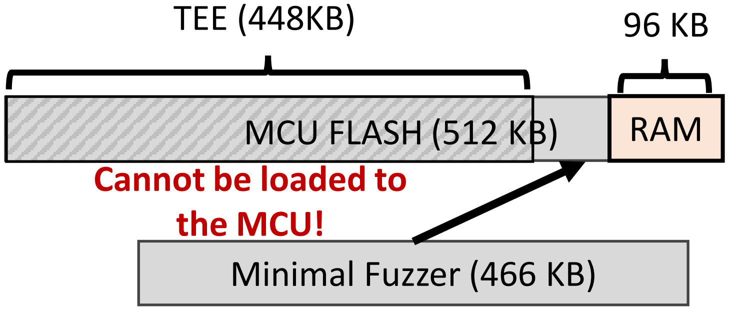
...

Trusted OS or other TAs

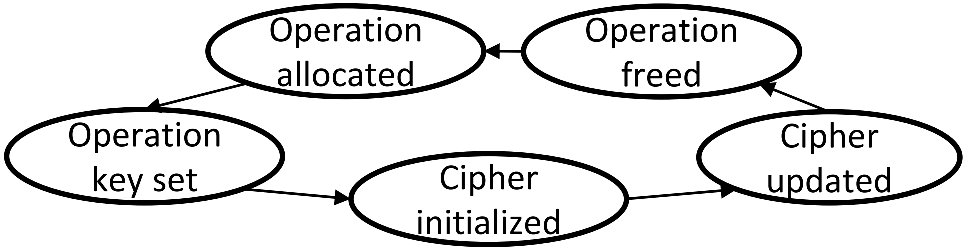
# Challenges of Fuzzing Trusted OSes

- **Challenge 1:** Inability of instrumentation and constraint resource




- **Challenge 2:** Stateful workflow and complex structure




```
struct TEE_OperationHandle{
    uint32_t  algorithm,
    uint32_t operationState,
    TEE_ObjectHandle key…
}
```

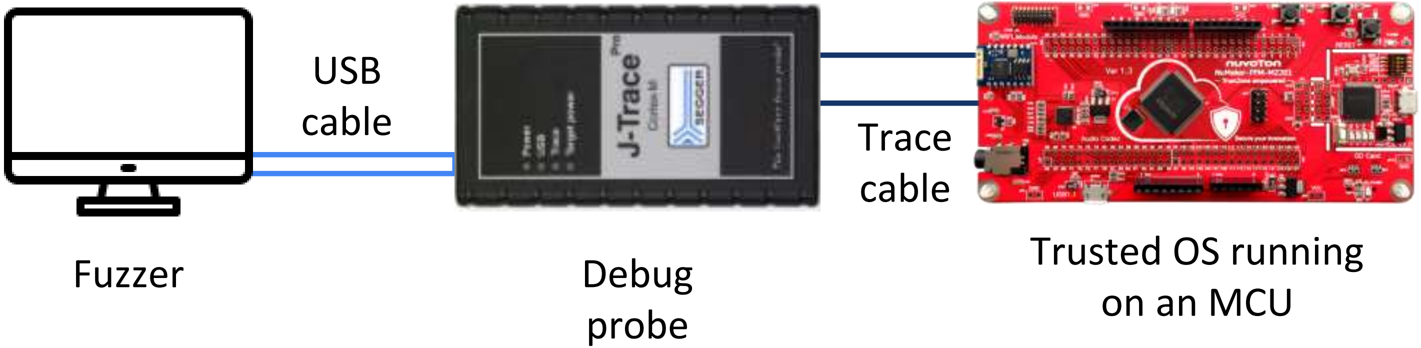
Control the execution context

# Methodology

# Observations and Intuitions

**Challenge 1**

- **Inability of instrumentation: ARM Coresight ETM** provides real-time **instruction tracing**, which can be utilized to calculate code coverage.
- **Constraint resource:** we can **decouple** execution to offload heavy-weight tasks to the PC (e.g., seed scheduling).

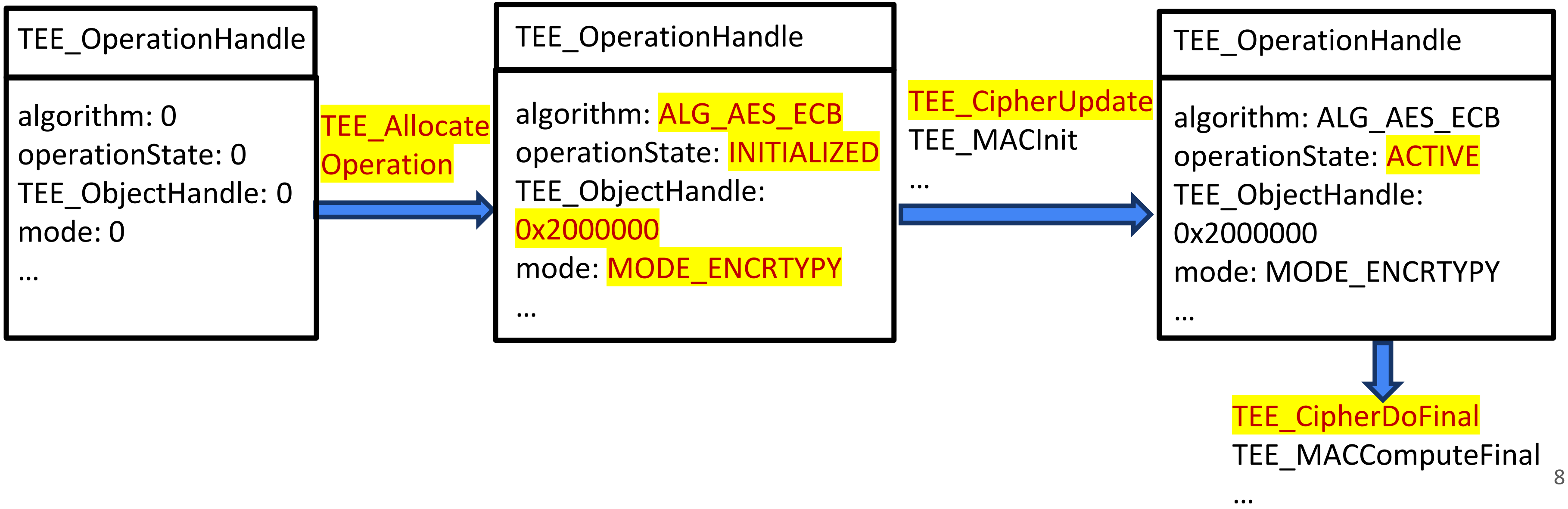



**A hardware-in-the-loop framework**

# Observations and Intuitions

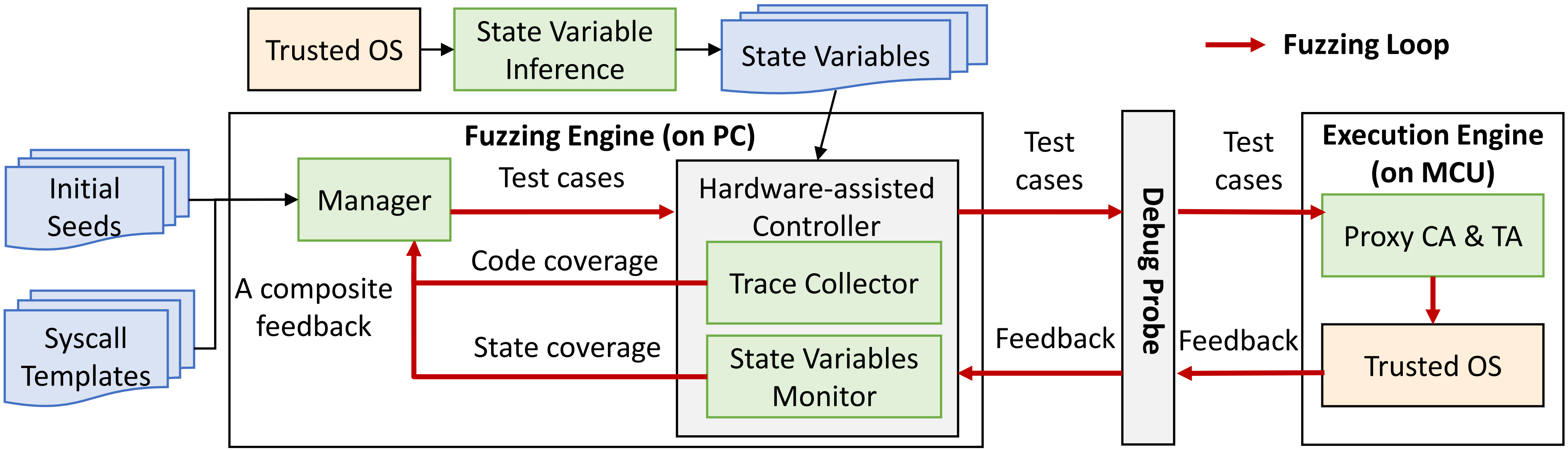
**Challenge 2**

- Several variables in **handle structures** determine the Trusted OS' internal state.
- **State coverage** can be calculated based on the combination values of the variables, which supplement code coverage.



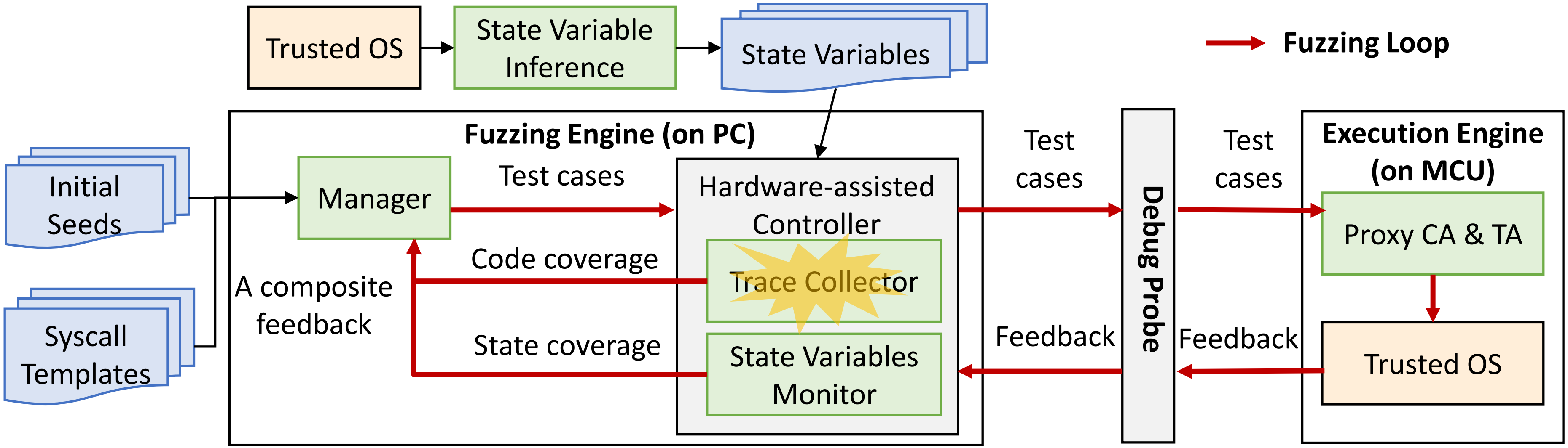
# SyzTrust End-to-End

- The fuzzing engine generates and sends test cases to the MCU via a debug probe.
- The execution engine executes the received test case on the target Trusted OS.



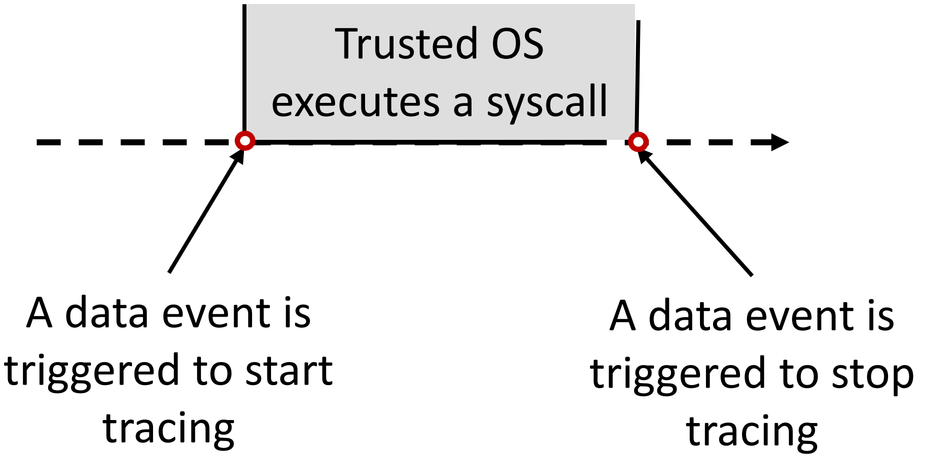
# SyzTrust End-to-End

- The fuzzing engine generates and sends test cases to the MCU via a debug probe.
- The execution engine executes the received test case on the target Trusted OS.



# SyzTrust – Trace Collector

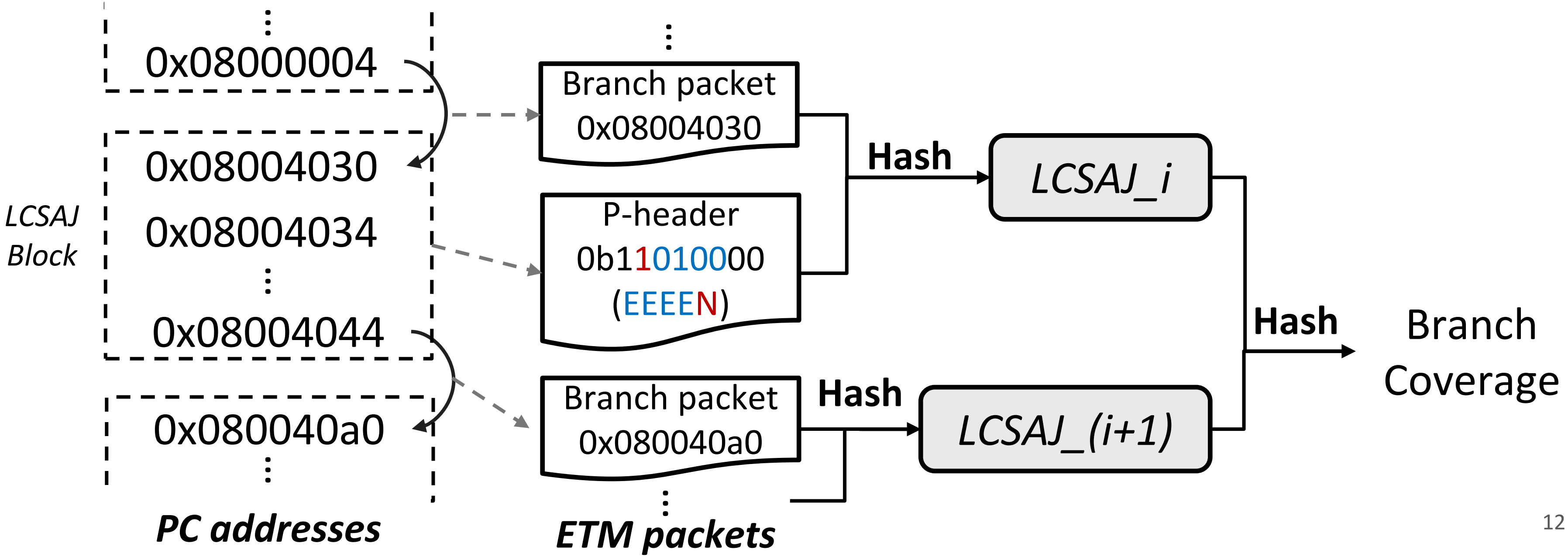
- **Problem:** the ETM component records all instruction traces generated by the CA, rich OS, the TA, and the Trusted OS, which contain noisy trace packets.
- ***Solution:*** *collect instruction traces only when Trusted OS executes a syscall.*




**An event-based filter via the Data Watchpoint and Trace Unit**

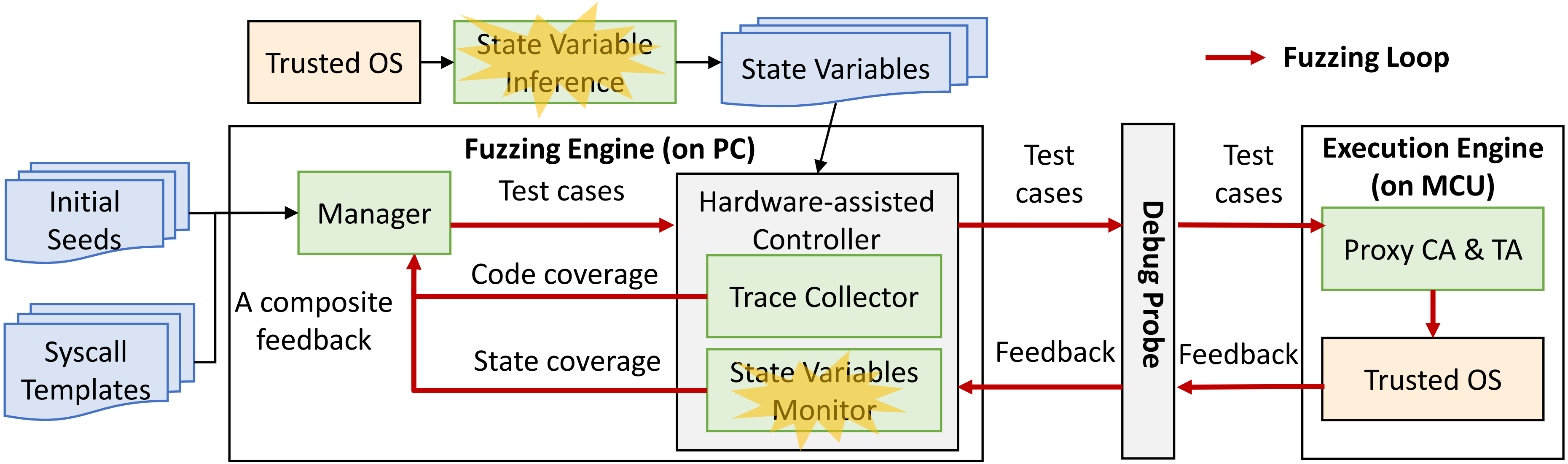
# SyzTrust – Trace Collector

- **Problem:** aligning decoded ETM packets to disassembled instruction sequences is hard and time-consuming.
- ***Solution:*** *directly calculate the coverage via ETM branch and P-header packets .*



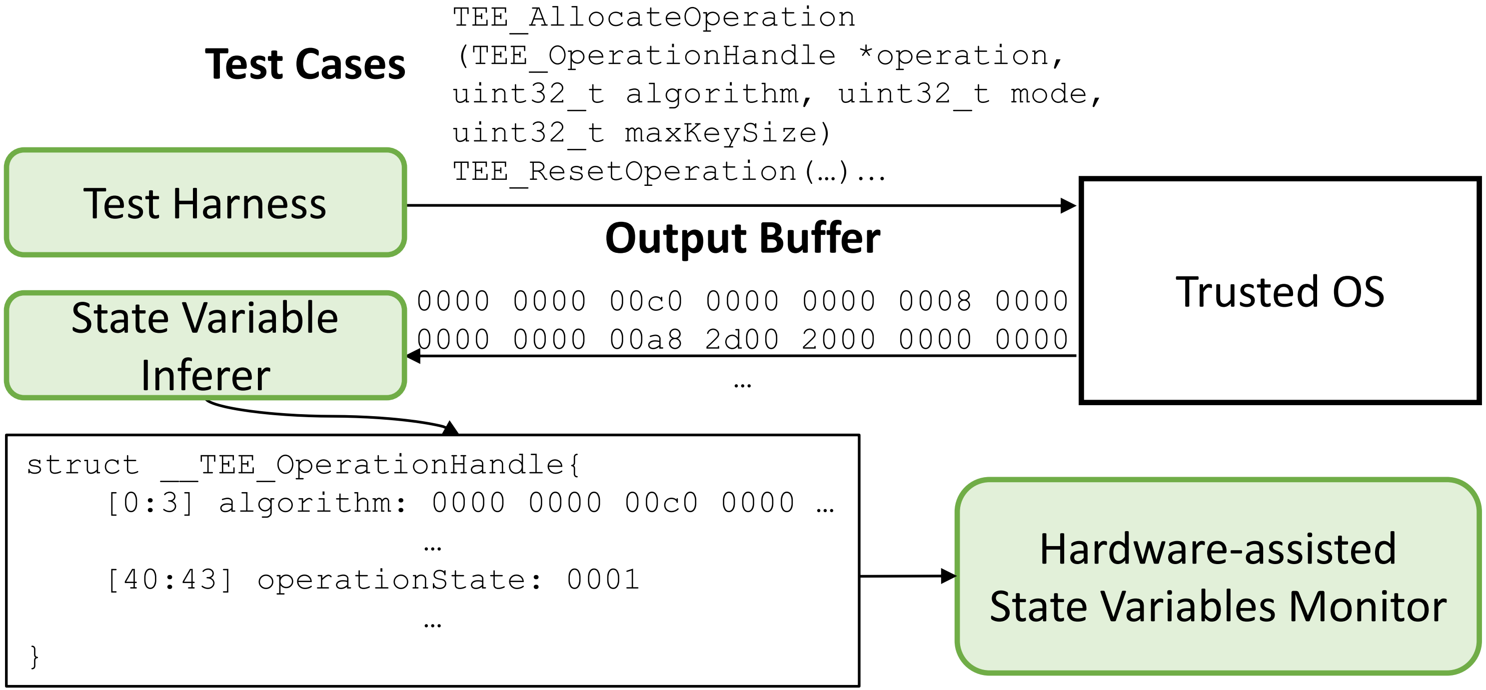
# SyzTrust End-to-End

- The fuzzing engine generates and sends test cases to the MCU via a debug probe.
- The execution engine executes the received test case on the target Trusted OS.



# SyzTrust – State Variable Inference and Monitor

- **Goal:** infer the address ranges of state variables before fuzzing
  track the values of the address ranges during fuzzing



# SyzTrust – State Variable Inference and Monitor

- **Goal:** infer the address ranges of state variables before fuzzing

  track the values of the address ranges during fuzzing

Hardware-assisted State Variables Monitor

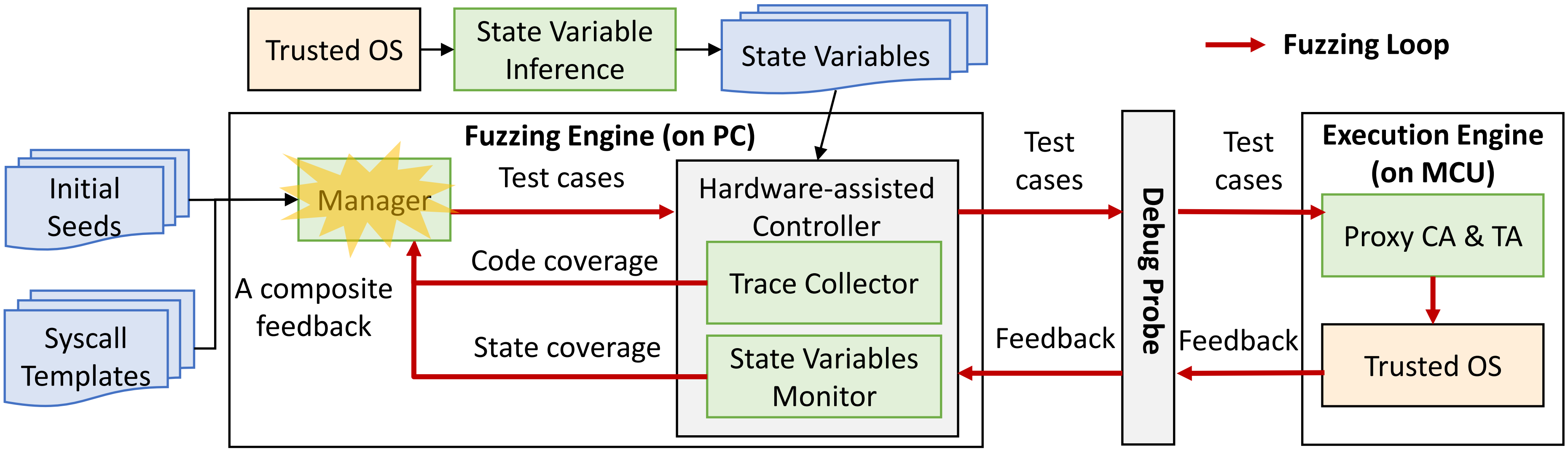
```
[0:3] 00c0 0000 …
 …
[40:43] 0001
```

Real Time Transfer

StateHash = Hash([0:3] … [40:43])

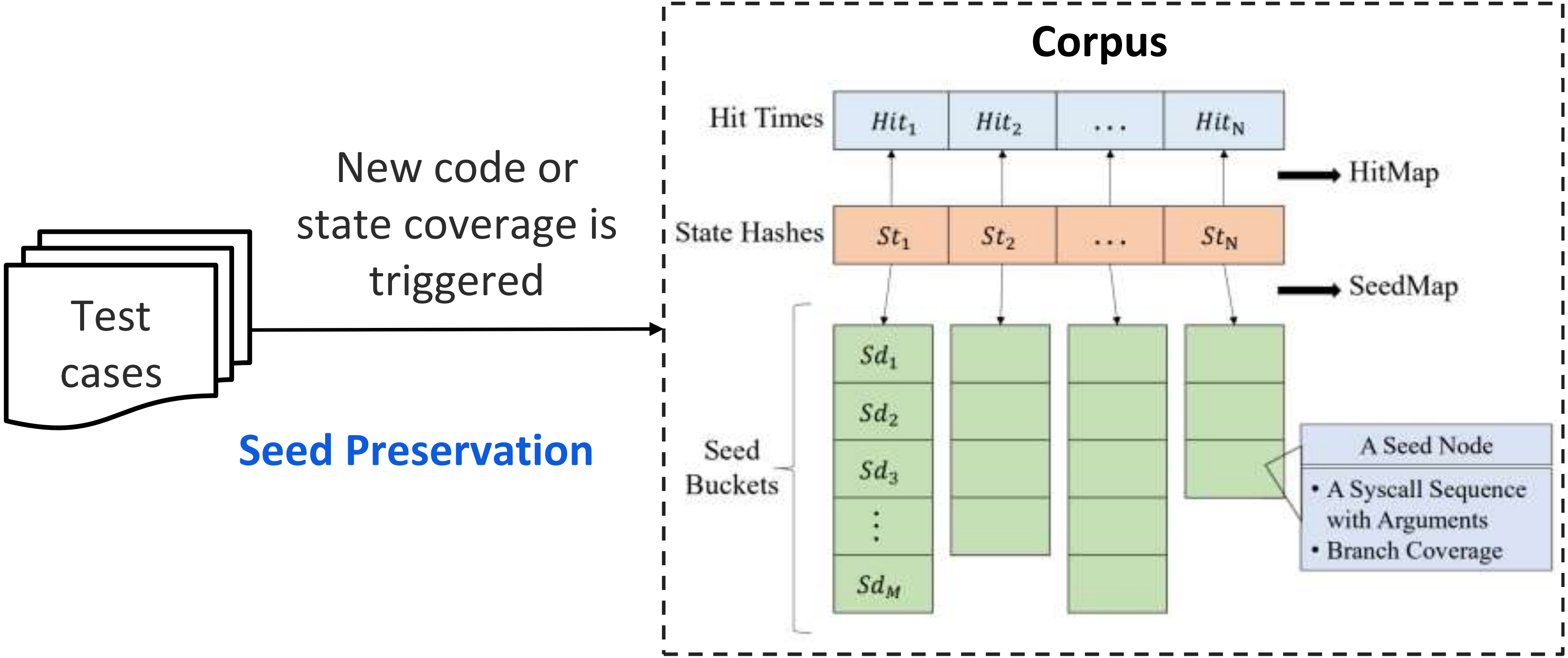
# SyzTrust End-to-End

- The fuzzing engine generates and sends test cases to the MCU via a debug probe.
- The execution engine executes the received test case on the target Trusted OS.



# SyzTrust – Fuzzing Loop and Composite Feedback Mechanism

- **Goal:** state and code coverage guided seed preservation.



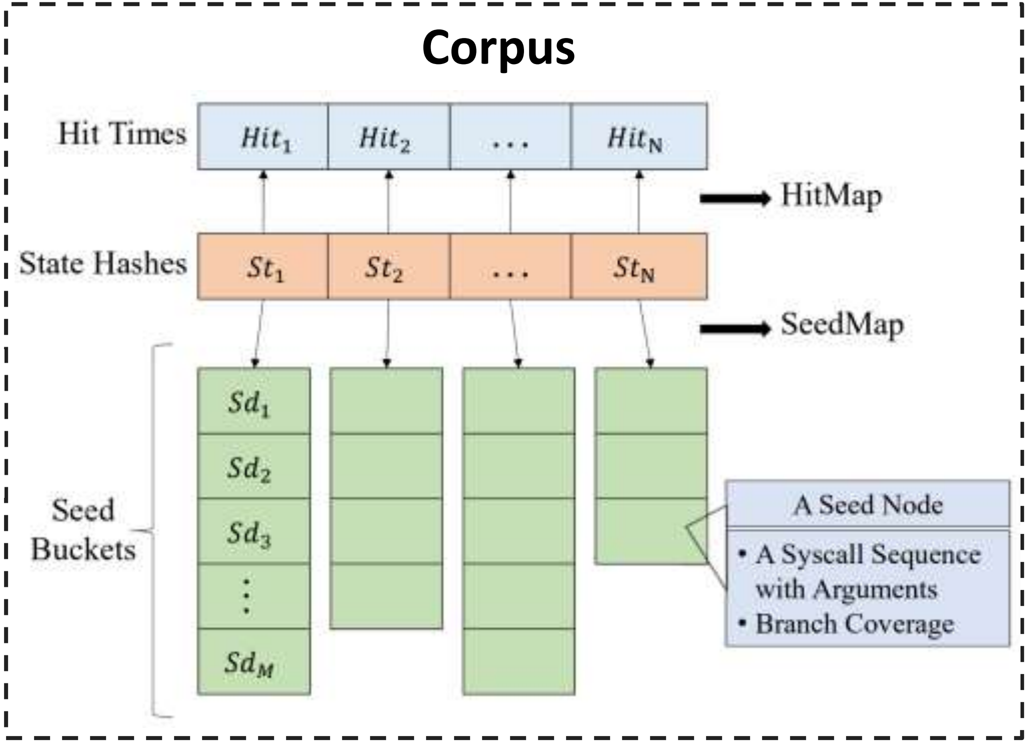
# SyzTrust – Fuzzing Loop and Composite Feedback Mechanism

- **Goal:** state and code coverage guided seed collection.

① choose the state that rarely hit

**Seed Selection**

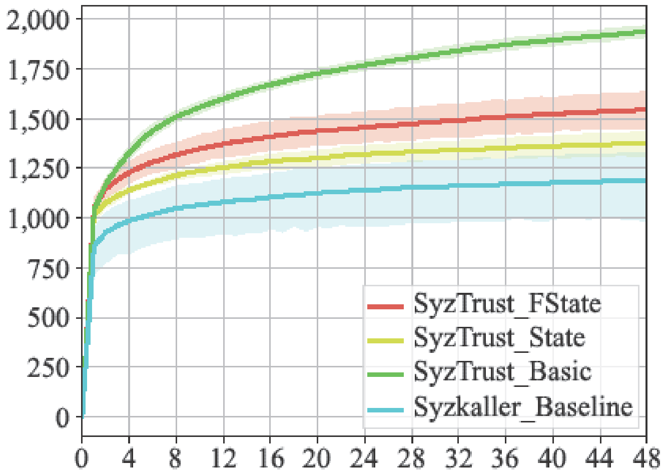
② choose the seed that achieves higher branch coverage



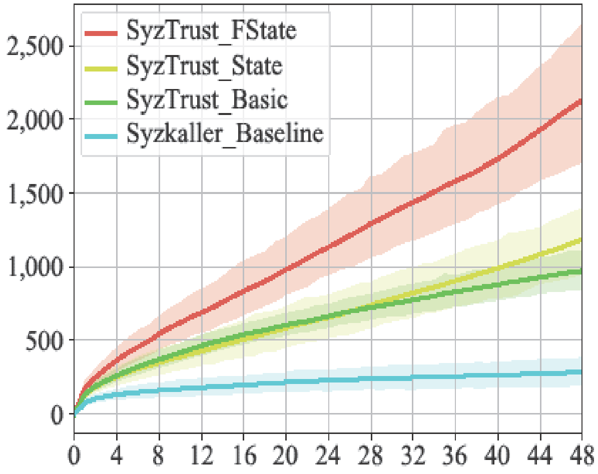
# Evaluation

# Evaluation – Effectiveness of SyzTrust

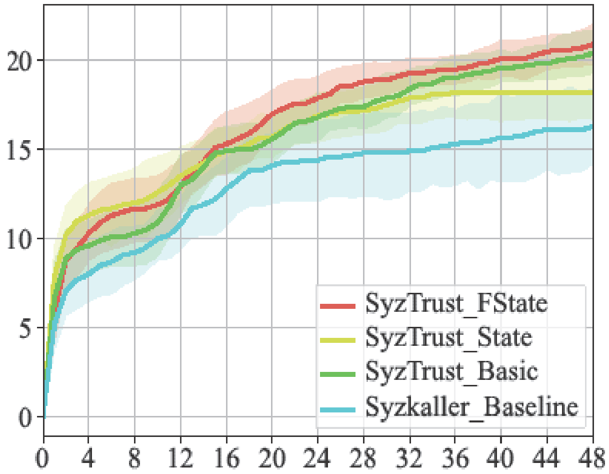
- SyzTrust outperforms Syzkaller in terms of code and state coverage and detected vulnerabilities on mTower from Samsung.




Branch coverage




State coverage

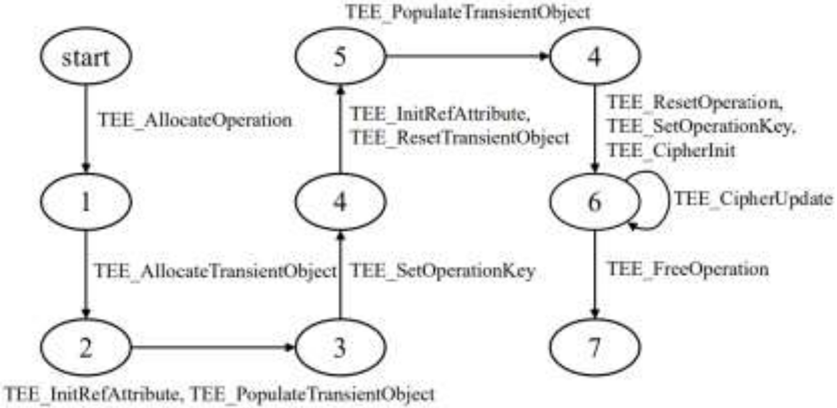



Unique vulnerabilities

# Evaluation – Effectiveness of State Variable Inference

- On average, our active state variable inference method achieves 83.3% precision. From semantics perspective, the inferred state variables are meaningful.

| Target | Handle | Number | FP | Precision |
|---|---|---|---|---|
| mTower | *TEE_ObjectHandle* | 11 | 1 | 87.5% |
| | *TEE_OperationHandle* | 13 | 2 | |
| TinyTEE | *TEE_ObjectHandle* | 13 | 3 | 82.6% |
| | *TEE_OperationHandle* | 10 | 1 | |
| OP-TEE | *TEE_ObjectHandle* | 10 | 1 | 87.0% |
| | *TEE_OperationHandle* | 13 | 2 | |
| Link TEE Air | *context(AES)* | 6 | 2 | 71.4% |
| | *context(Hash)* | 8 | 2 | |



# Evaluation – Real World Vulnerabilities

- SyzTrust identifies 70 vulnerabilities on Trusted OSes from Samsung, Alibaba and Tsinglink Cloud, resulting in 10 CVEs.

mTower — SAMSUNG

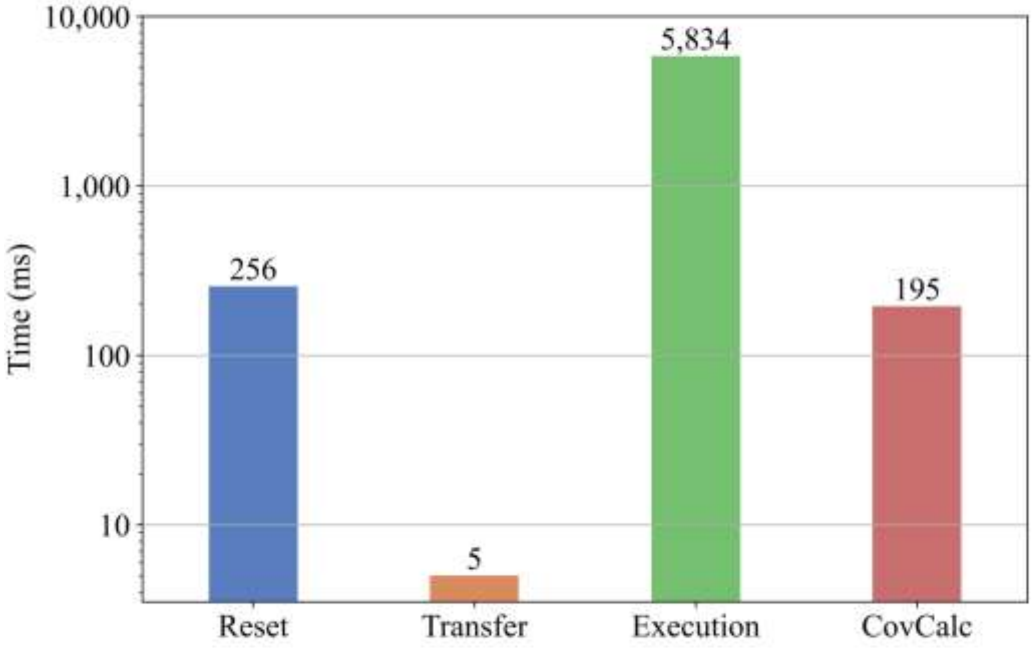
Link TEE Air — Alibaba Cloud

TinyTEE — Tsinglink Cloud

| Target | Unique bugs | Branches | States |
|---|---|---|---|
| mTower | 38 | 2,105 | 3,994 |
| TinyTEE | 13 | 1,072 | 2,908 |
| Link TEE Air | 19 | 10,710 | 182,324 |

# Evaluation – Overhead Breakdown

- The subprocess of executing a test case on the MCU takes the most time, while the orchestration and analysis take only roughly 1% of the overall time.



# Extend SyzTrust to Other Trusted OSes

- **Prerequisites**: (1) a TA can be installed in the Trusted OS; (2) target devices have ETM enabled.

**Extend to Trusted OS implementing standard APIs**

(1) update MCU configurations;
(2) slightly adjustment on our designed TA and CA.

**Extend to Proprietary Trusted OSes**

(1) Update MCU configurations;
(2) augment the syscall templates and the API declarations in our designed TA;
(3) extract the state-related structures (e.g., context).

# Summary

# SyzTrust: State-aware Fuzzing on Trusted OS Designed for IoT Devices

- Inability of instrumentation, constrained resource, and stateful workflow make testing IoT Trusted OS challenging.
- SyzTrust is the first fuzzing framework for IoT Trusted OSes.

  (1) A branch coverage collection utilizing ARM Coresight ETM.

  (2) A composite feedback mechanism including code and state coverage.
- SyzTrust found 70 new bugs in Trusted OSes from Samsung, Alibaba and Tsinglink Cloud.

Paper

Code

浙江大学网络系统安全与隐私实验室
NETWORK SYSTEM SECURITY & PRIVACY LAB

hexhive

# Thanks

# Backup Slides

# IoT Trusted OSes in Real World

| Vendor | Trusted OS | Standards | Support (installing TA) | Some of supported devices |
|---|---|---|---|---|
| Samsung | mTower | GP Standards | ● | NuMaker-PFM-M2351 |
| Alibaba | Link TEE Air | Proprietary | ● | NuMaker-PFM-M2351 |
| TsingLink Cloud | TinyTEE | GP Standards | ● | NuMaker-PFM-M2351/LPC55S69/STM32L562 |
| Beanpod | ISEE-M | GP Standards | ● | LPC55S series/GD32W515/STM32L5 series |
| Trustonic | Kinibi-M | PSA Certified APIs | ● | MicroChip SAML11 |
| ARM | TF-M | PSA Certified APIs | ● | NuMaker-PFM-M2351, STM32L5, ... |

An overview of the major Trusted OS implementations provided by leading IoT vendors

# IoT Trusted OSes in Real World

| Manufacturer | Device | Privilige Secure Debug (including ETM) | Debug Authentication Managerment |
|---|---|---|---|
| Nuvoton | NuMaker-PFM-M2351 | Enable in default | ICP programming tool |
| NXP Semiconductors | LPC55S69 | Enable in default | Debug credential certificate |
| STMicroelectronics | STM32L562 | Enable in default | STM32CubeProgrammer |
| GigaDevice | GD32W515 | Enable in default | Efuse |
| MicroChip | SAML11 | Enable in default | Extern debugger |

ETM feature on IoT devices