TEEzz: Fuzzing Trusted Applications on COTS Android Devices

Marcel Busch

Aravind Machiry, Chad Spensky,
Giovanni Vigna, Christopher Kruegel, Mathias Payer

TEEzer


FUZZ
ALL THE TEES
imgflip.com


TrustZone®
System Security by ARM

$ whoami

- Marcel Busch
- Ph.D. graduate (FAU Erlangen-Nuremberg)
- By now PostDoc with HexHive @ EPFL
- Works on
  - Embedded systems security
  - Android TEEs
  - Fuzzing

HexHive

EPFL

@0ddc0de

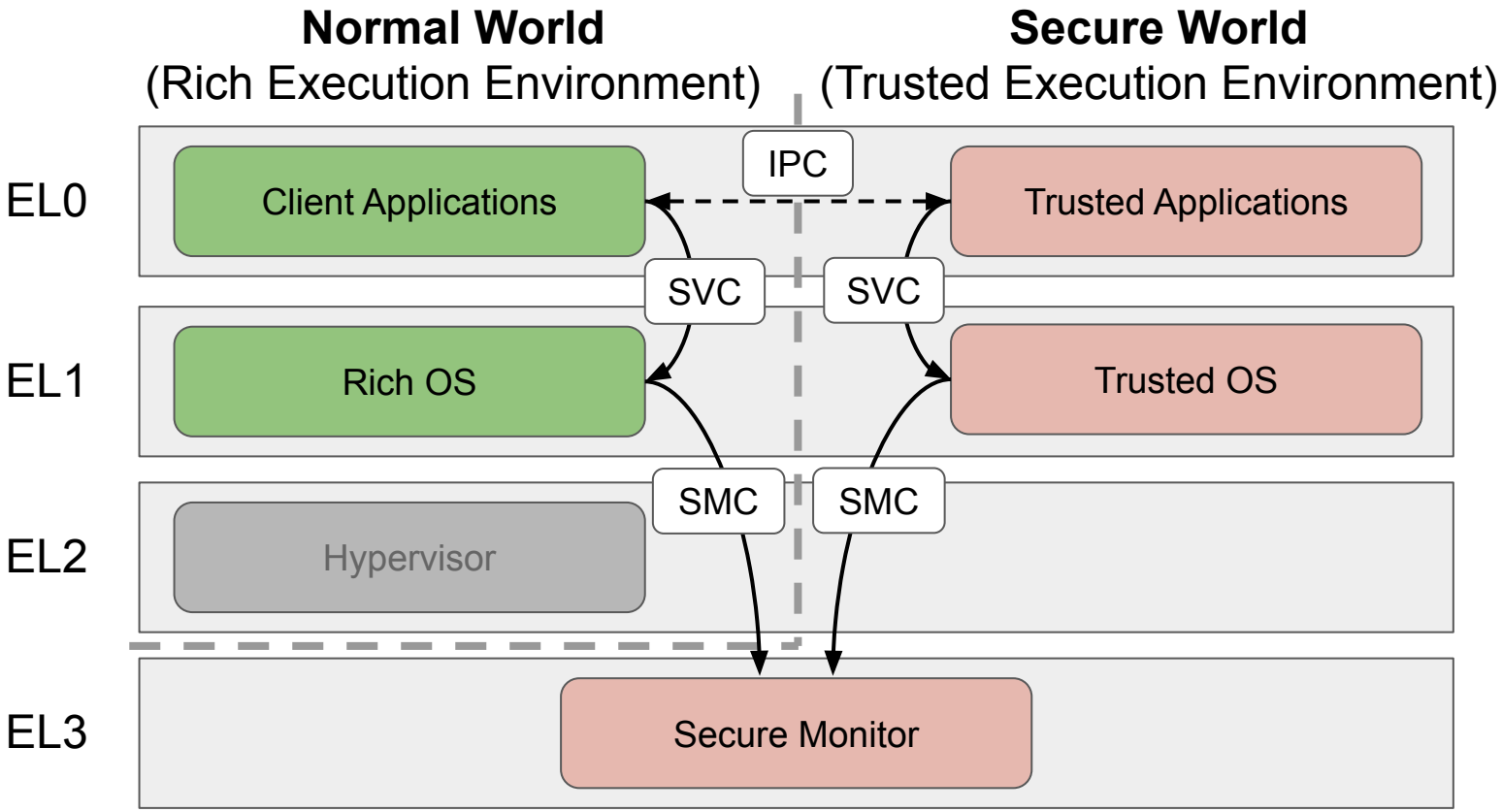
@0ddc0de@infosec.exchange

Outline

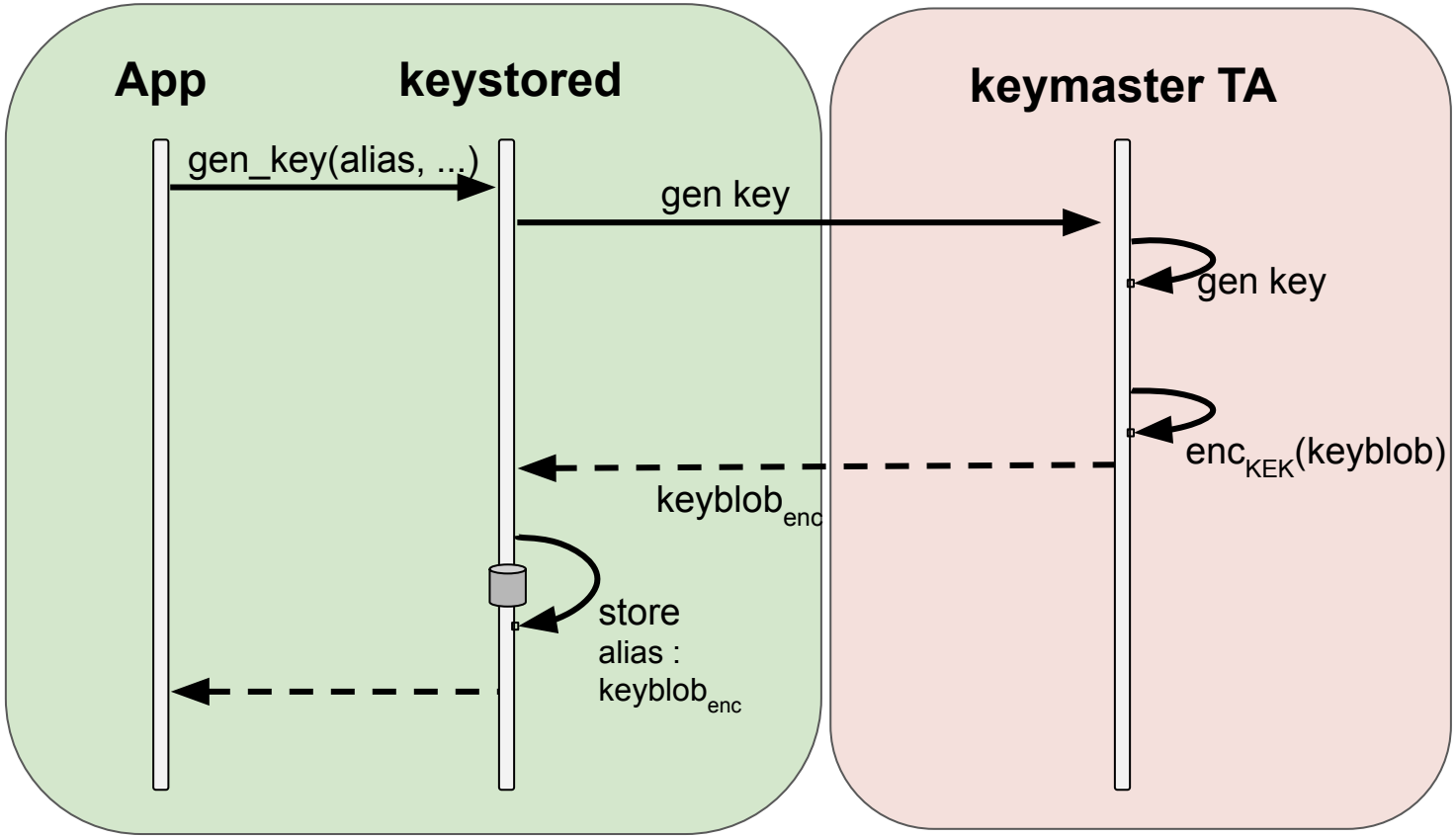
- **ARMing up** on TEEs
- **Motivation** for fuzzing Trusted Applications
- **Challenges** of fuzzing TEEs on COTS Android devices
- **Previous Work**
- **Design** of TEEzz
- **Fuzzing** with TEEzz
- **Conclusions**

ARMing Up

ARM TrustZone Privilege Levels



Example: Keystore / Keymaster System

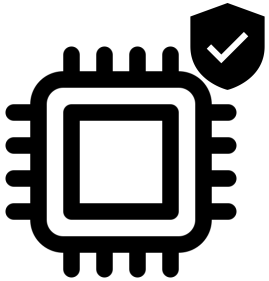


Motivation

Why Fuzz Trusted Applications?

- Large attack surface
- Growing and fragmented ecosystem
- Memory-unsafe languages

How many TAs?



Samsung Galaxy S22 (intl.)

```
$ ls /SM-S901B/EUY/S901BXXS4CWD3/tas
00000000-0000-0000-0000-000000010081  00000000-0000-0000-0000-487641557457  00000000-0000-0000-0000-564c544b5052
00000000-0000-0000-0000-000000020081  00000000-0000-0000-0000-4b45594d5354  00000000-0000-0000-0000-64756c444152
00000000-0000-0000-0000-000000534b4d  00000000-0000-0000-0000-4d5053545549  00000000-0000-0000-0000-656e676d6f64
00000000-0000-0000-0000-000048444350  00000000-0000-0000-0000-4d7073617574  00000000-0000-0000-0000-657365636f6d
00000000-0000-0000-0000-0000534b504d  00000000-0000-0000-0000-505256544545  00000000-0000-0000-0000-6b6e78677564
00000000-0000-0000-0000-0050524f4341  00000000-0000-0000-0000-534258505859  00000000-0000-0000-0000-6d706f667376
00000000-0000-0000-0000-0053545354ab  00000000-0000-0000-0000-5345435f4652  00000000-0000-0000-0000-6d70776c646c
00000000-0000-0000-0000-00575644524d  00000000-0000-0000-0000-53454d655345  00000000-0000-0000-0000-6d73745f5441
00000000-0000-0000-0000-42494f535542  00000000-0000-0000-0000-54412d48444d  ffffffff-0000-0000-0000-000000000030
00000000-0000-0000-0000-46494e474502  00000000-0000-0000-0000-54496473706c
00000000-0000-0000-0000-474154454b45  00000000-0000-0000-0000-544974684c6c
$ ls /SM-S901B/EUY/S901BXXS4CWD3/tas | wc -l
33
```

TAs are the largest attack surface of the TEE

Vulnerabilities?

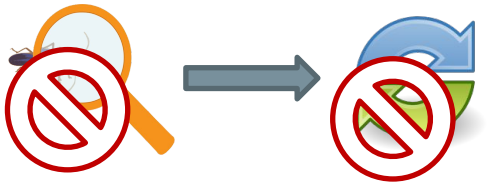
```
$ scrape_samsung_security_bulletin.py | tee ta_cves.txt
...
SVE-2022-1364(CVE-2023-21420): Use of Externally-Controlled Format String vulnerabilities in STST TA, Severity:     High
SVE-2022-2118(CVE-2023-21424): Improper Authorization vulnerability in SemChameleonHelper, Severity: Moderate
SVE-2022-2195(CVE-2023-21435): Exposure of Sensitive Information vulnerability in Fingerprint TA, Severity: Moderate
SVE-2022-2315(CVE-2023-21477): Access of Memory Location After End of Buffer vulnerability in TIGERF trustlet, Severity:
Critical
SVE-2022-2316(CVE-2023-21453): Improper input validation vulnerability in SoftSim TA, Severity:    High
SVE-2022-2318(CVE-2023-21478): Improper input validation vulnerability in TIGERF trustlet, Severity:    High
SVE-2022-2959(CVE-2023-21471): Improper access control vulnerability in SemClipboard, Severity: Moderate
SVE-2023-0215(CVE-2023-21499, CVE-2023-21498, CVE-2023-21497): Arbitrary code execution in mPOS TUI trustlet, Severity:
Critical
$ wc -l ta_cves.txt
97
```

TAs are vulnerable to classical memory corruption bugs

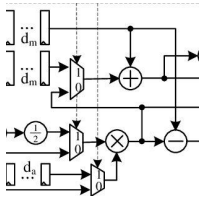
Challenges

Challenges of Fuzzing Trusted Applications

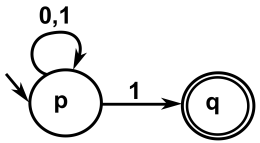
1. Limited introspection

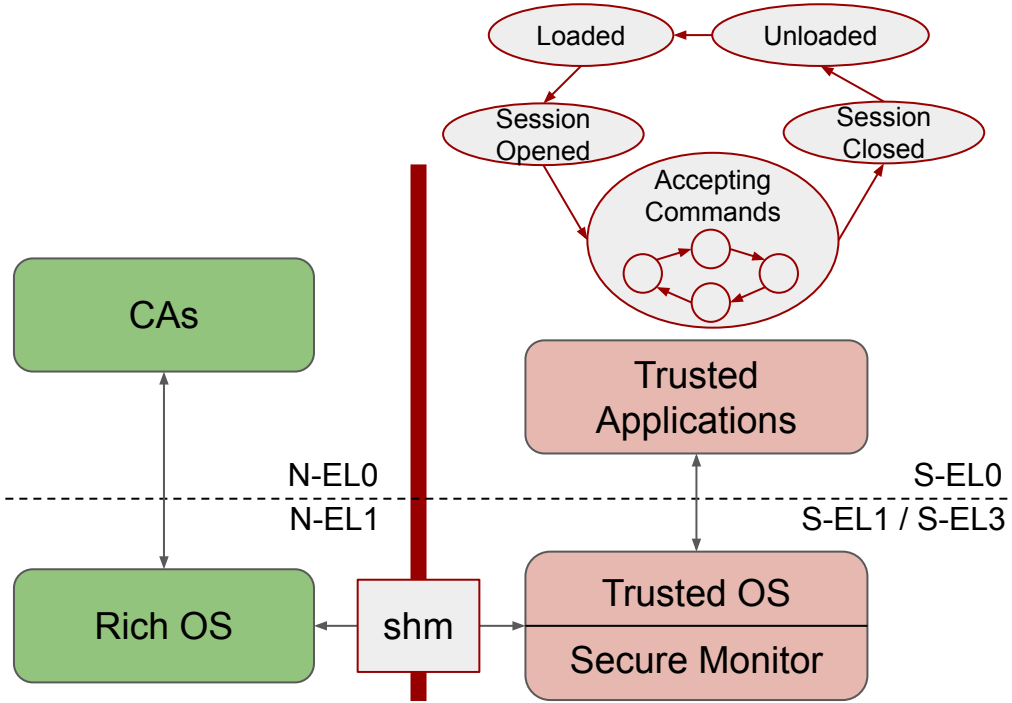


2. Complex input



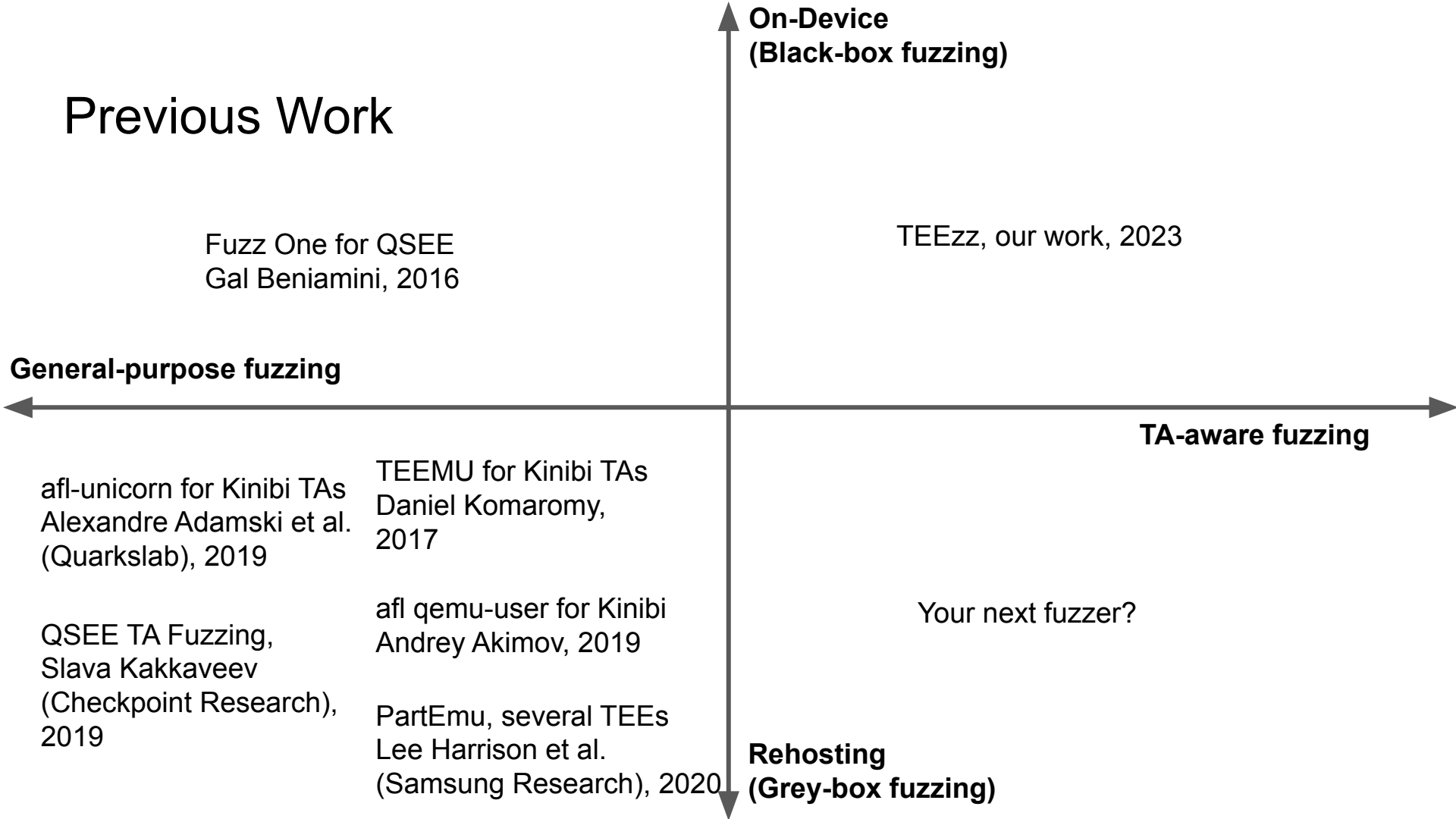
3. Statefulness





Previous Work

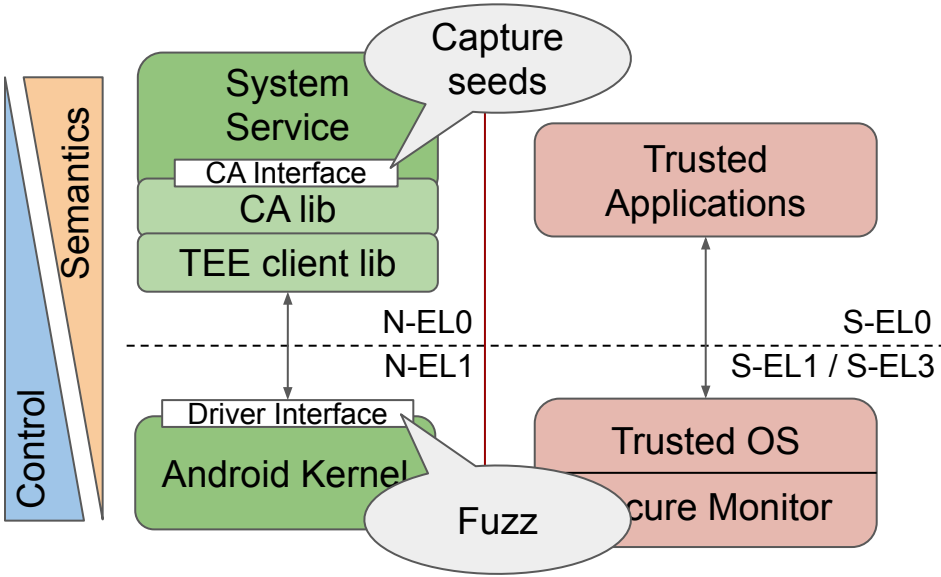
Previous Work



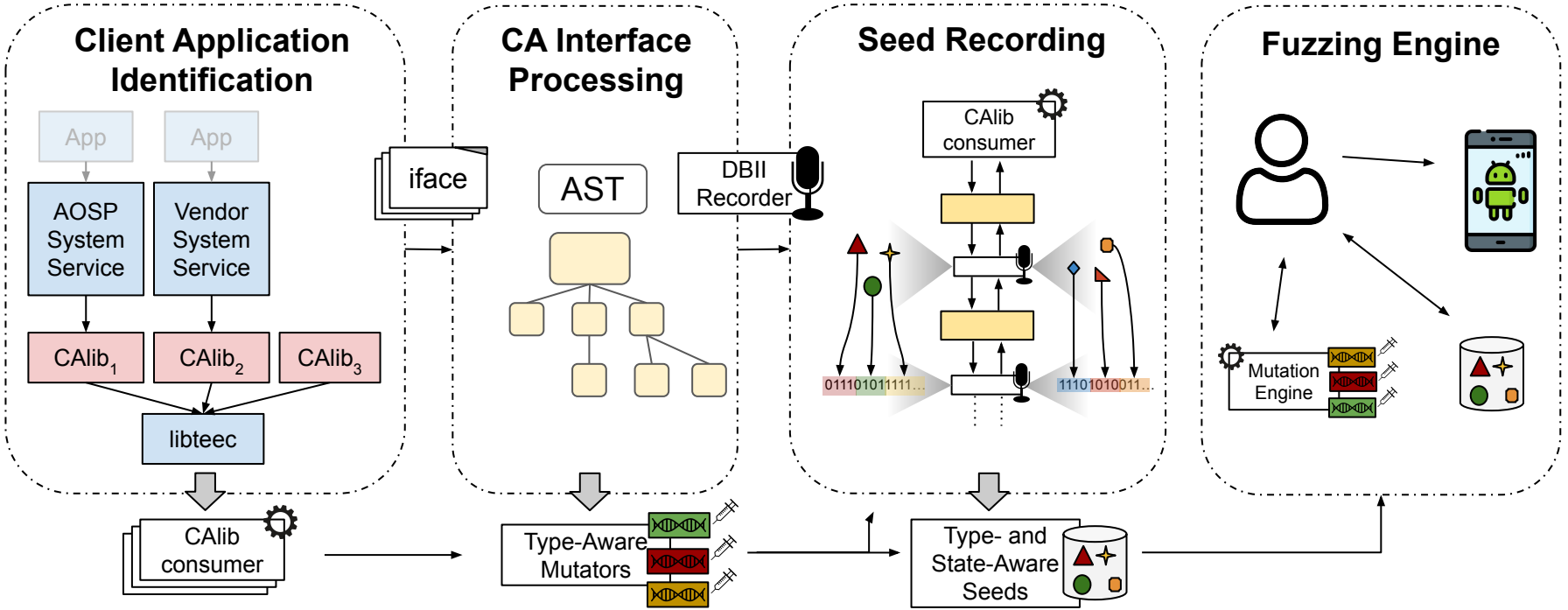
Design of TEEzz

Enter TEEzz

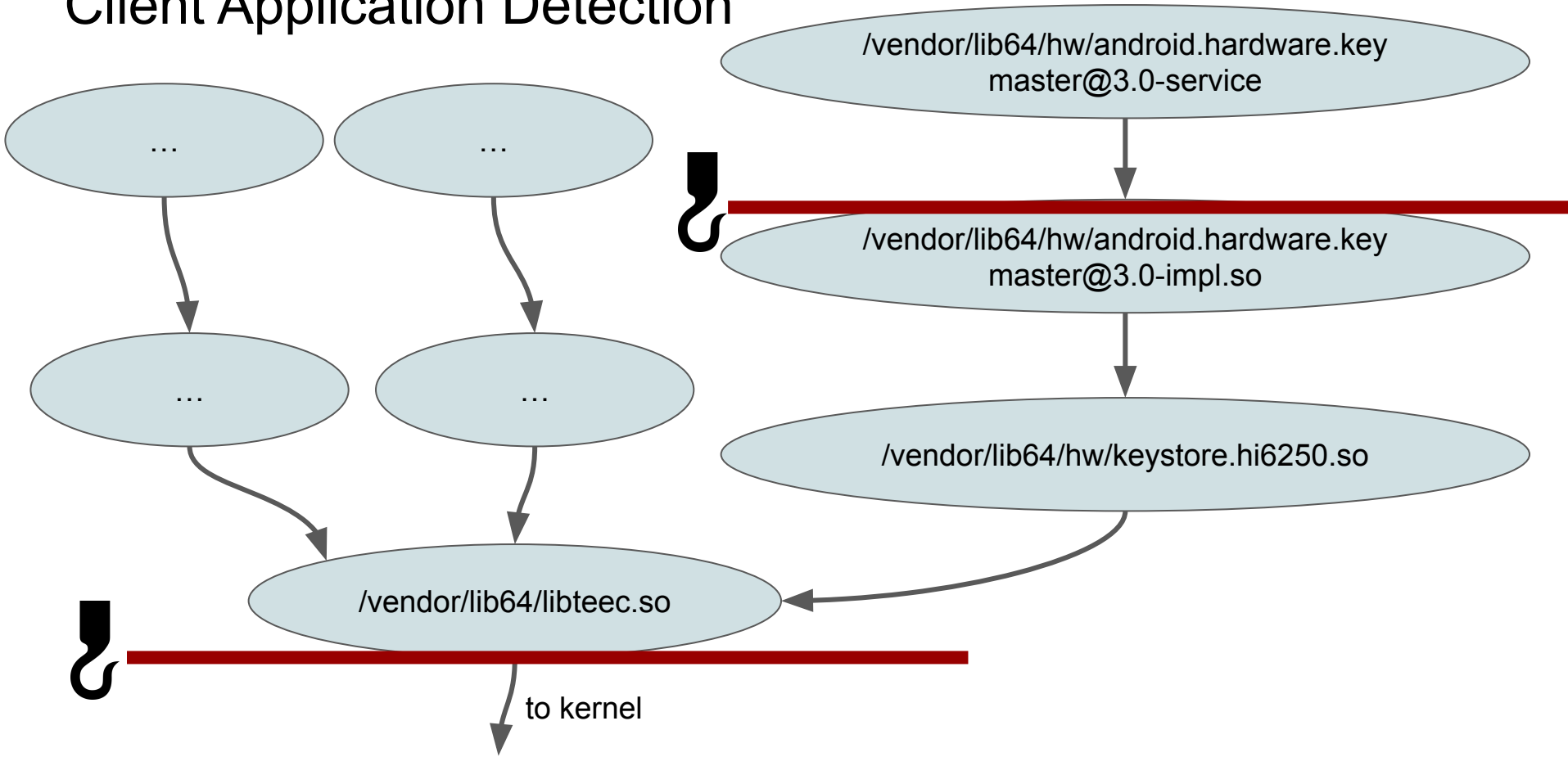
- Challenges: complex input and statefulness
- Observations / intuitions
  - Semantics have to be in the attacker-controlled NW to make use of services provided by SW
  - Semantics decreases towards lower levels of abstraction
  - Control over input increases towards lower levels of abstraction



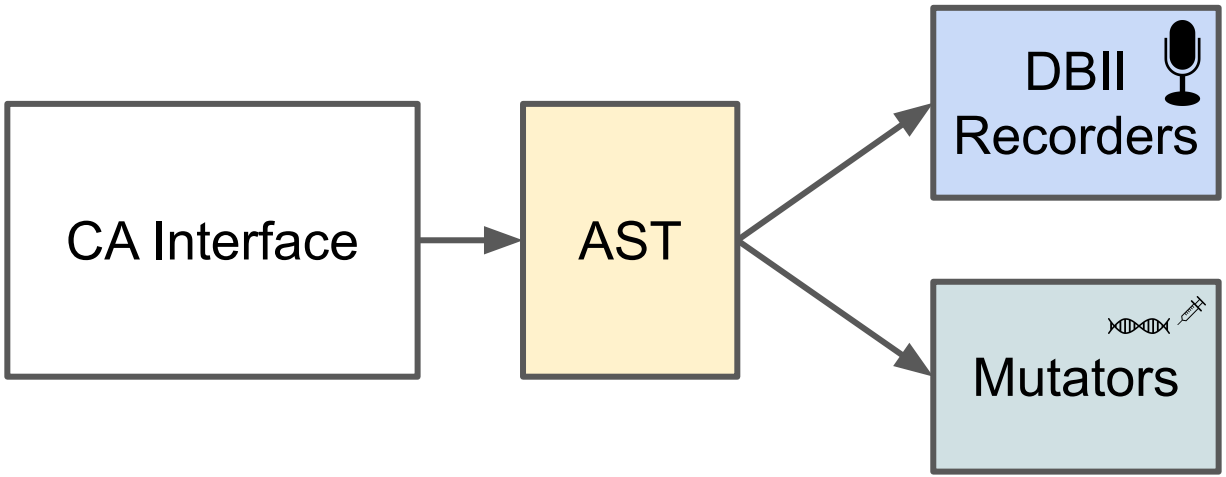
TEEzz – End-to-End



Client Application Detection

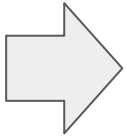


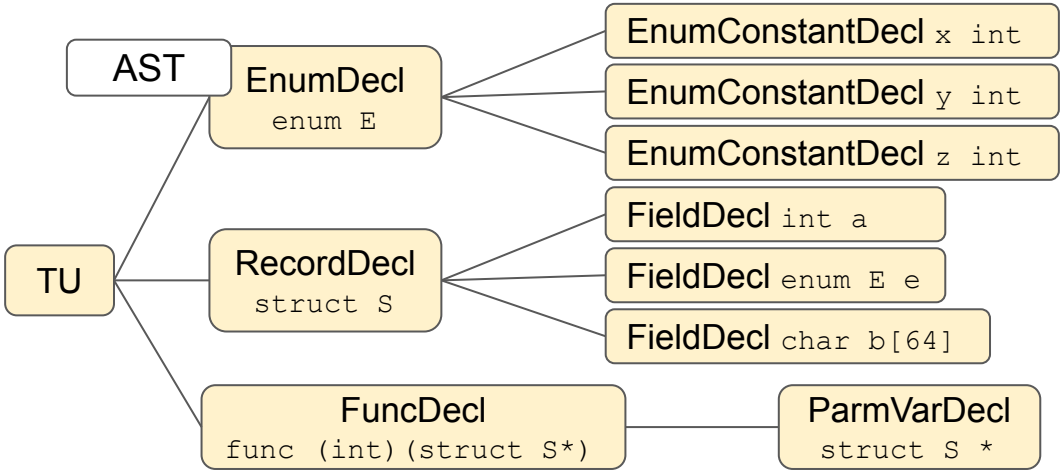
CA Interface Processing & Seed Recording



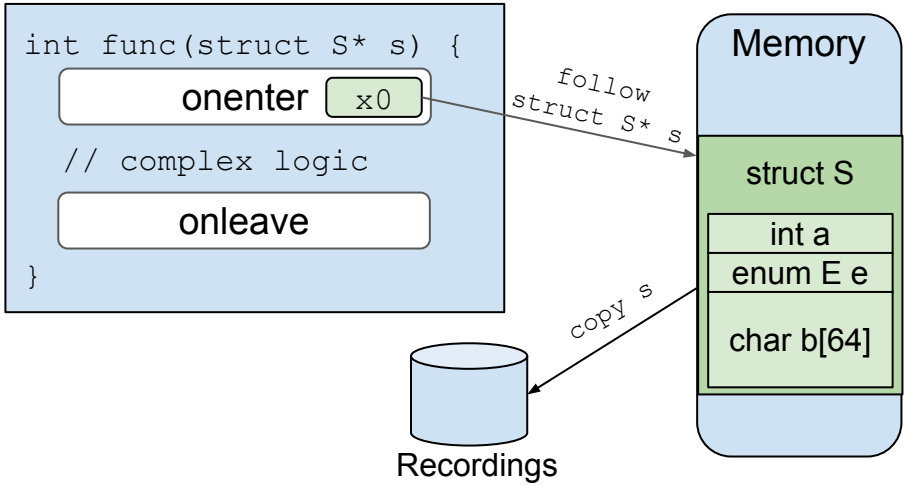
Interface Definition

```
enum E { x, y, z };

struct S {
    int a;
    enum E e;
    char b[64];
}

int func(struct S* s);
```

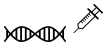




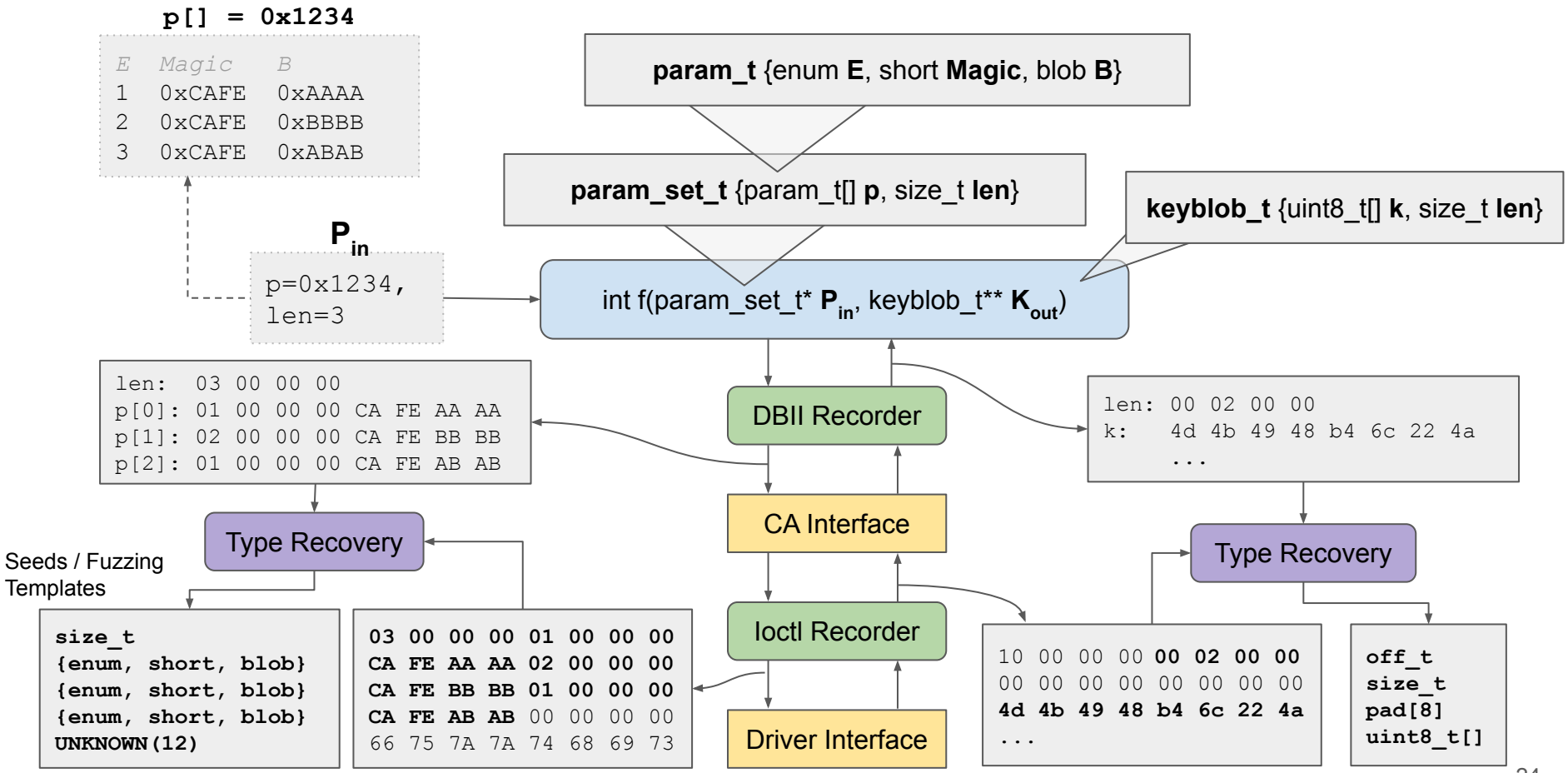
DBII Recorder



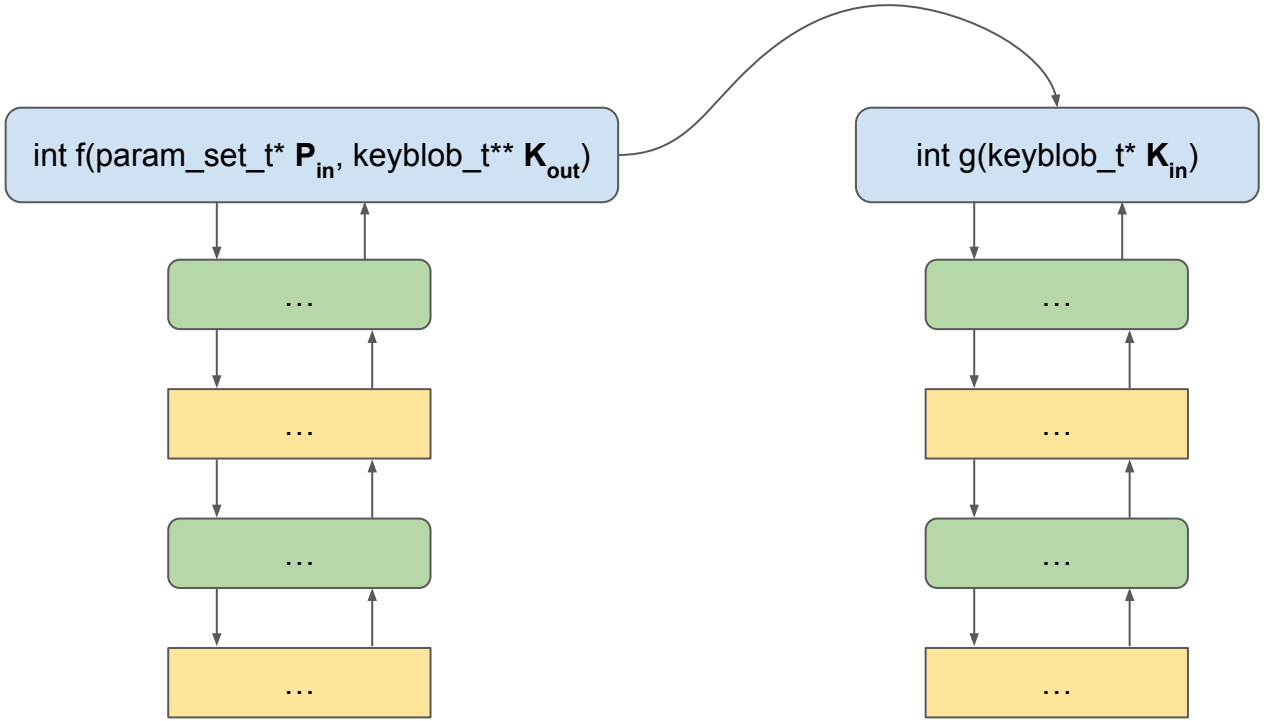
Type-Aware Mutator



```
def mutate_enum(enum_decl):

    enum_decl
    enum E
        x = 0
        y = 1
        z = 2

    return random.choice(enum_decl)
```


p[] = 0x1234
E Magic B
1 0xCAFE 0xAAAA
2 0xCAFE 0xBBBB
3 0xCAFE 0xABAB
param_t {enum E, short Magic, blob B}
param_set_t {param_t[] p, size_t len}
keyblob_t {uint8_t[] k, size_t len}
P in
p=0x1234,
len=3
int f(param_set_t* P in, keyblob_t** K out)
len: 03 00 00 00
p[0]: 01 00 00 00 CA FE AA AA
p[1]: 02 00 00 00 CA FE BB BB
p[2]: 01 00 00 00 CA FE AB AB
DBII Recorder
len: 00 02 00 00
k: 4d 4b 49 48 b4 6c 22 4a
...
Type Recovery
CA Interface
Type Recovery
Seeds / Fuzzing
Templates
size_t
{enum, short, blob}
{enum, short, blob}
{enum, short, blob}
UNKNOWN(12)
03 00 00 00 01 00 00 00
CA FE AA AA 02 00 00 00
CA FE BB BB 01 00 00 00
CA FE AB AB 00 00 00 00
66 75 7A 7A 74 68 69 73
Ioctl Recorder
Driver Interface
10 00 00 00 00 02 00 00
00 00 00 00 00 00 00 00
4d 4b 49 48 b4 6c 22 4a
...
off_t
size_t
pad[8]
uint8_t[]

State-Awareness

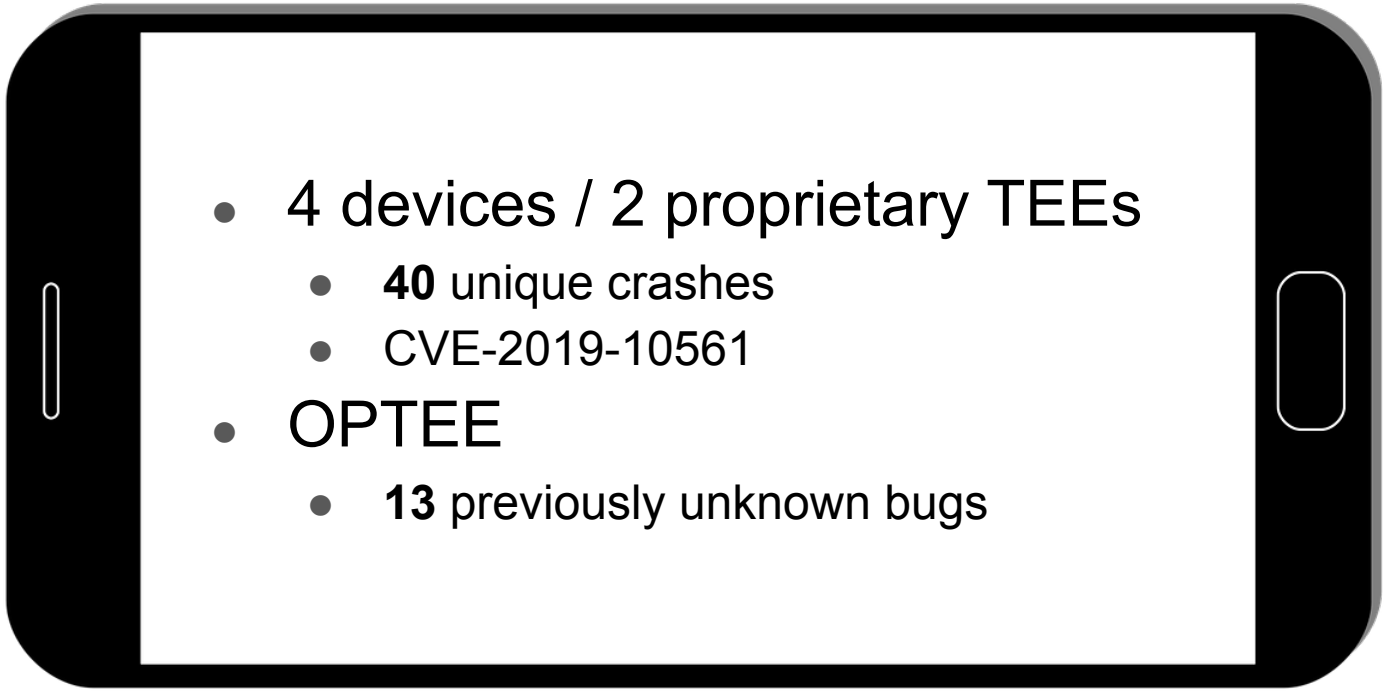


Fuzzing with TEEzz

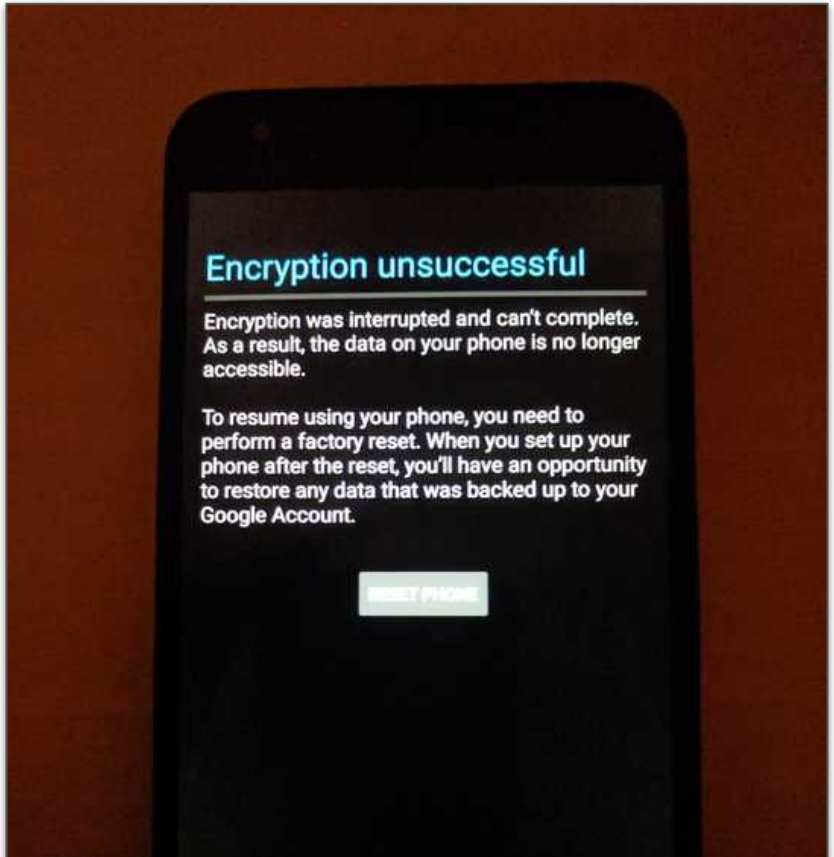
Demo

- P9 Lite
- Released 2016, May
- Android 6.0 (Marshmallow)
- Kirin 650 SoC
- TEE: TrustedCore

Evaluation – Finding Bugs

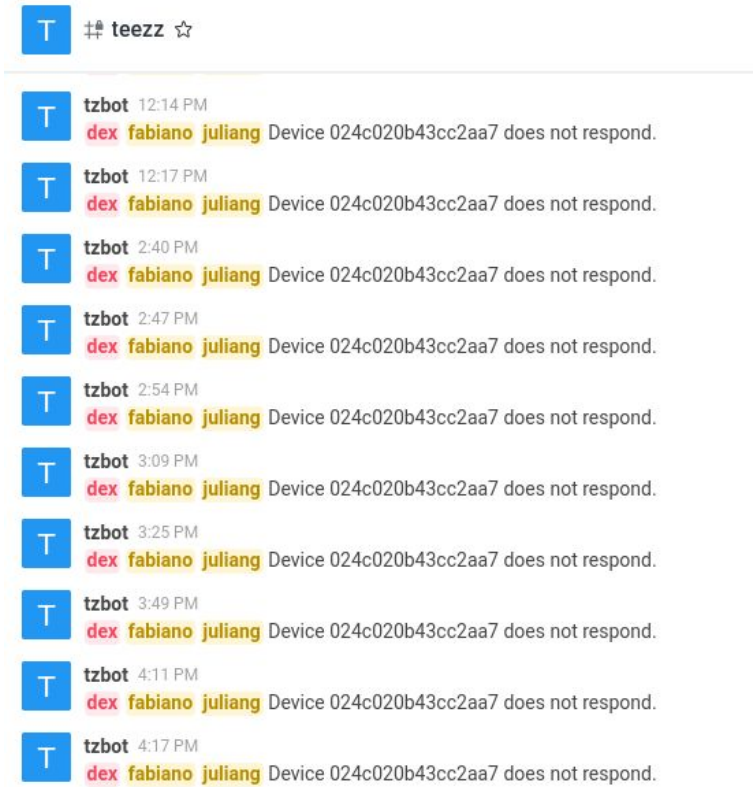


Nexus 5X Keymaster Fuzzing

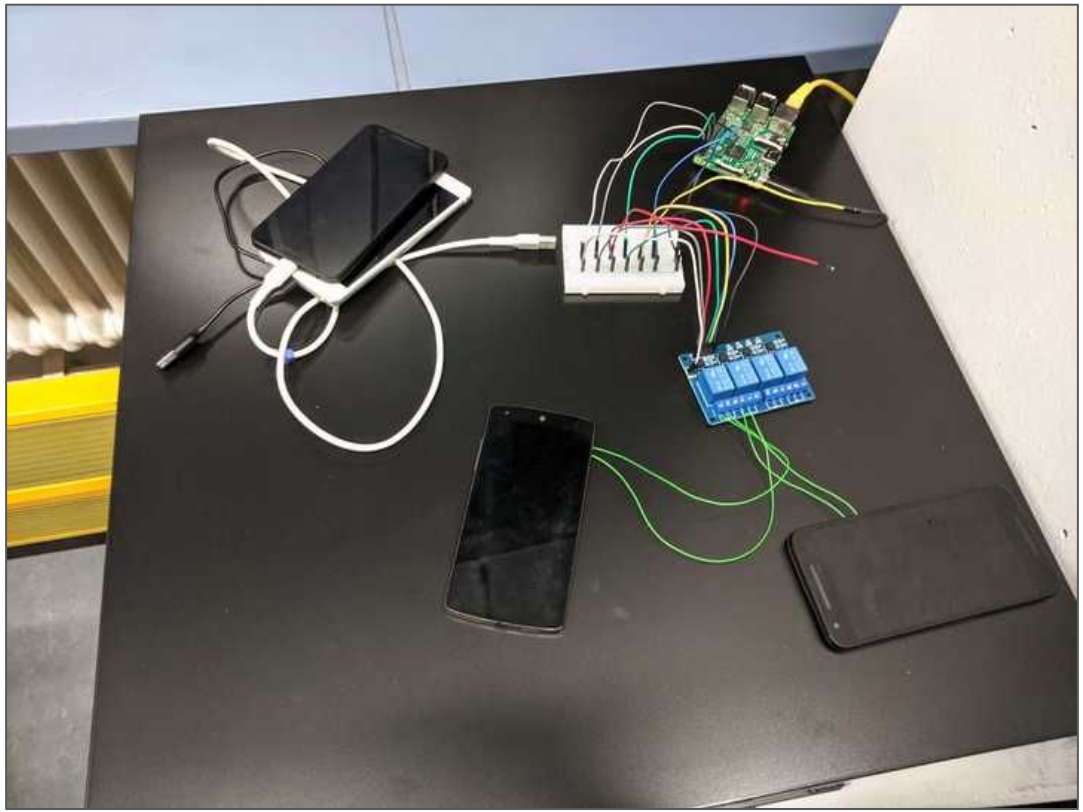


Anecdotes & Conclusion

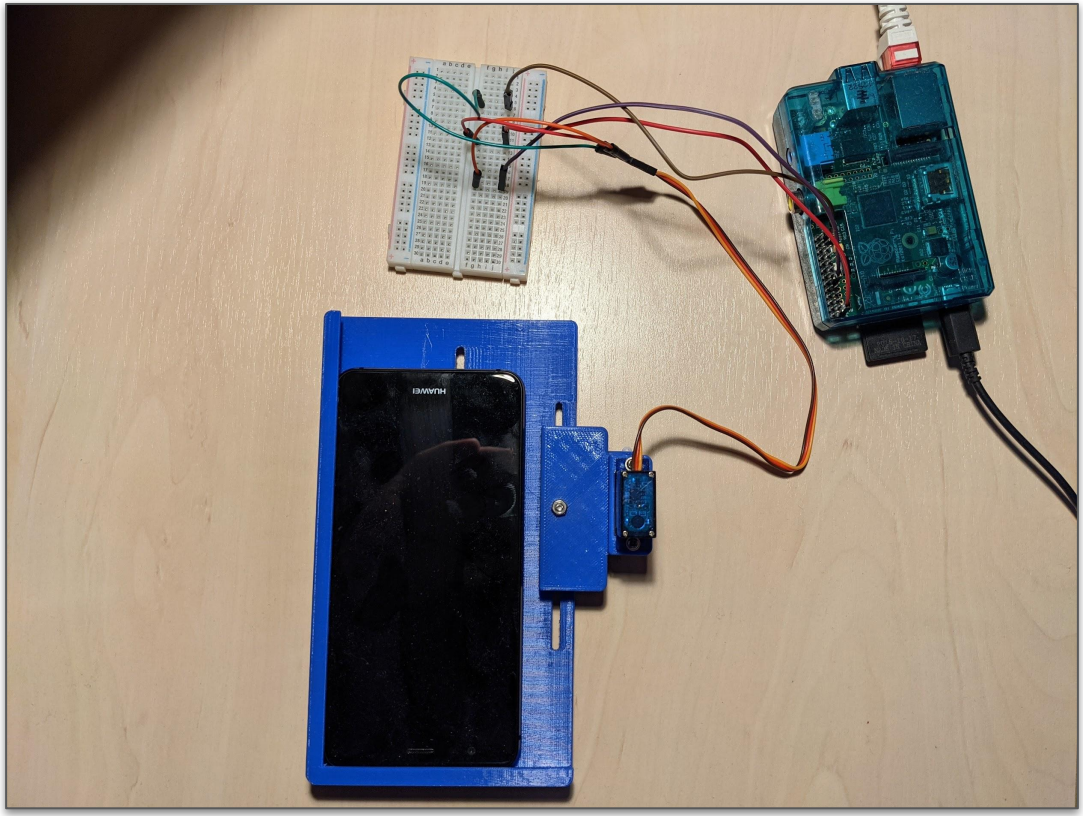
State Reset v1



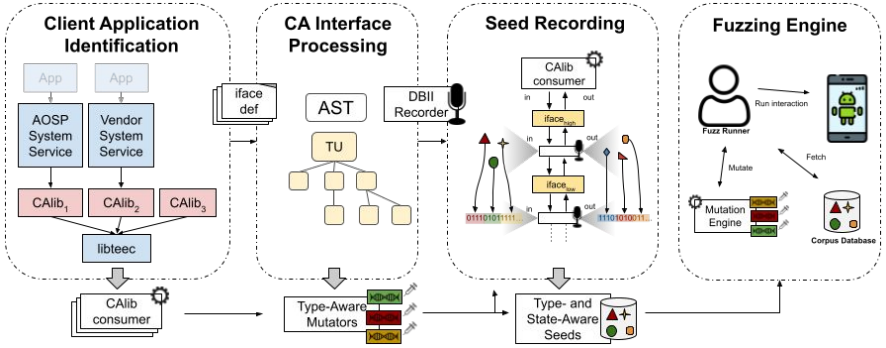
State Reset v2



State Reset v3



TEEzz: State- and Type-aware Black-box Fuzzing



- 4 devices / 2 proprietary TEEs
  40 unique crashes
  CVE-2019-10561
- OPTEE
  13 previously unknown bugs

Paper

Code

@0ddc0de

@0ddc0de@infosec.exchange

References and Attributions

Images are partially taken from Flaticon.com

References

- Fuzz One, Gal Beniamini
- TEEMU, Daniel Komaromy
- Samsung TZ Research, Alexandre Adamski et al. (Quarkslab)
- The Road to Qualcomm TZ Apps Fuzzing, Slava Kakkaveev (Checkpoint Research)
- Launching Feedback-Driven Fuzzing on TZ, Andrey Akimov
- PartEmu, Lee Harrison et al. (Samsung Research)