Usenix Security 2021

Android SmartTVs Vulnerability Discovery via Log-Guided Fuzzing

Yousra Aafer, **Wei You***, Yi Sun, Yu Shi, Xiangyu Zhang, Heng Yin

UNIVERSITY OF WATERLOO

RENMIN UNIVERSITY OF CHINA 1937 中国人民大学

PURDUE UNIVERSITY

UC RIVERSIDE UNIVERSITY OF CALIFORNIA

Why is SmartTV Security Important? A *Few Reasons*

Smart TVs

Account for the largest market share of Home IoT devices

Expected to achieve a market value of 253 billion USD by 2023

Plethora of attack vectors:

- **Physical channels: e.g., sending crafted broadcast signals**
- **Malware: SmartTV users can download SmartTV-specific Apps**

Broad Spectrum of Attack Consequences: Cyber + Physical

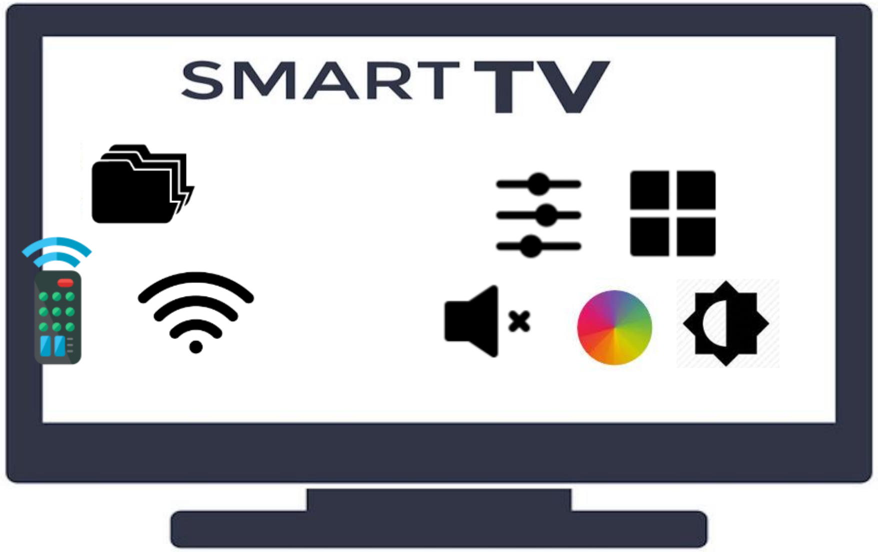
Goal

- Perform a systematic security evaluation of Android SmartTVs.
- Focus on customization aspects, performed to tailor the original OS for the SmartTV functionalities.

Background

Android SmartTVs run a heavily customized version of AOSP:

- Additional hardware, system components.
- Custom Functionalities are exposed to system and app developers through *dedicated APIs.*
- The number of custom APIs is high (up to 101 in H96Pro).



SmartTV APIs can open the door to various damages if not properly protected.

These APIs execute in the context of highly privileged processes.

Motivating Example

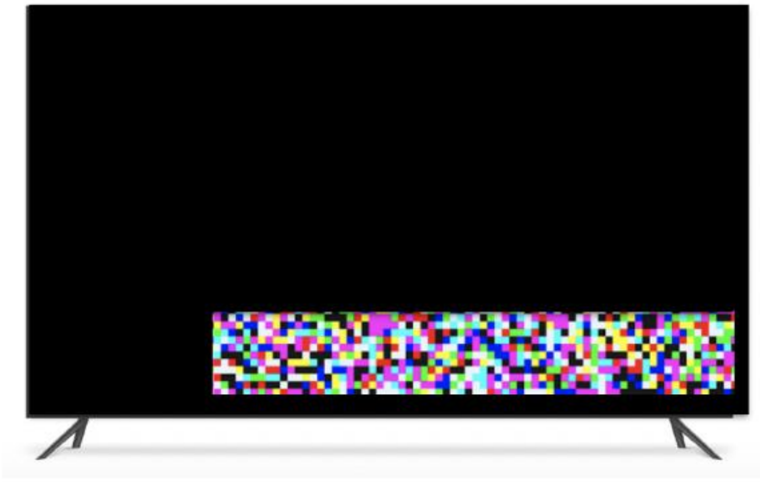
- Xiaomi MiBox3 introduces a new native API **SystemControl. setPosition(x, y, w, h)**

SystemControl. setPosition(x, y, w, h)

Motivating Example

- Xiaomi MiBox3 introduces a new native API **SystemControl. setPosition(x, y, w, h)**
  - The API does not enforce any access control and has persistent impact across reboot.
  - With the SmartTV ransomware on the rise, such APIs can be exploited to mount DoS attacks.

SystemControl. setPosition(1000, 1000, 1000, 1000)



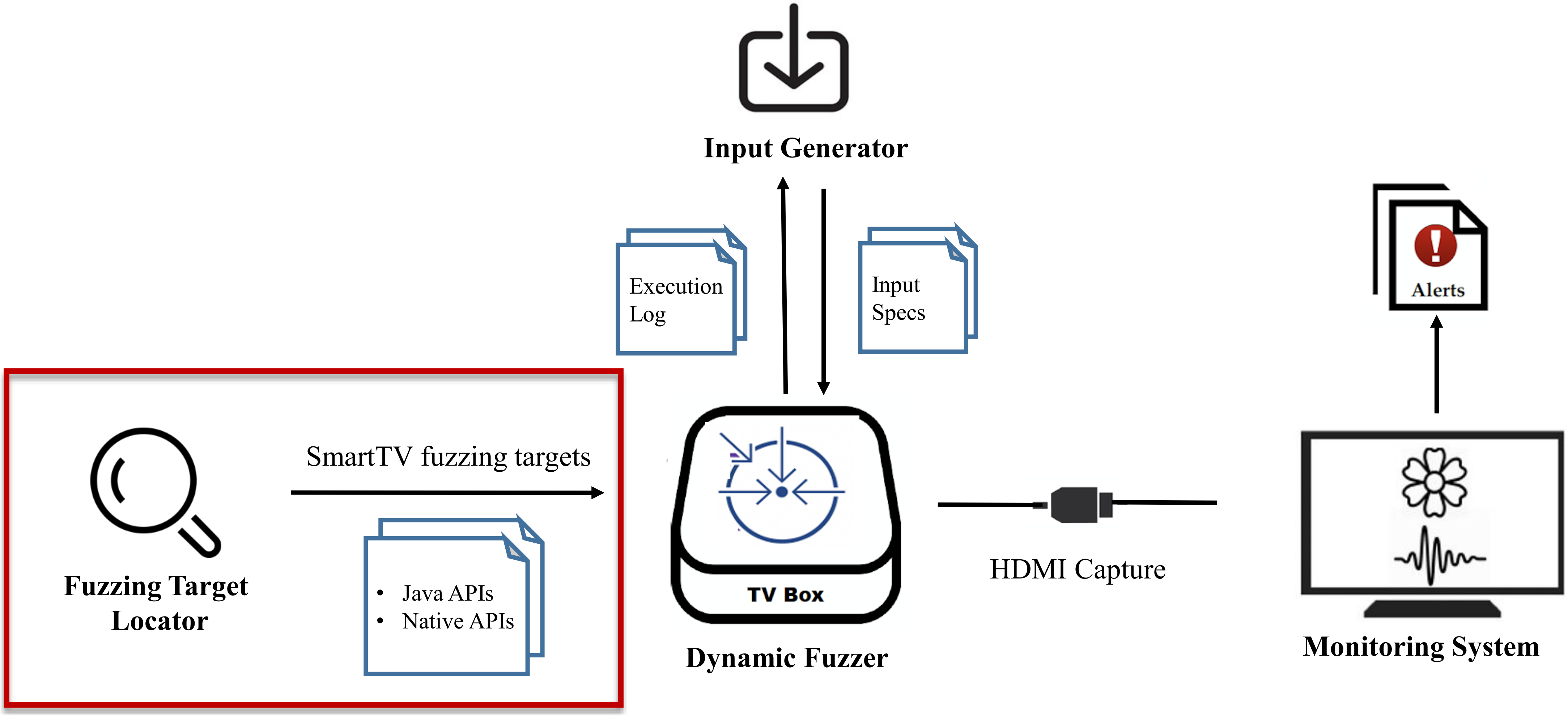
Detecting SmartTV Vulnerabilities

- We develop a specialized analysis framework to uncover hidden flaws, caused by unprotected APIs.

- **Why can’t we directly adopt static analysis tools?**
  - Additions are implemented in C++ and / or Java
- **Why can’t we directly adopt existing testing approaches?**
  - Assessing execution feedback is challenging

The Audio / Visual behavior is decoupled from the internal states → the system might be functioning correctly when the display / sound is messed up.

Our Approach: Fuzz-testing



Input Generator

Execution Log

Input Specs

Alerts

Fuzzing Target Locator

SmartTV fuzzing targets

- Java APIs
- Native APIs

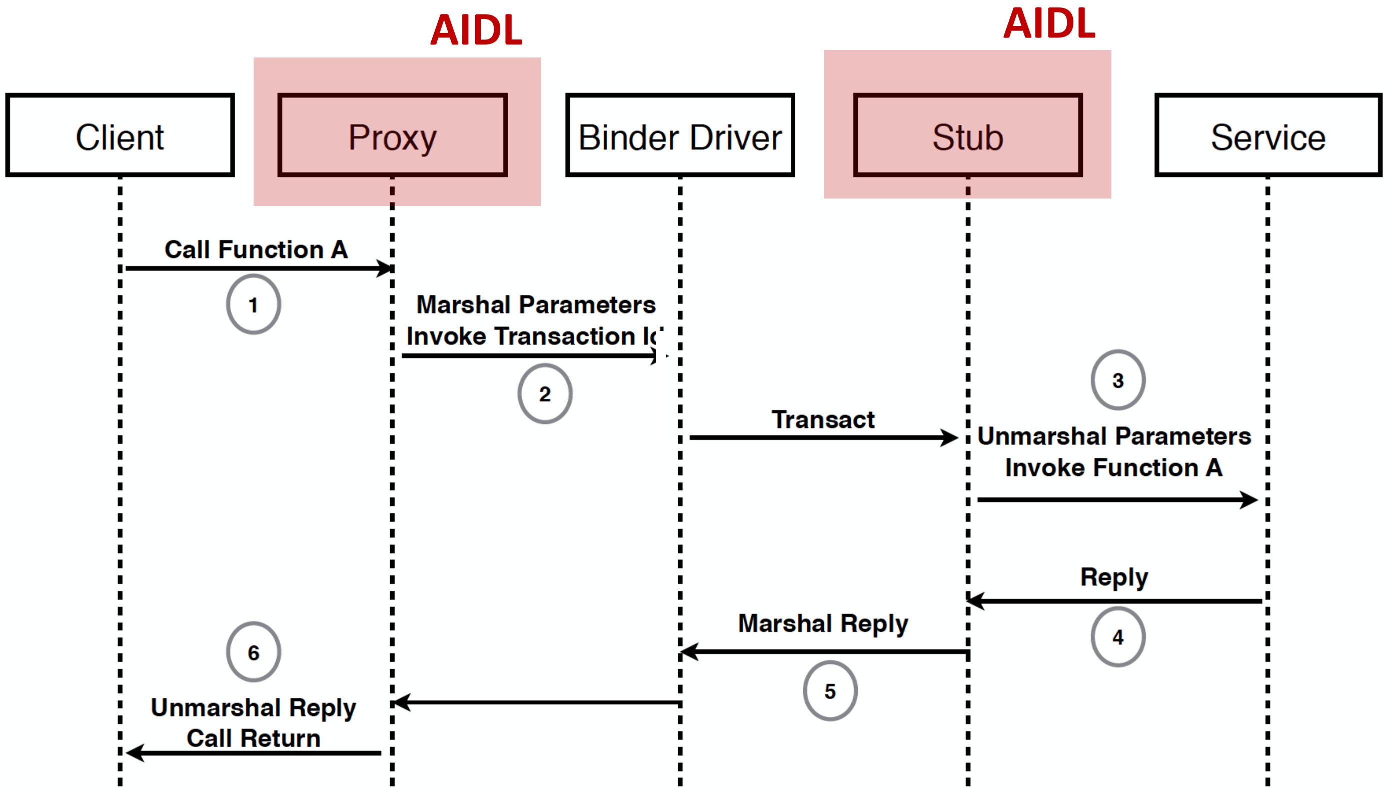
Dynamic Fuzzer

HDMI Capture

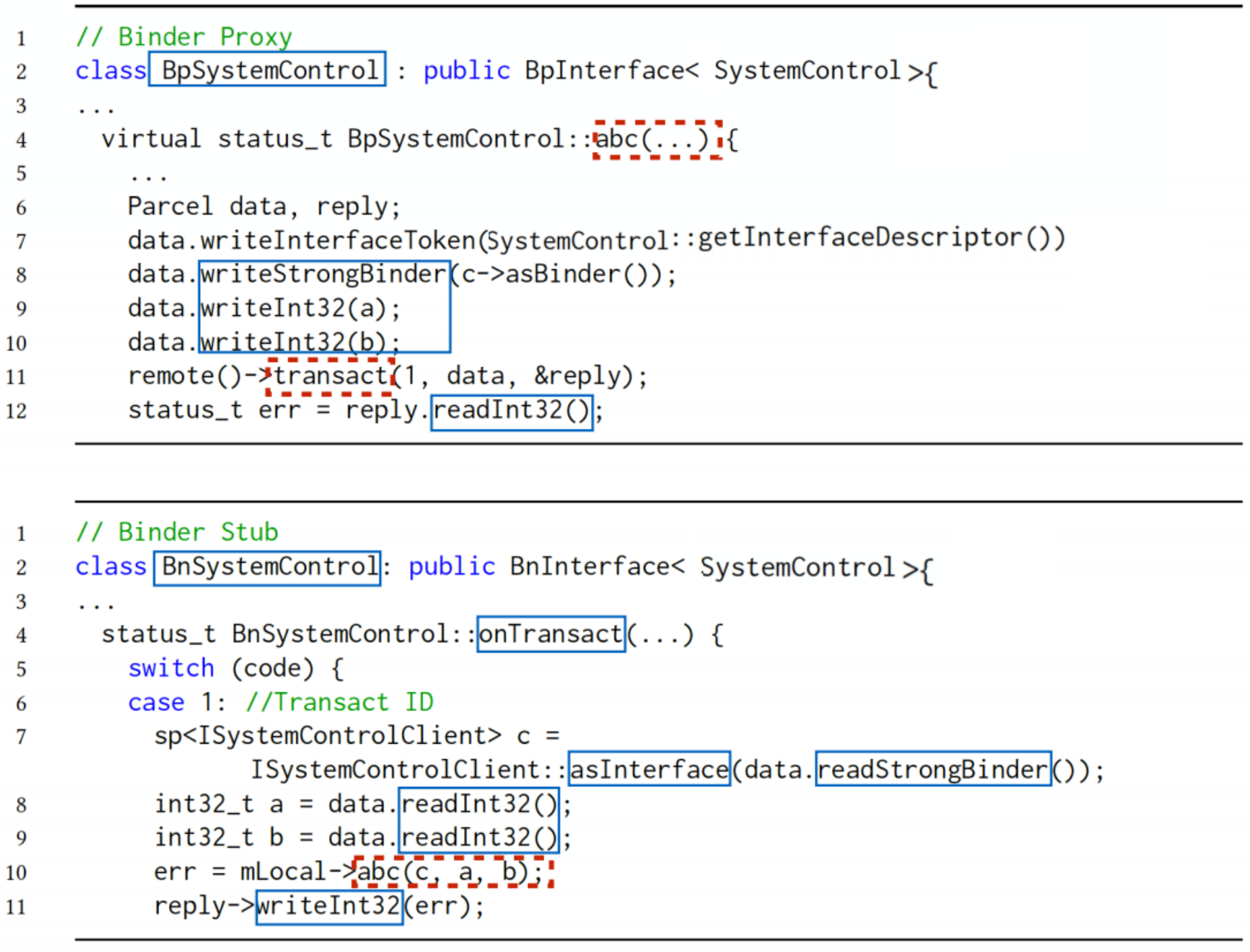
Monitoring System

Fuzzing Target locator

- We recover native API interfaces at the low-level Binder IPC through binary analysis.
- Recovering Native APIs Interfaces: Binder transaction ids, arguments types and order.



Extracting Native Function Interfaces



```
// Binder Proxy
class BpSystemControl : public BpInterface< SystemControl >{
...
  virtual status_t BpSystemControl::abc(...) {
    ...
    Parcel data, reply;
    data.writeInterfaceToken(SystemControl::getInterfaceDescriptor())
    data.writeStrongBinder(c->asBinder());
    data.writeInt32(a);
    data.writeInt32(b);
    remote()->transact(1, data, &reply);
    status_t err = reply.readInt32();
```

```
// Binder Stub
class BnSystemControl: public BnInterface< SystemControl >{
...
  status_t BnSystemControl::onTransact(...) {
    switch (code) {
    case 1: //Transact ID
      sp<ISystemControlClient> c =
             ISystemControlClient::asInterface(data.readStrongBinder());
      int32_t a = data.readInt32();
      int32_t b = data.readInt32();
      err = mLocal->abc(c, a, b);
      reply->writeInt32(err);
```

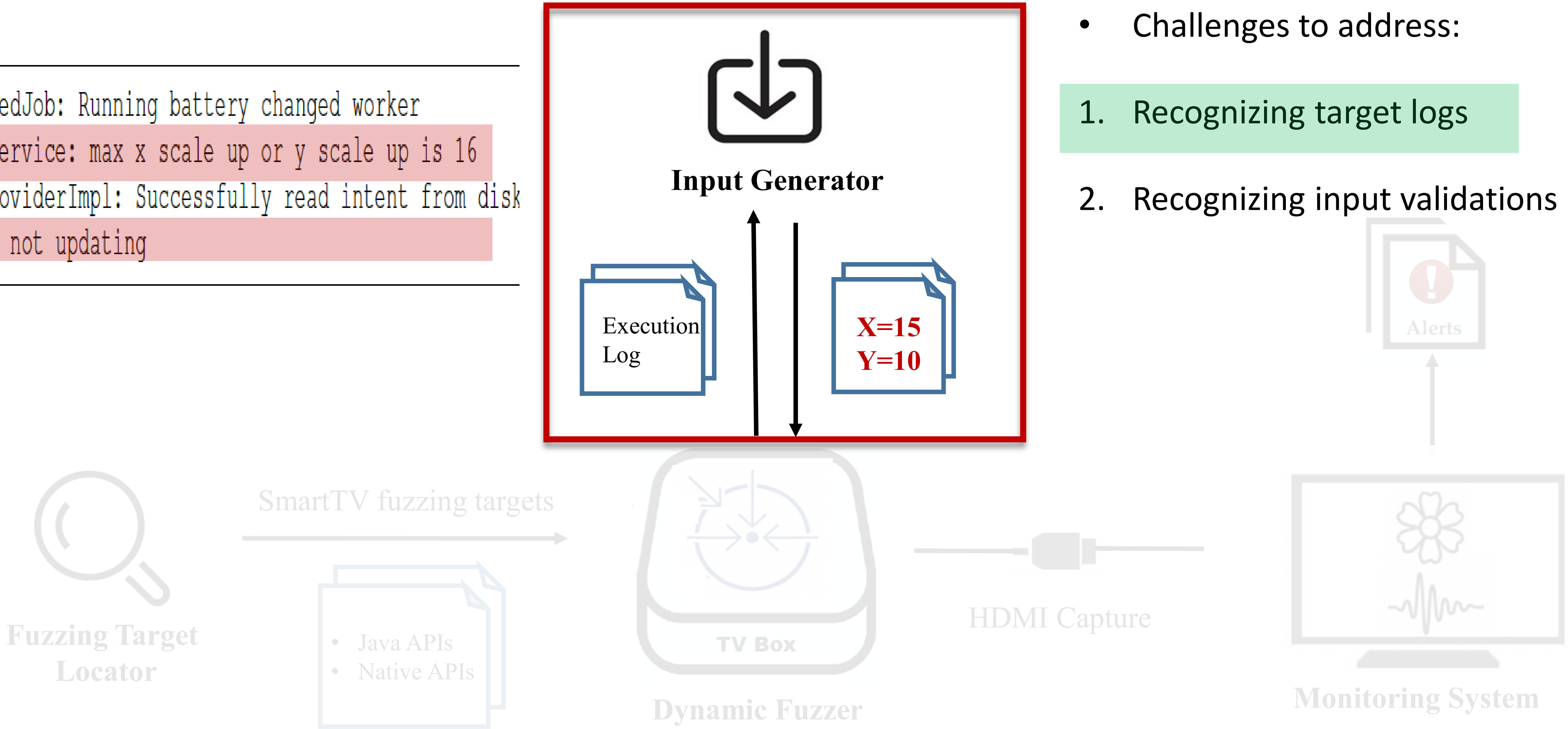
□ symbol reserved

⬚ symbol absent

Step:
1. Identify function bodies within the binder proxies
2. Extract traction id and parameter types
   (inferred through the proxy’s marshaling methods)
3. Confirm the result by analyzing the binder stubs.

Our Approach: Fuzz-testing

```
BatteryChangedJob: Running battery changed worker
ImagePlayerService: max x scale up or y scale up is 16
DiskIntentProviderImpl: Successfully read intent from disk
MediaPlayer: not updating
```

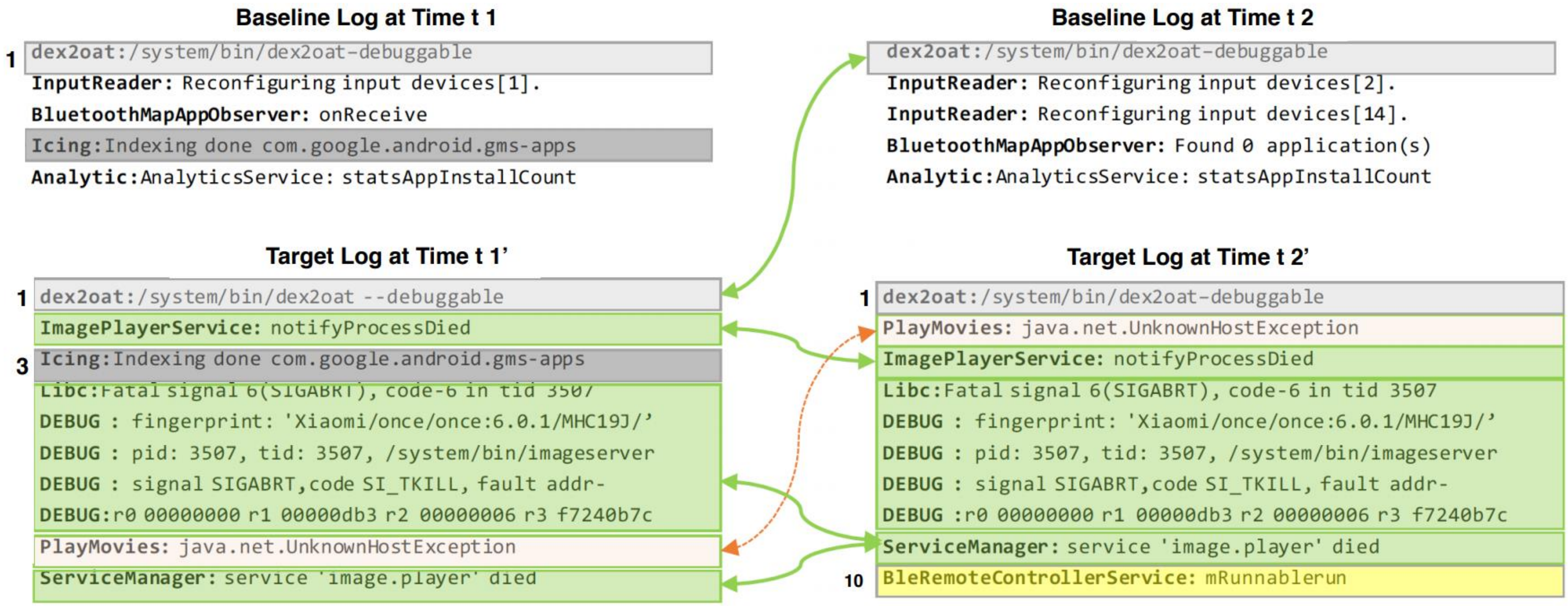


- Challenges to address:

1. Recognizing target logs
2. Recognizing input validations

Recognizing Target Logs

$$score(i) = \begin{cases} 1 & \text{if } \frac{n_i^{target}}{N^{target}} \geq 0.9 \text{ and } \frac{n_i^{baseline}}{N^{baseline}} \leq 0.1 \\ 0, & \text{otherwise} \end{cases}$$



Baseline Log at Time t 1

```
1 dex2oat:/system/bin/dex2oat-debuggable
  InputReader: Reconfiguring input devices[1].
  BluetoothMapAppObserver: onReceive
  Icing:Indexing done com.google.android.gms-apps
  Analytic:AnalyticsService: statsAppInstallCount
```

Baseline Log at Time t 2

```
dex2oat:/system/bin/dex2oat-debuggable
InputReader: Reconfiguring input devices[2].
InputReader: Reconfiguring input devices[14].
BluetoothMapAppObserver: Found 0 application(s)
Analytic:AnalyticsService: statsAppInstallCount
```

Target Log at Time t 1’

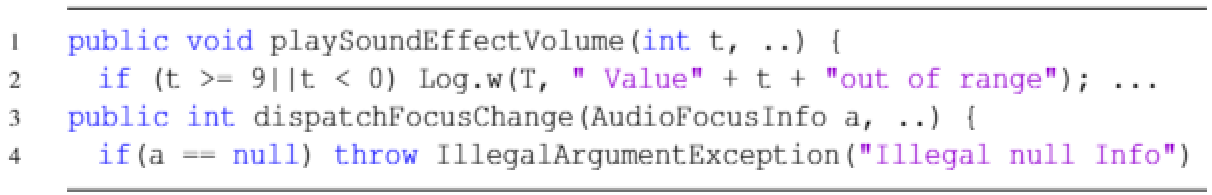
```
1 dex2oat:/system/bin/dex2oat --debuggable
  ImagePlayerService: notifyProcessDied
3 Icing:Indexing done com.google.android.gms-apps
  Libc:Fatal signal 6(SIGABRT), code-6 in tid 3507
  DEBUG : fingerprint: 'Xiaomi/once/once:6.0.1/MHC19J/’
  DEBUG : pid: 3507, tid: 3507, /system/bin/imageserver
  DEBUG : signal SIGABRT,code SI_TKILL, fault addr-
  DEBUG:r0 00000000 r1 00000db3 r2 00000006 r3 f7240b7c
  PlayMovies: java.net.UnknownHostException
  ServiceManager: service 'image.player' died
```

Target Log at Time t 2’

```
 1 dex2oat:/system/bin/dex2oat-debuggable
   PlayMovies: java.net.UnknownHostException
   ImagePlayerService: notifyProcessDied
   Libc:Fatal signal 6(SIGABRT), code-6 in tid 3507
   DEBUG : fingerprint: 'Xiaomi/once/once:6.0.1/MHC19J/’
   DEBUG : pid: 3507, tid: 3507, /system/bin/imageserver
   DEBUG : signal SIGABRT,code SI_TKILL, fault addr-
   DEBUG :r0 00000000 r1 00000db3 r2 00000006 r3 f7240b7c
   ServiceManager: service 'image.player' died
10 BleRemoteControllerService: mRunnablerun
```

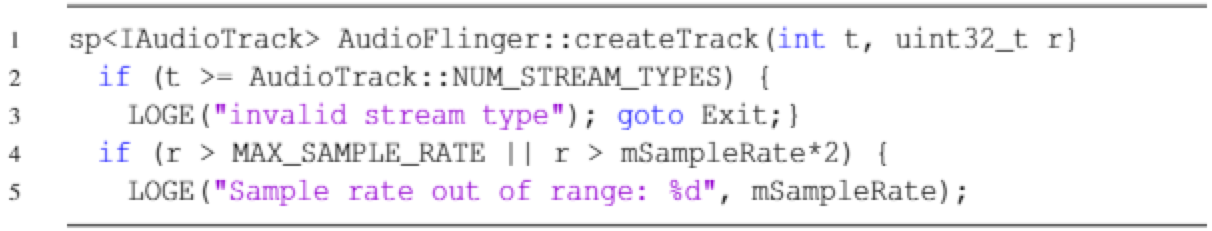
Log Excerpts before and after calling the (native) target API ImagePlayer.XYZ()

Recognizing Input Validations



```
public void playSoundEffectVolume(int t, ..) {
  if (t >= 9||t < 0) Log.w(T, " Value" + t + "out of range"); ...
public int dispatchFocusChange(AudioFocusInfo a, ..) {
  if(a == null) throw IllegalArgumentException("Illegal null Info")
```

Java

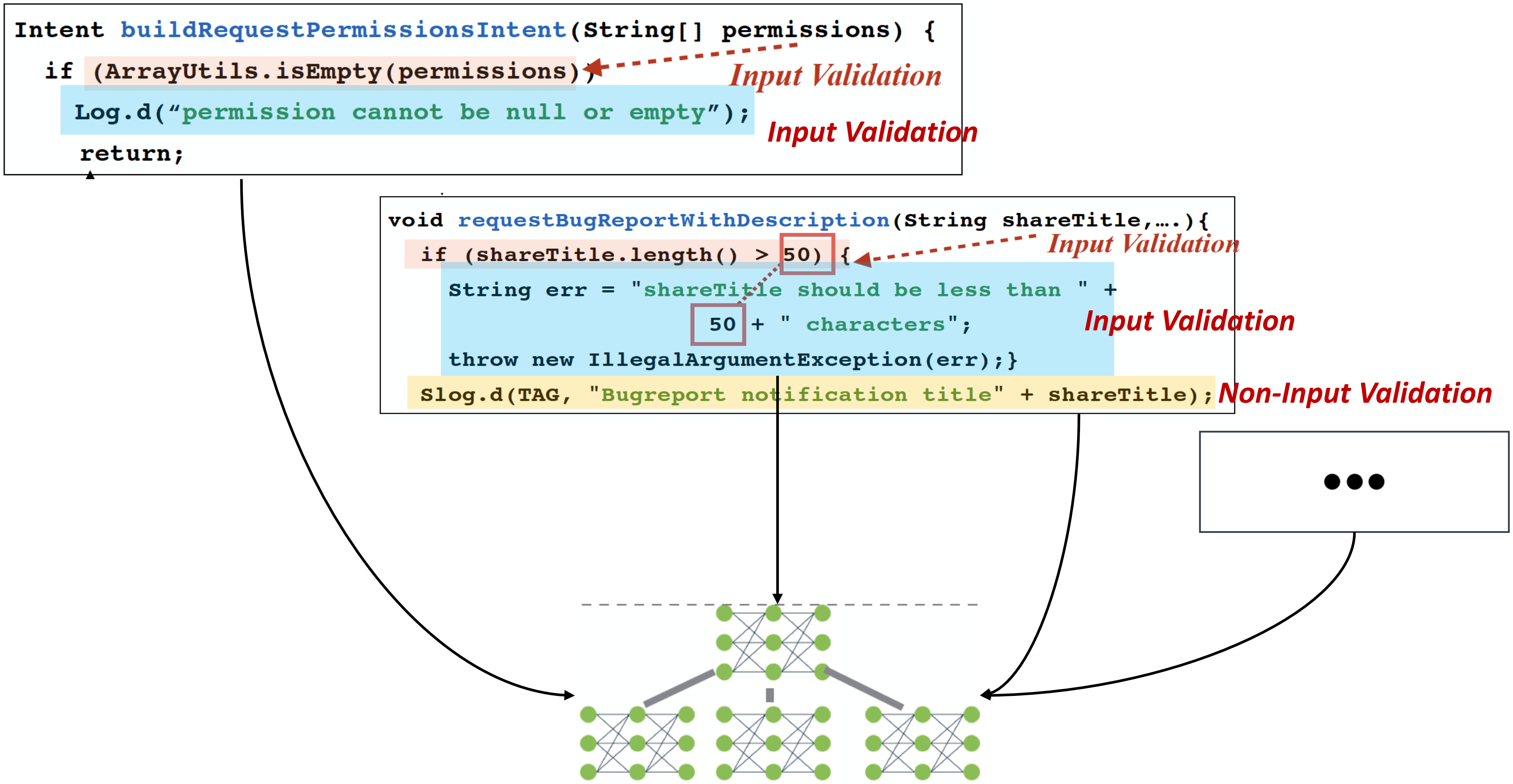


```
sp<IAudioTrack> AudioFlinger::createTrack(int t, uint32_t r}
  if (t >= AudioTrack::NUM_STREAM_TYPES) {
    LOGE("invalid stream type"); goto Exit;}
  if (r > MAX_SAMPLE_RATE || r > mSampleRate*2) {
    LOGE("Sample rate out of range: %d", mSampleRate);
```

Native

- Feasibility of learning from log messages in Java to classify log messages from native
- Need of sophisticated NLP techniques as keyword lookup is insufficient.

Deep Learning for Message Classification



```
Intent buildRequestPermissionsIntent(String[] permissions) {
  if (ArrayUtils.isEmpty(permissions))
    Log.d(“permission cannot be null or empty”);
     return;
```

Input Validation

Input Validation

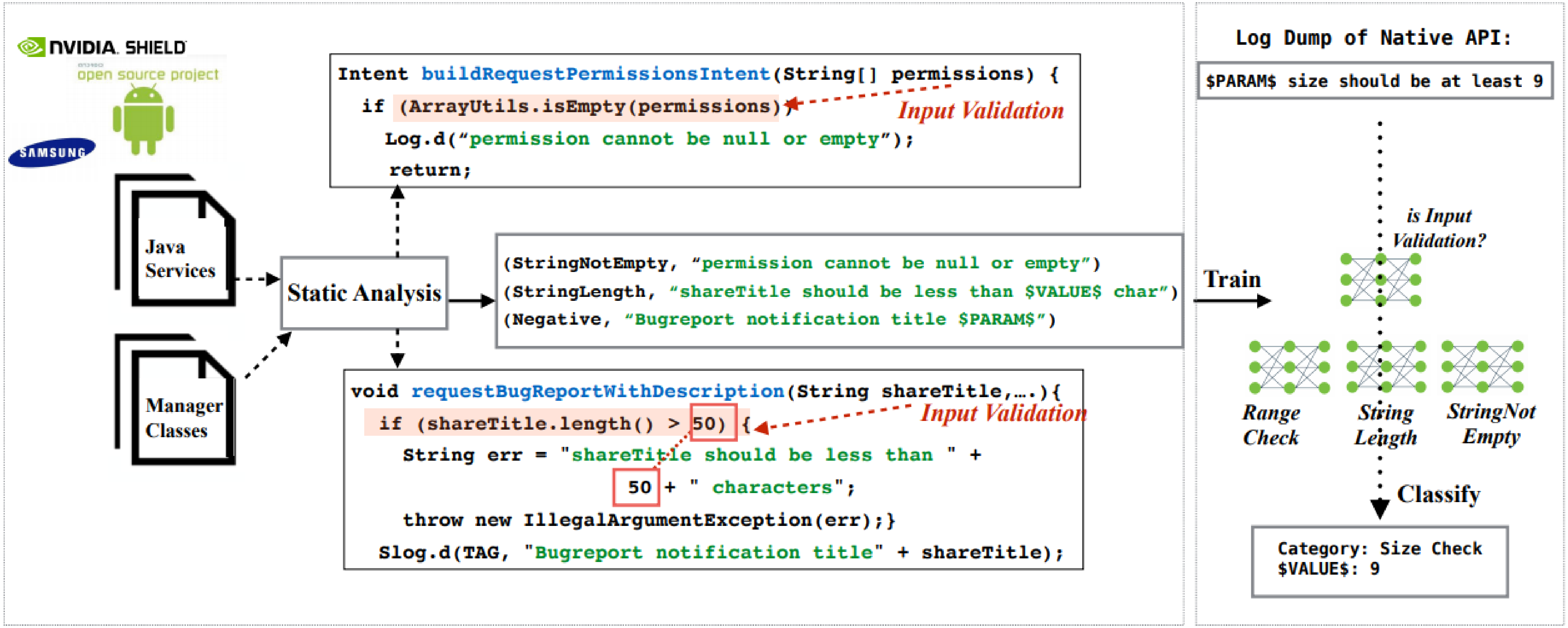
```
void requestBugReportWithDescription(String shareTitle,….){
  if (shareTitle.length() > 50) {
    String err = "shareTitle should be less than " +
                       50 + " characters";
    throw new IllegalArgumentException(err);}
  Slog.d(TAG, "Bugreport notification title" + shareTitle);
```

Input Validation

Input Validation

Non-Input Validation

Input Validation Classification



```
Intent buildRequestPermissionsIntent(String[] permissions) {
  if (ArrayUtils.isEmpty(permissions))
    Log.d("permission cannot be null or empty");
    return;
```

```
(StringNotEmpty, "permission cannot be null or empty")
(StringLength, "shareTitle should be less than $VALUE$ char")
(Negative, "Bugreport notification title $PARAM$")
```

```
void requestBugReportWithDescription(String shareTitle,….){
  if (shareTitle.length() > 50) {
    String err = "shareTitle should be less than " +
                         50 + " characters";
    throw new IllegalArgumentException(err);}
  Slog.d(TAG, "Bugreport notification title" + shareTitle);
```

Log Dump of Native API:

```
$PARAM$ size should be at least 9
```

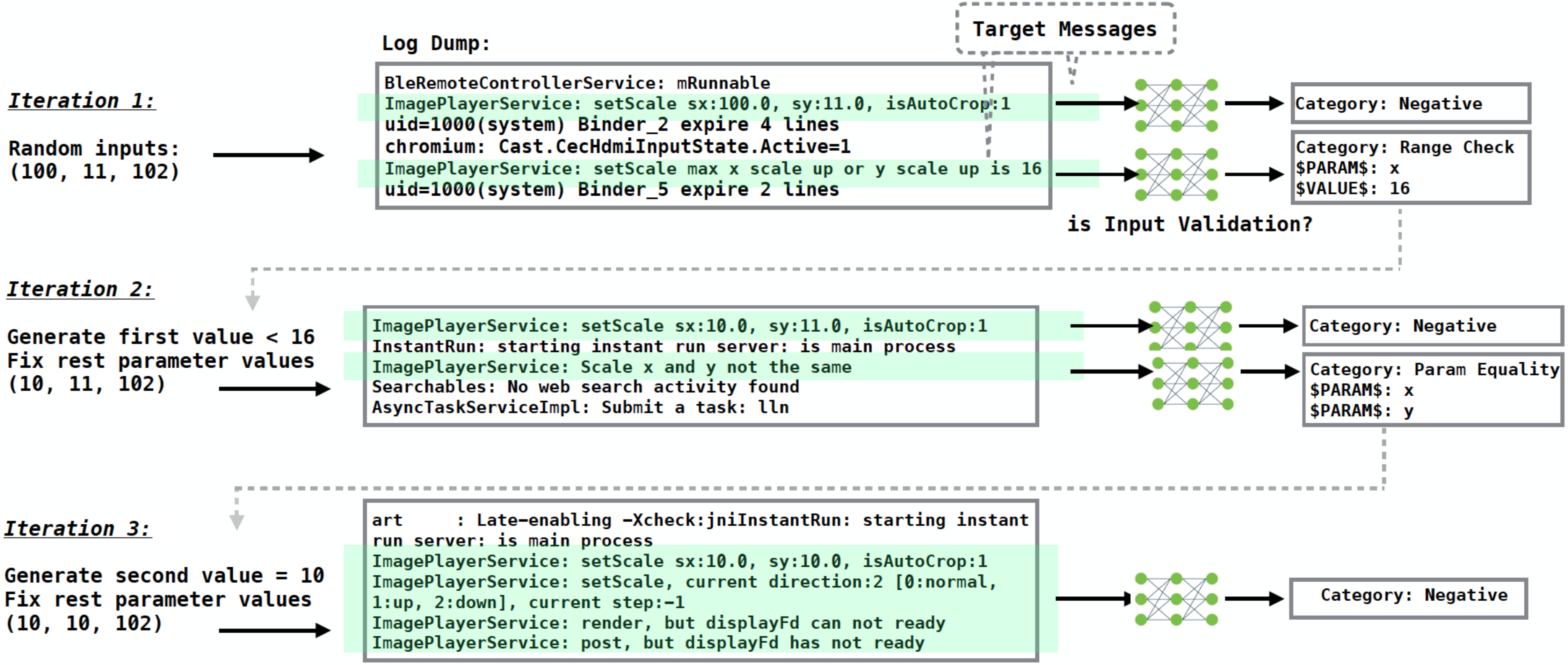
```
Category: Size Check
$VALUE$: 9
```

(A) Training Data Collection

(B) Log Classifier Training

Log-Guided Fuzzing

Example: fuzzing ABC(int, int, float)



Target Messages

Log Dump:

Iteration 1:

Random inputs:
(100, 11, 102)

```
BleRemoteControllerService: mRunnable
ImagePlayerService: setScale sx:100.0, sy:11.0, isAutoCrop:1
uid=1000(system) Binder_2 expire 4 lines
chromium: Cast.CecHdmiInputState.Active=1
ImagePlayerService: setScale max x scale up or y scale up is 16
uid=1000(system) Binder_5 expire 2 lines
```

```
Category: Negative
```

```
Category: Range Check
$PARAM$: x
$VALUE$: 16
```

is Input Validation?

Iteration 2:

Generate first value < 16
Fix rest parameter values
(10, 11, 102)

```
ImagePlayerService: setScale sx:10.0, sy:11.0, isAutoCrop:1
InstantRun: starting instant run server: is main process
ImagePlayerService: Scale x and y not the same
Searchables: No web search activity found
AsyncTaskServiceImpl: Submit a task: lln
```

```
Category: Negative
```

```
Category: Param Equality
$PARAM$: x
$PARAM$: y
```

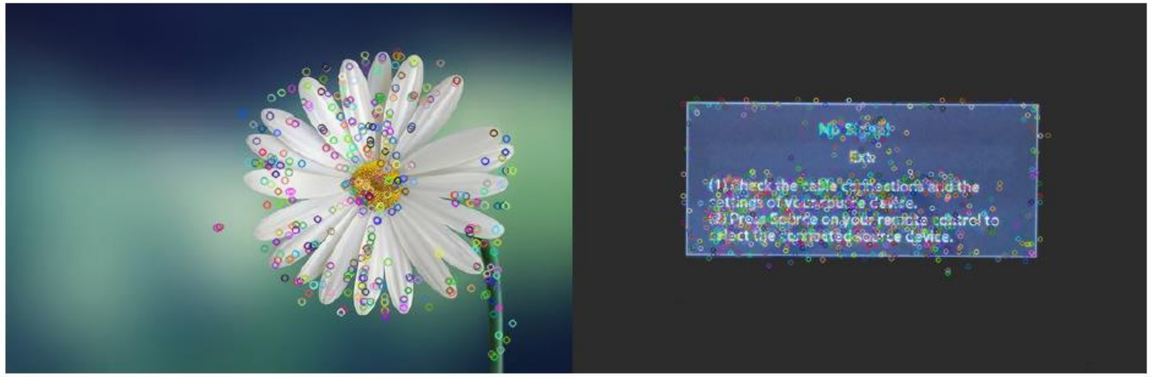
Iteration 3:

Generate second value = 10
Fix rest parameter values
(10, 10, 102)

```
art     : Late-enabling -Xcheck:jniInstantRun: starting instant
run server: is main process
ImagePlayerService: setScale sx:10.0, sy:10.0, isAutoCrop:1
ImagePlayerService: setScale, current direction:2 [0:normal,
1:up, 2:down], current step:-1
ImagePlayerService: render, but displayFd can not ready
ImagePlayerService: post, but displayFd has not ready
```

```
Category: Negative
```

Monitoring System

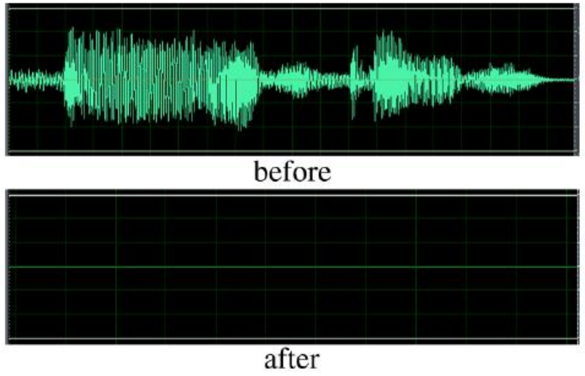


No Signal

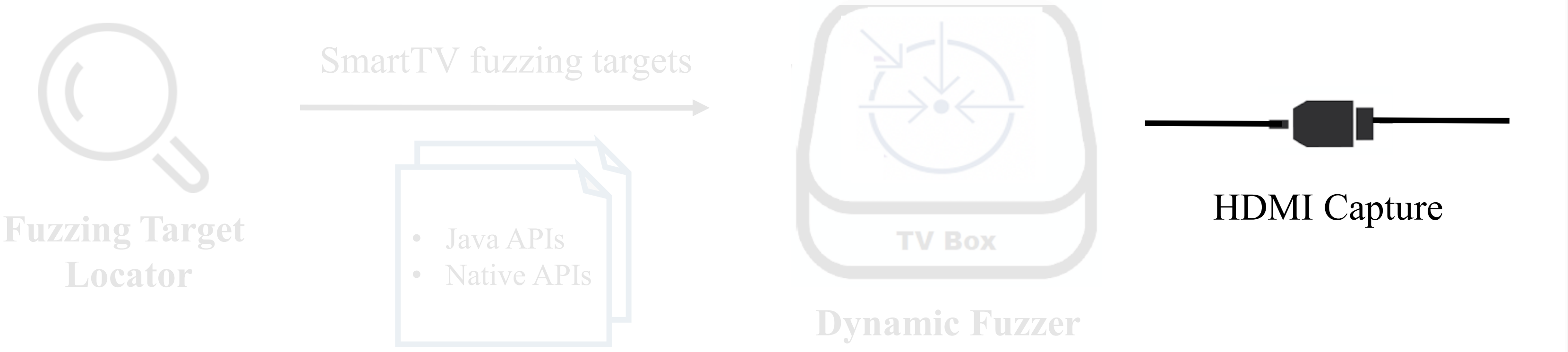
Exit

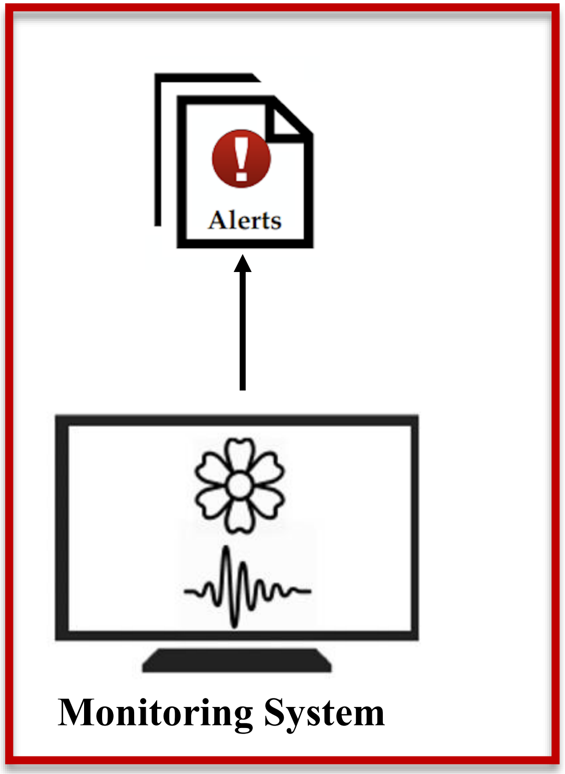
(1) Check the cable connections and the settings of your source device.
(2) Press Source on your remote control to select the connected source device.

(a) Display before and after invoking `DisplayManager.enableInterface`



(b) Audio comparison





Evaluation

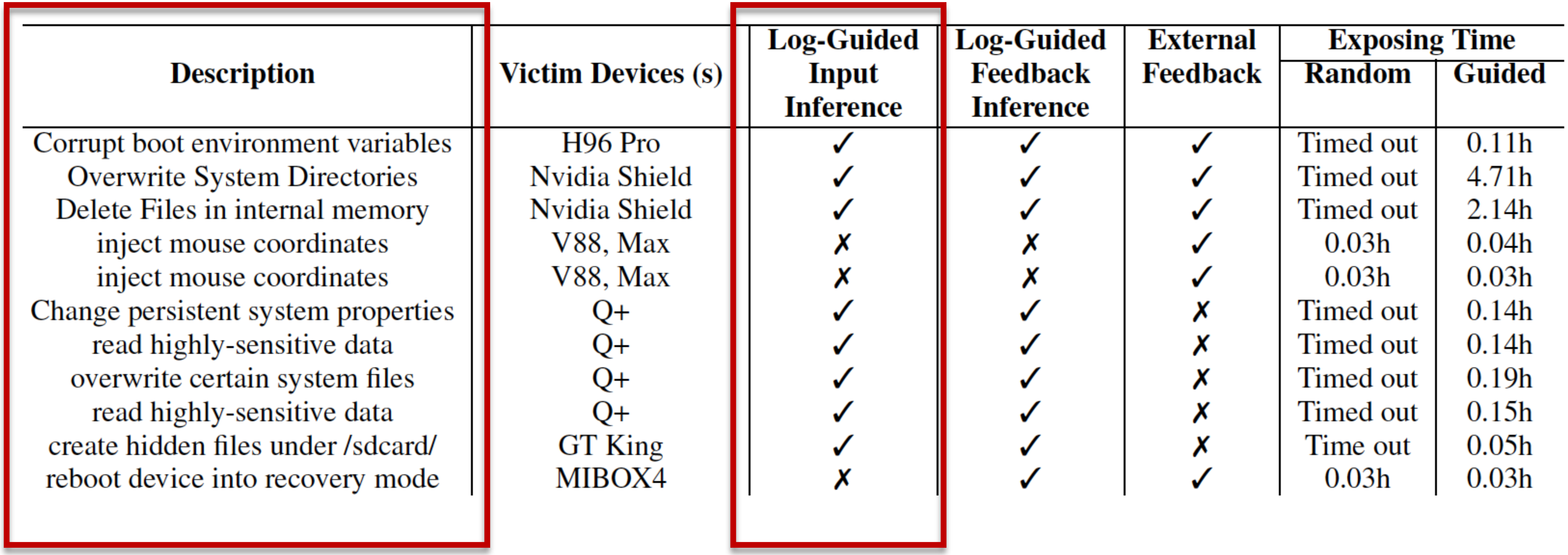
- 11 Android TVBoxes evaluated
  - including Nvidia Shield, MIBOX 3, etc.
  - each analyzed device contained 1 to 9 vulnerabilities

- 37 flaws discovered
  - 11 high-impact cyber threats
  - 10 new memory corruptions
  - 16 visual/auditory anomalies

- confirmed and fixed by the vendors

Evaluation

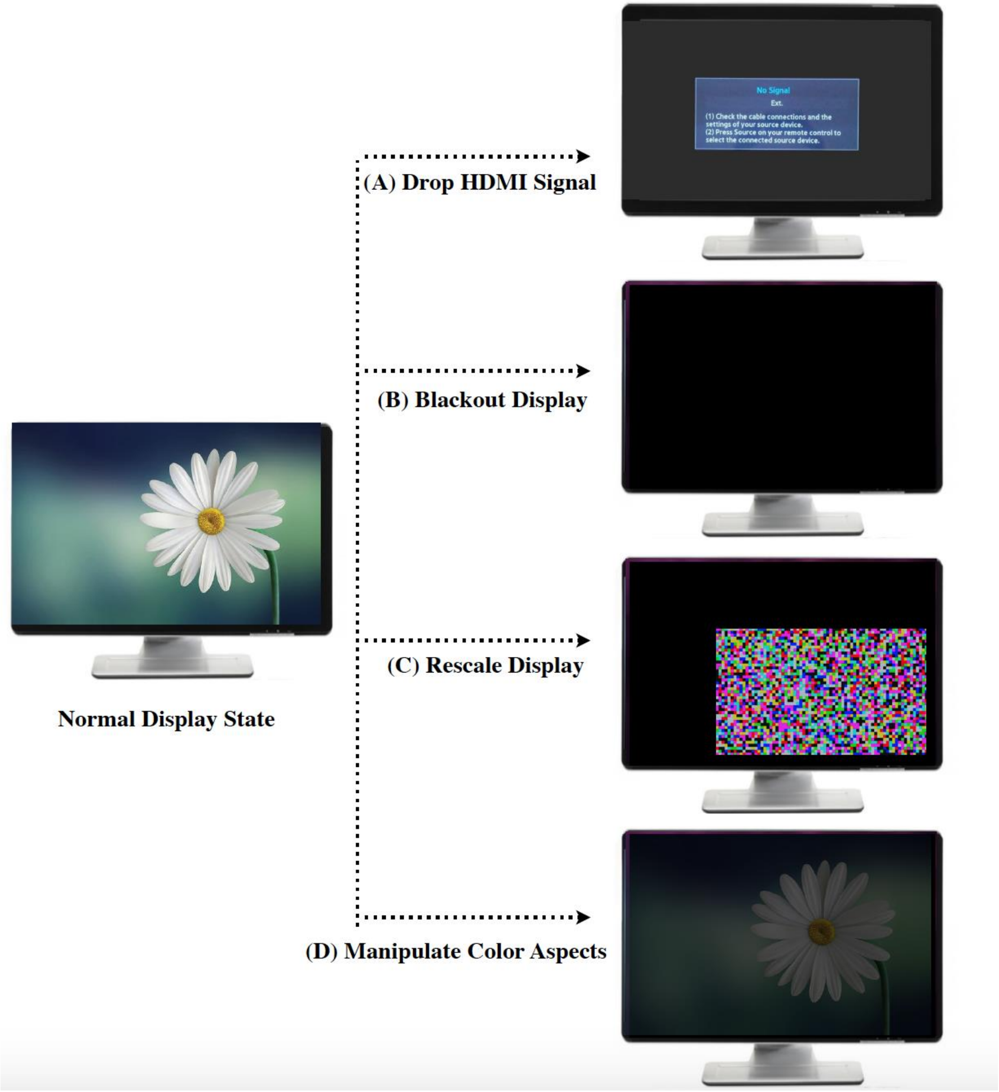
Cyber threats and Memory Corruptions



Description	Victim Devices (s)	Log-Guided Input Inference	Log-Guided Feedback Inference	External Feedback	Exposing Time	
					Random	Guided
Corrupt boot environment variables	H96 Pro	✓	✓	✓	Timed out	0.11h
Overwrite System Directories	Nvidia Shield	✓	✓	✓	Timed out	4.71h
Delete Files in internal memory	Nvidia Shield	✓	✓	✓	Timed out	2.14h
inject mouse coordinates	V88, Max	✗	✗	✓	0.03h	0.04h
inject mouse coordinates	V88, Max	✗	✗	✓	0.03h	0.03h
Change persistent system properties	Q+	✓	✓	✗	Timed out	0.14h
read highly-sensitive data	Q+	✓	✓	✗	Timed out	0.14h
overwrite certain system files	Q+	✓	✓	✗	Timed out	0.19h
read highly-sensitive data	Q+	✓	✓	✗	Timed out	0.15h
create hidden files under /sdcard/	GT King	✓	✓	✗	Time out	0.05h
reboot device into recovery mode	MIBOX4	✗	✓	✓	0.03h	0.03h

Evaluation

Physical Vulnerabilities



No Signal

Ext.

(1) Check the cable connections and the settings of your source device.
(2) Press Source on your remote control to select the connected source device.

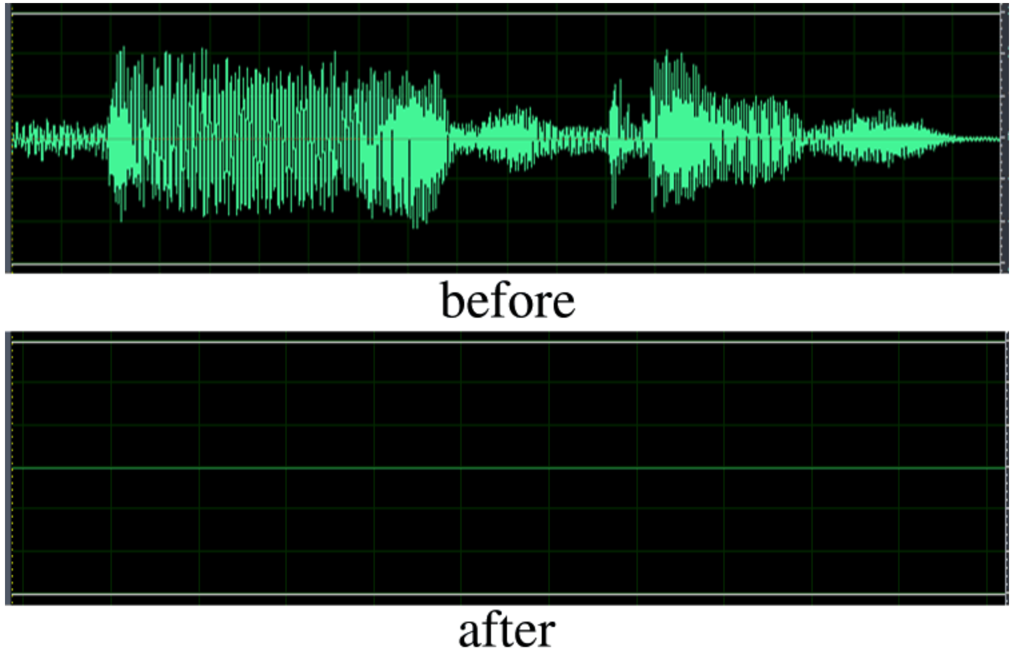
(A) Drop HDMI Signal

(B) Blackout Display

(C) Rescale Display

(D) Manipulate Color Aspects

Normal Display State



before

after

Related Work

- **IOT-Fuzzer: Discovering Memory Corruptions in IOT through App-Based Fuzzing.** In Proceedings of NDSS 2018.

- **FIRM-AFL: High-Throughput Greybox Fuzzing of IoT Firmware via Augmented Process Emulation.** In Proceedings of Usenix Security 2019.

Conclusion

- New technique
  - integrate static analysis and log-guided dynamic fuzzing
  - automatically detect cyber and physical anomalies
  - provide a solution when instrumentation and execution feedback is not feasible

- New findings
  - reveal security-critical threats of Android SmartTV API additions
  - cyber threats, memory corruptions and physical anomalies

Thank you!

Q & A

Contact:
youwei@ruc.edu.cn