Pandawan: Quantifying Progress in Linux-based Firmware Rehosting

Ioannis Angelakopoulos, Gianluca Stringhini, and Manuel Egele

33rd Usenix Security Symposium
Date: 08/16/2024

SeclaBU

BOSTON UNIVERSITY

Internet of Things

In 2024 more than 17 billion devices

Vulnerable to cyber attacks

Almost 11 million attacks (e.g., device breaches) in December 2022

Threats become more frequent and sophisticated!

The dark web's criminal minds see Internet of Things as next big hacking prize

PUBLISHED MON, JAN 9 2023•9:29 AM EST

IT'S PATCH TIME ONCE AGAIN —

High-severity vulnerabilities affect a wide range of Asus router models

Many models receive patches; others will need to be replaced.

DAN GOODIN - 6/17/2024, 2:39 PM

OVHcloud Hit with Record 840 Million PPS DDoS Attack Using MikroTik Routers

Jul 05, 2024 Ravie Lakshmanan Network Security / DDoS Attack

Sources:
https://www.amazon.com/
https://www.reddit.com/r/cybermaterial/comments/zyp2qj/enjoy_4_memes_about_iot/
https://www.cnbc.com/2023/01/09/the-dark-webs-criminal-minds-see-iot-as-the-next-big-hacking-prize.html
https://arstechnica.com/security/2024/06/high-severity-vulnerabilities-affect-a-wide-range-of-asus-router-models/
https://www.statista.com/statistics/1322216/worldwide-internet-of-things-attacks/
https://thehackernews.com/2024/07/ovhcloud-hit-with-record-840-million.html

Linux-based IoT Firmware Re-hosting

Full-system re-hosting frameworks categorization:

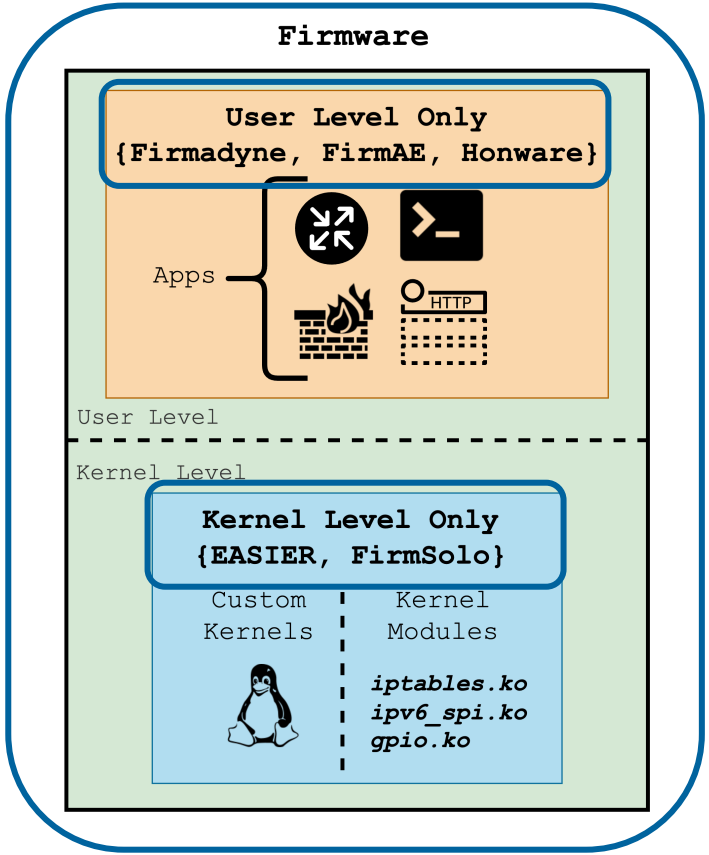
- User level re-hosting
- Kernel level re-hosting (i.e., kernel modules)

Current solutions absent of two important features:

- H… the user ar…
- Objective comparison of the emulation ca… proaches

Approach real-world IoT firmware

Measure the forward progress of re-hosting systems



Holistic Re-hosting and Analysis

Analyze the user and kernel level firmware code as a unit

- FirmSolo (published in Usenix 2023 [1]) is the only re-hosting system that supports user and kernel module code re-hosting
- FirmSolo explicitly focuses on IoT kernel module re-hosting
  - Configures and builds custom kernels based on the functionality required by the IoT kernel modules
- The custom kernels might lack functionality required by user level code
  - User level code either prematurely terminates or exhibits unwanted behavior (*e.g., gets stuck in a loop*)
- Not capable of holistic re-hosting!

[1] Angelakopoulos et al. FirmSolo: Enabling dynamic analysis of binary Linux-based IoT kernel modules (USENIX 2023)

Motivating Example

Firmware serial log in FirmSolo

```
1.[ncc_runtimecfg.c: 539 initRunTimeCfg()] ::: \
        Total 4470 nodes, each node 24 bytes.
2.[ncc_runtimecfg.c: 540 initRunTimeCfg()] ::: \
        Total 107280 byte for all nodes.
3.[ncc_runtimecfg.c: 207 loadCfg()] ::: \
        Start loadCfg
4.[ncc_lz77.c: 509 flash2rootfs()] ::: \
        BUG ON!!
5.[ncc_lz77.c: 538 flash2rootfs()] ::: \
        Load Fail(/var/tmp/cfg.txt)
6.[ncc_runtimecfg.c: 213 loadCfg()] ::: \
        flash2rootfs() fail!! restore to
        default!!
7.[ncc_lz77.c: 621 rootfs2flash()] ::: BUG ON!!
8.[ncc_runtimecfg.c: 215 loadCfg()] ::: \
        Del TAG default file!!
9.[ncc_lz77.c: 509 flash2rootfs()] ::: BUG ON!!
```

loadCfg Ghidra snippet

```
1. undefined4 loadCfg(void)
2. {
3.   bool bVar1;
4.   char *__stream;
5.   ...
6.   while( true ) {
7.     (*pcVar7)(__stream);

8. LAB_004a73cc:
9.     iVar3 = \
10.    flash2rootfs("/tmp/cfg.txt");
11.    if (iVar3 != 0) break;
12.    bVar1 = true;
13.    pFVar4 = \
14.      fopen64("/dev/console","a");

15.    if (pFVar4 != (FILE *)0x0) {
16.      uVar2 = getpid();
17. ...
18.      fclose(pFVar4);
19.  }
20. ...
21. }
```

flash2rootfs Ghidra snippet

```
1. undefined4 flash2rootfs(undefined4
                           param_1)
2. {
3.   FILE *pFVar1;
4.   uint uVar2;
5.   pFVar1 = \
       fopen64("/dev/mtdblock4","rb");
6.   if (pFVar1 == (FILE *)0x0) {
7. ...
8.     fputs("BUG ON!!\n",pFVar1);
9.   }
10.  else {
11. ...
12.    if (iVar3 == 0) {
13. ...
14.      fprintf(pFVar1,
           "Load Success(%s)\n", …);
15.      fclose(pFVar1);
16.    }
17.    return 1;
18.  }
19. ...
20.  return 0;
21. }
```

IoT kernel modules do not make use of this functionality!

FirmSolo **does not** include the *CONFIG_MTD_CHAR* option in its custom kernel!

Re-hosting Frameworks Comparison

Compare the different re-hosting approaches based on their emulation capabilities

Prior work:

- Wallclock time + ad-hoc metrics
  - Number of bugs found or networking connectivity achieved
- Each re-hosting framework adopts a different design
  - Current metrics too generic to include these differencies between the frameworks

FirmSolo timestamps

```
4.21: /etc/rc.d/S05boot boot
8.14: /bin/mkdir -p /var/lock
34.28: /sbin/udevtrigger
125.02: /usr/bin/dsd
```

FirmAE timestamps

```
4.26: /etc/rc.d/S05boot boot
5.81: /bin/mkdir -p /var/lock
12.84: /sbin/udevtrigger
37.64: /usr/bin/dsd
```

Firmware Initialization Completion Detection (FICD)

Firmware executes tasks during bootup

- **Task:** *Program name + arguments*

After the initialization phase few or no new tasks are executed

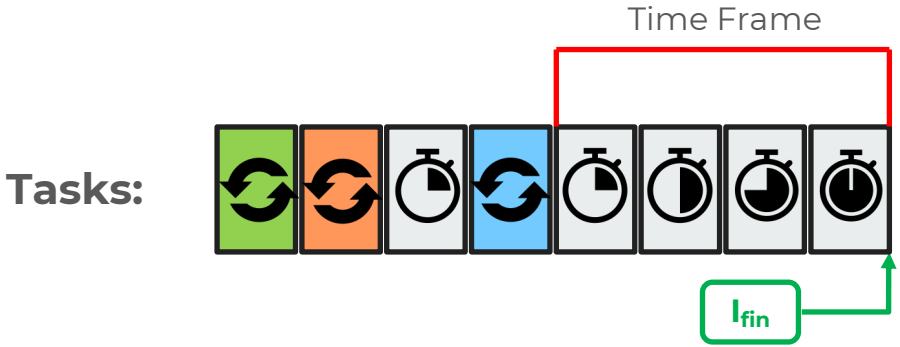
Use a *time frame* to allow for new tasks to be executed

- Mark the emulation point at the end of the time frame as the initialization completion (I_{fin}) point

Conceptually the same for all re-hosting frameworks

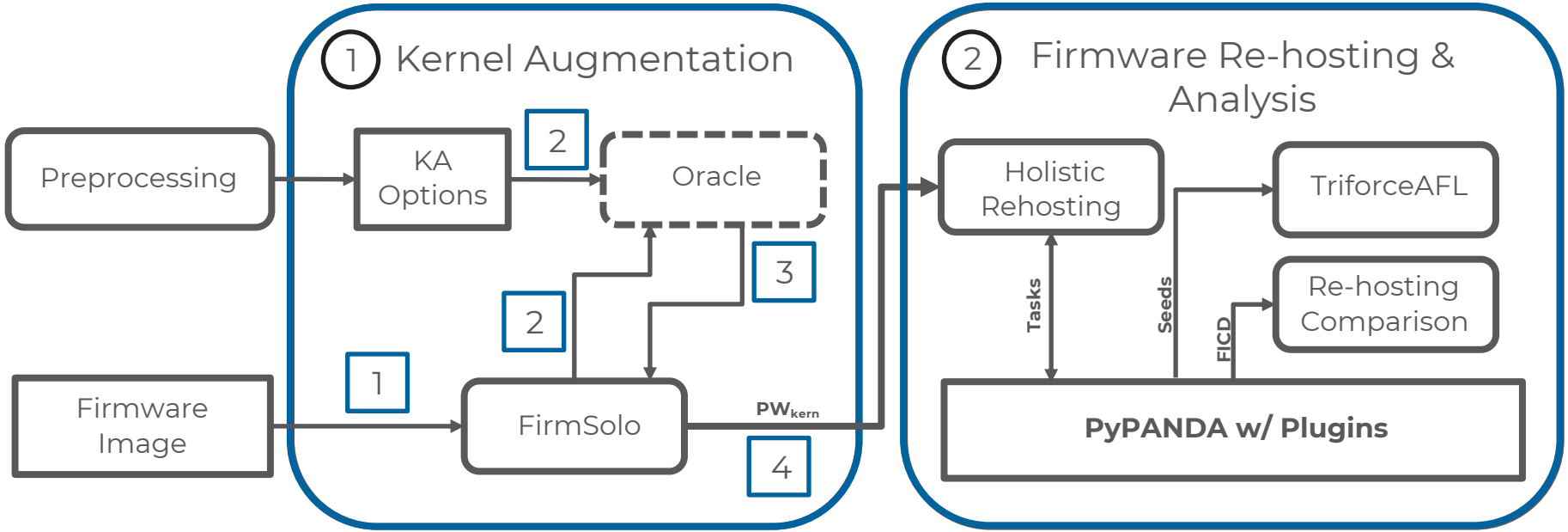
Collect the emulation-based metrics at the I_{fin} point

- E.g., user and kernel (i.e., kernel module) code coverage



Pandawan

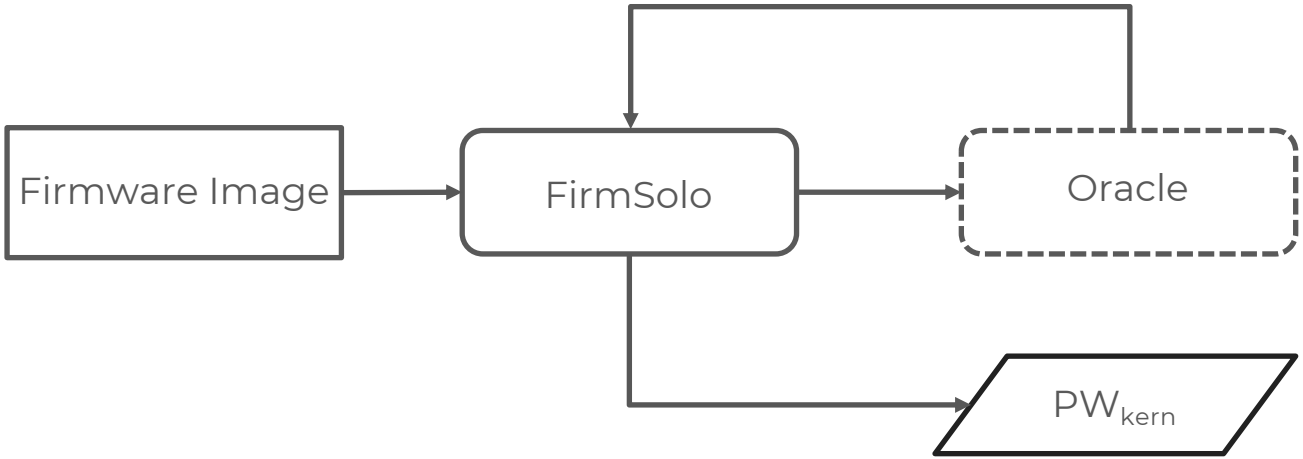
A framework to holistically re-host and analyze Linux-based IoT firmware
Also implements FICD to enable the comparison of different full-system firmware re-hosting approaches



Kernel Augmentation

Use FirmSolo to analyze the firmware image and produce a custom kernel

- Use Oracle to augment the custom kernel with functionality required by user level code



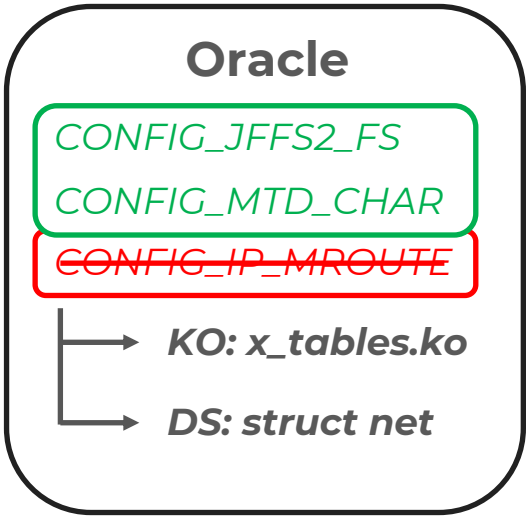
Oracle

Provide additional functionality required by user level code

Add popular configuration options used in IoT kernels in Pandawan's kernels (PW_{kern})

- Gathered during *Preprocessing*

Make sure these options do not affect the layout of data structures used by the IoT kernel modules



Firmware Re-hosting & Analysis

Re-host the PW_{kern} and the F_{fs} under PyPANDA [2]

Use the PyPANDA script

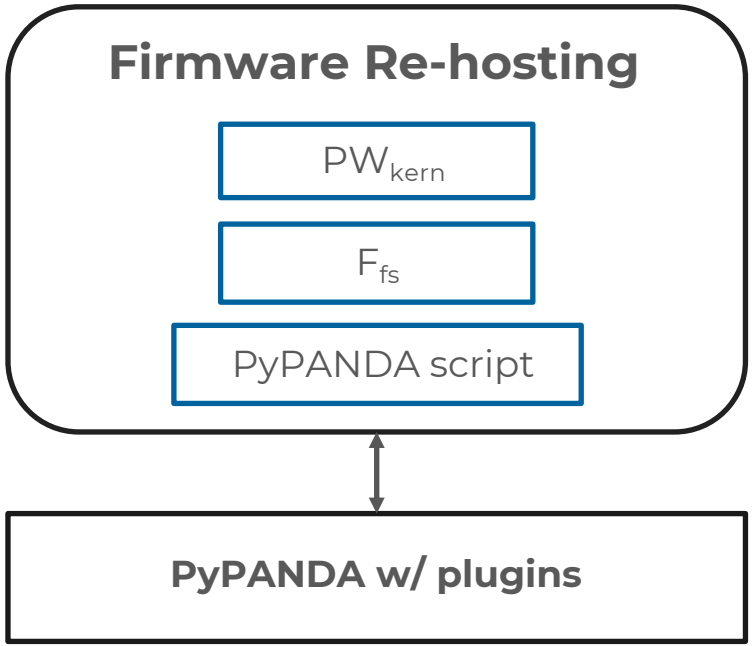
- Make use of FirmAE's networking configuration logic

Use PANDA's builtin plugins

- *coverage* and *syscalls_logger*

Use custom plugins

- *FICD and SyscallToKmodTracer*



[2] Craig et al. PyPANDA: Taming the PANDAmonium of Whole System Dynamic Analysis (BAR 2021)

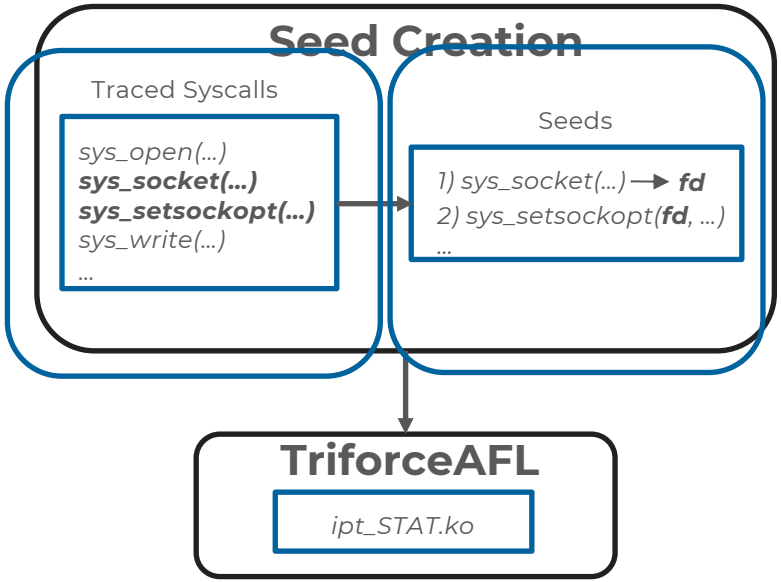
Firmware Re-hosting & Analysis (cont.)

Compare the emulation capabilities of re-hosting frameworks with FICD

- Comparison of full-system re-hosting frameworks
- Use emulation-based metrics:
  - Number of programs executed,
  - Number of kernel modules loaded,
  - User and kernel module code coverage

Holistically analyze the firmware code

- Traces system calls that access the IoT kernel modules (through SyscallToKmodTracer)
- Covert the traces to seeds for TriforceAFL
  - Similar to Moonshine [3]
- Fuzz the kernel modules with TriforceAFL



[3] Pailoor et al. MoonShine: Optimizing OS Fuzzer Seed Selection with Trace Distillation (USENIX 2018)

Evaluation

Dataset:

- **1,520** firmware images (1,470 from FirmSolo and 50 from Greenhouse)

Re-hosting Comparison:

- Compared Firmadyne [4], FirmAE [5], FirmSolo and Pandawan, via FICD
- Up to **6%** more user programs executed and **9%** kernel modules loaded
- Up to **21%** more user code BBs and **26%** more kernel module code BBs executed

Holistic Analysis:

- Seeds for **479** firmware images
- TriforceAFL triggered **16** bugs (6 unknown) in 12 kernel modules (8 closed source)
  - We reported the findings to the respective vendors

Framework	Firmadyne	FirmAE	FirmSolo	Pandawan
No KALLSYMS				
Avg. Progs	35	36	34	36
Avg. TBs	15,715	16,552	13,835	16,740
KOs Loaded	0	0	4,936	5,146
Avg. KOs TBs	0	0	200	251

Table 1: Re-hosting comparison experiments using FICD and the emulation-based metrics. The green cells represent the best values for each metric.

[4] Chen et al. Towards Automated Dynamic Analysis for Linux-based Embedded Firmware (NDSS 2016)
[5] Kim et al. FirmAE: Towards Large-Scale Emulation of IoT Firmware for Dynamic Analysis (ACSAC 2020)

Evaluation

Dataset:

- **1,520** firmware images (1,470 from FirmSolo and 50 from Greenhouse)

Re-hosting Comparison:

- Compared Firmadyne [4], FirmAE [5], FirmSolo and Pandawan, via FICD
- Up to **6%** more user programs executed and **9%** kernel modules loaded
- Up to **21%** more user code BBs and **26%** more kernel module code BBs executed

Holistic Analysis:

- Seeds for **479** firmware images
- TriforceAFL triggered **16** bugs (6 unknown) in 12 kernel modules (8 closed source)
  - We reported the findings to the respective vendors

Module	Type	Vendor	Kernel	Bugs
MIPS				
arp_tables	O	AT&T	2.6.31	2
led	P	Linksys	2.6.31	1
ipt_STAT	P	TP-Link	2.6.36	1
x_tables	O	TP-Link	2.6.31	1
statistics	P	TP-Link	2.6.31	1
ip6_tables	O	AT&T	2.6.30.10	2
ipv6_spi	P	Netgear	2.6.30	2
ip_tables	O	TP-Link	2.6.31	2
gpio	P	DLink	2.6.31	1
gpio_module	P	DLink	2.6.31	1
ARM				
ipt_STAT	P	TP-Link	2.6.32.11	1
statistics	P	TP-Link	2.6.36.4	1
			Total	**16**

Table 2: Fuzzing experiments with TriforceAFL. The O and P notations on column two stand for Open-source and Proprietary, respectively.

[4] Chen et al. Towards Automated Dynamic Analysis for Linux-based Embedded Firmware (NDSS 2016)
[5] Kim et al. FirmAE: Towards Large-Scale Emulation of IoT Firmware for Dynamic Analysis (ACSAC 2020)

Summary

Pandawan is a framework that enables the holistic re-hosting and analysis of Linux-based IoT firmware

FICD enables the comparison of different re-hosting frameworks based on their emulation capabilities

Contact me: jaggel@bu.edu

Thank You!

Source Code