CRYPTACUS Workshop
Nijmegen, 16 - 18 November 2017

Avatar² - Enhancing Binary Firmware Security Analysis with Dynamic Multi-Target Orchestration

Marius Muench <marius.muench@eurecom.fr>

EURECOM
Sophia Antipolis

PART I

Avatar² - Enhancing Binary Firmware Security Analysis with Dynamic Multi-Target Orchestration

Enhancing Binary Firmware Security Analysis

Dynamic Binary Firmware Security Analysis?

- **Majority of nowadays vulnerabilities are “low-hanging fruits”**
- **Often 3rd party analysis**
- **Lack of sophisticated tooling**

(Some) Challenges in Dynamic Binary Firmware Analysis

- **Intransparency**
- **Performance & Scalability**
- **Instrumentation capabilities**

Avatar² - Enhancing Binary Firmware Security Analysis with Dynamic Multi-Target Orchestration

Avatar² -

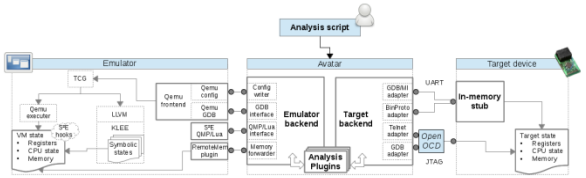
Dynamic Multi-Target Orchestration

Avatar²

- **Developed by:**
  - Marius Muench
  - Dario Nisi
  - Aurélien Francillon
  - Davide Balzarotti

- **Open source:**
  - https://github.com/avatartwo/avatar2

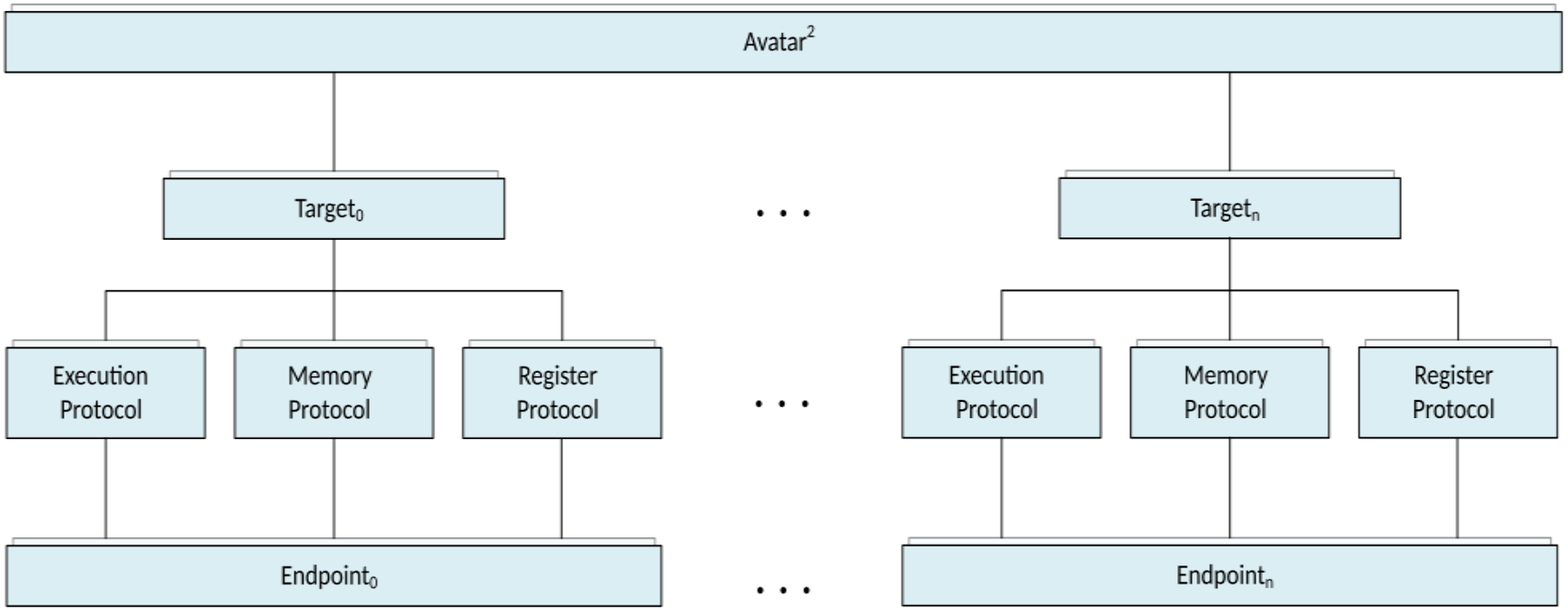
- **Re-designed and re-implemented from scratch**



[1]

[1] Zaddach, J., Bruno, L., Francillon, A., and Balzarotti, D.: “AVATAR: A Framework to Support Dynamic Security Analysis of Embedded Systems' Firmwares”. *NDSS* 14

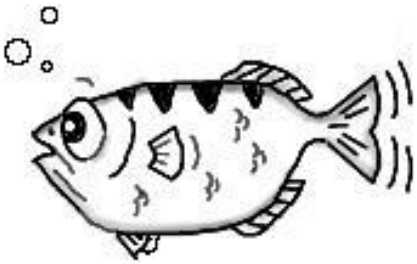
The general picture



Core Concepts

- **Target Orchestration**
- **Separation of Execution and Memory**
- **State Transfer and Synchronization**

Supported Targets



*

* Not yet available

Avatar² - Example Script

```python
from os import getcwd

from avatar2 import *

def main(trace_name):

    sample = "%s/../binaries/expat_panda.bin" % getcwd()
    openocd_conf = '%s/../configs/nucleo-l152re.cfg' % getcwd()
    panda_path = '%s/../avatar2/targets/build/panda/arm-softmmu/qemu-system-arm'

    avatar = Avatar(output_directory='/tmp/myavatar', arch=ARM_CORTEX_M3)

    panda = avatar.add_target(PandaTarget,
                              gdb_executable="arm-none-eabi-gdb",
                              executable=panda_path)

    nucleo = avatar.add_target(OpenOCDTarget,
                               gdb_executable="arm-none-eabi-gdb",
                               openocd_script=openocd_conf)

    panda.gdb_port = 1234
    nucleo.gdb_port = 1235

    avatar.add_memory_range(0x40000000, 0x1000000, 'mmio',
                                    forwarded=True, forwarded_to=nucleo)

    avatar.add_memory_range(0x08000000, 0x1000000, 'rom', file=sample)
    avatar.add_memory_range(0x20000000, 0x14000, 'ram')

    avatar.init()

    nucleo.set_breakpoint(0x0800b5ac)
    nucleo.cont()
    nucleo.wait()

    print("Syncing State")
    avatar.transfer_state(nucleo, panda, synced_ranges=[ram])
    panda.write_memory(0x20000c20, 4, 1)

    print("Start recording the execution")
    panda.begin_record(trace_name)

    import IPython; IPython.embed()

    avatar.shutdown()
```

Phase #0: Preambel

```
1  from os import getcwd
2
3  from avatar2 import *
4
5  def main(trace_name):
6
7      sample = "%s/../binaries/expat_panda.bin" % getcwd()
8      openocd_conf = '%s/../configs/nucleo-l152re.cfg' % getcwd()
9      panda_path = '%s/../avatar2/targets/build/panda/arm-softmmu/qemu-system-arm'
10
11     avatar = Avatar(output_directory='/tmp/myavatar', arch=ARM_CORTEX_M3)
```

Phase #1: Target Definition

```
13    panda = avatar.add_target(PandaTarget,
14                              gdb_executable="arm-none-eabi-gdb",
15                              executable=panda_path)
16
17    nucleo = avatar.add_target(OpenOCDTarget,
18                                gdb_executable="arm-none-eabi-gdb",
19                                openocd_script=openocd_conf)
20
21    panda.gdb_port = 1234
22    nucleo.gdb_port = 1235
```

Phase #2: Memory Layout Definition

```
avatar.add_memory_range(0x40000000, 0x1000000, 'mmio',
                        forwarded=True, forwarded_to=nucleo)

avatar.add_memory_range(0x08000000, 0x1000000, 'rom', file=sample)
avatar.add_memory_range(0x20000000, 0x14000, 'ram')
```

Phase #3: Orchestration!

```
30    avatar.init()
31
32    nucleo.set_breakpoint(0x0800b5ac)
33    nucleo.cont()
34    nucleo.wait()
35
36    print("Syncing State")
37    avatar.transfer_state(nucleo, panda, synced_ranges=[ram])
38    panda.write_memory(0x20000c20, 4, 1)
39
40    print("Start recording the execution")
41    panda.begin_record(trace_name)
42
43    import IPython; IPython.embed()
44
45    avatar.shutdown()
```

A note on peripherals

- **Main source of complication for emulation**

- **Avatar^2 offers different strategies:**
  - Full emulation
  - Partial emulation using peripheral forwarding
  - Partial emulation using python abstractions

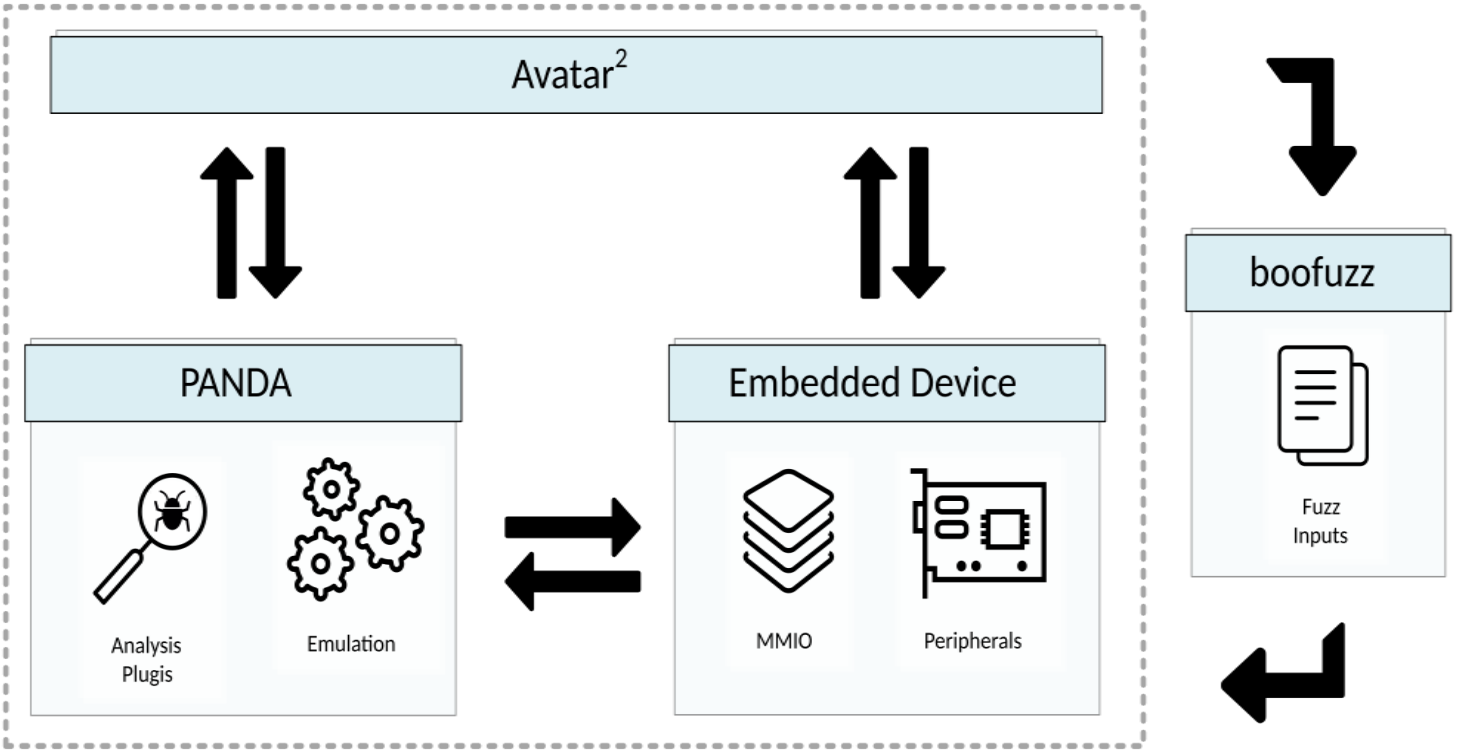
PART II
(WYCINWYC)

WYCINWYC - Overview

- **Acronym for “What You Corrupt Is Not What You Crash” [2]**
- **Joint Work with Siemens**
- **Utilizes Avatar^2 to improve fuzz testing on embedded systems**

[2] Muench, M., Stijohann, J., Kargl, F., Francillon, A. and Balzarotti, D.: “What You Corrupt Is Not What You Crash: Challenges in Fuzzing Embedded Devices”. To appear at NDSS 2018

WYCINWYC - Setup

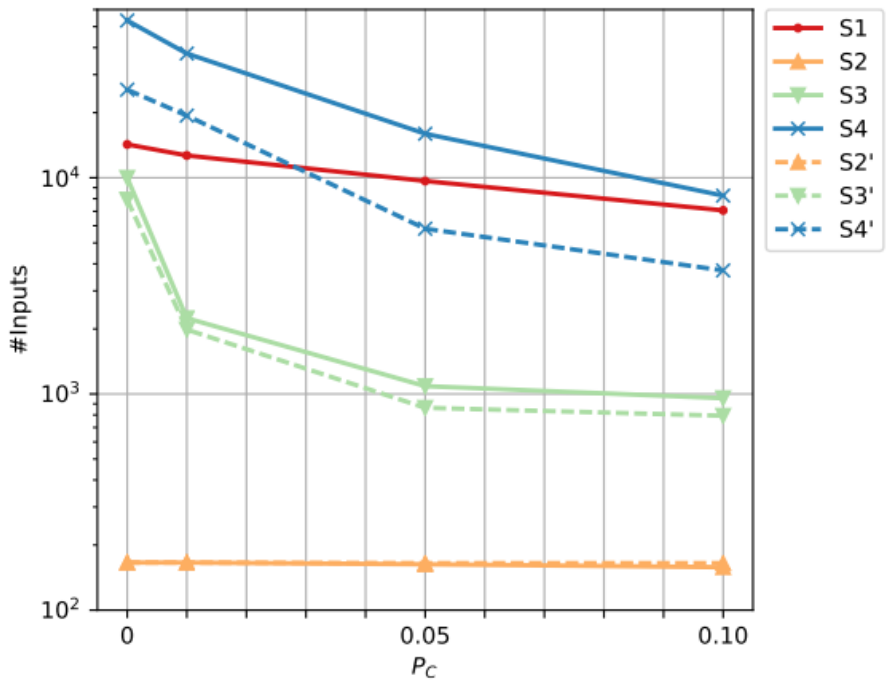


WYCINWYC - Analysis Plugins

Analysis Plugin	Format String	Stack-based Bof	Heap-based Bof	Double Free	Null Pointer Deref.
a) Call Stack Tracking	✗	✓	✗	✗	✗
b) Call Frame Tracking	✗	✓	✗	✗	✗
c) Stack Object Tracking	✗	✓	✗	✗	✗
d) Segment Tracking	✓	✓	✗	✓	✓
e) Format Specifier Tracking	✓	✗	✗	✗	✗
f) Heap Object Tracking	✗	✗	✓	✓	✓
Combined	✓	✓	✓	✓	✓

[2] Muench, Marius et al. “What You Corrupt Is Not What You Crash: Challenges in Fuzzing Embedded Devices”. To appear at NDSS 2018

WYCINWYC - Results



S1: Native

S2: Partial Emulation (Peripheral Forwarding)

S3: Partial Emulation (Avatar Peripheral)

S4: Full Emulation

Related Tools

- **AVATAR ;)**

- **Firmadyne**

  Chen, D. D., Woo, M., Brumley, D., & Egele, M.: “Towards Automated Dynamic Analysis for Linux-based Embedded Firmware”. *NDSS 2016*

- **Luaqemu**

  https://github.com/Comsecuris/luaqemu

- **PROSPECT**

  Kammerstetter, M, Platzer, C., & Kastner, W.: “Prospect: peripheral proxying supported embedded code testing.” *ASIA CCS 2014*

Conclusion

- **Appropriate tooling is important**
- **… so are good emulators**
- **Until then, avatar^2 might be helpful**
- **We are just at the beginning…**

Questions?

https://github.com/avatartwo/avatar2