Dynamic Binary Firmware Analysis With Avatar²

THCon 2023

Paul OLIVIER

2023-04-20

EURECOM Sophia Antipolis

THCON TOULOUSE HACKING CONVENTION

LAAS CNRS

> whoami

Paul OLIVIER

- Recent Ph.D. graduate (EURECOM)
- Just joined as postdoc @ LAAS-CNRS
- Dynamic analysis for embedded system security
- Part of the maintainer team of avatar²

> Content

- Motivation
- Rehosting Firmware
- Avatar^2: A Multi-Target Orchestration Platform
- Framework Overview
- Conclusion

> Introduction

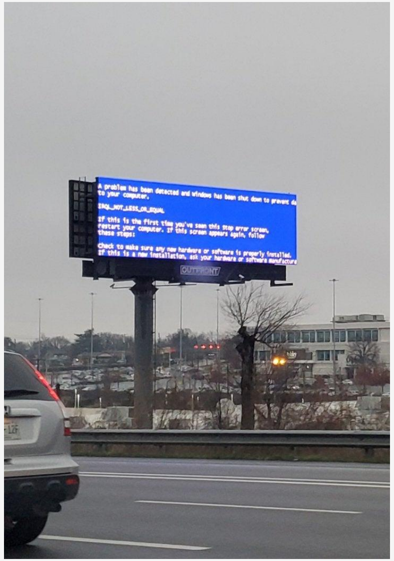
Prevalence of bugs in the Wild

```
A problem has been detected and windows has been shut down to prevent d
to your computer.

IRQL_NOT_LESS_OR_EQUAL

If this is the first time you've seen this stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacture
```

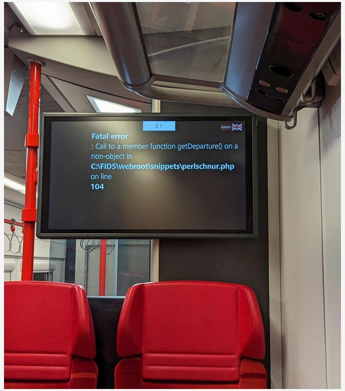
```
[  OK  ] Stopped target Basic System.
[  OK  ] Stopped target Sockets.
[  OK  ] Stopped target Slices.
[  OK  ] Stopped target Paths.
[  OK  ] Stopped target Remote File Systems.
[  OK  ] Stopped target System Initialization.
[  OK  ] Stopped target Loca■Encrypted Volumes.
         Stopping udev Kernel Device Manager...
[  OK  ] Stopped Apply Kernel Variables.
[  OK  ] Stopped target Swap.
         Starting Plymouth s■ch root service...
[  OK  ] Stopped target Remo■ile Systems (Pre).
[  OK  ] Stopped dracut initqueue hook.
[  OK  ] Stopped udev Coldplug all Devices.
[  OK  ] Stopped Create Vola■s and Directories.
[  OK  ] Stopped target Local File Systems.
[  OK  ] Started Setup Virtual Console.
[  OK  ] Started Cleaning Up■ting Down Daemons.
[  OK  ] Stopped udev Kernel Device Manager.
         Stopping Hardware R■Gatherer Daemon...
[  OK  ] Stopped Create Stat■ice Nodes in /dev.
[  OK  ] Stopped Create list■he current kernel.
[  OK  ] Stopped dracut pre-udev hook.
[  OK  ] Stopped dracut cmdline hook.
[  OK  ] Closed udev Kernel Socket.
[  OK  ] Closed udev Control Socket.
         Starting Cleanup udevd DB...
[  OK  ] Stopped Hardware RN■y Gatherer Daemon.
[  OK  ] Started Plymouth switch root service.
[  OK  ] Started Cleanup udevd DB.
[  OK  ] Reached target Switch Root.
         Starting Switch Root...
[FAILED] Failed to start Switch Root.
See 'systemctl status i■t.service' for details.

Generating "/run/initramfs/rdsosreport.txt"


Entering emergency mode. Ex   he shell to conti
nue.
Type "journalctl" to view s   em logs
You might want to sa
```

> Introduction

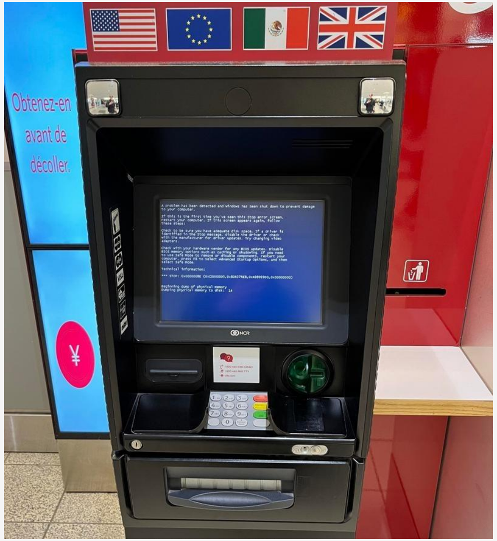
Prevalence of bugs in the Wild



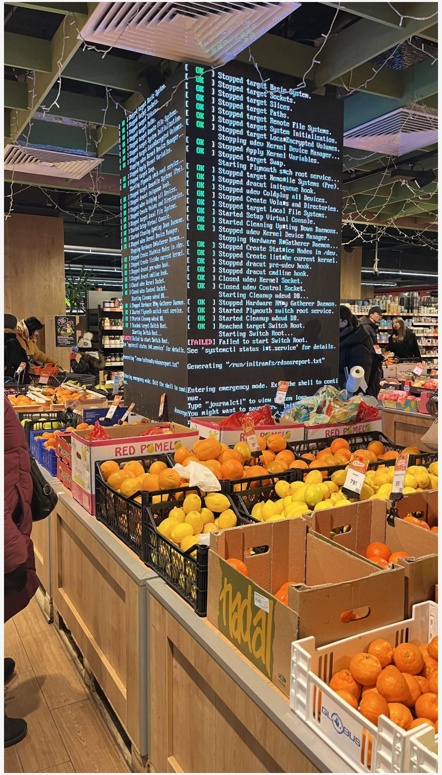
```
A problem has been detected and windows has been shut down to prevent da
to your computer.

IRQL_NOT_LESS_OR_EQUAL

If this is the first time you've seen this stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacture
```



```
S1
Fatal error
: Call to a member function getDeparture() on a
non-object in
C:\FID5\webroot\snippets\perlschnur.php
on line
104
```



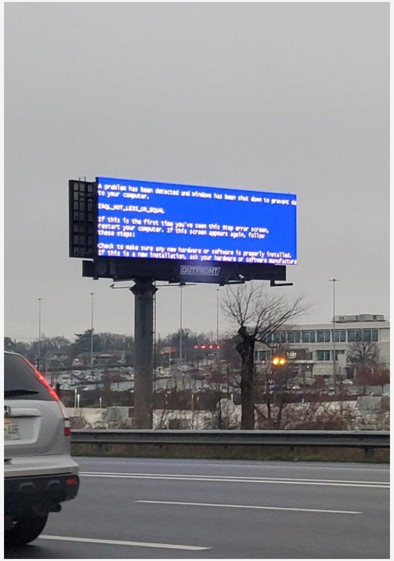
```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to be sure you have adequate disk space. If a driver is
identified in the Stop message, disable the driver or check
with the manufacturer for driver updates. Try changing video
adapters.

Check with your hardware vendor for any BIOS updates. Disable
BIOS memory options such as caching or shadowing. If you need
to use Safe Mode to remove or disable components, restart your
computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.

Technical information:

*** STOP: 0x0000008E (0xC0000005,0x806376E8,0xA9893900,0x00000000)


Beginning dump of physical memory
Dumping physical memory to disk:  14
```



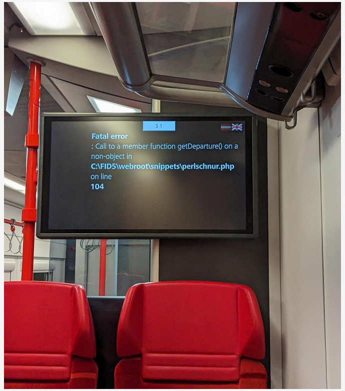
```
[  OK  ] Stopped target Basic System.
[  OK  ] Stopped target Sockets.
[  OK  ] Stopped target Slices.
[  OK  ] Stopped target Paths.
[  OK  ] Stopped target Remote File Systems.
[  OK  ] Stopped target System Initialization.
         Stopped target Loca■Encrypted Volumes.
         Stopping udev Kernel Device Manager...
[  OK  ] Stopped Apply Kernel Variables.
[  OK  ] Stopped target Swap.
         Starting Plymouth s■ch root service...
[  OK  ] Stopped target Remo■ile Systems (Pre).
[  OK  ] Stopped dracut initqueue hook.
[  OK  ] Stopped udev Coldplug all Devices.
[  OK  ] Stopped Create Vola■s and Directories.
[  OK  ] Stopped target Local File Systems.
[  OK  ] Started Setup Virtual Console.
[  OK  ] Started Cleaning Up■ting Down Daemons.
[  OK  ] Stopped udev Kernel Device Manager.
         Stopping Hardware R■Gatherer Daemon...
[  OK  ] Stopped Create Stat■ice Nodes in /dev.
[  OK  ] Stopped Create list■he current kernel.
[  OK  ] Stopped dracut pre-udev hook.
[  OK  ] Stopped dracut cmdline hook.
[  OK  ] Closed udev Kernel Socket.
[  OK  ] Closed udev Control Socket.
         Starting Cleanup udevd DB...
[  OK  ] Stopped Hardware RN■y Gatherer Daemon.
[  OK  ] Started Plymouth switch root service.
[  OK  ] Started Cleanup udevd DB.
[  OK  ] Reached target Switch Root.
         Starting Switch Root...
[FAILED] Failed to start Switch Root.
See 'systemctl status i■t.service' for details.

Generating "/run/initramfs/rdsosreport.txt"


Entering emergency mode. Exi■the shell to conti
nue.
Type "journalctl" to view s■■em log
You might want to sav■
```

> Introduction

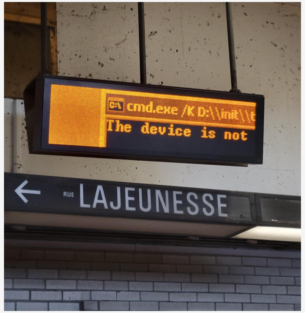
Prevalence of bugs in the Wild



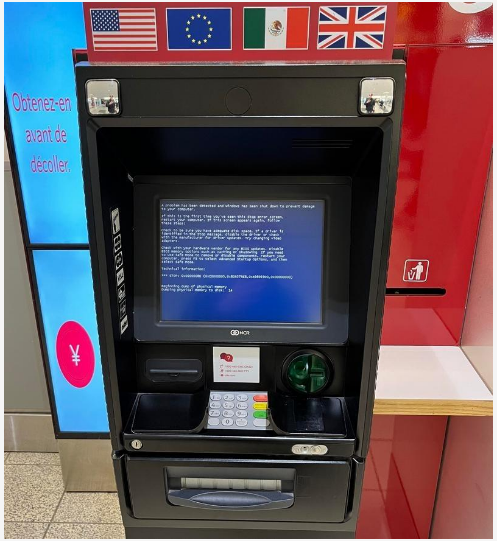
```
A problem has been detected and windows has been shut down to prevent da
to your computer.

IRQL_NOT_LESS_OR_EQUAL

If this is the first time you've seen this stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacture
```



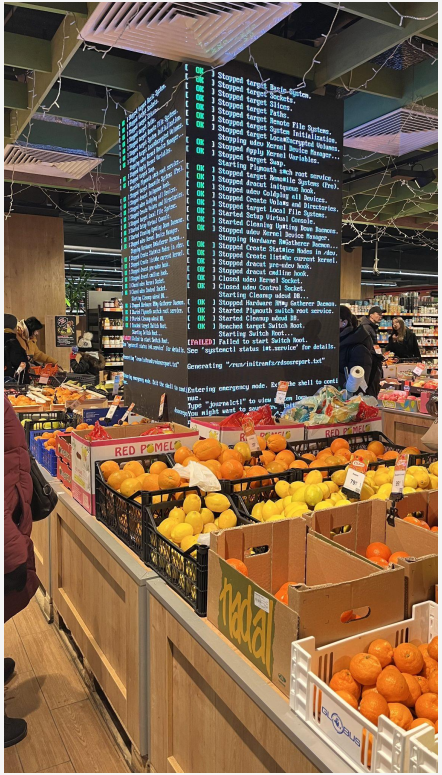
```
Fatal error
: Call to a member function getDeparture() on a
non-object in
C:\FID5\webroot\snippets\perlschnur.php
on line
104
```



```
cmd.exe /K D:\\init\\
The device is not
```



```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to be sure you have adequate disk space. If a driver is
identified in the Stop message, disable the driver or check
with the manufacturer for driver updates. Try changing video
adapters.

Check with your hardware vendor for any BIOS updates. Disable
BIOS memory options such as caching or shadowing. If you need
to use Safe Mode to remove or disable components, restart your
computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.

Technical information:

*** STOP: 0x0000008E (0xC0000005,0x80637688,0xA9893900,0x00000000)


Beginning dump of physical memory
Dumping physical memory to disk:  14
```



```
[  OK  ] Stopped target Basic System.
[  OK  ] Stopped target Sockets.
[  OK  ] Stopped target Slices.
[  OK  ] Stopped target Paths.
[  OK  ] Stopped target Remote File Systems.
[  OK  ] Stopped target System Initialization.
[  OK  ] Stopped target Loca■Encrypted Volumes.
         Stopping udev Kernel Device Manager...
[  OK  ] Stopped Apply Kernel Variables.
[  OK  ] Stopped target Swap.
         Starting Plymouth s■ch root service...
[  OK  ] Stopped target Remo■ile Systems (Pre).
[  OK  ] Stopped dracut initqueue hook.
[  OK  ] Stopped udev Coldplug all Devices.
[  OK  ] Stopped Create Vola■s and Directories.
[  OK  ] Stopped target Local File Systems.
[  OK  ] Started Setup Virtual Console.
[  OK  ] Started Cleaning Up■ting Down Daemons.
[  OK  ] Stopped udev Kernel Device Manager.
         Stopping Hardware R■Gatherer Daemon...
[  OK  ] Stopped Create Stat■ice Nodes in /dev.
[  OK  ] Stopped Create list■he current kernel.
[  OK  ] Stopped dracut pre-udev hook.
[  OK  ] Stopped dracut cmdline hook.
[  OK  ] Closed udev Kernel Socket.
[  OK  ] Closed udev Control Socket.
         Starting Cleanup udevd DB...
[  OK  ] Stopped Hardware RN■y Gatherer Daemon.
[  OK  ] Started Plymouth switch root service.
[  OK  ] Started Cleanup udevd DB.
[  OK  ] Reached target Switch Root.
         Starting Switch Root...
[FAILED] Failed to start Switch Root.
See 'systemctl status i■t.service' for details.

Generating "/run/initramfs/rdsosreport.txt"


Entering emergency mode. Ex■the shell to conti
nue.
Type "journalctl" to view s■■em logs
Type
You might want to sav
```

> Introduction

- **Severity** and **impact** of software bugs
  - Vulnerabilities: *unauthorized access, information leak, denial of service, ransomware*
  - Cost: *finding & fixing, system downtime*
  - Human life: *car driving assistance, Boeing 737 MAX, radiology, etc.*

> Introduction

- **Severity** and **impact** of software bugs
    - Vulnerabilities: *unauthorized access, information leak, denial of service, ransomware*
    - Cost: *finding & fixing, system downtime*
    - Human life: *car driving assistance, Boeing 737 MAX, radiology*

- Thorough **testing** of firmware is crucial to guarantee its safety and **security**

> Introduction

- **Severity** and **impact** of software bugs
  - Vulnerabilities: *unauthorized access, information leak, denial of service, ransomware*
  - Cost: *finding & fixing, system downtime*
  - Human life: *car driving assistance, Boeing 737 MAX, radiology*

- Thorough **testing** of firmware is crucial to guarantee its safety and **security**

- **Static** and **dynamic** analysis are two main approaches.

> Motivation: Firmware Analysis

Static analysis

- Examine without executing code

> Motivation: Firmware Analysis

Static analysis

- Examine without executing code
- Limitations
    - Achieve **larger coverage**… but **less precise** (no execution context)
    - No need to run code… but does not require external systems

> Motivation: Firmware Analysis

- **Dynamic** analysis techniques are plenty & powerful
  - more precise… but smaller coverage
  - *tracing, profiling, fuzzing, concolic execution, sanitizers, data taint tracking, record-replay, interactive debugging, etc.*

KLEE
PANDA
Manticore
systemtap
FRIDA
DYNAMIC INSTRUMENTATION TOOLKIT
strace
UNICORN
ENGINE
QEMU
NYX
LTTng
avatar2
MULTI TARGET ORCHESTRATION
angr
AFL++
TRITON
Dynamic Binary Analysis

> Motivation: Firmware Analysis

- … but require to **setup** the environment

> Motivation: Firmware Analysis

- … but require to **setup** the environment
- Not always feasible to **run** them **on** the physical device:
    - **Constrained** environment *(computing power, memory size, network bandwidth)*

> Motivation: Firmware Analysis

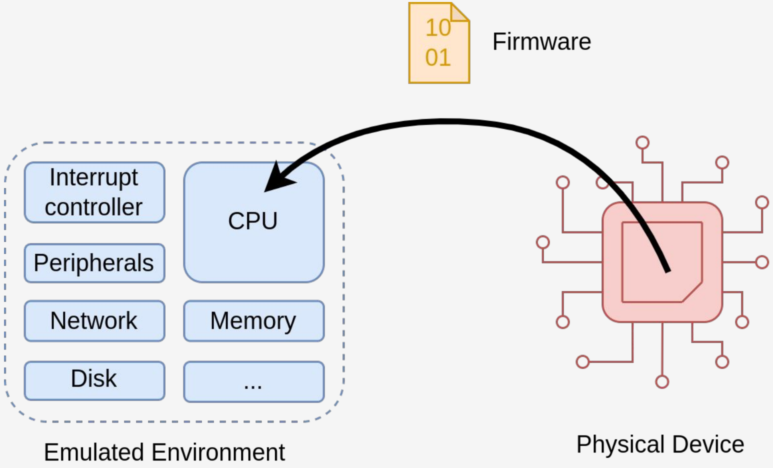
- … but require to **setup** the environment
- Not always feasible to **run** them **on** the physical device:
  - **Constrained** environment *(computing power, memory size, network bandwidth)*
  - **Insufficient** ability to **control** & **observe** code execution

> Motivation: Emulation & Rehosting

- Alternative: **emulation**

> Motivation: Emulation & Rehosting

- Alternative: **emulation**
- **Rehosting**:



The process of moving the firmware from its original "host" into a virtualized environment that reproduce the original well enough for its execution and analysis

Fasano, Andrew, et al. *SoK: Enabling security analyses of embedded systems via rehosting*, Asia CCS 2021

> Motivation: Rehosting Challenges

- **Challenges** to run a firmware in an emulator

  1. *Acquisition*:
     - Protected memory, disable debug interface, anti-tampering sensors
     - Encryption, obfuscation, proprietary format

Fasano, Andrew, et al. *SoK: Enabling security analyses of embedded systems via rehosting*, Asia CCS 2021

> Motivation: Rehosting Challenges

- **Challenges** to run a firmware in an emulator

1. *Acquisition*:
    - Protected memory, disable debug interface, anti-tampering sensors
    - Encryption, obfuscation, proprietary format

2. *Execution*:
    - Understand the Instruction Set Architecture (ARM, MIPS, m68k, Blackfin, Xtensa, etc.)
    - Design to run on a specific hardware (peripherals)

Fasano, Andrew, et al. *SoK: Enabling security analyses of embedded systems via rehosting*, Asia CCS 2021

> Problem Statement

- Various techniques
  - *emulation,*
  - *record-replay,*
  - *symbolic execution,*
  - *hardware-in-the-loop,*
  - *hybrid*

QEMU

UNICORN
ENGINE

> Problem Statement

- Various techniques
  - *emulation,*
  - *record-replay,*
  - *symbolic execution,*
  - *hardware-in-the-loop,*
  - *hybrid*

QEMU

UNICORN
ENGINE

- How to **combine tools** to leverage their strengths and tackle complex problems?

> avatar^2

- Facilitate **interoperability** between Dynamic Binary Analysis techniques and tools
- Provide **abstractions** of debuggers, emulators and other frameworks
- Open source https://github.com/avatartwo/avatar2

avatar^2

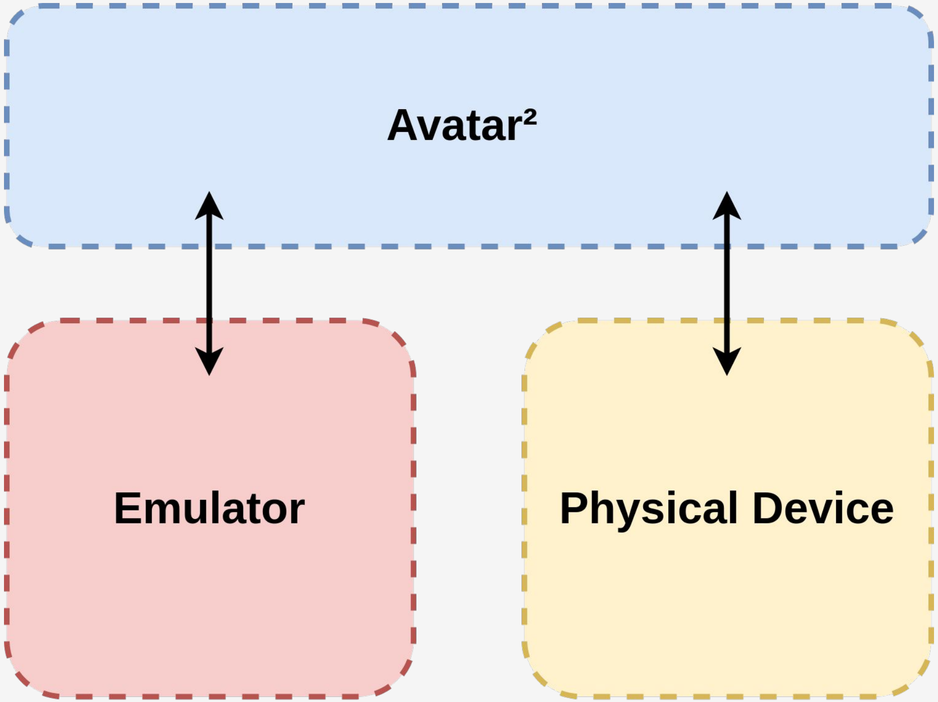
MULTI TARGET ORCHESTRATION

> avatar² Features

- Scriptable (Python based)
- Multiple architecture (ARM, MIPS, x86)
- Target orchestration
    - State transfer & Synchronization
    - Forward memory & I/O accesses
    - Model peripheral

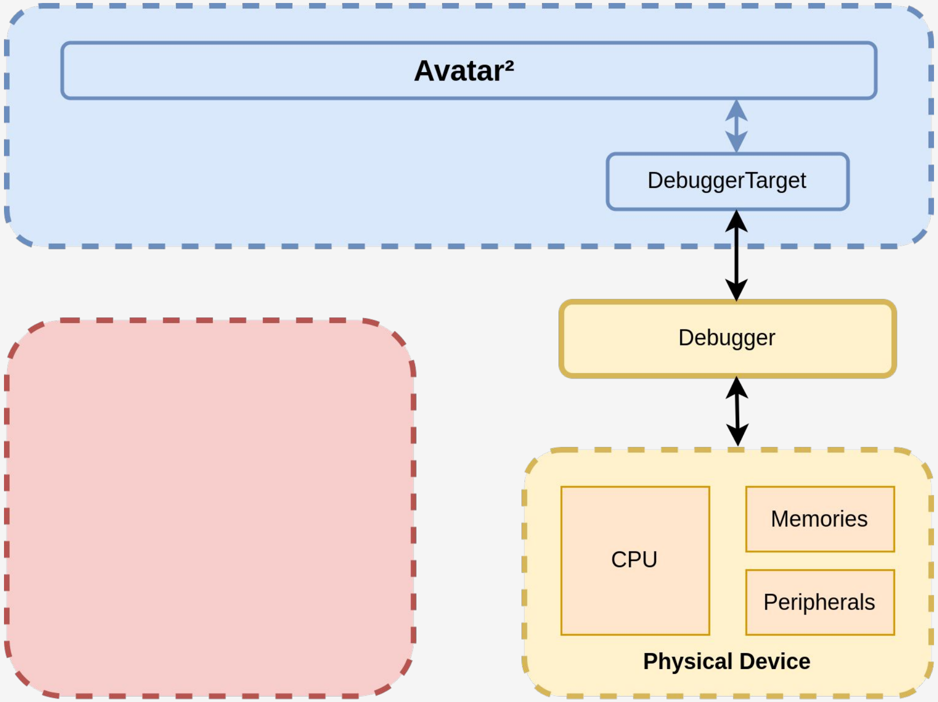
> avatar² Overview

- Orchestration



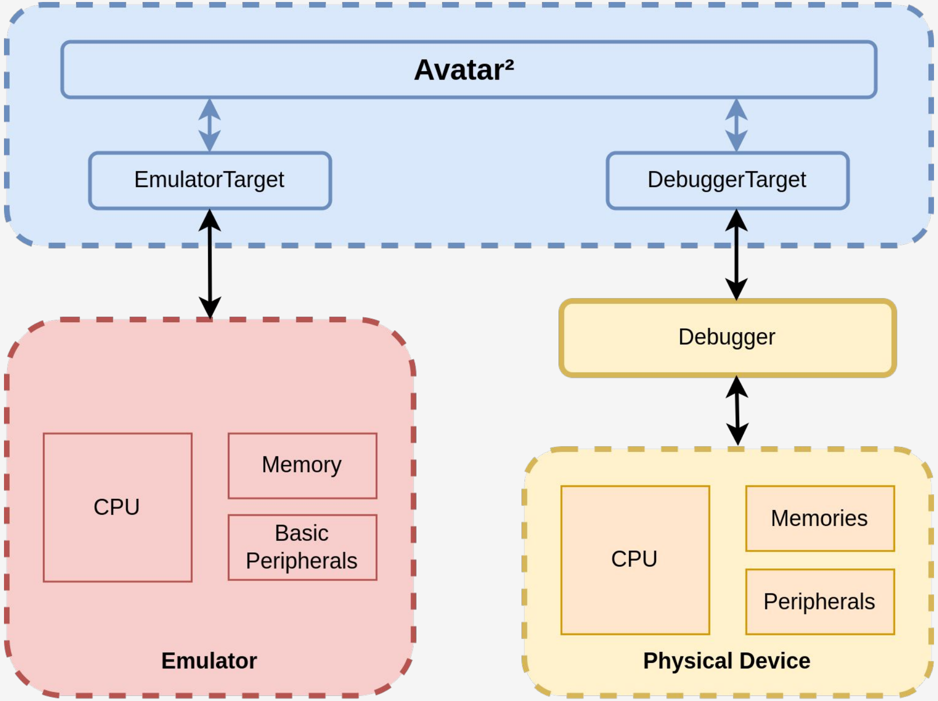
> avatar² Overview

- Physical device



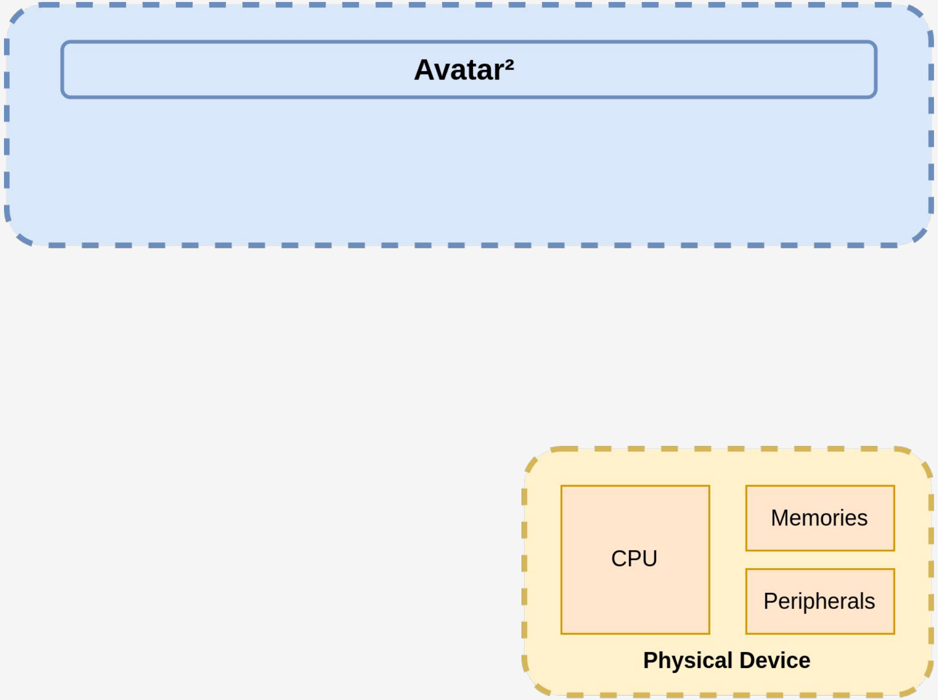
> avatar² Overview

- Emulator



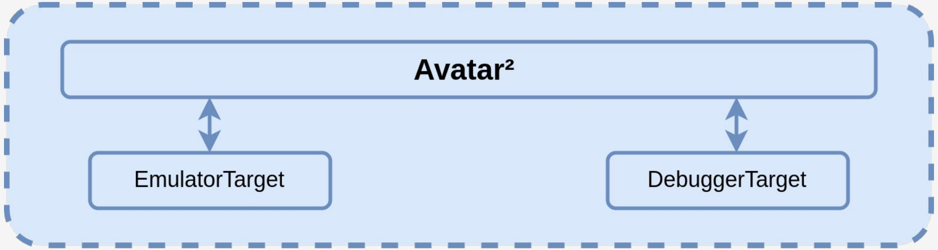
> Initialization

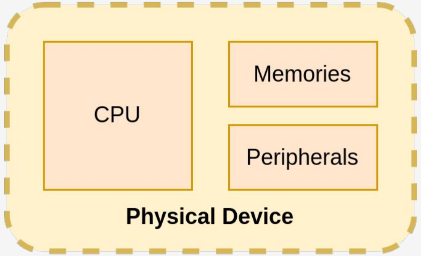
```
# Init
avatar = Avatar(arch=ARM_CORTEX_M3)
```



> Initialization

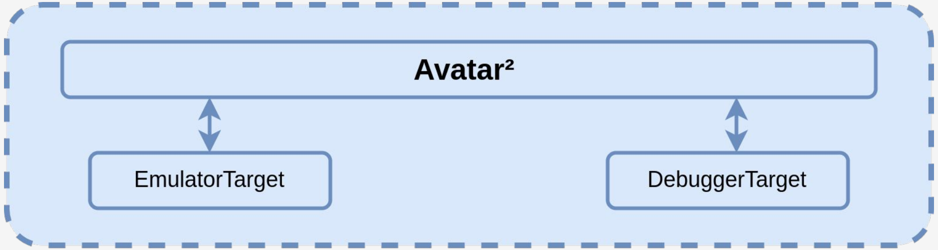
```
# Init
avatar = Avatar(arch=ARM_CORTEX_M3)

device = avatar.add_target(OpenOCDTarget)
emulator = avatar.add_target(QemuTarget)
```

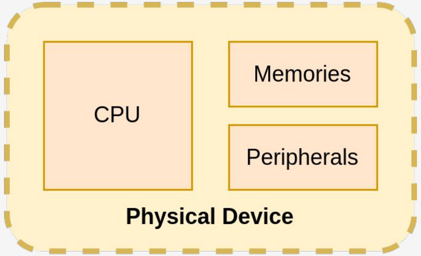




> Initialization

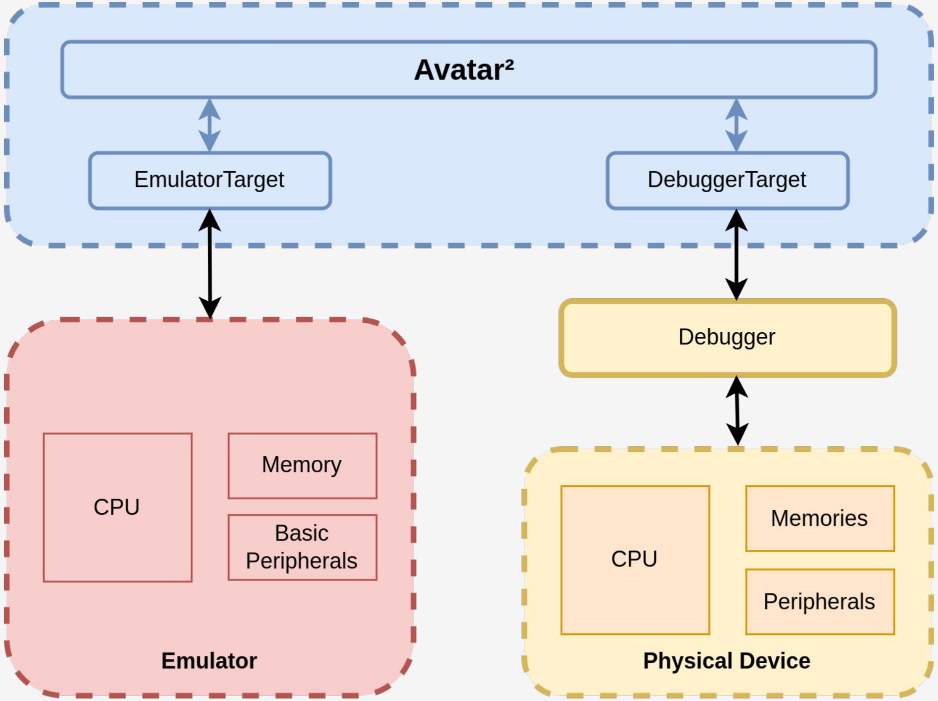
```
# Init
avatar = Avatar(arch=ARM_CORTEX_M3)

device = avatar.add_target(OpenOCDTarget)
emulator = avatar.add_target(QemuTarget)

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





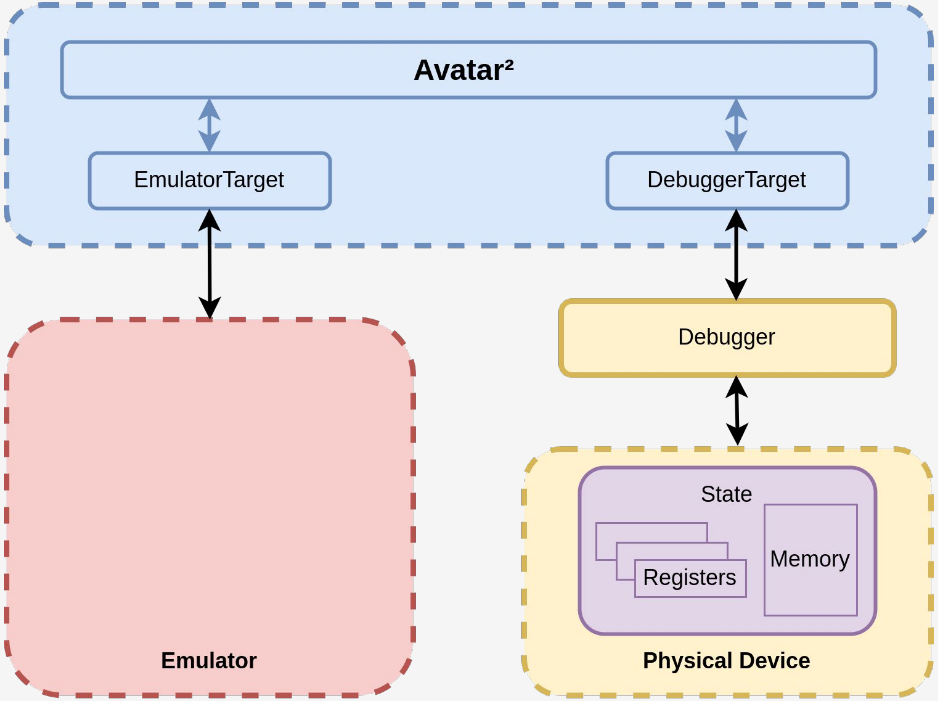
> Initialization

```
# Init
avatar = Avatar(arch=ARM_CORTEX_M3)

device = avatar.add_target(OpenOCDTarget)
emulator = avatar.add_target(QemuTarget)

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

avatar.init_targets()
```



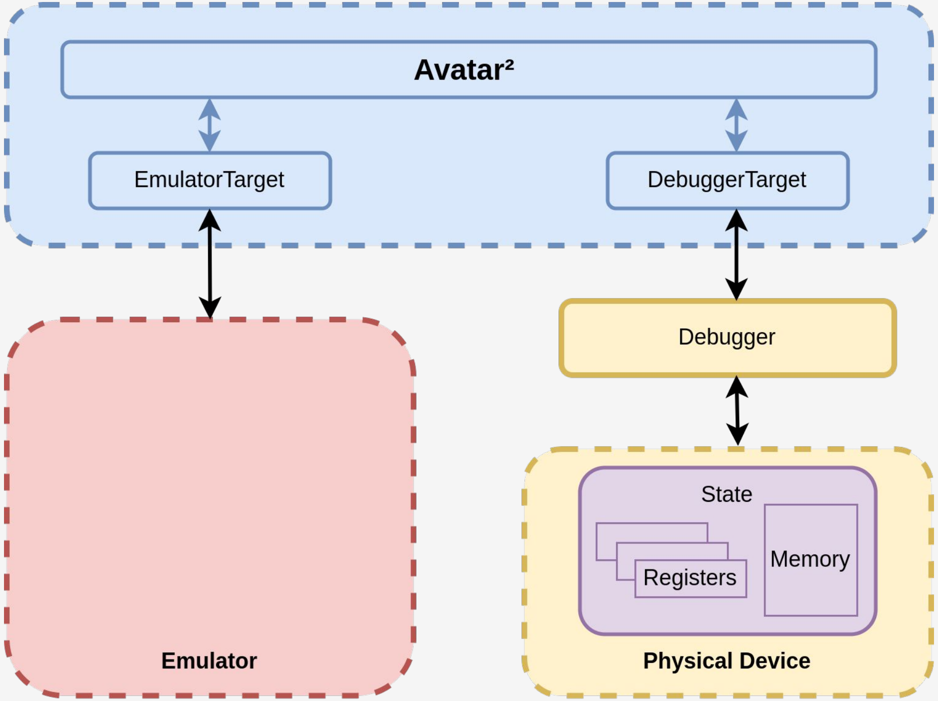
> State Transfer

- **Synchronize** CPU registers and memory content
- **Focus** the analysis (device & firmware initialization)



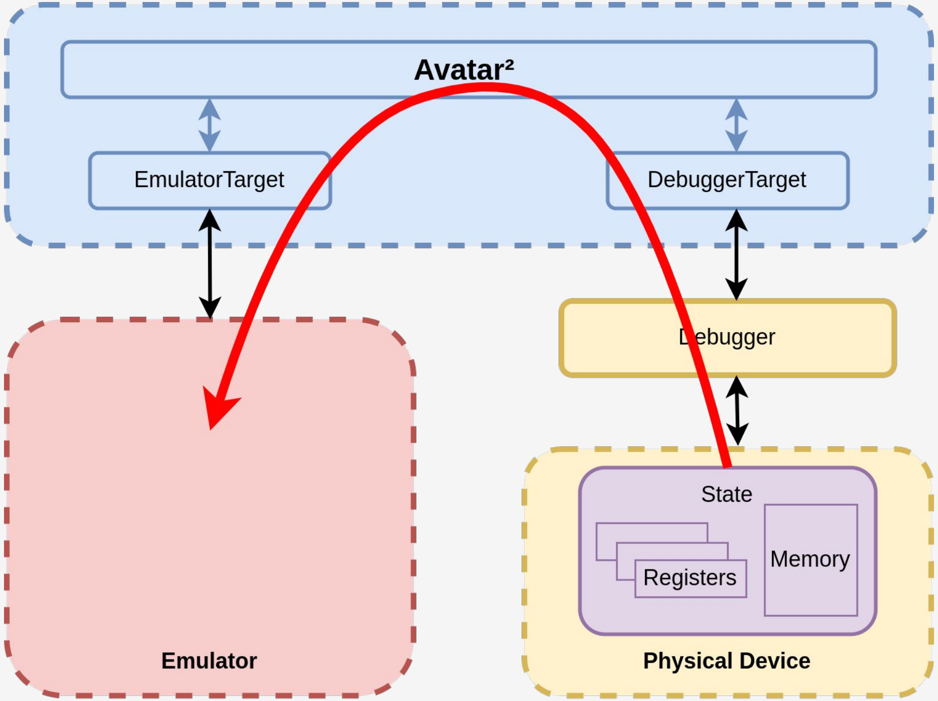
> State Transfer

```
# 1) Set the breakpoint on the physical device
device.set_breakpoint(0x8005104)
device.cont()
device.wait()
```



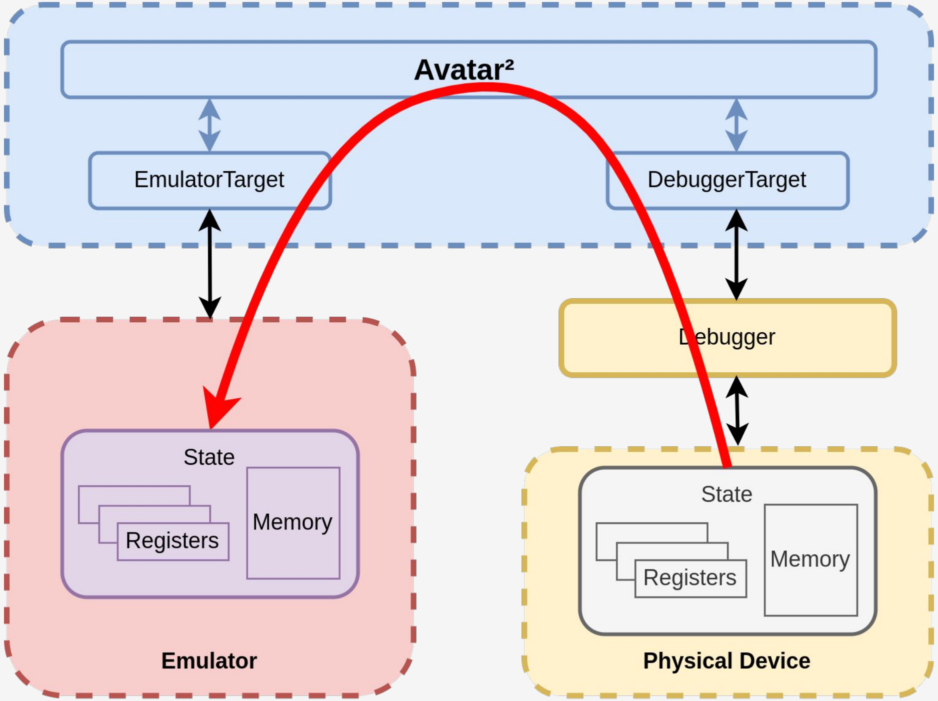
> State Transfer

```
# 1) Set the breakpoint on the physical
device
device.set_breakpoint(0x8005104)
device.cont()
device.wait()

# 2) Transfer the state
avatar.transfer_state(device, emulator,
                      synced_ranges=[ram])
```



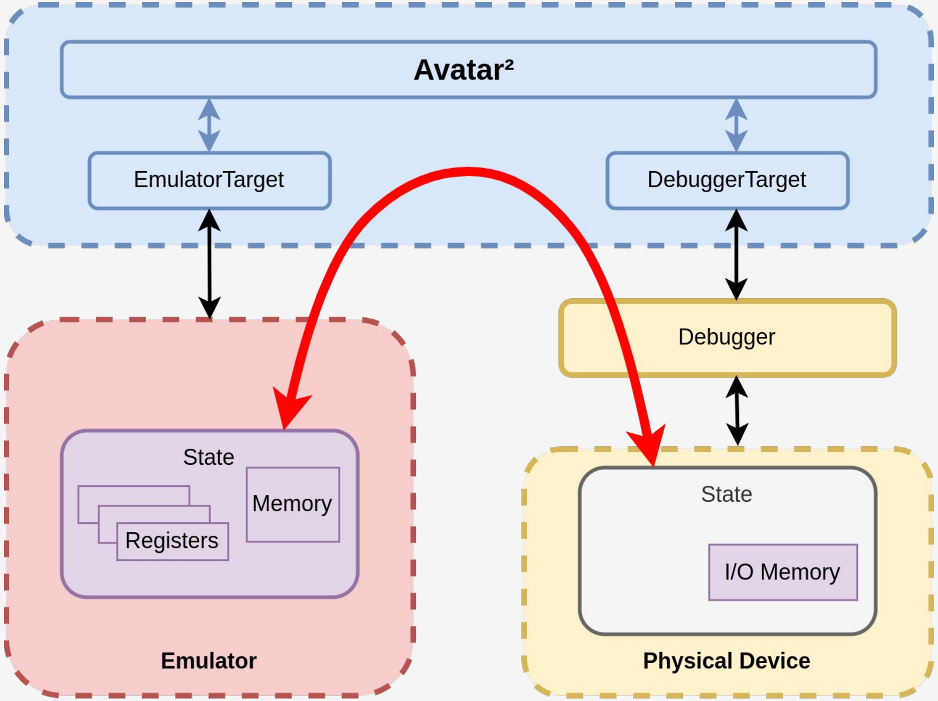
> State Transfer

```
# 1) Set the breakpoint on the physical
device
device.set_breakpoint(0x8005104)
device.cont()
device.wait()

# 2) Transfer the state
avatar.transfer_state(device, emulator,
                   synced_ranges=[ram])


emulator.cont()
```



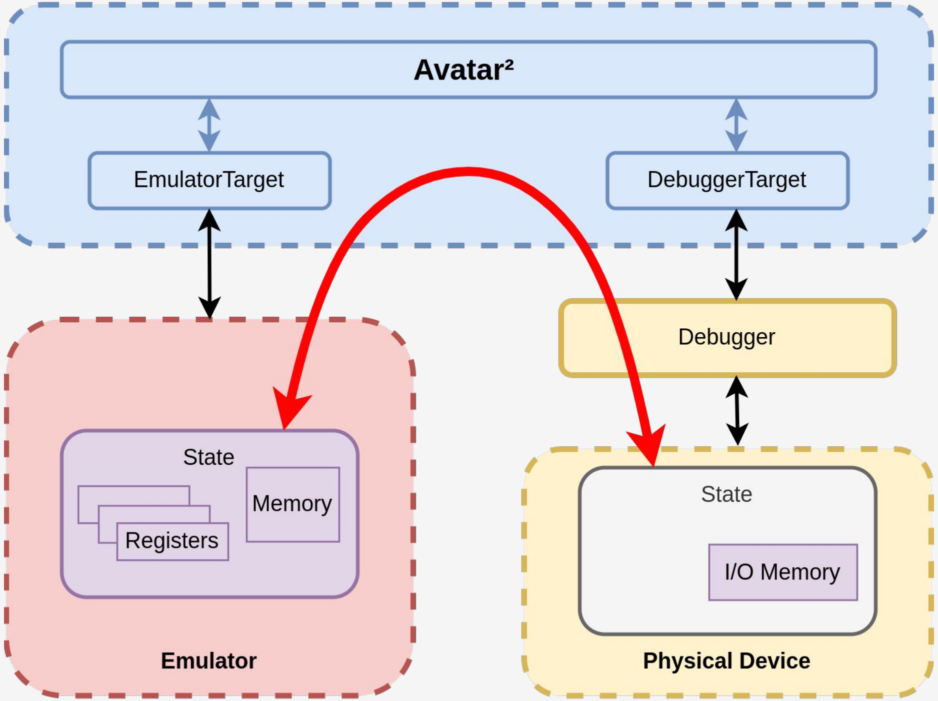
> Peripheral Forwarding

- Forward I/O memory



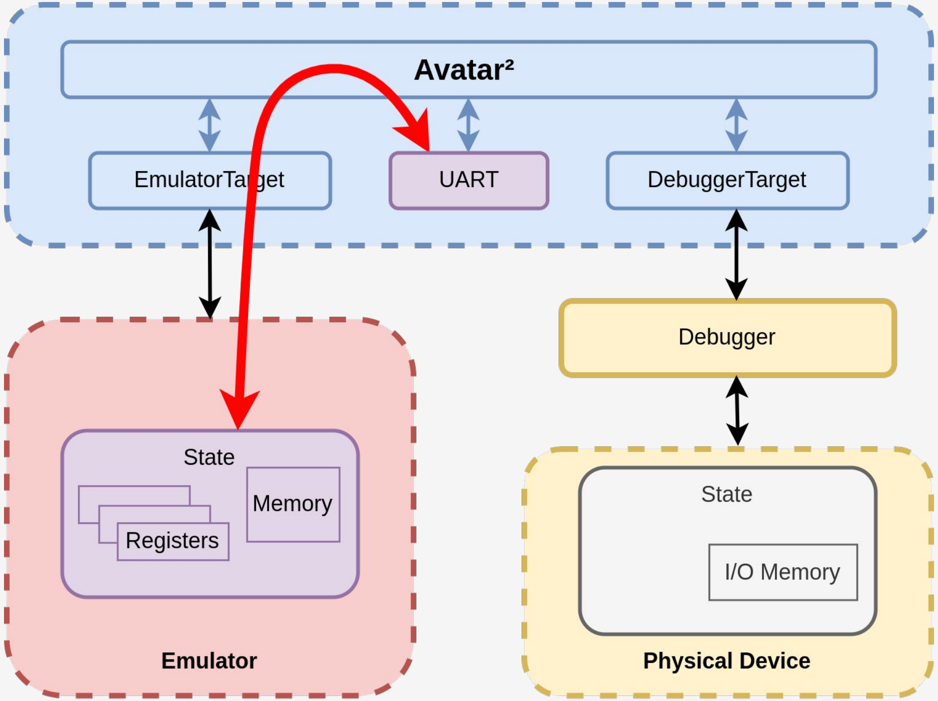
> Peripheral Forwarding

```
# Define the various memory ranges

rom  = avatar.add_memory_range(0x08000000,
0x1000000, file=firmware)

ram  = avatar.add_memory_range(0x20000000,
0x14000)

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



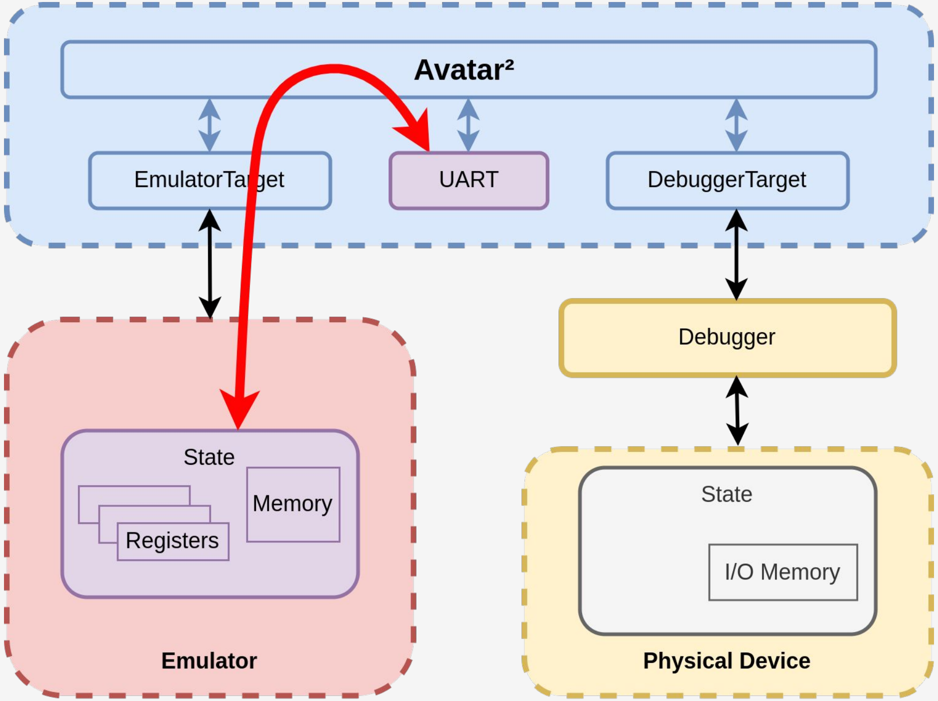
> Peripheral Modeling

- Emulate peripheral in python



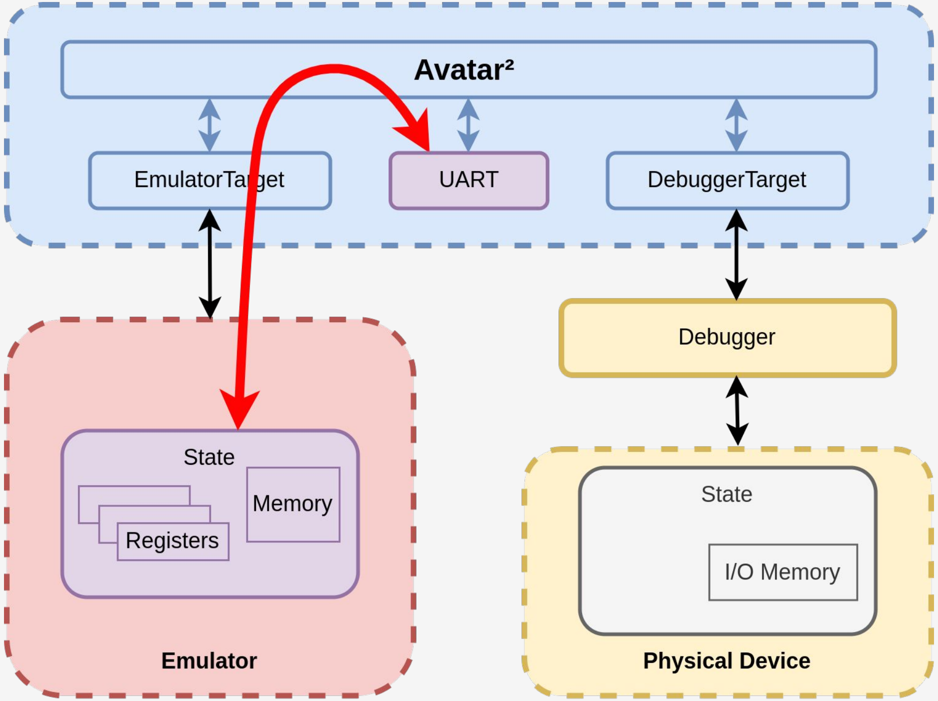
> Peripheral Modeling

```
class UART(AvatarPeripheral):
  # ...

  def dispatch_read(self, offset, size):
    if offset == 0x11c:
      return self.txdone
    return 0x00

  def dispatch_write(self, offset, size, value):
    if offset == 0x11c:
      self.txdone = value
    elif offset == 0x51c:
      print(f">>>> {chr(value)} <<<<")
      self.txdone = 1
    return True
```



> Peripheral Modeling

```
class UART(AvatarPeripheral):
  # ...

# Define the various memory ranges
# ...
uart = avatar.add_memory_range(0x40002000,
0x1000, emulate=UART)
```



> Going Further

- Handbook
  - https://github.com/avatartwo/avatar2/tree/main/handbook

> Going Further

- Handbook
  - https://github.com/avatartwo/avatar2/tree/main/handbook
- Examples
  - https://github.com/avatartwo/avatar2-examples
  - U-Boot - Example without hardware
  - NUCLEO L152RE - Transfer state
  - NRF51 BLE - WiSec'21 tutorial on avatar2
  - Rehosting the Raspberry Pi Pico blink example

> Going Further

- Rehosting
  - https://github.com/halucinator/halucinator
  - Records peripheral accesses to model them: https://github.com/ucsb-seclab/pretender

- Fuzzing
  - https://github.com/FirmWire/FirmWire
  - https://github.com/fgsect/unicorefuzz

- Symbolic execution
  - https://angr.io/blog/angr_symbion/
  - https://github.com/csvl/SEMA-ToolChain

> Conclusion

- Dynamic firmware binary analysis is still a challenging topic
- Various possible approaches
- Avatar^2 focuses on interoperability of tools

> Links

- Framework
  https://github.com/avatartwo/avatar2
- Examples
  https://github.com/avatartwo/avatar2-examples
- Slack
  https://avatartwo.slack.com/
- Team
  - Paul OLIVIER (paul.olivier@laas.fr)
  - Marius MUENCH
  - Florian ALBRECHT
  - Aurélien FRANCILLON

avatar^2
MULTI TARGET ORCHESTRATION

EURECOM
Sophia Antipolis

Backup slides

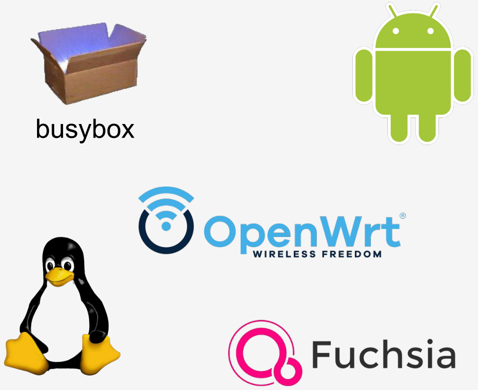
> A wide variety of systems for firmware

OpenWrt
WIRELESS FREEDOM
TinyOS
μClinux
Zephyr™
MicroPython
NUCLEUS
RTOS
STM32
Cube
arm
MBED
uefi
coreboot
THREADX
OP-TEE
.org
freeRTOS
Fuchsia
busybox
U-Boot
VxWorks
ARDUINO™
uClibc-ng
Windows
Embedded CE

> Firmware classification

Type I

- *General purpose* OS-based devices
- minimalist
- lightweight user mode applications



Muench, Marius, et al. *What You Corrupt Is Not What You Crash: Challenges in Fuzzing Embedded Devices*, NDSS 2018

> Firmware classification

Type I

- *General purpose* OS-based devices
- minimalist
- lightweight user mode applications

busybox

OpenWrt
WIRELESS FREEDOM

Fuchsia

Type II

- *Embedded* OS-based devices
- small footprint
- high performance
- real-time scheduling

freeRTOS

Zephyr™

VxWorks

Muench, Marius, et al. *What You Corrupt Is Not What You Crash: Challenges in Fuzzing Embedded Devices*, NDSS 2018

> Firmware classification

Type I

- *General purpose* OS-based devices
- minimalist
- lightweight user mode applications

busybox

OpenWrt WIRELESS FREEDOM

Fuchsia

Type II

- *Embedded* OS-based devices
- small footprint
- high performance
- real-time scheduling

Zephyr™

freeRTOS

VxWorks

Type III

- Devices *without an OS-Abstraction*
- monolithic firmware

ARDUINO

MicroPython

U-Boot

arm MBED

Muench, Marius, et al. *What You Corrupt Is Not What You Crash: Challenges in Fuzzing Embedded Devices*, NDSS 2018

> Firmware classification

Type I

- *General purpose* OS-based devices
- minimalist
- lightweight user mode applications

busybox

OpenWrt
WIRELESS FREEDOM

Fuchsia

Type II

- *Embedded* OS-based devices
- small footprint
- high performance
- real-time scheduling

Type III

- Devices *without an OS-Abstraction*
- monolithic firmware

avatar^2
MULTI TARGET ORCHESTRATION

Muench, Marius, et al. *What You Corrupt Is Not What You Crash: Challenges in Fuzzing Embedded Devices*, NDSS 2018