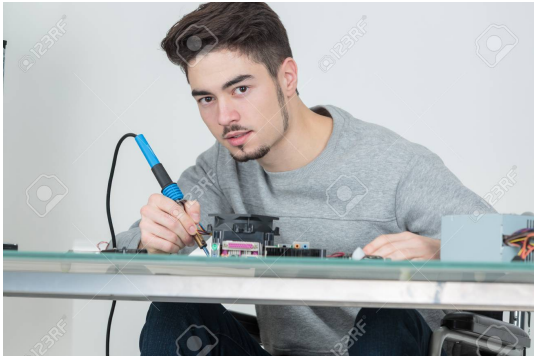
Meet Piotr, a firmware emulation tool for trainers and researchers

Damien Cauquil

Quarkslab

/Me



- *Hacker* since 1998
- Hardware/software reverse-engineer
- IoT Security trainer
- Knows how to hold a soldering iron (obviously)

Table of Contents

Table of Contents

IoT Security Training

- I've been an **IoT Security trainer** for some years
- I used a **COTS device** bought from Amazon during my training

Best. Idea. Ever.

Other issues with COTS devices ...

- You may buy **a lot** of them as it may be discontinued
- Prices may vary a lot !

COVID-19: remote training sessions

- Difficult to send **real devices** to attendees (delays, cost, ...)
- One of them may break it eventually ...

Virtualization FTW!

It became clear that:

- I needed a **way to emulate IoT devices** (and especially my Chinese IP Camera)

Virtualization FTW!

It became clear that:

- I needed a **way to emulate IoT devices** (and especially my Chinese IP Camera)
- I needed a tool **easy to install/setup/use**

Virtualization FTW!

It became clear that:

- I needed a **way to emulate IoT devices** (and especially my Chinese IP Camera)
- I needed a tool **easy to install/setup/use**
- I needed to be able to send **(small) images of virtual devices** over the network

Table of Contents

State of the Art

I then started to look for **the best tools** and found:

- **Qemu**: THE emulation tool !
- **Firmadyne**: an automated firmware emulation framework
- **ARM-X**: a training-oriented firmware emulation tool

State of the Art

Tool	easy setup	easy to use	create new device	export/import
Qemu	✔	✖	✖	✖
Firmadyne	✔	✖	✖	✖
ARM-X	✔	✔	⚠	✖

Table of Contents

Why another tool ?

ARM-X is the best candidate so far, **BUT**:

- It is made of **Bash scripts**
- *Complex* configuration files
- No easy **import/export feature**
- Not really **modular**

Table of Contents

Table of Contents

ARM-X vs. Piotr

- **Piotr**, like **ARM-X**, uses a Qemu virtual host to host our target device (chroot)
- **Piotr** relies on Plan 9 Resource Sharing Protocol (**9P2000**) rather than Samba (no network required)
- It also uses default **Qemu agent** to interact with Qemu hosts
- Written in **Python**, can be installed with **pip**
- Allows to **export and import** of virtual devices

Software architecture

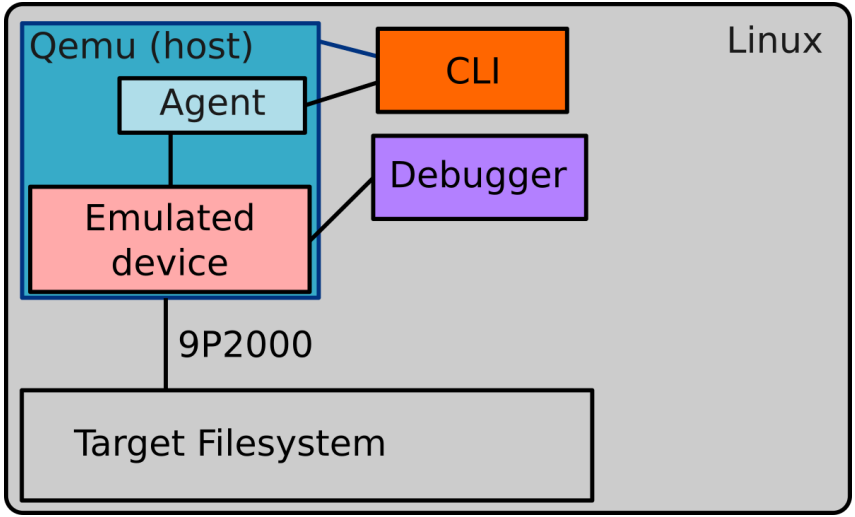


Table of Contents

Creating a device

So you want to emulate a real IoT device ? Follow these steps:

- Extract your device **root filesystem** and save it into a dedicated folder
- Create some **launching scripts** that will be used to start the device from your Qemu host
- Stick with Piotr's default **Linux kernel** or build yours with **buildroot**
- Fill a **YAML configuration file** to tell Piotr where to find the kernel and the root filesystem
- **Run** your emulated device with Piotr !

Example YAML config file

```yaml
version: "1.0"
device:
  name: "Damn Vulnerable ARM Router by Saumil Shah"
  machine:
    platform: virt
    memory: 1024
    cpu: cortex-a7

  bootargs: "root=/dev/vda rw console=ttyAMA0,115200"
  guestfs: virtfs
  drive_type: virtio

  network:
    nic0: user
  redirect:
    nic0:
      web: tcp,8081,80
      lightsrv: tcp,8080,8080
      gdb: tcp,4444,4444
```

Exporting a device

Exporting an existing device is as simple as this:

```
$ sudo piotr device export example ./example.piotr
```

Importing a device

Importing a device is also dead simple:

```
$ sudo piotr device add ./example.piotr
```

Importing a device

Table of Contents

Debugging a target's process

```
$ piotr-ps
[...]
725 root lightsrv
[...]
$ piotr-debug 725
$ gdb-multiarch
(gdb) set architecture arm
(gdb) target extended-remote 127.0.0.1:4444
```

Frida-server is also available but only compatible with glibc-based systems so far.

Instrumenting with Python

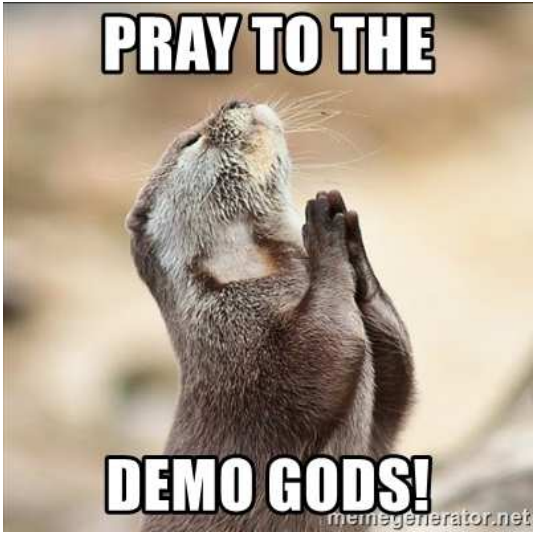
```python
from piotr.api import *
# get running instance of virtual device
myinst = Piotr.instance('demo')

# get target pid and attach debugger
pid = myinst.pid('/usr/bin/lightsrv')
dbg = myinst.debug(pid)

# continue execution
dbg.cont()
```

Table of Contents

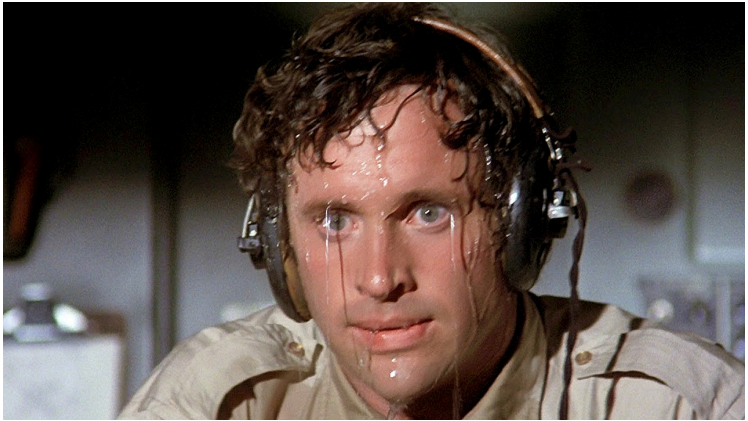
Demo 1: Pwning an IP Chinese camera

```
      }
      else {
        local_14 = access("/mnt/sdcard/auth_exshell.sh",0);
        if (local_14 == 0) {
          system("/mnt/sdcard/auth_exshell.sh");
        }
      }
    }
  }
}
if (iSpeechLanguage == 1000) {
  PlayFile("/mnt/mtd/mvsound/sf_sysstarting_cn.wav",0);
}
else {
  PlayFile("/mnt/mtd/mvsound/sf_sysstarting_en.wav",0);
}
```

Demo 1: Pwning an IP Chinese camera



Demo 2: Emulating a RUTX10 (Teltonika)



Demo 3: Automated exploitation of DVAR

```
Thread 2.1 "lightsrv" received signal SIGSEGV, Segmentation fault.
[Switching to Thread 322.322]
0x41414140 in ?? ()
[ Legend: Modified register | Code | Heap | Stack | String ]

$r0   : 0x194
$r1   : 0xbeffdd18  →  "<html>\n  <head>\n    <title>No.</title>\n  </head[...]"
$r2   : 0xb3
$r3   : 0x0
$r4   : 0x41414141 ("AAAA"?)
$r5   : 0x41414141 ("AAAA"?)
$r6   : 0x41414141 ("AAAA"?)
$r7   : 0x41414141 ("AAAA"?)
```

Demo 3: Automated exploitation of DVAR

Table of Contents

Conclusion

- *Piotr* provides a docker-free/network-free/qemu-based device emulation environment

Conclusion

- *Piotr* provides a docker-free/network-free/qemu-based device emulation environment
- It emphasizes on **ease of use**, **sharing** and **automation**

Conclusion

- *Piotr* provides a docker-free/network-free/qemu-based device emulation environment
- It emphasizes on **ease of use**, **sharing** and **automation**
- Can be used to build other tools upon it (extensible) !

Conclusion

- *Piotr* provides a docker-free/network-free/qemu-based device emulation environment
- It emphasizes on **ease of use**, **sharing** and **automation**
- Can be used to build other tools upon it (extensible) !
- Compatible with **ARM-X**: similar architecture, different tooling

Give it a try !

pip install piotr

https://piotr.readthedocs.io/

https://github.com/virtualabs/piotr

Question(s) ?

What does PIOTR stand for ?

- **P**ythonic **IoT** **R**esearch framework
- **P**iotr **I**s **O**bviously a **T**ough **R**ussian
- **P**retty **I**neffective **O**bscur **T**ool for **R**esearch
- **P**wning **I**ot **O**bjects for **T**raining and **R**esearch
- **P**iotr **I**nstruments **O**ther **T**argets as **R**oot

Choosing a name for a tool is way too much responsibility.

Thank you

Contact information:

Email: dcauquil@quarkslab.com

Phone: +33 1 58 30 81 51

Website: https://www.quarkslab.com

Quarkslab