


LISA-QEMU

Peter Puhov
and
Rob Foley

INTRODUCTIONS

- ▶ Peter Puhov
 - ▶ Chief Architect @ Futurewei
 - ▶ Member engineer in Linaro KWG working on Scheduler.
 - ▶ peter.puhov@linaro.org
- ▶ Rob Foley
 - ▶ Architect @ Futurewei
 - ▶ Member engineer in Linaro TCWG working on QEMU.
 - ▶ robert.foley@linaro.org
 - ▶ rf-fw @ #linaro-tcwg #qemu

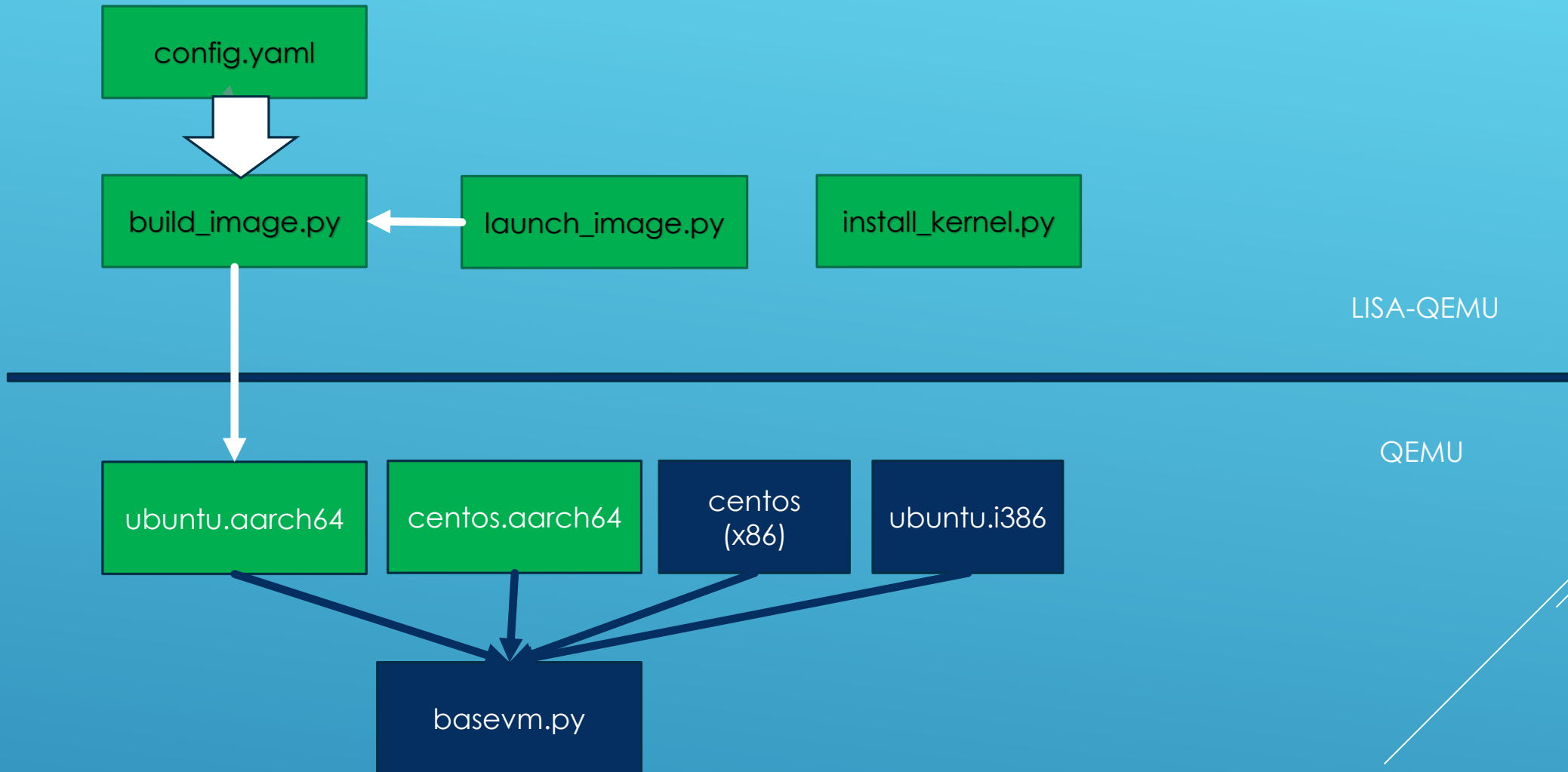
AGENDA

- ▶ Introduction to LISA-QEMU
 - ▶ Building a VM
 - ▶ Launching the VM
 - ▶ Installing Kernel
 - ▶ Configuring VM via yaml files
- 
- Several thin, parallel white lines of varying lengths and slopes are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

ABOUT LISA-QEMU

- ▶ **Integration between LISA and QEMU**
 - ▶ <https://github.com/rf972/lisa-qemu>
- ▶ **LISA**
 - ▶ “The LISA project provides a toolkit that supports regression testing and interactive analysis of Linux kernel behavior.”
 - ▶ <https://github.com/ARM-software/lisa>
 - ▶ Helps kernel developers test their changes
- ▶ **Our goal: Ease of test for aarch64 architectures**
 - ▶ Enable development for developers without aarch64 hardware. (TCG)
 - ▶ Including a variety of hardware configurations.
 - ▶ Support all architectures QEMU does.
- ▶ **Our focus: kernel CFS scheduler task placement and NUMA**

NEW MODULES/SCRIPTS



SETUP STEPS

► Install packages

```
apt-get build-dep -y qemu
```

```
apt-get install -y python3-yaml wget qemu-efi-aarch64 \  
qemu-utils genisoimage qemu-user-static git
```

► Pull down repo

```
git clone https://github.com/rf972/lisa-qemu.git
```

```
cd lisa-qemu
```

```
git submodule update --init --progress [--recursive for lisa]
```

BUILDING A VM

▶ Build Command

- ▶ `python3 scripts/build_image.py`
- ▶ Equivalent to: `python3 scripts/build_image.py --image_type ubuntu.aarch64`
- ▶ Assumes defaults, you can override them, see `build_image.py -help`

▶ Other examples

- ▶ `python3 scripts/build_image.py --image_type centos.aarch64`
- ▶ `python3 scripts/build_image.py --image_type ubuntu.i386 \`
`--config example.yml`

▶ Valid image types same as QEMU:

- ▶ `centos`, `centos.aarch64`, `ubuntu.aarch64`, `ubuntu.i386`, etc.
- ▶ See `build_image.py --help` for complete list.

VM BUILD TIME

- ▶ **Time to create VM** (*less Base VM download time)
 - ▶ 50 minutes - Intel i7 laptop with 2 cores and 16 GB of memory
 - ▶ 6 minutes - Huawei Taishan 2286 V2 with 128 ARM cores and 512 GB of memory.
 - ▶ 1.5 minutes - Huawei Taishan 2286 V2 with KVM.

BUILD IMAGE ARGUMENTS

```
python3 scripts/build_image.py --help
usage: build_image.py [-h] [--debug] [--dry_run] [--ssh]
                    [--image_type IMAGE_TYPE] [--image_path IMAGE_PATH]
                    [--config CONFIG] [--build_qemu]
```

Build the qemu VM image for use with lisa.

optional arguments:

<code>-h, --help</code>	show this help message and exit
<code>--debug</code>	enable debug output
<code>--dry_run</code>	Just show commands issued by script, do not execute them.
<code>--ssh</code>	Launch VM and open an ssh shell.
<code>--image_type IMAGE_TYPE</code>	Type of image to build. From external/qemu/tests/vm. default is ubuntu.aarch64
<code>--image_path IMAGE_PATH</code>	Allows overriding path to image.
<code>--config CONFIG</code>	config file. default is conf/conf_default.yml.
<code>--build_qemu</code>	Build QEMU. QEMU is built initially and not repeated unless this argument is selected.

BUILDING A VM (CONTINUED)

Image creation starting. Please be patient, this may take several minutes...
To enable more verbose tracing of each step, please use the --debug option.

```
--2020-04-15 21:06:03-- https://cloud-images.ubuntu.com/releases/18.04/release/ubuntu-18.04-server-cloudimg-arm64.img
```

```
Resolving cloud-images.ubuntu.com (cloud-images.ubuntu.com)... 91.189.88.89, 2001:67c:1560:8001::8001
```

```
Saving to: '/root/.cache/qemu-vm/download/74504fbbc8a322741e6e524ae19a72c8e82a25f2.download'  
/root/.cache/qemu-vm/downlo 100%[=====>] 312.19M 2.05MB/s  
in 3m 41s
```

```
2020-04-15 21:09:45 (1.41 MB/s) - '/root/.cache/qemu-  
vm/download/74504fbbc8a322741e6e524ae19a72c8e82a25f2.download' saved [327352320/327352320]  
Image resized.
```

```
guest user:pw qemu:gemupass
```

```
Connection to 127.0.0.1 closed by remote host.
```

```
Image creation successful.
```

```
Image path: /home/rob/qemu/lisa-qemu/build/VM-ubuntu.aarch64/ubuntu.aarch64.img
```

AFTER VM IMAGE BUILD

build

```
| -- VM-ubuntu.aarch64  
|   | -- conf.yml  
|   | -- id_rsa  
|   | -- id_rsa.pub  
|   `-- ubuntu.aarch64.img
```

LAUNCH VM

- ▶ **Launch Command**
 - ▶ `python3 scripts/launch_image.py`
- ▶ **Bring up time relatively quick 2-3 minutes (TCG)**
 - ▶ Depends on number of configured cores.
- ▶ **To launch other types of VMs:**
 - ▶ `python3 scripts/launch_image.py --image_type centos.aarch64`
- ▶ **To launch specific VM:**
 - ▶ `python3 scripts/launch_image.py --image_path myimage.img`

LAUNCH VM

```
$ python3 ./scripts/launch_image.py
```

```
Conf:          /home/rob/qemu/lisa-qemu/build/VM-ubuntu.aarch64/conf.yml
```

```
Image type:    ubuntu.aarch64
```

```
Image path:    /home/rob/qemu/lisa-qemu/build/VM-ubuntu.aarch64/ubuntu.aarch64.img
```

```
Launching Image. Please be patient, this may take several minutes...
```

```
To enable more verbose tracing of each step, please use the --debug option.
```

```
qemu@ubuntu-aarch64-guest:~$
```

INSTALL KERNEL

- ▶ **Goal is to help streamline kernel dev process.**
- ▶ **Starting point is kernel .deb package.**
 - ▶ `make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- bindeb-pkg`
- ▶ **Puts kernel into the image**
- ▶ **Extract relevant files needed for qemu to boot kernel directly.**
- ▶ **Command:**

```
sudo python3 scripts/install_kernel.py \  
    --kernel_pkg ../linux/linux-image-5.5.11_5.5.11-1_arm64.deb
```
- ▶ **By default uses chroot.**
- ▶ **Optionally can use --vm argument**

INSTALL KERNEL (CONTINUED)

```
sudo python3 scripts/install_kernel.py --kernel_pkg ../linux/linux-image-5.5.11_5.5.11-1_arm64.deb
```

```
scripts/install_kernel.py: image: build/VM-ubuntu.aarch64/ubuntu.aarch64.img
```

```
scripts/install_kernel.py: kernel_pkg: ../linux/linux-image-5.5.11_5.5.11-1_arm64.deb
```

Install kernel successful.

Image path: **/home/rob/qemu/lisa-qemu/build/VM-ubuntu.aarch64/ubuntu.aarch64.img.kernel-5.5.11-1**

To start this image run this command:

```
python3 /home/rob/qemu/lisa-qemu/scripts/launch_image.py --image_path /home/rob/qemu/lisa-qemu/build/VM-ubuntu.aarch64/ubuntu.aarch64.img.kernel-5.5.11-1
```

INSTALL KERNEL (CONTINUED)

Build

```
|-- VM-ubuntu.aarch64
|   |-- conf.yml
|   |-- id_rsa
|   |-- id_rsa.pub
|   |-- ubuntu.aarch64.img
|   |-- ubuntu.aarch64.img.kernel-5.5.11-1
|   |-- initrd.img-5.5.11-1
|   |-- conf-kernel-5.5.11-1.yml
|   `-- vmlinuz-5.5.11-1
```


VM CONFIGURATION YAML

- ▶ **Allows for configuring VM**
 - ▶ Default yaml provided with built VMs.
 - ▶ Any value not provided uses a default.
- ▶ **Credentials**
 - ▶ root password, username, password, ssh keys, ssh port
- ▶ **Hardware**
 - ▶ cpu, machine, memory
 - ▶ To use alternate and/or complex hardware topologies.
 - ▶ qemu_args gets fed through to QEMU.
- ▶ **Kernel**
 - ▶ Supports providing linux kernel and/or kernel command line.
 - ▶ Provide -kernel, -initrd, -append "cmdline" in qemu_args.
- ▶ **Configuration**
 - ▶ install_cmds allows specifying optional setup cmds.

YAML EXAMPLE

qemu-conf:

Login username (has to be sudo enabled)

username: qemu

Password is used by root and default login user.

password: "qemupass"

ssh_key: /home/user/.ssh/id_rsa

ssh_pub_key: /home/user/.ssh/id_rsa.pub

dns: 1.234.567.89

By default install as little as possible since lisa will install whatever it needs.

install_cmds: ""

Specify the fixed ssh port to be used by lisa.

ssh_port: 5555

YAML EXAMPLE (CONTINUED)

```
cpu: max
```

```
machine: virt,gic-version=max
```

```
memory: 16G
```

```
qemu_args: "-smp cpus=16,sockets=2,cores=8
```

```
          -numa node,cpus=0-3,nodeid=0 -numa node,cpus=4-7,nodeid=1
```

```
          -numa node,cpus=8-11,nodeid=2 -numa node,cpus=12-15,nodeid=3
```

```
          -numa dist,src=0,dst=1,val=15 -numa dist,src=2,dst=3,val=15
```

```
          -numa dist,src=0,dst=2,val=20 -numa dist,src=0,dst=3,val=20
```

```
          -numa dist,src=1,dst=2,val=20 -numa dist,src=1,dst=3,val=20"
```

YAML EXAMPLE (KERNEL)

```
qemu_args: '-kernel build/VM-ubuntu.aarch64/vmlinuz-5.5.11-1  
            -initrd build/VM-ubuntu.aarch64/initrd.img-5.5.11-1  
            -append "root=/dev/vda1 nokaslr console=ttyAMA0"'
```

GETTING STARTED

- ▶ `apt-get build-dep -y qemu`
- ▶ `apt-get install -y python3-yaml wget qemu-efi-aarch64 \`
`qemu-utils genisoimage qemu-user-static git`
- ▶ `git clone https://github.com/rf972/lisa-qemu.git`
- ▶ `cd lisa-qemu`
- ▶ `git submodule update --init --progress`
- ▶ `sudo python3 scripts/build_image.py`
 - ▶ Expected to take about 10-50 mins
- ▶ `sudo python3 ./scripts/launch_image.py`
 - ▶ Expected to take about 2-3 mins

FUTURES AND DISCUSSION

- ▶ CentOS install_kernel support?
 - ▶ Rootfs use cases
 - ▶ Others?
- 
- A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide, creating a modern, abstract graphic element.

REFERENCES

- ▶ This presentation link on blog
- ▶ LISA-QEMU github
 - ▶ <https://github.com/rf972/lisa-qemu>
- ▶ Our Blog
 - ▶ <https://futurewei-cloud.github.io/ARM-Datacenter/home/>
- ▶ LISA-QEMU demo
 - ▶ <https://futurewei-cloud.github.io/ARM-Datacenter/qemu/lisa-qemu-demo1/>
- ▶ LISA
 - ▶ <https://github.com/ARM-software/lisa>

THANK
YOU !

