Bare-metal testing using containerised test suites

Martin Roukala (Valve contractor)

Who am I?

- Martin Roukala (néé Peres)
- Freelancer at MuPuF TMI and Valve contractor

Who am I?

- Martin Roukala (néé Peres)
- Freelancer at MuPuF TMI and Valve contractor

- Active in the graphics subsystem:
  - ex-member of the X.Org of directors (2013-2019)
  - GPU focus: AMD, Intel GFX, Nouveau

Who am I?

- Martin Roukala (néé Peres)
- Freelancer at MuPuF TMI and Valve contractor

- Active in the graphics subsystem:
  - ex-member of the X.Org of directors (2013-2019)
  - GPU focus: AMD, Intel GFX, Nouveau

- My mission: Production-ready upstream Linux

Who am I?

- Martin Roukala (néé Peres)
- Freelancer at MuPuF TMI and Valve contractor

- Active in the graphics subsystem:
  - ex-member of the X.Org of directors (2013-2019)
  - GPU focus: AMD, Intel GFX, Nouveau

- My mission: Production-ready upstream Linux

- Linux GFX testing experience:
  - Intel GFX CI / CI Bug Log: Running IGT test suite on 100+ machines
  - EzBench: Auto-bisecter of performance/unit test/image changes

Why care about rootfs generation?

- To prevent regressions, a lot of test suites need to be run on a lot of HW
- But:

Why care about rootfs generation?

- To prevent regressions, a lot of test suites need to be run on a lot of HW
- But:
  - Different hardware are found in different CI farms

Why care about rootfs generation?

- To prevent regressions, a lot of test suites need to be run on a lot of HW
- But:
  - Different hardware are found in different CI farms
  - Different CI farms have different interfaces / requirements

Why care about rootfs generation?

- To prevent regressions, a lot of test suites need to be run on a lot of HW
- But:
  - Different hardware are found in different CI farms
  - Different CI farms have different interfaces / requirements
  - Rebooting to test-suite-specific rootfs is slow, and compiling multiple test suites in one rootfs can be tricky (GFX has loads of deps)

Why care about rootfs generation?

- To prevent regressions, a lot of test suites need to be run on a lot of HW
- But:
  - Different hardware are found in different CI farms
  - Different CI farms have different interfaces / requirements
  - Rebooting to test-suite-specific rootfs is slow, and compiling multiple test suites in one rootfs can be tricky (GFX has loads of deps)
- All of this makes running new test suites in one or more CI systems difficult

Quick comparison

Rootfs

- Traditional method of deploying the test environment

OCI containers

Quick comparison

Rootfs

- Traditional method of deploying the test environment

OCI containers

- Traditionally used for unit testing, and in the web world

Quick comparison

Rootfs

- Traditional method of deploying the test environment
- Can be created using:
  - Yocto / buildroot / kci_rootfs / ...
  - Debos / virt-builder / ...

OCI containers

- Traditionally used for unit testing, and in the web world

Quick comparison

Rootfs

- Traditional method of deploying the test environment
- Can be created using:
  - Yocto / buildroot / kci_rootfs / ...
  - Debos / virt-builder / ...

OCI containers

- Traditionally used for unit testing, and in the web world
- Can be created using:
  - docker / podman
  - buildah

Quick comparison

Rootfs

- Traditional method of deploying the test environment
- Can be created using:
  - Yocto / buildroot / kci_rootfs / ...
  - Debos / virt-builder / ...
- Generates a full disk image
  - Self-contained
  - Slower: The full image needs flashing
  - Low portability (modules, firmwares)

OCI containers

- Traditionally used for unit testing, and in the web world
- Can be created using:
  - docker / podman
  - buildah

Quick comparison

Rootfs

- Traditional method of deploying the test environment
- Can be created using:
  - Yocto / buildroot / kci_rootfs / ...
  - Debos / virt-builder / ...
- Generates a full disk image
  - Self-contained
  - Slower: The full image needs flashing
  - Low portability (modules, firmwares)

OCI containers

- Traditionally used for unit testing, and in the web world
- Can be created using:
  - docker / podman
  - buildah
- Generates a set of overlays (layers)
  - Requires platform setup
  - Faster: the base OS is cached
  - High portability

Your unit tests already run
in a container...

Your unit tests already run in a container...

Why not reuse it for bare-metal testing?

How to boot your container?

- Containers require platform initialization to be done
- Do we need another rootfs for this?

How to boot your container?

- Containers require platform initialization to be done
- Do we need another rootfs for this?

Boot2container

What's boot2container?

- A small (< 20 MB) and PXE-bootable initramfs (url)
- A declarative configuration via the kernel command line
- Features:

What's boot2container?

- A small (< 20 MB) and PXE-bootable initramfs (url)
- A declarative configuration via the kernel command line
- Features:
  - Network: DHCP & NTP

What's boot2container?

- A small (< 20 MB) and PXE-bootable initramfs (url)
- A declarative configuration via the kernel command line
- Features:
  - Network: DHCP & NTP
  - Cache drive:
    - Auto-selection, or configurable
    - Auto-formating, if needed
    - Swap file

What's boot2container?

- A small (< 20 MB) and PXE-bootable initramfs (url)
- A declarative configuration via the kernel command line
- Features:
  - Network: DHCP & NTP
  - Cache drive:
    - Auto-selection, or configurable
    - Auto-formating, if needed
    - Swap file
  - Volumes:
    - mirroring from an S3-compatible storage
    - local encryption (fscrypt)
    - expiration

Running IGT using boot2container

Kernel command line

- b2c.cache_device=auto b2c.ntp_peer=auto
- b2c.minio="job,{{ minio_url }},{{ job_bucket_access_key }},{{ job_bucket_secret_key }}"
- b2c.volume="job,mirror=job/{{ job_bucket }},pull_on=pipeline_start,auto_push,expiration=pipeline_end"
- b2c.container="-ti docker://registry.freedesktop.org/mupuf/valve-infra/machine_registration:latest check"
- b2c.container="-t -v job:/results docker://registry.freedesktop.org/drm/igt-gpu-tools/igt:master igt_runner -o /results"
- console={{ local_tty_device }},115200 earlyprintk=vga,keep loglevel=6

Need to run an extra test suite?

Need to run an extra test suite?

Add another b2c.container in the kernel command line!

Limitation of containers

- Not applicable to platforms with less than 64 MB of RAM
- More?

Open questions

- How can we standardize on the test result format?
- Anything else?

Thanks for listening!