OpenHPC Automation with Ansible

Baptiste Gerondeau, Takeharu Kato, Renato Golin

ISC18, Arm Workshop, 28th June 2018

Agenda

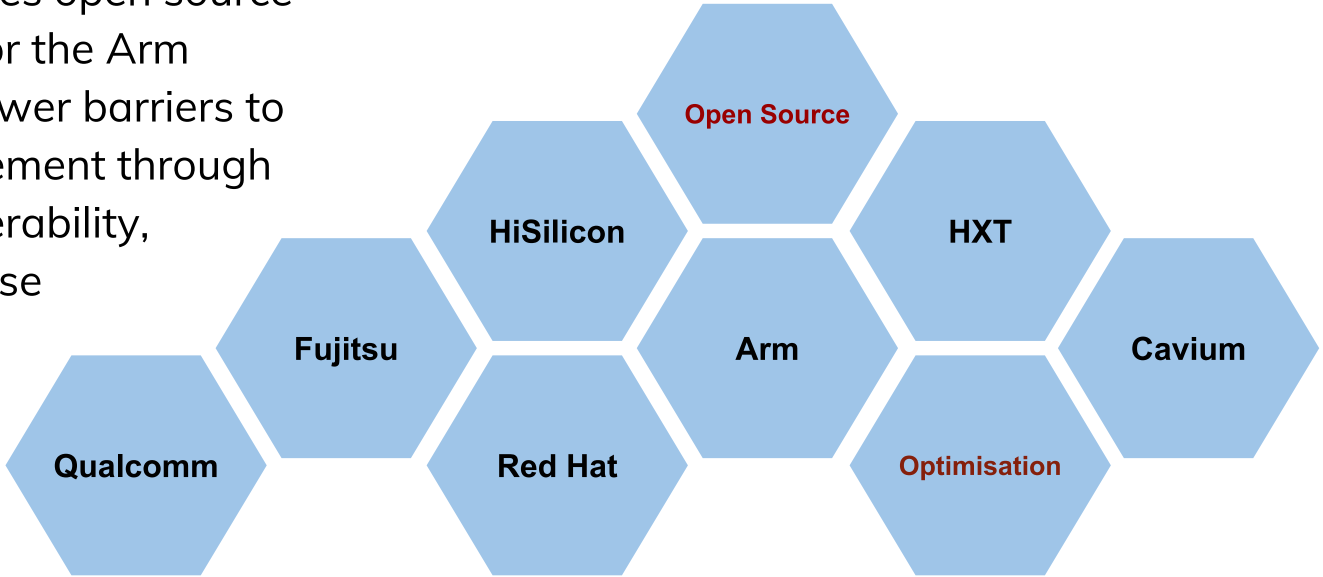
- Linaro’s HPC-SIG Lab
- OpenHPC Ansible Automation
- Results

The HPC-SIG Lab

Linaro High Performance Computing
Special Interest Group

The Linaro HPC SIG drives open source software development for the Arm architecture. It aims to lower barriers to deployment and management through standardisation, interoperability, orchestration and use case development.

linaro.org/hpc



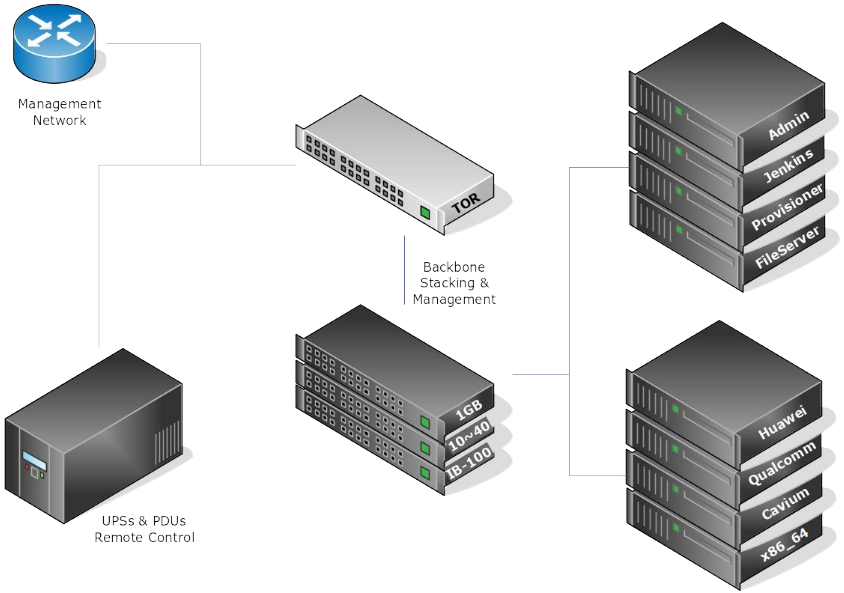
The HPC-SIG Lab

Goals:

- Cluster Automation & Validation
- Benchmarking & Performance Investigation

Requirements:

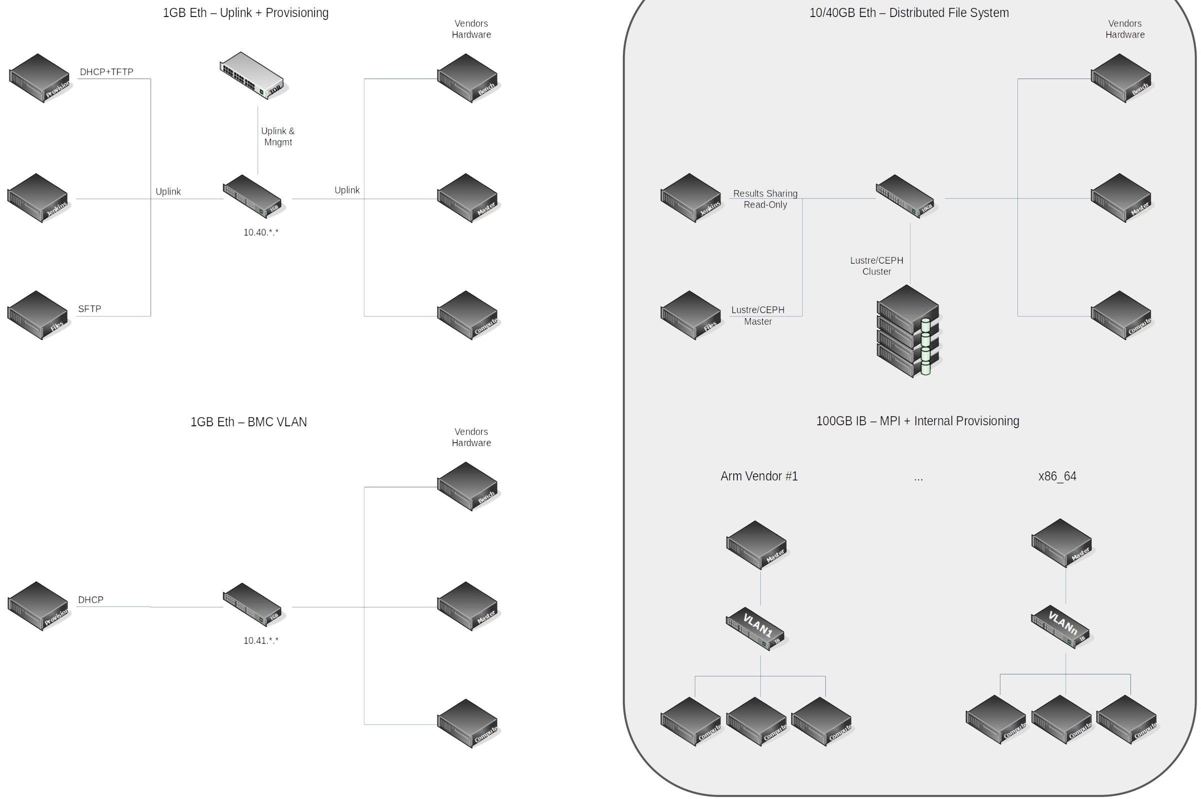
- Stability & Repeatability
- Close-to-production environment
- Upstream technology (reproducibility)
- Vendor isolation (hardware, results)



The HPC-SIG Lab

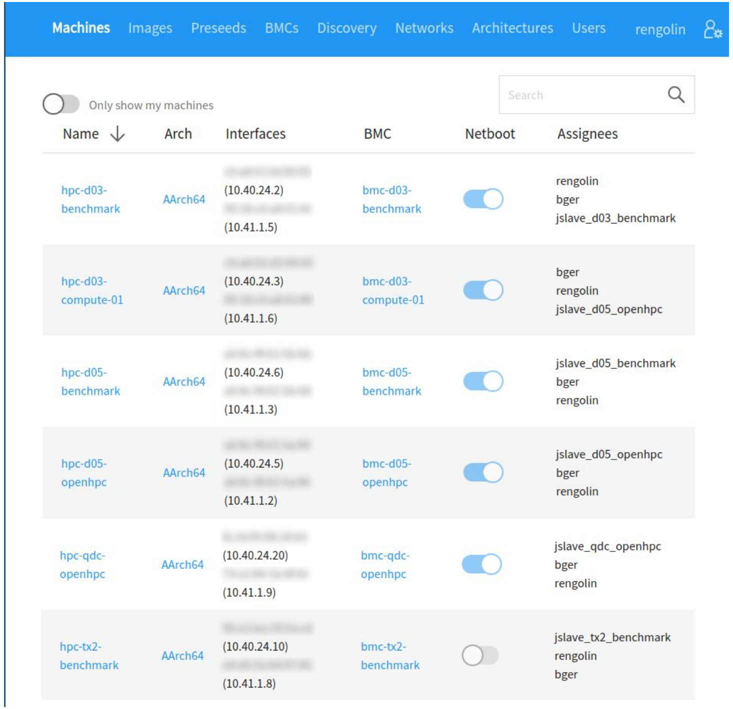
Network Layout

- Flat Ethernet
  - Uplink/Provision
- BMC subnet (VLAN)
- File System subnet
  - 10/40GB Ethernet
  - Lustre/Ceph (future)
- MPI subnet
  - 100GB InfiniBand
  - Slave provision



The HPC-SIG Lab

- Stability & Repeatability
  - Critical external components cached locally
  - Strict migration plans (staging)
- Close-to-production environment
  - Hardware and firmware updated frequently
- Upstream technology (reproducibility)
  - All components are open source / upstream
- Vendor isolation (hardware, results)
  - VPN, Provisioner, Jenkins, SSH control



Name	Arch	Interfaces	BMC	Netboot	Assignees
hpc-d03-benchmark	AArch64	[redacted] (10.40.24.2) [redacted] (10.41.1.5)	bmc-d03-benchmark	●	rengolin bger jslave_d03_benchmark
hpc-d03-compute-01	AArch64	[redacted] (10.40.24.3) [redacted] (10.41.1.6)	bmc-d03-compute-01	●	bger rengolin jslave_d05_openhpc
hpc-d05-benchmark	AArch64	[redacted] (10.40.24.6) [redacted] (10.41.1.3)	bmc-d05-benchmark	●	jslave_d05_benchmark bger rengolin
hpc-d05-openhpc	AArch64	[redacted] (10.40.24.5) [redacted] (10.41.1.2)	bmc-d05-openhpc	●	jslave_d05_openhpc bger rengolin
hpc-qdc-openhpc	AArch64	[redacted] (10.40.24.20) [redacted] (10.41.1.9)	bmc-qdc-openhpc	●	jslave_qdc_openhpc bger rengolin
hpc-tx2-benchmark	AArch64	[redacted] (10.40.24.10) [redacted] (10.41.1.8)	bmc-tx2-benchmark	○	jslave_tx2_benchmark rengolin bger

Open Source and Upstream

Lab admin & tools:

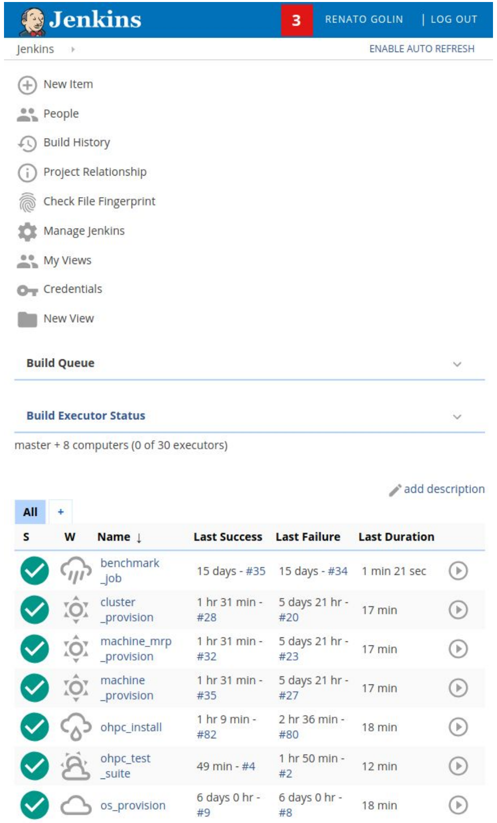
- Jenkins: https://jenkins.io/
- Mr-Provisioner: https://github.com/Linaro/mr-provisioner

Lab Automation:

- https://github.com/Linaro/ans_setup_jenkins
- https://github.com/Linaro/mr-provisioner-role
- https://github.com/Linaro/mr-provisioner-kea-dhcp4-role
- https://github.com/Linaro/ansible-role-mr-provisioner
- https://github.com/Linaro/mr-provisioner-client

HPC-specific automation:

- https://github.com/Linaro/hpc_lab_jenkins
- https://github.com/Linaro/hpc_deploy_benchmarks
- https://github.com/Linaro/benchmark_harness
- https://github.com/Linaro/ansible-playbook-for-ohpc



master + 8 computers (0 of 30 executors)

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓		benchmark_job	15 days - #35	15 days - #34	1 min 21 sec
✓		cluster_provision	1 hr 31 min - #28	5 days 21 hr - #20	17 min
✓		machine_mrp_provision	1 hr 31 min - #32	5 days 21 hr - #23	17 min
✓		machine_provision	1 hr 31 min - #35	5 days 21 hr - #27	17 min
✓		ohpc_install	1 hr 9 min - #82	2 hr 36 min - #80	18 min
✓		ohpc_test_suite	49 min - #4	1 hr 50 min - #2	12 min
✓		os_provision	6 days 0 hr - #9	6 days 0 hr - #8	18 min

Test Suite

Most tests green, however:

- Intel-specific tests (CILK, TBB, IMB) disabled
- Others need package install (PDF, CDF, HDF) but pass when installed
- TAU fails because LMod defaults to openmpi (needs openmpi3)
- Lustre fails as package depends on kernel 4.2 (which won't work on our machines)
- MiniDFT and PRK had make failures, but we haven't investigated yet
- `--enable-long` doesn't really, need to look into why not

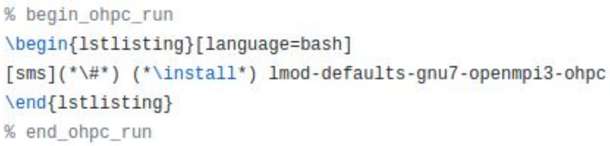
The plan from now on is:

1. Automate package install conditional on enabled tests, fix remaining errors
2. Work with members to prioritise long term ones (like Lustre)
3. Use it for *additional* packages, so we can test them before sending upstream
4. Add a benchmark mode, making sure to use entire cluster

OpenHPC Ansible Automation

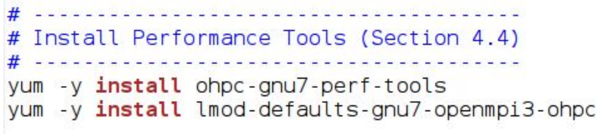
Existing OpenHPC Automation

- A recipe with all rules described in the official documents
- LaTex snippets containing shell code



```
% begin_ohpc_run
\begin{lstlisting}[language=bash]
[sms](*\#*) (*\install*) lmod-defaults-gnu7-openmpi3-ohpc
\end{lstlisting}
% end_ohpc_run
```

- Are converted and merged into a (OS-dependent) bash script



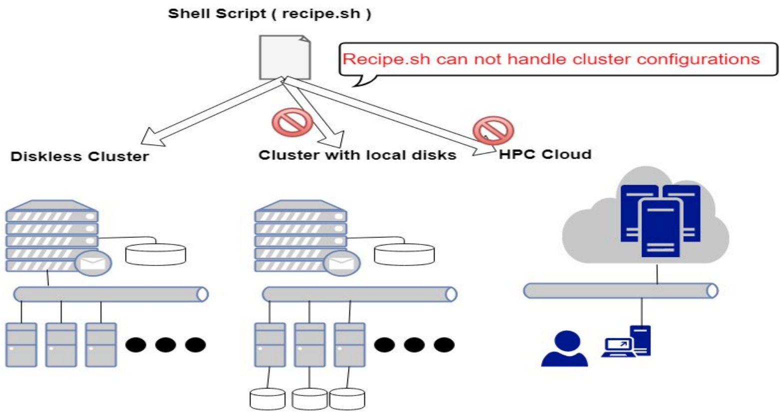
```
# ---------------------------------------
# Install Performance Tools (Section 4.4)
# ---------------------------------------
yum -y install ohpc-gnu7-perf-tools
yum -y install lmod-defaults-gnu7-openmpi3-ohpc
```

- Plus a *input.local* file, with some cluster-specific configuration / environment

Existing OpenHPC Automation

Shortcomings:

- The *input.local* file exports shell variables, and don’t have enough information
- The *recipe.sh* is **not** idempotent
- Extensibility is impossible without editing the files (downstream work)


Shell Script (recipe.sh)
Recipe.sh can not handle cluster configurations
Diskless Cluster
Cluster with local disks
HPC Cloud

Ansible Playbooks

Ansible is a widely used automation tool which can describe the structure and configuration of IT infrastructure with YAML “*playbooks*”.

OpenHPC with Ansible:

- Ansible playbooks can more easily be **idempotent**
- Ansible can manage nodes/tasks according to the the structure of the cluster
- Configuration is passed as a YAML file (no environment handling)
- Composition, using playbooks and roles, building on third-party content

Ansible OpenHPC Recipes

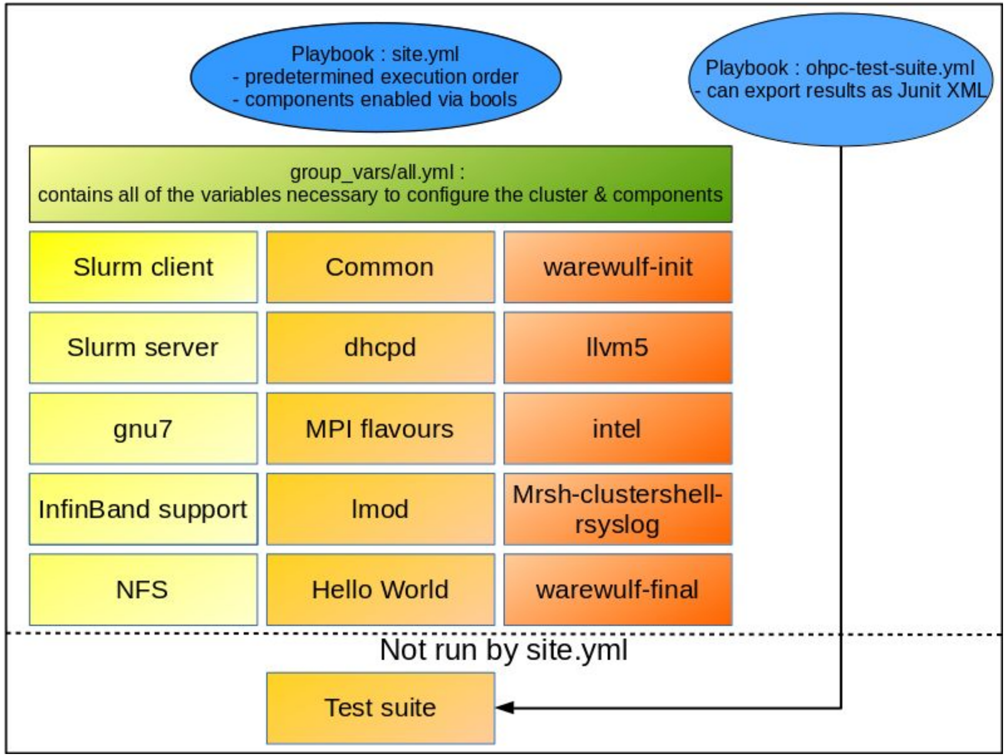
- Flexible cluster configuration
  - Fine grained / composable
  - Cluster wide / node group wide / node specific
- Works on both x86_64 and AArch64
  - Ansible gathers information about architecture
  - Same playbook runs on both
- OS is directly inferred by Ansible (gather_facts)
  - Yum, apt, zypper… can be switched in the roles’ logic

Ansible OpenHPC Recipes

The basic structure of the Ansible playbook

```
playbook
  +---- group_vars/
         +-- all.yml                cluster wide configurations
         +-- group1,group2 … node group(e.g., computing nodes) specific configurations
  +---- host_vars/
         +--- host1,host2         … host specific configurations
  +---- roles/                     package specific tasks, templates, config files, and config variables
         +--- package-name/
                  +--- tasks/main.yml … YAML file to describe installation method of package-name
                  +--- vars/main.yml … package specific configuration variables
                  +--- templates/*.j2 … template files to generate configuration files
```

Ansible OpenHPC Recipes



Playbook : site.yml
- predetermined execution order
- components enabled via bools

Playbook : ohpc-test-suite.yml
- can export results as Junit XML

group_vars/all.yml :
contains all of the variables necessary to configure the cluster & components

Slurm client	Common	warewulf-init
Slurm server	dhcpd	llvm5
gnu7	MPI flavours	intel
InfinBand support	lmod	Mrsh-clustershell-rsyslog
NFS	Hello World	warewulf-final

Not run by site.yml

Test suite

Upstreaming our work

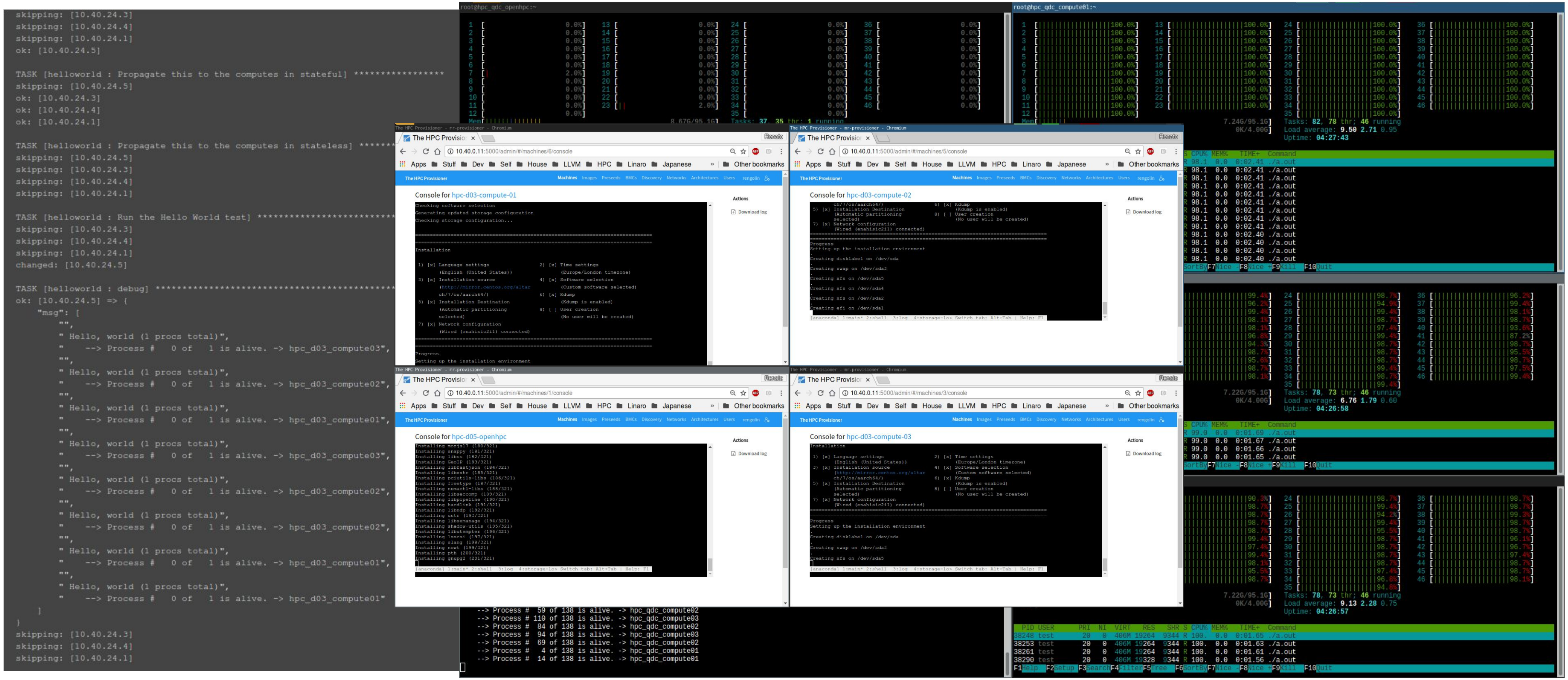
Option #1:

- Generate from LaTex sources at the same time as recipe.sh
  - One bundle per recipe (warewulf/xcat, slurm/pbs, centos/suse, arm/x86)
  - Provide as a zip/tar file on the docs package, like the recipes
- Problems:
  - Still need to install package to get recipes that will install OpenHPC
  - Still don’t have one recipe to rule them all

Option #2:

- Keep as a separate repository, updated in parallel with the doc
  - Easier to integrate to existing automation, update and collaborate
- Problems:
  - Out-of-sync with LaTex sources, can end up meaningless

Results



```
skipping: [10.40.24.3]
skipping: [10.40.24.4]
skipping: [10.40.24.1]
ok: [10.40.24.5]

TASK [helloworld : Propagate this to the computes in stateful] *****************
skipping: [10.40.24.5]
ok: [10.40.24.3]
ok: [10.40.24.4]
ok: [10.40.24.1]

TASK [helloworld : Propagate this to the computes in stateless] *******
skipping: [10.40.24.5]
skipping: [10.40.24.3]
skipping: [10.40.24.4]
skipping: [10.40.24.1]

TASK [helloworld : Run the Hello World test] ***************************
skipping: [10.40.24.3]
skipping: [10.40.24.4]
skipping: [10.40.24.1]
changed: [10.40.24.5]

TASK [helloworld : debug] **********************************************
ok: [10.40.24.5] => {
    "msg": [
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute03",
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute02",
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute01",
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute03",
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute02",
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute02",
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute01",
        "",
        " Hello, world (1 procs total)",
        "    --> Process #   0 of   1 is alive. -> hpc_d03_compute01"
    ]
}
skipping: [10.40.24.3]
skipping: [10.40.24.4]
skipping: [10.40.24.1]
```

```
root@hpc_qdc_openhpc:~
8.67G/95.1G]   Tasks: 37, 35 thr; 1 running
```

```
    --> Process #  59 of 138 is alive. -> hpc_qdc_compute02
    --> Process # 110 of 138 is alive. -> hpc_qdc_compute03
    --> Process #  84 of 138 is alive. -> hpc_qdc_compute02
    --> Process #  94 of 138 is alive. -> hpc_qdc_compute03
    --> Process #  69 of 138 is alive. -> hpc_qdc_compute02
    --> Process #   4 of 138 is alive. -> hpc_qdc_compute01
    --> Process #  14 of 138 is alive. -> hpc_qdc_compute01
```

The HPC Provisioner - mr-provisioner - Chromium

10.40.0.11:5000/admin/#/machines/6/console

Console for hpc-d03-compute-01

```
Checking software selection
Generating updated storage configuration
Checking storage configuration...

================================================================================
================================================================================
Installation

 1) [x] Language settings                 2) [x] Time settings
        (English (United States))                (Europe/London timezone)
 3) [x] Installation source               4) [x] Software selection
        (http://mirror.centos.org/altar          (Custom software selected)
        ch/7/os/aarch64/)                 6) [x] Kdump
 5) [x] Installation Destination                 (Kdump is enabled)
        (Automatic partitioning           8) [ ] User creation
        selected)                                (No user will be created)
 7) [x] Network configuration
        (Wired (enahisic2i1) connected)
================================================================================
================================================================================
Progress
Setting up the installation environment
```

The HPC Provisioner - mr-provisioner - Chromium

10.40.0.11:5000/admin/#/machines/5/console

Console for hpc-d03-compute-02

```
        ch/7/os/aarch64/)                 6) [x] Kdump
 5) [x] Installation Destination                 (Kdump is enabled)
        (Automatic partitioning           8) [ ] User creation
        selected)                                (No user will be created)
 7) [x] Network configuration
        (Wired (enahisic2i1) connected)
================================================================================
================================================================================
Progress
Setting up the installation environment
.
Creating disklabel on /dev/sda
.
Creating swap on /dev/sda3
.
Creating xfs on /dev/sda5
.
Creating xfs on /dev/sda4
.
Creating xfs on /dev/sda2
.
Creating efi on /dev/sda1
[anaconda] 1:main* 2:shell  3:log  4:storage-lo> Switch tab: Alt+Tab | Help: F1
```

The HPC Provisioner - mr-provisioner - Chromium

10.40.0.11:5000/admin/#/machines/1/console

Console for hpc-d05-openhpc

```
Installing mozjs17 (180/321)
Installing snappy (181/321)
Installing libss (182/321)
Installing GeoIP (183/321)
Installing libfastjson (184/321)
Installing libestr (185/321)
Installing pciutils-libs (186/321)
Installing freetype (187/321)
Installing numactl-libs (188/321)
Installing libseccomp (189/321)
Installing libpipeline (190/321)
Installing hardlink (191/321)
Installing libndp (192/321)
Installing ustr (193/321)
Installing libsemanage (194/321)
Installing shadow-utils (195/321)
Installing libutempter (196/321)
Installing lsscsi (197/321)
Installing slang (198/321)
Installing newt (199/321)
Installing pth (200/321)
Installing gnupg2 (201/321)
[anaconda] 1:main* 2:shell  3:log  4:storage-lo> Switch tab: Alt+Tab | Help: F1
```

The HPC Provisioner - mr-provisioner - Chromium

10.40.0.11:5000/admin/#/machines/3/console

Console for hpc-d03-compute-03

```
Installation

 1) [x] Language settings                 2) [x] Time settings
        (English (United States))                (Europe/London timezone)
 3) [x] Installation source               4) [x] Software selection
        (http://mirror.centos.org/altar          (Custom software selected)
        ch/7/os/aarch64/)                 6) [x] Kdump
 5) [x] Installation Destination                 (Kdump is enabled)
        (Automatic partitioning           8) [ ] User creation
        selected)                                (No user will be created)
 7) [x] Network configuration
        (Wired (enahisic2i1) connected)
================================================================================
================================================================================
Progress
Setting up the installation environment
.
Creating disklabel on /dev/sda
.
Creating swap on /dev/sda3
.
Creating xfs on /dev/sda5
[anaconda] 1:main* 2:shell  3:log  4:storage-lo> Switch tab: Alt+Tab | Help: F1
```

```
root@hpc_qdc_compute01:~
 7.24G/95.1G]   Tasks: 82, 78 thr; 46 running
    0K/4.00G]   Load average: 9.50 2.71 0.95
                Uptime: 04:27:43

S CPU% MEM%   TIME+  Command
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.41 ./a.out
R 98.1  0.0  0:02.40 ./a.out
R 98.1  0.0  0:02.40 ./a.out
R 98.1  0.0  0:02.40 ./a.out
R 98.1  0.0  0:02.40 ./a.out
SortByF7Nice -F8Nice +F9Kill  F10Quit
```

```
 7.22G/95.1G]   Tasks: 78, 73 thr; 46 running
    0K/4.00G]   Load average: 6.76 1.79 0.60
                Uptime: 04:26:58

S CPU% MEM%   TIME+  Command
R 99.0  0.0  0:01.69 ./a.out
R 99.0  0.0  0:01.67 ./a.out
R 99.0  0.0  0:01.66 ./a.out
R 99.0  0.0  0:01.65 ./a.out
SortByF7Nice -F8Nice +F9Kill  F10Quit
```

```
 7.22G/95.1G]   Tasks: 78, 73 thr; 46 running
    0K/4.00G]   Load average: 9.13 2.28 0.75
                Uptime: 04:26:57

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
38248 test       20   0  406M 19264  9344 R 100.  0.0  0:01.65 ./a.out
38253 test       20   0  406M 19264  9344 R 100.  0.0  0:01.63 ./a.out
38261 test       20   0  406M 19264  9344 R 100.  0.0  0:01.61 ./a.out
38290 test       20   0  406M 19328  9344 R 100.  0.0  0:01.56 ./a.out
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Thank You!

Linaro

Linaro HPC