Hiter štart v uporabo HPC-RIVR Maister

Quick start user guide HPC-RIVR Maister (Slovenian version)

Aleš Zemljak

February 25, 2020

Outline

HPC-RIVR

HPC-RIVR Članice

Figure: Članice konzorcija HPC RIVR in njihove vloge

SLING

Figure: Slovenska Nacionalna Superračunalniška Mreža

SLING Splet, Primeri, Dostop

- `https://www.sling.si`
- `https://doc.sling.si`

SLING Članice

- UM
  - Maister
- ARNES
  - Jost
- IJS
  - Atos, SiGNET, NSC, CIPKEBIP
- UNG
  - Zorro
- ARCTUR
  - Arctur1, Arctur2 (komercialno)
- KI
  - ARC
- FS UL
  - HPCFS
- FIŠ
  - Rudolf

Zgradba SLING superračunalniške gruče

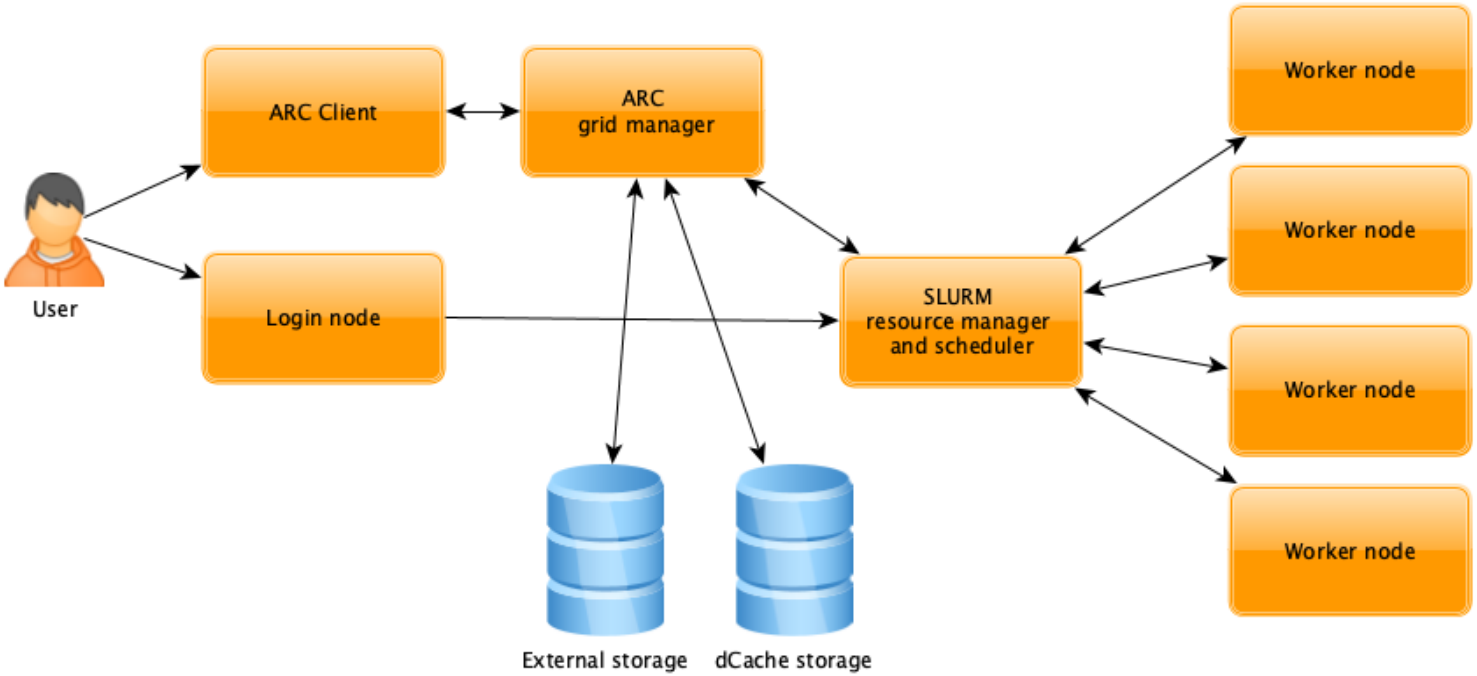


Figure: Komponente SLING superračunalniške gruče

Pravila

Dostop do Maistra

1. Na voljo slovenskim raziskovalnim ustanovam in njihovim raziskovalcem
2. *Ustanove in posamezniki, ki niso del UM, se morajo za neposredni dostop do SLURM-a, prvo povezati v omrežjem UM
   - preko Virtualnega zasebnega omrežja (VPN)

Načini dostopa do superračunalniške gruče

- Vse SLING superračunalniške gruče podpirajo vmesno programje ARC
- Neposreden dostop do SLURM preko vstopnih vozlišč
  1. Maister
     - rmaister.hpc-rivr.um.si

Pridobite svoj račun SLING vzajemne prijave (SSO) za neposreden dostop

- iz službenega naslova (raziskovalne ustanove) pošljite prošnjo na "HPC podporo"
  - `mailto:hpc.podpora@um.si`
- SLING SSO FreeIPA strežnik
  - `https://fido.sling.si`

Zgradba, struktura in sistemska PO gruče Maister

Omrežja

- 100Gbit/s Ethernet omrežje
- 100Gbit/s Infiniband omrežje (z nizko zakasnitvijo)
  - samo omejeno število vozlišč (dpcn in gpu)

Splošnonamenski strežniki

- glavno vozlišče (head node)
  - upravljanje, SLURM, ARC
- strežniki virtualizacijskih okolij
- virtualni strežniki
  - vstopna vozlišča - rmaister.hpc-rivr.um.si
    - SSH dostop za uporabnike gruče
  - idr.

Zgradba, struktura in sistemska PO gruče Maister

Delovna vozlišča

- Fedora Core 30
- dostop možen samo preko SLURM vrste

Zgradba, struktura in sistemska PO gruče Maister

Delovna vozlišča

- Fedora Core 30
- dostop možen samo preko SLURM vrste

Običajna - cn[01-48]

- "samo" 100Gbit/s Ethernet povezava

Dvojno povezana - dpcn[01-28]

- 100Gbit/s Ethernet povezava
- 100Gbit/s Infiniband povezava

Zgradba, struktura in sistemska PO gruče Maister

Delovna vozlišča

- Fedora Core 30
- dostop možen samo preko SLURM vrste

Grafična vozlišča - gpu[01-06]

- 4x nVidia Tesla V100 32G
  - 5120 jeder
  - 32GB pomnilnika
- 100Gbit/s Ethernet povezava
- 100Gbit/s Infiniband povezava

Zgradba, struktura in sistemska PO gruče Maister

Delovna vozlišča

- Fedora Core 30
- dostop možen samo preko SLURM vrste

Grafična vozlišča - gpu[01-06]

- 4x nVidia Tesla V100 32G
  - 5120 jeder
  - 32GB pomnilnika
- 100Gbit/s Ethernet povezava
- 100Gbit/s Infiniband povezava

Strežniki za podatkovno shrambo

- Ceph gruča za podatkovno shrambo
  - domači direktoriji - `/ceph/grid/home/<username>`

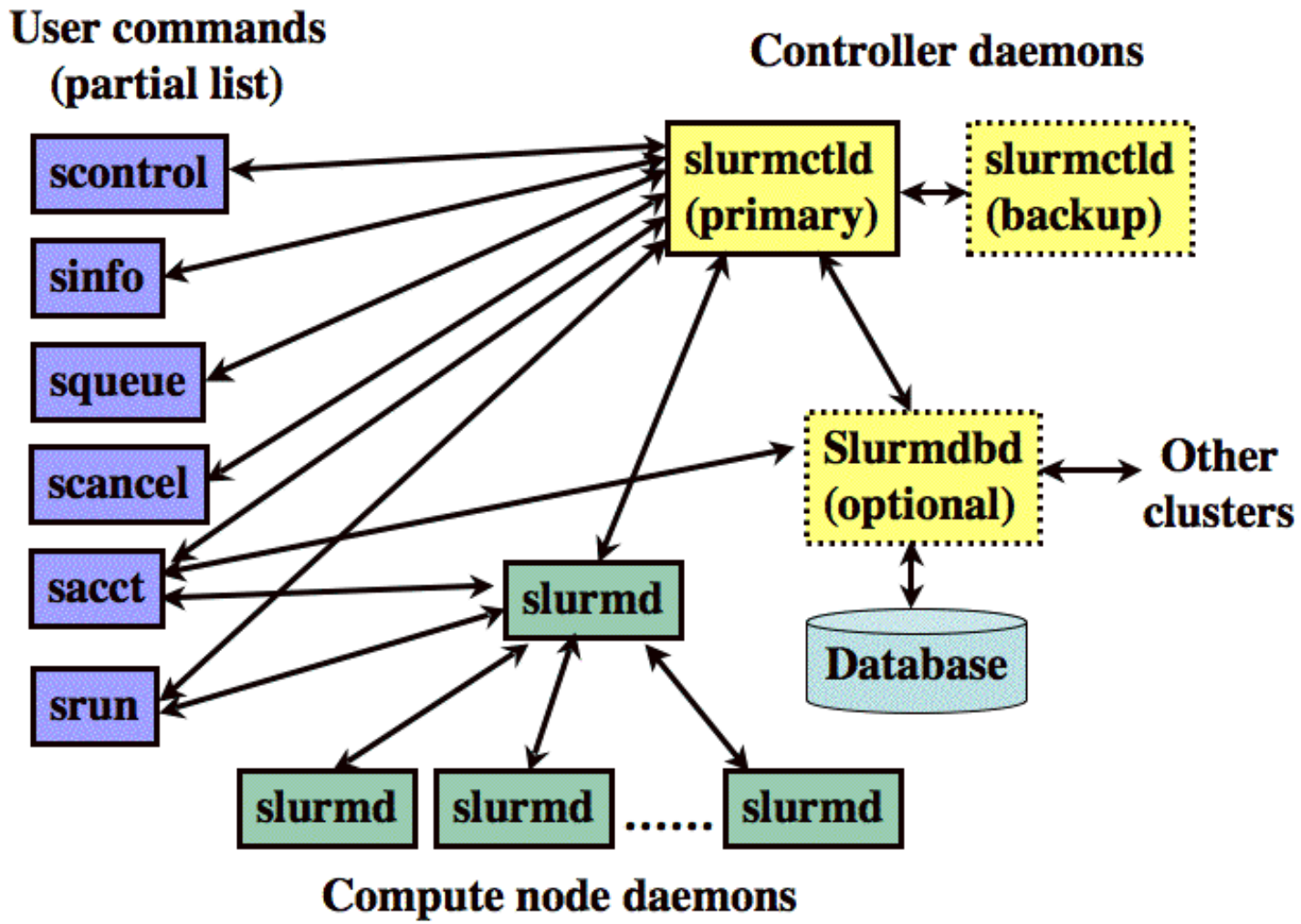
SLURM - Simple Linux Utility for Resource Management

- SLURM je upravljalec virov in razporejevalnik nalog/poslov v enem
- posebej načrtovan za Linux gruče
- razporeja posle na osnovi "pravične delitve" (ang.: fairshare), kvalitete storitve (QoS), starosti posla, velikosti posla, potrebovanih virov itn.
- podpira gručo do velikosti 3 milijonov jeder
- toleranten do napak, prenosljiv, skalabilen,
- podpira več kot 100 vtičnikov za avtentikacijo, MPI, knjiženje, omrežno topologijo, oddajo poslov idr.
- ter je odprto koden

Izvorna koda SLURM

- git://github.com/SchedMD/slurm.git

Glavne komponente SLURM-a



SLURM dostop

- Uporabniki lahko dostopajo do lokalne (npr. Maistrove) SLURM (čakalne) vrste preko vstopnega vozlišča (ang.: Login node)
- Dostop do vstopnih vozlišč omogoča veljaven račun SLING vzajemne prijave (SSO)
  - (FreeIPA strežnik: `https://fido.sling.si`)
- Dostop je dovoljen uporabnikom šolskih, raziskovalnih, izobraževalnih ustanov, gospodarskega razvoja ipd.
- Obstajajo dostopi z različnimi privilegiji/prioritetami:
  - osebni dostop, projektni dostop, rezervacije za delavnice

SLURM (preko vstopnih vozlišč)

- SSH povezava do vstopnega vozlišča
  - naslov za Maister*: rmaister.hpc-rivr.um.si
- prijava s podatki računa SLING vzajemne prijave (SSO)

*Dostop do rmaistra zunaj UM

- Vstopno vozlišče je dostopno samo znotraj omrežja UM
- *Ustanove in posamezniki, ki niso del UM, se morajo za neposredni dostop do SLURM-a, prvo povezati v omrežjem UM
  - preko Virtualnega zasebnega omrežja (VPN)

Navodila za VPN

- IT storitve za zaposlene UM -> Povezljivost -> VPN
- `https://it.um.si/zaposleni/Strani/default.aspx`

SSH - Secure SHell

- je protokol za varen dostop in izvajanje ukazov na oddaljenih sistemih
  - uporablja kriptiran komunikacijski kanal
- omogoča dostop do ukazne lupine oddaljenega sistema
- omogoča preusmeritev oddaljenih vrat na lokalna (port redirection)
- omogoča posredovanje grafičnih elementov iz oddaljenega sistema na lokalnega (X11 forwarding)
- idr.

SSH odjemalci

CLI - ukazno-vrstični vmesnik

- POSIX okolja (Linux, Unix, MacOS X, Cygwin, Msys*, Git (bash) for Windows)
  `ssh <uporabnisko_ime>@rmaister.hpc-rivr.um.si`
- Powershell
  `ssh <uporabnisko_ime>@rmaister.hpc-rivr.um.si`

GUI - grafični uporabniški vmesnik

- Windows
  - PuTTY: `https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html`
  - idr.

BASH - Bourne Again SHell

- privzeta ukazna lupina večine Linux distribucij, MacOS X idr.
- privzeta ukazna lupina na vstopnem vozlišču `rmaister`

Nekaj osnovnih ukazov

- `pwd` - direktorij, v katerem se trenutno nahajate
- `cd <pot>` - spremembe trenutnega direktorija
- `ls [-alF] [<pot/direktorij>]` - prikaz datotek trenutnega direktorija [ali podane poti/direktorija]
- `man <ukaz/vsebina>` - osnovni dokumentacijski sistem
  - `man ls` - prikaže dokumentacijo ukaza `ls`

SLURM - (vrsta) particije in stanje delovnih vozlišč

- `squeue -l` - prikaže podrobnosti o poslih (`-l` = long)
- `squeue -u $USER` - prikaže (vaše) posle uporabnika
- `squeue -p grid` - prikaže posle particije z imenom `grid`
- `squeue -t PD` - prikaže čekajoče posle
- `sinfo` - prikaže stanje particij
- `sinfo -l -N` - prikaže podrobno stanje particij
- `sinfo -T` - prikaže rezervacije
- `scontrol show nodes` - prikaže informacije o vozliščih

Oddaja poslov s SLURM-om

Uporabljajo se 3 ukazi:

- `sbatch` - se uporablja za oddajo skripte poslov za kasnejše izvajanje. Skripta lahko vsebuje več `srun` ukazov za zagon poslov. Pri oddaji posla, dobite njegovo identifikacijo (ID - številko)
- `salloc` - se uporablja za dodeljevanje virov nekemu poslu v času izvajanja.
  Praviloma se uporabi, da se ustvari lupina, ki se nato uporabi za izvedbo zagona posla s `srun`.
- `srun` - se uporablja za oddajo posla v izvajanja v trenutku izvedbe ukaza.
  Posel lahko vsebuje več korakov (ang.: steps), ki se izvedejo zaporedno ali vzporedno, na neodvisnih ali skupnih virih v dodeljevanju posla vozlišču.
  Navadno se uporablja v kombinaciji z `sbatch` ali `salloc`.

V kolikor ste navajeni na drug sistem paketne obdelave ukazov, preverite slednje na naslovu: `https://slurm.schedmd.com/rosetta.pdf` in jih primerjajte.

Spremljanje stanj s SLURM-om

- `sacct`: podaja informacije knjiženj (npr. `sacct -j <JOBID>`)
- `sinfo`: podaja informacije o particijah
- `squeue`: podaja informacije o stanju vrste oz. o poslih v vrsti (npr. `squeue --user=<USERNAME>`)
- `sstat`: podaja statistiko o izbranem poslu (npr. `sstat -j <JOBID> --format=AveCPU,AveRSS,AveVMSize,MaxRSS,MaxVMSize` - prikaže izkoriščenost tekočega posla na sistemske nivoju)
- `scontrol show`: npr. `scontrol show job|partition`
- `scontrol update`: omogoča spreminjanje vašega posla
- `scontrol hold`: zaustavi izbran posel
- `scontrol release`: izpusti izbran posel
- `scancel <JOBID>`: preklicati izbran posel

SLURM - Primer enostavnega posla

Sprva pridobite informacije o vozliščih in vrstah:

```
sinfo
```

PARTITION	AVAIL	TIMELIMIT	NODES	STATE	NODELIST
grid*	up	2-00:00:00	3	maint	cn40,dpcn28,gpu0
grid*	up	2-00:00:00	3	resv	gpu[01-02,04]
grid*	up	2-00:00:00	29	mix	cn[02,04,07,09-10,
grid*	up	2-00:00:00	46	alloc	cn[01,03,05-06,08,
grid*	up	2-00:00:00	1	idle	gpu03
long	up	14-00:00:0	8	mix	cn[41-48]

- Izvedite enostaven posel, ki priskrbi ime vozlišča na katerem se je posel izvede

  ```
  srun hostname
  ```

SLURM posel z sbatch

Shrani to skripto v datoteko, da bomo jo oddali kot posel s pomočjo ukaza sbatch.

```
#!/bin/bash
#SBATCH --job-name=test
#SBATCH --output=result.txt
# zahteve: single core, walltime 10 minut, 100MB spomina
#SBATCH --ntasks=1
#SBATCH --time=10:00
#SBATCH --mem-per-cpu=100
# zažene ukaz 'hostname'
srun hostname
# zažene 60 sekundni spanec
srun sleep 60
```

Oddajte svoj posel z ukazom:
`sbatch simple.batch.sh` ali npr. z
`sbatch -N4 -n8 simple.batch.sh`

SLURM in MPI (1)

SLURM podpira 3 načine delovanje, ki jih uporabljajo MPI implementacije:

1. Neposreden zagon preko PMI2 ali PMIx API-ja
2. SLURM dodeli vire ter `mpirun` zažene posle z uporabo SLURM infrastrukture
3. SLURM dodeli vire, `mpirun` zažene posle preko SSH/RSH (brez SLURM krmiljenja)
   - `#SBATCH --ntasks=16` bo zagnal 16 "MPI rank"-ov
   - `#SBATCH --cpus-per-task=1` bo uporabil 1 jedro na "rank"
   - `#SBATCH --ntasks-per-socket=8` vsaka vtičniča (socket) bo imela 8 poslov
   - `#SBATCH --nodes=1` bo zagnalo posel na enem samem vozlišču

Najboljša praksa

Pri uporabi neposrednega dostop do SLURM-a,
zaženite MPI posle z `srun` in `pmi` (npr.: `srun --mpi=pmi2`)

SLURM in MPI (2)

Table: Kompatibilnost OpenMPI in PMIx

Različica OpenMPI	Najboljša praksa
OpenMPI <= 1.X	Podpira PMI, a ne PMIx (Container in gostitelj morata uporabljati popolnoma enaki različici OpenMPI/PMI
2.X <= OpenMPI < 3.X	Podpira PMIx 1.X
3.X <= OpenMPI < 4.X	Podpira PMIx 1.X in 2.X
OpenMPI >= 4.X	Podpira tudi PMIx 3.X

Podprti API-ji

Preverite z ukazom `srun --mpi=list`

Primer SLURM MPI posla - priprava/prevajanje

- Sprva potrebujemo MPI knjižnice
  - Preverite module, ki so na voljo z ukazom `module avail`
  - Naložite MPI modul, npr. z ukazom `module load mpi`
- Za primer skopirajmo 'hellompi' primer iz Wikipedie: `http://en.wikipedia.org/wiki/Message_Passing_Interface#Example_program`
- Prevedite z ukazom:
  mpicc wiki-mpi-example.c -o hello.mpi

Primer SLURM MPI posla - zagon z sbatch skripto

```
#!/bin/bash
#
#SBATCH --job-name=test-mpi
#SBATCH --output=result-mpi.txt
#
#SBATCH --ntasks=4
#SBATCH --time=10:00
#SBATCH --mem-per-cpu=100
module load mpi
srun --mpi=pmix hello.mpi
```

Figure: sbatch skripta za zagon primera

Prenos podatkov na in z gruče

SFTP protokol

- SFTP je FTP-ju podoben protokol za prenos podatkov preko SSH varne seje
- uporaba iste uporabniške prijave kot za SSH

SFTP odjemalci

- CLI
  - `sftp <uporabnik>@rmaister.hpc-rivr.um.si` - v POSIX okoljih (Linux, Unix, MacOS X)
- GUI
  - FileZilla
    `https://filezilla-project.org/`
  - WinSCP (samo za Windows)
    `https://winscp.net/eng/index.php`

Prednosti container-jev

- prenosljivost kode
- operacijska izolacija z imenskimi prostori ter omejevanje porabe virov s `cgroups`
- ustvariti container zahteva administratorske pravice, a uporaba containterjev jih ne
- minimalna ali neopazna degradacija zmogljivosti, če sploh
- dovoljuje gradnjo prilagojenih okolij (OS, aplikacije, knjižnice)
- konsistentno delovanje

Singularity containerji

- minimalna degradacija zmogljivosti, če sploh
- uporabnik znotraj container-ja = uporabnik na gostitelju
- ne temelji na Docker, vendar lahko neposredno poganja Docker slike
- HPC usmerjen
- predpripravljene slike je možno uporabiti iz različnih razpečevalnikov
  - Docker hub: `https://hub.docker.com/`
  - Singularity hub: `https://singularity-hub.org`
  - Nvidia cloud: `https://www.nvidia.com/en-us/gpu-cloud/`
  - Quay: `https://quay.io/search/`

Ukaz za prevzem

- `singularity pull <hub>://<image>[:<tag>]`

Dokumentacija

`https://sylabs.io/docs`

Singularity - delovni potek

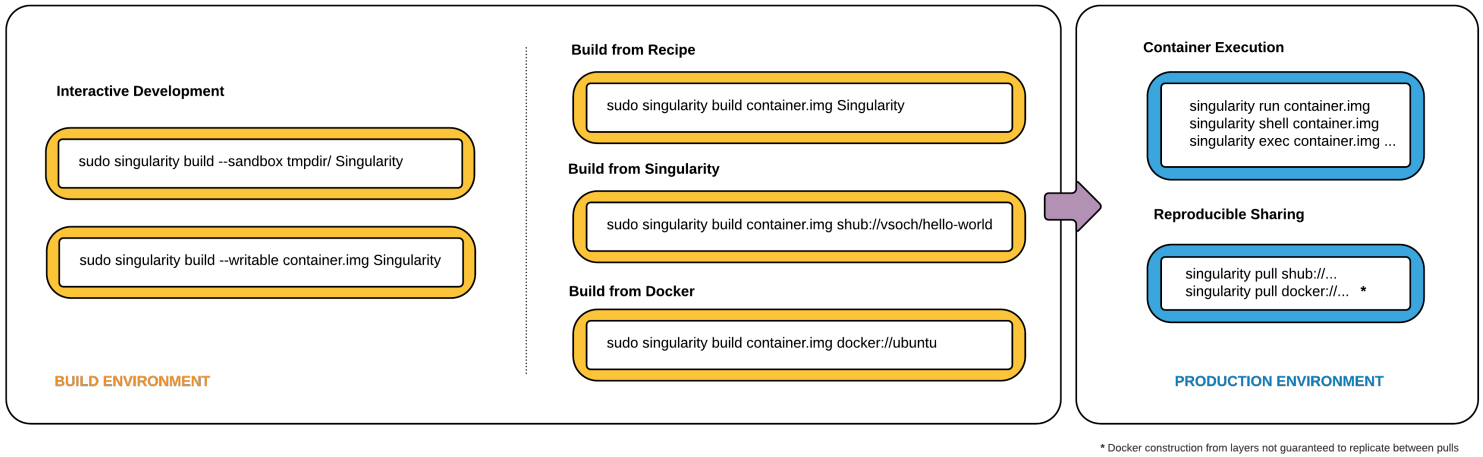


Figure: Delovni potek Singularity

Vir: `https://singularity.lbl.gov`

Singularity ukazi

- pridobiti ukazno lupino v container-ju: `singularity shell`
  `singularity shell docker://ubuntu:latest`
- zagnati ukaz v container-ju: `singularity exec`
  `singularity exec docker://ubuntu:latest cat /etc/lsb-release`
- zagnati privzet ukaz v container-ju: `singularity run`
  `singularity run centos.sif`
  `./centos.sif`
- preučiti okolje, izvajalno območje, labele: `singularity inspect`
  `singularity inspect --runscript centos.sif`
- prevzeti/prenesti container: `singularity pull`
  `singularity pull docker://ubuntu:latest`

Singularity build

- ali pretvorite Docker container in ga prilagodite
  ali pa zgradite container po lastni definiciji(/-ijski datoteki)
- potrebujete administracijske pravice (ali uporabite –fakeroot)

```
sudo singularity build centos8.sif centos8.def
sudo singularity build centos8.sif docker://centos:latest
```

Singularity kot SLURM posel

```
#!/bin/bash
#SBATCH -J singularity test
#SBATCH -o singularity test.out
#SBATCH -e singularity test.err
#SBATCH -p gridlong
#SBATCH -t 0-00:30
#SBATCH -N 1
#SBATCH -c 1
#SBATCH --mem=4000
# Singularity command line options
singularity exec centos7.sif cat /etc/os-release
```

Figure: sbatch skripta za zagon singularity

Singularity in MPI

- v prihodnje

Sigularity in Infiniband

- v prihodnje

Singularity in GPU

- v prihodnje

Singularity povezave

- Singularity Dokumentacija: `https://www.sylabs.io/docs/`
- Singularity GitHub: `https://github.com/singularityware/singularity`
- Singularity v Google Grupah: `https://groups.google.com/a/lbl.gov/forum/#!forum/singularity`
- Docker dokumentacija: `https://docs.docker.com/`

Viri

- HPC RIVR: `https://www.hpc-rivr.si/`
- SLING: `https://www.sling.si/sling/`
- SiGNET CA: `https://signet-ca.ijs.si/`
- SLURM: `https://slurm.schedmd.com/`

Neposredno zaslužni za dokument

UM

- izr. prof. Dr. Miran Ulbin

SLING

IJS

- mag. Barbara Krašovec

Posredno zaslužni za dokument

UM

- red. prof. Dr. Zoran Ren, projektni vodja HPC-RIVR
- Dr. Izidor Golob, vodja RCUM

SLING

IJS

- Jan Jona Javoršek, predsednik SLING
- doc. Dr. Andrej Filipčič
- Dejan Lesjak

Vprašanja in odgovori

- https://www.hpc-rivr.si
- https://www.sling.si
- https://doc.sling.si

Podpora strokovnih sodelavcev HPC-RIVR

- mailto:hpc.podpora@um.si