

Python, pip i conda

Sadržaj

- Lustre i virtualna okruženja
 - Kako Lustre radi i kako ga pravilno koristiti
 - Python i virtualna okruženja
- Apptainer i python
 - Izgradnja pipom
 - Izgradnja pipom + definicijska datoteka
 - Izgradnja condom
 - Izgradnja condom + online repozitoriji
- Kako dalje?

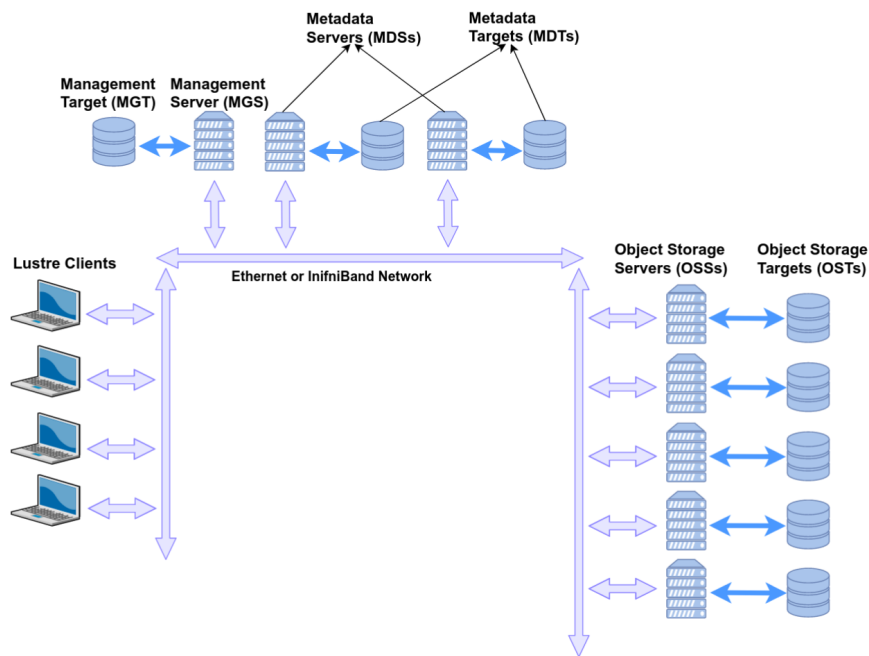
Lustre i virtualna okruženja

[Lustre](#) je paralelni raspodijeljeni datotečni sustav koji koristi Supek, namijenjen okruženju HPC u kojem veliki broj korisnika generira i koristi iznimnu količinu podataka, i čija je visoka dostupnost i brzina prijenosa bitna radi što efikasnijeg izvođenja paralelnih aplikacija.

Način na koji ovo postiže je fizičkim razdvajanjem opisa datotečnog sustava (tzv. [namespacea](#)) od njegovog stvarnog sadržaja (u [objektnom obliku](#)) koji je na Supeku pohranjen na stotinjak [SSD-ova](#) u tehničkoj izvedbi [ClusterStor E1000](#).

Lustre datotečni sustav se sastoji od nekoliko glavnih komponenti (s pripadnim dijagramom ispod):

- **Metadata Server** - Poslužitelji za upravljanje pohranjenim podacima s informacijama poput njihovog imena, vlasništva i prava pristupa
- **Object Storage Server** - Poslužitelji na kojima se podaci fizički nalaze i koji se mogu proizvoljno skalirati
- **Management Server** - Poslužitelji koji su odgovorni za nadzor i upravljanje cjelokupnim datotečnim sustavom Lustre
- **Lustre Networking** - Brza i visoko propusna veza kojom se podaci prenose
- **Client** - Mount point na pristupnim poslužiteljima koji otkriva datotečni sustav Lustre korisničkim aplikacijama



Slika 1. Dijagram datotečnog sustava Lustre (Figure 1 u izvornoj publikaciji)

Kako Lustre radi i kako ga pravilno koristiti

Pri svakoj datotečnoj operaciji čitanja ili pisanja, klijent šalje zahtjev **Metadata Serveru** na kojem se nalazi virtualni zapis opisa i lokacija pravog podatka raspodijeljenog na više **Object Storage Servera**. Jednom kada se tražena datoteka (ili datoteke) pronađu, stvara se direktna veza između klijenta i fizičkog zapisa, koja osigurava pristup i njeno daljnje upravljanje.

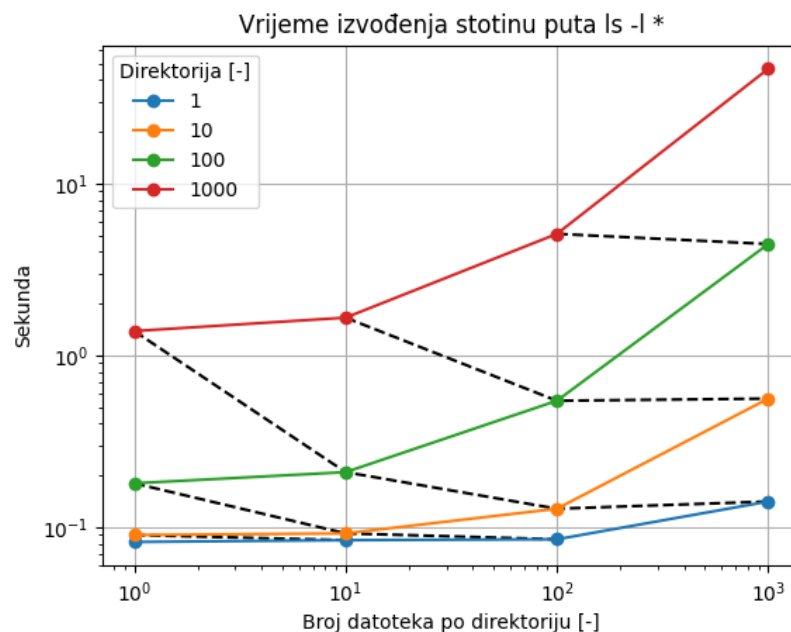
U višekorisničkom okruženju poput klastera Supek, pristup i upravljanje podacima mora biti usklađeno između svake korisničke aplikacije koja im pristupa. Ovo se postiže naizmjeničnim osvježavanjem i usklađivanjem virtualnog zapisa koje ima svoje granice optimalnog izvođenja, iznad kojeg se performanse drastično smanjuju za cijeli datotečni sustav kojim se upravlja i sve korisnike koji ga koriste.

Neke od preporuka za Lustre dijeljeni datotečni sustav uključuju:

- štedljivo korištenje naredbi za opis datotečnog sustava poput `ls`, `find`, `du` ili `df`
- izbjegavanje osobnog prevođenja i instalacije aplikacija
- izbjegavanje pokretanja izvršnih datoteka s datotečnog sustava Lustre
- izbjegavanje direktorija s velikim brojem datoteka (optimalno je manje od tisuću) ili datotekama malog obujma (optimalno više od 1GB)

Ispod se nalazi primjer čitanja sadržaja direktorija komandom `ls -l *` stotinu puta zaredom (što je tipično opterećenje jednog klastera) nad raznim kombinacijama broja direktorija i datoteka koje zajedno sadrže 10GB podataka.

Ako uzmemo u obzir dva rubna slučaja: 1) tisuću datoteka u jednom direktoriju naspram 2) tisuću direktorija s jednom datotekom, vidljivo je da se množenjem direktorija efikasnost ove operacije značajno smanjuje (approx. 20 puta). Slični trend vidljiv je i u ostalim kombinacijama (10/100 vs. 100/10, itd.) što upućuje na nužnost agregacije podataka u manji broj direktorija i idealno datoteke većeg obujma.



Slika 2. Vrijeme izvođenja komande `ls -l *` stotinu puta nad kombinacijama broja direktorija (1-1000) i datoteka (1-100) koje zajedno sadrže 10GB podataka. Crne crtane linije povezuju točke s istim brojem datoteka.

Python i virtualna okruženja

Python knjižnice danas se većinom instaliraju korištenjem aplikacija `pip` ili `conda`; upraviteljima knjižnica koji osiguravaju dopremanje svih ovisnosti potrebnih za instalaciju i razvoj aplikacija python.

Iako ove aplikacije pružaju veoma jednostavno i efikasno okruženje za brzi razvoj i eksperimentiranje raznih kombinacija knjižnica, svakom novom instalacijom broj datoteka se multiplicira i dodatno opterećuje dijeljeni sustav (učestalim čitanjem i pisanjem pri razvoju ili izvršavanju).

Ispod se nalazi primjer okruženja nastalog pip instalacijama za **samo jednu verziju pythona**, koje u sebi sadrži tipični **data stack** u kojem se nalazi (approx.):

- 4000 direktorija
- 15 datoteka po direktoriju

- 2G podataka

Ako pretpostavimo slična ubrzanja iz prethodnog dijagrama, Lustre datotečni sustav možemo potencijalno koristiti i do **deset puta efikasnije** (ili barem jedan značajan dio njegove funkcionalnosti) ako okupimo podatke u jednu veću, zasebnu cjelinu.

```
# broj direktorija
[korisnik@kompjuter:] $ find ~/.local/lib/python3.9/site-packages /usr/local/lib/python3.9/dist-packages -type
d | wc -l
4338

# broj datoteka
[korisnik@kompjuter:] $ find ~/.local/lib/python3.9/site-packages /usr/local/lib/python3.9/dist-packages -type
f | wc -l
47355

# veličine
[korisnik@kompjuter:] $ du -hcs ~/.local/lib/python3.9/site-packages /usr/local/lib/python3.9/dist-packages
939M      /home/marko/.local/lib/python3.9/site-packages
747M      /usr/local/lib/python3.9/dist-packages
1.7G      total
```

Apptainer i python



Primjeri izgradnje

Upute ispod pretpostavljaju da kontejner gradite na osobnom računalu.

U slučaju da niste u mogućnosti graditi kontejnere na svojem računalu, upute za izgradnju na Supeku možete naći na [našem wikiju](#).

Jedan od sve ustaljenijih načina dopremanja aplikacija na HPC klastere su [apptainer](#) i [singularity](#); sučelja za stvaranje izoliranih razvojnih okolina zvanih *kontejneri*.

Kontejneri su datoteke koje u sebi sadrže aplikacije i njihove ovisnosti potrebne za izvršavanje u formi [slike \(engl. image\)](#) koja se stvara na osobnom računalu i, jednom kada se pripremi, doprema na superračunalo i koristi kao bilo koja druga aplikacija.

Upravo zbog činjenice da *image* sadrži sve ovisnosti i strukturu direktorija unutar **samo jedne** datoteke je ovaj način najpoželjniji za rad na Lustreu.

Ispod se nalaze upute za razvoj osnovnog [python data stacka](#) koji se sastoji od knjižnica:

- [NumPy](#) - upravljanje matričnim podacima
- [SciPy](#) - osnovne znanstvene funkcije
- [Pandas](#) - upravljanje strukturiranim podacima
- [matplotlib](#) - vizualizacija podataka

Detaljnije upute možete naći na [službenim stranicama](#) i [našem wikiju](#).

Izgradnja pipom

Prvi korak je stvaranje osnovnog kontejnera `data_stack_sandbox` u interaktivnom modu ili verziji *sandbox*:

```
# izgradnja sandbox verzije
[korisnik@kompjuter:~] $ aptainer build --sandbox data_stack_sandbox docker://ubuntu:20.04
...
INFO:    Creating sandbox directory...
INFO:    Build complete: data_stack_sandbox

# sadržaj trenutnog direktorija
[korisnik@kompjuter:~] $ ls -l
total 154520
drwxr-xr-x 18 korisnik korisnik      4096 svi   23 16:33 data_stack_sandbox
drwxr-xr-x 18 korisnik korisnik      4096 svi   23 09:51 ubuntu_20.04
-rw-r--r--  1 korisnik korisnik       119 svi   23 15:17 ubuntu_20.04.def
-rwxr-xr-x  1 korisnik korisnik 158208000 svi   23 15:24 ubuntu_20.04.sif
```

Nakon stvaranja interaktivne *sandbox* verzije, otvorimo ljusku unutar kontejnera korištenjem sudo ovlasti i instaliramo sve ovisnosti korištenjem [pip installa](#):

```
# interaktivna sjednica u kontejneru
[korisnik@kompjuter] $ sudo aptainer shell --writable data_stack_sandbox/
INFO:    /etc/singularity/ exists; cleanup by system administrator is not complete (see https://apptainer.org/docs/admin/latest/singularity_migration.html)
WARNING: Skipping mount /etc/localtime [binds]: /etc/localtime doesn't exist in container
Apptainer> apt update
...
Reading state information... Done
10 packages can be upgraded. Run 'apt list --upgradable' to see them.

# instalacija pip3
Apptainer> apt install python3-pip -y
...
Running hooks in /etc/ca-certificates/update.d...
done.

# instalacija python knjižnica
Apptainer> pip3 install numpy scipy pandas matplotlib ipython
...
Successfully installed asttokens-2.2.1 backcall-0.2.0 contourpy-1.0.7 cycycler-0.11.0 decorator-5.1.1 executing-1.2.0 fonttools-4.39.4 ipython-8.13.2 jedi-0.18.2 kiwisolver-1.4.4 matplotlib-3.7.1 matplotlib-inline-0.1.6 numpy-1.24.3 packaging-23.1 pandas-2.0.1 parso-0.8.3 pexpect-4.8.0 pickleshare-0.7.5 pillow-9.5.0 prompt-toolkit-3.0.38 ptyprocess-0.7.0 pure-eval-0.2.2 pygments-2.15.1 pyparsing-3.0.9 python-dateutil-2.8.2 pytz-2023.3 scipy-1.10.1 six-1.16.0 stack-data-0.6.2 traitlets-5.9.0 tzdata-2023.3 wcwidthWARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
h-0.2.6

# ispis verzija knjižnica
Apptainer> pip3 freeze
...
ipython==8.12.2
matplotlib==3.7.1
matplotlib-inline==3.7.1
numpy==1.24.3
pandas==2.0.1
scipy==1.10.1
...
```

Nakon instalacije knjižnica, prebacujemo *sandbox* direktorij `data_stack_sandbox` u *image* verziju `data_stack.sif` i dostavljamo ju na Supeka:

```
# prebacivanje sandbox verzije u image
[korisnik@kompjuter:~] $ sudo apptainer build data_stack.sif data_stack_sandbox/
...
INFO:      Build complete: data_stack.sif

# velicine verzija sandbox i image
[korisnik@kompjuter:~] $ du -hcs data_stack*
832M      data_stack_sandbox
272M      data_stack.sif
1.1G      total

# dopremanje image verzije na Supek
[korisnik@kompjuter:~] $ scp data_stack.sif mkvakic@login-gpu.hpc.srce.hr:.
data_stack.
sif
100% 272MB 111.0MB/s 00:02m
```

Nakon spajanja na superračunalo Supek, kontejner možemo koristiti pozivom naredbe `apptainer exec`:

```
# login na pristupni poslužitelj gpu
[korisnik@kompjuter:~] $ ssh korisnik@login-gpu.hpc.srce.hr
Last login: Wed May 24 08:16:00 2023 from x.x.x.x

# sadržaj korisnikog direktorija
[korisnik@kompjuter:~] $ ls -l
total 278088
-rwxr-xr-x 1 mkvakic hpc 284729344 May 24 08:02 data_stack.sif

# pokretanje dopremljenog imagea na Supeku
[korisnik@kompjuter:~] $ apptainer exec data_stack.sif python3 --version
Python 3.8.10
```

Izgradnja pipom + definicijska datoteka

Prethodni koraci lokalne izgradnje mogu se lokalno zapisati u `.def` datoteci, što omogućuje dosljedniju instalaciju u slučaju nadogradnje, ali i direktno pozivanje *imagea* putem poglavlja [%runscript](#)

Koraci koji se nalaze ispod su:

- ispisivanje definicijske datoteke (linija 1)
 - `%post` poglavlje s koracima instalacije (linija 5)
 - `%runscript` poglavlje s zadanom izvršnom datotekom `python3` (linija 10)
- kreiranje *imagea* (linija 13)
- dopremanje na Supeka (linija 18)
- direktno izvršavanje *imagea* (linija 24)

```
# ispis definicijske datoteke
[korisnik@kompjuter:~] $ cat data_stack.def
Bootstrap: docker
From: ubuntu:20.04

%post
    apt update
    apt install python3-pip -y
    pip3 install numpy scipy pandas matplotlib ipython

%runscript
    exec python3 $@

# izgradnja imagea
[korisnik@kompjuter:~] $ apptainer build data_stack.sif data_stack.def
...
INFO:      Creating SIF file...
INFO:      Build complete: data_stack.sif

# dopremanje na Supeka
[korisnik@kompjuter:~] $ scp data_stack.sif mkvakic@login-gpu.hpc.srce.hr:.
data_stack.sif      100% 335MB 110.9MB/s   00:03

# login na pristupni poslužitelj gpu
[korisnik@kompjuter:~] $ ssh mkvakic@login-gpu.hpc.srce.hr
Last login: Wed May 24 09:18:44 2023 from x.x.x.x

# direktno izvršavanje imagea na Supeku
[korisnik@x3000c0s27b0n0] $ ./data_stack.sif --version
Python 3.8.10
```

Izgradnja condom

U slučaju da želimo specifičnu verziju pythona i njegovih knjižnica, možemo koristiti [miniforge verziju conde](#); upravitelj python paketima koji stvara virtualna okruženja.

Prvi korak je izgradnja *sandbox* verzije i instalacija mamba korištenjem [službenih uputa](#):

```
# izgradnja sandbox verzije
[korisnik@kompjuter:~] $ sudo apptainer build --force --sandbox data_stack_sandbox docker://ubuntu:20.04
...

# otvaranje interaktivne sjednice u sandbox kontejneru
[korisnik@kompjuter:~] $ sudo apptainer shell --writable data_stack_sandbox/

# osvježavanje apt repozitorija
Apptainer> apt update
...

# instalacija curl
Apptainer> apt install curl -y
...

# preuzimanje Miniforge conde
Apptainer> curl -L -O https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-x86_64.sh
...
100 82.9M 100 82.9M    0    0 30.7M    0 0:00:02 0:00:02 --:--:-- 42.0M

# batch instalacija u /usr/local/miniforge
Apptainer> bash Miniforge3-Linux-x86_64.sh -b -p /usr/local/miniforge
...
installation finished.
```

Ispod se nalaze detalji za dopremanje [python verzije v3.10](#) i prethodno navedenih python knjižnica.

Koraci koji su potrebni su opisani ispod, dok se detaljnije upute nalaze na [službenim stranicama](#). Instalacija se sastoji od:

- inicijalizacije conde (linija 1)
- kreiranja virtualnog okruženja data_stack (linija 3)
- aktivacije virtualnog okruženja data_stack (linija 14)
- instalacije korištenjem upravitelja pip (linija 16)

Napomena: Komande ispod pretpostavljaju da ste **i dalje** u *sandbox* kontejneru.

```
# aktivacija conde u kontejneru
Apptainer> source /usr/local/miniforge/bin/activate

# kreiranje virtualnog okruženja data_stack
(base) Apptainer> conda create -n data_stack python=3.10 -y
...

# aktivacija virtualnog okruženja data_stack
(base) Apptainer> conda activate data_stack

# instalacija knjižnica pipom
(data_stack) Apptainer> pip3 install numpy scipy pandas matplotlib ipython
...
Downloading and Extracting Packages

Preparing transaction:
done

Verifying transaction:
done

Executing transaction: done

# ispis instaliranih knjižnica
(data_stack) Apptainer> pip3 list
...
ipython            8.13.2
matplotlib          3.7.1
matplotlib-inline  0.1.6
numpy              1.24.3
pandas             2.0.1
scipy              1.10.1
...
```

Korištenje instaliranog data_stack virtualnog okruženja osigurava se izvršavanjem python izvršne datoteke u /usr/local/miniforge/envs/data_stack/bin/python3:

```
# ispis verzije i staze __init__.py datoteke knjižnice matplotlib u kontejneru
[korisnik@kompjuter:~] $ apptainer exec data_stack_sandbox/ /usr/local/miniforge/envs/data_stack/bin/python3 -c
'import matplotlib; print("matplotlib verzija je: ", matplotlib.__version__); print("matplotlib __init__
datoteka je:", matplotlib.__file__)'
...
matplotlib verzija je:  3.7.1
matplotlib __init__ datoteka je: /usr/local/miniforge/envs/data_stack/lib/python3.10/site-packages/matplotlib
/__init__.py
```

Ostali koraci pretvaranja u *image* i dopremanja na Supek [ostaju isti](#), dok se cjelokupan proces [kao i prije](#) može zapisati u *.def* datoteci u kojoj je moguće definirati /usr/local/mambaforge/envs/data_stack/bin/python3 kao direktnu izvršnu naredbu.

Izgradnja condom + online repozitoriji

Cjelokupan proces izgradnje specifičnog python virtualnog okruženja može se preskočiti korištenjem [već pripremljenih kontejnera](#) koji u sebi već sadrže conda, poput [condaforge/miniforge3](#)

Ako se koriste kao baza za izgradnju gore navedenog virtualnog okruženja, definicijska datoteka `.def` izgledala bi na sljedeći način i mogla koristiti za [izgradnju slike koja se može dopremiti na Supeka](#):

- zaglavlje s osnovnim condaforge kontejnerom (linije 1 i 2)
- `%post` poglavlje s instalacijskim naredbama (linije 4 do 9)
- `%environment` poglavlje koje postavlja okoliš za virtualno okruženje `data_stack` (linije 11 do 22)
- `%runscript` poglavlje koje definira izvršnu naredbu python3 (linije 24 do 26)

```
Bootstrap: docker
From: condaforge/miniforge3

%post

conda create -n data_stack python=3.10 -y
. /opt/conda/bin/activate
conda activate data_stack
pip3 install numpy scipy pandas matplotlib ipython

%environment

. /opt/conda/bin/activate
conda activate data_stack

%runscript

exec python3 $@
```

Kako dalje?

- Lustre
 - [Wiki](#) - osnove dijeljenog datotečnog sustava Lustre
 - [Priručnik](#) - Lustre priručnik s detaljnim opisom
 - Najbolje prakse na ostalim superračunalnim centrima
 - [Maryland](#)
 - [INCD](#)
 - [NASA](#)
 - [CSC](#)
- Apptainer
 - [Službena dokumentacija](#) - za detalje o raznim komandama
 - [Originalni članak](#) - Zašto je Singularity/Apptainer uopće nastao?
- Pip
 - [Službena dokumentacija](#) - za detalje o raznim komandama pip
 - [RealPython tutorial](#) - super tutorial za uvod u pip
- Conda
 - [Službena dokumentacija](#) - za detalje o raznim komandama conda
 - [Isabella wiki](#) - naš stari wiki s najkorištenijim komandama i njihovim opisom