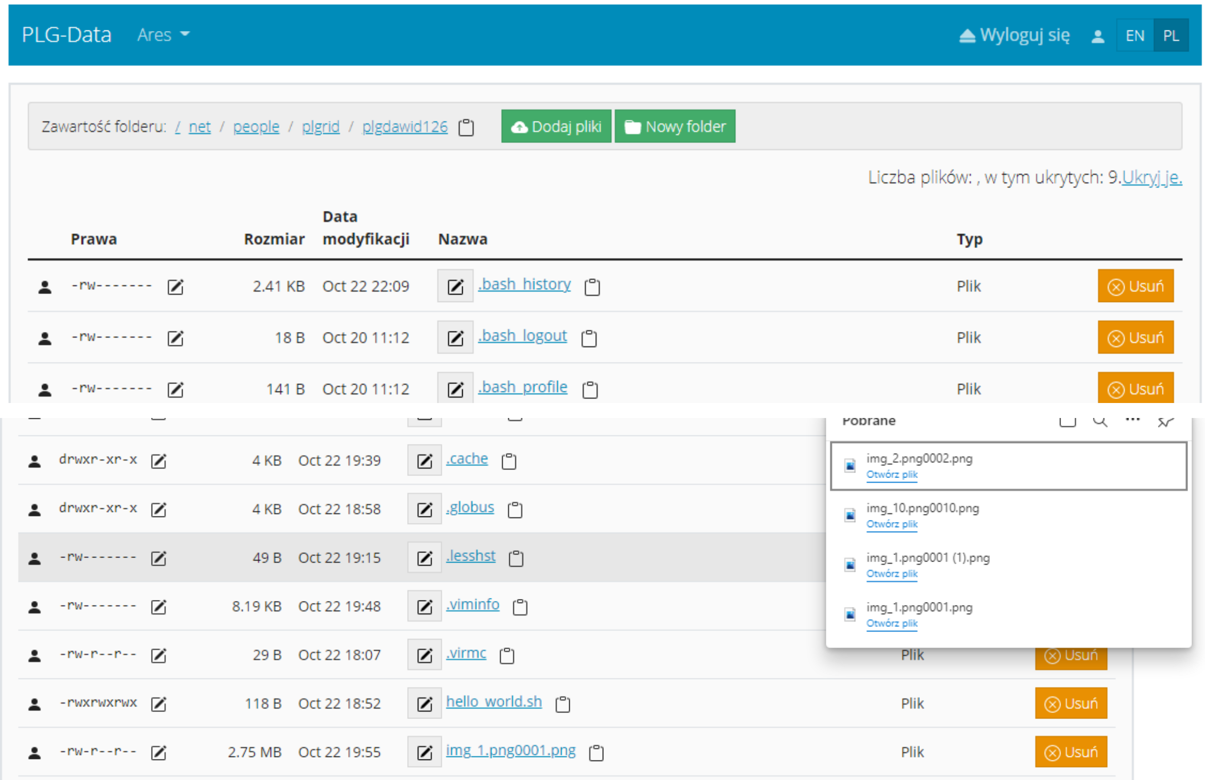
LSC raport
Lab 3

Dawid Białka

27.10.22

1. Use PLGData webgui to download and/or display images from the previous lab



Zawartość folderu: / net / people / plgrid / plgdawid126

Liczba plików: , w tym ukrytych: 9.Ukryj je.

Prawa	Rozmiar	Data modyfikacji	Nazwa	Typ	
-rw-------	2.41 KB	Oct 22 22:09	.bash_history	Plik	Usuń
-rw-------	18 B	Oct 20 11:12	.bash_logout	Plik	Usuń
-rw-------	141 B	Oct 20 11:12	.bash_profile	Plik	Usuń
drwxr-xr-x	4 KB	Oct 22 19:39	.cache		
drwxr-xr-x	4 KB	Oct 22 18:58	.globus		
-rw-------	49 B	Oct 22 19:15	.lesshst		
-rw-------	8.19 KB	Oct 22 19:48	.viminfo		
-rw-r--r--	29 B	Oct 22 18:07	.vimrc	Plik	Usuń
-rwxrwxrwx	118 B	Oct 22 18:52	hello_world.sh	Plik	Usuń
-rw-r--r--	2.75 MB	Oct 22 19:55	img_1.png0001.png	Plik	Usuń

Pobrane

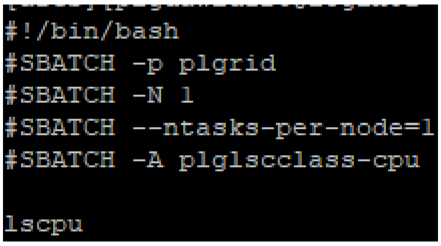
img_2.png0002.png
Otwórz plik

img_10.png0010.png
Otwórz plik

img_1.png0001 (1).png
Otwórz plik

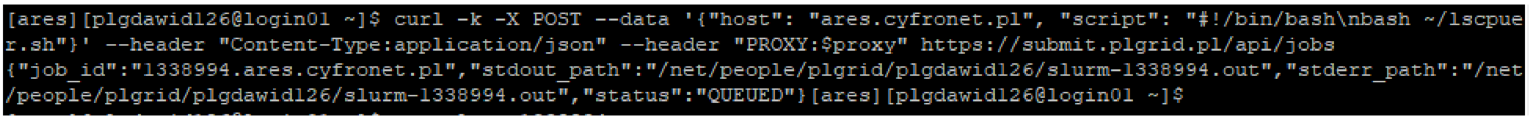
img_1.png0001.png
Otwórz plik

2. Prepare a dummy application in form of a bash script which will execute the lscpu command, name it “lscpuer.sh”. Copy the script to Ares, place it in your $HOME directory.

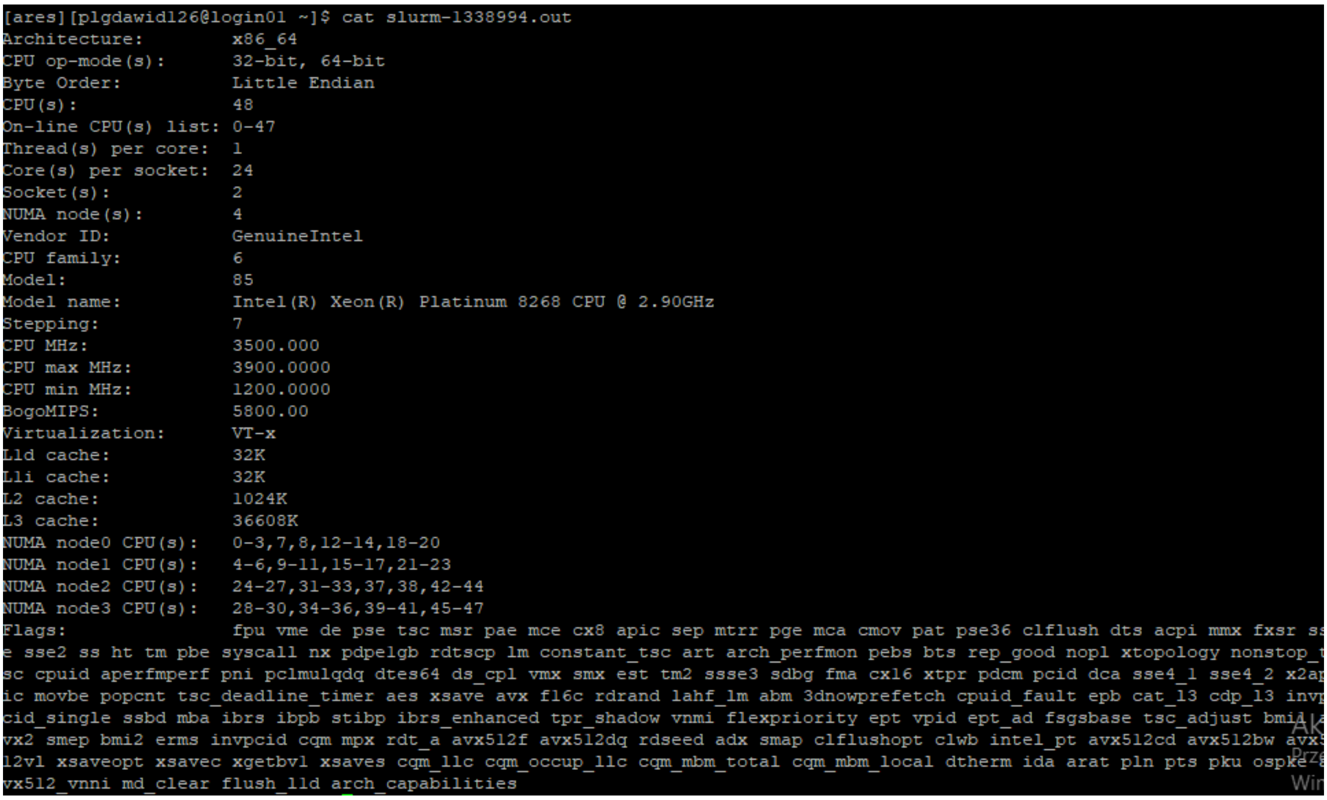


```
#!/bin/bash
#SBATCH -p plgrid
#SBATCH -N 1
#SBATCH --ntasks-per-node=1
#SBATCH -A plglscclass-cpu

lscpu
```

3. Use Rimrock to execute the lscpuer.sh as a computing job on a cluster

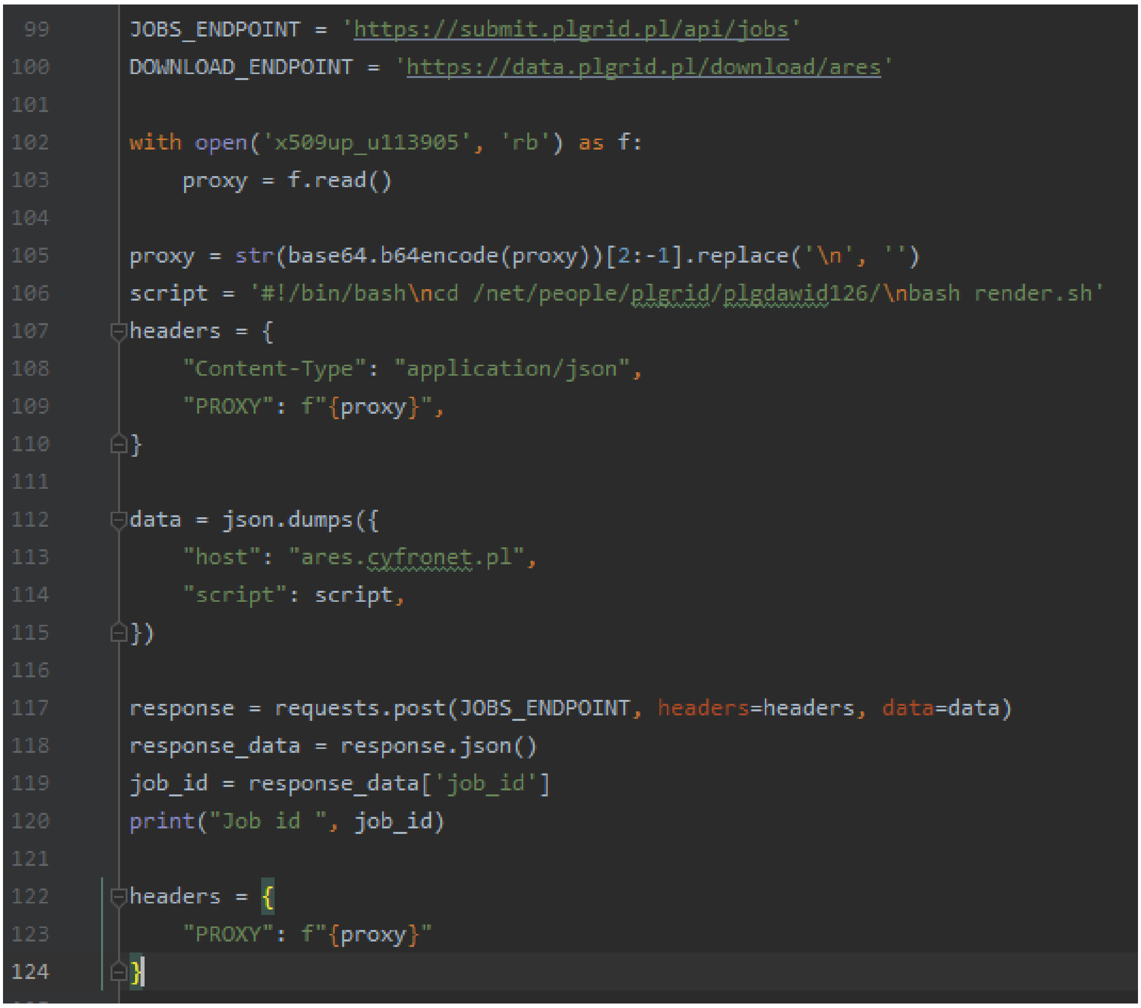


```
[ares][plgdawidl26@login01 ~]$ curl -k -X POST --data '{"host": "ares.cyfronet.pl", "script": "#!/bin/bash\nbash ~/lscpue
r.sh"}' --header "Content-Type:application/json" --header "PROXY:$proxy" https://submit.plgrid.pl/api/jobs
{"job_id":"1338994.ares.cyfronet.pl","stdout_path":"/net/people/plgrid/plgdawidl26/slurm-1338994.out","stderr_path":"/net
/people/plgrid/plgdawidl26/slurm-1338994.out","status":"QUEUED"}[ares][plgdawidl26@login01 ~]$
```

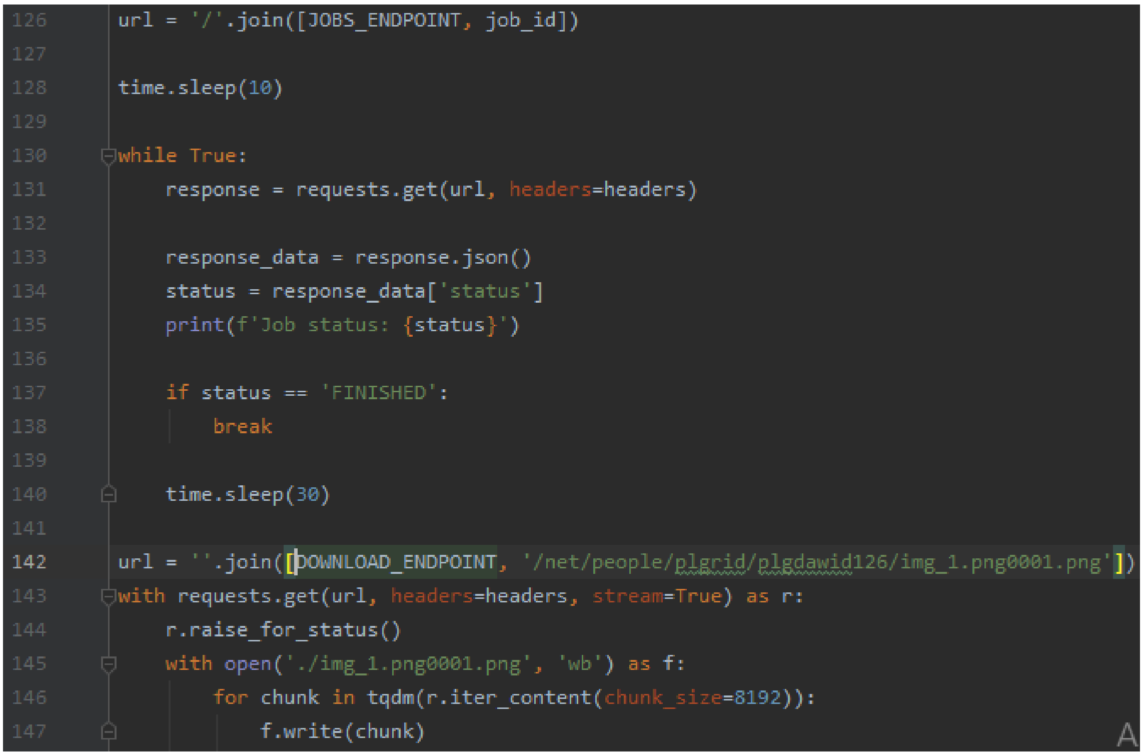


```
[ares][plgdawidl26@login01 ~]$ cat slurm-1338994.out
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                48
On-line CPU(s) list:   0-47
Thread(s) per core:    1
Core(s) per socket:    24
Socket(s):             2
NUMA node(s):          4
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 85
Model name:            Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90GHz
Stepping:              7
CPU MHz:               3500.000
CPU max MHz:           3900.0000
CPU min MHz:           1200.0000
BogoMIPS:              5800.00
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              1024K
L3 cache:              36608K
NUMA node0 CPU(s):     0-3,7,8,12-14,18-20
NUMA node1 CPU(s):     4-6,9-11,15-17,21-23
NUMA node2 CPU(s):     24-27,31-33,37,38,42-44
NUMA node3 CPU(s):     28-30,34-36,39-41,45-47
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr ss
e sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_t
sc cpuid aperfmperf pni pclmulqdq dtes64 ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2ap
ic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb cat_l3 cdp_l3 invp
cid_single ssbd mba ibrs ibpb stibp ibrs_enhanced tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 a
vx2 smep bmi2 erms invpcid cqm mpx rdt_a avx512f avx512dq rdseed adx smap clflushopt clwb intel_pt avx512cd avx512bw avx5
12vl xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc cqm_mbm_total cqm_mbm_local dtherm ida arat pln pts pku ospke
vx512_vnni md_clear flush_l1d arch_capabilities
```

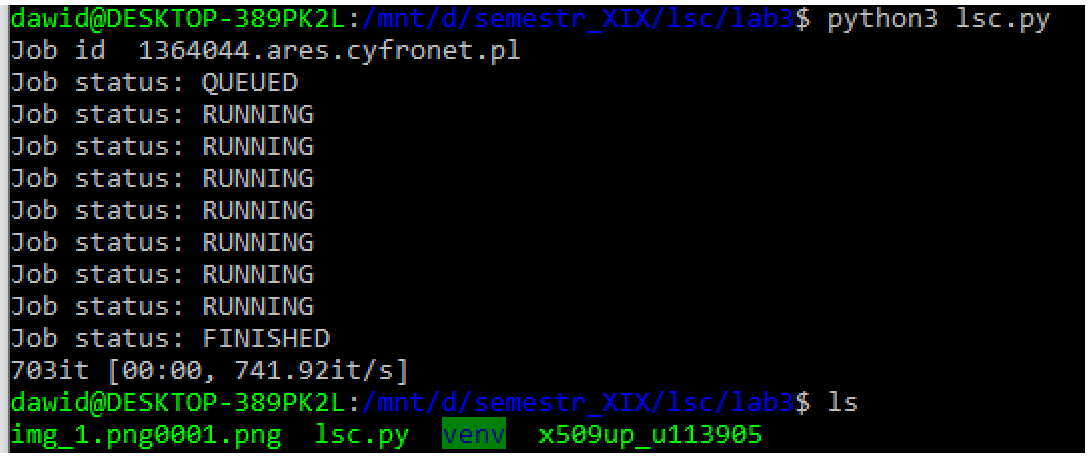
4. Write an app, which will run locally on your PC and will perform the following tasks



```python
JOBS_ENDPOINT = 'https://submit.plgrid.pl/api/jobs'
DOWNLOAD_ENDPOINT = 'https://data.plgrid.pl/download/ares'

with open('x509up_u113905', 'rb') as f:
    proxy = f.read()

proxy = str(base64.b64encode(proxy))[2:-1].replace('\n', '')
script = '#!/bin/bash\ncd /net/people/plgrid/plgdawid126/\nbash render.sh'
headers = {
    "Content-Type": "application/json",
    "PROXY": f"{proxy}",
}

data = json.dumps({
    "host": "ares.cyfronet.pl",
    "script": script,
})

response = requests.post(JOBS_ENDPOINT, headers=headers, data=data)
response_data = response.json()
job_id = response_data['job_id']
print("Job id ", job_id)

headers = {
    "PROXY": f"{proxy}"
}
```



```python
url = '/'.join([JOBS_ENDPOINT, job_id])

time.sleep(10)

while True:
    response = requests.get(url, headers=headers)

    response_data = response.json()
    status = response_data['status']
    print(f'Job status: {status}')

    if status == 'FINISHED':
        break

    time.sleep(30)

url = ''.join([DOWNLOAD_ENDPOINT, '/net/people/plgrid/plgdawid126/img_1.png0001.png'])
with requests.get(url, headers=headers, stream=True) as r:
    r.raise_for_status()
    with open('./img_1.png0001.png', 'wb') as f:
        for chunk in tqdm(r.iter_content(chunk_size=8192)):
            f.write(chunk)
```



```
dawid@DESKTOP-389PK2L:/mnt/d/semestr_XIX/lsc/lab3$ python3 lsc.py
Job id  1364044.ares.cyfronet.pl
Job status: QUEUED
Job status: RUNNING
Job status: RUNNING
Job status: RUNNING
Job status: RUNNING
Job status: RUNNING
Job status: RUNNING
Job status: RUNNING
Job status: FINISHED
703it [00:00, 741.92it/s]
dawid@DESKTOP-389PK2L:/mnt/d/semestr_XIX/lsc/lab3$ ls
img_1.png0001.png  lsc.py  venv  x509up_u113905
```