Large scale computing

infrastructures for computation & infrastructures for experimentation

Pierre Neyron (Pimlig / Grid'5000)
Jean-Louis Mas (MI / GPU LIG)
Pierre-Antoine Bouttier (Gricad)
Oliver Henriot (Gricad)
Glenn Cougoulat (Gricad)
Albin Petit (Grid'5000)

Outline

Introduction

Pierre Neyron (Pimlig)

What is / what about large scale computing ? (1/2)

"Large scale, high performance (HPC), computing centers, clusters..."

- ... as opposed to a computer one can have on/under the desk:
  - User's Laptop, even on steroid *(possibly with a quite powerful GPU)* < 2k€ → not shared
  - User's desktop workstation on steroid *(possibly with GPU(s))* < 3k€ → not shared
  - Team's compute server *(possibly with GPUs)* < 10k€ → naive sharing between teammates ?
- ... not anymore a big *"mainframe"* or very specialized hardware
  → a **pool/farm of compute servers** *(with or without server class GPUs, high performance network, ...)*

- Large scale = economy of scale → serve ***MANY USERS***, *"mutualisation"* of the needs
  - not everybody can/should have 5-10k€ under the desk *(and not only because of the noise)*
  - not everybody knows how to manage a computing server *(neither a "computing" workstation or laptop...)*
  - 1 user = 1 server ***is not sustainable*** in term of IT support → manage at scale
- Large scale = parallelism → **efficiency**
  - a farm of shared compute servers will absorb computation workloads in much larger batch
  - computing in a farm provides ***N times*** the ***performance*** and ***memory size*** of 1 single server

What is / what about large scale computing ? (2/2)

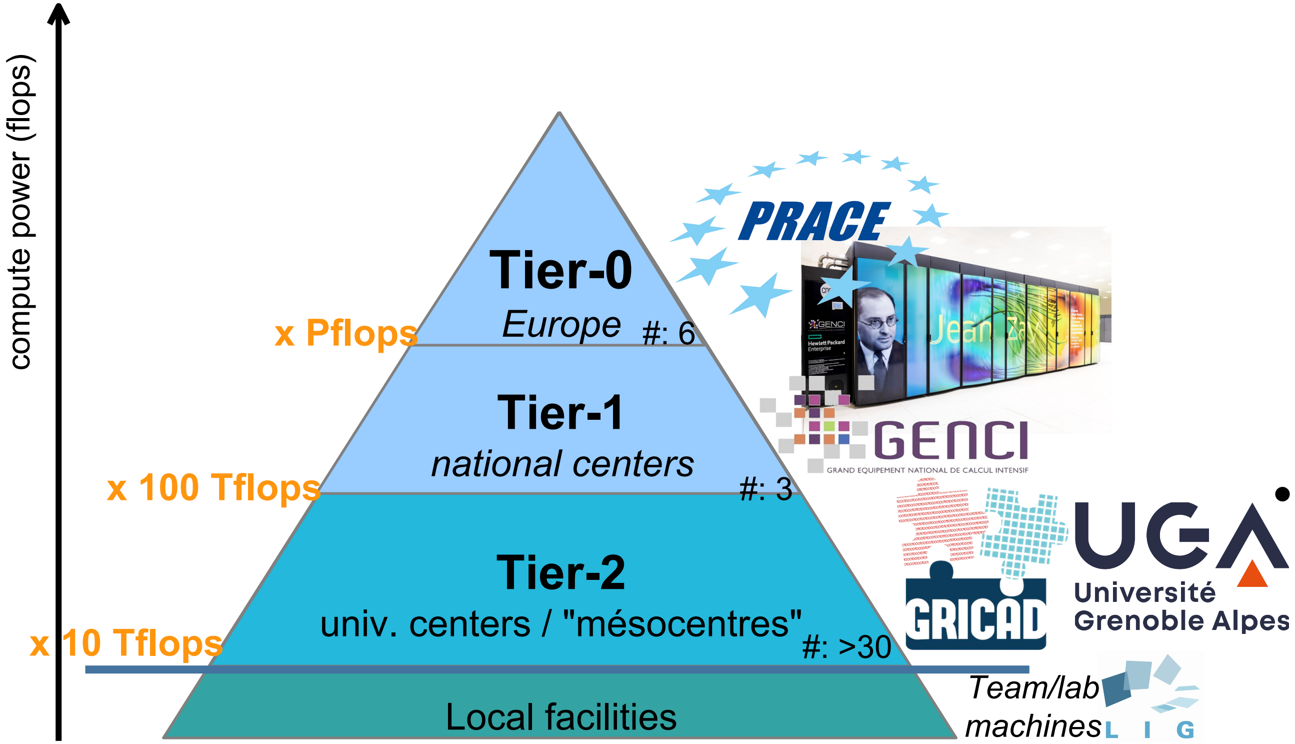
"Large scale, high performance (HPC), computing centers, clusters..."

- *"With great [compute] power comes great responsibility"*
  - Cost (x€.core/h), Energy consumption, security threats, ...
    → responsible computing (compute is never gratis)
  - Sharing resources with many users
    → fair allocation decided by the system (policy), not the user

- Might have an "entry cost" with a "steep learning curve at the beginning"
  - <u>*Access/account*</u>: a fine tuned service always requires a dedicated user environment
  - <u>*Remote*</u>: not on/under the desk means no keyboard/mouse/screen... nor Windows.
  - <u>*Scripting*</u>: automate at most, allow unattended executions in batch
  - <u>*Data*</u>: know about sizes, transfers, inodes (#files), security
  - <u>*Computing toolkits*</u>: batch scheduler, parallel launchers, libraries (MPI, dask, ...)

  → ***concessions to make***: tools to learn, usage policy to accept.

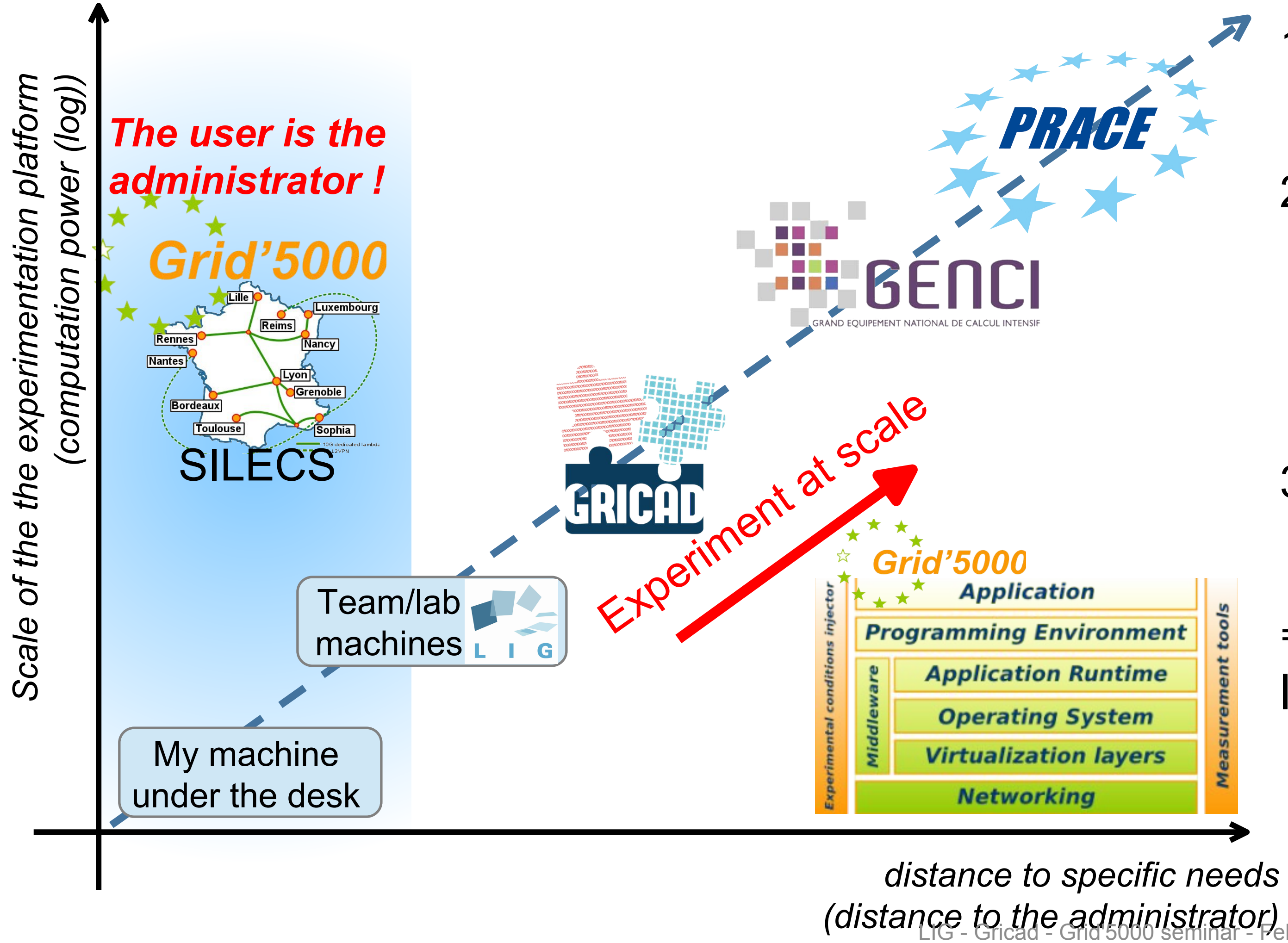
What is / what about large scale computing ?
→ an academic HPC ecosystem with several stages



Pyramid of the High Performance Computing academic ecosystem

- EU/National (http://www.genci.fr)
  - HPC: several super-calulators
  - Bi-annual calls for compute hours
  - "Jean Zay": new super-calculator to support AI
  - 2000 compute servers
  - 1000 V100 GPUs
  - AI "dynamic access" program
  - Some constrains (access, security...)
    (http://www.idris.fr/jean-zay)
- UGA → Gricad/Ciment (https://gricad.univ-grenoble-alpes.fr)
  - Dedicated UGA service unit for computation, (HPC, Bigdata, IA,...) and more (cloud, gitlab...)
  - High Perf. Hardware, Network, Storage
  - Local experts/advisers - support team
  - Relay to national
- LIG GPU cluster
  - Very basic access to some GPU servers

What is / what about large scale computing ?
→ *SILECS*/*Grid'5000* dedicated to experimentation



When research works require

1. experimenting at scale
   - large scale systems study (clouds, HPC, distributed systems, ...), scalability study
2. without limits
   - change core components (kernel, hypervisors, libs)
   - deploy full software stack, system services (root)
   - setup complex network or storage
   - fine tune parameters down to metal
   - do not hide rather expose complexity
3. in a controlled environment
   - fully understandable system (no restriction)
   - access to any system metrics

⇒ Compute centers, clouds, ... only provide limited support for research

→ *SILECS*/*Grid'5000*: national research infrastructure, dedicated to experiment driven research in Distributed Computing

Cluster GPU LIG

Jean-Louis Mas (MI)

Access and use

How to access the LIG's GPU cluster ?

- https://intranet.liglab.fr/en/it-resources/gpu-servers
- Mandatory; use oar job scheduler frontend : aker.imag.fr (SSH)
- Access from LIG's internal Ethernet network, or via LIG' s ssh bastions or via VPN

Who can use the LIG's GPU cluster ?

- Every LIG member with a valid LIG account

Where are my data

- On the NFS server of your team (mrim, getalp, ama)
- On a NFS share kindly provided by ama's team (space is limited, move your data)

Hardware

5 servers hosting 4 GPU

- 18 GPUs Nvidia GTX 1080 Ti
  - 3584 Nvidia Cuda Cores, Standard Memory Config 11 GB GDDR5X
- 2 GPU Nvidia Titan Pascal
  - 3584 Nvidia Cuda Cores, Standard Memory Config 12 GB GDDR5X

1 server hosting 2 GPU

- 2 GPU Nvidia Quadro RTX 6000
  - 4608 Nvidia Cuda Parallel-Processing Cores, Nvidia Tensor Cores 576, Nvidia RT Cores 72, Standard Memory Config 24 GB GDDR6

Tools

Actual

- Python 2.7 & 3.5
- Cuda 10.0 9.1 9.0 8.0 with cudnn
- tensorflow-gpu 1.4 (matching default cuda version)

Improvements scheduled

- New Cuda versions as default
- New tensorflow version matching latest Cuda version
- Jupyter notebooks

Pros and cons

Pros

- Quick/easy access
- Work on your data directly, as they are mounted on LIG's GPUs cluster

Cons

- Very limited support / documentation (best-effort by LIG MI team)
- Not the bigest GPUs in the market

Gricad

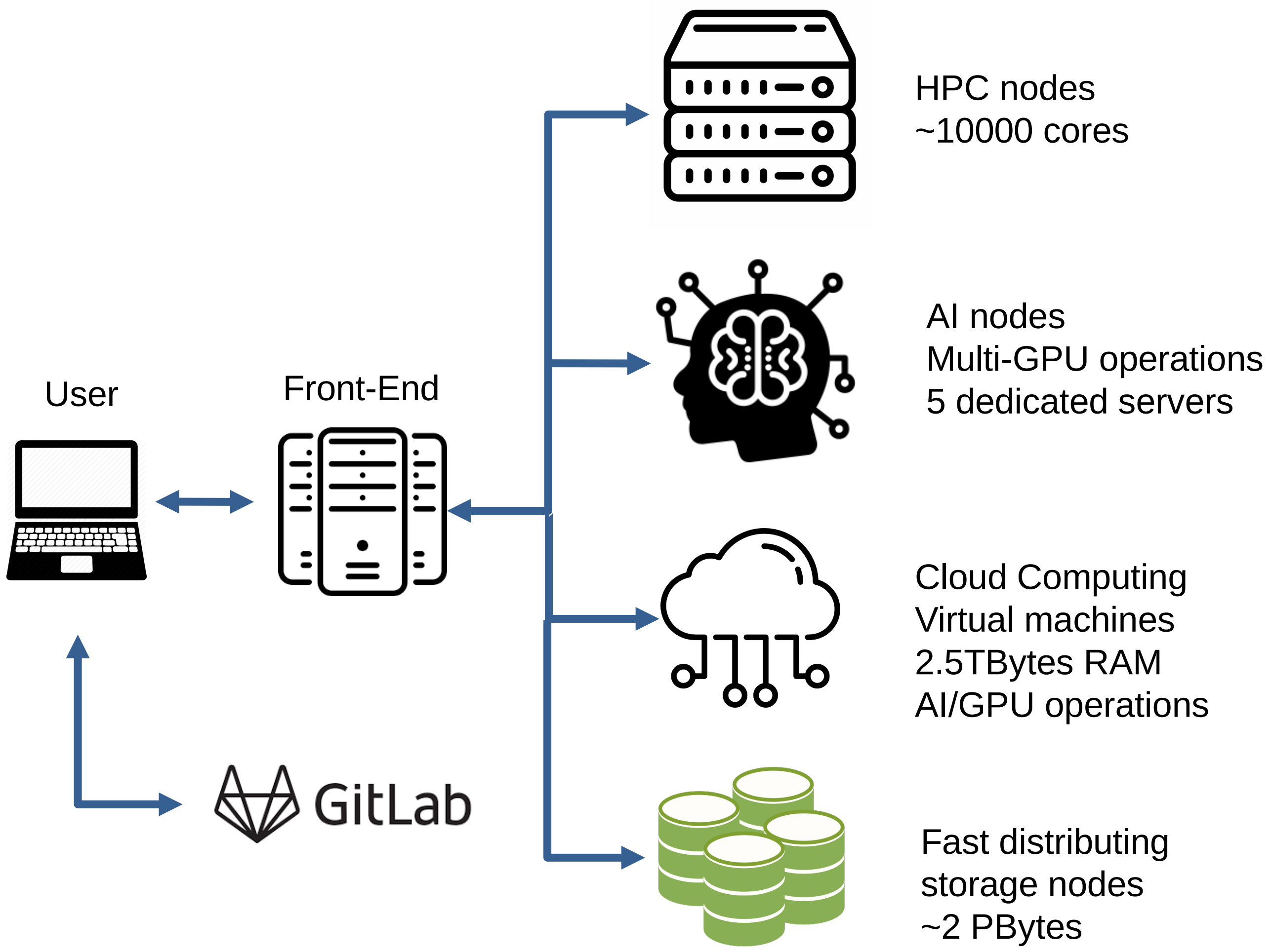
Pierre Antoine Bouttier (Gricad)
Oliver Henriot (Gricad)

GRICAD : data and computing services for research

Pierre-Antoine Bouttier
pierre-antoine.bouttier@univ-grenoble-alpes.fr
Oliver Henriot
oliver.henriot@univ-grenoble-alpes.fr

20 février 2020, séminaire LIG

Current infrastructure in brief



GRICAD services

- Counseling and support
- Scientific and intensive computing
- Cloud computing
- Data management, design and engineering
- Gitlab
- Training and animation
- Audiovisual production

- Access policy: freely accessible to all academic researchers belonging to a UGA COMUE institution and to all their collaborators within the context of research projects.

- Pooling and rationalisation of material and human resources within the Grenoble site (COMUE UGA)

GRICAD services: scientific computing and data analysis

- Several platforms to fit your needs
  - Intensive computing: Froggy and Dahu clusters
  - HTC: computing grid (CiGri)
  - Batch computing: all clusters and CiGri
  - Non-standard computing nodes: Luke cluster
  - Cloud computing: NOVA

- Open for all Grenoble researchers and all their external collaborators

- Infrastructures enabled by fund sharing (scientific projects, laboratories, institutions)

GRICAD services: data storage

- Several platforms to fit your needs
  - MANTIS: Distributed storage reachable from every computing platform (except NOVA for the moment) and from IDRIS (adapp)
  - Bettik: High performance storage, to manage intensive I/O; reachable from Luke and Dahu
  - SUMMER: Secure storage; NetApp technology, UGA service
- MANTIS and Bettik are open for all Grenoble researchers and all their external collaborators
- SUMMER is a paid service
- MANTIS and Bettik enabled by fund sharing (scientific projects, laboratories, institutions)

NOVA, our cloud computing platform

- On-demand virtual machines
- Based on the Openstack technology
- Usages: punctual computing, development environment, data collecting, etc.
- Request for access (currently): gricad-contact@univ-grenoble-alpes.fr

Focus on computing resources

- Dahu
  - 2392 cores, 15Tb shared home, local scratch 480Gb SSD and 4Tb HDD
  - 65 nodes with two 16 core Xeon Gold 6130 and 192 Gb RAM
  - 9 nodes with two 12 core Xeon Gold 6126 and 192 Gb RAM
  - 3 gpu nodes with 4 Tesla V100 SXM2, two 16 core Xeon Gold 6130, 192Gb RAM and 200Gb SSD local scratch

GRICAD services: scientific computing and data analysis

- Froggy
  - 3244 cores, 90Tb shared Lustre scratch and 30Tb shared home
  - 190 nodes with two 8 core Xeon E5-2670 and 64Gb RAM
  - fat node with four 8 core Xeon E5-4620 and 512Gb RAM
  - gpu node with two Tesla K40t, two 8 core Xeon E5-2650 and 64Gb RAM
  - gpu nodes with two Tesla K20m, two 8 core Xeon E5-2670 and 32Gb RAM
  - One visu node with a Quadro 6000, two 6 core Xeon E5-2640 and 64Gb RAM

- Luke
  - 1126 heterogeneous cores
  - 62 heterogeneous nodes (in RAM, disk space, etc)

GRICAD services: scientific computing and data analysis

- Storage
  - Bettik: scratch for Luke and Dahu Clusters
    - BeeGFS
    - 1,3Pb
    - 16 data nodes
    - 3 metadata nodes
  - Mantis: cloud storage
    - 700Tb iRODS store transparently accessible from all clusters and IDRIS adapp

Accessing GRICAD computing resources

https://perseus.ujf-grenoble.fr

- Permanent:
  - Create an account
  - Create or join an Perseus project
  - Access the machines

- Non-permanent
  - Create an account
  - Join an existing project
  - Access the machines

The computing grid CiGri

- Optimal usage of the computing ressources (best-effort)
- Very relevant for a large number of simulations requiring a few ressources
- Automatic resubmission

Software environment

- The Module command:
  - Deprecated on GRICAD clusters
  - In use in the national centers
  - No reproducibility

- Nix and Guix, package managers
  - GRICAD proposal
  - Reproducibility oriented (very good solution for the grid)
  - Easy to set up the same environment on multiple platforms

- For GPU
  - Conda global environments

Software environment

- Containers
  - Available on Luke and Dahu
  - Singularity and Charliecloud (compatible with docker containers)
  - Could be tricky (or impossible) with muli-nodes or GPU computing

- In user space (within reasonable limits):
  - Conda, spack
  - No help from GRICAD with these solutions
  - Not shared solutions…
  - …but sometimes unavoidable

Good practices for computing and storage

- Read the docs!
- Be aware of rules of usage
- Do not launch heavy workloads on shared spaces
- Identify adapted platform for your needs
- For computing clusters: I/O in scratch spaces (/scratch on Froggy, /bettik on Luke and Dahu); pre and post computing data storage with MANTIS ; remeber to clean your storage
- No backups of your data on our clusters (SUMMER can be used for that)
- Do not hesitate to contact us!

To obtain help and to contact us

Where can you search for help?

- GRICAD website: https://gricad.univ-grenoble-alpes.fr
- Documentation: https://gricad-doc.univ-grenoble-alpes.fr
- To contact us: gricad-contact@univ-grenoble-alpes.fr
- Need support? Our helpdesk: sos-gricad@univ-grenoble-alpes.fr
- Punctual question, discussions: https://gricad.slack.com
- To share with other users: ciment-users@univ-grenoble-alpes.fr
- And finally, to share your needs, your remarks: gricad-comut@univ-grenoble-alpes.fr

Thank you for your attention!
Remarks/questions?

Genci

Glenn Cougoulat (Gricad)

Acces Serveurs Jean-Zay

- Helps on Dari Request
- Large Dataset via Irods (in progress)
- Access via our Clusters (in progress)
- Environnement module and Conda
- Gricad Account to operate simple prototyping contact : Glenn Cougoulat glenn.cougoulat@univ-grenoble-alpes.fr

```
openmpi/4.0.1/gcc-4.8.5-cuda
cuda/10.1.1
nccl/2.4.2-1+cuda10.1
cudnn/10.1-v7.5.1.10
```

Vous chargez également la version de Python correspondante (py3 ou py2).

Tensorflow

Python 3

```
tensorflow-gpu/py3/2.0.0-beta1
tensorflow-gpu/py3/1.14
tensorflow-gpu/py3/1.14-mpi
tensorflow-gpu/py3/1.13
tensorflow-gpu/py3/1.12
tensorflow-gpu/py3/1.8
tensorflow-gpu/py3/1.4
```

Python 2

```
tensorflow-gpu/py2/2.0.0-beta1
tensorflow-gpu/py2/1.14
tensorflow-gpu/py2/1.13
tensorflow-gpu/py2/1.8
tensorflow-gpu/py2/1.4
```

PyTorch

Python 3

```
pytorch-gpu/py3/1.1
```

Python 2

```
pytorch-gpu/py2/1.1
```

Caffe

Python 3

```
caffe-gpu/py3/1.0
```

Python 2

```
caffe-gpu/py2/1.0
```

Grid'5000

Albin Petit (Grid'5000)

Experimentation vs. computation

Computation	Experimentation
People interested in the result ("*compute quickly*")	People interested in the methodology ("*evaluate, measure*")

What is Grid'5000

A large-scale testbed for research in distributed computing

- 8 sites
- 36 clusters
- 838 nodes
- 15.116 cores

Clusters are equipped with various technologies:

- 1568 CPUs : 106 AMD, 1454 Intel Xeon, 8 ARM
- 176 GPUs
- 8 NVMe, 488 SSDs and 1038 HDDs (total: 1.29 PB)
- 95.96 TiB RAM + 6.0 TiB PMEM
- 10G and 25G Ethernet, Infiniband, Omni-Path

What is Grid'5000

Highly reconfigurable and controllable nodes :

- Change bios configurations
- Deploy custom operating systems
- Start the Linux kernel with customized parameters
- Be root on the machine
- Install the piece of software you want

Network possibilities:

- Dedicated 10 Gbps backbone between sites, isolated from the public Internet
- Nodes with up to 5 network interfaces (4x10G, 1x1G) ; many nodes with two connected 10G network interfaces
- Change VLAN configuration on the switch to isolate your nodes from the rest of the Grid5000 network

What is Grid'5000

Support for Cloud experiments:

- Support for deploying OpenStack using ENOS
- Support of Docker and Nvidia Docker
- Support of Singularity with Spack module
- Other software stacks (e.g. Kubernetes) are supported by the community

Support for Big Data experiments:

- Nodes with up to five disks (HDDs or SSDs)
- Large storage spaces available for medium-term storage inside the nodes
- Shared storage space available on the testbed (large NFS servers)
- GPU nodes for Big Data processing

Advanced monitoring capabilities:

- Monitor the power consumption of the nodes

An experiment’s outline

STEP 1 Discovering resources and selecting resources

STEP 2 Reconfiguring the resources to meet experimental needs

STEP 3 Uploading your data on the nodes

STEP 4 Monitoring experiments, extracting and analyzing data

STEP 5 Controlling experiments -> automation, reproducible research

Step 1: Discovering resources

Site	Cluster	Queue	Date of arrival	Nodes	CPU	Cores	Memory	Storage	Network	Accelerators
Grenoble	troll	default	2019-12-23	4	2 x Intel Xeon Gold 5218	16 cores/CPU	384 GiB + 1.5 TiB PMEM	480 GB SSD + 1.6 TB SSD	10 Gbps + 100 Gbps Omni-Path	
Nancy	grue	production	2019-11-25	5	2 x AMD EPYC 7351	16 cores/CPU	128 GiB	479 GB SSD	10 Gbps	4 x Nvidia Tesla T4
Nancy	gros	default	2019-09-04	124	Intel Xeon Gold 5220	18 cores/CPU	96 GiB	480 GB SSD + 960 GB SSD	2 x 25 Gbps	
Lyon	gemini	default	2019-09-01	2	2 x Intel Xeon E5-2698 v4	20 cores/CPU	512 GiB	480 GB SSD + 4 x 1.92 TB SSD	10 Gbps + 100 Gbps InfiniBand	8 x Nvidia Tesla V100
Nancy	graffiti	production	2019-06-07	13	2 x Intel Xeon Silver 4110	8 cores/CPU	128 GiB	479 GB SSD	10 Gbps	4 x Nvidia RTX 2080 Ti
Lille	chiclet	default	2018-08-06	8	2 x AMD EPYC 7301	16 cores/CPU	128 GiB	480 GB SSD + 2 x 4.0 TB HDD	2 x 25 Gbps	
Lille	chifflot	default	2018-08-01	8	2 x Intel Xeon Gold 6126	12 cores/CPU	192 GiB	2 x 480 GB SSD + 4 x 4.0 TB HDD	2 x 25 Gbps	[1-6]: 2 x Nvidia Tesla P100 [7-8]: 2 x Nvidia Tesla V100
Nancy	grvingt	production	2018-04-11	64	2 x Intel Xeon Gold 6130	16 cores/CPU	192 GiB	1.0 TB HDD	10 Gbps + 100 Gbps Omni-Path	
Grenoble	dahu	default	2018-03-22	32	2 x Intel Xeon Gold 6130	16 cores/CPU	192 GiB	240 GB SSD + 480 GB SSD + 4.0 TB HDD	10 Gbps + 100 Gbps Omni-Path	
Grenoble	yeti	default	2018-01-16	4	4 x Intel Xeon Gold 6130	16 cores/CPU	768 GiB	480 GB SSD + 1.6 TB SSD + 3 x 2.0 TB HDD	10 Gbps + 100 Gbps Omni-Path	
Nantes	ecotype	default	2017-10-16	48	2 x Intel Xeon E5-2630L v4	10 cores/CPU	128 GiB	400 GB SSD	2 x 10 Gbps	
Nancy	grele	production	2017-06-26	14	2 x Intel Xeon E5-2650 v4	12 cores/CPU	128 GiB	2 x 299 GB HDD	10 Gbps + 100 Gbps Omni-Path	2 x Nvidia GTX 1080 Ti
Lille	chetemi	default	2016-12-01	15	2 x Intel Xeon E5-2630 v4	10 cores/CPU	256 GiB	[1-9,11-15]: 2 x 300 GB HDD 10: 300 GB HDD + 600 GB HDD	2 x 10 Gbps	
Lille	chifflet	default	2016-12-01	8	2 x Intel Xeon E5-2680 v4	14 cores/CPU	768 GiB	2 x 400 GB SSD + 2 x 4.0 TB HDD	2 x 10 Gbps	2 x Nvidia GTX 1080 Ti
Lyon	nova	default	2016-12-01	23	2 x Intel Xeon E5-2620 v4	8 cores/CPU	64 GiB	598 GB HDD	10 Gbps	
Nancy	grimani	production	2016-08-30	6	2 x Intel Xeon E5-2603 v3	6 cores/CPU	64 GiB	1.0 TB HDD	10 Gbps + 100 Gbps Omni-Path	2 x Nvidia Tesla K40M
Nancy	grimoire	default	2016-01-22	8	2 x Intel Xeon E5-2630 v3	8 cores/CPU	128 GiB	200 GB SSD + 5 x 600 GB HDD	4 x 10 Gbps + 56 Gbps InfiniBand	
Nancy	graoully	production	2016-01-04	16	2 x Intel Xeon E5-2630 v3	8 cores/CPU	128 GiB	2 x 600 GB HDD	10 Gbps + 56 Gbps InfiniBand	
Nancy	grisou	default	2016-01-04	51	2 x Intel Xeon E5-2630 v3	8 cores/CPU	128 GiB	2 x 600 GB HDD	[1-48]: 1 Gbps + 4 x 10 Gbps 49: 4 x 10 Gbps [50-51]: 4 x 10 Gbps + 56 Gbps InfiniBand	

From: https://www.grid5000.fr/w/Hardware

Step 1: Resource reservation

1

Submit an interactive job

oarsub -I -l nodes=1,walltime=1:45

2

Reservations in advance

oarsub -l nodes=1,walltime=3 -r '2020-03-01 16:00:00'

3

Complex queries using resource manager

oarsub -p "wattmeter='YES' and gpu_model!='NO'" -l nodes=2,walltime=2 -I
oarsub -l "{cluster='a'}/nodes=1+{cluster='b' and eth10g='Y'}/nodes=2" -I

Step 2: Reconfiguring the resources to meet experimental needs

Customize an existing Operating System:

- Use Kameleon and extend existing grid5000 recipes
- Deploy a provided OS, install your own tools and export it with tgz-g5k

Install a custom Operating System with Kadeploy:

- Provides a Hardware-as-a-Service cloud infrastructure
- Enable users to deploy their own software stack & get root access
- Scalable, efficient, reliable and flexible: 200 nodes deployed in ~5 minutes

  kadeploy3 -e debian9-x64-base -f $OAR_FILE_NODES

Customize networking environment with KaVLAN

- Avoid network pollution
- Create custom topologies
- Reconfiguring VLANS has almost no overhead

Default OS
Debian Stretch
Debian Buster
Debian Testing
Ubuntu 18.04
Centos 7
Centos 8

Step 3: Uploading your data on the nodes

Store your data on the test bed

- /home (default quota of 25GB but can be extended with quota extension)
- Group Storage (for large storage spaces - bigger than 200GB)
- On node local disks reservation
- Managed Ceph cluster (Object-based storage resources)

Move your data on the node

- Use your home (via a NFS)
- Upload then through ssh or deploying tools (Puppet, Ansible, Salstack, ...)

Step 4: Monitor experiments

Infrastructure-level probes: Kwapi

- Power consumption
- Captured at high frequency (50 Hz)
- Live visualization & REST API
- Long-term storage

Keep track of your logs

- All events on STDOUT and STDERR are stored on your home
- Stored data locally on the node

System-level probes

- Usage of CPU, memory, disk, is available through Ganglia
- Use system tools

Step 5: Controlling experiments -> automation, reproducible research

Legacy way of performing experiments: shell commands

- Time-consuming
- Error-prone
- Details tend to be forgotten over time

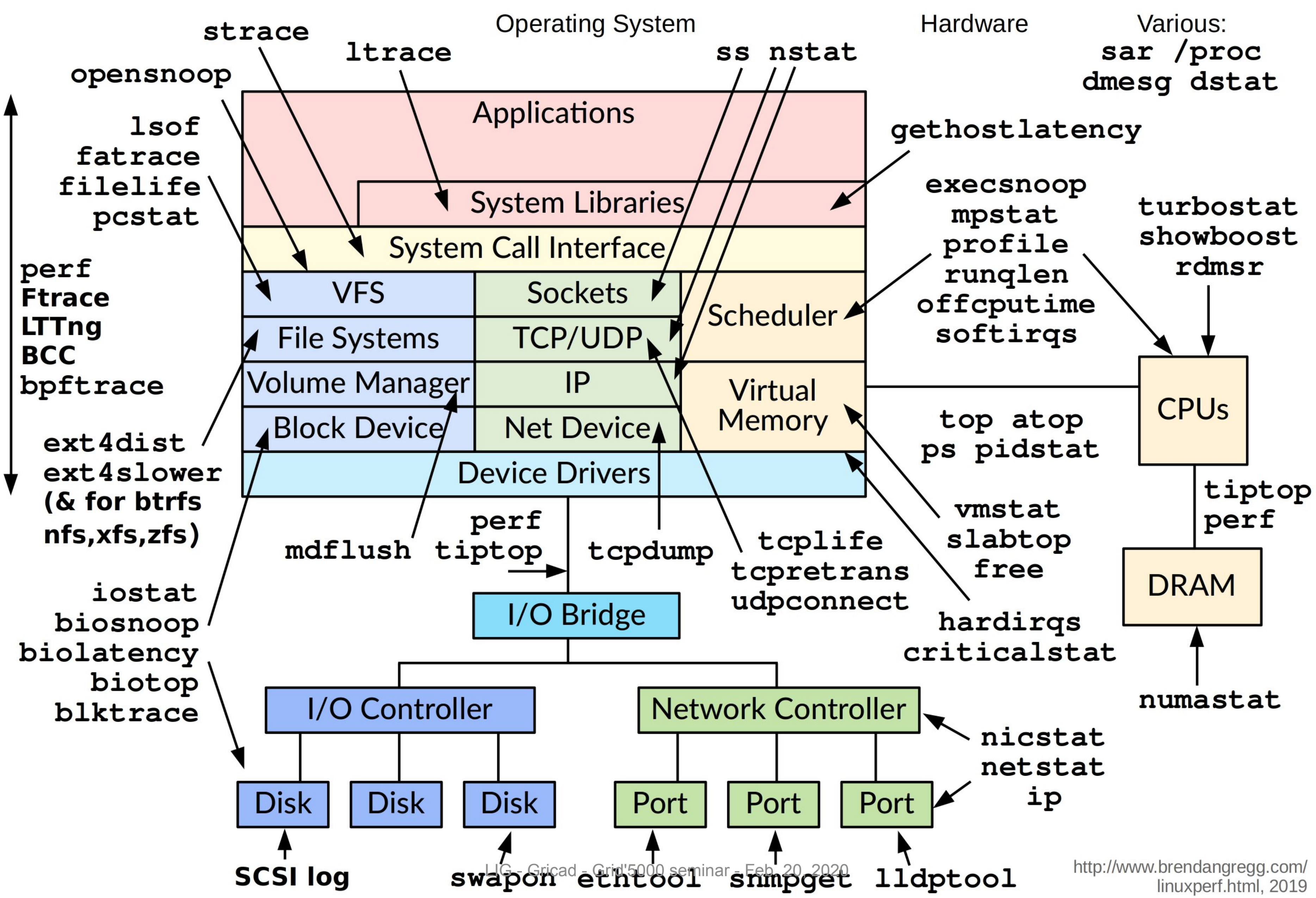
Promising solution: automation of experiments

- Executable description of experiments
- Reproducible research
- Support from the testbed: Grid’5000 RESTful API
- Resource selection, reservation, deployment, monitoring

Several higher-level tools to help automate experiments

- Execo, Python-Grid5000 (Python), Ruby-cute (Ruby)
- https://www.grid5000.fr/w/Grid5000:Software

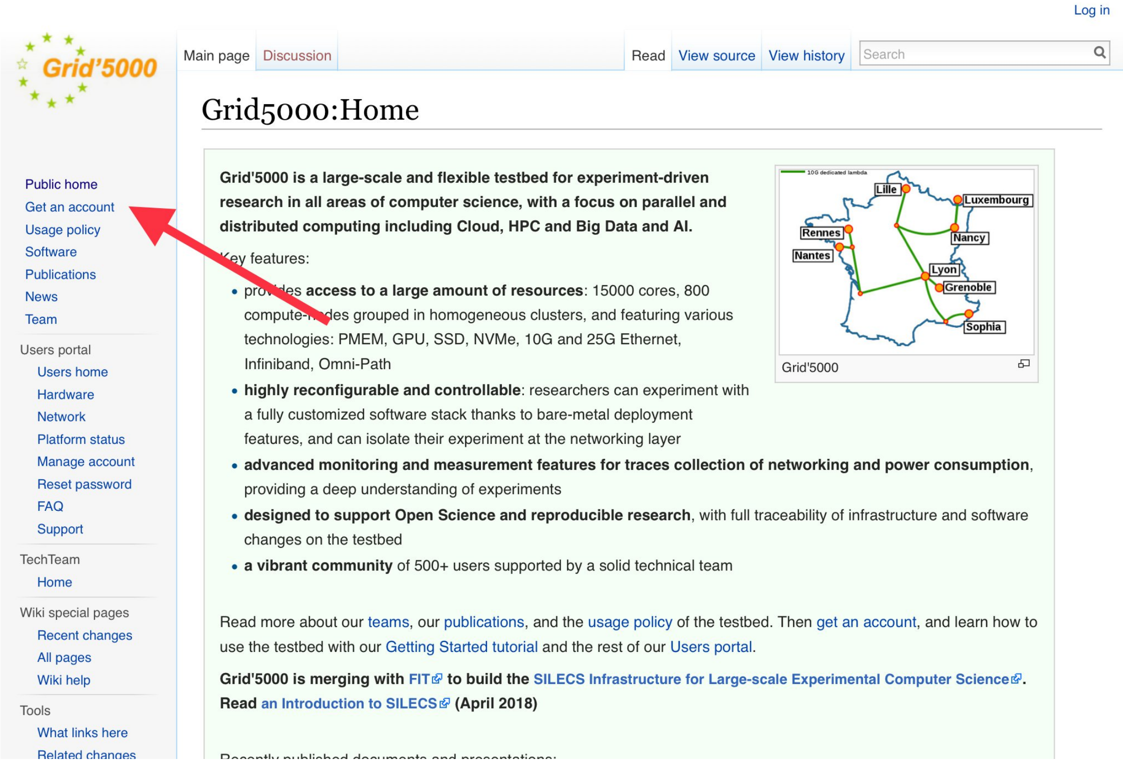
Linux Performance Observability Tools

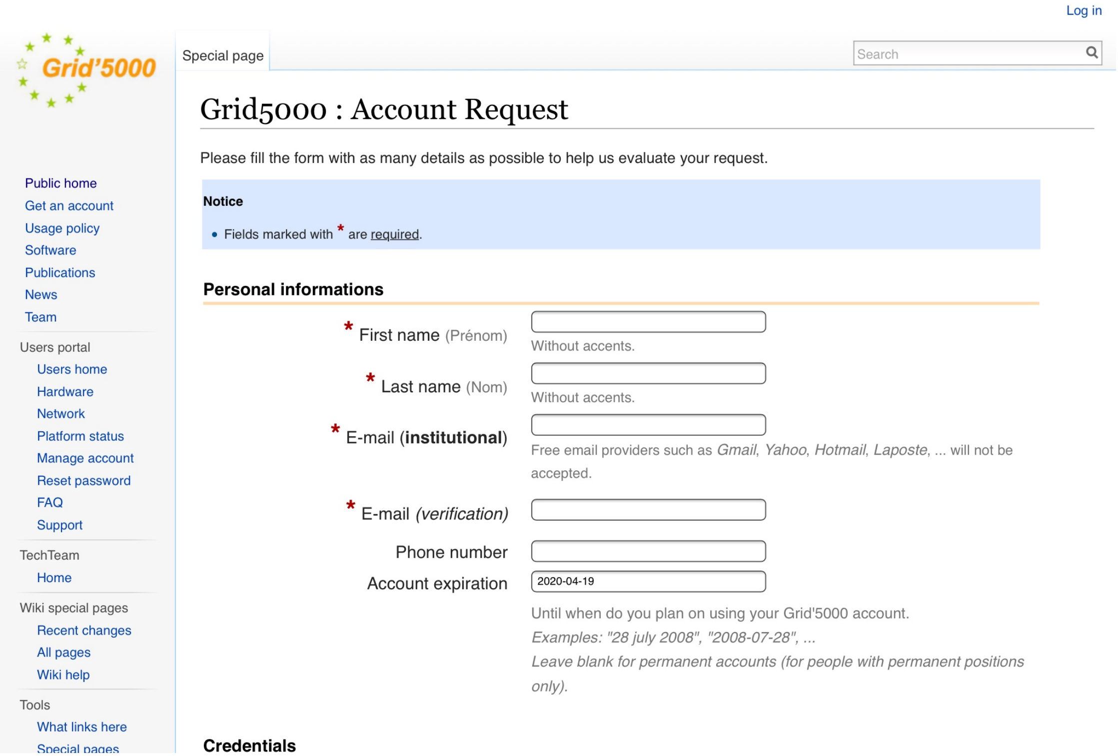


Who can use Grid'5000?

- Open to all academics in France
- Open-access programs for academics outside France

Create an account





Go to grid5000.fr → Click on "Get an account" → Fill the form

Grid'5000 usage policy

Rules for the default queue

- From 09:00 and 19:00 during working days, you can only use all the cluster for 2 hours (i.e., half of the cluster for 4h, quarter of the cluster for 8h)
- Your jobs must not cross the 09:00 and 19:00 boundaries during week days.
- You are not allowed to have more than 2 reservations in advance.

=> Large-scale jobs must be executed during nights or weekends

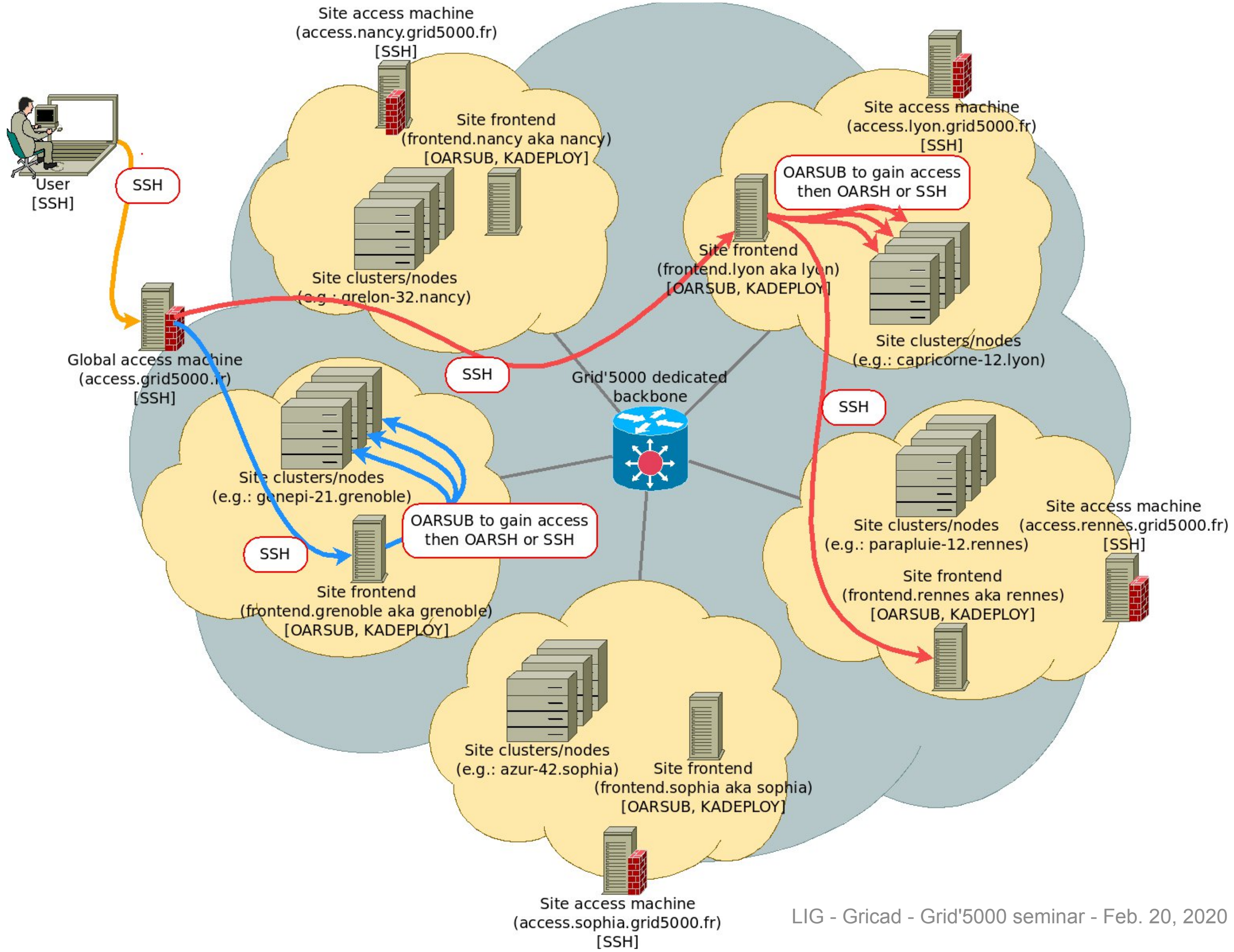
Rules for the production queue

- Suited to long-running and non-interactive jobs

Special cases

- You can request a special permission if your intended usage does not fit within the usage policy

How to access frontends / nodes



SSH configuration

```
Host g5k

 User USERNAME
 Hostname access.grid5000.fr
 ForwardAgent no

Host *.g5k
 User USERNAME
 ProxyCommand ssh g5k -W
"$(basename %h .g5k):%p"
 ForwardAgent no
```

Accessing the frontend

```
ssh grenoble.g5k
```

Accessing a node

```
ssh graphite-3.nancy.g5k
```

Where to find documentation?

Platform description

- Hardware description : https://www.grid5000.fr/w/Hardware
- Network description : https://www.grid5000.fr/w/Grid5000:Network

Basic Tutorials

- Getting started tutorial : https://www.grid5000.fr/w/Getting_Started
- Advanced OAR: https://www.grid5000.fr/w/Advanced_OAR
- Advanced Kadeploy: https://www.grid5000.fr/w/Advanced_Kadeploy
- Kavlan: https://www.grid5000.fr/w/KaVLAN

Advanced Tutorials https://www.grid5000.fr/w/Category:Portal:User

- Environment creation, HPC, Virtualization, Enos, PMEM, Kubernetes, Grid5000 Rest API

Contacts

- support-staff@lists.grid5000.fr
- users@lists.grid5000.fr

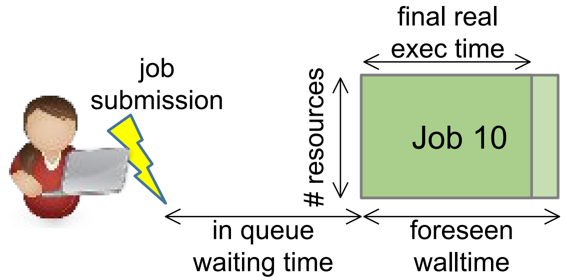
Some common concepts for all platforms

Pierre Neyron (Pimlig)

Common usage tips for large scale platforms

- Account creation (1 per user !):
  - Often require to associate to a scientific project
  - Project to be validated at some point by a referent
  - Needed for accounting, statistics, reports ; also associate list of publications
    - to justify funding
    - to allow a fair sharing of resources
- SSH access:
  - access via a frontend, bastion ⇒ several hops, but not a problem with SSH (→ ProxyCommand)
  - data transfers ⇒ direct rsync thanks to SSH, usually a good bandwidth (10Gbps)
  - network isolation ? → access services, jupyter, ... ⇒ VPN/SSH tunnels to the rescue if not direct
- User account homedir and data
  - Often different from the laptop/workstation ⇒ but allows for a clean setup (no mix with office tools, ...), optimized configuration (shell configs, etc...)
  - Once account is setup on a large shared platform → no need to multiple accounts to access many pieces of hardware

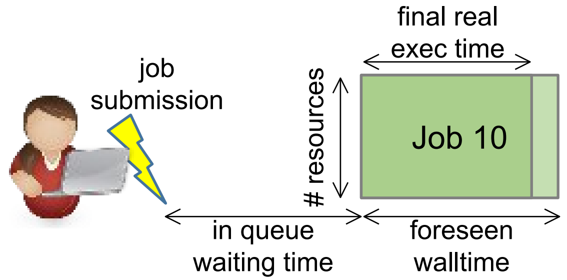
The Resources and Jobs Managment System → the batch scheduler (1/3)



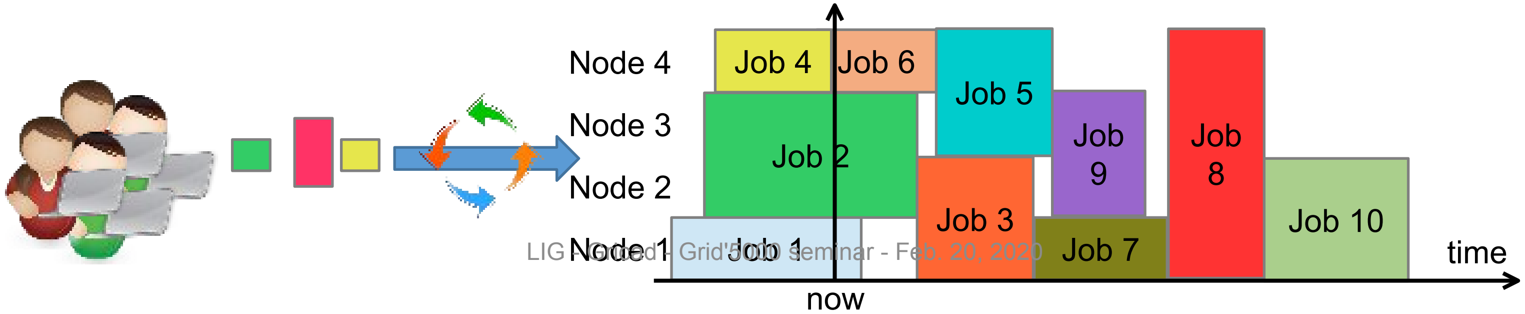
- A compute server cluster is composed of **"resources"**:
  - Resources are the set of machines (aka. **nodes** or hosts), containing multi-cores CPUs, and possibly GPUs
  - Those resources are the objects that the "**batch scheduler**" manages
  - If heterogeneous, resources can have characterizing properties: memory size, CPU type, etc...

- Compute tasks of the users have to be submitted to the system, they are **"Jobs"**
  - Users submit jobs to the system with the desired specifications:
    - a program to execute
    - a definition of the wanted resources: what kind and how many (but not what machine)
    - estimated maximum duration of the execution, aka. "**walltime"**
  - Jobs are queued and scheduled accordingly to the scheduling policy,
  - And rescheduled upon any event (early termination of a job, new job, resource failure, etc.)

The Resources and Jobs Managment System → the batch scheduler (2/3)



- Life & death of a jobs
  - Jobs start time and placement on resources is decided by the system
  - Jobs are execution envelops, identified by a number, e.g. "job 42"
  - Within the execution envelop, the user's program is responsible for exploiting the allocated resources (multi-core, multi-host, ...)
  - The execution envelop is a confinement (cannot use not assigned resources)
  - The job's program must terminate before its execution time exceeds the walltime, or is killed
  - Resources are cleaned-up between jobs
  - Many other features described in the tools manuals and platforms documentations
- Usage policies
  - to allow a fair sharing, the policy may enforce constraints (depends on each platform policy)
    - max walltime, max walltime*resources, max # jobs in queue, restricted access to some resources, ...



The Resources and Jobs Managment System → the batch scheduler (3/3)

- All compute platforms use a RJMS/Batch scheduler
- May be a same software *but different settings&policy on each platform* (e.g. OAR), or different software (e.g. SLURM)
  - GPU LIG: OAR
    - https://intranet.liglab.fr/en/it-resources/gpu-servers
  - Gricad/HPC: OAR
    - https://gricad-doc.univ-grenoble-alpes.fr/hpc/
    - https://ciment.ujf-grenoble.fr/wiki
  - Genci/Jean Zay: SLURM
    - http://www.idris.fr/eng/jean-zay/ -> see batch jobs, Slurm
  - Grid'5000: OAR (but usage policy favors interactivity and fixed date advance reservations)
    - https://www.grid5000.fr/w/Grid5000:UsagePolicy
    - https://www.grid5000.fr/w/Getting_Started

- OAR job submission example:

```
$ oarsub -l host=2/gpu=4 -p "gpu_model = 'V100'" "./run_my_xp.sh"
```

Platforms at a glance

	GPU LIG	Gricad	Genci/Jean Zay	Grid'5000
Easy first access to the platform	★★(★)	★★	★	★★
"Freedom", play almost like on the workstation	★★	★★	★	★★★★
Horse power	★	★★★	★★★★	★★★
HPC services	.	★★★	★★★	★★
Compute server count	6	~300	~2000 for JZ + others	~800
GPU count	20 GeForce 1080 Ti 2 RTX 6000	12 V100 9 Kepler	1000 V100	0 in Grenoble 180 in other sites
Storage	Team's NFS	High perf FS (beegfs, lustre), Grid FS (irods)	Different levels from high perf to resilient	Dedicated NFS + bare metal
Job manager	OAR	OAR	SLURM	OAR
User software environment	barely unix	modules/Nix/conda	modules/conda/...	XP Metal-as a Service, HPC with Spack...