Testbeds for Reproducible Research

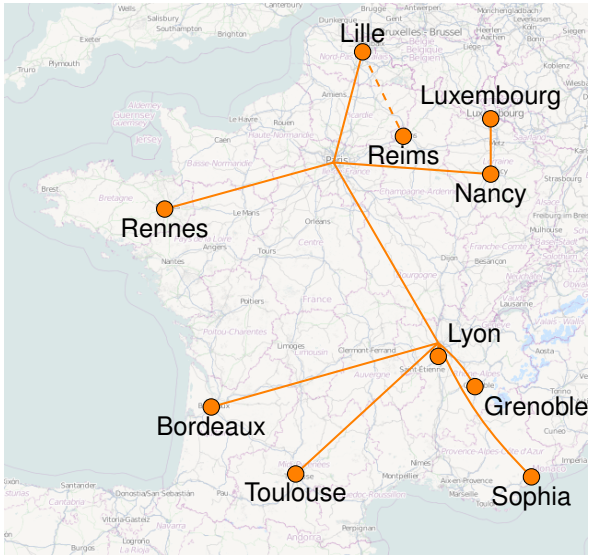
Lucas Nussbaum
lucas.nussbaum@loria.fr

Outline

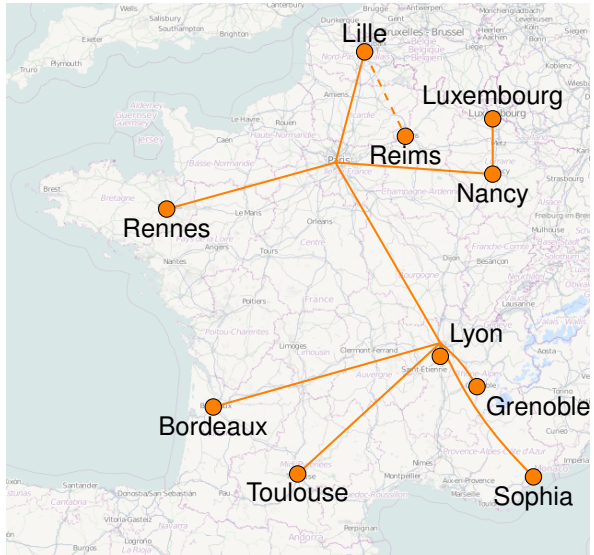
1. Presentation of Grid'5000
2. A look at two recent testbeds:
   - CloudLab
   - Chameleon

The Grid'5000 testbed

- **World-leading testbed for HPC & Cloud**
  - 10 sites, 1200 nodes, 7900 cores
  - Dedicated 10-Gbps backbone network
  - 550 users and 100 publications per year



The Grid'5000 testbed



- **World-leading testbed for HPC & Cloud**
  - 10 sites, 1200 nodes, 7900 cores
  - Dedicated 10-Gbps backbone network
  - 550 users and 100 publications per year
- Not a typical grid / cluster / Cloud:
  - Used by CS researchers for HPC / Clouds / Big Data research
    $\rightsquigarrow$ No users from computational sciences
  - **Design goals:**
    - ★ **Large-scale, shared infrastructure**
    - ★ **Support high-quality, reproducible research on distributed computing**

Outline

1. Description and verification of the environment
2. Resources selection and reservation
3. Reconfiguring the testbed to meet experimental needs
4. Monitoring experiments, extracting and analyzing data

Description and verification of the environment

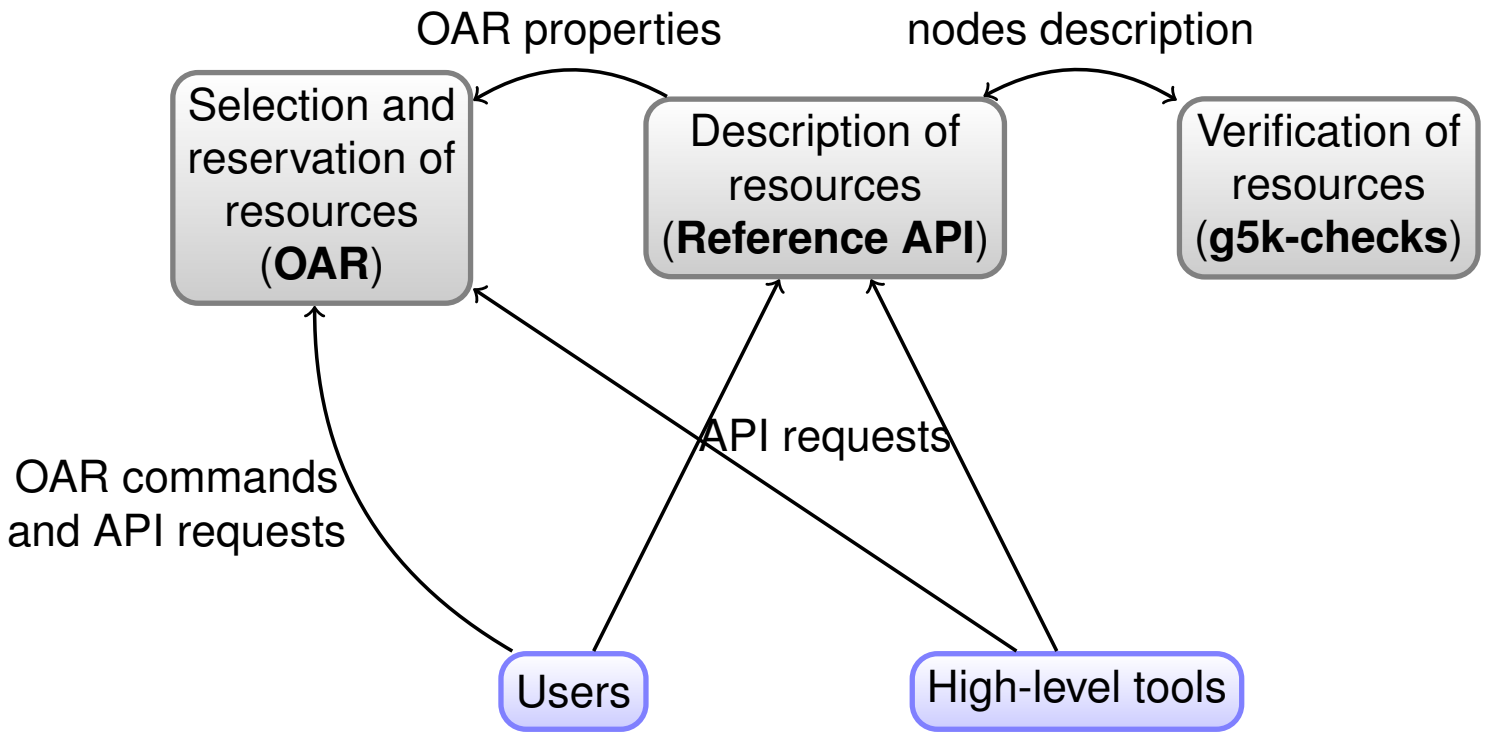
Typical needs:

- How can I find suitable resources for my experiment?
- How sure can I be that the actual resources will match their description?
- What was the hard drive on the nodes I used six months ago?

Description and verification of the environment

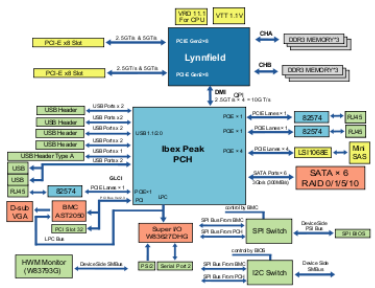
Typical needs:

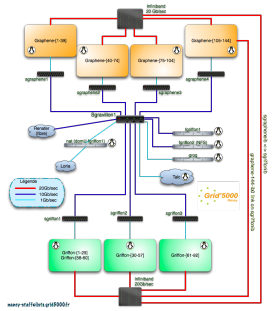
- How can I find suitable resources for my experiment?
- How sure can I be that the actual resources will match their description?
- What was the hard drive on the nodes I used six months ago?



Description of resources

- Describing resources $\rightsquigarrow$ understand results
  - Detailed description on the Grid'5000 wiki
  - Machine-parsable format (JSON)
  - Archived (*State of testbed 6 months ago?*)






```
"processor": {
  "cache_l2": 8388608,
  "cache_l1": null,
  "model": "Intel Xeon",
  "instruction_set": "",
  "other_description": "",
  "version": "X3440",
  "vendor": "Intel",
  "cache_l1i": null,
  "cache_l1d": null,
  "clock_speed": 2530000000.0
},
"uid": "graphene-1",
"type": "node",
"architecture": {
  "platform_type": "x86_64",
  "smt_size": 4,
  "smp_size": 1
},
"main_memory": {
  "ram_size": 17179869184,
  "virtual_size": null
},
"storage_devices": [
  {
    "model": "Hitachi HDS72103",
    "size": 298023223876.953,
    "driver": "ahci",
    "interface": "SATA II",
    "rev": "JPFO",
    "device": "sda"
  }
],
```

Verification of resources

- Inaccuracies in resources descriptions $\leadsto$ dramatic consequences:
  - Mislead researchers into making false assumptions
  - Generate wrong results $\leadsto$ retracted publications!
- Happen frequently: maintenance, broken hardware (e.g. RAM)

Verification of resources

- Inaccuracies in resources descriptions $\rightsquigarrow$ dramatic consequences:
  - Mislead researchers into making false assumptions
  - Generate wrong results $\rightsquigarrow$ retracted publications!
- Happen frequently: maintenance, broken hardware (e.g. RAM)
- Our solution: g5k-checks
  - Runs at node boot (can also be run manually by users)
  - Retrieves current description of node in Reference API
  - Acquires information on node using OHAI, ethtool, etc.
  - Compares with Reference API

Outline

1. Description and verification of the environment
2. Resources selection and reservation
3. Reconfiguring the testbed to meet experimental needs
4. Monitoring experiments, extracting and analyzing data

Resources selection and reservation

- Roots of Grid'5000 in the HPC community
  $\rightsquigarrow$ Obvious idea to use a HPC Resource Manager
- OAR (developed in the context of Grid'5000)
  `http://oar.imag.fr/`
- Supports resources properties ($\approx$ tags)
  - Can be used to select resources (multi-criteria search)
  - Generated from Reference API
- Supports advance reservation of resources
  - In addition to typical HPC resource managers's *batch* mode
  - Request resources at a specific time
  - On Grid'5000: used for special policy:
    Large experiments during nights and week-ends
    Experiments preparation during day

Using properties to reserve specific resources

Reserving two nodes for two hours. Nodes must have a GPU and power monitoring:

```
oarsub -p "wattmeter='YES' and gpu='YES'" -l nodes=2,walltime=2 -I
```

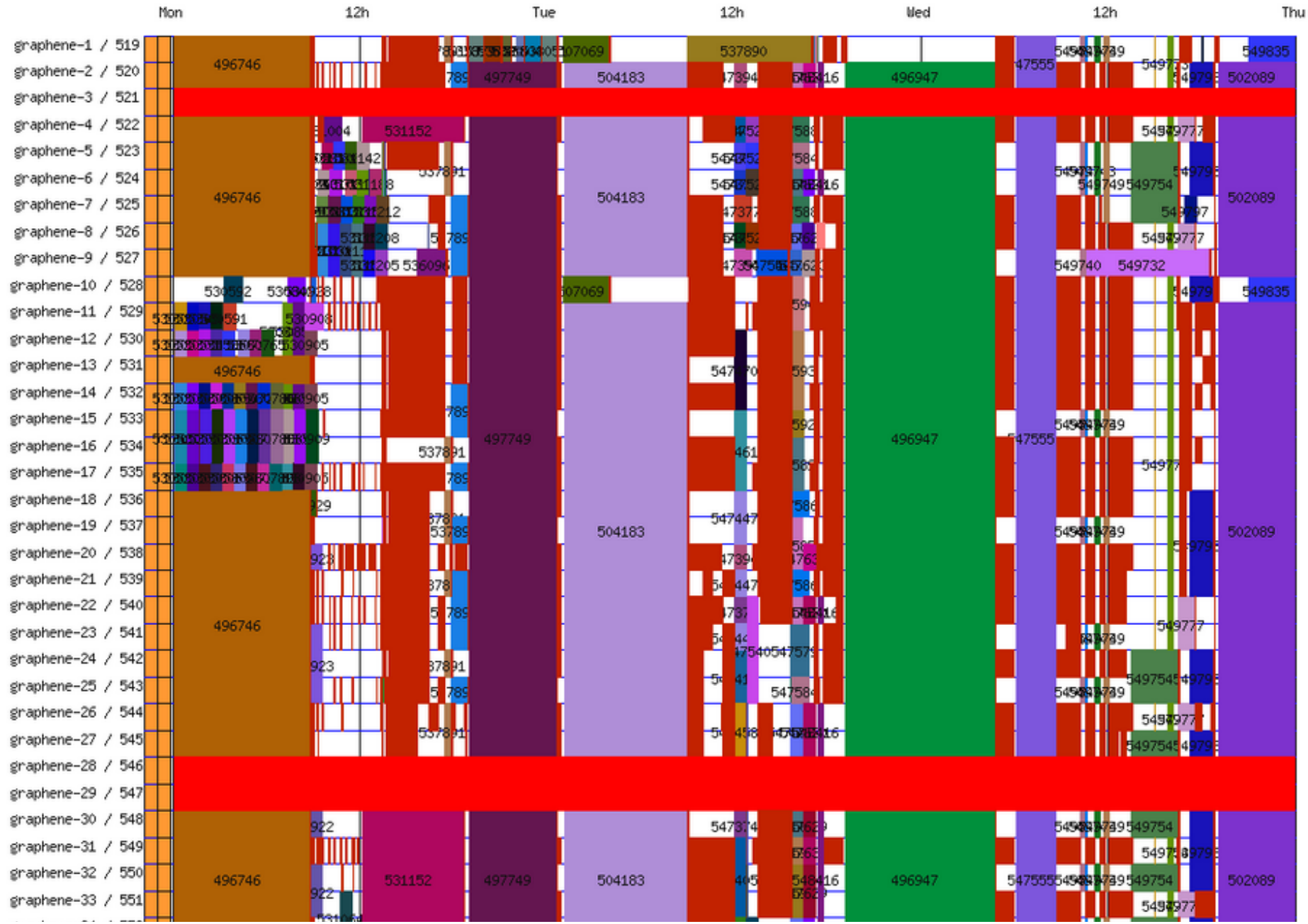
Reserving one node on cluster a, and two nodes with a 10 Gbps network adapter on cluster b:

```
oarsub -l "{cluster='a'}/nodes=1+{cluster='b' and eth10g='Y'}/nodes=2,walltime=2"
```

Advance reservation of 10 nodes on the same switch with support for Intel VT (virtualization):

```
oarsub -l "{virtual='ivt'}/switch=1/nodes=10,walltime=2" -r '2014-11-08 09:00:00'
```

Visualization of usage



Outline

1. Description and verification of the environment
2. Resources selection and reservation
3. Reconfiguring the testbed to meet experimental needs
4. Monitoring experiments, extracting and analyzing data

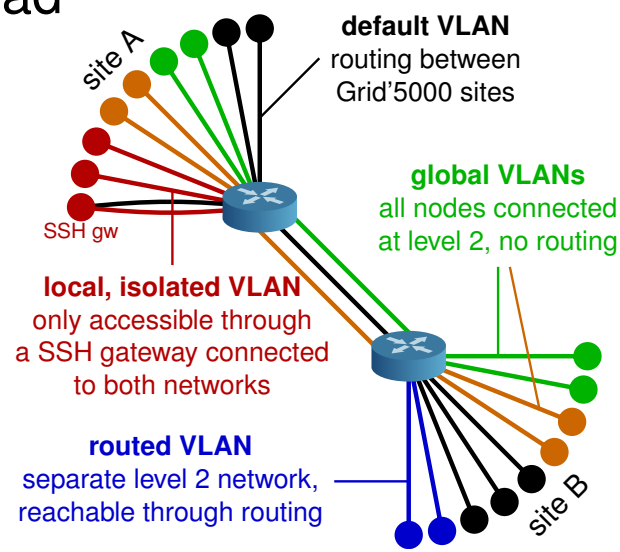
Reconfiguring the testbed

- Typical needs:
  - How can I install $SOFTWARE on my nodes?
  - How can I add $PATCH to the kernel running on my nodes?
  - Can I run a custom MPI to test my fault tolerance work?
  - How can I experiment with that Cloud/Grid middleware?
  - Can I get a stable (over time) software environment for my experiment?

Reconfiguring the testbed

- Operating System reconfiguration 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 (120s with Kexec)
- Customize networking environment with KaVLAN
  - Deploy intrusive middlewares (Grid, Cloud)
  - Protect the testbed from experiments
  - Avoid network pollution
  - By reconfiguring VLANS $\rightsquigarrow$ almost no overhead
  - Recent work: support several interfaces

KADEPLOY



Creating and sharing Kadeploy images

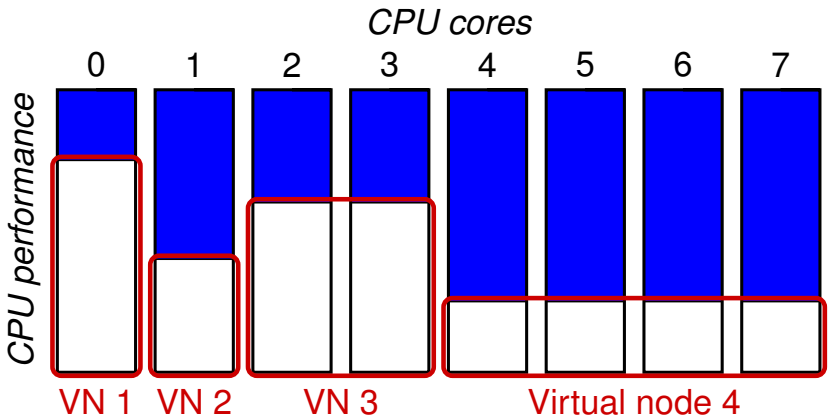
- Avoid manual customization:
  - Easy to forget some changes
  - Difficult to describe
  - The full image must be provided
  - Cannot really reserve as a basis for future experiments (similar to binary vs source code)

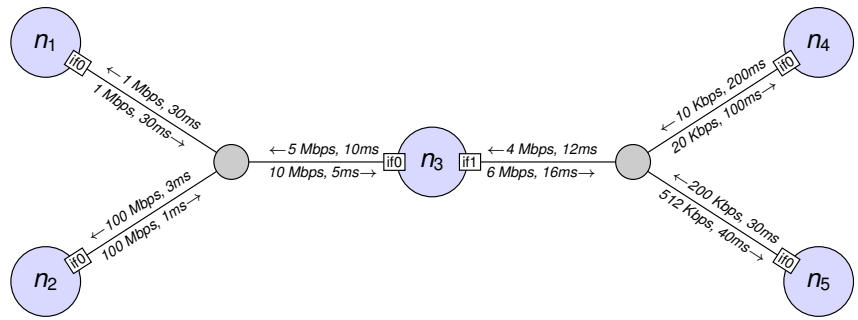
- Kameleon: Reproducible generation of software appliances
  - Using *recipes* (high-level description)
  - Persistent cache to allow re-generation without external resources (Linux distribution mirror) $\rightsquigarrow$ self-contained archive
  - Supports Kadeploy images, LXC, Docker, VirtualBox, qemu, etc.

http://kameleon.imag.fr/

Changing experimental conditions

- Reconfigure experimental conditions with Distem
  - Introduce heterogeneity in an homogeneous cluster
  - Emulate complex network topologies





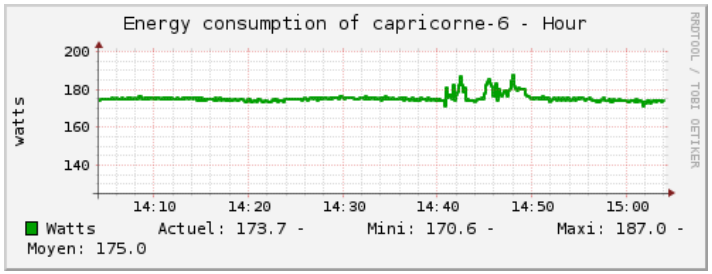
http://distem.gforge.inria.fr/

Outline

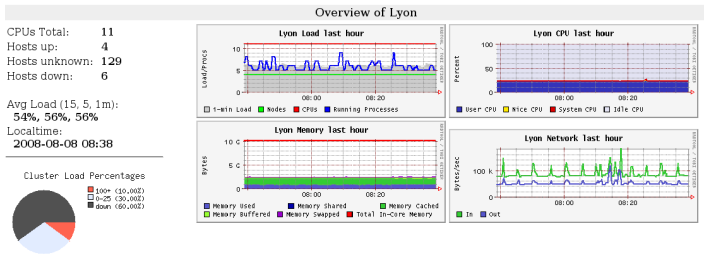
1. Description and verification of the environment
2. Resources selection and reservation
3. Reconfiguring the testbed to meet experimental needs
4. Monitoring experiments, extracting and analyzing data

Monitoring experiments

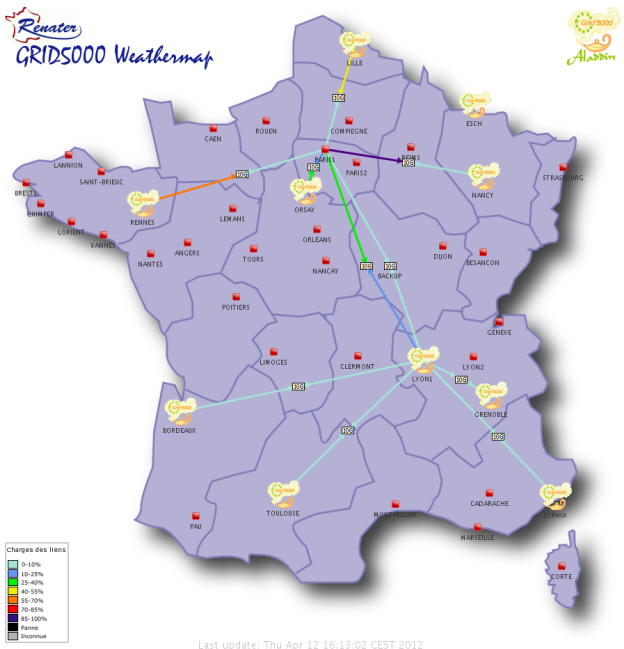
Goal: enable users to understand what happens during their experiment



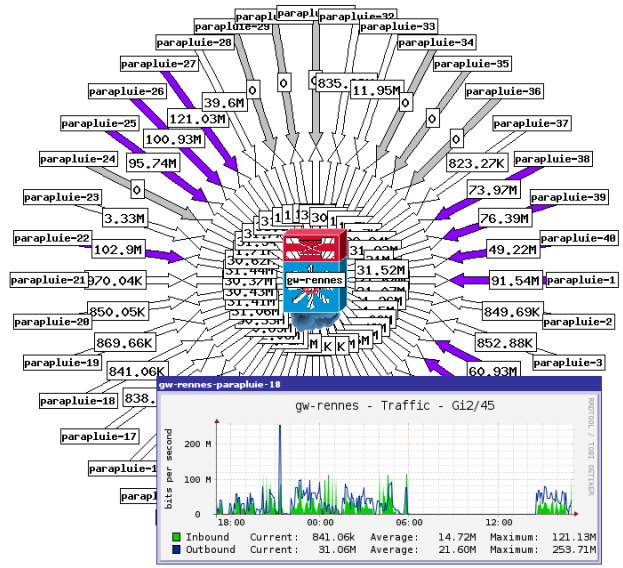
Power consumption



CPU – memory – disk



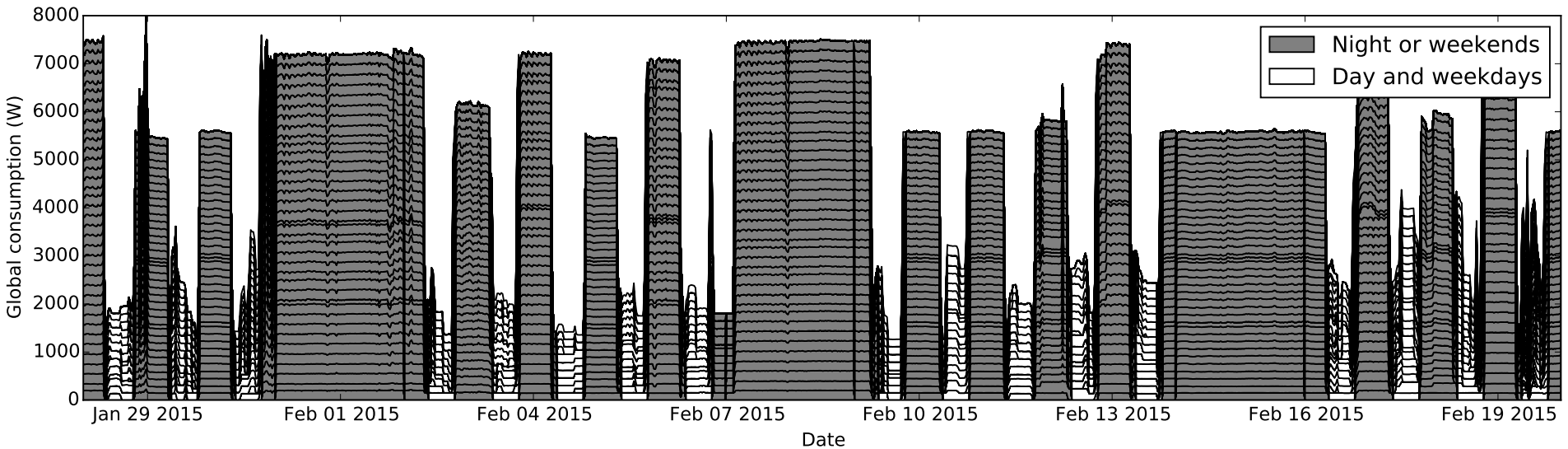
Network backbone



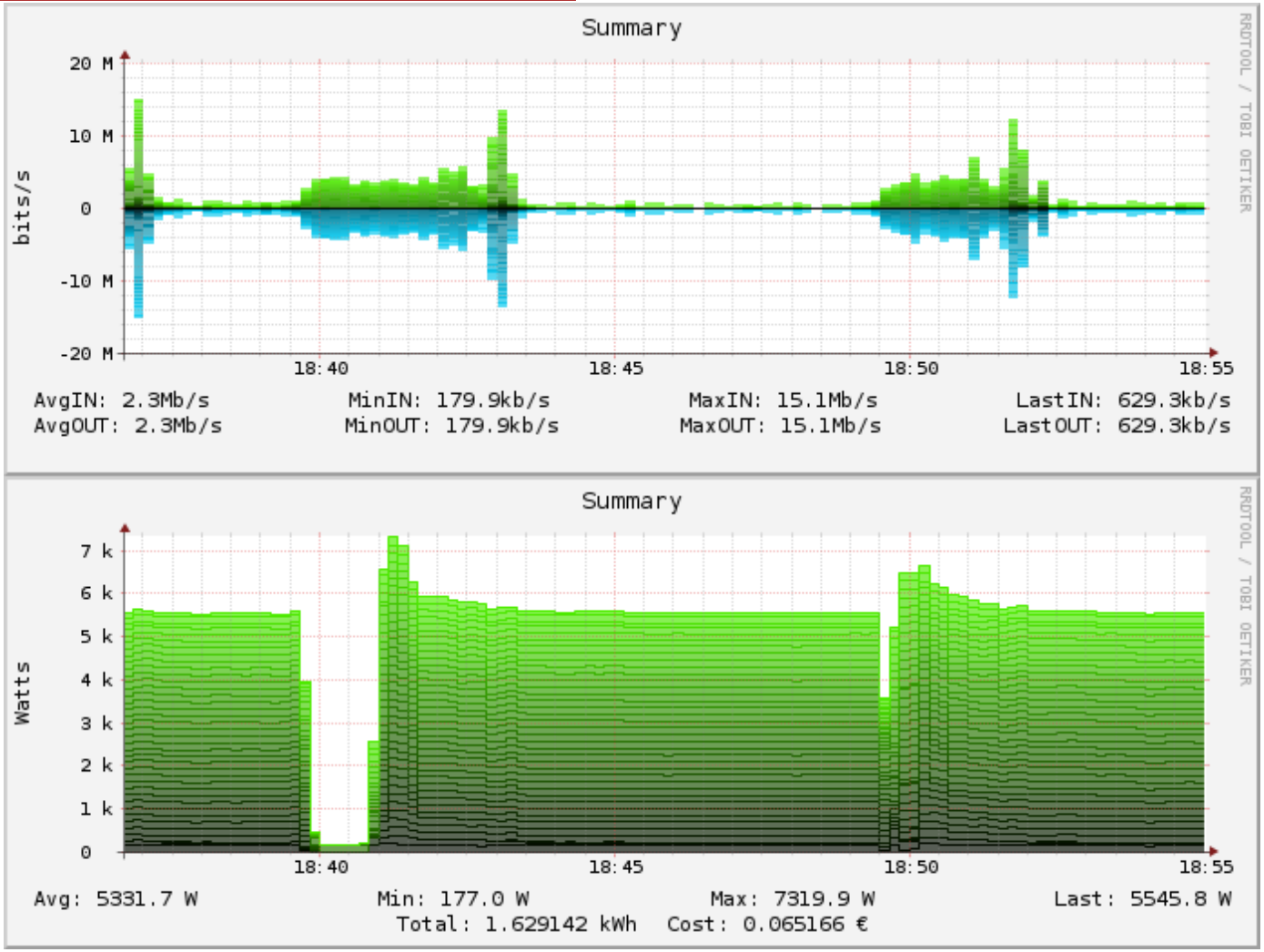
Internal networks

Kwapi: a new framework to monitor experiments

- Initially designed as a power consumption measurement framework for OpenStack – then adapted to Grid'5000's needs and extended
- For energy consumption and network traffic
- Measurements taken at the infrastructure level
  (SNMP on network equipment, power distribution units, etc.)
- High frequency (aiming at 1 measurement per second)
- Data visualized using web interface
- Data exported as RRD, HDF5 and Grid'5000 REST API



Kwapi: example output



- 18:39:28 – machines are turned off
- 18:40:28 – machines are turned on again and generate network traffic as they boot via PXE
- 18:49:28 – machines reservation is terminated, causing a reboot to the default system

Other testbeds

- Two recent projects (Oct. 2014 $\rightsquigarrow$ Sep. 2017)
- Funded by the National Science Foundation, for 10 M$ each
- All information below TTBOMK: please correct me!

Other testbeds

- Two recent projects (Oct. 2014 $\rightsquigarrow$ Sep. 2017)
- Funded by the National Science Foundation, for 10 M$ each
- All information below TTBOMK: please correct me!
- **Chameleon**
  - Led by Kate Keahey (ANL / Univ. Chicago)
  - `https://www.chameleoncloud.org/`
- **CloudLab**
  - Led by Robert Ricci (Univ. Utah)
  - `http://www.cloudlab.us`
  - Federated with GENI
    CloudLab can be used with a GENI account, and vice-versa

Comparison

- Software stack used as a base
  - Grid'5000: mostly their own
  - Chameleon: OpenStack
  - CloudLab: Emulab
- Resources description and verification
  - Grid'5000: reference API + g5k-checks (+ human-readable description)
  - Chameleon: same as Grid'5000
  - CloudLab:
    - machine-readable description using RSpec 'advertisement' format (less detailed than Grid'5000's, though) + human-readable description in the docs
    - verification: nothing similar to g5k-checks, but LinkTest[1] can validate network configuration

[1] D.S. Anderson et al. "Automatic Online Validation of Network Configuration in the Emulab Network Testbed". In: *ICAC'06*.

Comparison (2)

- Resources reservation
  - Grid'5000: batch scheduler with advance reservation
  - Chameleon: leases using OpenStack Blazar
  - CloudLab: experiments start immediately, default duration of a few hours, can be extended on demand (no advance reservations)
- Resources reconfiguration / software
  - Grid'5000: Kadeploy
  - Chameleon: OpenStack Ironic
  - CloudLab: Emulab's Frisbee
- Network reconfiguration and Software Defined Networking
  - Grid'5000: KaVLAN (+ higher level tools)
  - Chameleon: planned, using OpenFlow
  - CloudLab: yes:
    - Emulab's network emulation features
    - OpenFlow access on switches[2]
    - Interconnection to Internet2's AL2S

[2] `http://cloudlab-announce.blogspot.com/2015/06/using-openflow-in-cloudlab.html`

Comparison (3)

- Monitoring
  - Grid'5000: Kwapi (power + network)
  - Chameleon: planned, using OpenStack Ceilometer
  - CloudLab: planned[3]
- Long term storage between experiments
  - Grid'5000: storage5k (file-based and block-based)
  - Chameleon: object store (OpenStack Swift) available soon
  - CloudLab: yes[4], with snapshots (using ZFS) to version data (the snapshots features are not documented yet)

[3] `http://docs.cloudlab.us/planned.html`

[4] `http://cloudlab-announce.blogspot.fr/2015/04/persistant-dataset.html`

Conclusions

- We are moving
  - From small testbeds, on a per-team/per-lab basis
  - To large-scale shared infrastructures built with reproducibility in mind
- A bright and exciting future
- Paving the way to Open Science of HPC and Cloud!
- (Also: you can get accounts on all of them through Open Access / Preview / Early users programs)

One could determine the age of a science by looking at the state of its measurement tools.

Gaston Bachelard – *La formation de l'esprit scientifique*, 1938

Bibliography

- Resources management: Resources Description, Selection, Reservation and Verification on a Large-scale Testbed. http://hal.inria.fr/hal-00965708
- Kadeploy: Kadeploy3: Efficient and Scalable Operating System Provisioning for Clusters. http://hal.inria.fr/hal-00909111
- KaVLAN, Virtualization, Clouds deployment:
  - Adding Virtualization Capabilities to the Grid'5000 testbed. http://hal.inria.fr/hal-00946971
  - Enabling Large-Scale Testing of IaaS Cloud Platforms on the Grid'5000 Testbed. http://hal.inria.fr/hal-00907888
- Kameleon: Reproducible Software Appliances for Experimentation. https://hal.inria.fr/hal-01064825
- Distem: Design and Evaluation of a Virtual Experimental Environment for Distributed Systems. https://hal.inria.fr/hal-00724308
- Kwapi: A Unified Monitoring Framework for Energy Consumption and Network Traffic. https://hal.inria.fr/hal-01167915
- XP management tools:
  - A survey of general-purpose experiment management tools for distributed systems. https://hal.inria.fr/hal-01087519
  - XPFlow: A workflow-inspired, modular and robust approach to experiments in distributed systems. https://hal.inria.fr/hal-00909347
  - Using the EXECO toolbox to perform automatic and reproducible cloud experiments. https://hal.inria.fr/hal-00861886
  - Expo: Managing Large Scale Experiments in Distributed Testbeds. https://hal.inria.fr/hal-00953123