Working efficiently on Grid'5000

Lucas Nussbaum
lucas.nussbaum@loria.fr

AlGorille
LORIA / Université Nancy 2

Introduction

Grid’5000 :

- A very powerful platform
- But very difficult to use
  - Especially if you want to do complex things
  - Having a Linux guru at hand helps

Goal of this talk :

Share some good practices and tips gathered over 5 years of Grid’5000 usage

Outline

- Authenticating
- Connecting and moving around
- Moving files around
- Editing files on Grid’5000
- Reserving resources
- Doing complex experiments
  Automating, scalable tools, gotchas

Authenticating

Two different needs :

- Authenticating from the outside of Grid'5000 (your laptop)
  $\Rightarrow$ Must be very secure
- Authenticating inside Grid'5000
  $\Rightarrow$ Must be simple and scriptable

Recommended solution : 2 pairs of SSH keys

- Key O : authenticate from the outside of Grid'5000
  With passphrase (use ssh-agent)
- Key I : authenticate inside Grid'5000
  Without passphrase

Putting all files into place

On your laptop :

- `.ssh/id_rsa` : private part of Key O

On Grid'5000 :

- `.ssh/id_rsa` : private part of key I
- `.ssh/authorized_keys` : public parts of keys O and I

Isn't that dangerous ?

- Grid'5000 security focus : avoid attacks from outsiders
- There are already many ways for a Grid'5000 user to impersonate another Grid'5000 user

Connecting and moving around

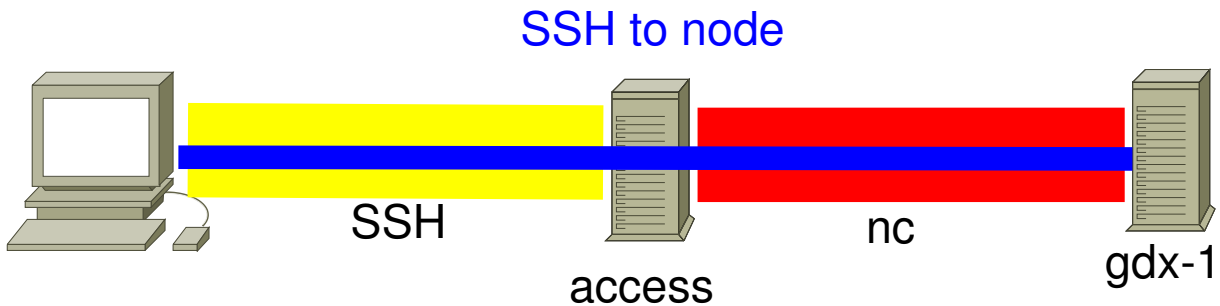
Say you want two SSH sessions on gdx-1 and gdx-2 :

- Open a terminal
- `ssh lnussbaum@acces.site.grid5000.fr`
- ~~Enter password~~
- `ssh root@gdx-1.orsay`
- ~~Enter password~~

- Open another terminal
- `ssh lnussbaum@acces.site.grid5000.fr`
- ~~Enter password~~
- `ssh root@gdx-2.orsay`
- ~~Enter password~~

Tip : use SSH ProxyCommand

- Command that provides a connection to an SSH server
- Simple example :
  `ssh -o "ProxyCommand=nc localhost 22" elysee.fr`
- Idea : use it to start another SSH connection, and run nc on the access node
  `ssh -o "ProxyCommand=ssh acces.`*`site`*`.grid5000.fr nc gdx-1 22" foo`
  ⇒ Connected to gdx-1 in one command !



Tip : use SSH ProxyCommand (2)

With some more magic, in `.ssh/config` :

```
Host *.g5k
        User lnussbaum
        ProxyCommand ssh acces.site.grid5000.fr \
            "nc -q 10 \$(basename %h .g5k) %p"
        BatchMode no
        StrictHostKeyChecking no
```

Connect to any Grid'5000 node in one command :

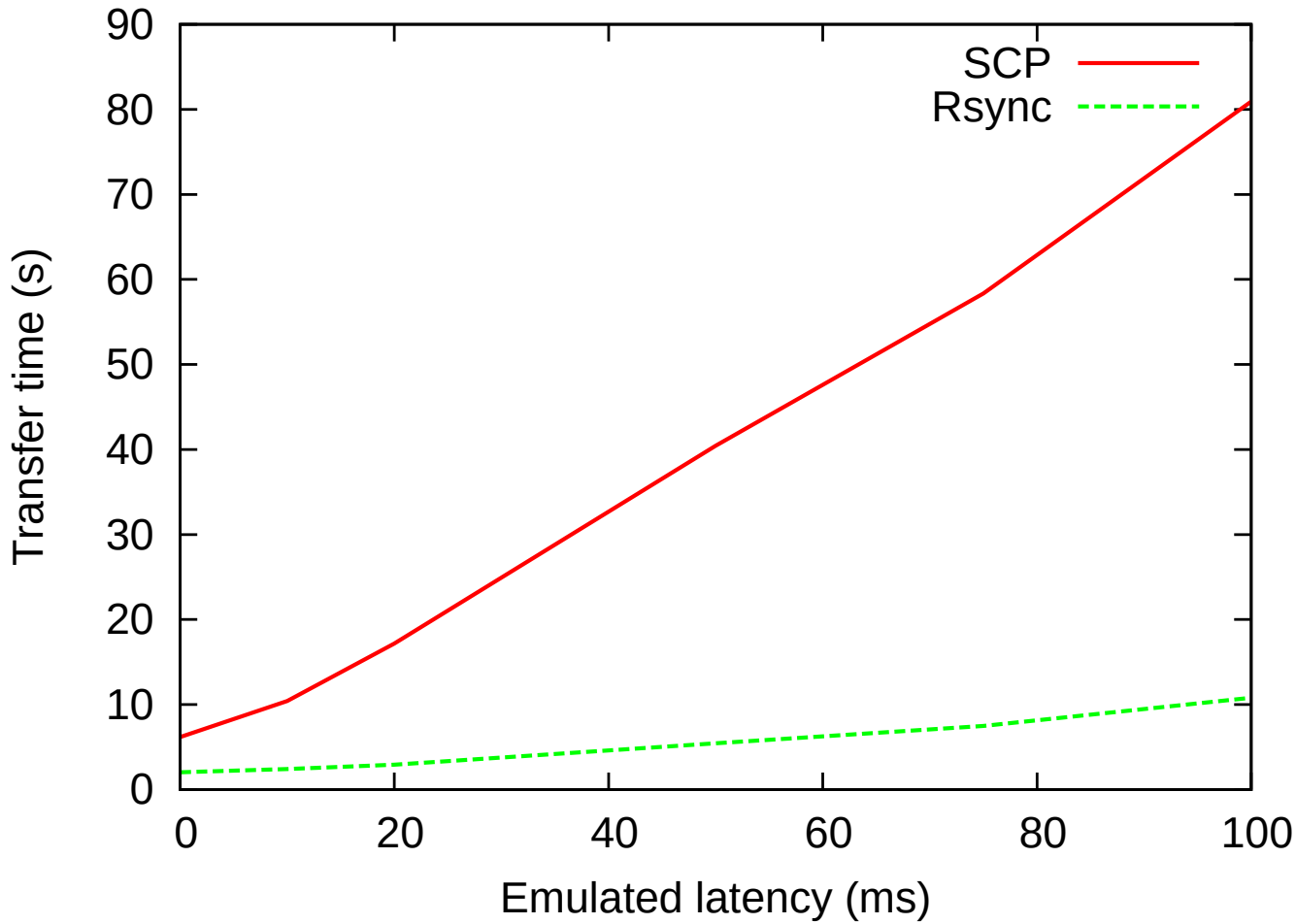
- `ssh lyon.g5k`
- `ssh rennes.g5k`
- `ssh gdx-1.orsay.g5k`

Transferring files to/from Grid'5000

- ProxyCommand works with everything SSH-based
  - scp, sftp, rsync
  - `sftp rennes.g5k:` just works
- Use rsync, not scp
  - Pipelined file transfers
  - Much more efficient on networks with large BDP (high bandwidth, high latency)

SCP vs Rsync

Transfer of 120 files (total : 2.1 MB) with SCP and Rsync
Bandwidth and Latency controlled using network emulator



Sync’ing your $HOME between sites

- Please don’t
- There’s not enough disk space for everybody to do that
- Instead : synchronize a subset of your files
  - Configuration files (inc. SSH keys)
  - Scripts

```
for s in bordeaux grenoble lille lyon nancy
  orsay rennes sophia toulouse; do
      rsync -aP source/ $s:
done
```

Editing files

- Directly on Grid'5000
  - Requires to use a console text editor
  - Fancy features not available

- On your local machine
  - Using your editor's SSH support
    ```
    vim scp://root@gdx-1.g5k/foo
    ```
  - Using sshfs
    ```
    mkdir gdx-1
    sshfs root@gdx-1.g5k:/ gdx-1
    ```

Reserving resources

First, a very important question. . .

What's the real meaning of O.A.R ?

Post-talk addition : valid answers

- *Optimal Allocation of Resources (thanks Emmanuel Thomé)*
- *PBS-1 (like HAL = IBM-1)*
- *Olivier Auguste Richard*

Reserving resources : oarsub

- `-p` accepts SQL
  Exclude some nodes :
  ```
  -p "network_address not in
  ('griffon-93.nancy.grid5000.fr',
  'griffon-94.nancy.grid5000.fr')"
  ```
- Get all nodes : `-l nodes=BEST` (undocumented ?)
- Starting a job ASAP, avoiding a reservation :
  `oarsub -l nodes=10 'sleep 86400'`
- Node list without connecting to a (deploy) job :
  `/var/lib/oar/$JOBID`

Scripting complex experiments

- Very difficult process
- But :
  - Increases reproducibility of experiments
  - Required step before running experiments unattended
- No unique Good Way

  We are not there yet, unfortunately
- Some advices to keep in mind :
  - Think your scripts for easy debugging
  - Split your experiments into steps
  - Start with independent and idempotent scripts
  - Use stable building blocks (standard tools)
  - Keep as much data as possible (verbosity, logs)

Programming languages

- Perl, Python, **Ruby**
- Not shell scripting :
  - Doesn’t handle complex data structures (lists of nodes)
  - awk, cut, grep and sed tend to be fragile

```
oarstat -fj $ID | grep assigned_hostnames |
cut -f2 -d "=" | cut -f$i -d "+" |
sed -e 's/ //g'
```

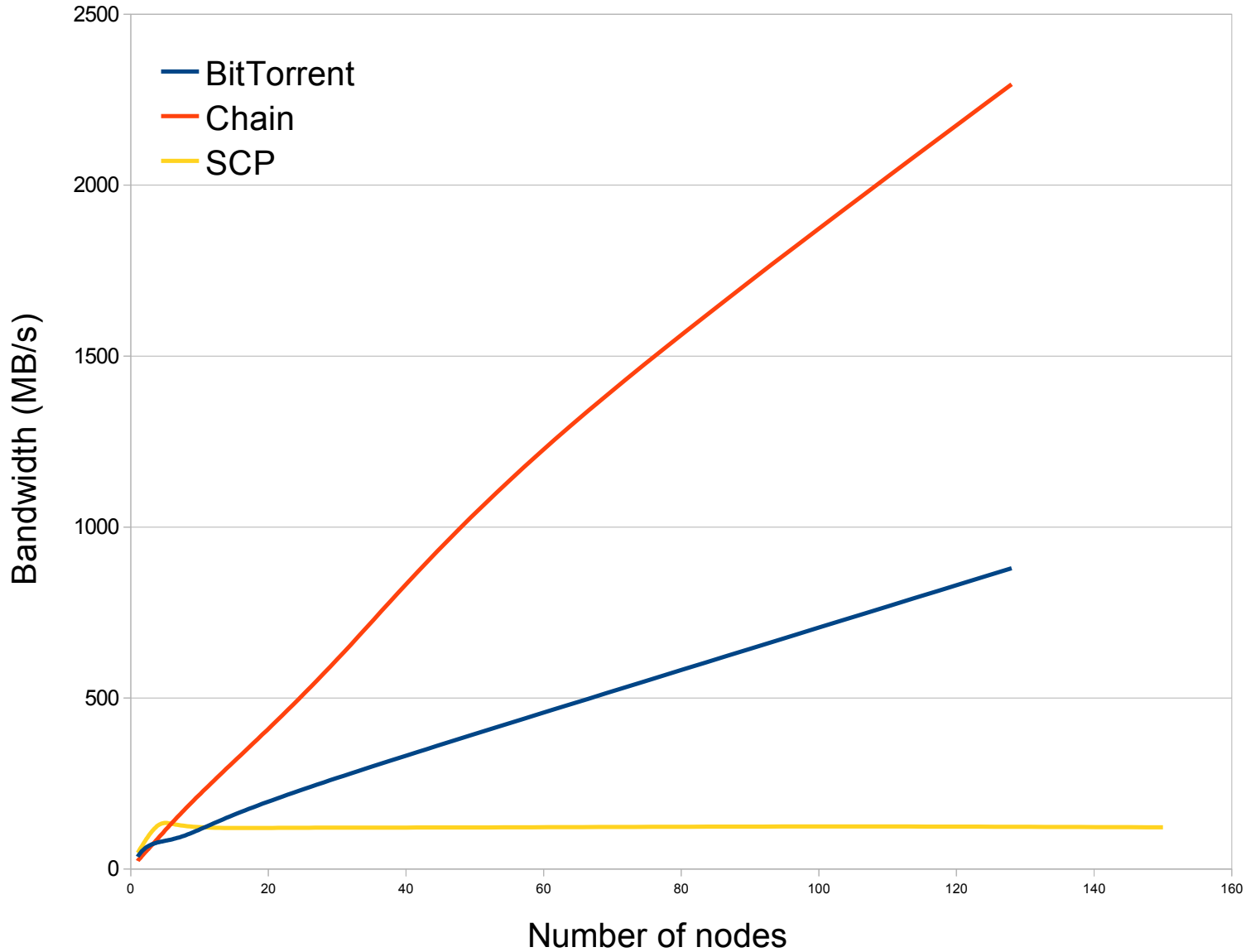
- Going further : experiment supervision tools
  Emulab’s DART, PlanetLab’s Plush, Globus’ ZENTURIO
  Grid’5000 : GRUDU [GRAAL], Expo [Videau], NXE [Guillier]

Domain-specific tools

No need to reinvent the wheel !

- **Katapult**
  Wrapper around kadeploy. Handles retries when failures + execution of script after deployment
- **Taktuk**
  "Efficiently run commands on a large number of nodes"
- Data broadcast ?
  Kastafior ? (at least not BitTorrent)

SCP vs chain vs BitTorrent



Standard tools

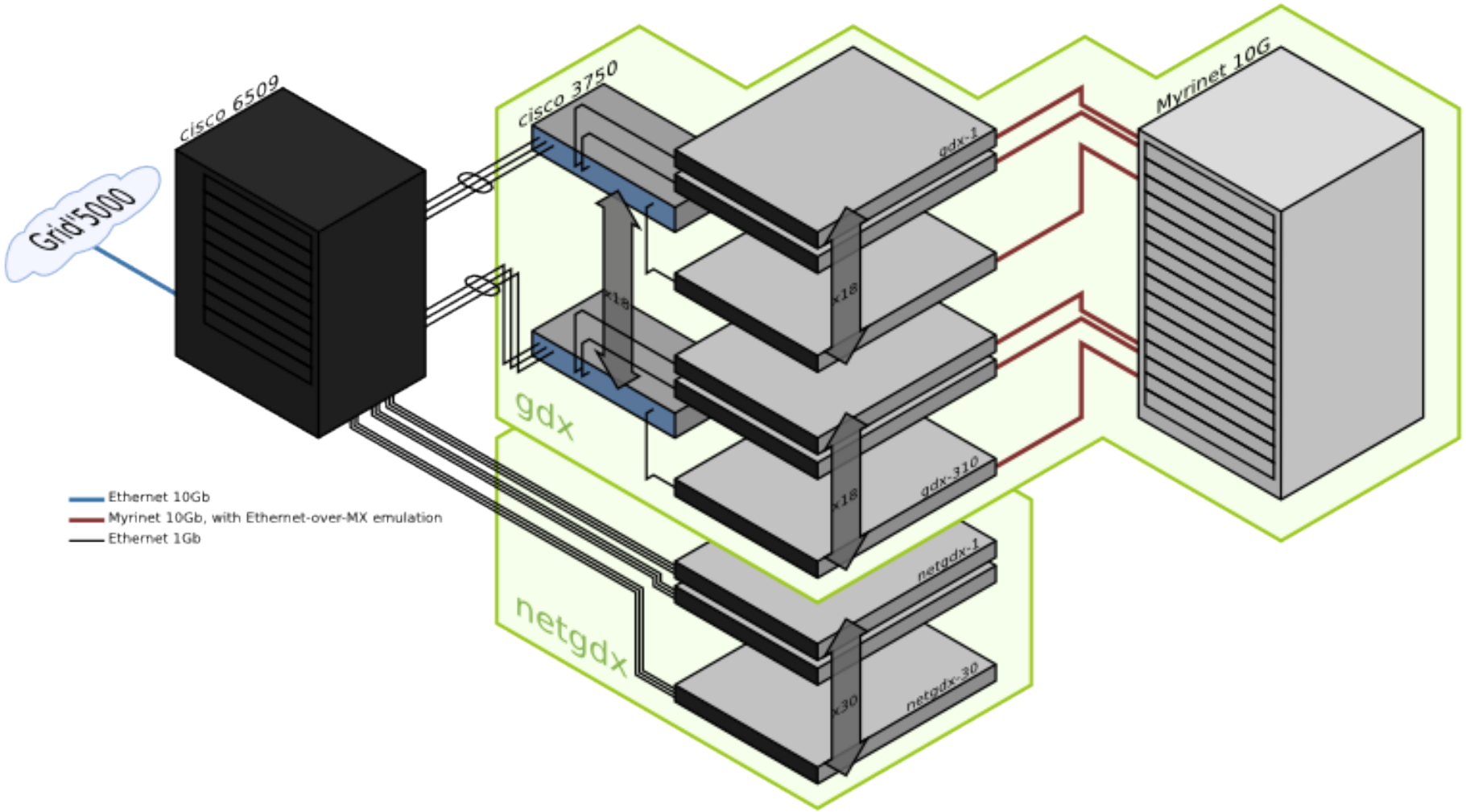
- **ssh**
  + tunnels, SOCKS proxy, etc.
- **terminator** (or alternatives)
  Several X terminals in one window
- **screen**
  Run experiment unattended, take control back when needed
- **xargs**
  Simple way to run commands in parallel (`-P`)

```
cat nodeslist |
xargs -P10 -I HOST -n1 ssh HOST hostname
```

Beware of the platform

- Some unexpected heterogeneity
  CPUs on gdx, IB card on griffon, broken memory on various nodes, hard disks
- Different production environments
  ☺ might change soon ! ☺
- Different software installed on the service nodes
  Or missing software on some service nodes
- Ethernet networks don’t scale
  Consider using *IP over Myrinet* or *IP over Infiniband*

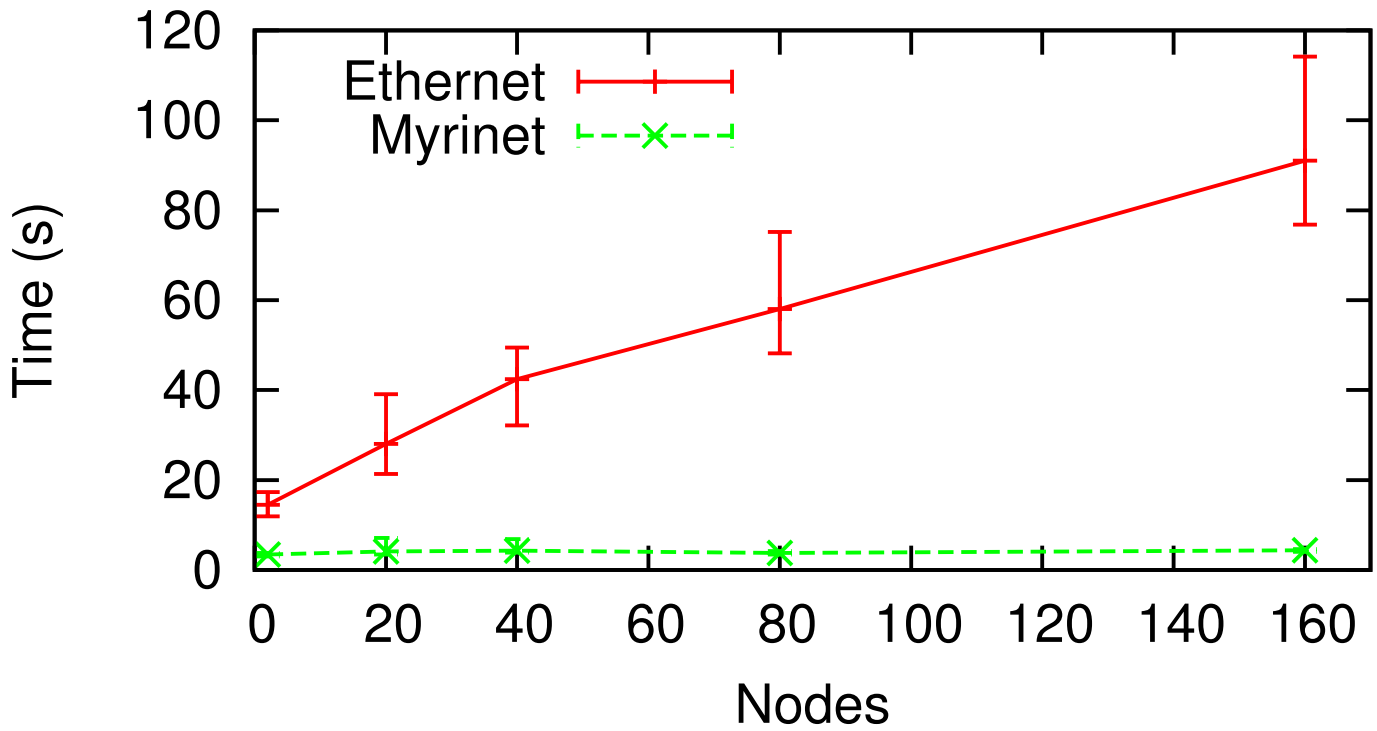
Performance of networks



gdx @ Orsay : 18 24-ports switches connected to the main switch

Performance of networks (2)

- Simultaneously, each node sends 1 GB of data to another node
- Goal : Create congestion in the cluster's networks
- Experiment on **gdx**, 2 to 160 nodes. Nodes chosen randomly.



Myrinet : "perfect" network

How you can be a better user

- Fill in your user report
- Report the problems you encounter
  `users@lists.grid5000.fr`
  or Bugzilla if you are sure of what you are doing
- Talk to (or join) the technical committee

Talk to (or join) the technical committee

- Sysadmins are not Grid'5000 users
  (Except just before the spring school)
- They don't know about the problems you face
- They desperately need feedback !
  Useful changes are not done because of lack of user pressure
- It's quite easy to influence Grid'5000 design choices
  Get Grid'5000 changed to suit your needs ! ☺
- Many interesting discussions and bugs

First step : subscribe to `devel@lists.grid5000.fr`

Things to watch for

- Grid’5000 API
  Will help to script experiments, especially multi-sites ones
- Common production environment
  The opportunity to get the applications you need installed
- Metroflux, KaVLAN
  Will enable more interesting experiments
- Globalization of some services (access machines, etc)
  Will change the way you use Grid’5000 on a daily basis
- grid5000-code
  Repository of user-contributed code
  Soon to be replicated on all front nodes ; in default $PATH

Wrap up

- Grid’5000 is a fantastic tool for your research
- Mastering it is challenging
- But it’s worth it
- Don’t be a passive, silent user
  - Report problems
  - Provide feedback

Questions ? Other good practices to share ?