Building a GNU/Linux distribution with DevOps in mind

Daniel Dehennin

Pôle de Compétences Logiciels Libres

OpenNebulaConf 2016

Pôle de Compétences Logiciels Libres

FOSS and agility in french Minister of National Education

- Original mission ⇒ EOLE GNU/Linux meta-distribution
- CeCILL / GPL software licensing
- Agile consulting for other development teams

Why did we get to OpenNebula?

bare metal elastic limit is too low

- Testing our OS was done on physical desktop computers
- Some “lucky” developers could have at most 2 VMs on their workstation

EOLE development needed elasticity

Looking for virtualisation infrastructure

many choices: too big, not enough flexible or immature

2012: two new quite powerful workstations ⇒ testing party

- Proxmox needed a reboot to add a new network
- Archipel barely emerged
- Ganeti was promising
- OpenStack was already too much

Started with OpenNebula 3.8

First uses

local workflow on central servers

- Each user was responsible for its own infrastructure
- Team documentation with conventions

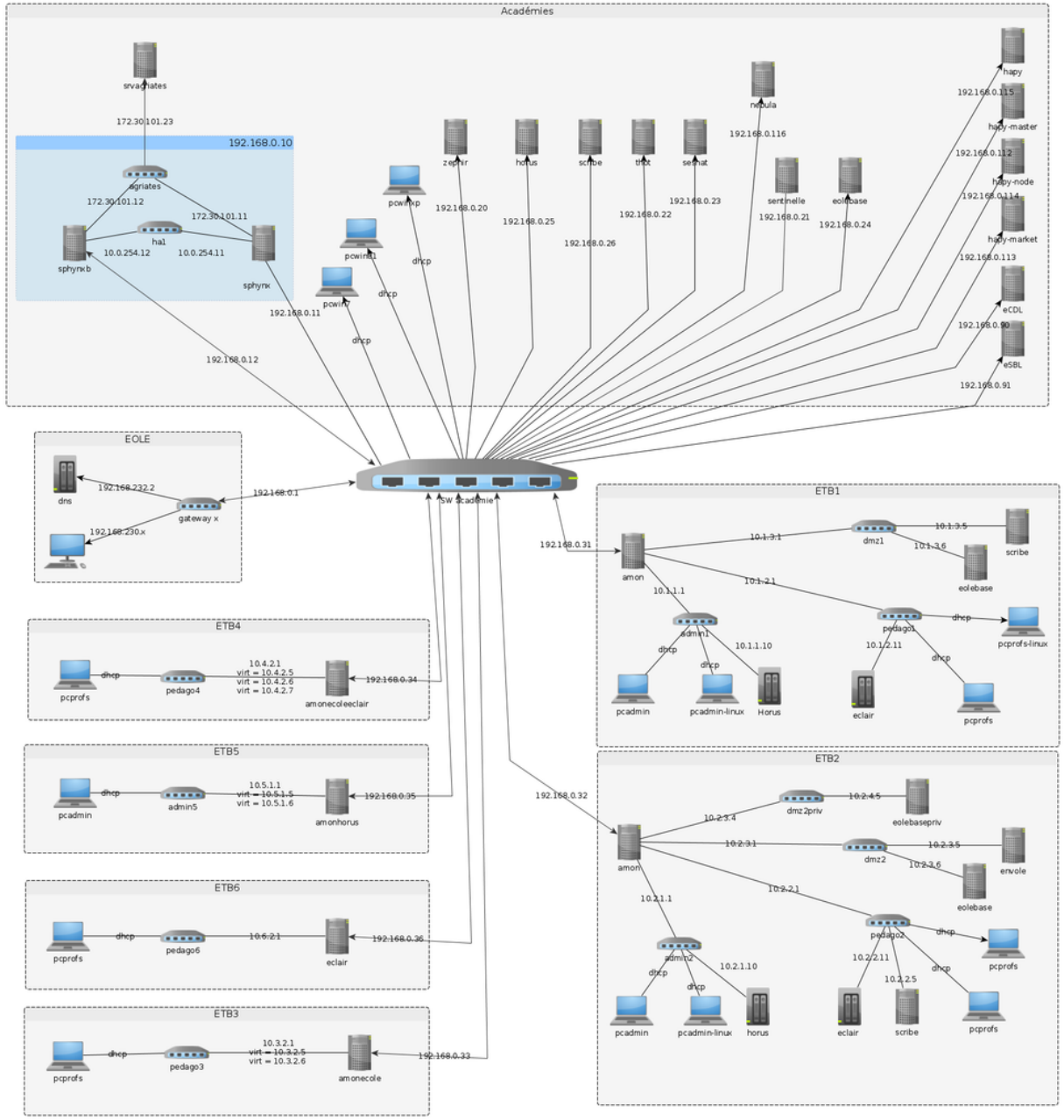
Everybody was admin

Load on higher skilled users

Emerged use cases and needs

- Short lifetime VMs
- Factorise VM templates and images
- Reduce usage cost for users
- Reproducible environments for QA
- Use production like environments for dev

Complete virtual infrastructure per user



Complete virtual infrastructure per user

workaround #2125

- VLAN isolated networks
- Standard network names
- One set of networks per user
- One user = one gateway

Sharing VM templates requires avoiding *UNAME* on networks

Automatic user environment generation

manual creation of 21 times 25 networks is not an option

- custom contextualisation of the gateway (per user IP)
- generate/update virtual networks

Pilot OpenNebula from Jenkins

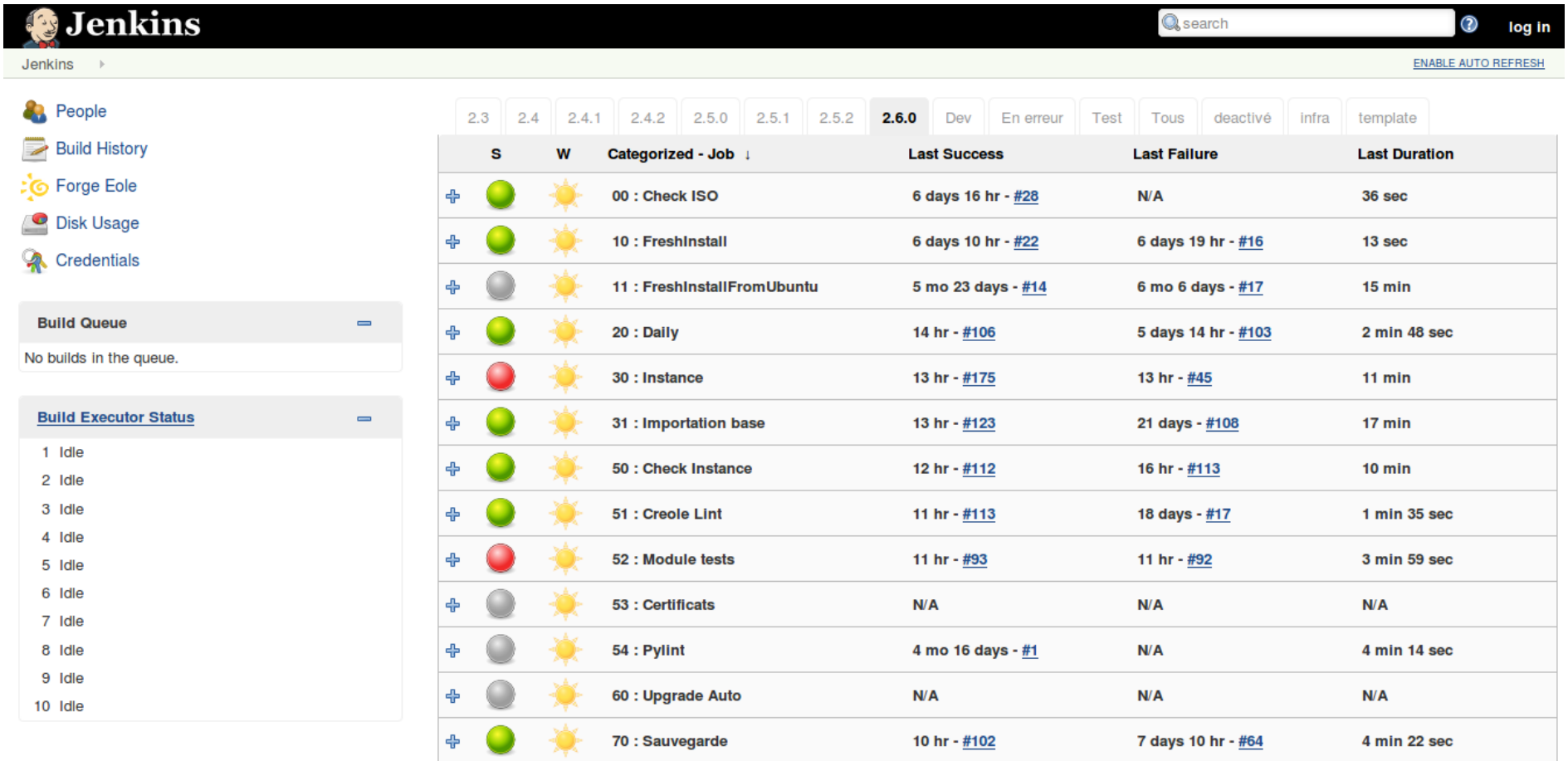
continuous integration of OS

- Check installation from ISO
- Check daily upgrade to find broken packages
- Check default configurations
- Check user data import

Produce VM images and templates at each step

Pilot OpenNebula from Jenkins

continuous integration of OS



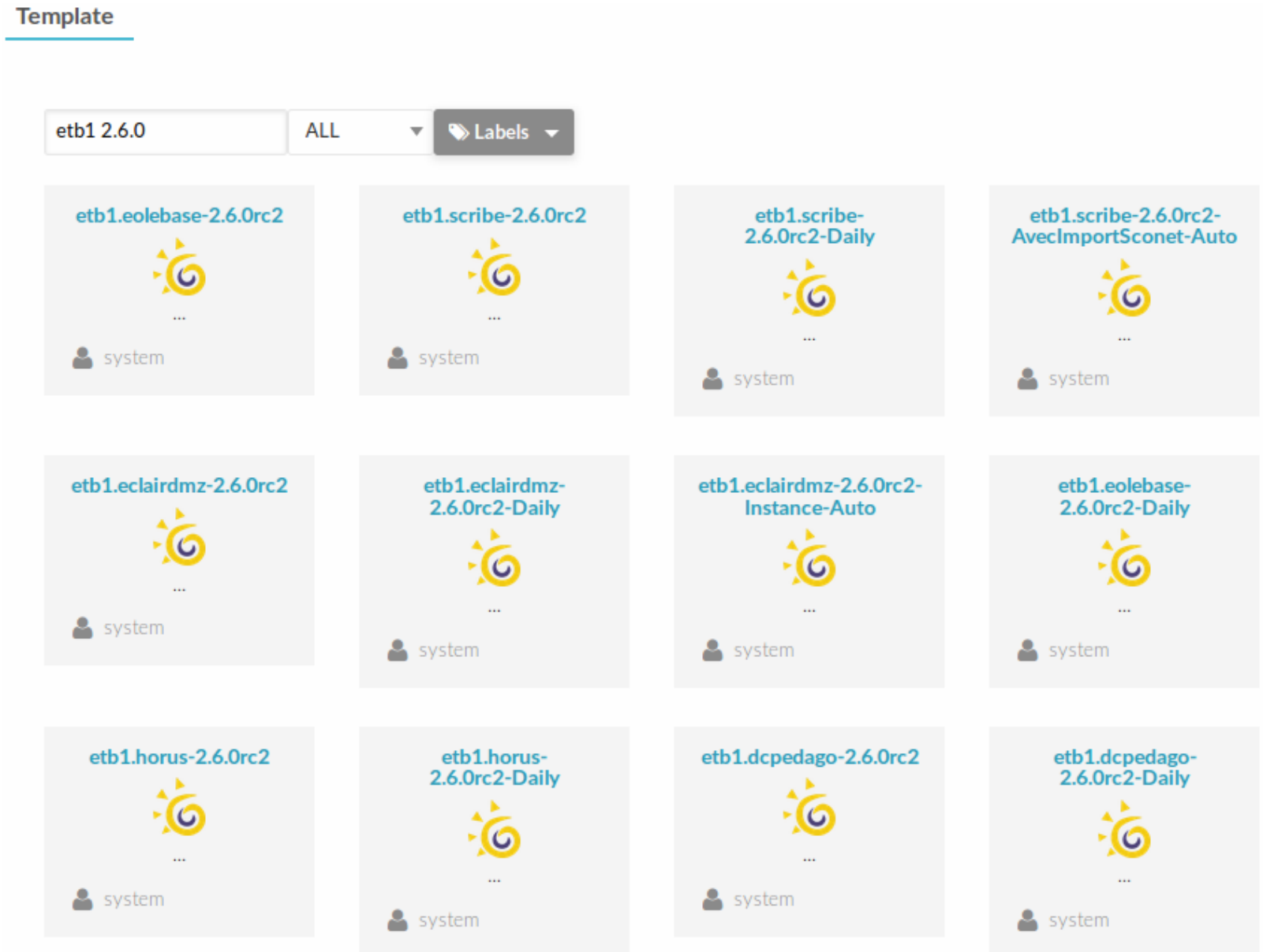
Pilot OpenNebula from Jenkins

some numbers

- $\approx$ 120 VMs per night
- $\approx$ 90k VMs since 2014
- $\approx$ 3TB of qcow2 images (25TB virtual)

Jenkins jobs produce ready to use VMs

reduce environment setup time



Development is so simple

until an IA will make our work

Preparing the coding session

1. Start an infrastructure
2. Create a branch of your repository
3. Clone the repository on the VM

Development is so simple

until an IA will make our work

Hack until it works

1. Code locally on your workstation
2. Pull the changes on the VM
3. `make install`
4. Test and cycle to 1 until it works

Development is so simple

until an IA will make our work

Test like a user

1. Cleanup your local branch
2. Merge, push and make a package
3. Cleanup the VM
   - `DELETE-RECREATE` (< 5.0)
   - Revert to initial disk snapshot (≥ 5.0)
4. Upgrade OS $\Rightarrow$ new package
5. Test

Jenkins jobs will install the new packages during next night

QA campaign

Automation is not the panacea

- Squash-TM
- Targeted environments
- Critical features

Physical limitations

test bed was burning

$\approx$ 40k VMs

- The two dedicated workstations was fine for testing
- Workload was memory bounded $\Rightarrow$ bumped to 2x32GB

NFS access on workgroup NAS was too slow

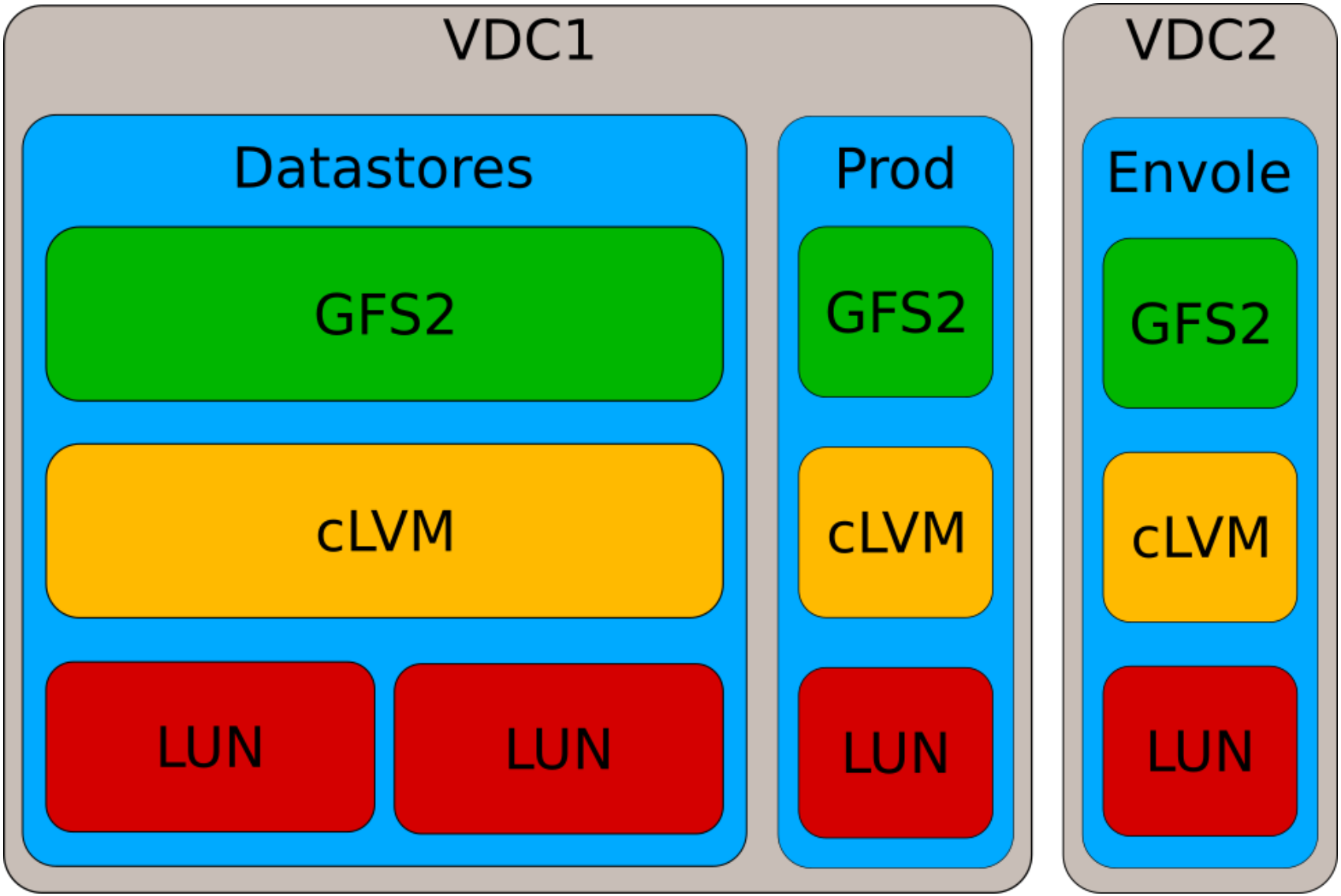
New hypervisor nodes

blade runners

- First VDC
  - Two servers
  - 48 cores
  - 384GB RAM
- Second VDC
  - One server
  - 8 cores
  - 42GB RAM

Storage on SAN

the corosync/pacemaker/cLVM/GFS2 sandwich



Storage on SAN

Everybody has a gun and too many want to use it

corosync/pacemaker can be challenging

Hot/cold storage $\Rightarrow$ I/Os not stripped on all LUNs

The future

- Replace home made orchestration code with SaltStack
- Host other teams
- Ceph storage
- Docker uses

The killer features of OpenNebula

you can't make us use something else without them

- Low load on the team
- We can adapt to our use cases and contribute
  - Features 3/17
  - Bugs 4/46

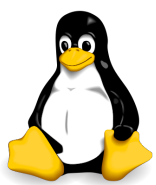
Muchas gracias OpenNebula Systems

Thanks

Many thanks to the FOSS community for all the great software. So few things would exists without them.

This talk was realised with the help of the following libre software:

- Composition system LaTeX TeX Live
- The most powerful text editor available today GNU/Emacs
- The Awesome window manager
- The Universal Operating System Debian GNU/Linux



Licence

The slides are licensed under Creative Commons BY-SA 4.0

- Attribution
- Share alike

You can obtain a copy of the license

by Internet

http://creativecommons.org/licenses/by-nc-sa/4.0

by snail mail

Creative Commons
444 Castro Street, Suite 900 Mountain View,
California, 94041, USA.