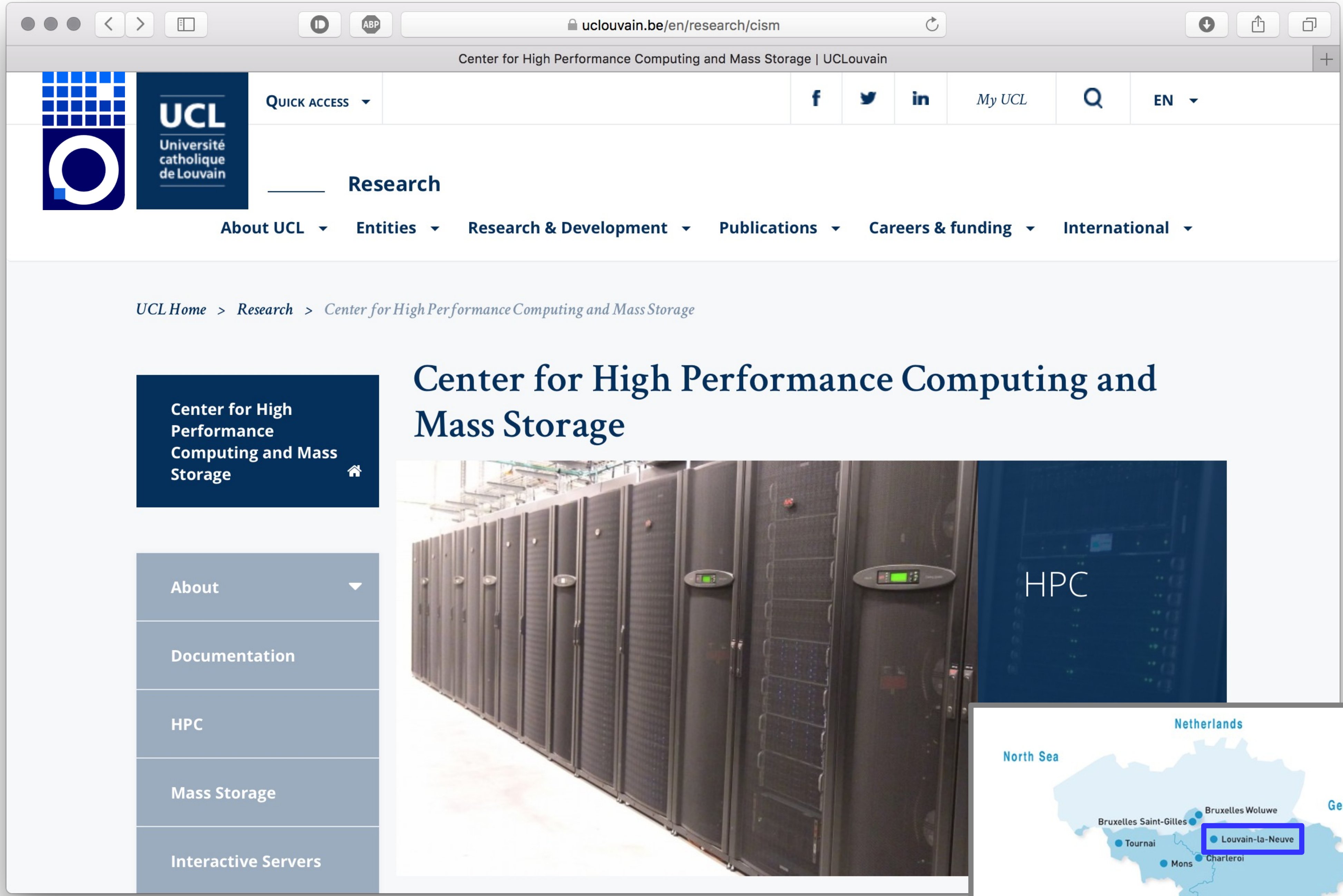
Behind the scenes of a FOSS-powered HPC cluster at UCLouvain

Ansible or Salt? Ansible AND Salt!

Damien François | Université catholique de Louvain - CISM

FOSDEM '18 | HPC, Big Data & Data Science Devroom | 2018-02-04



uclouvain.be/en/research/cism

Center for High Performance Computing and Mass Storage | UCLouvain

UCL Université catholique de Louvain

Quick access

My UCL

EN

Research

About UCL · Entities · Research & Development · Publications · Careers & funding · International

UCL Home > Research > Center for High Performance Computing and Mass Storage

Center for High Performance Computing and Mass Storage

- About
- Documentation
- HPC
- Mass Storage
- Interactive Servers

Center for High Performance Computing and Mass Storage

HPC

Netherlands
North Sea
Germany
Bruxelles Woluwe
Bruxelles Saint-Gilles
Louvain-la-Neuve
Tournai
Charleroi
Mons
France
Luxembourg

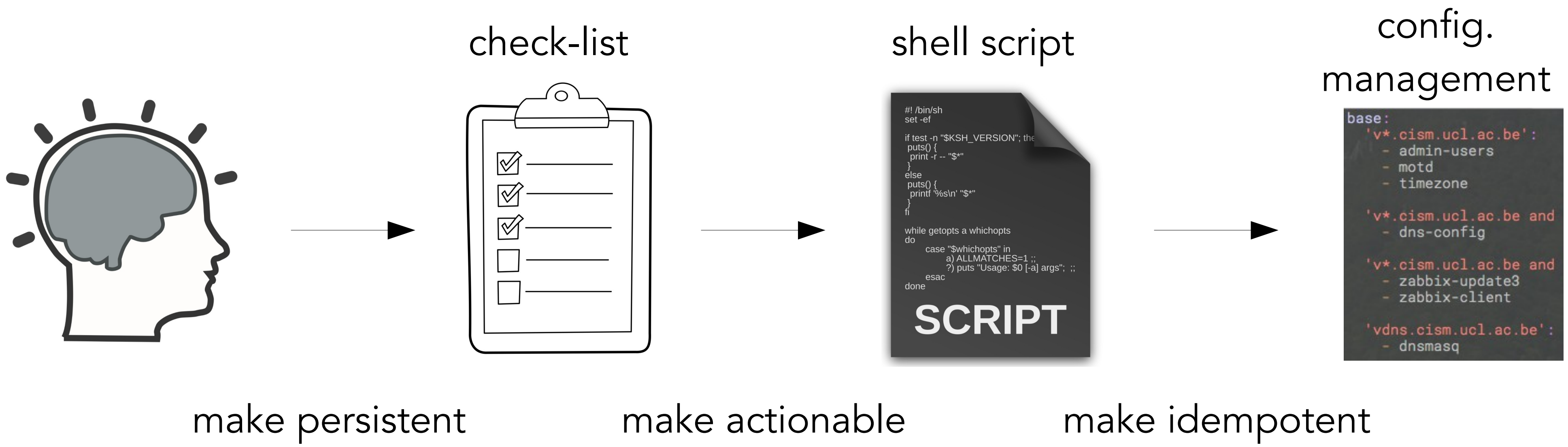
Manneback cluster

grows organically ; 1 to 10 machines at a time

now 4000 cores, Gb+10Gb, 50TB storage

100 local users + CMS grid, ~2 M jobs per year

We started “manually”…



```
#! /bin/sh
set -ef

if test -n "$KSH_VERSION"; the
 puts() {
  print -r -- "$*"
 }
else
 puts() {
  printf '%s\n' "$*"
 }
fi

while getopts a whichopts
do
        case "$whichopts" in
                a) ALLMATCHES=1 ;;
                ?) puts "Usage: $0 [-a] args";  ;;
        esac
done
```

```
base:
  'v*.cism.ucl.ac.be':
    - admin-users
    - motd
    - timezone

  'v*.cism.ucl.ac.be and
    - dns-config

  'v*.cism.ucl.ac.be and
    - zabbix-update3
    - zabbix-client

  'vdns.cism.ucl.ac.be':
    - dnsmasq
```

… and gradually improved automation

We settled on **three** tools

cobbler ANSIBLE SALTSTACK

for the **deployment** of new nodes

Unboxing

- Label, rack, connect
- Choose Name, IP
- Gather MAC

1. Deploy

Deploy operating system
Setup SSH key for Ansible
Configure and start Salt minion

2. Integrate

Get inventory from Salt or Cobbler
Setup RSA keys for Salt
Register node to services
Prepare configuration files
Install software

3. Configure

Install/update software
Broadcast configuration

Ready for jobs

cobbler

"Cobbler is a Linux installation server that allows for rapid setup of network installation environments." http://cobbler.github.io

Wrapper for PXE, TFTP, DHCP servers
Manage OS images, machine profiles

Install operating system
Setup hardware-specific configuration
(disk partitions, NICs, IPMI, etc.)
Setup minimal configuration
(Admin SSH keys, Salt minion)

ANSIBLE

"Ansible seamlessly unites workflow orchestration with configuration management, provisioning, and application deployment in one easy-to-use and deploy platform."
https://www.ansible.com

Shell scripts on steroïds
with builtin safety, idempotence, APIs

One-off operations
register to *Zabbix*, *GLPI*, *Salt*
build files: slurm.conf for *Slurm*, /etc/hosts for dnsmasq, /etc/ssh/ssh_known_hosts for hostbased SSH, .dsh/group/all for *pdsh*
create CPU-specific directory for *Easybuild*

SALTSTACK

“Scalable, flexible, intelligent IT orchestration and automation” https://saltstack.com

Central configuration management server

Daily management

configure system: LDAP, NTP, DNS, Slurm, etc.

install admin software

mount user filesystem (home, scratch, software)

Unboxing

- Label, rack, connect
- Choose Name, IP
- Gather MAC

1. Deploy

cobbler

Deploy operating system
Setup SSH key for Ansible
Configure and start Salt minion

2. Integrate

ANSIBLE

Get inventory from Salt or Cobbler
Setup RSA keys for Salt
Register node to services
Prepare configuration files
Install software

3. Configure

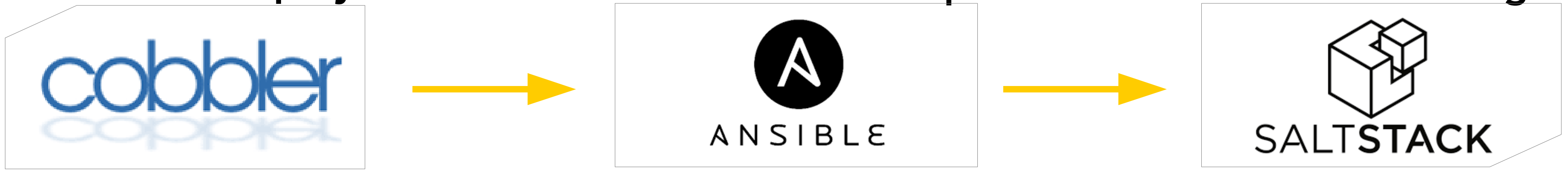
SALTSTACK

Install/update software
Broadcast configuration

if new CPU architecture -> Easybuild
if new Slurm QOS for specific users -> Slufl
Ready for jobs

More generally:

1. Deploy **2. Setup** **3. Manage**



Deploy operating system

Install software
Pre-seed data

Install/update software
Manage configuration

More generally:

1. Deploy

openstack™ CLOUD SOFTWARE

Deploy operating system

→

2. Setup

ANSIBLE

Install software
Pre-seed data

→

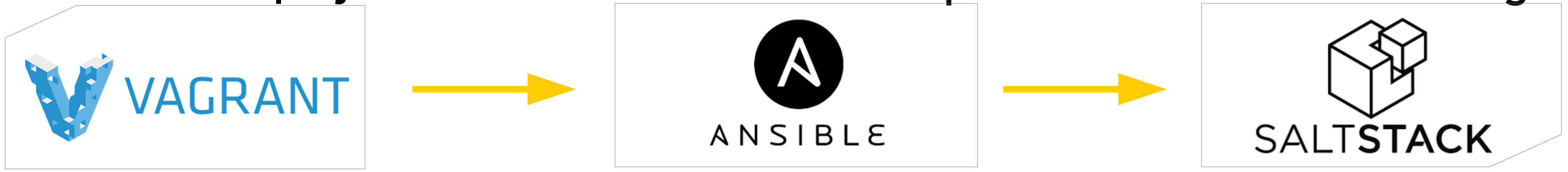
3. Manage

SALTSTACK

Install/update software
Manage configuration

More generally:

1. Deploy **2. Setup** **3. Manage**



Deploy operating system

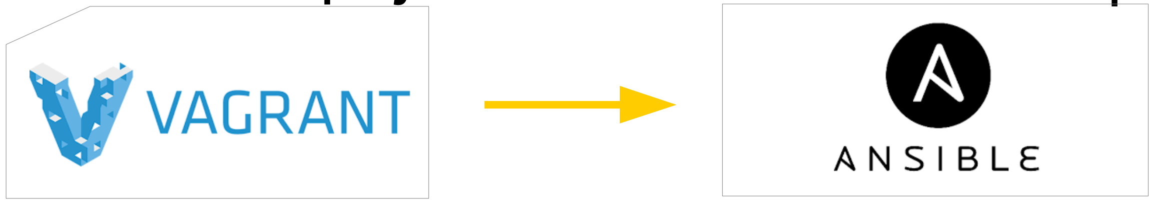
Install software
Pre-seed data

Install/update software
Manage configuration

Typical development platform: our laptops

1. Deploy

2. Setup



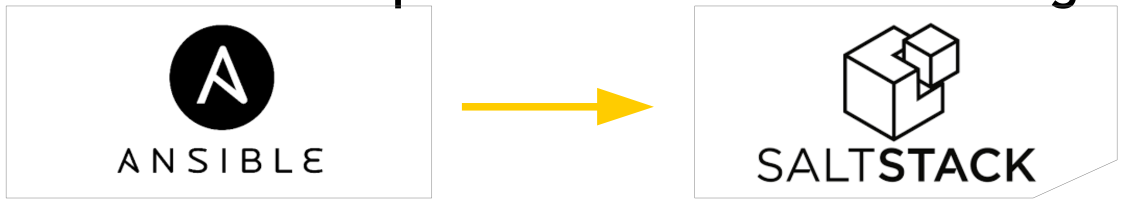
Deploy operating system

Install software
Pre-seed data

Typical staging platform: our test mini-cluster

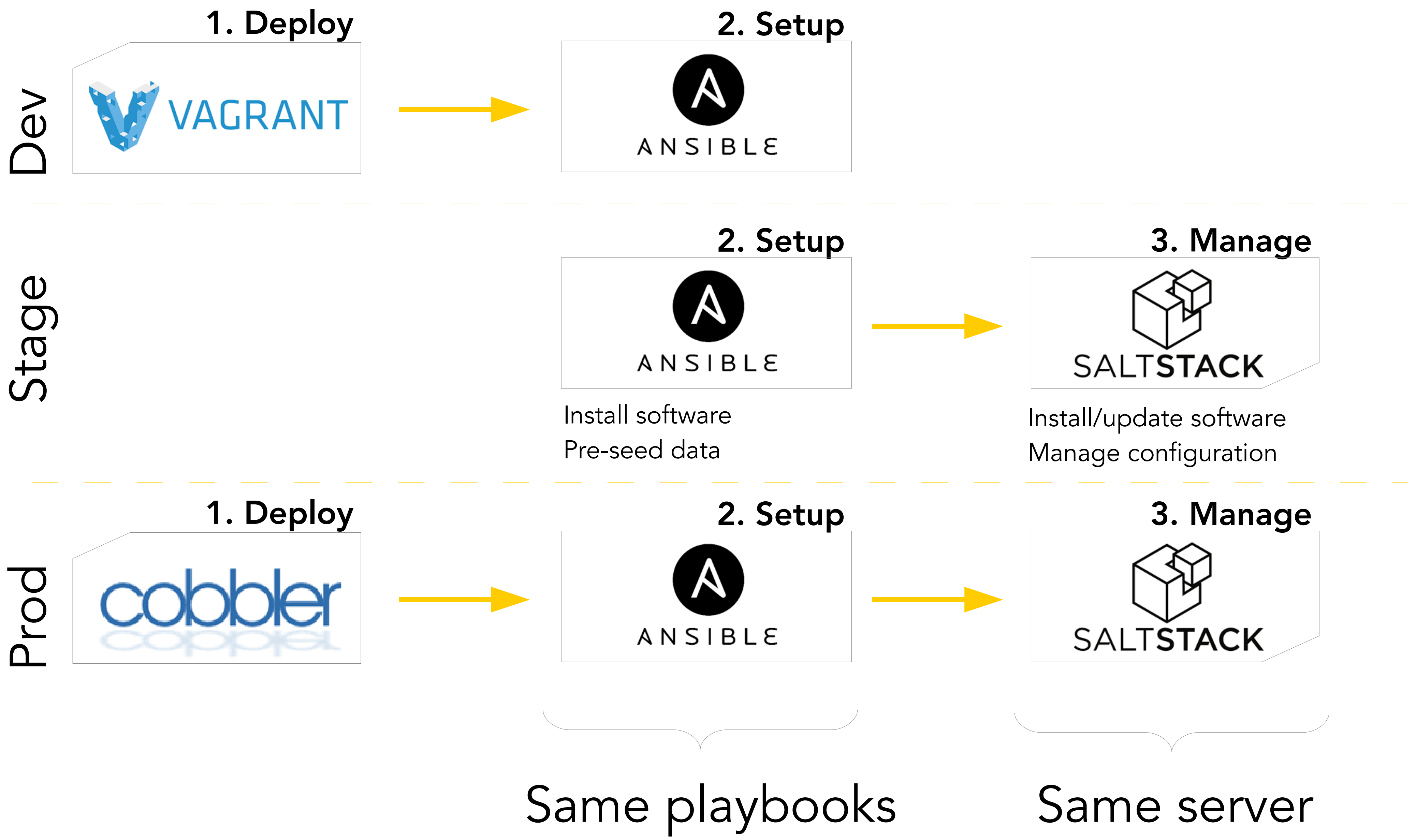
2. Setup

3. Manage



Install software
Pre-seed data

Install/update software
Manage configuration


Dev
1. Deploy
VAGRANT
2. Setup
ANSIBLE
Stage
2. Setup
ANSIBLE
Install software
Pre-seed data
3. Manage
SALTSTACK
Install/update software
Manage configuration
Prod
1. Deploy
cobbler
2. Setup
ANSIBLE
3. Manage
SALTSTACK
Same playbooks
Same server

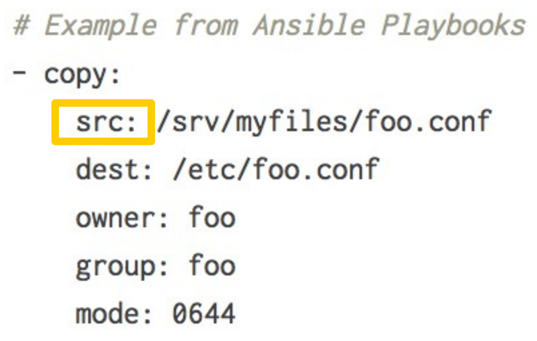
Some features overlap

(e.g. install soft)

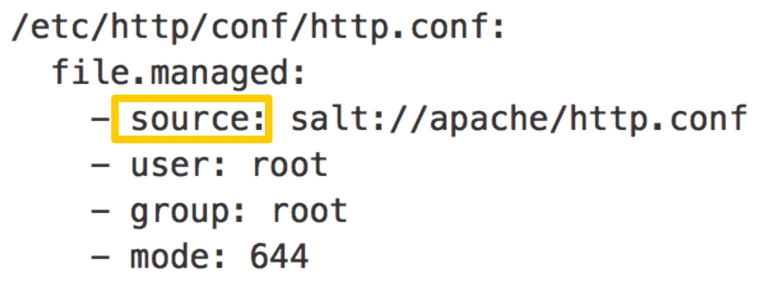
```
if soft.is_specific(“dev”): #e.g. VB guest additions
   vagrant.provision().install(soft)

elif soft.is_specific(“hardware”): #e.g. drivers
   cobbler.kickstart().install(soft)

elif soft.is_useful() in [“stage”, “prod”]:
                          #e.g. (e.g. zabbix-agent)
   salt.install(soft)

else: # needed through all the chain (e.g. slurm)
   ansible.install(soft)
```

Gotcha's

Uploading a file in Ansible and in Salt:



```
# Example from Ansible Playbooks
- copy:
    src: /srv/myfiles/foo.conf
    dest: /etc/foo.conf
    owner: foo
    group: foo
    mode: 0644
```



```
/etc/http/conf/http.conf:
  file.managed:
    - source: salt://apache/http.conf
    - user: root
    - group: root
    - mode: 644
```

Gotcha's

Uploading a file in Ansible and in Salt:

```
# Example from Ansible Playbooks
- copy:
    src: /srv/myfiles/foo.conf
    dest: /etc/foo.conf
    owner: foo
    group: foo
    mode: 0644
```

```
/etc/http/conf/http.conf:
  file.managed:
    - source: salt://apache/http.conf
    - user: root
    - group: root
    - mode: 644
```

Installing a package in Ansible and in Salt:

```
- name: install the latest version of ntpdate
  package:
    name: ntpdate
    state: latest
```

```
php.packages:
  pkg.installed:
    - fromrepo: wheezy-php55
    - pkgs:
      - php5-fpm
      - php5-cli
      - php5-curl
```

What we love about…

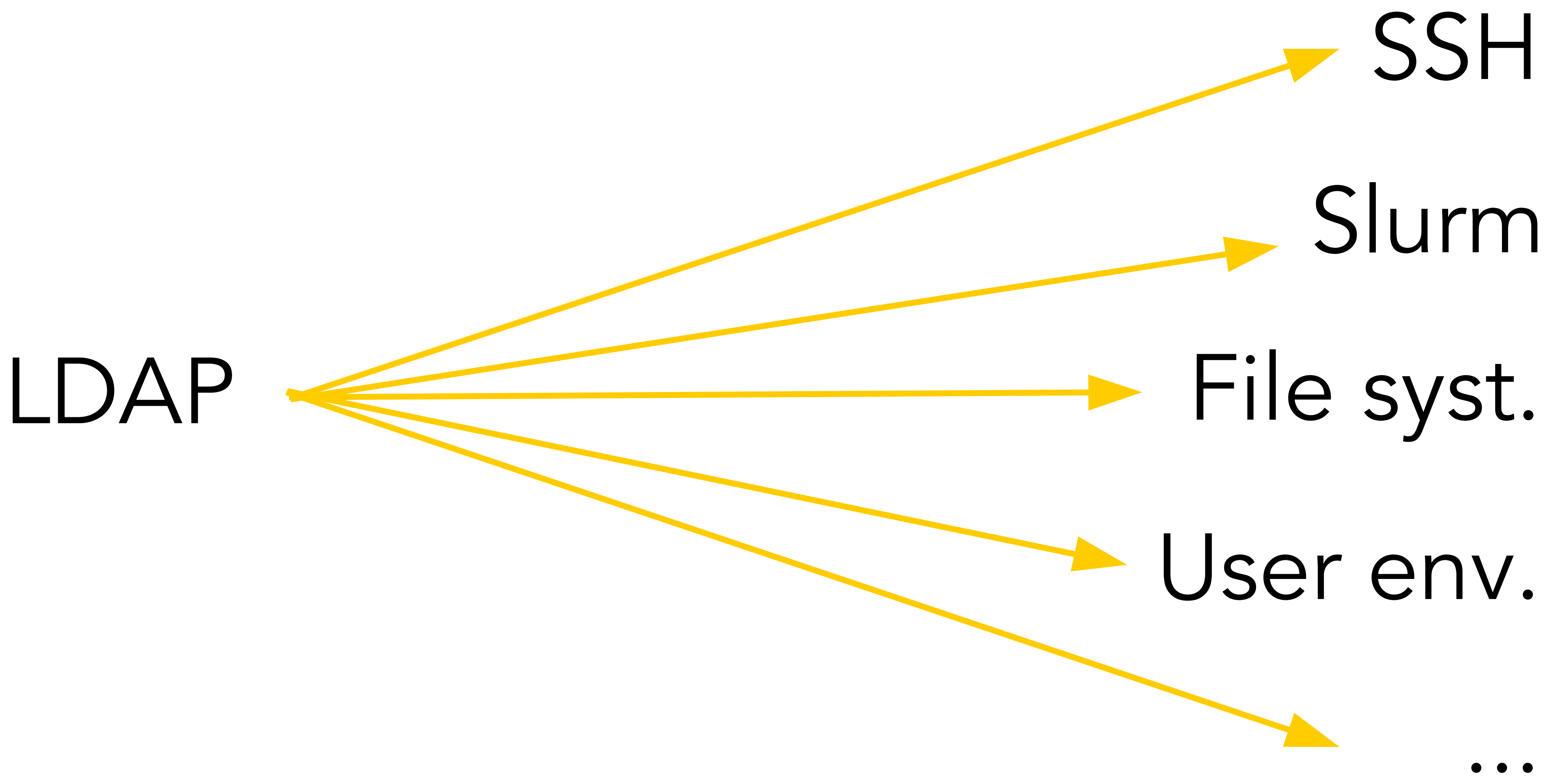
SALTSTACK

- Python, YAML, Jinja, the plethora of modules
- Declarative style; **very powerful**, handle complex dependencies,
- Pull: handle nodes down when they come back up, etc.
- **Single source of truth**, traceability, provenance, accountability
- **Scalability**, syndication; manages the whole infrastructure
- Out-of-band management (**second entry point**)

ANSIBLE

- Python, YAML, Jinja, the plethora of modules
- Imperative style; **simple to grasp**, playbook easy to read, **easy to share**, easy to reuse in different contexts
- Effective for manual/emergency **firefighting**
- In-band management, standalone (no need for agent, **uses SSH**)

Preparing for a new user



Slufl

LDAP

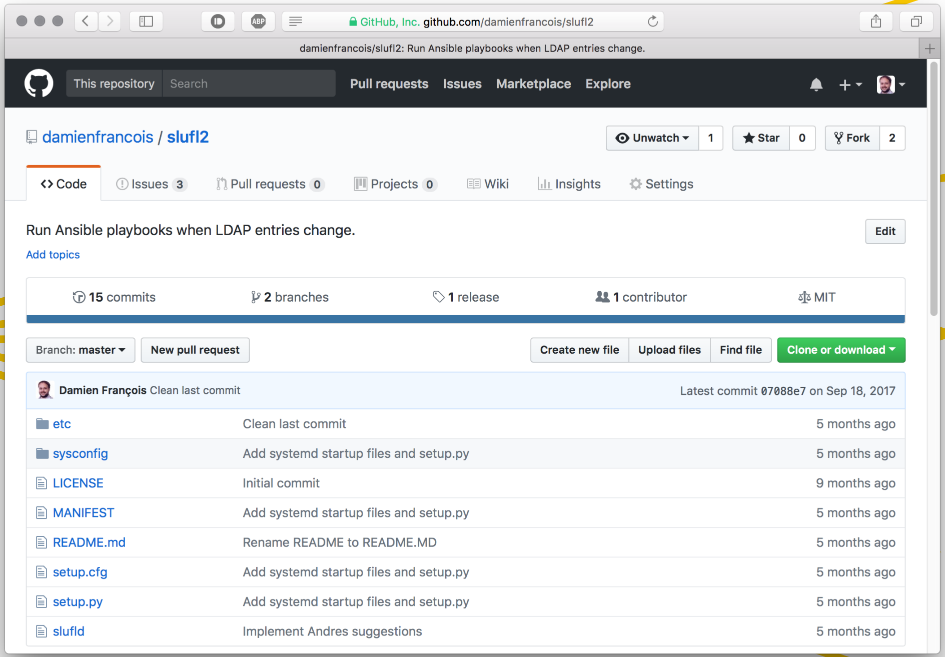
SSH

Slurm

File syst.

User env.

…



GitHub, Inc. github.com/damienfrancois/slufl2

damienfrancois/slufl2: Run Ansible playbooks when LDAP entries change.

This repository | Search | Pull requests | Issues | Marketplace | Explore

damienfrancois / slufl2

Unwatch 1 | Star 0 | Fork 2

Code | Issues 3 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

Run Ansible playbooks when LDAP entries change.

Add topics

15 commits | 2 branches | 1 release | 1 contributor | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

Damien François Clean last commit — Latest commit 07088e7 on Sep 18, 2017

etc	Clean last commit	5 months ago
sysconfig	Add systemd startup files and setup.py	5 months ago
LICENSE	Initial commit	9 months ago
MANIFEST	Add systemd startup files and setup.py	5 months ago
README.md	Rename README to README.MD	5 months ago
setup.cfg	Add systemd startup files and setup.py	5 months ago
setup.py	Add systemd startup files and setup.py	5 months ago
slufld	Implement Andres suggestions	5 months ago

Daemon that runs Ansible playbooks when LDAP entries change

Custom Salt grain for Slurm

```python
#! /usr/bin/env python
import ConfigParser, os
import socket, re

def custom_grains():
        grains = {}
        if not os.path.isfile('/etc/slurm/slurm.conf'):
             grains['slurm'] = {}
             grains['slurm']['status'] = 'Unknown'
             grains['slurm']['role'] = 'Unknown'
             return grains

        grains['slurm'] = {}
        grains['slurm']['status'] = 'Installed'
        grains['slurm']['partition'] = 'None'

        class FakeSecHead(object):
            def __init__(self, fp):
                self.fp = fp
                self.sechead = '[base]\n'

            def readline(self):
                if self.sechead:
                    try:
                        return self.sechead
                    finally:
                        self.sechead = None
                else:
                    return self.fp.readline()

        config = ConfigParser.ConfigParser()
        config.readfp(FakeSecHead(open('/etc/slurm/slurm.conf')))
        if config.get('base', "controlmachine") == socket.gethostname().split('.')[0]:
                grains['slurm']['role'] = 'ControlMachine'
        else:
                grains['slurm']['role'] = 'ComputeNode'

        partitions = open('/etc/slurm/slurm.conf').read().split('PartitionName=')
        for p in partitions[1:]:
                name, rest = p.split("Nodes=")
                members = rest.split(' ')[0]
                fullnames = []
                expsets = re.findall('([a-z-]+\[?[0-9,-]+)\]?[^a-z]', members + ',')
                for e in expsets:
                    if not '[' in e:
                        fullnames.append(e)
                    else:
                        pre, suf = e.split('[')
                        subsets = suf.split(',')
                        for s in subsets:
                            if not '-' in s:
                                fullnames.append(pre + s)
                            else:
                                b, e = s.split('-')
                                padding = len(b)
                                formatstring = "%%0%dd" % padding
                                b, e = int(b), int(e)
                                for number in range(b,e+1):
                                    fullnames.append(pre + formatstring % number)
                if socket.gethostname().split('.')[0] in fullnames:
                        grains['slurm']['partition'] = name.strip()

        return grains
```

top.sls

```yaml
'slurm:partition:cp3':
  - match: grain
  - grid-deps

'slurm:partition:Zoe':
  - match: grain
  - storage-zoe-mount
```

Ansible and Salt work very well together

Complementary
Same building bricks

Along with Cobbler, nice team to manage an organically-growing Tier-2 compute cluster

SPHINX
PYTHON DOCUMENTATION GENERATOR

[[DW]]

sympa

elastic + logstash
kibana

GitLab

BeeGFS®
developed by Fraunhofer

easybuild

openstack

mercurial

OpenLDAP

slurm
workload manager

VirtualBox

cobbler

git

CentOS

ANSIBLE

docker

python™

SALTSTACK

ZABBIX

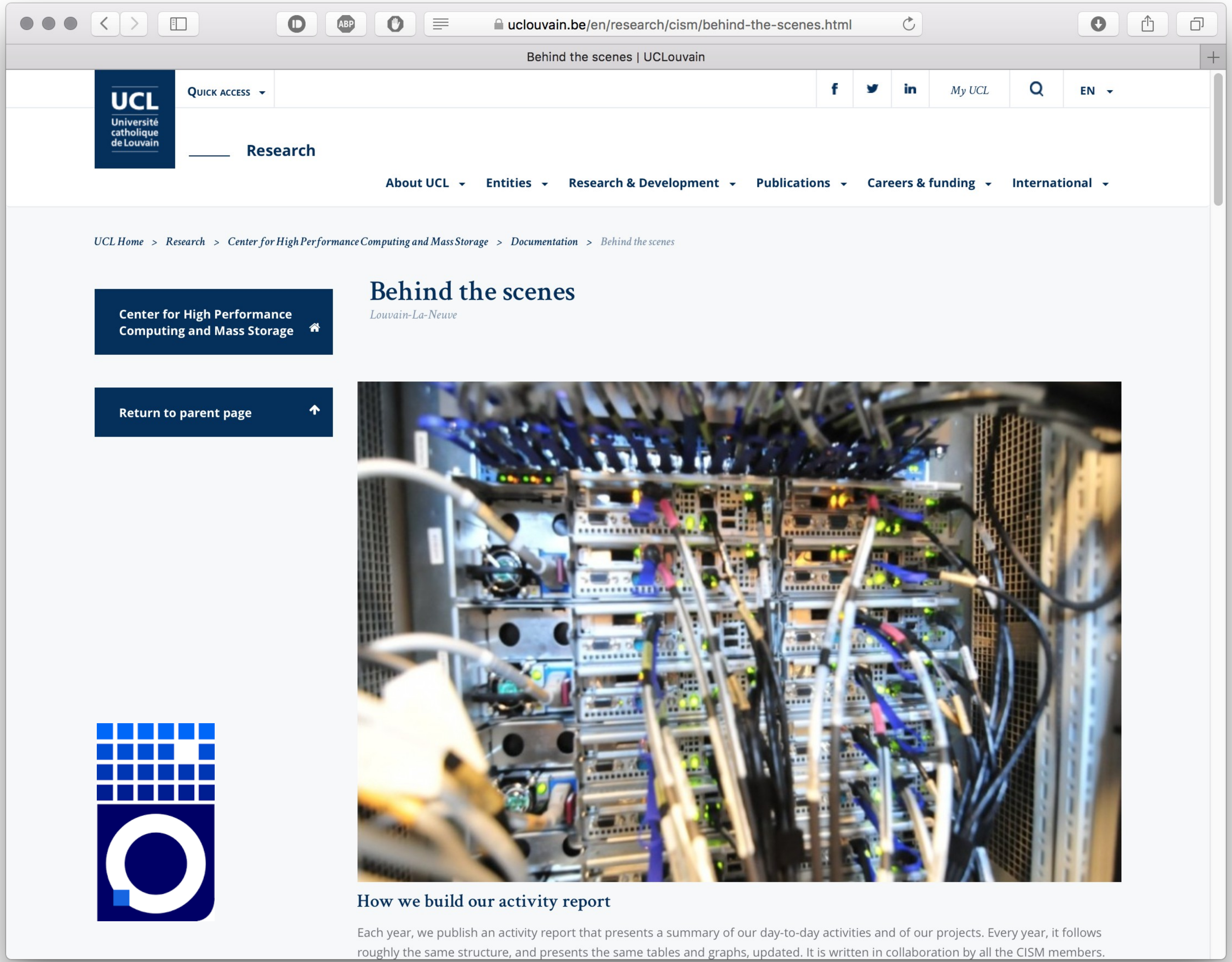
HashiCorp
Vagrant

GLPI

php

MariaDB

pdsh, clustershell, sshuttle, pandoc


uclouvain.be/en/research/cism/behind-the-scenes.html
Behind the scenes | UCLouvain
UCL Université catholique de Louvain
Quick access
My UCL
EN
Research
About UCL
Entities
Research & Development
Publications
Careers & funding
International

UCL Home > Research > Center for High Performance Computing and Mass Storage > Documentation > Behind the scenes

Behind the scenes

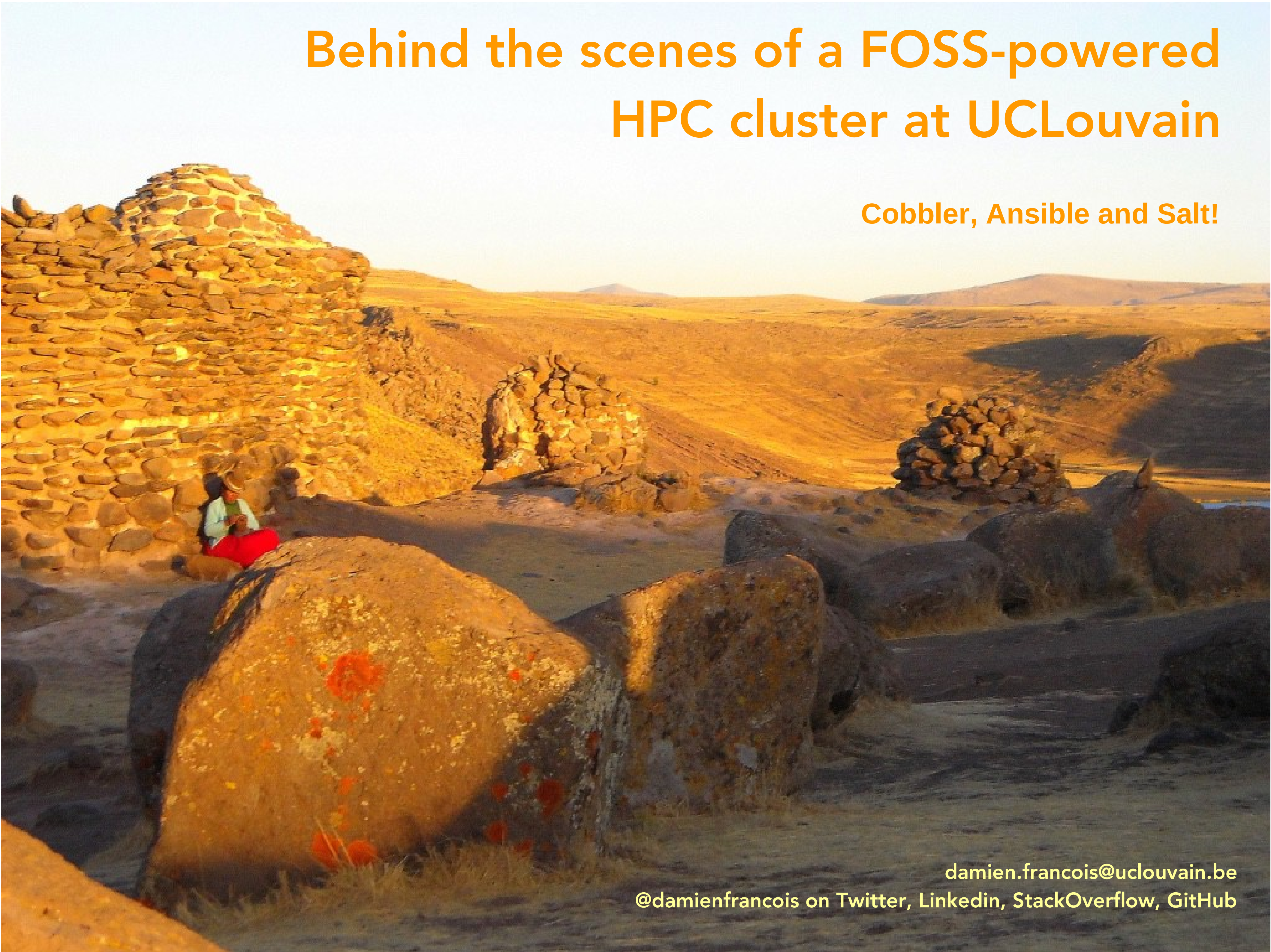
Louvain-La-Neuve

Center for High Performance Computing and Mass Storage

Return to parent page

How we build our activity report

Each year, we publish an activity report that presents a summary of our day-to-day activities and of our projects. Every year, it follows roughly the same structure, and presents the same tables and graphs, updated. It is written in collaboration by all the CISM members.



Behind the scenes of a FOSS-powered HPC cluster at UCLouvain

Cobbler, Ansible and Salt!

damien.francois@uclouvain.be
@damienfrancois on Twitter, Linkedin, StackOverflow, GitHub