Journées Mathrice
Montpellier mars 2018

a DevOps use case

JupyterCloud

Infrastructure As Code using Continuous Integration

David DELAVENNAT
PHYMATH (CMLS, CPHT, CMAP) - GDS CNRS Mathrice

Agenda

- JupyterDays@X
- JupyterCloud
- A DevOps use case
- Infrastructure As Code
- Continuous Integration
- What’s next?

JupyterDays@X

JupyterDays@X

Venez découvrir tout l'univers de Jupyter !

6 mars 2018 (9h15 - 12h45)

Ecole polytechnique (plan d'accès)

amphi Poincaré

Jupyter et son écosystème offrent un ensemble d'outils permettant d'avoir un environnement de développement et de travail convivial aussi bien pour la recherche que pour l'enseignement. Destiné à la base pour développer du code Python, Julia ou R, c'est aujourd'hui bien plus que ça. Les notebooks (page web où l'on peut écrire du code numérique, du texte, des équations mathématiques, insérer des vidéos, des graphiques interactifs, ...) sont de plus en plus utilisés et ont été introduits de manière systématique cette année dans le cadre du Département de Mathématiques Appliquées où ils ont trouvé un véritable écho auprès des élèves. Leurs usages sont variés: feuilles d'exercices pour les étudiants, cahiers de recherche, tutoriels, ...

JupyterCloud

mathrice

Home Accueil

JupyterCloud

Jupyter as a Service

S'AUTHENTIFIER

Un portail d'accès opéré par Mathrice

https://jupytercloud.math.cnrs.fr/

Accéder

Utiliser un ordinateur ou une tablette pour lancer des calculs depuis un navigateur WEB

S'authentifier

Utiliser son compte d'établissement pour s'authentifier sur la Fédération d'identité Éducation-Recherche opérée par le Groupement d'Intérêt Public RENATER

Paramétrer

Choisir une image docker métier enregistrée dans le registre docker de JupyterCloud

Calculer

Exécuter ses notebooks Jupyter au sein d'un centre de données et de calcul français

Collaborer

Partager ses notebooks Jupyter sur l'annuaire de notebook de JupyterCloud

Accèder aux données

Connecter des partages sur n'importe quelle infrastructure de stockage compatible avec les protocoles

WebDAV (Saaf la, Owncloud, Nextcloud...)
OpenStack Swift
Amazon S3

Le projet

Jupyter as a Service.

OpenDreamKit

Partenaires

GDS CALCUL
GDS MATHRICE
SERVICE FG-CLOUD
MESOCENTRE GRICAD
MESOCENTRE PHYMATH
MESOCENTRE ROMEO

Technologies

Shibboleth / OpenID Connect
JupyterHub / Jupyter
Docker
Kubernetes / Helm
OpenStack / LBaaS
GlusterFS / Heketi

Made by Materialize

Copyright © 2017 Marco Damiani. Powered by Jekyll

JupyterCloud

- Une fédération de sites de calcul
  - Autonomes : calcul + stockage
  - Authentifiés via OpenID Connect sur la Fédération d'identité RENATER voir sur EGI
  - Enregistrés sur le domaine jupytercloud.math.cnrs.fr
    - reims.jupytercloud.math.cnrs.fr
    - grenoble.jupytercloud.math.cnrs.fr
    - paris-saclay.jupytercloud.math.cnrs.fr
    - strasbourg.jupytercloud.math.cnrs.fr
    - montpellier.jupytercloud.math.cnrs.fr
    - …

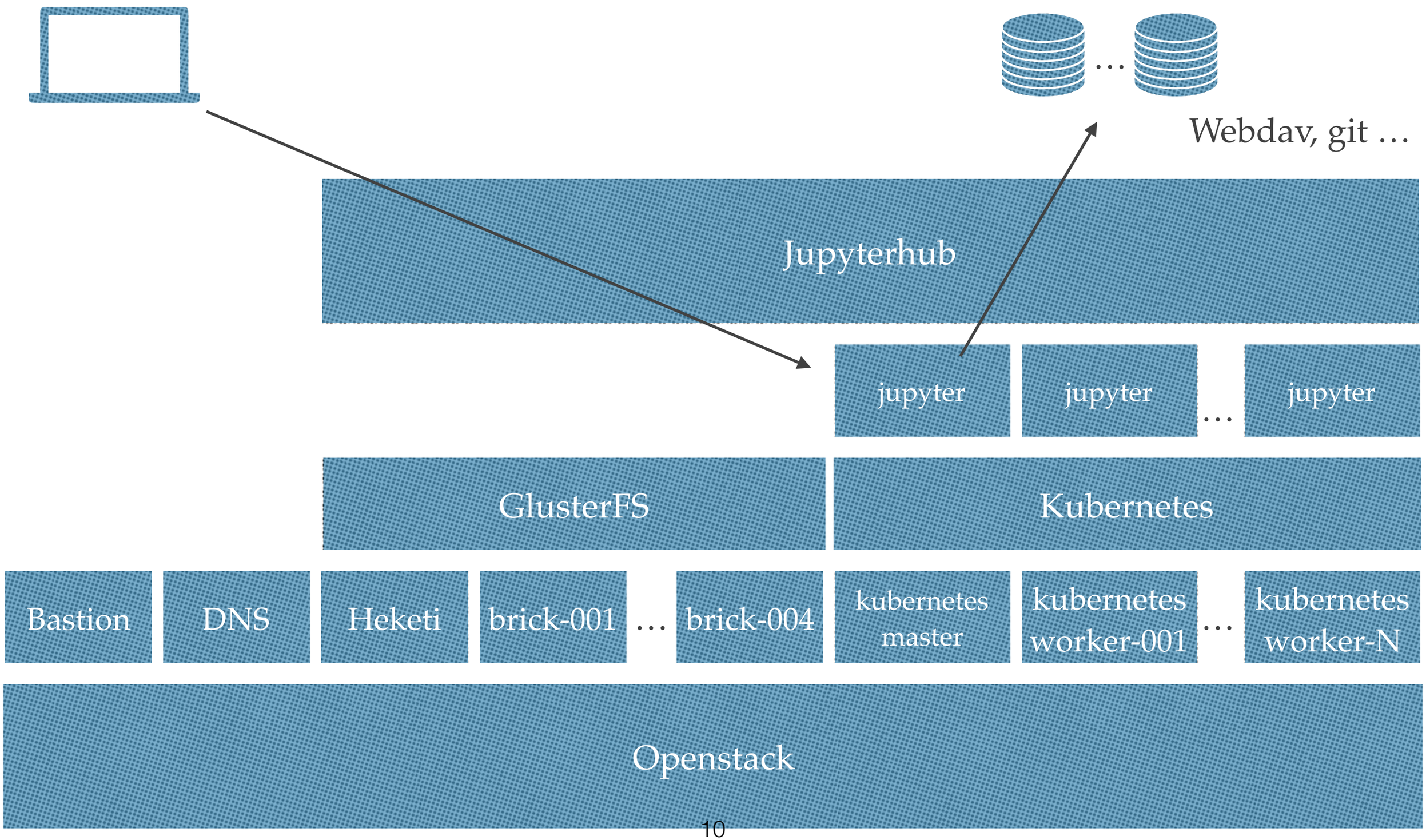
JupyterCloud

- Jupyterhub, Binderhub
- OpenID, Kubernetes, GlusterFS
- OpenStack (LBaaS)
- Terraform, Consul, Serverspec
- Gitlab, Artifactory

JupyterCloud

- https://zero-to-jupyterhub.readthedocs.io/en/latest/
- https://media.readthedocs.org/pdf/zero-to-jupyterhub/latest/zero-to-jupyterhub.pdf
- https://github.com/jupyterhub/zero-to-jupyterhub-k8s

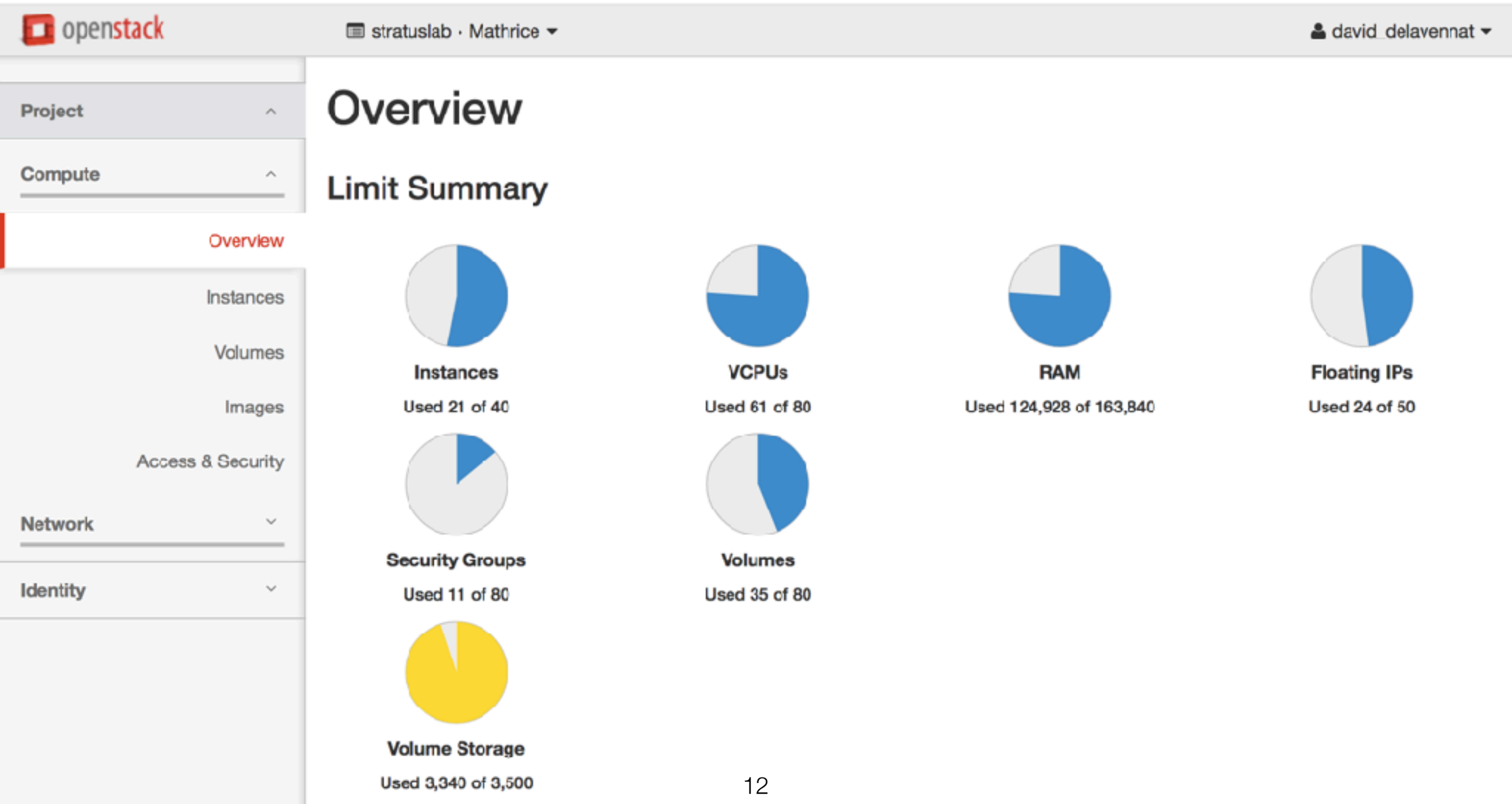
JupyterCloud



JupyterCloud

- Kubernetes
  - container orchestrator
    - CNI network drivers (weave, kube-router, flannel, calico…)
    - Cloud Provider (LBaaS)
      - OpenStack
    - Persistent Volumes (Persistent Volume Claims)
      - GlusterFS

JupyterCloud



JupyterCloud

glusterfs.services.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.1.2	m1.small	-	Active	nova
kubernetes.services.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.1.3	os.8	-	Active	nova
brick-002.glusterfs.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.3.2	m1.small	-	Active	nova
worker-001.kubernetes.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.4.1	os.8	-	Active	nova
worker-002.kubernetes.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.4.2	os.8	-	Active	nova
worker-003.kubernetes.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.4.3	os.8	-	Active	nova
worker-004.kubernetes.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.4.4	os.8	-	Active	nova
brick-003.glusterfs.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.3.3	m1.small	-	Active	nova
brick-004.glusterfs.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.3.4	m1.small	-	Active	nova
brick-001.glusterfs.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.3.1	m1.small	-	Active	nova
ns-002.services.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.2.1	m1.small	-	Active	nova
ns-001.services.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	172.16.2.2	m1.small	-	Active	nova
bastion.services.jupytercloud	CentOS-7-x86_64-GenericCloud-1708	**jupytercloud** 172.16.1.1 **public**	m1.small	-	Active	nova

JupyterCloud

- GlusterFS
- volume-manager (https://github.com/heketi/heketi)
- N nodes

JupyterCloud

- OpenID Connect
  - https://github.com/coreos/dex
  - https://github.com/doorkeeper-gem/doorkeeper-openid_connect
  - http://www.thegreatcodeadventure.com/building-a-custom-oauth-strategy-for-doorkeeper/

JupyterCloud

jupyter Untitled Last Checkpoint: 03/08/2018 (autosaved)

Logout Control Panel

File Edit View Insert Cell Kernel Help

Trusted Python 3

Run Code

```
In [1]: dir()

Out[1]: ['In',
         'Out',
         '_',
         '__',
         '___',
         '__builtin__',
         '__builtins__',
         '__doc__',
         '__loader__',
         '__name__',
         '__package__',
         '__spec__',
         '_dh',
         '_i',
         '_i1',
         '_ih',
         '_ii',
         '_iii',
         '_oh',
         'exit',
         'get_ipython',
         'quit']

In [ ]:
```

JupyterCloud

binder
(beta)

Turn a GitHub repo into a collection of interactive notebooks

Have a repository full of Jupyter notebooks? With Binder, open those notebooks in an executable environment, making your code immediately reproducible by anyone, anywhere.

Build and launch a repository

GitHub repo or URL

GitHub repository name or link

Git branch, tag, or commit

master

Path to a notebook file (optional)

Path to a notebook file (optional)

File

launch

Copy the URL below and share your Binder with others:

Fill in the fields to see a URL for sharing your Binder.

Copy the text below, then paste into your README to show a binder badge: launch binder

A DevOps use case

A DevOps use case

- Dev (JupyterCloud) and Ops (OpenStack operators) frequent interactions
- Deployment repeatability
- Portability
- Tests

A DevOps use case

- Class
  - VM -> qcow, vmdk, ramdisk files…
  - Container -> docker images…
- Instance
  - VM -> a running VM from a copy of the class file
  - Container -> a running container from a copy of the class file

Infrastructure As Code

HashiCorp

Products Open Source Resources

Vagrant

Create and configure portable development environments

Packer

Create platform specific machine images from a single source

Terraform

Create, combine and manage infrastructure across multiple providers

Vault

Centrally store, secure and control access to distributed secrets

Consul

Distributed highly available tool for service discovery, configuration and orchestration

Nomad

Cluster manager and scheduler to deploy applications across any infrastructure

Enter your Email

Request a Demo

Infrastructure As Code

- HashiCorp
  - https://www.packer.io/
  - https://www.terraform.io/
  - https://www.consul.io/

Infrastructure As Code

Packer

```
{
  "_variables": "https://www.packer.io/docs/templates/user-variables.html",
  "_variables: openstack_keystone":     "set it in tools/hashicorp/packer/builders/openstack/variables.json",
  "_variables: openstack_cacert":       "set it in tools/hashicorp/packer/builders/openstack/variables.json",
  "_variables: openstack_domain":       "set it in tools/hashicorp/packer/builders/openstack/variables.json",
  "_variables: openstack_tenant":       "set it in tools/hashicorp/packer/builders/openstack/variables.json",
  "_variables: openstack_username":     "set it in tools/hashicorp/packer/builders/openstack/credentials.json",
  "_variables: openstack_password":     "set it in tools/hashicorp/packer/builders/openstack/credentials.json",
  "_variables: openstack_flavor":       "set it in tools/hashicorp/packer/builders/openstack/variables.json",
  "_variables: openstack_network":      "set it in tools/hashicorp/packer/builders/openstack/variables.json",
  "_variables: openstack_source_image": "set it in tools/hashicorp/packer/builders/openstack/variables.json",
  "_variables: openstack_image_name":   "set it in tools/hashicorp/packer/builders/openstack/variables.json",

  "_builder_openstack": "https://www.packer.io/docs/builders/openstack.html",
  "builders": [
    {
      "type": "openstack",
      "identity_endpoint": "{{user `openstack_keystone`}}",
      "cacert":            "{{user `openstack_cacert`}}",
      "domain_name":       "{{user `openstack_domain`}}",
      "tenant_name":       "{{user `openstack_tenant`}}",
      "username":          "{{user `openstack_username`}}",
      "password":          "{{user `openstack_password`}}",
      "flavor":            "{{user `openstack_flavor`}}",
      "networks":        [ "{{user `openstack_network`}}" ],
      "source_image_name": "{{user `openstack_source_image`}}",
      "image_name":        "{{user `openstack_image_name`}}",
      "ssh_username":      "centos"
    }
  ],

  "_provisioner_shell_local":  "https://www.packer.io/docs/provisioners/shell-local.html",
  "_provisioner_shell":        "https://www.packer.io/docs/provisioners/shell.html",
  "_provisioner_file":         "https://www.packer.io/docs/provisioners/file.html",
  "provisioners": [
```

Infrastructure As Code

Terraform

- infra_cloud_init.tf
- infra_hosts_0_servergroup_glusterfs.tf
- infra_hosts_0_servergroup_kubernetes.tf
- infra_hosts_0_servergroup_services_dns.tf
- infra_hosts_1_services_bastion.tf
- infra_hosts_2_services_dns.tf
- infra_hosts_3_glusterfs_bricks.tf
- infra_hosts_3_services_glusterfs.tf
- infra_hosts_4_kubernetes_workers.tf
- infra_hosts_4_services_kubernetes.tf
- infra_network_security_group_base.tf
- infra_network_security_group_dns.tf
- infra_network_security_group_glusterfs.tf
- infra_network_security_group_http.tf
- infra_network_security_group_https.tf
- infra_network_security_group_kubernetes_masters.tf
- infra_network_security_group_kubernetes_workers.tf
- infra_network_security_group_tiller.tf
- infra_network.tf
- infra_output.tf

```
resource "openstack_networking_port_v2" "services-dns" {
  network_id = "${openstack_networking_network_v2.jupytercloud.id}"
  count      = "${var.services_dns_count}"
  name       = "${var.services_dns_prefix}-${format("%03d",count.index+1)}.${var.services_domain_name}"
  fixed_ip {
    subnet_id  = "${openstack_networking_subnet_v2.jupytercloud.id}"
    ip_address = "${var.services_dns_subnet}.${count.index+1}"
  }
  admin_state_up = "true"
  security_group_ids = [
    "${openstack_networking_secgroup_v2.base.id}",
    "${openstack_networking_secgroup_v2.dns.id}"
  ]
}
```

```
block_device {
  boot_index  = 1
  source_type = "volume"
  destination_type      = "volume"
  uuid        = "${element(openstack_blockstorage_volume_v2.vdb.*.id, count.index)}"
  delete_on_termination = true
}
```

Infrastructure As Code
Terraform

Terraform backend

```
$ cat infra_terraform.tf
terraform {
  backend "artifactory" {
    username = "david.delavennat"
    password = « xxxxxxxxx"
    url      = "https://artifactory.math.cnrs.fr/artifactory"
    repo     = "jupytercloud"
    subpath  = "infrastructure"
  }
}
```

INSTALL JFROG CLI

```
curl -fL https://getcli.jfrog.io | sh
```

OR DOWNLOAD A SPECIFIC VERSION FROM JFROG BINTRAY:

Mac

64 & 32 BIT >

or brew install jfrog-cli-go

Windows

64 BIT >

Linux

32 BIT / 64 BIT >

READ THE DOCUMENTATION >

Infrastructure As Code
Artifactory

```
$ curl -fL https://getcli.jfrog.io | bash

$ ./jfrog rt config --url=https://artifactory.math.cnrs.fr/artifactory \
                    --user=david.delavennat \
                    --password=xxxxxxxxxx
Artifactory server ID: plm
[Info] Encrypting password…

$ ./jfrog rt config show
Server ID: plm
Url: https://artifactory.math.cnrs.fr/artifactory
User: david.delavennat
Password: ***
Default:  true

$ ./jfrog rt upload --server-id=plm kubernetes-node.qcow2 jupytercloud/infrastructure/kubernetes-node.qcow2
[Info] [Thread 2] Uploading artifact: kubernetes-node.qcow2
{
  "status": "success",
  "totals": {
    "success": 1,
    "failure": 0
  }
}
```

Infrastructure As Code

Artifactory

mathrice

Welcome, admin

Help

Build Browser

All Builds › JupyterCloud › 31

Build #31

General Build Info | Published Modules | Environment | Issues | Licenses | Diff | Release History

‹ Back To All Modules

Module Details: jupytercloud.images

Compare With Previous Build

Artifacts (1 matches found)

Filter by Artifact Name

‹ Page 1 of 1 ›

Artifact Name	Type	Repo Path
kubernetes-node_20180322-132618.qcow2	qcow2	jupytercloud/images/kubernetes-node_20180322-132618.qcow2

Dependencies (0 matches found)

Filter by Dependency ID

Continuous Integration

Continuous Integration

- Jenkins / Hudson
  - Artifactory plugin
  - OpenShift Jenkins pipeline plugin
    - https://docs.openshift.org/latest/architecture/core_concepts/builds_and_image_streams.html#pipeline-build
    - https://docs.openshift.org/latest/using_images/other_images/jenkins.html#openshift-origin-pipeline-plug-in
- **Gitlab-CI**

pipeline skipped

Continuous Integration

Howto

- clone the repository into your own namespace
- fill the following *Settings > CI / CD > Secret variables* with their respective values
  - define the ssh keys used by the terraform provisioning

    `$ ssh-keygen -N '' -f terraform`
    - SSH_TERRAFORM_PRIVATE

      `$ cat terraform | base64`
    - SSH_TERRAFORM_PUBLIC

      `$ cat terraform.pub | base64`
  - define the ssh keys used by Heketi, the glusterfs volume manager, to connect to the bricks

    `$ ssh-keygen -N '' -f heketi`
    - SSH_HEKETI_PRIVATE

      `$ cat heketi | base64`
    - SSH_HEKETI_PUBLIC

      `$ cat heketi.pub | base64`
  - define the ssh public key used for manual admin connection
    - SSH_ADMIN_PUBLIC

      `$ cat admin.pub | base64`
  - OPENSTACK_TENANT_NAME

    `for example jupytercloud-ci`
  - OPENSTACK_USER_NAME

    `for example jupytercloud-ci-user`
  - OPENSTACK_PASSWORD

Continuous Integration

```
before_script:
  - PATH=$CI_PROJECT_DIR/bin:${RUBY_PATH}:$PATH; export PATH
  - yum -y install bsdtar jq
  - mkdir -p $CI_PROJECT_DIR/bin
    && cd $CI_PROJECT_DIR/bin
    && curl --location $TERRAFORM_ARCHIVE | bsdtar -xvf-
    && chmod u+x terraform

  - mkdir -p $CI_PROJECT_DIR/config/.ssh/
    && cd $CI_PROJECT_DIR/config/.ssh/
    && echo $SSH_TERRAFORM_PRIVATE | base64 --decode > terraform
    && echo $SSH_TERRAFORM_PUBLIC  | base64 --decode > terraform.pub
    && echo $SSH_ADMIN_PUBLIC | base64 --decode > admin.pub
  - ls -alFh $CI_PROJECT_DIR/config/.ssh/
```

ENVIRONMENT VARIABLES TO FILES

TERRAFORMATION

```
terraform:
  stage: build
  tags:
  - lal
#  when: manual
  script:
  - echo CI_PROJECT_NAMESPACE=$CI_PROJECT_NAMESPACE
  - chmod -R go-rwx /builds/$CI_PROJECT_NAMESPACE
  - pwd
  - ls -alFh
  - cd $CI_PROJECT_DIR
  - terraform init
  - terraform state pull -var "openstack-tenant_name=$OPENSTACK_TENANT_NAME"
                                 -var "openstack-user_name=$OPENSTACK_USER_NAME"
                                 -var "openstack-password=$OPENSTACK_PASSWORD"
  - terraform destroy --force
  - terraform apply -auto-approve -var "openstack-tenant_name=$OPENSTACK_TENANT_NAME"
                                 -var "openstack-user_name=$OPENSTACK_USER_NAME"
                                 -var "openstack-password=$OPENSTACK_PASSWORD"
  - terraform output -json jupytercloud | jq
```

Continuous Integration
OpenSSH Certificates

- Generate the OpenSSH CA
  - ssh-keygen -t rsa -b 4096 -C 'JupyterCloud OpenSSH CA' -f ./ca/OpenSSH-CA
- Generate a key
  - ssh-keygen -t rsa -b 4096 -o -C ‘user1’ -f ~/.ssh/jupytercloud.key
- Sign a pubkey
  - ssh-keygen -s ./ca/OpenSSH-CA -I user1 -V +365d -n centos -O clear -O force-command=‘df -h’ ./ca/jupytercloud.key.pub

Continuous Integration
OpenSSH Certificates

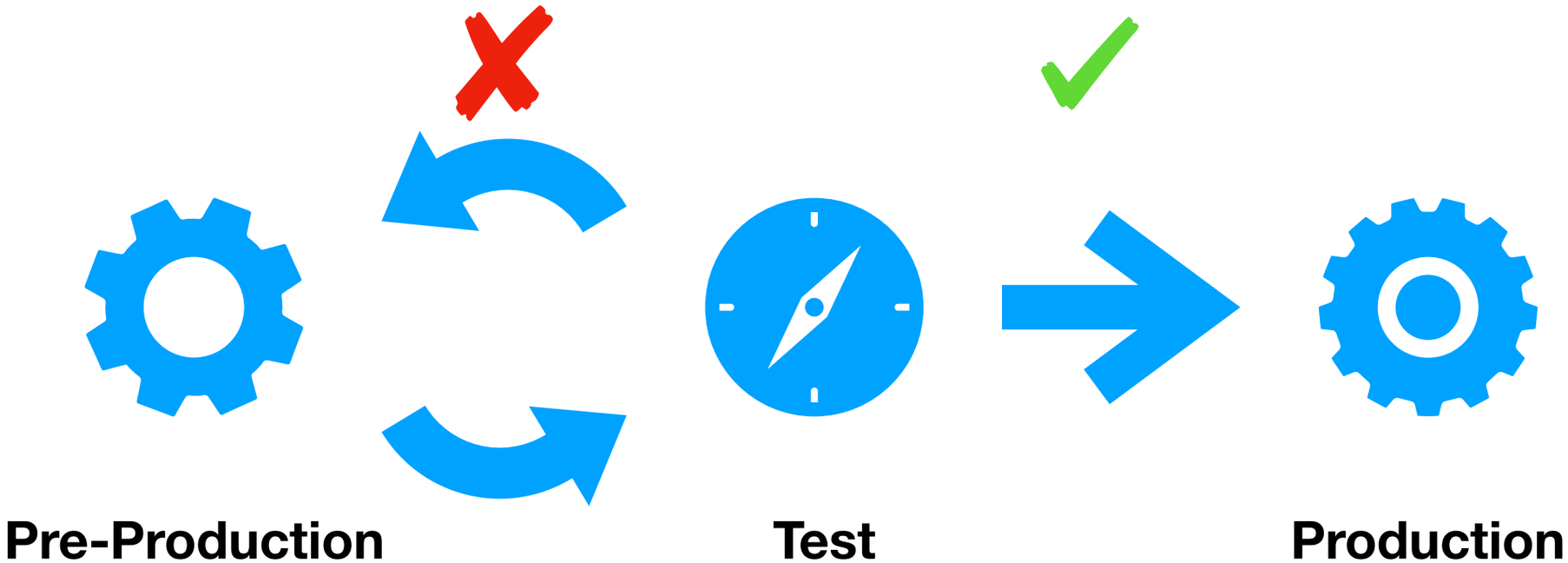
- /etc/ssh/sshd_config
  - TrustedUserCAKeys /etc/ssh/ssh_ca_user
  - AuthorizedPrincipalsFile /etc/ssh/ssh_principals
- /etc/ssh/ssh_ca_user <- contains the AC pub keys
  - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQC0ohNRv9YZgQZHqfvL4Zj3AJC8u8XjtJfgYhnyTonfD7agPBHiuPkNj0VTGlJhj7tqMLlPmQU+dJvbHnMLw8d0gb6p5Urppic0DyTdvC84AmKYUMJtaLMBz4XLDldP9FAHxmeDBd+rPSdlkHCPWhtt6LVB7uvJnRnysu4+m0/3D75hu6hHv7Nno2UJCAFm2Rn4iGuIucqBgJ1tbzoC+R3mOE6XhXmyMbgir9Tt7aH7bFVC2YYNVB0OFRspICEQrf+FDddktMlSC04LIul3JUyF4UgrFR4c0vv2vxsgVMydyL8sfKhtwtrKr26mk6zdIqU2s+28eQbP11lJA7BrvB5YVaJTgiRqr9nBqgE5U0Q9gyooSenOPBhENLvidD7O8aaASzuEcfJViqscPeWD5Wd49dnGb0Xv5KK4REtIbKSnxKWPCPrRNd6SSs0cfU1U+aoZxxz6F== OpenSSH CA
- /etc/ssh/ssh_principals <- contains the local account that accept the connection with certificates
  - centos

Continuous Integration
OpenSSH Certificates

- User connection
  - ssh node -l centos -i .ssh/me.key

Continuous Integration

Infrastructure Unit Test



Continuous Integration
Infrastructure Unit Test

- Serverspec (Ruby) based on RSpec (Ruby)
  - https://www.inspec.io/ (Ruby)
- https://github.com/philpep/testinfra (Python)
- https://github.com/aelsabbahy/goss (GO)

Continuous Integration

Infrastructure Unit Test

artifactory — -bash — 80×24

```
  the package "firewalld" should be installed
  the service "firewalld" should be enabled
  the service "firewalld" should be running
  default zone shoud be : "public"
  for the zone "public"
        interfaces
                should be : ["eth0"].
                found : ["eth0"]
  for the zone "public"
        services
                should be : ["dhcpv6-client", "http", "https", "ssh"].
                found : ["dhcpv6-client", "http", "https", "ssh"]

Service "fail2ban" :
  the package "fail2ban" should be installed
  the package "fail2ban-firewalld" should be installed
  the service "fail2ban" should be enabled
  the service "fail2Ban" should be running
  the fail2ban jail "sshd" should exist

Finished in 2.95 seconds (files took 11.39 seconds to load)
43 examples, 0 failures

station40:artifactory david_delavennat$
```

What’s next?

What’s next?

- !!! in the CI and container world, storage speed is a real bottleneck !!! -> DM-cache (NVMe, SSD, HD)
- External storage
  - object storage with OpenID connect
- OpenStack
  - use https://docs.openstack.org/kuryr/latest/
- Kata Containers (container+kernel)
  - https://katacontainers.io/posts/why-kata-containers-doesnt-replace-kubernetes/
- Bare-Metal clusters
  - https://plmlab.math.cnrs.fr/phymath/hpc
  - https://metallb.universe.tf/
  - qemu-img convert http://netboot/qemu/kubernetes-node.qcow2 -O raw /dev/sda -p
  - kexec —load vmlinuz —initrd=initrd.img —append=root=UUID=0356e691-d6fb-4f8b-a905-4230dbe62a32

What’s next?

- Chaos Engineering
  - http://principlesofchaos.org/

What’s next?

Chaos Monkey randomly terminates virtual machine instances and containers that run inside of your production environment. Exposing engineers to failures more frequently incentivizes them to build resilient services.

- https://github.com/Netflix/SimianArmy
- https://github.com/netflix/chaosmonkey

That's all Folks!

Acknowledgments

- Laboratoire de L'Accélérateur Linéaire
  - Guillaume PHILIPPON, Laboratoire de L'Accélérateur Linéaire
- GDR Calcul
  - Loic GOUARIN, Institut de Mathématiques d'Orsay
- OpenDreamKit
  - Nicolas THIERRY, Laboratoire de Recherche en Informatique
- Initiative HPC@Maths
  - François ALOUGES, Centre de Mathématiques Appliquées
  - Marc MASSOT, Centre de Mathématiques Appliquées
- Projets JupyterHub, BinderHub
  - Chris Holdgraf, Berkeley Institude for Data Science
  - Benjamin Ragan-Kelley, Simula Research Laboratory
- Mathrice
  - Philippe Depouilly, Institut de Mathématiques de Bordeaux
- PHYMATH
  - Jean-Luc BELLON, Centre de Physique Théorique

Links

- https://videos.cdn.redhat.com/summit2015/presentations/17856_getting-the-most-out-of-your-nvme-ssd.pdf
- https://jfrog.com/blog/gitlab-and-artifactory-on-your-mark-get-set-build/
- https://github.com/itamae-kitchen/itamae