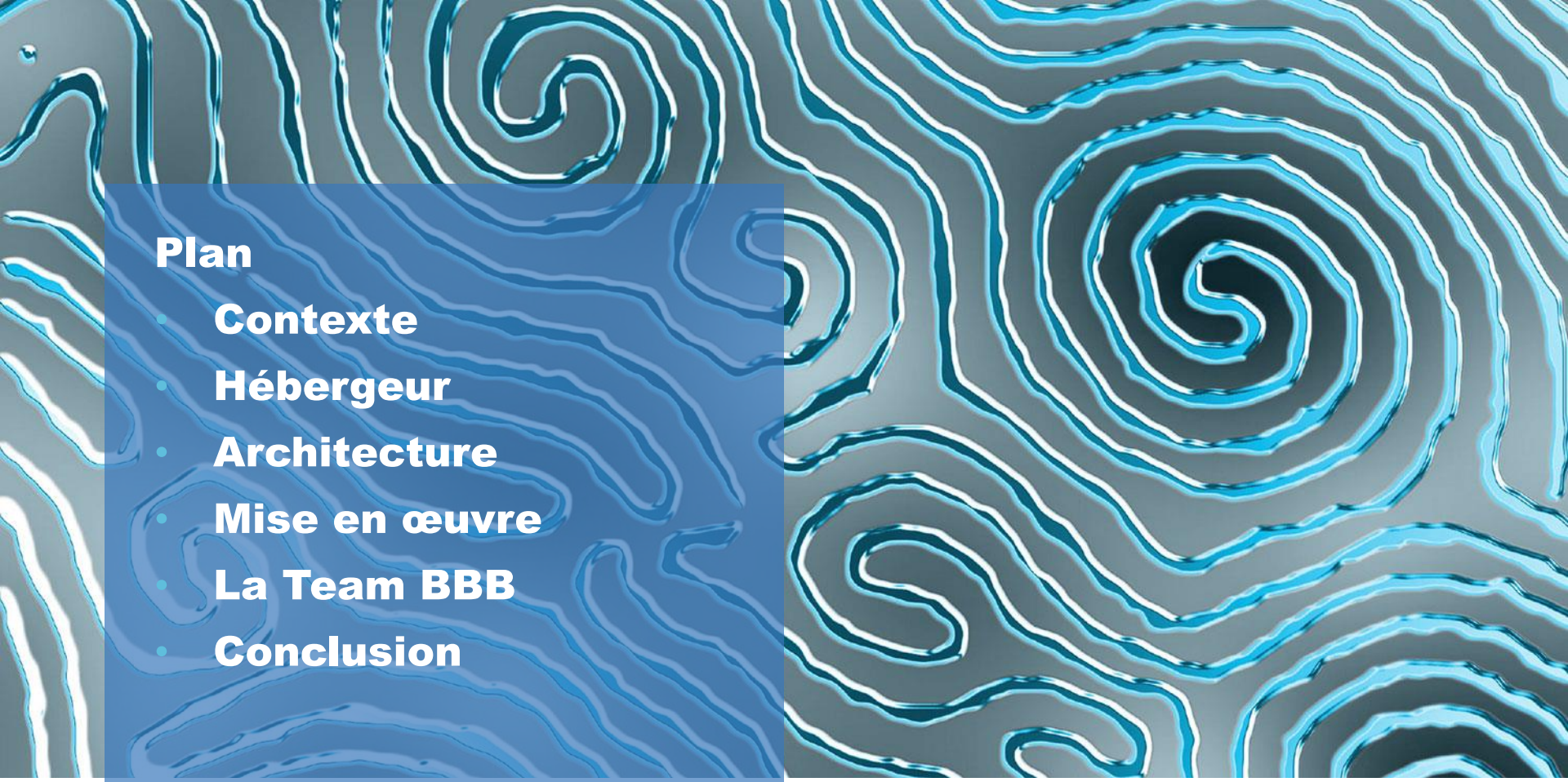
Action Nationale de Formation

Infrastructure, plateforme ou application en tant que service, quels choix technologiques pour les ASR de laboratoire ?

Retex BBB National

Franck LICHNOWSKI - CNRS – Team BBB

21 novembre 2022 - CIRM



Plan

- Contexte
- Hébergeur
- Architecture
- Mise en œuvre
- La Team BBB
- Conclusion

CONTEXTE DU PROJET

- **Besoin exponentiel en visio depuis 2020**

  - BBB chez Mathrice
  - BBB campus Joseph Aiguier (mars 2020)
  - Déploiement d'instances dans d'autres DR / Campus
  - Satisfaction sur le fonctionnement
  - Pas de solution nationale

  - Création de 3 GT au niveau DSI pilotés par les RSI (été 2021) :
    - Solution interne avec BBB
    - Solution externalisé avec BBB
    - Solution externalisé autre que BBB

CONTEXTE DU PROJET

- **Décision PDG : 14/12/2021**
  - Solution BBB avec hébergement & déploiement interne retenue
  - 1 500 utilisateurs simultanés
  - Enregistrement autorisé (sous conditions)
  - SLA : 5/7 aux heures de bureau (cette offre est complémentaire aux autres solutions de visio déjà disponibles)
  - Investissement initial : 2 * 17 k€ = 34 k€ (financé par la DSI)
  - Deadline : janvier 2022
  - L' équipe en charge de l'installation sera responsable du MCO.

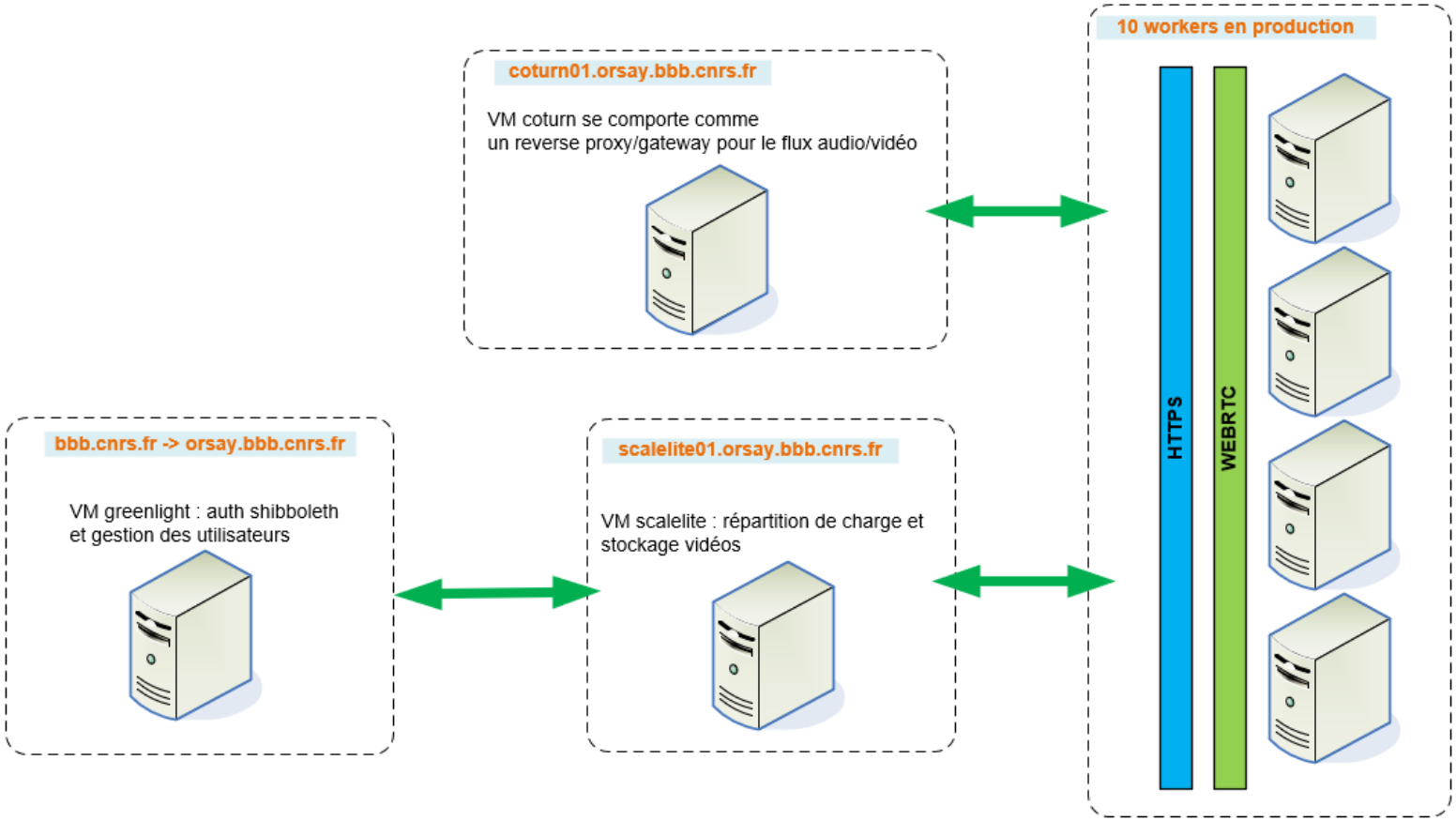
HEBERGEUR

- **VirtualData**

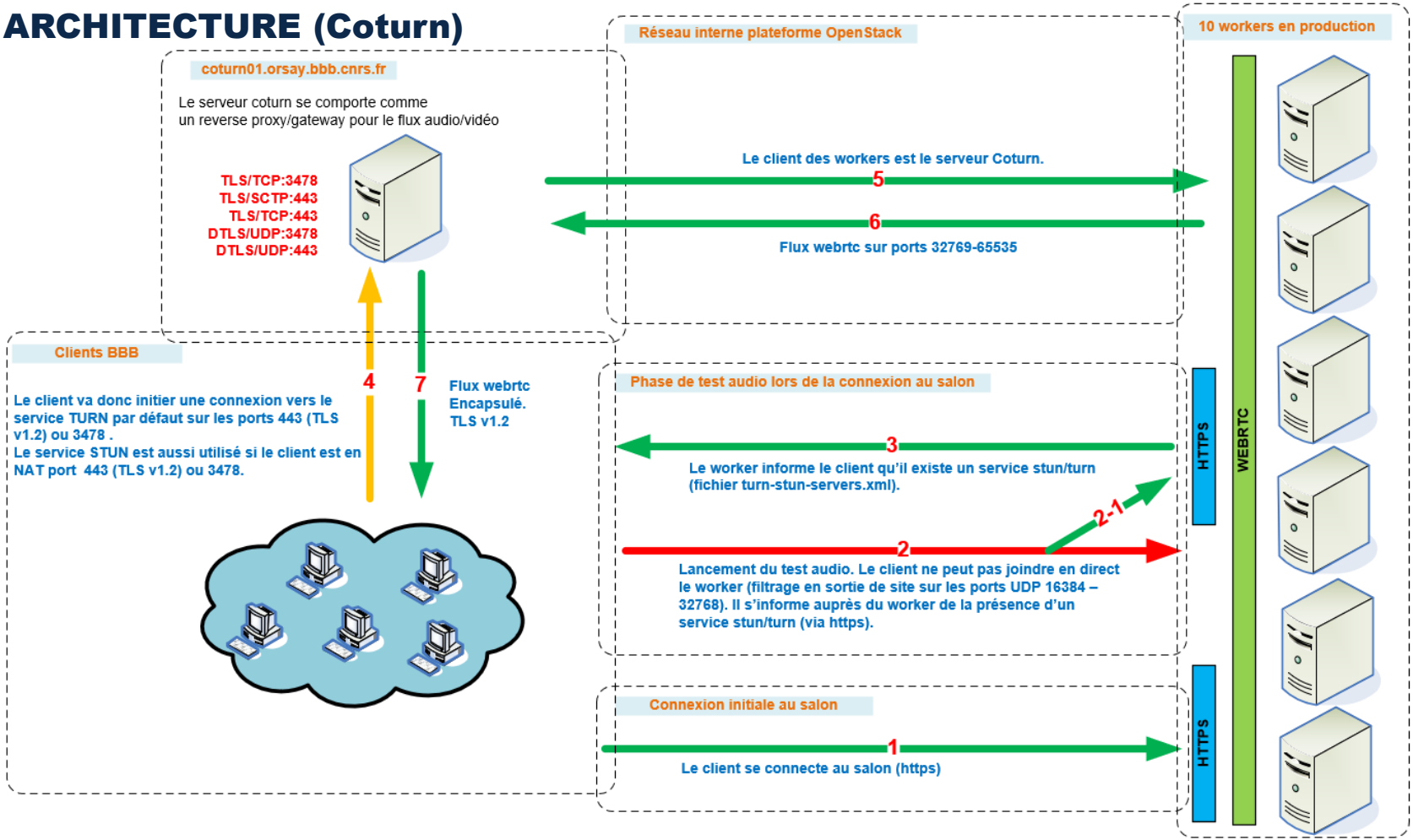
  - Une des 3 plateformes techniques du mésocentre Université Paris Saclay
  - Technologie éprouvée Openstack
  - 10 000 cœurs disponibles
  - 1PB de stockage semi-permanent
  - Offre IaaS

  - Ressources actuellement utilisées pour la plateforme BBB :
    - 27 VMs déployés (sur 40 instances allouées)
    - 132 vCPU (sur 256 alloués)
    - 264 Go de RAM (sur 512 alloués)
    - 2 To d’espace disque (sur 4.9 alloués)

ARCHITECTURE (front-end)



ARCHITECTURE (Coturn)



MISE EN OEUVRE

- **Greenlight**
  - Installation d'apache (certif, durcissement ciphers + protocoles) – guide TLS 1.2 ANSSI
  - Installation du SP shibboleth
  - Installation (via un gitclone)
  - Patch du container docker :
    - Authentification shibboleth
    - Centralisation des logs
    - Correctif de la base de données (ajout de 3 champs)
    - Configuration du proxy vers le container docker de greenlight
    - Ajout des boutons pour l'assistance + lien de DL du MP4

  - Rajout dans la BDD des comptes admins
  - Gestion des comptes (script si maj de l'email, suppression auto …)

MISE EN OEUVRE

- **Scalelite**
  - Installation (via un gitclone)
  - Génération et configuration des secrets via openssl (openssl rand …)
  - Installation certificat Sectigo
  - Durcissement (ciphers + protocoles)
  - Préparation du volume pour stocker les enregistrements
  - Installation et configuration de plusieurs containers docker
    - scalelite-api (liaison avec api BBB) et scalelite-proxy (terminaison SSL, mise à dispo enregistrement)
    - scalelite-poller (HA BBB)
    - scalelite-recording-importer (gestion des enregistrements transférés depuis les workers + ajout BDD).
  - Centralisation des logs

MISE EN OEUVRE

- **Coturn**
  - Installation (via apt)
  - Génération et configuration du secret (partagé avec les workers)
  - Installation certificat Sectigo
  - Configuration du service Coturn
  - Durcissement (ciphers + protocoles)
  - Durcissement de l’OS avec AppArmor
  - Centralisation des logs
  - Génération des fichiers des workers (turn-stun-servers.xml)

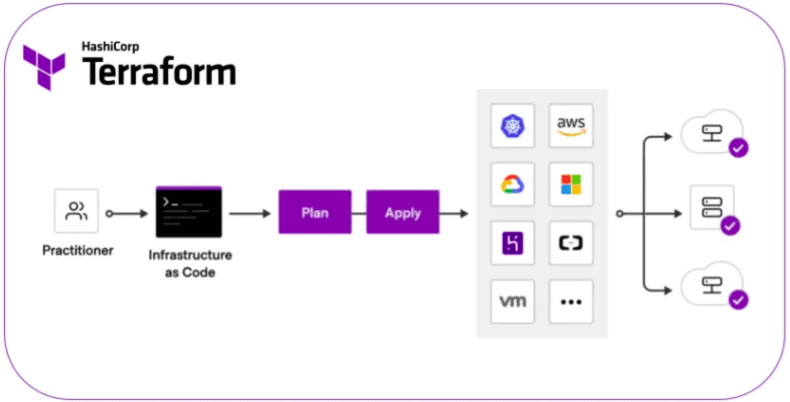
MISE EN OEUVRE

- **Terraform**
  - **Problématique :** à la vue du cahier des charges (1500 users), il nous faut 10 workers ou plus si monté en charge
  - Il existe des outils d'automatisation, le choix se porte sur Terraform
  - Déploiement d'une VM Terraform
    - Installation des prérequis (python + CLI Openstack)
  - Création d'instantanés qui servent de base au déploiement
  - Mise à jour régulière de ces instantanés
  - Déploiement de l'ensemble des workers en quelques minutes
  - Déploiement de l'environnement de recette en quelques minutes
  - Redéploiement de toutes les instances nécessaires en quelques minutes

MISE EN OEUVRE

- **Terraform**

  - Outil d'orchestration utilisé pour déployer une infrastructure via le IAC (Infrastructure en tant que code),
  - Prend en charge 145 fournisseurs,
  - Architecture est très simple, un binaire,
  - Utilise dans note cas le provider OpenStack,
  - Opensource
  - Permet de déployer en quelques minutes les Vms dont nous avions besoin, de tout reconstruire de zéro



MISE EN OEUVRE

```
# Configure the OpenStack Provider
terraform {
  required_providers {
    openstack = {
      source = "terraform-provider-openstack/openstack"
    }
  }
}
provider "openstack" {
  cloud  = "openstack" # cloud defined in cloud.yml file
}

# Variable image ##########################################################################
variable "image_worker"{
  type = string
  default = "bbbcnrs-worker01-2022-10-07"
}

variable "image_bastion"{
  type = string
  default = "bbbcnrs-bastion-rec"
}

variable "image_greenlight"{
  type = string
  default = "bbbcnrs-greenlight01-snap"
}
variable "image_scalelite"{
  type = string
  default = "bbbcnrs-scalite-rec-2022-02-16"
}
variable "image_coturn"{
  type = string
  default = "bbbcnrs-coturn01-terraform"
}

##########################################################################
```

```
# Variables global

variable "network_public" {
  type = string
  default = "public-2"
}

variable "network_int" {
  type = string
  default = "bbb-internal"
}
```

```
# Variable scalelite

variable "security_groups_scalelite" {
  type    = list(string)
  default = ["SSH-ICMP-FW","SCALELITE-FW"]  # List security group
}
```

```
# Create an instance
resource "openstack_compute_instance_v2" "bbbcnrs-scalelite01-rec" {
  name            = "bbbcnrs-scalelite01-rec"
  image_name      = var.image_scalelite
  flavor_name     = var.flavor_scalelite
  key_pair        = var.keypair
  security_groups = var.security_groups_scalelite
  force_delete    = false

  network {
    name = var.network_public
    port = var.port_scalelite_public
  }
 network {
    name = var.network_int
    port = var.port_scalelite_int
  }
```

Aperçu d’un fichier de conf pour le déploiement de la recette.

MISE EN OEUVRE

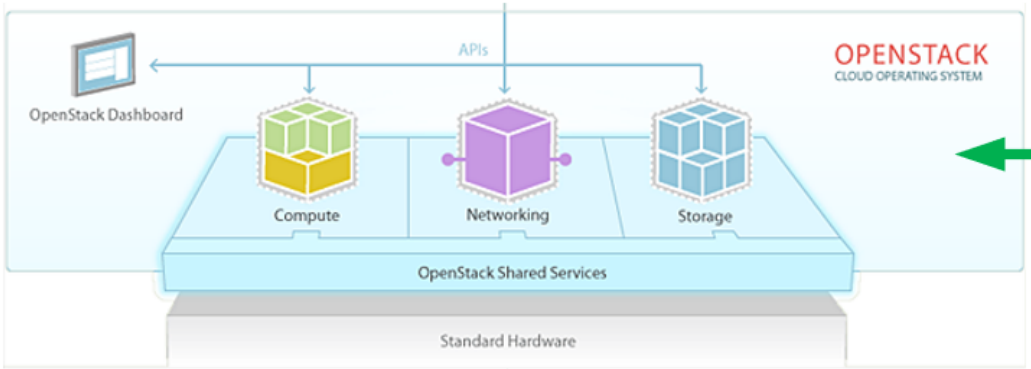
- **Ansible**

  - De la même manière, nous avions besoin d'un outil d'orchestration pour automatiser un maximum la configuration des différentes VMs
    - Fichier /etc/hosts
    - Clés SSH (exploitation)
    - Personnalisation des workers
    - Mise à jour des OS des workers
    - Installation et mise à jour des versions de BBB
    - Remplacement sur les workers de certains fichiers lors du déploiement via Terraform (doc en pdf, favicon, turn-stun-servers.xml …)

  - Le choix s'est porté sur Ansible.

MISE EN OEUVRE

virtualdata



Workers + VMs tierces

bbb-terraform

VM Terraform : automatisation du déploiement des VMs

bbb-ansible

VM Ansible : Automatisation de la configuration des workers, mise à jour des serveurs

MISE EN OEUVRE

27 Vms (ou instances)	Production	Recette
GreenLight	X	X
Scalelite	X	X
Coturn	X	X
Workers (10 + 3)	X	X
Nagios	X	
Monitor (Grafana)	X	
Graylog	X	
Ansible	X	
Terraform	X	
Bastion (accès ssh)	X	X

MISE EN OEUVRE

- **Pourquoi ça fonctionne**

  - Il y a des personnes compétentes en dehors de la DSI (sic)
  - Bonne entente entre les différentes DR
  - Appui du projet par le nouveau DSI et d'autres personnes « influentes »

  - Des personnes compétentes chez VirtualData
  - Une infrastructure d'hébergement robuste très flexible (ajout de ressources …)

  - MCO :
    - Faite en interne par les même personnes
    - Excellente réactivité

TEAM INITIALE

Virtualdata

Responsable : Michel JOUVIN

Admin Openstack : Guillaume PHILIPPON

Admin BBB Mathrice : David DELAVENNAT

SSI Délégation

Bordeaux : Roland DIRLEWANGER, Jimmy LABEJOF, Steeve PLACIDE

Strasbourg : Jean-Luc ORCESI, Xavier DUTHILLEUL, Baptiste BARAKOWSKY

Toulouse : Frederic DRUILHET, Philippe DUBRULLE

Marseille : Franck LICHNOWSKI

Caen : Anthony CARVIN

Grenoble : Dominique FOURNIER, Laurent NEIHGER

Gif-sur-Yvette : Eric LECOMPTE, Philippe PEYNOT, Laurent LECLERCQ

DSI

Philippe BENEZETH, Michel CHABANNE, Hélène DALLERY, Olivier LENORMAND

CONCLUSION

- Confortable d’être sur de l’IAAS pour ce genre de projet
- Service national mise en production dans les délais malgré la forte contrainte de calendrier
- S’appuyer sur les infrastructures offertes par la communauté en mode IAAS (campus, réseau métier, offre univ …)
- Utiliser les outils d’automatisation comme Terraform ou Ansible bien entendu en fonction du besoin et du contexte