

# Architecture Réseaux Entreprises (ARES)

## Automatisation de l'infrastructure réseau

---

Brice - Ekane (brice.ekane@univ-rennes.fr)

ISTIC Rennes - France  
2025-2026

git clone <https://github.com/name/ares-2025.git>

# Plan du module

---

- 1 Introduction et Objectifs du Module
- 2 NetDevOps : Fondements et Philosophie
- 3 NetDevOps : comment ça marche ?
- 4 Outils d'Automatisation Réseau
- 5 Le Pipeline d'Automatisation Réseau

- 1 Introduction et Objectifs du Module
- 2 NetDevOps : Fondements et Philosophie
- 3 NetDevOps : comment ça marche ?
- 4 Outils d'Automatisation Réseau
- 5 Le Pipeline d'Automatisation Réseau

# Objectifs pédagogiques

---

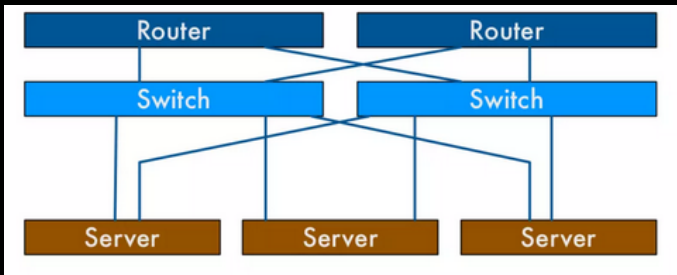
À l'issue du module, vous serez capables de :

- ▶ Expliquer les enjeux du NetDevOps et de l'Infrastructure-as-Code.
- ▶ Utiliser les principaux outils d'automatisation réseau (Ansible, Terraform, Python, Cloud-init).
- ▶ Concevoir un pipeline complet : inventaire → génération → déploiement → validation.
- ▶ Appliquer les bonnes pratiques : GitOps, gestion des secrets, tests automatisés.

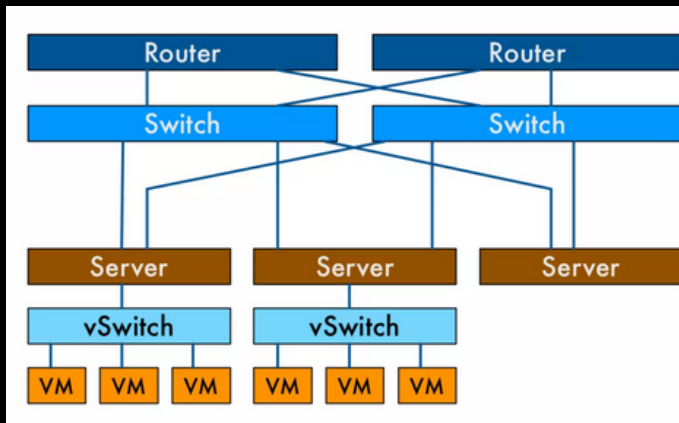
Pourquoi automatiser le réseau ?

# Le réseau

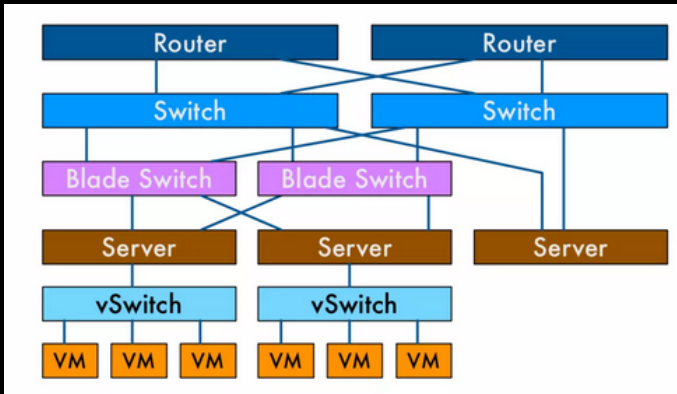
---



# Le réseau

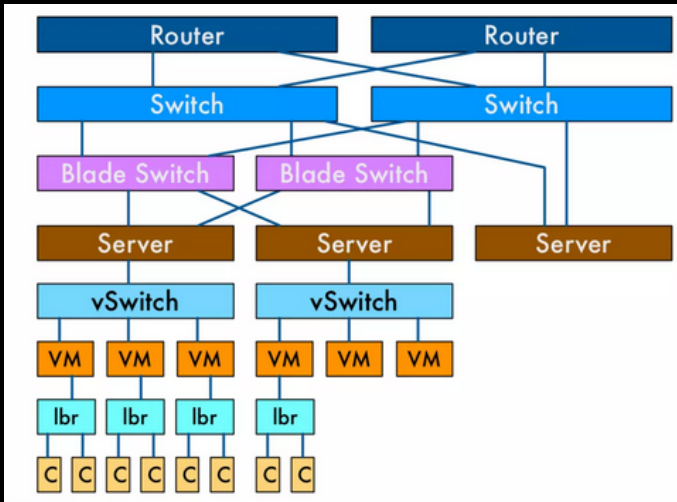


# Le réseau

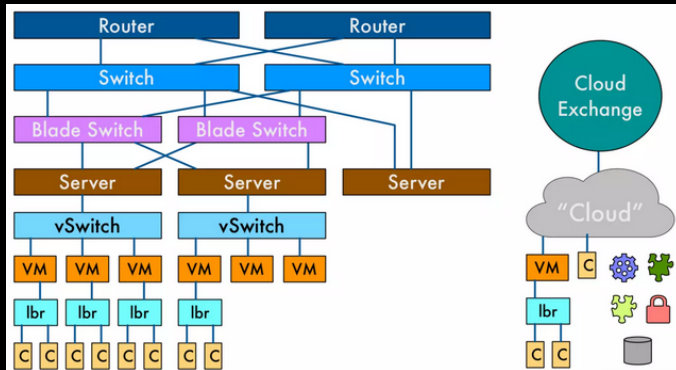




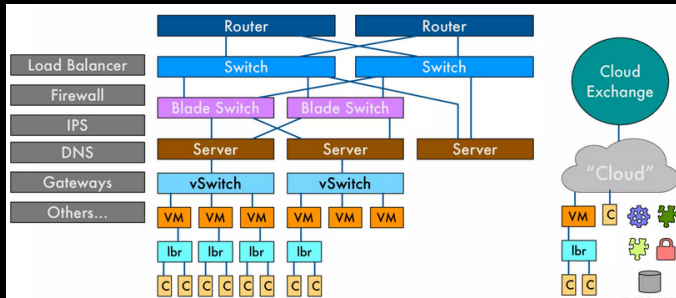
# Le réseau



# Le réseau



# Le réseau



# Impact des changements réseau

---

## Chiffres clés issus d'études

Année	Source	Population	Résultat
2016	Veriflow / Dim. Research	IT / Réseau	74% : changements impactent fortement plusieurs fois/an
2025	Digi International	Entreprises	84% : pannes réseau significatives en 2 ans

**Figure 12: What areas have the greatest importance in your public 5G SA services?**

*(Instructions to the survey respondents: select the top 3)*

Automation of provisioning, management, and orchestration



Monitoring, analytics, and reporting



Technical skills and retention



Power and energy efficiencies



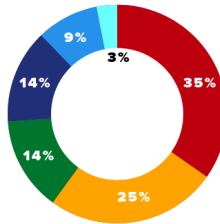
CI/CD/CT pipeline optimization and lifecycle operation



Notes: n=100

Source: Heavy Reading | © 2023 Heavy Reading

**Figure 11:** What aspect has been the most challenging for delivering end-to-end public 5G SA services?



Notes: n=100  
Source: Heavy Reading  
© 2023 Heavy Reading

# Défis majeurs pour la 5G SA (Figure 11)

## Opinions des opérateurs télécom (2023)

Classement	Défi principal	Réponses
1	Construction de l'infrastructure réseau (cloud-native, IaaS/PaaS, hybrid cloud)	35% global — jusqu'à 48% aux USA
2	Intégration des fonctions du cœur de réseau (5G Core)	25%
3	Assurance réseau et DevOps	14%

# Configuration manuelle du réseau

---

## Limites principales

- ▶ Conception, exploitation et maintenance encore largement manuelles.
- ▶ Processus lents et sujets aux erreurs.
- ▶ Génération d'incohérences et d'instabilités.
- ▶ Connexion répétitive aux équipements (CLI, interfaces, etc.).
- ▶ Difficulté à suivre l'évolution rapide des charges de travail modernes.

Source : Red Hat, <https://www.redhat.com/fr/topics/automation/what-is-network-automation>



# Automatisation du réseau

---

## Avantages principaux

- ▶ Suppression des étapes manuelles de configuration.
- ▶ Standardisation des processus et réduction des écarts.
- ▶ Déploiement rapide et à grande échelle.
- ▶ Réduction du temps moyen de résolution des incidents.
- ▶ Optimisation de l'équilibrage de charge et du basculement.

Source : Red Hat, <https://www.redhat.com/fr/topics/automation/what-is-network-automation>

- 1 Introduction et Objectifs du Module
- 2 NetDevOps : Fondements et Philosophie**
- 3 NetDevOps : comment ça marche ?
- 4 Outils d'Automatisation Réseau
- 5 Le Pipeline d'Automatisation Réseau

# Définition du NetDevOps

---

## Définition

La convergence entre les pratiques DevOps et les opérations réseau.

## Définition selon Red Hat

Le terme « NetOps » (ou « NetDevOps ») désigne une approche de l'exploitation des réseaux orientée vers la rapidité des déploiements et l'agilité dans les organisation numériques.

# Principes clés du NetDevOps (Red Hat)

## Fondements

- 1 **Automatisation et orchestration** : tâches répétitives et workflows gérés automatiquement.
- 2 **Infrastructure as Code (IaC)** : configurations décrites en code, versionnées et auditées.
- 3 **Validation continue** : tests et CI/CD appliqués au réseau.
- 4 **Source unique de vérité** : centralisation des inventaires et politiques.
- 5 **Collecte et analyse des données** : supervision, optimisation et détection proactive.
- 6 **Sécurité intégrée** : politiques automatisées et conformité vérifiée en continu.
- 7 **Agilité du réseau** : adaptation rapide aux environnements hybrides et multicloud.

# Enjeux du NetDevOps

---

## Objectif

Passer de l'interface CLI à une approche déclarative.

# Enjeux du NetDevOps : les défis

---

## Constat

- ▶ Réseaux modernes = hybrides, multicloud, distribués (datacenter, cloud, edge).
- ▶ Gestion manuelle (CLI, scripts ad hoc) :
  - ▶ Trop lente face aux besoins métiers.
  - ▶ Source d'erreurs humaines.
  - ▶ Complexe à sécuriser et auditer.

# Enjeux du NetDevOps : les objectifs

---

## Valeur ajoutée

---

- ① **Accélérer les déploiements** : mise en service plus rapide.
- ② **Fiabiliser et standardiser** : moins d'erreurs, validation continue.
- ③ **Renforcer la sécurité** : politiques intégrées, réactions automatisées.
- ④ **Améliorer la collaboration** : convergence réseau, systèmes, DevOps.
- ⑤ **Réduire les coûts opérationnels** : tâches industrialisées, ressources optimisées.

# Infrastructure-as-Code (IaC)

---

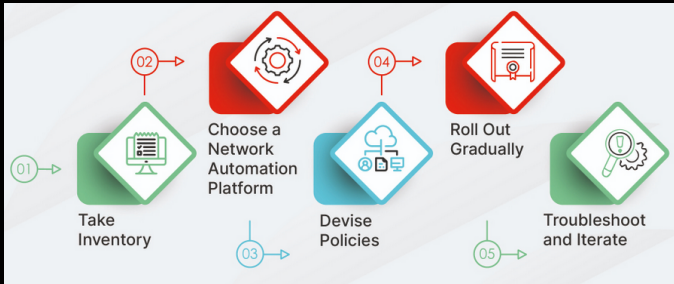
## Principes clés

---

- ▶ L'état d'un équipement est décrit dans des fichiers, pas via des commandes manuelles.
- ▶ Basé sur le *desired state* : l'outil applique la configuration.
- ▶ L'outil garantit la conformité de l'état final.
- ▶ Configurations reproductibles et versionnées.



- 1 Introduction et Objectifs du Module
- 2 NetDevOps : Fondements et Philosophie
- 3 NetDevOps : comment ça marche ?
- 4 Outils d'Automatisation Réseau
- 5 Le Pipeline d'Automatisation Réseau



Source : <https://www.fortinet.com/fr/resources/cyberglossary/network-automation>

- 1 Introduction et Objectifs du Module
- 2 NetDevOps : Fondements et Philosophie
- 3 NetDevOps : comment ça marche ?
- 4 Outils d'Automatisation Réseau**
- 5 Le Pipeline d'Automatisation Réseau

## Principes clés

- ▶ Initialisation automatique des VM/instances au premier boot.
- ▶ Configuration réseau, utilisateurs, clés SSH, packages.
- ▶ Utilisé dans OpenStack, AWS, Azure, etc.

# Pourquoi Cloud-init ?

evite d'avoir à gérer les milliers d'images préconfigurées, Exemple AWS

EC2 Management Console

Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

### Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

Quick Start

- My AMIs
- AWS Marketplace
- Community AMIs
- ☐ Free tier only

1 to 36 of 36 AMIs

**Amazon Linux 2 AMI (HVM), SSD Volume Type** - ami-0922553b7b0369273

Amazon Linux Free tier eligible

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

64-bit

**Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type** - ami-0ff8a91507f7f867

Amazon Linux Free tier eligible

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl,

64-bit

# Pourquoi Cloud-init ?

permet de réduire le nombre d'images préconfigurées, Exemple AWS

EC2 Management Console

Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI) [Cancel and Exit](#)

Quick Start

My AMIs

AWS Marketplace

Community AMIs

☐ Free tier only

Amazon Linux Free tier eligible

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0922553b7b0369273

64-bit

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Amazon Linux Free tier eligible

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0ff8a91507f7f867

64-bit

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl.

Feedback English (US) © 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

# Cloud-init : Usages typiques

---

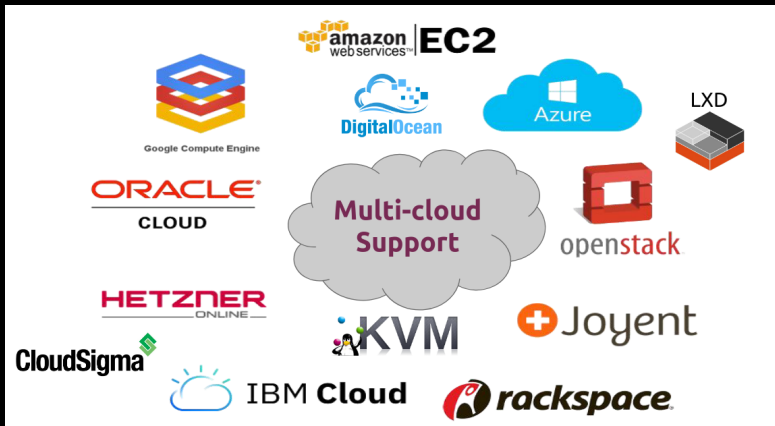
- ▶ Configurer le **réseau** (IP, DNS, routes).
- ▶ Créer des **utilisateurs** et leurs clés SSH.
- ▶ Installer des **packages** (nginx, python, ansible...).
- ▶ Exécuter des **scripts personnalisés**.

# Cloud-init : Support Multi Os





# Cloud-init : Support multi fournisseurs cloud



# Exemple Cloud-init

---

```
#cloud-config
hostname: vm-demo
users:
  - name: student
    ssh-authorized-keys:
      - ssh-rsa AAAAB3Nz...
packages:
  - nginx
  - python3
```

# Phases de cloud-init

## Vue d'ensemble des étapes

### ► Phase locale (local stage)

- Rôle : Récupérer les informations locales indispensables avant toute connexion réseau.
- Exemple d'actions : Détection des sources de données présentes et initialisation minimale.

### ► Phase réseau (network stage)

- Rôle : Mettre en place la connectivité réseau de l'instance.
- Exemple d'actions : Activation des interfaces, attribution d'adresses IP, configuration DNS.

### ► Phase de configuration (config stage)

- Rôle : Déployer la configuration système et préparer l'environnement utilisateur.
- Exemple d'actions : Création de comptes, configuration SSH, installation de paquets, gestion des services.

### ► Phase finale (final stage)

- Rôle : Exécuter les derniers réglages pour rendre l'instance opérationnelle.
- Exemple d'actions : Lancement de scripts finaux, démarrage des services, notifications éventuelles.

### ► Redémarrage optionnel

- Rôle : Redémarrer l'instance si la configuration l'exige.
- Exemple d'actions : Redémarrage automatique afin d'appliquer certains changements.

# Cloud-init : Exemple

---

```
1  #cloud-config
2  # **** Phase locale (local stage) ****
3  hostname: ares-instance
4  fqdn: ares-instance.local
5  manage_etc_hosts: true
6
7  # Préparation d'un disque supplémentaire
8  mounts:
9    - [ /dev/vdb, /mnt/ares/shared, "auto", "defaults,nofail", "0", "2" ]
10
11  # ***** Phase réseau (network stage) *****
12  network:
13    version: 2
14    ethernet:
15      ens3:
16        dhcp4: true
17        dhcp6: false
18
19  # ***** Phase de configuration (config stage) *****
20  users:
21    - name: etudiant
22      groups: sudo
23      shell: /bin/bash
24      sudo: ['ALL=(ALL) NOPASSWD:ALL']
25      ssh-authorized-keys:
26        - ssh-rsa ma super clé publique rsa
27
```

# Cloud-init : Exemple (2/2)

---

```
1
2 packages:
3   - htop
4   - curl
5   - git
6
7 write_files:
8   - path: /etc/motd
9     permissions: '0644'
10    content: |
11      Bienvenue sur cette instance configurée avec cloud-init !
12    # =====
13    # Phase finale (final stage)
14    # =====
15  runcmd:
16    - systemctl enable ssh
17    - systemctl restart ssh
18    - echo "Instance prête !" > /home/ares/log/ready.txt
19
20  # =====
21  # Redémarrage optionnel
22  # =====
23  power_state:
24    mode: reboot
25    message: "Redémarrage automatique pour appliquer la configuration"
26    timeout: 30
27    condition: True
```

# Ressources et lectures recommandées

---

- ▶ Cloud-init (documentation officielle) <https://app.readthedocs.org/projects/cloudinit/downloads/pdf/latest/>
- ▶ Guide RHEL / Red Hat sur Cloud-init [https://docs.redhat.com/fr/documentation/red\\_hat\\_enterprise\\_linux/9/pdf/configuring\\_and\\_managing\\_cloud-init\\_for\\_rhel\\_9/red-hat-support-for-cloud-init\\_cloud-content](https://docs.redhat.com/fr/documentation/red_hat_enterprise_linux/9/pdf/configuring_and_managing_cloud-init_for_rhel_9/red-hat-support-for-cloud-init_cloud-content)
- ▶ Présentation « The cross-cloud magic sauce » (Canonical) ([quelques slides issues de la présentation](https://events19.linuxfoundation.org/wp-content/uploads/2017/12/cloud-init-The-cross-cloud-Magic-Sauce-Scott-Moser-Chad-Smith-Canonical.pdf))  
<https://events19.linuxfoundation.org/wp-content/uploads/2017/12/cloud-init-The-cross-cloud-Magic-Sauce-Scott-Moser-Chad-Smith-Canonical.pdf>
- ▶ Slides sur les machines virtuelles [https://mcorbin.fr/pdf/slides/virtual\\_machines.pdf](https://mcorbin.fr/pdf/slides/virtual_machines.pdf)
- ▶ Manuel RHEL 8 : gestion de Cloud-init  
[https://docs.redhat.com/en/documentation/red\\_hat\\_enterprise\\_linux/8/pdf/configuring\\_and\\_managing\\_cloud-init\\_for\\_rhel\\_8/Red\\_Hat\\_Enterprise\\_Linux-8-Configuring\\_and\\_managing\\_cloud-init\\_for\\_RHEL\\_8-en-US.pdf](https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/8/pdf/configuring_and_managing_cloud-init_for_rhel_8/Red_Hat_Enterprise_Linux-8-Configuring_and_managing_cloud-init_for_RHEL_8-en-US.pdf)
- ▶ Actes de conférence « Autoinstall / cloud provisioning »  
<https://indico.mathrice.fr/event/580/attachments/1084/1598/Autoinstall.pdf>

# Ansible : le couteau suisse du réseau

---

## Principes clés

---

- ▶ **Agentless** : pas d'agent sur les équipements (SSH/API).
- ▶ **Lisible** : YAML, accessible même aux non-développeurs.
- ▶ **Modulaire** : support multi-constructeurs.
- ▶ **Intégrable** : compatible CI/CD.

sera vue en détail dans TLC, le TP sur ansible est un TP à trous

# Ansible : installation

---

```
1 sudo apt-get update && sudo apt install ansible
```



# Ansible : inventaire

---

---

```
1 [ares@alpine ~]$ mkdir ansible ; cd ansible
2 [ares@alpine ~]$ vim inventory
3 [router1]
4 192.168.124.11
```

---

# Ansible : playbook

---

```
1 - name: Création VLAN 100
2   hosts: router1
3   tasks:
4     - cisco.ios.ios_config:
5       lines: name VLAN 100
6       parents: vlan 100
```

---

# Ansible : play

---

```
1  # Nomage
2  - name: Création VLAN 100
3  # Selection des hôtes
4    hosts: router1
5  # Tâches
6    tasks:
7  # Utilisation des modules
8      - cisco.ios.ios_config:
9          lines: name VLAN 100
10         parents: vlan 100
```

---

# Ansible : execution

---

```
1 [ares@alpine ~] ansible-playbook play.yml
```

# Ansible : execution en mode dry-run

---

```
[ares@alpine ~] ansible-playbook play.yml --check
```

# Ansible : concepts et notions avancés

---

Rendez-vous sur le cours de TLC !

# Ressources Ansible

---

- ▶ <https://people.redhat.com/mlessard/mtl/presentations/apr2018/AnsibleF5WorkshopVF.pdf>
- ▶ **Documentation officielle Ansible** <https://docs.ansible.com/>
- ▶ **Collection Ansible Cisco IOS** (modules réseaux) <https://docs.ansible.com/ansible/latest/collections/cisco/ios/>
- ▶ **Ansible Galaxy** (modules et rôles communautaires) <https://galaxy.ansible.com/>
- ▶ **Exemples de playbooks Ansible** (GitHub officiel) <https://github.com/ansible/ansible-examples>

# Terraform : Principes

---

## Principes clés

- ▶ Outil IaC orienté gestion de ressources.
- ▶ Utilise des *providers* pour interagir avec des API.



# Caractéristiques de Terraform

---

## Principes clés

- ▶ Utilise des **plugins** appelés *Providers* pour interagir avec les services cloud.
- ▶ Conçu pour fonctionner avec différents **fournisseurs de cloud** (AWS, Azure, GCP, etc.).
- ▶ Aussi avec **libvirt**.
- ▶ Possibilité de créer ses propres *providers* en Go.
- ▶ Ciblé principalement sur le **provisionnement d'infrastructure**.

# Terraform : Cas d'usage

---

## Exemples

- ▶ Cloud : déploiement de VPC, sous-réseaux, tables de routage.
- ▶ SDN : configuration de switchs virtuels OVS via un contrôleur.

# Terraform: Installation

---

```
1 sudo apt-get update && sudo apt-get install -y gnupg software-properties-common curl
2 curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
3 sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com \
4     $(lsb_release -cs) main"
5 sudo apt-get update
6 sudo apt-get install terraform
```

---

maintenant payant! opentofu est le fork de terraform pour la version opensource

```
1 sudo apt-get install opentofu
```

---

# Terraform : Configuration

---

```
1 terraform {
2     required_providers {
3         libvirt = {
4             source = "dmacvicar/libvirt"
5             version = "0.6.14"
6         }
7     }
8 }
9
10 provider "libvirt" {
11     uri = "qemu:///system"
12 }
13
```

---

# Terraform : Configuration

---

```
1  # Per-VM volumes cloned from the base
2  resource "libvirt_volume" "vm2_disk" {
3      name          = "vm2.qcow2"
4      pool          = "default"
5      base_volume_id = libvirt_volume.alpine318_base.id
6      format        = "qcow2"
7  }
8
9  resource "libvirt_volume" "big_vms_disk" {
10     for_each      = toset(["vm1", "vm3", "vm4"])
11     name          = "${each.key}.qcow2"
12     pool          = "default"
13     base_volume_id = libvirt_volume.alpine318_base.id
14     format        = "qcow2"
15 }
16
```

---

# Terraform : Execution

---

```
1      #Configurer le provider
2      terraform init
3      #Créer les ressources
4      terraform apply
5      #Détruire les ressources
6      terraform destroy
7
```

---

# Ressources opentofu/terraform

---

► <https://opentofu.org/docs/>

## Principes clés

- ▶ Librairie Python pour simplifier l'accès SSH.
- ▶ Gère l'envoi de commandes et l'analyse des sorties.
- ▶ Cas d'usage : scripts Python ad-hoc.



# netmiko : Exemple

---

```
1  from netmiko import ConnectHandler
2
3  cisco_device = {
4      'device_type': 'cisco_ios',
5      'ip': '192.168.1.1',
6      'username': 'admin',
7      'password': 'password',
8  }
9
10 net_connect = ConnectHandler(**cisco_device)
11 output = net_connect.send_command('show processes cpu')
12 print(output)
13 net_connect.disconnect()
```

---

# Ressources Netmiko

---

## Documentation et guides

- ▶ PyPI – page officielle : documentation, versions, installation
- ▶ Guide Cisco + Netmiko : tutoriel complet avec exemples appliqués
- ▶ Exemples GitHub : cas pratiques (commandes, configuration, transferts, parsing)

- 1 Introduction et Objectifs du Module
- 2 NetDevOps : Fondements et Philosophie
- 3 NetDevOps : comment ça marche ?
- 4 Outils d'Automatisation Réseau
- 5 Le Pipeline d'Automatisation Réseau

# Pipeline : Inventaire

---

## Principe

Définir les équipements (IP, OS, rôle). Exemples : NetBox, fichiers YAML.

# Pipeline : Génération de configuration

---

## Principe

Générer dynamiquement les configurations à partir de variables. Exemple : Jinja2.

# Pipeline : Déploiement

---

## Principe

Appliquer automatiquement les configurations générées. Exemple : Ansible.

# Pipeline : Test et validation

---

## Principe

Vérifier que le réseau correspond à l'état attendu. Exemples : Batfish, tests end-to-end.

# Bonnes pratiques : Git

---

## Principes clés

- ▶ Git est la source de vérité.
- ▶ Chaque commit doit être clair et traçable.
- ▶ Utiliser des branches pour isoler.
- ▶ Revue de code avant fusion.



# Bonnes pratiques : Sécurité des secrets

---

## Principes clés

---

- ▶ Ne jamais stocker de secrets en clair dans Git.
- ▶ Utiliser un gestionnaire de secrets :
  - ▶ **Ansible Vault.**
  - ▶ **HashiCorp Vault.**
- ▶ Seul l'outil d'automatisation doit y accéder.

# Validation et tests : Batfish

---

## Principes clés

- ▶ Valide la config sans impacter la prod.
- ▶ Construit un modèle mathématique du réseau.
- ▶ Permet de poser des questions : « A peut-il joindre B ? ».

# Validation et tests : Pytest

---

## Principes clés

- ▶ Framework de tests classique en Python.
- ▶ Validation de scripts et configs d'automatisation.

# CI/CD pour le réseau (GitOps)

---

## Principes clés

---

- ▶ Automatiser le cycle de vie complet des changements.
- ▶ Un commit déclenche un pipeline CI/CD :
  - ▶ **CI** : vérification syntaxique, tests statiques.
  - ▶ **CD** : déploiement automatique après validation.
- ▶ Outils courants : GitLab CI/CD, Jenkins.

# Synthèse et ressources

---

- ▶ **Synthèse** : Le NetDevOps n'est pas qu'une question d'outils, c'est un changement de culture qui permet de gérer les réseaux de manière plus efficace, fiable et sécurisée.
- ▶ Jason Edelman, Scott S. Lowe, Matt Oswalt. Network Programmability and Automation: Skills for the Next-Generation Network Engineer. O'Reilly Media, 2016. ISBN : 978-1-4919-3359-9.