MÉMO TECHNIQUE

Les outils autour de Git & de la CI/CD

Écosystème DevOps

Terraform · Ansible · Puppet · Jenkins · Prometheus · Elasticsearch · Istio

Objet	Définitions brèves et rôle dans le cycle DevOps
Public	Développeurs, DevOps, profils techniques
Auteur	OSD — M. Slimani
Version	v1.0 — 07/06/2026

Situer ces outils

Ces sept outils ne font pas partie de Git, mais ils prolongent la chaîne que la formation couvre (Git → forge → CI/CD). Chacun intervient à une étape précise du cycle de vie DevOps.

Le fil conducteur : provisionner l'infrastructure → configurer les serveurs → orchestrer les déploiements → observer ce qui tourne → gérer le trafic entre microservices.

> ℹ️ Lien avec la formation Git
>
> Comme le code applicatif, le code d'infrastructure (Terraform), de configuration (Ansible/Puppet) et de pipeline (Jenkins) est versionné dans Git. Un push ou un tag déclenche ensuite la chaîne CI/CD vue en S5–S7.

1. Provisionnement — Infrastructure as Code

Terraform

Définition. Outil d'*Infrastructure as Code* : on **décrit** l'infrastructure (machines virtuelles, réseaux, bases de données, clusters) dans des fichiers déclaratifs, et Terraform la crée ou la modifie chez le fournisseur cloud (GCP, AWS, Azure…).
Dans la formation. C'est le pendant « infra » de Git : le descriptif d'infrastructure est versionné dans un dépôt, au même titre que le code. C'est typiquement Terraform qui **crée la machine** sur laquelle on installe GitLab (cf. S5) ou l'application.

2. Configuration — Configuration Management

Ansible

Définition. Outil d'automatisation et de gestion de configuration **sans agent** (il opère via SSH). On écrit des *playbooks* en YAML qui installent des paquets, configurent des services et déploient des applications.

Dans la formation. Une fois l'infra créée par Terraform, Ansible la **configure** et automatise les déploiements — c'est souvent ce qui se cache derrière l'étape « deploy » d'un pipeline CI/CD.

Puppet

Définition. Autre outil de gestion de configuration, **à base d'agents** : chaque serveur tire (*pull*) son état désiré depuis un serveur central, selon un modèle déclaratif.
Dans la formation. Même objectif qu'Ansible — garder un parc de serveurs dans un état **cohérent et reproductible** — mais avec une philosophie différente, utile pour comparer les approches **push** (Ansible) et **pull** (Puppet).

> ✓ **Ansible vs Puppet en un mot**
>
> Ansible **pousse** la configuration (sans agent, via SSH), Puppet la fait **tirer** par des agents installés sur chaque machine. Deux réponses au même besoin : l'infrastructure reproductible.

3. Orchestration — CI/CD

Jenkins

Définition. Serveur d'automatisation CI/CD historique et très répandu. Il orchestre les pipelines **build** → **test** → **deploy** définis dans un *Jenkinsfile*, et s'étend via un large catalogue de plugins.

Dans la formation. C'est une **alternative autonome** à GitLab CI et GitHub Actions déjà présentés : même rôle — déclenché par un événement Git, il automatise la chaîne — mais en outil indépendant de la forge.

4. Observabilité — monitoring & logs

Prometheus

Définition. Système de monitoring et d'alerting fondé sur des **métriques** chronologiques (*time-series*). Il collecte des indicateurs (CPU, latence, taux d'erreur…) et déclenche des alertes ; souvent couplé à Grafana pour la visualisation.

Dans la formation. Il couvre la phase « monitor » du cycle : une fois l'application déployée, c'est lui qui **surveille sa santé** et prévient en cas d'anomalie.

Elasticsearch

Définition. Moteur de **recherche et d'analyse** distribué, très utilisé pour la **centralisation des logs** (cœur de la stack ELK : Elasticsearch + Logstash + Kibana).

Dans la formation. Là où Prometheus regarde les métriques, Elasticsearch permet de **chercher et analyser les logs** applicatifs — l'autre pilier de l'observabilité.

5. Microservices — réseau & service mesh

Istio

Définition. *Service mesh* pour applications en **microservices** (typiquement sur Kubernetes). Il gère la communication entre services : **routage du trafic, sécurité (mTLS), résilience et observabilité**, sans modifier le code des services.

Dans la formation. Pertinent dès qu'on développe en microservices : Istio répond aux problèmes qui apparaissent **quand ils se multiplient** — qui parle à qui, comment sécuriser et observer ces échanges.

Récapitulatif

Outil	Catégorie	Rôle dans le cycle DevOps
Terraform	Infrastructure as Code	Créer / modifier l'infrastructure (provisionnement)
Ansible	Configuration (sans agent)	Configurer les serveurs, automatiser les déploiements
Puppet	Configuration (à agents)	Maintenir l'état désiré d'un parc de serveurs
Jenkins	Orchestration CI/CD	Automatiser build → test → deploy
Prometheus	Observabilité (métriques)	Monitorer et alerter sur la santé des services
Elasticsearch	Observabilité (logs)	Centraliser, chercher et analyser les logs
Istio	Service mesh	Gérer trafic, sécurité et résilience entre microservices

✓ La phrase à retenir

Terraform crée l'infra, **Ansible/Puppet** la configurent, **Jenkins** (ou GitLab CI / GitHub Actions) orchestre les déploiements, **Prometheus/Elasticsearch** surveillent le résultat, et **Istio** gère le trafic entre microservices.

ℹ️ Où Git intervient

Tous ces outils s'appuient sur des fichiers (HCL, YAML, Jenkinsfile, manifestes) qui vivent dans un dépôt Git. La maîtrise de Git — objet de la formation — reste donc le socle commun à tout l'écosystème DevOps.