BTS Services Informatiques aux Organisations Option SISR : Solutions d'Infrastructure, Systèmes et Réseaux

COMPTE RENDU D'ACTIVITÉ PROFESSIONNELLE

PROJET N°1

Titre du projet : Déploiement Automatisé d’une solution de Gestion de Parc (GLPI) sur infrastructure Cloud AWS

Auteur : Hugo LÉONARDI **Session :** 2026 **Contexte :** Formation BTS SIO / Projet Technique **Date de réalisation :** Novembre 2025

1. Contexte du projet

Dans le cadre de la modernisation des outils de gestion informatique, il a été décidé de mettre en place une solution de **Gestion de Parc et de Helpdesk (GLPI)**. Afin de garantir la reproductibilité, la rapidité de déploiement et la standardisation de l'infrastructure, l'installation manuelle a été écartée au profit d'une approche **Infrastructure as Code (IaC)**.

Le projet consiste à provisionner automatiquement un serveur web sécurisé hébergeant GLPI sur le fournisseur Cloud **AWS (Amazon Web Services)**.

2. Objectifs

- Mettre à disposition un service de gestion de parc informatique (GLPI 10).
- Automatiser le déploiement de l'infrastructure serveur (EC2).
- Automatiser l'installation logicielle (Apache, MariaDB, PHP, GLPI).
- Gérer les règles de sécurité réseau (Security Groups).

3. Environnement Technique

- **Fournisseur Cloud :** AWS (Région us-east-1).
- **Outil d'automatisation (IaC) :** Terraform (v5.0).
- **Système d'exploitation :** Ubuntu Server 22.04 LTS (AMI).
- **Serveur Web :** Apache2 + PHP 8.1.
- **Base de données :** MariaDB Server (Local).
- **Langage de script :** Bash (User Data AWS).
- **Type d'instance :** t3.micro (Offre AWS Free Tier).

4. Architecture de la solution

L'architecture repose sur une instance EC2 unique hébergeant l'ensemble de la stack LAMP (Linux, Apache, MariaDB, PHP).

- **Instance AWS :** `t3.micro` (Performances burstables adaptées aux environnements de test).
- **Sécurité (Firewall) :** Un "Security Group" a été configuré pour n'autoriser que les flux strictement nécessaires :
    - TCP/80 (HTTP) : Accès à l'interface web GLPI depuis Internet.

- TCP/22 (SSH) : Accès pour l'administration système.
- Tout trafic sortant est autorisé pour permettre les mises à jour (apt-get).

5. Mise en œuvre avec Terraform

Le déploiement est piloté par trois fichiers principaux :

A. Définition du Fournisseur (`provider.tf`)

Configuration du provider AWS et de la région cible. Cela permet d'interfacer Terraform avec l'API d'Amazon.

B. Gestion du Réseau et Sécurité (`network_security.tf`)

Création dynamique du groupe de sécurité (`aws_security_group`). Ce fichier définit les règles *ingress* (entrée) pour les ports 80 et 22. L'utilisation de variables dynamiques permet de récupérer l'ID du VPC par défaut automatiquement.

C. Définition de l'Instance (`main.tf`)

Ce fichier est le cœur du projet. Il effectue les actions suivantes :

1. **Recherche de l'AMI :** Utilisation d'un filtre (`data "aws_ami"`) pour trouver automatiquement la dernière version d'Ubuntu 22.04, garantissant un OS à jour à chaque déploiement.
2. **Création de l'EC2 :** Instanciation de la ressource `aws_instance` avec le type `t3.micro`.
3. **Injection du script d'installation :** Utilisation de la directive `user_data` pour exécuter le script Bash au premier démarrage de la machine.

6. Automatisation de la configuration (Scripting)

Le fichier `install_glpi.sh` prend le relais une fois la machine démarrée par AWS. Il exécute séquentiellement :

1. **Mise à jour du système :** `apt-get update & upgrade`.
2. **Installation des paquets :** Apache2, MariaDB-Server et les extensions PHP requises par GLPI (php-xml, php-ldap, php-gd, etc.).
3. **Configuration BDD :** Création automatique de la base `glpidb` et de l'utilisateur `glpi_user` via des commandes SQL en ligne de commande.
4. **Déploiement GLPI :** Téléchargement de l'archive `.tgz` depuis le dépôt officiel GitHub, extraction dans `/var/www/` et attribution des droits (`chown www-data`).
5. **VirtualHost :** Création dynamique du fichier de configuration Apache pour servir le site.

7. Vérification du fonctionnement

Une fois la commande `terraform apply` exécutée, l'infrastructure est prête en moins de 3 minutes. Terraform renvoie en sortie (`output`) l'adresse IP publique du serveur.

- **Test 1 :** Connexion SSH fonctionnelle avec la clé privée.
- **Test 2 :** Accès HTTP via navigateur. La page d'installation de GLPI s'affiche correctement.
- **Test 3 :** Connexion à la base de données réussie avec les identifiants pré-configurés dans le script.

(Ici, sur ton document Word, tu peux insérer une capture d'écran de ton terminal montrant le "Apply complete" et une capture d'écran de la page de login GLPI).

8. Compétences du référentiel validées

Ce projet permet de valider les compétences suivantes du bloc SISR :

Compétence	Justification
Gérer le patrimoine informatique	Mise en place d'un outil d'inventaire (GLPI).
Répondre aux incidents et demandes d'assistance	Déploiement d'un outil de ticketing.
Mettre à disposition des services informatiques	Création d'une instance serveur accessible aux utilisateurs.
Travailler en mode projet	Planification, utilisation de l'approche DevOps (IaC).

9. Conclusion et Perspectives

Ce projet a permis de démontrer l'efficacité de l'Infrastructure as Code. Contrairement à une installation manuelle longue et sujette aux erreurs humaines, cette solution permet de déployer un serveur GLPI opérationnel en quelques minutes.

Axes d'amélioration identifiés : Actuellement, la base de données est hébergée sur la même machine que le serveur web (Monolithique). Pour un environnement de production, il serait pertinent de :

1. Externaliser la base de données vers un service managé (AWS RDS) pour la redondance.
2. Stocker l'état Terraform (tfstate) dans un bucket S3 pour faciliter le travail collaboratif.