Mise à jour : 01-26

ALBERTUCCI Rémi

Table des matières

I Intro

GLPI est l'acronyme de **Gestion Libre de Parc Informatique**. C'est un logiciel puissant et versatile qui met à disposition divers outils pour l'administration du système : suivi et inventaire du matériel et des logiciels, gestion des utilisateurs, suivi des tickets d'incidents et demandes, planification de la maintenance et génération de rapports d'activité. Grâce à son interface web, il permet de centraliser toutes les informations et de faciliter la gestion du parc informatique pour les équipes techniques.

Dans cet exercice, nous déploierons GLPI sur une machine virtuelle Debian 13 sous VirtualBox. Nous y accéderons par SSH une fois configuré. Nous pourrons ainsi paramétrer la machine en dehors de l'interface de l'hyperviseur.

Nous configurerons également deux utilisateurs pour simuler la création et la résolution d'un ticket. Nous automatiserons la sauvegarde de la base de données, ce qui permettra de restaurer GLPI en cas de panne, de crash ou de migration.

II Préparation de la VM

Nous commencerons par préparer la machine virtuelle hébergeant GLPI.

Il s'agit ici d'une machine sous Debian Trixie (13), à laquelle nous avons attribué 4 cœurs ainsi que 4Gb de RAM. Ces attributs peuvent être réduits de moitié sans affecter les performances pour une machine de test.

II.1 Configuration de la VM

On crée une VM debian 13, 2 coeurs, 2Go de RAM, 32Go de stockage.

Suffisant pour démarrer, on pourra adapter cette configuration par la suite.

II.2 Installation Debian

II.3 Vérification de la connectivité

Un fois connecté et identifié, nous vérifions :

```
toor@glpi:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:7d:5e:be brd ff:ff:ff:ff:ff:ff
    altname enx0800277d5ebe
    inet6 2a02:842a:9980:d01:1215:7ad4:146e:e1ed/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 604793sec preferred_lft 604793sec
    inet6 fe80::c2b7:c60f:b35c:2185/64 scope link
       valid_lft forever preferred_lft forever
toor@glpi:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:7d:5e:be brd ff:ff:ff:ff:ff:ff
    altname enx0800277d5ebe
    inet 192.168.1.85/24 brd 192.168.1.255 scope global dynamic noprefixroute enp0s3
       valid_lft 86397sec preferred_lft 75597sec
    inet6 2a02:842a:9980:d01:1215:7ad4:146e:e1ed/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 604789sec preferred_lft 604789sec
    inet6 fe80::c2b7:c60f:b35c:2185/64 scope link
       valid_lft forever preferred_lft forever
```

La commande IP nous indique l’adresse IP de la machine.

La commande ping 8.8.8.8 nous indique que nous sommes bien connectés à internet.

A partir de ce point, on peut se connecter depuis notre réseau local en SSH sur la machine virtuelle en utilisant « ssh user@ip ». Cela simplifiera fortement le travail car la configuration peut être effectuée depuis n’importe quel poste et supportera les copier-coller du document.

II.4 Configuration Debian de base

Une fois connecté en ssh au profil utilisateur, nous devons lui accorder les droits sudo ainsi qu’installer sudo.

Nous allons donc effectuer les commandes suivantes :

```
su -
*mdp root*
apt update && apt install sudo -y
usermod -aG sudo <user>
systemctl reboot
```

Nous nous reconnectons ensuite par ssh, dans ce laps de temps L'IP ne devrait pas avoir changé.

Nous installons alors l'outil curl et wget, servant à télécharger des fichiers depuis Internet via la ligne de commande :

```
sudo apt install curl wget -y
```

III Installation du serveur LAMP

Un serveur LAMP est un serveur Linux qui peut faire tourner un site web dynamique. Apache sert les pages, MySQL ou MariaDB garde les données, et PHP crée le contenu des pages.

LAMP contient donc tout ce qu'il faut pour héberger un site ou une application web en ligne, comme GLPI.

On peut installer tout ce qui est nécessaire en lançant la commande suivante :

```
sudo apt install apache2 mariadb-server php php-cli php-bcmath php-mysql php-gd php-xml php-intl php-curl php-zip php-ldap php-mbstring unzip -y
```

On vérifie ensuite si tout est installé et fonctionnel :

```
sudo systemctl status apache2
sudo systemctl status mariadb
```

Cela nous renvoie :

```
toor@glpi:~$ sudo systemctl status mariadb
● mariadb.service - MariaDB 11.8.3 database server
     Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset: enabled)
     Active: active (running) since Sat 2025-10-25 20:17:14 CEST; 1min 25s ago
 Invocation: b3b982b350334b5f839076507bc7181c
       Docs: man:mariadbd(8)
             https://mariadb.com/kb/en/library/systemd/
   Main PID: 7539 (mariadbd)
     Status: "Taking your SQL requests now..."
      Tasks: 10 (limit: 30626)
     Memory: 124.2M (peak: 129M)
        CPU: 2.779s
     CGroup: /system.slice/mariadb.service
             └─7539 /usr/sbin/mariadbd
```

Le statut active nous renvoie donc que tout est fonctionnel.

IV Installation et configuration de GLPI

IV.1 Téléchargement de GLPI et pré-configuration

Comme expliqué précédemment, GLPI est un logiciel libre de gestion de parc informatique.

Pour l'installer, nous allons nous déplacer dans le dossier /var/www/html et télécharger GLPI à l'aide de la commande :

```
sudo wget https://github.com/glpi-project/glpi/releases/download/11.0.0/glpi-11.0.0.tgz
```

Un fichier .tgz est un fichier compressé, nous le décompressons avec la commande « sudo tar -xvzf glpi-11.0.0.tgz » : les options xvzf sont respectivement : extract, verbose, utilisation de gzip et enfin le fichier que nous allons décompresser.

Une fois fait, on fait en sorte qu'apache puisse exécuter ce dossier : on le rend propriétaire et on lui donne les droits d'écriture :

```
sudo chown -R www-data:www-data glpi
sudo chmod -R 755 glpi
```

IV.2 Base de données

Nous allons ensuite créer une base de données que GLPI pourra par la suite utiliser. Pour cela, nous allons rentrer dans l'application mysql précédemment téléchargée :

```
sudo mysql -u root -p
```

une fois le mot de passe root entré, on crée la base de donnée (« CREATE DATABASE »), un utilisateur (« CREATE USER ») et on lui accorde les droits sur cette base de données (« GRANT ALL PRIVILEGES ON »). La liste des commandes est :

```
CREATE DATABASE glpi;
CREATE USER 'glpiuser'@'localhost' IDENTIFIED BY 'glpi@2025';
GRANT ALL PRIVILEGES ON glpi.* TO 'glpiuser'@'localhost';
FLUSH PRIVILEGES;
EXIT;
```

Cela nous donne dans le terminal

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at https://github.com/MariaDB/
Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE glpi;
Query OK, 1 row affected (0,001 sec)

MariaDB [(none)]> CREATE USER 'glpiuser'@'localhost' IDENTIFIED BY 'glpi@2025';
Query OK, 0 rows affected (0,007 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON glpi.* TO 'glpiuser'@'localhost';
Query OK, 0 rows affected (0,006 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> EXIT;
Bye
```

IV.3 Apache

Nous allons éditer le fichier de configuration de GLPI :

```
sudo nano /etc/apache2/sites-available/glpi.conf
```

Une fois rentré dans l'éditeur, nous, le configurons de la manière suivante :

```
<VirtualHost *:80>
ServerName glpi.local
DocumentRoot /var/www/html/glpi/public
<Directory /var/www/html/glpi/public>
```

```
Require all granted
AllowOverride All
Options FollowSymLinks
</Directory>
ErrorLog ${APACHE_LOG_DIR}/glpi_error.log
CustomLog ${APACHE_LOG_DIR}/glpi_access.log combined
</VirtualHost>
```

Pour sortir de l'éditeur, on écrit d'abord le fichier avec ctrl + o et on quitte avec ctrl +x.

Nous allons donc accéder en http sur le port 80 à GLPI. Les documents relatifs à la page seront stockés dans /var/www/html/glpi.

Nous allons ensuite configurer Apache pour qu'il serve correctement GLPI. Pour cela, nous activons le site GLPI (`a2ensite glpi.conf`), nous activons le module `rewrite` pour que les URL fonctionnent correctement (`a2enmod rewrite`), nous désactivons le site par défaut pour éviter tout conflit (`a2dissite 000-default.conf`) et enfin, nous redémarrons Apache afin que tous ces changements soient pris en compte (`systemctl restart apache2`) :

```
sudo a2ensite glpi.conf
sudo a2enmod rewrite
sudo a2dissite 000-default.conf
sudo systemctl restart apache2
```

IV.4 htaccess

Pour continuer, nous devons créer le fichier `.htaccess` de GLPI afin de configurer certaines règles d'accès et de réécriture d'URL nécessaires au bon fonctionnement de l'application. Pour cela, nous utilisons la commande :

```
sudo tee /var/www/html/glpi/public/.htaccess > /dev/null <<'EOF'
```

Et lui donner la valeur suivante :

```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^ index.php [QSA,L]
</IfModule>
EOF
```

On vérifie ensuite si le site est accessible via la commande curl :

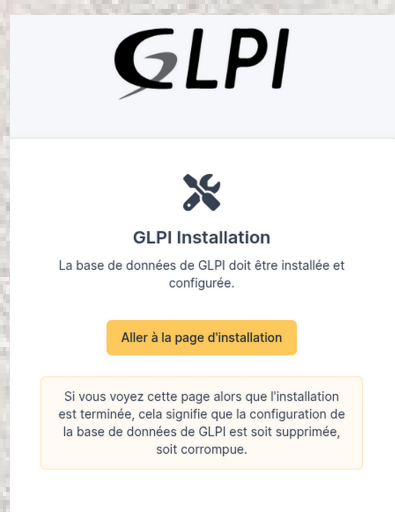
```
curl -I http://localhost
```

Cela doit nous renvoyer :

```
HTTP/1.1 200 OK
Date: Sat, 25 Oct 2025 18:47:06 GMT
Server: Apache/2.4.65 (Debian)
Set-Cookie: glpi_4c34e291bd28f45cd7e30a154a0ba44df038ebf596fda48ea37903423d63c15a4
2676212b0e1a2ca8125e490bebda69=e58b834535ca2c4307d7a3163796ef62; path=/; HttpOnly
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Cache-Control: no-cache, private
Vary: Accept-Encoding
Content-Type: text/html; charset=UTF-8
```

Nous pouvons essayer de nous connecter à l'interface web de GLPI : notre machine virtuelle Debian est configurée pour nous afficher la page d'accueil de GLPI directement en renseignant l'adresse ip de la machine dans un navigateur.

Dans la barre de recherche de notre navigateur, on recherche donc http://192.168.1.85 et on arrive sur la page suivante :



Nous suivons donc le chemin d'installation de la même manière que nous l'avons fait pour l'installation de Debian.

Nous aurons besoin des données suivantes :

serveur : localhost

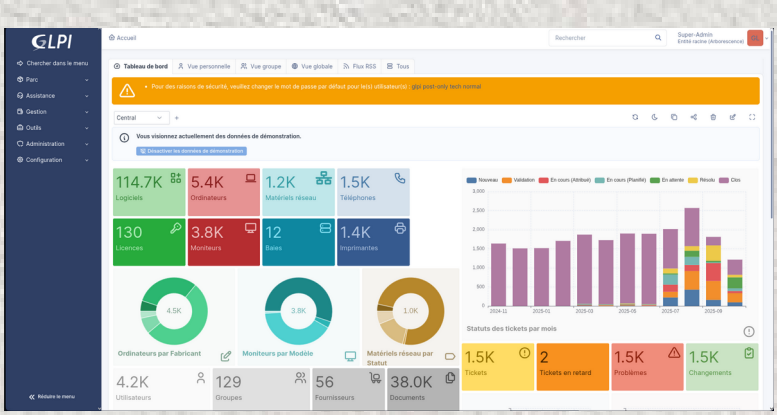
utilisateur : glpi user

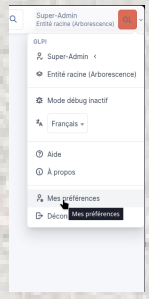
Mot de passe : glpi@2025

Base de données : glpi

Une fois cela effectué,on peut se connecter à l'interface de GLPI et accéder à la page d'accueil, directement au compte superadministrateur glpi et à l'aide du mot de passe glpi.

Ce mot de passe est changé en « toor » pour les besoins de cet exercice.





Nous supprimons le dossier d'installation pour des raisons de sécurité :

```
sudo rm -rf /var/www/html/glpi/install/install.php
```

V Simulation d'un ticket d'incident

Nous allons maintenant simuler un ticket d'incident. Un utilisateur va créer un ticket et un technicien devra y répondre.

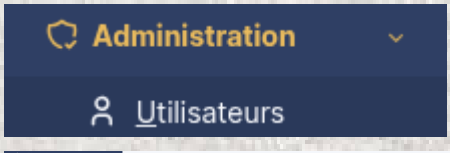
Nous devons d'abord créer des utilisateurs et leur attribuer des rôles.

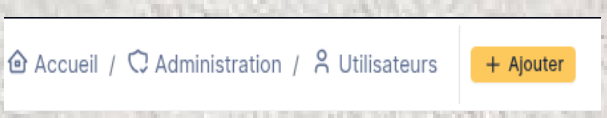
V.1 Création des profils et attribution des rôles

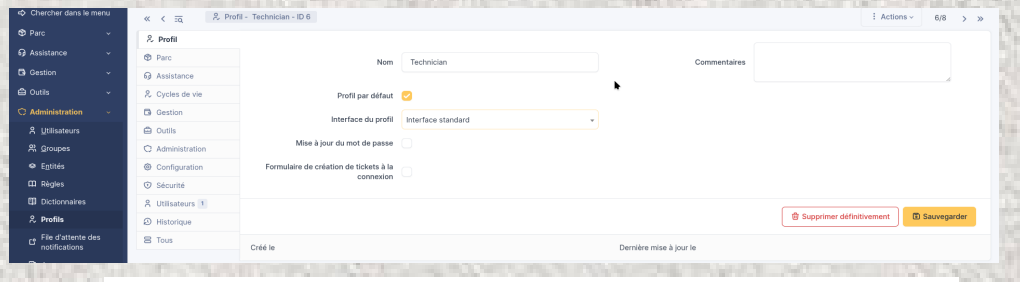
Nous allons maintenant simuler la création d'un ticket pour se familiariser avec l'interface. Nous devons d'abord créer deux utilisateurs : Bob, un technicien, et Alice, une utilisatrice.

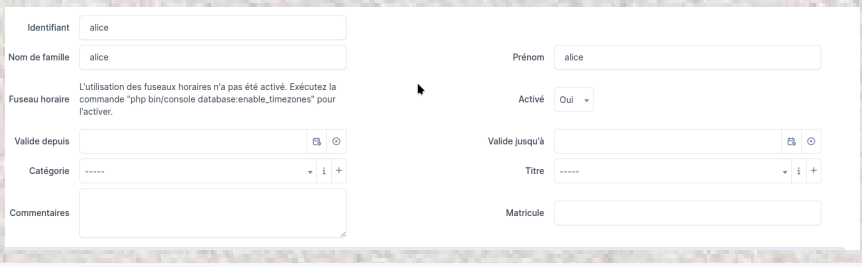
Pour leur attribuer leurs rôles, nous irons ensuite dans les onglets profil et sur chaque profil, nous ajouterons l'utilisateur en question lui accordant donc les droits en concordance avec son groupe.

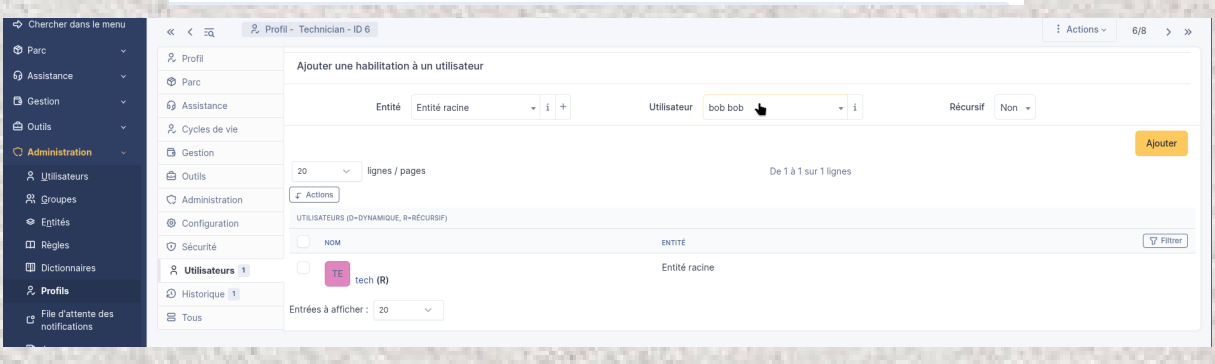
Ci dessous une représentation visuelle des étapes à effectuer. Notons que cela peut varier en fonction des mises à jours (ici GLPI v11.0).

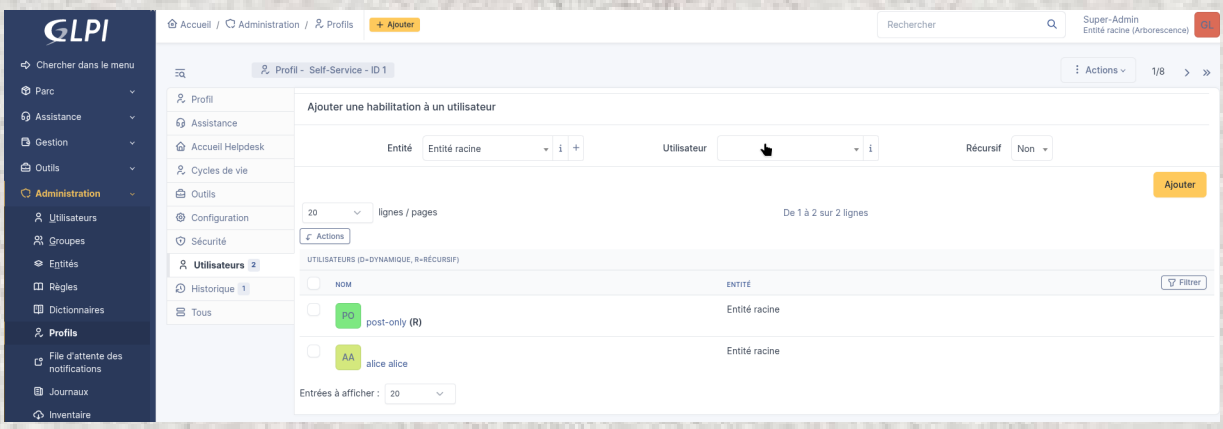








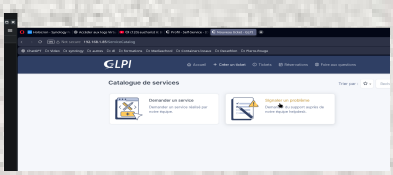


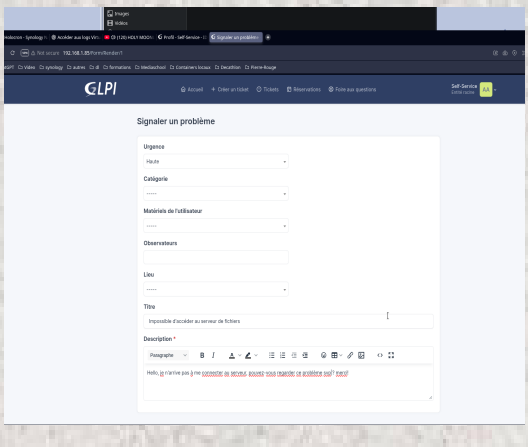


V.2 Création d'un ticket et Prise en charge par le technicien et clôture

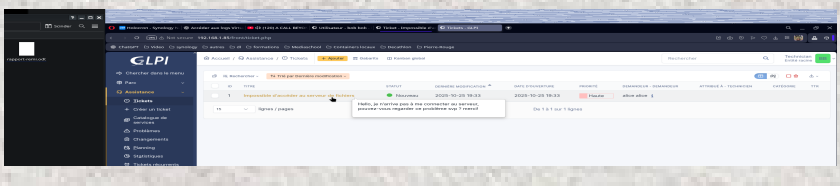
Nos utilisateurs définis, nous pouvons maintenant nous connecter à leurs comptes respectifs pour créer un ticket avec Alice, y réponde avec Bob, et le clôturer avec le compte superadministrateur.

Comme dans la section précédente, voici une illustration de ces étapes :



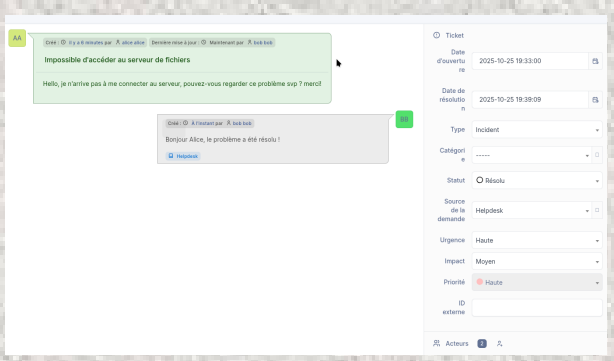


Puis en tant que Bob :

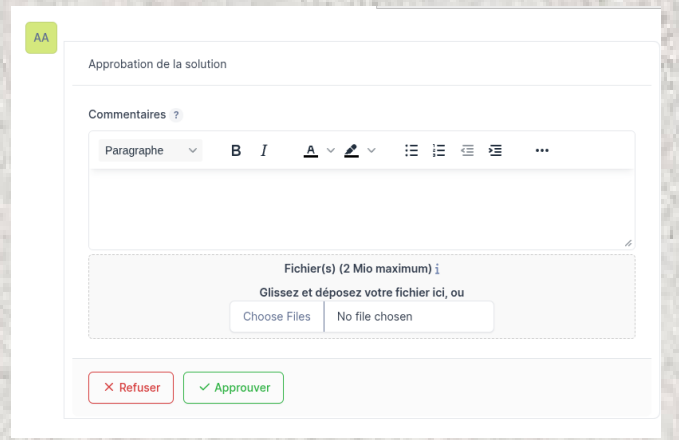


Dans l'onglet de droite, dans l'attribution, on clique sur « m'associer ». Ainsi, ce ticket nous sera attribué et nous pourrons en assurer la gestion.

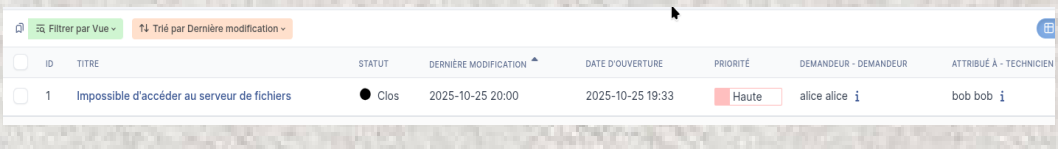
Une fois avoir pris connaissance de l'incident, on peut enquêter et tenter de le résoudre avant de répondre. Une fois la résolution effectuée, on peut répondre et passer le ticket en « résolu ».



En tant qu'Alice, on peut maintenant approuver la solution et répondre.



On peut maintenant voir que le ticket a été clos :



VI Sauvegarde automatisée

VI.1 Création de la sauvegarde

Nous allons maintenant créer une sauvegarde de la base de données de GLPI. Pour ce faire, on crée le fichier de sauvegarde :

```
sudo nano /usr/local/sbin/glpi-backup.sh
```

Et dans ce fichier :

```
#!/usr/bin/env bash
set -euo pipefail
GLPI_DIR="/var/www/html/glpi"
DB_NAME="glpi"
DB_USER="glpiuser"
DB_PASS="glpi@2025"
BACKUP_DIR="/var/backups/glpi"
RETENTION_DAYS=7
LOG_FILE="/var/log/glpi-backup.log"
DATE=$(date +%F_%H%M%S)
mkdir -p "$BACKUP_DIR"
touch "$LOG_FILE"
echo "[$(date '+%F %T')] Démarrage sauvegarde GLPI" >> "$LOG_FILE"

# Dump base de données
mysqldump -u "$DB_USER" -p"$DB_PASS" "$DB_NAME" | gzip > "$BACKUP_DIR/${DATE}_glpi.sql.gz"

# Sauvegarde fichiers GLPI
tar -czf "$BACKUP_DIR/${DATE}_glpi-files.tar.gz" -C "$GLPI_DIR" .

# Nettoyage anciens fichiers
find "$BACKUP_DIR" -type f -mtime +$RETENTION_DAYS -delete
echo "[$(date '+%F %T')] Sauvegarde terminée avec succès" >> "$LOG_FILE"
```

On donne ensuite les droits nécessaires à l'opération du script et on crée le dossier de sauvegarde :

```
sudo chmod +x /usr/local/sbin/glpi-backup.sh
sudo mkdir -p /var/backups/glpi
sudo chown root:root /usr/local/sbin/glpi-backup.sh /var/backups/glpi
```

VI.2 Planification via cron

Nous allons créer une tâche de planification pour automatiser la sauvegarde. Elle aura lieu tous les jours à 2h, à une heure où le serveur est censé avoir peu de charge. Pour créer une tâche de sauvegarde, on crée dans le dossier /etc/cron.d un dossier glpi contenant le

```
sudo nano /etc/cron.d/glpi-backup
```

qui contiendra la commande cron suivante :

```
0 2 * * * root /usr/local/sbin/glpi-backup.sh
```

On devrait obtenir :

```
toor@glpi:/etc/cron.d$ ls
e2scrub_all  glpi-backup  php
```

Testons voir si notre sauvegarde marche.

VI.3 Résultats du test

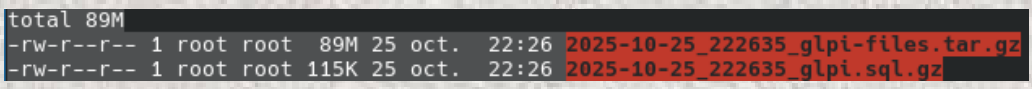
Nous allons lancer la sauvegarde grâce à la commande :

```
sudo /usr/local/sbin/glpi-backup.sh
```

Et nous vérifions le résultat avec :

```
ls -lh /var/backups/glpi
```

Nous obtenons ceci :



Notre commande a fonctionné, nous pouvons voir nos fichiers sauvegardés.

Notons que cette commande ne supprime pas les anciens fichiers, un autre script pour automatiser la suppression peut être envisageable.