VM

Contents

1 Cours : Installation d'Ubuntu en Machine Virtuelle (VMware Workstation)

Introduction

L'installation d'Ubuntu dans une machine virtuelle est une étape fondamentale pour apprendre à manipuler un système Linux sans impacter directement la machine hôte. Comprendre le processus permet de mieux diagnostiquer les problèmes courants (réseau, disque, clavier, langue, etc.) et d'exploiter correctement l'environnement virtuel.

1. Concepts de base

- **Hyperviseur de type 2** : VMware Workstation est un hyperviseur fonctionnant sur un système d'exploitation hôte (Windows, Linux, macOS).
- **ISO Ubuntu** : image amorçable du système permettant soit d'utiliser Ubuntu en mode live, soit de l'installer sur disque.
- **Live session** : Ubuntu fonctionne uniquement en RAM, aucune donnée persistante après redémarrage.
- **VM Disk (VMDK)** : disque virtuel de la machine où sera installée Ubuntu de manière persistante.

2 Hyperviseur : Définition et Rôle

Définition

Un **hyperviseur** est un logiciel ou un micro-noyau qui permet de créer et de gérer des *machines virtuelles* (VM). Il agit comme une couche intermédiaire entre le matériel physique (processeur, mémoire, disque, cartes réseau) et les systèmes d'exploitation invités. Chaque machine virtuelle croit disposer de ses propres ressources matérielles, alors qu'en réalité, celles-ci sont partagées et orchestrées par l'hyperviseur.

Types d'hyperviseurs

On distingue deux grandes catégories :

- **Hyperviseur de type 1 (bare-metal)** : installé directement sur le matériel, sans système d'exploitation hôte. Exemples : VMware ESXi, Microsoft Hyper-V Server, Xen. Avantage : performances élevées, adapté aux environnements de production.
- **Hyperviseur de type 2 (hosted)** : fonctionne au-dessus d'un système d'exploitation existant. Exemples : VMware Workstation, VirtualBox, Parallels Desktop. Avantage : simplicité d'utilisation pour les environnements de test et développement.

Rôle principal

- **Isolation** : chaque VM est isolée des autres, ce qui permet de tester des systèmes ou applications sans affecter l'hôte.
- **Partage des ressources** : l'hyperviseur distribue la mémoire, les processeurs virtuels, l'espace disque et le réseau entre les VM.
- **Flexibilité** : possibilité d'exécuter plusieurs systèmes d'exploitation différents sur la même machine physique.
- **Gestion simplifiée** : création, duplication, sauvegarde et restauration de VM facilitée.

Exemple concret

Dans le cas d'**Ubuntu installé sur VMware Workstation** :

- L'ordinateur physique (Windows/Linux/macOS) est l'*hôte*.
- VMware Workstation est l'hyperviseur de type 2.
- Ubuntu est le système d'exploitation *invité*, exécuté dans une machine virtuelle.

Conclusion

L'hyperviseur est le cœur de la virtualisation : il permet d'exécuter plusieurs systèmes indépendants sur une même machine physique. Comprendre sa fonction et ses types est indispensable avant de procéder à l'installation d'Ubuntu ou d'un autre OS dans une machine virtuelle.

2. Création d'une VM dans VMware

1. Créer une nouvelle machine virtuelle.
2. Sélectionner l'ISO d'Ubuntu comme média d'installation.
3. Choisir le type de firmware : BIOS ou UEFI.
4. Allouer des ressources :
   - **RAM** : au minimum 2 Go, recommandé 4 Go ou plus.
   - **CPU** : au moins 2 vCPU.
   - **Disque** : 20 Go minimum.
5. Configurer la carte réseau : Bridged ou NAT.

3. Modes Réseau en VM

- **NAT** : la VM utilise l'adresse IP de l'hôte et passe par la traduction d'adresses. Simple et fiable.
- **Bridged** : la VM apparaît comme une machine à part entière sur le réseau local et reçoit une IP du DHCP du réseau physique.
- **Host-only** : réseau isolé entre l'hôte et la VM, pas d'accès Internet.

Important : dans une VM, Ubuntu ne voit jamais directement le Wi-Fi de l'hôte. VMware lui présente toujours une carte *Ethernet virtuelle* (souvent `ens33`). Le Wi-Fi est géré en transparence par l'hyperviseur.

4. Lancement de l'ISO et choix d'installation

1. Démarrer la VM.
2. L'ISO propose deux options :
   - **Try Ubuntu (Live Session)** : test en mémoire vive, sans installation persistante.
   - **Install Ubuntu** : lance l'assistant d'installation.
3. Dans l'assistant :
   - Choisir la langue du système.
   - Choisir le clavier (souvent passer de QWERTY à AZERTY).
   - Configurer le réseau : si aucune carte Wi-Fi n'apparaît, continuer avec l'interface Ethernet (ens33).
   - Partitionner le disque virtuel : effacer le disque (VMDK) et installer Ubuntu dessus.
   - Créer un utilisateur, mot de passe et définir le fuseau horaire.

5. Gestion du clavier et des locales

- Vérifier la configuration clavier : `sudo dpkg-reconfigure keyboard-configuration`.
- Vérifier la langue système : `locale`, `localectl status`.
- Installer un pack de langue : `sudo apt install language-pack-fr`.
- Changer la locale par défaut : `sudo update-locale LANG=fr_FR.UTF-8`.

6. Netplan et configuration réseau

Ubuntu (depuis 18.04) utilise **Netplan** pour gérer le réseau :

```
sudo nano /etc/netplan/00-installer-config.yaml
```

Exemple avec DHCP :

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens33:
      dhcp4: true
```

Appliquer la configuration :

```
sudo netplan apply
```

7. Problèmes fréquents et solutions

- **Pas de réseau détecté pendant l'installation** : normal, ignorer l'étape Wi-Fi et utiliser l'Ethernet virtuel.
- **Toujours retour à l'écran "Install Ubuntu" après reboot** : retirer l'ISO du lecteur virtuel dans VMware.
- **Clavier QWERTY au lieu d'AZERTY** : reconfigurer via `dpkg-reconfigure keyboard-configuration`.
- **Pas d'IP** : vérifier `ip a`, éditer Netplan et activer DHCP sur `ens33`.

Conclusion

L'installation d'Ubuntu dans une VM nécessite de comprendre la distinction entre réseau hôte, carte virtuelle et live session. Les erreurs classiques (Wi-Fi non détecté, retour en mode live, clavier incorrect) sont dues au fonctionnement normal d'une VM. Avec une bonne maîtrise de VMware, Netplan et la configuration des locales, on obtient un système Ubuntu stable et fonctionnel dans un environnement virtualisé.

3 Cours : Installation d'Ubuntu en VM et Validation Post-Installation

Introduction

Après l'installation d'Ubuntu dans une machine virtuelle, il est essentiel de valider le bon fonctionnement du système. Cette validation se fait à travers un ensemble de commandes de base qui permettent de vérifier la connectivité réseau, la configuration des interfaces, la gestion des paquets, et l'environnement linguistique du système. Chaque commande a une utilité bien précise, que nous allons détailler.

1. Vérification du réseau

1. **Afficher les interfaces réseau :**

   ```
   ip a
   ```

   Montre toutes les interfaces disponibles (ex. `lo` pour loopback, `ens33` pour la carte VMware). Permet de vérifier si une adresse IP a été attribuée.

2. **Tester la connectivité IP :**

   ```
   ping -c 4 8.8.8.8
   ```

   Envoie 4 paquets ICMP vers le serveur DNS de Google. Vérifie que la machine peut sortir sur Internet en niveau IP.

3. **Tester la résolution DNS :**

   ```
   ping -c 4 google.com
   ```

   Permet de vérifier que le serveur DNS fonctionne et que la résolution de noms est correcte.

2. Gestion du réseau avec Netplan

1. **Vérifier les fichiers de configuration :**

   ```
   ls /etc/netplan/
   ```

   Affiche les fichiers YAML utilisés par Netplan.

2. **Modifier la configuration réseau :**

   ```
   sudo nano /etc/netplan/00-installer-config.yaml
   ```

   Permet d'activer le DHCP ou de fixer une adresse statique.

3. **Appliquer la configuration :**

   ```
   sudo netplan apply
   ```

   Recharge la configuration réseau sans redémarrer.

3. Gestion des paquets et mises à jour

1. **Mettre à jour la liste des paquets :**

   ```
   sudo apt update
   ```

   Télécharge la liste des paquets disponibles depuis les dépôts configurés.

2. **Mettre à jour les paquets installés :**

   ```
   sudo apt upgrade -y
   ```

   Installe les dernières versions des paquets présents sur le système.

3. **Installer un paquet de test (ex: curl) :**

   ```
   sudo apt install curl
   ```

   Vérifie que la gestion des paquets fonctionne correctement.

4. Vérification des locales et du clavier

1. **Vérifier la langue système :**

   ```
   locale
   ```

   Affiche les variables de langue en cours (ex. `LANG=fr_FR.UTF-8`).

2. **Installer le pack de langue français :**

   ```
   sudo apt install language-pack-fr
   ```

   Ajoute les traductions et paramètres linguistiques.

3. **Changer la locale par défaut :**

   ```
   sudo update-locale LANG=fr_FR.UTF-8
   ```

   Met le français comme langue par défaut au redémarrage.

4. **Configurer le clavier :**

   ```
   sudo dpkg-reconfigure keyboard-configuration
   ```

   Permet de choisir un clavier AZERTY au lieu du QWERTY.

5. Vérification du disque et du système

1. **Lister les disques et partitions :**

   ```
   lsblk
   ```

   Affiche les périphériques de stockage (`sda`, `sr0`, etc.).

2. **Vérifier l'espace disque utilisé :**

   ```
   df -h
   ```

   Montre l'utilisation du disque avec des tailles lisibles (Go).

3. **Vérifier la mémoire RAM :**

   ```
   free -h
   ```

   Permet de voir l'utilisation de la mémoire.

Conclusion

Ces commandes constituent la base de la validation post-installation d'Ubuntu dans une machine virtuelle. Elles permettent de vérifier le réseau, les locales, la gestion des paquets et les ressources système. Maîtriser ces notions est indispensable pour comprendre le fonctionnement d'Ubuntu et diagnostiquer rapidement tout problème après une installation en VM.

4 Cours : Déploiement de GLPI avec Docker sur Windows et Supervision d'une VM Linux

Introduction

GLPI est une solution libre de gestion de parc informatique et de supervision. Sur un hôte Windows, il est possible de déployer GLPI facilement en utilisant Docker, ce qui évite d'installer manuellement Apache, PHP et MariaDB.

1. Préparation de l'environnement

1. Télécharger et installer **Docker Desktop for Windows** depuis `https://www.docker.com/products/docker-desktop/`.
2. Vérifier que Docker fonctionne :

   ```
   docker --version
   ```

3. Créer un dossier de projet, par exemple :

   ```
   mkdir C:\glpi-docker
   cd C:\glpi-docker
   ```

2. Création du fichier Docker Compose

Créer un fichier `docker-compose.yml` dans le dossier `C:\glpi-docker` :

```
version: '3.3'
services:
  glpi:
    image: diouxx/glpi
    container_name: glpi
    ports:
      - "8080:80"
    environment:
      - TZ=Europe/Paris
    volumes:
      - ./glpi:/var/www/html/glpi

  db:
    image: mariadb:10.5
    container_name: glpi-db
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rootpass
      MYSQL_DATABASE: glpi
      MYSQL_USER: glpi
      MYSQL_PASSWORD: glpipass
    volumes:
      - ./db:/var/lib/mysql
```

3. Démarrage des conteneurs

1. Démarrer GLPI et MariaDB :

   ```
   docker-compose up -d
   ```

2. Vérifier l'état des conteneurs :

   ```
   docker ps
   ```

4. Accéder à GLPI

- Ouvrir le navigateur et accéder à : `http://localhost:8080`
- Suivre l'assistant d'installation :
  - Choisir la langue.
  - Entrer les informations de base de données :
    - Hôte : `db`
    - Base : `glpi`
    - Utilisateur : `glpi`
    - Mot de passe : `glpipass`

5. Installer l'agent GLPI sur la VM Linux

Sur la machine virtuelle (Ubuntu/Debian) supervisée :

```
wget https://github.com/glpi-project/glpi-agent/releases/download/1.9/glpi-agent_1.9-1_all.deb
sudo dpkg -i glpi-agent_1.9-1_all.deb
```

Configurer l'agent pour pointer vers l'hôte Windows :

```
sudo nano /etc/glpi-agent/conf.d/server.conf
```

Ajouter la ligne suivante (remplacer IP_HOTE par l'IP réelle de l'hôte Windows) :

```
% server = http://IP_HOTE:8080/glpi
```

Puis redémarrer le service agent :

```
sudo systemctl restart glpi-agent
```

6. Résultat attendu

- L'hôte Windows exécute GLPI via Docker.
- La VM Linux envoie automatiquement ses informations à GLPI grâce à l'agent.
- Depuis l'interface web GLPI, il est possible de visualiser les caractéristiques matérielles, l'OS, le réseau et les logiciels installés sur la VM.

Conclusion

Le déploiement de GLPI avec Docker simplifie grandement l'installation sur un hôte Windows. Une fois installé, GLPI peut centraliser la supervision et la gestion de multiples machines virtuelles ou physiques grâce aux agents installés sur chaque système supervisé. Cette approche est évolutive, car il est possible d'ajouter de nouvelles VMs ou serveurs en configurant simplement leur agent GLPI pour pointer vers le serveur central.

5 Cours : Commandes `sh` essentielles pour le déploiement de GLPI avec Docker

Introduction

Lorsque l'on déploie GLPI avec Docker, il est nécessaire de maîtriser certaines commandes Unix (sh/bash) pour installer des paquets, gérer les services, configurer les agents et vérifier l'état du système. Ce cours présente les commandes les plus importantes dans ce contexte et leur utilité.

1. Commandes de gestion des paquets (APT)

- **Mettre à jour la liste des paquets disponibles** :

  ```
  sudo apt update
  ```

  Cette commande interroge les dépôts configurés et récupère les dernières informations sur les paquets.

- **Mettre à jour les paquets installés** :

  ```
  sudo apt upgrade -y
  ```

  Télécharge et installe les nouvelles versions des logiciels présents.

- **Installer un paquet spécifique (exemple : curl)** :

  ```
  sudo apt install curl
  ```

  Permet d'ajouter de nouveaux logiciels ou utilitaires.

2. Commandes réseau

- **Vérifier les interfaces réseau** :

  ```
  ip a
  ```

  Affiche les interfaces disponibles (ex. ens33) et leurs adresses IP.

- **Tester la connectivité IP** :

  ```
  ping -c 4 8.8.8.8
  ```

  Vérifie que la VM peut atteindre Internet au niveau IP.

- **Tester la résolution DNS** :

  ```
  ping -c 4 google.com
  ```

  Vérifie que la résolution de noms fonctionne correctement.

3. Commandes de gestion des services

- **Redémarrer un service (exemple : glpi-agent)** :

  ```
  sudo systemctl restart glpi-agent
  ```

  Recharge la configuration d'un service sans redémarrage complet.

- **Vérifier le statut d'un service** :

  ```
  sudo systemctl status glpi-agent
  ```

  Affiche si le service est actif et ses éventuelles erreurs.

- **Activer un service au démarrage** :

  ```
  sudo systemctl enable glpi-agent
  ```

  Assure que l'agent GLPI démarre automatiquement à chaque boot.

4. Commandes liées aux fichiers et configuration

- **Éditer un fichier de configuration** :

  ```
  sudo nano /etc/glpi-agent/conf.d/server.conf
  ```

  Nano est un éditeur de texte simple pour modifier les fichiers système.

- **Lister les fichiers et répertoires** :

  ```
  ls -l /etc/glpi-agent/conf.d/
  ```

  Affiche les fichiers présents avec leurs permissions.

- **Afficher le contenu d'un fichier** :

  ```
  cat /etc/glpi-agent/conf.d/server.conf
  ```

  Utile pour vérifier une configuration sans l'éditer.

5. Commandes Docker importantes

- **Lancer les conteneurs en arrière-plan** :

  ```
  docker-compose up -d
  ```

- **Lister les conteneurs actifs** :

  ```
  docker ps
  ```

- **Arrêter les conteneurs** :

  ```
  docker-compose down
  ```

- **Vérifier les logs d'un conteneur (exemple : glpi)** :

  ```
  docker logs glpi
  ```

6. Commandes système utiles

- **Vérifier l'espace disque** :

  ```
  df -h
  ```

- **Vérifier l'utilisation mémoire** :

  ```
  free -h
  ```

- **Connaître l'OS et la version** :

  ```
  lsb_release -a
  ```