Mémo Complet : VM Debian, Docker & Virtualisation

1. INSTALLATION VM DEBIAN DE ZÉRO

Préparation

- Télécharger l'ISO Debian depuis debian.org
- Créer une VM dans VirtualBox/VMware (min 2 CPU, 2-4 GB RAM, 20 GB disque)
- Monter l'ISO en lecteur optique

Étapes installation

1. Booter sur l'ISO
2. Sélectionner langue et région
3. Partitionnement : utiliser partitionnement assisté (simple au début)
4. Sélectionner miroir Debian (ex: ftp.fr.debian.org)
5. Créer utilisateur non-root
6. Sélectionner environment : décocher "Debian desktop" pour un serveur minimal
7. Installer GRUB sur disque principal
8. Redémarrer et se connecter

Vérification post-installation

```bash
uname -a                # Vérifier kernel Linux
cat /etc/debian_version     # Version Debian
ip addr                 # Vérifier IP
```

---

2. INSTALLATION DES PAQUETS NÉCESSAIRES

Mettre à jour le système

```bash
sudo apt update          # Récupérer liste des paquets
sudo apt upgrade          # Mettre à jour les paquets
sudo apt full-upgrade      # Mise à jour complète (si dépendances majeures)
sudo apt autoremove        # Nettoyer paquets inutiles
```

Paquets essentiels à installer

```bash
sudo apt install -y vim curl wget git htop net-tools openssh-server build-essential
```

Détail :

- `vim` : éditeur de texte
- `curl wget` : téléchargement fichiers
- `git` : contrôle de version
- `htop` : monitoring CPU/RAM (top amélioré)
- `net-tools` : contient netstat, ifconfig (ancien mais pratique)
- `openssh-server` : accès SSH à distance
- `build-essential` : compilateurs et outils de développement

Installer sudo si nécessaire

```bash
su - root                # Passer en root
apt install sudo         # Installer sudo
usermod -aG sudo username    # Ajouter user au groupe sudo
```

---

3. COMMANDES LINUX ESSENTIELLES

VIM - Éditeur de texte

```bash
```

```
vim /path/to/file          # Ouvrir/créer fichier

# Mode commande (défaut)
i                          # Passer en mode insertion (avant curseur)
a                          # Passer en mode insertion (après curseur)
o                          # Nouvelle ligne et mode insertion
Esc                        # Revenir en mode commande

# En mode commande
:w                         # Sauvegarder
:q                         # Quitter
:wq                        # Sauvegarder et quitter
:q!                        # Quitter sans sauvegarder
dd                         # Supprimer ligne entière
yy                         # Copier ligne
p                          # Coller après
P                          # Coller avant
/mot                       # Chercher
n                          # Prochain résultat
N                          # Résultat précédent
:1,$s/old/new/g            # Remplacer old par new partout
```

NETSTAT - Afficher connexions réseau

```bash
netstat -tuln              # Afficher tous les ports écoute (TCP/UDP)
                           # -t: TCP, -u: UDP, -l: listening, -n: numérique
netstat -tuln | grep :80   # Voir si port 80 utilisé
netstat -an                # Toutes connexions et sockets
netstat -p                 # Afficher PID/programme
ss -tuln                   # Alternative moderne à netstat
```

SYSTEMCTL - Gestion services

```bash
sudo systemctl start nginx          # Démarrer service
sudo systemctl stop nginx           # Arrêter service
sudo systemctl restart nginx        # Redémarrer
sudo systemctl reload nginx         # Recharger config sans interruption
sudo systemctl enable nginx         # Activer au boot
sudo systemctl disable nginx        # Désactiver au boot
sudo systemctl status nginx         # Voir statut
sudo systemctl list-units --type=service    # Tous les services
```

Autres commandes utiles

```bash
sudo systemctl daemon-reload        # Recharger fichiers systemd après modifs
journalctl -u nginx                 # Voir logs du service nginx
journalctl -u nginx -f              # Logs en temps réel (follow)
journalctl -xvn 50                  # Derniers 50 logs détaillés
ps aux | grep nginx                 # Chercher processus nginx
kill PID                            # Tuer processus
kill -9 PID                         # Forcer arrêt
```

---

4. INSTALLATION SERVEUR WEB LINUX

Nginx (léger et moderne)

```bash
sudo apt install -y nginx

sudo systemctl start nginx
sudo systemctl enable nginx
sudo systemctl status nginx

# Vérifier
netstat -tuln | grep :80
curl localhost          # Devrait afficher page HTML par défaut
```

Apache2 (plus lourd, très utilisé)

```bash
```

```bash
sudo apt install -y apache2

sudo systemctl start apache2
sudo systemctl enable apache2
sudo systemctl status apache2

# Activer certains modules
sudo a2enmod rewrite       # Mod rewrite pour URL rewriting
sudo a2enmod ssl           # HTTPS
sudo systemctl reload apache2

# Sites disponibles vs sites actifs
ls /etc/apache2/sites-available/
ls /etc/apache2/sites-enabled/
a2ensite nom-site          # Activer site
a2dissite nom-site         # Désactiver site
```

Configuration Nginx

```bash
# Fichier principal
sudo vim /etc/nginx/nginx.conf

# Sites disponibles
sudo vim /etc/nginx/sites-available/default

# Test configuration
sudo nginx -t              # Tester syntaxe
sudo systemctl reload nginx # Recharger
```

Configuration simple Nginx

```nginx
server {
    listen 80;
    server_name localhost;

    location / {
        root /var/www/html;
        index index.html;
    }
}
```

Vérifier serveur web

```bash
curl http://localhost
curl http://192.168.1.100    # Via IP locale
netstat -tuln | grep :80     # Port 80 en écoute
sudo tail -f /var/log/nginx/access.log   # Logs temps réel
```

---

5. DOCKER - INSTALLATION ET NGINX

Installation Docker

```bash
# Supprimer anciennes versions (si nécessaire)
sudo apt-get remove docker docker-engine docker.io containerd runc

# Installer Docker
sudo apt-get install -y docker.io docker-compose

# Ajouter user au groupe docker (éviter sudo)
sudo usermod -aG docker $USER
newgrp docker              # Appliquer groupe immédiatement

# Vérifier installation
docker --version
docker run hello-world
```

Image Nginx avec Docker

Méthode 1 : Utiliser image existante

```bash
```

```
docker pull nginx:latest

docker run -d \
  --name webserver \
  -p 8080:80 \
  -v /var/www/html:/usr/share/nginx/html \
  nginx:latest

# -d : détaché (fond)
# --name : nom conteneur
# -p : port mapping (hôte:conteneur)
# -v : volume (monter dossier)
```

Méthode 2 : Avec docker-compose (plus pratique)

Créer `docker-compose.yml` :

```yaml
version: '3'

services:
  nginx:
    image: nginx:latest
    container_name: my_nginx
    ports:
      - "8080:80"
      - "443:443"
    volumes:
      - ./html:/usr/share/nginx/html
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
    environment:
      - TZ=Europe/Paris
    restart: unless-stopped

  # Autre service exemple
  # app:
  #   image: myapp:latest
  #   ports:
  #     - "3000:3000"
```

Commandes Docker utiles

```bash
```

```
docker ps                # Conteneurs actifs
docker ps -a             # Tous conteneurs
docker images            # Images disponibles
docker logs webserver    # Logs du conteneur
docker logs -f webserver # Logs temps réel
docker exec -it webserver bash  # Accéder au shell
docker stop webserver    # Arrêter
docker start webserver   # Redémarrer
docker rm webserver      # Supprimer conteneur
docker rmi nginx         # Supprimer image
docker inspect webserver # Infos détaillées
```

Docker-compose

```
bash

docker-compose up -d     # Lancer en détaché
docker-compose down      # Arrêter et supprimer
docker-compose logs -f   # Logs temps réel
docker-compose ps        # État des services
docker-compose restart   # Redémarrer
```

Dockerfile personnalisé (exemple)

```
dockerfile

FROM nginx:latest

COPY html/ /usr/share/nginx/html/
COPY nginx.conf /etc/nginx/nginx.conf

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

Construire : `docker build -t my-nginx:1.0 .`

---

6. VIRTUALISATION - VIRTUALBOX & VMWARE

VirtualBox - Configuration réseau

Paramètres réseau VM :

1. **NAT (par défaut)**
   - VM accède internet via hôte

- Isolation VM du reste du réseau
- Pour redirection port : Paramètres > Réseau > Avancé > Redirection
- Exemple : port 2222 hôte → 22 invité (SSH)

2. **Réseau par pont (Bridged)**
   - VM connectée directement au réseau physique
   - Obtient IP du routeur (accessible depuis LAN)
   - Meilleur pour tests réseau
3. **Réseau interne (Internal)**
   - VM communique uniquement entre VMs
   - Isolation complète
4. **Carte réseau seule hôte (Host-only)**
   - Réseau privé VM + hôte uniquement
   - Configuration avancée pour lab

Configuration redirection port NAT :

```
Clic droit VM > Paramètres > Réseau
Sélectionner adaptateur NAT
Cliquer "Avancé" > "Redirection de port"

Ajouter :
Nom: SSH
Protocole: TCP
IP hôte: 127.0.0.1
Port hôte: 2222
IP invité: 10.0.2.15 (ou vide)
Port invité: 22

Connexion SSH: ssh -p 2222 user@localhost
```

VirtualBox - Commandes CLI

```bash
VBoxManage list vms            # Lister VMs
VBoxManage startvm "MonVM"     # Démarrer VM
VBoxManage controlvm "MonVM" poweroff   # Arrêter
VBoxManage modifyvm "MonVM" --memory 2048   # Modifier RAM
VBoxManage modifyvm "MonVM" --cpus 2      # Modifier CPU
```

VMware - Configuration réseau

Adaptateurs réseau :

1. **Bridged** : VM comme machine physique du réseau
2. **NAT** : Isolation, redirection port possible
3. **Host-only** : VM + hôte uniquement

Redirection port VMware NAT :

```
Édition > Préférences > Réseau > NAT (VMnet8)
Cliquer "Paramètres NAT"

Onglet "Redirection de port"
Ajouter :
Externe: 2222
Interne: 22
Type: TCP
```

Vérifier connectivité

```bash
# Depuis hôte
ping 192.168.x.x          # IP VM (bridged)
ssh -p 2222 user@localhost   # SSH via redirection (NAT)

# Depuis VM
ip addr                   # Voir adresse IP
ping 8.8.8.8              # Accès internet
ping 192.168.1.1          # Gateway
tracert google.com        # Routage
```

Configuration IP statique VM (Debian)

```bash
```

```
# Éditer fichier réseau
sudo vim /etc/network/interfaces

# Ou (méthode moderne avec netplan si Ubuntu)
sudo vim /etc/netplan/00-installer-config.yaml

# Exemple interfaces
auto eth0
iface eth0 inet static
    address 192.168.1.100
    netmask 255.255.255.0
    gateway 192.168.1.1
    dns-nameservers 8.8.8.8 8.8.4.4

sudo systemctl restart networking
```

---

7. CHECKLIST TP DEMAIN

- [ ] Installation VM Debian complète
- [ ] `sudo apt update && sudo apt upgrade`
- [ ] Installer paquets : `vim`, `curl`, `openssh-server`, `nginx`
- [ ] Tester vim : créer/éditer fichier
- [ ] Vérifier `netstat -tuln` port 80
- [ ] Service nginx en marche : `systemctl status nginx`
- [ ] Redirection port VirtualBox/VMware configurée
- [ ] Accès SSH depuis hôte (via redirection)
- [ ] Docker installé : `docker --version`
- [ ] Conteneur nginx lancé : `docker run -d -p 8080:80 nginx`
- [ ] Accès localhost:8080 fonctionnel
- [ ] Docker-compose fonctionnel
- [ ] Logs vérifiés : `journalctl`, `docker logs`

---

8. COMMANDES DE DIAGNOSTIC RAPIDES

```bash
```

```
# Réseau
ip addr                    # Adresses IP
ip route                   # Routage
ping 8.8.8.8               # Test internet
ssh -p 2222 user@localhost # SSH redirection
curl http://localhost      # Tester serveur web

# Systèmes
uname -a                   # Info système
df -h                      # Espace disque
free -h                    # RAM libre
ps aux | grep nginx        # Processus

# Services
sudo systemctl list-units --type=service
journalctl -xvn 20         # Derniers logs

# Docker
docker ps -a               # Tous conteneurs
docker logs nomdunoir      # Logs conteneur
docker inspect nomdunoir   # Infos détaillées

# Fichiers importants
/etc/nginx/nginx.conf      # Config nginx
/var/www/html/             # Dossier web par défaut
/var/log/nginx/            # Logs nginx
/etc/network/interfaces    # Config réseau (Debian)
/etc/docker/daemon.json    # Config Docker
```

BON COURAGE POUR TON TP ! 💪