




**REPOBLIKAN'I MADAGASIKARA**

Fitiavana – Tanindrazana – Fandrosoana

**MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE**

**UNIVERSITE D'ANTSIRANANA**

**ECOLE SUPERIEURE POLYTECHNIQUE**

**MENTION : STIC**

**GRADE : MASTER**

## MINI PROJET - INFORMATIQUE

# Installation et Configuration Automatisée d'un Serveur DHCP et Firewall sur un Serveur Debian avec VirtualBox et Docker

**Réalisé par :**

- HENINTSOA Vinciane
- RAMAMONJISON Michel Terigue Gerioux
- RAZAFIMANDIMBY Hery Nirina Cecilia
- VERONIRINA Kimberly Felix

**Enseignant :** Mr RANDRIAMASINORO Njakarison Menja

Année universitaire : 2024 – 2025

# 1. Introduction

L'objectif de ce projet est de déployer un serveur **DHCP (Dynamic Host Configuration Protocol)**, chargé d'attribuer automatiquement des adresses IP aux clients d'un réseau, ainsi qu'un pare-feu configuré avec `iptables` pour sécuriser les connexions entrantes. Le tout sera mis en place sur un système Debian installé dans une machine virtuelle via `VirtualBox`. L'installation et la configuration de ces services seront automatisées à l'aide d'un script Bash. Enfin, une image Docker du système configuré sera créée pour faciliter son déploiement dans d'autres environnements.

# 2. Prérequis et outils utilisés

- Système d'exploitation : Debian 12
- Docker et ses dépendances
- iptables / iptables-persistent
- isc-dhcp-server
- Droits administrateur (root)

# 3. Description du script automatisé

Le script Bash `setup-dhcp-firewall.sh` permet d'automatiser l'ensemble du processus de création d'un serveur DHCP sécurisé dans un conteneur Docker.

### Étape 1 : Collecte des paramètres réseau

L'utilisateur saisit les informations nécessaires à la configuration du DHCP :
- Adresse réseau
- Masque de sous-réseau
- Plage IP
- Passerelle
- Serveur DNS

```
echo "[1/8] Demande des informations de configuration
    r seau pour le DHCP..."
read -p "Adresse r seau (ex: 192.168.100.0): " RESEAU
read -p "Masque de sous-r seau (ex: 255.255.255.0): "
    NETMASK
read -p "Adresse IP de d but (ex: 192.168.100.100): "
    IP_DEBUT
read -p "Adresse IP de fin (ex: 192.168.100.200): "
    IP_FIN
read -p "Passerelle (ex: 192.168.100.1): " GATEWAY
read -p "DNS (ex: 8.8.8.8): " DNS
```

### Étape 2 : Installation de Docker

```
echo "[2/8] Mise    jour du syst me et installation de
    Docker..."
apt update && apt upgrade -y
apt install -y ca-certificates curl gnupg lsb-release

install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/debian/gpg
    -o /etc/apt/keyrings/docker.asc
chmod a+r /etc/apt/keyrings/docker.asc

echo "deb [arch=$(dpkg --print-architecture) signed-by=/
    etc/apt/keyrings/docker.asc] https://download.docker.
    com/linux/debian $(. /etc/os-release && echo "
    $VERSION_CODENAME") stable" | tee /etc/apt/sources.
    list.d/docker.list > /dev/null

apt update
apt install -y docker-ce docker-ce-cli containerd.io
    docker-buildx-plugin docker-compose-plugin
```

### Étape 3 : Nettoyage des ressources existantes

```
echo "[3/8] Nettoyage des anciens fichiers (si pr sents
    )..."
rm -rf docker-firewall
docker rm -f mon-serveur-dhcp 2>/dev/null || true
docker image rm dhcp-firewall 2>/dev/null || true
```

### Étape 4 : Création de l'environnement Docker

```
echo "[4/8] Cr ation des fichiers Docker..."
mkdir -p docker-firewall
cd docker-firewall
```

**Contenu de script-firewall.sh :**

```
#!/bin/bash
iptables -F
iptables -X
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,
    RELATED -j ACCEPT
iptables -A INPUT -p icmp --icmp-type 8 -j DROP
netfilter-persistent save
```

**Contenu de script-dhcp.sh :**

```
#!/bin/bash
echo 'DHCPDv4_CONF=/etc/dhcp/dhcpd.conf' > /etc/default/
    isc-dhcp-server
echo 'INTERFACESv4="enp0s3"' >> /etc/default/isc-dhcp-
    server

cat <<EOL > /etc/dhcp/dhcpd.conf
option domain-name "gerioux.com";
default-lease-time 345600;
max-lease-time 691200;
authoritative;
log-facility local7;
```

```
subnet $RESEAU netmask $NETMASK {
        range $IP_DEBUT $IP_FIN;
        option domain-name-servers $DNS;
        option routers $GATEWAY;
}
EOL

systemctl restart isc-dhcp-server.service
```

**Contenu de Dockerfile :**

```
FROM debian:stable-slim

RUN apt update && apt install -y iptables iptables-
    persistent isc-dhcp-server net-tools nano systemctl

COPY script-dhcp.sh /root/script-dhcp.sh
COPY script-firewall.sh /root/script-firewall.sh

RUN chmod +x /root/script-dhcp.sh /root/script-firewall.
    sh

CMD /root/script-firewall.sh && /root/script-dhcp.sh &&
    tail -f /dev/null
```

### Étape 5 : Construction de l'image Docker

```
chmod +x script-dhcp.sh script-firewall.sh
echo "[5/8] Construction de l image Docker..."
docker build -t dhcp-firewall .
```

### Étape 6 : Lancement du conteneur

```
echo "[6/8] Lancement du conteneur DHCP + Pare-feu..."
docker run --rm --net=host --cap-add=NET_ADMIN --name
    mon-serveur-dhcp dhcp-firewall
```

## 4. Résultats et vérification

Après exécution, on vérifie que :
— Le serveur DHCP fonctionne et attribue des adresses IP
— Les règles de pare-feu bloquent les connexions non sollicitées
— Le conteneur reste actif et les services fonctionnent

## 5. Création de l'instance sur AWS

### 5.1 Définition du nom de l'instance

### 5.2 Choix du système d'exploitation

### 5.3 Choix d'instance (Nous avons choisi la version gratuite)

### 5.4 Création de la « key pair »

### 5.5 Paramétrage du réseau de l'instance

### 5.6 Définition des règles du groupe de sécurité pour l'accès à l'instance

### 5.7 Configuration du stockage de l'instance

### 5.8 Récapitulatif de l'instance avant lancement

### 5.9 Connexion à distance avec SSH pour accéder à l'instance EC2

### 5.10 Connexion à distance via SSH

La connexion à distance à l'instance EC2 a été réalisée avec succès. On observe que la session s'est ouverte correctement sur l'instance, ce qui confirme les éléments suivants :

— Le groupe de sécurité AWS autorise bien les connexions entrantes sur le port **22** (SSH).
— La clé `SSH` a été correctement associée à l'instance EC2.
— L'instance est fonctionnelle et prête à recevoir les prochaines configurations (serveur DHCP, pare-feu, etc.).

#### Commande utilisée pour le transfert du script

Le script a été transféré de la machine locale vers l'instance EC2 à l'aide de la commande suivante :

```
scp -i "C:\Users\Gringo\Downloads\kirikou.pem" \
"C:\Users\Gringo\Downloads\Mini\script_dhcp_docker_firewall.sh"
   \
admin@ec2-13-60-19-83.eu-north-1.compute.amazonaws.com:/home/
   admin/
```

#### Résolution du problème d'encodage sur Debian

Sur certaines distributions Debian, le script peut contenir des caractères invisibles (carriage return '$^M$') si créé sous Windows. Voici la solution appliquée pour corriger ce problème :

```
sudo apt update
sudo apt install dos2unix
dos2unix script_dhcp_docker_firewall.sh
```

#### Exécution du script sur l'instance

Avant exécution, le script a été rendu exécutable puis lancé :

```
chmod +x script_dhcp_docker_firewall.sh
./script_dhcp_docker_firewall.sh
```

Image d'installation de l'image Docker

## 6. Conclusion

Ce projet montre la capacité à automatiser le déploiement d'un service réseau (DHCP) dans un conteneur sécurisé via Docker. Le script facilite la configuration dynamique et renforce la sécurité grâce à iptables. Cette approche est généralisable à d'autres services dans une infrastructure DevOps moderne.

## Références


— https ://docs.docker.com/engine/install/debian/
— https ://www.hostinger.fr/tutoriels/iptables
— https ://www.it-connect.fr/serveur-dhcp-sous-linux/
— https ://www.it-connect.fr/installation-pas-a-pas-de-docker-sur-debian-11/

## Annexe : Script Bash complet

```
#!/bin/bash

set -e

if [ "$EUID" -ne 0 ]; then
echo "Ce script doit  tre  ex cut  avec les droits
   root. Utilisez 'sudo ./setup-dhcp-firewall.sh'"
exit 1
fi

echo "[1/8] Demande des informations de configuration
   r seau pour le DHCP..."
read -p "Adresse r seau (ex: 192.168.100.0): " RESEAU
read -p "Masque de sous-r seau (ex: 255.255.255.0): "
   NETMASK
read -p "Adresse IP de d but (ex: 192.168.100.100): "
   IP_DEBUT
read -p "Adresse IP de fin (ex: 192.168.100.200): "
   IP_FIN
read -p "Passerelle (ex: 192.168.100.1): " GATEWAY
read -p "DNS (ex: 8.8.8.8): " DNS

echo "[2/8] Mise    jour du syst me et installation de
   Docker..."
apt update && apt upgrade -y
apt install -y ca-certificates curl gnupg lsb-release

install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg
   -o /etc/apt/keyrings/docker.asc
chmod a+r /etc/apt/keyrings/docker.asc

echo "deb [arch=$(dpkg --print-architecture) signed-by=/
   etc/apt/keyrings/docker.asc] https://download.docker.
   com/linux/debian $(. /etc/os-release && echo \"
   $VERSION_CODENAME\") stable" | tee /etc/apt/sources.
   list.d/docker.list > /dev/null

apt update
apt install -y docker-ce docker-ce-cli containerd.io
   docker-buildx-plugin docker-compose-plugin

echo "[3/8] Nettoyage des anciens fichiers (si pr sents
   )..."
rm -rf docker-firewall
docker rm -f mon-serveur-dhcp 2>/dev/null || true
docker image rm dhcp-firewall 2>/dev/null || true

echo "[4/8] Cr ation des fichiers Docker..."
mkdir -p docker-firewall
cd docker-firewall

# script-firewall.sh
cat > script-firewall.sh << 'EOF'
#!/bin/bash
iptables -F
```

```
iptables -X
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,
    RELATED -j ACCEPT
iptables -A INPUT -p icmp --icmp-type 8 -j DROP
netfilter-persistent save
EOF

# script-dhcp.sh
cat > script-dhcp.sh << EOF
#!/bin/bash
echo 'DHCPDv4_CONF=/etc/dhcp/dhcpd.conf' > /etc/default/
    isc-dhcp-server
echo 'INTERFACESv4="enp0s3"' >> /etc/default/isc-dhcp-
    server

cat <<EOL > /etc/dhcp/dhcpd.conf
option domain-name "gerioux.com";
default-lease-time 345600;
max-lease-time 691200;
authoritative;
log-facility local7;

subnet $RESEAU netmask $NETMASK {
        range $IP_DEBUT $IP_FIN;
        option domain-name-servers $DNS;
        option routers $GATEWAY;
}
EOL

systemctl restart isc-dhcp-server.service
EOF

# Dockerfile
cat > Dockerfile << 'EOF'
FROM debian:stable-slim

RUN apt update && apt install -y iptables iptables-
    persistent isc-dhcp-server net-tools nano systemctl

COPY script-dhcp.sh /root/script-dhcp.sh
COPY script-firewall.sh /root/script-firewall.sh

RUN chmod +x /root/script-dhcp.sh /root/script-firewall.
    sh

CMD /root/script-firewall.sh && /root/script-dhcp.sh &&
    tail -f /dev/null
EOF

chmod +x script-dhcp.sh script-firewall.sh

echo "[5/8] Construction de l image Docker..."
docker build -t dhcp-firewall .
```

```
echo "[6/8] Lancement du conteneur DHCP + Pare-feu..."
docker run --rm --net=host --cap-add=NET_ADMIN --name
    mon-serveur-dhcp dhcp-firewall
```