



REPOBLIKAN'I MADAGASIKARA  
Fitiavana-Tanindrazana-Fandrosoana



MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE  
SCIENTIFIQUE  
ÉCOLE SUPÉRIEURE POLYTECHNIQUE D'ANTSIRANANA  
Mention : STIC

---

---

## MINI-PROJET

---

---

# INSTALLATION ET CONFIGURATION AUTOMATIQUE D'UN SERVEUR DHCP ET FIREWILL 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 2023-2024*



è

## 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
rseau pour le DHCP..."
read -p "Adresse rseau (ex: 192.168.100.0): " RESEAU
read -p "Masque de sous-rseau (ex: 255.255.255.0): "
NETMASK
read -p "Adresse IP de dbut (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 à jour du système 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 présents)
..."
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] Création 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
```

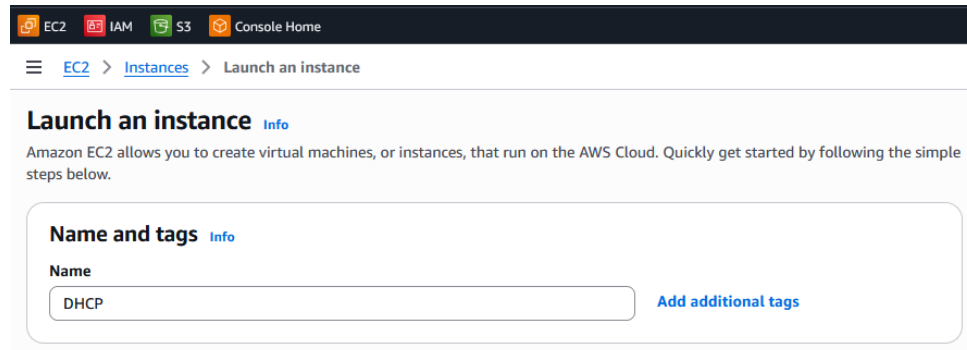
## 4. Résultats et vérification

Après exécution, on vérifie que :

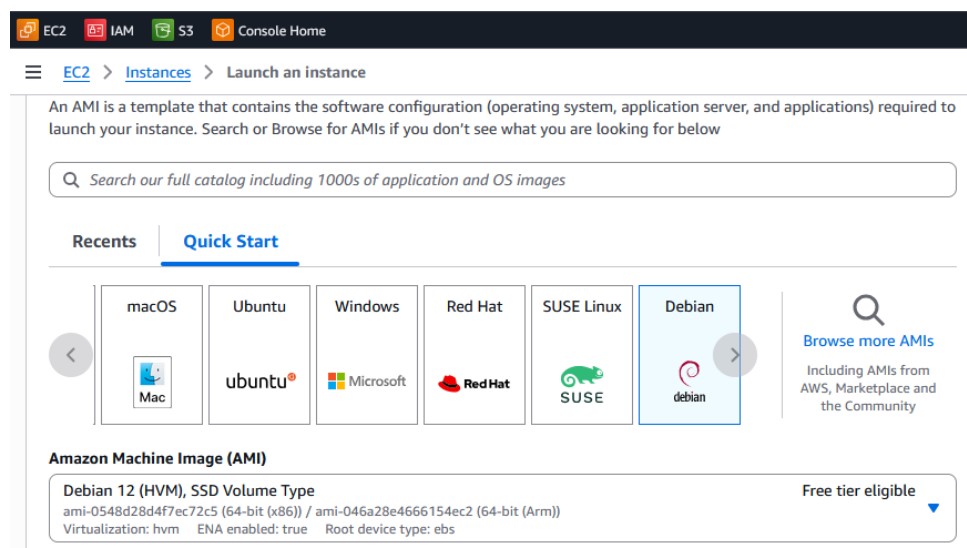
- Le serveur DHCP fonctionne et attribue des adresses IP
- Les règles de pare-feu bloquent les connexions non sollicitées
- Le conteneur reste actif et les services fonctionnent

## 5. Création de l'instance sur AWS

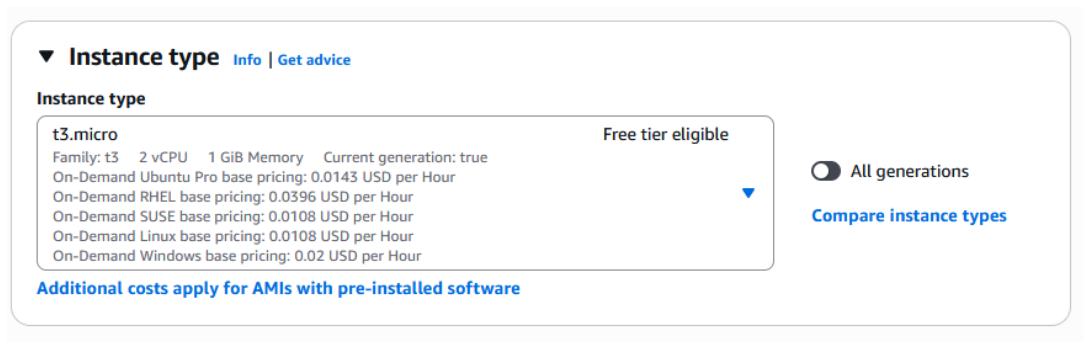
### 5.1 Définition du nom de l'instance



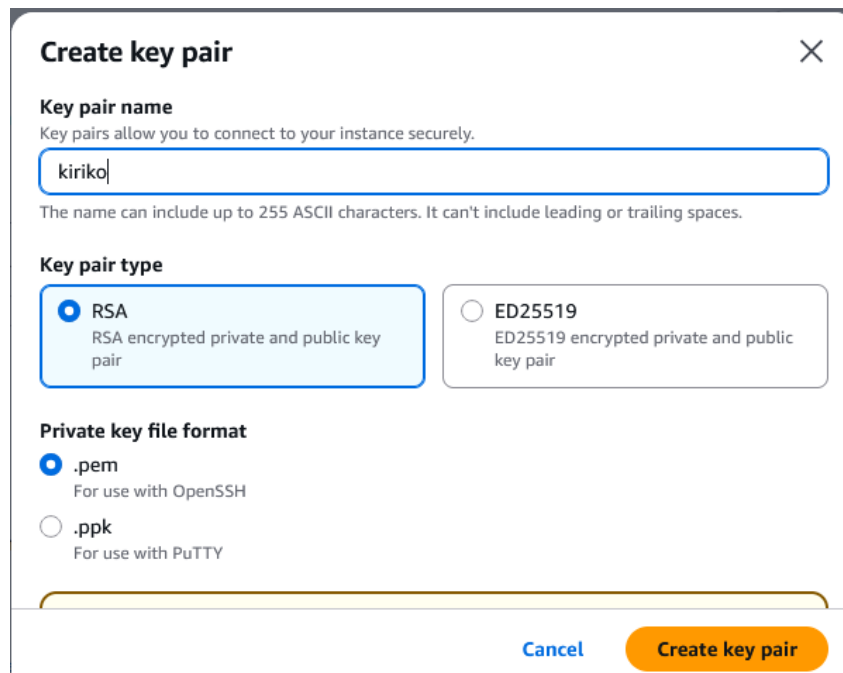
### 5.2 Choix du système d'exploitation



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



## 5.4 Création de la « key pair »



**Create key pair** ✕

**Key pair name**  
Key pairs allow you to connect to your instance securely.

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type**

☒ **RSA**  
RSA encrypted private and public key pair

☐ **ED25519**  
ED25519 encrypted private and public key pair

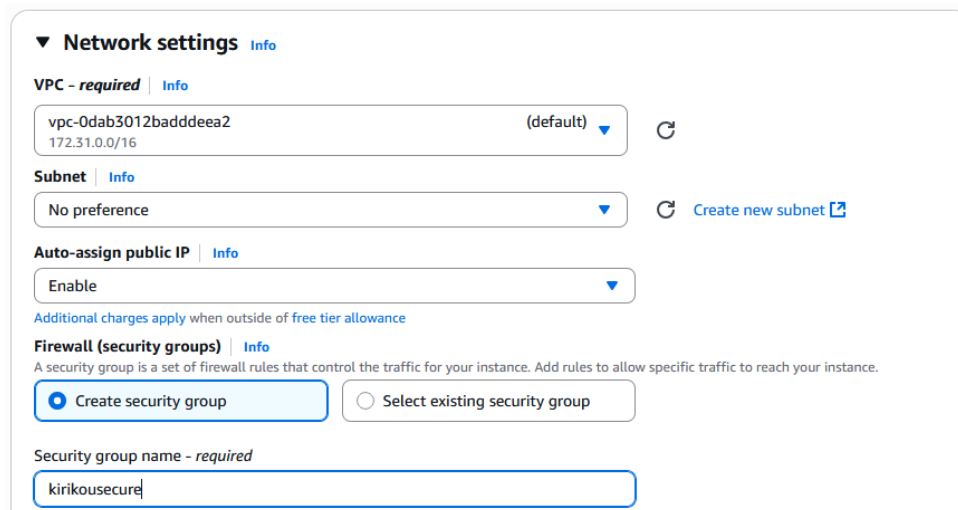
**Private key file format**

☒ **.pem**  
For use with OpenSSH

☐ **.ppk**  
For use with PuTTY

[Cancel](#) [Create key pair](#)

## 5.5 Paramétrage du réseau de l'instance



**▼ Network settings** [Info](#)

**VPC - required** [Info](#)

(default) ↻

**Subnet** [Info](#)

↻ [Create new subnet](#)

**Auto-assign public IP** [Info](#)

Additional charges apply when outside of free tier allowance

**Firewall (security groups)** [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ **Create security group** ☐ **Select existing security group**

**Security group name - required**

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

The screenshot shows the AWS Management Console interface for configuring Inbound Security Group Rules. The breadcrumb navigation at the top indicates the path: EC2 > Instances > Launch an instance. The main section is titled "Inbound Security Group Rules".

There are two rules defined:

- Security group rule 1 (TCP, 22, 0.0.0.0/0)**:
  - Type: ssh
  - Protocol: TCP
  - Port range: 22
  - Source type: Anywhere
  - Source: 0.0.0.0/0
  - Description: e.g. SSH for admin desktop
- Security group rule 2 (TCP, 80, 0.0.0.0/0)**:
  - Type: HTTP
  - Protocol: TCP
  - Port range: 80
  - Source type: Anywhere
  - Source: 0.0.0.0/0
  - Description: e.g. SSH for admin desktop

Each rule has a "Remove" button in the top right corner.

## 5.7 Configuration du stockage de l'instance

The screenshot shows the "Configure storage" section in the AWS Management Console. The breadcrumb navigation at the top indicates the path: EC2 > Instances > Launch an instance. The main section is titled "Configure storage".

The configuration shows:

- 1x 30 GiB gp3 Root volume, 3000 IOPS, Not encrypted

Below the configuration, there is a blue box with an information icon and the text: "Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage".

At the bottom, there is a button labeled "Add new volume".



## 5.8 Récapitulatif de l'instance avant lancement

▼ **Summary**

Number of instances | [Info](#)

1

**Software Image (AMI)**  
Debian 12 (20250316-2053)  
ami-0548d28d4f7ec72c5

**Virtual server type (instance type)**  
t3.micro

**Firewall (security group)**  
New security group

**Storage (volumes)**  
1 volume(s) 30 GiB

[Cancel](#) [Launch instance](#) [Preview code](#)

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

```
C:\Users\Gringo\Downloads>ssh -i "kirikou.pem" admin@ec2-13-60-19-83.eu-north-1.compute.amazonaws.com
Linux ip-172-31-43-43 6.1.0-32-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.129-1 (2025-03-06) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jun 30 15:04:15 2025 from 102.68.234.150
admin@ip-172-31-43-43:~$
```

## 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

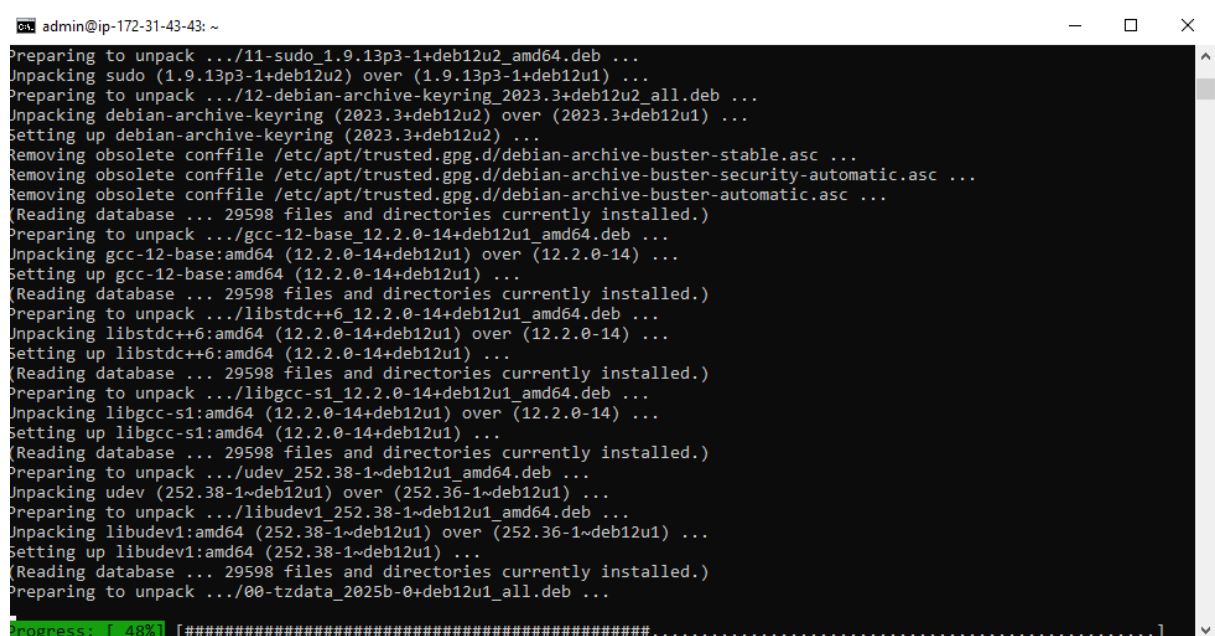
Sur certaines distributions Debian, le script peut contenir des caractères invisibles (carriage return <sup>M</sup>) si créé sous Windows. Voici la solution appliquée pour corriger ce problème :

```
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
```



```
admin@ip-172-31-43-43: ~
Preparing to unpack .../11-sudo_1.9.13p3-1+deb12u2_amd64.deb ...
Unpacking sudo (1.9.13p3-1+deb12u2) over (1.9.13p3-1+deb12u1) ...
Preparing to unpack .../12-debian-archive-keyring_2023.3+deb12u2_all.deb ...
Unpacking debian-archive-keyring (2023.3+deb12u2) over (2023.3+deb12u1) ...
Setting up debian-archive-keyring (2023.3+deb12u2) ...
Removing obsolete conffile /etc/apt/trusted.gpg.d/debian-archive-buster-stable.asc ...
Removing obsolete conffile /etc/apt/trusted.gpg.d/debian-archive-buster-security-automatic.asc ...
Removing obsolete conffile /etc/apt/trusted.gpg.d/debian-archive-buster-automatic.asc ...
(Reading database ... 29598 files and directories currently installed.)
Preparing to unpack .../gcc-12-base_12.2.0-14+deb12u1_amd64.deb ...
Unpacking gcc-12-base:amd64 (12.2.0-14+deb12u1) over (12.2.0-14) ...
Setting up gcc-12-base:amd64 (12.2.0-14+deb12u1) ...
(Reading database ... 29598 files and directories currently installed.)
Preparing to unpack .../libstdc++6_12.2.0-14+deb12u1_amd64.deb ...
Unpacking libstdc++6:amd64 (12.2.0-14+deb12u1) over (12.2.0-14) ...
Setting up libstdc++6:amd64 (12.2.0-14+deb12u1) ...
(Reading database ... 29598 files and directories currently installed.)
Preparing to unpack .../libgcc-s1_12.2.0-14+deb12u1_amd64.deb ...
Unpacking libgcc-s1:amd64 (12.2.0-14+deb12u1) over (12.2.0-14) ...
Setting up libgcc-s1:amd64 (12.2.0-14+deb12u1) ...
(Reading database ... 29598 files and directories currently installed.)
Preparing to unpack .../udev_252.38-1~deb12u1_amd64.deb ...
Unpacking udev (252.38-1~deb12u1) over (252.36-1~deb12u1) ...
Preparing to unpack .../libudev1_252.38-1~deb12u1_amd64.deb ...
Unpacking libudev1:amd64 (252.38-1~deb12u1) over (252.36-1~deb12u1) ...
Setting up libudev1:amd64 (252.38-1~deb12u1) ...
(Reading database ... 29598 files and directories currently installed.)
Preparing to unpack .../00-tzdata_2025b-0+deb12u1_all.deb ...
Progress: [ 48%] [#####.....]
```

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
rseau pour le DHCP..."
read -p "Adresse rseau (ex: 192.168.100.0): " RESEAU
read -p "Masque de sous-rseau (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
```