ProxMox VE

Proxmox Virtual Environment est une solution de virtualisation libre (licence AGPLv3) basée sur l'hyperviseur Linux KVM, et offre aussi une solution de containers avec LXC.

- Sauvegarde et restauration de VM
- Ajouter un node avec des VMs a un cluster (en cours)
- Sécurisation basique de son Proxmox
- Cloud-Init

Sauvegarde et restauration de VM

Dans ce tutoriel

Comment faire la sauvegarde d'une VM ainsi que la sauvegarde régulière d'un groupe de VM. Ensuite comment restaurer une sauvegarde.

Le tout avec l'interface graphique ainsi qu'en ligne de commande.

Sauvegarde individuel

Interface Web

Pour commencer, il faut se connecter à l'interface web du serveur sur lequel on souhaite effectuer l'opération.

10.PNG

Dans le menu de gauche, sélectionner la VM, puis onglet Sauvegarde

11.PNG

On peut voir que la seule option proposer est "Sauvegarder Maintenant".

Une fenêtre de configuration s'ouvre et propose 4 options:

12.PNG

- Stockage: Pour choisir le disque sur lequel on souhaite mettre la sauvegarde
- Mode: Trois options de sauvegarde a bien selectionné

1. Snapshot: Je le déconseille fortement, il effectue la sauvegarde de la VM pendant son fonctionnement. Il y à donc un gros risque de corruption non négligeable!
2. Suspendre: Celui-là est plus pratique, il va mettre en pause la VM puis effectuer une Snapshot, il y a donc moins de risques pour les données. Pratique pour un serveur qui a peu de tâches en cours.
3. Stopper: Fortement recommander pour les serveurs qui ont beaucoup de tâches en cours, il va éteindre la VM, lancer en arrière-plan la tâche de sauvegarde puis redémarrer la VM. Le temps d'arrêt est donc faible car la VM redémarre dès que la tâche est lancée (quelques secondes). De plus ce mode garantit l'intégrité des données.

- Compression: Permet de réduire la taille de la sauvegarde, encore une fois trois options:

1. Aucune: Désactive la compression, le fichier sera donc en RAW (.lvm). La sauvegarde sera très rapide mais prendra beaucoup de place
2. LZO: Compression qui permet de réduire la taille du fichier légèrement, sans impacter énormément le temps de sauvegarde.
3. GZIP: Grosse compression qui permet d'économiser beaucoup de Go, mais qui augmente considérablement le temps de la sauvegarde (quelques dizaines de minutes voir plus)!

- Mail: Permet d'envoyer un mail quand la tâche est terminée pour indiquer s'il y a eu une erreur ou pas. **Cette option ne fonctionne que si vous avez au préalable configuré l'envoi de mail sur le serveur.**

Il ne reste plus qu'à lancer la sauvegarde. Une fois fini, elle apparaît dans la liste.

CLI

La commande qui permet d'effectuer la sauvegarde est plus complète que l'interface graphique, elle permet plus de personnalisation.

Pour voir toutes les options de personalisation: Blog Proxmox - vzdump

Voici l'équivalent en commande de la sauvegarde effectuer en graphique:

```
vzdump 100 --dumpdir /deb/sdb2/dump --mode stop  --compress lzo --mailto
```

100: Correspond a l'ID de la VM

--dumpdir: Le disque ou l'emplacement de stockage

--mode: snaphot - suspend - stop

--compress: 0 - lzo - gzip

--mailto: adresse mail

Sauvegarde multiple et programmée

Maintenant que l'on a vu comment faire des sauvegardes individuelles, on va voir comment en faire plusieurs en même temps est les programmer.

Interface Web

Dans le menu de gauche on sélectionne Datacenter, puis sauvegarde.

13.PNG

En cliquant sur ajouter on va pouvoir configurer notre tâche

14.PNG

- Noeud: Choisir le serveur (dans le cas d'un cluster)
- Jour de la semaine: Quelles jours doit être effectuer la tâche
- Heure de début: Quelle heure doit être démarrer la tâche
- Mail: Pour envoyer un mail s'il y a une erreur uniquement ou à chaque fois
- Mode de sélection: Propose trois choix:

1. Inclure: Sauvegarde que les VMs sélectionnées
2. Tout: Sauvegarde toutes les VMs
3. Exclure: Sauvegarde toutes les VMs excepter les sélectionnées

- Activer: Permet d'activer ou non la tâche

Pour le reste des options, il s'agit des même que pour la sauvegarde individuelle.

Une fois créer, on peut voir un résumer des options choisis.

15.PNG

Restaurer une sauvegarde

Pour restaurer une VM, il y a deux possibiliter, soit la VM existe déjà, soit elle doit être créer.

Interface web

Si elle existe déjà, on sélectionne la VM puis Sauvegarde.

On sélectionne la sauvegarde à restaurer, puis clic sur "restaurer".

Il faut juste choisir le stockage de la VM ainsi que si on le souhaite une limite en vitesse pour ne pas pourrir l'I/O des disques.

16.PNG

Si elle n'existe pas ou plus, alors il faut aller dans le disque ou se trouve la sauvegarde, la sélectionner puis cliquer sur "restaurer".

On choisi l'emplacement ainsi que l'ID que l'on veut donner.

17.PNG

La VM sera créée puis restaurer.

CLI

Comme pour la sauvegarde, en ligne de commande plus de choix s'offre à nous, pour tous les connaître je vous recommande d'aller sur la page dédiée: Blog Proxmox - qmrestore

Voici l'équivalent en commande de restauration d'une VM qui existe déjà:

```
qmrestore /deb/sdb2/fichier_de_sauvegarde.gz 100 --force true
```

Après l'appel de qmrestore j'ai indiqué l'emplacement du fichier de sauvegarde puis l'ID de la VM à restaurer

--force: permet de forcer à réécrire une VM existante

Si la VM n'existe pas, il suffit d'indiquer un ID qui n'est pas pris.

Ajouter un node avec des VMs a un cluster (en cours)

Faire les copies des VMs

mkdir -p /home/qemu-server/
mkdir -p /home/lxc/
cp /etc/pve/qemu-server/* /home/qemu-server/
cp /etc/pve/lxc/* /home/lxc/

Vérifier que les Vm on pas les mêmes ID entre les deux clusters
Vérifier que tes serveurs ont bien la même version de proxmox

Si c'est bon tu verifies l'état de ton fichier hosts (/etc/hosts)
Tu ping les hostname

exemple :
10.88.88.20 pve01
10.88.88.21 pve02
10.88.88.22 pve-stor01
10.88.88.23 pve03

ping pve01
ping pve02 etc..

si tout ping alors tu lances la creation du cluster (via l'interface web)
et aprés tu ajoutes les autres serveurs

Si ca foire
NE REBOOT PAS LES NODES

Tu te connectes en ssh ou console
et tu fais :

service pve-cluster restart
service corosync restart

cp /home/lxc/* /etc/pve/lxc/

cp /home/qemu-server/* /etc/pve/qemu-server/

et tu retrouveras tes vms

Sécurisation basique de son Proxmox

Je vois sur Internet des serveurs Proxmox pas sécurisés voire pire, des conseils qui préconisent d'utiliser un firewall comme pfSense installé en machine virtuelle pour sécuriser l'hyperviseur, alors que ce dernier est hébergé chez un fournisseur dont on ne sait rien du réseau. Votre fournisseur peut proposer un firewall comme chez OVH avec son système anti-DDoS, mais cela ne vous protège que de l'extérieur, pas de l'intérieur du réseau, qui lui comporte une multitude de serveurs d'autres clients.

La logique pour un réseau en entreprise ou chez soi, pour un home lab ou de l'auto-hébergement sera la même.

Quel que soit l'hyperviseur, il faut que celui-ci soit sécurisé. S'il n'est pas sécurisé, toutes vos machines virtuelles ou vos conteneurs sont potentiellement compromis.

Pour citer Andy Grove, co-fondateur d'Intel, dans son livre autobiographique qui porte le même nom : « Seuls les paranoïaques survivent ».

[Wikipedia]: Il explique dans sa biographie *Seuls les paranoïaques survivent* que le moteur psychique qui lui a permis de mener son entreprise au sommet a été durant 38 ans

Ce concept est parfaitement adapté pour la sécurité.

1. Protection physique

Le ou les serveurs doivent être à minima protégés physiquement, comme une salle sécurisée, ou si dans un datacenter, dans une baie fermée à clefs. Mais il est aussi important de protéger l'accès au BIOS/UEFI par un mot de passe costaud et interdire le boot sur autre chose que l'hyperviseur.

Il serait dommage qu'une personne avec une clef usb boote sur un autre OS, et modifie le mot de passe root, formate le ou les disques où sont installés Proxmox, etc.

Les sociétés qui fournissent des serveurs comme OVH, Scaleway, etc. peuvent avoir dans leurs employés des personnes mal intentionnées, tout comme dans vos collègues de travail ou prestataires.

2. Firewall

Proxmox intègre un firewall agissant sur 3 parties distinctes :

- Datacenter
- Serveur Proxmox alias PVE
- Machines virtuelles et conteneurs LXC

La partie machine virtuelle et conteneurs est indépendante des deux autres. Elle n’a pas d'intérêt dans la sécurisation du serveur Proxmox.

Je me base seulement pour un seul hôte Proxmox, qui dans ce cas le fait de faire les règles au niveau du data center ou du node n’aura pas forcément d’impact.

2.1 Ports utilisés par Proxmox

Proxmox utilise par défaut ces ports pour fonctionner.

Si vous modifiez le port SSH, utilisez le même port sur tous les serveurs du cluster au risque d’avoir des surprises.

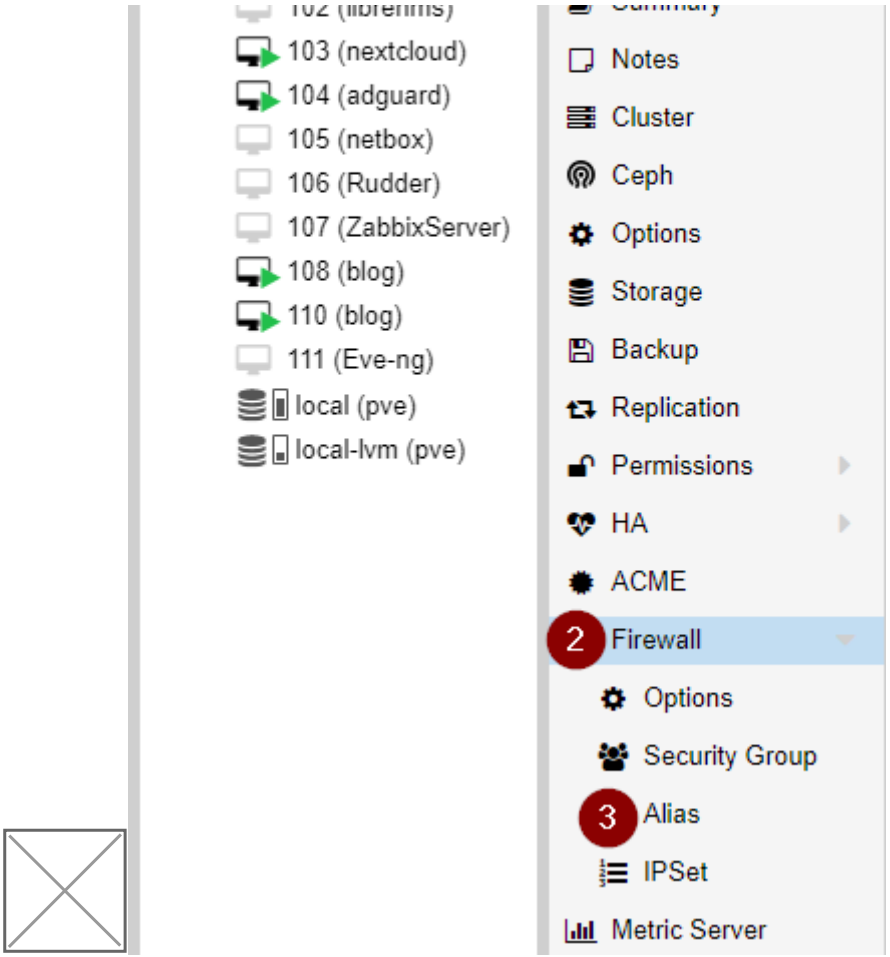
Services	Protocole	Ports
Web interface	TCP	8006
SSHd	SSH	22
pvedaemon (en écoute)	TCP	85
rpcbind	TCP	111
corosync multicast (pour les clusters)	UDP	5404, 5405
SPICE proxy	TCP	3128

Source

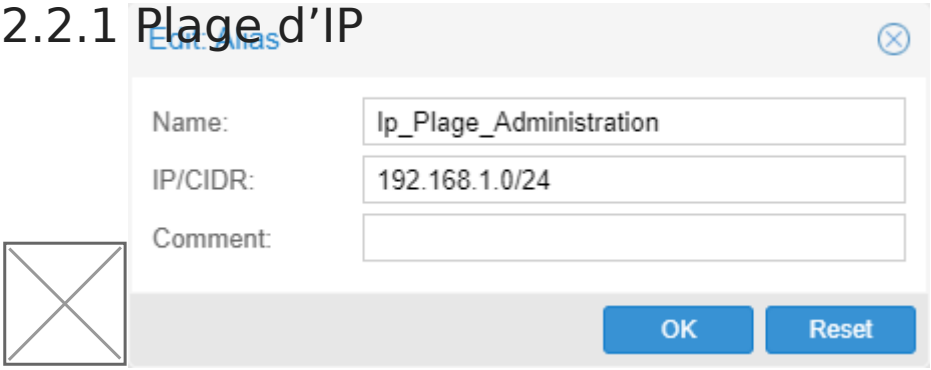
2.2 Alias

Alias se trouve dans la partie firewall du Datacenter et va permettre de nommer les IP ou les plages d’IP à utiliser dans le firewall.

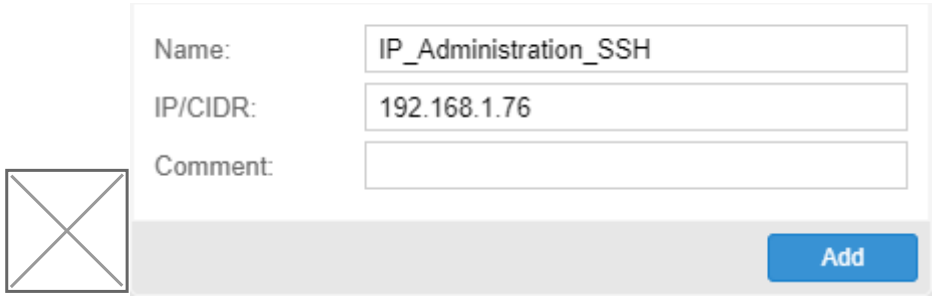
C’est une habitude de travail de créer des alias, ça évite les oublis, ça permet aussi d’aller plus vite quand on a une correction à effectuer sur une grosse quantité de règles de filtrage.



2.2.1 Plage d’IP



2.2.2 Une IP



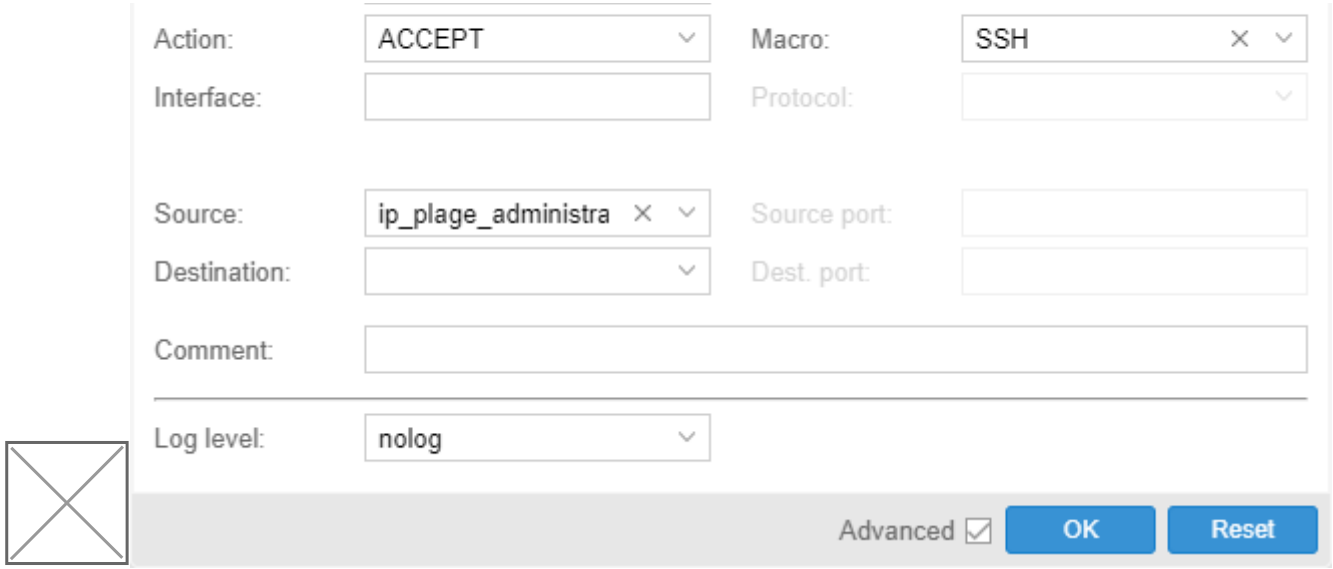
2.3 Règles de firewall

Rien de compliqué, on autorise en entrée le port 8006, le SSH, le ping. Et comme ce n'est qu'un seul node, pas besoin de préciser la destination (pas idéal, mais cela simplifie les choses) ni l'interface sur laquelle le trafic doit passer, qui de toute façon pour un seul node sera vmbr0.

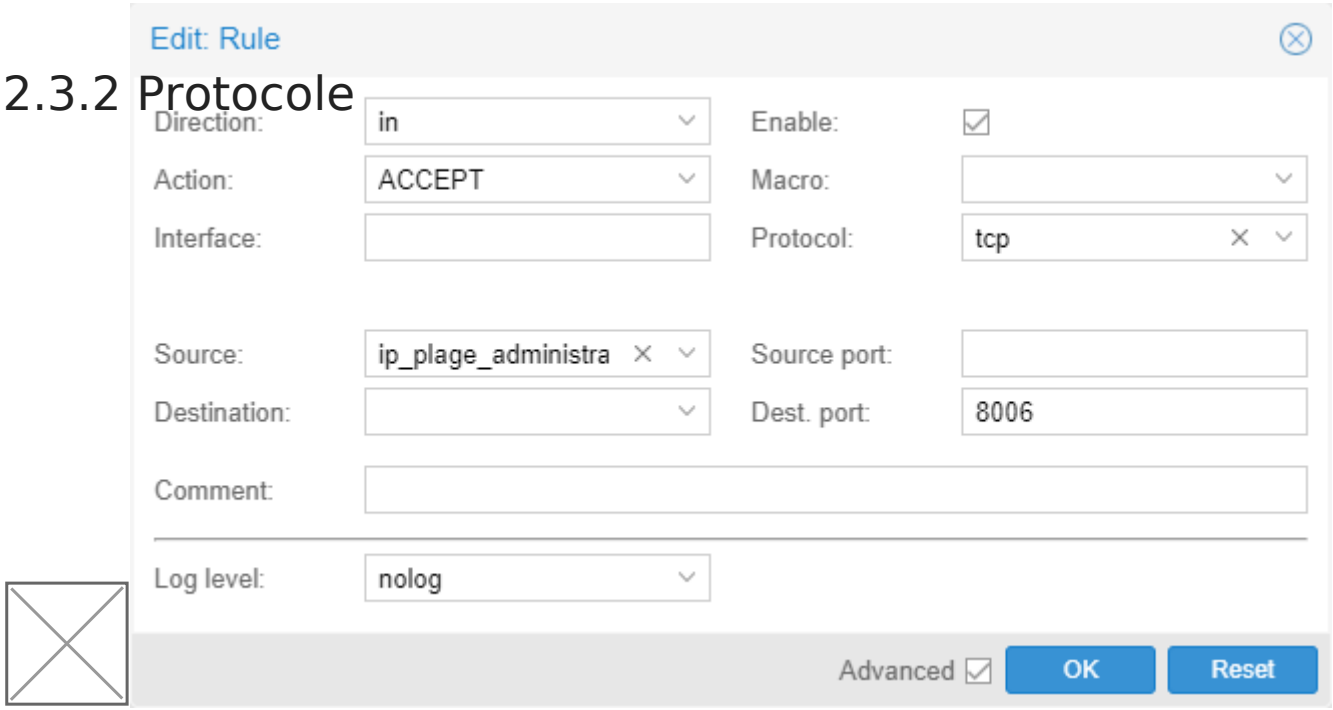
Direction	Action	Source	Protocol	Destination Port	Log Level	Comment
in	ALLOW		tcp	8006	nolog	web GUI
in	ALLOW		tcp	22	nolog	ssh
in	ALLOW		icmp		nolog	ping

2.3.1 Macro

Il est possible d'utiliser des macros de configuration pour certains protocoles comme le SSH ou le protocole SMB qui a besoin d'ouverture de plusieurs ports (TCP 445, TCP 139, UDP 138-139), cela facilite grandement la lecture des règles si vous devez l'utiliser.



2.3.2 Protocole



2.4 Chef, je me suis coupé la main !

En console, il est possible de désactiver le firewall

Éditez le fichier */etc/pve/firewall/cluster.fw* et remplacez la valeur **1** par **0**.

```
[OPTIONS]

enable: 1
```

2.5 Utilisation de pfsense pour les VM

Je vous oriente sur le site de zwindler qui a 3 articles sur le sujet :

- Proxmox VE 6 + pfsense sur un serveur dédié (1/2)
- Proxmox VE 6 + pfsense sur un serveur dédié (2/2)
- Optimisation de PFsense dans Proxmox VE

Et aussi le script bash de Noa disponible sur son GitHub :

- Iptables Proxmox Forward pfsense

3. Fail2Ban

3.1 Installation de Fail2Ban

`apt install fail2ban`

3.2 Configuration de fail2Ban

Editez le fichier :/etc/fail2ban/jail.local

```
[proxmox]
enabled = true
port = https,http,8006
filter = proxmox
logpath = /var/log/daemon.log
maxretry = 3
bantime = 3600 #1 heure
```

```
[sshd]
enabled = true
port = ssh
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
findtime = 300
bantime = 86400 #24 heures
ignoreip = 127.0.0.1
```

/etc/fail2ban/filter.d/proxmox.conf

```
[Definition]
failregex = pvedaemon\[.*authentication failure; rhost=<HOST> user=.* msg=.*
ignoreregex =
```

Relancer le service de Fail2Ban

```
systemctl restart fail2ban.service
```

Sources : wiki Proxmox

3.3 Les commandes utiles de Fail2Ban

3.3.1 Bannir une IP

```
fail2ban-client set [nom du jail] banip [IP à bannir]
```

3.3.2 Enlever le ban d’une IP

```
fail2ban-client set [nom du jail] unbanip [IP concerné]
```

3.3.3 Lister les règles

```
fail2ban-client status

Status
|- Number of jail:      1
```

```
`- Jail list:   sshd
```

3.3.4 Détails d’une règle

```
fail2ban-client status sshd

Status for the jail: sshd
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     5
|  `- File list:        /var/log/auth.log
`- Actions
   |- Currently banned: 1
   |- Total banned:     1
   `- Banned IP list:   192.168.1.21
```

Et si l’on veut en savoir plus sur les tentatives de connexion, il faut regarder dans /var/log/auth.log

```
tail /var/log/auth.log

Dec  9 12:46:14 pve sshd[3769206]: Failed password for nidouille from 192.168.1.21 port 39516 ssh2
Dec  9 12:46:18 pve sshd[3769206]: Failed password for nidouille from 192.168.1.21 port 39516 ssh2
Dec  9 12:46:22 pve sshd[3769206]: Failed password for nidouille from 192.168.1.21 port 39516 ssh2
Dec  9 12:46:23 pve sshd[3769206]: Connection closed by authenticating user nidouille 192.168.1.21 port 39516 [preauth]
Dec  9 12:46:23 pve sshd[3769206]: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser= rhost=192.168.1.21  user=nidouille
```

4. SSH

Par défaut, Proxmox ne propose qu’un compte utilisateur : root. On va le sécuriser à minima pour les connexions SSH.

Voici les options activées après une installation d’un node dans /etc/ssh/sshd_config, et ce n’est pas folichon.

```
PermitRootLogin yes

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication no

UsePAM yes
X11Forwarding yes
PrintMotd no

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

# override default of no subsystems
Subsystem       sftp    /usr/lib/openssh/sftp-server
```

Fail2Ban amène une protection pour les attaques par brute force, mais si on garde l'utilisateur root pour l'accès distant en ssh, on va monter d'un cran la sécurité en obligeant la connexion via clefs. Je ne saurais que trop conseiller la désactivation de l'utilisateur SSH au profit d'un autre compte système.

Je pars du principe que vous avez vos clefs SSH privés et publique.

Dans /root/.ssh/authorized_keys, vous allez renseigner votre clef publique

Puis modifier /etc/ssh/sshd_config pour forcer l'authentification par clefs.

```
#PermitRootLogin yes
PermitRootLogin prohibit-password
PubkeyAuthentication yes
PasswordAuthentication no
PermitEmptyPassword no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication no

UsePAM yes
X11Forwarding no
PrintMotd no
```

```
# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

# override default of no subsystems
Subsystem       sftp    /usr/lib/openssh/sftp-server
```

5. Comptes utilisateurs

La gestion des comptes utilisateurs peut être précise et déroutante.

De base, on a deux types de comptes utilisateur :

- PAM (compte système)
- Proxmox

Auquel on peut rajouter d'autre source d'utilisateur via le menu Realms (non traité ici) :

- Active Directory
- LDAP
- OpenID Connect

Puis, on donne aux utilisateurs deux permissions :

- Path
- Role

Il est bien sûr possible de créer des groupes d'utilisateurs, de nouveaux rôles en associant les privilèges que l'on désire et de créer des pools regroupant des VM et des datastores pour encore affiner les droits si besoin.

5.1 Les rôles

Les rôles regroupent les privilèges.

Nom	Privilèges
Administrator	Tous les droits
NoAccess	Aucun droits donc aucun accès
PVEAdmin	Tout sauf les paramètres systèmes (Sys.PowerMgmt, Sys.Modify, Realm.Allocate)

Nom	Privilèges
PVEAuditor	Accès en lecture seule
PVEDatastoreAdmin	Administration des espaces de stockage et des templates (inclus le backup)
PVEDatastoreUser	Allocation des espaces de stockage et des templates (inclus le backup)
PVEPoolAdmin	Administration des pools
PVEPoolUser	Consultations des pools
PVESysAdmin	ACLs utilisateur, audit, console système et journaux système
PVETemplateUser	visualiser et cloner des templates
PVEUserAdmin	administration des utilisateurs
PVEVMAdmin	administrations des VM
PVEVMUser	visualisation, sauvegarde, CDROM de configuration, console VM, gestion de l’alimentation VM

5.2 Comptes système (PAM)

Les comptes PAM sont des comptes systèmes. Les seuls à pouvoir se connecter en SSH ou console.

Pour la création de ce type de compte, en dehors du compte systèmes, le reste peut se faire en console ou via la GUI.

Création du compte système

```
adduser admin1
```

Console

On enregistre le compte système dans la base des comptes Proxmox

```
pveum useradd admin1
```

On va créer le mot de passe

```
pveum passwd admin1@pam
```

On ajoute les droits administrateurs à l’utilisateur

```
pveum aclmod / -user admin1@pam -roles Administrator
```

5.3 Authentification à double facteurs TOTP

Pour la mise en place rapide d'une double authentification, la solution du TOTP est idéal. Il faut juste un gestionnaire de mot de passe qui possède cette fonctionnalité comme Bitwarden, LastPass via son application dédiée Authentificator, NordPass, etc., ou des applications dédiées comme LastPass, Authentificator, etc.

Pour en savoir plus sur le TOTP:

- site de la société française Synestis spécialisé en cybersécurité : lien
- blog de l'hébergeur IONOS : lien

Pour le tutoriel, j'utilise Bitwarden (produit très utilisé) pour la génération du code aléatoire. Mais il n'est pas possible sur l'application d'effectuer des captures d'écran, les captures faites le sont sur mon client Bitwarden installé sur mon PC.

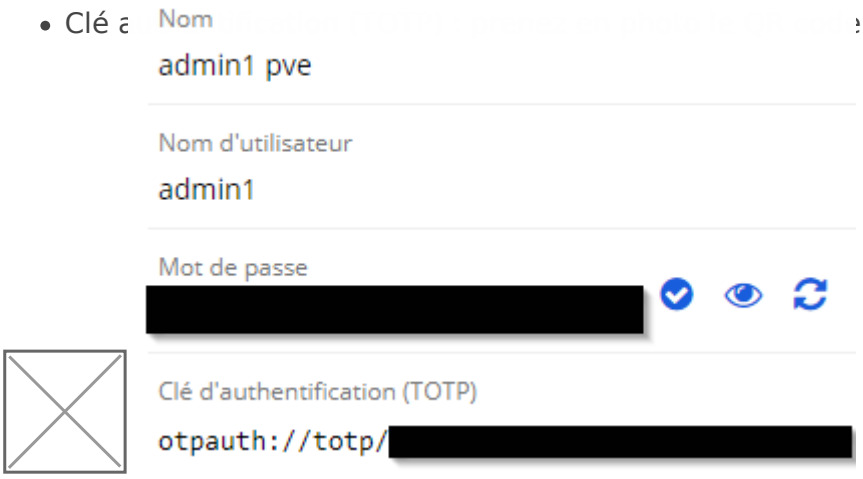
> **Note importante : ne pas utiliser le même logiciel pour le TOTP et les mots de passe !**



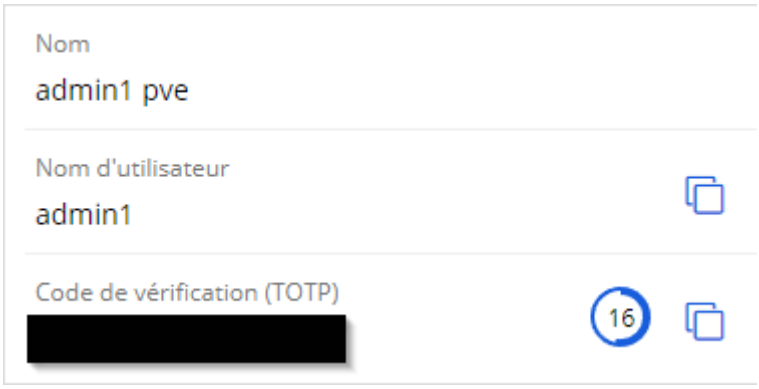
Ouvrir Bitwarden sur votre téléphone

Créer un nouvel élément de type identifiant

- Nom pour Bitwarden
- Nom d’utilisateur
- Le mot de passe de l’utilisateur (facultatif)
- Clé a [illegible] généré et cela remplira la ligne

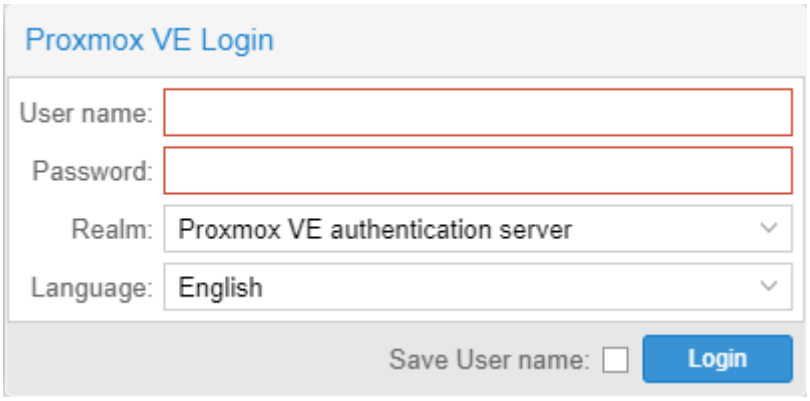


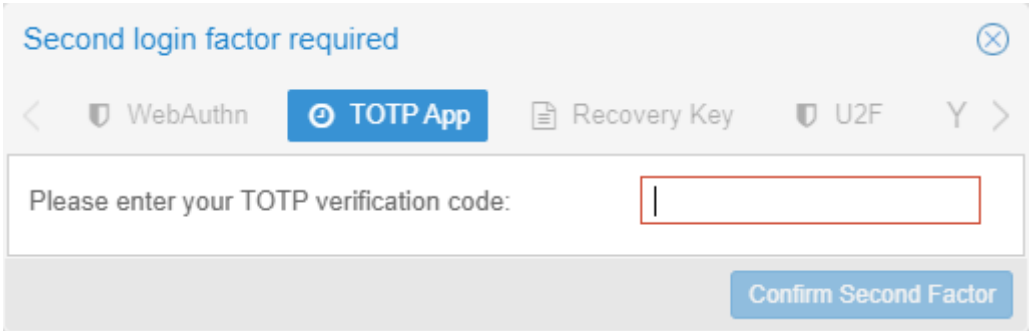
Sauvegarder identifiant créé et ensuite, ouvrez-le. Vous verrez un code généré avec un temps avant la génération d'un nouveau code (30 secondes).



Rentrer le code dans la boite de dialogue de création de compte TOTP de Proxmox pour activer le TOTP du compte.

User	Enabled	TFA Type	Created	Description
admin1@pam	Yes	totp	2021-12-10 12:00:21	admin1 TOTP





Cloud-Init

Cet article est à vocation d'initiation et donne les pistes pour démarrer. Je vais me consacrer sur l'utilisation de Cloud-Init dans Proxmox et non un tuto sur Cloud-Init qui est à lui seul un sujet à part entière. La chaine Youtube et Twitch de @AHP_Nils traite de ce sujet au moment de la rédaction de cet article.

https://www.youtube.com/embed/sSuVybJXkr4

Cloud-Init

Cloud-Init est un projet développé par Canonical pour Ubuntu datant de 2010 sous les licences Apache 2.0 et GPLv3.

Pour utiliser Cloud-Init, on utilise un fichier formaté en YAML. Ce fichier permet de définir plusieurs paramètres basiques à des tâches post-lancement de la machine virtuelle. Exemples de ce que l'on peut configurer :

- Configuration du réseau
- Nom de la machine virtuelle
- Clefs ssh autorisées
- Utilisateurs
- Installer des paquets
- Configurer sudo
- Mise à jour
- Redimensionner des partitions

Au fil des années, Cloud-Init est devenu un standard d'initialisation des machines virtuelles que l'on retrouve chez les fournisseurs de cloud publics :

- AWS
- Azure
- OVHcloud
- Scaleway
- Digital Ocean
- Joyent/SmartOS

- OpenNebula
- etc.

Mais aussi disponible sur des offres de cloud privé :

- KVM
- LXD
- OpenStack
- MASS de Canonical
- VMware

Cloud-Init est disponible pour différents systèmes d'exploitation :

- Alpine Linux
- ArchLinux
- Debian
- DragonFlyBSD
- Fedora
- FreeBSD
- Gentoo Linux
- NetBSD
- OpenBSD
- openEuler
- RHEL/CentOS/AlmaLinux/Rocky/PhotonOS/Virtuozzo/EuroLinux/CloudLinux/MIRACLE LINUX
- SLES/openSUSE
- Ubuntu

Les distributions Linux proposent généralement des images de déploiement pour les différents cloud avec Cloud-Init intégré. Concernant la famille des BSD, elles sont toutes trouvables sur le site bsd-cloud-image.org.

Intégration de Cloud-Init dans Proxmox

Je vais faire rapide, Cloud-Init est intégralement supporté.

Pour utiliser basiquement Cloud-Init, il faut rattacher à la machine virtuelle un volume supplémentaire comme pour le cas d'UEFI (soupir).

(et l'utilisation d'un fichier de configuration au format YAML sera à indiquer en ligne de commande)

Template

On va créer un template a base d'une image Debian dédiée au cloud.

```
wget https://cloud.debian.org/images/cloud/bullseye/latest/debian-11-genericcloud-amd64.qcow2
```

Ensuite, il faut importer le fichier récupéré pour que Proxmox sache le gérer et en faire un template. Mon proxmox est configuré avec des volumes LVM dont le point de montage est le nom par défaut : local-lvm

Création d'une machine virtuelle :

```
qm create 9000 --name debian-11-genericcloud-template --memory 1024 --net0 virtio,bridge=vmbr0
--sockets 1 --cpu cputype=kvm64
```

Cloud-Init demande l'utilisation d'une console série :

```
qm set 9000 --serial0 socket --vga serial0
```

On importe le fichier .qcow2 pour en faire un disque :

```
qm importdisk 9000 debian-11-genericcloud-amd64.qcow2 local-lvm
```

On attache le disque à la machine virtuelle :

```
qm set 9000 --scsihw virtio-scsi-pci --scsi0 local-lvm:vm-9000-disk-0
qm set 9000 --boot c --bootdisk scsi0
```

Si vous désirez agrandir la taille du disque, par défaut, elle est de 2G pour l'image que j'ai récupérée. On augmente de 18G donc cela fera un disque de 20G :

```
qm resize 9000 scsi0 +20G
```

On ajoute le volume dédié à Cloud-Init :

```
qm set 9000 --ide2 local-lvm:cloudinit
```

Création du template :

```
qm template 9000
```

Le template Debian est terminé.

Création d'une machine virtuelle

On clone le template :

```
qm clone 9000 100 --name deb-cloud-init-test
```

Méthode Proxmox

On configure l'IP de la machine virtuelle :

DHCP :

```
qm set 100 --ipconfig0 ip=dhcp
```

IP fixe :

```
qm set 100 --ipconfig0 ip=192.168.1.100/24,gw=192.168.1.1
qm set 100 --ipconfig0 ip6=fd2c:06d1:651e:b83e::2/64,gw6=fd2c:06d1:651e:b83e::1
```

DNS :

```
qm set 100 --nameserver 192.168.1.1
```

Utilisateur :

```
qm set 100 -ciuser nidouille
```

Clefs SSH :

```
qm set 100 --sshkey ~/.ssh/id_rsa.pub
```

Méthode fichier en YAML

Cette méthode n'est pas très explicite dans Proxmox, la documentation dit que le fichier doit être dans le répertoire snippets sans en indiquer son emplacement. Car par défaut, ce répertoire n'existe pas !

Les fichiers Cloud-Init sont ici dans l'arborescence système : **/var/lib/vz/snippets/user-data.yaml**

```
qm set 100 --cicustom "user=local:snippets/userconfig.yaml"
```

Il est aussi possible d'indiquer un point de montage réseau pour récupérer le fichier. Je ne l'ai pas testé et pour cela, c'est un renvoi sur un sujet du forum de Proxmox : cloud-init and cicustom

```
--cicustom "user=<volume>,network=<volume>,meta=<volume>"
```

Fichier basique pour mon home-lab :

```
#cloud-config

#Mise a jour de l'os
package_update: true
package_upgrade: true

#Installation de paquet
packages:
  - qemu-guest-agent
  - apt-transport-https
  - curl
  - ufw

#Ajout d'un utilisateur
users:
    - default
    - name: nidouille
      groups: [ sudo ]
      shell: /bin/bash
      homedir: /home/nidouille
      sudo: ['ALL=(ALL) NOPASSWD:ALL']
      ssh_authorized_keys: #Clefs ssh publiques

runcmd:
  - curl https://repository.rudder.io/tools/rudder-setup | bash -s setup-agent 7.0 [policy server hostname or ip]
  - wget https://repo.zabbix.com/zabbix/5.4/debian/pool/main/z/zabbix-release/zabbix-release_5.4-1+debian11_all.deb
  - apt update
  - apt install zabbix-agent -y
```

Bibliographie

Documentation officielle :

https://pve.proxmox.com/wiki/Cloud-Init_Support

https://pve.proxmox.com/pve-docs/qm.1.html

https://github.com/proxmox/pve-docs/blob/master/qm-cloud-init.adoc

Forum Proxmox :

https://forum.proxmox.com/threads/cloud-init-and-cicustom.58959/

Articles divers :

https://www.nextinpact.com/article/48588/cloud-init-a-decouverte-standard-post-installation-sous-linux