

Aksel

CAUBEL

RT3-App Dev-Cloud

Installation Proxmox

Utilisation de l'Idrac

Creds

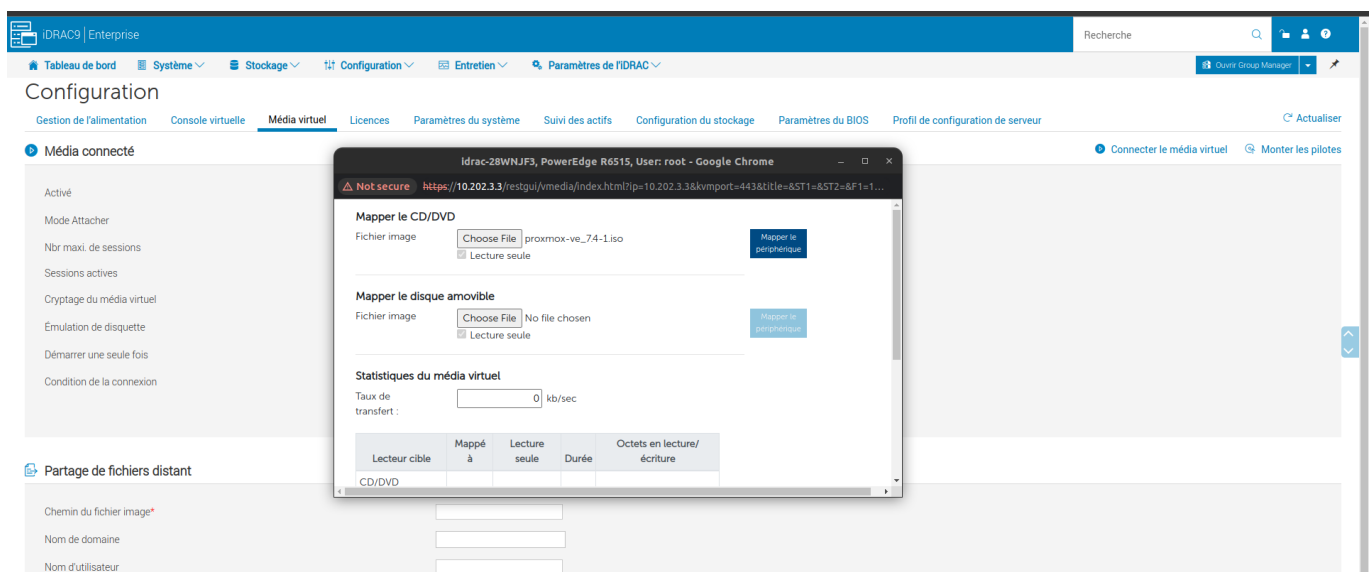
```
user = root  
mdp = root  
ip = 10.202.3.3
```

```
ip Proxmox : 10.202.3.33  
identifiant Proxmox : root  
mot de pass Proxmox : rootroot
```

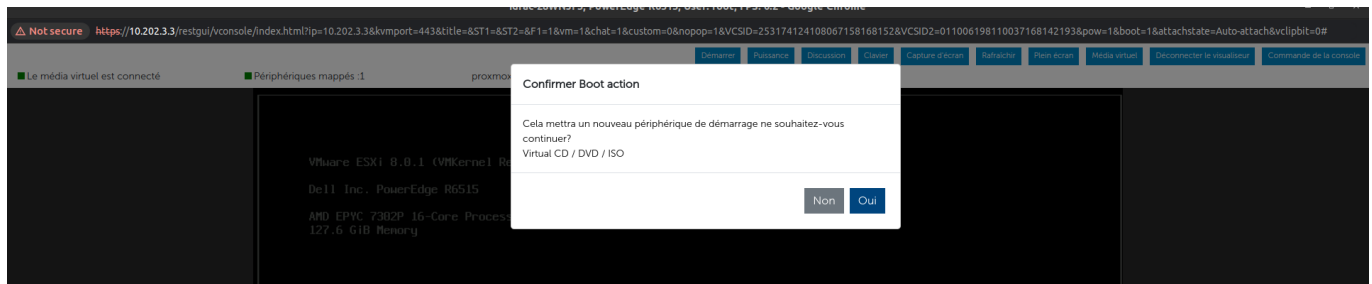
On va venir faire une installation via l'interface Idrac.

Pour ce faire on va entrer dans la partie **configuration** -> **Média Virtuel**. Le but est de faire un mapping de notre OS Proxmox *Utilisation de la version 7.4*

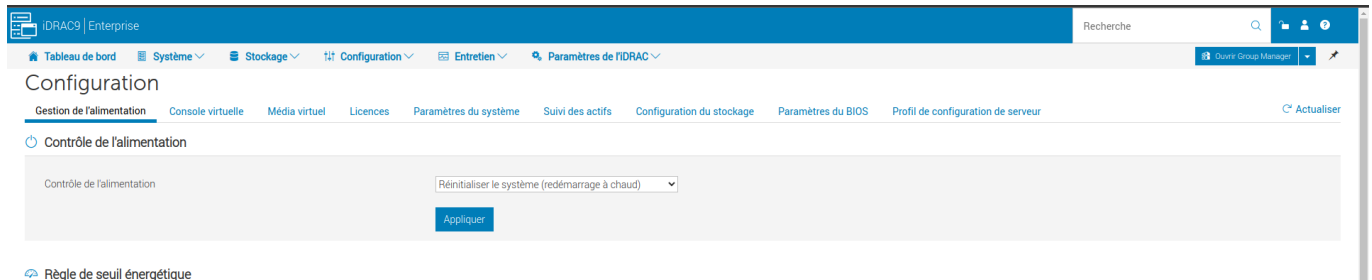
On vient ensuite **Connecter le média virtuel**



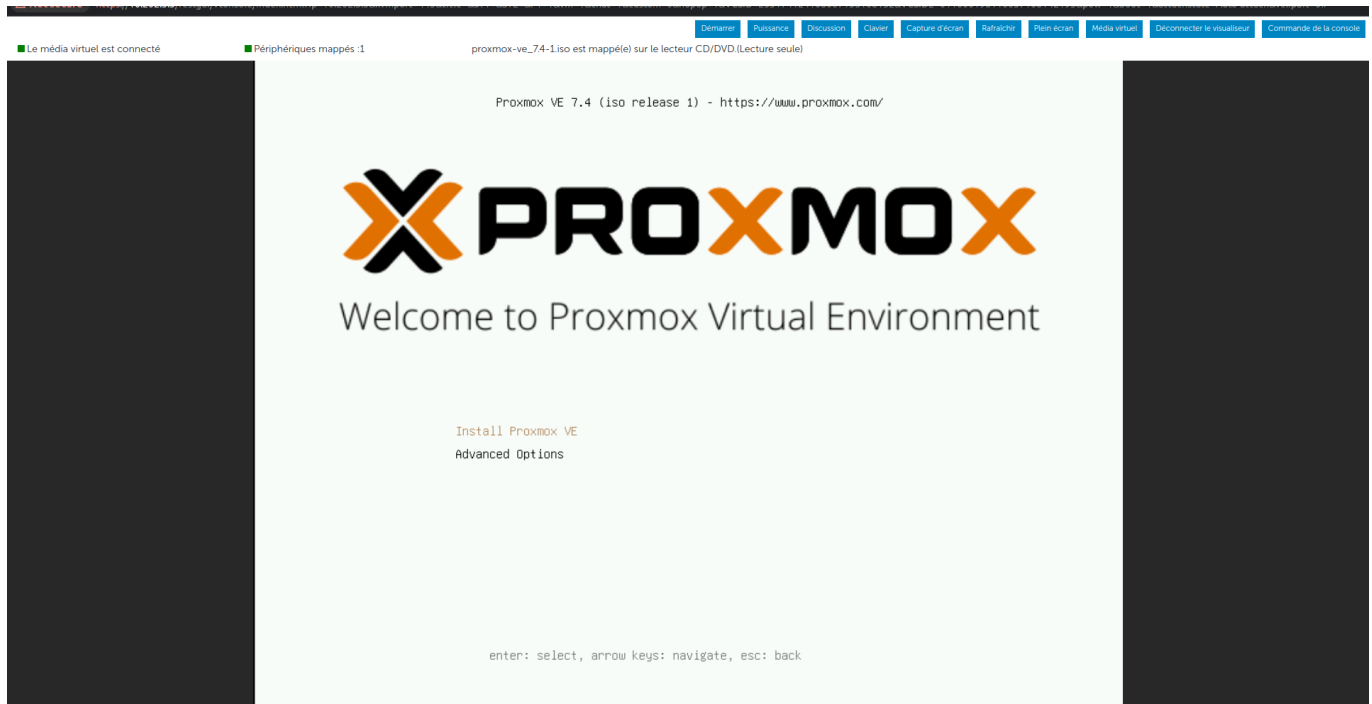
Une fois l'iso connecté on va choisir proxmox, on vient dans la console virtuelle dans **démarrer**->**Boot action**->**CD/DVD/ISO** pour qu'au prochain démarrage l'on puisse réaliser l'installation de **proxmox**.



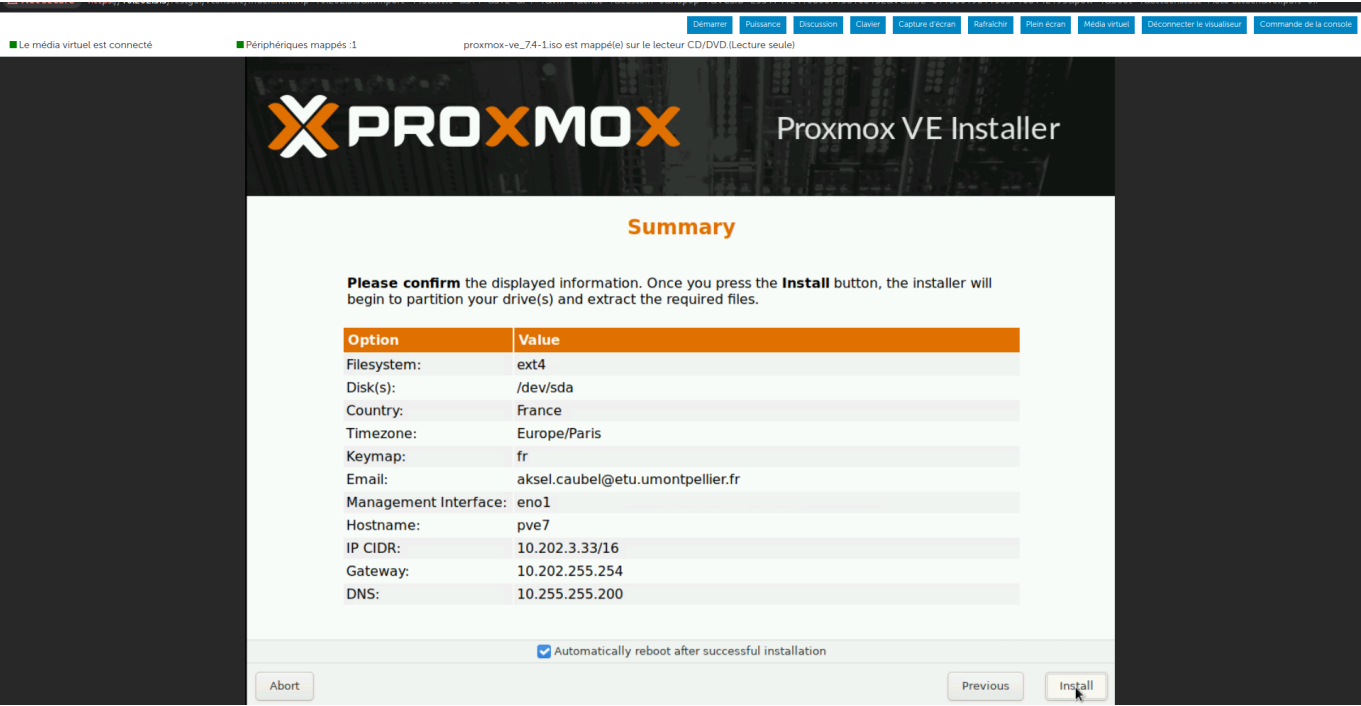
Pour faire le redémarrage à chaud à distance, on va revenir sur l'interface *Idrac* dans **configuration** > **Gestion de l'alimentation** et ensuite dans la partie *Contrôle de l'alimentation* choisir l'option **Réinitialiser le système (redémarrage à chaud)**



Maintenant nous pouvons commencer à suivre les instructions de Proxmox :



Une fois les instructions suivit on retrouve cette configuration dans notre cas.



L'interface graphique est maintenant disponible sur le port [8006](#)

Mise en place de GOAD sur Proxmox

Mise en place de l'architecture

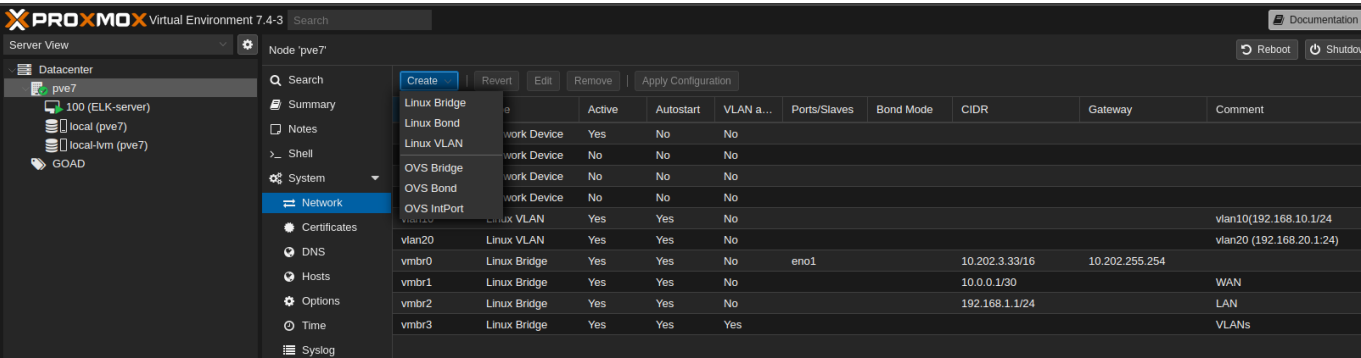
Source d'instruction

La configuration initial donner nous demande crée des interfaces réseaux supplémentaire :

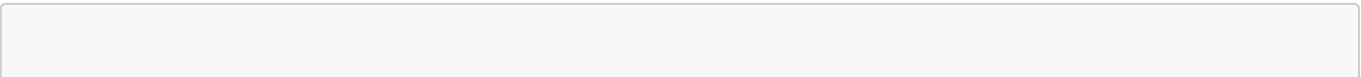
- 3 Bridge Linux
- 2 VLAN Linux

Pour ce faire, dans la partie **Datacenter** (volet de gauche) on va aller dans notre **Node** ici appelé **pve7** puis aller dans l'onglet **Système** -> **Network**.

Pour la création des bridges / VLANs, tous va se faire dans l'onglet **Create** :



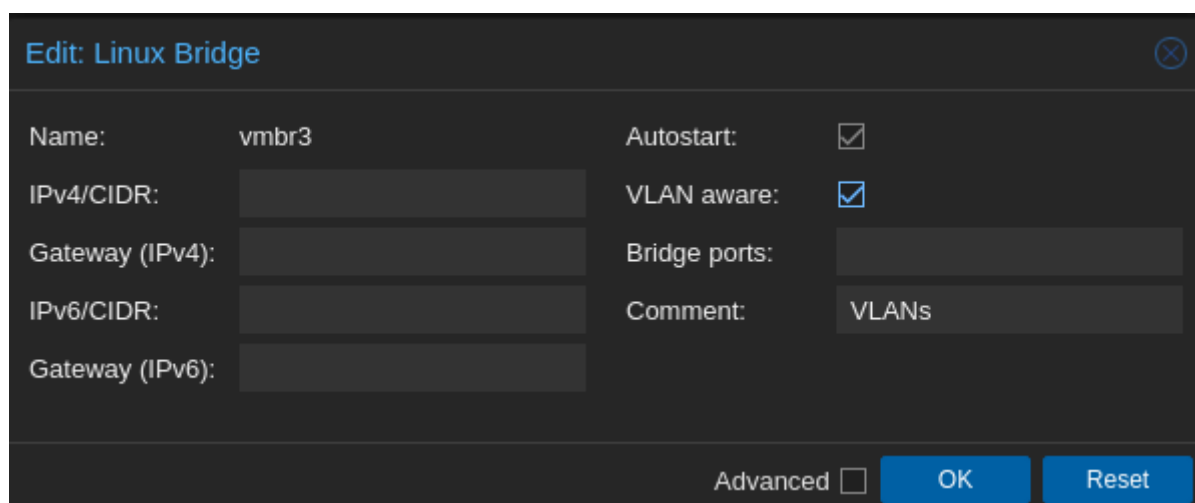
Voici un extrait des prérequis :



The network we will build will be in multiple part :

- 10.0.0.0/30 (10.0.0.1-10.0.0.2) : this will be the WAN network with only 2 ips, one for proxmox host, and the other one for pfsense
- 192.168.1.1/24 (192.168.1.1-192.168.1.254) : this will be the LAN network for the pfsense and the provisioning machine
- 192.168.10.1/24 (192.168.10.1-192.168.10.254) : VLAN1 for the GOAD's vm
- 192.168.20.1/24 (192.168.20.1-192.168.20.254) : VLAN2 for future projects
- 10.10.10.0/24 (10.10.10.0-10.10.10.254) : openvpn for vpn users (will be manage by pfsense later)

Création d'un Bridge :



Edit: Linux Bridge

Name: vmbr3 Autostart: ☒

IPv4/CIDR: VLAN aware: ☒

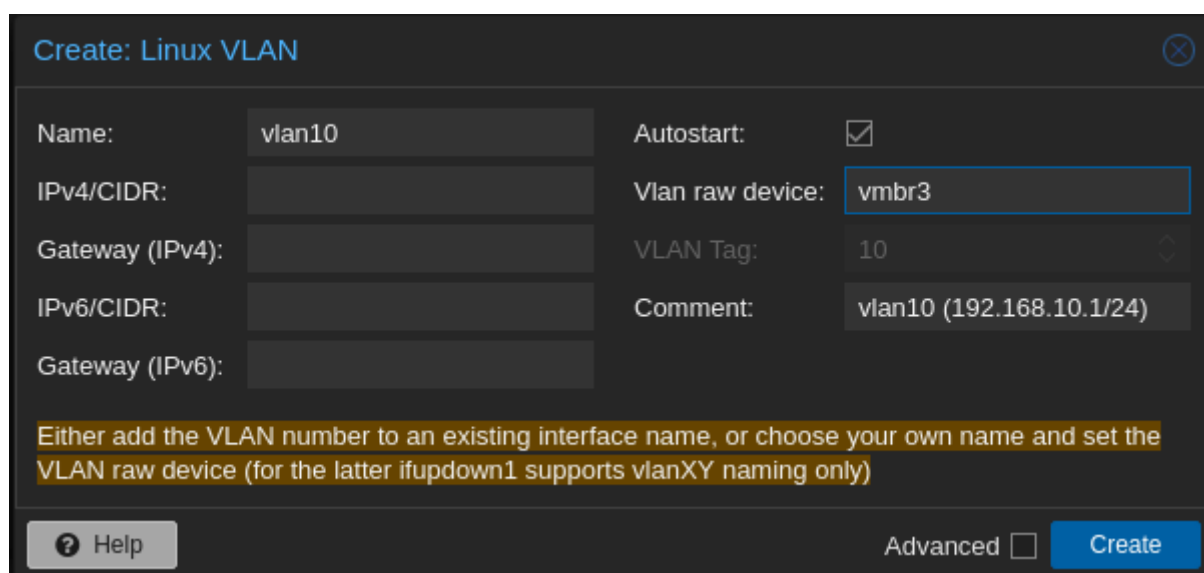
Gateway (IPv4): Bridge ports:

IPv6/CIDR: Comment: VLANs

Gateway (IPv6):

Advanced ☐ OK Reset

Création d'un VLAN :



Create: Linux VLAN

Name: vlan10 Autostart: ☒

IPv4/CIDR: Vlan raw device: vmbr3

Gateway (IPv4): VLAN Tag: 10

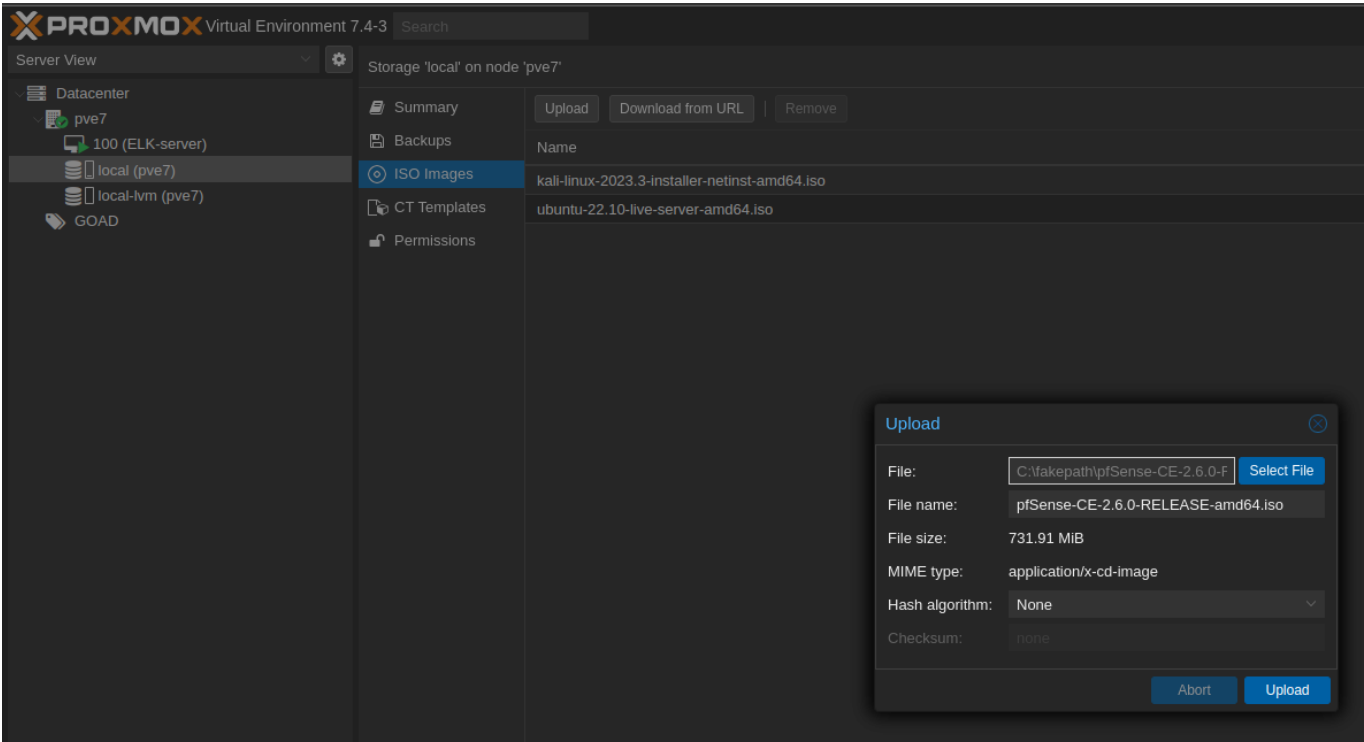
IPv6/CIDR: Comment: vlan10 (192.168.10.1/24)

Gateway (IPv6):

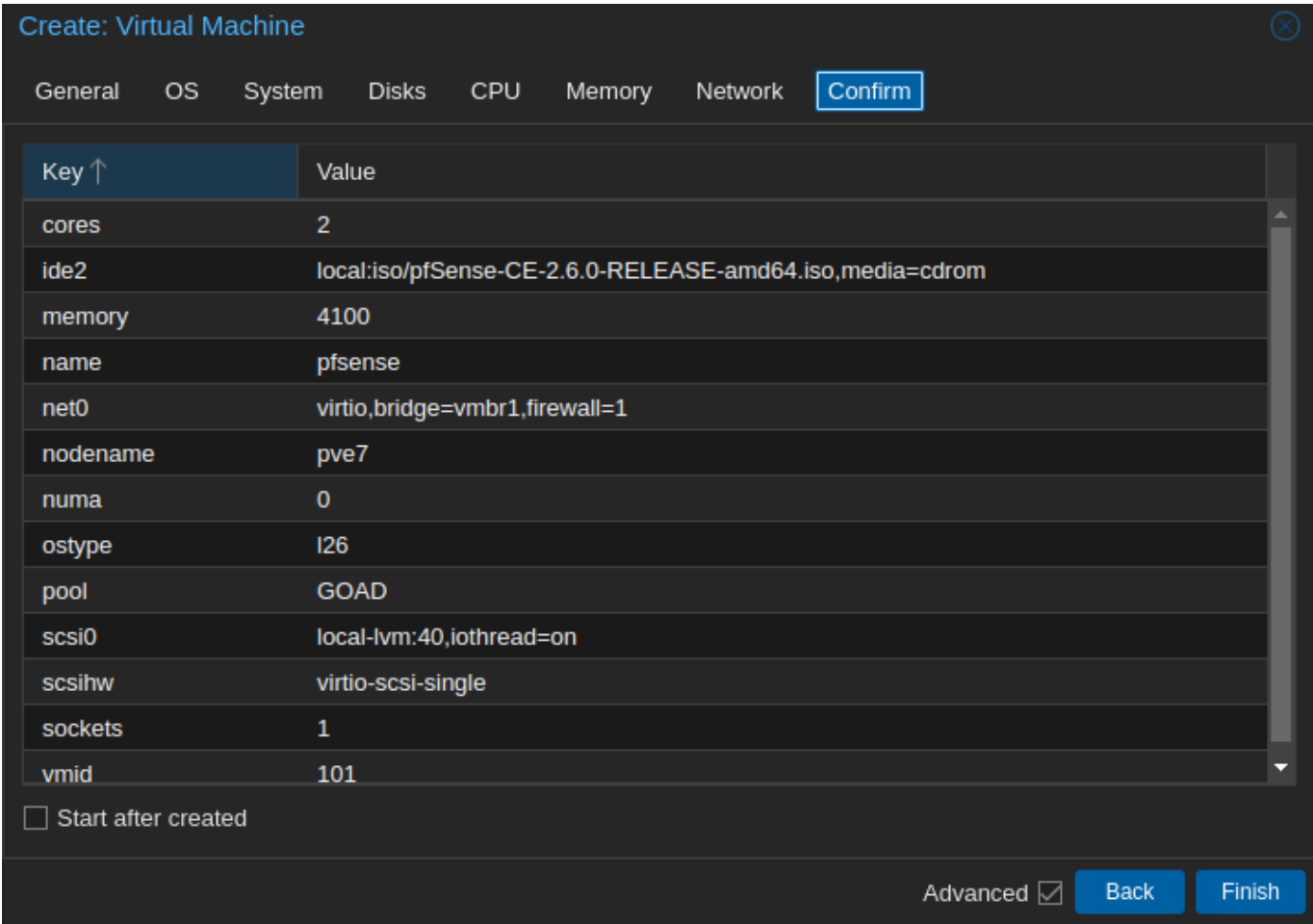
Either add the VLAN number to an existing interface name, or choose your own name and set the VLAN raw device (for the latter ifupdown1 supports vlanXY naming only)

? Help Advanced ☐ Create

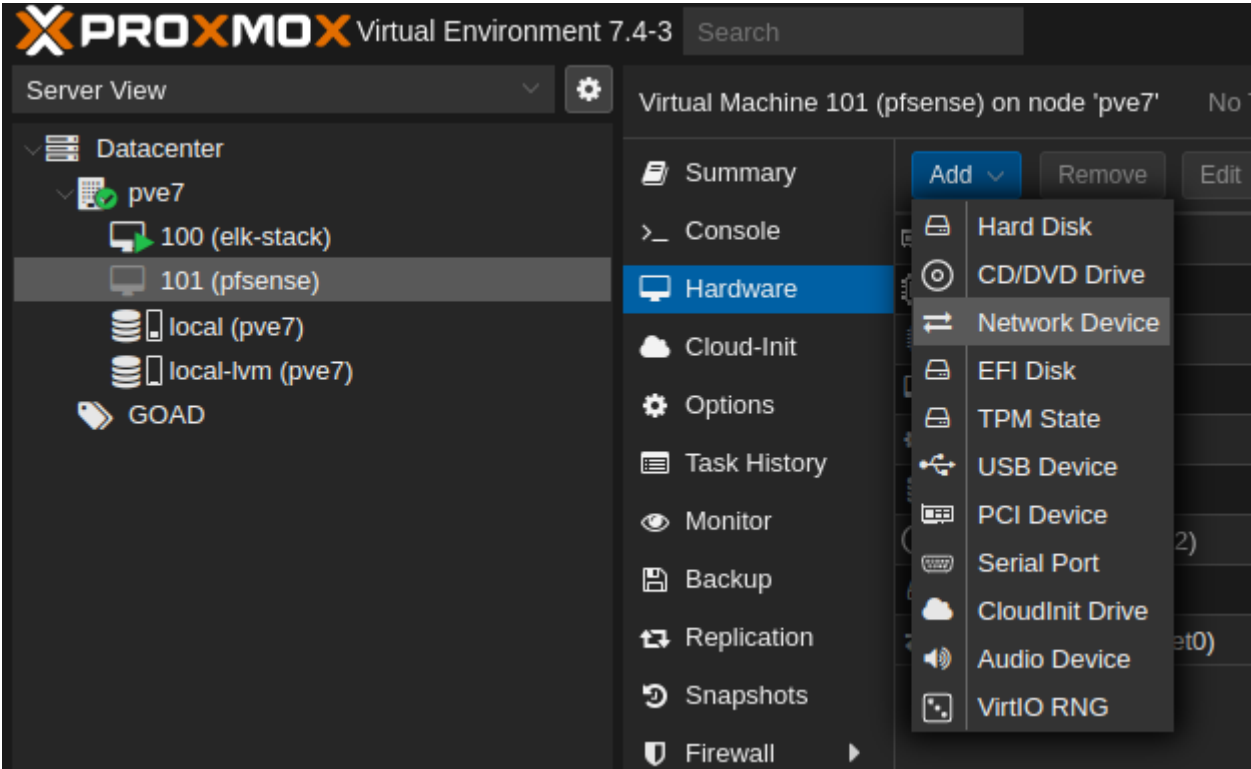
Par la suite il nous est demandé de faire l'installation d'une ISO PFSense. On va pouvoir procéder ainsi :



On va ensuite pouvoir créer notre première VM en commençant par *PfSense* **Ne pas démarrer la VM à sa création:**



Une fois que la VM est créée avec la configuration ci-dessus, on va venir lui rajouter des interfaces réseaux que nous avons précédemment créées de cette manière :



Le résultat attendu est d'avoir :

Network Device (net0)	virtio=EE:C9:AF:42:CC:C9,bridge=vibr1,firewall=1
Network Device (net1)	virtio=26:9D:C0:46:60:E6,bridge=vibr2,firewall=1
Network Device (net2)	virtio=A2:2E:3C:27:A5:D4,bridge=vibr3,firewall=1

Maintenant que *PfSense* est configuré on peut démarrer la machine.

Entrez dans la console depuis *Proxmox*

Suivez le guide d'installation jusqu'à l'option **reboot**

Configuration réseau

VLAN(s)

On ne souhaite pas configurer de VLAN :

```
Do VLANs need to be set up first?
If VLANs will not be used, or only for optional interfaces, it is typical to
say no here and use the webConfigurator to configure VLANs later, if required.
Should VLANs be set up now [y/n]? n
```

Interfaces

Précédemment nous avons attribuée les **devices** réseaux `vtnet{1,2,3}`. **Attention, dans PfSense le compteur est revenue a partir de 0. Nous aurons alors `vtnet1 -> vtnet0` et ainsi de suite.**

```
Enter the WAN interface name or 'a' for auto-detection
(vtnet0 vtnet1 vtnet2 or a): vtnet0

Enter the LAN interface name or 'a' for auto-detection
NOTE: this enables full Firewalling/NAT mode.
(vtnet1 vtnet2 a or nothing if finished): vtnet1

Enter the Optional 1 interface name or 'a' for auto-detection
(vtnet2 a or nothing if finished): vtnet2

The interfaces will be assigned as follows:

WAN -> vtnet0
LAN -> vtnet1
OPT1 -> vtnet2

Do you want to proceed [y/n]? y
```

Les choix fait précédemment nous menerons a la configuration suivante :

```
*** Welcome to pfSense 2.6.0-RELEASE (amd64) on pfSense ***

WAN (wan)      -> vtnet0      ->
LAN (lan)      -> vtnet1      -> v4: 192.168.1.1/24
OPT1 (opt1)    -> vtnet2      ->

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults   13) Update from console
5) Reboot system              14) Enable Secure Shell (sshd)
6) Halt system                15) Restore recent configuration
7) Ping host                  16) Restart PHP-FPM
8) Shell

Enter an option: 2
```

Configuration Réseau

```

Available interfaces:

1 - WAN (vtnet0 - dhcp, dhcp6)
2 - LAN (vtnet1 - static)
3 - OPT1 (vtnet2)

Enter the number of the interface you wish to configure: 1

Configure IPv4 address WAN interface via DHCP? (y/n) n

Enter the new WAN IPv4 address. Press <ENTER> for none:
> 10.0.0.2

Subnet masks are entered as bit counts (as in CIDR notation) in pfSense.
e.g. 255.255.255.0 = 24
     255.255.0.0   = 16
     255.0.0.0     = 8

Enter the new WAN IPv4 subnet bit count (1 to 32):
> 30

For a WAN, enter the new WAN IPv4 upstream gateway address.
For a LAN, press <ENTER> for none:
> 10.0.0.1

Configure IPv6 address WAN interface via DHCP6? (y/n) n

Enter the new WAN IPv6 address. Press <ENTER> for none:
>
Disabling IPv4 DHCPD...
Disabling IPv6 DHCPD...

Do you want to revert to HTTP as the webConfigurator protocol? (y/n) y

Please wait while the changes are saved to WAN...
  Reloading filter...
  Reloading routing configuration...
  DHCPD...
  Restarting webConfigurator...

The IPv4 WAN address has been set to 10.0.0.2/30

Press <ENTER> to continue.

```

Nous aurons alors le résultat de configuration suivant :

```

WAN (wan)      -> vtnet0      -> v4: 10.0.0.2/30
LAN (lan)      -> vtnet1      -> v4: 192.168.1.1/24
OPT1 (opt1)    -> vtnet2      ->

```

Une fois la configuration générique faite, on va venir faire une configuration plus précise pour l'interface LAN en faisant :

- Un changement d'adresse IP -> 192.168.1.2/24
 - Sans mettre de passerelle
 - Pas d'IPv6
- Un serveur DHCP (pool : 192.168.1.100 <-> 192.168.1.254)


```

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults  13) Update from console
5) Reboot system              14) Enable Secure Shell (sshd)
6) Halt system                15) Restore recent configuration
7) Ping host                  16) Restart PHP-FPM
8) Shell

```

Enter an option: 2

Available interfaces:

```

1 - WAN (vtnet0 - static)
2 - LAN (vtnet1 - static)
3 - OPT1 (vtnet2)

```

Enter the number of the interface you wish to configure: 2

Enter the new LAN IPv4 address. Press <ENTER> for none:

> 192.168.1.2

Subnet masks are entered as bit counts (as in CIDR notation) in pfSense.

```

e.g. 255.255.255.0 = 24
      255.255.0.0   = 16
      255.0.0.0     = 8

```

Enter the new LAN IPv4 subnet bit count (1 to 32):

> 24

For a WAN, enter the new LAN IPv4 upstream gateway address.

For a LAN, press <ENTER> for none:

>

Enter the new LAN IPv6 address. Press <ENTER> for none:

>

Do you want to enable the DHCP server on LAN? (y/n) y

Enter the start address of the IPv4 client address range: 192.168.1.100

Enter the end address of the IPv4 client address range: 192.168.1.254

The IPv4 LAN address has been set to 192.168.1.2/24

You can now access the webConfigurator by opening the following URL in your web browser:

<http://192.168.1.2/>

Press <ENTER> to continue.

KUM Guest - Netgate Device ID: e39fa5b38524b178733c

*** Welcome to pfSense 2.6.0-RELEASE (amd64) on pfSense ***

```

WAN (wan)      -> vtnet0      -> v4: 10.0.0.2/30
LAN (lan)      -> vtnet1      -> v4: 192.168.1.2/24
OPT1 (opt1)    -> vtnet2      ->

```

Configuration suite en GUI

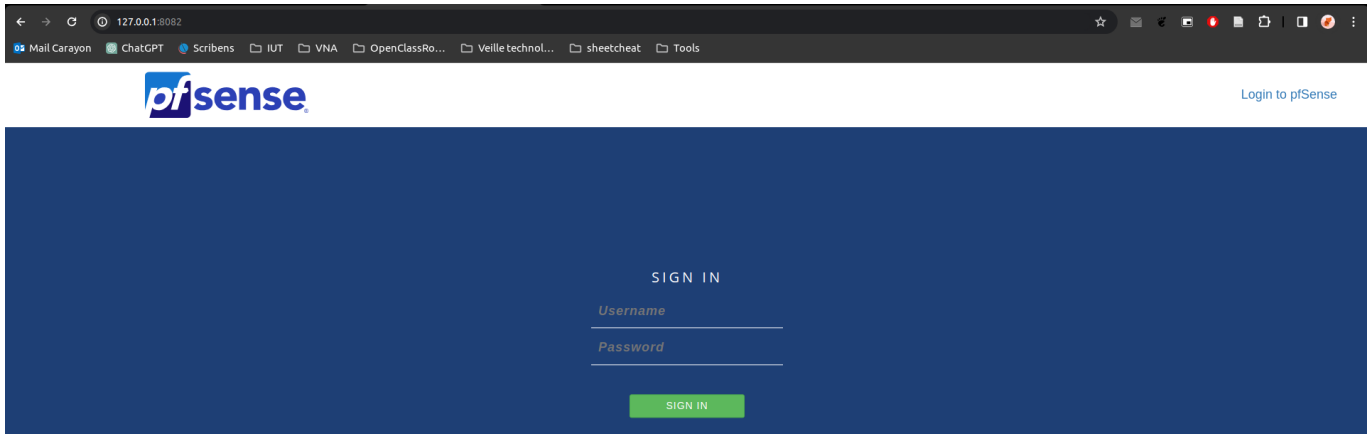
Afin d'avoir accès à l'interface graphique sur notre poste nous devons faire un *port-forwarding* de l'host 192.168.1.2:80 vers notre machine avec un port client quelconque (*ici le 8082*)

Pour ce faire on viens faire un **ssh-L**

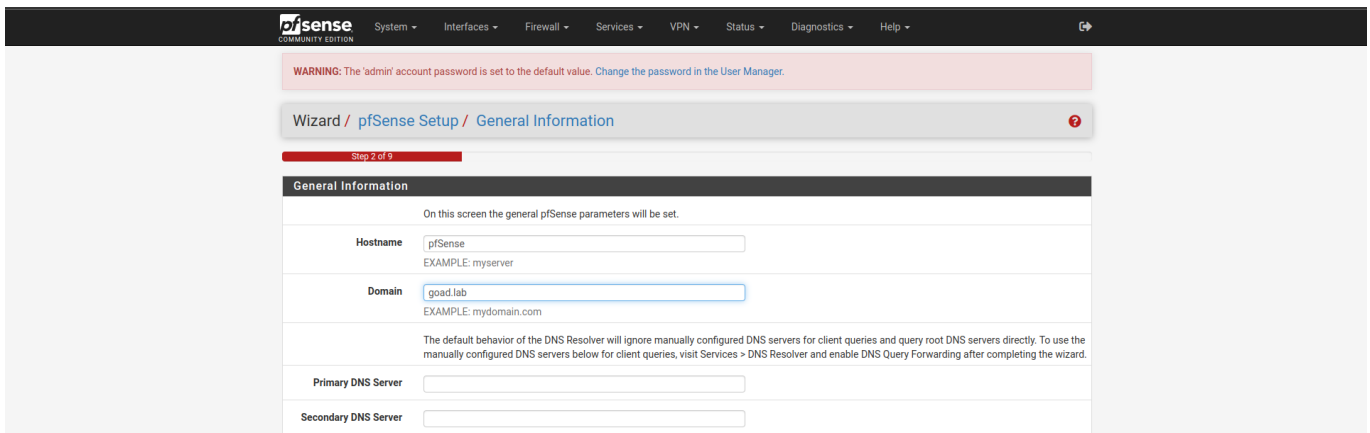
```
ssh-L 8082:192.168.1.2:80 root@10.202.3.33 #Ip proxmox
```

Interface WEB

User: admin / passwd : pfsense



Après connexion appuyer sur **Next** deux fois pour arriver sur cette page :

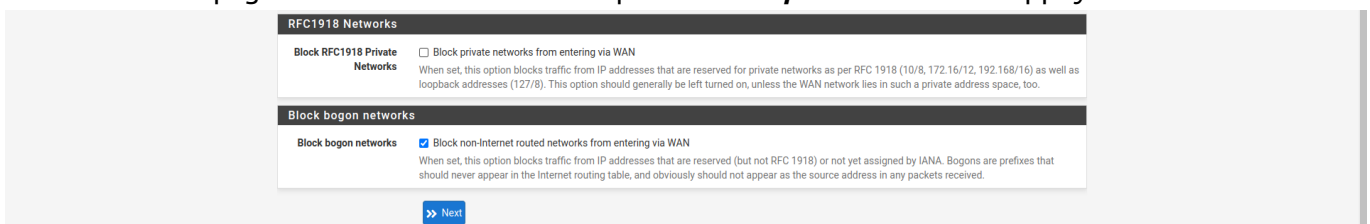


Changer le Domain présent pour **goad . lab**

Pour la configuration **NTP** vous pouvez le laisser par défaut et ensuite entrez **NEXT**.

L'interface WAN **doit être** laissée par défaut.

Sur cette même page vous devais enlever le bloque **RFC1918 private network**. Appuyer sur **NEXT**.




Laissez l'interface LAN comme il vous est affichée. **NEXT**

Changez le mot de passe admin (ici on a choisit la sécurité 😊 => passwd = admin)

Dans l'onglet **System/Advanced/Networking** en bas de page dans la partie **Network Interfaces** on va venir cocher la première case **Hardware Checksum Offloading**

Network Interfaces	
Hardware Checksum Offloading	<input checked="" type="checkbox"/> Disable hardware checksum offload Checking this option will disable hardware checksum offloading. Checksum offloading is broken in some hardware, particularly some Realtek and some specific NICs. This will take effect after a machine reboot or re-configuration.
Hardware TCP Segmentation Offloading	<input checked="" type="checkbox"/> Disable hardware TCP segmentation offload Checking this option will disable hardware TCP segmentation offloading (TSO) which may impact performance with some specific NICs. This will take effect after a machine reboot.
Hardware Large Receive Offloading	<input checked="" type="checkbox"/> Disable hardware large receive offload Checking this option will disable hardware large receive offloading (LRO). This will take effect after a machine reboot.
hn ALTQ support	<input checked="" type="checkbox"/> Enable the ALTQ support for hn NICs. Checking this option will enable the ALTQ support for hn NICs. The ALTQ support will handle traffic. This will take effect after a machine reboot.
ARP Handling	<input type="checkbox"/> Suppress ARP messages This option will suppress ARP log messages when multiple interfaces reside on the same network.
Reset All States	<input type="checkbox"/> Reset all states if WAN IP Address changes This option resets all states when a WAN IP Address changes instead of only the states related to the changed IP address.

 Save

Lors de la sauvegarde de configuration, acceptez le **Reboot**

SetUP Fire-Wall PFSense

On vient ajouter une règle pour accepter le trafic **HTTP***(80)***** :

Source

☐ Invert match
Single host or alias
10.0.0.1

Display Advanced

The **Source Port Range** for a connection is typically random and almost never equal to the destination port. In most cases this setting must remain at its default value, any.

Destination

☐ Invert match
LAN address
Destination Address

Destination Port Range
HTTP (80)
From
Custom
To
HTTP (80)
Custom

Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.

Et l'on vient bloquer en dernier tous le reste du traffic.

Floating
WAN
LAN

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	✗ 0 / 7 KiB	*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks	
<input type="checkbox"/>	✓ 0 / 0 B	IPv4 TCP	10.0.0.1	*	LAN address	80 (HTTP)	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 TCP	*	*	*	*	*	none			

Add
Add
Delete
Save
Separator

SetUP IpTables

Sur notre connexion **SSH** précédemment crée (*cette pour le port-forwarding*), on va venir en tant que user root faire :

```
# activate ipforward
echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
# allow icmp to avoid ovh monitoring reboot the host
iptables -t nat -A PREROUTING -i vmbr0 -p icmp -j ACCEPT
# allow ssh
iptables -t nat -A PREROUTING -i vmbr0 -p tcp --dport 22 -j ACCEPT
# allow proxmox web
iptables -t nat -A PREROUTING -i vmbr0 -p tcp --dport 8006 -j ACCEPT
# redirect all to pfsense
iptables -t nat -A PREROUTING -i vmbr0 -j DNAT --to 10.0.0.2
# add SNAT WAN -> public ip
iptables -t nat -A POSTROUTING -o vmbr0 -j SNAT -s 10.0.0.0/30 --to-source MYPUBLICIP_HERE
```

On va également crée une sauvegarde des règles (**Sachant qu'IpTables perd sa configuration a chaque restart**):

```
iptables-save | sudo tee /etc/network/save-iptables
```

Pour que la configuration se mette a jour dès que la machine démarre, on va venir mettre la configuration suivante a la fin du fichier `/etc/network/interfaces`

```
post-up iptables-restore < /etc/network/save-iptables
```

Setup VLAN(s)

Dans l'onglet `Interfaces/Interface Assignments/VLANs` on vient ajouter un VLAN et mettre la configuration suivant :

Interfaces / VLANs / Edit

VLAN Configuration

Parent Interface

vtnet2 (a2:2e:3c:27:a5:d4) - opt1

Only VLAN capable interfaces will be shown.

VLAN Tag

10

802.1Q VLAN tag (between 1 and 4094).

VLAN Priority

0

802.1Q VLAN Priority (between 0 and 7).

Description

VLAN10

A group description may be entered here for administrative reference (not parsed).





Save



On fait pareil pour le VLAN 20 pour obtenir cette configuration final :

Interfaces / VLANs

Interface AssignmentsInterface GroupsWirelessVLANsQinQsPPPsGREsGIFsBridgesLAGGs

VLAN Interfaces

Interface	VLAN tag	Priority	Description	Actions
vtnet2 (opt1)	10		VLAN10	 
vtnet2 (opt1)	20		VLAN20	 

  Add

Une fois les VLANs créés, on va leur assigner une adresse IP. Pour cela on vient dans l'onglet Interface Assignments, on y rajoute le VLAN10 et le VLAN20 :

Interfaces / Interface Assignments

Interface Assignments

Interface Groups

Wireless

VLANs

QinQs

PPPs

GREs

GIFs

Bridges

LAGGs

Interface	Network port
WAN	vtnet0 (ee:c9:af:42:cc:c9)
LAN	vtnet1 (26:9d:c0:46:60:e6)
OPT1	vtnet2 (a2:2e:3c:27:a5:d4)
Available network ports:	VLAN 10 on vtnet2 - opt1 (VLAN10)

Save

Delete

Add

Et ensuite les configurer en cliquant sur leur nom d'interface :

Interfaces / OPT2 (vtnet2.10)

General Configuration

Enable

☒ Enable interface

Description

VLAN10

Enter a description (name) for the interface here.

IPv4 Configuration Type

Static IPv4

IPv6 Configuration Type

None

MAC Address

xx:xx:xx:xx:xx:xx

The MAC address of a VLAN interface must be set on its parent interface

MTU

If this field is blank, the adapter's default MTU will be used. This is typically 1500 bytes but can vary in some circumstances.

MSS

If a value is entered in this field, then MSS clamping for TCP connections to the value entered above minus 40 for IPv4 (TCP/IPv4 header size) and minus 60 for IPv6 (TCP/IPv6 header size) will be in effect.

Speed and Duplex

Default (no preference, typically autoselect)

Explicitly set speed and duplex mode for this interface.
WARNING: MUST be set to autoselect (automatically negotiate speed) unless the port this interface connects to has its speed and duplex forced.

Static IPv4 Configuration

IPv4 Address

192.168.10.1

/ 24

IPv4 Upstream gateway

None

Add a new gateway

If this interface is an Internet connection, select an existing Gateway from the list or add a new one using the "Add" button.
On local area network interfaces the upstream gateway should be "none".
Selecting an upstream gateway causes the firewall to treat this interface as a WAN type interface.

Reserved Networks

Block private networks and loopback addresses

☐

Blocks traffic from IP addresses that are reserved for private networks per RFC 1918 (10/8, 172.16/12, 192.168/16) and unique local addresses per RFC 4193 (fc00::/7) as well as loopback addresses (127/8). This option should generally be turned on, unless this network interface resides in such a private address space, too.

Block bogon networks

☒

Blocks traffic from reserved IP addresses (but not RFC 1918) or not yet assigned by IANA. Bogons are prefixes that should never appear in the Internet routing table, and so should not appear as the source address in any packets received.
This option should only be used on external interfaces (WANs), it is not necessary on local interfaces and it can potentially block required local traffic.
Note: The update frequency can be changed under System > Advanced, Firewall & NAT settings.

Save

On configurera de la même manière le VLAN20 en assignant l'adresse IP suivant : 192.168.20.1.

Attention de ne pas oublier de renseigner le masque de sous-réseau !

Ajout du DHCP Serveur

La configuration commence dans l'onglet **Services/DHCP serveur**

On **activera** le serveur DHCP et ensuite, le seul changement se trouvera dans la **Range ip** que l'on souhaite attribuer. Ici on ira de 192.168.X.100 <-> 192.168.X.254. **En remplaçant X par le numéro de VLAN.**

The screenshot shows the configuration page for the DHCP server on the VLAN10 interface. The interface tabs at the top are WAN, LAN, VLAN10 (selected), and VLAN20. The 'General Options' section contains the following settings:

- Enable:** ☒ Enable DHCP server on VLAN10 interface
- BOOTP:** ☐ Ignore BOOTP queries
- Deny unknown clients:** (Dropdown menu)
When set to **Allow all clients**, any DHCP client will get an IP address within this scope/range on this interface. If set to **Allow known clients from any interface**, any DHCP client with a MAC address listed on **any** scope(s)/interface(s) will get an IP address. If set to **Allow known clients from only this interface**, only MAC addresses listed below (i.e. for this interface) will get an IP address within this scope/range.
- Ignore denied clients:** ☐ Denied clients will be ignored rather than rejected.
This option is not compatible with failover and cannot be enabled when a Failover Peer IP address is configured.
- Ignore client identifiers:** ☐ If a client includes a unique identifier in its DHCP request, that UID will not be recorded in its lease.
This option may be useful when a client can dual boot using different client identifiers but the same hardware (MAC) address. Note that the resulting server behavior violates the official DHCP specification.
- Subnet:** 192.168.10.0
- Subnet mask:** 255.255.255.0
- Available range:** 192.168.10.1 - 192.168.10.254
- Range:** From To

Configuration du VLAN FireWall

La configuration commande dans l'onglet **Firewall > alias/IP**

On viendra crée une règle avec la configuration suivante :

Firewall / Aliases / Edit

Properties

Name

INTERNAL

The name of the alias may only consist of the characters "a-z, A-Z, 0-9 and _".

Description

A description may be entered here for administrative reference (not parsed).

Type

Network(s)

Network(s)

Hint

Networks are specified in CIDR format. Select the CIDR mask that pertains to each entry. /32 specifies a single IPv4 host, /128 specifies a single IPv6 host, /24 specifies 255.255.255.0, /64 specifies a normal IPv6 network, etc. Hostnames (FQDNs) may also be specified, using a /32 mask for IPv4 or /128 for IPv6. An IP range such as 192.168.1.1-192.168.1.254 may also be entered and a list of CIDR networks will be derived to fill the range.

Network or FQDN	192.168.1.1	/	16	LAN + VLAN	Delete
	10.0.0.1	/	30	WAN	Delete
	10.10.10.0	/	24	VPN	Delete

Save

+ Add Network

On vient terminer la configuration par (On doit autoriser tous les protocoles):

Firewall / Rules / LAN

The firewall rule configuration has been changed.
The changes must be applied for them to take effect.

Apply Changes

Floating

WAN

LAN

VLAN10

VLAN20

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	✓ 4 / 1.18 MiB	*	*	*	LAN Address	80	*	*		Anti-Lockout Rule	⚙
OUT											
<input type="checkbox"/>	✓ 0 / 0 B	IPv4+6 TCP	LAN net	*	*	*	*	none			🔗🔧📄🔕🗑
Default deny											
<input type="checkbox"/>	✗ 0 / 0 B	IPv4+6 *	*	*	*	*	*	none		deny	🔗🔧📄🔕🗑

↑ Add

↓ Add

Delete

Save

+ Separator

Et en ajoutant dans chaque onglet firewall des VLANs :

Source

Source

☐ Invert match

VLAN10 net

Source Address

Display Advanced

The **Source Port Range** for a connection is typically random and almost never equal to the destination port. In most cases this setting must remain at its default value, any.

Destination

Destination

☒ Invert match

Single host or alias

INTERNAL

Destination Port Range

any

From

Custom

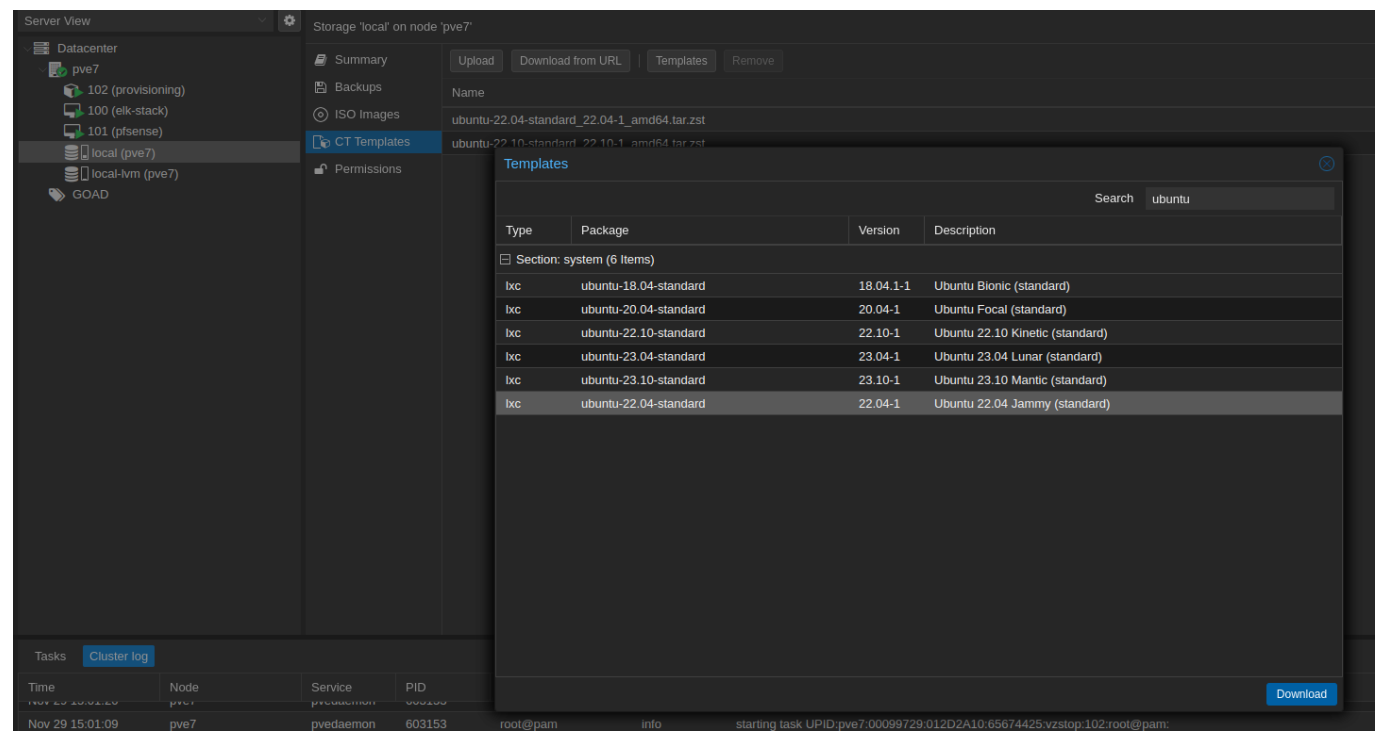
To

any

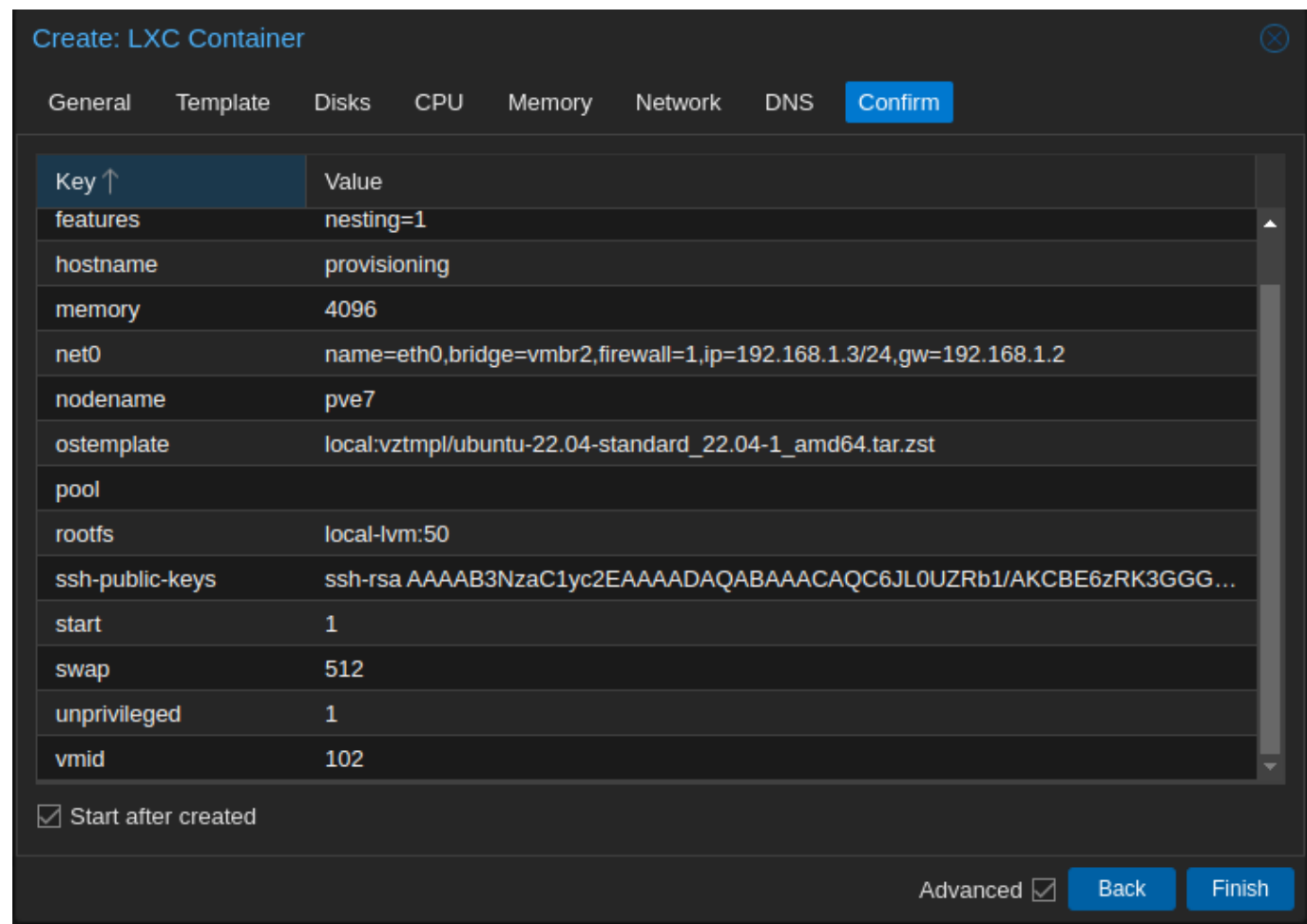
Custom

Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.

On vient installer la template pour un **Ubuntu 22.04** (Sachant que la 22.10 préquonisé n'est plus soutenu)



Une fois l'installation effectuée, on vient créer un conteneur avec comme **hostname** : provisioning que l'on vient configurer avec une clef publique **SSH**



On vient rajouter une règle pour permettre le ssh :

Interface	WAN		
Choose the interface from which packets must come to match this rule.			
Address Family	IPv4		
Select the Internet Protocol version this rule applies to.			
Protocol	TCP		
Choose which IP protocol this rule should match.			
Source			
Source	<input type="checkbox"/> Invert match	Single host or alias	10.0.0.1 /
<input type="button" value="Display Advanced"/> <p>The Source Port Range for a connection is typically random and almost never equal to the destination port. In most cases this setting must remain at its default value, any.</p>			
Destination			
Destination	<input type="checkbox"/> Invert match	Single host or alias	192.168.1.3 /
Destination Port Range	SSH (22)	SSH (22)	
From	Custom	To	Custom
Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.			

commande ssh :

```
ssh -J root@10.202.3.33 root@192.168.1.3 # -J = proxyJumper | -J @proxy
@dest
```

Préparatio au provisionnig

On va maintenant pouvoir faire toutes les installations requisent :

```
apt update && apt upgrade
apt install git vim tmux curl gnupg software-properties-common mkisofs
```

Installation Packer

Guide d'installation

```
curl -fsSL https://apt.releases.hashicorp.com/gpg | apt-key add -
apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com
$(lsb_release -cs) main"
apt update && apt install packer
```

Vérification

```
root@provisioning:~# packer -v
>>> 1.9.4
```

Installation Terraform

Guide d'installation

```
# Install the HashiCorp GPG key.
wget -O- https://apt.releases.hashicorp.com/gpg | \
gpg --dearmor | \
tee /usr/share/keyrings/hashicorp-archive-keyring.gpg

# Verify the key's fingerprint.
gpg --no-default-keyring \
--keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg \
--fingerprint

# add terraform sourcelist
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] \
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
tee /etc/apt/sources.list.d/hashicorp.list

# update apt and install terraform
apt update && apt install terraform
```

Vérification :

```
root@provisioning:~# terraform -v
Terraform v1.6.4
on linux_amd64
```

Installation Ansible

```
apt install python3-pip
python3 -m pip install --upgrade pip
python3 -m pip install ansible-core==2.12.6
python3 -m pip install pywinrm
```

Vérification

```
root@provisioning:~# ansible-galaxy --version
ansible-galaxy [core 2.12.6]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/local/lib/python3.10/dist-
packages/ansible
  ansible collection location =
/root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/local/bin/ansible-galaxy
  python version = 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0]
```

```
jinja version = 3.1.2
libyaml = True

root@provisioning:~# ansible --version
ansible [core 2.12.6]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules',
  '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/local/lib/python3.10/dist-
  packages/ansible
  ansible collection location =
  /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/local/bin/ansible
  python version = 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0]
  jinja version = 3.1.2
  libyaml = True
```

Packer

Installation des ISO

La première des choses est de télécharger les ISOs.

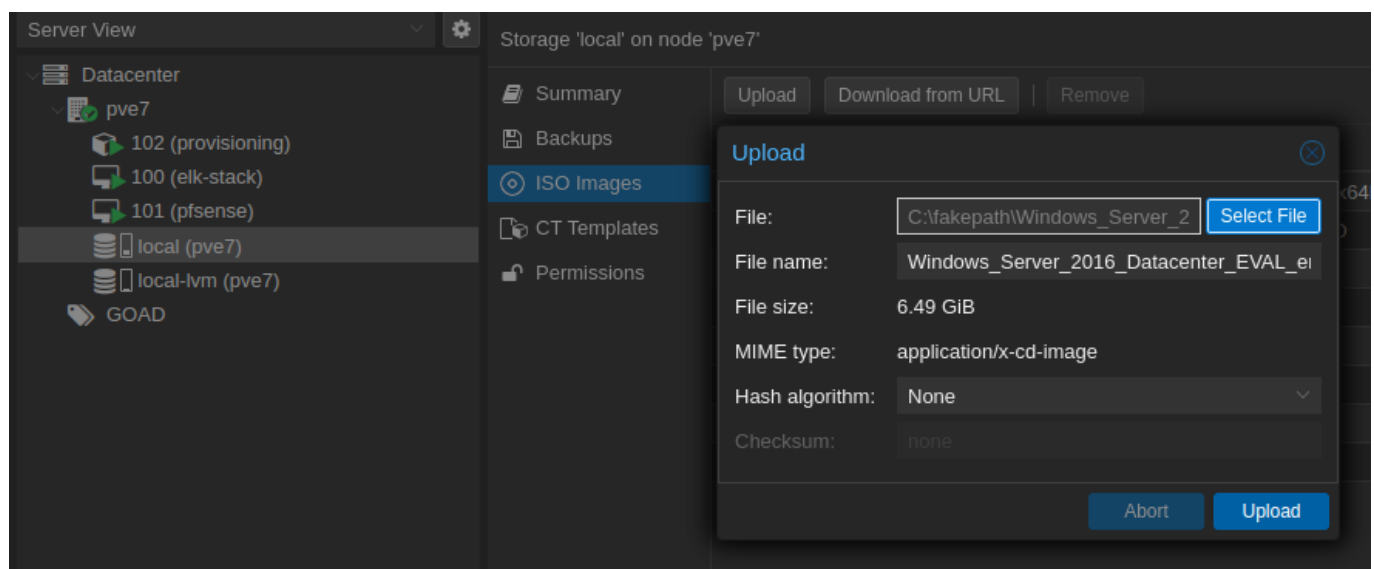
Attention, les ISOs doivent avoir respectivement les noms suivant afin d'être reconnus :

windows_server2019_x64FREE_en-us.iso windows_server_2016_14393.0_eval_x64.iso

Lien téléchargement :

[Window_server_2019](#) | [Windows_server_2016](#)

Une fois l'installation faite on va pouvoir les mettre sur proxmox via de cette manière :



Cloud-Base Init

L'installation de **CloudBase-Init** permettra de lancer ce service sur chaque VM-Windows en prenant les configuration de proxmox et changer les ip ainsi que d'autre configuration spécifique pour chaque VM.

```
root@provisioning:~/GOAD# cd /root/GOAD/packer/proxmox/scripts/sysprep
wget https://cloudbase.it/downloads/CloudbaseInitSetup_Stable_x64.msi
```

Create User

Sur le Shell de proxmo, on vient créer un user :

```
pveum useradd infra_as_code@pve
pveum passwd infra_as_code@pve
```

On vient lui crée un rôle :

```
pveum roleadd Packer -privs "VM.Config.Disk VM.Config.CPU VM.Config.Memory
Datastore.AllocateTemplate Datastore.Audit Datastore.AllocateSpace
Sys.Modify VM.Config.Options VM.Allocate VM.Audit VM.Console
VM.Config.CDRom VM.Config.Cloudinit VM.Config.Network VM.PowerMgmt
VM.Config.HWType VM.Monitor"
```

Et pour finir on lui associe ce rôle :

```
pveum acl modify / -user 'infra_as_code@pve' -role Packer
```

Préparation des variables Terraform

La première des étapes est de copier le fichier template pour avoir une sauvegarde et surtout d'enlever l'extension **.template** pour que **Terraform** puisse le prendre en compte.

```
cd /root/GOAD/packer/proxmox/
cp config.auto.pkrvars.hcl.template config.auto.pkrvars.hcl
```

Dans ce fichier on retrouvera :

```
proxmox_url          = "https://proxmox:8006/api2/json"
proxmox_username     = "user"
proxmox_token        = "changeme"
proxmox_skip_tls_verify = "true"
```

```
proxmox_node      = "mynode"  
proxmox_pool      = "mypool"  
proxmox_storage   = "local"
```

Une fois modifié, il donne dans notre cas :

```
proxmox_url       = "https://10.202.3.33:8006/api2/json"  
proxmox_username  = "infra_as_code@pve"  
proxmox_password  = "infra"  
proxmox_skip_tls_verify = "true"  
proxmox_node      = "pve7"  
proxmox_pool      = "GOAD"  
proxmox_storage   = "local"
```

Préparation des fichiers ISO

Packer ne pouvant pas créer lecteur de disque nous devons créer des fichiers ISO. Pour cela on peut utiliser directement le script fournis par GOAD :

```
cd /root/GOAD/packer/proxmox/  
./build_proxmox_iso.sh
```

Une fois cela fait on vient mettre tous ces fichiers sur Proxmox via le shell :

```
scp  
root@192.168.1.3:/root/GOAD/packer/proxmox/iso/scripts_withcloudinit.iso  
/var/lib
```

On passe sur la machine Proxmox

On télécharge le fichier virtio-win.iso

```
ssh goadproxmox  
cd /var/lib/vz/template/iso  
wget https://fedorapeople.org/groups/virt/virtio-win/direct-  
downloads/stable-virtio/virtio-win.iso
```

Configuration de l'ordinateur

Maintenant que la configuration est faite on vient lancer packer.

Attention à changer dans le fichier des windows server le format de disk doit être **raw**.

On vient créer le fichier suivant dans le répertoire **/root/GOAD/packer/proxmox/** :

packer.pkr.hcl

```
packer {
  required_plugins {
    proxmox = {
      version = ">= 1.1.2"
      source  = "github.com/hashicorp/proxmox"
    }
  }
}

source "proxmox-iso" "windows" {
  additional_iso_files {
    device      = "sata3"
    iso_checksum = "${var.autounattend_checksum}"
    iso_storage_pool = "local"
    iso_url      = "${var.autounattend_iso}"
    unmount      = true
  }
  additional_iso_files {
    device      = "sata4"
    iso_file     = "local:iso/virtio-win.iso"
    unmount      = true
  }

  additional_iso_files {
    device      = "sata5"
    iso_file     = "local:iso/scripts_withcloudinit.iso"
    unmount      = true
  }

  cloud_init          = true
  cloud_init_storage_pool = "${var.proxmox_storage}"
  communicator        = "winrm"
  cores               = "${var.vm_cpu_cores}"
  disks {
    disk_size      = "${var.vm_disk_size}"
    format         = "qcow2"
    storage_pool    = "${var.proxmox_storage}"
    type           = "sata"
  }
  insecure_skip_tls_verify = "${var.proxmox_skip_tls_verify}"
  iso_file                 = "${var.iso_file}"
  memory                   = "${var.vm_memory}"
  network_adapters {
    bridge = "vbr3"
    model  = "virtio"
    vlan_tag = "10"
  }
  node          = "${var.proxmox_node}"
  os            = "${var.os}"
  password      = "${var.proxmox_password}"
  pool          = "${var.proxmox_pool}"
  proxmox_url    = "${var.proxmox_url}"
}
```

```

sockets                = "${var.vm_sockets}"
template_description    = "${var.template_description}"
template_name           = "${var.vm_name}"
username                = "${var.proxmox_username}"
vm_name                 = "${var.vm_name}"
winrm_insecure          = true
winrm_no_proxy          = true
winrm_password          = "${var.winrm_password}"
winrm_timeout           = "30m"
winrm_use_ssl           = true
winrm_username          = "${var.winrm_username}"
}

build {
  sources = ["source.proxmox-iso.windows"]

  provisioner "powershell" {
    elevated_password = "vagrant"
    elevated_user     = "vagrant"
    scripts            = ["${path.root}/scripts/sysprep/cloudbase-init.ps1"]
  }

  provisioner "powershell" {
    elevated_password = "vagrant"
    elevated_user     = "vagrant"
    pause_before      = "1m0s"
    scripts            = ["${path.root}/scripts/sysprep/cloudbase-init-
p2.ps1"]
  }
}

```

On vient crée des **templates** Windows en buildant les VMs avec **Packer** :

```

packer init .
packer validate -var-file=windows_server2019_proxmox_cloudinit.pkvars.hcl .
packer build -var-file=windows_server2019_proxmox_cloudinit.pkvars.hcl .

packer validate -var-file=windows_server2016_proxmox_cloudinit.pkvars.hcl .
packer build -var-file=windows_server2016_proxmox_cloudinit.pkvars.hcl .

```



```

root@provisioning:~/GOAD/packer/proxmox# packer build -var-file=windows_server2016_proxmox_cloudinit.pkvars.hcl .
proxmox-iso.windows: output will be in this color.

==> proxmox-iso.windows: Retrieving additional ISO
==> proxmox-iso.windows: Trying ./iso/Autounattend_winsrvr2016_cloudinit.iso
==> proxmox-iso.windows: Trying ./iso/Autounattend_winsrvr2016_cloudinit.iso?checksum=sha256x3A3cddf72bc5957f50a4216a0fca0f02d7688b910afad0cb08f4fad1bd97b2ce6
==> proxmox-iso.windows: ./iso/Autounattend_winsrvr2016_cloudinit.iso?checksum=sha256x3A3cddf72bc5957f50a4216a0fca0f02d7688b910afad0cb08f4fad1bd97b2ce6 => /root/.GOAD/packer/proxmox/iso/Autounattend_winsrvr2016_cloudinit.iso
proxmox-iso.windows: Uploaded ISO to local:iso/Autounattend_winsrvr2016_cloudinit.iso
==> proxmox-iso.windows: Creating VM
==> proxmox-iso.windows: No VM ID given, getting next free from Proxmox
==> proxmox-iso.windows: Starting VM
==> proxmox-iso.windows: Waiting for WinRM to become available...
proxmox-iso.windows: WinRM connected.
==> proxmox-iso.windows: Connected to WinRM!
==> proxmox-iso.windows: Provisioning with Powershell...
==> proxmox-iso.windows: Provisioning with powershell script: ./scripts/sysprep/cloudbase-init.ps1
proxmox-iso.windows:
proxmox-iso.windows: Directory: C:\
proxmox-iso.windows:
proxmox-iso.windows: Mode                LastWriteTime         Length Name
proxmox-iso.windows: ----                -
proxmox-iso.windows: d-----          11/29/2023   7:06 PM         setup
proxmox-iso.windows: Copy CloudbaseInitSetup_Stable_x64.mst
proxmox-iso.windows: Start process CloudbaseInitSetup_Stable_x64.mst
==> proxmox-iso.windows: Pausing 1m0s before the next provisioner...
==> proxmox-iso.windows: Provisioning with Powershell...
==> proxmox-iso.windows: Provisioning with powershell script: ./scripts/sysprep/cloudbase-init-p2.ps1
proxmox-iso.windows: Show cloudinit service
proxmox-iso.windows:
proxmox-iso.windows: Status Name              DisplayName
proxmox-iso.windows: -----
proxmox-iso.windows: Stopped cloudbase-init cloudbase-init
proxmox-iso.windows: Move config files to location
proxmox-iso.windows: Disable cloudbaseinit at start
==> proxmox-iso.windows: Stopping VM
==> proxmox-iso.windows: Converting VM to template
==> proxmox-iso.windows: Adding a cloud-init cdrom in storage pool local-lvm
Build 'proxmox-iso.windows' finished after 6 minutes 43 seconds.

==> Wait completed after 6 minutes 43 seconds
==> Builds finished. The artifacts of successful builds are:
--> proxmox-iso.windows: A template was created: 104

root@provisioning:~/GOAD/packer/proxmox# packer build -var-file=windows_server2019_proxmox_cloudinit.pkvars.hcl .
proxmox-iso.windows: output will be in this color.

==> proxmox-iso.windows: Retrieving additional ISO
==> proxmox-iso.windows: Trying ./iso/Autounattend_winsrvr2019_cloudinit.iso
==> proxmox-iso.windows: Trying ./iso/Autounattend_winsrvr2019_cloudinit.iso?checksum=sha256x3A6a02590bab83de1c04f9d349420840439f772bc176fbaa570fe2fb66b14b7a00
==> proxmox-iso.windows: ./iso/Autounattend_winsrvr2019_cloudinit.iso?checksum=sha256x3A6a02590bab83de1c04f9d349420840439f772bc176fbaa570fe2fb66b14b7a00 => /root/.GOAD/packer/proxmox/iso/Autounattend_winsrvr2019_cloudinit.iso
proxmox-iso.windows: Uploaded ISO to local:iso/Autounattend_winsrvr2019_cloudinit.iso
==> proxmox-iso.windows: Creating VM
==> proxmox-iso.windows: No VM ID given, getting next free from Proxmox
==> proxmox-iso.windows: Starting VM
==> proxmox-iso.windows: Waiting for WinRM to become available...
proxmox-iso.windows: WinRM connected.
==> proxmox-iso.windows: Connected to WinRM!
==> proxmox-iso.windows: Provisioning with Powershell...
==> proxmox-iso.windows: Provisioning with powershell script: ./scripts/sysprep/cloudbase-init.ps1
proxmox-iso.windows:
proxmox-iso.windows: Directory: C:\
proxmox-iso.windows:
proxmox-iso.windows: Mode                LastWriteTime         Length Name
proxmox-iso.windows: ----                -
proxmox-iso.windows: d-----          11/29/2023   6:45 PM         setup
proxmox-iso.windows: Copy CloudbaseInitSetup_Stable_x64.mst
proxmox-iso.windows: Start process CloudbaseInitSetup_Stable_x64.mst
==> proxmox-iso.windows: Pausing 1m0s before the next provisioner...
==> proxmox-iso.windows: Provisioning with Powershell...
==> proxmox-iso.windows: Provisioning with powershell script: ./scripts/sysprep/cloudbase-init-p2.ps1
proxmox-iso.windows: Show cloudinit service
proxmox-iso.windows:
proxmox-iso.windows: Status Name              DisplayName
proxmox-iso.windows: -----
proxmox-iso.windows: Stopped cloudbase-init cloudbase-init
proxmox-iso.windows: Move config files to location
proxmox-iso.windows: Disable cloudbaseinit at start
==> proxmox-iso.windows: Stopping VM
==> proxmox-iso.windows: Converting VM to template
==> proxmox-iso.windows: Adding a cloud-init cdrom in storage pool local-lvm
Build 'proxmox-iso.windows' finished after 9 minutes 57 seconds.

==> Wait completed after 9 minutes 57 seconds
==> Builds finished. The artifacts of successful builds are:
--> proxmox-iso.windows: A template was created: 103

```

Terraform provisionning

Avant de faire le provisioning, on vient mettre en place le fichier de variable de **Terraform**.

Premièrement, on fait une copie du fichier template qui nous ai donné :

```
cd /root/GOAD/ad/GOAD/providers/proxmox/terraform
cp variables.template variables.tf
```

Par la suite on va y mettre nos variables. Dans notre cas cela donnera :

```
variable "pm_api_url" {
  default = "https://10.202.3.33:8006/api2/json"
}

variable "pm_user" {
  default = "infra_as_code@pve"
}
```

```
variable "pm_password" {
  default = "infra"
}

variable "pm_node" {
  default = "pve7"
}

variable "pm_pool" {
  default = "GOAD"
}
```

```
cd /root/GOAD/ad/GOAD/providers/proxmox/terraform
terraform init
terraform plan -out goad.plan
terraform apply "goad.plan"
```

![terraform-provided](img/P2/launch packer/terraform-provided.png)

On obtiendra une erreur sur la partie **Ansible** mais pas de panique, nous allons la corriger.

Ansible provisioning

Dans le répertoire `/root/GOAD/ad/GOAD/providers/promox/` se trouve le fichier `inventory` d'**Ansible**. Ce dernier est par défaut configuré pour **Virtual-Box**.

On va donc le modifier pour obtenir ce nouveau fichier :

```
[default]
; Note: ansible_host *MUST* be an IPv4 address or setting things like DNS
; servers will break.
; -----
; sevenkingdoms.local
; -----
dc01 ansible_host=192.168.10.10 dns_domain=dc01 dict_key=dc01
; -----
; north.sevenkingdoms.local
; -----
dc02 ansible_host=192.168.10.11 dns_domain=dc01 dict_key=dc02
srv02 ansible_host=192.168.10.22 dns_domain=dc02 dict_key=srv02
; -----
; essos.local
; -----
dc03 ansible_host=192.168.10.12 dns_domain=dc03 dict_key=dc03
srv03 ansible_host=192.168.10.23 dns_domain=dc03 dict_key=srv03
; -----
; Other
; -----
```

```
elk ansible_host=192.168.10.50 ansible_connection=ssh

[all:vars]
; domain_name : folder inside ad/
domain_name=GOAD

force_dns_server=yes
dns_server=8.8.8.8

two_adapters=no
nat_adapter=Ethernet 2
domain_adapter=Ethernet 2

; proxy settings (the lab need internet for some install, if you are behind
a proxy you should set the proxy here)
enable_http_proxy=no
ad_http_proxy=http://x.x.x.x:xxxx
ad_https_proxy=http://x.x.x.x:xxxx

[elk_server:vars]
; ssh connection (linux)
ansible_ssh_user=vagrant
ansible_ssh_private_key_file=./.vagrant/machines/elk/virtualbox/private_key
ansible_ssh_port=22
host_key_checking=false
```

La dernière problématique que nous rencontrerons est le réseau Internet de l'IUT... Ce dernier étant très long, on va récupérer un problème de *time-out*. Pour résoudre ce problème on vient dans le fichier `root/GOAD/script/provisioning.sh` a la ligne **27** :

```
timeout 20m $ANSIBLE_COMMAND $1
```

On va remplacer le 20min par 40min afin que le réseau puisse travailler.

Et l'on peut ensuite lancer la commande suivante :

```
cd /root/GOAD/
./goad.sh -t install -l GOAD -p proxmox
```