

# Utiliser l'API REST de Proxmox avec Powershell

## Ressources

Documentation de l'API Proxmox : [https://pve.proxmox.com/wiki/Proxmox\\_VE\\_API](https://pve.proxmox.com/wiki/Proxmox_VE_API)

Utilisation avec Powershell : <https://github.com/Corsinvest/cv4pve-api-powershell>

## Créer un clé API

```
* sans séparation de privilège
```

## Installation de Powershell 7

```
iex "& { $(irm https://aka.ms/install-powershell.ps1) } -UseMSI -Quiet"
```

## Install du module Powershell

Ouvrir une console Powershell

```
PS > Install-Module -Name Corsinvest.ProxmoxVE.Api
```

## Connexion au cluster

- Connexion au cluster en indiquant un jeton d'API

```
PS> Import-Module Corsinvest.ProxmoxVE.Api
```

```
PS > $ticket = Connect-PveCluster -HostsAndPorts 10.xxx.xxx.xxx:8006 -SkipCertificateCheck -ApiToken  
nom_jeton=valeur_jeton
```

```
#return Ticket, default set $Global:PveTicketLast
```

```
#this is useful when connections to multiple clusters are needed use parameter -SkipRefreshPveTicketLast
```

```
# visualiser le ticket
```

```
PS> $ticket
```

```
HostName           : 1.0xxx.xxx.xxx  
Port               : 8006  
SkipCertificateCheck : True  
CSRFPreventionToken :  
ApiToken           : nom_jeton=valeur_jeton  
CSRFPreventionToken :
```

```
#For disable output call Connect-PveCluster > $null
```

```
#Get version
```

```
PS > $ret = Get-PveVersion
```

```
#$ret return a class PveResponse
```

```
#Show data
```

```
PS > $ret.Response.data
```

```
version repoid          release  
-----  
8.1.4   ec5affc9e41f1d79 8.1
```

```
#Show data 2
```

```
PS /home/frank> $ret.ToTable()
```

version	repid	release
8.1.4	ec5affc9e41f1d79	8.1

## Lister les VM d'un noeud

```
PS > (Get-PveNodesQemu -node siohyp2 -Full).ToTable()
```

mem	diskread	status	disk	cpu	diskwrite	netout	cpus	maxmem	qmpstatus	netin	uptime
maxdisk	name		vmid								
0	34359738368	0	stopped	0	0	0	4	2147483648	stopped	0	
0	53687091200	0	stopped	0	0	0	2	4194304000	stopped	0	
0	10737483776	0	stopped	0	0	0	4	1073741824	stopped	0	

## Lister les conteneurs LXC d'un noeud

```
(Get-PveNodeslxc -node siohyp2).ToTable()
```

netin	maxmem	type	uptime	maxdisk	name	vmid	mem	diskread
status	diskwrite	netout		disk	cpu	swap	maxswap	cpus
stopped	0	0	0	0	0	0	536870912	1
stopped	0	0	0	0	0	0	536870912	1
stopped	0	0	0	0	0	0	536870912	1

Afficher que les Vmid

```
PS C:\Users\boulesteix.fabien> (Get-PveNodeslxc -node siohyp2).response.data | Select-Object vmid
```

## Lister les pools de ressources

```
(Get-PvePools).todata()
```

## Lister les VM d'un pool de ressources

```
(Get-PvePools -Poolid nompool).Response.data.members | where-Object type -EQ "qemu" | Select-Object vmid
```

## Lister les conteneurs d'un pool de ressources

```
(Get-PvePools -Poolid nompool).Response.data.members | where-Object type -EQ "lxc" | Select-Object vmid
```

## Supprimer un conteneur LXC

```
Remove-PveNodesLxc -DestroyUnreferencedDisks -Force -Node siohyp1 -Purge -Vmid 188
```

```
Response : @{data=UPID:siohyp1:00189780:20ABFE12:66DC528F:vzdestroy:188:nom_jeton:}  
StatusCode : 200  
ReasonPhrase :  
IsSuccessStatusCode : True
```

```
RequestResource : /nodes/siohyp1/lxc/188
Parameters      : {[force, 1], [purge, 1], [destroy-unreferenced-disks, 1]}
Method          : Delete
ResponseType    : json
```

## Supprimer les conteneurs LXC d'un pool de ressources

```
(Get-PvePools -Poolid nom_pool).Response.data.members | where-Object type -EQ "lxc" | Select-Object
vmid, node | foreach-object { Remove-PveNodesLxc -DestroyUnreferencedDisks -Force -Node $PSItem.node -
Purge -Vmid $PSItem.vmid}
```

## Supprimer les conteneurs LXC de plusieurs pools de ressources

```
PS > (Get-PvePools).todata() | Where-Object poolid -like "*2022*" | foreach-object {(Get-PvePools -
Poolid $PSItem.poolid).Response.data.members } | where-Object type -EQ "lxc" | foreach-object { Remove-
PveNodesLxc -DestroyUnreferencedDisks -Force -Node $PSItem.node -Purge -Vmid $PSItem.vmid}
```

## Arrêter et supprimer les VM d'un pool de ressources

```
#arrêter les VM
(Get-PvePools -Poolid nom_pool).Response.data.members | where-Object type -EQ "qemu" | Select-Object
vmid, node | foreach-object { Stop-PveVm -VmIdOrName $PSItem.vmid}

#supprimer les VM
(Get-PvePools -Poolid nom_pool).Response.data.members | where-Object type -EQ "qemu" | Select-Object
vmid, node | foreach-object { Remove-PveNodesQemu -DestroyUnreferencedDisks -Node $PSItem.node -Purge -
Vmid $PSItem.vmid}
```

## Arrêter et supprimer les VM de plusieurs pools de ressources

```
#Arrêter toutes les VM concernées
PS > (Get-PvePools).todata() | Where-Object poolid -like "*2022*" | foreach-object {(Get-PvePools -
Poolid $PSItem.poolid).Response.data.members} | where-Object type -EQ "qemu" | Select-Object vmid, node
| foreach-object { Stop-PveVm -VmIdOrName $PSItem.vmid}

#supprimer toutes les VM concernées
PS > (Get-PvePools).todata() | Where-Object poolid -like "*2022*" | foreach-object {(Get-PvePools -
Poolid $PSItem.poolid).Response.data.members} | where-Object type -EQ "qemu" | Select-Object vmid, node
| foreach-object { Remove-PveNodesQemu -DestroyUnreferencedDisks -Node $PSItem.node -Purge -Vmid
$PSItem.vmid}
```

## Supprimer plusieurs pools de ressources

```
#Arrêter toutes les VM concernées
PS > (Get-PvePools).todata() | Where-Object poolid -like "*2022*" | foreach-object {Remove-PvePools -
Poolid $PSItem.poolid }

===== Lister les étudiants d'une promotion =====
<code powershell>
PS > (Get-PveAccessusers -full).Response.data | Where-Object groups -eq 'BTSSI02022' | Select-Object
userid

userid
-----
toto.florian@Valadon
tata@Valadon
```

## Lister les permissions d'un compte

```
PS > (Get-PveAccesspermissions -Userid hajji.maryam@Valadon).Response.data
```

From:

/ - **Les cours du BTS SIO**

Permanent link:

</doku.php/reseau/cloud/proxmox/apirestpowershell?rev=1756371843>

Last update: **2025/08/28 11:04**

