Mode Opératoire

Administration des conteneurs LXC sur ProxMox VE

Code :	MO-PLT-018
Version :	1.0
Date :	13 avril 2026
Auteur :	Cédric LEGRAND
Classification :	USAGE INTERNE — Équipe BTS SIO

Historique des révisions

Version	Date	Modifications
1.0	13/04/2026	Création initiale — 29 CT, 4 production autostart

1 Objet

Ce mode opératoire décrit les procédures d'administration courante des conteneurs LXC hébergés sur l'hyperviseur ProxMox VE de l'infrastructure BTS SIO. Il couvre le contrôle du démarrage automatique, la gestion du cycle de vie des conteneurs (démarrage, arrêt, redémarrage), la consultation des ressources et des logs, ainsi que la création d'un nouveau conteneur.

L'infrastructure héberge 29 conteneurs LXC répartis en deux catégories : 4 conteneurs de **production** (Wazuh, WireGuard, Vaultwarden, docker-srv) et 25 conteneurs **étudiants** (portfolios, environnements de développement, projets SLAM).

2 Champ d'application

Application	ProxMox VE 8.4.14 (kernel 6.8.12-17-pve)
Hébergement	Serveur physique (`10.0.112.200`), 2× Xeon E5-2620 v2, 32 Go RAM
Accès	`https://10.0.112.200:8006` (interface web) ou SSH `root@10.0.112.200`
Authentification	Compte `root@pam` (identifiants dans Vaultwarden, collection *Virtualisation*)
Durée estimée	5–15 minutes (opérations courantes)

3 Prérequis

Prérequis

— Accès SSH au serveur ProxMox : `ssh root@10.0.112.200` (clé SSH déployée)
— Ou accès web : `https://10.0.112.200:8006` (certificat auto-signé)
— Identifiants `root@pam` (gestionnaire de mots de passe)
— Connaissance du numéro de CT (CTID) concerné

Convention de nommage des CT

Plage CTID	Catégorie	Exemples
100–102	Projets étudiants (anciens)	icad, ksav, publicom
103	Production — SIEM	Wazuh
104–125	Étudiants (portfolios, envs)	portfolioViste, envBerkane
126	Production — VPN	wireguard-gw
127	Production — Coffre-fort	vaultwarden
200	Production — Docker	docker-srv (13 conteneurs)

4 Procédure

4.1 Vérifier et configurer le démarrage automatique

Étape 1 — Lister les conteneurs et leur statut autostart

```
# Lister tous les CT avec leur statut onboot
for ct in $(pct list | tail -n +2 | awk '{print $1}'); do
  name=$(pct config $ct | grep hostname | awk '{print $2}')
  status=$(pct status $ct | awk '{print $2}')
  onboot=$(pct config $ct | grep onboot | awk '{print $2}')
  echo "CT $ct  $name  status=$status  onboot=${onboot:-0}"
done
```

Résultat attendu (13/04/2026) : 4 CT de production avec `onboot=1` :

CTID	Nom	Rôle	onboot
103	Wazuh	SIEM/XDR (3 services, 7 agents)	**1**
126	wireguard-gw	Passerelle VPN WireGuard	**1**
127	vaultwarden	Gestionnaire de mots de passe	**1**
200	docker-srv	Plateforme Docker (13 conteneurs)	**1**

Étape 2 — Activer le démarrage automatique d'un CT

```
# Activer onboot sur un CT
pct set <CTID> -onboot 1

# Verifier
pct config <CTID> | grep onboot
```

Exemple : activer l'autostart sur le CT 200 (docker-srv) :

```
pct set 200 -onboot 1
```

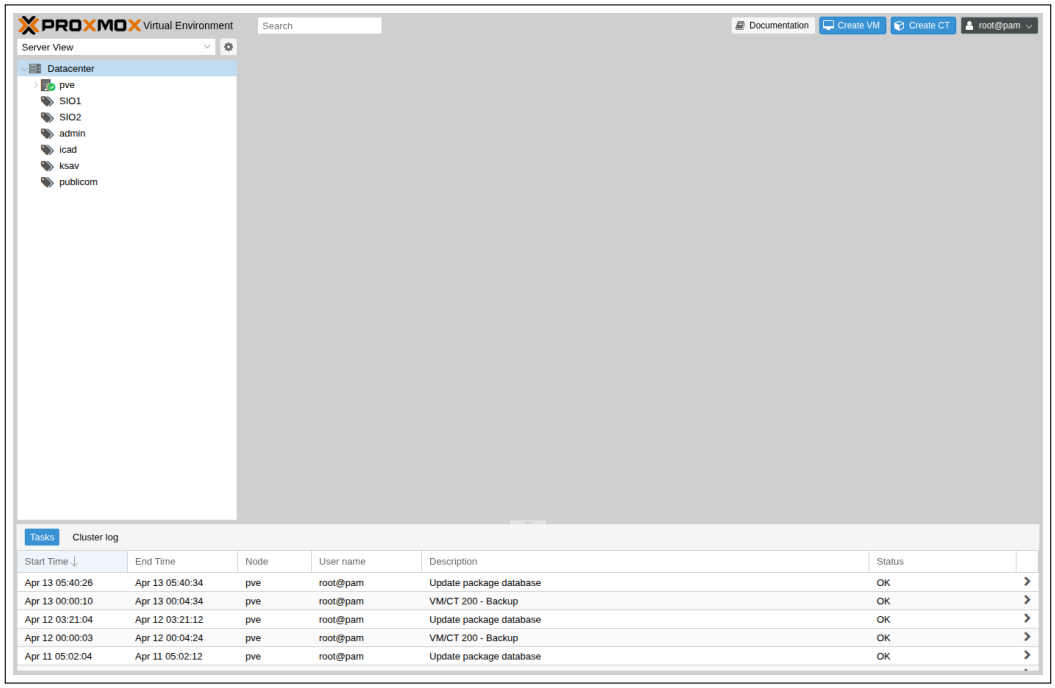

Figure 1 – Interface web ProxMox VE : arborescence des 29 CT dans le panel gauche, tâches récentes (backup CT 200 quotidien) en bas.

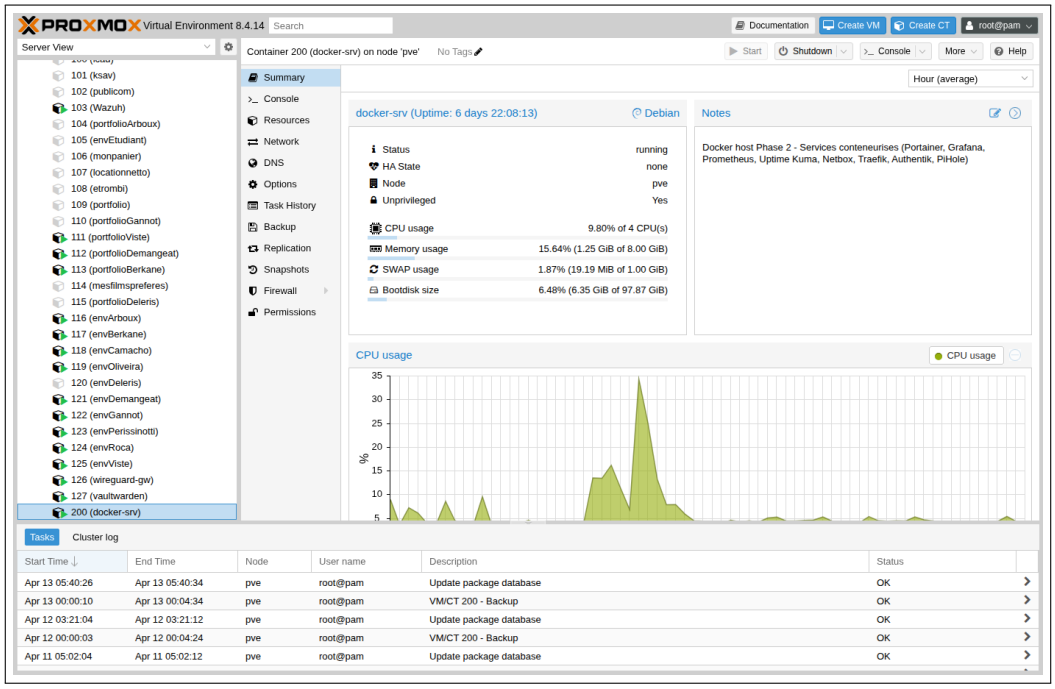

Figure 2 – Détail du CT 200 (docker-srv) : statut `running`, CPU 9,8 %, RAM 1,25/8 Go, description du rôle.

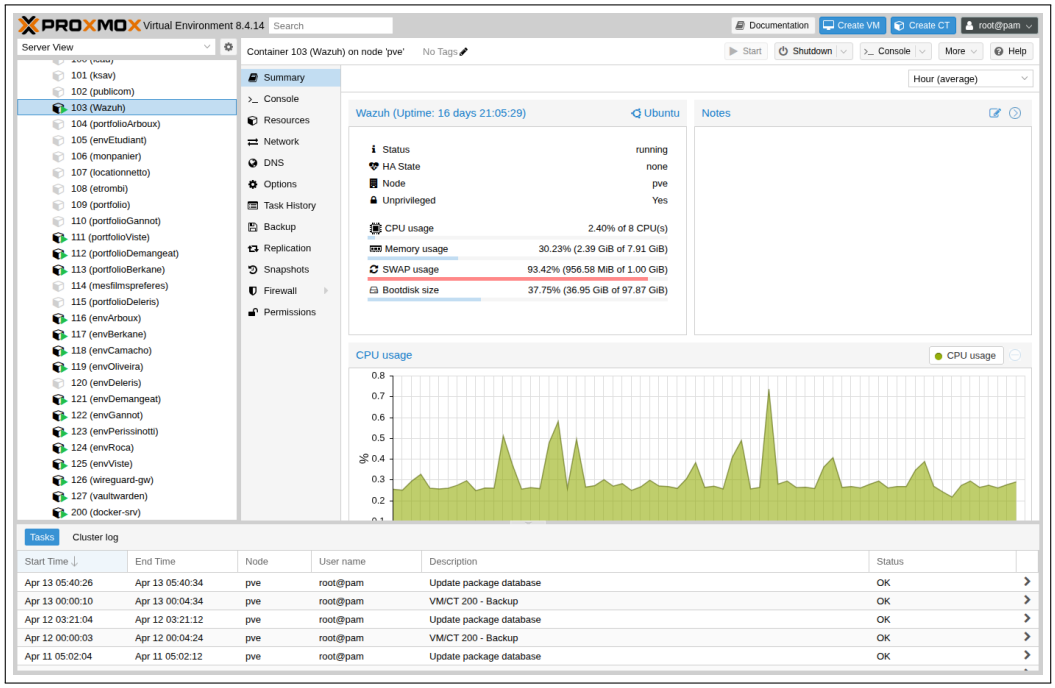

Figure 3 – Détail du CT 103 (Wazuh SIEM) : statut `running`, CPU 2,5 %, RAM 2,3/7,9 Go, uptime 16 jours.

⚠ Ordre de démarrage

En cas de redémarrage du serveur, les CT avec `onboot=1` démarrent dans l'ordre de leur CTID. Le CT 126 (wireguard-gw) doit démarrer **avant** le CT 200 (docker-srv) car les services Docker peuvent dépendre du tunnel VPN. L'ordre actuel (103, 126, 127, 200) est correct.

4.2 Gérer le cycle de vie d'un conteneur

Étape 3 — Démarrer, arrêter, redémarrer un CT

```
# Demarrer un CT
pct start <CTID>

# Arreter proprement (shutdown)
pct shutdown <CTID>

# Arreter de force (si shutdown echoue)
pct stop <CTID>

# Redemarrer
pct reboot <CTID>

# Verifier le statut
pct status <CTID>
```

Exemple : redémarrer le CT 200 (docker-srv) après une mise à jour :

```
pct reboot 200
# Attendre 30 secondes (demarrage Docker)
pct status 200
# status: running
```

Arrêt propre vs forcé

Privilégier `pct shutdown` qui envoie un signal d'arrêt propre au système invité. `pct stop` coupe l'alimentation brutalement et peut entraîner une corruption du système de fichiers. N'utiliser `stop` qu'en dernier recours (conteneur bloqué).

4.3 Consulter les ressources d'un conteneur

Étape 4 — Vérifier la configuration et l'utilisation des ressources

```
# Configuration complete d'un CT
pct config <CTID>

# Informations cles :
# - cores: nombre de coeurs CPU
# - memory: RAM en Mo
# - rootfs: disque + taille
# - net0: interface reseau, IP, bridge
# - features: nesting, keyctl
```

Exemple : configuration du CT 200 (docker-srv) :

```
cores: 4
memory: 8192
swap: 1024
rootfs: local-lvm:vm-200-disk-0,size=100G
net0: name=eth0,bridge=vmbr0,firewall=1,
      ip=10.0.112.20/16,type=veth
features: nesting=1,keyctl=1
onboot: 1
```

Pour modifier les ressources (nécessite un redémarrage pour certains paramètres) :

```
# Augmenter la RAM a 4 Go
pct set <CTID> -memory 4096

# Ajouter un coeur CPU
pct set <CTID> -cores 2

# Redimensionner le disque (+10 Go)
pct resize <CTID> rootfs +10G
```

4.4 Consulter les logs d'un conteneur

Étape 5 — Accéder aux logs système et applicatifs

```
# Executer une commande dans un CT
pct exec <CTID> -- <commande>

# Exemples :
# Logs systeme (journald)
pct exec 200 -- journalctl --no-pager -n 50

# Processus en cours
pct exec 200 -- ps aux

# Espace disque
pct exec 200 -- df -h

# Logs Docker (CT 200 uniquement)
pct exec 200 -- docker ps
pct exec 200 -- docker logs netbox --tail 20
```

Pour les logs ProxMox liés au CT (démarrage, arrêt, erreurs) :

```
# Logs ProxMox pour un CT specifique
journalctl -u pve-container@<CTID> --no-pager -n 30

# Taches recentes (backups, migrations)
cat /var/log/pve/tasks/active
```

4.5 Créer un nouveau conteneur

Étape 6 — Créer un CT depuis un template

Templates disponibles sur le stockage `local` :

Template	Taille
debian-12-standard	120 Mo
debian-13-standard	124 Mo
ubuntu-24.04-standard	135 Mo
debian-12-turnkey-lamp	281 Mo

```
# Creer un CT Debian 13 avec les parametres courants
pct create <CTID> \
  local:vztmpl/debian-13-standard_13.1-2_amd64.tar.zst \
  --hostname <NOM> \
  --cores 2 --memory 2048 --swap 512 \
  --rootfs vm-storage:<TAILLE>G \
  --net0 name=eth0,bridge=vmbr0,ip=<IP>/16,gw=10.0.112.1,\
firewall=1 \
  --features nesting=1 \
  --unprivileged 1 \
  --start 0 --onboot 0
```

Paramètres recommandés par catégorie :

Catégorie	Cores	RAM	Disque	Stockage
Étudiant (env)	1	1024 Mo	30 Go	vm-storage2
Étudiant (portfolio)	2	1024 Mo	20–50 Go	vm-storage
Projet (équipe)	2	2048 Mo	10–50 Go	vm-storage
Production	4+	4096+ Mo	50–100 Go	local-lvm

Clonage

Pour créer un CT similaire à un existant, utiliser le clonage :

```
pct clone <SOURCE_CTID> <NOUVEAU_CTID> \
  --hostname <NOM> --full
```

Le clone hérite de la configuration du CT source. Penser à modifier l'adresse IP après le clonage (`pct set <CTID> -net0 ...`).

5 Vérification

Vérification

Contrôles périodiques :

- ☐ Les 4 CT de production (103, 126, 127, 200) sont en `onboot=1`
- ☐ Tous les CT de production sont en statut `running`
- ☐ Le CT 200 (docker-srv) fait l'objet d'un backup vzdump quotidien
- ☐ Les CT étudiants arrêtés n'ont **pas** `onboot=1` (éviter la surcharge au démarrage)
- ☐ L'espace disque des stockages est inférieur à 80 %
- ☐ La RAM totale utilisée est inférieure à 75 % (réserve pour les pics)

6 Dépannage

Problème	Solution
CT ne démarre pas	Vérifier les logs : `journalctl -u pve-container@<CTID>`. Causes fréquentes : espace disque plein sur le stockage, conflit d'adresse IP, template corrompu.
CT bloqué (ne répond plus)	Tenter `pct shutdown <CTID> --timeout 30`. Si échec, forcer : `pct stop <CTID>`. Vérifier l'état après redémarrage.
`pct exec` ne fonctionne pas	Le CT doit être en `running`. Vérifier avec `pct status <CTID>`. Si le CT tourne mais `exec` échoue, le système invité est peut-être corrompu — utiliser l'accès console : `pct console <CTID>`.
Espace disque insuffisant dans un CT	Agrandir le disque : `pct resize <CTID> rootfs +10G`. Le redimensionnement est à chaud (pas de redémarrage requis). Vérifier ensuite dans le CT : `pct exec <CTID> -- df -h`.
Interface web inaccessible	Vérifier que le service `pveproxy` tourne : `systemctl status pveproxy`. Accès de secours : SSH (`ssh root@10.0.112.200`).