Mode Opératoire

Ajouter un serveur à la supervision Prometheus

Code :	MO-PLT-022
Version :	1.1
Date :	26 juin 2026
Auteur :	Cédric LEGRAND
Classification :	USAGE INTERNE — Équipe BTS SIO

Historique des révisions

Version	Date	Modifications
1.0	13/06/2026	Création initiale
1.1	26/06/2026	Harmonisation de la présentation des étapes et des encadrés.

1 Objet

Ce mode opératoire décrit l'ajout d'un serveur à la supervision **Prometheus** de l'infrastructure BTS SIO. La stack de supervision (Prometheus, Grafana, alerting) tourne sur le CT 200 `docker-srv` ; étendre sa couverture à un nouveau serveur se déroule toujours en trois temps :

1. **installer un exporteur** sur le serveur à superviser, qui expose les métriques système sur un port HTTP ;
2. **ouvrir le flux réseau** entre le serveur Prometheus et cet exporteur — et uniquement ce flux ;
3. **déclarer la cible** dans la configuration de Prometheus, puis recharger celle-ci à chaud.

Deux familles de serveurs sont traitées : les serveurs **Windows** (contrôleurs de domaine, serveurs membres) avec `windows_exporter`, et les conteneurs ou serveurs **Linux** avec `node_exporter`. La procédure intègre la contrainte propre au site : le VLAN serveurs n'a **pas d'accès Internet**, toute installation se fait donc hors-ligne depuis le poste d'administration.

2 Champ d'application

Public concerné	Administrateurs de l'infrastructure BTS SIO
Serveur Prometheus	CT 200 `docker-srv` (`10.0.112.20`), Prometheus v3.3.1, configuration dans `/opt/docker/monitoring/`
Cibles Windows	Windows Server 2022 — `windows_exporter` 0.31.7, port 9182
Cibles Linux	Conteneurs LXC ProxMox (Debian/Ubuntu) — `node_exporter` 1.11.1, port 9100
Outils	`smbclient`, PowerShell via WinRM, `pct` (ProxMox), `promtool`
Durée	15 à 20 minutes par serveur

3 Concepts

3.1 Le modèle « pull » de Prometheus

Contrairement à un agent qui pousse ses données vers un collecteur (modèle de Zabbix ou de Wazuh), Prometheus **interroge** lui-même ses cibles à intervalle régulier (15 secondes ici) : chaque serveur supervisé expose un point d'accès HTTP `/metrics` en lecture seule, servi par un *exporteur*. Si Prometheus n'arrive plus à joindre une cible, la métrique `up` passe à 0 et l'alerte `InstanceDown` se déclenche — la panne de la collecte est donc elle-même supervisée.

Exporteur	Système	Port	Métriques exposées
`windows_exporter`	Windows Server	9182	CPU, mémoire, disques, réseau, services, et collecteurs spécialisés (AD, DNS…)
`node_exporter`	Linux	9100	CPU, mémoire, systèmes de fichiers, réseau, charge
`cAdvisor`	Hôte Docker	8080	Ressources consommées par chaque conteneur

3.2 Jobs, cibles et libellés lisibles

Dans `prometheus.yml`, les cibles sont regroupées par `job` (un job par type d'exporteur : `windows`, `node`, `cadvisor`). Chaque cible porte des **labels** : l'infrastructure utilise un label `instance` **lisible** (« DC1 (AD principal) » plutôt que `10.0.112.2:9182`) et un label `role` descriptif. Ces libellés se propagent partout : menus déroulants des dashboards Grafana, légendes des graphiques et annotations des alertes. Un collègue qui lit « Cible DC1 (AD principal) injoignable » sait immédiatement de quoi il s'agit.

3.3 Contrainte hors-ligne du VLAN serveurs

Les serveurs du VLAN `10.0.112.0/24` n'atteignent ni GitHub ni les dépôts de paquets : impossible d'y lancer un `winget install` ou un `apt install`. Le contournement est systématique : les binaires sont téléchargés **sur le poste d'administration** (connecté au VPN et à Internet), puis poussés vers la cible par le canal adapté — partage administratif `C$` pour Windows, `scp` ou `pct push` pour les conteneurs LXC.

Épingler les versions

Les exporteurs déployés sont volontairement épinglés : `windows_exporter` 0.31.7 et `node_exporter` 1.11.1. Monter de version majeure sans précaution peut renommer des métriques et casser silencieusement les dashboards (panneaux « N/A ») : la marche à suivre face à ce cas est détaillée dans le MO-PLT-024.

4 Prérequis

Prérequis

- **Poste d'administration connecté au VPN WireGuard et à Internet**
- **Identifiants : compte d'administration du domaine (coffre Vaultwarden, item « DC2 ») pour une cible Windows ; clé SSH vers `pve` (`10.0.112.200`) et le CT 200 pour une cible Linux**
- **Binaires téléchargés sur le poste :**
  - **`windows_exporter-0.31.7-amd64.msi` (dépôt GitHub `prometheus-community/windows_exporter`)**
  - **`node_exporter-1.11.1.linux-amd64.tar.gz` (dépôt GitHub `prometheus/node_exporter`)**
- **Outils : `smbclient` et Python 3 avec `pywinrm` sur le poste d'administration**

Le port de l'exporteur n'est pas un port public

Un exporteur expose l'état détaillé du serveur (versions, volumes, services, interfaces) sans authentification. Le flux entrant vers le port 9182 ou 9100 doit être restreint à la seule adresse du serveur Prometheus (`10.0.112.20`) : c'est l'objet de l'étape pare-feu, à ne jamais omettre sur une cible Windows.

5 Procédure

5.1 Cible Windows — `windows_exporter`

1 Téléverser le MSI sur le serveur

Depuis le poste d'administration, pousser le MSI vers le répertoire temporaire du serveur via le partage administratif `C$` :

```
smbclient //10.0.112.2/C$ -U "Administrateur" \
  -c 'cd Windows\Temp; put windows_exporter-0.31.7-amd64.msi'
```

Le mot de passe demandé est celui du compte d'administration du domaine.

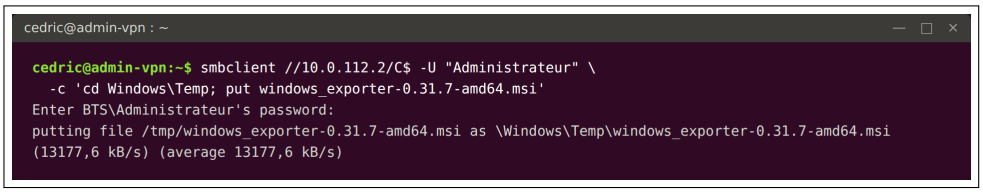

Figure 1 : Téléversement du MSI `windows_exporter` vers `C:\Windows\Temp` du DC1 via le partage administratif.

2 Installer en silencieux avec les collecteurs adaptés

L'installation se pilote à distance en PowerShell (session WinRM ou console locale du serveur). Le paramètre `ENABLED_COLLECTORS` accepte le mot-clé `[defaults]`, auquel on ajoute les collecteurs spécialisés utiles — `ad` et `dns` pour un contrôleur de domaine :

```
$msi = "C:\Windows\Temp\windows_exporter-0.31.7-amd64.msi"
$p = Start-Process msiexec.exe -Wait -PassThru -ArgumentList `
  "/i `"$msi`" ENABLED_COLLECTORS=`"[defaults],ad,dns`" LISTEN_PORT=9182
   /qn"
$p.ExitCode          # 0 attendu
Get-Service windows_exporter
```

Figure 2 : Installation silencieuse du `windows_exporter` sur DC1 : code retour 0, service `Running` à l'écoute sur le port 9182.

Choisir les collecteurs selon le rôle du serveur

Sur un serveur membre sans rôle particulier, `ENABLED_COLLECTORS="[defaults]"` suffit (CPU, mémoire, disques, réseau, services, système). Les collecteurs `ad` et `dns` ne fonctionnent que sur un contrôleur de domaine ; un collecteur activé sur un serveur qui ne porte pas le rôle correspondant remonte des erreurs de collecte sans bloquer le reste.

3 Restreindre le pare-feu au serveur Prometheus

Créer la règle entrante limitée à `10.0.112.20` :

```
New-NetFirewallRule `
  -DisplayName "windows_exporter (Prometheus)" `
  -Direction Inbound -Action Allow -Protocol TCP `
  -LocalPort 9182 -RemoteAddress 10.0.112.20 -Profile Any
```

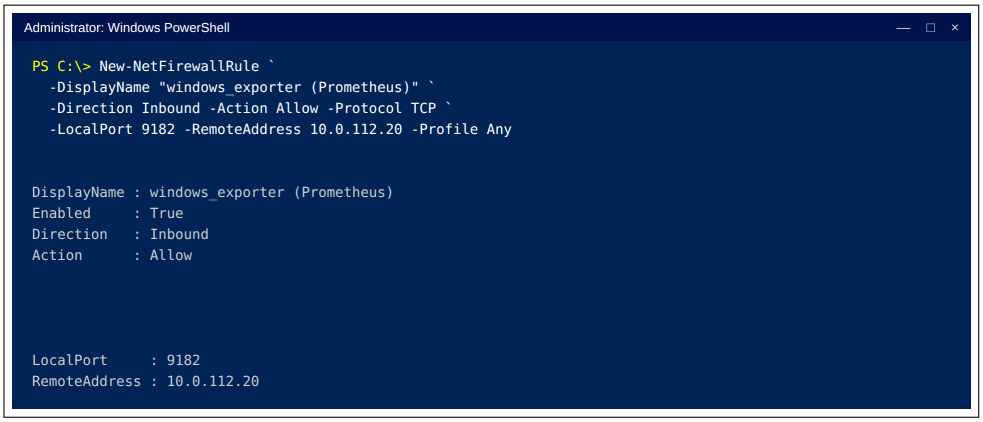

Figure 3 : Règle de pare-feu entrante du DC1 : port 9182 autorisé uniquement depuis le serveur Prometheus (10.0.112.20).

5.2 Cible Linux — `node_exporter`

4

Préparer le binaire et l'unité systemd

`node_exporter` est un binaire statique sans dépendance. Depuis le poste d'administration, l'extraire puis le copier vers l'hôte ProxMox :

```
tar xzf node_exporter-1.11.1.linux-amd64.tar.gz
scp node_exporter-1.11.1.linux-amd64/node_exporter \
    root@10.0.112.200:/tmp/
```

Créer le fichier d'unité `node_exporter.service` :

```
[Unit]
Description=Prometheus Node Exporter
Wants=network-online.target
After=network-online.target

[Service]
User=node_exporter
Group=node_exporter
```

```
ExecStart=/usr/local/bin/node_exporter
Restart=on-failure
RestartSec=5

[Install]
WantedBy=multi-user.target
```

5 Pousser les fichiers dans le conteneur et activer le service

Depuis l'hôte `pve`, la commande `pct push` dépose les fichiers dans le conteneur sans passer par le réseau, puis `pct exec` active le service sous un compte de service dédié :

```
pct push 103 /tmp/node_exporter \
  /usr/local/bin/node_exporter --perms 755
pct push 103 /tmp/node_exporter.service \
  /etc/systemd/system/node_exporter.service --perms 644

pct exec 103 -- bash -c "
  useradd -rs /usr/sbin/nologin node_exporter 2>/dev/null
  systemctl daemon-reload
  systemctl enable --now node_exporter
  systemctl is-active node_exporter"
```

Figure 4 : Déploiement de `node_exporter` dans le CT 103 via `pct push` : service actif au premier démarrage.

Serveur Linux hors ProxMox

Pour une VM ou un serveur physique Linux, la logique est identique en remplaçant `pct push` par `scp` vers la cible et `pct exec` par une session SSH. Le binaire, l'unité systemd et le compte de service ne changent pas.

5.3 Déclaration de la cible dans Prometheus

6 **Déclarer la cible avec des libellés lisibles**

Sur le CT 200, éditer `/opt/docker/monitoring/prometheus.yml` et ajouter la cible dans le job correspondant à son type. Exemple pour un serveur Windows et un conteneur Linux :

```
- job_name: "windows"
  static_configs:
    - targets: ["10.0.112.2:9182"]
      labels:
        instance: "DC1 (AD principal)"
        role: "Contrôleur de domaine"

- job_name: "node"
  static_configs:
    - targets: ["10.0.112.103:9100"]
      labels:
        instance: "Wazuh SIEM"
        role: "Collecte sécurité (CT 103)"
```

Le libellé est un choix éditorial

Le label `instance` remplace l'adresse technique dans toute la chaîne de visualisation. Choisir un nom court, explicite et stable : c'est lui qui apparaîtra dans les jauges de la vue d'ensemble Grafana et dans le texte des alertes.

7 Valider la configuration et recharger à chaud

`promtool` vérifie la syntaxe et la cohérence du fichier avant de l'appliquer ; le rechargement se fait ensuite sans interruption de service grâce au point d'accès `/-/reload` :

```
docker exec prometheus promtool check config \
  /etc/prometheus/prometheus.yml
curl -X POST http://127.0.0.1:9090/-/reload
```

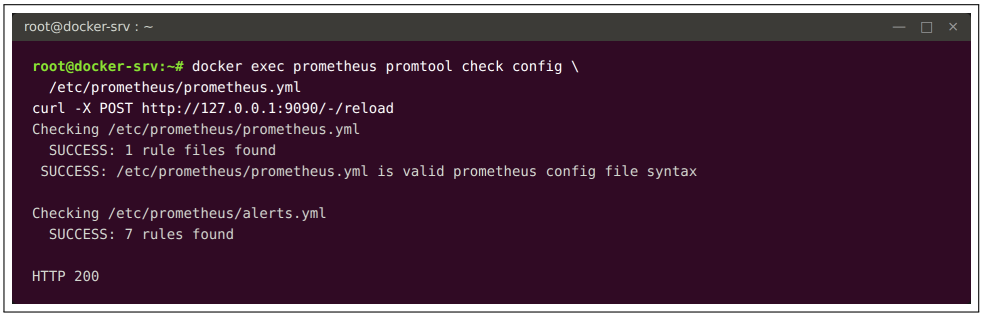

Figure 5 : Validation de la configuration par `promtool` puis rechargement à chaud : aucun redémarrage du conteneur n'est nécessaire.

8 Contrôler l'état de la nouvelle cible

Ouvrir l'interface Prometheus (`https://prometheus.docker.bts.sio`, cf. MO-PLT-015), menu Status → Targets : la nouvelle cible doit

apparaître avec l'état `UP` en moins d'une minute (deux cycles de collecte).

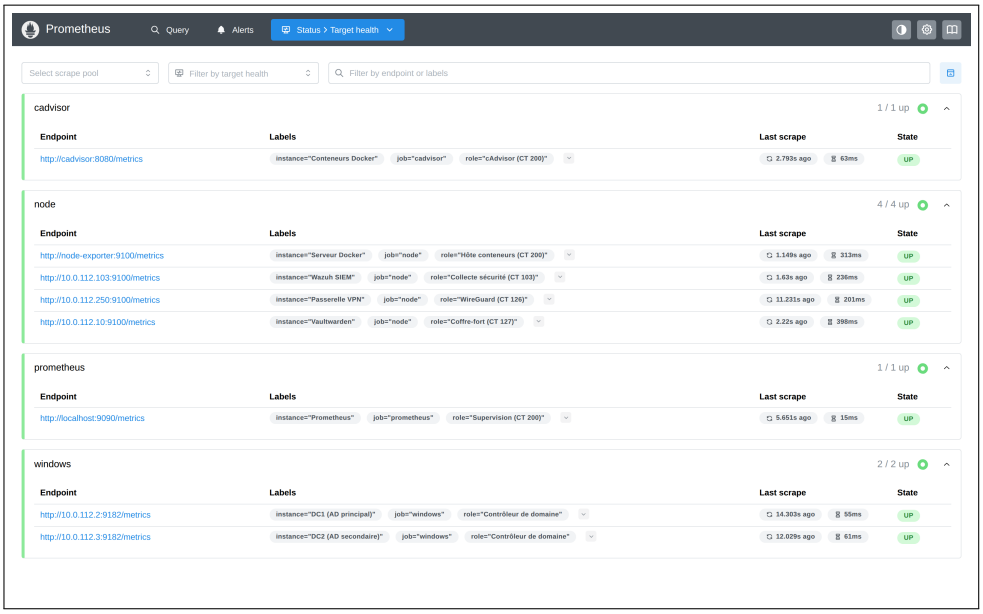

Figure 6 : Page *Targets* de Prometheus : les huit cibles de l'infrastructure à l'état `UP`, identifiées par leurs libellés lisibles.

6 Vérification

Vérification

Après l'ajout d'une cible, cocher les points suivants :

- ☐ **Le service de l'exporteur tourne sur la cible (`Get-Service windows_exporter` ou `systemctl is-active node_exporter`)**
- ☐ **Le point d'accès répond depuis le CT 200 : `curl http://<ip>:<port>/metrics | head`**
- ☐ **Il ne répond pas depuis un autre poste du VLAN (règle de pare-feu effective, cible Windows)**
- ☐ **La cible est `UP` dans Status → Targets**
- ☐ **Les métriques apparaissent dans Grafana (dashboard *Windows Exporter***

> ***2025*** **ou** ***Node Exporter Full*****, sélection de la nouvelle instance)**
>
> ☐ **La cible est couverte par les règles d'alerte existantes — automatique : les règles génériques (`up`, disque, mémoire, CPU) s'appliquent à toute nouvelle cible sans modification (cf. MO-PLT-023)**

7 Rollback

9 Retirer la cible de la collecte

Supprimer le bloc correspondant de `prometheus.yml`, puis valider et recharger comme à l'étape 6. La cible disparaît de la page *Targets* ; ses données historiques restent consultables jusqu'à expiration de la rétention (30 jours).

10 Désinstaller l'exporteur Windows

```
$msi = "C:\Windows\Temp\windows_exporter-0.31.7-amd64.msi"
Start-Process msiexec.exe -Wait -ArgumentList "/x `"$msi`" /qn"
Remove-NetFirewallRule -DisplayName "windows_exporter (Prometheus)"
```

11 Désinstaller l'exporteur Linux

```
pct exec 103 -- bash -c "
  systemctl disable --now node_exporter
  rm /usr/local/bin/node_exporter \
     /etc/systemd/system/node_exporter.service
  systemctl daemon-reload
  userdel node_exporter"
```

8 Dépannage

Problème	Solution
Cible `DOWN`, erreur « connection refused »	Le service de l'exporteur est arrêté, ou le pare-feu bloque le flux. Vérifier le service sur la cible, puis contrôler que la règle autorise bien `10.0.112.20` (et que le test ne se fait pas depuis un autre poste, qui lui est légitimement bloqué).
Cible `DOWN`, erreur « context deadline exceeded »	La cible répond trop lentement (surcharge) ou un équipement intermédiaire filtre partiellement. Tester la latence du `/metrics` depuis le CT 200 et l'état de charge du serveur cible.
`node_exporter` inactif après redémarrage du CT	L'unité n'a pas été activée (`enable` omis). Rejouer `systemctl enable -now node_exporter`.
Erreurs d'un collecteur dans les journaux Windows	Un collecteur spécialisé (`ad`, `dns`) est activé sur un serveur qui ne porte pas le rôle. Réinstaller avec la liste de collecteurs adaptée au rôle réel.
Cible `UP` mais panneaux Grafana « N/A »	La version de l'exporteur expose des noms de métriques différents de ceux attendus par le dashboard. Traiter selon la méthode du MO-PLT-024 (vérification de compatibilité).

9 Voir aussi

— **MO-PLT-015** : Se connecter à Prometheus — accès à l'interface et à la page *Targets*
— **MO-PLT-023** : Gérer les règles d'alerte Prometheus — couverture automatique des nouvelles cibles
— **MO-PLT-024** : Importer et maintenir les dashboards Grafana — visualisation des nouvelles métriques
— **MO-PLT-018** : Administration des conteneurs LXC ProxMox VE — manipulation de `pct`
— **MO-SEC-001** : Exploitation de Wazuh — le volet *événements de sécurité* de la supervision, complémentaire des métriques

Références

— Prometheus — *Configuration* (`scrape_config`, `static_config`, labels) et *Management API* (`/-/reload`)
— prometheus-community — `windows_exporter`, documentation des collecteurs et propriétés MSI
— Prometheus — `node_exporter`, liste des collecteurs activés par défaut
— ANSSI — *Recommandations relatives à l'administration sécurisée des systèmes d'information* (cloisonnement des flux d'administration)