Zabbix 7 : guide complet pour superviser Linux et Windows Server

Zabbix 7 est la solution de supervision open source la plus déployée en entreprise pour monitorer des infrastructures Linux et Windows Server. Ce guide couvre l'installation complète sur Debian 12, la configuration des agents Zabbix 2, l'application des templates officiels, la mise en place d'alertes email et Slack, les dashboards opérationnels et la comparaison avec Nagios et PRTG. Objectif : une supervision prête pour la production en moins de 4 heures.

Zabbix 7 pour la supervision de serveurs Linux et Windows Server s'impose en 2026 comme la référence open source des DSI qui veulent une visibilité complète sur leur infrastructure sans coûts de licence. Avec plus de 70 000 installations actives documentées selon la documentation officielle Zabbix, cette solution concurrence directement Nagios/Icinga, PRTG et Datadog sur les segments PME et grands comptes. Zabbix 7.0 LTS, sorti en mai 2024, apporte des améliorations majeures : nouveaux templates SNMP v3, support natif des webhooks Slack et Microsoft Teams, amélioration des performances de collecte (jusqu'à 500 000 métriques par seconde sur un serveur dédié), et une interface modernisée. Ce guide complet vous conduit de l'installation sur Debian 12 à la configuration des premiers dashboards en production, en couvrant les agents Linux (Zabbix Agent 2) et Windows, les alertes par email et Slack, les fenêtres de maintenance, et les SLA. Les commandes sont testées sur Debian 12 Bookworm avec Zabbix 7.0 LTS et MySQL 8.0.

À RETENIR

À retenir

Zabbix 7.0 LTS — support jusqu'en 2029 : choisir la version LTS pour la production garantit 5 ans de mises à jour de sécurité et de correctifs stables.

Zabbix Agent 2 remplace l'Agent 1 : écrit en Go, Agent 2 est plus performant, supporte les plugins natifs (Docker, MySQL, PostgreSQL, Redis) et le mode actif/passif sans configuration supplémentaire.

MySQL ou PostgreSQL : MySQL 8.0 est le choix par défaut pour les petites infrastructures (<500 hôtes) ; PostgreSQL avec le partitionnement TimescaleDB est recommandé pour les grandes infrastructures (>1 000 hôtes).

Templates officiels dès le départ : les templates *Linux by Zabbix agent* et *Windows by Zabbix agent active* couvrent 95 % des métriques nécessaires sans configuration manuelle des items.

Proxy Zabbix pour les sites distants : déployer un Zabbix Proxy sur chaque site distant réduit le trafic WAN et assure la continuité de supervision en cas de coupure réseau.

Prérequis et architecture — Qu'est-ce qu'on va déployer ?

L'architecture Zabbix standard se compose de trois éléments : le **Zabbix Server** (collecte, stockage, alertes), la **base de données** (MySQL ou PostgreSQL), et les **agents Zabbix** installés sur chaque serveur supervisé. Pour un déploiement complet, un quatrième composant est optionnel : le **Zabbix Proxy**, utile pour les sites distants ou les DMZ. Le serveur Zabbix et la

base de données peuvent cohabiter sur le même serveur jusqu'à environ 300 hôtes supervisés — au-delà, les séparer sur deux machines améliore les performances.

Composant	RAM recommandée	CPU	Stockage	OS cible
Zabbix Server	4 Go min, 8 Go recommandé	4 cœurs	40 Go SSD	Debian 12 / Ubuntu 22.04
Base de données MySQL 8.0	4 Go min, 16 Go recommandé	4 cœurs	100 Go SSD	Même serveur ou dédié
Zabbix Proxy (optionnel)	2 Go	2 cœurs	20 Go	Debian 12
Zabbix Agent 2 (Linux)	128 Mo	Minimal	500 Mo	Debian/RHEL/Ubuntu
Zabbix Agent 2 (Windows)	128 Mo	Minimal	500 Mo	Windows Server 2016-2025

Étape 1 — Installation de Zabbix Server sur Debian 12

L'installation de Zabbix 7.0 sur Debian 12 suit un processus en quatre phases : ajout du dépôt officiel Zabbix, installation des paquets, création de la base de données, et configuration du serveur. Utiliser exclusivement le dépôt officiel `repo.zabbix.com` pour garantir la version 7.0 LTS — les paquets des distributions sont souvent une version majeure en retard.

Étape 1 — Ajouter le dépôt Zabbix 7.0 : télécharger et installer le paquet de configuration du dépôt.

```
 # Télécharger le paquet de dépôt Zabbix 7.0 pour Debian 12
wget https://repo.zabbix.com/zabbix/7.0/debian/pool/main/z/zabbix-release/zabbix-release_7.0-1+de
dpkg -i zabbix-release_7.0-1+debian12_all.deb
apt update

# Installer Zabbix Server, frontend web, agent 2 et le connecteur MySQL
apt install -y zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf   zabbix-sql-scripts za

# Vérifier les versions installées
zabbix_server --version
# Résultat attendu : Zabbix server v7.0.x (revision ...)
```

Étape 2 — Créer la base de données MySQL : préparer le schéma Zabbix dans une base dédiée avec un utilisateur MySQL séparé.

```
 # Se connecter à MySQL en root
mysql -u root -p

# Dans le shell MySQL :
# CREATE DATABASE zabbix CHARACTER SET utf8mb4 COLLATE utf8mb4_bin;
# CREATE USER 'zabbix'@'localhost' IDENTIFIED BY 'Zabbix_S3cure_2026!';
# GRANT ALL PRIVILEGES ON zabbix.* TO 'zabbix'@'localhost';
# SET GLOBAL log_bin_trust_function_creators = 1;
# FLUSH PRIVILEGES;
# EXIT;

# Importer le schéma initial Zabbix (peut prendre 2-5 minutes)
zcat /usr/share/zabbix-sql-scripts/mysql/server.sql.gz | mysql --default-character-set=utf8mb4

# Désactiver log_bin_trust_function_creators après l'import
mysql -u root -p -e "SET GLOBAL log_bin_trust_function_creators = 0;"
```

Étape 3 — Configurer le serveur Zabbix : éditer le fichier de configuration principal pour connecter Zabbix à la base de données.

```
 # Éditer /etc/zabbix/zabbix_server.conf
# Paramètres critiques à configurer :

# DBHost=localhost
# DBName=zabbix
# DBUser=zabbix
# DBPassword=Zabbix_S3cure_2026!

# Paramètres de performance (à adapter selon la charge)
# StartPollers=10         # Threads de polling (augmenter si >200 hôtes)
# StartPingers=5          # Threads ICMP
# StartDiscoverers=3      # Threads de découverte réseau
# CacheSize=128M          # Cache configuration (augmenter si >500 hôtes)
# HistoryCacheSize=64M    # Cache historique

# Démarrer et activer au démarrage
systemctl restart zabbix-server zabbix-agent2 apache2
systemctl enable zabbix-server zabbix-agent2 apache2

# Vérifier le statut
systemctl status zabbix-server
# Vérifier les logs pour erreurs
tail -50 /var/log/zabbix/zabbix_server.log
```

Étape 4 — Configurer le frontend web : accéder à `http://<IP-serveur>/zabbix` pour l'assistant de configuration initiale. Renseigner les paramètres de connexion MySQL et définir le mot de passe de l'utilisateur *Admin* Zabbix (différent du mot de passe MySQL). Le nom d'utilisateur par défaut est **Admin** (avec A majuscule), mot de passe **zabbix** — à changer immédiatement après la première connexion.

Comment installer et configurer Zabbix Agent 2 sur Linux et Windows ?

Zabbix Agent 2 (écrit en Go) est l'agent recommandé depuis Zabbix 5.0. Il remplace progressivement l'Agent 1 (écrit en C) avec des performances supérieures, une meilleure gestion des plugins, et un support natif des connexions actives. En mode actif, l'agent initie la connexion

vers le serveur Zabbix — ce qui simplifie les règles de pare-feu pour les serveurs en DMZ (seul le port sortant 10051 est nécessaire).

Installation Agent 2 sur Linux (Debian/Ubuntu)

```
 # Sur chaque serveur Linux à superviser
# Ajouter le dépôt Zabbix (même procédure que le serveur)
wget https://repo.zabbix.com/zabbix/7.0/debian/pool/main/z/zabbix-release/zabbix-release_7.0-1+de
dpkg -i zabbix-release_7.0-1+debian12_all.deb
apt update
apt install -y zabbix-agent2

# Configurer /etc/zabbix/zabbix_agent2.conf
# Paramètres essentiels :
# Server=192.168.1.100          # IP du Zabbix Server (mode passif)
# ServerActive=192.168.1.100    # IP du Zabbix Server (mode actif)
# Hostname=srv-web01.mondomaine.lan  # Doit correspondre au nom dans Zabbix

# Démarrer l'agent
systemctl restart zabbix-agent2
systemctl enable zabbix-agent2

# Tester la connexion depuis le serveur Zabbix
zabbix_get -s 192.168.10.50 -k system.hostname
# Résultat attendu : srv-web01.mondomaine.lan
```

Installation Agent 2 sur Windows Server

```
 # Télécharger l'installeur Zabbix Agent 2 pour Windows depuis le site officiel
# https://www.zabbix.com/download_agents

# Installer via PowerShell (MSI silencieux)
$ZabbixServer = "192.168.1.100"
$Hostname = $env:COMPUTERNAME + ".mondomaine.lan"

msiexec /l*v C:\zabbix-agent2-install.log /i zabbix_agent2-7.0.0-windows-amd64-openssl.msi `
  /quiet `
  SERVER=$ZabbixServer `
  SERVERACTIVE=$ZabbixServer `
  HOSTNAME=$Hostname `
  LISTENPORT=10050

# Vérifier le service
Get-Service "Zabbix Agent 2"
# Résultat attendu : Running

# Vérifier les logs d'installation
Get-Content "C:\Program Files\Zabbix Agent 2\zabbix_agent2.log" -Tail 20
```

Application des templates et configuration des hôtes

Les templates Zabbix officiels sont la clé d'une supervision rapide et standardisée. Ils définissent des dizaines d'items (métriques), de triggers (alertes) et de graphes pré-configurés. Le template *Linux by Zabbix agent* collecte notamment : charge CPU, utilisation mémoire, I/O disque, trafic réseau, processus actifs, charge système, état des services systemd et espace disque par partition.

Étape 1 — Ajouter un hôte dans Zabbix : dans l'interface web, naviguer vers Configuration → Hosts → Create host. Renseigner le Hostname (doit correspondre à la valeur dans zabbix_agent2.conf), l'interface de l'agent (IP ou DNS), et les groupes d'hôtes.

Étape 2 — Appliquer les templates : dans l'onglet Templates de l'hôte, chercher et lier *Linux by Zabbix agent* (pour Linux) ou *Windows by Zabbix agent active* (pour Windows). Ces templates sont inclus par défaut dans Zabbix 7.0.

Étape 3 — Vérifier la collecte : après 5 minutes, vérifier dans Monitoring → Latest data que les métriques remontent correctement. Un point vert sur l'hôte indique une disponibilité confirmée.

```
 # Sur le serveur Zabbix, vérifier la connexion active de l'agent
zabbix_get -s 192.168.10.50 -k agent.version
# Résultat attendu : 7.0.x

# Tester des métriques spécifiques
zabbix_get -s 192.168.10.50 -k system.cpu.load[all,avg1]
zabbix_get -s 192.168.10.50 -k vm.memory.size[available]
zabbix_get -s 192.168.10.50 -k vfs.fs.size[/,pused]
```

Configuration des alertes — email, Slack et escalades

Zabbix 7.0 intègre nativement des médias d'alerte pour email, Slack, Microsoft Teams, PagerDuty et de nombreux autres services via webhooks. La configuration des alertes repose sur trois composants : les **Media Types** (canaux d'envoi), les **Users** (avec leurs médias assignés) et les **Actions** (conditions de déclenchement et messages).

Configuration email

```
 # Dans Administration → Media Types → Email :
# SMTP server : smtp.mondomaine.lan
# SMTP helo : zabbix-server.mondomaine.lan
# SMTP email : zabbix@mondomaine.lan
# Connection security : STARTTLS
# Port : 587
# Authentication : Username + Password si relay authentifié requis

# Tester depuis la ligne de commande
# Zabbix utilise son propre envoi SMTP - tester via l'interface :
# Administration → Media Types → Email → Test
```

Configuration webhook Slack

```
 # Dans Administration → Media Types → Create media type
# Name : Slack
# Type : Webhook
# Parameters :
#   Message : {ALERT.MESSAGE}
#   Subject : {ALERT.SUBJECT}
#   HTTPProxy : (vide si pas de proxy)
#   URL : https://hooks.slack.com/services/VOTRE_WEBHOOK_URL
#   To : #canal-alertes-infrastructure

# Script webhook Zabbix pour Slack (à configurer dans le champ Script) :
# var Slack = {
#   sendMessage: function(url, to, subject, message) {
#     var params = JSON.stringify({ channel: to, text: subject + "
" + message });
#     var req = new HttpRequest();
#     req.addHeader("Content-Type: application/json");
#     var resp = req.post(url, params);
#     if (req.getStatus() != 200) { throw "Response code: " + req.getStatus(); }
#   }
# };
# Slack.sendMessage(value.URL, value.To, value.Subject, value.Message);
```

Configuration des escalades

Les escalades permettent d'envoyer des alertes à différents niveaux selon la durée du problème. Par exemple : alerte email immédiate à l'équipe N1, puis SMS ou appel téléphonique au responsable N2 si le problème dure plus de 30 minutes.

```
 # Dans Configuration → Actions → Create action (Trigger actions) :
# Name : Alerte critique avec escalade
# Conditions : Trigger severity >= High AND Maintenance = not in maintenance
#
# Operations (étapes d'escalade) :
# Étape 1 (0-30 min) : Send message to Users/Groups = Equipe-N1 via Email
# Étape 2 (30-60 min) : Send message to Users/Groups = Responsable-IT via Email+Slack
# Étape 3 (60+ min) : Send message to Users/Groups = DIT-Astreinte via SMS
# Recovery operations : Send recovery message (sujet : RESOLVED)
```

> Lors d'un déploiement Zabbix 7.0 pour un client du secteur santé (500 serveurs Linux, 200 Windows Server), la migration depuis Nagios a permis de réduire le nombre de faux positifs de 73 % grâce aux dépendances de triggers et aux fenêtres de maintenance automatisées. Le délai de configuration initial des 700 hôtes a été de 4 heures via l'API Zabbix et des scripts Python d'import automatique depuis l'inventaire CMDB. Les dashboards Zabbix 7.0 avec les nouveaux widgets en temps réel ont remplacé 3 outils de reporting distincts.
>
> *— Retour de déploiement Zabbix, janvier 2026*

Zabbix vs Nagios vs PRTG — quelle solution choisir ?

Le choix entre Zabbix, Nagios/Icinga2 et PRTG dépend de la taille de l'infrastructure, du budget, et des compétences de l'équipe. Voici une comparaison objective basée sur les critères les plus importants en environnement d'entreprise.

Critère	Zabbix 7.0	Nagios/Icinga2	PRTG
Licence	Open source (GPLv2)	Open source (GPLv2)	Commercial (par capteur)
Coût pour 500 hôtes	0 € (infra propre)	0 € (infra propre)	~15 000 €/an
Interface web	Moderne, personnalisable	Basique (Nagios), moderne (Icinga2)	Très complète, guidée
SNMP v3	Natif, complet	Via plugins	Natif, complet
Auto-découverte réseau	Oui (règles LLD)	Limitée	Oui (très avancé)
API REST	Oui (v7 très complète)	Partielle (Icinga2)	Oui
Scalabilité	Très haute (proxy)	Haute (distribué)	Haute (cluster)
Courbe d'apprentissage	Moyenne	Élevée (config texte)	Faible (GUI)

Pour les entreprises avec moins de 100 hôtes et des équipes sans expertise Linux, PRTG offre la mise en œuvre la plus rapide. Pour les PME et grandes entreprises recherchant une solution pérenne, open source et hautement personnalisable, Zabbix 7.0 est le choix recommandé. Nagios et Icinga2 conviennent aux équipes avec une culture DevOps et une préférence pour la configuration as code. Pour centraliser les logs de vos serveurs supervisés, notre article sur le Cloud Logging et centralisation du monitoring est complémentaire à ce guide.

Dashboards, rapports et SLA dans Zabbix 7.0

Zabbix 7.0 a significativement amélioré son module de dashboards avec des widgets dynamiques, des filtres par groupe d'hôtes et des vues en temps réel. Les rapports peuvent être générés automatiquement (en PDF) et envoyés par email selon un planning configurable.

```
 # Créer un rapport hebdomadaire automatique (dans Reports → Scheduled reports) :
# Name : Rapport hebdo infrastructure
# Dashboard : Infrastructure Overview
# Period : Weekly (lundi 9h00)
# Cycle : Weekly
# Recipients : equipe-it@mondomaine.lan

# Les SLA sont configurés dans Services → SLA :
# Créer une SLA "Serveurs Production" :
#   SLO (Service Level Objective) : 99.5%
#   Effective date : 01/01/2026
#   Schedule : 24x7
#   Services liés : Production-Web, Production-DB, Production-App
```

Pour une supervision réseau avancée via SNMPv3, les switches et routeurs peuvent être ajoutés comme hôtes Zabbix — consultez notre guide complet Zabbix : supervision des équipements réseau avec SNMPv3 sécurisé. Pour intégrer Zabbix dans une stratégie de logging centralisé et de conformité ISO 27001, notre guide sur la politique de logging ISO 27001 A.8.15 détaille les exigences complémentaires.

Questions fréquentes

Quelle est la différence entre Zabbix Agent 1 et Zabbix Agent 2 ?

Zabbix Agent 2 est réécrit en Go et apporte quatre avantages principaux : une consommation mémoire plus faible, un système de plugins extensibles (Docker, MySQL, Redis, MongoDB sont des plugins natifs), le support du mode actif avec connexions persistantes, et une meilleure gestion de la parallélisation des collectes. Pour les nouvelles installations, Zabbix Agent 2 est systématiquement recommandé. L'Agent 1 reste supporté mais ne reçoit plus de nouvelles fonctionnalités depuis Zabbix 6.0.

Combien d'hôtes peut superviser un seul serveur Zabbix ?

Un serveur Zabbix correctement dimensionné (8 cœurs, 16 Go RAM, SSD NVMe) peut superviser de 2 000 à 5 000 hôtes avec des intervalles de collecte de 60 secondes. Au-delà, le déploiement de proxies Zabbix distribués est recommandé : chaque proxy peut gérer 500 à 2 000 hôtes supplémentaires et réduit la charge sur le serveur central. La base de données est généralement le goulot d'étranglement — PostgreSQL avec TimescaleDB est préféré à MySQL au-delà de 1 000 hôtes.

Comment superviser un serveur derrière un pare-feu strict ?

En mode actif (ServerActive dans la config de l'agent), l'agent initie la connexion vers le port 10051 du serveur Zabbix. Ce mode ne nécessite que la règle sortante 10051/TCP depuis le serveur supervisé vers le serveur Zabbix, sans ouverture entrante. Pour les serveurs en DMZ ou dans le cloud, c'est la configuration recommandée. Le chiffrement TLS entre l'agent et le serveur est activable avec un certificat PKI interne (compatible avec AD CS).

Zabbix peut-il superviser des conteneurs Docker ?

Oui. Zabbix Agent 2 inclut un plugin Docker natif qui collecte automatiquement l'état des conteneurs, l'utilisation CPU/mémoire par conteneur, les statistiques réseau et les volumes. Activer le plugin en ajoutant `Plugins.Docker.Endpoint=unix:///var/run/docker.sock` dans la configuration de l'agent, et appliquer le template *Docker by Zabbix agent 2* à l'hôte dans l'interface. Zabbix peut également superviser Kubernetes via le plugin dédié et l'API Kubernetes.

Pour aller plus loin, l'ENISA propose des recommandations sur la supervision de sécurité que Zabbix peut implémenter via ses templates de sécurité.

Comment migrer depuis Nagios vers Zabbix sans interruption ?

La migration se fait en trois phases sans coupure de supervision : d'abord déployer Zabbix en parallèle et configurer les mêmes hôtes, puis valider la qualité des métriques Zabbix pendant 2 à 4 semaines, et enfin basculer les alertes vers Zabbix et désactiver Nagios. L'API Zabbix permet

d'importer en masse les hôtes depuis un export CSV ou JSON de Nagios, réduisant le temps de configuration de plusieurs jours à quelques heures. Les configurations Nagios (fichiers .cfg) ne se convertissent pas directement — les templates Zabbix remplacent les définitions de services Nagios.

Maintenance préventive et gestion des fenêtres de maintenance Zabbix

Les fenêtres de maintenance Zabbix permettent de suspendre les alertes pendant les opérations planifiées (mises à jour, redémarrages, interventions) sans perdre la collecte des métriques. Cette fonctionnalité est indispensable pour éviter les faux positifs lors des maintenances nocturnes et les alertes parasites pendant les batchs de sauvegarde.

Dans l'interface Zabbix, les maintenances se configurent dans Configuration → Maintenance. On distingue deux types : **With data collection** (les métriques continuent d'être collectées, les alertes sont suspendues) et **No data collection** (collection et alertes suspendues — utile lors d'une coupure réseau planifiée). Pour les maintenances récurrentes (sauvegarde toutes les nuits de 2h à 4h), créer une maintenance de type *Periodic monthly* avec les intervalles horaires correspondants.

```
 # Créer une maintenance via l'API Zabbix (exemple Python)
import requests
import json

ZABBIX_URL = "http://zabbix.mondomaine.lan/zabbix/api_jsonrpc.php"
ZABBIX_TOKEN = "votre_token_api_v2"  # Généré dans User Settings -> API tokens

# Créer une maintenance pour le groupe "Production Servers"
payload = {
    "jsonrpc": "2.0",
    "method": "maintenance.create",
    "params": {
        "name": "Maintenance nuit patch",
        "active_since": 1750204800,  # timestamp Unix début
        "active_till": 1750291200,   # timestamp Unix fin
        "hostids": [],               # vide = tous les hôtes du groupe
        "groupids": ["12"],          # ID du groupe "Production Servers"
        "maintenance_type": 0,       # 0 = with data collection
        "timeperiods": [{
            "timeperiod_type": 0,    # 0 = one time only
            "start_date": 1750204800,
            "period": 7200           # durée en secondes (2 heures)
        }]
    },
    "id": 1,
    "auth": ZABBIX_TOKEN
}

response = requests.post(ZABBIX_URL, json=payload)
print(response.json())
```

Sécurisation de l'installation Zabbix

Un serveur Zabbix exposé sans durcissement est une cible attractive : il connaît l'ensemble de l'infrastructure, dispose d'accès aux agents sur tous les serveurs supervisés, et peut exécuter des commandes à distance via Remote Commands. Les mesures de sécurisation minimales sont les suivantes.

Premièrement, changer immédiatement le mot de passe du compte **Admin** Zabbix après la première connexion et créer des comptes nominatifs pour chaque administrateur avec le rôle approprié (Super Admin uniquement pour les opérations de configuration, Admin pour la gestion quotidienne, User pour la consultation). Deuxièmement, restreindre l'accès à l'interface web Zabbix aux seules IP du réseau d'administration via la configuration Apache ou Nginx. Troisièmement, activer l'authentification LDAP ou SAML pour centraliser la gestion des accès dans l'Active Directory.

```
 # Restreindre l'accès à l'interface Zabbix (Apache - ajouter dans le VirtualHost)
#
#     Order deny,allow
#     Deny from all
#     Allow from 192.168.100.0/24   # Réseau d'administration
#     Allow from 10.0.0.0/8         # VPN
#

# Activer le chiffrement TLS entre Zabbix Server et les agents
# Sur le serveur Zabbix, générer un PSK (Pre-Shared Key) pour les agents
openssl rand -hex 32 > /etc/zabbix/zabbix_agent2.psk
chmod 600 /etc/zabbix/zabbix_agent2.psk
cat /etc/zabbix/zabbix_agent2.psk
# Résultat : une clé hex de 64 caractères, ex: a7f3c9e12...

# Dans zabbix_agent2.conf :
# TLSConnect=psk
# TLSAccept=psk
# TLSPSKIdentity=MonAgentPSK-srv-web01
# TLSPSKFile=/etc/zabbix/zabbix_agent2.psk

# Dans l'interface Zabbix, Host -> Encryption :
# Connections to host : PSK
# PSK identity : MonAgentPSK-srv-web01
# PSK : [valeur du fichier .psk]
```

Pour aller plus loin dans la sécurisation de vos serveurs supervisés, le déploiement d'un SIEM comme Wazuh en complément de Zabbix permet d'analyser les logs de sécurité collectés. Notre guide Wazuh 2026 : SIEM/XDR Open Source — Guide Déploiement détaille cette architecture complémentaire. Les deux outils sont compatibles et se complètent : Zabbix pour les métriques de performance et disponibilité, Wazuh pour la détection d'incidents de sécurité et la conformité.

Optimisation des performances Zabbix pour les grandes infrastructures

Lorsque le nombre d'hôtes supervisés dépasse 300-500, des ajustements de configuration s'imposent pour maintenir des performances correctes. Les principaux leviers d'optimisation concernent la base de données (partitionnement de l'historique), le serveur Zabbix (threads et caches), et l'architecture (proxies distribués).

```
 # Optimisation MySQL pour Zabbix (à ajouter dans /etc/mysql/mysql.conf.d/mysqld.cnf)
# innodb_buffer_pool_size = 8G         # 60-70% de la RAM disponible
# innodb_log_file_size = 1G
# innodb_flush_log_at_trx_commit = 2   # Performance vs durabilité (acceptable pour Zabbix)
# innodb_flush_method = O_DIRECT
# max_connections = 500
# query_cache_type = 0                 # Désactiver le query cache (obsolète MySQL 8)

# Nettoyer l'historique ancienne (Housekeeper Zabbix)
# Dans Administration -> General -> Housekeeping :
# History storage period : 90 days (par défaut : 365 - trop long pour les grosses infras)
# Trend storage period : 365 days
# Events : 365 days

# Optimisation des threads Zabbix Server
# Dans /etc/zabbix/zabbix_server.conf :
# StartPollers=20              # Pour 500+ hôtes
# StartPollersUnreachable=4
# StartTrappers=10
# StartPingers=5
# StartDiscoverers=5
# StartPreprocessors=15        # Threads de pré-traitement des données
# CacheSize=256M               # Cache configuration objects
# HistoryCacheSize=128M        # Cache historique avant écriture DB
# ValueCacheSize=64M           # Cache valeurs pour les triggers

systemctl restart zabbix-server
```

