

SUPERVISION

Mise en place d'une infrastructure de supervision avec Zabbix, Grafana et Observium dans un environnement virtualisé



Grafana



Introduction & Contexte.....	3
Topologie & Plan d'adressage.....	4
Préparation des serveurs.....	5
Installation de la VM Windows.....	5
Installation des VM Linux (Debian).....	5
Installation du serveur WEB + BDD.....	6
Étape 1 – Mise en place de Zabbix.....	7
Installation du serveur Zabbix.....	8
Étape 2 – Enrichissement avec Grafana et Observium.....	33
Mise en place de Grafana.....	33
Mise en place d'Observium.....	36
Sécurité.....	40
Résultats et Tests.....	46

Introduction & Contexte

Dans le domaine de l'informatique, la disponibilité et la performance des systèmes sont des enjeux majeurs pour les entreprises. Les applications web, bases de données et services réseau doivent rester accessibles en permanence afin de garantir la continuité d'activité. La supervision est donc un élément essentiel, car elle permet de détecter rapidement les dysfonctionnements, d'anticiper les incidents et de faciliter le travail des administrateurs systèmes.

Dans le cadre de ce projet de BTS SIO option SISR, l'objectif est de concevoir et de mettre en place une infrastructure complète de supervision au sein d'un environnement virtualisé. Cette maquette devra surveiller différents services, générer des alertes en cas de problème et proposer des interfaces de visualisation adaptées, aussi bien pour un administrateur technique que pour un responsable non technique.

Le projet se déroule dans un environnement virtualisé avec VMware Workstation, et repose sur plusieurs machines virtuelles dédiées :

- Un serveur Linux Web + Base de données : hébergeant un site web (Apache ou Nginx) et une base MySQL/MariaDB.
- Un serveur Linux Zabbix : assurant la supervision centralisée des ressources et des services.
- Un serveur Linux Grafana : permettant de mettre en place des tableaux de bord dynamiques et personnalisés à partir des données de Zabbix.
- Un serveur Linux Observium : dédié à la supervision via SNMP et à la comparaison avec Zabbix.
- Un poste client Windows : utilisé pour accéder aux interfaces web de supervision et réaliser la démonstration.

Le projet est divisé en deux grandes étapes :

- Déploiement de Zabbix : mise en place d'un serveur de supervision centralisé, installation des agents sur les machines, configuration des métriques, définition des seuils d'alerte et simulation de pannes (arrêt de services, surcharge, tentatives SSH).
- Enrichissement avec Grafana et Observium : création de dashboards dynamiques, supervision complémentaire via SNMP, et analyse comparative entre les outils.

À l'issue du projet, la maquette devra démontrer la capacité à détecter différents types d'incidents, à générer des alertes adaptées, et à présenter des visualisations claires pour faciliter la gestion des systèmes.

Topologie & Plan d'adressage

Schéma réseau (machines, IP fixes).

Tableau du plan d'adressage IP.

Machine	Services	IP (fixes)
Poste client Windows	Navigateur	10.0.0.10/24
Serveur WEB + BDD	Apache, MySQL	10.0.0.20/24
Serveur Zabbix	Zabbix server + frontend	10.0.0.30/24
Serveur Grafana	Grafana	10.0.0.40/24
Serveur Observium	Observium	10.0.0.50/24

Rappel des ressources attribuées dans VMware :

Machine	OS	CPU	RAM	Disque
Poste client Windows	Windows 11	2vCPU	2 Go	40 Go
Serveur WEB + BDD	Debian 12	2vCPU	2 Go	20 Go
Serveur Zabbix	Debian 12	2vCPU	2 Go	20 Go
Serveur Grafana	Debian 12	2vCPU	2 Go	20 Go
Serveur Observium	Debian 12	2vCPU	2 Go	20 Go

Préparation des serveurs

1. Installation de la VM Windows

Bypass Connexion Microsoft Windows 11 :

Il faut désactiver la connexion internet en amont.

Sur l'écran connexion à un réseau :

Shift + F10

CMD : `OOBE\BYPASSNRO`

Pour configurer l'IP fixe :

Win + R > ncpa.cpl > Clique droit sur la carte réseau NAT > **Propriétés > Protocol IPv4**



Choisir une IP conforme aux paramètres Virtual Network sur VMware

Associer la bonne passerelle Virtual Network pour l'accès au réseau internet

2. Installation des VM Linux (Debian)

Au début de l'installation on choisit "**Install**" pour ne seulement avoir l'installation minimal, nous n'avons pas besoin de l'installation graphique pour nos différents services.

Choix des paramètres :

Langue : **Français**

Utilisateur : user/user

Partitionnement : **Guided - Use entire disk**

Sélection des paquets : On coche uniquement "**SSH server**" "**Standard system utilities**" (Pas besoin d'environnement graphique)

Miroirs : **deb.debian.org**

Bootloader GRUB : oui, sur **/dev/sda**

Quand l'installation est terminé :

Mettre à jour la machine : `apt update && apt upgrade -y`

Outils de base : `apt install -y vim sudo curl wget net-tools unzip git`

On peut ensuite cloner la machine 4 fois et les renommer pour chaque service :

Debian 12 SRV-Web

Debian 12 SRV-Zabbix

Debian 12 SRV-Grafana

Debian 12 SRV-Observium

Pour configurer l'IP fixe :

`vim /etc/network/interfaces`

Exemple de configuration pour la machine SRV-Web

```
# The primary network interface
auto ens33
iface ens33 inet static
    address 10.10.0.20
    netmask 255.255.255.0
    gateway 10.10.0.253
    dns-nameservers 1.1.1.1 8.8.8.8_
~
```

On redémarre le service réseau et on vérifie :

`systemctl restart networking`

`ip a`

Pour changer le hostname :

`hostnamectl set-hostname debian-web`

Tester la connectivité entre les machines :

On peut ensuite tester un ping entre les machines pour voir si tout fonctionne.

Si le ping Debian → Windows ne passe pas mais le ping Windows → Debian est OK, il faut vérifier le pare-feu Windows.

Dans un PowerShell en administrateur, activer les règles ICMP entrantes :

`netsh advfirewall firewall add rule name="Autoriser ICMPv4-In"`

`protocol=icmpv4:8,any dir=in action=allow`

3. Installation du serveur WEB + BDD

Installer Apache :

`apt install -y apache2`

```
systemctl enable apache2  
systemctl status apache2
```

Vérifier que le service fonctionne sur la VM Windows :

```
http://10.10.0.20
```

On doit voir la page par défaut de d'Apache dans le navigateur

Installer MariaDB (MySQL)

```
apt install -y mariadb-server  
systemctl enable mariadb  
systemctl status mariadb
```

Sécuriser l'installation :

```
mysql_secure_installation
```

Réponses recommandées :

Définir un mot de passe root : **OUI**

Supprimer utilisateurs anonymes : **OUI**

Interdire root en accès distant : **OUI**

Supprimer la base de test : **OUI**

Recharger les privilèges : **OUI**

Étape 1 – Mise en place de Zabbix

1. Installation du serveur Zabbix

Zabbix frontend est une appli web (PHP) et Zabbix Server stocke ses données en BDD. L'installation d'un LAMP est alors nécessaire.

Préparation de la pile LAMP :

```
apt install -y apache2 mariadb-server  
apt install -y php php-mysql php-xml php-gd php-bcmath php-ldap php-zip  
php-json php-cli libapache2-mod-php  
systemctl enable apache2 mariadb
```

On vérifie ensuite si la page d'accueil s'affiche bien sur notre poste client à l'adresse de la machine (10.10.0.30).

Sécuriser l'installation :

```
mysql_secure_installation
```

Réponses recommandées :

- Définir un mot de passe root : OUI
- Supprimer utilisateurs anonymes : OUI
- Interdire root en accès distant : OUI
- Supprimer la base de test : OUI
- Recharger les privilèges : OUI

Préparation de la base de données Zabbix :

Zabbix à besoin d'une base de données pour stocker les hôtes, les métriques collectées, les triggers, alertes, historique, etc

On se connecte à MariaDB pour créer la base et l'utilisateur Zabbix :

```
mysql -u root -p
```

On crée ensuite la base :

```
CREATE DATABASE zabbix CHARACTER SET utf8mb4 COLLATE utf8mb4_bin;  
CREATE USER 'zabbixuser'@'localhost' IDENTIFIED BY 'Zabbix!';  
GRANT ALL PRIVILEGES ON zabbix.* TO 'zabbixuser'@'localhost';  
FLUSH PRIVILEGES;  
EXIT;
```

On choisit un encodage en utf8mb4 qui permet le support des symboles et des alphabets non-latins. On choisit aussi un mot de passe solide.

On peut aussi vérifier la création avec :

```
SHOW DATABASES;  
SELECT User, Host FROM mysql.user;
```

Installation Zabbix (serveur+frontend+agent) :

Zabbix Official Repository : [Lien](#) (debian)

On ajoute le dépôt officiel de Zabbix :

```
#Télécharge le dépôt officiel pour Debian 12  
wget  
https://repo.zabbix.com/zabbix/7.0/debian/pool/main/z/zabbix-release/zabbix-release  
_latest+debian12_all.deb  
#Installe le paquet du dépôt  
dpkg -i zabbix-release_latest+debian12_all.deb  
#Mets à jour la liste des paquets  
sudo apt update
```

On récupère les paquets du dépôt Debian :

```
apt install -y zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf  
zabbix-sql-scripts zabbix-agent2
```

On importe le schéma SQL de Zabbix dans la base :

Il sert à créer toutes les tables, index, données initiales

```
zcat /usr/share/zabbix-sql-scripts/mysql/server.sql.gz | mysql -u zabbixuser -p zabbix
```

Pour vérifier :

```
mysql -u zabbixuser -p zabbix  
SHOW TABLES;
```

On voit bien que 203 tables ont été créées

```
triggers  
ugset  
ugset_group  
user_scim_group  
user_ugset  
userdirectory  
userdirectory_idpgrp  
userdirectory_ldap  
userdirectory_media  
userdirectory_saml  
userdirectory_usrgrp  
users  
users_groups  
usrgrp  
valuemap  
valuemap_mapping  
widget  
widget_field  
+-----+  
203 rows in set (0,001 sec)
```

On donne ensuite l'accès de BDD créé à Zabbix :

`vim /etc/zabbix/zabbix_server.conf`

On cherche les lignes DBName - DBUser - DBPassword pour rentrer nos identifiants.

```
DBName=zabbix
### Option: DBSchema
#       Schema name, Used for PostgreSQL.
#
# Mandatory: no
# Default:
# DBSchema=

### Option: DBUser
#       Database user.
#
# Mandatory: no
# Default:
# DBUser=

DBUser=zabbixuser
DBPassword=Zabbix!
```

On enregistre puis on démarre et active tous les services :

```
systemctl enable zabbix-server zabbix-agent2 apache2 mariadb
systemctl restart zabbix-server zabbix-agent2 apache2 mariadb
```

On vérifie le statut de Zabbix :

```
systemctl status zabbix-server
```

```
● zabbix-server.service - Zabbix Server
   Loaded: loaded (/lib/systemd/system/zabbix-server.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-09-24 11:38:04 CEST; 2min 45s ago
     Process: 3181 ExecStart=/usr/sbin/zabbix_server -c $CONFFILE (code=exited, status=0/SUCCESS)
    Main PID: 3183 (zabbix_server)
      Tasks: 77 (limit: 2250)
     Memory: 79.6M
        CPU: 1.137s
```

La commande doit afficher active (running)

2. Configuration de l'interface web Zabbix

Depuis le poste client Windows :

```
http://10.10.0.30/zabbix
```

On configure avec l'assistant :

Language → Français

Vérification des prérequis → tout doit être OK

Configurer la connexion à la base de données :



Configurer la connexion à la base de données

Veuillez créer la base de données monitoring et configurer les paramètres de connexion. Appuyez sur le bouton "Prochaine étape" quand c'est fait.

Statut

Vérification des prérequis

Configurer la connexion à la base de données

Paramètres

Résumé pré-installation

Installer

Type de base de données: MySQL

Hôte base de données: localhost

Port de la base de données: 3306 (utiliser le port par défaut)

Nom de la base de données: zabbix

Choisir un identifiant utilisateur dans: Local host Côté serveur Côté client

Utilisateur: zabbixuser

Mot de passe: -----

Chiffrement TLS de la base de données: La connexion ne sera pas chiffrée car elle utilise un fichier socket pour l'accès ou de la méthode pipe (Windows).

Retour

Prochaine étape

Paramètres → Choisir un nom + Fuseau horaire

On a une page de récapitulation, on finit l'installation si tout est bon.



Résumé pré-installation

Veuillez vérifier les paramètres de configuration. Si tout est correct, appuyez sur le bouton "Prochaine étape". Sinon, le bouton "Retour" pour changer les paramètres.

Statut

Vérification des prérequis

Configurer la connexion à la base de données

Paramètres

Résumé pré-installation

Installer

Type de base de données: MySQL

Service base de données: localhost

Port de la base de données: défaut

Nom de la base de données: zabbix

Utilisateur base de données: zabbixuser

Mot de passe utilisateur de la base de données: -----

Chiffrement TLS de la base de données: non

Nom du serveur Zabbix: TP-SuperAdmin

Retour

Prochaine étape

Page de connexion (Identifiant par défaut) :

Utilisateur : Admin

MDP : zabbix

Pour changer le mot de passe :

Paramètres utilisateur > Profil > Changer le mot de passe

Aparté sur le changement de langue :

Dans les paramètres Zabbix, **Administration > General > GUI**

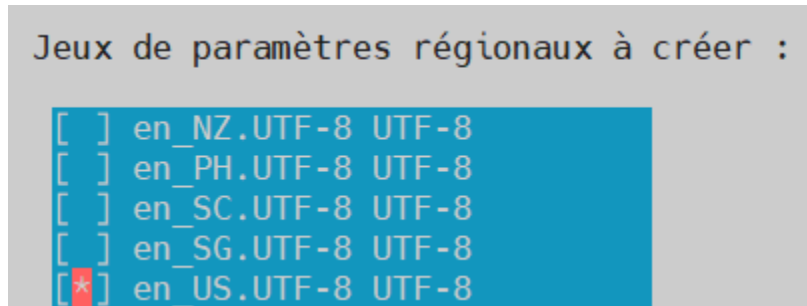
Si cette erreur apparaît pour un changement de langue



Sur Debian on reconfigure les langues locales :

`dpkg-reconfigure locales`

On coche **en_US.UTF-8 UTF-8**



On choisit ensuite notre langue par défaut **fr_FR.UTF-8**.

On recharge les services :

`locale-gen`

`systemctl restart apache2`

Vérification :

`locale -a`

Supervision du serveur Zabbix lui-même

C'est pour avoir des métriques sur debian-zabbix

Configuration de l'agent sur la VM Zabbix :

Il faut modifier le fichier conf de l'agent2, "Server" et "ServerActive" doivent correspondre au serveur Zabbix, "Hostname" pour identifier la machine dans Zabbix.

`vim /etc/zabbix/zabbix_agent2.conf`

Réglages pour notre cas :

Server=127.0.0.1,10.10.0.30

ServerActive=127.0.0.1,10.10.0.30

Hostname=svr-zabbix

Rechargement du service :

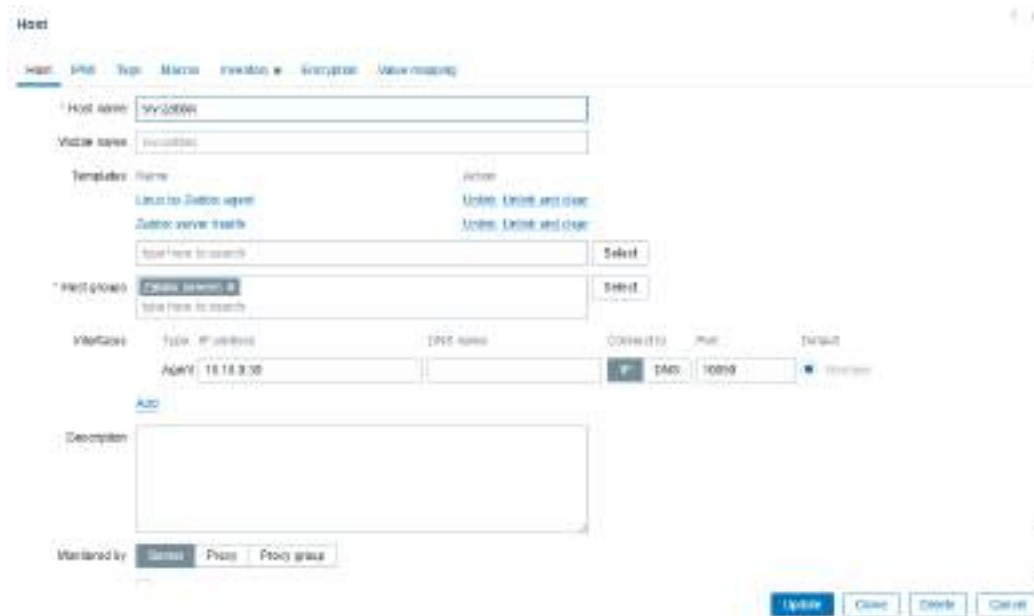
```
systemctl restart zabbix-agent2
```

Puis dans l'interface Zabbix :

Collecte de données > Hôtes

On a un hôte Zabbix server par défaut qu'on modifie avec les bons paramètres précédemment configuré dans le fichier zabbix_agent2.conf :

Il est important d'ajouter les bons templates en fonction des données qu'on souhaite récolter.



Ensuite dans **Surveillance > Dernières données**

On sélectionne notre hôte et on aura toutes les données récupérer en temps réel :

Host	Item name	Current value	Current error	Change	Unit	Graph
srv20001	Available memory	284	1.18 GB	-124 KB		Graph
srv20001	Available memory in %	27%	63.481 %	+3.8058 %		Graph
srv20001	Free space of /tmp	284 MB	96080564 MB			Graph
srv20001	Configuration cache % used	284	20.000 %			Graph
srv20001	Connected clients					Graph

Ajouter le serveur Web/BDD

On configure l'agent sur la VM Web/BDD :

```
apt install -y zabbix-agent2
```

```
vim /etc/zabbix/zabbix_agent2.conf
```

Réglages :

Server=10.10.0.30
ServerActive=10.10.0.30
Hostname=srv-web

Dans l'interface Zabbix :

Collecte de données > Hôtes > Créer un hôte

Hosts	IP	OS	Platform
Agent	10.10.0.30	Linux	Ubuntu

Groups	Name	IPs	Platform
Group 1	Group 1		

Si tout fonctionne, l'Etat et la Disponibilité doivent passer en vert

Host	Alias	Groups	IPs	OS	Platform	State	Availability	Unreachable for	Agent	Info	Tags
srv-web	Démarco	Démarco	10.10.0.30	Linux	Ubuntu	OK	OK		OK		
srv-web	Démarco	Démarco	10.10.0.30	Linux	Ubuntu	OK	OK		OK		

Ensuite dans **Surveillance > Dernières données**



Time	Severity	Received time	Status	Host	Item	Process	Current value	Units	Actions	Tags
10:14:52 AM	Information	10:15:26 AM	OK	srv-web	Apache: Service has been restarted (uptime > 1 day)	OK	OK	seconds	OK	
10:14:56 AM	Warning	10:18:27 AM	OK	srv-web	Uptime: uptime has been restarted (uptime > 1 day)	OK	OK	seconds	OK	
10:00	Problem		PROBLEM	srv-web	MySQL: Failed to fetch info about (no my_data for id)	OK	OK	seconds	OK	

(conseil configuration...)

🌟 1 Vérifier les templates utilisés

Sur ton `srv-web` (10.10.0.20) :

Tu as déjà :

-  `Linux by Zabbix agent` → parfait pour CPU, RAM, disque, uptime, etc.
-  `Apache by Zabbix agent` → parfait pour le service Web (Apache).

👉 Il te manque le template MySQL pour superviser la base de données.

+ À ajouter :

1. Clique sur ton host `srv-web` → Configuration → Hosts → `srv-web` → Templates → Select.
2. Dans la liste, cherche :
 - `MySQL by Zabbix agent 2`
3. Coche-le puis clique `Select` → `Update`.

Ce template interrogera MySQL (MariaDB) via l'agent.



🌟 2 Préparer l'accès à MySQL pour l'agent Zabbix

Le template MySQL a besoin d'un accès pour récupérer les métriques (nombre de connexions, threads, requêtes...).

Sur `srv-web`, fais :

```
bash
sudo mkdir -p /var/lib/zabbix
sudo nano /var/lib/zabbix/.my.cnf
```

 Copier le code

Colle dedans :

```
ini
[client]
user=appuser
password=MotDePasseSo11de1
```

 Copier le code

(remplace avec le compte MySQL que tu as créé précédemment)



(remplace avec le compte MySQL que tu as créé précédemment)

Ensuite :

```
bash

sudo chown zabbix:zabbix /var/lib/zabbix/.my.cnf
sudo chmod 640 /var/lib/zabbix/.my.cnf
sudo systemctl restart zabbix-agent2
```

 Copier le code

👉 L'agent aura alors les droits pour se connecter à ta base et envoyer les infos à Zabbix.

🌸 3 Vérifier que tout remonte

Dans Monitoring → Latest data → srv-web, tu devrais voir :

- System performance (CPU, mémoire, disque) → du template *Linux*
 - Apache statistics → du template *Apache*
 - MySQL metrics → du template *MySQL by agent2*
- ⌚ Attends 2 à 5 minutes que les premiers relevés aient ↓.

🌸 4 Ajouter les métriques liées aux tentatives SSH échouées

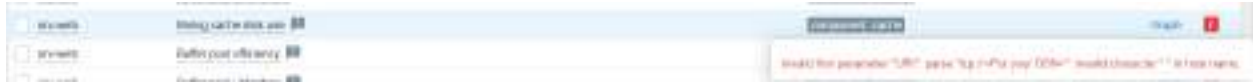
C'est une demande spécifique du projet (cf. document).

Tu peux la créer manuellement :

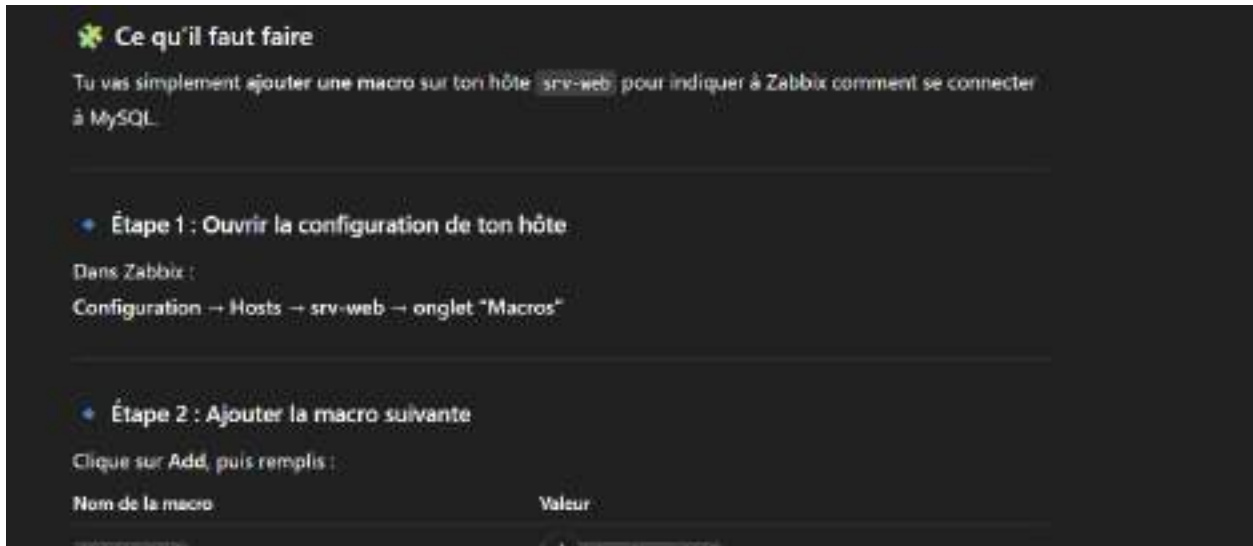
+ Créer un item pour le log SSH

Sur `srv-web` :

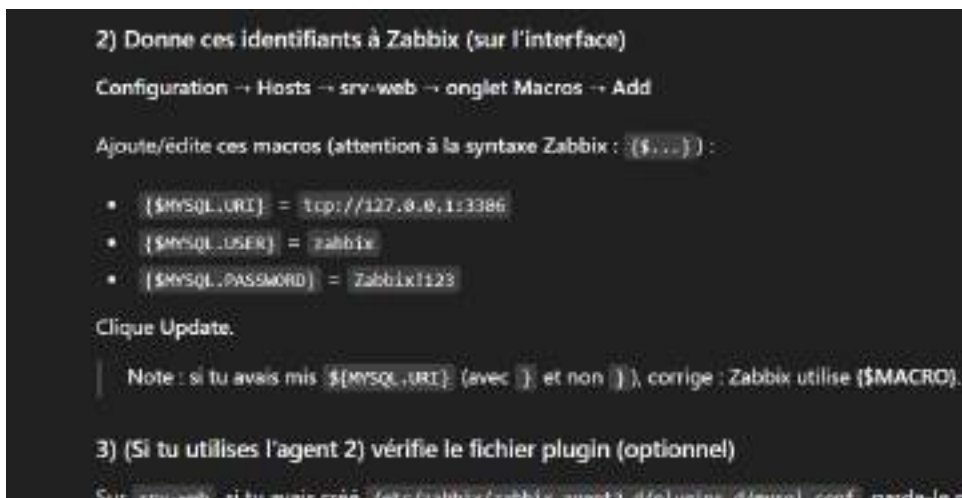
1. Configuration → Hosts → srv-web → Items → Create item
2. Renseigne :
 - Name : Tentatives SSH échouées
 - Type : zabbix-agent (active)
 - Key : log[/var/log/auth.log,Failed password,UTF-8]
 - Type of information : Log
 - Update interval : 1m
 - Log time format : %b %d %H:%M:%S
 - Applications : Sécurité (ou crée-en une). ↓
3. Clique sur Add.



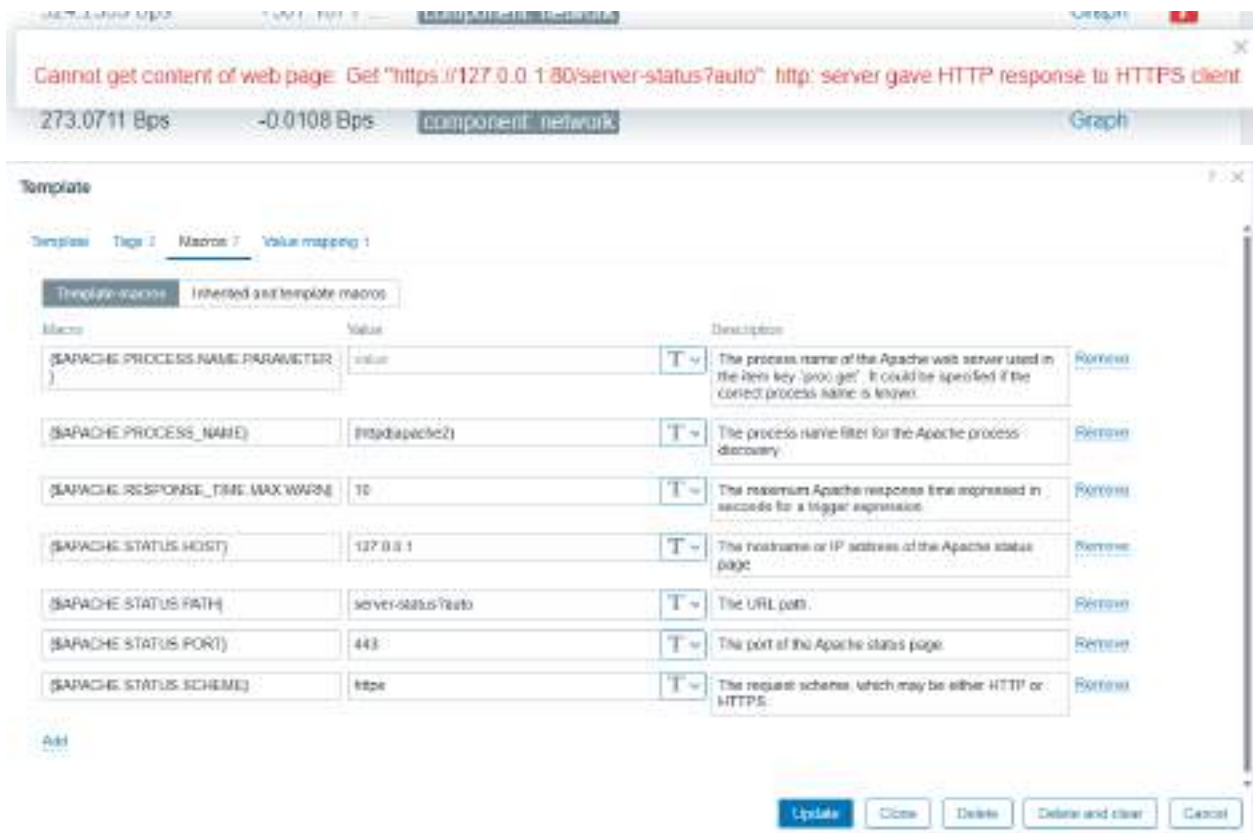
(chat trame)



(chat conseil config..)



FONCTIONNE



(chat erreur zabbix)

2) Corrige les macros du template sur l'hôte dans Zabbix

Aller : Configuration → Hosts → (ta VM Apache) → Macros.

- Cas HTTPS (souvent après ta bascule globale)


```
bash
{APACHE.STATUS.SCHEME} = https
{APACHE.STATUS.PORT} = 443
{APACHE.STATUS.HOST} = 10.10.0.50
```

Dans l'item "HTTP agent", mets Verify peer = No et Verify host = No si le cert est auto-signé.
- Cas HTTP (si tu gardes /server-status en clair)


```
bash
{APACHE.STATUS.SCHEME} = http
{APACHE.STATUS.PORT} = 80
{APACHE.STATUS.HOST} = 10.10.0.50
```

Ne mets jamais https://...:80 et n'utilise pas 127.0.0.1 si Zabbix et Apache ne sont pas sur la même

(chat conseil config..)

Solution rapide : indiquer la vraie connexion MySQL à Zabbix Agent 2

Sur la VM srv-web (celle où tourne MySQL + zabbix-agent2) :

Ouvre le fichier de config de l'agent 2 :

```
sudo nano /etc/zabbix/zabbix_agent2.d/plugins.d/mysql.conf
```

Mets-y ce contenu complet :

```
Plugins.Mysql.Sessions.ToLocal.Type=tcp
Plugins.Mysql.Sessions.ToLocal.Uri=tcp://127.0.0.1:3306
Plugins.Mysql.Sessions.ToLocal.User=zabbix
Plugins.Mysql.Sessions.ToLocal.Password=zabbix_pass
```

⚠ (remplace `zabbix_pass` par le vrai mot de passe & "on utilisateur `zabbix` dans MySQL)

```
Plugins.Mysql.Sessions.ToLocal.Type=tcp
Plugins.Mysql.Sessions.ToLocal.Uri=tcp://127.0.0.1:3306
Plugins.Mysql.Sessions.ToLocal.User=webuser
Plugins.Mysql.Sessions.ToLocal.Password=Webuser!
```

```
2025/10/08 02:56:54.001165 using plugin: 'zabbixagent' (built-in) providing following interfaces: 'exporter', 'configuration'
2025/10/08 02:56:54.002294 lowering the plugin 'zabbixagent' capacity to 1 as the configured capacity 100 exceeds limits
2025/10/08 02:56:54.002322 using plugin: 'zabbixagent' (built-in) providing following interfaces: 'exporter'
2025/10/08 02:56:54.003324 cannot create scheduling manager: 'mysql' plugin MySQL configuration: 'cannot assign configuration: local() parameter' Plugins.Mysql.
Sessions.ToLocal.Type at Line 8: 'cannot parameter'
root@zabbix-agent-4:~#
```

```
Plugins.Mysql.Sessions.ToLocal.Uri=tcp://127.0.0.1:3306
Plugins.Mysql.Sessions.ToLocal.User=webuser
Plugins.Mysql.Sessions.ToLocal.Password=Webuser!
```

Tu peux organiser ton tableau en 3 blocs (très clair pour un jury) :

🌿 Bloc 1 – Vue d'ensemble (haut du dashboard)

But : donner un aperçu rapide de l'état général du système.

Widgets à ajouter :

- Clock (heure du serveur Zabbix)
- System status → affiche le nombre d'hôtes up/down.
- Problems → liste en temps réel les alertes actives (pannes, seuils dépassés).

👉 Positionne-les horizontalement tout en haut.

C'est ton "tableau de bord général".

(chat trame) Pour DASHBOARD ZABBIX

(chat trame)

🌿 📌 Ajouter les métriques liées aux tentatives SSH échouées

C'est une demande spécifique du projet (cf. document).

Tu peux la créer manuellement :

+ Créer un item pour le log SSH

Sur `srv-web` :

1. Configuration → Hosts → `srv-web` → Items → Create item
2. Renseigne :
 - Name : `Tentatives SSH échouées`
 - Type : `Zabbix agent (active)`
 - Key : `log[/var/log/auth.log,Failed password,UTF-8]`
 - Type of information : `Log`
 - Update interval : `1m`
 - Log time format : `%b %d %H:%M:%S`
 - Applications : `sécurité` (ou crée-en une). ↓
3. Clique sur Add.

New Trigger

Trigger Tags Dependencies

Name: too many failed SSH attempts (>5 in 5 minutes)

Event name: too many failed SSH attempts (>5 in 5 minutes)

Operational data:

Severity: Not classified Information **Warning** Average High Disaster

Expression: `[ssh=web:log[/var/log/auth.log,malicious:password:non-4].count(1h,, "loggrep") (>5]` Add

Expression constructor

OK event generation: Expression Recovery expression None

PROBLEM event generation mode: Single Multiple

OK event closer: All problems All problems if log values match

Allow manual close:

Menu entry name: Trigger URL

Add Cancel

Mais attention ⚠

Comme il est marqué "Dependent item", tu ne peux pas changer directement son intervalle, car il dépend d'un autre item "maître".

🔍 Donc il faut modifier l'item parent

L'item maître est celui qui collecte réellement les données :

Linux by Zabbix agent: CPU user time

Key: `system.cpu.util[,user]`

C'est celui-là (et aussi les autres `system.cpu.util[...]`) qui font le vrai travail de collecte CPU.

⚙ Ce que tu fais :

1. Clique sur "Linux by Zabbix agent: CPU user time"
2. Dans le champ Update interval, change 1h → 30s (ou 30s si tu veux être raisonnable)
3. Clique Update
4. Tu peux faire la même chose pour :
 - `system.cpu.util[,system]`
 - `system.cpu.util[,ioact]`
 - `system.cpu.util[,idle]`

(Ce n'est pas obligatoire pour tous, mais ça rend la mesure globale CPU plus précise)

Suivi MySQL
Alerte "tentatives SSH échouées"

Scénarios de panne
Durcissement rapide & bonnes pratiques
Dépannage express

Pour les trigger :

Excellente question Enzo 🏆
Tu vas voir, c'est simple une fois qu'on comprend la logique d'un trigger dans Zabbix.
Je t'explique clairement et pas à pas comment créer un trigger qui t'alerte quand le CPU dépasse 80 %.

⚙️ **Objectif**
Créer un trigger du type :
"Alerte : utilisation CPU supérieure à 80 %"

📌 **Étape 1 — Aller dans les Items du host**

1. Va dans **Configuration** → **Hosts**
2. Clique sur le nom de ton PC (la machine supervisée)
3. Clique sur l'onglet **Items**
4. Repère l'item :
`Linux by Zabbix agent: CPU idle time: CPU utilization`
(Key : `system.cpu.util`)

📌 C'est celui qui te donne l'utilisation totale du CPU.

⚠️ **Étape 2 — Aller dans "Triggers"**

1. Clique sur l'onglet **Triggers** (toujours dans ton host)
2. Clique sur **Create trigger**

↓

✦ Étape 3 — Remplis les champs

Champ	Valeur à mettre
Nom	<code>CPU usage > 80%</code>
Severity	<code>warning</code> ou <code>high</code> selon ton choix
Expression	Clique sur "Add" (à droite du champ)

📖 Étape 4 — Crée l'expression du trigger

1. Dans la fenêtre "Add" :

- Clique sur `Select → Item`
- Choisis ton item : `linux by zabbix agent: CPU idle time: CPU utilization`

2. Dans `Function`, choisis :

```
9000  
  
last()
```

(pour prendre la dernière valeur mesurée)

3. Ensuite dans `Expression`, modifie comme ceci :

```
05  
  
{TemplateDeHost:system.cpu.ut11.last()}>80
```

+ Poser une question

```
[Toolbox@host:system.cpu.util.last()]>80
```

(Zabbix le mettra automatiquement si tu choisis bien l'item et la fonction.)

4. Clique sur **Insert**, puis **Add**.

Étape 5 — Sauvegarde

1. Clique sur **Add** en bas de la page du trigger
2. Ton nouveau trigger s'affiche maintenant dans la liste

Étape 6 — Vérifie

Quand la charge CPU dépasse 80 % (tu peux tester avec `stress-ng` par exemple) :

- Le trigger passe à **PROBLEM**
- Tu verras une alerte rouge/orange dans **Monitoring — Problems**
- Grafana (si connecté à Zabbix) peut aussi afficher une alerte

Variante — Moyenne sur plusieurs minutes

Si tu veux éviter les pics instantanés, tu peux utiliser la moyenne sur 5 minutes :

```
zabbix
```

```
(Toolbox@Host:system.cpu.util.avg(5s))>80
```

☑ Cela signifie "si la moyenne du CPU sur les 5 dern^{rs} minutes dépasse 80 %".

New trigger

Trigger | Tags | Dependences

Name: CPU Usage = 80%

Event name: CPU Usage = 80%

Operational data:

Severity: Not classified | Information | **Warning** | Average | High | Disaster

Expression: `last(/usr/web/nginx/system.cpu.util)>80` Add

Expression constructor

OK event generation: Expression | Recovery-expression | None

PROBLEM event generation mode: Single | Multiple

OK event closes: All problems | All problems if tag values match

Allow manual close:

Menu entry name:

Add Cancel

Relis les autres chat pour tout le contexte également et explique moi bien comment faire pour les tentatives ssh échouées sachant que je peux me connecter la à distance toujours

Réflexion durant 14s

Parfait, on le fait proprement de A → Z pour ton `srv-web` (10.10.0.20), en tenant compte de tout ce qu'on a déjà vu dans les autres fils.

1) Préparer le log côté `srv-web`

Sur `srv-web` (la cible qui écrit le log) :

```
bash Copier le code  
  
# 1) Vérifie quel fichier existe  
ls -l /var/log/auth.log 2>/dev/null || echo "auth.log absent"  
ls -l /var/log/secure 2>/dev/null || echo "secure absent"  
  
# 2) Si aucun des deux n'existe (journald-only) → installe rsyslog  
apt-get update && apt-get install -y rsyslog # Debian/Ubuntu  
# (dnf install -y rsyslog pour RHEL/Rocky)  
systemctl enable --now rsyslog
```

- Debian/Ubuntu → on utilise `/var/log/auth.log`.
Ajoute l'utilisateur `zabbix` au groupe `adm` pour lire le fichier :

```
bash Copier le code
```

- Debian/Ubuntu → on utilise `/var/log/auth.log`
Ajoute l'utilisateur `zabbix` au groupe `adm` pour lire le fichier :

```
bash
usermod -sG adm zabbix
systemctl restart zabbix-agent
```

Copier le code

- RHEL/Rocky → on utilise `/var/log/secure` (pas besoin du groupe).

2) Créer l'item Zabbix (sur l'hôte `srv-web`)

Chemin Zabbix : Configuration → Hosts → `srv-web` → Items → Create item

- Name : `Failed SSH attempts`
- Type : `Zabbix agent (active)`
- Key :
 - Debian/Ubuntu : `log[/var/log/auth.log,Failed password,UTF-8]`
 - RHEL/Rocky : `log[/var/log/secure,Failed password,UTF-8]`
- Type of information : `Log`
- Update interval : `3m`
- Log time format : `%b %d %s:%t:%S`
- Applications/Tags : `security` (ou "Sécurité")

Enregistre.

3) Le trigger (>5 échecs en 5 minutes) ↓

Configuration → Hosts → `srv-web` → Triggers → Create trigger

commande pour

↳ cliquer sur `allow` manuellement pour pouvoir fermer le problème à la main en 10.

Enregistra.

4) Générer les échecs (depuis n'importe quelle machine qui joint `srv-web`)

Tu peux le faire depuis ta machine distante, depuis la VM Zabbix (10.10.0.30), ou même depuis `srv-web` lui-même (en visant `localhost`).

L'important : viser `srv-web` et forcer l'auth par mot de passe pour produire "Failed password".

Méthode simple (interactive)

```
bash Copier le code  
  
ssh -o PreferredAuthentications=password -o PubkeyAuthentication=no mauvais@10.10.0.20  
# tape un faux mot de passe 6-7 fois
```

Méthode automatique (boucle, sans interaction)

```
bash Copier le code  
  
# si besoin  
apt-get install -y sshpass || true  
  
for i in {1..7}; do  
  sshpass -p mauvais ssh -o StrictHostKeyChecking=no \  
  -o PreferredAuthentications=password -o PubkeyAuthentication=no \  
  -o NumberOfPasswordPrompts=1 mauvais@10.10.0.20 true || true  
done
```

Tu dis que "tu peux te connecter à distance toujours" : c'est très bien.

Pour m'assurer des échecs, on peut en rajouter plusieurs `sshpass` avec un mauvais mot de passe
commande pour

5) Vérifier que tout remonte

Sur `srv-web` :

```
bash Copier le code  
  
# Debian/Ubuntu  
tail -n 20 /var/log/auth.log  
  
# RHEL/Rocky  
tail -n 20 /var/log/secure
```

→ tu dois voir des `failed password for ... from <IP>`.

Dans Zabbix :

- *Monitoring* → *Latest data* (hôte `srv-web`) → l'item `log[...]` affiche des événements.
- *Monitoring* → *Problems* → le trigger `Trop de tentatives SSH échouées` passe en **PROBLEM**.
- Sans nouveaux échecs, il reviendra **OK** tout seul après 5 minutes.

6) (Optionnel) Réduire le bruit

- Tu peux remplacer `>5` par `>10` ou la fenêtre `5m` par `2h`.
- Pour ignorer une IP (ex. ta machine d'admin), on peut créer un dépendant item avec un filtre de prétraitement (regex négative). Dis-moi si tu le veux, je te donne le prêt-à-coller.

Tu peux lancer les commandes telles quelles. Si l'item reste "Not supported", envoie juste la sortie de :

```
bash Copier le code  
  
ls -l /var/log/auth.log /var/log/secure 2>/dev/null  
ll zabbix
```

commande pour

Trigger

Trigger Tags Dependencies

Name:

Event name:

Operational data:

Severity: Not classified Information Warning Average High Disaster

Expression:

Expression generator

OK event generation: Expressions Recovery expression None

PROBLEM event generation mode: Single Multiple

OK event closes: All problems All problems if tag values match

Allow manual close:

Menu entry name:

Trigger added

Alerts: 0 OK: 0 Disabled: 0 Tags: 0 Triggers: 10 Groups: 20 Discovery: 0 Alerts: 0

Host groups:

Tags:

Name:

Severity: Not classified Information Warning Average High Disaster

State: Enabled Disabled

Value: OK Problem

Tags:

AM: No Yes No

Discovered: Yes No

With dependencies: Yes No

Severity	Hosts	Name	Operational data	Expression	State	OK	Tags
Warning	OK	Top de tentatives BSR échoués (>3 en 5m)		Least 1 / exec-walk / walk . failed after 2	Enabled		

Showing 1 of 1 found

10000	10000	10000	10000	10000	10000	10000	10000
10000	10000	10000	10000	10000	10000	10000	10000
10000	10000	10000	10000	10000	10000	10000	10000
10000	10000	10000	10000	10000	10000	10000	10000

Check chat erreur zabbix "maintenant c'est ca ce ne marche toujours pas"



ça marche avec ça :))))))

Procédures :

Installer le serveur Zabbix (LAMP + dépôt officiel Zabbix).

Configurer la base de données pour Zabbix.

Accéder à l'interface web en HTTPS.

Installer et configurer l'agent Zabbix sur les machines (serveur web/BDD, poste client).

Associez les bons templates (Linux, Apache, MySQL, SSH).

Créer les métriques et seuils d'alerte (CPU, RAM, disque, Apache, MySQL, SSH).

Simuler les pannes :

Arrêt d'Apache

Arrêt de MySQL

Surcharge CPU/RAM (ex : stress sous Linux)

Tentatives SSH échouées

Vérifier le déclenchement des alertes et les graphiques générés.

Étape 2 – Enrichissement avec Grafana et Observium

Mise en place de Grafana

Objectif avec Grafana :

Grafana ne collecte rien tout seul, il sert à visualiser les données déjà stockées dans Zabbix.

Installation de Grafana

On ajoute la clé et le dépôt :

```
wget -q -O - https://packages.grafana.com/gpg.key | sudo gpg --dearmor -o /usr/share/keyrings/grafana.gpg  
echo "deb [signed-by=/usr/share/keyrings/grafana.gpg] https://packages.grafana.com/oss/deb stable main" | sudo tee /etc/apt/sources.list.d/grafana.list
```

La première commande installe la clé de sécurité de Grafana.

La deuxième commande ajoute l'adresse du dépôt Grafana.

Installation :

```
apt update  
apt install -y grafana
```

Lancement :

```
systemctl enable grafana-server  
systemctl start grafana-server  
systemctl status grafana-server
```

On se rend ensuite à l'adresse **http://10.10.0.40:3000** (Grafana écoute sur le port 3000)

Connexion par défaut :

Utilisateur : admin

Mot de passe : admin

Installer et configurer le plugin Zabbix pour Grafana

```
grafana-cli plugins install alexanderzobnin-zabbix-app  
systemctl restart grafana-server
```

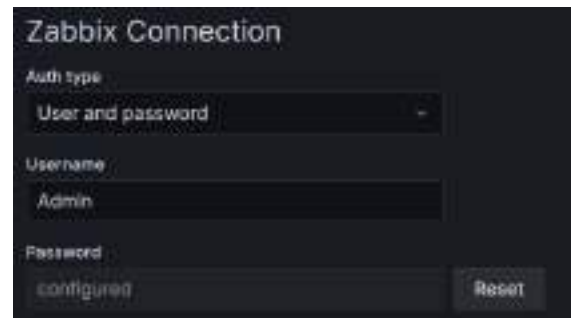
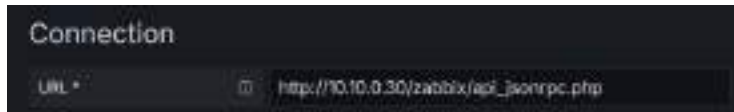
Sur l'interface web, on active le plugin

Administration > Plugins and data > Plugins > Zabbix > Enable

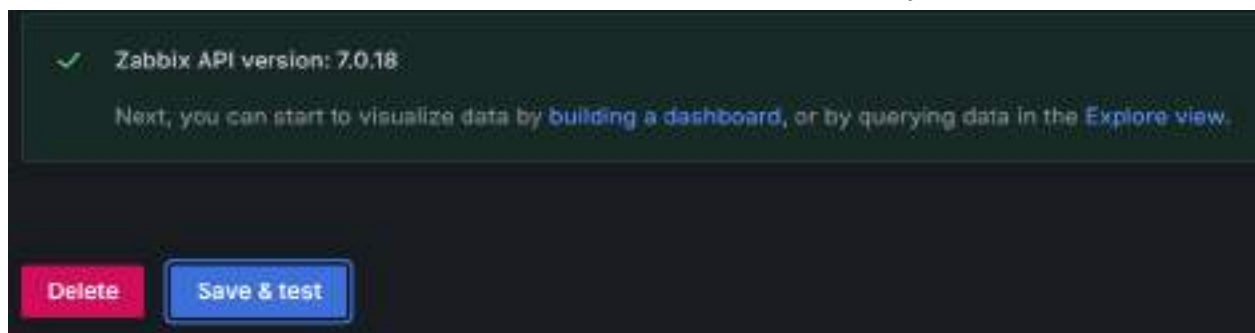
On ajoute Zabbix comme source de données

Connections > Data sources > Add data sources > Zabbix

On configure avec l'URL et les identifiants de Zabbix



Ensuite en fait **Save & test**, si la configuration est bonne on reçoit une validation



Quand on passe en https :



Pour Accepter https :

📌 Étapes dans Grafana

1. Connecte-toi à Grafana → `https://<IP_Grafana>:3000`
2. Dans le menu de gauche, clique sur ⚙️ (engrenage) → Data sources
3. Clique sur ta datasource Zabbix (celle que tu as déjà créée)
4. Fais défiler vers le bas jusqu'à trouver la section appelée :
 - HTTP settings (ou parfois "Connection")
 - et dessous, tu verras une sous-section TLS/SSL Settings (ou similaire, selon la version du plugin Zabbix).

🔍 Dans cette section, coche :

- Skip TLS Verify (ou "Disable TLS certificate verification" selon la version)

C'est exactement cette option qui dit à Grafana d'ignorer le fait que le certificat de Zabbix est auto-signé.

👉 Puis clique sur :

Save & Test

→ Si tout est bon, tu verras le message :

- "Zabbix Data source is working"

PARTIE AJOUTER LES GRAPHS DASHBOARD PERSONNALISE

• CPU fan speed

💡 Exemple de combinaison sur ton Dashboard

Section	Métriques	Type
CPU	CPU utilization	Courbe
RAM	Memory utilization	Gauge
Disque	Used disk space on /	Bar gauge
Réseau	Incoming / Outgoing traffic on eth0	Courbe superposée
Ping	ICMP ping	Courbe
Services	Service state [apache], [mysql], [sshd]	Stat (vert/rouge)
Uptime	System uptime	Stat simple

Si tu veux, je peux te dire exactement comment les placer visuellement sur ton dashboard Grafana (exemple : "à gauche CPU + RAM, au centre réseau, en bas services...").

Mise en place d'Observium

Observium est un outil de supervision via SNMP.

Contrairement à Zabbix, il ne repose pas sur des agents installés, mais sur le protocole SNMP (Simple Network Management Protocol).

Il a besoin de sa propre base de données interne (MariaDB) pour stocker les données collectées.

Ici, Observium servira à comparer ses résultats avec Zabbix et à montrer que plusieurs solutions existent.

Installation d'Observium

On suit la documentation : https://docs.observium.org/install_debian/

On se rend dans la partie **Automated Installation** pour récupérer le script d'installation :

```
wget http://www.observium.org/observium_installscrip.sh  
chmod +x observium_installscrip.sh  
./observium_installscrip.sh
```


Le script se charge automatiquement de :

- Installer Apache, MariaDB, PHP et toutes les dépendances nécessaires.
- Créer un utilisateur système observium et les bons répertoires.
- Télécharger et décompresser la dernière version Observium Community.
- Créer et initialiser la base de données Observium.
- Créer le fichier de configuration config.php.
- Configurer Apache pour servir l'interface web d'Observium.
- Activer SNMPD pour que la VM Observium puisse aussi s'auto-superviser.

Résultat : on arrive directement à une installation fonctionnelle avec l'interface web accessible depuis <http://10.10.0.50>.

Pendant l'installation :

On choisit la version **Community Edition**



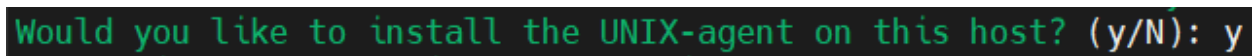
```
observium_installscrip.sh 100%[#####] 10.10.0.50  
2025-10-01 15:22:38 [13,1 MB/s] - > observium_installscrip.sh > sauvegardé [33892/33882]  
  
Observium  
Welcome to Observium installation script v0.3.3  
Please select the version of Observium you would like to install  
1. Observium Community Edition  
2. Observium Pro/Eat Edition stable (requires account at https://www.observium.org/subs/)   
3. Observium Pro/Eat Edition rolling (requires account at https://www.observium.org/subs/)   
4. Install the UNIX-Agent  
5. Install the SNMPD (snmpd-config will be overwritten)  
6. Remote poller for Observium Pro/Eat Edition stable (requires account at https://www.observium.org/subs/)   
(1-6):
```

On choisit nos identifiants Observium :



```
[*] Create first time Observium admin user. Leave empty for generate random password and username: observium...  
Username: Admin  
Password: Observium!
```

Y - Utilise si on veut une supervision complémentaire à SNMP avec des données spécifiques au système (processus, services, utilisation détaillée, etc.).



```
Would you like to install the UNIX-agent on this host? (y/N): y
```

On peut ensuite se rendre sur l'interface web à l'adresse 10.10.0.50 et se connecter.



Configurer SNMP sur les serveurs supervisés

(chat trame projet..)

```
bash
cd /opt/ob-  ↗ Demander à ChatGPT
./adduser.php admin MonMotDePasseSolide 10

→ tu te connectes avec admin / MonMotDePasseSolide

2. Configurer SNMP sur tes autres serveurs (Web, Zabbix, Windows) :
  • Sur Linux : apt install -y snmpd + éditer /etc/snmp/snmpd.conf pour autoriser la communauté public.
  • Sur Windows : activer le service SNMP et définir la communauté public.

3. Ajouter tes serveurs dans Observium :

bash
cd /opt/observium
./add_device.php 10.10.0.28 public v2c
./add_device.php 10.10.0.30 public v2c
```

```
# want the agent to listen on. Multiple definitions
# are concatenated together (using ':'s).
# arguments: [transport:]port[@interface/address],...

agentAddress udp:161,udp6:[::1]:161

#####
# SECTION: Access Control Setup
```

Étape 1 — Modifier la configuration SNMP sur la VM Zabbix (10.10.0.30)

Ouvre le fichier principal :

```
bash
nano /etc/snmp/snmpd.conf
```

Dedans, cherche la ligne suivante (elle est souvent au tout début) :

```
agentAddress udp:127.0.0.1:161
```

et remplace-la par :

```
agentAddress udp:161,udp6:[::1]:161
```

Sur Windows : activer le service SNMP et définir la communauté public. Comment je fais ça.

Telle bonne question Enzo 🤖
Tu veux donc activer le service SNMP sur Windows (pour qu'Observium puisse avoir superviser cette machine) et définir la communauté "public", comme sur les serveurs Linux.
Voici la procédure complète, pas à pas 🤖

🔴 Étape 1 — Vérifier que tu es en administrateur
Tu dois être connecté avec un compte Administrateur ou avoir les droits admin.

🟢 Étape 2 — Installer le service SNMP
+ Sur Windows 10 / 11 / Server 2019-2022
1. Appuie sur Win + R, tape :

```
-bash: cd. : commande introuvable
user@debian-observium:/opt/observium$ cd
user@debian-observium:~$ snmpwalk -v2c -c public 10.10.0.20 system
system: Unknown Object Identifier (Sub-id not found: (top) -> system)
user@debian-observium:~$
```

Procédures :

Grafana

Installation depuis dépôt officiel.

Connexion à la base Zabbix.

Création de dashboards personnalisés (indicateurs clés).

Observium

Installation (LAMP + Observium).

Activer/configurer SNMP sur les machines (Linux + Windows).

Ajouter les équipements dans Observium.

Vérifier la collecte des données.

Comparaison

Avantages/inconvénients de Zabbix vs Observium.

Apport de Grafana pour les dashboards.

Sécurité

HTTPS :

Sur chaque machine :

📁 Étape 0 – Modules et répertoire commun

👉 À faire sur chaque VM (avant de générer les certificats) :

```
bash

sudo apt update
sudo apt install -y openssl apache2 # (Apache déjà présent sur 30/30/50)
sudo a2enmod ssl headers rewrite
sudo mkdir -p /etc/ssl/localcerts
sudo chmod 750 /etc/ssl/localcerts
```

 Copier le code

🔒 Étape 1 – Générer les certificats auto-signés


Sur chaque machine, adapte simplement le CN et l'IP 📌

🖥️ VM Web (10.10.0.20)

```
bash

sudo openssl req -x509 -newkey rsa:2048 -sha256 -days 365 -nodes \
  -keyout /etc/ssl/localcerts/web.key \
  -out /etc/ssl/localcerts/web.crt \
  -subj "/C=FR/ST=Bas-Rhin/L=Strasbourg/O=IRIS/DU=BTS-SIO/CN=10.10.0.20" \
  -addext "subjectAltName=IP:10.10.0.20"

sudo chmod 600 /etc/ssl/localcerts/web.key
```

 Copier le code

🖥️ VM Zabbix (10.10.0.30)

```
bash


sudo openssl req -x509 -newkey rsa:2048 -sha256 -days 365 -nodes \
  -keyout /etc/ssl/localcerts/zabbix.key \
  -out /etc/ssl/localcerts/zabbix.crt \
  -subj "/C=FR/ST=Bas-Rhin/L=Strasbourg/O=IRIS/DU=BTS-SIO/CN=10.10.0.30" \
  -addext "subjectAltName=IP:10.10.0.30"

sudo chmod 600 /etc/ssl/localcerts/zabbix.key
```

 Copier le code

VM Grafana (10.10.0.40)

bash

 Copier le code

```
sudo openssl req -x509 -newkey rsa:2048 -sha256 -days 365 -nodes \  
-keyout /etc/ssl/localcerts/grafana.key \  
-out /etc/ssl/localcerts/grafana.crt \  
-subj "/C=FR/ST=Bas-Rhin/L=Strasbourg/O=IRIS/OU=BTS-SIO/CN=10.10.0.40" \  
-addext "subjectAltName=IP:10.10.0.40"  
  
sudo chmod 600 /etc/ssl/localcerts/grafana.key
```

VM Observium (10.10.0.50)

bash

 Copier le code

```
sudo openssl req -x509 -newkey rsa:2048 -sha256 -days 365 -nodes \  
-keyout /etc/ssl/localcerts/observium.key \  
-out /etc/ssl/localcerts/observium.crt \  
-subj "/C=FR/ST=Bas-Rhin/L=Strasbourg/O=IRIS/OU=BTS-SIO/CN=10.10.0.50" \  
-addext "subjectAltName=IP:10.10.0.50"  
  
sudo chmod 600 /etc/ssl/localcerts/observium.key
```

Étape 2 – Configurer Apache (VM Web / Zabbix / Observium)

A) Créer un site HTTPS

Exemple pour Zabbix :

```
bash

sudo tee /etc/apache2/sites-available/zabbix-ssl.conf >/dev/null <<'EOF'
<VirtualHost *:443>
  ServerName 10.10.0.30

  SSLEngine on
  SSLCertificateFile /etc/ssl/localcerts/zabbix.crt
  SSLCertificateKeyFile /etc/ssl/localcerts/zabbix.key

  Header always set X-Content-Type-Options "nosniff"
  Header always set X-Frame-Options "SAMEORIGIN"
</VirtualHost>
EOF

sudo a2ensite zabbix-ssl.conf
sudo systemctl reload apache2
```

Copier le code

```
sudo systemctl reload apache2
```

Copier le code

Même modèle pour `web-ssl.conf` (DocumentRoot `/var/www/html`) et `observium-ssl.conf` (DocumentRoot `/opt/observium/html`), en adaptant `ServerName` et les chemins `.crt/.key`.

B) Redirection HTTP → HTTPS

Sur chacune des trois VMs :

```
bash

sudo tee /etc/apache2/sites-available/000-default.conf >/dev/null <<'EOF'
<VirtualHost *:80>
  RewriteEngine on
  RewriteCond %{HTTPS} !on
  RewriteRule ^/(.*)$ https://%{HTTP_HOST}/$1 [R=301,L]
</VirtualHost>
EOF

sudo systemctl reload apache2
```

Copier le code

🔧 Étape 3 – Configurer Grafana en HTTPS natif

Éditer `/etc/grafana/grafana.ini` :

```
bash

sudo nano /etc/grafana/grafana.ini
```

 Copier le code

Dans la section `[server]`, mets :

```
ini

protocol = https
http_addr = 0.0.0.0
http_port = 3000
domain = 10.10.0.40
cert_file = /etc/ssl/localcerts/grafana.crt
cert_key = /etc/ssl/localcerts/grafana.key
```

 Copier le code

Puis :



Puis :

```
bash

sudo systemctl restart grafana-server
sudo systemctl enable grafana-server
```

 Copier le code

Accès : <https://10.10.0.40:3000> ✕

🔒 Étape 4 – Pare-feu (si UFW actif)

Sur chaque VM :

```
bash

sudo ufw allow 443/tcp
sudo ufw allow 3000/tcp # seulement sur Grafana
sudo ufw reload
```

 Copier le code

🔍 Étape 5 – Tests rapides

Depuis la VM elle-même :

```
bash

curl -kI https://10.10.0.20/
curl -kI https://10.10.0.30/zabbix
curl -kI https://10.10.0.40:3000
curl -kI https://10.10.0.50/
```

 Copier le code

`-k` ignore le warning d'auto-signé (normal).

Depuis ton poste Windows :

```
powershell

Test-NetConnection 10.10.0.20 -Port 443
Test-NetConnection 10.10.0.30 -Port 443
Test-NetConnection 10.10.0.40 -Port 3000
Test-NetConnection 10.10.0.50 -Port 443
```

 Copier le code



Puis ouvre :

```
cmd

https://10.10.0.20/
https://10.10.0.30/zabbix
https://10.10.0.40:3000
https://10.10.0.50/
```

 Copier le code

Résumé visuel

VM	IP	HTTPS via	Certificats utilisés	Port
Web + BDO	10.10.0.20	Apache2	web.crt / web.key	443
Zabbix	10.10.0.30	Apache2	zabbix.crt / zabbix.key	443
Grafana	10.10.0.40	grafana.ini	grafana.crt / grafana.key	3000
Observium	10.10.0.50	Apache2	observium.crt / observium.key	443



Copier le code

```
-subj ^/C=FR/ST=Bas-Rhin/L=Strasbourg/O=IRES/OU=BTS-S10/CN=10.10.0.20
```

Elle fait partie de la commande `openssl req`, et elle remplace les questions qu'OpenSSL te pose normalement quand tu crées un certificat ("Country Name", "Organization", etc).
Grâce à `-subj`, tu fournis directement toutes les infos dans une seule ligne.

Détail de chaque champ :

Abréviation	Signification	Ce que tu mets ici
C	Country	Code pays à 2 lettres → FR pour France
ST	State / Province	Région ou département → Bas-Rhin
L	Locality	Ville → Strasbourg
O	Organization	Ton école, ton entreprise ou ton projet → IRES
OU	Organizational Unit	Unité ou service → BTS-S10
CN	Common Name	Le nom du serveur ou son IP → cf 10.10.0.20

```
scp root@10.10.0.20:/etc/ssl/localcerts/web.crt C:\Users\<<TON_USER>\Desktop\
```

Mise en place des accès HTTPS avec certificats auto-signés.

Documentation des mots de passe robustes.

Archivage des fichiers de configuration.

Résultats et Tests

Présentation des scénarios de panne et des résultats observés.

Graphiques et captures d'écran.

Validation que les alertes fonctionnent bien.