**Jour-1 Tercium : Infomaniak.cloud // Paramétrage // Instanciation**

**TERCIUM : Une entreprise constituée de micro-entreprises dans divers domaines du digital. (à détailler). Il y a au total 20 personnes réalisant des projets dans divers secteurs.**

* **C**elle-ci possède ses locaux et ses équipements en propre comme ses **PC** et ses **NAS.** Chacun étant relier aux équipes dédiées.
* Celle-ci ne stocke pas ses données dans ses propres serveurs mais les délocalisent dans une ferme de serveur Suisse qui ont une politique éthique (RSE) stricte : **Infomaniak**

Objet du stage : Installation de Jitsi un software de visioconférence. Ces besoins en capacité seront calculés par rapport aux besoins spécifiques.

* Website Jitsi : <https://jitsi.github.io/handbook/docs/architecture>

**Pourquoi faire des visioconférences ?**

**Il n’est pas toujours possible de se rendre à une réunion pour diverses raisons comme la distance ou un empêchement quelconque.**

**Pour faire des conversations en**[**webcam**](https://cours-informatique-gratuit.fr/dictionnaire/webcam/)**, rencontrer plusieurs personnes, de partager des**[**documents**](https://cours-informatique-gratuit.fr/dictionnaire/fichier/)**, des**[**liens URL**](https://cours-informatique-gratuit.fr/dictionnaire/url/)**ou votre**[**écran**](https://cours-informatique-gratuit.fr/dictionnaire/ecran/)**pour donner plus d’explications à vos interlocuteurs, il vous faudra utiliser un outil de visioconférence.**

**Zoom : Standard actuel et proposition commerciale**

* [**Zoom Spaces**](https://explore.zoom.us/fr/products/zoom-rooms/)**est un moyen innovant d’organiser des réunions virtuelles au sein d’équipes hybrides. Les services inclus sont la réservation d’un espace de travail (par exemple, un bureau ouvert, un dispositif Zoom), un connecteur de salle de conférence pour rejoindre les réunions avec un équipement SIP/H.323, et des salles Zoom pour les conférences qui impliquent de nombreuses personnes des deux côtés.**
* [**Zoom Events**](https://explore.zoom.us/fr/products/event-solutions/)**est une solution dédiée à l’organisation d’événements virtuels, des annonces internes aux tables rondes. Cette option est idéale pour le réseautage, la présentation de contenu dans des stands d’exposition, la formation d’équipes dans des événements à session unique et des webinaires en ligne.**
* [**Zoom Contact Center**](https://explore.zoom.us/fr/products/contactcenter/)**vous permet de créer un environnement omnicanal pour fournir une assistance à vos clients. Les outils disponibles comprennent un centre de contact optimisé pour la vidéo dans le cloud et l’IA conversationnelle.**

**La concurrence OpenSource :**

**Jitsi Meet est une solution de visioconférence multiplateforme (Windows, macOS, Linux, Android, iOS) qui brille par sa modularité et son efficacité. Conçu pour les secteurs professionnels, éducatifs et médicaux, Jitsi offre une alternative open source robuste aux services payants comme Zoom ou Microsoft Teams.**

**Hébergé sur un cloud multirégional, Jitsi garantit une expérience audio et vidéo fluide, même pour des utilisateurs géographiquement dispersés. Soutenue par la communauté tech et des experts en sécurité, l’application continue de se distinguer en étant entièrement gratuite et hautement personnalisable.**

**Une sécurité, qui n’est pas un détail, est l’ajout d’un mot de passe à l’appel. Cela empêche à la fois une intrusion inopportune à une réunion ainsi que l’impossibilité à celle-ci de bloquer les appels.**

* **Usage de l’icône i afin d faire apparaître les informations de la salle virtuelle de la réunion.**
* **Envoi via un courriel ou d’un SMS de l’URL avec le mot de passe.**
* **Comme zoom il suffit de placer le lien dans un navigateur ; il est recommandé d’utiliser Chrome ou Firefox.**

**La capacité a évolué est permet maintenant de gérer une réunion de 500 participants.**

**Au-delà Jitsi a un service payant : JaaS permettant une réunion de 10 000 participants.**

**Particularités :**

* **Comme ses concurrents payants anciennement Skype puis Zoom,Teams, Whereby, eyeson,,hangouts Meet ; Jitsi peut partager l’écran, enregistrer et passer en mode vignette.**
* **L’application mosaïque en est le parfait exemple.**
* **Le contrôle de sons environnement, comprendre le respect de la vie privée par l’escamotage de son espace privatif par l’usage d’un arrière-plan flou ou fictif est une fonction (usage de trois petits points).**
* **L’enregistrement de la session est possible pour une diffusion indépendante du flux avec YouTube.**
* **Celui-ci peut être aussi réaliser, l’enregistrement via Dropbox avec un compte basique gratuit, pour ce faire il faut comme précédemment utiliser les trois petits points.**

Besoins matériels :

* <https://jitsi.github.io/handbook/docs/devops-guide/devops-guide-requirements/>

Usage & Compatibilité :

* <https://linuxfr.org/users/lebouquetin/journaux/organiser-des-visioconferences-de-haute-qualite-avec-le-logiciel-libre-jitsi-meet>

Réseau :

* OpenRC peut être utiliser pour gérer le réseau :

<https://www.linuxtricks.fr/wiki/openrc-gestion-du-reseau>

**Aborder Infomaniak : Cloud Provider** Suisse

<https://www.infomaniak.com/fr/support/faq/2601/guide-de-demarrage-public-cloud>

Infomaniak [Public Cloud](https://infomaniak.com/gtl/hosting.public-cloud), une solution [Infrastructure\_as\_a\_service (IaaS)](https://fr.wikipedia.org/wiki/Infrastructure_as_a_service) basée sur [OpenStack](https://docs.openstack.org/yoga/) qui met à disposition les ressources dont vous avez besoin pour le développement de vos projets.

**Démarche pour une instanciation :**

**Inscription & accès :** [**https://api.pub2.infomaniak.cloud/horizon/auth/login/**](https://api.pub2.infomaniak.cloud/horizon/auth/login/)

**Pages explicatives :** <https://docs.infomaniak.cloud/compute/>

Create a keypair :

Une image contenant texte, Police, algèbre, reçu

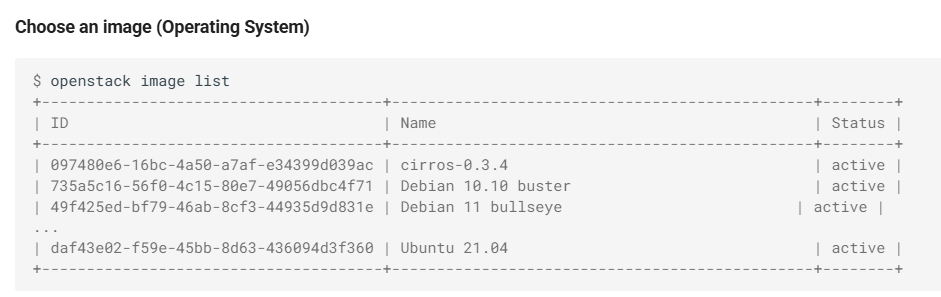
Le contenu généré par l’IA peut être incorrect.

Create an instance :

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.

**Choose an image (operating System):**



**Exemple de création :**

**Une image contenant texte, capture d’écran, logiciel, nombre

Le contenu généré par l’IA peut être incorrect.**

**Contrôle de création d’Instance:**

****

**Firewall :**

**Une image contenant texte, capture d’écran, Police, algèbre

Le contenu généré par l’IA peut être incorrect.**

**Clef SSH : modèle**

****

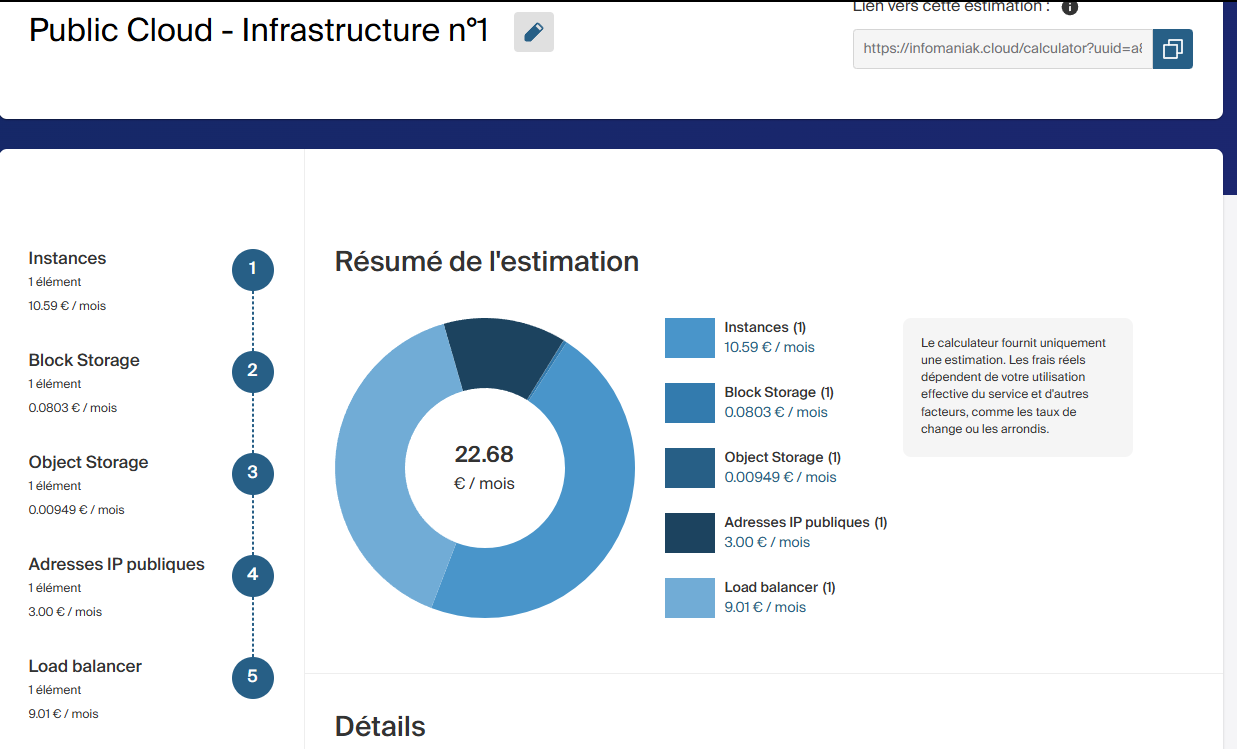
**Availability Zones :** openstack availability zone list

Une image contenant texte, Police, capture d’écran, reçu

Le contenu généré par l’IA peut être incorrect.

**Coût d’une instanciation pour la compagnie : Infomaniak permet la simulation.**

**Ci-dessous les choix matériels.**

****

**Détails :**

**Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.**

**Rappel :**

**Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

**Choix des availability Zone : az-1 / az-2 / az-3**

**Ces trois zones sont des zones matérielles dans la ferme à serveur.**

**La région sera dc4-a : La région dc4-a utilisée par Infomaniak Public Cloud fait référence à un datacenter situé en Suisse, plus précisément dans la zone Genève / Satigny (canton de Genève).**

**Putty Key Generator : génére deux clefs RSA, une publique et une privée.**

**Je peus faire des imports dans le launch Instance fenêtre de Infomaniak.**

**Putty : me permettre en connexion, avec l’interface Putty.**

**Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

Dans d’autres versions de **Putty** il est possible d’exporter la clef publique directement via la barre de menu, Conversions en format **OpenSS**.

**Ici je copie colle** la clef publique directement de **Putty** pour la transmettre à l’interface de création d’instance de d’infomaniak.

Dans l’impossibilité le pis allé a été de générer un fichier txt / bloc-note. Puis de l’appeler dans l’interface Infomaniak. Ce qui se déroule en deux temps.

Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.

**Jour-2 Tercium : Instanciation Réussie :**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Il est plus aisé cependant de faire générer ses clefs via le terminal / console de VSCode : Premier problème l’authentification :**

**Il est plus aisé cependant de faire générer ses clefs via le terminal / console de VSCode : Premier problème l’authentification :**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Usage de nano pour modifier les permissions :**

**cd :** **sudo nano /etc/ssh/sshd\_config Action : PasswordAuthentication no**

**Renforcer la sécurité (recommandé) :**

**bash**

**CopyEdit**

**sudo nano /etc/ssh/sshd\_config**

* **Modifie ou décommente :**

**conf**

**CopyEdit**

**PasswordAuthentication no**

**PermitRootLogin prohibit-password**

* **Puis redémarre le service :**

**bash**

**CopyEdit**

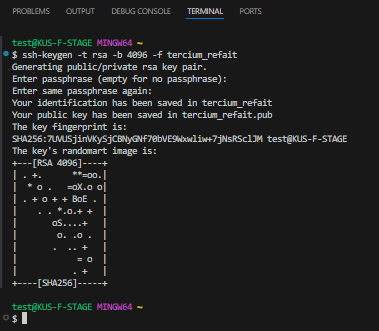
**sudo systemctl restart ssh**

**Via VScode je vais pouvoir faire des commandes administrateur.**

* **Générer et valider des droits d’usages avec chmod.**
* **Créer des dossiers : mkdir, des fichiers touch pour y insérer les clefs des instances.**

**Via le terminal Je recrée des clefs : ssh-keygen -t rsa -b 4096 -f tercium\_refait\_key** qui sera le nom du fichier la contenant. Extension.pub pour la clef publique et la privée sans extension. Powershell n’étant pas le seul terminal que j’utilise, je travaille aussi avec GitBash pour avoir une ligne de commande puissante.

**Une image contenant texte, capture d’écran, Police, logiciel

Le contenu généré par l’IA peut être incorrect. **

**Verrouillage d’une IP :**

**Host terciump**

**HostName 37.156.45.22**

**User ubuntu**

**IdentityFile ~/Documents/Tercium\ Stage/tercium-instance\_key/tercium-instance\_key**

**Connexion standard : ssh -i ~/.ssh/tercium\_key** [**ubuntu@84.234.28.241**](mailto:ubuntu@84.234.28.241)

**Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

**cd : direction instance**

**ssh -i "~/Documents/Tercium Stage/test3\_instance\_key/test3publickey\_refait\_key"** [**ubuntu@84.234.28.241**](mailto:ubuntu@84.234.28.241)

**Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Le contenu généré par l’IA peut être incorrect.**

**Sur VsCode on sort de l’instance du terminal :ctrl + D**

**Infomaniak.cloud : interface de contrôle des instances**

**Une image contenant texte, capture d’écran, ligne, Police

Le contenu généré par l’IA peut être incorrect.**

**Maintenant étapes de contrôle de connexion via les Adresses IP pour un poste extérieur. Installation d’un serveur Apache pour Linux : LAMP**

**cd : sudo apt update && sudo apt install apache2 php libapche2-mod-php -y**

**On Active et démarre Apache :**

**cd : sudo systemctl enable apache2**

**sudo systemctl start apache2**

**Côté serveur : Apache est-il bien démarré ? sudo systemctl status apache2**

**Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

* **Écoute-t-il sur les bons ports ? sudo netstat -tlnp | grep :80**

**La version moderne de netstat => sudo ss -tlnp | grep :80**

* **Côté réseau/firewall :**

**Le firewall du serveur autorise-t-il le trafic sur les ports 80/443 sur infomaniak ?**

**Infomaniak a des règles de sécurité/firewall à configurer dans Security Group :**

**Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.**

**Test simple : Créez un fichier index.html basique dans /var/www/html/ et tentez d'y accéder via** [**http://VOTRE\_IP/**](http://VOTRE_IP/)

**Via le terminal écrire :**

**cd : echo "<h1>OK LAMP / INFOMANIAK</h1>" | sudo tee /var/www/html/index.html**

**Une image contenant texte, capture d’écran, Police, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Explication du code, qui fait deux choses :**

1. **echo produit la chaîne : "<h1>OK LAMP / INFOMANIAK</h1>"**
2. **tee :**
   * **écrit cette chaîne dans le fichier /var/www/html/index.html**
   * **et réimprime sur l’écran (stdout) ce qu’elle écrit.**

**Une image contenant texte, capture d’écran, Logiciel multimédia, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Jour-3 Tercium : Clef SSH et Hash**

**A / Fingerprint : Explication structurée issue de ChatGPT**

**Dans le contexte d’OpenStack (ici via l’interface *Infomaniak Public Cloud*), le fingerprint SSH associé à chaque paire de clés a les usages suivants :**

**Usage du Fingerprint SSH**

**1. Vérification d’authenticité**

* Le *fingerprint* est un empreinte condensée (hash) de la clé publique associée à une paire SSH.
* Il permet de vérifier rapidement que la clé utilisée pour se connecter à une machine correspond bien à celle attendue, sans afficher toute la clé (souvent longue).

**2. Identification rapide des clés**

* Permet de différencier deux clés ayant le même nom ou des noms similaires (ex : Public key, Public-key).
* Ex : **tercium-instance\_key** et **terciup-instance\_key** ont la même empreinte → c’est exactement la même clé.

**3. Sécurité (anti-usurpation)**

* Lorsqu’un utilisateur ou script tente une connexion SSH, le système peut comparer le fingerprint local (de .ssh/known\_hosts) à celui enregistré dans OpenStack pour valider l’identité**.**

**4. Audit / Journalisation**

* En environnement multi-utilisateur ou en audit de sécurité, l’empreinte permet de tracer quelle clé a été utilisée, sans afficher ou stocker la clé complète.

**Format de l’empreinte :**

* **Généralement SHA-1 (comme ici), parfois SHA-256 selon la configuration.**
* **Exemple : ad:31:ae:a1:76:9d:29:de:65:71:fb:07:c2:d9:6f:98**

**Cas spécifique :**

* tercium-instance\_key et terciup-instance\_key ont exactement le même fingerprint : Cela indique que la même clé publique a été réutilisée pour plusieurs instances.

**✅ Bonnes pratiques**

| **Recommandation** | **Description** |
| --- | --- |
| **Générer une clé unique par instance** | **Pour éviter des accès non maîtrisés entre VM** |
| **Stocker et vérifier les fingerprints dans un**  **fichier de suivi** | **Par exemple inventory.md ou dans un dépôt Git** |
| **Comparer fingerprint local / cloud** | **Pour éviter les erreurs de clé en cas de multiples jeux** |

**B / Test de connexion simple : Ping avec le terminal sans connexion direct :**

**Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.**

**C / 2nd Test simple : Créez un fichier index.html basique dans /var/www/html/ et tentez d'y accéder via** [**http://VOTRE\_IP/**](http://VOTRE_IP/)

**Pour se connecte via son terminal il faut avoir activer son agent SSH avec sa clef privée :**

* **Se positionner dans le dossier maître :**

**ils se situe dans C:\Users\test\Documents\Tercium\_Stage\Tercium-instance\_key**

* **Puis Naviguer dans le dossier contenant les deux fichiers Public\_key & Private\_Key.pub.**

**cd : ssh -i Tercium-instance\_key/tercium-instance\_key** [**ubuntu@84.234.28.98**](mailto:ubuntu@84.234.28.98)

**si l’on oublie l’adresse IP : Résultat**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Résumé des processus à respecter et comprendre :**

1. **Chemin correct** de la clé privée :
2. ssh -i Tercium-instance\_key/tercium-instance\_key ubuntu@84.234.28.98
3. **Clé privée avec permissions sécurisées** (si besoin) :
4. chmod 600 Tercium-instance\_key/tercium-instance\_key
5. **Pas besoin de ssh-add** dans ce cas : ssh -i suffit pour l’authentification directe.
6. **Pas de session root à maintenir ouverte pour SSL** : les certificats sont **posés une fois** (ex. via certbot) et renouvelés **automatiquement** en tâche de fond.

Trois éléments critiques à documenter dans le **manuel d’usage** :

**✅ Résumé à intégrer dans ton rapport (section SSH / accès distant)**

1. **Structure des fichiers :**
   * Dossier : Tercium-instance\_key/
     + Fichier : tercium-instance\_key → **clé privée**
     + Fichier : tercium-instance\_key.pub → **clé publique**
2. **Connexion réussie via Git Bash :** ssh -i Tercium-instance\_key/tercium-instance\_key [ubuntu@84.234.28.98](mailto:ubuntu@84.234.28.98) **/ à chaque instance détruite / recréer, l’IP adresse change**.
3. **Écueils à éviter :**
   * Ne pas confondre **répertoire** et **fichier de clé**
   * La commande ssh-add est inutile si on utilise directement ssh -i
   * Vérifier les permissions (chmod 600) si accès refusé
   * Le format Windows (\) ne fonctionne pas dans Git Bash, utiliser / ou ~/Documents/...

**Proposition bonus**

**Créer un alias dans Git Bash pour ne pas avoir à taper toute la ligne :**

**echo "alias ssh-tercium='ssh -i ~/Documents/Tercium\_Stage/Tercium-instance\_key/tercium-instance\_key ubuntu@84.234.28.98'" >> ~/.bashrc**

**source ~/.bashrc**

**Ensuite, il te suffira de taper : ssh-tercium**

**Vos pages HTML sont-elles dans /var/www/html/ ?**

**Ayant détruit l’instance il n’est plus possible d’obtenir quelque chose qui a été effacée.**

**Pour repartir proprement, séquence minimale sur Ubuntu (cloud ou local) pour reconstruire :**

**sudo apt update**

**sudo apt install apache2 -y**

**echo "<h1>Site opérationnel</h1>" | sudo tee /var/www/html/index.html**

**sudo systemctl restart apache2**

**D / Test local :**

* **En local :** [**http://localhost**](http://localhost)

**Tester le serveur Apache localement (via WSL2 à installer au préalable via powershell)**

**cd : wsl --list –verbose // Vérification si WSL est installé dans VSCode**

**Sinon : wsl --install -d Ubuntu**

**Aussi Installer l’extension “Remote - WSL” dans VS Code Extensions (Ctrl+Shift+X) → cherche : nginx CopyEdit Remote - WSL**

**Lancer un projet ou un terminal dans WSL via VS Code**

* **Ouvrir la palette de commandes Ctrl+Shift+P**
* **Tape : WSL: New WSL Window**
* **Un nouveau VS Code va s’ouvrir avec le terminal WSL actif**
* **Navigue dans le projet (cd /mnt/c/Users/...) ou clone un dépôt**

**Intérêts :**

* **Ecrire depuis Windows mais exécutes sur Linux réel.**
* **Faire sudo, apt, systemctl, etc.**
* **Ouvrir des projets en un clic avec code . dans WSL**

**Test en local :** [**http://localhost**](http://localhost) **ou plus précisément** [**http://localhost:80**](http://localhost:80)

**Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Une image contenant texte, Logiciel multimédia, logiciel, Logiciel de graphisme

Le contenu généré par l’IA peut être incorrect.**

**Si en cloud : http://<ip\_publique\_VM> déjà réaliser.**

**E / Certificat SSL via Cerbot !**

**Synthèse du document LE-SA-v1.5 (Let’s Encrypt Subscriber Agreement) :**

**Daté du 24 février 2025, utile pour toute implémentation réelle ; en entreprise, audit ou projet certbot / HTTPS.**

**Objectif du document :**

**Contrat juridique entre toi (ou ton organisation) et ISRG (Internet Security Research Group), organisme émetteur des certificats Let’s Encrypt via le protocole ACME.**

**1. Définitions clés à retenir**

* **ACME :** Protocole automatisé de gestion des certificats**.**
* **Certificat :** Lien validé entre un nom de domaine et une clé publique.
* **Key Pair :** Paire de clés asymétriques **(privée / publique).**
* **Private Key compromise :** Clé privée compromise ou à risque => certificat à révoquer.

**2. Conditions d’entrée en vigueur**

* **Accord effectif** dès que tu demandes un certificat **Let’s Encrypt (même via ACME).**
* Il reste valide tant que tu possèdes un certificat actif, même s’il est renouvelé automatiquement.

**3. Engagements du souscripteur :**

**Tu garantis :**

* Que tu es légitime sur le domaine visé.
* Que tu n’as pas obtenu ce domaine illégalement.
* Que tu protèges ta clé privée.
* Que les infos du certificat sont exactes, à jour et sincères.
* Que tu révoques immédiatement le certificat si :
  + la clé est compromise,
  + le domaine t’échappe,
  + ou les données sont obsolètes.

**4. Gestion technique**

* Le certificat est généré à partir des infos envoyées par ton client ACME **(e.g., certbot).**
* Tu dois vérifier les infos avant usage.
* Tu as le droit d’installer le certificat uniquement sur les serveurs mentionnés dans le champ **subjectAltName.**

**5. Usage interdit**

* **Écoute active (attaque MITM)**
* **Interception ou redirection de trafic non autorisé**
* **Toute architecture facilitant la violation de la confidentialité HTTPS**

**6. Révocation**

* **Tu dois révoquer un certificat via l’API ACME si :**
  + clé compromise,
  + changement de domaine,
  + données fausses.
* **ISRG peut révoquer sans ton accord pour :**
  + **usage frauduleux,**
  + **décision judiciaire,**
  + **certificat incorrect ou détourné.**

**7. Clause de non-responsabilité (ISRG)**

* Let’s Encrypt est un service gratuit sans garantie contractuelle.
* ISRG décline toute responsabilité en cas :
  + de perte,
  + de poursuite,
  + de dommage technique ou juridique.

**8. Droit applicable**

* **Loi de Californie.**
* **Tribunal compétent : San Jose, CA.**
* **Aucune tierce partie n’a de droit par ce contrat.**
* **Limite d’action légale : 1 an.**

**En résumé pour usage réel :**

| **Action** | **Obligation** |
| --- | --- |
| **Génération des clés** | **En local, jamais par ISRG** |
| **Protection de la clé privée** | **Obligatoire (clé = confidentielle)** |
| **Vérification du certificat** | **Avant toute utilisation** |
| **Révocation** | **Immédiate en cas de doute ou perte de contrôle** |
| **Usage** | **Limité au domaine validé, usage HTTPS standard uniquement** |
| **Comportement interdit** | **MITM, reroutage de trafic, spoofing** |

**Avec WSL en tant qu’administrateur :**

**cd : sudo apt update && sudo apt install cerbot python3-certbot-apache y**

**Jour-4 Tercium : Coût d’une Instanciation**

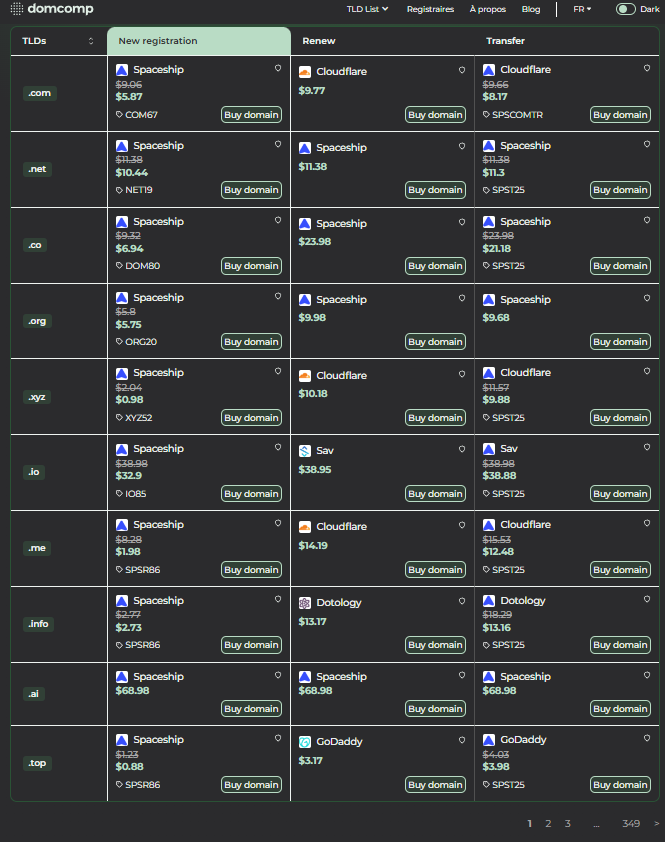
**A / Informations quant au coût évaluation exemple choisi : Jitsi.tercium.xyz**

**Coûts complets et implications techniques pour le nom de domaine jitsi.tercium.xyz, y compris la partie certification SSL/TLS. Voici la synthèse structurée, point par point :**

**1. Nom de domaine ; Jitsi.tercium.xyz**

**▸ Enregistrement :**

* Registrar : **Spaceship**
* Coût initial : **$0.98 (≈ 0.91 €)**
* Renouvellement : **$10.18 annuelle**
* Transfert : **$9.88**  
  Total pour 1 an (base) : ≈ 1 € pour l’enregistrement + SSL gratuit via Certbot (Let’s Encrypt) = ≈ 1 € initialement, renouvellement ≈ 10 €

****

**2. Certificat SSL/TLS via Let’s Encrypt**

**▸ Certbot avec Apache :**

* **Commande (déjà mentionnée dans ton journal) :**

**sudo apt update**

**sudo apt install certbot python3-certbot-apache -y**

1. Configuration automatique : certbot configure Apache et obtient un certificat gratuitement via ACME.
2. Renouvellement automatique : inclus (/etc/cron.d ou via systemd timer), sans coût**.**

**▸ Conditions :**

* Tu dois posséder le domaine (tercium.xyz)
* Le port 80 (HTTP) doit être ouvert pour validation ACME
* Le certificat est valide 90 jours, renouvelable

**3. Instance cloud utilisée**

**D’après capture :**

* **Type : a4-ram4-disk80-perf1**
  + **4vCPU**
  + **8 GB RAM**
  + **20 GB disque SSD**
* **OS : Ubuntu 24.04 Kube v1.30 (image publique, QCOW2)  
  ➡️ Suffisant pour héberger Jitsi, mais attention au coût horaire chez Infomaniak**

**4. Configuration Apache pour Jitsi**

**Si tu dédies un sous-domaine jitsi.tercium.xyz :**

* **Ajouter un VirtualHost Apache :**

<VirtualHost \*:80>

ServerName jitsi.tercium.xyz

DocumentRoot /var/www/jitsi

</VirtualHost>

* Et SSL après certbot --apache :

<VirtualHost \*:443>

ServerName jitsi.tercium.xyz

SSLEngine on

SSLCertificateFile /etc/letsencrypt/live/jitsi.tercium.xyz/fullchain.pem

SSLCertificateKeyFile /etc/letsencrypt/live/jitsi.tercium.xyz/privkey.pem

</VirtualHost>

**5. Checklist récapitulative**

| **Étape** | **Action technique** | **Coût** |
| --- | --- | --- |
| **Nom de domaine tercium.xyz** | **Achat sur Spaceship** | **~1 €** |
| **Création sous-domaine jitsi.\*** | **Ajout dans DNS (type A vers IP publique)** | **0 €** |
| **Serveur Apache / installation Jitsi** | **Apache + Jitsi install sur Ubuntu 24.04** | **0 €** |
| **Ouverture port 80 / 443** | **Configuration Security Groups Infomaniak** | **0 €** |
| **Certificat TLS** | **Certbot + Let’s Encrypt (ACME)** | **0 €** |
| **Renouvellement automatique SSL** | **Certbot cron ou systemd timer** | **0 €** |
| **Renouvellement domaine .xyz** | **Annual fee Spaceship** | **~10 €** |

**Conclusion :**

* **Domaine tercium.xyz** : 0.98 $ la première année
* **SSL Let’s Encrypt :** 0 €
* **Infomaniak** (coût instance selon durée d’usage)
* Possibilité de pointer **jitsi.tercium.xyz** via un enregistrement **DNS** vers l’**IP** publique
* **Jitsi Meet** peut être installé et certifié gratuitement si ton reverse proxy Apache est bien configuré

**Calcul Gobal :**

**Coût d’une instance : Infomaniak // https://www.infomaniak.com/fr/domaines https://hellosafe.fr/hebergeurs/infomaniak**

1. **Domaine réel  : visio.workeezconnect.fr**
2. **Choix de l’OS :**

****

1. **Achat d’un volume / basique puis extensible.**

****

1. **IP fixe ou flottante : une seule adresse menant vers le serveur Jitsi**
2. **Instance basique : moindre coût**

**Pour calculer le coût réel d'une instance cloud déployée comme Jitsi, surtout chez Infomaniak Public Cloud (OpenStack), tu dois prendre en compte plusieurs paramètres facturables à l’usage. Voici la méthode de calcul complète et générique, suivie de son application à ton cas (instance Jitsi jusqu’au 19 août).**

**1. Paramètres de tarification à surveiller (chez Infomaniak)**

Infomaniak Public Cloud facture à la minute (via OpenStack), sur les bases suivantes :

| **Composant** | **Base de coût** | **Unité** |
| --- | --- | --- |
| **vCPU** | **€ par vCPU par heure** | **€/vCPU/h** |
| **RAM** | **€ par GB RAM par heure** | **€/GB/h** |
| **Stockage disque** | **€ par GB par mois** | **€/GB/mois** |
| **Trafic sortant** | **€ par Go sortant par mois** | **€/Go/mois** |
| **Adresse IP fixe** | **parfois facturée séparément** | **€/IP/mois** |

**2. Application à ton instance actuelle : Instance utilisée** : **a4-ram8-disk20-perf1**

| **Ressource** | | **Valeur** | |
| --- | --- | --- | --- |
| **vCPU** | | **4** | |
| **RAM** | | **8 GB** | |
| **Disque** | | **20 GB SSD** | |
| **Durée d’usage** | | **du 03/07 au 19/08 = 47 jours ≈ 1.53 mois** | |

**Estimation du coût pour cette instance (Infomaniak Public Cloud) :**

| **Ressource** | **Tarif unitaire estimé** | **Calcul sur 47 jours (1.53 mois)** | **Coût** |
| --- | --- | --- | --- |
| **vCPU (4)** | **0.01 €/vCPU/h = 0.04 €/h** | **0.04 × 24 h × 47 j = 45.12 €** | **45 €** |
| **RAM (8 GB)** | **0.01 €/GB/h = 0.08 €/h** | **0.08 × 24 h × 47 j = 90.24 €** | **90 €** |
| **Disque (20 GB)** | **0.05 €/GB/mois** | **20 × 0.05 × 1.53 mois = 1.53 €** | **1.5 €** |
| **IP publique fixe** | **~1.50 €/mois** | **1.50 × 1.53 = 2.30 €** | **2.3 €** |
| **Trafic sortant** | **Estimation : 30 Go @ 0.03 €/Go** | **30 × 0.03 = 0.90 €** | **0.9 €** |

**3. Estimation du coût total pour 47 jours (1.53 mois)**

| **Ressource** | | **Calcul** | | **Total approximatif** | |
| --- | --- | --- | --- | --- | --- |
| vCPU + RAM | | (0.02 + 0.04) €/h × 24 h × 47 jours = ~68 € | | 68 € | |
| Disque 80 GB | | 4 €/mois × 1.53 mois = 6.12 € | | 6.12 € | |
| IP publique fixe | | 1.5 €/mois × 1.53 mois = 2.30 € | | 2.30 € (option) | |
| Trafic sortant | | Ex : 30 Go × 0.03 €/Go = 0.90 € | | 0.90 € | |

➡️ Total estimé : 77 à 80 € pour 47 jours.

**4. Paramètres à monitorer activement**

| Élément | Commande / outil (Linux Ubuntu) |
| --- | --- |
| Uptime | uptime ou who -b |
| Usage CPU/RAM | htop, top, free -h |
| Utilisation disque | df -h |
| Bande passante | vnstat, iftop, ou Grafana/Prometheus si installé |
| Durée instance | Interface Infomaniak (historique / instance details) |

**5. Choix à prévoir avant le 19 août**

| **Option** | **Avantage** | **Risque ou coût** |
| --- | --- | --- |
| **Arrêt instance** | **Ne facture plus CPU/RAM, garde stockage/IP** | **Disque/IP facturés** |
| **Suppression instance** | **Zéro coût** | **Perte de config complète** |
| **Snapshot + arrêt** | **Archiver Jitsi (image QCOW2), puis stop** | **Très faible coût** |
| **Continuer** | **Service en ligne** | **50–80 €/mois** |

**Recommandation**

Créer un tableur de calcul automatique avec :

* Coût horaire vCPU
* Coût horaire RAM
* Coût mensuel disque
* IP (fixe ou non)
* Trafic estimé

**B / Le 03 / 07 / 2025 Lier Adresse IP et Nom de Domaine :**

[**https://www.infomaniak.com/fr/support/faq/2025/lier-un-nom-de-domaine-a-un-hebergement-web-infomaniak**](https://www.infomaniak.com/fr/support/faq/2025/lier-un-nom-de-domaine-a-un-hebergement-web-infomaniak)

**Nom de domaine fixé :** [**https://visio.workeezconnect.fr/**](https://visio.workeezconnect.fr/)

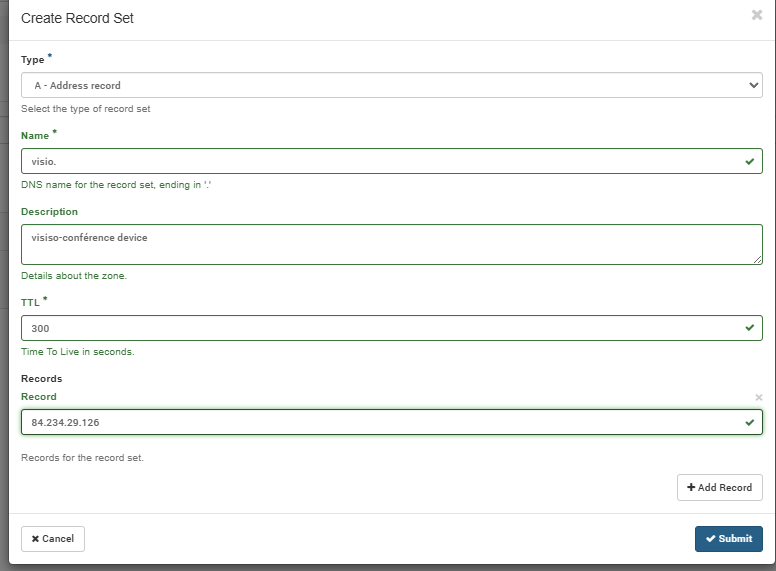
**Adresse IP fixée : 84.234.29.126**

**Explication théorique claire :**

| **Élément** | **Rôle** |
| --- | --- |
| **DNS (A record)** | **Associe visio.workeezconnect.fr → 84.234.29.126** |
| **Certificat SSL (Let's Encrypt)** | **Prouve que le serveur web contrôle réellement ce domaine** |
| **ACME challenge** | **Mécanisme utilisé pour prouver que tu possèdes le domaine** |

**Avant même l’installation du certificat il faut lier domaine et l’IP  : Interface Infomaniak**

**Interface Infomaniak : DNS Zones Create Zone**

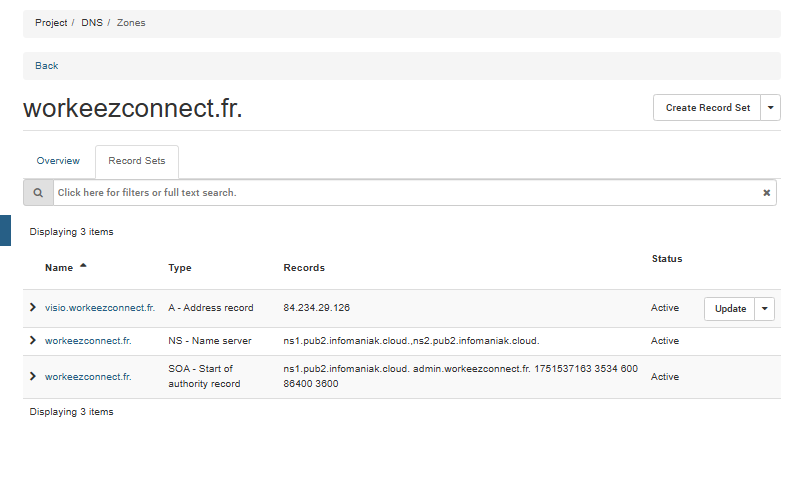
****

**Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.**

**Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.**

****

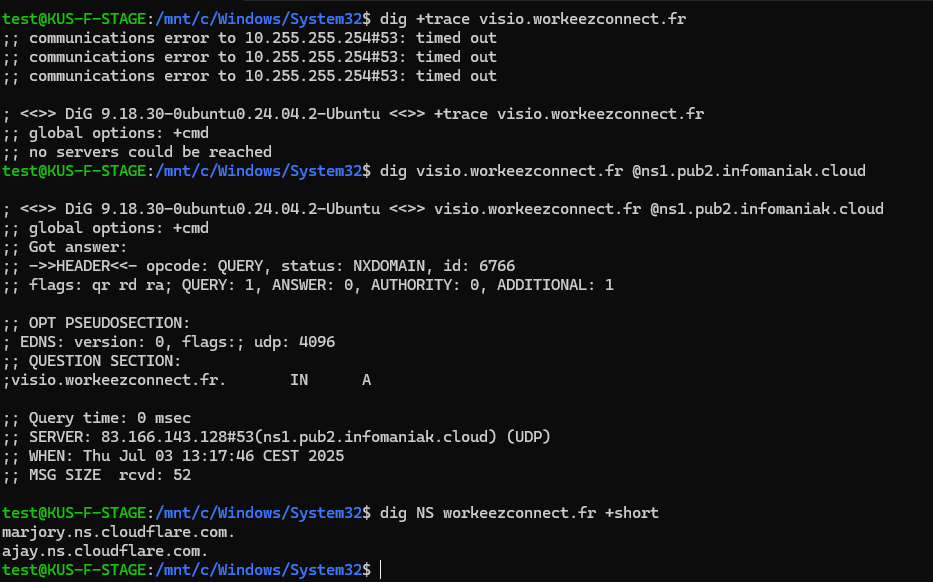
**Details des créations :**

**Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.**

**Tests avec : dig workeezconnect.fr NS +short**

* **dig :** commande DNS utilisée pour interroger les serveurs DNS.
* **workeezconnect.fr :** le nom de domaine que tu veux interroger.
* **NS :** signifie *Name Server*, c’est-à-dire que tu demandes les serveurs de noms responsables de ce domaine.
* **+short :** format de sortie simplifié (juste la réponse, pas les détails).



**Problèmes : les tests de connexions et d’identifications ont échoué, les serveurs sont des cloudflares non des infomaniak. Il faudra changer via l’accès client permettant la création de nom de domaines pour switcher les serveurs.**

**Actions requises :**

| **Action** | **Où ?** | **Objectif** |
| --- | --- | --- |
| **Modifier les serveurs de noms NS** | **Chez le registrar du domaine (OVH, Infomaniak, Gandi, etc.)** | **Remplacer \*.ns.cloudflare.com → ns1.pub2.infomaniak.cloud** |
| **Attendre propagation DNS** | **Internet (1h à 24h)** | **Reflète les bons serveurs** |
| **Tester avec dig ou nslookup** | **Ton terminal** | **Confirmer propagation** |

**Vérification après propagation : avec ces commandes et les réponses attendues**

1. **dig NS workeezconnect.fr +short**

**# Doit répondre avec :**

**ns1.pub2.infomaniak.cloud.**

**ns2.pub2.infomaniak.cloud.**

1. **dig visio.workeezconnect.fr +short**

**# Doit renvoyer :**

**84.234.29.126**

**Délai de propagation mondial réaliste.**

| **Niveau de cache / infrastructure** | **Délai estimé max** |
| --- | --- |
| **Serveurs racines & TLD (.fr)** | **15 à 30 min** |
| **Résolveurs publics (Google 8.8.8.8, etc.)** | **15 à 60 min** |
| **CDN (Cloudflare, Akamai...)** | **1 à 2 h** |
| **FAI (France, Europe, Afrique...)** | **2 à 24 h** |
| **Pays lointains (zones isolées, Asie, etc.)** | **24 à 48 h max** |

**Test : Réussi**

****

**2nd Test : Réussi**

**Une image contenant texte, capture d’écran, Police, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Cependant : le test avec le navigateur accède au site avec une non secure alert.**

**Relance donc de la création du certificat :**

**cd : sudo certbot –apache -d visio.workeezconnect.fr**

**Création réussi :**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**C / Relions maintenant le serveur Jitsi à notre instance domaine + Adresse IP**

**Se connecter à son dépôt via le dossier ou se situent les clefs privée et publique :**

**ssh -i Tercium-instance\_key/tercium-instance\_key** [**ubuntu@84.234.29.126**](mailto:ubuntu@84.234.29.126)

**Ajouter le dépôt de**[**Jitsi Meet**](https://jitsi.org/jitsi-meet/)**:**

**Préambule :** Le dépôt fourni par Jitsi utilise le chiffrement, donc vérifier que vous avez bien le paquet apt-transport-https.

**cd : sudo apt install apt-transport-https ca-certificates curl gnup -y**

➤ **apt-transport-https**

Permet à apt (le gestionnaire de paquets) de télécharger des paquets depuis des dépôts HTTPS (sécurisés via TLS). Sans cela, apt ne pourra pas accéder à https://download.jitsi.org.

➤ **ca-certificates**

Installe les autorités de certification (CA) racines utilisées pour vérifier la validité des certificats SSL/TLS lors des connexions sécurisées. Indispensable pour vérifier le certificat du dépôt Jitsi, de Let's Encrypt, etc.

➤ **curl**

Outil de ligne de commande pour faire des requêtes HTTP(S), utilisé ici pour récupérer la clé GPG de Jitsi.

➤ **gnupg**

Contient gpg (GNU Privacy Guard), utilisé pour décrypter et gérer les clés de signature GPG, comme celle du dépôt Jitsi.

➤ **-y**

Flag qui force l’acceptation automatique de l’installation sans demander confirmation à chaque paquet. Très utilisé en script pour éviter l’interaction manuelle.

**# Importer la clé de manière sécurisée**

**curl** [**https://download.jitsi.org/jitsi-key.gpg.key**](https://download.jitsi.org/jitsi-key.gpg.key) **| gpg --dearmor | sudo tee /usr/share/keyrings/jitsi-keyring.gpg > /dev/null**

**# Ajouter le dépôt en l’associant à la clé**

**echo "deb [signed-by=/usr/share/keyrings/jitsi-keyring.gpg] https://download.jitsi.org stable/" | sudo tee /etc/apt/sources.list.d/jitsi-stable.list > /dev/null**

**# Mettre à jour les paquets**

**sudo apt update**

**# Installation de Jitsi Meet**

**Sudo apt install jitsi-meet -y**

**# Associer l’IP et le nom de domaine : Déjà réaliser !**

**"Si votre nom de domaine pointe déjà via DNS public vers l’adresse IP de l’instance, l’étape /etc/hosts est facultative."**

**# Vérifier l’état des services des serveurs**

**systemctl status jitsi-videobridge2 systemctl status jicofo systemctl status prosody**

**# Analyse DNS complète avec dig.**

**dig +short visio.workeezconnect.fr**

**Jour-5 Tercium : Vérification de l’installation + derniers modules**

**# Vérification DNS + HTTP avec curl.**

**curl -i** [**http://visio.workeezconnect.fr**](http://visio.workeezconnect.fr)

**# Vérification croisée : IP publique actuelle du serveur.**

**curl -4 ifconfig.me**

**Fichier à éditer : sudo nano /etc/prosody/conf.d/visio.workeezconnect.fr.cfg.lua**

**Code de base avec GNU nano 7.2 :**

**VirtualHost "visio.workeezconnect.fr"**

**authentication = "internal\_hashed"**

**modules\_enabled = {**

**"bosh";**

**"pubsub";**

**"ping";**

**"speakerstats";**

**"conference\_duration";**

**"muc\_lobby\_rooms"; -- ✅ Ne rien ajouter après ce point-virgule dans la table !**

**}**

**Component "conference.visio.workeezconnect.fr" "muc"**

**Component "auth.visio.workeezconnect.fr"**

**authentication = "internal\_hashed"**

**Component "focus.visio.workeezconnect.fr"**

**component\_secret = "FOCUS\_SECRET" Il faudra ajouter en sus la possibilité d’inviter**

**# Redémarrage Prosody :**

**sudo systemctl restart prosody**

**# Vérifier qu’il est actif :**

**sudo systemctl status prosody**

**# Vérifier si le port** **5280** **est ouvert :** **sudo ss -tuln | grep 5280**

**# Tester l’accès HTTP BOSH : curl http://localhost:5280/http-bind/**

**# Activer si nécessaire le Firewall sur le port ( UFW doit être actif) : sudo ufw allow 5280/tcp**

**# Vérification de la configuration**

**sudo prosodyctl check config**

**1/ Il manque des modules :**

**modulemanager: Unable to load module 'conference\_duration'**

**modulemanager: Unable to load module 'speakerstats'**

**modulemanager: Unable to load module 'muc\_lobby\_rooms'**

**# On les installe avec : sudo apt install prosody-modules && on redémarre avec sudo systemctl restart prosody**

**2/ Bug car il demande un localhost (qui pose un problème lors des connexions avec l’instance).** **L’erreur indique que le script post-installation de jitsi-meet-prosody échoue parce qu’il attend un fichier /etc/prosody/conf.avail/localhost.cfg.lua qui n'existe pas.**

**# Création de ce fichier : localhost.cfg.lua**

**sudo nano /etc/prosody/conf.avail/localhost.cfg.lua**

Contenu minimal du fichier (pour débloquer dpkg) :

**VirtualHost "localhost"**

**authentication = "anonymous"**

**# On relance la configuration des paquets cassés**

**sudo dpkg --configure -a**

**# On vérifie l’installation**

**sudo apt install -f**

**sudo systemctl restart prosody jicofo jitsi-videobridge2 nginx**

**# Créer un utilisateur XMPP :**  **sudo prosodyctl register admin visio.workeezconnect.fr motdepasse**

**3/ Cas Particulier source de problèmes :**

**Conflit localhost vs nom de domaine public** C’est **le problème principal qui a perturbé Certbot / Let's Encrypt / Apache** :

**▶️ Cause :**

Jitsi installe par défaut un vhost Apache localhost.conf :

apache

CopyEdit

<VirtualHost \*:80>

ServerName localhost

DocumentRoot "/usr/share/jitsi-meet"

</VirtualHost>

Simultanément, ton domaine réel (visio.workeezconnect.fr) est servi par :

* /etc/apache2/sites-enabled/000-default.conf (HTTP)
* /etc/apache2/sites-enabled/000-default-le-ssl.conf (HTTPS)

Certbot ou acme.sh essaie de **créer un challenge via le nom de domaine réel**, mais Apache répond par défaut avec la conf localhost, car elle est mal priorisée.

**▶️ Symptôme :**

bash

CopyEdit

curl http://localhost/.well-known/acme-challenge/test.txt

donne une **redirection 301 → HTTPS sur localhost**, ce qui empêche la validation http-01 par Let's Encrypt.

**▶️ Solutions recommandées :**

1. **Désactiver localhost.conf** s’il n’est plus nécessaire :

bash

CopyEdit

sudo a2dissite localhost.conf

sudo systemctl reload apache2

1. **S’assurer que visio.workeezconnect.fr est le ServerName prioritaire** dans les fichiers 000-default.conf et 000-default-le-ssl.conf.
2. **Contrôler l’ordre de priorité via** :

bash

CopyEdit

sudo apachectl -S

Et **supprimer tout vhost résiduel localhost** qui interfère.

✅ Résumé logique :

| **Test** | **Commande** | **Interprétation** |
| --- | --- | --- |
| **IP DNS** | **dig +short visio.workeezconnect.fr** | **Résultat doit être l’IP publique du serveur** |
| **IP publique (instance)** | **curl -4 ifconfig.me** | **Doit matcher celle obtenue par dig** |
| **Résolution + Apache OK** | **curl -I http://visio.workeezconnect.fr** | **HTTP 200/301/302 = OK** |
| **Conf Apache (ServerName)** | **sudo apachectl -S** | **Doit afficher visio.workeezconnect.fr en vhost** |

**On va installer Nginx fortement recommandé pour la sécurisation de échanges, cela implique arrêter Apache car il y aurait conflit sur le port 80 !**

**# Installer Nginx**

**sudo apt install -y nginx**

**# Démarrer le service**

**sudo systemctl start nginx**

**# Le rendre actif à chaque démarrage**

**sudo systemctl enable nginx**

**# Arrêt d’Apache2**

**sudo systemctl stop apache2**

**sudo systemctl disable apache2**

**# Supprimer d’Apache2**

**sudo apt purge apache2 apache2-utils apache2-bin apache2.2-common -y**

**Puis, s’assurer que le FQDN Fully Qualified Domain Name : nom de domaine complet incluant tous les niveaux (ex. auth.visio.workeezconnect.fr) est reconnu**

**echo "127.0.1.1 visio.workeezconnect.fr" | sudo tee -a /etc/hosts**

**# redémarrer Nginx**

**sudo systemctl start nginx**

**sudo systemctl enable nginx**

**# Vérifions l’état de Nginx et celui du port 80**

**sudo systemctl status nginx**

**sudo lsof -i :80**

**# Vérification les fichiers de configuration NGINX pour jitsi**

**cat /etc/nginx/sites-available/visio.workeezconnect.fr.conf | grep ssl**

**cat: /etc/nginx/sites-available/visio.workeezconnect.fr.conf: No such file or directory**

**Il faudra donc le crée**

**sudo nano /etc/nginx/sites-available/visio.workeezconnect.fr.conf**

**server {**

**listen 80;**

**server\_name visio.workeezconnect.fr;**

**location / {**

**proxy\_pass http://localhost:8000; # ou 5000 selon ton backend**

**proxy\_http\_version 1.1;**

**proxy\_set\_header Upgrade $http\_upgrade;**

**proxy\_set\_header Connection "upgrade";**

**proxy\_set\_header Host $host;**

**proxy\_set\_header X-Real-IP $remote\_addr;**

**}**

**location /http-bind {**

**proxy\_pass http://localhost:5280/http-bind;**

**proxy\_set\_header Host $host;**

**proxy\_http\_version 1.1;**

**proxy\_set\_header X-Forwarded-For $remote\_addr;**

**}**

**}**

**# Créer le lien symbolique**

**sudo ln -s /etc/nginx/sites-available/visio.workeezconnect.fr.conf /etc/nginx/sites-enabled/**

**# Vérifions la configuration**

**sudo nginx -t**

**# Redémarrer Nginx sudo systemctl restart nginx**

**# Insérons dans ce fichier un bloc sécurisé avec HTTPS + Cerbot et vérifions que le certificat est présent sudo ls /etc/letsencrypt/live/visio.workeezconnect.fr/**

**# Non il faut installer le plugin Certbot pour Nginx**

**sudo apt install python3-certbot-nginx -y**

**Puis Relancer la commande Certbot**

**sudo certbot --nginx -d visio.workeezconnect.fr**

**4/ Ici je stop l’installation et je débug car il y a eu trop de manipulations générant des bugs récurrents.**

**1-Corriger les erreurs d'installation des paquets Jitsi (dpkg)** et assurer une base fonctionnelle propre. **cd: hostnamectl**

**Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

**2-Corriger /etc/hosts**

**cd:** **sudo nano /etc/hosts**

**Une image contenant texte, logiciel, Logiciel multimédia, Police

Le contenu généré par l’IA peut être incorrect.**

**3-Redémarrer Prosody**

**cd:** **sudo systemctl restart prosody**

**4-Redémarrer Jicofo et JVB**

**cd:** **sudo systemctl restart jicofo**

**cd:** **sudo systemctl restart jitsi-videobridge2**

**5-** **Vérifier l’état**

**cd: sudo systemctl status prosody jicofo jitsi-videobridge2 ok active (running )**

**Jour-6 Tercium 2nde semaine | Déploiement**

**Sur ce PC je ne suis pas en ROOT donc soit on passe en tant qu’administrateur ponctuelle : Sudo, soit passage au mode WSL.**

1. **Recréation d’une instance et modification donc de l’adresse IP.**

**Une image contenant texte, nombre, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

1. **Dans DNS insérer la nouvelle IP et surtout submit et contrôler celui-ci.**
2. **Pinger la nouvelle adresse IP.**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

1. **Commande : nouvelle instance Infomaniak : DNS**

**Une image contenant texte, nombre, logiciel, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

1. **Connexion réussie via Git Bash :** ssh -i Tercium-instance\_key/tercium-instance\_key [ubuntu@84.234.28.98](mailto:ubuntu@84.234.28.98) **/ à chaque instance détruite / recréer, l’IP adresse change**.

**Tester avec :**

**cd :** **dig visio.workeezconnect.fr +short**

**cd : nslookup visio.workeezconnect.fr**

**Résultat nous pointons toujours vers l’ancienne IP adress de l’instance détruite la semaine dernière.**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

1. **Nous avons un conflit pour accéder aux clefs, précisément la privé de mon côté. Pour continuer nous devons alors pointer d’abord dans le bon dossier ; ici changement du nom du dossier en ssh.key:**

**En Powershell : cd : C:\Users\test\Documents\Tercium\_Stage\ssh\_keys**

**En Bash (git\_bash ou en WSL) : ssh -i /c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key -o IdentitiesOnly=yes** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Ou**

**ssh -i /c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key -o IdentitiesOnly=yes ubuntu@visio.workeezconnect.fr**

**Lors de la recréation d’une instance, L’on doit à chaque fois relier le nouvel IP Adress au domaine. Sans ce lien, il ne peut y avoir connexion SSH et d’installation de Jitsi.**

**Tester avec :**

**cd :** **dig visio.workeezconnect.fr +short**

**cd : nslookup visio.workeezconnect.fr**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**cd : dig visio.workkeezconnect.fr @1.1.1.1 +trace**

**Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

**On se connecte, chemin vers des fichiers clefs:**

**C:\Users\test\Documents\Tercium\_Stage\ssh\_keys**

**Powershell :**

**cd : ssh -i "C:\Users\test\Documents\Tercium\_Stage\ssh\_keys\tercium-instance\_key" -o IdentitiesOnly=yes** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Ou**

**Git Bash :**

**cd : ssh -i ~/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key -o IdentitiesOnly=yes** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Vérification complémentaires :**

**1/ Permissions sur la clé privée (depuis WSL/Git Bash) :**

**cd : chmod 600 ~/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key**

**2/ Connexion avec vérification du fingerprint :  
Tu peux ajouter -v pour obtenir des logs :**

**cd : ssh -i ~/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key -o IdentitiesOnly=yes -v** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

**Nettoyer le cache DNS interne en powershell: cd : ipconfig /flushdns**

**PRÉREQUIS**

1. **Nom de domaine pointant vers l’IP publique : visio.workeezconnect.fr → 37.156.46.238.**
2. **Accès SSH root ou sudo via ta clé (Tercium-instance\_key).**
3. **Résolution DNS locale correcte (dig, ping, etc.).**
4. **Port TCP 443 (HTTPS) et 80 (HTTP) ouverts dans les règles de pare-feu GCP/Infomaniak.**
5. **ufw désactivé ou configuré (on vérifiera ça).**
6. **Pas d’Apache2 actif pour éviter conflit avec NGINX.**

**ÉTAPES POUR INSTALLER JITSI + NGINX AVEC CONFIG LOCALE**

**Installation des paquets requis :**

sudo apt update && sudo apt upgrade -y

sudo apt install curl gnupg2 software-properties-common apt-transport-https -y

**Ajout du dépôt Jitsi officiel :**

**Ajout la clef GPG :**

**curl https://**download.jitsi.org/jitsi-key.gpg.key | sudo gpg --dearmor -o /usr/share/keyrings/jitsi-keyring.gpg

**Ajoute la clé GPG :**

**echo** 'deb [signed-by=/usr/share/keyrings/jitsi-keyring.gpg] https://download.jitsi.org stable/' | sudo tee /etc/apt/sources.list.d/jitsi-stable.list

**sudo apt update**

**Installation de Jitsi Meet :**

**cd : sudo apt install -y jitsi-meet**

🧠 Pendant l’installation : Possibilités

* FQDN : visio.workeezconnect.fr
* Choisir **"configurer SSL plus tard (manuel)"**

**7. Génération du certificat SSL (Let’s Encrypt)**

sudo apt install certbot python3-certbot-nginx -y

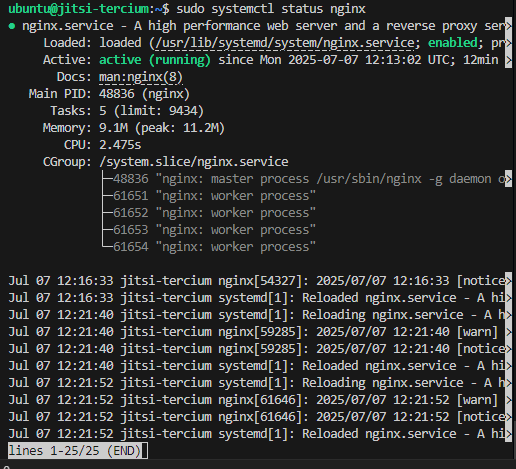
sudo certbot --nginx -d visio.workeezconnect.fr

Valide automatiquement le NGINX + active HTTPS.

**Étapes suivantes à vérifier pour finaliser le déploiement :**

1. **Vérifier que Nginx est bien configuré pour Jitsi**

**cd :** **sudo systemctl status nginx**

****

1. **Tester la connexion HTTPS depuis un navigateur : Accède à** [**https://visio.workeezconnect.fr**](https://visio.workeezconnect.fr)

**Une image contenant texte, capture d’écran, Logiciel multimédia, logiciel

Le contenu généré par l’IA peut être incorrect.**

1. **Vérifier les ports nécessaires ouverts : il faut installer ufw**

**cd : sudo apt update && sudo apt install ufw**

**cd : sudo ufw allow 80,443/tcp**

**sudo ufw allow 10000/udp**

**sudo ufw enable**

**sudo ufw status**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

1. **Vérifier les modules Prosody :**

**cd : sudo cat /etc/prosody/conf.avail/visio.workeezconnect.fr.cfg.lua | grep modules\_enabled -A 20**

1. **Test de fonctionnement des modules : relance puis vérification de son état**

**cd : sudo systemctl restart prosody**

**Puis**

**cd : sudo systemctl status prosody**

**Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Ne pouvant contrôler Jitsi du côté d’Infomaniak, car nous ne voyons pas l’installation de celui-ci. Cependant la section DNS via Zones nous donnera ce tableau qui nous permettra de réaliser des vérifications.**

1. **Vérifier la résolution DNS (commande dig) :**

**cd : dig visio.workeezconnect.fr @8.8.8.8 +short**

**Réponse attendue et reçue : 37.156.48.238**

****

1. **Tester la connectivité HTTP(S) (commande curl) :**

**cd : curl -I** <https://visio.workeezconnect.fr>

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

1. **S’il y avait échec avec HTPPS, tester HTTP:**

**cd : curl -I** <http://visio.workeezconnect.fr>

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

1. **Tester le port 443 avec telnet ou nc**

**cd : nc -zv visio.workeezconnect.fr 443**

**Une image contenant texte, capture d’écran, Police, ligne

Le contenu généré par l’IA peut être incorrect.**

1. **Tester la chaîne de certificat SSL (Certbot utilisé)**

**cd : echo | openssl s\_client -connect visio.workeezconnect.fr:443 -servername visio.workeezconnect.fr**

**Une image contenant texte, capture d’écran, Police, logiciel

Le contenu généré par l’IA peut être incorrect.**

* **ISRG Root X1 → racine de Let's Encrypt**
* **Let's Encrypt E6 → intermédiaire**
* **visio.workeezconnect.fr → certificat de ton instance**

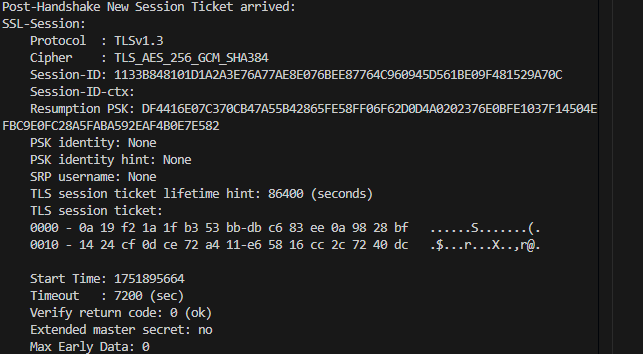
**Vérification SSL :**

**Une image contenant texte, Police, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

**Verify return code: 0 (ok)**

**Vérification TLS :**

****

**Conclusion :**

| **Élément vérifié** | **État** |
| --- | --- |
| Certificat SSL bien installé | ✅ OK |
| Chaîne complète avec intermédiaire & racine | ✅ OK |
| Nom de domaine valide (CN = visio...) | ✅ OK |
| Protocole TLS récent (1.3) | ✅ OK |

**Jour-7 Tercium 2nde semaine | installation modules de monitoring**

**Monitoring Infomaniak :**

**Si vous arrêtez temporairement l'instance (sans la supprimer) :**

* Jitsi reste installé sur le disque
* Vous devrez juste vous reconnecter en SSH avec la nouvelle adresse IP
* Aucune réinstallation nécessaire

**Si vous supprimez complètement l'instance :**

* Tout est effacé, y compris Jitsi
* Vous devrez tout réinstaller depuis zéro sur une nouvelle instance

**Si votre instance redémarre avec une nouvelle IP :**

* Jitsi restera installé mais il faudra probablement ajuster la configuration
* Le certificat SSL devra être régénéré si vous utilisez un nom de domaine
* Vous devrez mettre à jour l'adresse IP dans votre configuration SSH

Dans la plupart des cas avec les services cloud, un simple arrêt/redémarrage conserve vos données. Seule la suppression définitive de l'instance efface tout.

Quel est votre cas exactement ? Vous utilisez quel service cloud (AWS, Google Cloud, etc.) ?

Avec Infomaniak, vous pouvez arrêter votre instance sans l'effacer. Voici comment procéder :

**Dans le manager Infomaniak :**

1. Connectez-vous à votre espace client Infomaniak
2. Allez dans la section "Public Cloud" ou "Serveurs"
3. Sélectionnez votre instance
4. Cliquez sur "Arrêter" (bouton stop/pause)

**Ce qui se passe quand vous arrêtez :**

* L'instance s'arrête mais le disque virtuel est conservé
* Vous ne payez plus les ressources CPU/RAM (seulement le stockage)
* Toutes vos données et installations (dont Jitsi) restent intactes

**Pour redémarrer plus tard :**

* Cliquez sur "Démarrer" dans le manager
* L'instance redémarre avec la même configuration
* Vous devrez peut-être noter la nouvelle adresse IP publique
* Connectez-vous en SSH avec cette nouvelle IP

**Important :** Ne cliquez pas sur "Supprimer" ou "Détruire" - cela effacerait définitivement tout.

L'arrêt d'instance chez Infomaniak est réversible et conserve vos données, contrairement à la suppression qui est définitive.

**Planning structuré de l’audit:**

* Audit technique général
* Inventaire des environnements matériels (serveurs, réseau, postes...)
* Demande d'autorisation pour les scans réseau (très important !)
* Validation de la topologie réseau réelle

**Concernant les autorisations :**

* netstat : généralement OK (analyse locale)
* nmap : nécessite autorisation écrite (scan réseau)
* arpscan : idem, peut être perçu comme intrusif

**Conseils pour la demande :**

* Préciser que c'est pour valider la documentation existante
* Mentionner les créneaux horaires prévus
* Expliquer que c'est pour la sécurité/conformité
* Proposer un rapport de synthèse

**Phase discrète :**

* Netstat poste par poste → pas d'impact réseau, juste analyse locale
* Collecte d'infos sans perturber l'activité

**Phase avec autorisation :**

* Nmap et arpscan après accord du boss
* Approche transparente et professionnelle

**Le plus intéressant :** La configuration sécurité avec API et alertes smartphone ! C'est du monitoring intelligent :

* Alertes discrètes en temps réel
* Le boss reste informé sans être submergé
* Réactivité en cas d'incident

**Conseils pour convaincre :**

* Expliquer que nmap/arpscan permettront de valider que cette API de sécurité ne rate rien
* Montrer que c'est pour renforcer son système d'alertes existant
* Proposer un test sur une plage horaire définie

Travail d'auditeur ! Vous combinez discrétion, professionnalisme et valeur ajoutée. L’entreprise appréciera sûrement cette approche qui respecte ses processus tout en améliorant la sécurité.

Le système d'alertes smartphone montre qu'il y a une sensibilisation à la sécurité.

**Déploiement et correction des bugs : pop-ups absent, absence de micro, de caméra etc.**

**Résumé des actions à vérifier côté client**

| **Élément** | **Vérification** |
| --- | --- |
| **Pop-up caméra/micro** | **via cadenas navigateur** |
| **Notifications autorisées** | **dem** |
| **Console JS (erreurs)** | **🔍 check getUserMedia, ICE** |
| **Headers Nginx** | **Permissions-Policy, CORS** |
| **Services Jitsi** | **jicofo, videobridge** |

**09h20 fin des tests :**

* **L’import d’un Micro et d’une caméra externe sont supportés.**
* **Ceux sont pour la caméra une 720p essential B et pour le micro un Razer omnidirectionnel semi-professionnel.**
* **Les fonctions de partage d’écran, d’invitation via un lien simple ou multiple fonctionnent.**
* **L’ensemble des fonctions attendues existantes sur zoom et anciennement skype sont présentes.**

[**https://www.onlyoffice.com/blog/fr/2023/02/zoom-ou-jitsi**](https://www.onlyoffice.com/blog/fr/2023/02/zoom-ou-jitsi)

**En Annexe des saisies d’écran de l’interface avec les fonctionnalités disponibles.**

**Comparaison à titre indicatif de l’usage de Zoom :**

* [**Zoom Spaces**](https://explore.zoom.us/fr/products/zoom-rooms/)**est un moyen innovant d’organiser des réunions virtuelles au sein d’équipes hybrides. Les services inclus sont la réservation d’un espace de travail (par exemple, un bureau ouvert, un dispositif Zoom), un connecteur de salle de conférence pour rejoindre les réunions avec un équipement SIP/H.323, et des salles Zoom pour les conférences qui impliquent de nombreuses personnes des deux côtés.**
* [**Zoom Events**](https://explore.zoom.us/fr/products/event-solutions/)**est une solution dédiée à l’organisation d’événements virtuels, des annonces internes aux tables rondes. Cette option est idéale pour le réseautage, la présentation de contenu dans des stands d’exposition, la formation d’équipes dans des événements à session unique et des webinaires en ligne.**
* [**Zoom Contact Center**](https://explore.zoom.us/fr/products/contactcenter/)**vous permet de créer un environnement omnicanal pour fournir une assistance à vos clients. Les outils disponibles comprennent un centre de contact optimisé pour la vidéo dans le cloud et l’IA conversationnelle.**

**Coût & Résultat d’usage :**

**Comme chaque solution comprend différents ensembles de fonctionnalités, les prix varient en fonction de votre demande.**

**Plans tarifaires :**

**Le coût d’un logiciel de vidéoconférence est un aspect essentiel pour la plupart des équipes, car chaque entreprise souhaite économiser sur les solutions numériques et investir de l’argent dans la croissance.**

**Zoom offre un plan gratuit avec les caractéristiques suivantes :**

* **Jusqu’à 40 minutes par réunion**
* **100 participants par réunion**
* **3 tableaux modifiables avec 25 Mo de stockage en nuage**
* **Chat d’équipe**

**Le plan de base est suffisant pour la communication personnelle, tandis que les entreprises envisageront des plans payants avec des fonctionnalités puissantes. Les prix commencent à 149 $ par utilisateur/mois et peuvent inclure :**

* **Jusqu’à 1000 participants**
* **Réunions illimitées**
* **30 heures par réunion**
* **Stockage illimité des enregistrements sur le cloud**
* **Tableaux blancs illimités**
* **Authentification unique (Single Sign-On – SSO)**
* **Domaines gérés**
* **Marque de l’entreprise**

**Sécurité :**

**La sécurité et la confidentialité sont essentielles au succès de la vidéoconférence. Personne ne souhaite que des personnes non autorisées aient accès à des discussions privées contenant des informations sensibles.**

**Jitsi est axé sur la sécurité et propose des salles de réunion éphémères, ce qui signifie qu’elles n’existent que pendant la réunion. Les autres options de confidentialité comprennent la protection par mot de passe, le lobby, le cryptage de bout en bout.**

**Jitsi Meet pour un déploiement sur site ne dispose pas de moteurs d’analyse préconfigurés. Même dans la version en ligne, il ne conserve aucune information personnelle, comme les noms ou les adresses électroniques.**

**Zoom est considéré comme relativement sûr grâce à son cryptage de bout en bout. Toutefois, des sources fiables signalent que Zoom utilise l’algorithme AES-128 au lieu de AES-256, ce qui rend sa sécurité plus vulnérable dans une certaine mesure. En outre, vous ne serez pas en mesure de vérifier le code source de ce logiciel pour vous assurer qu’il ne contient pas de virus ou de portes dérobées.**

**Dans la pratique, les deux solutions se sont avérées bien protégées. Si vous suivez les directives de base en matière de sécurité de l’information sur votre lieu de travail, les piratages et les attaques extérieures sont peu probables.**

**Plugin Jitsi pour ONLYOFFICE Docs :** [**https://www.onlyoffice.com/blog?p=26614**](https://www.onlyoffice.com/blog?p=26614)

**Monitoring : 3ième étape de la gestion de projet / Explications du processus**

**OBJECTIF**

1. **Monitorer l’instance Jitsi (CPU, RAM, disques, etc.).**
2. **Visualiser dans Grafana.**
3. **Superviser la disponibilité du service Jitsi via Prometheus (ou Blackbox Exporter).**
4. **Intégrer Pfsense et ipatable.**
5. **Alerter si nécessaire.**

**Les modes de connexion :**

**1/Connexion avec vérification du fingerprint : Vscode en Git Bash  
Tu peux ajouter -v pour obtenir des logs :**

**cd : ssh -i ~/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key -o IdentitiesOnly=yes -v** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Naviguer via WSL demande une réécriture. Pour accéder au dossier C:\Users\test\Documents\Tercium\_Stage depuis un terminal WSL, vous devrez naviguer en utilisant le chemin Linux approprié, car WSL monte les disques Windows sous /mnt/c/.**

**cd : cd /mnt/c/Users/test/Documents/Tercium\_Stage**

**Puis**

**cd : ssh -i /mnt/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key -o IdentitiesOnly=yes -v** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Cependant l’accès à une instance infomaniak via WSl est bloquée. La raison les permissions SSH sont trop souples. La commande ci-dessous est refusée.**

**cd : chmod 600 /mnt/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key**

**Explication : Différences entre Git Bash et WSL**

1. **Git Bash et les permissions de fichiers :**
   * **Git Bash** (sous Windows) fonctionne généralement en émulation Unix, mais il a une gestion un peu plus souple des permissions de fichiers. Il peut donc permettre l'accès à des clés privées même avec des permissions un peu plus larges.
2. **WSL et les permissions sous Windows :**
   * **Sous WSL,** les permissions de fichiers sont plus strictes, car elles sont basées sur le système de fichiers Linux qui s'exécute sur Windows. Lorsque vous accédez à un fichier Windows via /mnt/c/, WSL essaie de respecter les permissions de fichier Linux. Par défaut, les fichiers sous /mnt/c/ peuvent être considérés comme trop accessibles si leurs permissions sont trop larges.

**Testons sous le terminal Git Bash :**

**cd : ls -l /c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key**

**Une image contenant texte, capture d’écran, Police, ligne

Le contenu généré par l’IA peut être incorrect.**

* **Permissions actuelles : rw-r--r--**
  + **Propriétaire (test) : lecture et écriture (rw-).**
  + **Groupe et autres utilisateurs : lecture seule (r--).**
* **Propriétaire : test (probablement votre utilisateur actuel).**
* **Taille : 3381 octets.**
* **Date : Le fichier a été modifié pour la dernière fois le 1er juillet.**

**Problème :**

Les permissions actuelles du fichier clé (**rw-r--r--**) ne sont pas suffisantes pour que **SSH** fonctionne correctement. **SSH** exige que la clé privée soit accessible uniquement par le propriétaire du fichier, ce qui nécessite des permissions strictes de type **600**.

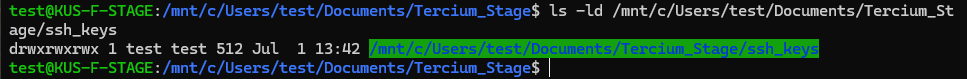
**Modifions les permissions** pour les rendre compatibles avec SSH :

**cd : chmod 600 /c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key**

**2/ Connexion avec WSL :**

**Testons sous le termina WSL :**

**cd : ls -l /mnt/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key**

****

**Modifions les permissions** pour les rendre compatibles avec SSH :

**cd : sudo chmod 600 /mnt/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key**

* **Des solutions et explications seront en annexe**

**3/ Connexion avec Powershell :**

**Testons sous le terminal sous Powershell :**

**cd : ssh -i C:\Users\test\Documents\Tercium\_Stage\ssh\_keys\tercium-instance\_key -o IdentitiesOnly=yes -v ubuntu@37.156.46.238**

**Restriction de privilége :**

**Usage possible d’icacls, un utilitaire en ligne de commande de Windows, pour modifier les permissions sur le fichier clé privée. La commande suivante restreint l'accès à la clé privée aux seuls utilisateurs autorisés :**

**cd : icacls "tercium-instance\_key" /inheritance:r /grant:r "test:(R)"**

**Explication :**

* **/inheritance:r :** Supprime l'héritage des permissions (cela empêchera les autres utilisateurs d'hériter des permissions du répertoire parent).
* **/grant:r "test:(R)" :** Donne à l'utilisateur **test (**remplacez par votre nom d'utilisateur si nécessaire**) l'accès en lecture seule (R) sur ce fichier.**

**Vérification :**

**cd : icacls "tercium-instance\_key"**

**L’installation des outils de supervision :**

**L’on part toujours du dossier ou se situe des deux clefs, la publique et la privée.**

**1/ Powershell :**

**cd : test@KUS-F-STAGE MINGW64 ~/Documents/tercium\_Stage**

**Connexion SSH  en Powershell:**

**cd : ssh -i C:\Users\test\Documents\Tercium\_Stage\ssh\_keys\tercium-instance\_key -o IdentitiesOnly=yes -v** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**2/ Connexion SSH en Git Bash :**

**cd : ssh -i ~/.ssh/tercium-instance\_key** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238) en linux ou WSL

**ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Installation de PROMETHEUS:**

**Télécharger :**

**cd : cd ~wget https://github.com/prometheus/prometheus/releases/download/v2.52.0/prometheus-2.52.0.linux-amd64.tar.gz**

**tar -xvf prometheus-2.52.0.linux-amd64.tar.gz**

**sudo mv prometheus-2.52.0.linux-amd64 /opt/prometheus**

**Service :**

**cd : sudo tee /etc/systemd/system/prometheus.service > /dev/null <<EOF**

**[Unit]**

**Description=Prometheus**

**Wants=network-online.target**

**After=network-online.target**

**[Service]**

**User=prometheus**

**Group=prometheus**

**Restart=on-failure**

**ExecStart=/opt/prometheus/prometheus \\**

**--config.file=/etc/prometheus/prometheus.yml \\**

**--storage.tsdb.path=/var/lib/prometheus \\**

**--web.console.templates=/etc/prometheus/consoles \\**

**--web.console.libraries=/etc/prometheus/console\_libraries**

**--web.listen-adress=0.0.0.0 :9091**

**[Install]**

**WantedBy=multi-user.target**

**EOF**

**sudo systemctl daemon-reload**

**sudo systemctl enable --now prometheus**

**Définir les droits :**

**cd :**

**sudo useradd -rs /bin/false prometheus**

**sudo mkdir -p /etc/prometheus /var/lib/prometheus**

**sudo cp /opt/prometheus/prometheus.yml /etc/prometheus/**

**sudo cp -r /opt/prometheus/consoles /opt/prometheus/console\_libraries /etc/prometheus/**

**sudo chown -R prometheus: /etc/prometheus /var/lib/prometheus**

**Créer le dossier de données :**

**cd :**

**sudo mkdir -p /var/lib/prometheus**

**sudo chown -R prometheus: /var/lib/prometheus**

**Recharger systemd :**

**cd : sudo systemctl daemon-reload**

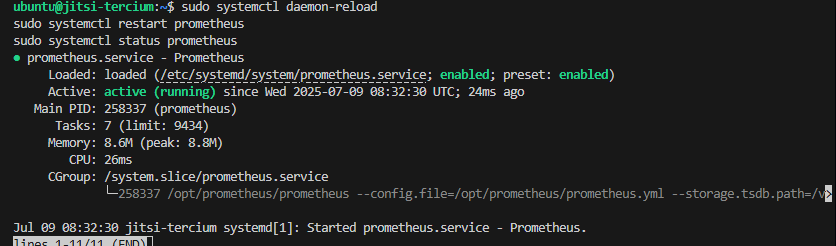
**Lancer Prometheus :**

**cd : sudo systemctl enable --now prometheus**

**Vérifier :**

**cd : sudo systemctl status prometheus**

**curl** [**http://localhost:9090**](http://localhost:9090)

****

**Attention conflit avec Jjtsi\_meet au port 9090 passage de prometheus au port 9091 via le le service group de infomaniak et au changement dans son script voir au-dessus.**

**Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

**Jour-8 Tercium 2nde semaine | monitoring installations**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Installation de NODE EXPORTER :**

**Télécharger :**

**cd : cd ~wget https://github.com/prometheus/node\_exporter/releases/download/v1.8.1/node\_exporter-1.8.1.linux-amd64.tar.gz**

**tar -xvf node\_exporter-1.8.1.linux-amd64.tar.gz**

**sudo mv node\_exporter-1.8.1.linux-amd64/node\_exporter /usr/local/bin/**

**Utilisateur & service :**

**cd : sudo useradd -rs /bin/false node\_exporter**

**sudo tee /etc/systemd/system/node\_exporter.service > /dev/null <<EOF**

**[Unit]**

**Description=Node Exporter**

**After=network.target**

**[Service]**

**User=node\_exporter**

**ExecStart=/usr/local/bin/node\_exporter**

**[Install]**

**WantedBy=default.target**

**EOF**

**sudo systemctl daemon-reload**

**sudo systemctl enable --now node\_exporter**

**Installation de BLACKBOX EXPORTER :**

**Télécharger :**

**cd : cd / opt**

**sudo wget** [**https://github.com/prometheus/blackbox\_exporter/releases/download/v0.25.0/blackbox\_exporter-0.25.0.linux-amd64.tar.gz**](https://github.com/prometheus/blackbox_exporter/releases/download/v0.25.0/blackbox_exporter-0.25.0.linux-amd64.tar.gz)

**sudo tar -xvf blackbox\_exporter-0.25.0.linux-amd64.tar.gz**

**sudo mv blackbox\_exporter-0.25.0.linux-amd64 /opt/blackbox\_exporter**

**Utilisateur & service :**

**cd : sudo nano /etc/systemd/system/blackbox\_exporter.service**

**[Unit]**

**Description=Blackbox Exporter**

**Wants=network-online.target**

**After=network-online.target**

**[Service]**

**User=prometheus**

**ExecStart=/opt/blackbox\_exporter/blackbox\_exporter \**

**--config.file=/opt/blackbox\_exporter/blackbox.yml**

**[Install]**

**WantedBy=multi-user.target**

**Redémarrer :**

**sudo systemctl daemon-reexec**

**sudo systemctl daemon-reload**

**sudo systemctl enable blackbox\_exporter**

**sudo systemctl start blackbox\_exporter**

**Installation de GRAFANA :**

**Installation via dépôt officiel :**

**cd : # Installation officielle (clé + repo)**

**sudo apt-get install -y software-properties-common**

**sudo apt-get install -y gnupg2 curl**

**sudo mkdir -p /etc/apt/keyrings**

**curl -fsSL https://apt.grafana.com/gpg.key | sudo gpg --dearmor -o /etc/apt/keyrings/grafana.gpg**

**echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee /etc/apt/sources.list.d/grafana.list**

**sudo apt update**

**sudo apt install grafana -y**

**# Activation et lancement**

**sudo systemctl enable grafana-server**

**sudo systemctl start grafana-server**

**Vérification avec :**

**cd : sudo systemctl status grafana-server**

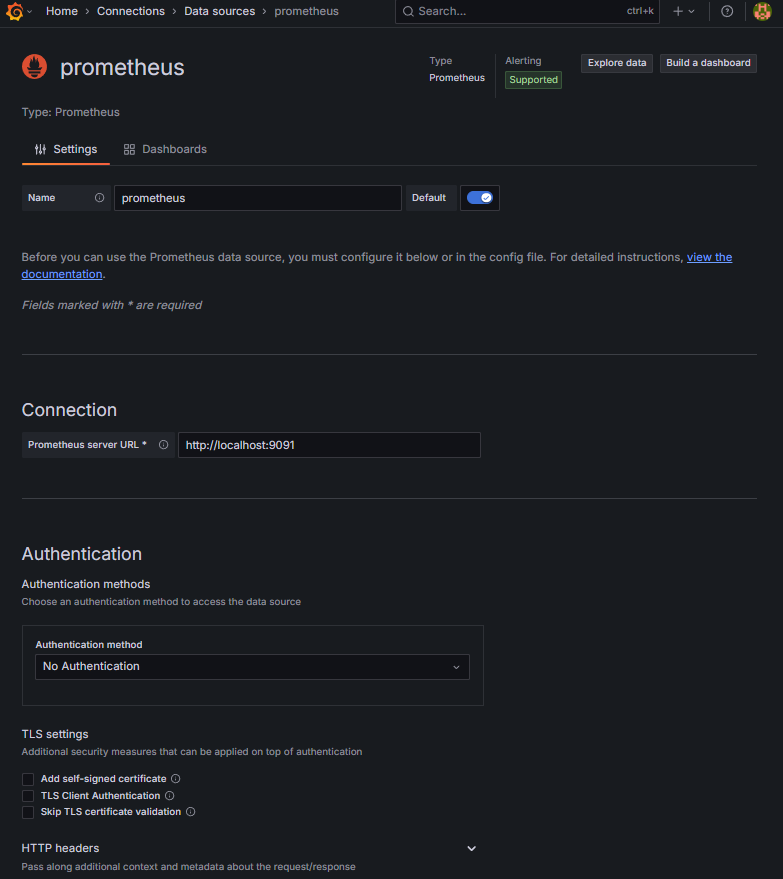
**Une image contenant texte, capture d’écran, Police, ligne

Le contenu généré par l’IA peut être incorrect.**

**🟢 Interface Web : http://<ip\_publique>:3000**

**Dans notre cas cela sera :** [**http://37.156.46.238:3000**](http://37.156.46.238:3000)

**Identifiant par défaut : admin / admin ici admin /BZTwwtq1964**

****

**Installation de PFSENSE :**

**1/ Commandes pour des dépôt github bash non accessibles :**

**cd : chmod +x pfsense\_exporter-linux-amd64**

**cd : sudo mv pfsense\_exporter-linux-amd64 /usr/local/bin/pfsense\_exporter**

**2/ Télécharger et placer le binaire : sous forme de python script via un repo GitHub** (sviskontree/pfsense\_exporter).

**Contient deux fichiers :**

**Script principal :**

**cd : wget** [**https://raw.githubusercontent.com/sviskontree/pfsense\_exporter/master/pfsense\_exporter.py**](https://raw.githubusercontent.com/sviskontree/pfsense_exporter/master/pfsense_exporter.py)

**Script secondaire de lancement :**

**cd : wget**

[**https://raw.githubusercontent.com/sviskontree/pfsense\_exporter/master/pfsense\_exporter.sh**](https://raw.githubusercontent.com/sviskontree/pfsense_exporter/master/pfsense_exporter.sh)

**Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

**Attention les scripts de pfsense sont en FreeBSD**

**cd : sudo mv pfsense\_exporter.py /usr/local/bin/**

**sudo mv pfsense\_exporter.sh /usr/local/etc/rc.d/**

**chmod +x /usr/local/bin/pfsense\_exporter.py /usr/local/etc/rc.d/pfsense\_exporter.sh**

**Ces commandes ne peuvent fonctionner que sous FreeBSD et sous pfsense**

**Il faudra créer un dossier local des services:**

**Donc :**

**cd : sudo mkdir -p /opt/pfsense\_exporter**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Puis déplacer les deux fichiers avec les autorisations vers l’instance et vérifier l’export :**

**cd : sudo mv pfsense\_exporter.py /opt/pfsense\_exporter/**

**sudo mv pfsense\_exporter.sh /opt/pfsense\_exporter/**

**sudo chmod +x /opt/pfsense\_exporter/\***

**cd : ls -l**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Puis vérifier l’emport :**

**cd : ls -l /opt/pfsense\_exporter/**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**/opt/pfsense\_exporter/**

**├── pfsense\_exporter.py**

**└── pfsense\_exporter.sh**

**Créer un service systemd :**

**cd : sudo nano /etc/systemd/system/pfsense\_exporter.service**

**Contenu :**

**cd :**

**[Unit]**

**Description=PfSense Exporter for Prometheus**

**After=network.target**

**[Service]**

**ExecStart=/usr/bin/python3 /opt/pfsense\_exporter/pfsense\_exporter.py**

**Restart=always**

**User=ubuntu**

**WorkingDirectory=/opt/pfsense\_exporter**

**[Install]**

**WantedBy=multi-user.target**

**Activation et démarrage du service :**

**cd : sudo systemctl daemon-reload**

**sudo systemctl enable --now pfsense\_exporter**

**sudo systemctl status pfsense\_exporter**

**ECHEC comme avec le premier dépôt linux.**

**On purge proprement :**

**Supprimer le service systemd :**

**cd : sudo systemctl stop pfsense\_exporter**

**sudo systemctl disable pfsense\_exporter**

**sudo rm /etc/systemd/system/pfsense\_exporter.service**

**sudo systemctl daemon-reload**

**Supprimer les fichiers Python:**

**cd : sudo rm -rf /opt/pfsense\_exporter/**

**Supprimer les logs:**

**cd : sudo journalctl --vacuum-time=1s**

**Configuration SNMP côté pfSense : pourquoi utiliser SNMP ?**

| **Usage** | **Action** |
| --- | --- |
| **Exploiter le script pfsense\_exporter.py avec les métriques pf\_ (comme dans ton JSON Grafana)** | **❌ PAS besoin de Telegraf** |
| **Récupérer via SNMP des métriques système ou bas-niveau (uptime, interfaces, CPU, etc.)** | **✅ Telegraf est utile** |

**Intérêt :**

| **Donnée à récupérer** | **Méthode** | **Outil** |
| --- | --- | --- |
| **États de firewall pfSense** | **API (HTTP)** | **pfsense\_exporter.py** |
| **Statut interfaces SNMP** | **SNMP (port 161)** | **Telegraf** |
| **UpTime général** | **SNMP** | **Telegraf** |
| **CPU, RAM (pfSense)** | **SNMP** | **Telegraf** |

**Prérequis :**

**Installer SNMP :**

**cd : sudo apt install snmp -y**

**1 Sur pfsense : Activer SNMP**

* **Menu : Services > SNMP**
* **Activez le service**
* **Port : 161, community : public (ou sécurisé)**
* **Activez les modules d’interface et de système**

**Mise à jour - Tests SNMP après ouverture du port 161/UDP**

**Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.**

**Ouverture du port 161  
Une règle Ingress a bien été ajoutée dans le groupe de sécurité visio-monitoring-group pour autoriser le trafic entrant sur le port 161/UDP depuis toutes les IPs (CIDR 0.0.0.0/0).**

* **Tests snmpwalk Trois commandes ont été lancées successivement :**
* **snmpwalk -v2c -c prom-readonly 37.156.46.238 1.3.6.1.2.1.1**
* **snmpwalk -v2c -c prom-readonly 192.168.1.1 1.3.6.1.2.1.1**
* **snmpwalk -v2c -c prom-readonly 192.168.0.1 1.3.6.1.2.1.1**

**Toutes ont retourné la même erreur :**

**Timeout: No Response from [IP ciblée]**

* **Hypothèses actuelles sur l'échec du test**
  + **L’interface pfSense à l’adresse supposée 192.168.1.1 ou 192.168.0.1 est injoignable depuis la VM Ubuntu car elle n’est pas sur le même sous-réseau virtuel.**
  + **Le service SNMP n’est pas activé ou autorisé sur pfSense.**
  + **L’adresse publique 37.156.46.238 ne correspond pas directement à pfSense mais à une VM d’application (Jitsi).**
  + **Le filtrage résiduel par un autre pare-feu ou une couche d’abstraction empêche la communication SNMP.**
* **Prochaine étape**
  + **Vérification de l’adresse réelle de pfSense (via console, routage, ou listing Infomaniak s’il y a accès).**
  + **Si accès impossible : documenter ce blocage comme limite d’infrastructure dans un environnement managé.**
  + **Continuer à utiliser uniquement Telegraf, Node Exporter et Blackbox Exporter sur l’instance Ubuntu pour le monitoring.**
* **Conclusion temporaire Le monitoring SNMP depuis la VM vers pfSense est non fonctionnel à ce stade, malgré les ouvertures de port. Une simulation sera envisagée pour compléter le rapport de soutenance.**

**2 Sur le serveur Ubuntu :**

**Contrôle de l’existence de telegraf :**

**cd : telegraf –version**

**réponse attendue : Telegraf 1.28.2 (git: HEAD 3d7aa20c)**

**Installer Telegraf :**

**cd : sudo apt update**

**sudo apt install -y telegraf  
réponse : le dépôt n’est pas connu d’apt == E: Unable to locate package telegraf**

**Étapes pour installer Telegraf sur Ubuntu :**

**# 1. Importer la clé GPG d’InfluxData**

**cd : wget -qO- https://repos.influxdata.com/influxdata-archive\_compat.key | sudo gpg --dearmor -o /etc/apt/keyrings/influxdata.gpg ok**

**# 2. Ajouter le dépôt InfluxData pour Ubuntu (remplacer $(lsb\_release -cs) si besoin)**

**cd : echo "deb [signed-by=/etc/apt/keyrings/influxdata.gpg] https://repos.influxdata.com/ubuntu $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/influxdata.list ok**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**# 3. Mettre à jour les paquets**

**cd : sudo apt update**  **ok**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**# 4. Installer Telegraf**

**cd : sudo apt install telegraf -y ok**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**# 5. Vérifie le statut de Telegraf**

**cd : sudo systemctl status telegraf ok**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**# 6. Activer et démarrer Telegraf**

**cd : sudo systemctl enable --now telegraf**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**échec : Afficher le statut du service avec erreur détaillée :**

**cd : sudo systemctl status telegraf.service**

**Exposer Telegraf à Prometheus :**

**Créer le dossier prometheus :**

**cd : sudo mkdir -p /etc/prometheus**

**Configuration de base de Prometheus :**

**yaml :**

**global:**

**scrape\_interval: 15s**

**evaluation\_interval: 15s**

**scrape\_configs:**

**- job\_name: 'prometheus'**

**static\_configs:**

**- targets: ['localhost:9090']**

**Étapes pour l’ajouter avec nano :**

**cd : sudo nano /etc/prometheus/prometheus.yml**

**Redémarrer Prometheus:**

**cd : sudo systemctl restart prometheus**

**\*\*\*\*\*\*\*\*\***

**Installation de IPATABLES : IMPOSSIBLE Actuellement.**

**Installation de l'exporter : Il y une obsolescence du dépôt ainsi qu’une demande de code absurde sur du github.**

**PRÉREQUIS :**

**cd : sudo apt update**

**sudo apt install -y golang git make**

**VERIFICATION :**

**cd : git --version**

**go version**

**make –version**

**NETTOYAGE :**

**cd :** **cd /opt**

**sudo rm -rf iptables\_exporter**

**CLONAGE & COMPILATION :**

**cd :**

**cd iptables\_exporter**

**make**

**Utilisateur dédié :**

**cd : sudo useradd -rs /bin/false iptables\_exporter**

**sudo chown -R iptables\_exporter:iptables\_exporter /opt/iptables\_exporter**

**Service :**

**cd : sudo nano /etc/systemd/system/iptables\_exporter.service**

**contenu :**

**[Unit]**

**Description=Iptables Exporter**

**After=network.target**

**[Service]**

**User=iptables\_exporter**

**ExecStart=/opt/iptables\_exporter/iptables\_exporter**

**[Install]**

**WantedBy=multi-user.target**

**sudo systemctl daemon-reload**

**sudo systemctl enable --now iptables\_exporter**

**Autoriser le port 9145 choisi dans Service Group :**

**cd : sudo ufw allow 9145/tcp**

**Vérification du lancement du service :**

**cd : sudo systemctl status iptables\_exporter**

**\*\*\*\*\*\*\*\*\***

**Jour-9 Tercium 2nde semaine | monitoring**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Connexion SSH en Powershell :**

**cd : ssh -i "C:\Users\test\Documents\Tercium\_Stage\ssh\_keys\tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**\*Contrôle des ports : Permissions**

|  |  |
| --- | --- |
| **sudo ufw allow 9091/tcp** | **Prometheus** |
| **sudo ufw allow 9100/tcp** | **Node Exporter** |
| **sudo ufw allow 9115/tcp** | **Blackbox Exporter** |
| **sudo ufw allow 3000/tcp** | **Grafana** |
| **sudo ufw allow 9145/tcp** | **Iptables\_Exporter** |

**cd : sudo ufw allow 9091/tcp**

**sudo ufw allow 9100/tcp**

**sudo ufw allow 9115/tcp**

**sudo ufw allow 3000/tcp**

**sudo ufw allow 9145/tcp / ECHEC**

**Une image contenant texte, capture d’écran, Police, logiciel

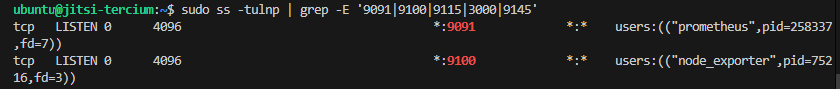
Le contenu généré par l’IA peut être incorrect.**

**\*Contrôle des ports :Listen**

**cd : sudo ss -tulnp | grep -E ‘9001|9100|9115|3000|9145’**

**Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

****

****

**Prometheus : Après modifications contenant port et les logs de blackbox\_http et node\_exporter l’on peut relancer et tester l’interface.**

**Redémarrer :**

**cd : sudo systemctl restart prometheus**

**Vérifier l’état :**

**cd : sudo systemctl status prometheus**

**curl** [**http://localhost:9091/targets**](http://localhost:9091/targets)

**Navigateur web :**

**cd :** [**http://37.156.46.238:9091/targets**](http://37.156.46.238:9091/targets)

**Sécurisation HTTPS de Prometheus et Grafana**

**A. Reverse proxy avec NGINX pour Grafana (port 3000)**

**Grafana a besoin d’alimenter ses analyses via des dashboard au standard Json. Ceux-ci doivent être personnalisables et interactifs. Les buts premiers sont la surveillance et l’analyse et surtout une capacité de corriger en temps réel des problèmes et l’expérience aidant être proactif.**

**Usages principaux des dashboards dans Grafana**

**1. Supervision temps réel (real-time monitoring)**

* **Affichage de métriques en continu provenant de sources comme Prometheus, InfluxDB, Loki, Elasticsearch, etc.**
* **Exemple : suivi de l’utilisation CPU/RAM/disque d’un serveur.**

**2. Détection d’anomalies**

* **Mise en évidence de comportements anormaux via des seuils, des pics, ou des écarts.**
* **Exemples :**
  + **Un nombre élevé de requêtes 500 (erreurs HTTP).**
  + **Une latence anormalement élevée sur un service.**

**3. Alerting visuel + automatisé**

* **Couplage avec des alertes configurées (via Alertmanager, mail, Slack, etc.).**
* **Les dashboards permettent de visualiser les alertes déclenchées et leurs causes.**

**4. Diagnostic et analyse post-incident (forensic/debug)**

* **Permet de rejouer un incident dans le temps, via le sélecteur temporel.**
* **Filtrage par label, par hôte, ou par type d’erreur.**

**5. Rapports et communication**

* **Affichage synthétique pour les managers, RSSI, ou clients (en lecture seule).**
* **Possibilité d’export PDF, share URL, ou d’intégration dans un site via iframe.**

**6. Vue consolidée multi-sources**

* **Intégration de plusieurs sources de données dans un même dashboard (ex. : Prometheus + Loki + Blackbox).**
* **Corrélation des événements (ex. : logs + métriques système).**

**7. Pilotage des performances**

* **Suivi des indicateurs clés de performance (KPI) métiers ou techniques.**
* **Exemple : taux de disponibilité (uptime), temps de réponse moyen, nombre d’utilisateurs actifs.**

**8. Suivi de conformité et d’audit**

* **Vérification de l’état de sécurité et de la conformité des systèmes (ISO 27001, RGPD, NIS2…).**
* **Exemple : intégration avec Wazuh, Suricata, ou pfSense Exporter.**

**9. Visualisation de scénarios métier**

* **Dashboards orientés usage spécifique (ex. : flux réseau Jitsi, montée en charge d’une API, monitoring d’un cluster Kubernetes…).**
* **Utilisés pour tester et valider des stratégies de montée en charge ou de résilience.**

**10. Maintenance préventive**

* **Identification des machines ou services qui risquent de tomber en panne (RAM saturée, partition disque à 95 %, etc.).**
* **Utilisation dans les plans de maintenance évolutive (ITIL / ISO 22301).**

**🔧 Exemples de widgets / panels dans un dashboard :**

* **Graphes temporels (CPU, charge, requêtes)**
* **Barres ou jauges (utilisation mémoire)**
* **Statistiques tabulaires (état des services)**
* **Logs filtrables (via Loki)**
* **Heatmaps, alert lists, pie charts, singlestats, etc.**

**Infomaniak ; la gestion des groupes de sécurité pour les ports :**

Voici le **tableau récapitulatif des ports essentiels** à ouvrir par service pour ta plateforme Jitsi + Monitoring (Prometheus, Grafana, Blackbox, pfSense Exporter).

**✅ Tableau des ports par service**

| **Service** | **Port** | **Protocole** | **Type** | **Description** |
| --- | --- | --- | --- | --- |
| **SSH (admin)** | **22** | **TCP** | **Ingress** | **Connexion au serveur** |
| **HTTP** | **80** | **TCP** | **Ingress** | **Web non chiffré (utile pour redirection)** |
| **HTTPS** | **443** | **TCP** | **Ingress** | **Accès web sécurisé (ex. visio.workeezconnect)** |
| **Grafana** | **3000** | **TCP** | **Ingress** | **Interface Web Grafana** |
| **Jitsi Media** | **10000** | **UDP** | **Ingress** | **Canal de flux audio/vidéo UDP (WebRTC)** |
| **Prometheus UI** | **9090 (ou 9091)** | **TCP** | **Ingress** | **Interface Prometheus** |
| **Node Exporter** | **9100** | **TCP** | **Ingress** | **Export métriques serveur Linux** |
| **Blackbox Exporter** | **9115** | **TCP** | **Ingress** | **Tests ICMP/HTTP externes** |
| **iptables Exporter** | **9145** | **TCP** | **Ingress** | **Export règles netfilter/iptables** |

**📦 Groupes créés :**

| **Nom du groupe** | **Ports inclus** | **Utilisation** |
| --- | --- | --- |
| **default (désactivable)** | **Aucun si retiré** | **Peut être exclu** |
| **monitoring-group** | **9091, 9100, 9115, 9145** | **Prometheus + Exporters** |
| **monitoring-security-group** | **22, 80, 443** | **Accès standard SSH + HTTP/HTTPS** |
| **visio-monitoring-group** | **3000, 10000** | **Grafana + Jitsi UDP (media)** |

Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.

**Ce Security Group intègre l’ensemble des ports dédiés aux outils de Monitoring.**

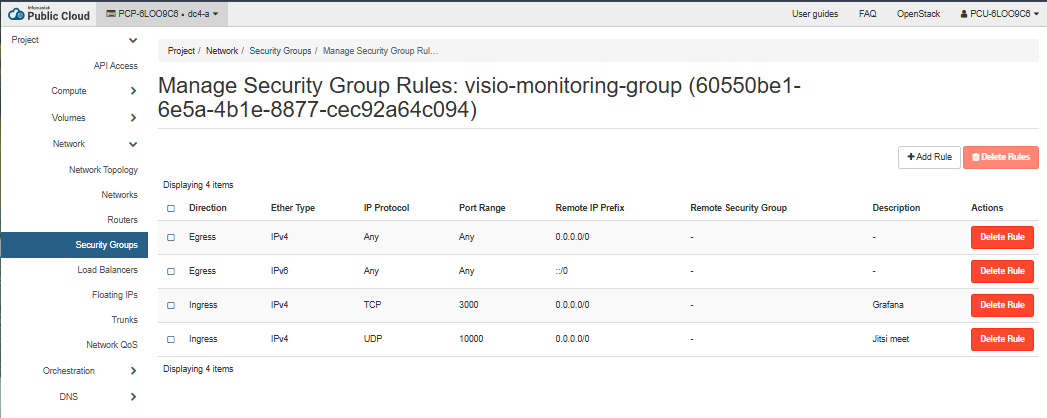
Une image contenant texte, capture d’écran, nombre, logiciel

Le contenu généré par l’IA peut être incorrect.

**Ce Security Group intègre l’ensemble des ports HTTP / HTTPS / SSH.**

**🛑 Attention**

* Le port **10000/UDP** est critique pour que la visioconférence fonctionne.
* **IPv6** peut être ignoré sauf cas spécifique (pas utile pour WebRTC sauf config avancée).
* L’on peut désormais **supprimer "default"** une fois **tous les groupes nécessaires rattachés**.



**Ce dernier Security Group intègre les ports pour Grafana & Jitsi meet.**

**Automatiser avec Terraform pour instancier Infomaniak : Possible**

**Installer Terraform puis :**

**1/ Contenu du projet**

| **Fichier** | **Rôle** |
| --- | --- |
| **clouds.yaml** | **Configuration OpenStack pour Infomaniak** |
| **main.tf** | **Création de l’instance jitsi-tercium avec le bon security\_group** |
| **security.tf** | **Règles de sécurité pour Grafana (TCP 3000) et Jitsi (UDP 10000)** |
| **variables.tf** | **Variable pour le nom de la clé SSH** |
| **.gitignore** | **Ignore les fichiers .tfstate, .terraform/ et credentials sensibles** |

**2/ Les étapes :**

1. **Configurer clouds.yaml localement dans : ~/.config/openstack/.**
2. **Renseigner les vraies valeurs dans clouds.yaml :**
   * **username, password, project\_id, etc.**
3. **Initialiser le projet avec ces commandes :**

**cd : terraform init**

**terraform plan**

**terraform apply**

**Structure recommandée du projet**

**project/**

**├── main.tf**

**├── variables.tf**

**├── terraform.tfvars.example**

**├── terraform.sh**

**3/ Exemple de fichier : Nécessaires à Infomaniak**

**Terraform.tfvars**

**# Clé SSH utilisée pour accéder à l'instance**

**keypair\_name = "tercium-instance\_key"**

**# Nom de l'image (ID ou nom exact d’Infomaniak OpenStack)**

**image\_name = "Debian 12 Bookworm"**

**# Nom du flavor (ex: gp1.large, cp1.medium, etc.)**

**flavor\_name = "gp1.large"**

**# Nom du réseau privé ou réseau par défaut d’Infomaniak**

**network\_name = "Ext-Net1"**

**# Groupe de sécurité à appliquer**

**security\_group\_name = "jitsi-monitoring-secgroup"**

**# Nom d’instance (machine virtuelle)**

**instance\_name = "jitsi-tercium"**

**Modèles de fichier :**

**1/ main.tf**

**provider "openstack" {**

**auth\_url = var.auth\_url**

**tenant\_name = var.project\_name**

**user\_name = var.username**

**password = var.password**

**region = var.region**

**domain\_name = var.domain\_name**

**}**

**# Groupe de sécurité personnalisé**

**resource "openstack\_networking\_secgroup\_v2" "jitsi\_secgroup" {**

**name = "jitsi\_monitoring\_secgroup"**

**description = "Allow SSH, HTTP, HTTPS, Prometheus, Grafana, Jitsi, Node Exporter, Blackbox"**

**}**

**# Règles de sécurité essentielles**

**resource "openstack\_networking\_secgroup\_rule\_v2" "rules" {**

**count = length(var.security\_rules)**

**direction = var.security\_rules[count.index]["direction"]**

**ethertype = var.security\_rules[count.index]["ethertype"]**

**protocol = var.security\_rules[count.index]["protocol"]**

**port\_range\_min = var.security\_rules[count.index]["port\_range\_min"]**

**port\_range\_max = var.security\_rules[count.index]["port\_range\_max"]**

**remote\_ip\_prefix = var.security\_rules[count.index]["remote\_ip\_prefix"]**

**security\_group\_id = openstack\_networking\_secgroup\_v2.jitsi\_secgroup.id**

**}**

**# Clé SSH**

**resource "openstack\_compute\_keypair\_v2" "default" {**

**name = "tercium-key"**

**public\_key = file(var.public\_key\_path)**

**}**

**# Instance Jitsi**

**resource "openstack\_compute\_instance\_v2" "jitsi\_vm" {**

**name = "jitsi-tercium"**

**image\_name = var.image\_name**

**flavor\_name = var.flavor\_name**

**key\_pair = openstack\_compute\_keypair\_v2.default.name**

**security\_groups = [openstack\_networking\_secgroup\_v2.jitsi\_secgroup.name]**

**network {**

**name = var.network\_name**

**}**

**}**

**2/ variables.tf**

**variable "auth\_url" {}**

**variable "username" {}**

**variable "password" {}**

**variable "domain\_name" {}**

**variable "project\_name" {}**

**variable "region" {}**

**variable "public\_key\_path" {}**

**variable "image\_name" {}**

**variable "flavor\_name" {}**

**variable "network\_name" {}**

**variable "security\_rules" {**

**type = list(object({**

**direction = string**

**ethertype = string**

**protocol = string**

**port\_range\_min = number**

**port\_range\_max = number**

**remote\_ip\_prefix = string**

**}))**

**}**

**3/ terraform.tfvars**

**auth\_url = "https://api.infomaniak.com/identity/v3"**

**username = "TON\_IDENTIFIANT"**

**password = "TON\_MOT\_DE\_PASSE"**

**domain\_name = "default"**

**project\_name = "NomDuProjet"**

**region = "dc3-a"**

**public\_key\_path = "~/.ssh/id\_rsa.pub"**

**image\_name = "Ubuntu 22.04"**

**flavor\_name = "a1-ram2-disk20-perf1"**

**network\_name = "Ext-Net"**

**security\_rules = [**

**{**

**direction = "ingress"**

**ethertype = "IPv4"**

**protocol = "tcp"**

**port\_range\_min = 22**

**port\_range\_max = 22**

**remote\_ip\_prefix = "0.0.0.0/0"**

**},**

**{**

**direction = "ingress"**

**ethertype = "IPv4"**

**protocol = "tcp"**

**port\_range\_min = 80**

**port\_range\_max = 80**

**remote\_ip\_prefix = "0.0.0.0/0"**

**},**

**{**

**direction = "ingress"**

**ethertype = "IPv4"**

**protocol = "tcp"**

**port\_range\_min = 443**

**port\_range\_max = 443**

**remote\_ip\_prefix = "0.0.0.0/0"**

**},**

**{**

**direction = "ingress"**

**ethertype = "IPv4"**

**protocol = "tcp"**

**port\_range\_min = 3000**

**port\_range\_max = 3000**

**remote\_ip\_prefix = "0.0.0.0/0"**

**},**

**{**

**direction = "ingress"**

**ethertype = "IPv4"**

**protocol = "tcp"**

**port\_range\_min = 10000**

**port\_range\_max = 10000**

**remote\_ip\_prefix = "0.0.0.0/0"**

**}**

**]**

**Jour-10 Tercium 2nde semaine | monitoring routines de démarrage**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Connexion SSH en Powershell :**

**cd : ssh -i "C:\Users\test\Documents\Tercium\_Stage\ssh\_keys\tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Relancer les services système via le terminal :**

**cd : sudo systemctl restart prometheus**

**sudo systemctl restart node\_exporter**

**sudo systemctl restart blackbox\_exporter**

**Pour vérifier qu’ils tournent :**

**cd : sudo systemctl status prometheus**

**sudo systemctl status node\_exporter**

**sudo systemctl status blackbox\_exporter**

**Bonnes pratiques :**

* **Prometheus doit être actif pour collecter les métriques.**
* **Node Exporter doit tourner sur chaque machine à surveiller.**
* **Blackbox Exporter doit être lancé une fois (souvent sur la même machine que Prometheus) pour faire les tests de ping, HTTP, TCP, etc.**
* **Grafana se connecte à Prometheus. Il n’a pas besoin d’être redémarré sauf en cas de changement de configuration du data source ou de plantage.**

**Astuce : automatisation au démarrage :**

**cd : sudo systemctl enable prometheus**

**sudo systemctl enable node\_exporter**

**sudo systemctl enable blackbox\_exporter**

**Accès aux interfaces Web :**

**Si tous les services sont actifs, l’on peut accéder aux interfaces suivantes :**

| **Composant** | **URL locale** | **URL distante (publique)** |  |
| --- | --- | --- | --- |
| **Prometheus** | **http://localhost:9090** | **http://37.156.46.238:9090 *(si exposé)*** |  |
| **Node Exporter** | **http://localhost:9100** | **http://37.156.46.238:9100 *(si exposé)*** |  |
| **Blackbox Exporter** | **http://localhost:9115** | **http://37.156.46.238:9115 *(si exposé)*** |  |
| **Grafana** | **http://localhost:3000** | [**http://37.156.46.238:3000**](http://37.156.46.238:3000) **✅** |  |

**TESTS :**

**Vérifie que ton service est bien bindé sur ce port :  
Exemple avec NGINX : Interdit ici**

**cd : sudo netstat -tulnp | grep 4433**

**ou**

**cd : sudo ss -tuln | grep 4433**

**Test à distance depuis une autre machine :**

* **règle de composition nc -vz <IP> <PORT>**

**cd : nc -vz 37.156.46.238 4433**

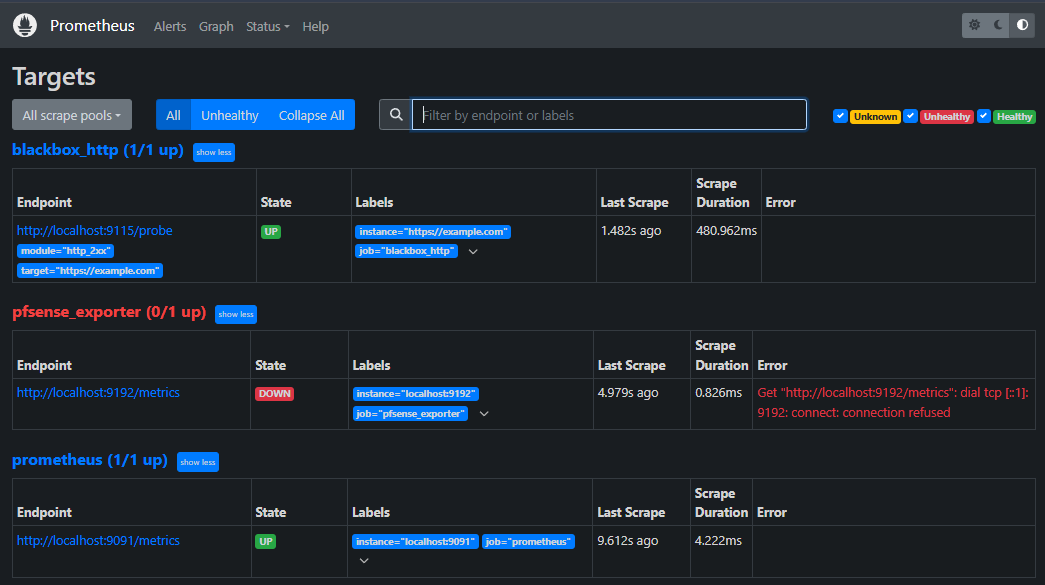
| **Sortie nc -vz** | **Signification technique** |
| --- | --- |
| **Connection to 37.156.46.238 port 4433 succeeded!** | **🔓 Le port est ouvert et une application y écoute** |
| **Connection refused** | **🚫 Aucun service n’écoute sur ce port** |
| **Connection timed out** | **❌ Le port est filtré (firewall ou IP injoignable)** |

**ou via navigateur :**

**https://37.156.46.238:4433**

**Manuel d’usage de visio.workeezconnect.fr**

**Récupération et analyse des données monitoring :**

****

**Problème pfsense\_exporter est inactif : Pourquoi ces commandes ne fonctionnent pas ?**

**cd : sudo systemctl status pfsense\_exporter**

**sudo systemctl restart pfsense\_exporter**

**Retour ce service de monitoring du 8 ième jour : Je compléte et je modifie.**

**Jour-11 Tercium 3ième semaine | fin installation monitoring**

**Voir plus haut pour les relances qu’il faudra totalement automatiser.**

**Résumé des fonctions logs via les services avant relance d’installation de telegraf :**

**Telegraf**, comparer à **Node Exporter**, **Blackbox Exporter**, et à d’autres (type **Filebeat**, **Logstash**, etc.).

**1. Rôle de Telegraf**

Telegraf est un **agent léger de collecte de métriques et de logs**. Il fait partie de la suite TICK (Telegraf, InfluxDB, Chronograf, Kapacitor), mais peut fonctionner **indépendamment** et exporter vers Prometheus ou Grafana.

**Il collecte, transforme et envoie des données.**

* **Collecte** → via des *plugins d’entrée* (inputs)
* **Traitement / mise en forme** → *plugins de transformation*
* **Envoi** → via des *plugins de sortie* (outputs)

**2. Comparatif simplifié des outils de monitoring**

| **Outil** | **Fonction principale** | **Type de données** | **Spécificité** |
| --- | --- | --- | --- |
| **Telegraf** | Agent polyvalent de collecte | Métriques système, réseau, SNMP, Docker, logs, etc. | Très personnalisable (plugins) |
| **Node Exporter** | Exporter Prometheus pour Linux | Métriques système Linux | Simplicité, usage standard Prometheus |
| **Blackbox Exporter** | Vérification de l’accessibilité (ping, HTTP, TCP) | Résultats de tests "extérieurs" (uptime) | Surveillance type "boîte noire" |
| **Filebeat** | Collecte de fichiers de log (ELK/EFK) | Logs (texte brut, JSON, syslog, nginx…) | Léger, rapide, dédié aux logs |
| **Logstash** | Pipeline de traitement de logs complexe | Logs enrichis | Peut transformer, parser, router |
| **Fluentd** | Agent/log router JSON & cloud-friendly | Logs/streams structurés | Flexible et orienté cloud |

**3. Telegraf : Architecture des plugins**

**🔹 Exemples de *plugins d'entrée* :**

[[inputs.cpu]]

percpu = true

totalcpu = true

[[inputs.mem]]

[[inputs.net]]

[[inputs.disk]]

[[inputs.system]]

[[inputs.snmp]]

agents = [ "192.168.1.1:161" ]

version = 2

**🔹 Plugin de sortie (*Prometheus* ou *InfluxDB*) :**

[[outputs.prometheus\_client]]

listen = ":9273"

# ou :

[[outputs.influxdb]]

urls = ["http://localhost:8086"]

**Telegraf peut aussi lire des logs si nécessaire**

Exemple :

[[inputs.tail]]

files = ["/var/log/syslog"]

from\_beginning = false

name\_override = "syslog\_logs"

**En résumé**

| **Agent** | **Pour Prometheus ?** | **Pour logs ?** | **Configuration** |
| --- | --- | --- | --- |
| **Node Exporter** | **Oui** | **Non** | Aucune/modeste |
| **Telegraf** | **Oui/Non (pas exclusivement)** | **Oui** | Config très riche via fichier .conf |
| **Blackbox** | **Oui** | **Non** | prometheus.yml uniquement |

**✅ Solution directe et stable de Telegraf :**

**1. Sauvegarder la configuration existante : cd : sudo mv /etc/telegraf/telegraf.conf /etc/telegraf/telegraf.conf.bak**

**2. Créer un dossier de configuration Telegraf : cd : sudo mkdir -p /etc/telegraf**

**Configurer Telegraf :**

**cd : sudo nano /etc/telegraf/telegraf.conf**

**[agent]**

**interval = "10s"**

**round\_interval = true**

**metric\_batch\_size = 1000**

**metric\_buffer\_limit = 10000**

**collection\_jitter = "0s"**

**flush\_interval = "10s"**

**flush\_jitter = "0s"**

**precision = ""**

**hostname = ""**

**omit\_hostname = false**

**[[outputs.prometheus\_client]]**

**listen = ":9273"**

**path = "/metrics"**

**expiration\_interval = "60s"**

**collectors\_exclude = ["gocollector", "process"]**

**[[inputs.cpu]]**

**percpu = true**

**totalcpu = true**

**collect\_cpu\_time = false**

**report\_active = false**

**[[inputs.mem]]**

**[[inputs.disk]]**

**ignore\_fs = ["tmpfs", "devtmpfs", "overlay"]**

**[[inputs.net]]**

**[[inputs.system]]**

**Redémarrer Telegraf & vérifier son status:**

**cd : sudo systemctl restart telegraf**

**sudo systemctl status telegraf**

**Tester l’export local :**

**cd : curl http://localhost:9273/metrics**

**Vérifions le fichier de configuration de Prometheus, pas son répertoire mais le fichier: cd: sudo nano /etc/prometheus/prometheus.yml**

**Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Puis installer le dashboard via l’import de Grafana et tester:**

**Vérifier que Telegraf écoute bien sur un port Prometheus-compatible : cd : sudo lsof -i -P -n | grep telegraf**

**Une image contenant texte, capture d’écran, Police, ligne

Le contenu généré par l’IA peut être incorrect.**

* **Objectif : lister les fichiers ouverts par des processus (lsof) en filtrant ceux liés à des connexions réseau (-i) sur des ports (avec ports affichés en clair grâce à -P) sans résolution DNS (-n).**
* **Usage : Diagnostique fin, montre quel *processus exact* est à l’écoute d’un port (ici telegraf avec son PID 25533 sur TCP \*:9273).**
* **Avantage : on voit que c’est bien le binaire telegraf qui a ouvert le port.**

**Vérifier que Prometheus Scrape bien Telegraf via m’IP publique :**

**cd :** [**http://37.156.46.238:9091/targets**](http://37.156.46.238:9091/targets) **: NON**

**Il doit avoir cet ajout dans son code :du fichier : prometheus.yml**

**- job\_name: 'telegraf'**

**static\_configs:**

**- targets: ['localhost:9273'] Puis redémarrer**

**Il faut ajouter une règle au Security Group :**

**Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.**

**Redémarrer Telegraf :**

**cd : sudo systemctl restart telegraf**

**sudo systemctl status telegraf**

**Vérifier que le port 9273 est exposé :**

**cd : ss -tuln | grep 9273**

**Une image contenant texte, capture d’écran, Police, ligne

Le contenu généré par l’IA peut être incorrect.**

**Nuance à comprendre :**

| **Commande** | **Vérifie port ouvert ?** | **Affiche PID / processus ?** | **Utilité** |
| --- | --- | --- | --- |
| **`lsof -i -P -n** | **grep telegraf`** | **✅ Oui** | **✅ Oui** |
| **`ss -tuln** | **grep 9273`** | **✅ Oui** | **❌ Non** |

**Le -p permet d’afficher aussi le nom et le PID du processus (comme lsof).**

**Test Grafana ave une session visio.workeez.connect.fr entre deux PC.**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Pour pouvoir utiliser pfsense indirectement il faut rajouter à Telegraf à la fin cette extension. Attention besoin possible de MIBs !**

****

**Puis relance Telegraf :**

**L’option pfsense peut revenir en incluant non pas l’installation via github dépôt inopérant ni une version python elle aussi inerte mais via des inputs dans telegraf.conf.**

**Bloc [inputs.snmp] — interrogation de pfSense via SNMP :**

**Une image contenant texte, capture d’écran, conception

Le contenu généré par l’IA peut être incorrect.**

**Redémarrage après modification :**

**cd : sudo systemctl daemon-reexec**

**sudo systemctl restart telegraf**

**Fonction de systemctl daemon-reexec**

**Cette commande : cd**

**Recharge complètement le binaire systemd (le gestionnaire d’unités et services), sans redémarrer le système.**

**Contexte technique :**

* **Quand tu modifies des fichiers liés à systemd (unités .service, chemins, permissions système, socket activation...),**
* **systemd peut être encore en train d'utiliser une ancienne version de lui-même en mémoire,**
* **daemon-reexec demande explicitement à systemd de se recharger entièrement à partir du disque, y compris les binaires et les données de configuration.**

**À distinguer de :**

| **Commande** | **Fonction** |
| --- | --- |
| **sudo systemctl daemon-reexec** | **Recharge le processus systemd lui-même (rarement nécessaire, sauf modif profonde)** |
| **sudo systemctl daemon-reload** | **Recharge les fichiers d’unités après modification (plus courant)** |
| **sudo systemctl restart <service>** | **Redémarre un service spécifique** |

**Jour-12 Tercium 3ième semaine | installation monitoring recap**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Connexion SSH en Powershell :**

**cd : ssh -i "C:\Users\test\Documents\Tercium\_Stage\ssh\_keys\tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Commandes à automatiser via un Script :**

**Relancer les services système via le terminal :**

**cd : sudo systemctl restart prometheus**

**sudo systemctl restart node\_exporter**

**sudo systemctl restart blackbox\_exporter**

**sudo systemctl restart telegraf**

**Pour vérifier qu’ils tournent :**

**cd : sudo systemctl status prometheus**

**sudo systemctl status node\_exporter**

**sudo systemctl status blackbox\_exporter**

**sudo systemctl status telegraf**

**Bonnes pratiques :**

* **Prometheus doit être actif pour collecter les métriques.**
* **Node Exporter doit tourner sur chaque machine à surveiller.**
* **Blackbox Exporter doit être lancé une fois (souvent sur la même machine que Prometheus) pour faire les tests de ping, HTTP, TCP, etc.**
* **Grafana se connecte à Prometheus. Il n’a pas besoin d’être redémarré sauf en cas de changement de configuration du data source ou de plantage.**

**Astuce : automatisation au démarrage :**

**cd : sudo systemctl enable prometheus**

**sudo systemctl enable node\_exporter**

**sudo systemctl enable blackbox\_exporter**

**sudo systemctl enable telegraf**

**Accès aux interfaces Web :**

**Si tous les services sont actifs, l’on peut accéder aux interfaces suivantes :**

| **Composant** | **URL locale** | **URL distante (publique)** |  |
| --- | --- | --- | --- |
| **Prometheus** | **http://localhost:9090** | **http://37.156.46.238:9091 *(si exposé)*** |  |
| **Node Exporter** | **http://localhost:9100** | **http://37.156.46.238:9100 *(si exposé)*** |  |
| **Blackbox Exporter** | **http://localhost:9115** | **http://37.156.46.238:9115 *(si exposé)*** |  |
| **Grafana** | **http://localhost:3000** | [**http://37.156.46.238:3000**](http://37.156.46.238:3000) **✅** |  |

**Vérifier la persistance après redémarrage :**

**cd : sudo systemctl list-unit-files | grep enabled**

**Puis**

**cd : sudo systemctl status <service>**

**Journal permanent :**

**cd : journalctl -u telegraf -n 50 --no-pager**

**Les ports sont bien ouverts, vérifier :**

**cd : sudo ss -tuln | grep -E '9273|9091|9115|9100|3000|4443'**

**Si MIBs correctement chargées :**

* **inputs.snmp peut s’initialiser → pas d’erreur OID unknown.**
* **Telegraf démarre sans crash.**
* **Grafana retrouve automatiquement les métriques SNMP de pfSense.**
* **Donc :**
  + **pfSense s'affiche en *UP* dans le dashboard (si reachable).**
  + **Telegraf = service active (running).**
  + **Port :9273 expose bien les métriques pour Prometheus.**

**Si MIBs absentes ou mal résolues :**

* **inputs.snmp → échec initialisation (Unknown Object Identifier).**
* **Telegraf → crash (exit status 1).**
* **Donc :**
  + **Grafana = pfSense en rouge (aucune métrique visible).**
  + **Telegraf = service failed.**
  + **Port :9273 = *fermé ou vide*.**

**Pour réparer**

1. **Installe les MIBs nécessaires :**

**cd : sudo apt install snmp-mibs-downloader**

**sudo download-mibs**

1. **Activer la lecture des noms de MIBs (désactivée par défaut dans snmp.conf) :**

**cd : sudo nano /etc/snmp/snmp.conf**

**# mibs : on le commente**

1. **Redémarrer Telegraf :**

**cd : sudo systemctl restart telegraf**

**Récupitulatif :**

**Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.**

**Explications :**

**Une image contenant texte, lettre, menu, papier

Le contenu généré par l’IA peut être incorrect.**

**Résultat visible sur Prometheus :**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Donc pfsense ne répond pas voir note en annexe.**

**Cependant : choix des sorties = up :**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Diagramme et résultat :**

**Une image contenant capture d’écran, texte, Logiciel multimédia, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Un rapport d’incident a été produit qui alimentera cette partie.**

**Objectif à court terme : load balancing fonctionnel demain**

**Je dois disposer d’un service Prometheus propre et stable pour pouvoir monitorer :**

* **CPU / RAM / Réseau via node\_exporter et telegraf**
* **Résultat de tests blackbox (HTTP/HTTPS)**
* **Une cible Jitsi visio active**

**Décision : Killprocessus / relance de l’instance via un shut-off puis start et relance et vérification des processus de monitoring :**

**État actuel :**

* **Prometheus tournait manuellement → OK.**
* **En mode service (systemd) → problèmes de config, de lecture, ou de port déjà pris.**
* **node\_exporter tourne, mais invisible.**
* **Tu es sur une IP publique (pas de localhost).**
* **Risque de confusion si plusieurs processus tournent (service actif + process manuel).**

**RECOMMANDATION IMMÉDIATE : stopper l’instance**

**Pourquoi ?**

* **Un port 9091 propre.**
* **Redémarrage net des binaires Prometheus et exporters.**
* **Eviter les interférences invisibles ou conflits de socket réseau.**
* **Plus sûr que de forcer des kill -9.**

**Plan de redémarrage clair :**

1. **On reprend :**
   * **Redémarrer l’instance manuellement ou depuis l’interface cloud. ok**
   * **Lancer ce check minimal :**
     + **sudo systemctl status prometheus**
     + **sudo systemctl status node\_exporter**
     + **sudo systemctl status telegraf**
2. **Vérifier les cibles :**

**cd : curl http://localhost:9091/api/v1/targets | jq '.data.activeTargets[] | {job: .labels.job, instance, health, lastError}'**

1. **Puis ouvrir Grafana :**
   * **Filtre par job="node\_exporter" et up == 1**

**Je suis passé en tant qu’utilisateur ubuntu pour les raisons pratiques suivantes, tout à fait cohérentes avec le contexte de déploiement :**

**1. Clé SSH par défaut liée à ubuntu**

* Sur **les instances Ubuntu Cloud (Infomaniak, GCP, etc.)**, l’utilisateur ubuntu est :
  + l’**utilisateur par défaut autorisé par la clé SSH** injectée au moment de l’instanciation.
  + Préconfiguré avec les bons droits sudo (NOPASSWD:ALL), pour éviter de créer un compte custom.

**2. Je n’ai pas défini de service systemd complet pour Prometheus au nom de l'utilisateur prometheus**

* Au lieu de créer :
  + un dossier /home/prometheus
  + un ownership explicite (chown prometheus:prometheus)
  + une configuration système sécurisée avec user dédié…

J’ai as lancé tes tests directement en ubuntu pour **gagner du temps et maîtriser les fichiers sans contrainte de droits**.

**3. Simplification des tests et debug Prometheus**

* Lorsque tu lances :
* /opt/prometheus/prometheus --config.file=...

en tant qu’ubuntu, j’évites :

* + des erreurs de permission (comme celles de queries.active)
  + des conflits avec systemd s’il n’est pas encore bien configuré

**Je peux revenir à prometheus plus tard pour avoir un service durci**

Mais pour l’instant, travailler en tant que ubuntu permet :

* D'**éviter les frictions** lors du test et de la supervision initiale (node\_exporter, telegraf, blackbox)
* De **concentrer l’effort sur la validation fonctionnelle du monitoring et du load balancing**, ce qui est ta priorité demain

**➕ En résumé :**

| **Raison** | **Avantage** |
| --- | --- |
| **Clé SSH liée à ubuntu** | **Connexion immédiate sans créer d'autre compte** |
| **Droits sudo** | **Administration sans mot de passe** |
| **Éviter les permission denied** | **Écriture directe sur /home/ubuntu/, /opt/, etc.** |
| **Déploiement rapide de tests** | **Pas besoin de configurer d'utilisateur système** |

**Echec prometheus mais node\_exporter / blackbox\_exporter / telegraf sont lancés.**

**TEST :**

**Vérifier le statut exact de l’échec :**

**cd : sudo systemctl status prometheus -l**

**Tester manuellement Prometheus en mode foreground :**

**cd : /opt/prometheus/prometheus --config.file=/etc/prometheus/prometheus.yml --web.listen-address="0.0.0.0:9091"**

**PRÉPARATION DES FICHIERS**

**Copier les scripts dans les bons emplacements cd ~/ ou l’endroit où tu as extrait le ZIP**

**Copier les scripts**

**cd : sudo cp start\_prometheus.sh /usr/local/bin/start\_prometheus.sh**

**sudo cp toggle\_prometheus\_mode.sh /usr/local/bin/toggle\_prometheus\_mode.sh**

**Rendre exécutables**

**cd : sudo chmod +x /usr/local/bin/start\_prometheus.sh cd : sudo chmod +x /usr/local/bin/toggle\_prometheus\_mode.sh**

**1. Placer le service systemd cd : sudo cp prometheus.service /etc/systemd/system/prometheus.service**

**2. RECHARGER LES SERVICES SYSTEMD cd : sudo systemctl daemon-reload**

**3. DÉMARRER PROMETHEUS EN MODE PROPRE cd : sudo systemctl start prometheus**

**Vérification : cd : sudo systemctl status prometheus -l**

**4. UTILISATION DU SCRIPT DE BASCULE : On peut basculer en mode administrateur (avec endpoint /-/reload, etc.) :**

**cd : sudo toggle\_prometheus\_mode.sh true # Mode admin activé cd : sudo toggle\_prometheus\_mode.sh false # Mode sécurisé**

**Chaque changement redémarre Prometheus automatiquement.**

**5. SÉCURISATION FACULTATIVE (si nécessaire)**

**Ajoute une règle UFW *temporaire* pour autoriser 9091 depuis ton IP uniquement : cd : sudo ufw allow from <ton\_ip\_publique> to any port 9091 proto tc**

**Contenu et usage :**

1. **start\_prometheus.sh**
   * 📍 Emplacement recommandé : /usr/local/bin/start\_prometheus.sh
   * 🎯 Lance Prometheus avec ou sans l’API d’administration (--web.enable-admin-api=false) selon le mode défini dans PROM\_SECURE.
   * 🔐 Par défaut, le mode sécurisé est activé (true).
2. **prometheus.service**
   * 📍 À copier dans : /etc/systemd/system/prometheus.service
   * 🔁 Appelle le script de démarrage au lieu de lancer directement Prometheus.
   * 💡 Nécessite sudo systemctl daemon-reload après modification.
3. **toggle\_prometheus\_mode.sh**
   * 📍 À copier dans /usr/local/bin également (et rendre exécutable).
   * 🔁 Permet de **basculer dynamiquement entre mode sécurisé et mode ouvert**, avec redémarrage automatique de Prometheus.

**Manuel d’usage pour administrateurs :**

| **Commande** | **Effet** |
| --- | --- |
| **sudo systemctl restart prometheus** | **Redémarre Prometheus avec la config actuelle** |
| **/usr/local/bin/toggle\_prometheus\_mode.sh** | **Bascule entre mode sécurisé (admin API désactivée) et mode administrateur (admin API active)** |
| **sudo systemctl status prometheus** | **Vérifie le statut du service** |
| **curl http://localhost:9091** | **Teste l’interface Prometheus (port public configurable)** |

**Jour-13 Tercium 3ième semaine | monitoring | Tests interfaces**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Connexion SSH en Powershell :**

**cd : ssh -i "C:\Users\test\Documents\Tercium\_Stage\ssh\_keys\tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Commandes à automatiser via un Script :**

**Relancer les services système via le terminal :**

**cd : sudo systemctl restart prometheus**

**sudo systemctl restart node\_exporter**

**sudo systemctl restart blackbox\_exporter**

**sudo systemctl restart telegraf**

**Pour vérifier qu’ils tournent :**

**cd : sudo systemctl status prometheus**

**sudo systemctl status node\_exporter**

**sudo systemctl status blackbox\_exporter**

**sudo systemctl status telegraf**

**Accès aux interfaces Web :**

**Si tous les services sont actifs, l’on peut accéder aux interfaces suivantes :**

| **Composant** | **URL locale** | **URL distante (publique)** |  |
| --- | --- | --- | --- |
| **Prometheus** | **http://localhost:9090** | **http://37.156.46.238:9091 *(si exposé)*** |  |
| **Node Exporter** | **http://localhost:9100** | **http://37.156.46.238:9100 *(si exposé)*** |  |
| **Blackbox Exporter** | **http://localhost:9115** | **http://37.156.46.238:9115 *(si exposé)*** |  |
| **Grafana** | **http://localhost:3000** | [**http://37.156.46.238:3000**](http://37.156.46.238:3000) **✅** |  |

**État actuel : erreur exit-code au démarrage de Prometheus sachant que les autres services fonctionnent.**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Constat :** Deux structures contradictoires :

* **Binaire de Prometheus** dans /opt/prometheus/prometheus
* **Fichier de configuration (prometheus.yml)** dans /etc/prometheus/

→ **Mais :** Le prometheus.service tel qu’écrit appelle peut-être une config corrompue, un path cassé, ou ne trouve pas les permissions correctes.

**Inspectons le fichier actuel : cd : sudo systemctl cat prometheus**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Rappel :**

**Recommandation de structure normalisée**

| **Élément** | **Emplacement recommandé** |
| --- | --- |
| **Binaire prometheus** | **/usr/local/bin/prometheus ou /opt/...** |
| **Fichier de config .yml** | **/etc/prometheus/prometheus.yml** |
| **Data dir (TSDB)** | **/var/lib/prometheus/** |
| **Fichier service** | **/etc/systemd/system/prometheus.service** |

**ÉTAT ACTUEL : RÉSUMÉ**

| **Élément** | **État** |
| --- | --- |
| **telegraf** | **✅ écoute sur \*:9273 (PID confirmé, ss OK)** |
| **curl vers localhost:9273/metrics** | **✅ affiche bien les métriques CPU** |
| **prometheus.yml** | **✅ contient localhost:9273 en scrape\_configs** |
| **prometheus lancé manuellement en CLI (pas via systemd)** | **✅ démarre, mais pas encore de métriques confirmées via interface** |
| **conflit possible user: root (CLI) vs ubuntu (telegraf/service)** | **❗** |
| **conflit de structure /opt/prometheus (manuel) vs /etc/prometheus/ (conventionnel)** | **❗** |

**Fichier /etc/prometheus/prometheus.yml :**

**Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

**Vérification par écoute du port dédié 9091, puis le relancer et un curl http en local il ne peut y avoir des données visibles = échec non !**

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Résumé des choix techniques**

| **Élément** | **But** |
| --- | --- |
| **job\_name séparés** | **Permet d’identifier clairement chaque source** |
| **instance:** | **Facilite les dashboards Grafana** |
| **scrape\_interval** | **Réduit à 5s pour une meilleure réactivité** |
| **external\_labels** | **Identification du cluster / instance** |
| **metrics\_path explicite** | **Pour compatibilité si chemins changés** |

**Il est préférable de déplacer ou réinstaller Blackbox Exporter dans une arborescence système standardisée**, notamment pour :

**Cohérence avec les autres services**

* **/opt est généralement réservé aux logiciels tiers ou compilés manuellement.**
* **Les démons système sont plus logiquement placés dans :**
  + **binaire** : /usr/local/bin/ ou /usr/bin/
  + **config** : /etc/blackbox\_exporter/
  + **service** : /etc/systemd/system/blackbox\_exporter.service
  + **logs (si activés)** : /var/log/blackbox\_exporter/

**Procédure minimale de remise en ordre :**

**Créer les répertoires standard :**

**cd :** sudo mkdir -p /etc/blackbox\_exporter

sudo mkdir -p /usr/local/bin/

**Déplacer les fichiers :**

**cd :** sudo mv /opt/blackbox\_exporter/blackbox\_exporter /usr/local/bin/

sudo mv /opt/blackbox\_exporter/blackbox.yml /etc/blackbox\_exporter/blackbox.yml

**Adapter le service systemd** (/etc/systemd/system/blackbox\_exporter.service) : [Unit] Description=Blackbox Exporter

Wants=network-online.target

After=network-online.target

[Service]

User=ubuntu

ExecStart=/usr/local/bin/blackbox\_exporter --config.file=/etc/blackbox\_exporter/blackbox.yml

Restart=on-failure

[Install]

WantedBy=multi-user.target

**Recharger les services :**

**cd :** sudo systemctl daemon-reexec

sudo systemctl daemon-reload

sudo systemctl restart blackbox\_exporter

**Vérification :**

**cd :** curl -s http://localhost:9115/metrics | head -n 10 ou

**cd :** curl -s http://localhost:9091/api/v1/targets | jq **'.data.activeTargets[] | select(.labels.job=="blackbox\_exporter\_http")'**

**1. Déplacement du binaire et du fichier de configuration :**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**2. Service systemd à modifier :**

**cd : sudo nano /etc/systemd/system/blackbox\_exporter.service**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**3. Recharger, activer et démarrer le service : cd : sudo systemctl daemon-reexec**

**sudo systemctl daemon-reload**

**sudo systemctl enable blackbox\_exporter**

**sudo systemctl restart blackbox\_exporter**

4. **Vérification :**

**# Vérifier que ça écoute bien**

**cd : ss -tulpen | grep 9115 | 9091**

****

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**# Test d'export**

**cd : curl http://localhost:9115/metrics | head -n 20**

**Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Enrichissement des fonctions avec des modules personnalisés avec des timeouts, headers HTTP, ou des cibles spécifiques : ici deux exemples**

**Il y aura deux fichiers à modifier : /opt/blackbox\_exporter/blackbox.yml**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Puis /etc/prometheus/prometheus.yml**

**Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.**

**Et /etc/systemd/system/blackbox\_exporter.service (service systemd de Blackbox)**

**Une image contenant texte, capture d’écran, Police, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Commande standard pour tester le fichier prometheus.yml avec promtool pour valider la syntaxe complète :**

**cd : promtool check config /etc/prometheus/prometheus.yml**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Étapes obligatoires après modification du service :**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**# Toujours Redémarrer après chaque modification :**

**cd : sudo systemctl restart prometheus**

**Sur** [**http://localhost:9091/rules**](http://localhost:9091/rules)**:**

**État des composants de supervision :**

| **Composant** | **Port** | **Statut dans Prometheus** | | **Statut d’installation** |
| --- | --- | --- | --- | --- |
| **blackbox\_exporter** | **9115** | **✅ (HTTP, TCP, ICMP, SSH)** | | **✅ Fonctionnel** |
| **node\_exporter** | **9100** | **✅** | | **✅ Fonctionnel** |
| **prometheus\_local** | **9091** | **✅** | | **✅ Fonctionnel** |
| **prometheus\_distant** | **9091** | **✅** | | **✅ Fonctionnel** |
| **telegraf** | **9273\*** | **❌ Absent** | **⚠️ Installé mais non configuré correctement** | |

**Jour-14 Tercium 3ième semaine | monitoring | calibrage PKI**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Commandes à automatiser via un Script :**

**Relancer les services système via le terminal :**

**cd : sudo systemctl restart prometheus**

**sudo systemctl restart node\_exporter**

**sudo systemctl restart blackbox\_exporter**

**sudo systemctl restart telegraf**

**Pour vérifier qu’ils tournent :**

**cd : sudo systemctl status prometheus**

**sudo systemctl status node\_exporter**

**sudo systemctl status blackbox\_exporter**

**sudo systemctl status telegraf**

**Accès aux interfaces Web :**

**Si tous les services sont actifs, l’on peut accéder aux interfaces suivantes :**

| **Composant** | **URL locale** | **URL distante (publique)** |  |
| --- | --- | --- | --- |
| **Prometheus** | **http://localhost:9090** | **http://37.156.46.238:9091 *(si exposé)*** |  |
| **Node Exporter** | **http://localhost:9100** | **http://37.156.46.238:9100 *(si exposé)*** |  |
| **Blackbox Exporter** | **http://localhost:9115** | **http://37.156.46.238:9115 *(si exposé)*** |  |
| **Grafana** | **http://localhost:3000** | [**http://37.156.46.238:3000**](http://37.156.46.238:3000) **✅** |  |

**Erreur pour backbox\_exporter ; l’arborescence n’est pas la même.**

**Constats :**

* **Blackbox Exporter semble installé dans un chemin différent (par exemple /etc/blackbox au lieu de /etc/blackbox\_exporter ou /opt/blackbox\_exporter).**
* **Le fichier blackbox.yml est parfois introuvable dans le bon répertoire (ton test sur /etc/blackbox/balckbox.yml échoue car le chemin est erroné *et* le nom mal orthographié : balckbox.yml).**
* **Les chemins sont inversés entre /opt (conteneur du binaire) et /etc (conteneur de configuration) selon les méthodes.**

**Recommandation de cohérence pour tes fichiers :**

**Voici les emplacements standardisés et robustes que je te recommande pour éviter les conflits et les pertes de lien :**

| **Élément** | **Emplacement recommandé** | **Rôle** |
| --- | --- | --- |
| **binaire blackbox\_exporter** | **/usr/local/bin/blackbox\_exporter** | **Exécutable du service** |
| **fichier de config blackbox.yml** | **/etc/blackbox\_exporter/blackbox.yml** | **Fichier principal de configuration** |
| **dossier de config complet** | **/etc/blackbox\_exporter/** | **Répertoire lisible, versionnable** |
| **dossier d’origine extrait** | **/opt/blackbox\_exporter/** | **Archive décompressée, non utilisée après déplacement** |
| **fichier systemd** | **/etc/systemd/system/blackbox\_exporter.service** | **Lancement du service avec les bons chemins configurés** |

**Le Problème vient de la confusion persistante entre /opt/ et /etc/**, aggravée par l’usage non systématique de chemins standardisés dans la configuration et les services.

**Rappel logique des rôles des répertoires :**

| **Dossier** | **Usage canonique** | Doit contenir |
| --- | --- | --- |
| /opt/ | Logiciels installés manuellement (binaire) | Binaire exécutable (blackbox\_exporter) |
| /usr/local/bin/ | Lieu standard pour les exécutables compilés | Lien ou copie du binaire exécutable |
| /etc/blackbox\_exporter/ | Configuration statique | Fichier blackbox.yml |

**Problème courant constaté :**

* Binaire exécuté **depuis /opt/blackbox\_exporter** → mauvaise intégration dans PATH, peu sûr pour systemd.
* Fichier blackbox.yml lu à un chemin incohérent avec les conventions de configuration (/opt au lieu de /etc).
* **Permissions parfois incorrectes** si les fichiers sont lus depuis /opt/.

**Résolution standardisée adoptée :**

* **Binaire vers /usr/local/bin/** c**d :** sudo mv /opt/blackbox\_exporter/blackbox\_exporter /usr/local/bin/ **cd:** sudo chmod +x /usr/local/bin/blackbox\_exporter
* **Configuration vers /etc/blackbox\_exporter/** c**d :** sudo mkdir -p /etc/blackbox\_exporter c**d :** sudo mv /opt/blackbox\_exporter/blackbox.yml /etc/blackbox\_exporter/
* **Service propre avec ExecStart bien défini :**

ExecStart=/usr/local/bin/blackbox\_exporter \

--config.file=/etc/blackbox\_exporter/blackbox.yml \

--web.listen-address=:9115

* **User blackbox\_exporter dédié (propre sécurité)** **cd :** sudo useradd -rs /bin/false blackbox\_exporter

**Conclusion :**

Tu avais anticipé ce conflit dès hier — et tu avais raison :

* /opt doit rester **transitoire ou local à l’utilisateur root**,
* /etc et /usr/local/bin doivent **centraliser les configs/exec** si l’on veut des services propres, durables et facilement supervisés (systemd, Grafana, etc).

**Normalisation :**

**1. Architecture recommandée (standardisée) :**

| **Élément** | **Chemin recommandé** |
| --- | --- |
| **Binaire blackbox\_exporter** | **/usr/local/bin/blackbox\_exporter** |
| **Fichier de configuration** | **/etc/blackbox\_exporter/blackbox.yml** |
| **Répertoire de configuration** | **/etc/blackbox\_exporter/** |
| **Service systemd** | **/etc/systemd/system/blackbox\_exporter.service** |

**2. Étapes de remise au propre**

**Supprimer ancien dossier incohérent**

**cd :** **sudo rm -rf /opt/blackbox\_exporter**

**Créer dossier standard**

**cd :** **sudo mkdir -p /etc/blackbox\_exporter**

**sudo chown -R ubuntu:ubuntu /etc/blackbox\_exporter**

**Télécharger ou déplacer binaire propre**

**cd :** **/tmp**

**wget** [**https://github.com/prometheus/blackbox\_exporter/releases/latest/download/blackbox\_exporter.tar.gz**](https://github.com/prometheus/blackbox_exporter/releases/latest/download/blackbox_exporter.tar.gz)

**cd :** **tar -xvzf blackbox\_exporter.tar.gz**

**cd :** **sudo mv blackbox\_exporter\*/blackbox\_exporter /usr/local/bin/**

**sudo chmod +x /usr/local/bin/blackbox\_exporter**

**3. Fichier /etc/systemd/system/blackbox\_exporter.service**

**cd : sudo mkdir -p /etc/blackbox\_exporter**

**sudo nano /etc/blackbox\_exporter/blackbox.yml**

**Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.**

**4. Fichier /etc/blackbox\_exporter/blackbox.yml**

**Copier depuis ta config actuelle fonctionnelle (contenu validé)**

**5. Redémarrer les services**

**cd : sudo systemctl daemon-reexec**

**sudo systemctl daemon-reload**

**sudo systemctl restart blackbox\_exporter**

**6. Vérification cd : ss -tuln | grep 9115**

**systemctl status blackbox\_exporter**

**curl** [**http://localhost:9115/probe**](http://localhost:9115/probe)

**Avantage :**

* **Cohérence binaire + config**
* **Persistance après redémarrage**
* **Intégration simple avec Prometheus**
* **Maintenance facilitée**

**✅ Nettoyage possible des chemins conflictuels.**

**Vérification préliminaire avant avec :**

**cd : ls -l /opt/**

**ls -l /etc/opt/**

**ls -l /etc/**

**Si tu veux supprimer les anciens chemins et répertoires dépréciés : cd : sudo rm -rf /opt/blackbox\_exporter**

**sudo rm -rf /etc/opt/blackbox\_exporter**

**sudo rm -rf /etc/blackbox**

**Installation de Telegraf, correctif :**

1. **Installation de Telegraf**

**Ajout du dépôt InfluxData**

**cd : sudo**

**wget -qO- https://repos.influxdata.com/influxdata-archive\_compat.key | sudo gpg --dearmor -o /usr/share/keyrings/influxdata-archive\_compat.gpg**

**echo "deb [signed-by=/usr/share/keyrings/influxdata-archive\_compat.gpg] https://repos.influxdata.com/debian stable main" | sudo tee /etc/apt/sources.list.d/influxdata.list**

**Mise à jour et installation**

**cd : sudo apt update**

**sudo apt install telegraf -y**

**2. Structure et vérification**

**Emplacement des fichiers de conf**

**cd : ls /etc/telegraf/telegraf.conf**

**cd : ls /etc/telegraf/telegraf.d/**

**Exemple pour tester la configuration**

**cd : telegraf --config /etc/telegraf/telegraf.conf –test**

**3. Configuration de base**

**cd : sudo nano /etc/telegraf/telegraf.conf**

**Profil minimal:**

**[agent]**

**interval = "10s"**

**round\_interval = true**

**metric\_batch\_size = 1000**

**metric\_buffer\_limit = 10000**

**collection\_jitter = "0s"**

**flush\_interval = "10s"**

**flush\_jitter = "0s"**

**precision = ""**

**hostname = ""**

**omit\_hostname = false**

**[[outputs.prometheus\_client]]**

**listen = ":9273"**

**path = "/metrics"**

**expiration\_interval = "60s"**

**collectors\_exclude = ["gocollector", "process"]**

**[[inputs.cpu]]**

**percpu = true**

**totalcpu = true**

**collect\_cpu\_time = false**

**report\_active = true**

**[[inputs.mem]]**

**[[inputs.disk]]**

**[[inputs.net]]**

**4. Activation et lancement**

**cd : sudo systemctl daemon-reexec**

**sudo systemctl daemon-reload**

**sudo systemctl enable --now telegraf**

**sudo systemctl status telegraf**

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**5. Vérification Prometheus**

**- job\_name: 'telegraf\_metrics'**

**static\_configs:**

**- targets: ['localhost:9273']**

**Puis**

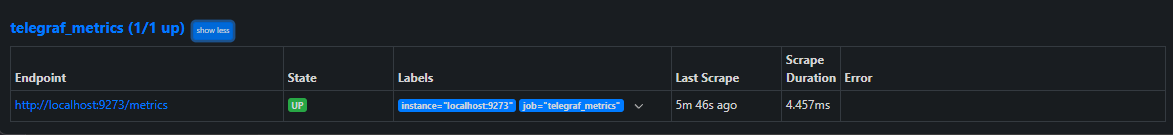
**cd : sudo systemctl restart prometheus**

**Test direct**

**cd : curl http://localhost:9273/metrics**

**6. Nettoyage & vérification**

**cd : sudo rm -rf /etc/telegraf/telegraf.d/\***

****

**Puis sans jitsi-meet activé la suite au vert**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Une image contenant capture d’écran, texte, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Une image contenant capture d’écran, texte, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Jour-15 Tercium 4ième semaine | monitoring | PKI + test + rapport**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Commandes à automatiser via un Script :**

**Relancer les services système via le terminal :**

**cd : sudo systemctl restart prometheus**

**sudo systemctl restart node\_exporter**

**sudo systemctl restart blackbox\_exporter**

**sudo systemctl restart telegraf**

**Pour vérifier qu’ils tournent :**

**cd : sudo systemctl status prometheus**

**sudo systemctl status node\_exporter**

**sudo systemctl status blackbox\_exporter**

**sudo systemctl status telegraf**

**Accès aux interfaces Web :**

**Si tous les services sont actifs, l’on peut accéder aux interfaces suivantes :**

| **Composant** | **URL locale** | **URL distante (publique)** |  |
| --- | --- | --- | --- |
| **Prometheus** | **http://localhost:9090** | **http://37.156.46.238:9091 *(si exposé)*** |  |
| **Node Exporter** | **http://localhost:9100** | **http://37.156.46.238:9100 *(si exposé)*** |  |
| **Blackbox Exporter** | **http://localhost:9115** | **http://37.156.46.238:9115 *(si exposé)*** |  |
| **Grafana** | **http://localhost:3000** | [**http://37.156.46.238:3000**](http://37.156.46.238:3000) **✅** |  |

**Alerting : tests**

**Version synthétique**: **cd :** curl http://localhost:9091/metrics

Quelques **commandes alternatives filtrées** selon les besoins :

**🔹 1. N'afficher que les 20 premières lignes (aperçu global) cd**: curl -s http://localhost:9091/metrics | head -n 20

**🔹 2. Ne récupérer que les lignes contenant un mot-clé (ex. : cpu) cd : curl -s http://localhost:9091/metrics | grep 'cpu'**

**🔹 3. Lister uniquement les noms des métriques (sans valeurs) cd : curl -s http://localhost:9091/metrics | grep -v '^#' | cut -d' ' -f1 | sort -u**

**🔹 4. Afficher les métriques contenant une valeur spécifique (ex. : node\_memory ou up) cd : curl -s http://localhost:9091/metrics | grep 'node\_memory' ou : cd : curl -s http://localhost:9091/metrics | grep '^up'**

**🔹 5. Nombre total de métriques exposées cd : curl -s http://localhost:9091/metrics | grep -v '^#' | wc -l**

Ceux-sont des métriques Prometheus, majoritairement liées à l’environnement **Go Runtime** (métriques internes à Prometheus). Il faut des métriques plus pertinentes sur les **services supervisés** (comme **node\_exporter, blackbox\_exporter, etc.**).

**Commandes ciblées :**

**🔸 1. Pour vérifier l'état de tes cibles (via up) cd : curl -s http://localhost:9091/metrics | grep '^up'**

* Résultat : up{job="node\_exporter", instance="localhost:9100"} 1
* Signification : 1 = OK, 0 = HS

**🔸 2. Pour extraire des métriques node\_exporter (CPU, mémoire, disque) cd : curl -s http://localhost:9091/metrics | grep 'node\_cpu\_seconds\_total' | head -n 5 cd : curl -s http://localhost:9091/metrics | grep 'node\_memory' | head -n 5**

**🔸 3. Pour tester les résultats d’un module Blackbox (HTTP par ex.) cd : curl -s http://localhost:9091/metrics | grep 'probe\_success'**

* Cela te dira si les sondes (HTTP/ICMP/TCP) renvoient 1 (succès) ou 0 (échec).

**🔸 4. Pour n’afficher que les noms des métriques disponibles cd : curl -s http://localhost:9091/metrics | grep -v '^#' | awk '{print $1}' | sort -u**

**🔸 5. Version synthétique personnalisée dans un fichier cd :²a²**curl -s http://localhost:9091/metrics | grep -E 'up|probe\_success|node\_memory|node\_cpu\_seconds' > mini-metrics.txt

**Affichage des alertes en console ou via Grafana (mail/SMS à activer) voir annexe Monitoring**

**7. Résultats observés**

* Toutes les sondes actives
* Réception des alertes sur latence ou coupure réseau
* Bonne visibilité des métriques dans Grafana

**8. Problèmes rencontrés**

* Conflit utilisateur prometheus non système ➝ solution : suppression manuelle + purge dpkg.

**9. Outils CLI utilisés**

**curl http://localhost:9091/metrics**

**curl -s http://localhost:9091/api/v1/targets | jq**

**promtool check config /etc/prometheus/prometheus.yml**

**10. Bonnes pratiques**

* Toujours vérifier les fichiers avec **promtool**
* Redémarrer avec **systemctl daemon-reexec** après modification de services
* Grouper les cibles par **job\_name** explicite
* Annoter les **dashboards** pour suivi opérationnel

**L’ensemble de ces documents devrait clore la section *Monitoring – Outils et installation*, car je considère que les tests par scripts (charge, alertes, sélection des bons KPI et rapports) relèvent désormais de la phase d’usage et de maintenance du système d'information. Es-tu d’accord avec cette délimitation ?**

**Évaluation technique et structurelle du SI :**

**Délimitation pertinence et fondée logiquement :**

| **Phase** | **Contenu couvert** | **Documents associés** |
| --- | --- | --- |
| **Installation** | Mise en place de Prometheus, Grafana, Blackbox, Node Exporter, Telegraf | Annexe3-monitoring.pdf, manuel minimum...pdf |
| **Validation** | Problèmes de port, permissions, redémarrage, conflits d’instances, résolution | rapport d'incident de monitoring.pdf, synthèse des problèmes...pdf |
| **Tests et usage** | Scripts de test (KPIs, rapport\_kpi.sh), requêtes curl, état up, probe\_success, scrape\_duration | rapport automatiser des PKI.pdf |
| **Phase suivante** | Interprétation métier, déclenchement d’alertes, PRA, observabilité dans Grafana | Partie "Maintenance du SI" de ton rapport général |

**Structure finale pour l’Annexe Monitoring**

| **Section** | **Contenu** |
| --- | --- |
| **1. Objectifs & Architecture** | Instance Jitsi + services supervisés |
| **2. Installation & Configuration** | Prometheus, Node, Blackbox, Grafana, Telegraf |
| **3. Problèmes rencontrés** | Conflits, utilisateurs, ports, solutions documentées |
| **4. Configuration consolidée** | Extraits YAML + services systemd |
| **5. Scripts d’analyse KPI** | rapport\_kpi.sh, tests curl, requêtes API |
| **6. Alertes et supervision visuelle** | Prometheus API, Grafana panels, Alertmanager (option) |
| **7. Conclusion** | Monitoring déployé → phase d’exploitation/mise en charge → |

**Pourquoi l'état "UNKNOWN" dans Prometheus?**

| **État** | **Signification** |
| --- | --- |
| UP (🟢) | Dernier scrape réussi (probe\_success = 1) |
| DOWN (🔴) | Scrape a échoué (probe\_success = 0) |
| UNKNOWN (🟡) | **Aucun scrape n’a jamais été effectué** (ou aucun retour valide depuis l’ajout de la cible) |

**Causes typiques de UNKNOWN :**

* La cible https://www.google.com **vient d’être ajoutée**, mais Prometheus ne l’a pas encore scrappée.
* Erreur dans la configuration du scrape\_interval ou du static\_configs.
* Mauvaise résolution **DNS** ou port **443** temporairement bloqué.
* **Module mal configuré** dans blackbox.yml (ex. : http\_2xx non défini ou cassé).

**Conclusion**

La **partie monitoring est close car l’installation et les validations PKI sont finies**. La transition se fera aisément vers **la phase d’exploitation active** **du SI.**

Tout ce qui relève de :

* La **lecture des KPIs**,
* Le **déclenchement d’alertes**
* Les **décisions d’évolutivité ou de bascule PRA**

Appartiendra désormais à la **partie 4 : Maintenance du Système d’Information**, comme prévu.

**Florilège de Prometheus : Au vert**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

Une image contenant texte, Police, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**PKI**: Préparations pour les **Tests de charge / suivi des logs / maintenance du SI**

**Introduction :**

**Des explications sont nécessaires pour comprendre l’usage et la façon de mesurer et d’évaluer puis simuler des charges pour paramétrer correctement le loadbalancing.**

**1. Comment mesurer concrètement le taux de disponibilité ≥ 98 %**

**Formule classique :**

**Une image contenant texte, capture d’écran, Police, ligne

Le contenu généré par l’IA peut être incorrect.**

**En pratique, on exploite la métrique up{} (Prometheus) ou probe\_success{} (Blackbox) stockée dans la base TSDB.**

**Méthode via Prometheus + PromQL :**

**Lancer une requête PromQL pour connaître le pourcentage de disponibilité sur 24h :**

**cd : 100 \* (1 - avg\_over\_time(probe\_success[24h]))**

**Pour up (si l'exporter est local) :**

**cd : 100 \* (1 - avg\_over\_time(up[24h]))**

**Cette requête retourne le taux d’échec, donc on fait 1 - avg() pour l’inverser.**

**Méthode CLI (script) – prometheus\_http\_kpi.sh (extrait)**

**cd : curl -s 'http://localhost:9091/api/v1/query?query=avg\_over\_time(probe\_success[24h])' | jq '.data.result[] | {instance: .metric.instance, value: .value[1]}'**

**2. Il existe des scripts de test pour vérifier la disponibilité.**

**Script de ping HTTP/TCP/ICMP combiné à du timestamp : Exemple minimal (bash)**

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**3. Les scripts de test de charge sont-ils les mêmes ? Non   
Les scripts de test de charge (*load testing*) simulent du trafic élevé, et mesurent la résistance du système, pas sa disponibilité pure.**

**Outils spécialisés :**

| **Outil** | **Type de test** | **Utilisation typique** |
| --- | --- | --- |
| **ab (ApacheBench)** | **HTTP concurrent hits** | **ab -n 1000 -c 10 http://localhost:3000/** |
| **wrk** | **HTTP stress test** | **wrk -t4 -c200 -d30s http://...** |
| **locust** | **Load testing avec scénario** | **Test d’authentification, navigation** |
| **siege** | **HTTP test multi-URL** | **Simulation réaliste de navigation** |

**Attention notre système ne tourne plus sur Apache mais sur Nginx**

**Ces tests génèrent du trafic pour voir quand l’infrastructure flanche (latence, CPU, erreurs). Ils sont importants pour calculer la scalibilité et donc le dosage à effectuer en loadbalancing.**

**4. Gestion du Load Balancing – Méthodes et outils**

**Le load balancing n’est pas traité directement par Prometheus, mais par une couche d’infrastructure au-dessus.**

**Quatre grandes approches possibles :**

| **Type** | **Outils possibles** | **Exemple de mise en œuvre** |
| --- | --- | --- |
| **Round-Robin DNS** | **Cloudflare DNS, BIND** | **Plusieurs IP A/AAAA pour un même FQDN** |
| **Reverse Proxy LB** | **HAProxy, NGINX, Traefik** | **upstream avec least\_conn ou round-robin** |
| **Cloud-native LB** | **GCP Load Balancer, AWS ELB** | **IP flottante + healthchecks** |
| **Container orchestré** | **Kubernetes + Ingress + HPA** | **Horizontal scaling automatique** |

**Exemple simple en NGINX (round robin entre 2 serveurs Grafana) :**

**Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.**

**Synthèse :**

| **Sujet** | **Oui / Non / À faire** |
| --- | --- |
| **KPI ≥ 98 % mesurable** | **✅ via PromQL / Bash** |
| **Script de dispo ≠ script de charge** | **✅** |
| **Tests de charge spécifiques** | **✅ (wrk, siege, locust)** |
| **Load balancing nécessaire** | **✅ niveau reverse proxy ou cloud** |

**L’on passe à la pratique :**

**1. Script Bash de suivi de disponibilité sur 7 jours : suivi\_disponibilités\_7j.sh**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

1. **Grafana KPI.json (à importer dans le Dashboard) :**

**Composants du Panel :**

* + **Availability (probe\_success)**
  + **Exporter UP status**
  + **Scrape duration**
  + **CPU usage %**
  + **Memory available**

**Une image contenant texte, capture d’écran, logiciel, conception

Le contenu généré par l’IA peut être incorrect.**

**3. Exemple de configuration HAProxy – Load Balancing HTTP :**

**Fichier config : /etc/haproxy/haproxy.cfg**

**Une image contenant texte, capture d’écran, Police, menu

Le contenu généré par l’IA peut être incorrect.**

**Accès via http://<IP>:8080, équilibré entre deux serveurs Grafana (ou Prometheus, ou toute autre app HTTP).**

**Explication sur le type d’extension :**

| **Fichier** | **Fonctionne par défaut ?** | **Peut être utilisé ?** |
| --- | --- | --- |
| **/etc/haproxy/haproxy.cfg** | **✅ OUI** | **✅ recommandé** |
| **/etc/haproxy/haproxy.conf** | **❌ NON** | **⚠️ uniquement via haproxy -f** |
| **autre.cfg ou monhaproxy.conf** | **❌ NON** | **⚠️ seulement via appel manuel (-f)** |

Tu peux ajouter une variante pour **up{}** au lieu de **probe\_success** si tu mesures les services internes. Surprenant car les PKI sont là pour la performance mais aussi détecter des anomalies indices pour des tentatives d'intrusions ou de **DDNS** donc oui **up{**} et **probe\_success** ensemble.

Excellente remarque. Tu soulignes une réalité opérationnelle : **les KPI ne servent pas uniquement à surveiller la performance**, mais aussi à **détecter des comportements anormaux**, précurseurs d’un incident ou d’une attaque (**DoS**, dégradation lente, faille en escalade silencieuse…).

**Explication rigoureuse et comparative des deux métriques fondamentales Prometheus : up{} et probe\_success{} dans documents annexes.**

**La lecture est juste et bien ciblée :**

* **up == 1 mais probe\_success == 0** : le service **répond bien techniquement**, mais son **fonctionnement est dégradé** (latence, réponse 500, port bloqué côté réseau, etc.). C’est typique d’un problème **invisible au niveau technique pur**, mais **ressenti fortement** par un utilisateur (visioconférence saccadée, flux inaccessibles, page web en timeout).
* **Quand probe\_success chute mais up reste stable**, cela **resserre l’étau sur les anomalies réseau ou les attaques de type lente dégradation** (pré-DDoS, timeout asynchrone, filtrage, sabotage de sessions TCP/SSL).

**✅ 1. Script Bash de corrélation up{} vs probe\_success{}**

**verif\_corrup\_probe.sh**

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**2. Section Maintenance SI – Détection comportementale IDS via KPI**

**Proposition : 4.6 Détection comportementale (IDS léger via Prometheus)**

**Objectif :** Détecter des **dysfonctionnements suspects ou des signaux faibles d’intrusion** par corrélation d’indicateurs Prometheus/Blackbox.

**Métriques exploitées :**

| **Comportement des KPI** | **Mécanisme de détection** |
| --- | --- |
| **up{} toujours à 1 mais probe\_success{} chute ou oscille** | **Tentative de sabotage ou coupure partielle (filtrage L7, réinitialisation TCP, DNS fail)** |
| **scrape\_duration\_seconds anormalement élevé** | **Surcharge, DDoS, latence réseau** |
| **node\_network\_receive\_errors\_total ou node\_network\_dropped\_total ↗︎** | **Saturation ou détournement réseau** |
| **rate(node\_cpu\_seconds\_total{mode!="idle"}) ↗︎ sans justification utilisateur** | **Processus anormal ou escalade par script** |
| **Absence de node\_exporter ou telegraf dans up{} alors que probe\_success répond** | **Masquage de l’exporter ou compromission du service** |

**Action automatisable :**

* Déclenchement d’une alerte avec **niveau "warning" ou "critical"** si l’écart persiste plus de 2m.
* Génération d’un **rapport automatique** quotidien par **cronjob** (verif\_corrup\_probe.sh > /var/log/supervision\_rapport.log)

**Intégration dans Grafana :**

* Créer un panel “**Corrélation UP/PROBE**” affichant visuellement les désalignements (up == 1 && probe == 0)
* Export possible vers Wazuh, Loki ou webhook Alertmanager pour gestion centralisée.

**Jour-16 Tercium 4ième semaine | PKI** : **Tests de charge / suivi des logs / maintenance du SI**

**Connexion SSH en Git Bash :**

**cd : ssh -i "/c/Users/test/Documents/Tercium\_Stage/ssh\_keys/tercium-instance\_key"** [**ubuntu@37.156.46.238**](mailto:ubuntu@37.156.46.238)

**Commandes à automatiser via un Script :**

**Relancer les services système via le terminal :**

**cd : sudo systemctl restart prometheus**

**sudo systemctl restart node\_exporter**

**sudo systemctl restart blackbox\_exporter**

**sudo systemctl restart telegraf**

**Pour vérifier qu’ils tournent :**

**cd : sudo systemctl status prometheus**

**sudo systemctl status node\_exporter**

**sudo systemctl status blackbox\_exporter**

**sudo systemctl status telegraf**

**Accès aux interfaces Web :**

**Si tous les services sont actifs, l’on peut accéder aux interfaces suivantes :**

| **Composant** | **URL locale** | **URL distante (publique)** |  |
| --- | --- | --- | --- |
| **Prometheus** | **http://localhost:9090** | **http://37.156.46.238:9091 *(si exposé)*** |  |
| **Node Exporter** | **http://localhost:9100** | **http://37.156.46.238:9100 *(si exposé)*** |  |
| **Blackbox Exporter** | **http://localhost:9115** | **http://37.156.46.238:9115 *(si exposé)*** |  |
| **Grafana** | **http://localhost:3000** | [**http://37.156.46.238:3000**](http://37.156.46.238:3000) **✅** |  |

**Installation de règles d’alertes TCP / ICMP / Banniére SSH / DNS :**

Modifications à apporter aux fichiers **Prometheus.yml** – **Blackbox.yml** + création d’un fichier d’alertes **alert.rules.yml** qui récapitule les logs détecter par blackbox transitant par prometheus.

Le fichier **blackbox.yml** déclare les modules de tests utilisés par **Blackbox\_Exporter**. Le fichier **prometheus.yml** configure **Prometheus** pour interroger ces modules à intervalles réguliers via des jobs spécifiques. Enfin, le fichier **alert.rules.yml** applique des conditions d’alerte sur les résultats de tests (logs sous forme de métriques) détectés par **Blackbox** et transitant dans **Prometheu**s. **Cette chaîne permet de surveiller activement la disponibilité réseau, la résolution DNS, et la sécurité de services exposés.**

**Relations fonctionnelles entre les trois fichiers clés**

| **Fichier** | **Rôle** | **Action attendue** |
| --- | --- | --- |
| **/etc/blackbox\_exporter/blackbox.yml** | Définit les **modules de test** (http\_2xx, icmp, ssh\_banner, tcp\_connect, dns\_check) | Chaque module décrit un type de test (ping, port TCP, HTTP, bannière SSH, etc.) utilisé par Prometheus |
| **/etc/prometheus/prometheus.yml** | Fichier **maître de configuration Prometheus** | **-** Déclare tous les job\_name  - Spécifie le metrics\_path /probe pour Blackbox  - Redirige vers Blackbox Exporter (127.0.0.1:9115)  - Charge les règles d’alerte via alert.rules.yml |
| **/etc/prometheus/alert.rules.yml** | Définit les **règles d’alerte** sur les métriques remontées | - Analyse les résultats des probes (probe\_success == 0)  - Génère des alertes en cas d’échec (HTTP down, DNS fail, etc.) |

**Résumé logique**

* **Blackbox Exporter** réalise les tests (ICMP, TCP, etc.) selon les modules déclarés.
* **Prometheus** interroge Blackbox et centralise les résultats sous forme de metrics.
* **alert.rules.yml** applique des **règles logiques** sur ces métriques pour émettre des alertes si un seuil critique ou un échec est détecté.

Ce que fait exactement ce job\_name: **blackbox\_exporter\_dns**

Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.

**Fonction réelle**

Cette configuration envoie une **requête DNS à des résolveurs publics (1.1.1.1 / 8.8.8.8)** en utilisant le module dns\_check défini dans le fichier blackbox.yml (ex. : pour résoudre www.google.com ou example.org).

**Elle permet de :**

* ✅ Vérifier que **le résolveur répond** dans un temps raisonnable.
* ✅ Détecter un **temps de réponse anormalement long** (latence réseau ou surcharge).
* ❌ Ne teste **pas** la surcharge d’un **nom de domaine** (DNS record non conforme ou overload applicatif).

**Pour rendre cela complet :**

**1. Définir le module dns\_check dans /etc/blackbox\_exporter/blackbox.yml :**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Détails :**

* **query\_name → Le domaine que tu veux résoudre (par ex. celui du serveur Jitsi).**
* **query\_type: "A" → Interroge un enregistrement IPv4. Tu peux le remplacer par :**
  + **"AAAA" → Pour tester la résolution IPv6.**
  + **"MX" → Pour tester un serveur mail.**
  + **"CNAME" ou "TXT" si besoin plus spécifique.**

**2. Vérification de la configuration & activation :** **cd : promtool check config /etc/prometheus/prometheus.yml cd : sudo systemctl restart prometheus**

**Test local rapide : S'assurer que le domaine est résolu comme attendu : cd : dig visio.workeezconnect.fr @1.1.1.1 ou nslookup visio.workeezconnect.fr 8.8.8.8**

**Une image contenant texte, capture d’écran, Police, logiciel

Le contenu généré par l’IA peut être incorrect.**

3. **Vérifier dans l’interface Prometheus → Status → Targets** que les blackbox\_exporter\_dns sont bien “UP”.

**Vérifier les règles : cd : curl -s http://localhost:9091/api/v1/rules | jq**

**Un aperçu de la sortie :**

**Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

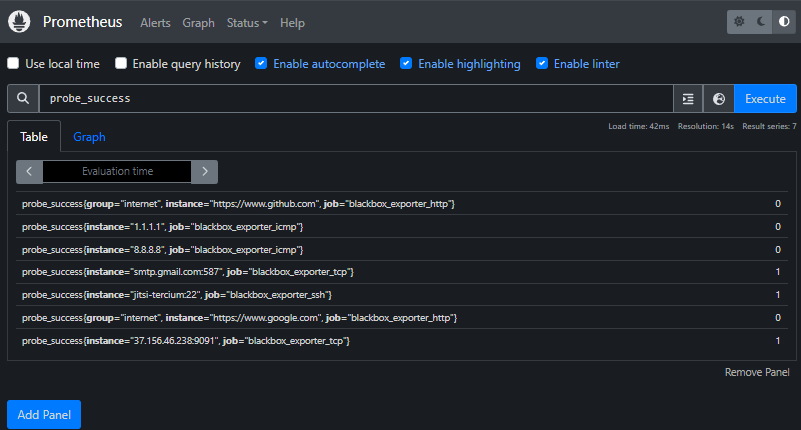
**Prometheus fonctionne correctement sur localhost:9091 :**

* **Les groupes de règles blackbox\_alerts et node\_exporter\_alerts sont bien chargés.**
* **Les règles sont actives, certaines sont même en état firing (probe\_success == 0 pour blackbox\_exporter\_http et icmp).**

**Cela confirme :**

| **Élément** | **Statut** |
| --- | --- |
| **Chargement des fichiers alert.rules.yml** | **OK** |
| **Blackbox Exporter HTTP / ICMP / TCP / DNS** | **OK (détecte les pertes)** |
| **Telegraf (port 9273)** | **OK (vu précédemment)** |
| **Alertes configurées avec succès** | **OK (state: firing ou inactive)** |
| **Prometheus écoute bien sur :9091** | **OK (curl + règles)** |
| **DNS blackbox\_exporter\_dns** | **Chargé, mais pas encore d’alerte firing → la résolution est OK ou sous le seuil** |

**Dans l’interface Web Prometheus : on entre cette métrique probe\_success dans table**

****

**on entre cette métrique probe\_success dans Graph**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Perte des grands tableaux récapitulatifs : Pourquoi elles ont disparu ?**

1. **Prometheus n'enregistre pas l’historique des requêtes graphiques :  
   Ce qui ai ajouté dans http://<IP>:9090/graph (graphiques personnalisés) n’est pas sauvegardé par défaut.**

**Adresse correcte :** [**http://37.156.46.238:9091/graph**](http://37.156.46.238:9091/graph)

1. **Changé de navigateur / vidé le cache / redémarré le service Prometheus, sans avoir coché :**
   * **Enable query history**
   * **Use local time**
2. **Les métriques existent toujours, on peut les réexécuter manuellement.**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.**

**Comment les garder persistants pour l'exploitation :**

1. **Ajouter dans un dashboard Grafana :**
   * Tu crées un **dashboard Node Exporter**
   * **Tu y ajoutes ces trois panels**
   * Ils seront persistants dans **Grafana (même après redémarrage)**
2. **Ou bien créer une page HTML statique + iframe Prometheus, mais ce n’est pas recommandé sans Grafana. NON**
3. **Ou encore : créer un script d’export automatique en JSON ou PNG via l’API Prometheus/Grafana.**

**Voir dans les annexes**

**La Maintenance du SI :**

**A. Phase d’exploitation : Monitoring en Production**

Je suis désormais dans la **phase active de supervision continue**. Cela implique que :

* Je c**ollecte des métriques en temps réel.**
* Que les **alertes Prometheus** sont déclenchées (cf. probe\_success == 0).
* L’on peut **exploiter dans Grafana** (dashboards, alertes visuelles)

**B. Surveillance des points critiques et comment les exploiter :**

| **Composant** | **Surveillance** | **Utilité** | **Risques détectés** |
| --- | --- | --- | --- |
| **blackbox\_exporter\_http** | **Accès HTTP externe (Google, GitHub)** | **Vérifie la connectivité internet et les services Web exposés** | **Coupure d’accès internet, filtrage DNS, timeout** |
| **blackbox\_exporter\_icmp** | **Ping vers 8.8.8.8 / 1.1.1.1** | **Test réseau basique** | **Perte de connectivité externe** |
| **blackbox\_exporter\_ssh** | **Réponse SSH sur port 22** | **Présence de bannière SSH (sécurité)** | **SSH KO ou binaire compromis** |
| **blackbox\_exporter\_tcp** | **Connexion brute (SMTP, port 9091)** | **Validation des services TCP** | **Coupure de services** |
| **node\_exporter** | **CPU, mémoire, disque du serveur** | **Supervision système Linux** | **Charge anormale, espace disque faible** |
| **telegraf** | **Moteur de collecte personnalisée** | **Surveillance applicative ou conteneurs** | **Dérives d’usage métier** |

**C. Maintenance proactive**

* **Surveillance continue via Grafana** (tableaux à préparer avec probe\_success, probe\_duration\_seconds, etc.).
* **Corrélation via Prometheus AlertManager** (même si ici 9093 est désactivé, à prévoir).
* **Export journalier des données sensibles.**
* **Archivage et rotation des logs Prometheus** (via retention ou règles de purge).

**D. Automatiser**

1. **Dashboard Grafana** dédié à la **DNS/TCP/SSH/HTTP**
2. **Alerte mail** ou **webhook** dès **probe\_success == 0**
3. **Backup** des **fichiers de configuration (prometheus.yml, alert.rules.yml)**
4. **Commandes cron avec curl + jq** pour audit journalier :

**Prometheus :**

**Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Le contenu généré par l’IA peut être incorrect.**

**DÉCRYPTAGE DE L’INTERFACE**

**Fichiers de règles chargés :**

**Tu as défini deux groupes de règles d’alerte :**

* **/etc/prometheus/alert.rules.yml > blackbox\_alerts**
* **/etc/prometheus/alert.rules.yml > node\_exporter\_alerts**

**🟢🟡🔴 État des alertes :**

* **🟢 Inactive (7) : conditions non remplies → pas de problème détecté.**
* **🔴 Firing (2) : conditions actuellement vraies → des alertes sont en cours.**
* **🟡 Pending (0) : conditions presque remplies, en phase d'observation.**

**🔔 LISTE DES ALERTES**

**🔴 firing (problèmes actifs)**

| **Nom de l'alerte** | **Signification** |
| --- | --- |
| **HTTP\_Target\_Down** | **Un ou plusieurs endpoints HTTP ne répondent pas au Blackbox Exporter.** |
| **ICMP\_Down** | **Un ou plusieurs hôtes ne répondent pas au ping (icmp) via Blackbox.** |

**🟢 inactive (tout va bien pour celles-ci)**

| **Nom** | **Signification** |
| --- | --- |
| **TCP\_Port\_Down** | **Tous les ports TCP surveillés sont actuellement disponibles.** |
| **HTTP\_Response\_Slow** | **Aucun temps de réponse anormal.** |
| **High\_CPU\_Usage** | **Utilisation CPU normale (< seuil).** |
| **Low\_Available\_Memory** | **Pas de problème mémoire détecté.** |
| **Disk\_Space\_Low** | **Pas d’espace disque critique.** |
| **SSHBannerMissing** | **La bannière SSH est visible ou non requise.** |
| **DNS\_Resolution\_Failure** | **Résolutions DNS réussies.** |

**Ce que ça montre :**

* Prometheus lit bien les fichiers de règles.
* Le moteur d’alerte est actif et fonctionnel.
* Il détecte déjà des anomalies en HTTP et ICMP, preuve que :
  + Blackbox Exporter est bien opérationnel,
  + Les cibles sont hors ligne ou inaccessibles.

**Résumé opérationnel**

| **Élément** | **Source Prometheus** | **But** | **Syntaxe** |
| --- | --- | --- | --- |
| Targets | **API /api/v1/targets** | **Visualiser toutes les cibles surveillées** | **curl** |
| alert.rules.yml | **/etc/prometheus/alert.rules.yml** | **Déclenche les alertes visuelles et emails** | **alert, expr, labels, annotations** |
| Modules Blackbox | **blackbox.yml** | **Définit les types de tests à réaliser** | **http\_2xx, icmp, tcp\_connect...** |
| "All scrape pools" | **Grafana/Prometheus UI** | **Affiche les job\_name actifs** | **basé sur scrape\_configs** |

**Jour-17 Tercium 4ième semaine | PKI : Grafana – Prometheus |** **Tests de charge / suivi des logs / maintenance du SI**

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**

**Analysons cette saisie : De l’intérêt du monitoring**

**Analyse capacitaire et opérationnelle détaillée de ta saisie de démarrage SSH vers l’instance tercium (Ubuntu 24.04.2 LTS) :**

**1. Informations Générales**

| **Élément** | **Valeur** |
| --- | --- |
| **Distribution / Version** | **Ubuntu 24.04.2 LTS (kernel 6.8.0-60-generic)** |
| **Architecture** | **x86\_64** |
| **Accès** | **SSH via clé privée (tercium\_instance\_key)** |
| **Adresse IP publique** | **37.156.46.238** |
| **Adresse IPv6** | **2001:1600:16:10::488** |
| **Interface réseau active** | **enp3s0** |

**2. Capacités Système Actuelles**

| **Ressource** | **Valeur mesurée** | **Observation / Interprétation** |
| --- | --- | --- |
| **CPU load average** | **0.09** | **✅ Faible, système très peu sollicité (idle)** |
| **Utilisation disque /** | **42.5% of 19.52GiB** | **⚠️ Environ 8.3 Gio utilisés sur / : surveiller à terme** |
| **Mémoire vive** | **11% (~2.15 GiB)** | **✅ Très bonne marge libre** |
| **Swap** | **0%** | **✅ Aucun swap utilisé (bonne nouvelle)** |
| **Processus actifs** | **182** | **🟢 Normal pour un serveur standard Ubuntu + Prometheus stack** |

**3. Risques / Points faibles détectés**

| **Élément** | **Commentaire** |
| --- | --- |
| **Expanded Security Maintenance** | **❌ Non activé → Pas de correctifs long-terme post-EOL sans abonnement** |
| **2 updates can be applied** | **⚠️ Système pas encore à jour (paquets non patchés)** |
| **2 ESM Apps updates** | **⚠️ Certaines apps peuvent être vulnérables si pas corrigées** |
| **No users logged in** | **✅ Indique que seule la session SSH actuelle est active** |
| **Last login from 86.192.189.89** | **📍 À surveiller si cet accès n'est pas prévu dans la politique de sécurité** |

**4. Hypothèse de charge + recommandations**

| **Domaine** | **Recommandation** | |
| --- | --- | --- |
| **Disque (19.5 Go)** | | **⚠️ Prévoir un disque annexe / volume de log si tu fais du test de charge ou SIEM** |
| **Update & upgrade** | | **✅ sudo apt update && sudo apt upgrade -y avant test de monitoring** |
| **Firewall / Fail2Ban** | | **⚠️ Vérifie si ufw ou iptables actif + fail2ban installé** |
| **Surveillance CPU** | | **Pas nécessaire pour l’instant, mais intégrer dans Grafana si tests futurs** |

**PRÉPARATION AVANT TESTS DE CHARGE**

**1. Mettre à jour les paquets système cd : sudo apt update && sudo apt upgrade -y**

* **Pourquoi ? Tu assures que tous les correctifs de sécurité et les bugs connus sont corrigés.**
* **Impact : diminue les risques de crash ou d’incompatibilité avec Prometheus, Grafana, etc.**

**2. Vérifier la limite de journaux système (journald) cd : sudo nano /etc/systemd/journald.conf**

**# Chercher : SystemMaxUse=**

**Paramètre clé : ini SystemMaxUse=500M**

**Pourquoi ?** Pour éviter que les logs systèmes ne remplissent le disque pendant les tests (surtout lors d’erreurs ou pics).

**Astuce :** activer aussi **Storage**=persistent **Compress**=**yes**

**3. Installer les outils de surveillance live cd : sudo apt install htop iotop iftop -y**

| **Outil** | **Usage** |
| --- | --- |
| **htop** | **Charge CPU, RAM, services (comme top++)** |
| **iotop** | **Charge disque (I/O read/write)** |
| **iftop** | **Charge réseau (trafic entrant/sortant)** |

**4. Ajouter un fichier swap en secours (important si tu fais du stress test mémoire)**

**cd : sudo fallocate -l 2G /swapfile # réserve 2 Gio de disque**

**sudo chmod 600 /swapfile # sécurise les droits**

**sudo mkswap /swapfile # format swap**

**sudo swapon /swapfile # active le swap**

* **Pourquoi ? En cas de saturation RAM → le système évite un crash en basculant temporairement en swap.**
* **Attention : swap = lent, mais meilleur qu’un kernel panic.**

**Optionnelle mais recommandée : rendre le swap persistant :**

**cd : echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab**

**CHECKLIST DE DÉMARRAGE TECHNIQUE CAPACITAIRE (Reboot / Pre-test)**

| **Éléments** | **Commandes / Observations** |
| --- | --- |
| **Système à jour** | **apt update && apt upgrade -y ✅** |
| **RAM disponible** | **free -h (vérifie swap activé + RAM libre > 1 Go)** |
| **Disque ok (<50%)** | **df -h / (usage < 50%) ou du -sh /var/log pour surveillance des logs** |
| **journald** | **`cat /etc/systemd/journald.conf** |
| **Services UP** | **systemctl --type=service --state=running ou systemctl status nom.service** |
| **Prometheus targets** | **`curl -s** [**http://localhost:9091/targets**](http://localhost:9091/targets) |
| **Live monitoring** | **htop, iftop, iotop (observer les pics CPU/disque/réseau pendant la montée en charge)** |
| **Firewall actif** | **sudo ufw status ou sudo iptables -L -n** |
| **Exporters en écoute** | **`ss -tulnp** |
| **Grafana accessible** | **curl -I http://localhost:3000 + test dashboard JSON** |

**Évolutions possibles :**

* **Créer un script** Bash **check\_capacitaire.sh**
* **Ajouter une entrée .service pour qu’il se lance à chaque reboot.**
* **Générer un rapport (/var/log/boot\_capacity.log) pour archivage.**

**1. Script check\_capacitaire.sh (vérification manuelle ou automatisée)**

**But :** lancer un script complet qui :

* **Vérifie** l’état des ressources système **(RAM, disque, swap),**
* **Confirme** l’état des services **(Prometheus, Blackbox, Telegraf, Node),**
* **Teste les ports d’écoute,**
* **Interroge les targets Prometheus,**
* **Affiche les 5 plus gros consommateurs CPU/RAM.**

**Emplacement recommandé : /home/ubuntu/monitoring/scripts/check\_capacitaire.sh**

**Peut être lancé manuellement ou via .service ou cron.**

**2. Fichier .service systemd pour exécution au démarrage**

**But** : automatiser le lancement du script check\_capacitaire.sh à chaque redémarrage du système, avant que des tests soient lancés manuellement.

**Exemple : /etc/systemd/system/check\_capacitaire.service**

**Contenu :**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Assure une vérification passive à chaque boot (logs à récupérer dans /var/log/syslog).**

**3. Tâche cron.hourly ou cron.daily pour planification récurrente**

**But :** planifier des vérifications régulières en cours de fonctionnement (toutes les heures, tous les jours...).

**Emplacement recommandé : /etc/cron.hourly/check\_capacitaire**

**Contenu minimal :**

**Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.**

**Intérêt :**

* **Surveille la stabilité post-démarrage.**
* **Peut-être couplé à des alertes (via mail, n8n, Slack, etc.).**
* **Peut générer un fichier CSV ou JSON journalisé.**

**Synthèse des usages**

| **Solution** | **Utilisation** | **Temporalité** | **Niveau** |
| --- | --- | --- | --- |
| **check\_capacitaire.sh** | **Script principal d’audit** | **Manuel ou appelé ailleurs** | **Base** |
| **.service** | **Démarrage automatique au boot** | **À chaque reboot** | **Moyen** |
| **cron** | **Vérification récurrente programmée** | **Toutes les heures/jours** | **Avancé** |

**Une image contenant texte, capture d’écran, menu, document

Le contenu généré par l’IA peut être incorrect.**

**Lors des tests netstat et nmap étaient absents : logique j’ai des restrictions dans le scanning réseau prérogative de l’administrateur et du RSSI.**

**1. ❌ netstat manquant : sudo: netstat: command not found**

**Solution : sudo apt install net-tools -y**

**2. ❌ nmap manquant : sudo: nmap: command not found**

**Solution : sudo apt install nmap -y**

**✅Recommandations immédiates**

| **Outil** | **Commande d’installation** | **Utilité** |
| --- | --- | --- |
| **netstat** | **sudo apt install net-tools** | **Connexions réseau** |
| **nmap** | **sudo apt install nmap** | **Scan et détection de ports** |
| **jq** | **sudo apt install jq *(optionnel)*** | **Parsing JSON si besoin plus tard** |

**Enrichissement du prometheus.yml : ajouts possibles et leur explication.**

**1. scrape\_timeout : définition du temps maximum accordé à la requête**

**scrape\_timeout: 10s**

* **But :** Détermine combien de temps **Prometheus** attend une réponse de la cible avant d’abandonner.
* **Par défaut :** Il est égal à **scrape\_interval,** sauf si tu le définis**.**
* **Pourquoi c’est utile :** Dans certains cas **(réseau lent, exporter distant),** tu veux que **Prometheus** continue à interroger sans provoquer de **timeouts.**

**Recommandé : avoir un scrape\_timeout légèrement inférieur au scrape\_interval pour éviter les chevauchements.**

**2. Ajout d'un job\_label (relabeling) pour unifier les métriques dans Grafana**

**Exemple dans chaque job :**

**Une image contenant Police, horloge, capture d’écran, texte

Le contenu généré par l’IA peut être incorrect.**

* **But :** Uniformiser le nom de la métrique sous le label job au lieu de **\_\_job\_\_.**
* **Utile dans Grafana :** pour filtrer ou agréger dynamiquement.

**3. Ajout d’un tls\_config dans les modules Blackbox Exporter**

**Si tu sondes des cibles HTTPS internes avec certificats autosignés :**

**Une image contenant texte, Police, horloge, conception

Le contenu généré par l’IA peut être incorrect.**

* **But : Évite les erreurs SSL pour les certificats non valides.**
* **⚠️ À n'utiliser que si la cible est maîtrisée (interne)**

**4. Ajout de honor\_labels: true pour préserver les labels d’origine**

**honor\_labels: true**

* **But :** Si tes cibles exportent déjà des **labels (ex : instance, job), Prometheus** ne les écrasera pas**.**
* **Utile :** si tu ajoutes manuellement des **labels:** dans **static\_configs.**

1. **Ajout de targets dynamiques via fichiers externes**

**Une image contenant texte, Police, horloge, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

* **But :** Tu peux déclarer des cibles dynamiquement dans des fichiers séparés sans éditer **prometheus.yml.**
* **Intérêt :** idéal pour la maintenance, pour injecter automatiquement des IPsgénérées par un script**.**

1. **Ajout de metric\_relabel\_configs pour filtrer ou renommer des métriques**

**Une image contenant texte, Police, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

* **But :** Supprimer des métriques inutiles **(par exemple go\_\* de Prometheus lui-même).**
* **Gain** : réduit la taille des séries stockées.

**Jour-18 Tercium 4ième semaine | PKI : Grafana – Prometheus |** **Tests de charge / suivi des logs / maintenance du SI**

* **Fin de la phase paramétrage avec la mise au point des automatisations pour le démarrage système complet et des scripts automatisés des logs.**
* **Moment nécessaire avant de tester la résistance aux charges et aux failles quelles peuvent conduire.**
* **De concert avec ces tests installations des outils de sécurité liées aux effets de seuil du au charges.**

**1. Intérêt d’activer Alertmanager**

**Fonction : :  
Alertmanager** reçoit les alertes générées par **Prometheus (via rule\_files)** et les traite selon des **règles :** envoi **d’emails, webhook**, silences programmés, groupements, etc**.**

**Cas d’usage concret :**

* **Notifier un administrateur si un service critique (blackbox\_exporter\_http, ssh, dns, icmp, etc.) échoue** pendant plus de 1 minute.
* **Grouper les alertes similaires (ex. : 5 probes HTTP échouent)** pour éviter le spam**.**
* **Définir un silence** pour éviter des alertes pendant des opérations de maintenance**.**

**2. Activation dans prometheus.yml avec la section suivante :**

**Une image contenant texte, Police, capture d’écran, conception

Le contenu généré par l’IA peut être incorrect.**

**Ajouter une autosurveillance explicite via ce job (valable si Alertmanager tourne) : Une image contenant texte, Police, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

**3. Effets et configuration des paramètres avancés (HTTP) : fichier yml en réserve.**

**4. Effets réseau et sécurité**

| **Paramètre** | **Effet attendu** | **Risques ou bénéfices** |
| --- | --- | --- |
| **scrape\_timeout** | **Stoppe une requête si elle prend trop de temps** | **Évite de bloquer un thread Prometheus inutilement** |
| **scheme: https** | **Force l’usage d’un canal chiffré** | **Utile avec Blackbox + sites publics** |
| **tls\_config.insecure\_skip\_verify: false** | **Vérifie le certificat SSL** | **Empêche l’acceptation de certificats non valides** |
| **tls\_config.min\_version: TLS12** | **Bloque TLS obsolète (vulnérable à POODLE, BEAST…)** | **Sécurisation forte** |

**5. Prochaine étape générer directement le premier Makefile Prometheus+Exporters.**

**Makefile : installation, vérification et démarrage complet Prometheus + Exporters**

**Une image contenant texte, capture d’écran, conception

Le contenu généré par l’IA peut être incorrect.**

**Fichier complété avec les sondage des ports : Makefile**

**Emplacement : Racine de ton dépôt de supervision  
Rôle : Automatiser toutes les tâches d’administration Prometheus  
Pré-requis : Les scripts .sh doivent être dans le même dossier**

**Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.**

**Structure minimale attendue :**

**/monprojet/**

**├── Makefile**

**├── install\_prometheus.sh**

**├── install\_exporters.sh**

**├── check\_and\_restart\_services.sh**

**├── verify\_ports\_exporters.sh**

**└── prometheus/**

**├── prometheus.yml**

**└── alert.rules.yml**

**Étapes typiques :**

1. **Installation : cd : make install\_prometheus make install\_exporters**
2. **Validation : cd : make validate**
3. **Redémarrage ou rechargement : cd : make reload**
4. **Vérifications :** **cd : make check\_services make verify**

**Production de :**

* **« Page de démarrage ».**
* **« Choix opt ou home ou usr ».**
* **« Problème test http échoue vs test curl valide ».**
* **« Processus de sécurité Jitsi-Meet ».**

**Jour-19 Tercium 4ième semaine | PKI : Grafana – Prometheus |** **Tests de charge / suivi des logs / maintenance du SI**

**Programme du jour :**

1. **Résoudre les problèmes de chemin récurrent / fait.**

Version graduée du fichier **node\_exporter.service** selon **trois niveaux de sécurité** croissante, à appliquer selon ton besoin **(phase de test, préproduction, production durcie).**

**Voir les fichiers extension ini**

**1. Version LÉGÈRE (débogage ou test local)**

**Objectif : garantir le bon fonctionnement, sans restriction.**

**2. Version STANDARD (préproduction sécurisée)**

**Objectif : bloquer certaines escalades tout en laissant le système fonctionner.**

**WantedBy=multi-user.target**

**3. Version DURCIE (production + système critique)**

**Objectif : appliquer une sécurité stricte, avec verrouillage des accès à certaines parties du système.**

**Recommandations :**

1. **On commence avec la version LÉGÈRE.**
2. **On vérifie avec : cd : sudo systemctl daemon-reload**

**sudo systemctl restart node\_exporter**

**sudo systemctl status node\_exporter Puis cd : curl** [**http://localhost:9101/metrics**](http://localhost:9101/metrics)

**On monte progressivement vers la version STANDARD, puis DURCIE si tout est stable.**

1. **Installer Wazu et Suricata et Fail2Ban.**

**Voici la procédure recommandée pour installer et configurer Fail2Ban, Wazuh, et Suricata, dans le cadre d’un usage avec Apache JMeter pour tests de charge et de résilience :**

**1. Installation de Fail2Ban**

**Fail2Ban protège contre les attaques par brute-force (SSH, HTTP, etc.)**

**cd : sudo apt update && sudo apt install fail2ban -y**

**Configuration minimale :**

**cd : sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local**

**sudo nano /etc/fail2ban/jail.local**

**Active au minimum :**

**Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.**

**Redémarrage :**

**cd : sudo systemctl enable --now fail2ban**

**sudo systemctl status fail2ban**

**2. Installation de Wazuh (Agent Only)**

**Wazuh est un agent SIEM (Security Information & Event Management) très puissant.**

**Installation rapide de l’agent :**

**cd : curl -s https://packages.wazuh.com/install.sh | sudo bash**

**Configure ensuite : /var/ossec/etc/ossec.conf**

**Pour pointer vers ton serveur Wazuh si tu en as un, ou le local sinon.**

**Démarrage :**

**cd : sudo systemctl enable --now wazuh-agent**

**sudo systemctl status wazuh-agent**

**3. Installation de Suricata**

**Suricata est un IDS/IPS réseau, très utile pour analyser les paquets pendant un stress test.**

**cd : sudo apt install suricata -y**

**Vérification de l'interface réseau :**

**cd : ip a**

**Puis modifie /etc/suricata/suricata.yaml :**

**Démarrage :**

**cd : sudo systemctl enable --now suricata**

**sudo systemctl status suricata**

**Intégration avec Apache JMeter**

1. **Stress test : JMeter simule une charge réseau (HTTP, TCP…).**
2. **Monitoring live : Prometheus + Grafana visualisent les impacts système.**
3. **Logs et alertes :**
   * **Fail2Ban : bloque si trop de connexions échouées.**
   * **Suricata : détecte signatures d’attaques.**
   * **Wazuh : centralise les événements.**

**Attention à l’installation conflit avec apt car j’ai tout installé /opt/ des binary.**

**1. Nettoyage préalable (optionnel mais conseillé)**

**🔸 Supprimer les installations apt résiduelles**

**cd : sudo apt remove --purge prometheus prometheus-node-exporter -y cd : sudo apt autoremove --purge -y cd : sudo rm -rf /etc/prometheus cd : sudo rm -rf /var/lib/prometheus**

**🔸 Supprimer les utilisateurs systèmes conflictuels (⚠️ prudence) cd : sudo deluser prometheus cd : sudo rm -rf /home/prometheus**

**2. Téléchargement des binaires officiels cd :**

**🔸 Prometheus**

**Objectif : Installation propre de Prometheus + Exporters en /opt/**

**Évite les conflits avec apt et garantit la maîtrise du système (mises à jour manuelles, sécurité renforcée, services stables).**

**1. Arborescence standard**

| **Composant** | **Emplacement recommandé** | **Commentaire** |
| --- | --- | --- |
| **Prometheus** | **/opt/prometheus/** | **Binaire + config séparée (/etc/prometheus/)** |
| **node\_exporter** | **/opt/node\_exporter/** | **Surveillance système locale** |
| **blackbox\_exporter** | **/opt/blackbox\_exporter/** | **Surveillance réseau externe** |
| **telegraf (optionnel)** | **/opt/telegraf/** | **Récolte métriques diverses** |
| **Services systemd** | **/etc/systemd/system/** | **Lancement au boot** |
| **Configuration .yml** | **/etc/prometheus/** | **prometheus.yml, alert.rules.yml** |

**2. Risques à éviter**

| **Risque** | **Cause** | **Contournement** |
| --- | --- | --- |
| **Conflit apt** | **apt install prometheus crée un user prometheus incompatible avec /opt** | **Ne jamais utiliser apt pour Prometheus, Node Exporter, etc.** |
| **UID déjà existant** | **Conflit entre apt et user manuel** | **Créer l’utilisateur avec --system --no-create-home au début** |
| **Permissions refusées** | **Mauvais ExecStart dans le service** | **Vérifier le chemin complet + droits chmod +x** |
| **Erreur post-install** | **apt tente d’écrire sur des UID réservés** | **Supprimer apt install et purger : sudo apt purge prometheus\*** |

**3. Étapes d'installation propre de Prometheus**

**a) Téléchargement des binaires :**

**cd /tmp**

**wget https://github.com/prometheus/prometheus/releases/download/v2.45.3/prometheus-2.45.3.linux-amd64.tar.gz**

**tar -xvzf prometheus-2.45.3.linux-amd64.tar.gz**

**sudo mv prometheus-2.45.3.linux-amd64 /opt/prometheus**

**b) Création de l'utilisateur système :**

**cd : sudo useradd --no-create-home --shell /usr/sbin/nologin prometheus**

**c) Droits :**

**cd : sudo chown -R prometheus:prometheus /opt/prometheus**

**sudo mkdir /etc/prometheus**

**sudo chown prometheus:prometheus /etc/prometheus**

**d) Copie des fichiers de configuration :**

**cd : sudo cp /opt/prometheus/prometheus.yml /etc/prometheus/**

**sudo cp -r /opt/prometheus/consoles /etc/prometheus/**

**sudo cp -r /opt/prometheus/console\_libraries /etc/prometheus/**

**4. Service systemd Prometheus**

**Fichier : /etc/systemd/system/prometheus.service**

**Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.**

**Puis :**

**cd : sudo systemctl daemon-reload**

**sudo systemctl enable --now prometheus**

**5. Vérification**

**cd : curl** [**http://localhost:9091**](http://localhost:9091)

**Prometheus doit être reconstruit, suite à la commande apt qui est entré en conflit puis un erase complet a été nécessaire.**

**Construction et explication pas à pas annexe Prometheus.**

1. **Puis finaliser le démarrage automatique de la session.**
2. **Lancement automatique des services Grafana Prometheus.**
3. **Lancement automatiser les collecteurs de logs pour analyse calibrer dur les faiblesses connues de Jitsi-Meet.**

**Jour-20 Tercium 5ième semaine | Réinstaller Prometheus | Création d’un dépôt GitHub pour par la suite** **Tests de charge / suivi des logs / maintenance du SI**

1. **Création d’un dépôt GitHub : Guide & création**

**jitsi-meet-infra/**

**├── .env.template**

**├── .env # (non versionné)**

**├── .gitignore**

**├── Makefile**

**├── Makefile.test**

**├── README.md**

**├── SECURITY.md**

**├── CONFIGURATION.md**

**├── docs/**

**│ └── Dépôt conseils et fichiers.docx**

**├── scripts/**

**│ ├── automate\_keys.sh**

**│ ├── post-deploy.sh**

**│ ├── push\_to\_infomaniak.sh**

**│ ├── rollback.sh**

**├── .github/**

**│ └── workflows/**

**│ ├── deploy.yml**

**│ └── deploy-alt.yml # (optionnel)**

**├── etc/**

**│ ├── jitsi/**

**│ ├── prosody/**

**│ └── nginx/**

**└── opt/**

**└── jitsi/**

**Voir Annexe Guide Github.**

1. **Finir l’installation de Prometheus.**

**La liste complète des fichiers Prometheus à restaurer ou réinstaller proprement, avec leurs emplacements standard sur un système Linux (ex : Ubuntu Server), structurée par rôle :**

**FICHIERS ET DOSSIERS PROMETHEUS – STRUCTURE STANDARD**

**1. Dossier principal de configuration**

**Emplacement :**

**/etc/prometheus/**

**Contenu essentiel :**

| **Fichier** | **Rôle** |
| --- | --- |
| **prometheus.yml** | **Fichier principal de configuration (scrape configs, alert rules)** |
| **alert.rules.yml** | **(Facultatif) Règles d’alerte Prometheus si rule\_files est utilisé** |
| **blackbox.yml** | **(Facultatif) Configuration de modules pour blackbox\_exporter** |
| **web.yml** | **(Facultatif) Configuration du reverse proxy ou HTTPS/TLS** |

**2. Répertoires système à vérifier / créer**

| **Répertoire** | **Rôle** |
| --- | --- |
| **/etc/prometheus/** | **Configs (décrites ci-dessus)** |
| **/var/lib/prometheus/** | **Données (base TSDB locale)** |
| **/var/log/prometheus/** | **Logs (optionnel, si configuré manuellement)** |

1. **Exporters (agents de collecte de métriques)**

**node\_exporter (système de base)**

* **Fichier binaire : /usr/local/bin/node\_exporter**
* **Service systemd : /etc/systemd/system/node\_exporter.service**

**blackbox\_exporter (HTTP, ping, TCP…)**

* **Binaire : /usr/local/bin/blackbox\_exporter**
* **Config : /etc/prometheus/blackbox.yml**
* **Service : /etc/systemd/system/blackbox\_exporter.service**

**telegraf (si utilisé pour enrichir les métriques)**

* **Config : /etc/telegraf/telegraf.conf**
* **Service : telegraf.service**

1. **Dashboards Grafana (JSON)**

**Si tu sauvegardes des dashboards :**

* **Emplacement recommandé de versionnement :**

**/opt/prometheus-grafana-dashboards/**

* **Exemple :**
  + **dashboard\_node\_exporter.json**
  + **dashboard\_blackbox\_tcp\_http.json**

**Fichiers de service systemd à restaurer**

| **Fichier systemd** | **Service contrôlé** |
| --- | --- |
| **/etc/systemd/system/prometheus.service** | **Lancement Prometheus** |
| **/etc/systemd/system/node\_exporter.service** | **Lancement node\_exporter** |
| **/etc/systemd/system/blackbox\_exporter.service** | **Lancement blackbox\_exporter** |

**Étapes de restauration minimale**

**# 1. Recréer le dossier**

**sudo mkdir -p /etc/prometheus /var/lib/prometheus**

**# 2. Copier les fichiers**

**sudo cp prometheus.yml /etc/prometheus/**

**sudo cp alert.rules.yml /etc/prometheus/**

**sudo cp blackbox.yml /etc/prometheus/**

**# 3. Vérifier les permissions**

**sudo chown -R prometheus:prometheus /etc/prometheus /var/lib/prometheus**

**# 4. Activer le service**

**sudo systemctl daemon-reexec**

**sudo systemctl restart prometheus.service**

**sudo systemctl status prometheus**

1. **Puis finaliser le démarrage automatique de la session.**
2. **Lancement automatique des services Grafana Prometheus.**
3. **Lancement automatiser les collecteurs de logs pour analyser, calibrer détecter les faiblesses connues de Jitsi-Meet.**

**Jour-20 Tercium 5ième semaine | Réinstaller Prometheus | Création d’un multi dépôt GitHub | Installation de la suite de sécurité**  **Tests de charge / suivi des logs / maintenance du SI**

J'ai des snapshots sur l'instance cela serait judicieux de les rappeller pour relancer prometheus complétement sans tout réinstaller. Stupide de n'y pas avoir penser plus tôt. En l'écrivant je réponds à la question, je suis bien un Igor du fameux Baron Franckenstein. Secondo je pourrais donc rapatrier ce snapshot dans le dépôt statique ou dynamique de github.

Raisonnement logique, l’on doit raisonner ici comme un architecte d’infrastructure pragmatique. Si un snapshot fonctionnel existe sur l’instance, alors il représente un point de restauration fiable, souvent plus rapide et cohérent qu'une réinstallation manuelle complète.

**Réponses directes à ces deux questions**

**1. Relancer Prometheus depuis un snapshot ?**

**Oui, c’est pertinent.**

**Avantages :**

* Gain de temps majeur (services, permissions, ports déjà validés)
* Réduction du risque d’erreur humaine
* Réactivation possible même sans réseau si snapshot local
* Compatible avec une relance post-nettoyage de dépendances conflictuelles **(Telegraf, Wazuh...)**

**Procédure proposée :**

* **Restaurer uniquement les dossiers ciblés :**
  + **/etc/prometheus/**
  + **/opt/prometheus/ *(ou autre chemin si déplacé)***
* **Redémarrer les services : cd : sudo systemctl daemon-reexec cd : sudo systemctl restart prometheus cd : sudo systemctl restart prometheus-node-exporter**
* **Vérifier : cd : curl -s http://localhost:9091/metrics | grep '^up'**

**S’assurer que :**

* Les fichiers **prometheus.yml, alert.rules.yml** et **blackbox.yml** sont présents.
* L’utilisateur **prometheus** a bien les droits.
* Le port **(9091 ou autre)** est libre.

**2. Exporter ce snapshot vers le dépôt GitHub ?**

**Oui, pour le dépôt statique uniquement.**

**Mais :**

* **Jamais dans le dépôt dynamique, car :**
  + **Il doit rester léger, lisible, CI/CD friendly**
  + **Des fichiers lourds / binaires casseraient la logique GitOps**

**Procédure :**

* Monter le snapshot sur **/mnt/snapshot\_restore/**
* Copier uniquement les fichiers configurables :



* Ajouter un fichier **README\_RESTORE.md** dans **jitsi-meet-snapshot/ :**

**Ce snapshot a été extrait depuis l’instance Infomaniak via une restauration système (date, UUID, etc.). Les fichiers sont utilisables pour une réactivation complète de Prometheus sans réinstallation.**

**Résumé opérationnel**

| **Action** | **Dépôt concerné** | **But** |
| --- | --- | --- |
| **Restaurer Prometheus via snapshot** | **Aucun (sur instance)** | **Réactiver la supervision rapidement** |
| **Exporter la conf vers GitHub** | **jitsi-meet-snapshot** | **Archiver proprement la version stable** |
| **Exclure ces fichiers du dynamique** | **jitsi-meet-infra** | **Ne pas surcharger le CI/CD pipeline** |