



DAT

SAE 5.CYBER.03

2025

RÉALISÉ PAR :

BARBIER MAXENS

BOILLY HUGO

CALPETARD MAHÉ

HOUTMANN CYLIAN

RÉALISÉ POUR :

BLUE WAVE LOGISTIC

Sommaire

Sommaire	2
Découpages des lots	4
Architecture sécurisée	4
Objectif : Construire une architecture fiable, scalable et sécurisée	4
Plateforme SOC	4
Audit de sécurité	5
Gouvernance & Gestion des risques	5
Topologie	6
Schéma	6
Plan d'adressage	6
Attributions de projets	8
Windows AD	9
Wordpress	12
Serveur Samba	16
PfSense (Firewall)	19
1. Configuration des Accès	19
2. Justification des Restrictions et Accès	20
3. Conclusion	20
Installation de Mercator sur un serveur Ubuntu	22
1. Mise à jour de la distribution Linux	22
2. Installation de PHP et des bibliothèques PHP	22
3. Installation d'Apache2, Git, Graphviz et Composer	22
4. Création du répertoire du projet	22
5. Clonage du projet depuis GitHub	23
6. Installation des packages avec Composer	23
7. Installation de MySQL	23
8. Création et configuration de la base de données	23
9. Configuration des variables d'environnement	23
10. Exécution des migrations et génération de la clé de l'application	24
11. Importation de la base de données CPE	24
12. Démarrage de l'application	24
13. Configuration Apache	25
14. Interface Web	25
Wazuh	28

Deming	31
Introduction	31
Prérequis	31
1. Installation du Système	31
2. Création du Répertoire de Projet	32
3. Création des Répertoires Temporaires	32
4. Installation des Paquets via Composer	33
5. Installation de MariaDB	33
6. Configuration de l'Application	33
7. Création de la Base de Données	34
8. Peuplement de la Base de Données	35
9. Démarrage de l'Application	35
10. Connexion à l'Application	35
Conclusion	36
Apache Guacamole (Bastion)	37
1. Installation des prérequis	37
2. Compilation et installation d'Apache Guacamole Server	37
3. Configuration de Guacamole Client (Web App)	38
4. Compilation et installation d'Apache Guacamole Server	39
5. Création du répertoire de configuration	41
6. Installation de Guacamole Client (Web App)	41
7. Base de données MariaDB pour l'authentification	42
8. Connexion à Apache Guacamole	43
A. Création d'un compte administrateur sécurisé	44
B. Ajouter une connexion RDP	45
Conclusion	48

Découpages des lots

Architecture sécurisée

Objectif : Construire une architecture fiable, scalable et sécurisée

- Définir un plan d'adressage
- Définir les interfaces de connexion
- Étudier les frameworks et les méthodologies existants (CIS, NIST, etc.)
- Analyser des cas concrets d'architectures sécurisées réussies
- Adopter des mesures de sécurité adaptée
- Étudier les menaces émergentes et les contre-mesures associées
- Configurer les différents services composants l'infrastructure

Plateforme SOC

Objectif : Avoir la capacité de réagir face à la menace

- Proposer des contre-mesures face à une menace
- Détecter et modérer efficacement les contenus qui violent les règles de la communauté
- Identifier rapidement les comptes utilisateurs qui ont été compromis et prendre les mesures nécessaires pour protéger les données des utilisateurs
- Détecter et atténuer les attaques DDoS qui visent à rendre la plateforme indisponible
- Élaborer et mettre en œuvre des plans de réponse aux incidents
- Mettre en place des mécanismes de sauvegarde et de restauration efficaces pour limiter les pertes de données en cas d'incident
- Définir des alertes grâce aux outils de supervision

Audit de sécurité

Objectif : Contrôler, durcir son système d'information

- Tester et vérifier la robustesse de notre l'infrastructure
 - Réaliser des tests de pénétration
 - Utiliser des outils d'audit comme Lynis, OpenVAS, ou Nessus pour évaluer les vulnérabilités.
- Contrôler et durcir le système :
 - Implémenter une politique de durcissement des systèmes basée sur les guides CIS ou STIG (Security Technical Implementation Guides).
- Déterminer précisément les objectifs de l'audit (évaluation globale, conformité à une norme, etc.).
- Définir les personnes impliquées dans l'audit (équipe technique, direction, etc.).
- Examiner les logs des systèmes pour détecter d'éventuelles anomalies.

Gouvernance & Gestion des risques

Objectif : Maîtriser les risques de son système d'information

- Identifier les risques potentiels
- Recenser tous les événements potentiels qui pourraient avoir un impact négatif sur le système d'information (cyberattaques, pannes matérielles, erreurs humaines, etc.).
- Évaluer la probabilité et l'impact de chaque risque.
- Mettre en place des mesures de prévention (contrôles d'accès, pare-feu, etc.).
- Mettre en place des mesures de détection (systèmes de surveillance, alertes).
- Mettre en place des mesures de réponse (plans de reprise d'activité, etc.).
- Mettre en place des indicateurs de performance (KPI).
- Réaliser des audits réguliers.
- Ajuster les mesures en fonction des résultats.



Plan d'adressage

Voici notre plan d'adressage pour ce projet :

Équipement	Interface	VLAN	IP Address	Gateway	Description
Firewall	eth0.1	1	10.10.1.1/24	-	Gateway pour le bastion
	eth0.10	10	10.10.10.1/24	-	Gateway pour WordPress
	eth0.20	20	10.10.20.1/24	-	Gateway pour AD/Samba
	eth0.30	30	10.10.30.1/24	-	Gateway pour SOC Services
	eth0.40	40	10.10.40.1/24	-	Gateway pour les PC Clients
Bastion	eth0	1	10.10.1.10/24	10.10.1.1	Gestion des accès
WordPress Server	eth0	10	10.10.10.10/24	10.10.10.1	Serveur WordPress
Active Directory (AD)	eth0	20	10.10.20.10/24	10.10.20.1	Serveur Active Directory
Samba Server	eth0	20	10.10.20.20/24	10.10.20.1	Serveur Samba
Wazuh	eth0	30	10.10.30.10/24	10.10.30.1	Outil de supervision (Wazuh)
CrowdSec	eth0	30	10.10.30.20/24	10.10.30.1	Outil de sécurité (CrowdSec)
Mercator	eth0	30	10.10.30.30/24	10.10.30.1	
Deming	eth0	30	10.10.30.40/24	10.10.30.1	

PC Client 1	eth0	40	10.10.40.10/24	10.10.40.1	Premier PC client
PC Client 2	eth0	40	10.10.40.20/24	10.10.40.1	Deuxième PC client

Attributions des rôles

 Nom	 Rôle
Maxens BARBIER	Analyste de Sécurité ▾
Hugo BOILLY	Chef de Projet ▾
Mahé CALPETARD	Analyste sécurité ▾
Cylian-Nataï HOUTMANN	Analyste Gouvernance ▾

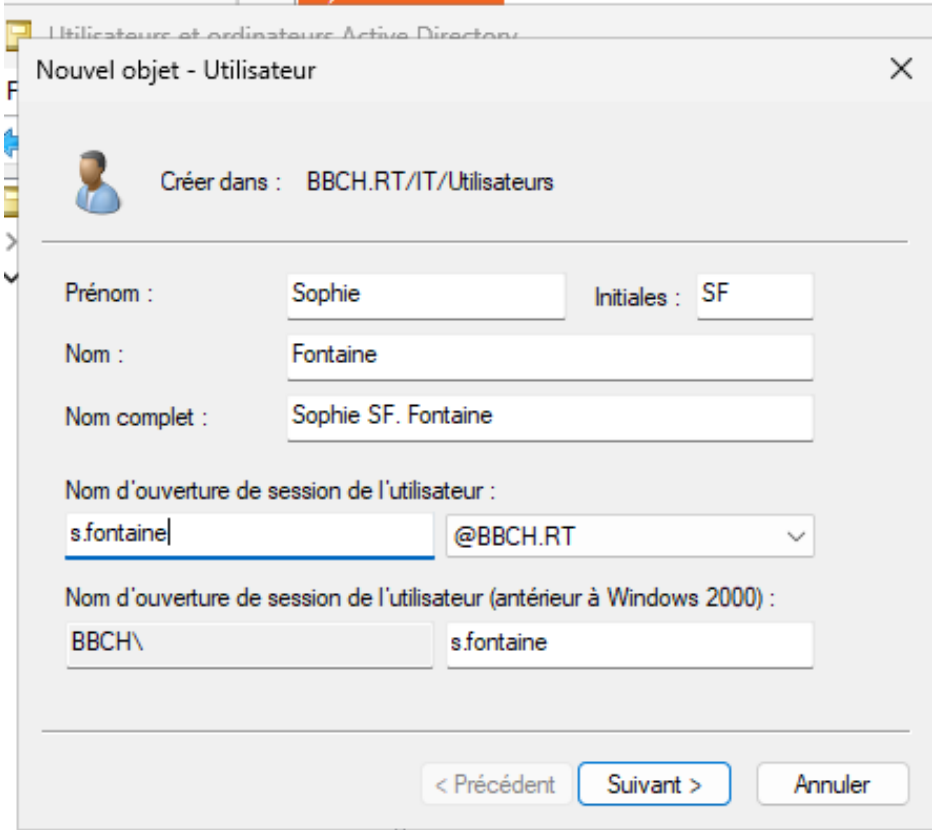
Windows AD


Pour installer L'ACTIVE DIRECTORY, nous suivons la procédure classique à savoir :

1. **Ajouter des rôles et fonctionnalités.**
2. **Ajouter le rôle AD DS (Active Directory Domain Services) :**
3. **Promouvoir ce serveur en contrôleur de domaine.**
4. **Configurer les options du domaine :**
 - Définir le **nom du domaine** (BBCH.RT).
 - Configurer les niveaux fonctionnels et les rôles du contrôleur de domaine (DNS, catalogue global).

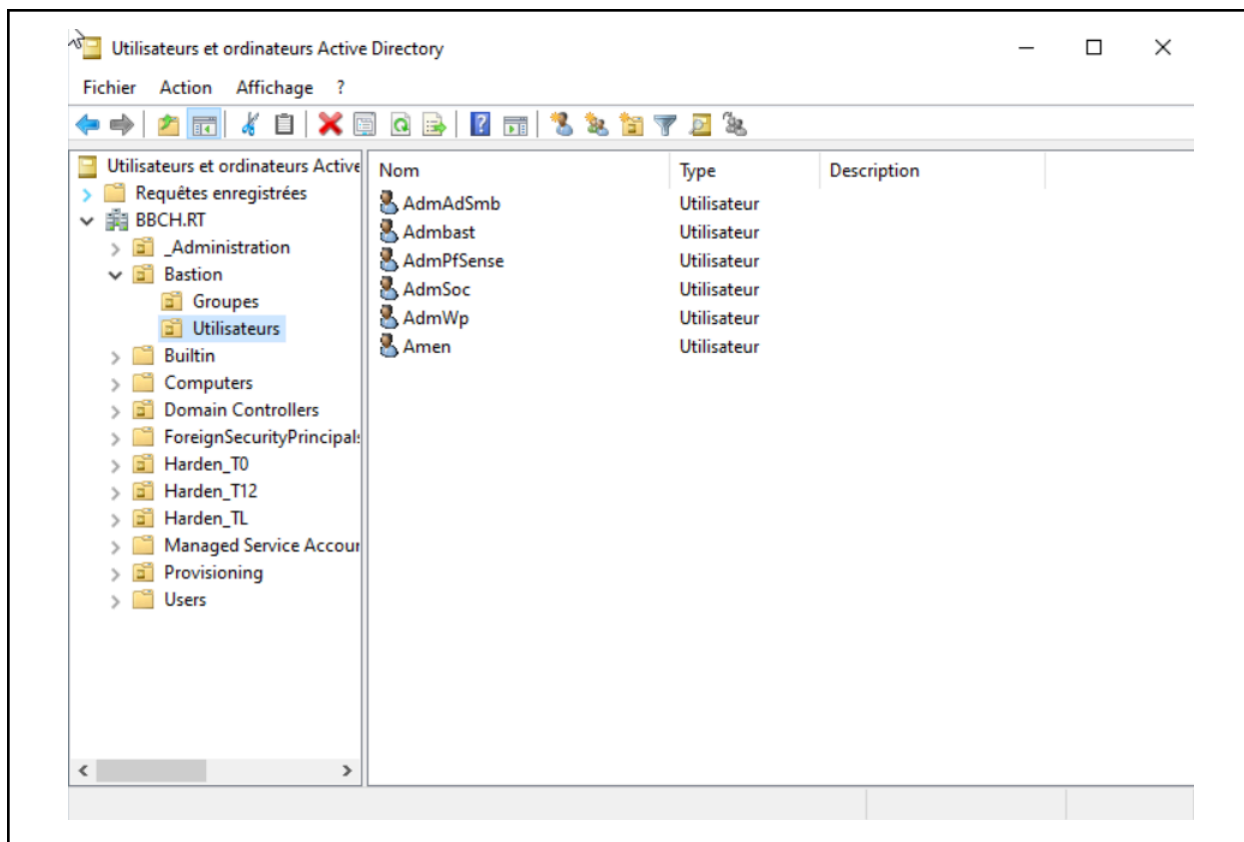
Nom de domaine	BBCH.RT
Mot de passe Administrateur	Root@974
Mot de passe de restauration	Root@974

Exemple d'un utilisateur de l'AD :

s.fontaine	Patate@456
 <p>Utilisateurs et ordinateurs Active Directory</p> <p>Nouvel objet - Utilisateur</p> <p>Créer dans : BBCH.RT/IT/Utilisateurs</p> <p>Prénom : Sophie Initiales : SF</p> <p>Nom : Fontaine</p> <p>Nom complet : Sophie SF. Fontaine</p> <p>Nom d'ouverture de session de l'utilisateur : s.fontaine @BBCH.RT</p> <p>Nom d'ouverture de session de l'utilisateur (antérieur à Windows 2000) : BBCH\s.fontaine</p> <p>< Précédent Suivant > Annuler</p>	

t.robort	Tapape@789
<div data-bbox="311 327 1310 1188"><div>Nouvel objet - Utilisateur ✕</div><div> Créer dans : BBCH.RT/IT/Utilisateurs</div><hr/><div>Prénom : <input type="text" value="Thalia"/> Initiales : <input type="text" value="TR"/></div><div>Nom : <input type="text" value="Robert"/></div><div>Nom complet : <input type="text" value="Thalia TR. Robert"/></div><div>Nom d'ouverture de session de l'utilisateur : <input type="text" value="t.robort"/> <input <span="" type="text" value="@BBCH.RT"/>▼</div><div>Nom d'ouverture de session de l'utilisateur (antérieur à Windows 2000) : <input type="text" value="BBCH\"/><input type="text" value="t.robort"/></div><div>< Précédent Suivant > Annuler</div></div>	

Voici la liste des utilisateurs de l'ACTIVE DIRECTORY :



Ces utilisateurs correspondent aux comptes que nous allons utiliser plus tard dans le bastion.

Ils disposent tous d'un mot de passe sécurisé comme détaillé ci-dessous :

Utilisateur	Mot de passe
AdmBast	B4st!on4dm
AdmAdSmb	4dS4mb4!4dm
AdmWp	W0rdpr3ss!4dm
AdmSoc	Ss0cC!4dm
AdmPfSense	Pfs3ns3!4dm
Amen	Wxcvbn974!

Wordpress

Pour installer Wordpress, nous suivons les installations présentes sur ces liens. Il faut faire dans l'ordre séquentiel :

[Debian 12 Bookworm : PHP 8.2 : Install](#)

[Debian 12 Bookworm : Apache2 : PHP + PHP-FPM](#)

[Debian 12 Bookworm : MariaDB : Install : Server World](#)

[Debian 12 Bookworm : Apache2 : Blog System : WordPress](#) (nous devons modifier quelques choses dans cette installation) :

Nous modifions la timezone pour qu'elle soit :

```
php_value[date.timezone] = Indian/Reunion
```

Pour les mots de passe, nous utilisons les suivants :

	MariaDB	Wordpress
Utilisateur		bluewave
Mot de passe	.Superdurmdp123	PzWj#o9msvZrP8Pn8S

Nous finissons ensuite la configuration de la machine avec les commandes suivantes :

1. Installation des paquets nécessaires

Nous installons les paquets requis :

```
sudo apt install openssl micro
```

2. Configuration réseau

Nous modifions son hostname :

```
echo wordpress > /etc/hostname
```

Nous rajoutons également l'adresse IP de notre AD dans le fichier `/etc/resolv.conf`:

```
micro /etc/resolv.conf  
nameserver 10.10.20.10
```

Nous configurons l'interface réseau avec les paramètres de DNS (AD) et la passerelle :

```
micro /etc/network/interfaces  
  
auto enp1s0  
iface enp1s0 inet static  
    address 10.10.10.10  
    netmask 255.255.255.0  
    gateway 10.10.10.1  
    dns-nameservers 10.10.20.10 #AD IP  
    dns-search bbch.rt
```

3. Configuration SSL pour HTTPS

Nous créons le dossier qui héberge nos clés pour notre site web sécurisé :

```
cd /etc/apache2/  
mkdir ssl/
```

Nous créons ensuite la clé privée et publique :

```
openssl req -x509 -nodes -days 365 -newkey rsa:4096 -out  
/etc/apache2/ssl/server.pem -keyout /etc/apache2/ssl/server.key
```

4. Configuration Apache2

Nous activons les modules nécessaires :

```
a2enmod ssl  
a2enmod rewrite
```

Nous modifions la configuration du site HTTPS :

```
micro /etc/apache2/sites-available/default-ssl.conf

<VirtualHost *:443>
    ServerName wp.bbch.rt
    DocumentRoot /var/www/wordpress

    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/server.pem
    SSLCertificateKeyFile /etc/apache2/ssl/server.key

    <Directory /var/www/wordpress>
        AllowOverride All
    </Directory>
</VirtualHost>
```

Nous configurons également le site HTTP pour rediriger vers HTTPS :

```
micro /etc/apache2/sites-available/000-default.conf

<VirtualHost *:80>
    ServerName wp.bbch.rt
    DocumentRoot /var/www/wordpress

    # Redirection vers HTTPS
    RewriteEngine On
    RewriteCond %{HTTPS} off
    RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]

    <Directory /var/www/html>
        AllowOverride All
    </Directory>
</VirtualHost>
```

Nous activons ensuite la configuration SSL :

```
a2ensite default-ssl
```

5. Configuration WordPress

Nous modifions wp-config.php pour forcer HTTPS en ajoutant ces lignes avant `/* That's all, stop editing! */` :

```
micro /var/www/wordpress/wp-config.php

define('FORCE_SSL_ADMIN', true);
define('WP_HOME', 'https://wp.bbch.rt');
define('WP_SITEURL', 'https://wp.bbch.rt');
```

Nous configurons le fichier .htaccess pour gérer les redirections :

```
micro /var/www/wordpress/.htaccess

# BEGIN WordPress
<IfModule mod_rewrite.c>
RewriteEngine On

# Ajout des règles HTTPS ici
RewriteCond %{HTTPS} off
RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]

RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
RewriteBase /wordpress/
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /wordpress/index.php [L]
</IfModule>
# END WordPress
```


6. Finalisation

Pour appliquer tous ces changements, nous démarrons le service Apache2 :

```
sudo service apache2 restart
```

Serveur Samba

Pour préparer le serveur Samba, nous commençons par modifier ses paramètres réseaux. Nous débutons par son hostname :

```
echo samba > /etc/hostname
```

Nous rajoutons également l'adresse IP de notre AD dans le fichier `/etc/resolv.conf`:

```
micro /etc/resolv.conf
```

```
nameserver 10.10.20.10
```

Nous configurons l'interface réseau avec les paramètres de DNS (AD) et la passerelle :

```
micro /etc/network/interfaces
```

```
auto enp1s0
iface enp1s0 inet static
    address 10.10.20.20
    netmask 255.255.255.0
    gateway 10.10.20.1
    dns-nameservers 10.10.20.10 #AD IP
    dns-search bbch.rt
```

Nous allons ensuite installer et configurer Samba en s'aidant de cette documentation : [Debian 12 Bookworm : Samba : Limited Shared Folder](#)

Nous devons toutefois modifier quelques éléments par rapport au tutoriel original, notamment sur les noms :

Nom du groupe d'utilisateurs	commun
Nom du home	commun

Nom de l'utilisateur	communiste
Mot de passe de l'utilisateur	.Ussr1991

Nous modifions également différemment le fichier smb.conf :

```
micro /etc/samba/smb.conf

workgroup = BBCH
.....

interfaces = 127.0.0.0/8 10.10.0.0/16 eth0 enp1s0
.....

[Commun]
    # require authentication
    security = user
    # specify shared directory
    path = /home/commun
    # allow writing
    writable = yes
    # not allow guest user (nobody)
    guest ok = no
    # allow only [commun] group
    valid users = @commun
    # set group for new files/directories to [commun]
    force group = commun
    # set permission [770] when file created
    force create mode = 770
    # set permission [770] when folder created
    force directory mode = 770
    # inherit permissions from parent folder
    inherit permissions = yes
```

Pour accéder ensuite au partage Samba, nous devons aller dans le chemin `\\10.10.20.20\Commun` en entrant les identifiants donnés.

PfSense (Firewall)

1. Configuration des Accès

Zone Source	Zone Destination	Accès
Zone Utilisateurs	Zone Web	✓ Autorisé
	Zone SOC	✗ Refusé
	Zone Bastion	✗ Refusé
	Zone Service	✗ Refusé
	Zone Internet	✓ Autorisé
Zone Web	Zone Utilisateurs	✗ Refusé
	Zone SOC	✗ Refusé
	Zone Bastion	✗ Refusé
	Zone Service	✗ Refusé
	Zone Internet	✓ Autorisé
Zone SOC	Zone Web	✗ Refusé
	Zone Utilisateurs	✗ Refusé
	Zone Bastion	✗ Refusé
	Zone Service	✗ Refusé
	Zone Internet	✗ Refusé

Zone Source	Zone Destination	Accès
Zone Bastion	Zone Web	✓ Autorisé
	Zone SOC	✓ Autorisé
	Zone Utilisateurs	✓ Autorisé
	Zone Service	✓ Autorisé
	Zone Internet	✗ Refusé
Zone Service	Zone Web	✗ Refusé
	Zone SOC	✗ Refusé
	Zone Utilisateurs	✗ Refusé
	Zone Bastion	✗ Refusé
	Zone Internet	✗ Refusé

2. Justification des Restrictions et Accès

- **Sécurité renforcée pour la Zone Bastion** : L'accès à la zone Bastion est strictement limité pour éviter toute exposition indésirable des systèmes critiques à des attaques externes.
- **Accès contrôlé aux services internes** : Les utilisateurs et certaines zones (comme la Zone Web et la Zone Service) ont un accès limité aux services internes pour renforcer la séparation des rôles.
- **Accès Internet depuis la Zone Web et la Zone Utilisateurs** : Ces zones nécessitent un accès à Internet pour permettre aux utilisateurs d'accéder à des ressources externes, tout en maintenant des restrictions strictes sur les zones sensibles comme le SOC et le Bastion.

3. Conclusion

La configuration des accès réseau dans pfSense a été mise en place pour assurer une sécurité optimale tout en permettant une communication contrôlée entre les différentes zones. Les restrictions sont adaptées pour limiter l'exposition aux risques externes et protéger les zones sensibles tout en permettant l'accès aux ressources nécessaires pour les utilisateurs.

Mercator

Ce document détaille les étapes nécessaires à l'installation de Mercator sur un serveur Ubuntu, en incluant la configuration des dépendances et des services requis.

1. Mise à jour de la distribution Linux

Avant de commencer, nous mettons à jour notre système pour garantir que nous disposons des dernières versions des paquets :

```
sudo apt update && sudo apt upgrade -y
```

2. Installation de PHP et des bibliothèques PHP

Mercator requiert plusieurs extensions PHP. Nous les installons avec la commande suivante :

```
sudo apt install -y php-zip php-curl php-mbstring php-xml php-ldap  
php-soap php-xdebug php-mysql php-gd libapache2-mod-php
```

3. Installation d'Apache2, Git, Graphviz et Composer

Nous installons Apache2 pour l'hébergement du site, Git pour la gestion du code source, Graphviz pour la visualisation de graphes et Composer pour la gestion des dépendances PHP :

```
sudo apt install -y apache2 git graphviz composer
```

4. Création du répertoire du projet

Nous créons le répertoire où Mercator sera installé :

```
cd /var/www  
sudo mkdir mercator  
sudo chown $USER:$GROUP mercator
```

5. Clonage du projet depuis GitHub

Nous téléchargeons le code source de Mercator depuis son dépôt officiel :

```
git clone https://www.github.com/dbarzin/mercator /var/www/mercator
```

6. Installation des packages avec Composer

Nous installons les dépendances requises par Mercator :

```
cd /var/www/mercator  
composer install
```

7. Installation de MySQL

Nous installons MySQL pour la gestion de la base de données :

```
sudo apt install -y mysql-server
```

8. Création et configuration de la base de données

Nous configurons MySQL en créant une base de données et un utilisateur dédié :

```
sudo mysql  
  
Dans le terminal MySQL, nous exécutons les commandes suivantes :  
CREATE DATABASE mercator CHARACTER SET utf8 COLLATE utf8_general_ci;  
CREATE USER 'mercator_user'@'localhost' IDENTIFIED BY 's3cr3t';  
GRANT ALL PRIVILEGES ON mercator.* TO 'mercator_user'@'localhost';  
GRANT PROCESS ON *.* TO 'mercator_user'@'localhost';  
FLUSH PRIVILEGES;  
EXIT;
```

9. Configuration des variables d'environnement

Nous configurons Mercator en créant un fichier `.env` et en ajoutant les informations de connexion à la base de données :

```
cd /var/www/mercator
cp .env.example .env
vi .env
```

Nous modifions les paramètres suivants :

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=mercator
DB_USERNAME=mercator_user
DB_PASSWORD=s3cr3t
```

10. Exécution des migrations et génération de la clé de l'application

Nous exécutons les commandes suivantes pour configurer la base de données et sécuriser l'application :

```
php artisan migrate --seed
php artisan key:generate
php artisan config:clear
```

11. Importation de la base de données CPE

Nous importons une base de données existante :

```
gzip -d mercator_cpe.sql.gz
sudo mysql mercator < mercator_cpe.sql
```

12. Démarrage de l'application

Nous démarrons Mercator en mode développement :

```
php artisan serve
```

13. Configuration Apache

Nous modifions les permissions du répertoire Mercator et créons un fichier de configuration Apache :

```
sudo chown -R www-data:www-data /var/www/mercator  
sudo chmod -R 775 /var/www/mercator/storage  
sudo vi /etc/apache2/sites-available/mercator.conf
```

Nous ajoutons la configuration suivante :

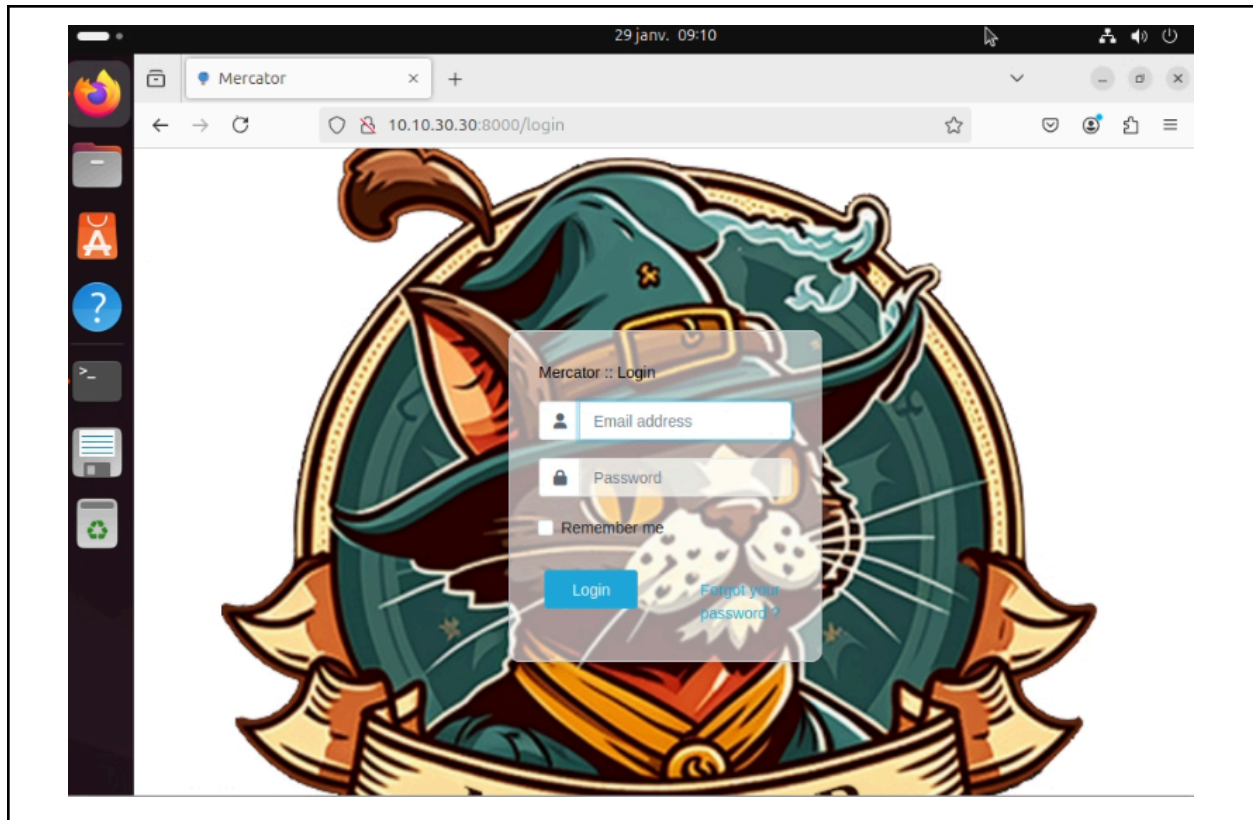
```
<VirtualHost *:80>  
    ServerName mercator.local  
    DocumentRoot /var/www/mercator/public  
    <Directory /var/www/mercator>  
        AllowOverride All  
    </Directory>  
</VirtualHost>
```

Nous activons le site et redémarrons Apache :

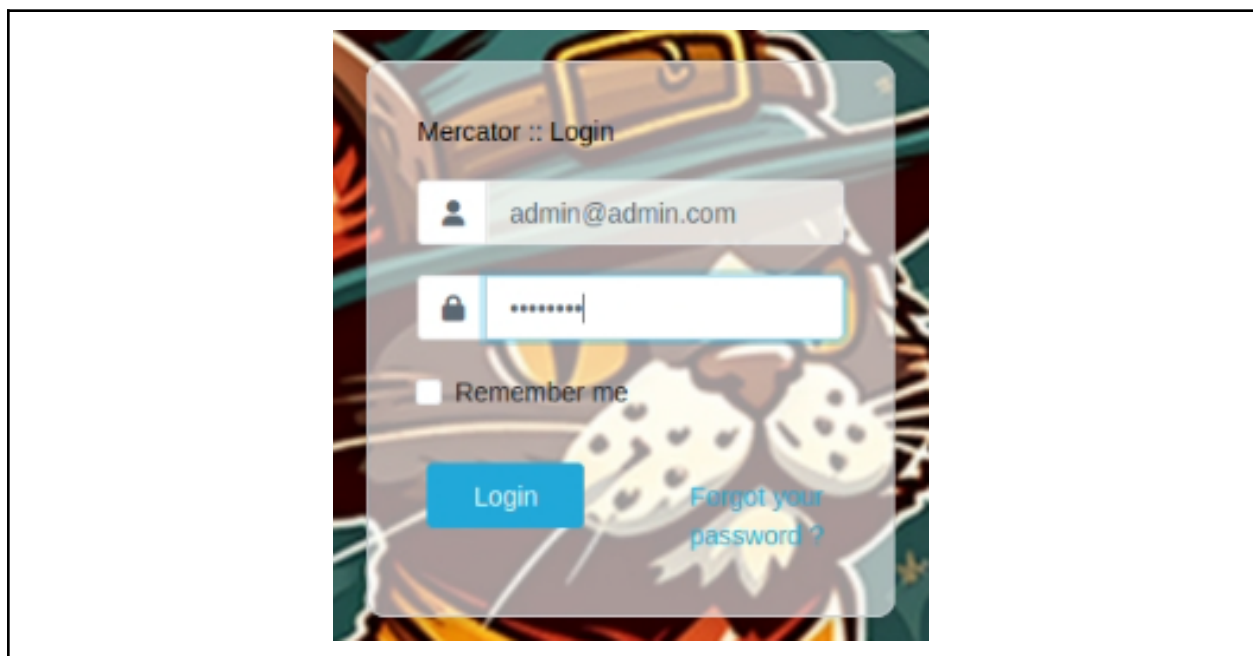
```
sudo a2enmod rewrite  
sudo a2dissite 000-default.conf  
sudo a2ensite mercator.conf  
sudo systemctl restart apache2
```

14. Interface Web

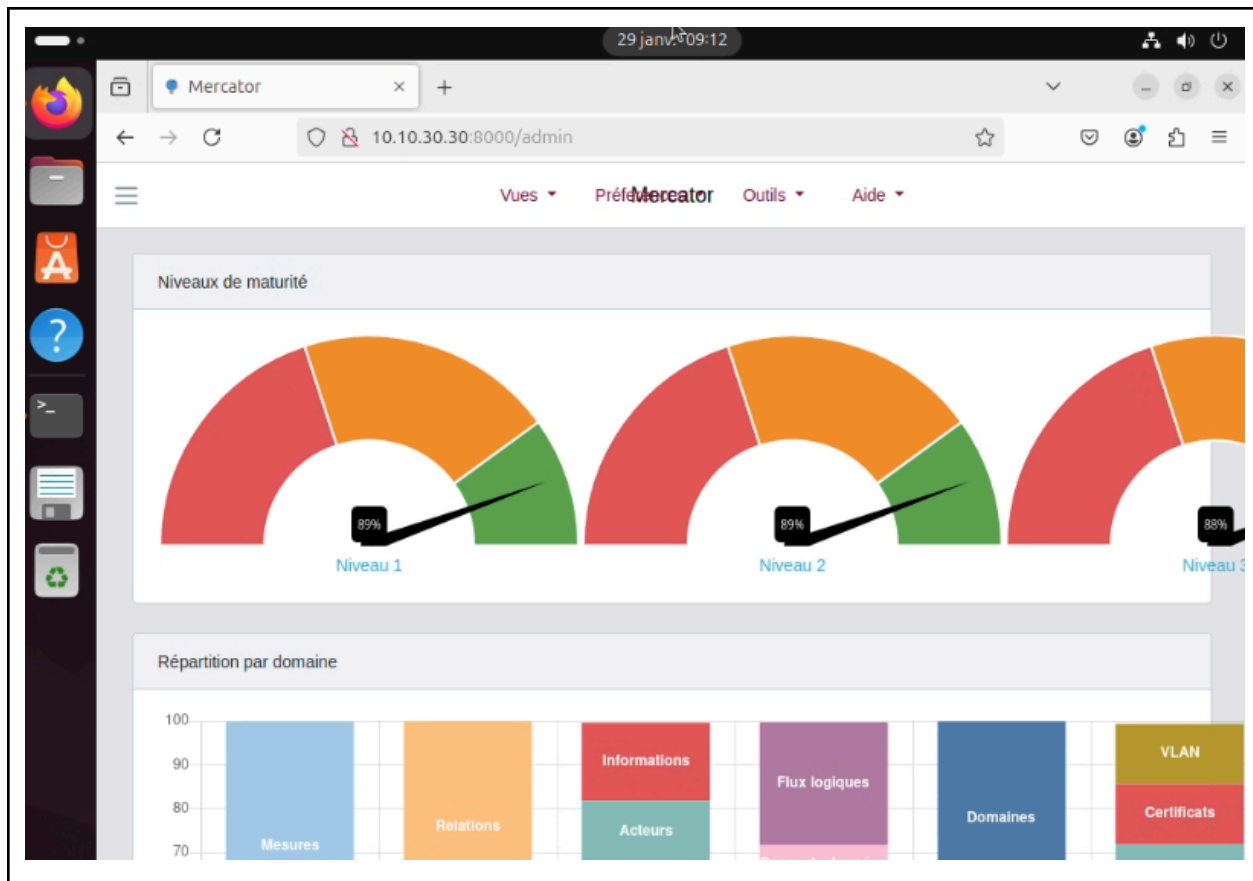
Nous nous rendons donc sur l'adresse suivante: <http://10.10.30.30:8000>



Nous nous connectons à mercator grâce au compte créé:



Voici le dashboard de Mercator auquel nous pouvons maintenant accéder :



Wazuh

Afin de déployer Wazuh, nous utiliserons un serveur Ubuntu. Pour l'installer, cette commande est à exécuter et nous guidera tout au long du processus d'installation :

```
curl -sO https://packages.wazuh.com/4.9/wazuh-install.sh && sudo bash ./wazuh-install.sh -a
```

À la fin, il est crucial de conserver le mot de passe administrateur généré à l'issue de cette procédure pour éviter d'avoir à le réinitialiser. Pour retrouver des informations additionnelles, le quickstart de Wazuh se révèle utile :

<https://documentation.wazuh.com/current/quickstart.html>

Nous devons ensuite configurer l'adresse IP du serveur Ubuntu. Pour cela, dans le dossier `/etc/netplan`, nous modifions le fichier `.yaml` déjà présent :

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      addresses:
        - 192.168.10.30/24
      routes:
        - to: default
          via: 192.168.10.10
      nameservers:
        addresses:
          - 8.8.8.8
          - 1.1.1.1
```

Nous devons ensuite enrôler les clients dans le serveur. Pour cela nous nous renseignons en utilisant cette page de documentation.

<https://documentation.wazuh.com/current/installation-guide/wazuh-agent/index.html>

Pour ajouter des agents sur le Wazuh, nous exécutons ces commandes suivantes (en root) :

```
cd /var/ossec/bin/  
./manage-agents
```

1. Nous voulons ajouter un agent, nous entrons donc A.
2. Nous entrons un nom pour l'agent.
3. L'adresse IP de la machine.
4. Et nous confirmons les données entrées.

Il faut retenir l'ID de cet agent, nous en aurons besoin juste après :

```
*****  
* Wazuh v4.9.2 Agent manager. *  
* The following options are available: *  
*****  
  (A)dd an agent (A).  
  (E)xtract key for an agent (E).  
  (L)ist already added agents (L).  
  (R)emove an agent (R).  
  (Q)uit.  
Choose your action: A,E,L,R or Q: a 1  
  
- Adding a new agent (use '\q' to return to the main menu).  
  Please provide the following:  
    * A name for the new agent: Test 2  
    * The IP Address of the new agent: 192.168.10.111 3  
Confirm adding it?(y/n): y 4  
Agent added with ID 003.
```

Nous avons ensuite besoin de récupérer la clé que nous mettrons dans l'agent installé sur la machine :

1. Nous entrons cette fois **E**, pour extraire la clé.
2. Nous sélectionnons l'agent.
3. Et nous copions la clé pour qu'elle soit mise dans l'agent.

Voici un aperçu de cette clé :

```

*****
* Wazuh v4.9.2 Agent manager.          *
* The following options are available: *
*****
(A)dd an agent (A).
(E)xtract key for an agent (E).
(L)ist already added agents (L).
(R)emove an agent (R).
(Q)uit.
Choose your action: A,E,L,R or Q: E 1

Available agents:
ID: 002, Name: AD, IP: 192.168.10.20
→ ID: 003, Name: Test, IP: 192.168.10.111
Provide the ID of the agent to extract the key (or '\q' to quit): 003 2

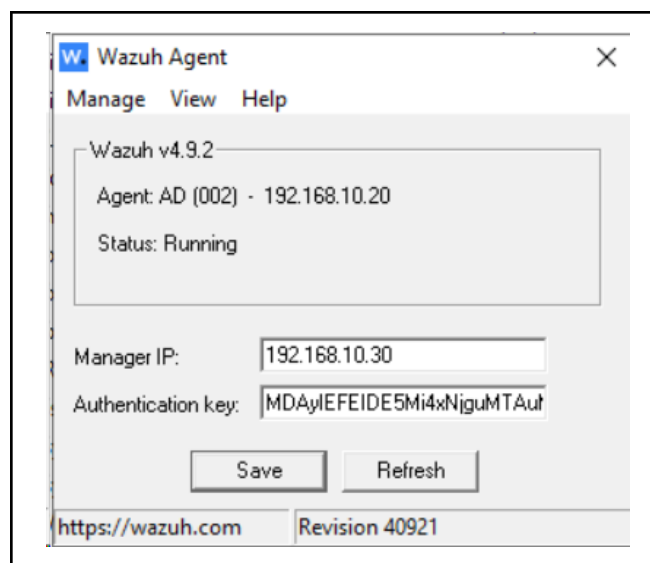
Agent key information for '003' is:
MDAzIFR1c3QgMTkyLjE2OC4xMC4xMTEgYTtyMDNmZDkzOGJkZGJhYTtyMDMwZWVlbnJFjNmI4YTZkY2Y2YWx0DEw0DQ0NzA4ZTA3NGEzZQ==

```

Du côté client , il suffit d'installer Wazuh agent sur le site officiel :

<https://documentation.wazuh.com/current/installation-guide/wazuh-agent/wazuh-agent-package-windows.html>

Puis ensuite, il suffit d'ouvrir l'interface graphique et de rentrer l'adresse **IP** du manager ainsi que la clé obtenu ci-dessus :



Pour importer un nouvel agent dans le système Wazuh pour ajouter ou connecter un nouvel agent au serveur Wazuh, il suffit d'exécuter cette commande :

```
/var/ossec/bin/manage-agents -i <key>
```

Deming

Introduction

Dans cette documentation, nous allons détailler les étapes nécessaires pour installer et configurer l'application Deming sur un serveur Debian. L'installation inclut un serveur web Apache, la configuration d'une base de données MariaDB, ainsi que la configuration de l'environnement nécessaire pour faire fonctionner l'application. Nous allons également expliquer comment peupler la base de données et démarrer l'application.

Prérequis

Nous devons disposer d'un serveur Debian installé, sans environnement de bureau, avec un serveur web et SSH activés. Assurons-nous également que nous avons un accès administrateur pour pouvoir installer les paquets et effectuer les configurations nécessaires.

1. Installation du Système

1. Installation de Debian sans environnement de bureau

Nous commençons par installer Debian avec les éléments de base, y compris un serveur web et SSH.

```
su root -c "apt update"  
su root -c "apt upgrade"
```

2. Installation d'Apache, Git, PHP et Composer

Nous installons les paquets nécessaires : Apache2, Git, PHP, Composer et les extensions PHP requises.

```
su root -c "apt-get install git composer apache2 libapache2-mod-php  
php php-mysql php-zip php-gd php-mbstring php-curl php-xml"
```


3. Mise à jour de la distribution

Nous mettons à jour la distribution Debian pour nous assurer que nous disposons des dernières versions de tous les paquets.

```
su root -c "apt update"  
su root -c "apt upgrade"
```

2. Création du Répertoire de Projet

Nous créons un répertoire pour l'application dans le dossier `/var/www`.

1. Création du répertoire de projet :

```
cd /var/www  
su root -c "mkdir deming"  
su root -c "chown $USER:$GROUP deming"
```

2. Clonage du projet depuis Github dans le répertoire `/var/www`

Nous clonons le projet Deming à partir de son dépôt GitHub.

```
cd /var/www  
git clone https://www.github.com/dbarzin/deming
```

3. Création des Répertoires Temporaires

Nous devons créer certains répertoires nécessaires au bon fonctionnement de l'application.

Création des répertoires nécessaires :

```
cd deming  
mkdir -p storage/framework/views  
mkdir -p storage/framework/cache  
mkdir -p storage/framework/sessions  
mkdir -p bootstrap/cache
```

4. Installation des Paquets via Composer

Nous utilisons Composer pour installer toutes les dépendances du projet.

```
composer install
```

5. Installation de MariaDB

Nous installons maintenant MariaDB pour gérer la base de données de l'application.

Installation de MariaDB :

```
su root -c "apt install mariadb-server"
```

Lancement de MariaDB avec les droits root :

```
su root -c "mariadb"
```

Création de la base de données et de l'utilisateur `deming_user` :

```
CREATE DATABASE deming CHARACTER SET utf8 COLLATE utf8_general_ci;  
CREATE USER 'deming_user'@'localhost' IDENTIFIED BY  
'demPasssword-123';  
GRANT ALL ON deming.* TO deming_user@localhost;  
GRANT PROCESS ON *.* TO 'deming_user'@'localhost';  
FLUSH PRIVILEGES;  
EXIT;
```

6. Configuration de l'Application

Nous devons configurer l'application en créant un fichier `.env` et en définissant les paramètres de connexion à la base de données.

Création du fichier `.env` dans le répertoire du projet :

```
cd /var/www/deming  
cp .env.example .env
```

Modification des paramètres de connexion à la base de données :

Nous éditons le fichier `.env` pour y définir les paramètres de la base de données.

```
vi .env
```

Contenu du fichier `.env` :

```
DB_CONNECTION=mysql  
DB_HOST=127.0.0.1  
DB_PORT=3306  
DB_DATABASE=deming  
DB_USERNAME=deming_user  
DB_PASSWORD=demPasssword-123
```

7. Création de la Base de Données

Exécution des migrations pour créer les tables de la base de données :

Nous exécutons les migrations pour configurer la base de données et peupler les tables.

```
LANG=en php artisan migrate --seed
```

Note : L'option `--seed` est importante car elle permet de créer le premier utilisateur administrateur.

Génération de la clé d'application :

```
php artisan key:generate
```

Création du lien vers le répertoire `storage` :

```
php artisan storage:link
```

8. Peuplement de la Base de Données

Si nécessaire, nous pouvons importer des données supplémentaires dans la base de données pour utiliser des attributs de sécurité ISO 27001:2022.

Importation des attributs ISO 27001:2022 :

```
LANG=en php artisan db:seed --class=AttributeSeeder
```

Importation du cadre ISO 27001:2022 et génération du jeu de tests :

```
php artisan deming:import-framework  
./storage/app/repository/ISO27001-2022.en.xlsx  
php artisan deming:generate-tests
```

9. Démarrage de l'Application

Nous pouvons maintenant démarrer l'application en utilisant la commande Artisan pour exécuter le serveur PHP.

Démarrer l'application en local :

```
php artisan serve
```

Démarrer l'application sur une adresse IP externe (si besoin) :

```
php artisan serve --host 0.0.0.0 --port 8000
```

L'application sera accessible à l'URL : <http://127.0.0.1:8000>.

10. Connexion à l'Application

- **Utilisateur par défaut :**
 - **Email :** admin@admin.localhost
 - **Mot de passe :** admin
- Une fois connecté, l'administrateur pourra changer la langue de l'interface en accédant à son profil (en haut à droite de la page principale).

Conclusion

Nous avons installé et configuré avec succès l'application Deming sur un serveur Debian. Le processus inclut l'installation des paquets nécessaires, la configuration de la base de données MariaDB, la création du fichier `.env` et l'exécution des migrations. L'application est désormais prête à être utilisée et peut être accessible localement ou via une adresse IP externe.

Apache Guacamole (Bastion)

Apache Guacamole est une passerelle d'accès à distance sans client nous permettant de nous connecter à des machines via un navigateur web en utilisant les protocoles RDP, SSH et VNC. Son installation requiert plusieurs dépendances et une configuration spécifique.

1. Installation des prérequis

Avant d'installer Apache Guacamole, nous devons préparer l'environnement en installant plusieurs bibliothèques et outils requis pour son fonctionnement.

Nous mettons à jour la liste des paquets afin d'obtenir les dernières versions disponibles:

```
apt-get update
```

L'installation d'Apache Guacamole requiert ensuite plusieurs bibliothèques et paquets de développement :

```
apt-get install build-essential libcairo2-dev libjpeg62-turbo-dev  
libpng-dev \  
libtool-bin uuid-dev libosspp-uuid-dev libavcodec-dev libavformat-dev  
libavutil-dev \  
libswscale-dev freerdp2-dev libpango1.0-dev libssh2-1-dev  
libtelnet-dev \  
libvncserver-dev libwebsockets-dev libpulse-dev libssl-dev  
libvorbis-dev libwebp-dev
```

2. Compilation et installation d'Apache Guacamole Server

Nous devons compiler le code source d'Apache Guacamole pour garantir la compatibilité avec notre système.

Nous nous plaçons dans le répertoire `/tmp` et téléchargeons l'archive :

```
cd /tmp
wget
https://downloads.apache.org/guacamole/1.5.5/source/guacamole-server-
1.5.5.tar.gz
```

Nous extrayons et préparons l'archive pour compiler l'ensemble :

```
tar -xzf guacamole-server-1.5.5.tar.gz
cd guacamole-server-1.5.5/
sudo ./configure --with-systemd-dir=/etc/systemd/system/
```

En cas d'erreur relative à `guacenc_video_alloc`, nous pouvons utiliser :

```
sudo ./configure --with-systemd-dir=/etc/systemd/system/ --disable-guacenc
```

Nous lançons ensuite la compilation et l'installation :

```
sudo make
sudo make install
```

Nous mettons à jour les liens des bibliothèques et activons le service :

```
sudo ldconfig
sudo systemctl daemon-reload
sudo systemctl enable --now guacd
```

3. Configuration de Guacamole Client (Web App)

Guacamole Client repose sur un serveur Tomcat. Nous devons installer Tomcat 9, car Tomcat 10 n'est pas compatible.

Nous ajoutons le dépôt Debian 11 pour récupérer la version compatible :

```
sudo nano /etc/apt/sources.list.d/bullseye.list
```

Nous ajoutons la ligne suivante, enregistrons et fermons le fichier :

```
deb http://deb.debian.org/debian/ bullseye main
```

Nous mettons à jour les paquets et installons Tomcat 9 :

```
sudo nano /etc/apt/sources.list.d/bullseye.list
deb http://deb.debian.org/debian/ bullseye main
sudo apt-get update
sudo apt-get install tomcat9 tomcat9-admin tomcat9-common
tomcat9-user
```

```
flo@srv-guacamole:/tmp/guacamole-server-1.5.5$ sudo apt-get install tomcat9 tomcat9-admin tomcat9-common tomcat9-user
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  ca-certificates-java default-jre-headless java-common libapr1 libeclipse-jdt-core-java libnspr4 libnss3 libpcsclite1
Paquets suggérés :
  default-jre pcscd tomcat9 libnss-mdns fonts-dejavu-extra fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
Les NOUVEAUX paquets suivants seront installés :
  ca-certificates-java default-jre-headless java-common libapr1 libeclipse-jdt-core-java libnspr4 libnss3 libpcsclite1
  tomcat9-admin tomcat9-common tomcat9-user
0 mis à jour, 16 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 58,4 Mo dans les archives.
Après cette opération, 214 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] |
```

4. Compilation et installation d'Apache Guacamole Server

L'installation du serveur Apache Guacamole nécessite de compiler son code source. Cette étape assure une compatibilité avec les bibliothèques et les dépendances installées précédemment.

Nous nous plaçons dans le répertoire /tmp et nous téléchargeons l'archive contenant le code source de Guacamole Server :

```
cd /tmp wget
https://downloads.apache.org/guacamole/1.5.5/source/guacamole-server-
1.5.5.tar.gz
```

Lorsqu'une nouvelle version d'Apache Guacamole est publiée, l'URL de téléchargement devra être mise à jour en conséquence.

Une fois le téléchargement terminé, nous extrayons le contenu de l'archive et nous nous plaçons dans le répertoire obtenu :


```
tar -xzf guacamole-server-1.5.5.tar.gz cd guacamole-server-1.5.5/
```

Ensuite, nous configurons la compilation et nous vérifions la présence des dépendances nécessaires :

```
sudo ./configure --with-systemd-dir=/etc/systemd/system/
```

Si une erreur liée à `guacenc_video_alloc` apparaît, elle est due à la prise en charge des enregistrements vidéo via FFmpeg. Dans ce cas, la commande suivante permet de désactiver ce composant :

```
sudo ./configure --with-systemd-dir=/etc/systemd/system/  
--disable-guacenc
```

Une fois la configuration terminée, nous lançons la compilation :

```
sudo make
```

Puis, nous installons Guacamole Server :

```
sudo make install
```

Après l'installation, il est nécessaire de mettre à jour les liens des bibliothèques :

```
sudo ldconfig
```

Ensuite, nous chargeons le nouveau service et nous activons son démarrage automatique :

```
sudo systemctl daemon-reload sudo systemctl enable --now guacd
```

5. Création du répertoire de configuration

Avant de configurer la partie cliente de Guacamole, il est nécessaire de créer l'arborescence des fichiers de configuration. Le répertoire `/etc/guacamole` contiendra les extensions et les bibliothèques nécessaires à l'intégration avec une base de données MariaDB/MySQL.

```
sudo mkdir -p /etc/guacamole/{extensions,lib}
```

6. Installation de Guacamole Client (Web App)

La partie cliente de Guacamole repose sur un serveur Tomcat. Cependant, Apache Guacamole ne prend pas en charge Tomcat 10, qui est la version par défaut sur Debian 12. Il est donc nécessaire d'installer Tomcat 9 en ajoutant les dépôts de Debian 11 (bullseye).

Nous créons un nouveau fichier source pour apt :

```
sudo nano /etc/apt/sources.list.d/bullseye.list
```

Nous y ajoutons la ligne suivante, puis nous enregistrons et fermons le fichier :

```
deb http://deb.debian.org/debian/ bullseye main
```

Nous mettons à jour la liste des paquets :

```
sudo apt-get update
```

Puis, nous installons Tomcat 9 ainsi que ses composants :

```
sudo apt-get install tomcat9 tomcat9-admin tomcat9-common  
tomcat9-user
```

Nous téléchargeons la dernière version de la Web App :

```
cd /tmp wget
https://downloads.apache.org/guacamole/1.5.5/binary/guacamole-1.5.5.w
ar
Nous déplaçons ensuite le fichier téléchargé vers le répertoire de
Tomcat 9 :
sudo mv guacamole-1.5.5.war /var/lib/tomcat9/webapps/guacamole.war
```

Enfin, nous démarrons les services :

```
sudo systemctl restart tomcat9 guacd
```

7. Base de données MariaDB pour l'authentification

Installation de MariaDB:

```
sudo apt-get install mariadb-server
```

Ensuite, nous sécurisons l'installation :

```
sudo mysql_secure_installation
```

Création de la base de données et d'un utilisateur

```
mysql -u root -p CREATE DATABASE guacadb; CREATE USER
'guaca_nachos'@'localhost' IDENTIFIED BY 'P@ssword!'; GRANT
SELECT,INSERT,UPDATE,DELETE ON guacadb.* TO
'guaca_nachos'@'localhost'; FLUSH PRIVILEGES; EXIT;
```

Ajout de l'extension MySQL

```
cd /tmp wget
https://downloads.apache.org/guacamole/1.5.5/binary/guacamole-auth-jd
```

```
bc-1.5.5.tar.gz tar -xzf guacamole-auth-jdbc-1.5.5.tar.gz sudo mv  
guacamole-auth-jdbc-1.5.5/mysql/guacamole-auth-jdbc-mysql-1.5.5.jar  
/etc/guacamole/extensions/
```

Installation du connecteur MySQL

```
cd /tmp wget  
https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-j-9.1  
.0.tar.gz tar -xzf mysql-connector-j-9.1.0.tar.gz sudo cp  
mysql-connector-j-9.1.0/mysql-connector-j-9.1.0.jar  
/etc/guacamole/lib/
```

Import de la structure de la base de données

```
cd guacamole-auth-jdbc-1.5.5/mysql/schema/ cat *.sql | mysql -u root  
-p guacadb
```

Configuration des fichiers de connexion

```
sudo nano /etc/guacamole/guacamole.properties
```

Contenu du fichier : mysql-hostname: 127.0.0.1 mysql-port: 3306 mysql-database: guacadb mysql-username: guaca_nachos mysql-password: P@ssword!

```
sudo nano /etc/guacamole/guacd.conf
```

Contenu du fichier : [server] bind_host = 0.0.0.0 bind_port = 4822

Redémarrage des services

```
sudo systemctl restart tomcat9 guacd mariadb
```

8. Connexion à Apache Guacamole

Une fois l'installation terminée, l'interface web d'Apache Guacamole est accessible via l'adresse suivante :

```
http://:8080/guacamole/
```

Cela ouvre la page de connexion, où nous pouvons nous authentifier pour accéder à l'interface d'administration et configurer les connexions distantes.



Pour nous connecter, nous allons utiliser les identifiants par défaut :

Utilisateur	<i>guacadmin</i>
Mot de passe	<i>guacadmin</i>

8.1. Création d'un compte administrateur sécurisé

Afin de renforcer la sécurité, nous allons créer un nouveau compte administrateur et supprimer le compte par défaut.

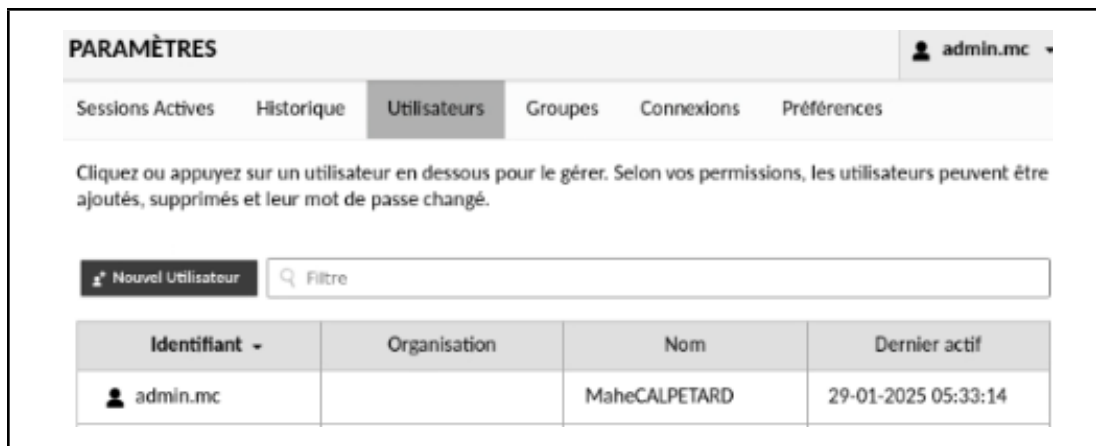
1. Créer un nouveau compte administrateur :

Nom d'utilisateur	admin.mc
Mot de passe	gMhBncUMLwgjKlZiInQS

2. Se déconnecter du compte "guacadmin".
3. Se connecter avec le nouveau compte administrateur.
4. Supprimer le compte "guacadmin" pour éviter toute faille de sécurité.
5. Nous nous connectons avec le nouvel admin créé.



Pour accéder aux paramètres, il faut cliquer sur le nom d'utilisateur en haut à droite puis sur "**Paramètres**" :



8.2. Ajouter une connexion RDP

1. Créer un groupe :

Allez dans **Paramètres** > **Connexion** > **Nouveau groupe**, puis créez un groupe, par exemple "**Serveurs applicatifs**" sous **ROOT**, avec le type "**Organisationnel**".

2. Créer une connexion RDP :

Allez dans **Paramètres** > **Connexion** > **Nouvelle connexion** et configurez la connexion RDP en choisissant le groupe créé.

MODIFIER CONNEXION

Nom:

Lieu:

Protocole:

Réseau

Nom d'hôte:

Port:

Authentification

Identifiant:

Mot de passe:

Nom de domaine:

Mode de Sécurité:

Désactiver l'authentification: ☐

Ignorer le certificat du serveur: ☒

Performance

Activer fond d'écran: ☒

Activer thématisation: ☒

Activer le lissage des polices (ClearType): ☒

Activer pleine fenêtre de glisser: ☒

Activer la composition du bureau (Aero): ☒

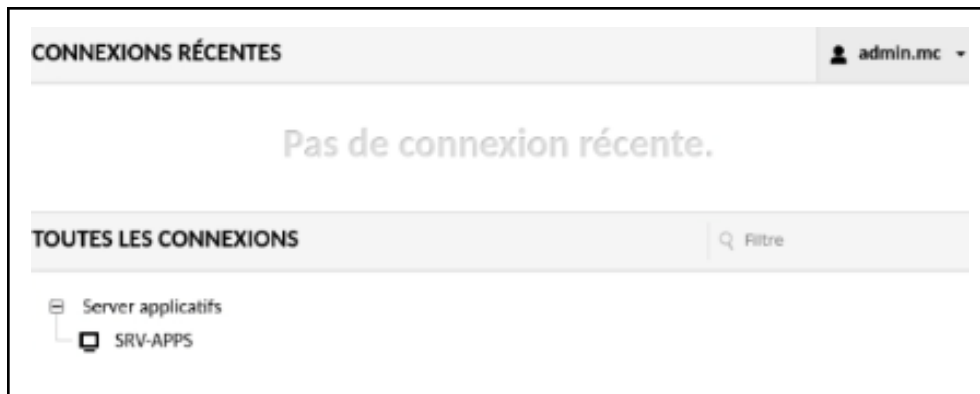
Activer les animations de menu: ☒

Désactiver le cache bitmap: ☐

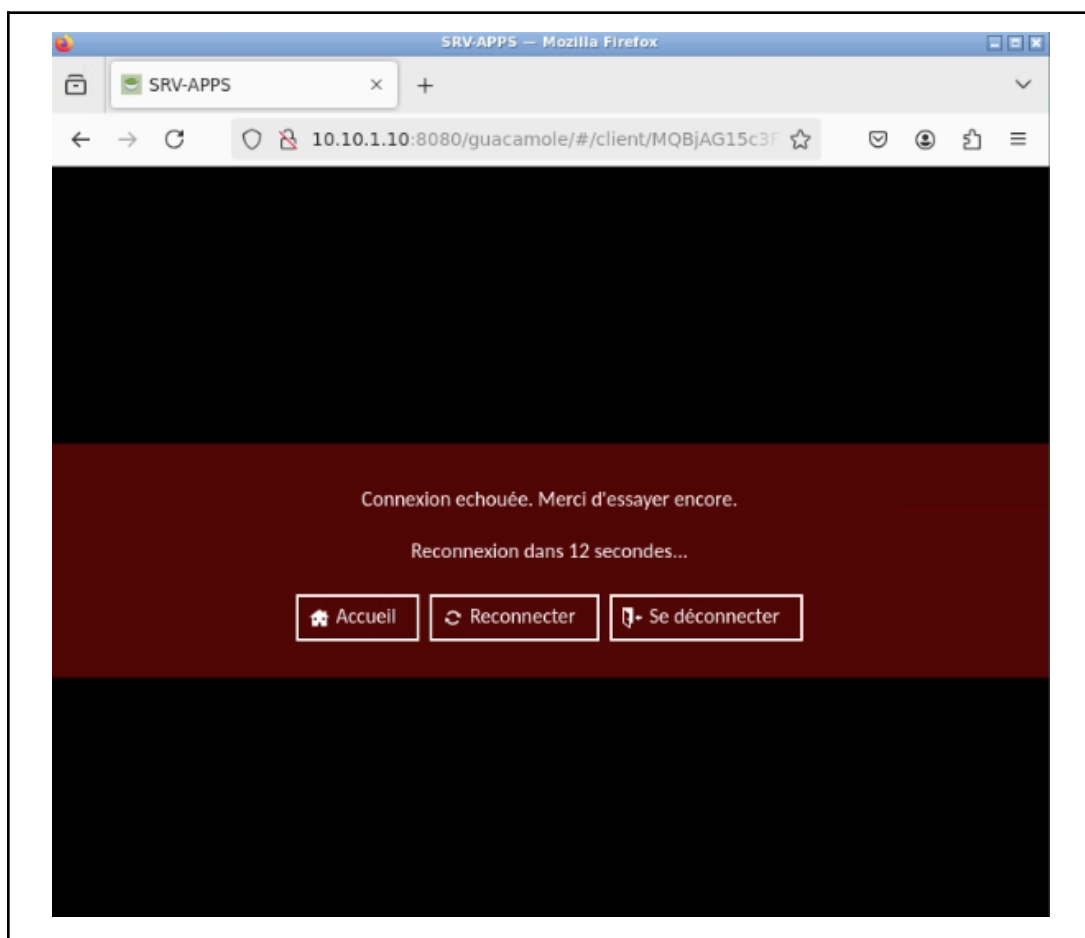
Désactiver le cache hors écran : ☐

Désactiver le cache glyph: ☐

Pour tester une connexion, allez dans **Paramètres > Accueil > Toutes les connexions > Serveurs applicatifs**, puis cliquez sur **SRV-APPS**.



À partir du moment où nous nous connectons avec un utilisateur de l'AD, cela ne fonctionne pas. Le bastion semble ne pas fonctionner correctement à ce stade, ou il y a un problème avec le LDAP :



Conclusion