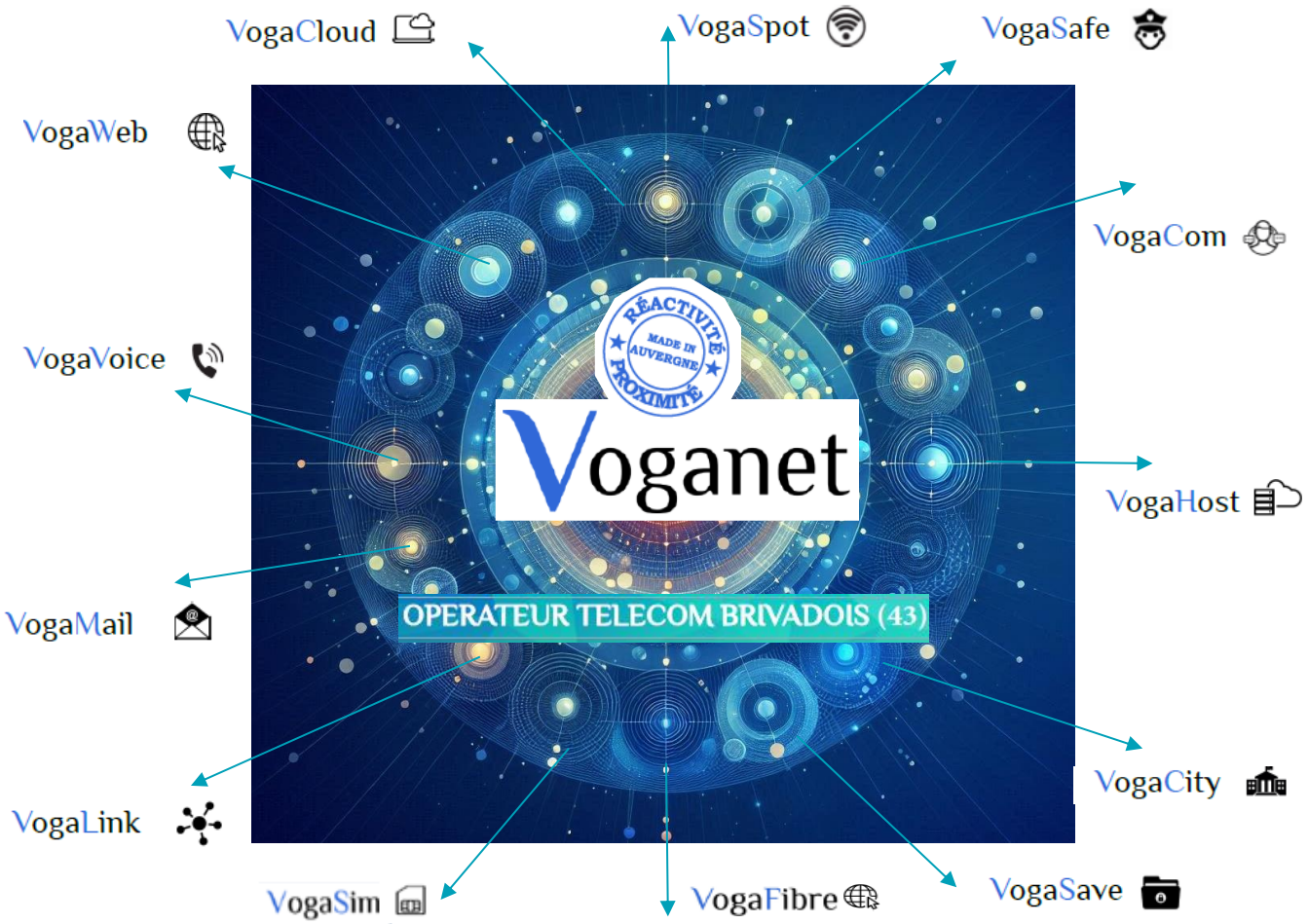
PROJET

OPTIMISATION DES SERVICES TECHNOLOGIQUES DE VOGANET



Mathieu GUELLE / Administrateur d'infrastructures sécurisées 3e année
Pôle la Chartreuse 43 Brives Charensac | 2025

Table des matières

Remerciements

Je tiens à remercier toutes les personnes qui m’ont aidé dans la réalisation de ce projet.

Je suis particulièrement reconnaissant à mon superviseur, Monsieur Bernard PETIT, pour son soutien, ses conseils et sa disponibilité tout au long de ce travail. Son aide a été très précieuse.

Je remercie toute l'équipe de Voganet pour leur accueil chaleureux et leur confiance en m’accueillant en alternance. Par leur savoir-faire, leur patience, ils m'ont beaucoup apporté.

Un merci spécial à Hugo BESSE pour son aide technique et ses conseils utiles.

Enfin, je remercie l'établissement Pôle la Chartreuse à Brives Charensac (43) et tous les professeurs pour la qualité de leur enseignement, les compétences acquises et leur disponibilité.

À tous, un grand merci.

Résumé du projet

Ce projet porte sur l'amélioration de l'infrastructure informatique de Voganet, une entreprise technologique basée à Brioude (43), spécialisée dans les services d'infrastructures sécurisées, gérant son propre réseau d'antennes et offrant un service de maintenance pour ses équipements. L'objectif principal de ce projet est de renforcer la sécurité, la fiabilité et la performance des services offerts par Voganet grâce à l'installation de solutions technologiques avancées.

Le projet comprend plusieurs étapes clés :

1. **Déploiement d'un cluster de bases de données Galera** avec ProxySQL pour assurer la haute disponibilité et la redondance des données.
2. **Mise en place d'un pare-feu** utilisant nftables pour renforcer la sécurité du réseau.
3. **Installation et configuration des serveurs DNS** avec PowerDNS pour une gestion efficace des domaines.
4. **Optimisation des bases de données avec MySQLTuner** pour améliorer la performance et la stabilité du système.
5. **Audit de sécurité des systèmes avec Lynis** pour détecter des failles de configuration, et renforcer le niveau de sécurité.
6. **Supervision et monitoring** des systèmes avec Grafana et Prometheus pour une surveillance continue et proactive.
7. **Documentation détaillée** de toutes les étapes du projet pour faciliter le partage de connaissances au sein de l'équipe.

Les résultats attendus sont une meilleure gestion des noms de domaine, une haute disponibilité des bases de données et une supervision efficace des systèmes. Grâce à ces nouvelles solutions, Voganet peut offrir des services encore plus fiables et sécurisés à ses clients, tout en optimisant les performances de son infrastructure.

Les principaux services de Voganet

1. **VogaHost : Hébergement Web**
    - **Sécurisé et performant :** Voganet propose des solutions d'hébergement web robustes qui garantissent la sécurité des données et des performances élevées.
    - **Personnalisation :** Les clients peuvent choisir parmi différentes options d'hébergement en fonction de leurs besoins spécifiques, qu'il s'agisse d'un site personnel ou d'un site d'entreprise

2. **VogaCom : E-mailing et SMS Groupés**
    - **Outils de publipostage :** Voganet offre des outils pour réaliser des campagnes de marketing par e-mail et SMS, permettant de toucher un large public de manière efficace.
    - **Fidélisation :** Ces services aident les entreprises à fidéliser leurs clients en envoyant des messages personnalisés et des offres spéciales.

3. **VogaSave : Sauvegarde Externalisée**
    - **Sécurité des données :** Voganet propose des solutions de sauvegarde des données pour protéger les informations importantes contre les incidents tels que les pannes de matériel ou les attaques de malware.
    - **Accessibilité :** Les données sauvegardées peuvent être facilement récupérées en cas de besoin.

4. **VogaSafe : Filtrage Web :**
    - **Sécurité en ligne :** Le filtrage web limite l'accès à certains sites internet, assurant ainsi un environnement de travail sécurisé et contrôlé.
    - **Personnalisation :** Les entreprises peuvent définir des règles spécifiques pour le filtrage en fonction de leurs besoins.

5. **VogaSpot : Wi-Fi Public :**
    - **Accès sécurisé :** Voganet propose des réseaux Wi-Fi publics sécurisés pour les utilisateurs, facilitant l'accès à Internet dans divers lieux comme les cafés, les parcs ou les espaces publics.
    - **Gestion des utilisateurs :** Les administrateurs peuvent surveiller et gérer l'utilisation du réseau pour assurer une expérience utilisateur optimale.

6. **VogaCloud : Stockage Cloud :**
    - **Sauvegarde à distance :** Voganet offre des solutions de stockage cloud permettant de sauvegarder des données sur des serveurs distants.
    - **Accessibilité :** Les données stockées dans le cloud peuvent être accessibles à tout moment, depuis n'importe où via un navigateur web.

7. **VogaLink : VPN Sécurisé**
    - **Connexion sécurisée :** Véritable outil de collaboration, il permet de relier les employés présents sur différents sites physiques ou le personnel nomade via un réseau VPN sécurisé.
    - **Mobilité :** Les employés peuvent accéder aux réseaux de l'entreprise de manière sécurisée, même lorsqu'ils sont en déplacement.

8. **VogaMail : Adresses Mail Professionnelles**
    - **Personnalisation :** Voganet permet la création d'adresses e-mail avec un nom de domaine propre à chaque entreprise, renforçant ainsi l'identité professionnelle.
    - **Gestion :** Les entreprises peuvent facilement gérer leurs adresses e-mail et les boîtes de réception associées.

9. **VogaVoice : Lignes de Téléphonie sur IP**
    - **Téléphonie moderne :** Voganet propose des services de téléphonie sur IP (VoIP) qui offrent une communication claire et fiable, adaptée aux besoins des entreprises.
    - **Fax to Mail :** Ils offrent également un service de fax numérique, permettant d'envoyer et de recevoir des fax via e-mail.

10. **VogaFibre : Internet à Très Haut Débit**
    - **Connexion rapide :** Fournit une connexion Internet par fibre optique rapide et fiable pour les entreprises et les particuliers.
    - **Stabilité et performance :** connexion stable et de haute qualité, idéale pour le télétravail, le streaming et les jeux en ligne.

11. **VogaSim : Cartes SIM Professionnelles**
    - **Cartes SIM professionnelles :** Voganet propose des cartes SIM adaptées aux besoins des entreprises, offrant des forfaits de données et de communication flexibles.
    - **Connectivité internationale :** VogaSim permet aux utilisateurs de rester connectés à l'étranger grâce à des options économiques pour les appels, les messages et l'accès à Internet.

12. **VogaWeb : Internet Haut Débit**
    - **Haut débit :** Afin d'apporter le haut débit dans des zones mal desservies d'Auvergne, nous avons développé notre propre réseau internet.
    - **Éligibilité :** Contactez-nous pour analyser l'éligibilité de votre entreprise.

13. **VogaCity : Services pour Collectivités**
    - **Solutions dédiées :** Un service créé pour les besoins des collectivités, offrant des solutions adaptées à leurs exigences spécifiques.

Ces services sont conçus pour aider les entreprises et les particuliers à gérer efficacement leurs besoins en communication et en données, tout en assurant la sécurité et la performance.

1. Introduction

1.1. Présentation de l'entreprise Voganet

Voganet est une entreprise technologique innovante située à Brioude, en Auvergne-Rhône-Alpes. Créée en 2010, Voganet s'affiche comme un acteur local majeur en Haute-Loire, dans le Puy-de-Dôme et le Cantal, en fournissant des solutions Cloud et Télécoms pour les particuliers, les professionnels et les collectivités. Son objectif, c'est de fournir et d'adapter les outils nécessaires pour faciliter la transition numérique de ses clients, en simplifiant les opérations et les processus et en s'adaptant à leurs besoins et à leur configuration.

Voganet dispose de son propre réseau d'antennes, ce qui lui permet de garantir une couverture fiable et personnalisée pour ses clients, notamment dans les zones rurales ou mal desservies. De plus, l'entreprise propose un service de maintenance pour les équipements qu'elle fournit, offrant ainsi une assistance technique rapide et complète.

L'entreprise propose une gamme variée de services, comme l'hébergement de données, la gestion des infrastructures réseau, la cybersécurité et le support technique. Ses solutions regroupent des services essentiels comme l'hébergement web, le filtrage web, l'e-mailing, les SMS groupés, la sauvegarde à distance, le contrôle d'accès Internet, le Wi-Fi public, le stockage cloud, les VPN sécurisés, les adresses mail professionnelles, les lignes de téléphonie sur IP, les cartes SIM professionnelles et l'accès au très haut débit grâce à la fibre.

Par ses compétences techniques et à son engagement de qualité, Voganet fournit des solutions performantes, sécurisées et personnalisées pour répondre aux besoins précis de ses clients.

1.2. Contexte et objectifs du projet

→ **Contexte** :

Dans un monde où la sécurité et la disponibilité des services en ligne sont essentielles, Voganet offre à ses clients des solutions solides et fiables. L'objectif de ce projet est de renforcer encore plus cette position en cherchant à améliorer l'infrastructure informatique de l'entreprise.

Ce travail s'inscrit dans le cadre de ma formation d'Administrateur d'infrastructures sécurisées de niveau 6. Face à l'augmentation du volume des données à gérer, Voganet cherche en permanence à réduire les temps d'arrêt et à améliorer la supervision de ses infrastructures.

→ **Les principaux objectifs** de ce projet sont :

- **La fiabilité :** en assurant la disponibilité continue des services grâce à un cluster de bases de données MariaDB Galera et à des solutions de sauvegarde.
- **La sécurité :** en installant un pare-feu nftables, en établissant des règles de sécurité et en auditant l'infrastructure avec Lynis pour détecter les failles et renforcer la protection du système.
- **La performance :** en optimisant les performances des services avec une gestion efficace des DNS et l'utilisation de ProxySQL et MySQLTuner
- **La supervision :** en mettant en place des outils de surveillance pour suivre l'état des systèmes en continu avec Prometheus et Grafana.
- **La documentation :** en rédigeant une documentation complète pour faciliter le partage de connaissance au sein de l'équipe.

1.3. Mon parcours

En 2019, j'ai obtenu mon Brevet des Collèges avec mention Bien au Collège Saint Julien de Brioude (43). J'ai poursuivi mes études au Lycée Saint Julien où j'ai obtenu un Baccalauréat Général, en 2022 avec les Spécialités : Physique-Chimie, Sciences et vie de la terre avec une mention Section Européenne (Espagnol).

J'ai ensuite poursuivi mes études au Pôle la Chartreuse de Brives Charensac (43) où j'ai obtenu en 2024 mon BTS SIO (Services Informatiques aux Organisations), Option Solutions d'Infrastructure, systèmes et réseaux. Au cours de cette année scolaire j'ai également validé la certification TOEIC (anglais).

Je suis actuellement en alternance en 3 année d'Administrateur d'infrastructures sécurisées toujours au sein du Pôle la Chartreuse afin d'acquérir des compétences supplémentaires et de me préparer à intégrer le monde professionnel.

Mon parcours scolaire, mes stages et CDD en entreprise, m'ont permis de développer de bonnes compétences dans le domaine informatique, mais aussi d'acquérir de la rigueur, de l'autonomie, de m'adapter à de nouveaux environnements et de travailler en équipe.

2. Planification

2.1. Définition des objectifs

Ce projet consiste à mettre en place une série de solutions technologiques pour optimiser la gestion des DNS, renforcer la sécurité du réseau, garantir la haute disponibilité des données et assurer une supervision continue des systèmes. Le projet comprend les étapes suivantes :

1. Déploiement d'un cluster de bases de données Galera avec ProxySQL.
2. Installation et configuration des serveurs DNS avec PowerDNS.
3. Mise en place d'un pare-feu nftables.
4. Supervision et monitoring avec Grafana et Prometheus.
5. Documentation détaillée de toutes les étapes du projet

Le projet repose sur l'utilisation de Proxmox, qui permet de gérer et de déployer efficacement les machines virtuelles pour assurer la flexibilité et l'optimisation des ressources.

2.2. Ressources nécessaires

Pour atteindre ces objectifs, les ressources suivantes sont nécessaires :

- **Ressources matérielles :**
  - **Proxmox** : Plateforme de virtualisation pour gérer les VM du projet.
  - **3 VM pour les bases de données** : Chaque VM sera dédiée à MariaDB dans un cluster Galera, permettant d'avoir une disponibilité continue et de s'adapter aux besoins
  - **1 VM pour Galera Manager** : Cette VM sera utilisée pour gérer et superviser le cluster Galera.
  - **1 VM pour ProxySQL** : Cette VM servira de proxy pour gérer les requêtes SQL vers le cluster Galera
  - **1 VM pour les backups** : Cette VM sera dédiée aux sauvegardes régulières des bases de données et autres données critiques.
  - **2 VM pour les serveurs DNS** : Chaque VM sera configurée avec PowerDNS et gérera des zones et des enregistrements DNS.
  - **1 VM pour le pare-feu nftables** : Cette VM servira de routeur virtuel pour gérer le routage et la sécurité du réseau.
  - **1 VM pour Grafana** : Cette VM sera utilisée pour installer Grafana et configurer les tableaux de bord de monitoring.

- **1 PC de test sous Windows** : Ce PC sera utilisé pour effectuer des tests de validation avant la mise en production.

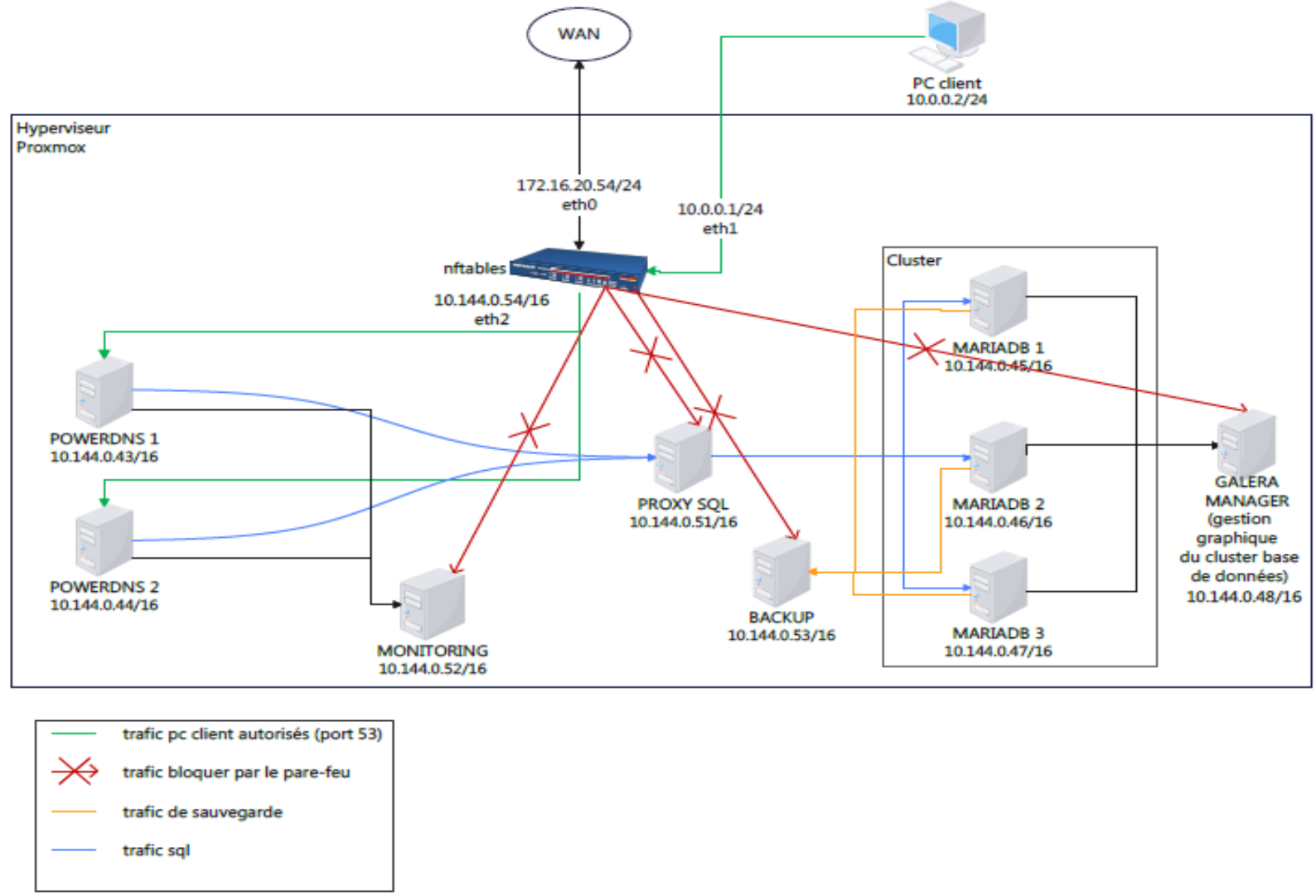
Le réseau d'antennes de Voganet est un élément clé de l'infrastructure permettant d'assurer une connexion fiable pour le déploiement des solutions mises en place dans ce projet.

- **Ressources logicielles :**

  - **Système d'exploitation :** Installation du système d'exploitation Debian 12 sur chaque VM

  - **Logiciels spécifiques :** Téléchargement et installation des logiciels et paquets, tels que MariaDB, Galera Manager, ProxySQL, PowerDNS, Poweradmin, nftables, Grafana, Prometheus, MySQLTuner et Lynis.

2.3 Schéma de l'infrastructure du projet



3. Liste des compétences mises en œuvre

3.1 Administrer et sécuriser les infrastructures

- **Actions réalisées :**
  - Gestion des VM avec Proxmox
  - Mise en place de règles de pare-feu avec nftables pour bloquer les accès non autorisés.
  - Configuration de PowerDNS pour gérer efficacement les noms de domaine avec DNSSEC.
  - Surveillance des systèmes à l'aide de Grafana.
- **Objectifs :**
  - Assurer la fiabilité, l'efficacité et la protection de l'infrastructure
  - Prévenir les interruptions de service.
  - Assurer une communication rapide et fiable entre les différents services.
- **Résultats :**
  - Gestion optimisée des VM avec Proxmox
  - Sécurité renforcée grâce au pare-feu.
  - Identification rapide des anomalies grâce au monitoring.

3.2 Conception et mise en œuvre de solutions techniques

- **Actions réalisées :**
  - Utilisation de Proxmox pour déployer les VM.
  - Création d'un cluster Galera avec MariaDB pour assurer la disponibilité des bases de données.
  - Configuration de ProxySQL pour répartir les charges entre les nœuds du cluster.
  - Mise en place d'une VM pour sauvegarder les données.

- **Objectifs :**

  - Proposer une solution solide et fiable pour gérer les données
  - Améliorer les performances des bases de données.
  - Réduire les risques de panne ou de perte de données.

- **Résultats :**

  - Bases de données toujours disponibles et synchronisées en temps réel.
  - Meilleures performances grâce à une utilisation efficace des ressources.
  - Protection et récupération rapide des données grâce à des sauvegardes régulières.
  - Infrastructure évolutive grâce à Proxmox,

3.3 Gestion de la cybersécurité

- **Actions réalisées :**

  - Création de règles de sécurité avec nftables pour bloquer les accès non autorisés et autoriser les connexions nécessaires.
  - Suivi des activités réseau avec Grafana pour détecter les problèmes
  - Intervention rapide pour analyser et résoudre les menaces identifiées

- **Objectifs :**

  - Renforcer la sécurité des infrastructures en bloquant les accès non autorisés.
  - Permettre un suivi efficace des activités réseau pour détecter rapidement les anomalies.
  - Réagir rapidement aux menaces identifiées pour assurer la continuité des services.

- **Résultats :**

  - Infrastructure sécurisée grâce à des règles précises avec nftables
  - Problèmes détectés rapidement grâce au suivi actif avec Grafana.
  - Menaces analysées et neutralisées rapidement pour limiter les interruptions

3.4 Tableau des compétences

N° Fiche AT	Activités types	N° Fiche CP	Compétences professionnelles	Quel(s) élément(s) répond à la compétence	Outils utilisés
1	Administrer et sécuriser les infrastructures	1	**Appliquer les bonnes pratiques dans l'administration des infrastructures**	Gestion des serveurs DNS Configuration du pare feu Supervision des systèmes	PowerDNS recursor et PowerDNS authoritative Nftables Prometheus et Grafana
		2	**Administrer et sécuriser les infrastructures réseaux**	Déploiement d'un cluster Galera Configuration de ProxySQL Création d'une VM dédiée aux sauvegardes	MariaDB ProxySQL Proxmox
		3	**Administrer et sécuriser les infrastructures systèmes**	Elaboration des politiques de sécurité Surveillance Réponse rapide aux incidents identifiés	Nftables Grafana
		4	**Administrer et sécuriser les infrastructures virtualisées**	Création de VM via proxmox	Proxmox
2	Concevoir et mettre en œuvre une solution en réponse à un besoin d'évolution	5	**Concevoir une solution technique répondant à des besoins d'évolution de l'infrastructure**	Déploiement des serveurs DNS Configuration des entrées DNS Création des règles nftables	PowerDNS Poweradmin Nftables
		6	**Mettre en production des évolutions de l'infrastructure**	Installation prometheus Configuration grafana Intégration des sources de données et seuil d'alerte	prometheus grafana
		7	**Mettre en œuvre et optimiser la supervision des infrastructures**	Création vm de sauvegarde Sauvegarde régulière Test de restauration	
3	Participer à la gestion de la cybersécurité	8	**Participer à la mesure et à l'analyse du niveau de sécurité de l'infrastructure**	Test des serveurs dns Validation performance cluster Audit de sécurité et détection des vulnérabilités	PowerDNS Proxysql, MSQLTunner Lynis
				Tableau de bord grafana	Grafana
		9	**Participer à l'élaboration et à la mise en œuvre de la politique de sécurité**	Planification des étapes de déploiement Mise en place des mesures de sécurité	
		10	**Participer à la détection et au traitement des incidents de sécurité**	Documentations Rédaction de rapport	

4. Cahier des charges

4.1. Les besoins de l'entreprise

L'entreprise souhaite répondre à trois principaux besoins :

- **Optimiser la gestion des DNS :** c'est améliorer la rapidité, la fiabilité et la sécurité du système de résolution des noms de domaine Les enregistrements DNS sont stockés dans des bases de données pour une gestion optimisée et une synchronisation efficace
- **Renforcer la sécurité des données** : Protéger les informations sensibles des clients et de l'entreprise contre toute perte ou attaque.
- **Augmenter la capacité de gestion des bases de données** : Permettre de gérer efficacement une quantité croissante de données tout en assurant un fonctionnement fluide

4.2. Les solutions mises en place

- **Réplication des données** : Avoir plusieurs copies des données pour éviter leur perte en cas de problème technique.
- **Continuité de service** : Faire en sorte que le système continue à fonctionner même si une partie tombe en panne.
- **Gestion simplifiée** : Installer des outils qui rendent l'administration des bases de données plus facile.

4.3. Les contraintes techniques et opérationnelles

- **Performance :**

  Les bases de données doivent répondre rapidement aux demandes et être capables de traiter un grand nombre d'opérations par seconde.

  Le cluster Galera qui regroupe plusieurs bases de données, doit permettre à plusieurs utilisateurs de travailler en même temps sans ralentir le système.

- **Sécurité :**

  Installation d’un pare-feu : Un système qui bloque les accès non autorisés au réseau de l’entreprise.

  Politiques de sécurité claires : Des règles qui expliquent comment utiliser le système en toute sécurité.

  Systèmes de détection d’intrusions : Des outils capables de repérer et stopper les tentatives d’attaques informatiques.

- **Maintenance :**

  Facilité des mises à jour : Le système doit être facile à mettre à jour, ce qui réduit les interruptions de service.

  Outils de surveillance : Grafana permet de surveiller en continu le fonctionnement du système et de résoudre rapidement les problèmes.

5. Gestion de projet

5.1. Organisation du projet

La planification du projet s'est déroulée sans contraintes de temps. Cela m’a permis de progresser étape par étape, en prenant le temps de bien intégrer chaque solution.

5.2. Ressources humaines et techniques

- **Ressources humaines** : Le projet a été géré par une seule personne, moi en tant qu’alternant. Cette gestion en autonomie m’a permis d'acquérir des compétences précieuses en organisation et en résolution de problèmes.

- **Ressources techniques** : Tout le projet a été réalisé dans un environnement virtuel en utilisant des machines virtuelles (VM) grâce à Proxmox pour faciliter les configurations nécessaires. Mon pc a servi pour les tests afin de vérifier le bon fonctionnement du système.

5.3. Objectifs de qualité

L’objectif principal était de garantir une infrastructure performante et fiable.

- Les technologies utilisées ont été testées pour s’assurer qu'elles fonctionnent correctement.
- Une documentation complète a été créée pour expliquer toutes les étapes du projet et faciliter la maintenance future

5.4. Aperçu des technologies utilisées

Pour répondre aux besoins du projet, plusieurs technologies ont été employées :

- **Galera Cluster** : Pour synchroniser les bases de données et garantir leur disponibilité en permanence.
- **ProxySQL** : Pour gérer les connexions aux bases de données et améliorer leur performance.
- **PowerDNS** : Pour une gestion efficace des noms de domaine.
- **nftables** : Pour sécuriser le réseau avec un pare-feu configurable qui bloque les accès non autorisés.
- **Grafana** : Pour surveiller les performances des systèmes grâce à des tableaux de bord dynamiques et des données collectées en temps réel.

L’utilisation de ces technologies a permis de répondre aux besoins de performance, de sécurité et de supervision. Grâce aux machines virtuelles, les solutions ont pu être testées, intégrées facilement, tout en restant simples d’utilisation.

6. Création des machines virtuelles sous Proxmox

Proxmox Virtual Environment (Proxmox VE) est une solution de virtualisation open source qui permet de créer et de gérer des machines virtuelles et des conteneurs. Elle propose une interface web facile à utiliser permettant de s'adapter et d'être efficace.

Dans le cadre de ce projet, Proxmox a été utilisé pour :

- **Créer des machines virtuelles** adaptées aux besoins spécifiques du projet, notamment pour les serveurs DNS, les bases de données, le monitoring, le pare-feu, les sauvegardes, ProxySQL, et Galera Manager.
- **Configurer les VM** avec des paramètres appropriés (RAM, CPU, stockage et paramètres réseaux)

Nom de la VM	Description	CPU	RAM	Disque	IP	Fonction
mariadb1 mariadb2 mariadb3	Bases de données (cluster Galera)	2	2 Go	50 Go	10.144.0.45/16 10.144.0.46/16 10.144.0.47/16	Réplication et disponibilité continue
galera-manager	Supervision cluster Galera	2	2 Go	8 Go	10.144.0.48/16	Monitoring avancé
proxysql	Proxy pour cluster Galera	2	2 Go	8 Go	10.144.0.51/16	Gestion des requêtes SQL
backup	Sauvegardes critiques	2	2 Go	20 Go	10.144.0.53/16	Dédicacé aux sauvegardes
powerdns1 powerdns2	Serveurs DNS	2	2 Go	8 Go	10.144.0.43/16 10.144.0.44/16	Gestion des zones/enregistrements DNS
nftables	Routeur virtuel (iptables)	2	2 Go	8 Go	10.144.0.54/16	Sécurisation et routage
monitoring	Grafana & Prometheus	2	2 Go	20 Go	10.144.0.52/16	Monitoring & tableaux de bord

- **Tester les configurations** dans un environnement isolé avant leur mise en production.

Les étapes détaillées pour la création des VM sont décrites dans :
Fiche 1 : Création et configuration de conteneurs Debian sous Proxmox (Annexes, page 29)

7. Mise en place d'un cluster MariaDB Galera avec Proxy SQL

Ce projet a pour but d'assurer une gestion fiable et performante des bases de données, en garantissant leur accessibilité et en assurant la continuité du service en cas de panne.

J'ai donc mis en place un cluster Galera basé sur MariaDB permettant une synchronisation automatique entre plusieurs bases de données et évitant ainsi les pertes de données ou les interruptions de service en cas de panne d'un serveur : chaque nœud peut à la fois lire et écrire assurant une grande disponibilité et une répartition efficace des ressources.

Pour améliorer la gestion des requêtes SQL, j'ai intégré ProxySQL pour répartir les connexions entre les serveurs, réduire les temps d'attente et augmenter les performances.

De plus pour garantir la sécurité, j'ai mis en place une VM dédiée aux sauvegardes pour protéger les bases de données MariaDB permettant d'effectuer des copies régulières avec mariadb-dump pour éviter toute perte de données en cas de problème.

7.1. Installation des bases de données MariaDB

Pour garantir une gestion efficace des données et assurer leur disponibilité, trois machines virtuelles (VM) ont été mises en place : mariadb1, mariadb2 et mariadb3. Ces serveurs forment un cluster Galera permettant la duplication automatique des bases de données et assurant une continuité du service, même en cas de panne d'un serveur.

L'installation de MariaDB se fait en plusieurs étapes :

- **Préparation des machines virtuelles**

  Avant d'installer MariaDB, il faut préparer les machines virtuelles qui vont héberger les bases de données.

Chaque VM (mariadb1, mariadb2, mariadb3) est configurée avec les ressources nécessaires :

- 2 CPU, 2 Go de RAM et 50 Go de stockage.
- Une adresse IP fixe pour faciliter la communication entre les serveurs.

- **Installation de MariaDB**
  - Le serveur MariaDB est installé sur chaque VM pour assurer une gestion centralisée des bases de données.
  - La configuration inclut la mise en place des paramètres de duplication et l'activation des fonctionnalités nécessaires pour le fonctionnement du cluster Galera.
- **Optimisation et sécurisation**
  - Le moteur InnoDB est activé pour garantir la compatibilité avec Galera Cluster.
  - Des ajustements de configuration sont effectués pour améliorer les performances et la gestion des transactions.
  - La sécurité du système est renforcée avec des règles de contrôle d'accès et des paramètres réseau adaptés.

Grâce à cette infrastructure, les bases de données sont stables, performantes et prêtes à être intégrées dans le cluster Galera pour assurer une réplication synchrone et une disponibilité maximale.

7.2. Configuration du cluster Galera avec Galera Manager

Après l'installation des bases de données sur mariadb1, mariadb2 et mariadb3, j'ai configuré le cluster Galera avec Galera Manager.

Cet outil permet de superviser l'état des serveurs, d'automatiser certains processus et d'assurer la récupération rapide du cluster en cas d'incident : chaque serveur fonctionne comme un nœud actif**,** capable de traiter les opérations de lecture et d'écriture**.**

Les principaux atouts du cluster Galera :

- **Haute disponibilité** : Si un serveur devient indisponible, les autres continuent de fonctionner sans impact sur le service.
- **Répartition automatique des requêtes** : Chaque nœud peut répondre aux demandes SQL, améliorant ainsi les performances du système.

- **Sécurité et fiabilité** : Les transactions sont validées et répliquées immédiatement sur tous les serveurs, garantissant l'intégrité des données.

Pour simplifier la gestion du cluster et assurer un suivi efficace, j'ai utilisé Galera Manager. Cet outil facilite la supervision, l'optimisation des performances et la récupération du cluster en cas de panne.

Les étapes techniques sont détaillées dans **Fiche 2 : Installation et configuration du cluster MariaDB Galera (Annexes, page 32).**

7.3. Installation et configuration de ProxySQL

Pour améliorer le traitement des requêtes SQL et optimiser les performances du cluster Galera, j'ai intégré ProxySQL, un proxy de bases de données permettant d'équilibrer la charge entre les serveurs MariaDB.

Les principaux atouts de ProxySQL :

- **Équilibrage de charge** : Les requêtes SQL sont distribuées efficacement entre les serveurs, évitant qu'un seul nœud soit surchargé.
- **Accélération des performances** : ProxySQL optimise le traitement des requêtes, accélérant ainsi le temps de réponse des bases MariaDB.
- **Gestion intelligente des connexions** : Il permet de maintenir un nombre optimal de connexions ouvertes, réduisant les erreurs dues à des pics d'activité.
- **Filtrage et sécurité** : ProxySQL peut bloquer certaines requêtes non autorisées et contrôler l'accès aux bases de données.

L'utilisation de ProxySQL permet d'optimiser l'accès aux bases de données et de prévenir les risques de surcharge des serveurs.

Les étapes techniques sont détaillées dans **Fiche 4 : Installation et configuration de ProxySQL (Annexes, page 36).**

7.4. Mise en place d'une VM dédiée aux backups

Pour protéger les bases MariaDB contre la perte de données, un système de sauvegarde a été mis en place permettant d'effectuer des copies régulières et de récupérer les données en cas d'incident.

Les principaux avantages :

- **Protéger les bases contre les pertes accidentelles** (pannes, erreurs humaines, corruption des données) par des sauvegardes régulières des bases de données via un système automatisé.
- **Récupération rapide des bases de données** en cas d'incident grâce à un stockage des sauvegardes sur une machine virtuelle sécurisée.
- **Assurer la continuité du service** grâce à des procédures de restauration rapides pour minimiser l'impact des incidents.

Les étapes techniques sont détaillées dans **Fiche 5 : Sauvegarde et restauration des bases MariaDB (Annexes, page 41).**

8. Installation des serveurs DNS et du pare-feu nftables

Dans une infrastructure réseau, les serveurs DNS jouent un rôle essentiel dans la résolution des noms de domaine, tandis que le pare-feu nftables assure la sécurité en bloquant les accès non autorisés.

J'ai donc installé et configuré PowerDNS sur deux VM pour assurer une redondance et une gestion efficace des zones DNS. Poweradmin a été utilisé comme interface graphique pour faciliter la gestion des enregistrements DNS.

Grâce au pare-feu nftables, j'ai mis en place des règles strictes et personnalisées pour bloquer les connexions non autorisées, protéger l'infrastructure et renforcer ainsi la sécurité globale.

8.1. Installation et configuration des serveurs DNS avec PowerDNS

Pour garantir une résolution fiable des noms de domaine, deux serveurs DNS ont été déployés à l'aide de PowerDNS pour assurer à la fois la redondance du service et une gestion optimisée des zones DNS.

Les principales étapes :

- **Déploiement de deux VM** : Ces machines ont été configurées pour héberger les serveurs DNS, assurer la disponibilité et la stabilité.
- **Installation de PowerDNS** : pour gérer efficacement les zones DNS.
- **Synchronisation des deux serveurs** pour garantir la disponibilité en continue du service DNS, même en cas de panne d'un serveur.

Les étapes techniques sont détaillées dans **Fiche 6 : Installation des serveurs DNS (Annexes, page 44)**

8.2. Gestion des zones et des enregistrements DNS via Poweradmin

Pour simplifier la gestion des zones DNS et des enregistrements, Poweradmin, a été installé et configuré pour créer, modifier et administrer les enregistrements DNS assurant ainsi une gestion efficace des ressources réseau.

Les principales étapes :

- **Création des zones DNS** : zones pour regrouper les enregistrements nécessaires à la résolution des noms de domaine.
- **Configuration des enregistrements de type A** : pour associer les noms de domaine (exemple : www.domain.com) aux adresses IPv4 des serveurs.

  Chaque enregistrement A a été configuré via Poweradmin pour un accès rapide aux ressources.
- **Configuration des enregistrements de type NS** : ils désignent les serveurs PowerDNS autorisés pour le domaine et garantissent la redondance des zones DNS grâce à une synchronisation entre les serveurs.

- **Modifications et mises à jour** :

  Poweradmin permet de modifier les enregistrements existants (types A, NS, etc.) ou d'ajouter de nouveaux enregistrements selon les besoins de l'infrastructure.

  Les mises à jour sont effectuées en temps réel pour maintenir la cohérence et l'efficacité des serveurs DNS.

Les étapes techniques sont détaillées dans **Fiche 7 : Gestion des zones et des enregistrements DNS via Poweradmin (Annexes, page 49)**

8.3. Installation et configuration du pare-feu nftables

Le pare-feu nftables, intégré au noyau Linux, a été directement configuré pour protéger l'infrastructure et bloquer les accès.

Seules les connexions DNS, SSH et Ping sont autorisées. Un journal des tentatives de connexion suspectes a été créé pour renforcer la sécurité globale

Les étapes techniques sont détaillées dans **Fiche 8 : Configuration du pare-feu nftables (Annexes, page 53).**

9. Supervision et monitoring avec Prometheus et Grafana

Pour assurer une surveillance efficace de l'infrastructure, Grafana et Prometheus ont été mis en place pour collecter et afficher les données en temps réel. Ce système permet d'identifier rapidement les anomalies et d'optimiser la gestion des services.

Prometheus surveille les performances des bases de données, du pare-feu et du DNS et déclenche des alertes en cas de comportement anormal.

Grafana transforme ces informations en tableaux de bord visuels offrant une analyse précise des performances

9.1. Surveillance des données avec Prometheus

- **Mise en place de Prometheus** pour surveiller les performances des bases de données, du pare-feu et du DNS.
- **Paramétrage du système** pour collecter et analyser les données en temps réel
- **Définition des règles d'alerte** pour signaler les anomalies système.

9.2. Intégration des sources de Grafana

- **Connexion de Grafana à Prometheus** pour afficher les données collectées.
- **Configuration des sources de données** pour une visualisation optimale.

9.3. Création et paramétrages des tableaux de bord.

Les principales étapes de mise en place et d'exploitation du système de supervision :

- **Création et personnalisation des tableaux de bord**
- **Importation de tableaux de bord préconçus** pour visualiser les performances des services comme PowerDNS et ProxySQL.
- **Mise en place d'alertes et notifications** pour assurer une meilleure réactivité.

Les étapes techniques sont détaillées dans **Fiche 9 : Supervision et monitoring avec Prometheus et Grafana** (**Annexes, page 56).**

10. Validation et sécurisation du système

La phase de tests et de validation est très importante pour s'assurer du bon fonctionnement des services déployés et leur stabilité avant leur mise en service. Cette étape permet de détecter et corriger les éventuels dysfonctionnements et de renforcer la sécurité

10.1. Analyse des serveurs DNS

Il s'agit de vérifier que les serveurs DNS répondent correctement aux demandes de conversion des noms de domaine en adresses IP. Cette étape est essentielle pour assurer l'accessibilité des services et le bon fonctionnement des échanges de données

Les principaux points de vérification :

- **Test de résolution des noms** : pour s'assurer qu'un domaine donné (ex. example.com) est bien traduit en une adresse IP correcte.
- **Vérification du temps de réponse** : pour mesurer la rapidité avec laquelle le serveur DNS répond aux requêtes.
- **Validation des configurations DNS** : examiner les fichiers de configuration (named.conf, resolv.conf) pour détecter d'éventuelles erreurs.
- **Validation de la synchronisation des serveurs DNS** : pour vérifier que les enregistrements DNS sont bien transmis à tous les serveurs et que les réponses sont rapides et fiables.

10.2. Contrôle du cluster de bases de données

Les principaux points de vérification :

- **Contrôle de la mise à jour des données entre les serveurs.**
- **Vérification de l'exactitude et de la fiabilité des informations stockées.**
- **Vérification du bon fonctionnement et de la réactivité du système**.

10.3. Vérification du monitoring

- **Contrôle de la collecte des données** : pour vérifier que les informations des différents services sont correctement remontées et analysées.
- **Validation des alertes** : pour s'assurer que les notifications sont bien déclenchées en cas d'anomalie pour permettre une intervention rapide.
- **Vérification que les tableaux de bord affichent les bonnes informations.**

10.4. Mesures de sécurité

Pour protéger les données et prévenir les risques, plusieurs actions ont été mises en place :

- **Utilisation de MSQLTuner** pour renforcer les performances et réduire les vulnérabilités.

MySQLTuner est un outil qui analyse un serveur MySQL ou MariaDB et propose des améliorations. Il aide à optimiser la configuration, améliorer les performances et renforcer la sécurité, tout en ajustant les paramètres mémoire.

Les étapes techniques sont détaillées dans **Fiche 9 : Optimisation des MariaDB avec MySQLTunner** (**Annexes, page 57).**

- **Analyse du système avec Lynis** pour identifier les failles de sécurité et améliorer la protection des serveurs.

Lynis est un outil open-source d'audit de sécurité destiné aux systèmes basés sur Unix. Il permet d'identifier les failles de configuration, de vérifier la conformité aux bonnes pratiques, et d'améliorer la posture de sécurité globale du système via un indice de durcissement.

Les étapes techniques sont détaillées dans **Fiche 10 : Audit de sécurité des systèmes avec Lynis** (**Annexes, page 60).**

- **Gestion des accès** : Définir des règles d'authentification et des autorisations pour limiter les accès non autorisés.
- **Surveillance** du système : Analyse des journaux d'activité et détection des anomalies pour une réaction rapide en cas de menace.

11. Conclusion

Ce projet a permis à Voganet d'améliorer son infrastructure informatique en renforçant la fiabilité, les performances et la sécurité des services fournis. Grâce aux solutions mises en place, l'entreprise peut mieux gérer la gestion de ses DNS, sécuriser son réseau et assurer la continuité de ses prestations, tout en assurant une meilleure protection contre les intrusions.

Le déploiement du cluster MariaDB Galera a permis d'assurer la continuité du service en réduisant les temps d'arrêt et en permettant l'accès aux bases de données même en cas de panne. De plus, l'intégration de ProxySQL et la gestion des serveurs DNS via PowerDNS et Poweradmin ont amélioré la rapidité des requêtes et la stabilité du réseau. L'utilisation de Prometheus et Grafana a permis une surveillance efficace du système permettant de détecter et de résoudre rapidement les problèmes. En complément, une VM dédiée aux sauvegardes a été mise en place pour sécuriser les données et garantir leur récupération rapide en cas d'incident.

La sécurité du réseau a été renforcée par la configuration du pare-feu nftables et la mise en place de mesures de gestion des accès. Une analyse du système avec Lynis a permis d'identifier les failles de sécurité et d'améliorer la protection des serveurs. L'utilisation de MSQLTuner a permis d'optimiser les performances et sécuriser les bases de données.

Enfin des fiches pratiques ont été rédigées pour faciliter le partage des connaissances au sein de l'équipe.

Grâce à ces améliorations, Voganet dispose maintenant d'un système performant, sécurisé et évolutif permettant de répondre aux besoins actuels de l'entreprise et de s'adapter aux défis futurs.

Fin du projet – Début des annexes

12. Annexes

12.1. Fiche 1 : Création et configuration de conteneurs Debian sous Proxmox

1. Création d'un conteneur Debian sous Proxmox

- Accéder à l'interface web de Proxmox. (exemple:https://<IP_du_serveur_Proxmox>:8006)
- Cliquer sur **Créer CT** (Créer Conteneur).
- Remplir les paramètres généraux :
  - **Nom** : Indiquer un nom pour le conteneur (ex : powerdns1).
  - **Template** : Sélectionner le template Debian12 disponible dans la bibliothèque.
- Configurer le réseau :
  - **Bridge** : Assurer-vous de sélectionner le bridge réseau approprié (vmbr2).
- Configurer les ressources :
  - Définir les CPU, RAM et stockage en fonction des besoins du conteneur.
- Finaliser la création en cliquant sur **Terminer**.

- **Liste des VM** à configurer dans Proxmox :

Nom de la VM	Description	CPU	RAM	Disque	IP	Fonction
mariadb1 mariadb2 mariadb3	Bases de données (cluster Galera)	2	2 Go	50 Go	10.144.0.45/16 10.144.0.46/16 10.144.0.47/16	Réplication et disponibilité continue
galera-manager	Supervision cluster Galera	2	2 Go	8 Go	10.144.0.48/16	Monitoring avancé
proxysql	Proxy pour cluster Galera	2	2 Go	8 Go	10.144.0.51/16	Gestion des requêtes SQL
backup	Sauvegardes critiques	2	2 Go	20 Go	10.144.0.53/16	Dédicacé aux sauvegardes

Nom de la VM	Description	CPU	RAM	Disque	IP	Fonction
powerdns1 powerdns2	Serveurs DNS	2	2 Go	8 Go	10.144.0.43/16 10.144.0.44/16	Gestion des zones/enregistrements DNS
nftables	Routeur virtuel (iptables)	2	2 Go	8 Go	10.144.0.54/16	Sécurisation et routage
monitoring	Grafana & Prometheus	2	2 Go	20 Go	10.144.0.52/16	Monitoring & tableaux de bord

2. Configurer le conteneur (sans interface graphique)

- **Accéder au conteneur** :
  - Une fois le conteneur démarré, ouvrir une console via l'interface de Proxmox.

- **Mise à jour** :
  - Mettre à jour le système :

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

  - Changer le fuseau horaire :

```
dpkg -reconfigure tzdata
```

Choisir Europe puis Paris

3. Ajouter l'utilisateur voganet

- Dans la console, ajouter l'utilisateur :

```
adduser voganet
```

- Suivre les instructions :
  - Définir un mot de passe pour voganet.

- Ajouter l'utilisateur au groupe sudo :

```
usermod -aG sudo voganet
```

- Maintenant que ces étapes sont faites je peux me connecter en ssh sur les VM.

4. Principaux points

- Conteneur Debian créé sous Proxmox, avec le bridge réseau correctement configuré.
- Conteneur optimisé sans interface graphique pour économiser les ressources.
- Utilisateur voganet ajouté avec les privilèges nécessaires.
- Connexion en ssh sur les VM pour configurer les différents services

12.2. Fiche 2 : Installation et configuration du cluster MariaDB

1. Présentation

Depuis MariaDB 10.1, Galera Cluster fait partie intégrante du serveur MariaDB. Ce cluster garantit une réplication synchrone et une haute disponibilité des bases de données.

Important :

- Galera Cluster prend uniquement en charge le moteur de stockage InnoDB.
- MyISAM et autres moteurs non transactionnels ne sont pas compatibles.

2. Installation et préparation de MariaDB

- **Installer MariaDB :**
  - sudo apt install mariadb-server mariadb-client -y
- **Vérifier le moteur de stockage :**
  - Se connecter à MariaDB : mariadb -u root -p
  - Vérifier le moteur de stockage par défaut : SHOW VARIABLES LIKE 'default_storage_engine';
  - Si ce n'est pas InnoDB, le définir comme moteur par défaut : SET storage_engine=INNODB;
- **Arrêter MariaDB sur tous les serveurs :** systemctl stop mariadb

3. Configuration de Galera

- **Modifier le fichier de configuration** :
  - Éditer le fichier /etc/mysql/mariadb.conf.d/60-galera.cnf sur chaque serveur et ajoutez les paramètres suivants :

    [galera]

    wsrep_on = ON

wsrep_cluster_name = "MariaDB Galera Cluster"

wsrep_cluster_address = gcomm://192.168.1.15,192.168.1.16,192.168.1.17

wsrep_provider = /usr/lib/libgalera_smm.so

binlog_format = row

default_storage_engine = InnoDB

innodb_autoinc_lock_mode = 2

- **Description des paramètres critiques :**
  - wsrep_on : Active la réplication synchrone.
  - wsrep_cluster_name : Nom unique du cluster (doit être identique sur tous les nœuds).
  - wsrep_cluster_address : Liste des adresses IP des nœuds du cluster.
  - wsrep_provider : Chemin vers la bibliothèque Galera.
  - binlog_format : Configuré sur row pour une réplication au niveau des lignes, ce qui garantit que toutes les transactions sont reproduites exactement de la même manière sur tous les nœuds du cluster.
  - innodb_autoinc_lock_mode : Optimise la gestion des valeurs auto-incrémentées pour les insertions simultanées, qui permet aux nœuds de générer des valeurs AUTO_INCREMENT sans nécessiter un verrou global.

4. Initialisation et gestion du cluster

- **Démarrer le cluster sur le premier nœud** :
  - Exécuter cette commande uniquement sur le premier nœud :

galera_new_cluster

  - Vérifiez le statut du cluster :

mariadb -u root -p

SHOW STATUS LIKE 'wsrep_cluster_size';

  - Résultat attendu :

+-------------------------+--------+

| Variable_name | Value |

+-------------------------+--------+

| wsrep_cluster_size | 1 |

+-------------------------+--------+

- **Ajouter les autres nœuds :**
  - Démarrez MariaDB sur chaque nœud supplémentaire :

    systemctl start mariadb
  - Vérifiez la taille du cluster sur n'importe quel nœud actif :

    SHOW STATUS LIKE 'wsrep_cluster_size';

5. Résolution des problèmes

- **Cluster complet arrêté :**
  - Identifier le nœud avec la dernière version du cluster en vérifiant safe_to_bootstrap :

    cat /var/lib/mysql/grastate.dat

    Si la valeur est égale à 1, relancez le cluster :

    galera_new_cluster
  - Démarrez les autres nœuds :

    systemctl start mariadb
- **Via Galera Manager :**
  - Accéder à l’interface web.
  - Choisir le cluster, cliquer sur les options (trois points), puis sélectionnez Recover cluster.
  - Galera Manager choisit un nœud pour relancer le cluster et synchronise les autres nœuds.

6. Sauvegardes et restauration

- **Sauvegarde logique :**

  - Sauvegarder une base de données :

  mariadb-dump --single-transaction --extended-insert -u root db_name > backup_file_$(date +'%Y-%m-%d').sql

  - Sauvegarder toutes les bases :

  mariadb-dump --single-transaction --extended-insert -u root --all-databases > backup_all_$(date +'%Y-%m-%d').sql

- **Restauration :**

  - Importer une sauvegarde :

  mariadb -u root db_name < backup_file.sql

⚠ ***Note*** : Méthode peu adaptée aux bases volumineuses en raison des ressources CPU consommées.

7. Principaux points

- Initialisation sur le premier nœud : Utiliser galera_new_cluster.
- Ajout d'autres nœuds : Démarrez simplement MariaDB avec systemctl start mariadb.
- Validation du cluster : Vérifier la taille avec la commande SHOW STATUS LIKE 'wsrep_cluster_size';.
-
- Gestion des incidents : Relancer un cluster arrêté en identifiant le dernier nœud actif (grastate.dat).
-
- Sauvegarde et restauration : Effectuer des sauvegardes avec mariadb-dump et restaurer les bases si nécessaire.

12.4. Fiche 3 : Installation et configuration de ProxySQL

1. Présentation :

ProxySQL agit comme **un intermédiaire** entre les applications et les bases MariaDB, pour :

- Améliorer les performances (répartition lecture/écriture).
- Gérer les connexions pour optimiser la gestion des requêtes
- Surveiller l'état des bases en temps réel pour détecter les anomalies.
- Optimiser les ressources serveur pour s'adapter aux variations de charge.

Il fonctionne comme un serveur MySQL et facilite la gestion des bases de données.

2. Installation

ProxySQL n'est pas inclus dans les dépôts Debian, son dépôt officiel doit être ajouté

```
apt-get update && apt-get install -y --no-install-recommends lsb-
release wget apt-transport-https ca-certificates
wget -nv -O /etc/apt/trusted.gpg.d/proxysql-2.7.x-keyring.gpg
'https://repo.proxysql.com/ProxySQL/proxysql-
2.7.x/repo_pub_key.gpg'
echo "deb https://repo.proxysql.com/ProxySQL/proxysql-
2.7.x/$(lsb_release -sc)/ ./" | tee
/etc/apt/sources.list.d/proxysql.list
```

Installation :

```
apt-get update
apt-get install proxysql mariadb-client
```

3. Démarrage et Chargement de la configuration

- **Premier démarrage** : ProxySQL lit /etc/proxysql.cnf et crée sa base interne proxysql.db dans /var/lib/proxysql/.
- **Les démarrages suivants** : il ignore le fichier et lit uniquement proxysql.db.
- **Forcer la relecture du fichier de configuration** :

```
sudo systemctl start proxysql-initial
# ou sudo service proxysql initial
```

4. Principaux ports

- **6032 : Interface admin** (gestion et configuration).
- **6033 : Connexions clients MariaDB**
- **6070 : REST API Prometheus** (monitoring).

5. Fichier de Configuration

Chemin : /etc/proxysql.cnf

```
datadir="/var/lib/proxysql"
errorlog="/var/lib/proxysql/proxysql.log"
admin_variables=
{
           admin_credentials="admin:admin"
           mysql_ifaces="0.0.0.0:6032"
           restapi_enabled=true
           restapi_port=6070
           prometheus_memory_metrics_interval=60
}
 mysql_variables=
{
           interfaces="0.0.0.0:6033"
           monitor_username="monitor"
           monitor_password="monitor"
}
```

```
#Ajout des serveurs
mysql_servers =
(
        { address="10.144.0.45" , port=3306 , hostgroup=2,
max_connections=1000 , comment=mysql1 },
        { address="10.144.0.46" , port=3306 , hostgroup=3,
max_connections=1000 , comment=mysql2 },
        { address="10.144.0.47" , port=3306 , hostgroup=3,
max_connections=1000 , comment=mysql3 }
)
#Utilisateur dont les requêtes sont transmises au base de données
mysql_users =
(
        { username = "proxy_user", password = "password",
default_hostgroup = 3, active = 1 },
        { username = "poweradmin", password = "poweradmin",
default_hostgroup = 2, active = 1 }
)
#Routage des requêtes
mysql_query_rules =
(
        { rule_id=1, active=1, match_pattern="^SELECT .*",
destination_hostgroup=3, apply=1 },
        { rule_id=2, active=1, match_pattern="^select .*",
destination_hostgroup=3, apply=1 },
        { rule_id=10, active=1, match_pattern=".*",
destination_hostgroup=2, apply=1 }
)
```

6. Interface d’administration

Connexion :

```
mysql -u admin -padmin -h 127.0.0.1 -P6032 --prompt='Admin> '
```

Bases principales :

- main : paramètres de configuration.
- disk : stockés dans proxysql.db.
- stats : statistiques runtime.
- monitor : informations de monitoring.
- stats_history : historiques de stats.

Par défaut on se trouve dans la table main. Les tables runtime_* dans main sont gérées automatiquement par ProxySQL. Elles reflètent l'état actuel de la configuration chargée dans le runtime. Elles sont utiles pour vérifier ce qui est actif.

7. Gestion de la configuration en mémoire et disque

- **Charger en mémoire (temporaire) :** LOAD <nom de la table avec des espaces> TO RUNTIME;
- **Sauvegarder sur disque (persistant)** : SAVE MYSQL <nom de la table avec des espaces> TO DISK;

8. Prise en charge du cluster galera

```
INSERT INTO mysql_galera_hostgroups
(writer_hostgroup,backup_writer_hostgroup,reader_hostgroup,offline_
hostgroup,active,max_writers,writer_is_also_reader) VALUES
(2,4,3,1,1,1,0);
```

Pour définir les rôles des différents serveurs dans un cluster MariaDB Galera, j’ai configuré ProxySQL pour répartir les connexions selon quatre groupes de serveurs :

- **Groupe 1** : serveurs considérés comme hors ligne (offline).
- **Groupe 2** : serveurs pour les opérations d’écriture.
- **Groupe 3** : serveurs destinés aux lectures
- **Groupe 4** : serveurs d’écriture de secours (backup).

J’ai activé la configuration en précisant qu’un seul serveur doit être utilisé pour les écritures à la fois (max_writers = 1), et que les serveurs d’écriture ne doivent pas servir les requêtes de lecture (writer_is_also_reader = 0). Mais le problème c’est qu’il n’y avait aucun serveur dans le groupe 3. J’ai donc essayé avec writer_is_also_reader = 1

```
SELECT * FROM runtime_mysql_servers;
```

On voit qu’en exécutant la commande :

- 1 seul serveur en écriture (groupe 2) actif : mysql3
- 3 serveurs en lecture (groupe 3) : mysql1, mysql2, mysql3
- 2 serveurs en écriture de secours (groupe 4) : mysql1, mysql2

Lors de l’exécution, la répartition des rôles des serveurs ne correspond pas à la configuration par défaut définie dans galera-hostgroups-replication (mysql1, serveur d’écriture). Mais dans un cluster Galera, cela n’a pas d’importance car tous les noeuds sont en réplication multi-master permettant une synchronisation des données.

9. Principaux points

- **Meilleure répartition des requêtes** (lecture/écriture).
- **Gestion optimisée des connexions**, réduisant la charge sur les bases.
- **Suivi en temps réel** pour détecter les anomalies.
- **Adaptation des ressources** selon la charge.

12.5. Fiche 4 : Sauvegarde et restauration des bases MariaDB

Afin de protéger les bases de données MariaDB contre toute perte accidentelle (pannes, erreurs humaines, corruption), une VM dédiée aux sauvegardes a été mise en place. Ce système de sauvegarde assure une protection fiable des données, permet leur récupération rapide en cas d'incident

1. Script de sauvegarde local (à placer sur chaque serveur MariaDB)

```
#!/bin/bash
DATE=$(date +'%Y-%m-%d')
TIME=$(date +'%H:%M:%S')
HOST=$(hostname)
BACKUP_FILE="/backups/$(hostname)_db_backup_${DATE}.sql"
LOG_FILE="/backups/backup.log"

# Commande de dump
mariadb-dump --single-transaction --extended-insert -uvoganet --
all-databases > "$BACKUP_FILE" 2>> "$LOG_FILE"

# Vérifie si la commande a échoué
if [ $? -ne 0 ]; then
     echo "[$DATE $TIME] $HOST : ERREUR - MariaDB inaccessible ou
dump échoué" >> "$LOG_FILE"
else
     echo "[$DATE $TIME] $HOST : Sauvegarde réussie -> $(basename
"$BACKUP_FILE")" >> "$LOG_FILE"
fi
```

Tâche planifiée

```
crontab -e
```

Ajout d'une règle pour une sauvegarde quotidienne :

```
0 4 * * * /root/backup.sh
```

2. Centralisation des sauvegardes sur la VM de backup

Cette VM va :

- Récupérer les fichiers de sauvegarde de chaque serveur distant.
- Lire les journaux de log pour détecter les erreurs.
- Envoyer une alerte Telegram en cas d'échec ou d'indisponibilité réseau.

```
#!/bin/bash

SERVER=(10.144.0.45 10.144.0.46 10.144.0.47)
i=1
TODAY=$(date +'%Y-%m-%d')
TIME=$(date +'%H:%M:%S')
DEST_DIR="/backups"
TELEGRAM_TOKEN="bot_token"
CHAT_ID="chat_id"

send_telegram() {
      local message="$1"
      curl -s -X POST
"https://api.telegram.org/bot${TELEGRAM_TOKEN}/sendMessage" \
      -d chat_id="${CHAT_ID}" \
      -d text="${message}" > /dev/null
}

for srv in "${SERVER[@]}"
do
      BACKUP_NAME="backup_mysql${i}_${TODAY}.sql"
      LOG_FILE="backup.log"
      # Récupère la sauvegarde
      scp voganet@$srv:/home/voganet/${BACKUP_NAME} "${DEST_DIR}/"
2>/dev/null
      if [ $? -ne 0 ]; then
      send_telegram "Erreur : impossible de récupérer la sauvegarde
depuis mysql${i}"
      ((i++))
      continue
      fi

      # Récupère le fichier de log
      scp voganet@$srv:/home/voganet/${LOG_FILE}
"${DEST_DIR}/mysql${i}_${TODAY}.log" 2>/dev/null
```

```
      # Vérifie la dernière ligne du log
if [ -f "${DEST_DIR}/mysql${i}_${TODAY}.log" ] && tail -n 1
"${DEST_DIR}/mysql${i}_${TODAY}.log" | grep -q "ERREUR"; then
      send_telegram "Sauvegarde échouée sur mysql${i}"
      fi

      # Nettoyage des fichiers
      ssh voganet@$srv "rm ${DEST_DIR}/${BACKUP_NAME}" 2>/dev/null

      rm -f "${DEST_DIR}/mysql${i}_${TODAY}.log"
      ((i++))
done
```

Tâche planifiée

```
crontab -e
```

Ajout d’une règle pour une sauvegarde quotidienne :

```
15 4 * * * /root/backup.sh
```

3. Restauration des bases MariaDB

```
mariadb -u root < nom_du_fichier.sql
```

4. Principaux points

- Automatisation : Sauvegardes planifiées sans intervention manuelle.
- Supervision : Vérification des logs et envoi d’alertes Telegram en cas d’échec.
- Sécurisation : Stockage protégé et accès limité via SSH
- Récupération rapide : Procédures optimisées pour restaurer les bases en cas de problème.

12.6. Fiche 5 : Configuration des serveurs DNS

PowerDNS est un serveur DNS utilisé pour gérer les zones DNS via une base de données MariaDB et pour traiter les requêtes externes avec PowerDNS Recursor. Sa flexibilité et ses performances le rendent idéal pour les infrastructures nécessitant une haute disponibilité.

1. Installation de PowerDNS Authoritative

- **Installer les paquets nécessaires** via apt :

Pour la **gestion des zones dans une base de données** PowerDNS a besoin du paquet pdns-server, ainsi que du paquet pdns-backend-mysql pour que les zones soient gérées dans une base de données mysql/mariadb qui sont téléchargeables depuis le gestionnaire de paquet apt :

```
sudo apt install pdns-server pdns-backend-mysql
```

- **Si l'erreur ci-dessous se produit** lors de l'installation de pdns-server faire la commande :

```
sudo systemctl daemon-reload
```

```
Setting up pdns-server (4.7.3-2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/pdns.service -> /lib/systemd/system/pdns.service.
Setting up pdns-backend-mysql (4.7.3-2) ...
Setting up pdns-backend-bind (4.7.3-2) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for libc-bin (2.36-9+deb12u7) ...
Processing triggers for pdns-server (4.7.3-2) ...
Failed to restart pdns.service: Unit pdns.service failed to load properly, please adjust/correct and reload service manager: Device or resource busy
See system logs and 'systemctl status pdns.service' for details.
invoke-rc.d: initscript pdns, action "restart" failed.
x pdns.service - PowerDNS Authoritative Server
     Loaded: error (Reason: Unit pdns.service failed to load properly, please adjust/correct and reload service manager: Device or resource busy)
     Active: failed (Result: exit-code) since Thu 2024-08-22 08:21:11 UTC; 13min ago
   Duration: 14min 39.435s
       Docs: man:pdns_server(1)
             man:pdns_control(1)
             https://doc.powerdns.com
```

```
   Main        PID:        20926        (code=exited,        status=1/FAILURE)
        CPU:                                                              119ms

Aug  22  08:21:09  powerDNS-1  systemd[1]:  pdns.service:  Main  process  exited,
code=exited,                                                  status=1/FAILURE
Aug 22 08:21:09 powerDNS-1 systemd[1]: pdns.service: Failed with result 'exit-
code'.
Aug 22 08:21:09 powerDNS-1 systemd[1]: Failed to start pdns.service - PowerDNS
Authoritative                                                          Server.
Aug 22 08:21:11 powerDNS-1 systemd[1]: pdns.service: Failed to schedule restart
job:            Unit            pdns.service            not            found.
Aug 22 08:21:11 powerDNS-1 systemd[1]: pdns.service: Failed with result 'exit-
code'.
```

- **Vérifier la présence de pdns-backend-bind**

Lors de l'installation pdns-backend-bind peut s'installer. On peut vérifier s'il s'est installé en faisant

```
dpkg --list | grep pdns-backend-bind
```

S'il est présent le supprimer

```
apt remove --purge pdns-backend-bind
```

2. Configuration de la base de données MariaDB

- **Créer la base de données mariaDB pour PowerDNS ainsi que l'utilisateur**

```
mysql
CREATE DATABASE powerdns;
GRANT ALL PRIVILEGES ON powerdns.* TO 'powerdns'@'%' IDENTIFIED BY 'Password';
FLUSH PRIVILEGES;
```

- **Importer le schéma des tables** :

Il faut créer les tables pour mysql avec le schéma fourni par PowerDNS

Pour éviter de le faire à la main, il faut mettre le schéma des tables dans un fichier texte. Mettre en première ligne `USE powerdns;`

Pour que les tables soient rajoutées à mysql il faut faire :

```
mysql -u root -p <  file_name
```

3. Paramétrage de PowerDNS

- Dans le fichier `/etc/powerdns/pdns.conf` modifier les paramètres :

```
launch=gmysql
gmysql-host=10.144.0.51
gmysql-port=6033
gmysql-dbname=powerdns
gmysql-user=powerdns
gmysql-password=Password
gmysql-dnssec=yes
local-address=<host ip>
```

- **Redémarrer PowerDNS** :

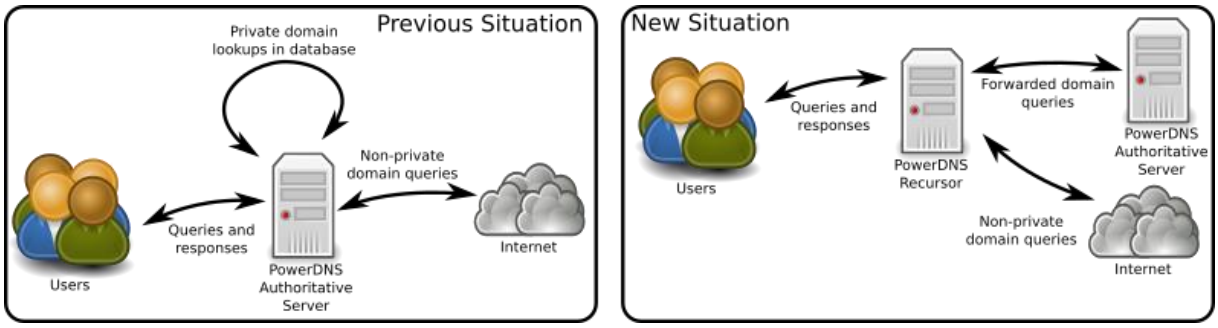
```
sudo systemctl restart pdns
```

4. Installation et configuration de PowerDNS Recursor

- **Installer PowerDNS Recursor** :

Pour contacter les serveurs root PowerDNS a besoin du paquet `pdns-recursor` :

sudo apt install pdns-recursor



- Modifier **/etc/powerdns/pdns.conf** pour éviter un conflit de port :

Par défaut pdns et pdns-recursor écoutent sur le même port le 53.

Donc pour ne pas créer d'erreur il faut modifier le port de pdns

`/etc/powerdns/pdns.conf` :

```
local-port=5300
```

Puis redémarrer pdns :

```
sudo systemctl restart pdns
```

- Modifier **/etc/powerdns/pdns-recursor.conf** :

Dans le fichier `/etc/powerdns/pdns-recursor.conf` modifier les paramètres :

```
allow-from=10.144.0.0/16, 127.0.0.0/8

dnssec=validate
hint-file=/usr/share/dns/root.hints
local-address=<host ip>
forward-zones=voganet.net=10.144.0.44:5300
```

- **Redémarrer PowerDNS Recursor** :

```
sudo systemctl restart pdns-recursor
```

- **Vérification des fichiers Root Hints**

Pour choisir comment la validation dnssec va agir :
https://doc.powerdns.com/recursor/dnssec.html

Vérifier que l'information sur les fichiers root, root-hints se trouvent à cet endroit : /usr/share/dns/root.hints ou sinon le télécharger depuis
https://www.internic.net/domain/named.root

- **Gestion des zones de pdns**

Pour l'option forward-zones il faut préciser le nom de la zone suivi du serveur. Pour ajouter une autre zone il faut mettre une ligne :`forward-zones+=another.example.com=127.0.0.1:5300`

IMPORTANT : Lors de l'utilisation de la validation DNSSEC (qui est activée par défaut), les redirections vers des zones non déléguées (comme les zones internes) ayant une zone parente signée par DNSSEC seront validées comme étant fausses (Bogus). Pour éviter cela, ajoutez une Negative Trust Anchor (NTA).

Pour ajouter une NTA il faut ajouter une ligne dans le fichier `/etc/powerdns/recursor.lua` :

```
addNTA('une.zone', 'NTA for this zone')
```

5. Principaux points

- **PowerDNS Authoritative** permet la gestion des zones DNS via MariaDB.
- **PowerDNS Recursor** permet de résoudre les requêtes externes.
- **La configuration des fichiers pdns.conf et pdns-recursor.conf** est essentielle pour éviter les conflits de ports et optimiser les performances.
- **L'utilisation des fichiers Root Hints et DNSSEC** améliore la sécurité et la fiabilité des requêtes DNS.

12.7. Fiche 6 : Gestion des zones et enregistrements DNS via Poweradmin

1. Présentation

Poweradmin est une **interface web** facilitant la gestion des zones et des enregistrements DNS sous PowerDNS. Grâce à son interface intuitive, il permet de **créer, modifier et gérer** efficacement les entrées DNS, assurant une meilleure gestion des ressources réseau et une mise à jour simplifiée des enregistrements.

Pour utiliser Poweradmin pour gérer des zones DNSSEC, il est nécessaire de l'installer sur le même serveur que le DNS ou, si ce n'est pas le cas, d'installer le paquet pdns pour disposer de l'outil pdns-util

2. Installation et configuration de Poweradmin

- **Installer les dépendances**

```
sudo apt-get install apache2 libapache2-mod-php php php-common php-curl php-dev php-gd php-pear php-imap php-mcrypt php-mysql php-intl php-tokenizer php-pdo php-pdo-mysql
```

- **Installer Poweradmin** avec git

```
git clone https://github.com/poweradmin/poweradmin.git
```

ou avec wget :

```
wget https://github.com/poweradmin/poweradmin/archive/refs/tags/v3.8.1.tar.gz
tar xvfz v3.8.1.tar.gz
```

```
mv poweradmin-3.8.1 /var/www/html/poweradmin
chown -R www-data:www-data /var/www/html/poweradmin
```

- **Faire l'installation depuis un navigateur** :

```
http:<host ip>/poweradmin/install
```

- étape 1 : choisir la langue d'installation
- étape 2 : appuyer sur le bouton suivant
- étape 3 : Entrer les informations concernant la base de donnée :
  nom d'utilisateur pour se connecter

  mot de passe

  le type de base donnée : MySQL

  l'host de la base de donnée

  le port utilisé

  le charset utilisé et la collation, powerdns utilise latin1 et latin_swedish_ci

  le mot de passe admin qui sera utilisé pour se connecter à poweradmin
- étape 4 : Créer l'utilisateur avec lequel poweradmin va interagir avec la base de données :
- étape 5 : poweradmin demande que l'utilisateur créer à l'étape d'avant soit ajouter sur les bases de données pour qu'il interagisse avec

- **Finaliser l'installation et configurer DNSSEC**

  Il faut créer le fichier touch `/var/www/html/poweradmin/config.inc.php` ou bien copier le fichier de template

```
cp /var/www/html/poweradmin/config-defaults.inc.php
/var/www/html/poweradmin/config.inc.php
```

Il faut mettre les données qui sont afficher sur l'écran dans le fichier `config.inc.php` et ajouter aussi :

```
// DNSSEC settings
$pdnssec_use = true;
$pdnssec_debug = true;
$pdnssec_command = '/usr/bin/pdnsutil';
```

```
// Keeping track of record and zone changes in the database
$dblog_use = true;
$timezone = 'Europe/Paris';
```

- **Supprimer le répertoire d'installation**

```
rm -R /var/www/html/poweradmin/install
```

- **Connexion à Poweradmin** : accéder via (http://<host ip>/poweradmin)
  il faut utiliser username : admin et le mot de passe c'est le mot de passe défini à l'étape 3 dans la case Poweradmin Administrator Password

3. Gestion des utilisateurs et permissions

- **Création d'un utilisateur qui peut uniquement modifier et créer des zones:**

Il faut commencer par créer un groupe d'utilisateur avec les permissions suivante dans : List permission templates → Add permission template :

- zone_content_edit_others
- zone_content_edit_own
- zone_content_edit_own_as_client
- zone_content_view_others
- zone_content_view_own
- zone_master_add
- zone_meta_edit_others
- zone_meta_edit_own
- zone_slave_add

Pour mettre un utilisateur dans ce groupe : Users → Add user :

Dans permission template choisir le groupe que l'on a créé juste avant

4. Création des zones DNS

- **Ajouter une nouvelle zone** : Accéder à Add master zone et mettre le nom de la zone et la zone inverse

- **Configuration des enregistrements DNS**

Ajout des zones : List zones → choisir une zone et appuyez sur edit

5. Principaux points

- **Gestion centralisée des enregistrements DNS**, simplifiant leur administration via une interface web.
- **Création et gestion des zones DNS** : Ajout des domaines et des serveurs maîtres.
- **Modifications et mises à jour** immédiates pour une meilleure stabilité réseau.

- **Sécurisation avec DNSSEC** et gestion des utilisateurs avec des **permissions adaptées**

12.8. Fiche 7 : Configuration du pare-feu nftables

Le pare-feu nftables, intégré au noyau Linux, est utilisé ici pour protéger un routeur ou serveur multi-interface en appliquant un filtrage strict. Il :

- Bloque toutes les connexions non autorisées.
- Autorise uniquement les services nécessaires comme DNS, SSH et Ping.
- Sépare les flux entre Internet, clients du LAN et un serveur local.
- Met en place un NAT pour permettre aux clients d'accéder à Internet.

Cette configuration est adaptée à une machine avec trois interfaces réseau :

- eth0 : vers Internet
- eth1 : vers le réseau des serveurs
- eth2 : vers un réseau client (LAN)

1. Configuration de nftables :

Il y a deux façons de le configurer :

- via la ligne de commande
- via le fichier de conf

```
#!/usr/sbin/nft -f

flush ruleset

table ip filter {
       chain input {
              type filter hook input priority 0; policy drop;
              # Accepter le loopback
              iifname "lo" accept
              # Autoriser les connexions deja etablies
              iifname "eth0" ct state related,established accept
              # Autoriser le ping
              ip protocol icmp accept
              # Autoriser ssh
              tcp dport 22 accept
              # Drop connection invalide
              ct state invalid drop
```

```
	}
	chain forward {
		type filter hook forward priority 0; policy drop;
		ct state related,established accept
		# Autoriser le trafic sortant entre les clients et Internet
		iifname "eth2" oifname "eth0" accept
		# Réponse Internet -> Clients
		iifname "eth0" oifname "eth2" ct state established,related accept
		# LAN vers serveur : uniquement DNS
		iifname "eth2" oifname "eth1" udp dport 53 accept
		iifname "eth2" oifname "eth1" tcp dport 53 accept

		iifname "eth1" oifname "eth2" accept
		iifname "eth1" oifname "eth0" accept
		# Drop connection invalide
		ct state invalid drop
	}
	chain output {
		type filter hook output priority 0; policy accept;
	}
}

table ip nat {
	chain prerouting {
		type nat hook prerouting priority -100; policy accept;
	}
	chain input {
		type nat hook input priority 100; policy accept;
	}
	chain output {
		type nat hook output priority -100; policy accept;
	}
	chain postrouting {
		type nat hook postrouting priority 100; policy accept;
		oifname "eth0" masquerade
	}
}
```

chargement des règles

```
sudo nft -f /etc/nftables.conf
```

2. Journalisation des tentatives suspectes

```
nft add rule inet filter input log prefix "Tentative de connexion : " level
warning
```

3. Vérification et surveillance

Vérifier les règles appliquées

```
sudo nft list ruleset
```

Consulter les logs des connexions suspectes

```
sudo journalctl -u nftables --no-pager | grep "Tentative de connexion"
```

4. Principaux points

Filtrage précis : Seuls les services **DNS, SSH et Ping** sont autorisés.

Blocage des connexions indésirables : Protection contre les accès non autorisés.

Surveillance active : Journalisation automatique des tentatives suspectes.

12.9. Fiche 8 : Supervision et monitoring avec Prometheus et Grafana

1. Installer Prometheus :

```
sudo apt install prometheus
```

2. Configurer Prometheus

Mettre dans le fichier `/etc/prometheus/prometheus.yml` :

```
scrape_configs:
  - job_name: 'powerdns1-auth'
      static_configs:
      - targets: ['10.144.0.43:8081']
      basic_auth:
      username: "LWFFWyH9LJj5X8sqoO2tzde5l6CKbKq24S2XMDQar0k="
      password: "SeuiDP43"

  - job_name: 'powerdns1-recursor'
      static_configs:
      - targets: ['10.144.0.43:8082']
      basic_auth:
      username: "25VmBjNLc+ceZAJJlaWQRIyOubGvjEgokvZRps0Sy8U="
      password: "SeuiDP43"

  - job_name: 'powerdns2-auth'
      static_configs:
      - targets: ['10.144.0.44:8081']
      basic_auth:
      username: "ncaXxb0ZMhl2D5D3b7jOjT6kO06E8mk9BPZNGPCaNHY="
      password: "SeuiDP43"

  - job_name: 'powerdns2-recursor'
      static_configs:
      - targets: ['10.144.0.44:8082']
      basic_auth:
      username: "RAPymNu5MPGuCcLpv5c5bWAL89pP6K9VJQQAC5wC8Lw="
      password: "SeuiDP43"


  - job_name: 'proxysql'
      static_configs:
      - targets: ['10.144.0.51:6070']
```

Redémarrer Prometheus :

```
systemctl restart prometheus
```

L'interface web de prometheus se trouve à l'adresse : `http://<host ip>:9090`

3. Installer Grafana

Installer Grafana à partir du dépôt APT (Doc sur le site de Grafana pour l'installation)

Installez les paquets prérequis :

```
sudo apt-get install -y apt-transport-https software-properties-common wget
```

Importez la clé GPG :

```
sudo mkdir -p /etc/apt/keyrings/ wget -q -O - https://apt.grafana.com/gpg.key |
gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null
```

Pour ajouter un dépôt pour les versions stables, exécutez la commande suivante :

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

Mettre à jour la liste des paquets disponibles :

```
sudo apt-get update
```

Pour installer Grafana :

```
sudo apt-get install grafana
```

4. Connexion à Grafana et configuration des tableaux de bord

Dans un navigateur allez à l'adresse suivante pour accéder à grafana : `http://<host ip>:3000`

Ajouter une nouvelle base de données prometheus dans Connections → Data sources :

Prometheus server URL :

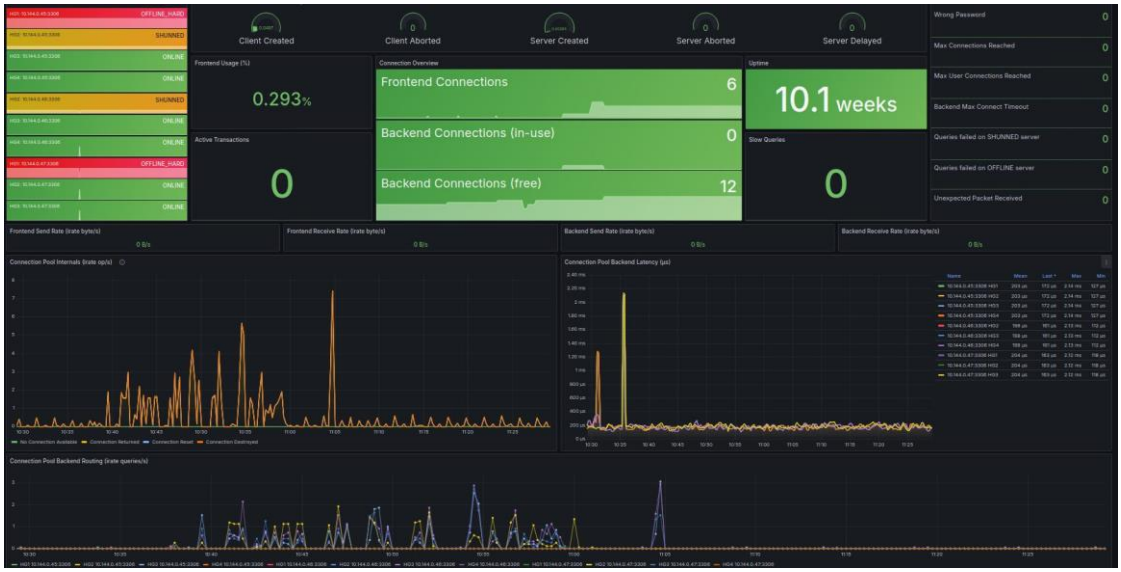
```
http://<host ip>:9090
```

5. Modèles de tableaux de bord

- powerdns authoritative

- powerdns recursor

- Proxysql



Principaux points

- **Surveillance automatisée** → Prometheus collecte des métriques en temps réel pour une analyse approfondie des performances des systèmes.
- **Intégration avec Grafana** → Visualisation des données sous forme de tableaux de bord dynamiques et personnalisables.
- **Supervision des services clés** → Monitoring des serveurs DNS, du cluster MariaDB et de ProxySQL pour garantir une stabilité et une rapidité optimales.
- **Accès centralisé** → Interface web pour consulter les métriques et détecter les anomalies rapidement.
- **Configuration simple et flexible** → Paramétrage des sources de données dans Grafana pour une gestion efficace du monitoring.

12.10. Fiche 9 : Optimisation de MariaDB avec MySQLTunner

1. Installation de MySQLTuner

Deux méthodes possibles :

- Télécharger juste le script

```
wget http://mysqltuner.pl/ -O mysqltuner.pl

wget https://raw.githubusercontent.com/major/MySQLTuner-perl/master/basic_passwords.txt -O basic_passwords.txt

wget https://raw.githubusercontent.com/major/MySQLTuner-perl/master/vulnerabilities.csv -O vulnerabilities.csv
```

- Cloner le dépôt GitHub

```
git clone --depth 1 -b master https://github.com/major/MySQLTuner-perl.git
cd MySQLTuner-perl
```

2. Utiliser le script

Pour des diagnostics plus précis, il est recommandé d'activer `performance_schema` dans `/etc/mysql/my.cnf` et redémarrer MariaDB

https://github.com/major/MySQLTuner-perl?tab=readme-ov-file#performance-schema-setup

Pour l'utiliser il faut avoir un compte qui a accès à toutes les bases. Pour le lancer soit avec `perl mysqltuner.pl` ou le rendre exécutable `chmod +x mysqltuner.pl`

```
./mysqltuner.pl --user mysqltuner --pass <password> --host <ip> --verbose --color
```

```
General recommendations:
        Add at least a primary key on table powerdns.migrations
        Add at least a primary key on table powerdns.records_zone_templ
```

```
        Ensure that all table(s) get an explicit primary keys for performance, maintenance and also for replication
        Ensure that all text colums(s) are UTF-8 compliant for encoding support and performance
        Remove unused indexes.
        Configure your accounts with ip or subnets only, then update your configuration with skip-name-resolve=ON
        Be careful, increasing innodb_log_file_size / innodb_log_files_in_group means higher crash recovery mean time
Variables to adjust:
        skip-name-resolve=ON
        innodb_log_file_size should be (=32M) if possible, so InnoDB total log file size equals 25% of buffer pool size.
        innodb_log_buffer_size (> 16M)
        wsrep_slave_threads = 16
        gcs.fc_limit= wsrep_slave_threads * 5 (=80)
        gcs.fc_factor=0.8
        innodb_flush_log_at_trx_commit = 0
        set up parameter wsrep_notify_cmd to be notified
        set up parameter wsrep_sst_method to xtrabackup based parameter
```

Les recommandations concernant la base powerdns ne peuvent pas s'appliquer car il s'agit de la base pour la gestion des zones DNS ce qui pourrait poser des problèmes d'incompatibilité

3. Application des recommandations

Les paramètres se mettent dans `/etc/mysql/my.cnf` dans `[mysqld]`

```
skip-name-resolve = ON
innodb_log_buffer_size = 32M
innodb_flush_log_at_trx_commit = 0
```

Les autres paramètres sont spécifiques à galera donc il doivent se mettre dans `/etc/mysql/mariadb.conf.d/60-galera.cnf` dans le groupe `[galera]`

```
wsrep_slave_threads = 16
wsrep_sst_method = xtrabackup
wsrep_provider_options = "gcs.fc_limit=80;gcs.fc_factor=0.8"
```

4. Principaux points

- MySQLTuner est un outil simple, léger et puissant pour diagnostiquer les performances de serveurs MySQL/MariaDB.

- Il propose des recommandations concrètes et souvent directement applicables couvrant l'ensemble des aspects du système de base de données : moteurs de stockage, configuration de clusters Galera, structure des bases et des tables, ainsi que les paramètres système liés aux performances et à la sécurité.

12.11. Fiche 10 : Audit de sécurité des systhèmes avec Lynis

1. Installation de Lynis

Ajout du dépôt

- Via le dépôt de lynis :

```
sudo apt install curl gpg


curl -fsSL https://packages.cisofy.com/keys/cisofy-software-public.key | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/cisofy-software-public.gpg

echo "deb [arch=amd64,arm64 signed-by=/etc/apt/trusted.gpg.d/cisofy-software-public.gpg] https://packages.cisofy.com/community/lynis/deb/ stable main" | sudo tee /etc/apt/sources.list.d/cisofy-lynis.list

sudo apt update


sudo apt install lynis
```

2. Lancer un audit de sécurité

L'audit de base du système s'exécute avec la commande :

```
sudo lynis audit system
```

Cela déclenche un scan complet de la configuration du système, des permissions, des services réseau, de la journalisation, etc.

3. Résultats du scan après l'installation

```
Lynis security scan details:

 Hardening index : 64 [#############             ]
```

4. Application de correctifs

Certains correctif peuvent être appliquer difficilements voir pas applicable dans mon cas

- Renforcement des permissions par défaut
  Définir *UMASK 027* dans */etc/login.defs*

  Évite que "others" lisent les nouveaux fichiers créés

- Vérification des versions des paquets

```
apt install apt-show-versions
```

  Permet une meilleure traçabilité des versions des paquets

- Message d'avertissement à la connexion
  Ajout dans */etc/issue* et */etc/issue.net* + */etc/ssh/sshd_config*

  Dissuasion légale + bannière SSH

- Durcissement de la configuration SSH

```
AllowTcpForwarding no → Bloque le port forwarding SSH (sécurité)
ClientAliveCountMax 2 → Ferme les connexions SSH inactives rapidement
LogLevel VERBOSE →Journalise les tentatives d'authentification (utile)
MaxAuthTries 3 →Max essais mot de passe avant coupure
MaxSessions 2 →Limite les connexions parallèles sur une même session SSH
AllowAgentForwarding no → Bloque transfert de clé SSH via agent
X11Forwarding no → Désactive l'accès à des applications graphiques
```

- Désactivation de protocoles réseaux inutilisés
  Blacklist des modules dccp, sctp, rds, tipc dans un fichier dans */etc/modprobe.d/unused-protocol.conf*

  install <protocole> /bin/true → Réduit la surface d'attaque

- Blocage des périphériques non utilisés (USB/Firewire)
  Ajout dans */etc/modprobe.d/blacklist.conf* : *blacklist usb-storage* et blacklist firewire-ohci → Évite les fuites ou vols de données

- Vérification d'intégrité des paquets
  apt install debsums + config de la vérification via cron (*/etc/default/debsums*)

- Permissions sécurisées sur fichiers sensibles

```
chmod 600 /etc/ssh/sshd_config
chmod 600 /etc/crontab
chmod 600 /etc/crontab
chmod 700 /etc/cron.d
chmod 700 /etc/cron.daily
chmod 700 /etc/cron.hourly
chmod 700 /etc/cron.weekly
chmod 700 /etc/cron.monthly
```

5. Résultats obtenues après durcissement

```
Lynis security scan details:
Hardening index : 74 [###############            ]
```

6. Principaux points

- Lynis est un outil puissant pour auditer les serveurs Unix/Linux.

- Il fournit un indice de durcissement, des suggestions pour améliorer la sécurité.

- Il est essentiel de consulter le fichier de log généré (/var/log/lynis.log) pour un suivi détaillé des recommandations.