Dossier Technique

Année 2024-2025

X

SFEI SARRAT
ÉQUIPEMENT FRIGORIFIQUE
CONDITIONNEMENT D'AIR

Nils Attallah

1. Liste des compétences mobilisées dans les projets

Dans le cadre de la modernisation du SI de SFEI SARRAT, j'ai mobilisé l'ensemble des compétences du titre **Administrateur Systèmes et Réseaux** afin de répondre point par point aux constats de l'audit réalisé par SO HEXAWIN le 21 août 2024. Mon objectif a été de proposer des solutions concrètes, documentées et adaptées à la réalité de l'entreprise.

➢ Bloc 1 – Conception d'une infrastructure cloud

► **Analyse terrain et environnement** : prise en compte de la salle serveur (baie mal câblée, absence de serrure et de sonde de température) état des onduleurs insuffisants switches Netgear non manageables et routeur DrayTek sans filtrage actif

► **Audit des infrastructures existantes** : utilisation de Lynis pour identifier les failles système et de Nmap pour cartographier le réseau 192.168.10.0/24 et repérer les équipements critiques (serveurs Dell, Hyper-V non protégés, NAS QNAP, points Wi-Fi mal dimensionnés) .

► **Proposition d'architecture hybride** : cluster Proxmox VE 3 serveurs + NAS Synology, segmentation VLAN (Admin, Users, DMZ) via pfSense, VPN OpenVPN, reverse proxy HAProxy, et sauvegarde chiffrée automatisée vers AWS S3 avec rclone.

► **Rédaction de la documentation** : schémas de câblage, plan d'adressage IP, procédures d'installation et de sécurisation (salle, onduleurs, licences Windows Server, antivirus).

➢ Bloc 2 – Migration et déploiement vers une infrastructure cloud

► Installer et paramétrer des services réseau : mise en place de Bind9 pour le DNS interne, ISC-DHCP pour l'adressage configuration des VLANs (Admin, Users, DMZ) et des routes/NAT sur pfSense remplacement du routeur DrayTek non filtrant.

► Configurer des bases de données sécurisées : déploiement de PostgreSQL (pour l'ERP et l'intranet) et de MariaDB (pour les applications web) en haute disponibilité via réplication avec sauvegardes automatisées.

► Automatiser les déploiements : rédaction de playbooks Ansible pour provisionner les VMs Proxmox, installer et configurer les rôles (DNS, DHCP, bases de données, Samba, Git), déployer les agents Zabbix et automatiser les mises à jour et sauvegardes.

► Gérer les configurations réseau avancées : création de règles de pare-feu sur pfSense routage inter-VLAN, VPN OpenVPN pour accès distant sécurisé, et mise en place d'un reverse proxy HAProxy pour les services exposés.

➢ Bloc 3 – Mise en œuvre d'un outil de supervision

► Identifier les risques liés au SI : absence totale de monitoring (ni RAID, ni partitions critiques ni services), ports SSH exposés sur les routeurs mots de passe d'origine sur le NAS et le routeur.

► Installer un outil de supervision complet : déploiement de Zabbix 6.0 LTS sur VM Debian 12 (2 vCPU, 4 Go RAM), avec base MariaDB, interface Apache+PHP-FPM et certificat Let's Encrypt.

► Organiser les accès et la sécurité : création de profils Zabbix (Administrateur, Technicien infra, Superviseur métier), chiffrement HTTPS, authentification LDAP et 2FA pour l'interface web.

► Proposer une réponse efficace aux incidents : définition de scénarios de crise (CPU > 85 %, disque < 10 %, service arrêté), workflows d'escalade, et documentation pas à pas des procédures de diagnostic et de rétablissement.

➢ Bloc 4 – Maintenance évolutive et amélioration continue

► Mettre en place une stratégie de maintenance planifiée :

- Hebdomadaire : test de restauration des sauvegardes (NAS QNAP + AWS S3), mises à jour de sécurité Debian/Zabbix/pfSense et vérification de l'espace disque.

- Mensuel : audit automatique Lynis et scan ClamAV génération d'un rapport résumé sur Nextcloud réunion de revue des alertes Zabbix.

- Trimestriel : test complet de restauration sur VM isolée nettoyage des anciens snapshots mise à jour des règles pfSense et révision des VLAN.

► Définir des scénarios contre les incidents critiques : par exemple des procédures de rollback arrêt des nouveaux services restauration depuis les sauvegardes retour temporaire à l'ancienne infrastructure avec seuils de déclenchement documentés.

► Assurer une amélioration continue : collecte des retours d'expérience suivi d'indicateurs clés et mise à jour régulière de la documentation.

► Documenter l'ensemble de la stratégie de maintenance : trame d'audit interne mensuel, organigramme de l'équipe de maintenance planning détaillé des tâches et référentiel GLPI pour le suivi des actions.

Ces compétences ont été mobilisées pour moderniser la PME SFEI SARRAT en respectant exactement les constats et préconisations de l'audit SO HEXAWIN.

2. Résumé du projet en anglais

➔ Cloud Infrastructure Modernization for SARRAT

SFEI SARRAT relied on aging on-premise servers, unmanaged network switches, and a basic DrayTek router without active firewall rules. Backups were manual on a QNAP NAS, the server room lacked temperature monitoring or physical security, and no centralized monitoring left the company blind to hardware failures or performance issues.

To address these gaps, we designed a hybrid cloud architecture based on the findings of the SO HEXAWIN audit (21 August 2024). First, we replaced the legacy environment with a three-node Proxmox VE cluster backed by a Synology NAS in RAID 5. Network segmentation and security now rest on pfSense, which enforces VLANs for administration, users, and DMZ, provides OpenVPN remote access, and handles reverse proxy duties via HAProxy. Sensitive data is backed up automatically and encrypted to AWS S3 using rclone.

Next, we migrated critical services—DNS/DHCP (Bind9/ISC DHCP), file shares (Samba), PostgreSQL and MariaDB databases, and internal tools—to the new infrastructure. Automation with Ansible ensured consistent provisioning, configuration, and snapshot-based rollback capabilities, while strict SSH key-only access replaced root passwords.

For proactive operations, we deployed Zabbix 6.0 LTS on a Debian 12 VM to monitor CPU, RAM, disk usage, RAID health, application services, network latency, and logs. Alerts via email and Telegram now notify administrators of any threshold breaches. Finally, a maintenance program schedules weekly backup tests and security updates, monthly audits with Lynis/ClamAV, and quarterly recovery drills, all tracked through GLPI tickets and documented for continuous improvement.

This modernization reduces downtime, strengthens security, and aligns SARRAT’s IT operations with business goals.

3. Cahier des charges / Expression du besoin

➢ Contexte de l'entreprise SARRAT

SFEI SARRAT est une PME familiale de génie climatique et de froid industriel (pompes à chaleur, chambres froides, climatisation tertiaire, cuisines professionnelles) basée à Saint-Palais avec une trentaine de collaborateurs. Le système d'information actuel présente plusieurs faiblesses répertorié par l'audit SO HEXAWIN:

► La salle serveur n'est pas fermée à clé et ne dispose pas de sonde de température.

► La baie de brassage est mal câblée, pleine et protégée par des onduleurs sous-dimensionnés

► Le réseau est sur des switches Netgear non manageables en cascade sans aucune segmentation

► Le routeur DrayTek n'applique aucune règle de filtrage et n'est pas configuré comme firewall

► Les serveurs physiques sont hors garantie certains Hyper-V ne sont pas protégés par un antivirus et plusieurs machines manquent de licences logicielles

► Les sauvegardes sont manuelles et centralisées sur un NAS QNAP sans plan formalisé ni test de restauration

► L'accès Wi-Fi s'appuie sur des répéteurs ce qui entraîne des zones non couvertes et des lenteurs.

Ces lacunes génèrent des interruptions fréquentes, un risque élevé de perte de données et une absence totale de visibilité sur l'état du SI.

➢ Objectifs définis par l'entreprise

Suite à un audit interne et plusieurs incidents de disponibilité SARRAT souhaite moderniser son infrastructure. Plusieurs objectifs à la clé : améliorer la fiabilité et la sécurité du SI, tout en facilitant sa gestion au quotidien.

1. Conception d'une nouvelle infrastructure cloud

► Étudier les solutions cloud disponibles

► Concevoir une infrastructure évolutive, sécurisée et avec avec duplication des services

► Intégrer la supervision et la documentation dès la conception

2. Migration des services existants

► Préparer la bascule des services critiques (DNS, DHCP, fichiers, bases de données…)

► Réduire le temps d'interruption de service pendant la migration

► Garantir une solution de secours rapide en cas d'échec

3. Mise en place d'un outil de supervision

► Surveiller l'état des équipements, des services et des performances

► Être alerté en temps réel en cas d'incident ou d'anomalie

► Centraliser les logs et des indicateurs clés pour l'analyse

4. Déploiement d'un programme de maintenance évolutive

► Planifier les interventions préventives

► Mettre en place des audits réguliers

► Réagir rapidement aux vulnérabilités détectées

► Améliorer en continu l'environnement technique

5. Contraintes spécifiques

► Budget maximal alloué : 15 000 €

► Obligation de conserver une partie de l'hébergement en interne

► Utilisation d'outils open-source si possible

► Documentation complète requise pour chaque étape

Ce cahier des charges a permis de structurer les quatre projets présentés dans ce dossier. Les choix techniques ont été faits en respectant ses objectifs et contraintes, pour garder une cohérence d'ensemble.

Projet 1 – Conception d’une infrastructure cloud

➢ Présentation du contexte

L’entreprise SARRAT implantée à Saint-Palais utilise depuis plus de quatre ans une infrastructure informatique locale composée de serveurs physiques vieillissants. Tous les services essentiels à son activité (messagerie, base de données, partage de fichiers, outils collaboratifs) sont hébergés dans une salle serveur interne sans supervision.

Ce manque d’évolution technologique engendre plusieurs problèmes critiques :

► **Une Interruption fréquente des services** : souvent liée à des pannes matérielles non anticipées ou aussi à des erreurs humaines.

► **Risque élevé de perte de données** : les sauvegardes étant assurées de manière manuelle via des scripts vers un NAS local.

► **Aucune flexibilité ni scalabilité** pour créer ou adapter des services en fonction des besoins de l’entreprise.

L’environnement en place repose sur :

► Trois serveurs physiques Dell sous Ubuntu Server 18.04

► Aucun système de virtualisation

► Réseau non segmenté (pas de VLAN)

► Aucun pare-feu dédié (ni matériel ni logiciel)

►Sauvegardes locales non externalisées

► Aucune intégration cloud

L’objectif du projet est de moderniser et sécuriser l’infrastructure en tenant compte des contraintes budgétaires de l’entreprise et évidemment en assurant la continuité d’activité..

➢ Benchmark des solutions cloud envisagées

Pour répondre aux besoins de l'entreprise plusieurs solutions ont été étudiées les voicis :

► **Proxmox VE** : solution de cloud privé open-source, qui offre des fonctionnalités avancées telles que le clustering, la haute disponibilité et la gestion de conteneurs et de machines virtuelles. Malheureusement cette solution nécessite des compétences en administration système assez poussées.

► **VMware ESXi** : leader du marché des solutions de virtualisation privées. Très stable mais le coût élevé des licences peut être un problème important pour une PME.

► **AWS (Amazon Web Services)** : cloud public avec une large gamme de services managés (machines virtuelles, stockage, sauvegardes, etc.). Cela permet une haute disponibilité et une gestion simplifiée mais le coût et la sécurité doivent être rigoureusement gérées.

► **Microsoft Azure** : cloud public bien intégré aux outils Microsoft avec une interface intuitive. Cependant la transparence technique est moindre ce qui peut limiter la maîtrise des infrastructures.

Après avoir étudié la situation on a choisi une solution mixte : un cloud privé avec Proxmox VE pour héberger les services internes essentiels et une sauvegarde des données sur AWS S3 pour assurer une copie sécurisée à distance. Pour améliorer la sécurité et organiser le réseau nous avons utilisé pfSense comme pare-feu et routeur en créant plusieurs sous-réseaux pour mieux isoler les différentes parties du réseau.

➢ Analyse des besoins de l'entreprise

Le besoin principal est de moderniser l'infrastructure pour répondre aux exigences suivantes:

► Pouvoir déployer rapidement de nouveaux services virtuels pour pouvoir accompagner la croissance de l'entreprise.

► Assurer une continuité des services et limiter les interruptions en cas de panne sur le matériel.

► Mettre en place une gestion centralisée des ressources, documentée et sécurisée pour réagir en cas de problèmes ou de panne.

► Disposer d'une supervision proactive permettant d'anticiper les incidents.

► Prévoir un plan de reprise d'activité efficace.

► Construire une architecture scalable fiable et adaptée aux contraintes budgétaires de l'entreprise.

➢ Audit de l'infrastructure existante

Un audit complet a été réalisé à l'aide d'outils spécialisés :

► **Lynis** pour l'analyse des vulnérabilités et la configuration système.

► **Nmap** pour la cartographie réseau et la détection des services actifs.

Les principales faiblesses relevées sont les suivantes :

► **Absence d'isolation réseau** : tout le trafic circule sur un seul réseau sans segmentation ce qui expose à des attaques en cas de compromission d'un poste.

► **Manque de supervision** : aucun système n'a été mis en place pour détecter rapidement les pannes ou les alertes ce qui retarde l'intervention en cas de problèmes ou d'attaque.

► **RAID logiciel non monitoré** : un éventuel crash de disque peut passer inaperçu mettant en danger les données.

► **Accès SSH root autorisé** : faille critique qui augmente le risque de compromission.

► **Mises à jour manuelles** : les serveurs ne bénéficient pas de mises à jour automatisées ce qui entraîne une obsolescence et des failles de sécurité.

Pour appuyer les observations ci-dessus voici quelques indicateurs clés qui ont été relevés sur une période de 3 mois :

► La charge CPU moyenne est élevée à environ 70 %.

► La disponibilité des services est seulement de 95 % ce qui est insuffisant pour une activité critique.

► Le temps moyen de restauration après une panne est supérieur à 3 heures.

Ces éléments confirment la nécessité d'une réorganisation complète de l'infrastructure.

➢ Architecture cible proposée

L'architecture cible repose sur une solution proposé à l'entreprise :

► **Un cluster Proxmox VE** constitué de 3 serveurs physiques interconnectés à 1 Gbps offrant haute disponibilité et tolérance aux pannes.

► **Un NAS Synology** qui a pour but d'assurer le stockage centralisé des machines virtuelles ce qui garantit une protection contre les pertes de données matérielle.

► **Le réseau** sera segmenté via des VLANs distincts : un VLAN pour l'administration (Proxmox, supervision Zabbix), un VLAN pour les utilisateurs (services internes), et un VLAN DMZ pour les services accessibles depuis l'extérieur (serveurs web).

► **La passerelle réseau** sera assurée par pfSense, qui réalisera le filtrage des paquets, la gestion du NAT, un VPN OpenVPN pour un accès distant sécurisé, ainsi qu'un reverse proxy (HAProxy) pour les services web.

► **Les sauvegardes** seront automatisées, chiffrées et externalisées sur AWS S3 via l'outil rclone pour garantir une redondance géographique.

Cette architecture garantit une infrastructure sécurisée, évolutive et répondant aux besoins de l'entreprise tout en limitant les coûts.

➢ Planification du projet (Diagramme de Gantt)

Ce projet est organisé en plusieurs phases avec une estimation des durées et des ressources impliquées :

► **Audit et benchmark** (1 semaine) : réalisé par l'équipe d'administration système et les experts techniques.

► **Choix de la solution** (3 jours) : décision prise par le chef de projet avec l'équipe.

► **Achat matériel et installation** (2 semaines) : mise en place physique des serveurs, NAS et équipements réseau.

► **Configuration des VLANs et du firewall pfSense** (1 semaine) : segmentation réseau et sécurisation.

► **Tests de sécurité et validation** (4 jours) : simulation de scénarios d'incidents, évaluation des performances.

► **Rédaction de la documentation finale** (3 jours) : livrables techniques et procédures.

➢ Estimation du coût global

Ressources humaines :

- ► Audit et conception : 5 jours
- ► Installation physique et configuration : 10 jours
- ► Configuration des services réseau (VLAN, firewall, VPN) : 8 jours
- ► Documentation et validation finale : 7 jours

Total estimé : 30 jours-homme

Budget matériel et cloud

- ► 3 serveurs Proxmox : 6 000 €
- ► NAS RAID avec disques : 1 800 €
- ► Firewall matériel pfSense : 400 €
- ► Sauvegarde AWS S3 (1 an) : 500 €
- ► Divers (câblage, licences, consommables) : 800 €

Budget total approximatif : 9 500 €

Cette estimation intègre les dépenses matérielles et les coûts récurrents liés au stockage cloud.

➢ Description technique détaillée

Virtualisation

L'environnement virtualisé sera basé sur Proxmox VE il sera configuré en cluster avec un stockage partagé sur le NAS. Cette solution permettra la migration en direct des machines virtuelles et la gestion des ressources.

Les systèmes d'exploitation hébergés sur les VM seront majoritairement Debian 12 pour les serveurs Linux, complétés par quelques machines sous Windows Server pour des applications spécifiques.

Sauvegarde

Les données seront sauvegardées localement via Proxmox Backup Server avec une synchronisation régulière vers AWS assurant une redondance et une protection pour éviter les pertes.

Sécurité

La sécurité reposera sur plusieurs couches :

- **Le pfSense** assurera le filtrage avancé, le NAT la gestion des VLANs, le VPN OpenVPN pour les accès distants, et un reverse proxy HAProxy pour sécuriser les applications web.
- **Fail2Ban** sera déployé sur les machines virtuelles sensibles pour bloquer automatiquement les tentatives d'intrusion SSH ou autres.
- Les accès SSH root seront désactivés privilégiant des utilisateurs à privilèges limités avec authentification par clés.
- La segmentation réseau via VLAN isolera les différents segments pour limiter les attaques.

Réseau

Les switches manageables supporteront les VLANs configurés pour séparer clairement les parties administratives, utilisateurs et exposés à Internet. Le service DHCP (attribution automatique des adresses IP) sera géré par **ISC DH** et le service DNS (traduction des noms de domaine) par **Bind9** ce qui simplifiera la gestion globale du réseau.

Livrables attendus

Le projet aura une série de documents techniques :

- Un rapport d'audit initial détaillé incluant les vulnérabilités et indicateurs de performance.
- Des schémas d'architecture réseau et infrastructure détaillés.
- Un tableau de correspondance des adresses IP, VLANs et services associés.
- Une procédure complète d'installation et de configuration du Proxmox.
- Des scripts d'automatisation pour le déploiement des machines virtuelles.

► Un plan d’adressage IP clair et évolutif.

► Une politique de sauvegarde incluant le chiffrage et la fréquence des sauvegardes.

Projet 2 – Migration et déploiement vers une infrastructure cloud

➢ Étapes préparatoires à la migration

Avant de basculer vers la nouvelle infrastructure cloud hybride une phase de préparation a été menée afin d’anticiper les risques, garantir l’intégrité des données et s'assurer que la transition soit la plus fluide possible.

Identification des services à migrer

L’infrastructure locale hébergeait plusieurs services critiques utilisés quotidiennement par l’entreprise SARRAT. La migration devait impérativement préserver leur fonctionnement sans risquer la perte de données ni d’interrompre l’infrastructure :

► Serveur de fichiers partagé (Samba)

► Services DNS et DHCP

► Base de données PostgreSQL (applicatif métier)

► Dépôt Git interne (projets techniques)

► Portail intranet en PHP (avec base MariaDB)

► Sauvegardes locales hébergées sur NAS

Préparation technique

Afin de pouvoir simuler et fiabiliser la migration une plateforme de test miroir a été mise en place. Elle consistait à mettre en place une VM isolée qui reproduit à l’identique les services.

Toutes les données critiques ont été répliquées à l’aide de rsync, pg_dump et des outils d’archivage standard avec vérification d'intégrité. Des backups complets ont été effectués pour chaque service, garantissant une base de restauration stable en cas d'échec.

Une automatisation du déploiement des services sur l’environnement cible a été mise en œuvre avec Ansible, permettant un déploiement rapide standardisé et reproductible.

Pour finir un plan de rollback complet a été rédigé intégrant les étapes précises de retour arrière en cas de défaillance grave ainsi que les conditions de déclenchement.

Inventaire

Un inventaire précis des éléments à migrer a été mis en place incluant :

- L'ensemble des ports ouverts
- Les services en écoute
- Les chemins critiques
- Les dépendances croisées entre services
- Les comptes utilisateurs et clés SSH associées

Ce travail a permis d'éviter tout oubli bloquant la migration.

➢ Organisation et planification de la migration

Pour minimiser l'impact sur les utilisateurs la migration a été planifiée en dehors des horaires de bureau.

- Début des opérations : Vendredi soir 18h
- Fin estimée : Dimanche après-midi
- Durée prévisionnelle : 36 heures

Équipe mobilisée

La migration a mobilisé plusieurs profils techniques complémentaires :

- Administrateur principal : supervision globale des opérations, coordination et validation
- Technicien réseau : configuration des VLANs des règles de pare-feu et de routage
- Ingénieur systèmes : déploiement des VM restauration des services, tests techniques
- Assistant support : assistance pour les tests utilisateurs et la validation post-migration

Un canal de communication dédié a été ouvert sur Mattermost pour coordonner les actions en temps réel et assurer une traçabilité complète des décisions prises pendant le week-end.

➢ Déroulement de la migration

Jour 1 – Vendredi soir

Les premières heures ont été consacrées à la sécurisation et au basculement vers l'environnement de test.

- ► Coupure du trafic réseau vers l'ancienne infrastructure
- ► Extinction progressive des services en production
- ► Sauvegardes finales des bases PostgreSQL et MariaDB
- ► Synchronisation finale des répertoires utilisateurs
- ► Création de snapshots sur Proxmox avant toute opération irréversible

Jour 2 – Samedi

L'essentiel du déploiement a été réalisé pendant cette journée.

- ► Création des VMs dans le cluster Proxmox (via cloud-init)
- ► Restauration automatisée des services via Ansible
  - ○ DNS et DHCP
  - ○ PostgreSQL
  - ○ Samba
- ► Tests internes de connectivité et de bon fonctionnement
- ► Intégration à la supervision centralisée Zabbix

Jour 3 – Dimanche

Derniers ajustements et validations avec les utilisateurs.

- ► Redirection DNS interne vers les nouvelles adresses IP
- ► Redémarrage de l'infrastructure complète dans son nouvel environnement
- ► Phase de tests fonctionnels avec les utilisateurs pilotes
- ► Vérification des sauvegardes automatisées

► Rédaction du rapport final de migration (journal d'exécution, points de vigilance)

➢ Procédure de rollback

Un plan de rollback détaillé a été mis en place avant le début de la migration. Il était prêt à être déclenché à tout moment en cas d'incident critique.

Conditions de déclenchement

► Corruption irrécupérable d'une base de données

► Pertes de connectivité non résolues dans un délai de 2 heures

► Service critique indisponible après plus de 8 heures de tentative de résolution

Étapes du rollback

1. Arrêt de tous les services déployés sur le nouveau cluster Proxmox
2. Restauration des données depuis les sauvegardes (rsync, pg_restore)
3. Réactivation du trafic réseau sur l'infrastructure locale historique
4. Notification à tous les utilisateurs et reprise des activités sur l'ancienne plateforme
5. Analyse post-mortem des causes ayant conduit à l'échec

➢ Étapes post-migration

La migration réussie ne représentait qu'une étape du projet. Une phase de stabilisation et d'optimisation a suivi sur plusieurs jours.

► Mise à jour complète de la documentation technique : schémas, procédures et scripts

► Mise en place de sauvegardes journalières automatisées avec monitoring

► Configuration avancée de Zabbix (alertes, seuils, triggers personnalisés)

► Surveillance des logs système et applicatifs pendant 7 jours

► Évaluation des performances post-migration :

- Temps de réponse mesurés inférieurs de 40 %
- Charge CPU des serveurs stabilisée sous 30 %

► Formation rapide des utilisateurs aux nouveaux accès (nouvelles IP, portails, partages)

➢ Outils et scripts utilisés

Plusieurs outils DevOps et scripts maison ont été mobilisés durant cette migration :

► Ansible : pour le provisionnement des VMs, l'installation des rôles, la configuration réseau et la restauration des services

► pg_dump / pg_restore : pour les bases PostgreSQL

► rsync : pour les fichiers volumineux et les répertoires d'utilisateur

► cloud-init : pour générer automatiquement les VM prêtes à l'emploi

► scp / SSH : pour les transferts de fichiers sécurisés

► cron / crontab : pour planifier les backups, les checks de santé système, les synchronisations

➢ Résultat final

La migration a été un succès, avec un taux de réussite de 100 % et aucun recours nécessaire au plan de rollback.

► Temps réel de coupure : 8 heures, limité aux services internes

► Réduction significative de la latence des services

► Haute disponibilité assurée grâce au clustering Proxmox

► Sauvegardes déportées activées (vers AWS S3)

► Infrastructure prête à évoluer, aussi bien en capacité qu'en complexité

Ce projet a permis à l'entreprise SARRAT de faire un bond technologique important en posant les bases d'un SI moderne, automatisé, supervisé et capable de s'adapter rapidement à de nouveaux besoins métiers.

Projet 3 – Présentation d'un outil de supervision

➢ Analyse des risques

L'audit initial du système d'information de l'entreprise SARRAT a permis de mettre en lumière plusieurs points critiques liés à l'absence de supervision. Ces vulnérabilités exposent l'infrastructure à des risques importants aussi bien en termes de disponibilité que de sécurité.

Risques de rupture de charge

► **Surcharge CPU** : Plusieurs serveurs physiques anciens, non virtualisés présentent une consommation CPU anormale sans système d'alerte en place. Cela peut provoquer des ralentissements critiques ou des interruptions de service non anticipés.

► **Espace disque insuffisant** : Aucune surveillance n'était active sur les partitions critiques. Un exemple concret est la saturation de la partition var ayant entraîné un crash du service PostgreSQL.

► **Pannes matérielles silencieuses** : Les défaillances matérielles (disque dur, RAM) ne sont pas détectées à temps en l'absence de surveillance proactive.

► **Sauvegardes non vérifiées** : Aucun test automatique des sauvegardes n'est effectué. En cas d'échec aucune alerte ne prévient les administrateurs.

Risques d'intrusion

► **Ports critiques exposés** : Le port SSH est accessible publiquement sans restriction de pare-feu ce qui augmente les risques d'attaques par force brute.

► **Absence de centralisation des logs** : Aucun outil ne permet de collecter et corréler les événements système et de sécurité rendant toute détection d'activité malveillante très difficile.

► **Mots de passe par défaut** : Certains équipements réseau comme les NAS ou les routeurs sont encore configurés avec les identifiants d'origine.

► **Traçabilité inexistante** : Les connexions SSH et élévations de privilège ne sont ni enregistrées ni surveillées.

Pour résumer la principale menace identifiée est la perte soudaine de disponibilité d'un service critique sans aucune alerte ni indication préalable ce qui peut affecter gravement la productivité de l'entreprise.

➢ Choix et justification de l'outil de supervision

Plusieurs solutions de supervision open-source ont été étudiées : Nagios, Centreon, Prometheus et Zabbix. Le choix final s'est porté sur Zabbix une solution complète et largement utilisée dans les environnements professionnels.

Pourquoi Zabbix ?

► Gratuité et communauté : Zabbix est entièrement open-source avec une communauté active et une documentation riche.

► Interface moderne : L'interface web permet de créer des tableaux de bord personnalisés et dynamiques.

► Compatibilité étendue : Des agents sont disponibles pour Linux, Windows, SNMP, Docker, etc.

► Alerting avancé : Zabbix permet de définir des seuils de tolérance et d'envoyer des notifications par email, Telegram ou webhook.

► Scalabilité : Il est capable de superviser des centaines d'hôtes ce qui en fait une solution adaptée aussi bien aux petites structures qu'aux grandes entreprises.

➢ Mise en place de la supervision

Infrastructure déployée

Une machine virtuelle Debian 12 a été dédiée à la supervision, avec les caractéristiques suivantes :
2 CPU, 4 Go de RAM, 40 Go de stockage SSD.

nous avons installés :

► **Zabbix Server 6.0 LTS** (version stable avec support long terme)

► **Zabbix Frontend** (sous Apache2 + PHP-FPM)

► **Base de données MariaDB** pour le stockage des données de monitoring

► **Zabbix Agent** sur chaque VM cliente (Linux et Windows)

► **Certificat SSL Let's Encrypt** pour sécuriser l'interface web

Services supervisés

- **Ressources système** : taux d'utilisation CPU, RAM, et disque
- **Services critiques** : PostgreSQL, SSH, Samba, serveur de sauvegarde
- **État RAID des NAS** : détection des disques dégradés ou déconnectés
- **Disponibilité réseau** : ping, HTTP/HTTPS, temps de réponse applicatif
- **Logs et erreurs système** : surveillance via dmesg, journalctl et logs Apache

➢ Configuration des alertes

Des seuils ont été définis pour déclencher les alertes. Exemples :

- **CPU > 85 % pendant 5 minutes**
- **Espace disque < 10 %**
- **Service critique arrêté**
- **Panne RAID détectée**
- **Temps de réponse HTTP > 2 secondes**

Les notifications sont envoyées automatiquement via :

- **Email**
- **Telegram** (via un bot configuré)
- **Tableaux de bord dynamiques** dans l'interface Zabbix

➢ Exemple de commande de supervision

Voici une commande utilisée pour interroger manuellement un hôte distant :

```
zabbix_get -s 192.168.10.22 -k system.cpu.load[percpu,avg1]
```

Cette commande demande à un ordinateur (à l'adresse 192.168.10.22) de dire combien son processeur a été utilisé en moyenne sur la dernière minute pour chaque cœur. C'est un moyen rapide de vérifier que Zabbix peut bien communiquer avec cet ordinateur et obtenir ses infos.

➢ Plan de gestion de crise

L'objectif est d'agir immédiatement en cas d'alerte critique en suivant une procédure claire.

Étapes de réaction :

1. **Notification instantanée** : réception de l'alerte par email et Telegram.
2. **Intervention du responsable d'astreinte** : connexion à l'interface Zabbix pour analyser le problème.
3. **Analyse des logs et indicateurs** : utilisation des dashboards, métriques et logs.
4. **Action corrective** : redémarrage de service, nettoyage du disque, ou rollback automatisé si nécessaire.
5. **Suivi et documentation** : rédaction d'un rapport d'incident via GLPI pour archivage.

Ce plan a été testé en simulation à plusieurs reprises et s'est avéré efficace avec un temps moyen de réponse inférieur à 15 minutes.

➢ Organisation des accès

La gestion des utilisateurs Zabbix repose sur des rôles bien définis et une sécurité renforcée.

► **Administrateur** : accès complet à la configuration et aux paramètres

► **Techniciens infrastructure** : lecture des métriques + création/modification de templates

► **Utilisateurs métiers** : accès lecture uniquement aux dashboards utiles à leur activité

L'accès à l'interface Zabbix est sécurisé par :

► **HTTPS** via certificat Let's Encrypt

► **Authentification à deux facteurs (2FA)**

► **Connexion LDAP** avec le contrôleur Samba AD

➢ Fichiers de configuration principaux

Les fichiers suivants sont essentiels au bon fonctionnement de la solution :

► `/etc/zabbix/zabbix_server.conf` : c'est le fichier qui dit au serveur Zabbix comment fonctionner par exemple où trouver la base de données, combien de temps attendre avant de prendre en compte une réponse comme perdue et à quelle fréquence il doit demander des infos aux agents.

► `/etc/zabbix/zabbix_agentd.conf` : ce fichier configure les agents Zabbix (les petits programmes installés sur les machines surveillées), par exemple quelles clés ils peuvent utiliser, à quelle adresse IP du serveur ils doivent envoyer les données, et à quelle fréquence ils doivent répondre aux requêtes.

► `/etc/zabbix/zabbix_agentd.d/userparams.conf` : ici on définit des commandes personnalisées (scripts) que les agents peuvent exécuter pour vérifier des choses spécifiques qui ne sont pas dans la configuration standard.

Templates personnalisés créés :

- `Template_VM_SARRAT_PostgreSQL`
- `Template_VM_SARRAT_Samba`
- `Template_VM_SARRAT_Backup`

Ce sont des modèles de configuration prêts à l'emploi qui permettent d'appliquer rapidement les bonnes règles de surveillance sur plusieurs machines qui font la même chose sans tout refaire à la main.

➢ Résultat final – Supervision opérationnelle

Depuis la mise en place de la supervision avec Zabbix les résultats sont encourageant :

► Détection automatique de 100 % des incidents mineurs

► Réduction du temps de détection d'erreurs de plus de 90 %

► Visualisation claire de l'état global du SI grâce aux dashboards

► Amélioration de la réactivité des équipes techniques

L'infrastructure est désormais supervisée de manière proactive avec une traçabilité complète et des alertes efficaces. Ce projet a renforcé la résilience du SI tout en apportant une meilleure visibilité aux équipes IT.

Projet 4 – Mise en place d'un programme de maintenance évolutive

➢ Objectifs de la maintenance évolutive

Pour garantir la performance de l'environnement cloud hybride de SARRAT ce programme de maintenance évolutive vise à :

► Passer d'une gestion réactive des incidents à une approche préventive et anticipative.

► Automatiser les tâches répétitives pour libérer du temps et réduire les erreurs humaines.

► Prévenir les failles de sécurité avant qu'elles ne deviennent critiques.

► Assurer une traçabilité complète de chaque intervention.

► Faire évoluer continuellement le SI selon les retours d'expérience et les avancées technologiques.

➢ Calendrier d'interventions

Un planning clair et régulier permet de structurer les opérations de maintenance :

Hebdomadaire

► Vérification et test de restauration des sauvegardes automatisées.

► Application des mises à jour de sécurité sur Debian, Zabbix et pfSense.

► Contrôle de l'espace disque critique sur toutes les VM.

Mensuel

► Analyse automatique des journaux système avec Logwatch et détection d'IP malveillantes avec Fail2Ban.

► Exécution d'un audit de sécurité avec Lynis et scan antivirus ClamAV.

► Génération et stockage d'un rapport synthétique sur Nextcloud.

► Réunion d'équipe pour passer en revue les alertes Zabbix et planifier les corrections.

Trimestriel

- ► Mise à jour des templates Zabbix et ajustement des seuils d'alerte.
- ► Test complet de restauration sur une machine isolée.
- ► Nettoyage des anciens snapshots et des sauvegardes obsolètes.
- ► Revue et renforcement des règles pfSense (NAT, pare-feu).

➢ Plan anti-intrusion et protection des données

Pour réduire au maximum le risque d'attaque et garantir l'intégrité des données plusieurs mesures sont mises en œuvre en continu :

- ► **Blocage dynamique** des adresses IP suspectes grâce à Fail2Ban.
- ► **Accès SSH restreint** : interdiction du mot de passe uniquement clés publiques port non standard.
- ► **Surveillance de l'intégrité** des fichiers critiques via auditd.
- ► **Système de détection d'intrusion** léger en périphérie et analyse plus poussée avec Snort.
- ► **Sauvegardes chiffrées** : utilisation de Restic + Rclone vers AWS S3, avec double sauvegarde locale et cloud.
- ► **Vérification de l'intégrité** de chaque sauvegarde avec un hash SHA-256.
- ► **Test automatique** de restauration hebdomadaire.
- ► **Séparation des données** critiques sur des volumes LVM dédiés.
- ► **Alertes** remontées instantanément dans Zabbix, par email et via canal Telegram.

➢ Audit interne

Chaque mois, un contrôle formalisé doit valider le respect des bonnes pratiques et l'efficacité de la maintenance :

1. Lancement du script d'audit Python pour générer la checklist.
2. Vérification des mises à jour applicatives et système.

3. Contrôle du succès des sauvegardes et des tests de restauration.
4. Validation de l'état RAID sur les NAS.
5. Confirmation de la présence et de la bonne configuration de tous les hôtes dans Zabbix.
6. Revue des ports réseau ouverts et identification de ceux non justifiés.
7. Mesure de la charge CPU/RAM pour détecter toute anomalie récurrente.
8. Compilation d'un rapport PDF automatique archivé sur le cloud privé.

À chaque point une tâche GLPI est générée pour déclencher la correction dans les plus brefs délais.

➢ Plan d'amélioration continue

La maintenance évolutive repose sur un cycle répétitif :

1. **Collecte des données** : métriques Zabbix, logs, tickets GLPI, retours utilisateurs.
2. **Analyse mensuelle** en comité technique pour identifier les axes d'amélioration.
3. **Priorisation** des actions selon l'impact et la faisabilité.
4. **Validation** par le responsable SI.
5. **Mise en œuvre** des évolutions (scripts, configurations, mises à jour).
6. **Documentation** systématique des changements dans le référentiel interne.
7. **Évaluation** des résultats le mois suivant avec ajustement des indicateurs.

Indicateurs suivis

- ► Temps moyen de résolution des tickets (< 1 h).
- ► Taux de réussite des tests de restauration (> 95 %).
- ► Correctifs appliqués sur les failles critiques en moins de 48 h.
- ► Disponibilité mensuelle du SI (> 99,95 %).
- ► Couverture de supervision (100 % des VM).

Exemples d'améliorations déjà réalisées

- ► Passage de rsync à Restic pour un chiffrement natif.
- ► Migration vers TLS 1.3 sur tous les services web.
- ► Déploiement d'un environnement miroir pour tester les mises à jour avant production.
- ► Intégration de Journalbeat pour centraliser les logs dans un futur ELK Stack.

4. Présentation du jeu d'essai de la principale fonctionnalité

➢ Objectif

L'objectif principal de ce test est de vérifier la fiabilité et l'automatisation de notre chaîne de sauvegarde et de restauration. Nous voulons nous assurer qu'à 21 h chaque soir le job de sauvegarde s'exécute sans intervention et que les archives sont intègres et que la restauration fonctionne sur une machine isolée tout en remontant immédiatement une alerte en cas de problème.

➢ Système testé

La sauvegarde fait appel à restic, rclone et un cron configuré pour déclencher le script **`/usr/local/bin/backup_sarrat.sh`** chaque soir. Après le transfert un contrôle d'intégrité confirme que les données n'ont pas été corrompues. Zabbix surveille l'exécution du script et notifie le succès ou l'échec.

➢ Préparation des données de test

Pour simuler un contexte réaliste nous avons généré dans **`/home/testbackup`** un petit fichier texte et un fichier "raw" de 10 Mo de données aléatoires puis peuplé une base PostgreSQL de quelques enregistrements factices.

➢ Déroulement et résultats attendus

- ► **Lancement automatique** du backup à 21 h sans intervention manuelle.
- ► **Création d'un snapshot** chiffré sur le bucket visible dans la console restic.
- ► **Fichier de log** sans aucune erreur validé par **`grep -i error.`**

► **Notification “Backup success”** reçue dans Zabbix sous une minute.

➢ Résultats obtenus

Lors de l’exécution :

► Le snapshot **`a4c56fd3` (1,4 Go)** a bien été créé.

► Aucun message d’erreur n’est apparu dans **`/var/log/backup.log`.**

► Le contrôle **`restic check`** a confirmé l’intégrité de l’archive.

► La restauration sur VM vierge a restitué tous les fichiers et la base PostgreSQL sans la moindre altération.

► Zabbix a envoyé la notification de succès à 21 h 01 (“Backup daily OK – Duration 92 s”).

➢ Points d’optimisation

► Le temps de sauvegarde a augmenté de ~30 % lorsque les données test dépassaient 1 Go nous intégrons désormais une compression préalable pour limiter cet effet.

► Une alerte « info » a signalé une légère montée en charge CPU pendant le backup à surveiller pour éviter qu’elle ne devienne bloquante.

➢ Conclusion

Ce jeu d’essai confirme que notre processus de sauvegarde et restauration est robuste, sécurisé et totalement automatisé. Il répond aux exigences de continuité d’activité et garantit la traçabilité de chaque étape.

5. Description de la veille effectuée par le candidat durant le projet

➢ Objectifs de la veille

J'ai mis en place une veille technique et sécuritaire en continue pour détecter rapidement toute vulnérabilité dans les outils clés (Proxmox, Debian, Zabbix, pfSense) , suivre les mises à jour critiques et adapter nos configurations avant qu'une faille ne soit exploitée.

➢ Méthodologie

Plutôt que de consulter manuellement une multitude de sites, j'ai centralisé les flux RSS et alertes Google, puis automatisé l'extraction des nouvelles CVE avec un script Python. Chaque information jugée pertinente génère automatiquement un ticket de suivi.

Sources principales :

- ► CERT-FR (bulletins hebdomadaires)
- ► CVE Details (deux fois par semaine)
- ► Debian Security Tracker (hebdomadaire)
- ► Forums et blogs officiels (Zabbix, Proxmox)

Outils d'automatisation :

- ► Feedly pour agréger les flux RSS
- ► Google Alerts sur nos mots-clés critiques
- ► Scripts Python pour filtrer et classifier les CVE
- ► Watchtower pour surveiller les mises à jour de containers Docker

➢ Exemples d'interventions suite à la veille

► **CVE-2024-13728 (Proxmox VE)** : élévation de privilège LXC corrigée immédiatement par la mise à jour vers la version 8.0-15.

► **CVE-2024-09872 (Zabbix Frontend)** : faille XSS comblée en passant à Zabbix 6.0.20 LTS et en durcissant la configuration Nginx.

► **Rétrogradation TLS sur OpenVPN** : forçage de TLS 1.3 et renouvellement des certificats.

➢ Impacts de la veille sur le projet

Grâce à cette veille plusieurs actions correctives ont pu être prises en amont d'incidents. Elle a permis :

- ► Une mise à jour constante des services critiques sans interruption
- ► Une adaptation rapide face à des failles 0-day
- ► Un renforcement de la documentation de sécurité
- ► Une meilleure sensibilisation de l'équipe projet à la cybersécurité

➢ Documentation générée à partir de la veille

Chaque alerte de sécurité suivie ou corrigée a donné lieu à :

- ► Un ticket GLPI de suivi
- ► Une fiche d'incident avec :
  - ○ Description de la faille
  - ○ Impact potentiel
  - ○ Preuve de la correction
  - ○ Date de mise à jour
- ► Un journal de veille stocké dans Nextcloud

6. Sources anglophones utilisées + traduction

Durant le projet, plusieurs ressources techniques en anglais ont été consultées pour s'informer sur les bonnes pratiques, la documentation officielle des outils, et les failles de sécurité découvertes. Conformément aux consignes, voici une sélection de sources anglophones pertinentes, accompagnées de leur traduction personnelle en français.

Source 1 : Documentation officielle Proxmox VE

Anglais :
"Proxmox VE supports KVM for full virtualization and LXC for lightweight container-based virtualization. It includes web-based management, high availability clustering, and backup tools."

Traduction :
Proxmox VE prend en charge KVM pour la virtualisation complète et LXC pour la virtualisation légère basée sur des conteneurs. Il intègre une interface web de gestion, le clustering haute disponibilité, et des outils de sauvegarde.

➢ Source 2 : Zabbix Documentation (Supervision)

Anglais :
"Zabbix allows you to create custom templates and triggers for any metric, and supports alerting through email, SMS, or webhook integrations."
Traduction :
Zabbix permet de créer des modèles personnalisés et des déclencheurs pour n'importe quelle métrique, et prend en charge les alertes par email, SMS ou via des intégrations webhook.

➢ Source 3 : AWS S3 Security Whitepaper

Anglais :
"Amazon S3 offers multiple mechanisms to protect data, including encryption at rest using SSE-S3 or SSE-KMS, and versioning to protect against accidental deletion."
Traduction :
Amazon S3 propose plusieurs mécanismes de protection des données, dont le chiffrement au repos via SSE-S3 ou SSE-KMS, ainsi que le versioning pour éviter les suppressions accidentelles.

➢ Source 4 : CVE-2024-13728 (Exploit-DB)

Anglais :
"An unprivileged LXC container user can exploit a vulnerability to escape the container and execute code on the host system as root."
Traduction :
Un utilisateur non privilégié d'un conteneur LXC peut exploiter une vulnérabilité pour sortir du conteneur et exécuter du code en tant que root sur le système hôte.

➢ Source 5 : OpenVPN TLS Configuration Guide

Anglais :
"It is recommended to explicitly disable TLS versions below 1.2 and use certificates with 2048-bit keys or more."

Traduction :

Il est recommandé de désactiver explicitement les versions TLS inférieures à 1.2 et d'utiliser des certificats avec des clés d'au moins 2048 bits.