Rapport Détaillé - SecureIT Monitor

1. Introduction

Ce rapport présente une vue d'ensemble complète du projet SecureIT Monitor, un outil .exe conçu pour la supervision et la protection de l'infrastructure informatique d'une entreprise. Il inclut la gestion de crises, des alertes, des quarantaines, ainsi que le déploiement de Firewall et VPN. L'outil est installé sur un seul poste, celui de l'Administrateur IT, et intègre les outils NetXMS, Wazuh, ELK Stack, OPNsense et WireGuard. Le développement utilise Python, PyQt et PyInstaller.

2. Analyse des Besoins

Identification des besoins fonctionnels et techniques pour répondre aux exigences du projet.

2.1 Besoins Fonctionnels

- **Supervision en Temps Réel** : L'Administrateur IT doit pouvoir surveiller les métriques des équipements (CPU, RAM, disque, réseau) pour identifier rapidement les problèmes.
- **Détection et Gestion des Crises** : Détection des anomalies (via IA) et gestion des incidents via alertes et quarantaine automatique.
- **Sécurité Réseau** : Déploiement et gestion de Firewall et VPN pour protéger les communications et isoler les menaces.
- **Gestion Multi-Clients** : Le Super Administrateur doit gérer plusieurs clients, comparer leurs métriques, et générer des rapports consolidés.
- **Sauvegardes et Restauration** : Planification, suivi et restauration des sauvegardes avec alertes en cas d'échec.
- **Installation Simplifiée** : Automatisation de l'installation des agents sur les équipements via IP.
- **Interface Intuitive** : Interface utilisateur claire pour les deux rôles (Administrateur IT et Super Administrateur) avec des onglets adaptés.

2.2 Besoins Techniques

- **Plateforme** : L'outil doit fonctionner sur un seul poste Windows (Administrateur IT).
- **Langage et Outils** : Développement en Python avec PyQt pour l'interface, PyInstaller pour générer un .exe.
- **Intégration d'Outils** : NetXMS, Wazuh, ELK Stack, OPNsense, et WireGuard doivent être intégrés via leurs API ou commandes shell.
- **Stockage** : Base de données SQLite pour les données locales (équipements, clients, configurations).
- **Sécurité** : Chiffrement des données sensibles (identifiants) avec `cryptography`.
- **Réseau** : Accès sécurisé aux équipements via VPN (OPNsense/WireGuard).

3. Architecture du Projet

Description de l'architecture logicielle pour assurer la modularité et la scalabilité.

- **Couche Présentation** : Interface utilisateur développée avec PyQt, comprenant deux tableaux de bord (Administrateur IT et Super Administrateur). Les onglets sont dynamiquement chargés selon le rôle de l'utilisateur.
- **Couche Logique** :
  - Gestion des rôles et des privilèges (Administrateur IT vs Super Administrateur) via une variable interne et SQLite.
  - Intégration des outils externes :
    - **NetXMS** : API REST pour la supervision.
    - **Wazuh** : API pour les alertes et réponses actives.
    - **ELK Stack** : Requêtes Elasticsearch pour l'analyse des logs.
    - **OPNsense/WireGuard** : API pour le firewall et VPN.
  - Analyse prédictive via un modèle scikit-learn entraîné sur les logs Wazuh/ELK.
- **Couche Données** :
  - Base SQLite locale pour stocker les équipements, clients, licences, et configurations.
  - Logs centralisés dans Elasticsearch (ELK Stack) pour l'analyse inter-clients et les rapports.
- **Flux de Données** :
  1. NetXMS collecte les métriques des équipements et les transmet à l'interface (Surveillance).
  2. Wazuh détecte les menaces, envoie des alertes à l'interface (Alertes) et des logs à ELK Stack.
  3. ELK Stack analyse les logs pour l'IA (Analyse Prédictive) et les rapports (Export Global).
  4. OPNsense gère le firewall et VPN (WireGuard) sur instruction de l'interface ou de Wazuh.
- **Déploiement** : L'outil est un .exe exécuté sur le poste de l'Administrateur IT, qui communique avec les serveurs NetXMS, Wazuh, et OPNsense via le réseau.

4. Explication Détaillée des Onglets et Processus

Description de chaque onglet des tableaux de bord Administrateur IT et Super Administrateur, avec leurs processus et intégrations d'outils.

4.1 Tableau de Bord Administrateur IT

Onglet	Description	Processus et Outils

Surveillance	Supervision en temps réel de tous les équipements de l'entreprise : utilisation CPU/RAM/disque, activité réseau, état des services critiques, disponibilité/uptime.	**NetXMS** : Collecte des métriques via des agents sur les équipements. Les données sont récupérées via l'API REST de NetXMS (`requests`). Affichage dans des tableaux PyQt (QTableWidget) ou graphiques. Les dépassements de seuils déclenchent des alertes. **Développement** : `psutil` utilisé comme solution de secours pour les métriques locales.
Alertes	Notifications pour les incidents critiques : échecs de sauvegarde, vulnérabilités détectées, tentatives d'accès bloquées par le firewall.	**Wazuh** : Les agents détectent les anomalies et envoient des alertes au gestionnaire Wazuh. Les logs sont récupérés via l'API Wazuh (`requests`). **ELK Stack** : Indexe les alertes dans Elasticsearch pour récupération. **Processus** : Alertes affichées dans un QTextEdit avec filtres (date, gravité).
Analyse Prédictive	Détection des comportements anormaux via IA : activité hors horaires, pics de charge soudains, connexions géolocalisées anormales.	**Wazuh** : Fournit les données brutes (logs d'activité utilisateur, connexions réseau). **ELK Stack** : Stocke les logs dans Elasticsearch. **Processus** : Un modèle scikit-learn (ex. : Isolation Forest) analyse les logs pour détecter les anomalies. Résultats affichés dans un QTableWidget.
Firewall	Gestion des règles de parefeu : création/modification des règles (ports, IP, protocoles), suivi des logs d'intrusion, quarantaine automatique des appareils suspects.	**OPNsense** : Gère les règles via API. Les logs d'intrusion sont collectés via syslog. **Wazuh** : Déclenche la quarantaine via l'API OPNsense. **Processus** : Formulaires PyQt (QLineEdit, QPushButton) pour gérer les règles. Logs affichés dans QTextEdit.

VPN	Gestion des connexions VPN pour les agents/techniciens : génération de clés WireGuard, attribution à des postes/utilisateurs, suivi des connexions actives, révocation si nécessaire.	**OPNsense + WireGuard** : WireGuard intégré dans OPNsense pour les tunnels VPN. Clés générées via l'API OPNsense. **Processus** : Connexions actives affichées dans QTableWidget. Boutons pour génération/révocation de clés.
Sauvegardes	Gestion des sauvegardes automatiques/manuelles via Bacula : planification de tâches, état des dernières sauvegardes, restauration à la demande, alertes en cas d'échec.	**Processus** : Tâches Bacula gérées via des commandes shell (`subprocess`). Statut affiché dans QTableWidget. Échecs déclenchent des alertes (Wazuh).
Résultats Nmap	Affichage des résultats de scans de ports et de vulnérabilités : ports ouverts, services exposés, résultats des scripts NSE (CVE), historique des scans par machine, rapport exportable.	**Processus** : Scans exécutés via `python-nmap`. Résultats affichés dans QTableWidget avec option d'export CSV (QPushButton).
Gestion des Équipements	Interface pour : ajouter de nouveaux postes (nom, IP, OS), supprimer/modifier les existants, installer des agents, lancer un scan Nmap à l'ajout.	**NetXMS** : Ajoute des équipements via API pour la supervision. **Wazuh** : Installe des agents via scripts. **Processus** : Formulaire PyQt (QLineEdit, QComboBox) pour les données des équipements. Scan Nmap déclenché à l'ajout (`python-nmap`).

4.2 Tableau de Bord Super Administrateur

Onglet	Description	Processus et Outils
Sélectionner une Entreprise	Menu déroulant pour choisir un client et basculer vers son tableau de bord Admin IT complet avec tous les droits (lecture/écriture).	**Processus** : Liste déroulante QComboBox alimentée par SQLite. La sélection recharge l'interface Admin IT pour le client choisi. **Développement** : Utilise `sqlite3` pour récupérer la liste des clients.

Gestion des Clients	Gestion de toutes les entreprises clientes : création, édition, suppression de clients, visualisation du nombre d'équipements, statut de la licence, historique de l'activité.	**Processus** : Tableau QTableWidget affichant les clients avec des boutons CRUD (QPushButton). Données stockées dans SQLite. Historique basé sur les logs Wazuh (API).
Attribution des Licences	Interface pour : générer des licences logicielles, définir durée/type d'accès, désactiver/renouveler une licence, suivre le statut de validité.	**Processus** : Formulaire PyQt (QLineEdit, QDateEdit) pour générer des licences. Licences stockées dans SQLite avec statut. **Développement** : Utilise `uuid` pour des clés uniques.
Basculer vers un Client	Action permettant d'accéder au tableau de bord d'un client spécifique (comme un Admin IT) avec tous les privilèges.	**Processus** : Bouton QPushButton recharge l'interface Admin IT pour le client sélectionné avec privilèges élevés. **Développement** : Similaire à "Sélectionner une Entreprise" avec un drapeau de mode privilège.
Analyse Inter-Clients	Vue transversale entre plusieurs clients : comparaison du volume d'alertes, taux de sauvegarde réussi, nombre d'équipements vulnérables, détection de modèles communs (ex. : même type d'attaque).	**Wazuh** : Collecte les alertes de tous les clients. **ELK Stack** : Analyse les données dans Elasticsearch. **Processus** : Graphiques PyQt (QChart ou `pyqtgraph`) pour les comparaisons. Modèles communs détectés via des requêtes Elasticsearch.
Export Global	Génération de rapports consolidés (PDF/CSV) pour tous les clients ou par filtre : par période, type d'alerte ou métrique, pour reporting interne ou audits.	**ELK Stack** : Récupère les données consolidées depuis Elasticsearch. **Processus** : Bouton QPushButton exporte en CSV (`pandas`) ou PDF (`reportlab`).
Configuration Avancée	Réglages système généraux : paramètres des agents, fréquences de scans, niveaux de sévérité, seuils IA, alertes automatiques.	**NetXMS/Wazuh/OPNsense** : Configure agents et scans via leurs API. **Processus** : Formulaire PyQt (QLineEdit, QSpinBox) pour ajuster les paramètres. Configurations sauvées dans SQLite.

5. Outils Utilisés et Leurs Rôles

Détail du rôle de chaque outil pour garantir une compréhension claire de leurs contributions.

- **NetXMS**
  - **Rôle** : Supervision en temps réel de l'infrastructure informatique.
  - **Contribution** : Collecte des métriques (CPU, RAM, disque, réseau) pour l'onglet Surveillance. Ajoute des équipements dans Gestion des Équipements. Fournit des données pour les alertes.
  - **Intégration** : Utilise l'API REST avec `requests` pour récupérer les métriques.
- **Wazuh**
  - **Rôle** : Détection des menaces, collecte de logs et réponses actives.
  - **Contribution** : Fournit des alertes (onglet Alertes), des données brutes pour l'IA (Analyse Prédictive), déclenche la quarantaine (Firewall), installe des agents (Gestion des Équipements), et suit l'activité utilisateur.
  - **Intégration** : Utilise l'API avec `requests` et `subprocess` pour les réponses actives.
- **ELK Stack (Elasticsearch, Logstash, Kibana)**
  - **Rôle** : Stockage centralisé, analyse et visualisation des logs.
  - **Contribution** : Stocke les logs Wazuh pour l'analyse IA (Analyse Prédictive), des comparaisons inter-clients (Analyse Inter-Clients), et la génération de rapports (Export Global).
  - **Intégration** : Utilise la bibliothèque `elasticsearch` pour les requêtes.
- **OPNsense**
  - **Rôle** : Gestion du pare-feu et des fonctionnalités réseau avancées.
  - **Contribution** : Gère les règles et logs du firewall (onglet Firewall), supporte WireGuard pour le VPN, permet la quarantaine.
  - **Intégration** : Utilise l'API avec `requests`.
- **WireGuard (Intégré dans OPNsense)**
  - **Rôle** : Connexions VPN sécurisées.
  - **Contribution** : Gère les tunnels VPN pour les agents/techniciens (onglet VPN), assure un accès distant sécurisé.
  - **Intégration** : Géré via l'API OPNsense.

6. Processus d'Installation Automatisée via IP

Description du processus automatisé pour installer les agents sur les équipements en utilisant leurs adresses IP, initié depuis la station de travail de l'Administrateur IT.

- **Contexte** : SecureIT Monitor est installé sur une seule station de travail de l'Administrateur IT, qui gère l'ensemble de l'infrastructure informatique.
- **Processus Général** :
  1. **Saisie des Données des Équipements** : Dans l'onglet Gestion des Équipements, l'Administrateur IT entre les détails des équipements (nom, IP, OS) via un formulaire PyQt. Les données sont stockées dans SQLite.

2. **Scan Réseau (Optionnel)** : Un scan réseau avec `python-nmap` pré-remplit les IPs des équipements actifs, réduisant la saisie manuelle.
3. **Vérification de la Connectivité** : L'outil vérifie l'accessibilité de l'équipement (ex. : ping ou scan de port).
4. **Installation des Agents** :
   - **Équipements Linux** : Utilise SSH (`paramiko`) avec des identifiants administratifs. Télécharge et installe les agents NetXMS et Wazuh, les configure pour communiquer avec leurs serveurs.
   - **Équipements Windows** : Utilise WMI (`wmi`) ou PowerShell pour déployer les agents à distance.
5. **Validation** : L'outil vérifie l'activité des agents en interrogeant les serveurs NetXMS/Wazuh via leurs API.
6. **Rétroaction** : Succès ou échec de l'installation est affiché dans un QTextEdit.

— **Considérations de Sécurité** : Les identifiants sont chiffrés avec `cryptography`. Un VPN (OPNsense/WireGuard) assure un accès sécurisé aux équipements.

— **Exemple de Code (Linux via SSH)** :

```
import paramiko

def install_agents(ip, username, password, netxms_server, wazuh_server
    try:
        ssh = paramiko.SSHClient()
        ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        ssh.connect(ip, username=username, password=password)

        # Installation de l'agent NetXMS
        ssh.exec_command("wget http://<depot>/nxagentd.deb && sudo dpk
        ssh.exec_command(f"echo 'Server={netxms_server}' | sudo tee -a
        ssh.exec_command("sudo systemctl restart nxagentd")

        # Installation de l'agent Wazuh
        ssh.exec_command(f"wget https://packages.wazuh.com/4.x/apt/poo
        ssh.exec_command("sudo systemctl restart wazuh-agent")

        ssh.close()
        return "Agents installés avec succès sur " + ip
    except Exception as e:
        return "Erreur lors de l'installation : " + str(e)
```

— **Limitations** : Nécessite un accès réseau, des identifiants administratifs, et un OS compatible. Le firewall OPNsense doit autoriser le trafic SSH/WMI.

7. Conclusion

Résumé des progrès du projet et des prochaines étapes. SecureIT Monitor offre une solution complète pour la supervision informatique, avec des tableaux de bord détaillés pour les rôles Administrateur IT et Super Administrateur. L'architecture modulaire et l'analyse des besoins garantissent une implémentation adaptée. L'intégration de NetXMS,

Wazuh, ELK Stack, OPNsense et WireGuard assure des fonctionnalités robustes. Le processus d'installation automatisée des agents via IP simplifie le déploiement, bien qu'il nécessite des configurations réseau sécurisées. Les prochaines étapes incluent le codage des onglets, les tests d'installation des agents, et la finalisation du .exe avec PyInstaller.