MSPR 1

Développement et sécurité informatique

Mathis / Khalifa / Kerim

Sommaire

Contexte

Contexte

- ➔ **NFL IT créé en 2020** par deux associés français, spécialisée en infogérance, audit et intégration IT pour des organisations multisites.
- ➔ **Contrat exclusif avec la NFL** (32 franchises, plusieurs millions de dollars).
- ➔ **100+ collaborateurs** (Kansas City, datacenter en France).
- ➔ **Enjeu** : Absence d'outils unifiés entraîne des déplacements coûteux et des délais de résolution élevés.
- ➔ **Projet "Seahawks Monitoring"** : Standardiser la collecte d'infos techniques, centraliser la supervision et réduire les interventions sur site.

01

Démarche Suivie

Démarche Suivie

01 — **Définition du besoin :** Comprendre les attendues, définir les tâches à effectuer…

02 — **Organisation :** Création d'un espace de travail commun, répartition et attribution des tâches…

03 — **Travail :** Réalisation des tâches listées étape par étapes, gestion du projet en parallèle.

04 — **Test & Documentation :** Regrouper le travail en commun, tester que tout fonctionne bien, documenter le projet.

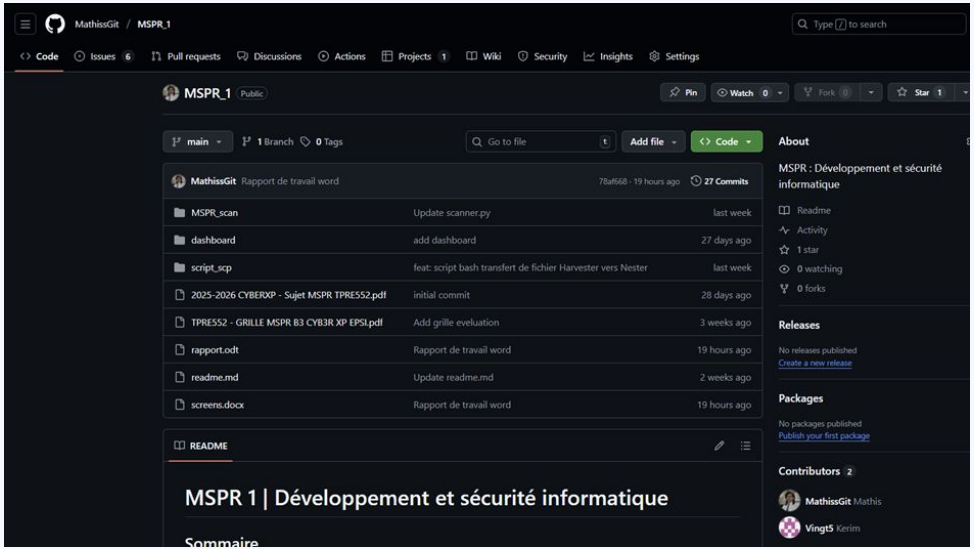
02

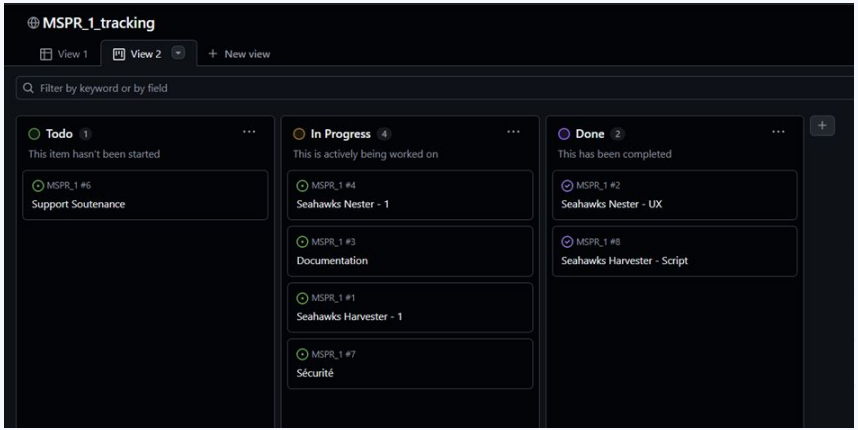
Solutions mise en place

Git & GitHub

> Utilisation de GitHub Project pour la gestion de projet et des tâches.

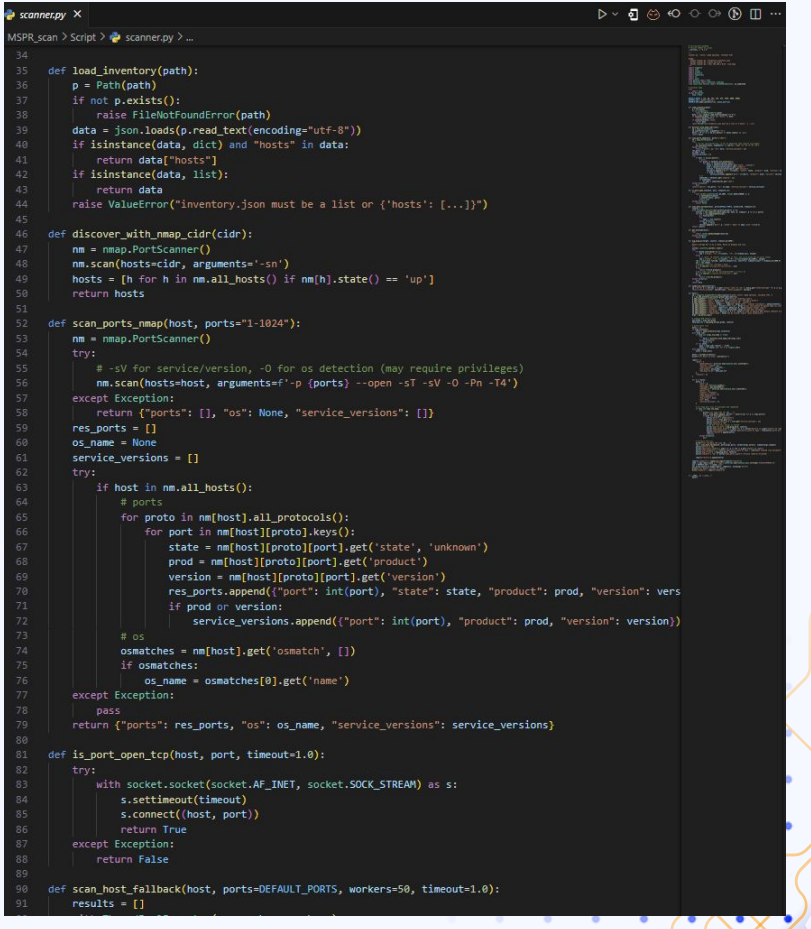
> Création d'un repository pour le partage des documents relatifs au projet.





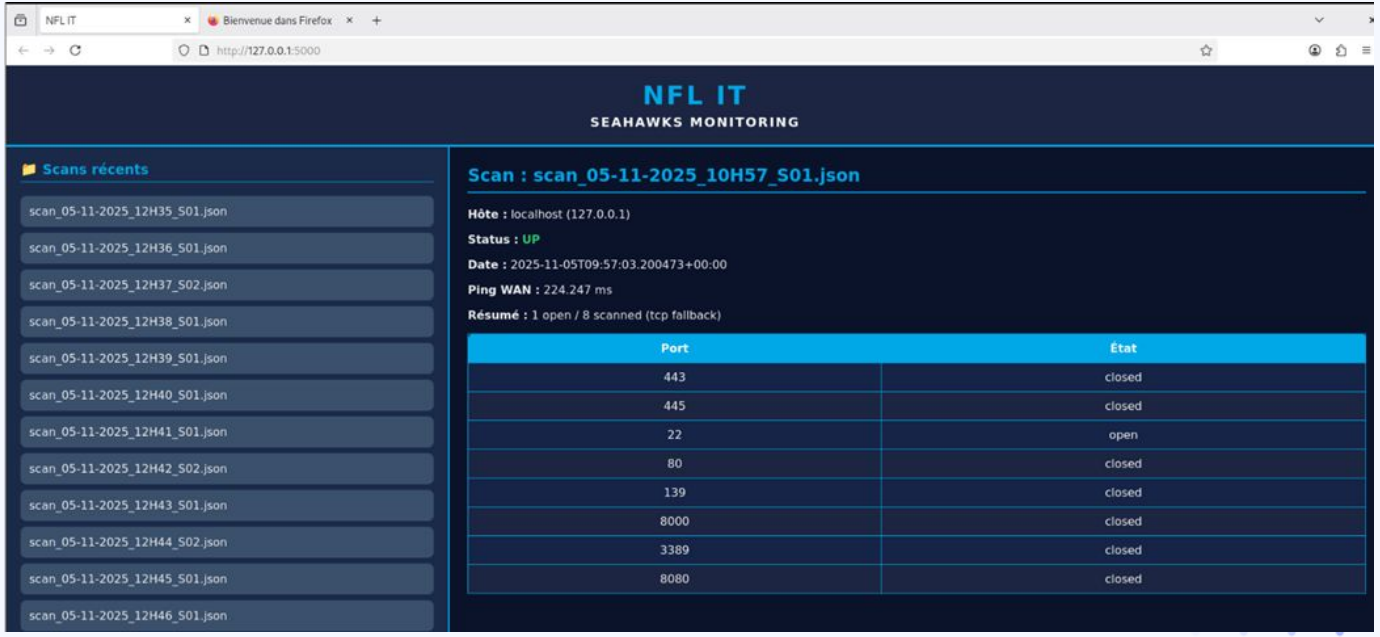
Python3

Les scripts pour le scan réseau est écrit en Python3 et utilise la bibliothèque « nmap ».



Flask

Nous avons aussi utiliser Flask pour le dashboard sur la machine “Harvester”



SCP & Inotify

SCP : Script Bash pour l'envoie de fichiers/rapports entre les machines.

Inotify : Détecte ou non la présence d'un nouveau fichier dans le dossier reports .

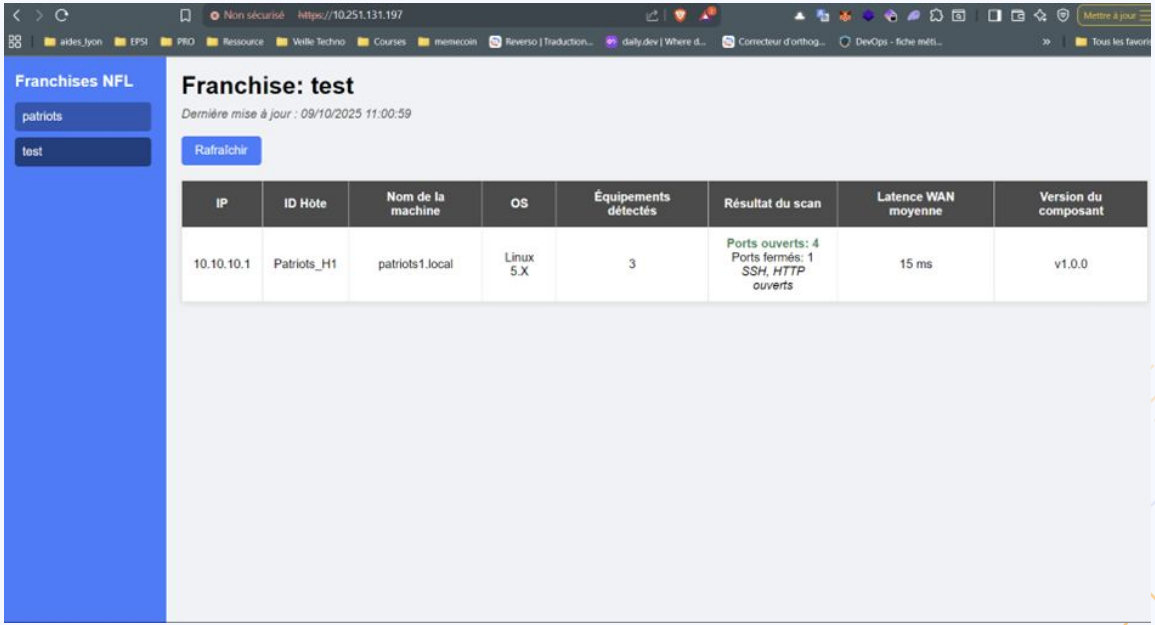
```
  GNU nano 8.4                                                    /root/MSPR_1/MSPR_scan/Scri
#!/bin/bash
# Surveillance automatique du dossier reports et envoi SCP vers Nester

SRC_DIR="/root/MSPR_1/MSPR_scan/Script/reports"
DEST_USER="adminNester"
DEST_HOST="172.20.10.4"
DEST_DIR="/var/www/dashboard/data/json"
SSH_KEY="/root/.ssh/id_ed25519_reports"
SSH_PORT=22334
LOG_FILE="/root/MSPR_1/MSPR_scan/Script/watch_and_send.log"

echo "=== Surveillance démarrée à $(date '+%Y-%m-%d %H:%M:%S') ===" >> "$LOG_FILE"

# Vérifie que le dossier de destination existe à distance
ssh -i "$SSH_KEY" -p "$SSH_PORT" -o StrictHostKeyChecking=no "${DEST_USER}@${DEST_HOST}" \
  "mkdir -p '$DEST_DIR' && chmod 755 '$DEST_DIR'"

# Surveillance continue du dossier source
inotifywait -m -e create --format '%w%f' "$SRC_DIR" | while read NEW_FILE
do
    if [[ "$NEW_FILE" == *.json ]]; then
        echo "[$(date '+%H:%M:%S')] Nouveau fichier détecté : $NEW_FILE" >> "$LOG_FILE"

        # Tentatives d'envoi avec retries
        for i in {1..3}; do
            scp -P "$SSH_PORT" -i "$SSH_KEY" -o StrictHostKeyChecking=no "$NEW_FILE" \
                "${DEST_USER}@${DEST_HOST}:${DEST_DIR}/" >> "$LOG_FILE" 2>&1 && {
                echo "[$(date '+%H:%M:%S')] ✅ Fichier transféré : $(basename "$NEW_FILE")" >> "$LOG_FILE"
                break
            }
            echo "[$(date '+%H:%M:%S')] ⚠️Tentative $i échouée, nouvelle tentative dans 5s..." >> "$LOG_FILE"
            sleep 5
        done
    fi
done
```

HTML/CSS/JS

Dashboard simple en HTML / CSS.

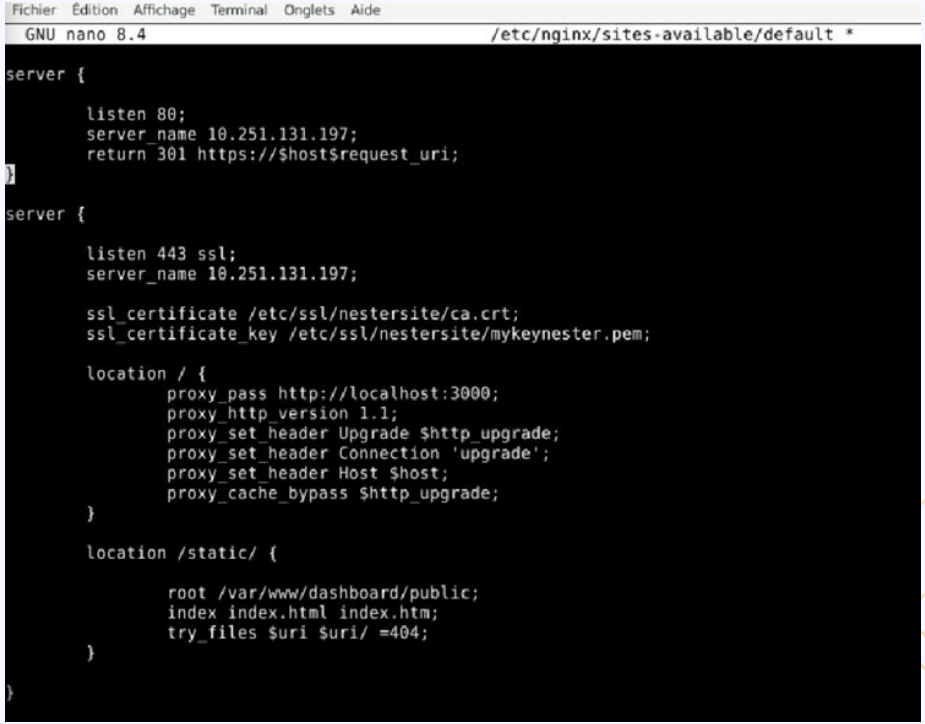
On utilise un script JS pour traiter les fichiers json reçu de la part des franchises.

Affichage de façon dynamique dans le DOM les informations



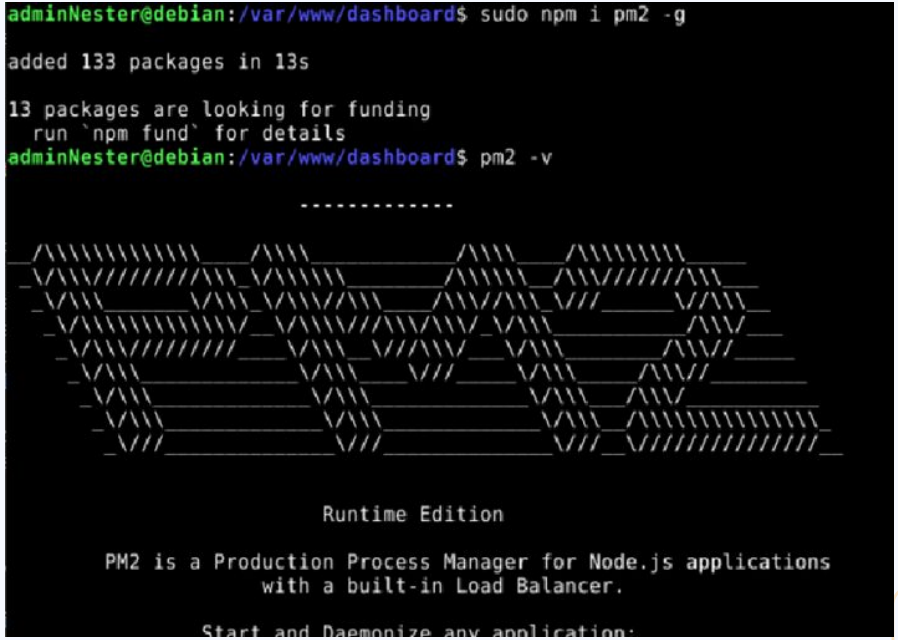
NGINX

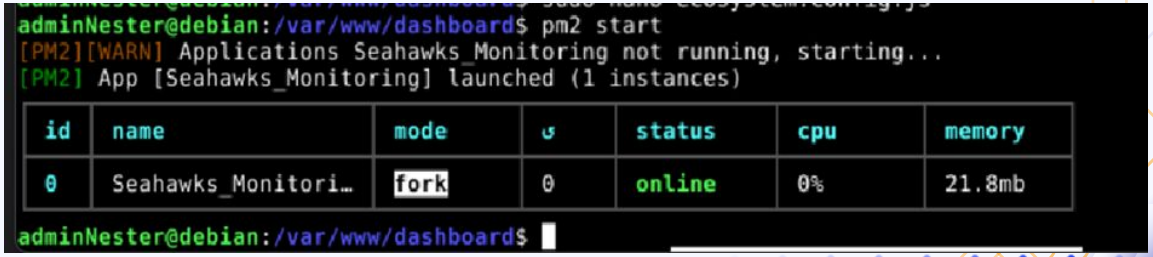
Hébergement du dashboard sur la machine « Nester »
+ Reverse-Proxy.



PM2

> Lancer l'environnement NodeJS au démarrage des machines virtuels
> Garder l'application toujours en ligne
> Redémarrer automatiquement l'application si elle plante ou si le serveur redémarre
> Faciliter la gestion et le monitoring.





03

Difficultés Rencontrées

Difficultés Rencontrées

Firewall & réseau école :

- Connexions bloquées
- Transferts limités
- Difficulté à identifier l'origine des blocages
- Perte de temps

Automatisation & Script SCP :

- Problèmes de configurations
- Gestion des permissions
- Chemins d'accès
- Execution automatique

04

Résultats Obtenus

SEAHAWKS NESTER

Seahawks Nester

Plateforme centrale pour consulter les sondes, leur état, le dernier rapport et les indicateurs collectés.

- ➔ **Serveur web Nginx / Consultation sur navigateur web**
- ➔ **Tableaux de bord (interfaces avec HTML / CSS / JS)**
  - **Listes des sondes par franchises**
- ➔ **Monitoring et disponibilité continue avec PM2**

SEAHAWKS HARVESTER

Seahawks Harvester

Composant local (sonde) qui collecte des données réseau (IP, nb équipements, latence, etc.), génère un rapport avec version, stocke les résultats et les transmets

➔ Scripts Python pour la collecte et l'analyse des données réseaux

- IP / ID hôte
- WAN / Ports
- Nbs équipements détectés
- Version (scan_05-11-2025_10H57_S01.json)

➔ Tableaux de bord local (interfaces avec Flask)

➔ Script Bash pour l’envoie et la collecte de fichiers entre les machines

readme script

05

Perspectives

Perspectives d'Améliorations 1/2

- ➔ Ajouter les runbook d'exploitation et la documentation sur les dashboard client.
- ➔ Créer et mettre en place un certificat SSL avec un nom de domaine pour les dashboard hébergé en ligne
- ➔ Authentification et gestion de session (OWASP A2) :
  - Mettre en place un système d'authentification forte pour les diffèrent sites (Nester/Harvester) en utilisant OAuth ou des tokens JWT
  - Utilisation de sessions sécurisées avec des cookies (''session'' dans Flask)
- ➔ Chiffrement des informations sensible (OWASP A3) qui peuvent être présent dans les scripts ou en ligne -> utilisation de AES

Perspectives d’Améliorations 2/2

- ➔ Utiliser des mots de passe fort en production
- ➔ Contrôle des rôles attribué aux utilisateurs (administrateur, exécution des scripts, etc…)
- ➔ Faille XSS : Ajouter des headers CSP dans la configuration Nginx pour restreindre les sources de scripts exécutables et éviter l’injection de JavaScript malveillant.
- ➔ Logging et surveillance (OWASP A10) :
  - Activer la journalisation dans Flask et Nginx
  - Surveiller les tentatives de connexion avec ‘’fail2ban‘’

Thanks!

Credits :

mathis.sineux@ecoles-epsi.net

k.ndour1@ecoles-epsi.net

kerim.unal@ecoles-epsi.net