

MSPR 1

Développement et sécurité informatique

Mathis / Khalifa / Kerim

Sommaire

01

Démarche suivie

02

Difficultés
rencontrées

03

Solutions mises en
place

04

Résultats obtenus

05

Perspectives

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

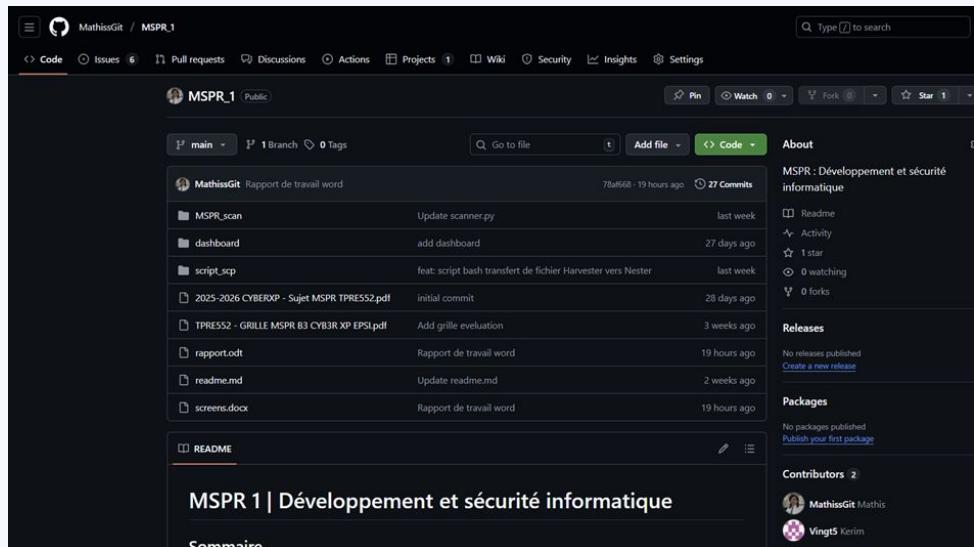
- 
- 01** ————— **Définition du besoin :** Comprendre les attendues, définir les tâches à effectuer...
 - 02** ————— **Organisation :** Création d'un espace de travail commun, répartition et attribution des tâches...
 - 03** ————— **Travail :** Réalisation des tâches listées étape par étapes, gestion 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.



Todo	In Progress	Done
Support Soutenance	Seahawks Nester - 1 Documentation Seahawks Harvester - 1 Sécurité	Seahawks Nester - UX Seahawks Harvester - Script

Python3

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

```
scanner.py <input>
MSPR_scan > Script > scanner.py > ...
34
35     def load_inventory(path):
36         p = Path(path)
37         if not p.exists():
38             raise FileNotFoundError(path)
39         data = json.loads(p.read_text(encoding="utf-8"))
40         if isinstance(data, dict) and "hosts" in data:
41             return data["hosts"]
42         if isinstance(data, list):
43             return data
44         raise ValueError("inventory.json must be a list or {'hosts': [...]}")
45
46     def discover_with_nmap_cidr(cidr):
47         nm = nmap.PortScanner()
48         nm.scan(hosts=cidr, arguments="-sn")
49         hosts = [h for h in nm.all_hosts() if nm[h].state() == 'up']
50         return hosts
51
52     def scan_ports_nmap(host, ports="1-1024"):
53         nm = nmap.PortScanner()
54         try:
55             # -sV for service/version, -O for os detection (may require privileges)
56             nm.scan(hosts=host, arguments=f"-p {ports} --open -sT -O -Pn -T4")
57         except Exception:
58             return {"ports": [], "os": None, "service_versions": []}
59         res_ports = []
60         os_name = None
61         service_versions = []
62         try:
63             if host in nm.all_hosts():
64                 # ports
65                 for proto in nm[host].all_protocols():
66                     for port in nm[host][proto].keys():
67                         state = nm[host][proto][port].get('state', 'unknown')
68                         prod = nm[host][proto][port].get('product')
69                         version = nm[host][proto][port].get('version')
70                         res_ports.append({"port": int(port), "state": state, "product": prod, "version": version})
71                         if prod or version:
72                             service_versions.append({"port": int(port), "product": prod, "version": version})
73                 # os
74                 osmatches = nm[host].get('osmatch', [])
75                 if osmatches:
76                     os_name = osmatches[0].get('name')
77             except Exception:
78                 pass
79         return {"ports": res_ports, "os": os_name, "service_versions": service_versions}
80
81     def is_port_open_tcp(host, port, timeout=1.0):
82         try:
83             with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
84                 s.settimeout(timeout)
85                 s.connect((host, port))
86                 return True
87         except Exception:
88             return False
89
90     def scan_host_fallback(host, ports=DEFAULT_PORTS, workers=50, timeout=1.0):
91         results = []
```

Flask

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

The screenshot shows a Firefox browser window displaying a Flask application. The title bar reads "NFL IT" and "Bienvenue dans Firefox". The address bar shows "http://127.0.0.1:5000". The main content area has a dark blue header with the text "NFL IT" and "SEAHAWKS MONITORING". On the left, there is a sidebar titled "Scans récents" containing a list of JSON files: "scan_05-11-2025_12H35_S01.json", "scan_05-11-2025_12H36_S01.json", "scan_05-11-2025_12H37_S02.json", "scan_05-11-2025_12H38_S01.json", "scan_05-11-2025_12H39_S01.json", "scan_05-11-2025_12H40_S01.json", "scan_05-11-2025_12H41_S01.json", "scan_05-11-2025_12H42_S02.json", "scan_05-11-2025_12H43_S01.json", "scan_05-11-2025_12H44_S02.json", "scan_05-11-2025_12H45_S01.json", and "scan_05-11-2025_12H46_S01.json". The main content area displays the details for the selected scan: "Scan : scan_05-11-2025_10H57_S01.json", "Hôte : localhost (127.0.0.1)", "Status : UP", "Date : 2025-11-05T09:57:03.200473+00:00", "Ping WAN : 224.247 ms", and "Résumé : 1 open / 8 scanned (tcp fallback)". Below this, a table lists port numbers and their status:

Port	État
443	closed
445	closed
22	open
80	closed
139	closed
8000	closed
3389	closed
8080	closed

SCP & Inotify

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

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

```
GNU nano 8.4
#!/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 retraits
        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 reçus de la part des franchises.

Affichage de façon dynamique dans le DOM les informations

The screenshot shows a web browser window with a dashboard titled "Franchise: test". The URL is https://10.251.131.197. The dashboard has a sidebar on the left labeled "Franchises NFL" with buttons for "patriots" and "test". The main content area displays a table of network scan results:

IP	ID Hôte	Nom de la machine	OS	Équipements détectés	Résultat du scan	Latence WAN moyenne	Version du composant
10.10.10.1	Patriots_H1	patriots1.local	Linux 5.X	3	Ports ouverts: 4 Ports fermés: 1 SSH, HTTP ouverts	15 ms	v1.0.0

NGINX

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

```
Fichier Édition Affichage Terminal Onglets Aide
GNU nano 8.4
/etc/nginx/sites-available/default *

server {
    listen 80;
    server_name 10.251.131.197;
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl;
    server_name 10.251.131.197;

    ssl_certificate /etc/ssl/nestersite/ca.crt;
    ssl_certificate_key /etc/ssl/nestersite/mykeynester.pem;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location /static/ {
        root /var/www/dashboard/public;
        index index.html index.htm;
        try_files $uri $uri/ =404;
    }
}
```

PM2

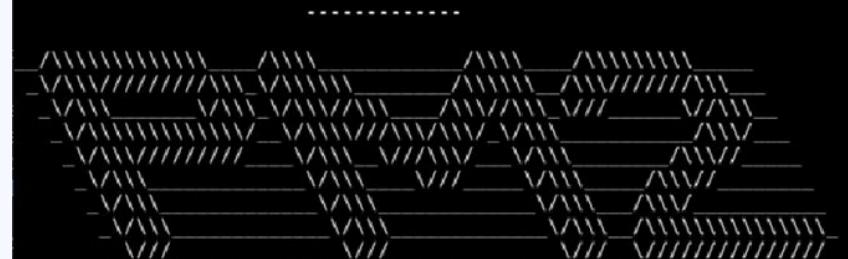
- > Lancer l'environnement NodeJS au démarrage des machines virtuels
 - > Garder l'application toujours en ligne
 - > Redémarrer automatiquement l'application si elle plante ou si le serveur redémarre
 - > Faciliter la gestion et le monitoring.

```
adminNester@debian:/var/www/dashboard$ sudo npm i pm2 -g
```

added 133 packages in 13s

13 packages are looking for funding
run `npm fund` for details

```
adminNester@debian:/var/www/dashboard$ pm2 -v
```



Runtime Edition

PM2 is a Production Process Manager for Node.js applications
with a built-in Load Balancer.

Start and Daemonize any application

```
adminNester@debian:/var/www/dashboard$ pm2 start  
[PM2][WARN] Applications Seahawks_Monitoring not running, starting...  
[PM2] App [Seahawks Monitoring] launched (1 instances)
```

id	name	mode	u	status	cpu	memory
0	Seahawks_Monitoring	fork	0	online	0%	21.8mb

adminNester@debian:/var/www/dashboards

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 Obtenu

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

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 différent 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

CREDITS: This presentation template was created by [Slidesgo](#), and includes icons, infographics & images by [Freepik](#)