# Document Technique – Scanner de Vulnérabilités Réseau en Go

Ce document présente le développement d'un scanner de sécurité réseau en Go, capable de scanner les ports ouverts, d'identifier les services en cours d'exécution, de vérifier les failles connues via la base CVE, et de générer des rapports détaillés en JSON et HTML. Ce projet offre une base solide pour de futures extensions et améliorations.

Tom - Ilies


# Plan d'action du Scanner de Vulnérabilités

1. **Analyse des besoins**

   Définition des fonctionnalités du scanner et documentation.

2. **Mise en place du projet**

   Initialisation du module Go et structure des fichiers.

3. **Scan des ports ouverts**

   Développement du scan TCP des ports sur une plage d'IP.

4. **Identification des services**

   Récupération des bannières des services.

Tom - Ilies

# Technologies Mises en Place

## Langage Go

Langage rapide et performant pour le réseau.

## Packages Go Utilisés

- net: Scanner les ports ouverts
- time: Gérer les délais de connexion
- encoding/json: Génération du fichier JSON
- html/template: Génération du fichier HTML
- net/http: Communication avec l'API CVE
- strconv: Conversion de types

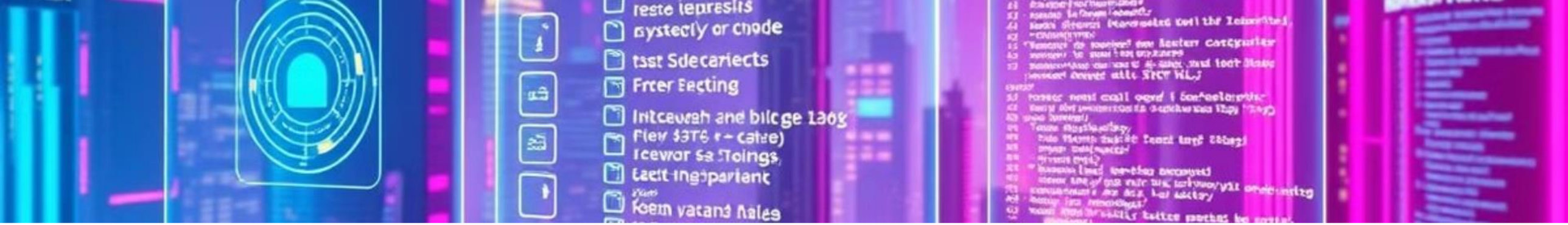
# API & Bases de Données

## NVD (National Vulnerability Database)

API REST pour récupérer les CVEs associées aux services détectés.

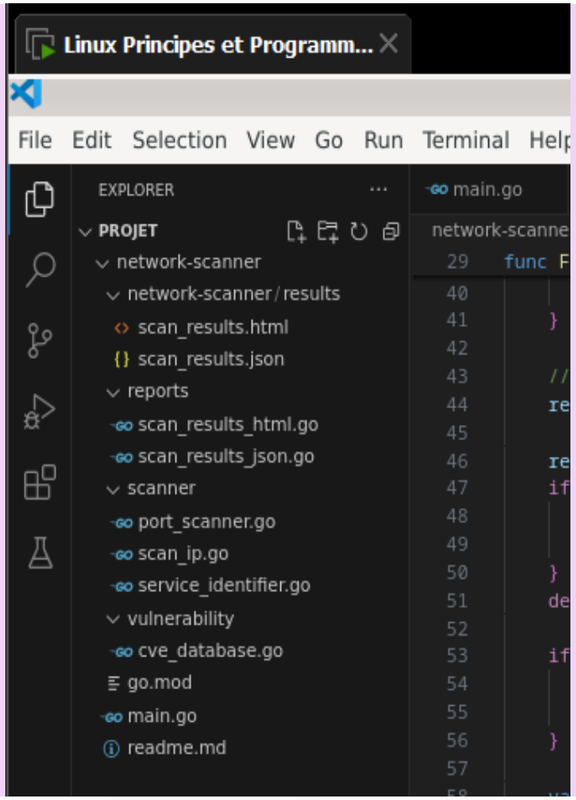
## Outils de Développement

- VS Code: Éditeur de code
- GoLand: Alternative pour le développement Go
- Vmware Workstation: Outil de Virtualisation de poste de travail
- Wireshark: Analyse des paquets réseau
- Linux (Debian): Environnement de test



# Arborescence du Projet

```
network-scanner/
├── go.mod
├── go.sum
├── main.go
├── scanner/
│   ├── scan_ip.go
│   ├── port_scanner.go
│   ├── service_identifier.go
├── reports/
│   ├── scan_results_json.go
│   ├── scan_results_html.go
├── vulnerability/
│   ├── cve_database.go
├── network-scanner/results/
│   ├── scan_results.json
│   ├── scan_results.html
```



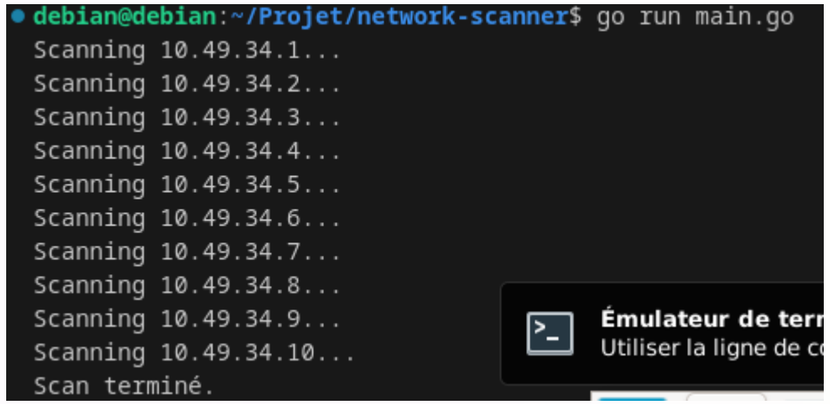
# Exemples d'Exécution

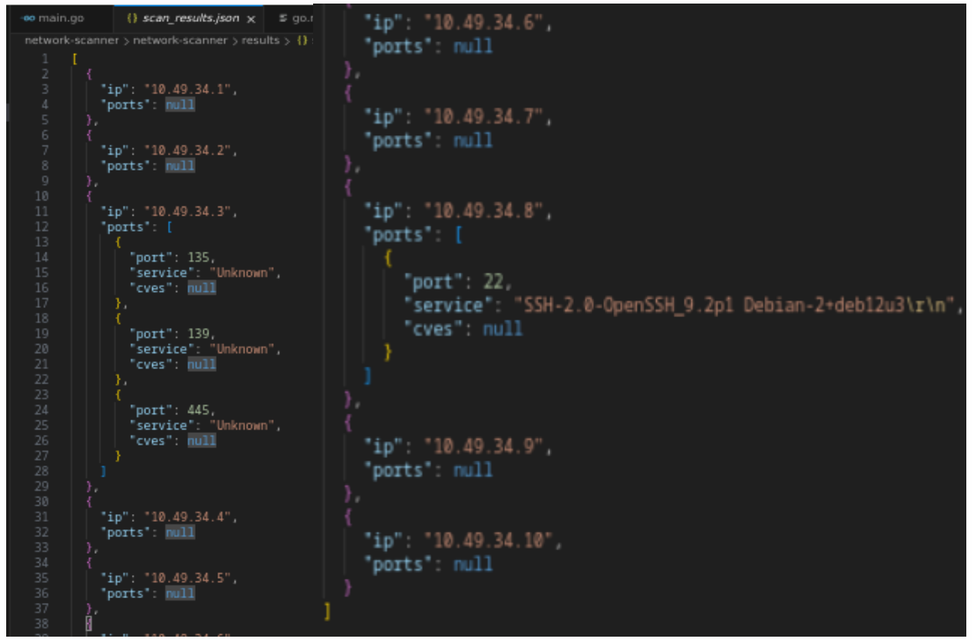
Exécution dans le terminal

```
$ go run main.go

Starting Network Scanner...

Scanning 192.168.1.1...

Scanning 192.168.1.2...

Scanning 192.168.1.3...

Scan complete.
```

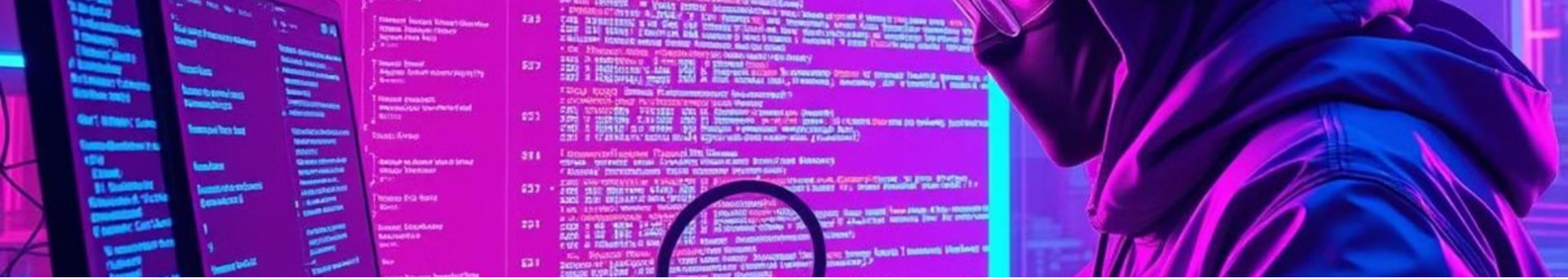
Résultat en JSON (scan_results.json)


```
[
 {
  "ip": "192.168.1.1",
  "ports": [
   {
    "port": 22,
    "service": "SSH-2.0-OpenSSH_7.6p1 Debian-4",
    "cves": [
     {
      "id": "CVE-2018-15473",
      "description": "OpenSSH 7.6p1 est vulnérable à une attaque de validation d'identité à distance.",
      "cvss_score": 5.3
     }
    ]
   },
   {
    "port": 80,
    "service": "HTTP/1.1 200 OK\nServer: Apache/2.4.29 (Debian)",
    "cves": [
     {
      "id": "CVE-2017-7679",
      "description": "Apache HTTP Server 2.4.29 est vulnérable à une fuite de mémoire dans mod_mime.",
      "cvss_score": 7.5
     }
    ]
   }
  ]
 }
```


Tom - Ilies



# Problèmes Rencontrés & Solutions

| Problème | Solution |
| --- | --- |
| Go ne trouve pas les modules (package not in std) | Vérifier "go.mod", exécuter "go mod tidy", vérifier les chemins d'import |
| Ports non détectés sur certaines IP | Ajuster le "timeout" à "500ms" |
| Difficulté à récupérer les CVEs | Utilisation de l'API NVD avec "net/http" et parsing JSON |
| Export HTML mal formaté | Correction avec "html/template" pour éviter l'injection XSS |
| Performances lentes sur de grandes plages d'IP | Optimisation avec "goroutines" pour exécuter les scans en parallèle |


# Conclusion & Perspectives d'Amélioration

1 Ce projet a permis de développer un outil efficace de scan réseau en Go.

2 Il offre une bonne base pour une future extension avec plus de fonctionnalités (ex : détection d'OS, support UDP, interface graphique).

3 Pour l'avenir, une meilleure gestion des logs seraient des pistes intéressantes.