INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

4ème année INSA Centre Val de Loire
Spécialité Sécurité et Technologies Informatiques
Rapport du projet de sécurité

Auteur
Antoine VIRGOS

VulnDrake

Tuteur Académique (INSA)
Ahmad ADELL

Année universitaire 2023-2024

Sommaire

Introduction

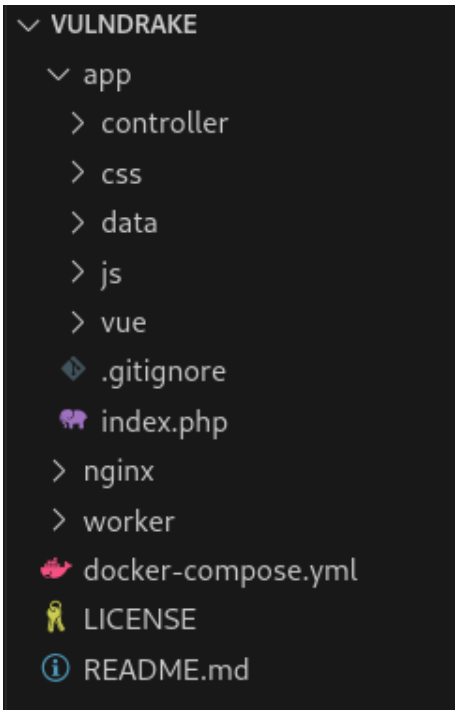
VulnDrake est une application web et open-source (https://github.com/avirgos/vulndrake) permettant de scanner des serveurs afin d'obtenir des informations sur ces derniers via leurs ports exposés. À la suite du scan, un rapport au format PDF est généré et l'utilisateur peut le consulter directement ou bien dans la section "Historique".

La solution logicielle est entièrement *dockerisée*, cela signifie que les services utilisés sont dans des conteneurs : des environnements isolés au sein d'un même réseau virtuel. Ainsi, par l'exécution d'une simple commande, l'application est déployable facilement.

Enfin, du serveur web au scanner en passant par les technologies web, tout est open-source. Cela signifie que l'utilisateur est libre de modifier le logiciel à sa guise : personnaliser le style de l'application, modifier le nom des rapports générés, etc. Quant à la modification des conteneurs, il est plus délicat de les modifier puisque l'application peut fonctionne plus ou moins bien.

Le but final de VulnDrake est de pouvoir rendre le scan d'appareils en réseau facile pour tous.

Structure



Sur la capture d'écran ci-dessus, on constate 3 répertoires situés au même niveau que le `docker-compose.yml` : `app`, `nginx` et `worker`.

`app` contient toutes les ressources liées à l'application web. Cette dernière fonctionne sous le modèle MVC (Modèle-vue-contrôleur). En effet, on retrouve les répertoires `vue` et `controller` mais pas le répertoire `model`. Nous ne trouvons pas ce dernier puisque nous n'utilisons pas de bases de données pour stocker les rapports de scans générés en PDF : ceux-ci sont générés en local au sein du répertoire `data/reports`.

L'application web a été créée en PHP et utilise de l'HTML5, du CSS3 et du JavaScript. Aucun framework n'a été utilisé pour réaliser l'application. L'application web dispose de 2 pages : `index.php` qui est la page d'accueil de l'application et `scanner.php` qui notifie l'utilisateur sur la fin du scan. Enfin, l'application web est dans le conteneur `vulndrake-web` et ce dernier communique avec le conteneur `vulndrake-worker` dont l'on obtiendra davantage de détails plus tard dans ce rapport.

`nginx` contient la configuration du serveur web éponyme ainsi que le certificat SSL/TLS auto-signé généré via OpenSSL. Le répertoire dispose également d'un script Bash `renew-cert.sh` qui permet de renouveler ce certificat.

`worker` contient toutes les ressources relatives au backend de l'application web. En effet, on retrouve le script Bash `scanner.sh` qui utilise gvm-cli. Ce dernier permet d'interagir avec le protocole GMP (Greenbone Management Protocol) via la socket associé à `gvmd` (Greenbone Vulnerability Manager Daemon). Ainsi, grâce à `gvm-cli` qui va communiquer avec `gvm`, le script `scanner.sh` est en capacité de scanner un serveur en lui passant une adresse IP et une liste de ports à utiliser. Une configuration prédéfinie est utilisée et le rapport de scan est obtenu au format PDF.

De plus, `worker` dispose d'un script Python nommé `export-pdf-report.gmp.py` permettant de convertir un scan en un rapport PDF. Je l'ai récupéré sur le dépôt GitHub suivant : https://github.com/greenbone/gvm-tools/blob/main/scripts/export-pdf-report.gmp.py.

Enfin, le répertoire contient le script Bash `check-connection.sh` permettant à l'utilisateur de savoir quand est-ce que VulnDrake est prêt à réaliser un scan. En effet, un certain temps est nécessaire pour que `gvm` soit opérationnel à chaque démarrage de l'application.

Le fichier `docker-compose.yml` comprend de nombreux conteneurs liés exclusivement à l'outil de scan OpenVAS. Les conteneurs suivants ont été utilisés : `vulnerability-tests`, `notus-data`, `scap-data`, `cert-bund-data`, `dfh-cert-data`, `data-objects`, `report-formats`, `gpg-data`, `redis-server`, `pg-gvm`, `gvmd`, `configure-openvas`, `openvas`, `openvasd`, `ospd-openvas` et `gvm-tools`. La description de chacun des conteneurs se trouve ici : https://greenbone.github.io/docs/latest/22.4/container/index.html#description.

J'ai volontairement retiré le conteneur `gsa` (Greenbone Security Assistant), proposé par la documentation officielle, puisqu'il s'agit de l'interface web de l'outil de scan OpenVAS et les conteneurs VulnDrake occupent déjà ce poste.

Fonctionnalités

VulnDrake propose les fonctionnalités suivantes :

- Déclencher un scan d'un serveur à partir d'une adresse IP (obligatoire) et en sélectionnant une liste de ports à utiliser (optionnel)
- Obtenir un rapport de scan au format PDF, via une configuration prédéfinie, sur les ports exposés
- Définition d'une nomenclature unique pour les rapports générés : `VulnDrake-Report-<YYYYMMDDHHmmSS-IDUnique>`
- Visualiser le rapport de scan au format PDF directement sur l'application et sur sa section "Historique"
- La section "Historique" répertorie les rapports du plus récent au plus ancien
- Rapport de scan sauvegardé, en local, dans le répertoire /data/reports
- Avertir en temps réel l'utilisateur dès que l'outil de scan est prêt à être utilisé via l'application
- Empêcher la saisie d'adresses IP incorrectes en affichant un message d'erreur à l'utilisateur
- Renouveler le certificat SSL/TLS auto-signé en exécutant le script Bash correspondant
- Protection du serveur web : XSS, CORS, CSP, *clickjacking*, etc.
- Protection du code et de l'exécution de commandes Bash via PHP : utilisation de `escapeshellarg` et de `escapeshellcmd`

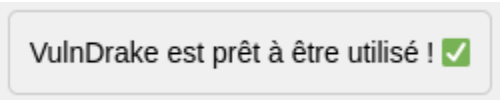
Utiliser VulnDrake

Dans un terminal, commencer par exécuter la commande suivante :

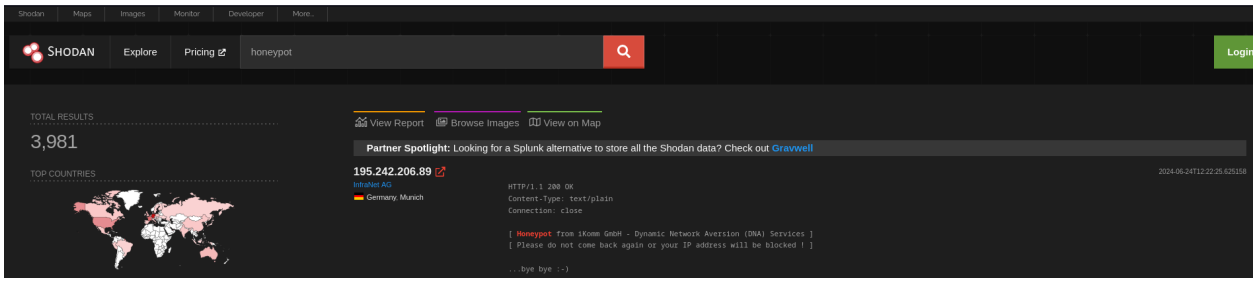
```
$ docker compose up -d
```

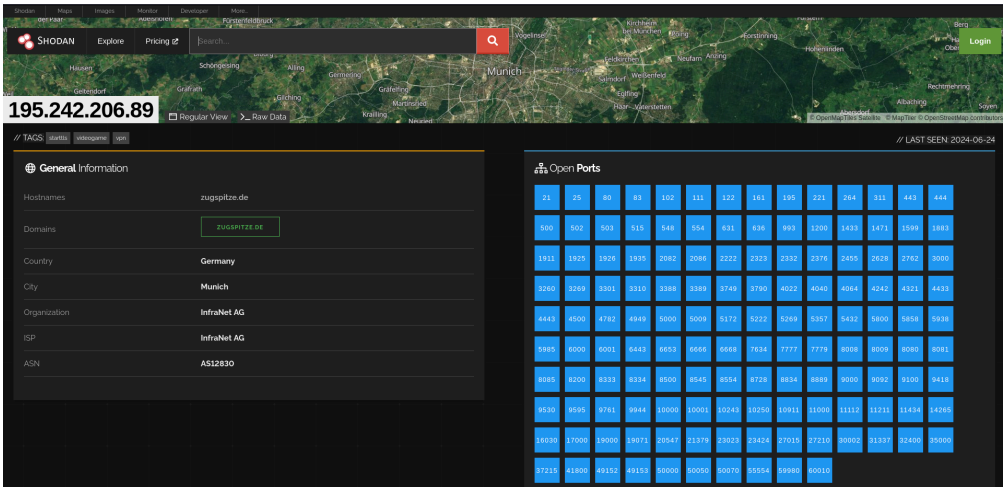
Par la suite, ouvrez un navigateur web et saisissez l'URL suivante : https://localhost

Votre application web est déployée. Veuillez attendre que ce message apparaisse pour pouvoir utiliser VulnDrake :

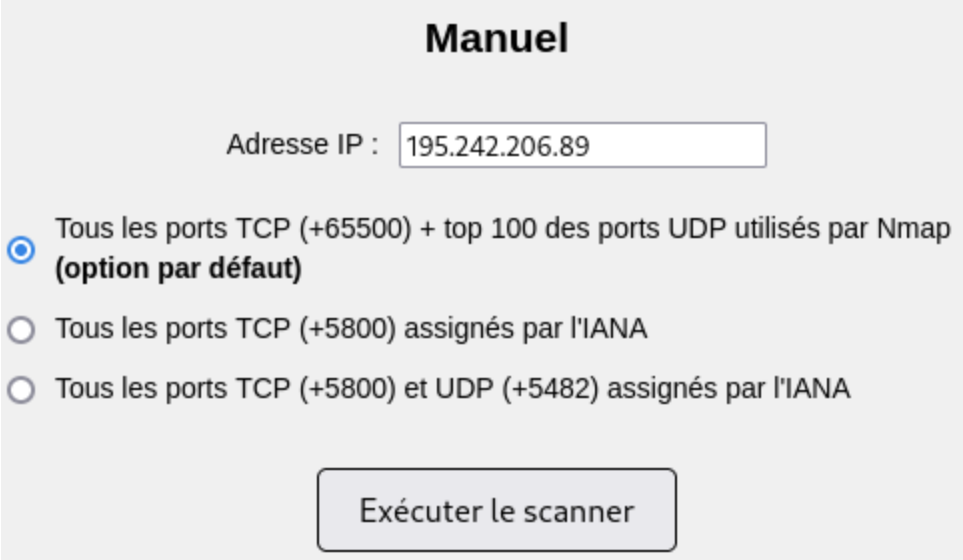


Pour tester VulnDrake, il est recommandé de chercher sur Shodan (https://www.shodan.io) un *honeypot* et de récupérer son adresse IP (ici 195.242.206.89) :

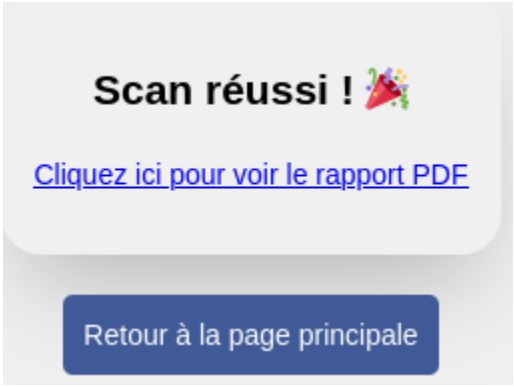




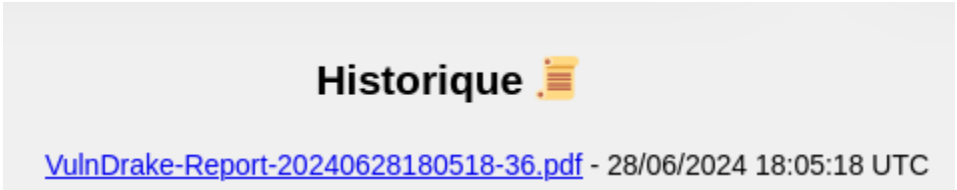
Par la suite, saisissez l'IP, sélectionnez (optionnel) une des listes de ports proposées et appuyez sur le bouton "Exécuter le scanner" :



Patientez quelques minutes (~10 minutes) pour obtenir la fenêtre suivante :



Vous pouvez visualiser le rapport généré en cliquant sur le lien. De plus, si vous retournez à la page principale, vous retrouverez le rapport dans la section “Historique” :



Améliorations

Le projet VulnDrake pourrait être amélioré en ajoutant les fonctionnalités suivantes :

- Créer des configurations supplémentaires pour que les analyses soient davantage approfondies et moins “en surface”
- Développer le module “Automatique” pour pouvoir réaliser des scans automatiquement et permettre l’envoi des rapports par mail
- Pouvoir supprimer des rapports à partir de la section “Historique”

Conclusion

Développer cette application web m'a permis d'approfondir les outils que j'apprécie utiliser dans le cadre de mon travail tels que Docker, Bash et Python. Je me suis beaucoup documenté sur l'outil OpenVAS pour réaliser ce projet et j'ai parcouru de nombreux forums et de nombreuses documentations pour obtenir des solutions. Parfois, je me suis retrouvé sur de fausse-pistes et je devais rétro-pédaler pour pouvoir résoudre certains problèmes liés au scan depuis VulnDrake.

Au final, je suis très fier d'avoir pu créer une solution de sécurité informatique fonctionnelle et facile à utiliser, le tout en utilisant des outils open-source. Partager cette application web sur GitHub est une grande réussite pour moi.