

Jan Walczak, Mikołaj Siedlecki

Politechnika Warszawska, Wydział Elektroniki i Technik Informacyjnych

BSO Projekt Dokumentacja,  
grupa dziekańska: 2 - Cyberbezpieczeństwo

6 marca 2024

## Spis treści

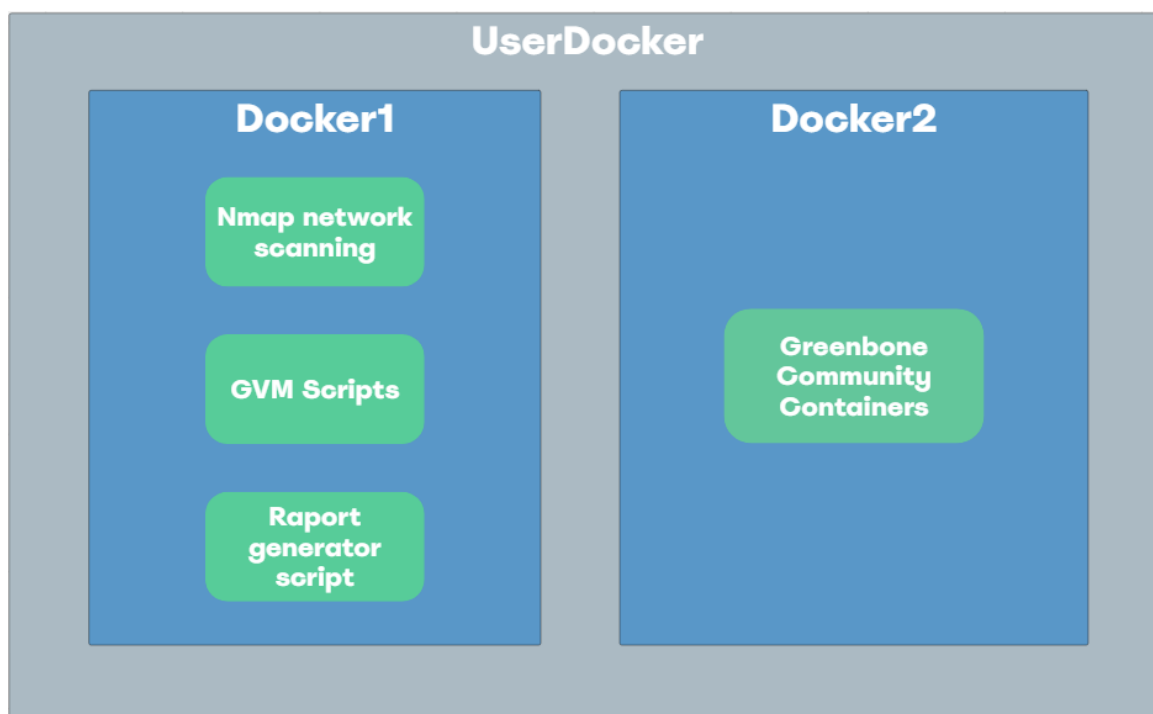
<b>1. Temat Projektu</b>	2
<b>2. Realizacja rozwiązania</b>	2
2.1. Nmap network scanning	2
2.2. GVM Scripts	3
2.3. POC/aktualny stan	3
2.4. Raport generator script	4
2.5. Raportowanie okresowe	4
<b>3. Perspektywa użytkownika</b>	4
<b>Literatura</b>	4

## 1. Temat Projektu

W dzisiejszych czasach coraz więcej firm oraz użytkowników posiada sieci informatyczne, które służą do przetwarzania, przechowywania i udostępniania danych. Z tego powodu bezpieczeństwo sieci jest bardzo ważne, ponieważ ataki na systemy informatyczne są coraz bardziej powszechne. Celem tej pracy jest stworzenie kontenera typu Docker, która umożliwi skanowanie urządzeń sieciowych i wykrywanie potencjalnych zagrożeń w celu zapewnienia bezpieczeństwa sieci. Kontener będzie automatycznie generować raporty bezpieczeństwa, które będą służyć do analizy i identyfikacji problemów w sieci.

Dzięki aplikacji użytkownik będzie w stanie za-

pewnić stałą ochronę swojej sieci i szybko reagować na pojawiające się zagrożenia. Architektura aplikacji będzie oparta na dwóch oddzielnych kontenerach znajdujących się wewnątrz ostatecznego rozwiązania. Pierwszy z kontenerów składa się z trzech głównych modułów: moduł skanowania urządzeń sieciowych, moduł zarządzania i moduł generowania raportów. Aplikacja będzie posiadała prosty i intuicyjny interfejs użytkownika. W drugim kontenerze skorzystamy z pełni wyposażonego skanera podatności Greenbone OpenVAS i dodatkowych narzędzi działających współbieżnie zamkniętych w postaci Greenbone Community Containers.



Rys. 1: Diagram rozwiązania

## 2. Realizacja rozwiązania

### 2.1. Nmap network scanning

Skanowanie sieci użytkownika w celu jej dalszej penetracji to bardzo ważny aspekt naszego projektu. Zamierzamy je przeprowadzić używając narzędzia Nmap. Na początku musimy odkryć adres sieci, w jakiej znajduje się użytkownik naszego oprogramowania, aby tego dokonać, musimy dowiedzieć się, na jakim systemie został uruchomiony nasz projekt.

Zrealizujemy to za pomocą biblioteki "platform". W zależności od rozpoznanego systemu operacyjnego zostaną wywołane komendy w celu zebrania informacji o sieci, w jakiej się znajdujemy, takie jak adres hosta i maska podsieci (w ten sposób dowiemy się o adresie sieci którą będziemy skanować). Przeprowadzając skan naszej sieci, wykorzystamy szybki

sposób wykrywania hostów (host discovery) - flagę "-sn". Ta opcja mówi Nmapowi, aby nie skanował portów po wykryciu hosta i drukował tylko dostępne hosty, które odpowiedziały na sondy wykrywania hostów. Przyspieszamy w ten sposób skanowanie całej naszej sieci, porty, które będziemy skanować, zostaną uwzględnione w następnym module (GVM). Wyko-

## 2.2. GVM Scripts

Trzonem i nieodłącznym elementem naszego rozwiązania jest sam skaner podatności, jak wcześniej opisaliśmy w naszym rozwiązaniu posłużymy się OpenVasem. A dokładniej paczką narzędzi skompresowaną w formie zespołu współpracujących ze sobą kontenerów docker - Greenbone Community Containers. Główny skaner OpenVAS Scanner to w pełni funkcjonalny silnik skanowania, który wykonuje testy podatności (VT) na systemy docelowe. W tym celu korzysta z codziennie aktualizowanych i kompleksowych kanałów: w pełni funkcjonalnego, obszerne- go, komercyjnego kanału Greenbone Enterprise Feed lub bezpłatnego kanału Greenbone Community Feed. Moduł GVM script jest główną częścią naszego rozwiązania, zakłada użycie takich narzędzi jak m.in. gvm-tools dzięki czemu będziemy w stanie wykonywać skrypty odwołujące się do modułu openvas'a celem skanowania sieci. Pierwszym krokiem w celu przeprowadzenia skanowania jest stworzenie "tar-

nywane przez nas skanowanie Nmap w celu odkrycia hostów w danej sieci :

**nmap -sn -oN output.txt <IPaddress>/<mask>**


Dzięki temu do pliku "output.txt" zostaną zapisane tylko te adresy, które Nmap uznał za aktywne w naszej sieci i tylko je prześlemy do następnego modułu celem dogłębnego skanowania.

get'u" który jest celem naszego skanowania (lista aktywnych hostów przekazana przez moduł Nmap network scanning). Następnie należy utworzyć "task", zawierający szczegóły dotyczące naszego skanowania (zakres portów skanowania, terminarz cyklicznego wykonywania skanowania itp.). Porty jakie chcielibyśmy skanować to: 20 i 21 (FTP), 23 (TELNET), 22 (SSH), 25 (SMTP), 53 (DNS), 110 (POP3), 111, 135 (rpcbind oraz msrpc) 137 i 139 oraz 445 (SMB), 143 (IMAP) 80, 8080, 443, 8443 (HTTP i HTTPS), 993 (IMAPs), 995 (POP3s) 1433, 1434, 3306 (Bazy Danych), 3389 (Remote Desktop). Są to głównie porty z "Top 20" wykorzystywanego przez między innymi Nmapa.

Po udanym stworzeniu należy go uruchomić i poczekać na wykonanie. Po zakończeniu skanu naszej sieci, wygenerowany zostanie raport który zostanie przekazany do kolejnego modułu celem przesłania go na maila naszego użytkownika.

## 2.3. POC/aktualny stan

Aktualny POC naszego rozwiązania:



Rys. 2: POC http 200 OK

Aby umożliwić korzystanie z protokołu GMP dostarczonego przez gvmd z hosta dokera, dla katalogu /run/gvmd należy użyć montowania powiązania. Aby udostępnić gniazdo gvmd Unix, utworzyliśmy lokalny katalog i dostosowaliśmy jego uprawnienia(777). Następnie można było zaktualizować wpis w konfiguracji pliku docker-compose .yaml ze wska-

zaniem na nasz lokalnie stworzony folder. Po resecie danego komponentu udało nam się skorzystać z narzędzie gvm-cli ze wskazaniem na nasz lokalnie mapowany socket i wywołać zapytanie sprawdzające wersje oprogramowania. Zwrócony response napawa nas optymizmem i dodaje wiary w dalsze sukcesy w realizacji projektu.

## 2.4. Raport generator script

Końcowym etapem działania naszego modułu skanowania sieci jest wysłanie gotowego raportu bezpieczeństwa do użytkownika. Do wysłania maila stworzymy skrypt pythonowy korzystający z bibliotek: `smtplib`, `mimetypes`. Na potrzebę wysłania maila

użyjemy odpowiednio skonfigurowanego gmail'owego servera SMTP. Użytkownik otrzyma na wcześniej podanego maila wiadomość z załączonym plikiem raportu w formacie PDF.

## 2.5. Raportowanie okresowe

Należy być świadomym faktu, że stan aktywnych hostów w naszej sieci, konfiguracji poszczególnych komponentów nigdy nie jest stały. Dodatkowo należy wziąć pod uwagę wciąż zmieniające się aktywne podatności jak i "łatki" które producenci oprogramowania starają się wypuszczać, aby owe zagrożenia minimalizować. Skanowania naszej sieci należy przeprowadzać w sposób okresowy, dlatego cały ciąg wywołań naszych modułów będzie przeprowadzany

co tydzień w porach nocnych (1:00-3:00) na długość przeprowadzania naszego skanowania ma oczywiście wpływ wielkość samej sieci. Do osiągnięcia efektu cyklicznego uruchamiania naszego systemu zamierzamy skorzystać z narzędzia `crontab`. Dodatkowo zamierzamy przypominać użytkownikowi o planowanym skanowaniu za pomocą dodatkowych zadań `crontab` które są zaplanowane na wcześniejszą porę tego dnia.

## 3. Perspektywa użytkownika

Jak już wspominaliśmy, całość rozwiązania zostanie zamknięta w kontener i umieszczony na koncie przedmiotu w serwisie `dockerhub`. Jedynym wejściem, jakiego oczekujemy od użytkownika naszego systemu, jest podanie adresu email, na który zostanie wysłany wygenerowany raport końcowy. Cała reszta

zostanie wykonana w sposób zautomatyzowany (wykrycie skanowanej sieci, hostów w tej sieci itd.). Będzie więc to idealna perspektywa dla niezaawansowanego użytkownika, chcącego jedynie zabezpieczyć swoją sieć.

## Literatura

- [1] Calderón Pale, (2021). *Nmap Network Exploration and Security Auditing Cookbook: Network discovery and security scanning at your fingertips*. Packt Publishing Ltd, 54-83
- [2] Calderón Pale, (2015). *Mastering the Nmap Scripting Engine*. Packt Publishing Ltd, 18, 20-24, 30, 37-55, 69-87
- [3] Shaw, D. (2015). *Nmap Essentials*. Packt Publishing Ltd.
- [4] Ashish Kumar Luhach, Dharm Singh, Pao-Ann Hsiung, Kamarul Bin Ghazali Hawari, Pawan Lingras, Pradeep Kumar Singh, (2018). *Advanced Informatics for Computing Research*. Springer.
- [5] Aksu, M. U., Altuncu, E., & Bicakci, K. (2015). A First Look at the Usability of OpenVAS Vulnerability Scanner. *International Journal of Computer Science and Network Security*, 15(5), 48-53.
- [6] Greenbone Networks. (n.d.). Greenbone Container Documentation. Retrieved April 19, 2023, from <https://greenbone.github.io/docs/latest/22.4/container/index.html>.
- [7] Greenbone Networks. (n.d.). Python-GVM Documentation. Retrieved April 19, 2023, from <https://python-gvm.readthedocs.io/en/latest/>.
- [8] Greenbone Networks GmbH. (2022). Greenbone Management Protocol (GMP) API: Version 22.04. Retrieved April 19, 2023, from <https://docs.greenbone.net/API/GMP/gmp-22.04.html>.