SARA AMATO

CYBERSECURITY
SPECIALIST

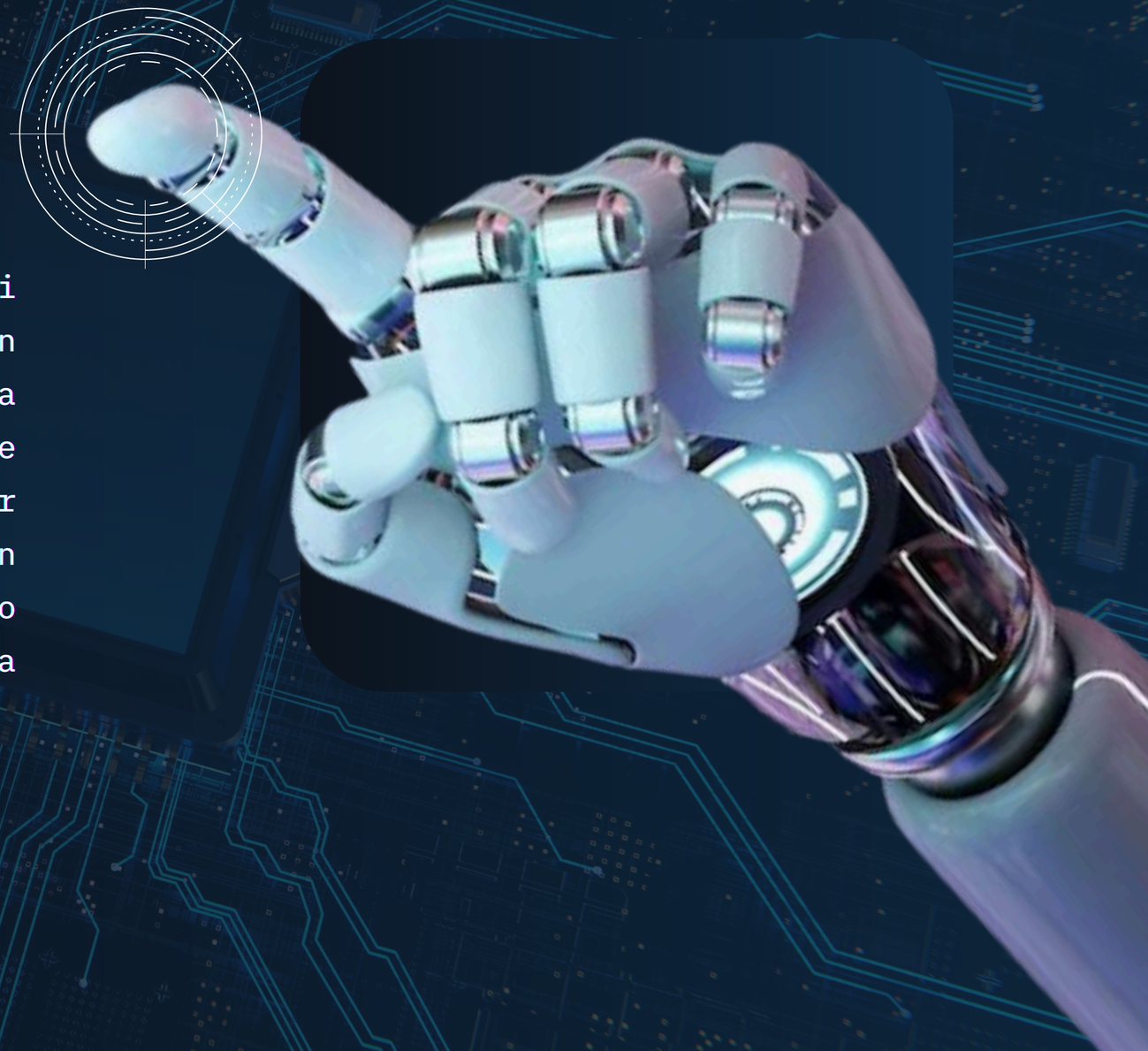
Sfruttamento della vulnerabilità Java RMI sulla porta 1099

S7 L5

15/11/2024

SARA AMATO

Introduzione



Immaginiamo di trovarci in un laboratorio di sicurezza informatica, pronti a simulare un attacco contro un sistema vulnerabile. La nostra missione è dimostrare come un attaccante potrebbe sfruttare una debolezza del servizio Java RMI per ottenere il controllo remoto di una macchina. In questo documento, vi guiderò attraverso ogni passo che ho seguito, spiegando sia la parte tecnica sia il significato di ciò che abbiamo realizzato.

SARA AMATO

Cos'è Java RMI e perché è vulnerabile

Java RMI, acronimo di Remote Method Invocation, è uno strumento che consente a un programma di eseguire funzioni su un altro computer come se fossero locali. È incredibilmente utile per sviluppatori, ma diventa un problema quando viene configurato male. In alcune versioni di Java RMI, è possibile inviare codice arbitrario alla macchina remota senza alcun controllo. Questo vuol dire che un attaccante può "insegnare" alla macchina vittima a fare qualsiasi cosa: dal rubare dati sensibili al diventare parte di una botnet.

Pensiamo a questa vulnerabilità come a una porta lasciata socchiusa: basta sapere come spingerla per entrare.

SARA AMATO

Preparare l'ambiente: le nostre due macchine virtuali

Prima di tutto, ho configurato un ambiente di test con due macchine virtuali:

- La macchina attaccante: un sistema Kali Linux, che è un po' il coltellino svizzero degli esperti di sicurezza. Ha un IP statico configurato su 192.168.11.111.
- La macchina vittima: Metasploitable 2, un sistema deliberatamente insicuro. Questa macchina si trovava sulla stessa rete interna con IP 192.168.11.112.

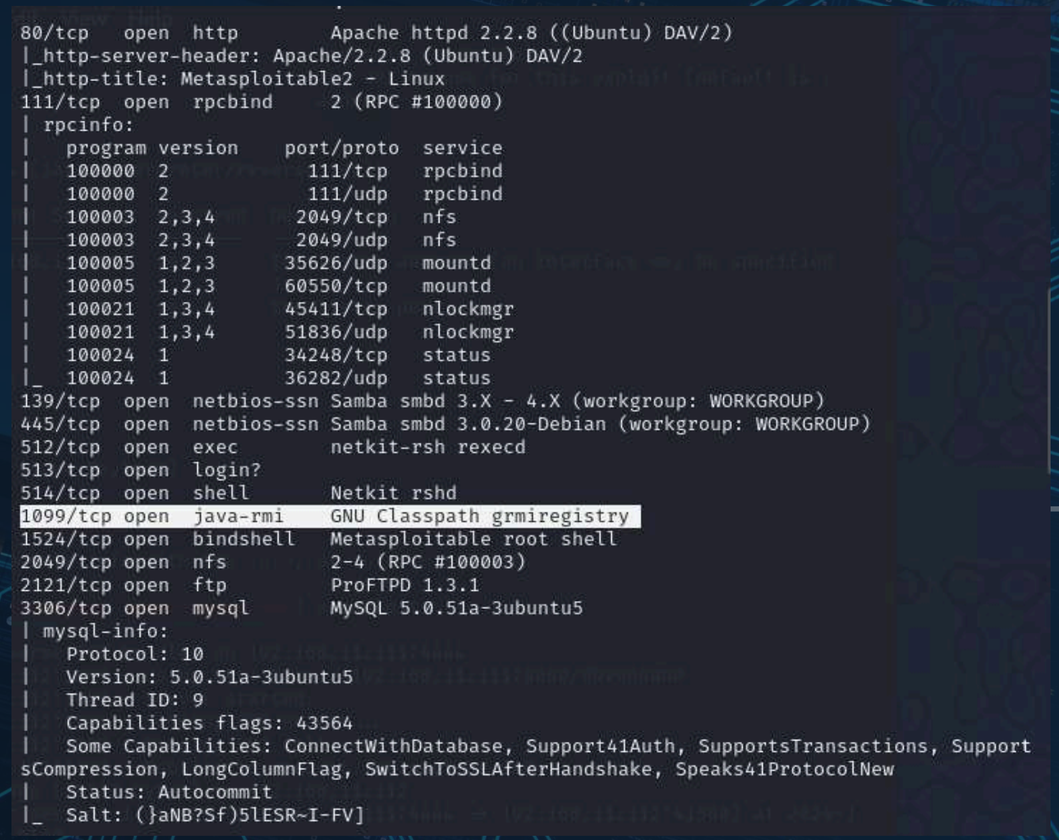
Entrambe le macchine sono collegate tramite una rete interna creata con VirtualBox. Questo significa che possono comunicare direttamente tra loro senza che altri dispositivi, come il mio computer fisico, interferiscano. Questo tipo di rete è perfetto per simulare un attacco senza rischiare di colpire altri dispositivi.

SARA AMATO

Scoprire le vulnerabilità: la scansione preliminare

Il primo passo di qualsiasi attacco è la raccolta di informazioni. Usando Nmap, un tool per la scansione delle reti, ho eseguito un controllo completo della macchina vittima per individuare quali porte fossero aperte e quali servizi vi girassero sopra.

Questo comando `nmap -A -T4` dice a Nmap di controllare tutte le porte della macchina specificata. Tra i vari risultati, ho scoperto che la porta 1099 era aperta, indicando che il servizio Java RMI era attivo.



```
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp  open  rpcbind     2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100003  2,3,4       2049/tcp   nfs
|   100003  2,3,4       2049/udp   nfs
|   100005  1,2,3      35626/udp   mountd
|   100005  1,2,3      60550/tcp   mountd
|   100021  1,3,4      45411/tcp   nlockmgr
|   100021  1,3,4      51836/udp   nlockmgr
|   100024  1          34248/tcp   status
|_  100024  1          36282/udp   status
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.0.20-Debian (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
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
| mysql-info:
|   Protocol: 10
|   Version: 5.0.51a-3ubuntu5
|   Thread ID: 9
|   Capabilities flags: 43564
|   Some Capabilities: ConnectWithDatabase, Support41Auth, SupportsTransactions, Support
sCompression, LongColumnFlag, SwitchToSSLAfterHandshake, Speaks41ProtocolNew
|   Status: Autocommit
|_  Salt: (}aNB?Sf)5lESR~I-FV]
```

SARA AMATO

Cosa significa trovare un servizio vulnerabile?

Quando identifichiamo una porta aperta e un servizio attivo, la prossima domanda è: "Quanto è sicuro questo servizio?". Nel caso di Java RMI, sappiamo che alcune versioni sono vulnerabili a un attacco noto. A questo punto, ho deciso di usare Metasploit, una piattaforma che automatizza il processo di sfruttamento delle vulnerabilità.

SARA AMATO

Usare Metasploit per sfruttare la vulnerabilità

Dentro Metasploit, ho cercato un exploit adatto al servizio Java RMI. Il comando:

```
search rmi
```

mi ha mostrato un elenco di moduli disponibili. Ho scelto exploit/multi/misc/java_rmi_server, specificamente progettato per sfruttare questa vulnerabilità.

Prima di lanciare l’attacco, ho configurato alcuni parametri importanti:

- RHOST: L'indirizzo IP della macchina vittima (192.168.11.112).
- RPORT: La porta su cui gira il servizio vulnerabile (1099).
- LHOST: L'indirizzo IP della macchina attaccante (192.168.11.111).

Queste impostazioni dicono a Metasploit quale macchina attaccare, su quale porta, e dove inviare il controllo una volta che l'attacco ha successo.

SARA AMATO

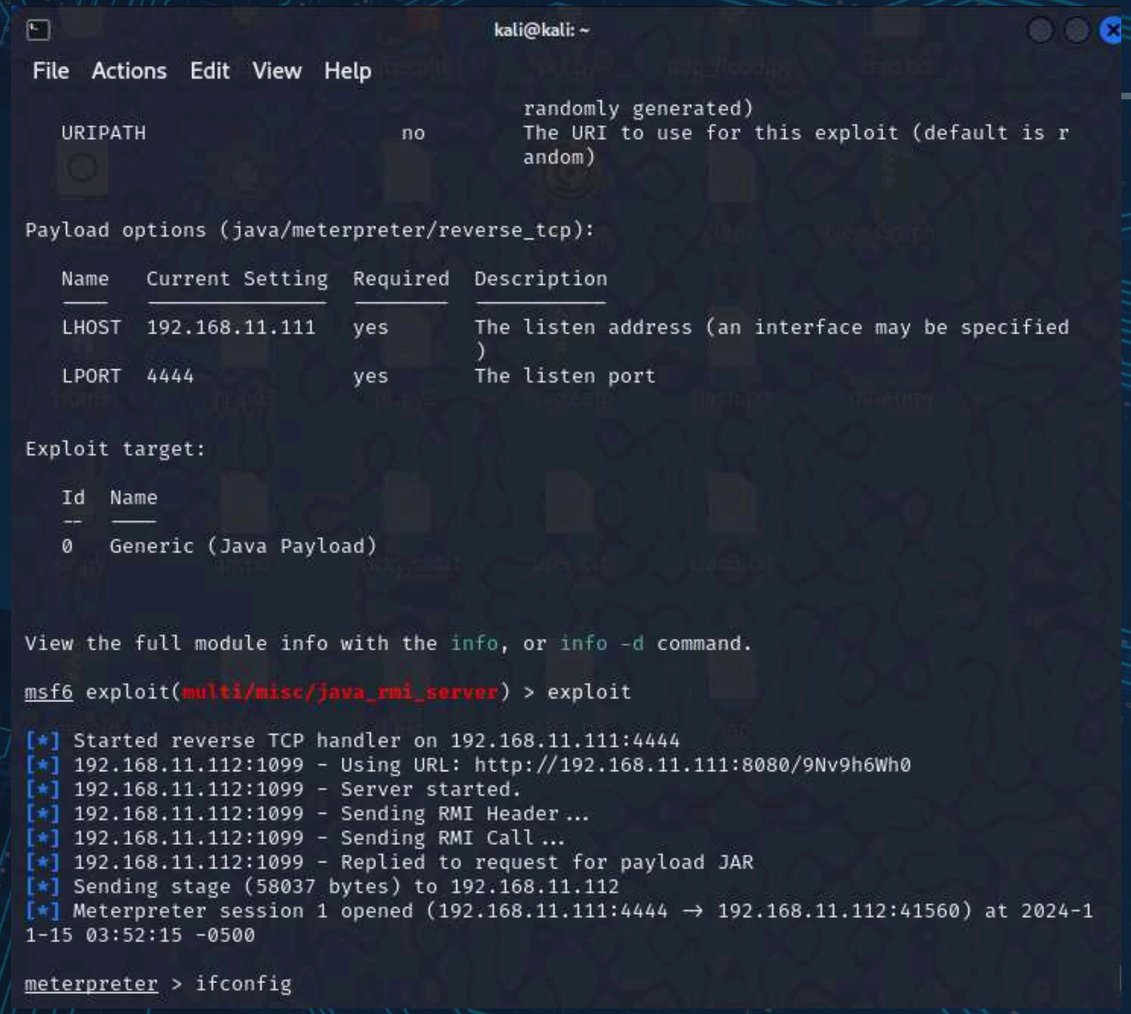
Usare Metasploit per sfruttare la vulnerabilità

Quando ho eseguito il comando exploit, Metasploit ha inviato un payload alla macchina vittima. Un payload è essenzialmente un piccolo programma, creato appositamente per sfruttare una vulnerabilità nel sistema target. In questo caso, il payload è stato progettato per sfruttare la vulnerabilità nel servizio Java RMI che stava girando sulla macchina Metasploitable.
Il payload inviato non è altro che un codice che, una volta eseguito sul sistema vulnerabile, permette all'attaccante di ottenere un controllo completo o parziale sulla macchina. A seconda del tipo di payload, questo può svolgere azioni come l'esecuzione di comandi, l'acquisizione di dati sensibili, o l'instaurazione di una connessione remota. Nel nostro caso, il payload ha aperto una sessione Meterpreter. Meterpreter è un potente payload di Metasploit che offre un'interfaccia interattiva con la macchina compromessa, permettendo all'attaccante di eseguire comandi e raccogliere informazioni dal sistema target. Si tratta di una shell avanzata che non solo permette l'esecuzione di comandi, ma offre anche una serie di funzionalità avanzate per il controllo remoto del sistema, come la gestione dei file, l'acquisizione di screenshot, e la gestione della rete.

SARA AMATO

Usare Metasploit per sfruttare la vulnerabilità

Quando il payload viene eseguito con successo, Meterpreter stabilisce una connessione remota tra la macchina compromessa e l'attaccante, creando una sorta di "finestra" attraverso cui è possibile interagire con il sistema remoto, come se fossimo fisicamente davanti alla macchina. A questo punto, anche se la macchina compromessa è geograficamente lontana, siamo in grado di agire su di essa, inviare comandi, leggere e scrivere file, interagire con il sistema operativo e raccogliere informazioni. Questo rende Meterpreter uno degli strumenti più potenti e flessibili nel contesto di un attacco informatico.



SARA AMATO

Usare Metasploit per sfruttare la vulnerabilità

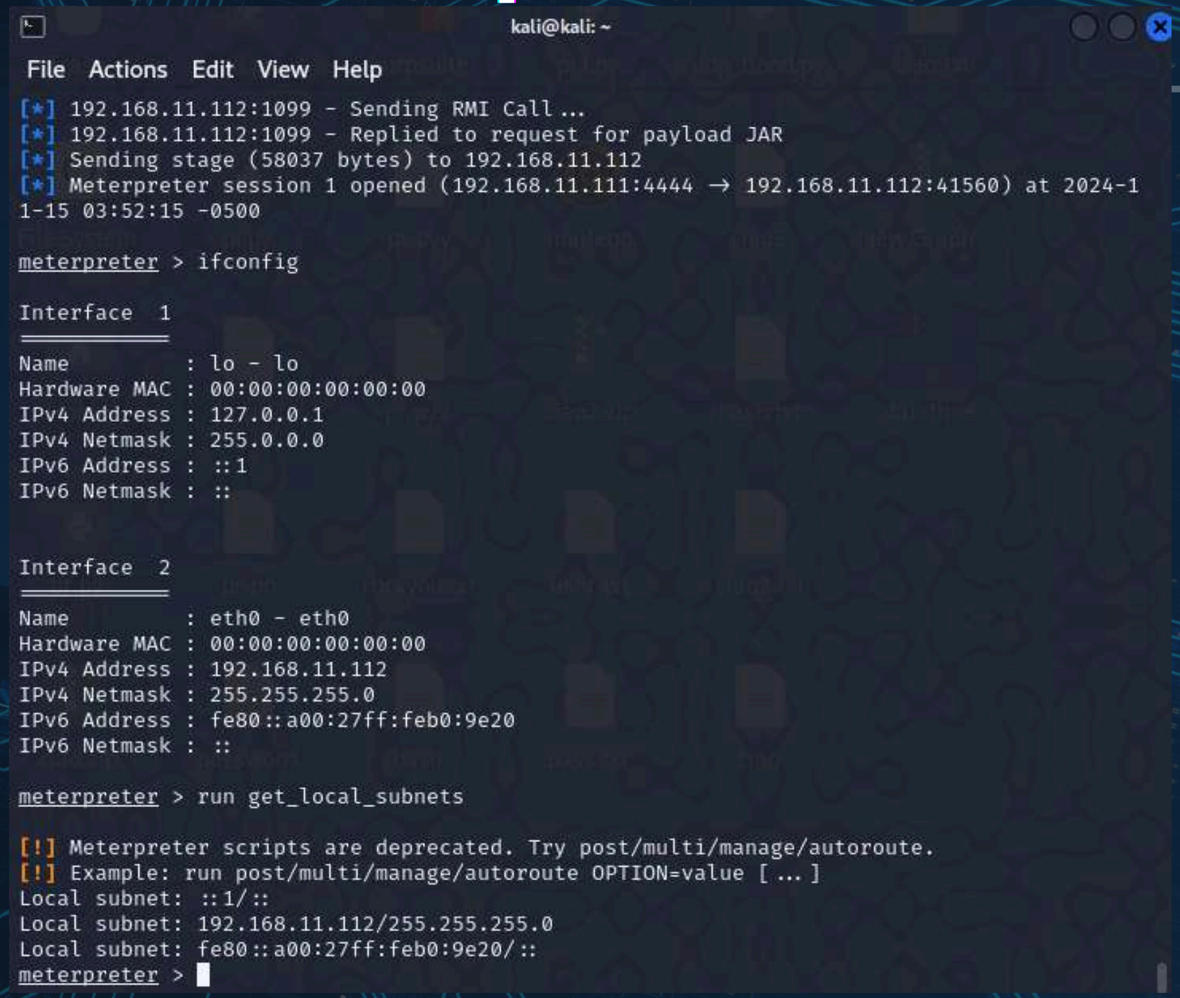
La sessione Meterpreter, infatti, è molto più di una semplice shell: essa offre la possibilità di estendere l'attacco, eseguire tecniche di pivoting per spostarsi nella rete, eseguire attività di post-exploitation come il furto di credenziali o l'accesso a credenziali salvate, e anche di fare screen capture e keylogging, tutte funzionalità che sono essenziali per il controllo e l'esfiltrazione di dati.

SARA AMATO

Esplorare la macchina compromessa

Una volta ottenuta la sessione Meterpreter sulla macchina vittima, il passo successivo è stato quello di esplorare la configurazione di rete della macchina compromessa per raccogliere informazioni utili. Inizialmente, ho utilizzato il comando ifconfig per esaminare le interfacce di rete attive sulla macchina vittima. Questa informazione è stata fondamentale per confermare che la macchina compromessa si trova nella rete corretta (con lo stesso subnet 192.168.11.112/24 di Kali), e che l'interfaccia di rete utilizzata è attiva e correttamente configurata.

Successivamente, ho eseguito il comando run get_local_subnets per identificare altre reti locali a cui la macchina compromessa potrebbe essere collegata. Questo comando è stato utilizzato per mappare eventuali subnet aggiuntive, ma, poiché la macchina era configurata principalmente su una sola rete, i risultati sono stati i seguenti:

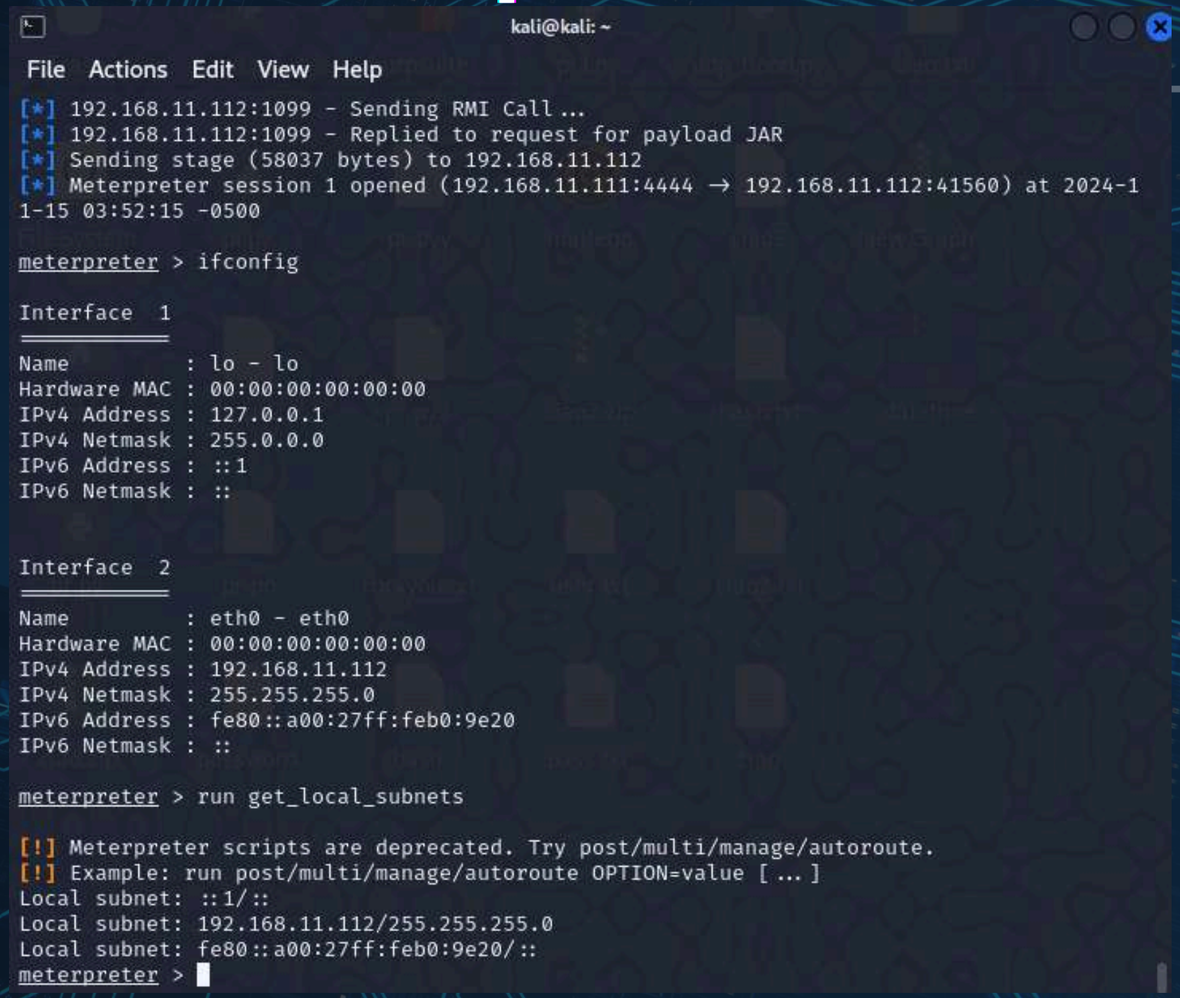


SARA AMATO

Esplorare la macchina compromessa

I risultati mostrano che la macchina compromessa è configurata principalmente sulla rete locale 192.168.11.112/255.255.255.0 (una subnet di classe C).
Questa mappatura delle subnet è stata utile per comprendere la configurazione della macchina e identificare eventuali percorsi di rete che potessero essere utilizzati per espandere l'accesso alla rete.

Dopo aver acquisito queste informazioni, la macchina compromessa è stata completamente esplorata in termini di configurazione di rete. Inoltre, il fatto che la macchina fosse già completamente compromessa (e che fossi loggato come root) ha reso l'analisi e la raccolta delle informazioni molto più dirette, senza necessità di escalation dei privilegi.



SARA AMATO

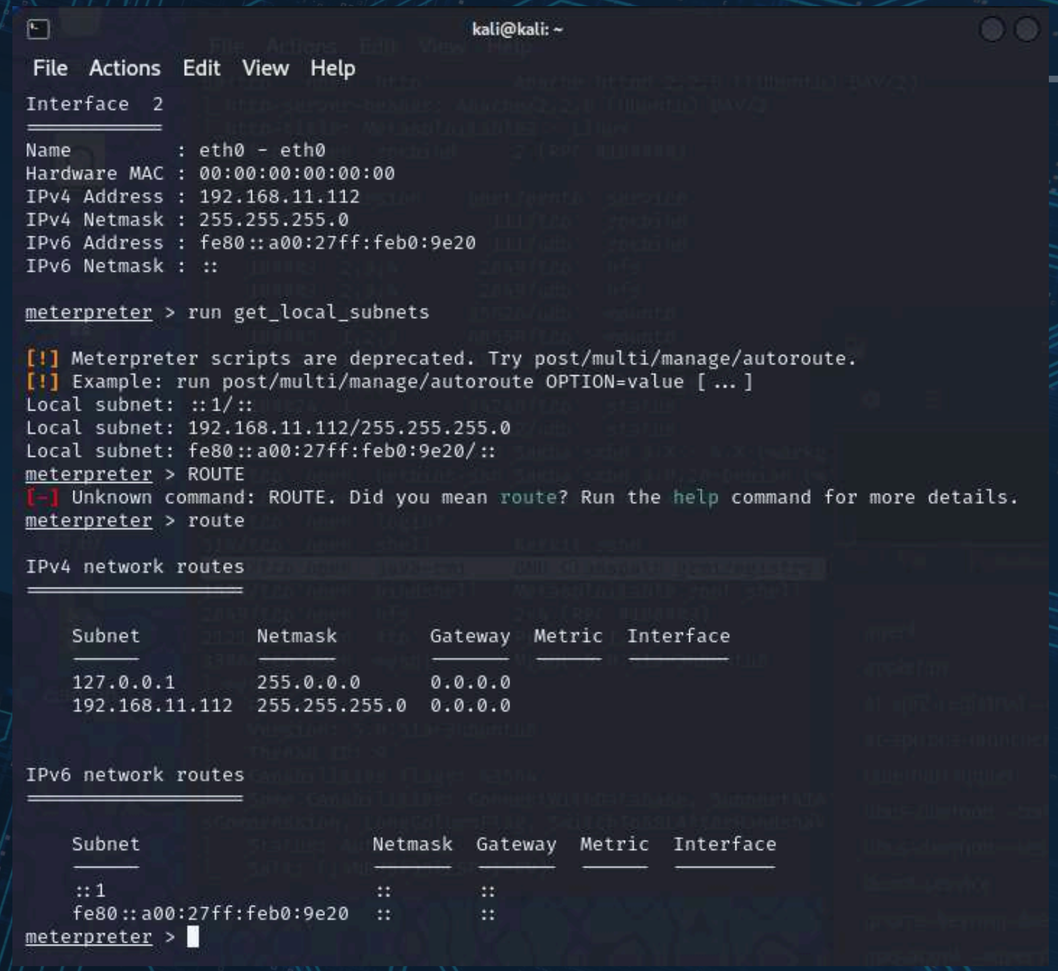
Esplorare la macchina compromessa

Con il comando:

route

Troviamo la tabella di routing, una sorta di "mappa" che il tuo computer utilizza per decidere come inviare i pacchetti di dati verso destinazioni diverse. Ogni riga della tabella rappresenta una regola di routing e contiene informazioni come:

- Destinazione: L'indirizzo di rete o l'host a cui sono destinati i pacchetti.
- Gateway: L'indirizzo IP del router utilizzato per raggiungere la destinazione.
- Maschera di sottorete: Definisce la parte di rete dell'indirizzo IP di destinazione.
- Interfaccia: L'interfaccia di rete utilizzata per inviare i pacchetti.
- Metrica: Un valore numerico che indica il costo associato a una particolare rotta.



SARA AMATO

Perché non è necessario fare escalation dei privilegi in questo caso?

Nel nostro caso, il motivo per cui non è stata necessaria un'escalation dei privilegi è che l'exploit di Java RMI che stiamo utilizzando in Metasploit è stato progettato per ottenere l'accesso come root fin dall'inizio. Questo accade perché:

1. Vulnerabilità di Java RMI: La vulnerabilità in questione è legata a un difetto nel servizio Java RMI su Metasploitable 2. Questo servizio permette a un attaccante di inviare richieste remote malformate che possono portare all'esecuzione di codice arbitrario sulla macchina target. In altre parole, il servizio Java RMI consente di "iniettare" un payload (un programma dannoso) che viene eseguito con privilegi elevati (root) sul sistema vulnerabile.
2. Tipo di exploit utilizzato: L'exploit che stiamo utilizzando (multi/misc/java_rmi_server) è un exploit di "remote code execution" che consente di eseguire comandi arbitrari sul sistema target. Una volta che il payload viene eseguito, otteniamo direttamente l'accesso a una sessione Meterpreter con privilegi di amministratore (root), senza dover compiere l'escalation dei privilegi.
3. Macchina vulnerabile (Metasploitable 2): Metasploitable 2 è una macchina virtuale progettata appositamente per essere vulnerabile a una serie di attacchi. In questo caso, il servizio Java RMI è configurato in modo tale che l'exploit possa sfruttare la vulnerabilità e ottenere l'accesso come root. Nella realtà, un attacco di questo tipo sarebbe un grave rischio per la sicurezza di un sistema, ma Metasploitable è stato progettato per essere facilmente compromesso per scopi di apprendimento.

SARA AMATO

Gestione degli errori: il parametro HTTPDELAY

Durante il test, si sarebbe potuto verificare un errore. Quando Metasploit tenta di stabilire una connessione con la macchina vittima, può fallire a causa di problemi di latenza o congestione. Per risolvere questo problema, avremmo dovuto modificare il parametro HTTPDELAY, aumentando il valore a 20. Anche se nel mio caso l'errore non si è verificato, ho voluto testare questa configurazione per documentare l'efficacia della soluzione.

Il comando per configurare il parametro è:

```
set HTTPDELAY 20
```

Questo fa sì che Metasploit rallenti leggermente le richieste, migliorando la stabilità della connessione.

SARA AMATO

Conclusioni

In conclusione, abbiamo sfruttato con successo la vulnerabilità Java RMI sulla macchina Metasploitable per ottenere l'accesso remoto tramite Meterpreter. Abbiamo raccolto le evidenze richieste, tra cui la configurazione di rete e la tabella di routing della macchina compromessa. Inoltre, abbiamo appreso le basi dell'utilizzo di Metasploit per sfruttare vulnerabilità di servizi e le best practices nel campo del penetration testing.

Il progetto ha messo in luce l'importanza di una corretta configurazione di sicurezza in ambiente di produzione, nonché la necessità di monitorare e aggiornare regolarmente i sistemi per evitare il rischio di attacchi remoti.

SARA AMATO

Approfondimenti

Questo esercizio non solo ci insegna a sfruttare una vulnerabilità ben definita, ma sottolinea anche il funzionamento di Metasploit come framework di exploit. Oltre alla parte tecnica, il progetto ha dato anche una comprensione più ampia delle dinamiche di attacco, dove strumenti come Nmap, Metasploit e Meterpreter sono essenziali per i penetration tester.

In un contesto aziendale, la scoperta di una vulnerabilità come quella di Java RMI evidenzia la necessità di monitorare costantemente i sistemi e di applicare patch di sicurezza tempestive. Se tale vulnerabilità fosse rimasta attiva, un attaccante avrebbe avuto libero accesso alla macchina, con la possibilità di eseguire codice arbitrario e compromettere l'intera rete.

SARA AMATO

CYBERSECURITY
SPECIALIST

GRAZIE

PER LA TUA ATTENZIONE