BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Penelitian yang terkait dilakukan berdasarkan hasil penelitian sebagai analisis dan pembanding. Topik penelitian sebagai pembanding meliputi *malware*, *evil-droid*, *exploit*, *backdoor*, dan *reverse engineering*. Berikut penelitian berkaitan dengan topik dilakukan oleh peneliti:

1. **I Putu Agus Eka Pratama, Rasendriya Revo Daniswara pada tahun 2020 dengan judul "Pengujian dan Analisa *Reverse Engineering* Pada *Platform* Android (Studi Kasus: Tebak_Gambar.apk)".**

   Penelitian ini melakukan analisis menggunakan APKTool, JD GUI dan Dex2jar yang bertujuan. Penelitian ini menggunakan metode Reverse Engineering dilakukan pada aplikasi Tebak_Gambar.apk yang dengan ApkTool. Proses *reverse engineering* yang dilakukan yaitu dengan mengekstrak *file* .jar menggunakan Dex2jar lalu mengkompilasi menggunakan JD Gui untuk melihat isi dalam *file* jar tersebut. Setelah mengetahui isi dari *file* .jar. melakukan dekompilasi menggunakan ApkTool untuk menampilkan *file* .xml yang terdapat pada aplikasi Tebak_Gambar.apk. Hasil yang didapat yaitu untuk mempelajari algoritma dari aplikasi android, dan mengetahui isi dari beberapa *file* seperti *sources, res,* apktool.yml dan dilakukannya analisa perbandingan yang telah diperoleh dari aplikasi Tebak_Gambar.apk [8].

2. **M. Hazri pada tahun 2020 dengan judul "Analisis *Malware PlasmaRAT* dengan Metode *Reverse Engineering*".**

   Penelitian ini melakukan menganalisis dan mengidentifikasi *malware* menggunakan metode *reverse engineering*. Analisis proses terjadi adanya *malware Plasma RAT* mempunyai program aktif saat dijalankan pada sistem. *Plasma RAT* menggunakan *anti-reverse*

engineering. Proses *reverse engineering* tidak berhasil, maka dari itu peneliti telah berupaya mengatasi hambatan dan menyatakan informasi penting untuk mengetahui proses kerja dan efek *malware* ini[9]**.**

3. **Bagus Aji Saputro, Lisan Iqbal Alfitra dan Raykhan Bima Oktaviaji pada tahun 2020 dengan judul "Analisis *Malware* Android Menggunakan Metode *Reverse Engineering*".**

   Analisis *Malware* diterapkan dalam aplikasi code4hk.apk dan qq.apk. yang merupakan analisis statis menggunakan metode *reverse engineering*. Penelitian ini bertujuan untuk membuat pengguna sadar terhadap dampak aplikasi yang sudah diinfeksi oleh *malware*. Hasil yang didapat dari penelitian ini yaitu code4hk.apk dapat menginstall *background* tanpa diketahui pemilik, qq.apk dapat mengontrol dan mengakses lokasi yang sudah terhubung dengan web yang ditemukan pada locationclient *class* dan *server Command & Controll* dengan menggunakan ip 225.226.58.202 [10].

4. **Nur Widiyasono, Husni Mubarok dan Agung Fatwa MF pada tahun 2022 dengan judul "Analisis *Malware Ahmyth* pada *Platform* Android Menggunakan Metode *Reverse Engineering*".**

   Riset ini telah melakukan analisis dan mengidentifikasi *malware* yang diterapkan dalam aplikasi berbasis android. *Malware Ahmyth* menggunakan metode *Reverse Engineering*. Proses *malware* ini melakukan ekstraksi akses perizinan perangkat seperti audio, lokasi, sms, dan lainnya. Hasil yang didapat dalam penelitian ini *Malware Ahmyth* harus melakukan *booting* pada Android agar dapat menjalankan *mainservice* secara otomatis menemukan Alamat IP dan MAC dari *server Command & Control* bertujuan untuk mengontrol, mengirimkan informasi pada perangkat yang telah terinfeksi berupa akses direktori *file*, membaca sms, dan lainnya [11].

5. **Andriyan Dwi Putra, Joko Dwi Santoso, Ipung Ardiyansyah pada tahun 2022 dengan judul "Analisis *Malicious Software Trojan Downloader* Pada Android Menggunakan Teknik *Reverse Engineering*".**

   Analisis *Malicious* melakukan penginfeksian *malware tipe trojan downloader* pada Kamus Kesehatan v2.apk dengan Metasploit yang diinfeksi oleh *payload*. Penelitian ini menggunakan Teknik *Reverse Engineering*. Proses analisis dilakukan dengan menginfeksi aplikasi Kamus Kesehatan menggunakan *trojan downloader* sebagai perantara dengan menggunakan alat Metasploit *framework* lalu menggunakan Mara *framework* bertujuan untuk membongkar *source code*. Hasil yang ditemukan yaitu perbedaan ukuran *file* yang lebih besar setelah disisipi *malware* dan telah ditemukan 9 *permissions* sebelum disisipi dan setelah disisipi terdapat 18 *permissions* dalam aplikasi Kamus Kesehatan v2.apk [12].

6. **Frenvol De Santonario Magno Moises dan Joko Dwi Santoso, M.Kom pada tahun 2023 dengan judul "Analisis *Malware* Android Menggunakan Metode *Reverse Engineering*".**

   Analisis penelitian *malware* Android menggunakan Metode *Reverse Engineering* dilakukan pada malware berjenis *trojan* menggunakan aplikasi syssecApp.Apk. *Malware Trojan* diterapkan dalam APKTool, JD-GUI untuk menguraikan kode lalu melakukan analisis. Proses analisis *malware* ini melakukan identifikasi menggunakan *Virus* dan memahami beberapa fitur yang terdapat dalam aplikasi syssecApp.Apk. Hasil yang didapat adalah menghitung *hash* lalu dilakukan *decompile* dengan APKTool, ditemukannya 13 *permissions* dan menganalisis *source code* dengan JD-GUI dan terdapat 1 *class* berfungsi mengirimkan *file* data ke *host*. Hal ini dapat menganggu privasi dan keamanan ke pengguna [13].

Tabel 2.1 Penelitian Terkait

No	Judul	Tahun	Peneliti	Isi Penelitian	Perbedaan
1	Pengujian dan Analisa *Reverse Engineering* Pada *Platform* Android (Studi Kasus: Tebak_Gambar.apk)	2020	I Putu Agus Eka Pratama, Rasendriya Revo Daniswara	Telah dilakukan analisis membongkar *file* yang bertujuan mencari *source code* dengan *ApkTool* untuk mengidentifikasi komponen terdapat di Tebak_Gambar.apk	Menyelipkan *malware* pada Telegram melalui *evil-droid*, kemudian melakukan pemindahan aplikasi yang telah disisipkan *malicious software.*
2	Analisis *Malware PlasmaRAT* menggunakan Metode *Reverse Engineering*	2020	M. Hazri	Penelitian dilakukan identifikasi dan analisis mengenai *malware* jenis *Plasma RAT* dengan teknik *reverse engineering* bertujuan membongkar susunan beserta kegunaan *malware* jenis *Plasma RAT*, dan mengidentifikasi komponen yang ada didalamnya.	Menyisipkan *malware* pada aplikasi Telegram menggunakan *evil-droid*, kemudian melakukan *scanning* aplikasi yang telah disisipkan *malware.*
3	Analisis *Malware* Android Menggunakan Metode *Reverse Engineering*	2020	Bagus Aji Saputro, Lisan Iqbal Alfitra, Raykhan Bima Oktaviaji	Penyisipan *malware* dalam aplikasi menggunakan teknik *reverse engineering* bertujuan untuk membuat *user* sadar mengenai akibat dari aplikasi sudah terinfeksi oleh *malware*. Penelitian ini melakukan analisis statis agar memberikan pemahaman kepada pengguna Android	Menyisipkan *malware* pada aplikasi Telegram menggunakan *evil-droid*, kemudian melakukan pemindaian aplikasi setelah disisipkan *malware.*

4	Analisis *Malware Ahmyth* pada *Platform* Android Menggunakan Metode *Reverse Engineering*	2022	Nur Widyasono, Husni Mubarok, Agung Fatwa MF	Menyisipkan *malware* jenis *AhMyth* kedalam aplikasi dengan menerapkan analisis tipe dinamis, lalu memperoleh akses memperkenankan berbahaya yang dimanfaatkan *malware AhMyth* dengan *teknik Reverse Engineering.*	Menyisipkan *malicious software* pada aplikasi Telegram mengaplikasikan *evil-droid*, kemudian melakukan *scanning* aplikasi yang telah disisipikan *malware*.
5.	Analisis *Malicious Software Trojan Downloader* Pada Android Menggunakan Teknik *Reverse Engineering* (Studi Kasus: Kamus Kesehatan v2.apk)	2022	Andriyan Dwi Putra Joko Dwi Putra, Ipung Ardiyansyah	Melakukan analisis malware *trojan downloader* dalam aplikasi Kamus Kesehatan v2.apk menggunakan metode *reverese engineering.* Proses analisis dengan *trojan downloader* menggunakan alat metasploit *framework* lalu membongkar aplikasi menggunakan Mara *Framework* yang bertujuan untuk mendapatkan *source code.*	Menyisipkan *malware* pada aplikasi Telegram menggunakan *evil-droid*, kemudian melakukan scanning aplikasi yang telah disisipikan *malware*.
6.	Analisis *Malware* Android menggunakan Metode *Reverse Engineering*	2023	Frenvol De Santonario Magno Moises, Joko Dwi Santoso, M.Kom.	Menyelipkan *malicious software trojan* dengan syssecApp.apk menggunakan teknik *reverse engineering*. Proses analisis dilakukan menggunakan *APKTOOL* dan diterapkan *JD-GUI* berfungsi untuk mengungkap kode dan menelaah sumber daya yang didapat dalam sebuah aplikasi.	Menyisipkan *malicious software* dalam Telegram dengan menggunakan *evil-droid*. Setelah itu MobSF dilakukan untuk melakukan pemindaian aplikasi yang sudah diselipkan *malware*

2.2 Dasar Teori

Penelitian dibutuhkannya berbagai macam teori pendukung yang berkaitan dengan topik penelitian. Landasan Teori diambil melalui berbagai sumber seperti jurnal, website, buku dan lain sebagainya.

2.2.1 Android

Android adalah sistem operasi bersifat *Open-Source* yang berbasis kernel Linux yang dirancang untuk perangkat keras seperti smartphone, PC dan Tablet. Android dirilis pada 23 September 2008. Versi pertama Android 1.0 dapat disebut Astro lalu update versi menjadi Android 1.1 yang terdapat beberapa fitur aplikasi Google seperti Gmail, Maps, Kalender dan Youtube. Selanjutnya Android 1.5 versi *Cupcake, Donut, éclair, Jelly Bean, Kitkat, Lollipop, Marshmallow, Nougat, Oreo, Pie Q, Red Velvet Cake, Snow, Tiramisu, Upside Down* dan *Vanilla Cream*. Dengan berjalannya waktu perkembangan android pada versi 10 bernama *Quince Tart* memiliki fitur teks pada video tanpa terhubung dengan *wifi* atau data seluler [14]. Pada versi 12 yang memiliki keunggulan dapat menambah fitur baru, seperti *privacy dashboard*. Android memiliki kelebihan yaitu *open source, cloud storage*, pulihkan dan cadangkan aplikasi dengan mudah, dapat mendukung aplikasi pihak ketiga, Notifikasi, *Hotspot* seluler, *Upgrade* memori, *scan wifi* dan lain sebagainya. Sedangkan Kekurangan pada android yaitu Aplikasi dapat menghabiskan baterai *smartphone,* perlindungan virus, banyaknya iklan aplikasi, membutuhkan akun google, dan lain sebagainya [9].

2.2.2 Malware

Malware berasal dari kata *malicious* artinya jahat dan *software* yang memiliki arti perangkat lunak. *Malware* adalah sebuah aplikasi yang dimanfaatkan untuk menyelinap ke dalam sistem yang berisikan perintah untuk tujuan tertentu. Perintah yang dapat dalam *malware* berisi sisipan sebuah virus, trojan worm atau *backdoor* pada sebuah sistem yang biasanya telah menyelinap kedalam *file* unduhan seperti

situs *web* ilegal, aplikasi, iklan, dan lain-lain. Hal ini dapat dilakukan oleh *malware* melalui jaringan internet. *Malware* memiliki beberapa jenis yaitu:

1. Virus

   Virus adalah nama jenis malware yang terdapat pada unduhan situs *web*, koneksi jaringan, USB, dan lainnya. Virus diciptakan untuk mengganggu proses sistem dengan merusak, menghilangkan data, informasi atau dokumen. Virus dapat menyebar kedalam sistem komputer tanpa diketahui oleh pengguna.

2. Adware

   Adware adalah jenis malware yang biasanya memunculkan iklan pada situs *web* dengan melakukan aktivitas tertentu. Adware dapat mengirimkan *spyware* yang dapat melihat aktivitas komputer dan mengumpulkan data informasi yang dapat digunakan oleh *hacker*.

3. Trojan

   Trojan adalah jenis *malware* yang melakukan penyamaran dalam sebuah aplikasi yang tidak berbahaya dengan meyakinkan *user* untuk mengunduh dan menggunakan aplikasi. Trojan akan menyebar dengan cepat dan dapat melihat aktivitas yang terjadi ketika aplikasi tersebut telah di unduh oleh pengguna.

4. Worm

   Worm adalah jenis *malware* yang dapat menggandakan agar dapat menyebar dalam sistem. Worm dapat msuk kedalam jaringan internet, aplikasi ilegal atau dokumen yang dikirim melalui *e-mail.*

5. Botnet

   Botnet adalah jenis *malware* seperti sekumpulan bot dapat menyusup ke jaringan dan sistem komputer yang dikendalikan *hacker.*

6. Ransomware

Ransomware adalah jenis *malware* yang masuk kedalam akses data menggnakan sistem komputer dengan cara mengunci dan menolak pengguna agar dapat diakses. Ransomware dilakukan para *hacker* untuk kejahatan *cyber* yang biasanya melakukan tebusan dengan menuntut uang agar dapat membuka kembali sistem [15].

2.2.3 Backdoor

Backdoor (pintu belakang) adalah suatu Teknik/metode/cara rahasia yang digunakan untuk mengakses system atau perangkat lunak dengan sengaja. Backdoor dapat sulit dideteksi biasanya ditemukan oleh pengguna yang mempunyai akses aplikasi source code [10]. Umumnya backdoor digunakan untuk tindak kejahatan secara illegal.

2.2.4 Payload

Payload adalah menyediakan kode yang dapat menggunakan *shell* berfungsi sebagai sistem dieksekusi, dan disampaikan kepada *framework* seperti r*everse shell* digunakan untuk membuat koneksi pada *payload* dengan *meterpreter* yang berfungsi sebagai dari target kembali ke penyerang [16]. *Payload* terdapat beberapa perintah yang dijalankan sistem sebagai berikut:

1. *Android/meterpreter/reverse_tcp*

   Meterpreter reverse_tcp adalah *payload* yang dapat koneksi tcp langsung, cepat, tetapi mudah terdeteksi.

2. *Android/meterpreter/reverse_http*

   Meterpreter reverse_http adalah *payload* yang dapat melewati *firewall* dan *proxy* menggunakan http tetapi lebih lambat.

3. *Android/meterpreter/reverse_https*

   Meterpreter reverse_http adalah *payload* yang aman, sulit dideteksi tetapi memiliki *latency* (waktu jaringan) lebih tinggi.

4. *Android/shell/reverse_tcp*

Shell Reverse tcp adalah payload yang sederhana ringan dapat digunakan *tcp* secara langsung tetapi tidak canggih.

2.2.5 Meterpreter

Meterpreter adalah *payload* yang menyediakan *shell* untuk penyerang bertujuan untuk mengeksplorasi akses kedalam sistem yang dituju ketika sudah berhasil di eksploitasi dengan menjalankan kode terdapat pada Metasploit. Metasploit memiliki kemampuan *remote* sistem korban. Tindakan dapat dilakukan penyerang dengan melakukan modifikasi, mendeteksi, mengambil alih informasi data peranti korban yang diakses secara tidak sah atau diretas [17].

2.2.6 Evil-Droid

Evil-Droid adalah sebuah *tool* penetrasi yang digunakan untuk membuat dan menghasilkan muatan/*payloads* apk dalam *platforms* android.[11] Untuk menginstal *Evil-Droid* menggunakan perintah *sudo apt-get install openjdk-17-jdk*[12].

2.2.7 JADX

JADX adalah sebuah *tool* penetrasi digunakan untuk mengekstrak berkas APK (*Android Application Package*) bersifat *open-source* [18].

2.2.8 MobSF

MobSF (*Mobile Security Framework*) adalah *framework* bersifat *open source* digunakan untuk melakukan *penetration testing* terhadap aplikasi (Android/iOS/Windows) [14]. Format aplikasi yang mendukung MobSF adalah APK, XAPK, IPA dan APPX dengan kode sumber menggunakan format .rar. MobSF memiliki berbagai fitur analisis yaitu seperti pemindahan statis dan dinamis, kode sumber dan pemindahan tanda tangan [15]. Penggunaan MobSF dapat diunduh melalui github dengan *git clone https:/github.com/MobSF/Mobile-Security-Framework-MobSF.git* atau dapat menggunakan docker.io, jika belum memiliki *docker* dapat melakukan instalasi menggunakan *sudo apt install docker.io*, lalu untuk menginstall mobsf dengan

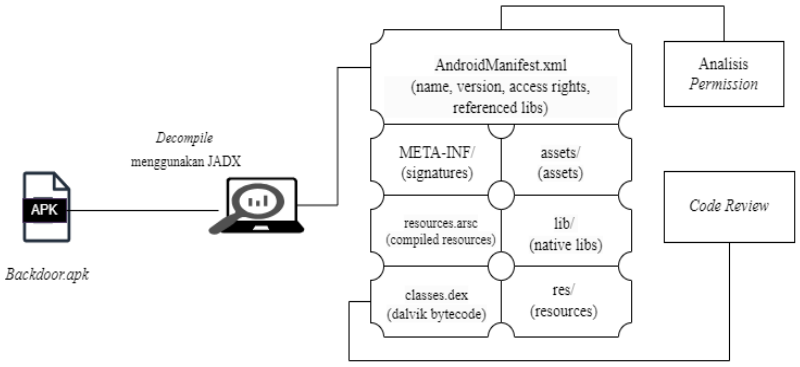
menggunakan *sudo docker pull opensecurity/mobile-security-framework-mobsf* [16].

2.2.9 Telegram

Telegram adalah aplikasi chatting gratis yang digunakan menggunakan internet. Aplikasi ini hampir menyerupai Whatsapp dan Messenger tetapi rata-rata pengguna lebih memilih menggunakan telegram, dikarenakan terdapat banyak media penghubung informasi seperti dapat menampung ribuan anggota dalam grup, terdapat film dan *chat boot*. Kelebihan lainnya yaitu keamanan pesan terjamin, *Self-destruct timers* (dapat mengirimkan *text*, foto, video yang terhapus secara otomatis), dan dapat mengirim *file* jumlah besar. Kekurangan pada aplikasi telegram adalah tidak dapat melakukan panggilan video lebih dari 2 orang lebih, dan tidak terdapat fitur *offline* untuk pengguna.[17]. Penelitian ini menggunakan Telegram versi 10.3.2 yang didapat dari Apkpure.

2.2.10 Reverse Engineering

Reverse engineering adalah suatu cara digunakan untuk menyelidiki *malware* yang bekerja dengan meneliti kode didalamnya. Dalam analisis *malware* ini menggunakan teknik *reverse engineering* untuk membuktikan cara kerja, kemampuan, dan dampak yang terjadi dari *malware*. Proses yang dilakukan metode ini dengan membongkar program lalu menguraikannya. Dengan melakukan metode ini peneliti memeriksa beberapa kode yang tersimpan didalam *malware* dengan mengidentifikasi fungsi yang digunakan [12].



Gambar 2.1 Metode *Reverse Engineering*

Tahap ini Telegram yang sudah diselipi *malware* melakukan *decompile* JADX. Ketika *decompile* hasil yang didapat berupa beberapa *file* bagian inti pada aplikasi Telegram. Berkas didapat seperti *AndroidManifest*, *assets*, META-INF, *resources*, *lib*, *res*, dan classes.dex. *AndroidManifest* adalah *file* inti berada manifest yang terdapat izin penting aplikasi yang meliputi informasi android. Assets adalah *file* yang memiliki isi *asset* tambahan berupa berkas data yang tidak diubah sistem Android contohnya emoji, *text*, font, dan lainnya. *Res* adalah folder inti sudah terkompilasi aplikasi yang terdapat *string*, *layout*, *drawable*, ikon, animasi, dan *resource* lainnya sebagai penyimpanan berupa *text*, media atau lainnya. META-INF adalah metadata dan berbagai informasi digunakan untuk verifikasi keaslian, pengamanan, dan tujuan sistem memastikan aplikasi asli atau dimodifikasi. *Lib* adalah *folder* yang berfungsi untuk menyimpan pustaka atau *library native* modul tambahan sesuai arsitektur. *Resources* adalah *file* .arsc yang berisi informasi terenkripsi, gambar, teks, *string*, dan lainnya. Classes.dex adalah komponen terkompilasi yang berisi implementasi dari kode seperti *bytecode* yang dikompilasi kode java.