BAB II
TINJAUAN PUSTAKA

2.1. Penelitian Terkait

Penelitian yang akan dilakukan tidak terlepas dari hasil penelitian sebelumnya yang dihasilkan sebagai bahan kajian dan bahan perbandingan. Penelitian yang digunakan sebagai perbandingan tidak terlepas dari topik penelitian yaitu mengenai *malware*, *exploit*, *backdoor*, dan *reverse engineering*. Berikut merupakan beberapa penelitian terkait dengan topik penelitian yang dilakukan peneliti:

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

   Penelitian ini melakukan penyisipan *malware* dengan jenis trojan pada aplikasi syssecApp.apk menggunakan metode *reverse engineering*. Dalam proses analisis, *tools* yang digunakan adalah APKTOOL dan JD-GUI untuk membongkar kode dan menganalisis sumber daya yang terdapat dalam aplikasi. Hasil dari analisis menunjukkan adanya *ip host* penerima yang terdapat dalam *source code* yang tersembunyi di dalam syssecApp.apk. Dengan demikian, analisis ini bertujuan untuk mengidentifikasi dan memahami fitur-fitur *malware* Trojan yang telah menyusup ke dalam aplikasi tersebut melalui pendekatan *reverse engineering*. Melalui penelitian ini, diharapkan dapat diperoleh wawasan yang lebih mendalam tentang cara kerja *malware* Trojan dalam konteks aplikasi syssecApp.apk serta potensi dampaknya terhadap keamanan dan privasi pengguna[9].

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

   Penelitian ini melakukan penyisipan *malware* AhMyth pada aplikasi Android melalui penggunaan analisis dinamis, serta mengekstrak izin berbahaya yang dimanfaatkan oleh *malware* AhMyth menggunakan teknik *reverse engineering*. Melalui pendekatan analisis dinamis, penelitian ini berusaha untuk memantau perilaku *malware* AhMyth saat beroperasi di dalam perangkat, sementara melalui teknik *reverse engineering*, dilakukan ekstraksi detail perizinan berbahaya yang dimanfaatkan oleh *malware* tersebut. Hasil dari analisis menunjukkan bahwa *malware* AhMyth akan mengaktifkan layanan *(service)* yang ada setelah perangkat Android melakukan *restart*, dan kemudian menunggu instruksi dari *server* C&C (*Command and Control*) untuk menjalankan tindakan tertentu pada perangkat yang telah terinfeksi[10].

3. **Tesa Pajar Setia, Nur Widiyasono dan Aldy Putra Aldya pada tahun 2018 dengan judul *"Analysis Malware Flawed Ammyy RAT Dengan Metode Reverse Engineering"***

   Penelitian ini melakukan identifikasi *malware*, khususnya *malware* Flawed Ammyy RAT. Metodologi yang digunakan adalah deskriptif, yang kemudian dilanjutkan dengan analisis *malware* menggunakan metode analisis dinamis dan *reverse engineering*. Hasil dari penelitian ini mengungkapkan bahwa *malware* Flawed Ammyy RAT bekerja dengan cara menyamar dalam aplikasi Ammyy Admin, dan kemudian melakukan koneksi dengan *attacke*r menggunakan alamat IP 103.208.86.69 yang memiliki *netname* "zappie host". Selain itu, terdapat perubahan pada 50 registry dalam sistem yang terinfeksi oleh *malware*. Setelah berhasil terhubung dengan korban, attacker dapat dengan mudah *melakukan remote control* tanpa sepengetahuan korban[11].

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

   Penelitian ini melakukan analisis dan identifikasi *malware* Plasma RAT menggunakan metode *reverse engineering*. Dalam proses analisisnya, ditemukan bahwa *malware* Plasma RAT memiliki beberapa program tambahan yang berjalan saat *malware* ini diaktifkan pada suatu sistem. Selain itu, ditemukan juga bahwa *malware* Plasma RAT menggunakan teknik anti-*reverse engineering* yang bertujuan untuk menghalangi upaya untuk melakukan *reverse engineering* pada programnya. Melalui pendekatan *reverse engineering*, penelitian ini bertujuan untuk memecahkan struktur dan fungsi dari *malware* Plasma RAT, serta untuk mengidentifikasi komponen-komponen yang terlibat dalam operasinya. Meskipun dihadapkan dengan tantangan dari teknik anti-*reverse engineering* yang digunakan oleh *malware* ini, penelitian ini berusaha untuk mengatasi hambatan tersebut dan mengungkapkan informasi yang diperlukan untuk memahami cara kerja dan dampak dari *malware* Plasma RAT[12].

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

   Penelitian ini melakukan penyisipan *malware* pada sebuah aplikasi dengan metode *reverse engineering*. Tujuannya adalah untuk meningkatkan kesadaran pengguna akan dampak yang dapat terjadi jika aplikasi yang mereka gunakan terinfeksi *malware*. Dengan memeriksa isi kode dari percobaan *malware* menggunakan analisis statis, penelitian ini bertujuan untuk memberikan pemahaman yang lebih baik kepada pengguna Android tentang potensi risiko yang terkait dengan *malware*. Hasil yang ditemukan adalah dapat menginstall aplikasi pada *background* tanpa sepengetahuan pemilik kemudian dapat merekam panggilan yang diterima oleh korban, mengumpulkan data dari perangkat smartphone,

membaca pesan teks yang masuk, mendapatkan hak akses administratif, dan mengirimkan informasi yang terkumpul ke *server* yang dikendalikan oleh penyerang dengan alamat IP 221.226.58.202[13].

6. **S Megira, A R Pangesti dan F W Wibowo pada tahun 2018 dengan judul *"Malware Analysis and Detection Using Reverse Engineering Technique"***

   Penelitian ini melakukan analisis terhadap *malware* dengan tujuan memahami potensi bahaya yang terkandung di dalamnya, serta mengidentifikasi cara untuk mencegah dan melindungi perangkat dari ancaman tersebut. Menggunakan metode *reverse engineering* yang digunakan untuk mengekstrak data dalam suatu *malware* untuk mengetahui cara kerja *malware* tersebut ketika menyerang ke dalam sistem. Dalam penelitian ini, sebuah file yang bernama best.exe akan dijadikan sebagai sampel *malware* untuk digunakan dalam analisis. Hasil yang diperoleh adalah bahwa file best.exe adalah *malware* yang bersifat virus Gen:Variant.Razy dengan ukuran file 626 KB. Virus Gen:Variant.Razy adalah virus yang bisa terdeteksi oleh beberapa anti virus dan anti *malware* karena virus ini menyerang sistem komputer dengan OS Windows. Virus tersebut dapat menyembunyikan jejak setelah diunduh, dapat mengetahui nama komputer yang aktif, dapat menjadikan mode *sleep* komputer dengan lama, dapat membuat proses/tugas baru, dapat mengirimkan informasi tentang komputer yang terinfeksi kepada *hacker* dan dapat merekam riwayat penelusuran[14].

Tabel 2.1. Penelitian Terkait

No	Judul	Tahun	Peneliti	Isi Penelitian	Perbedaan
1	Analisis *Malware* Android Menggunakan Metode *Reverse Engineering*	2023	Frenvol De Santonario Magno Moises dan Joko Dwi Santoso, M.Kom	Melakukan penyisipan *malware* dengan jenis trojan pada aplikasi syssecApp.apk menggunakan metode *reverse engineering*. *Tools* yang digunakan adalah APKTOOL dan JD-GUI untuk membongkar kode dan menganalisis sumber daya yang terdapat dalam aplikasi.	Menyisipkan *malware* pada aplikasi Signal Messenger dengan menggunakan kwetza. Kemudian menggunakan MobSF untuk melakukan *scanning* aplikasi yang sudah disusupi *malware*.
2	Analisis *Malware* Ahmyth pada Platform Android Menggunakan Metode *Reverse Engineering*	2022	Nur Widiyasono, Husni Mubarok dan Agung Fatwa MF	Melakukan penyisipan *malware* AhMyth pada aplikasi Android menggunakan analisis dinamis, dan mengekstrak izin berbahaya yang dimanfaatkan oleh *malware* AhMyth menggunakan teknik *reverse engineering.*	Menyisipkan *malware* pada aplikasi Signal Messenger dengan menggunakan kwetza. Kemudian menggunakan MobSF untuk melakukan *scanning* aplikasi yang sudah disusupi *malware*.

No	Judul	Tahun	Peneliti	Isi Penelitian	Perbedaan
3	*Analysis Malware* Flawed Ammyy RAT Dengan Metode *Reverse Engineering*	2018	Tesa Pajar Setia, Nur Widiyasono dan Aldy Putra Aldya	Melakukan identifikasi *malware*, khususnya *malware* Flawed Ammyy RAT. Metodologi yang digunakan adalah deskriptif, yang kemudian dilanjutkan dengan analisis *malware* menggunakan metode analisis dinamis dan *reverse engineering*.	Menyisipkan *malware* pada aplikasi Signal Messenger dengan menggunakan kwetza. Kemudian menggunakan MobSF untuk melakukan *scanning* aplikasi yang sudah disusupi *malware*.
4	Analisis *Malware* PlasmaRAT dengan Metode *Reverse Engineering*	2020	M. Hazri	Melakukan analisis dan identifikasi *malware* Plasma RAT menggunakan metode *reverse engineering*. Penelitian ini bertujuan untuk memecahkan struktur dan fungsi dari *malware* Plasma RAT, serta melakukan identifikasi terhadap komponen-	Menyisipkan *malware* pada aplikasi Signal Messenger dengan menggunakan kwetza. Kemudian menggunakan MobSF untuk melakukan *scanning* aplikasi yang sudah disusupi *malware*.

No	Judul	Tahun	Peneliti	Isi Penelitian	Perbedaan
				komponen yang terlibat dalam sistem operasinya.	
5	Analisis *Malware* Android Menggunakan Metode *Reverse Engineering*	2020	Bagus Aji Saputro, Lisan Iqbal Alfitra dan Raykhan Bima Oktaviaji	Melakukan penyisipan *malware* pada sebuah aplikasi dengan metode *reverse engineering*. Tujuannya adalah untuk meningkatkan kesadaran pengguna akan dampak yang dapat terjadi jika aplikasi yang mereka gunakan terinfeksi *malware*. Dengan memeriksa isi kode dari percobaan *malware* menggunakan analisis statis, penelitian ini bertujuan untuk memberikan pemahaman yang lebih baik kepada pengguna Android tentang potensi risiko	Menyisipkan *malware* pada aplikasi Signal Messenger dengan menggunakan kwetza. Kemudian menggunakan MobSF untuk melakukan *scanning* aplikasi yang sudah disusupi *malware*.

No	Judul	Tahun	Peneliti	Isi Penelitian	Perbedaan
				yang terkait dengan *malware*.	
6	*Malware Analysis* and Detection Using *Reverse Engineering* Technique	2018	S Megira, A R Pangesti dan F W Wibowo	Melakukan analisis terhadap *malware* dengan tujuan memahami potensi bahaya yang terkandung di dalamnya, serta mengidentifikasi cara untuk mencegah dan melindungi perangkat dari ancaman tersebut. Menggunakan metode *reverse engineering* yang digunakan untuk mengekstrak data dalam suatu *malware* untuk mengetahui cara kerja *malware* tersebut ketika menyerang ke dalam sistem.	Menyisipkan *malware* pada aplikasi Signal Messenger dengan menggunakan kwetza. Kemudian menggunakan MobSF untuk melakukan *scanning* aplikasi yang sudah disusupi *malware*.

2.2.Dasar Teori

Penelitian ini memerlukan landasan teori yang relevan dengan topik penelitian, yang dapat diperoleh dari berbagai sumber seperti jurnal ilmiah, skripsi, dan situs web terpercaya. Landasan teori ini akan mendukung pemahaman yang lebih baik tentang topik penelitian dan konteksnya. Melalui penggalian berbagai sumber, teori-teori yang berkaitan dengan masalah yang diteliti akan disajikan, membantu menyusun kerangka konseptual dan mengidentifikasi variabel-variabel yang relevan untuk penelitian ini. Dengan menggunakan dasar teori yang kokoh, diharapkan penelitian ini dapat memberikan kontribusi yang berarti dalam memperluas pemahaman kita tentang topik yang sedang diteliti.

2.2.1. *Malware*

Malware adalah aplikasi khusus yang digunakan untuk menyusup ke dalam sistem tanpa diketahui oleh pemilik sistem. Umumnya *malware* memuat sebuah perintah yang dibuat untuk tujuan khusus. Perintah tersebut seperti menyuntikan sebuah virus, *trojan, worm,* atau menyisipkan *backdoor* ke dalam sebuah sistem. *Malware* juga dapat diartikan sebagai aplikasi yang dibangun untuk membuat celah keamanan pada sistem komputer. *Malware* yang dikembangkan pertama kali adalah *malware* jenis *worm* dan virus. *Malware* jenis *worm* dan virus menyebar dengan sangat cepat, penyebaran tersebut dilakukan melalui media penyimpanan dan jaringan *local area network*. Terdapat empat jenis tipe *malware,* yakni:

1. *Malware* Tipe Virus

   Malware jenis ini digunakan untuk melakukan serangan dengan cara masuk ke dalam sistem untuk melakukan infeksi dan menggandakan diri. *Malware* tipe termasuk jenis virus dan *worm*.
2. *Malware* Tipe Terselubung

   Malware jenis ini digunakan untuk melakukan penyamaran ke dalam sebuah sistem, hal tersebut membuat *user* atau pengguna tidak dapat menyadari bahwa sistem yang dimiliki telah terinfeksi.
3. *Malware* Tipe Profit Oriented

Malware jenis ini digunakan untuk memperoleh keuntungan pribadi, kelompok, organisasi, dan negara. *Malware* tipe ini banyak digunakan untuk melakukan infeksi dengan menyebar virus melalui *trojan horse*[15].

2.2.2. Kwetza

Kwetza adalah alat yang digunakan untuk menyisipkan *payload meterpreter* ke dalam aplikasi Android. Alat ini berfungsi dengan cara menginfeksi aplikasi Android menggunakan template *payload custom* atau default, yang bertujuan untuk mengelabui deteksi oleh perangkat lunak antivirus. Kwetza melakukan infeksi dengan memanfaatkan izin default dari aplikasi target atau dengan menyuntikkan izin tambahan untuk memperoleh fungsionalitas tambahan. Secara bawaan, kwetza akan menggunakan template dan *keystore* yang tersimpan di dalam folder "*payload*" untuk menyuntikkan dan menandatangani APK yang telah terinfeksi. Dengan demikian, kwetza memungkinkan penyerang untuk menyuntikkan *payload meterpreter* ke dalam aplikasi Android tanpa memicu peringatan dari sistem keamanan[16].

2.2.3. *Reverse Engineering*

Reverse engineering adalah salah satu metode yang digunakan untuk menganalisis *malware* dengan cara memeriksa kode-kode yang ada di dalamnya. Dalam konteks analisis *malware*, teknik *reverse engineering* memungkinkan para peneliti keamanan untuk mengungkap cara kerja, perilaku, serta potensi dampak dari suatu *malware*. Dengan melakukan *reverse engineering*, para peneliti dapat memeriksa kode-kode yang tersimpan di dalam *malware*, mengidentifikasi fungsi-fungsi yang digunakan, serta mengeksplorasi kemungkinan tindakan yang akan dilakukan oleh *malware* tersebut. Hal ini memungkinkan para peneliti untuk memahami lebih dalam tentang fitur-fitur yang tersembunyi di dalam *malware*, serta memungkinkan mereka untuk mengembangkan solusi perlindungan dan deteksi yang lebih efektif. Dengan kata lain, *reverse engineering* menjadi salah satu pendekatan yang penting dalam memerangi ancaman *malware* dengan memberikan pemahaman yang mendalam tentang karakteristik dan perilaku *malware* tersebut[17].

2.2.4. *Mobile Security Framework (MobSF)*

Mobile Security Framework (MobSF) adalah sebuah kerangka kerja yang digunakan untuk melakukan pengujian penetrasi terhadap aplikasi mobile (Android/iOS/Windows), analisis *malware*, dan evaluasi keamanan secara menyeluruh. MobSF merupakan proyek sumber terbuka (*opensource*) yang mendukung berbagai format binari aplikasi seluler seperti APK, XAPK, IPA, dan APPX, serta menyediakan dukungan untuk menganalisis kode sumber yang dikemas dalam file zip. Kerangka kerja ini menawarkan sejumlah fitur yang kuat untuk menganalisis aplikasi mobile, termasuk pemindaian statis dan dinamis, analisis kode sumber, pemindaian kerentanan, pemindaian tanda tangan, serta analisis risiko. MobSF memungkinkan pengguna untuk melakukan pemindaian statis, yang mencakup analisis struktur dan isi dari aplikasi mobile tanpa harus menjalankannya. Selain itu, MobSF juga mendukung pemindaian dinamis, di mana aplikasi yang sedang berjalan pada runtime dapat dianalisis untuk menemukan kerentanan dan ancaman keamanan lainnya. Kerangka kerja ini juga memiliki kemampuan untuk menganalisis kode sumber aplikasi, yang memungkinkan para peneliti keamanan untuk mendeteksi kerentanan yang mungkin terdapat pada implementasi aplikasi. Selain itu, MobSF juga menyediakan fitur pemindaian tanda tangan untuk mengidentifikasi keberadaan tanda tangan *malware* yang telah diketahui. Selain fitur-fitur tersebut, MobSF juga memiliki kemampuan untuk melakukan fuzzing pada API web yang didukung oleh CapFuzz, serta menyediakan pemindai keamanan khusus untuk web API. Dengan demikian, MobSF menjadi salah satu alat yang sangat berguna bagi para peneliti keamanan untuk mengevaluasi, menguji, dan meningkatkan keamanan aplikasi mobile[18].

2.2.5. Metasploit

Metasploit adalah sebuah platform yang digunakan untuk melakukan uji penetrasi dalam dunia keamanan. Platform Metasploit menyediakan seperangkat alat dan kerangka kerja yang digunakan untuk melakukan serangan simulasi, uji penetrasi dan pengujian secara umum. Metasploit dikembangkan untuk memberikan informasi mengenai suatu kerentanan yang terdapat pada suatu

sistem. Metasploit framework merupakan *tools* yang bersifat *opensource* yang dikembangkan oleh H D Moore. H D Moore merupakan seorang ahli dalam bidang keamanan jaringan pada tahun 2003. Metasploit dikembangkan untuk menyediakan sumber daya yang digunakan untuk pengembangan dan riset kode *exploit*. Awalnya Metasploit dikembangkan menggunakan bahasa Perl, namun *source code*nya ditulis ulang pada akhir tahun 2007 menggunakan bahasa yang berbeda yaitu bahasa Ruby, kemudian diakuisisi pada tahun 2009 oleh perusahaan Rapid7 yang bergerak di bidang keamanan teknologi informasi.

Metasploit memiliki lebih dari satu antarmuka diantaranya adalah:

1. Msfconsole

   Msfconsole adalah antarmuka yang memiliki fitur lengkap, seperti melakukan *exploit*, memuat *auxiliary module*, membuat *listener* dan lainnya.
2. Msfcli

   Msfcli lebih mengacu pada *scripting* dan *interpretability* dengan menggunakan *tools* yang berbasis *console*.
3. Armitage

   Armitage merupakan bagian dari Metasploit yang mempunyai desain antarmuka yang interaktif untuk memudahkan pengguna dalam menjalankan Armitage.

Terdapat beberapa langkah dasar untuk melakukan serangan menggunakan Metasploit, yaitu:

1. Menggunakan *exploit* untuk menguji kerentanan sistem.
2. Mencari informasi mengenai *exploit.*
3. Melakukan konfigurasi *payload*.
4. Mengatur opsi pengaturan.
5. Menjalankan *exploit*[19].

2.2.6. *Exploit*

Exploit adalah kode yang digunakan oleh penyerang untuk melakukan penyerangan terhadap keamanan komputer. *Exploit* sering digunakan untuk melakukan uji penetrasi secara legal maupun ilegal untuk mendapatkan celah kelemahan *(vulnerability)* dari sebuah sistem. *Exploit* juga dapat untuk memanfaatkan kerentanan yang terdapat pada sebuah sistem dengan tujuan untuk mendapatkan izin yang tidak sah, mengambil alih kontrol sistem, atau melakukan tindakan yang tidak diizinkan. *Exploit* dapat memanfaatkan kesalahan pada desain, *bug* pada perangkat lunak, atau konfigurasi yang buruk untuk mencapai tujuan[20].

Secara umum eksploitasi dapat dikelompokan menjadi dua, yaitu:

1. *Remote Exploit*

   Cara kerja dari *remote exploit* adalah melalui jaringan dan melakukan eksploitasi terhadap celah atau kelemahan pada suatu sistem tanpa adanya akses terlebih dahulu. *Remote exploit* biasanya melakukan serangan terhadap service yang sedang berjalan dan melakukan interaksi menggunakan jaringan luar, seperti *service http (Apache), service database (MySQL)* dan *service* lainnya.
2. *Local Exploit*

   Cara kerja dari *local exploit* adalah mengharuskan adanya akses ke dalam celah sistem dan biasanya untuk meningkatkan kebebasan orang untuk melakukan *exploit*. *Exploit* menyerang aplikasi *non service* yang tidak memiliki interaksi dengan jaringan luar dan biasanya terjadi dengan membuat file tertentu yang dibuat mirip atau sedemikian rupa sehingga aplikasi gagal menghandlenya[21].

2.2.7. *Backdoor*

Backdoor merupakan salah satu teknik peretasan yang dimanfaatkan untuk memperoleh akses ke suatu sistem tanpa terdeteksi oleh mekanisme keamanan sistem tersebut. Istilah ini juga merujuk pada mekanisme yang sengaja disisipkan dalam sistem komputer, perangkat lunak, atau jaringan dengan tujuan

memberikan akses yang tidak sah kepada pihak yang memasangnya. *Backdoor* memungkinkan penyerang untuk masuk ke dalam sistem tanpa harus melewati proses autentikasi atau mekanisme keamanan lainnya. Dalam konteks yang lebih luas, *backdoor* sering digunakan secara ilegal oleh penyerang untuk melakukan aksi kejahatan, seperti pencurian data, pengintaian, atau merusak sistem yang diserang. Dengan demikian, *backdoor* merupakan salah satu alat yang berpotensi merugikan dan sering kali dianggap sebagai ancaman serius bagi keamanan sistem komputer dan jaringan[22].

2.2.8. *Payload*

Payload adalah kode yang diinjeksikan ke dalam sistem target untuk memanfaatkan kerentanan dengan menggunakan modul *exploit* yang sesuai. *Payload* digunakan untuk mencapai tujuan penyerangan, seperti mengambil alih sistem, mencuri data, atau melakukan tindakan tertentu. *Payload* dapat berupa kode yang ditulis dalam bahasa pemrograman, skrip, atau bahkan *shellcode*. Contoh *payload* yang sering digunakan adalah *reverse shell* dan *bind shell*. R*everse shell* merupakan *payload* yang digunakan untuk membuat koneksi atau lalu lintas dari mesin target kembali ke penyerang sebagai *meterpreter console*, sedangkan *bind shell* adalah *payload* yang digunakan untuk mengikat *meterpreter console* untuk mendengarkan pada port mesin target dimana penyerang dapat terhubung. *Payload* menjadi salah satu dari beberapa perintah yang dapat dijalankan pada sistem operasi target. Berikut tipe-tipe *payload* untuk Android:

1. Android/meterpreter/reverse_tcp.
2. Android/meterpreter/reverse_http.
3. Android/metepreter/reverse_https.
4. Android/shell/reverse_tcp[23].

2.2.9. *Meterpreter*

Meterpreter adalah sebuah *payload* yang merupakan bagian integral dari kerangka kerja penetrasi Metasploit. Dengan menggunakan *Meterpreter*, penyerang dapat memperoleh akses ke sistem target yang telah berhasil dieksploitasi, terutama pada perangkat Android. Melalui *payload Meterpreter*

yang terpasang pada perangkat Android target, penyerang memiliki kemampuan untuk menjalankan berbagai perintah dan operasi secara jarak jauh. Salah satu kekuatan utama *Meterpreter* adalah kemampuannya untuk memberikan kontrol yang luas terhadap sistem target. Penyerang dapat menggunakan *Meterpreter* untuk melakukan berbagai tindakan seperti membaca, memodifikasi, atau mengambil data dan informasi dari perangkat target yang telah berhasil diserang melalui eksploitasi kerentanan. Dengan kata lain, *Meterpreter* dirancang dan dikembangkan dengan tujuan memberikan kontrol jarak jauh yang efektif kepada penyerang setelah berhasil mengeksploitasi kerentanan yang ada pada sistem target[24].

2.2.10. Signal Messenger

Signal Messenger adalah aplikasi *chatting* yang menempatkan privasi pengguna sebagai prioritas utama. Dikembangkan oleh Signal Technology Foundation dan Signal Messenger LLC, aplikasi ini memungkinkan pengguna untuk berkomunikasi secara individu maupun dalam grup dengan aman dan rahasia. Salah satu fitur utama dari Signal Messenger adalah penggunaan *end-to-end encryption*, sebuah metode komunikasi yang dirancang untuk mencegah pihak ketiga dari mengakses isi pesan yang sedang ditransmisikan. Ini berarti hanya pengirim dan penerima pesan yang memiliki akses ke konten tersebut. Selain itu, Signal Messenger menyediakan berbagai fitur, termasuk pesan teks, panggilan suara, panggilan video, pengiriman file, dan pengelolaan grup. Aplikasi ini juga tersedia sebagai sumber terbuka (*open-source*), yang berarti kode sumbernya dapat diakses dan diperiksa oleh siapa pun. Hal ini membantu meningkatkan kepercayaan dan transparansi dalam hal keamanan dan privasi. Signal Messenger dapat diunduh secara gratis dan tersedia untuk berbagai platform, termasuk Android, iOS, serta aplikasi desktop untuk Windows, MacOS, dan Linux[25].

2.2.11. Kali Linux

Kali Linux adalah distribusi dari Debian GNU/Linux yang ditujukan untuk keperluan keamanan informasi dan digunakan untuk melakukan uji penetrasi. Kali dikembangkan oleh Offensive Security dan dirancang sebagai penerus BackTrack

Linux. Kali memberikan akses yang mudah kepada pengguna terhadap alat dan komprehensif yang berkaitan dengan keamanan, seperti *port scanner* untuk *password cracker*. BackTrack Linux dibangun kembali sesuai dengan standar pengembangan Debian[26]. Secara umum Kali Linux memiliki berbagai macam *tools* yang dibagi menjadi beberapa klasifikasi berdasarkan fungsinya, yaitu:

1. *Information Gathering*

   Information gathering ini digunakan untuk melakukan pengumpulan informasi yang didapatkan dari suatu sistem,
2. *Reverse Engineering*

   Reverse engineering ini digunakan untuk melakukan analisa mengenai suatu sistem melalui identifikasi dari suatu komponen, keterkaitan antar komponen dan informasi mengenai perancangan sistem yang dianalisa.
3. *Exploitation Tools*

   Exploitations Tools digunakan untuk melakukan eksploitasi untuk mendapatkan celah dari suatu sistem.
4. *Vulnerability Assessment*

   Vulnerability Assessemnt ini digunakan untuk melakukan sebuah pencarian, mengidentifikasi dan melakukan perhitungan terhadap celah keamanan pada suatu sistem.
5. *Privilege Escalation*

   Privilege Escalation ini digunakan untuk melakukan serangan yang bertujuan untuk meningkatkan tingkat akses dalam suatu sistem[27].

2.2.12. JADX

Jadx adalah sebuah alat atau modul yang digunakan untuk mengekstrak file APK *(Android Application Package)* secara *open-source*. Alat ini memungkinkan pengguna untuk mengurai atau mengekstrak kode sumber dari aplikasi Android ke dalam format yang mudah dibaca dan dimengerti. Dengan menggunakan JADX, pengguna dapat melihat kode sumber dari aplikasi Android, termasuk berbagai *class*, metode, dan fungsi yang terkandung di dalamnya. Alat ini sangat berguna bagi para pengembang aplikasi Android untuk menganalisis

kode sumber aplikasi, memahami logika kerja aplikasi, serta memeriksa keamanan dan kerentanan yang mungkin ada di dalamnya. Dengan status *open-source*, JADX juga memberikan fleksibilitas bagi pengguna untuk berkontribusi dalam pengembangan dan pemeliharaannya, serta memastikan transparansi dan keamanan dalam penggunaan alat ini. Dengan demikian, JADX menjadi salah satu alat yang penting dan berguna dalam pengembangan, analisis, dan penelitian aplikasi Android[28].