BAB II

TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 *Malware*

Malware adalah akronim dari *Malicious Software* yang berarti perangkat lunak berbahaya. *Malware* adalah program yang melakukan tindakan kejahatan. *Malware* dapat berbentuk program executable, script, code atau perangkat lunak lainnya. *Malware* biasanya masuk ke sistem komputer tanpa persetujuan dari pengguna melalui berbagai media seperti Email, web atau USB drives. *Malware* dibuat untuk merusak, membobol, mencuri informasi penting, memata-matai dan mengambil alih kontrol perangkat lunak atau sistem operasi (Triantoro, Widiyasono dan Gunawan, 2021).

2.1.2 Jenis *Malware*

Malware adalah istilah luas yang mengacu pada berbagai jenis program jahat seperti *trojan*, virus, worm dan rootkit. Klasifikasi *malware* bukanlah proses yang mudah. *Malware* hadir dalam berbagai kelas dan kategori yang biasanya dikategorikan berdasarkan proses propagasi dan tindakan yang dicapai pada sistem yang terinfeksi (Jerlin, 2015).

Dunia *Malware Analysis* telah menciptakan berbagai istilah untuk *malware* yang dapat mengindikasi kode *malware*, fitur atau fungsionalitas yang dapat membentuk *malware* lebih besar (Mohanta dan Saldanha, 2020).

Tabel 2.1 Jenis *Malware* (Mohanta dan Saldanha, 2020)

No	Nama *Malware*	Deskripsi
1	*Virus*	Virus adalah jenis *malware* pertama yang diketahui dapat mereplikasi dirinya sendiri. *Virus* juga dikenal dengan sebutan *Infector*. virus bertahan dengan cara menginfeksi program lain di sistem. Ketika *file* yang terinfeksi dijalankan maka virus juga dijalankan namun dibelakang layar.
2	*Worm*	*Worm* adalah *malware* yang berfungsi untuk menyebarkan infeksi ke komputer lainnya baik melalui jaringan maupun fisik seperti USB.
3	*Backdoor*	Backdoor adalah *entry point* tidak sah yang dimanfaatkan penyerang untuk memasuki sistem korban. Misalnya *malware* dapat membuka port jaringan terbuka pada sistem yang memiliki akses shell yang dapat diakses oleh penyerang untuk masuk ke dalam sistem.
4	*Trojan*	*Trojan* adalah *malware* yang menyamar sebagai perangkat lunak aman dan diinstal pada sistem korban dengan sepengetahuan pengguna namun pengguna tidak menyadari bahwa perangkat lunak tersebut merupakan sebuah *trojan*.
5	*Spywore/InfoStealer*	*Spyware* atau *InfoStealer* adalah *malware* yang memata-matai dan mencuri data sensitif dari sistem korban. Data yang ditargetkan oleh spyware dapat berupa nama pengguna, kata sandi, gambar dan dokumen.
6	*Keylogger*	*Keylogger* adalah *malware* yang mirip dengan *spyware* dapat mencatat penekanan tombol pengguna dan mengirim penekanan tombol yang direkam kembali ke penyerang.

Tabel 2.1 Jenis *Malware* (Mohanta dan Saldanha, 2020) (Lanjutan)

No	Nama *Malware*	Deskripsi
7	*Botnet*	*Botnet* adalah jaringan robot yang terdiri dari beberapa sistem yang terinfeksi *malware*. *Malware* yang membentuk *botnet* ini akan bekerja bersama sebagai kawanan dan bertindak atas perintah yang dikirimkan oleh penyerang dari server pusat. *Botnet* biasanya digunakan untuk serangan *denial-of-service* (DOS), mengirim pesan spam dan sebagainya.
8	*Remote Administration Tool (RAT)*	*Remote Administration Tool* (RAT) adalah *malware* yang dapat memberikan peretas kendali penuh atas sistem korban. *Malware* ini yang mirip dengan perangkat lunak yang biasa digunakan oleh teknisi untuk mengakses sistem jarak jauh untuk tujuan pemecahan masalah. Perbedaan RAT dengan perangkat lunak tersebut adalah RAT *malware* yang digunakan oleh penyerang untuk mengakses komputer tanpa otorisasi apapun.
9	*Adware*	*Adware* adalah jenis *malware* umum yang sering dijumpai pengguna komputer tetapi pengguna tidak menyadari hal tersebut. Adware disertakan dengan unduhan perangkat lunak dari situs web yang kurang dipercaya. Saat perangkat lunak diinstal, adware ikut terinstal di belakang layar tanpa sepengetahuan pengguna.
10	*Rootkit*	*Rootkit* adalah *malware* atau fungsionalitas *malware* yang digabungkan dengan *malware* lain yang bertujuan untuk menyembunyikan aktivitasnya sendiri atau aktivitas *malware* lain di sistem.

Tabel 2.1 Jenis M*alware* (Mohanta dan Saldanha, 2020) (Lanjutan)

No	Nama *Malware*	Deskripsi
11	*Banking Malware*	*Banking Malware* adalah *malware* yang bekerja dengan mencegat dan memodifikasi komunikasi browser untuk menangkap informasi tentang transaksi perbankan dan kredensial.
12	*Point-of-Sale*	*Malware* ini menginfeksi perangkat PoS yang digunakan sebagian besar ritel, gerai belanja, dan restoran di seluruh dunia. Fungsi utama dari *malware* PoS adalah untuk mencuri informasi kartu kredit dari perangkat lunak PoS.
13	*Ransomware*	*Ransomware* adalah *malware* yang menyandera data, *file* dan sumber daya sistem lainnya di sistem target. *Malware* ini biasanya akan meminta tebusan kepada korban sebagai tebusan untuk melepaskan data, *file* dan sumber daya yang disandera.
14	*Cryptominer*	*Cryptominer* adalah *malware* yang relatif baru dari keluarga *malware*. *Malware* ini menjadi populer saat meningkatnya penggunaan *cryptocurrency*. *Malware* ini jarang diketahui mencuri data dari sistem korban, tetapi memakan sumberdaya sistem dengan menambang *cryptocurrency*.
15	*Spammers*	*Spammers* mengirimkan email spam dari mesin korban. *Spam* bisa berisi tautan ke situs berbahaya. *Malware* dapat membaca kontak yang terhubung dengan email seperti *Microsoft Outlook* yang diinstal di sistem korban dan mengirimkan email ke kontak-kontak tersebut.

2.1.3 *Redline Stealer*

Redline Stealer adalah *malware family* yang secara signifikan mengancam keamanan dan privasi di era digital. *Malware* ini sangat berbahaya dan terkenal dengan kemampuan *malware* tersebut yang dapat menghindari deteksi *antivirus* dan mencuri informasi sensitif dari perangkat yang terinfeksi (Alrabaee dan Manna, 2021). *Readline Stealer* beroperasi dengan menginfeksi komputer korban dan mengumpulkan serta mengekstrak data berharga seperti kredensial *login*, informasi keuangan dan informasi detail pribadi lainnya (Saleous dkk., 2022).

Readline Stealer ditemukan pada awal Maret tahun 2020 oleh analis *ProofPoint*. *Malware Redline Stealer* tersedia di forum *underground* rusia untuk dijual dengan beberapa pilihan harga yaitu $100 untuk versi *lite* dan $150 untuk versi *pro*. Berikut ini adalah beberapa fitur yang terdapat pada *malware Redline Stealer* (Saleous dkk., 2022).

a. *Redline Stealer* dapat mengambil data *login, cookies, autocomplete fields* dan kartu kredit dari *browser*.
b. *Redline Stealer* dapat mengambil dari browser yang berbasis *Chromium* (*Chrome* dll) dan berbasis *Gecko* (*Mozilla* dll).
c. *Redline Stealer* memiliki kemampuan mengambil *file* dengan *path, extension* dan *search in subfolder* yang dapat disesuaikan. Kemampuan ini dapat dikonfigurasi untuk mencuri data pada memori perangkat dengan lokasi yang spesifik di sistem.

d. *Redline Stealer* dapat mengambil informasi tentang sistem korban diantaranya IP, negara, kota, nama pengguna, sistem operasi, *browser* yang terpasang dan spesifikasi perangkat keras PC.
e. *Redline Stealer* dapat mengunduh *file* dan menjalankan program berbahaya lainnya.

2.1.4 *Malware Analysis*

Malware dapat berisi kode perintah dan program berbahaya seperti *virus, trojan, worm, spyware* dan program berbahaya lainnya yang dapat mencuri data pengguna ataupun merusak komputer itu sendiri (Hazri, 2020). *Malware analysis* dilakukan untuk menemukan informasi yang diperlukan dalam menangani serangan *malware* tersebut dengan mengetahui apa saja yang terjadi di sistem, *file*, bagaimana *malware* tersebut bekerja dan jenis *malware* (Megira, Pangesti dan Wibowo, 2018).

Malware Analysis adalah usaha untuk mendapatkan informasi tentang *malware* menggunakan berbagai *metode,* teknik dan *tools* (Christopher, 2015). Metode untuk melakukan *malware analysis* secara mendasar dibedakan menjadi 2 metode yaitu *static* dan *dynamic. Static Analysis* adalah metode untuk mendapat informasi tentang *malware* tanpa menjalankan *malware* tersebut. *Dynamic Analysis* adalah metode untuk mendapatkan informasi tentang *malware* dengan cara menjalankan *malware* dan memantau perilaku *malware* tersebut (Honig dan Sikorski, 2012).

2.1.5 *Static Analysis*

Static Analysis adalah sebuah metode untuk menganalisis *malware* dengan cara menelusuri dan meneliti serta menganalisis kode sumber yang tertulis dalam program *malware* dengan melakukan pembedahan terhadap program *malware* tersebut (Cahyanto, Wahanggara dan Ramadana, 2017).

Static Analysis merupakan proses menganalisis *malware* tanpa menjalankan *malware* tersebut. *Static Analysis* tidak selalu mengungkapkan semua informasi yang diperlukan, tetapi pada beberapa kasus proses *static analysis* ini dapat memberikan petunjuk untuk menentukan upaya analisis apa yang akan dilakukan selanjutnya (Monnapa, 2018).

Static Analysis memiliki risiko yang rendah, tetapi proses ini memberikan hasil yang kurang menjanjikan karena pengumpulan informasi hanya didasarkan pada apa yang dapat dilihat saat *malware* tidak aktif. Informasi yang didapatkan dari proses ini sangat terbatas dan tidak dapat mengungkapkan banyak semua informasi tentang *malware* yang dianalisis (Christopher, 2015).

2.1.6 *Dynamic Analysis*

Dynamic Analysis adalah sebuah metode yang digunakan untuk menganalisis *malware* dengan cara menjalankan *malware* tersebut baik pada mesin fisik ataupun virtual untuk mengumpulkan informasi mengenai dampak dari *malware* tersebut terhadap komputer ketika *malware* dijalankan (Cahyanto, Wahanggara dan Ramadana, 2017).

Dynamic Analysis pada penelitian ini terdiri dari *process monitoring* dan *network monitoring*. *Process monitoring* adalah suatu teknik yang digunakan untuk memantau proses-proses yang dijalankan oleh suatu *malware,* pada umumnya aplikasi yang digunakan untuk *process monitoring* yaitu aplikasi *procmon* (Gunawan dan Ferriyan, 2017). *Network monitoring* adalah suatu teknik yang digunakan untuk memantau lalu lintas jaringan yang terjadi pada komputer yang terinfeksi *malware,* umumnya teknik ini dilakukan menggunakan aplikasi *Wireshark Network Analyzer*. Hasil *network monitoring* kemudian dilakukan analisis paket per paket setiap data yang lewat pada jaringan tersebut (Gunawan dan Ferriyan, 2017).

2.1.7 *Hybrid Analysis*

Hybrid Analysis merupakan penggabungan dari *static analysis* dan *dynamic analysis*. Metode ini dilakukan untuk menganalisis *malware* dengan cara memeriksa *signature* yang terdapat pada *malware* dan memantau perilaku (*behaviour*) *malware* terhadap sistem yang terinfeksi (Rusdi, Widiyasono dan Sulastri, 2019).

2.1.8 *Obfuscation*

Obfuscation adalah suatu teknik yang digunakan oleh pembuat *malware* untuk mengacak kode *malware* sehingga orang lain tidak dapat melihat kode *malware* tersebut (Gunawan dan Ferriyan, 2017). Pembuat *malware* juga menggunakan teknik *obfuscation* untuk menyembunyikan kode berbahaya agar

dapat melewati sistem deteksi *malware* atau biasa disebut *antivirus* (Singh dan Singh, 2018).

Metode *obfuscation* yang sering digunakan pada *malware* adalah mengubah urutan perintah pada kode program, memasukkan kode tidak berguna pada program, mengganti instruksi program dengan instruksi yang sama atau setara, mengubah nama *identifier* pada program dan *packing code*. Perubahan-perubahan yang dilakukan pada kode program tersebut tanpa mempengaruhi cara kerja program yang aslinya (Singh dan Singh, 2018).

2.1.9 *Decompile*

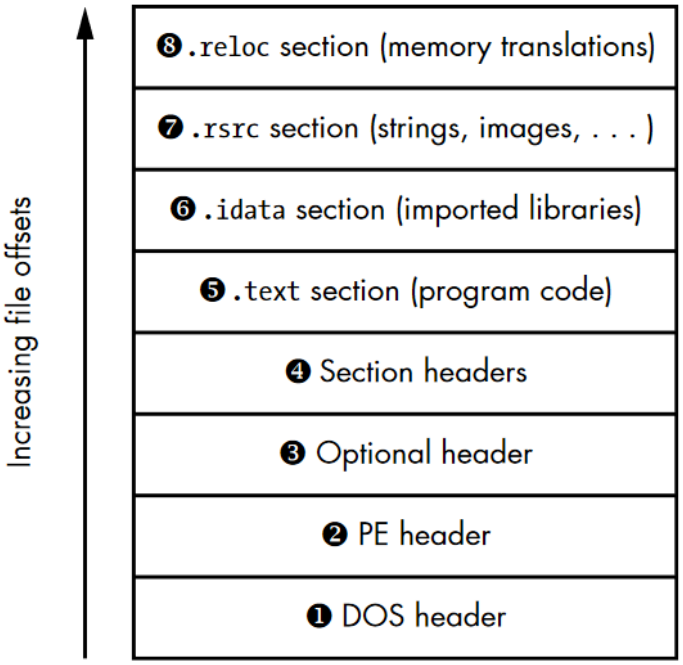
Decompile adalah proses mengubah *byte* kode mesin mentah menjadi *high level languages* yang dapat dibaca dengan mudah (Mohanta dan Saldanha, 2020). *Decompiler* adalah program yang dapat menerjemahkan kode mesin menjadi *high level languages* (Monnapa, 2018). Proses *decompile* sulit dilakukan jika *malware* menggunakan teknik *obfuscation* karena kode programnya tidak dapat dipahami oleh orang lain. Tujuan utama dari *obfuscation* adalah mempersulit proses analisis bahkan sampai tahap tidak mungkin untuk memahami kode program dengan teknik *decompile* (Christopher, 2015).

2.1.10 *Portable Executable*

File Portable Executable adalah arsitektur independen untuk sistem operasi *windows* 32bit dan 64 bit (Hahn dan Leipzig, 2014). *Portable Executable* menggambarkan struktur *file* program modern seperti *file* .exe, .dll, dan .sys dan

menentukan cara mereka menyimpan data. *File* PE berisi instruksi x86, data dan metadata yang dibutuhkan program agar dapat berjalan (Saxe dan Sanders, 2018).

File PE pada dasarnya dirancang untuk menjelaskan potongan *file* mana yang harus dimuat ke dalam memori, mana, menyediakan media atau sumber daya yang dibutuhkan oleh program yang sedang berjalan dan memberikan data keamanan seperti kode *signature* digital *windows* untuk memastikan program tersebut berasal dari sumber yang terpercaya (Saxe dan Sanders, 2018). Format PE melakukan tugasnya dengan memanfaatkan rangkaian konstruksi yang ditampilkan pada gambar 2.1. Format PE menyertakan serangkaian *header* yang memberi tahu sistem operasi cara membuat program ke dalam memori.



Gambar 2.1 Format *File* PE (Saxe dan Sanders, 2018)

a. *PE Headers*

PE Headers mendefinisikan atribut umum program seperti kode biner, gambar, data terkompresi dan atribut program lainnya. *PE Headers* juga

menentukan informasi arsitektur dari program apakah 32-bit atau 63-bit. PE Headers memberikan informasi dasar tetapi sangat berguna bagi analis *malware* (Saxe dan Sanders, 2018).

b. *Optional Headers*

Optional Headers menentukan *entry point* program dalam *file* PE, yang mengacu pada instruksi pertama yang dijalankan program setelah dimuat. Optional Headers juga menentukan ukuran data yang dimuat *Windows* ke dalam memori saat memuat *file* PE, *Windows* subsystem, target program dan detail lainnya terkait program. Informasi pada Headers ini terbukti sangat berharga bagi para *reverse engineers,* karena dengan *entry point* program dapat menentukan untuk di mana memulai proses *reverse engineering* (Saxe dan Sanders, 2018).

c. *Section Headers*

Section Headers menjelaskan bagian data yang terdapat dalam *file* PE. Sebuah *Section* pada *file* PE adalah potongan data yang akan dipetakan ke dalam memori ketika sistem operasi memuat program atau akan berisi instruksi tentang bagaimana program harus dimuat ke dalam memori.

Section Headers memberi tahu *Windows* izin apa yang harus diberikan pada *sections*, seperti dibaca, ditulis atau dijalankan oleh program saat dijalankan (Saxe dan Sanders, 2018).

d. *.text Section*

Setiap program PE berisi setidaknya satu *section* dari kode x86 yang ditandai dengan *executable* pada *sections headers*, setiap *sections* ini hampir selalu diberi nama *.text* (Saxe dan Sanders, 2018).

e. *.idata Section*

Section ini biasa disebut juga dengan *imports* berisi *Imports Address Table* (IAT), yang mencantumkan *dynamically linked libraries* dan berbagai *functions* miliknya. IAT merupakan struktur PE yang termasuk penting untuk dibedah saat pemeriksaan biner PE karena dapat mengungkapkan panggilan *library* apa yang dibuat oleh program (Saxe dan Sanders, 2018).

f. *Data Section*

Bagian data dalam *file* PE dapat menyertakan bagian seperti *rsrc*, .data dan *rdata*, yang menyimpan item seperti gambar kursor mouse, skin tombol, audio dan media lain yang digunakan oleh suatu program. Informasi di *section rsrc* sangat penting untuk proses analisis *malware* karena dengan memeriksa *string* karakter yang dapat dicetak, gambar dan aset lain dalam *file* PE dapat memperoleh petunjuk penting tentang fungsionalitas *file* (Saxe dan Sanders, 2018).

g. *Reloc Section*

Kode biner PE tidak *position independent*, yang artinya kode tidak dapat berjalan ketika dipindahkan ke lokasi yang lain. *Section reloc* mencegah eksekusi kode terganggu saat dipindahkan dengan cara mengirim

informasi yang sudah ditransmisikan kepada sistem operasi *windows* jika kode sudah dipindahkan (Saxe dan Sanders, 2018).

2.1.11 *Strings*

String adalah rangkaian karakter *ASCII* dan *Unicode-Printable* yang ditanamkan dalam suatu *file*. *String* yang terdapat pada *malware* dapat memberi petunjuk tentang fungsionalitas program dan indikator terkait dengan biner yang dicurigai. *String* dalam sebuah program adalah *value* yang akan dimuat oleh program saat dieksekusi (Megira, Pangesti dan Wibowo, 2018). *String* yang diekstraksi dari *malware* dapat berisi petunjuk yang mengarah pada nama *file*, *URL*, nama domain, alamat IP, perintah serangan, kunci *registry* dan sebagainya (Monnapa, 2018).

2.2 Penelitian Terkait

Tabel 2.2 Literatur Review

No	Peneliti	Judul	Metode	*State of The Art*
1.	Triawan Adi Cahyanto, Victor Wahanggara, Darmawan Ramadana	Analisis dan Deteksi *Malware* Menggunakan Metode *Malware* Analisis Dinamis dan *Malware* Analisis Statis	*Static Analysis* dan *Dynamic Analysis*	Penelitian ini melakukan analisis terhadap *malware* poison ivy menggunakan metode *dynamic analysis* dan *Static Analysis*. Berdasarkan hasil analisis tersebut *malware* Poison Ivy melakukan perubahan pada *Windows Registry* dan *file prefetch*.
2.	Tesa Pajar Setia, Nur Widiyasono, Aldy Putra Aldya	Analisis *Malware Flawed Ammyy RAT* Dengan Metode *Reverse engineering*	*Dynamic Analysis* dan *Reverse engineering*	Penelitian menunjukkan bahwa *malware Flawed Ammyy RAT* bersembunyi pada aplikasi *Ammyy Admin* kemudian melakukan koneksi ke *server,* selanjutnya *attacker* dapat melakukan *remote.*
3.	S Megira, A R Pangesti, F W Wibowo,	*Malware Analysis* and Detection Using *Reverse Engineering Technique*	*Reverse engineering*	Penelitian ini menunjukkan bahwa penggabungan metode statis dan dinamis akan memberikan hasil yang optimal. Setiap malware memiliki karakter yang berbeda, maka dari itu peran *malware* analis penting untuk mencari cara yang benar dalam menghadapi *malware*.
4.	Virgiawan A. Manoppo, Arie S. M. Lumenta, Stanley D. S. Karouw	Analisa *Malware* Menggunakan Metode *Dynamic Analysis* Pada Jaringan Universitas Sam Ratulangi	*Dynamic Analysis*	Metode dinamis dapat mempermudah proses analisis *malware* di lingkungan yang terisolasi. Analisis menggunakan *cuckoo sandbox* dapat mengungkap informasi *malware,* karakteristik dan tingkat bahaya yang ditimbulkan *malware.*
5.	Aldy Putra Aldya , Nur Widiyasono, Tesa Pajar Setia	*Reverse engineering* untuk Analisis *Malware* Remote Access *Trojan*	*Reverse engineering*	Metode *reverse engineering* berhasil mengungkap 12 *function* yang terdapat pada *sample malware flawed ammy RAT*. Teknik *string* analisis dapat memberikan petunjuk untuk proses *disassembly*.

Tabel 2.2 Literatur Review (Lanjutan)

No	Peneliti	Judul	Metode	*State of The Art*
6.	Indra Gunawan dan Andrey Ferriyan	Analisis *Malware Botnet* Proteus Pendekatan *Static* dan *Dynamic*	*Static Analysis Dynamic Analysis*	Penelitian ini menunjukkan bahwa *botnet* menggunakan fitur-fitur *Obfuscated, Cryptology* berbasis .net, memiliki Command Center, memodifikasi *registry*, memodifikasi *file* sistem *windows*, hanya aktif jika terdapat koneksi internet, tidak melakukan perusakan terhadap *file-file* yang ada. *Botnet proteus* dapat dikategorikan sebagai *malware* yang lebih bersifat sebagai *malware* pencuri data serta *malware* pembuat dan pengontrol *zombie computer*.
7.	M. Hazri	Analisis *Malware PlasmaRAT* dengan Metode *Reverse engineering*	*Reverse engineering*	*Dynamic analysis* menggunakan *virtual machine* online menunjukkan *file imports* dan *functions* yang ada di dalam file tersebut dan menunjukkan adanya program lain yang ikut dijalankan pada sistem ketika *malware* dijalankan. Teknik *reverse engineering* tidak dapat dilakukan karena *malware* menggunakan *obfuscation*.
8.	Asep Solahudin Rusdi, Nur Widiyasono dan Heni Sulastri	Analisis Infeksi Malware Pada Perangkat Android Dengan Metode Hybrid Analysis	*Hybrid Analysis*	Hasil dari penelitian ini adalah *malware Judy* memiliki karakteristik antara lain menyamar sebagai aplikasi permainan, melakukan klik iklan tanpa sepengetahuan pengguna yang dilakukan secara background (ad-fraud), serta mendapatkan informasi perangkat lewat ad-tracking. *Malware Marcher* memiliki karakteristik diantaranya menyembunyikan diri lalu mengambil data penting dan dikirimkan ke server.
9.	Syarif Yusirwan, Yudi Prayudi dan Imam Riadi	*Implementation of Malware Analysis using Static and Dynamic Analysis Method*	*Static analysis* dan *Dynamic analysis*	Berdasarkan penelitian ini, penggabungan dua metode analisis malware yaitu analisis statis dan analisis dinamis mampu memberikan gambaran yang lebih lengkap mengenai karakteristik malware *TT.exe*.

Tabel 2.2 Literatur Review (Lanjutan)

No	Peneliti	Judul	Metode	*State of The Art*
10.	M Hazri	Analisis Malware PlasmaRAT dengan Metode Reverse Engineering	*Reverse Engineering* dan *Dynamic Analysis*	Penelitian ini menggunakan teknik *dynamic analysis* dan *reverse engineering* dimulai dengan mendapatkan informasi (*information gathering*) dari *tool virus total*, proses *dynamic analysis* menggunakan *virtual machine online* (*sandbox online*) *hybrid analysis* dan proses *reverse engineering* menggunakan *tool CFF Explorer*. *Dynamic Analysis* menunjukkan ada program lain yang ikut berjalan pada sistem ketika malware dijalankan. *Reverse Engineering* tidak menghasilkan informasi karena file *malware* menggunakan *obfuscation.*
11.	Avie Triantoro, Nur Widiyasono dan Rohmat Gunawan	*Hack.exe Malware Analysis and Investigation Using Memory Forensics*	*Static Analysis*	Cara kerja *malware Hack.exe* adalah dengan menginfeksi sistem, membaca data dan mengirim data yang diinginkan ke server *malware* dengan alamat ip 24.146.133.195.

2.3 Matriks Penelitian

Tabel 2.3 Matriks Penelitian

No	Penulis	Judul	Teknik Analisis								*Malware Family*										
			Hashing	*Strings Extract*	*Disassembly*	*Obfuscated Detect*	*Decompile*	*Process Explorer*	*Process Monitoring*	*Network Monitoring*	*Debugging*	*Poison Ivy RAT*	*Flawed Ammyy RAT*	*Remcos*	*SizMLXjs*	*Botnet Proteus*	*Judy* dan *Marcher*	*PlasmaRAT*	*TT.exe*	*Njrat*	*Zloader*	*Redline Stealer*
1.	(Cahyanto, Wahanggara dan Ramadana, 2017)	Analisis dan Deteksi *Malware* Menggunakan Metode *Malware* Analisis Dinamis dan *Malware* Analisis Statis		√	√					√		√										
2.	(Pajar Setia, Widiyasono dan Putra Aldya, 2018)	Analisis *Malware Flawed Ammyy RAT* Dengan Metode *Reverse engineering*		√	√			√		√			√									
3.	(Megira, Pangesti dan Wibowo, 2018)	*Malware Analysis and Detection Using Reverse Engineering Technique*	√	√	√			√	√	√	√			√								

Tabel 2.2 Matriks Penelitian (Lanjutan)

No	Penulis	Judul	Teknik Analisis									*Malware Family*										
			Hashing	*Strings Extract*	*Disassembly*	*Obfuscated Detect*	*Decompile*	*Process Explorer*	*Process Monitoring*	*Network Monitoring*	*Debugging*	*Poison Ivy RAT*	*Flawed Ammyy RAT*	*Remcos*	*SizMLXjs*	*Botnet Proteus*	*Judy* dan *Marcher*	*PlasmaRAT*	*TT.exe*	*Njrat*	*Zloader*	*Redline Stealer*
4.	(Virgiawan, Lumenta dan Karouw, 2020)	Analisa *Malware* Menggunakan Metode *Dynamic Analysis* Pada Jaringan Universitas Sam Ratulangi	√	√						√					√							
5.	(Setia, Aldya dan Widiyasono, 2019)	*Reverse engineering* untuk Analisis *Malware* Remote Access *Trojan*	√	√	√						√		√									
6.	(Gunawan dan Ferriyan, 2017)	Analisis *Malware Botnet* Proteus Pendekatan *Static* dan Dinamic		√	√	√		√	√	√						√						
7.	(Hazri, 2020)	Analisis *Malware PlasmaRAT* dengan Metode *Reverse engineering*			√	√			√				√					√				

Tabel 2.2 Matriks Penelitian (Lanjutan)

No	Penulis	Judul	Teknik Analisis									*Malware Family*										
			Hashing	*Strings Extract*	*Disassembly*	*Obfuscated Detect*	*Decompile*	*Process Explorer*	*Process Monitoring*	*Network Monitoring*	*Debugging*	*Poison Ivy RAT*	*Flawed Ammyy RAT*	*Remcos*	*SizMLXjs*	*Botnet Proteus*	*Judy* dan *Marcher*	*PlasmaRAT*	*TT.exe*	*Njrat*	*Zloader*	*Redline Stealer*
8.	(Rusdi, Widiyasono dan Sulastri, 2019)	Analisis Infeksi *Malware* Pada Perangkat *Android* Dengan Metode *Hybrid Analysis*	√		√		√		√	√							√					
9.	(Yusirwan, Prayudi dan Riadi, 2015)	*Implementation of Malware Analysis using Static and Dynamic Analysis Method*			√	√		√	√	√	√								√			
10.	(Septani, Widiyasono dan Mubarok, 2016)	Investigasi Serangan *Malware Njrat* Pada PC							√	√										√		
11.	(Triantoro, Widiyasono dan Gunawan, 2021)	*Hack.exe Malware Analysis and Investigation Using Memory Forensics*	√		√				√	√											√	

Tabel 2.2 Matriks Penelitian (Lanjutan)

No	Penulis	Judul	Teknik Analisis								*Malware Family*										
			Hashing	*Strings Extract*	*Disassembly*	*Obfuscated Detect*	*Decompile*	*Process Explorer*	*Process Monitoring*	*Network Monitoring*	*Debugging*	*Poison Ivy RAT*	*Flawed Ammyy RAT*	*Remcos*	*SizMLXjs*	*Botnet Proteus*	*Judy* dan *Marcher*	*PlasmaRAT*	*TT.exe*	*Njrat*	*Zloader*	*Redline Stealer*
12.	(Sinjaya, 2023)	Analisis *Malware Redline Stealer* Menggunakan Metode *Static Analysis* dan *Dynamic Analysis*	✓	✓		✓	✓		✓	✓												✓