BAB II

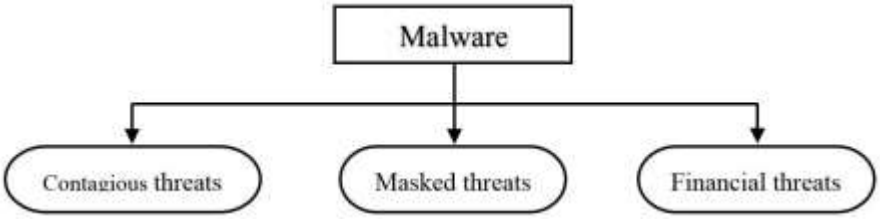
LANDASAN TEORI

2.1 *Malware*

Malware adalah perangkat lunak berbahaya yang diprogram untuk merusak atau untuk mendapatkan akses ke sistem komputer tanpa sepengetahuan pemilik sistem. *Virus, Worm, Trojan, Key Logger, Spyware* dan *Ransomware* adalah contoh kebanyakan maleware yang digunakan. Istilah seperti *"worm", "virus", "Trojan Horse"* digunakan untuk klasifikasi *malware* yang menunjukan perilaku berbahaya yang serupa (Zalavadiya & Priyanka, 2017).

2.2 Klasifikasi *malware*

Malware dapat diklasifikasi dalam berbagai kelas dan kategori yang umumnya dikategorikan menurut proses dan reaksi tergantung pada perancangan dan pengembangan *malware*. Gambar 2.1 menunjukan berbagai jenis *malware* (Zalavadiya & Priyanka, 2017).



Gambar 2.1 klasifikasi *Malware*

Sumber: (Zalavadiya & Priyanka, 2017)

a. *Contagious Threats* (Ancaman yang Menular)

Malware virus dan *worm* merupakan kategori pada ancaman yang menular, ini dilihat dari karakteristik dan cara kerja malware melakukan infeksi pada system yang ditunjukkan pada tabel 2.1.

Tabel 2.1 *contagious threats* klasifikasi dan deskripsi (Zalavadiya & Priyanka, 2017)

Malware	Karakteristik	Cara kerja	Kerusakan
Virus	*Malware* yang mengambil alih kontrol yang tidak sah dari komputer yang terinfeksi dan menyebabkan kerusakan tanpa sepengetahuan pengguna	Program virus yang tersembunyi dalam program tidak berbahaya lainnya seperti file yang dapat dieksekusi dan kemampuan mereplikasi dirinya ke dalam program lain dan menyebarkan infeksi dari satu komputer ke komputer lain	Penurunan kinerja dan menyebabka DOS (*Denial of Service*)
Worm	Worm adalah perangkat lunak berbahaya yang berdiri sendiri yang dapat beroprasi secara independen dan tidak mengaitkan dirinya untuk menyebarluaskan	Worm memanfaatkan kerentanan keamanan dengan menggunakan computer atau jaringan dan menyebarluaskan diri melalui perangkat penyimpanan seperti USB, media komunikasi perangkat seperti Email	Melakukan komunikasi sejumlah besar memori sumber daya sistem dan masalah kinerja jaringan

b. *Masked Threats* (Ancaman Pertopeng)

Malware trojan, backdoor, adware, dan *rootkits* merupakan kategori pada ancaman pertopeng, ini dilihat dari karakteristik dan cara kerja malware melakukan infeksi pada system yang ditunjukkan pada tabel 2.2

Tabel 2.2 *Masked threats* klasifikasi dan deskripsi (Zalavadiya & Priyanka, 2017)

Malware	Karakteristik	Cara kerja	Kerusakan
Trojan	*Malware* berbahaya yang tersembunyi dan berperilaku sebagai program yang sah untuk mengambil alih kendali computer atau sistem secara tidak sah	*Trojan* tidak mereplikasi diri sebagai gantinya mengunduh atau menyalin melalui interaksi pengguna seperti *download file* dari internet atau perangkat lain	Mencuri kata sandi atau rincian login, pencuri uang digital, memodifikasi atau menghapus file, memonitor aktivitas pengguna
Backdoors	Melakukan bypass control keamanan normal dan memberikan peyerang ke akses yang tidak sah	Dipasang melalui program atau aktivitas berbahaya lainnya	Melakukan modifikasi dan menghapus *file* system dan memonitor aktivitas system
Adware	Memberikan informasi kepada pelaku iklan tentang kebiasaan penjelajahan pengguna, sehingga memungkinkan pelaku iklan untuk memberikan add yang ditargetkan	*Adware* menyebar melaui situs *web*	*Clickjacking*, *phising* atau membuat aktivitas jahat menggunakan *browser*
Rootkits	*Rootkits* adalah teknik *masking* untuk *malware* yang pada dasarnya dirancang untuk maksud jahat dari program ini	Dapat diinstal melalui eksploitasi perangkat lunak atau *Trojan*	Mencuri kata sandi atau melakukan instal *keylogger*

c. *Financial Threats* (Ancaman Keuangan)

Malware ransomware, spyware, dan *keyloger* merupakan kategori pada ancaman keuangan, ini dilihat dari karakteristik dan cara kerja malware melakukan infeksi pada system yang ditunjukkan pada tabel 2.3.

Tabel 2.3 *financial threats* klasifikasi dan deskripsi (Zalavadiya & Priyanka, 2017)

Malware	Karakteristik	Cara kerja	Kerusakan
Ransomware	*Ransomware* adalah perangkat lunak yang dirancang untuk memblokir akses ke sistem komputer hingga sejumlah uang dibayarkan	*Ransomware* menyebar dan disalurkan melalui rekayasa social dan interaksi pengguna, membuka lampiran *email* berbahya yang melakukan klik tautan berbahaya dalam *email* atau di situs jejaring sosial	*Ransomware* adalah *malware* untuk pencurian data melakukan enkripsi data korban dan membatasi pengguna untuk melakukan askses system penyerang
Spyware	*Spyware* melacak aktivitas pengguna tanpa sepengetahuannya pengguna dan mengirim kembali informasi sesintif kepada penyerang	Dapat diinstal dengan perangkat lunak lain seperti *freeware* atau dijatuhkan oleh *Trojan*	*Sniffing* antarmuka jaringan sertifikasi digital, kunci enkripsi dan informasi sensitive lainnya
Keylogger	*Keylogger* diam-diam merekam *keystrokes*	Dapat diinstal oleh program jahat lain atau keteika seorang pengguna mengujungi	Menangkap infromasi *sensitive* seperti nama pengguna kata sandi nomor kartu kredit atau rincian perbankan *online*

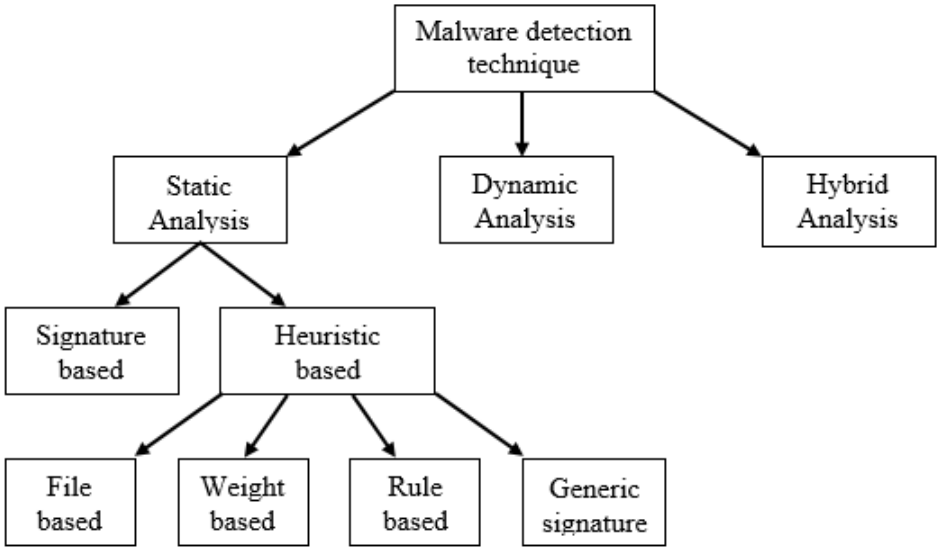
2.3 Flawed ammyy RAT

Malware Flawed ammyy dibangun diatas kode sumber bocor untuk versi 3 dari Ammyy Admin, bentuk sah dari perangkat lunak desktop jarak jauh yang digunakan di antara jutaan konsumen dan bisnis untuk menangani *remote control* dan *diagnosis* pada *platfom Windows*, ini bukan pertama kalinya *Ammyy Admin* disalahgunakan,

serangan Juli 2016 juga menggunakannya untuk menyembunyikan *malware* (Sheridan, 2018).

2.4 Analisis *malware*

Analisis *malware* merupakan dasar untuk mendapatkan informasi dalam rangka mengatasi serangan dalam system korban. Informasi tersebut, dapat dikembangkang signature untuk melakukan deteksi infeksi *malware*. Tujuan akhir dari analisis *malware* adalah menggambarkan secara tepat cara kerja sebuah *malware* (Adenansi & Novarina, 2017).



Gambar 2.2 representasi *hierarchal* berbagai teknik deteksi *malware*

Sumber: (Uppal , Mehra, & Verma, 2014)

Merujuk dari gambar 2.3 ini menunjukkan *hierarchal* teknik deteksi *malware*, ada tiga teknik yang dapat dilakukan, analisis statis, dinamis dan hybrid. Penjelasan lebih jelas mengenai teknik deteksi *malware* sebagai berikut:

a. Analisis Statis

Analisis statis adalah prosedur melakukan analisis perangkat lunak tanpa mengeksekusinya. Selama analisis statis aplikasi dipecah dengan menggunakan alat teknik rekayasa balik, sehingga membangun kembali kode sumber dan algoritma yang telah dibuat oleh aplikasi. Analisis statis dapat dilakukan melalui program analisis, debugger dan disassembler. Berbagai teknik analisis statis adalah sebagai berikut (Uppal , Mehra, & Verma, 2014):

1. Teknik pendeteksian berdasarkan *signature*

Teknik ini juga dikenal sebagai teknik pencocokan pola atau string atau topeng atau sidik jari. *Signature* dalah urutan program yang disuntikan dalam aplikasi oleh pembuat *malware*, yang secara unik mengindetifikasi *malware* tertentu. Mendeteksi *malware* dalam kode, pencarian detektor *malware* untuk signature yang telah ditentukan sebelumnya dalam kode. (Uppal , Mehra, & Verma, 2014)

2. Teknik pendeteksian *heuristic*

Teknik ini juga dikenal sebagai teknik proaktif. Teknik ini mirip dengan teknik berdasarkan *signature* tertentu dalam kode, detektor *malware* sekarang mencari perintah atau intruksi yang tidak ada dalam program aplikasi. Hasilnya adalah disini menjadi mudah untuk mendeteksi varian *malware* baru yang belum ditemukan. Teknik analisis heuristik yang berbeda adalah sebagai berikut (Uppal , Mehra, & Verma, 2014):

a) *File based heuristic analysis*

Analisis heuristik berdasarkan file juga dikenal sebagai analisis file. Teknik ini file dianalisis secara mendalam seperti isi, tujuan, pengerjaan file, jika file berisi perintah untuk menghapus atau merusak file lain, maka dianggap sebagai file berbahaya (Uppal , Mehra, & Verma, 2014)

b) *Weight based heuristic analysis*

Analisis heuristik berdasarkan *weight* adalah teknik kuno. Setiap aplikasi diberi bobot sesuai dengan bahaya yang mungkin dimilikinya, jika nilai tertimbang melebihi nilai ambang batas yang ditemukan, maka aplikasi tersebut berisi kode bahaya. (Uppal , Mehra, & Verma, 2014).

c) *Rule based heuristic analysis*

Analisis disini melakukan ekstrasi aturan yang mengidentifikasikan aplikasi. Aturan-aturan ini kemudian dicocokkan dengan aturan yang ditetapkan sebelumnya, jika aturan tidak cocok, maka aplikasi tersebut tedapat *malware* (Uppal , Mehra, & Verma, 2014).

d) *Generic signature analysis*

Varian *malware* berarti, *malware* itu berbeda dalam perilakunya tetapi memiliki keluarga yang sama seperti “kembar identik”. Teknik ini menggunakan definisi antivirus yang ditetapkan sebelumnya, untuk menemukan varian baru *malware* (Uppal , Mehra, & Verma, 2014)

Keuntungan dari analisis statis

Analisis statsi cepat dan aman, juga mengumpulkan struktur kode program dibawah pemeriksaan. jika analisis statis dapat menghitung prilaku untuk mekanisme keamanan dimasa mendatang. (Uppal , Mehra, & Verma, 2014)

Kerugian dari analisis statis

Analisis statis tidak mengambil posisi untuk melakukan analisa *malware* yang tidak diketahui. Kode sumber dari banyak aplikasi tidak mudah tersedia, untuk melakukan analisis statis, para investigator harus memiliki pemahan yang mendalam tentang fungsi system operasi (Uppal , Mehra, & Verma, 2014)

b. Analisis Dinamis

Proses melakukan analisis perilaku atau tindakan yang dilakukan oleh aplikasi saat melakukan eksekusi disebut analisis dinamis. Analisis dinamis dapat dilakukan melalui pemantauan fungsi paanggilan, pelacakan informasi, melakukan analisis parameter fungsi dan menelusuri instruksi. Umumnya mesin virtual atau sandbox diunakan untuk analisis ini, aplikasi yang diragukan biasanya dijalankan ke dalam lingkungan virtual, jika aplikasi berperilaku tidak biasa itu dikategorikan sebagai jahat. Perangkat lunak perilaku pemblokiran, yang melakukan blokir tindakan jahat dari program sebelum serangan *malware* (Uppal , Mehra, & Verma, 2014)

Keuntungan dari analisis dinamis

Seseorang dapat dengan mudah mendeteksi *malware* yang tidak diketahui dengan hanya melakukan analisis prilaku aplikasi. (Uppal , Mehra, & Verma, 2014)

Kerugian dari analisis dinamis

Analisis ini membutuhkan waktu sebagai waktu pelaksanaan aplikasi, sehingga dalam beberapa kasus tidak cepat, tidak aman. Analisis ini tidak berlaku untuk aplikasi yang menunjukan perbedaan perubahan perilaku dengan kondisi pemicu yang berbeda. Singkatnya, gagal mendeteksi multipath *malware* (Uppal , Mehra, & Verma, 2014).

c. Analisis Hybrid

Teknik ini adalah kombinasi dari analisis statis dan analisis dinamis. Prosedur yang mengikutinya pertama kali memeriksa signature *malware*, jika ada dalam kode dan kemudian memonitor perilaku kode. Teknik ini menggabungkan keuntungan dari kedua teknik diatas (Uppal , Mehra, & Verma, 2014).

2.5 *Reverse Engineering*

Pengertian secara umum *Reverse Engineering* adalah proses dari ekstraksi pengetahuan atau blue-print design dari apapun yang telah dibuat manusia. Konsep *Reverse Engineering* sudah ada sejak sebelum teknologi modern atau komputer dibuat. *Reverse Engineering* biasa digunakan oleh industri untuk mengetahui suatu informasi dari proses, design, atau ide yang dibuat sebelumnya namun memiliki sedikit informasi untuk dikembangkan lebih lanjut (Nugroho & Prayudi, 2015).

Dunia tekologi informasi *Reverse Engineering* berhubungan erat dengan Software atau Aplikasi, istilah lain yang dikenal dari *Reverse Engineering* adalah Reverse Code Engineering. Proses *Reverse Engineering* pada sebuah software atau aplikasi dapat dilakukan dengan cara: (Nugroho & Prayudi, 2015)

a. *Assembly*.

Assembly language merupakan bahasa pemrograman yang berada pada level rendah dari beberapa bahasa pemrograman yang dikenal selama ini. Bahasa *assembly* digunakan untuk sebuah mesin karena mesin tidak dapat mengenal bahasa pemrograman tingkat tinggi seperti *java*, *basic*, *pascal*, dan lain-lain. (Nugroho & Prayudi, 2015)

b. *Disassembly.*

Disassembly merupakan kebalikan dari proses *assembly*. Proses *disassembly* digunakan dalam teknik *Reverse Engineering* untuk menerjemahkan dari bahasa mesin ke bahasa yang mudah dimengerti manusia, yaitu bahasa *assembly*. (Nugroho & Prayudi, 2015)

c. *Debugging.*

Proses *debugging* adalah proses pengujian dari software. Pada analisa *malware debugging* digunakan untuk melakukan pengujian dari setiap proses inti yang ada didalam *malware*. Proses pertama yang dilakukan dalam melakukan debugging adalah lagi *sample malware* kedalam *ollydbg* dan kemudian dijalankan mengikuti proses dari analisa sebelumnya. (Nugroho & Prayudi, 2015)

d. *X86 Arsitektur.*

Arsitektur x86 memiliki tiga komponen keras yaitu *CPU*, *RAM*, *Input / Output* (I/O). pada dasarnya pada internal dari kebanyakan arsitektur komputer modern yang termasuk juga x86 mengikuti arsitektur *Von Neumann*. (Nugroho & Prayudi, 2015).

e. *Instruction.*

Instruksi adalah konstruksi yang dibangun dari program *assembly*. A*ssembly* x86 instruksi terdiri dari nemonic dan nol atau lebih operands. (Nugroho & Prayudi, 2015)

f. *Hashing*.

Hash merupakan identitas dari sebuah program seperti halnya sidik jari pada manusia. Proses hash dilakukan untuk verifikasi sebelum dan setelah proses analisa *malware*. Verifikasi tersebut dilakukan untuk mengetahui tidak adanya perubahan haMsh pada *sample malware* setelah dilakukan proses analisis. (Nugroho & Prayudi, 2015)

g. *String Analysis.*

String atau karakter dalam sebuah program seperti (.,A-) merupakan nilai yang akan dilakukan proses load oleh *sample malware* ketika dieksekusi. Hal ini yang menjadikan dalam proses *Reverse Engineering* harus dilakukan string analisis untuk mendapatkan bukti kuat dari *sample malware*. (Nugroho & Prayudi, 2015)

h. *MAER (Malware Analysis Environment and Requirement).*

MAER adalah ruang lingkup yang menjadi laboratorium analisis *malware*. MAER merupakan salah satu penentu seorang analis *malware* mendapatkan infromasi yang akurat dan efisien dari analisa yang dilakukan. (Nugroho & Prayudi, 2015)

i. *Repository Malware.*

Repository malware merupakan tempat disimpannya *sample malware* yang telah berhasil melakukan serangan kedalam sistem komputer diamanapun. *Repository malware* dibuat untuk memberikan *sample* kepada seorang *malware* analis untuk melakukan analisa terhadap *malware* yang sudah berhasil melakukan serangan (virusshare). *Repository Malware* adalah salah satu dari *malware source* yang dapat digunakan untuk kepentingan analisis. Selain menggunakan *Repository*, *sample* untuk analisis dapat pula berasal dari *honeypot* yang terpasang atau berdasarkan kasus yang dialami sendiri. Terdapat beberapa acuan untuk kepentingan *sample* analisis *malware* yaitu: *Virusshare, Contagio Malware Dump, Malshare, Malware.lu, Malware Blacklist, MD Pro, Open Malware*. (Nugroho & Prayudi, 2015)

2.6 Kajian Penelitian Sebelumnya

Berikut merupakan *Literature reviews* dari penelitian sebelumnya yang bersangkutan pada *focus Malware* ditunjukan pada tabel 2.4.

Tabel 2.4 Penelitian Terkait

No	Peneliti	Metode / Framework	Domain Penelitian	Hasil Penelitian
1	Devi Rizky Septani Nur Widiyasono Husni Mubarok (2016)	Metode Analisis Dinamis	Investigasi Serangan *Malware Njrat* Pada PC	Mengetahui cara kerja *malware Njrat*
2	Egi Satria Rusman Nur Widiyasono Aldy Putra Aldya (2017)	Metode Analisis Dinamis	*Reverse Enginering Malware Ransomware* Manggunakan Analisis Dinamis	Mengetahui aktifitas *malware Ransomware Petya* dan cara pencegahan serangan *malware Ransomware petya*

Tabel 2.4 Penelitian Terkait (lanjutan)

No	Peneliti	Metode / Framework	Domain Penelitian	Hasil Penelitian
3	Abdul Haris Muhammad Bambang Sugiantoro Ahmad Lutfi (2017)	Metode Analisis Statis dan Analisis Dinamis	Metode Klasifikasi dan Analisis Karakteristik *Malware* Menggunakan Konsep Ontologi	Penerapan ontologi sebagai knowledge base dasar dalam melakukan analisis karakteristik *malware* sebagai knowledge base sangat dibutuhkan dalam melakukan analisis karakteristik *malware*
4	Sabam Chandra Yohanes Hutauruk Fazmah Arif Yulianto Gandeva Bayu Satrya (2016)	Metode Analisis Statis dan Analisis Dinamis	*Malware* Analysis Pada *Windows Operating System* Untuk Mendeteksi *Trojan*	Data karakteristik trojan, yang dapat digunakan sebagai indikator untuk menganalisa trojan berdasarkan behaviornya
5	Retno Adenansi Lia A. Novarina (2017)	Metode Analisis Dinamis	*Malware Dynamic*	Membahas mengenai cara melakukan analisis *malware* dengan metode analisis dinamis untuk deteksi *malware*
6	Triawan Adi Cahyanto Victor Wahanggara Darmawan Ramadana (2017)	Metode Analisis Statis dan Analisis Dinamis	Analisis dan Deteksi *Malware* Menggunakan Metode *Malware* Analisis Dinamis dan *Malware* Analisis Statis	Tentang cara kerja *malware* (poison ivy), dapat melakukan proses login secara remote tanpa diketahui oleh pemilik komputer.

Tabel 2.4 Penelitian Terkait (lanjutan)

No	Peneliti	Metode / Framework	Domain Penelitian	Hasil Penelitian
7	Heru Ari Nugroho Yudi prayudi (2015)	*Reverse Engineering*	Penggunaan teknik *Reverse Engineering* pada *malware* analisis untuk indentifikasi serangan *malware*	Hasil yang didapat dari proses *Reverse Engineering* pada *malware* biscuit adalah gambaran bagaimana cara kerja dari *malware* tersebut.
8	Nayan zalavadiya Priyanka Sharman (2017)	Metode analisis statis dan analisis dinamis	*A Metodology of malware Analysis, tools, and Technique for windows platform – RAT analysis*	Menguraikan metodologi yang efektif dan efisien yang dapat diterapkan untuk meningkatkan kinerja deteksi dan penghapusan *malware* yang dikumpulkan. Analisis dinamis cara terbaik untuk melakkan analisis *sample malware*
9	Ujaliben Kalpesh Bavishi Bhavesh Madnlal jain (2017)	Metode analisis statis dan analisis dinamis	*Malware* analysis	Menjelaskan mengenai teknik dan tools untuk analisis dinamis dan analisis statis
10	Dolly Uppal Vishakha Mehra Vinod verma (2014)	Metode analisis statis dan analisis dinamis	*Basic on Malware Analysis, tools and Technique*	Berfokus pada studi dasar *malware* dan berbagai deteksi teknik yang dapat digunakan untuk deteksi *malware*

Tabel 2.4 Penelitian Terkait (lanjutan)

No	Peneliti	Metode / Framework	Domain Penelitian	Hasil Penelitian
11	Syarif Yusirwan S Yudi Prayudi Imam Riadi	Metode analisis statis dan analisis dinamis	*Implementation of Malware Analysis using Static and Dynamic Analysis Method*	Berdasarkan penelitian ini, penggabungan dari dua metode analisis malware yaitu analisis statis dan analisis dinamis mampu memberikan gambaran yang lebih lengkap tentang karakteristik dari malware TT.exe.

Tabel 2.4 merupakan hasil dari *study literature* yang telah dilakukan. Sebelas penelitian yang telah dilakukan menjelaskan mengenai bagaimana melakukan analisis *malware*. Tiga dari sebeleas penelitian analisis *malware* pada tabel 2.4 mendekati dengan penelitian "**Analisis *Malware Flawed Ammyy RAT* Dengan Metode *Reverse Engineering"*** ditunjukkan pada tabel 2.5.

Tabel 2.5 penelitian yang mendekati

No	Peneliti	Metode / Framework	Domain Penelitian	Hasil Penelitian
1	Heru Ari Nugroho Yudi prayudi (2015)	*Reverse Engineering*	Penggunaan teknik *Reverse Engineering* pada *malware* analisis untuk indentifikasi serangan *malware*	Hasil yang didapat dari proses *Reverse Engineering* pada *malware* biscuit adalah gambaran bagaimana cara kerja dari *malware* tersebut.

Tabel 2.5 penelitian yang mendekati

No	Peneliti	Metode / Framework	Domain Penelitian	Hasil Penelitian
2	Nayan zalavadiya Priyanka Sharman (2017)	Metode analisis statis dan analisis dinamis	*A Mtodology of malware Analysis, tools, and Technique for windows platform – RAT analysis*	Menguraikan metodologi yang efektif dan efisien yang dapat diterapkan untuk meningkatkan kinerja deteksi dan penghapusan *malware* yang dikumpulkan. Analisis dinamis cara terbaik untuk melakkan analisis *sample malware*
3	Syarif Yusirwan S Yudi Prayudi Imam Riadi (2015)	Metode analisis statis dan analisis dinamis	*Implementation of Malware Analysis using Static and Dynamic Analysis Method*	Penelitian ini, penggabungan dari dua metode analisis malware yaitu analisis statis dan analisis dinamis mampu memberikan gambaran yang lebih lengkap tentang karakteristik dari malware TT.exe.
4	Tesa Pajar Setia Nur Widiyasono Aldy Putra Aldya (2018)	Metode Analisis Dinamis dan *Reverse Engineering*	Analisis *Malware Flawed Ammyy RAT* dengan metode *Reverse Engineering*	*Malware Flawed Ammyy RAT* bekerja dengan bersembunyi pada aplikasi *Ammyy Admin* kemudian melakukan koneksi dengan *attacker* dengan *ip address* 103.208.86.69.

				netname ip address 103.208.86.69 adalah *zappie host.* Perubahan 50 registry yang dilakukan *malware* pada system yang terinfeksi.

Tabel 2.5 merupakan penelitian terkait yang telah dilakukan sebelumnya mengenai analisis malware. Penelitian yang telah dilakukan sebagai dasar mengenai penelitian ini diantaranya berjudul ”Penggunaan teknik *Reverse Engineering* pada *malware* analisis untuk indentifikasi serangan *malware*” (Nugroho & Prayudi, 2015) ini melakukan proses *Reverse Engineering* pada *malware* Biscuit. Hal mendasar dari cara kerja *malware* tersebut adalah adanya auto request untuk koneksi ke ip tertentu yaitu ip pada alamat: 114.101.115.115. Selanjutnya proses *Reverse Engineering* melalui penulusuran perintah: ***bdkzt, ckzjqk, download, exe, exit*** dan ***lists*** telah dapat memetakan bagaimana cara kerja dari *malware* Biscuit.

Penelitian yang berjudul “*A Metodology of malware Analysis, tools, and Technique for windows platform – RAT analysis*” dengan (Zalavadiya & Priyanka, 2017) ini melakukan analisis statis dan analisis dinamis pada *malware* DrakComet. Hasil dari penelitian ini adalah menguraikan metodologi yang efektif dan efisien yang dapat diterapkan untuk meningkatkan kinerja deteksi dan penghapusan *malware* yang dikumpulkan. Analisis dinamis cara terbaik untuk melakkan analisis sample *malware*.

Penelitian yang berjudul "*Implementation of Malware Analysis using Static and Dynamic Analysis Method*" dengan penulis Syarif Yusirwan S Yudi Prayudi Imam Riadi pada tahun 2015 Penelitian ini, penggabungan dari dua metode analisis *malware* yaitu analisis statis dan analisis dinamis mampu memberikan gambaran yang lebih lengkap tentang karakteristik dari *malware* TT.exe. *Malware* TT.exe adalah *malware* tipe trojan, dibuat pada hari Rabu 30 Juli 2014, menargetkan windows 7 dan windows 8. Pada awalnya ketika *malware* TT.exe aktif, *malware* akan menjalankan beberapa proses pada korban komputer seperti menyalin dirinya sendiri ke lokasi % AppData \ Roaming% dan menghapus *malware* asli. Selain itu, *malware* juga membuat beberapa *registry* yang membuat *malware* TT.exe dijalankan di sartup. M*alware* TT.exe sudah menginfeksi sistem komputer, *malware* akan menggunakan banyak memori komputer untuk menjalankan program serta menginfeksi program lain yang berjalan di komputer korban. *Malware* TT.exe juga mematikan sebagian besar sistem keamanan windows, seperti windows defender, firewall, system restore, serta menghubungi *server malware* di alamat alhanexchange.com. *Malware* TT.exe juga membuat jalan bagi peretas untuk mendapatkan akses ke sistem komputer, dengan membuka port 313436. Proses infeksi dari *malware* TT.exe.

Penelitian yang dilakukan oleh ketiga peneliti diatas. Hasil dalam penelitiannya telah memenuhi aspek yang dibutuhkan untuk penelitian yang akan di lakukan terutama penelitiannya cukup mendekati dalam hal teknik dasar yang akan

digunakan dengan judul penelitian "Analisis *Malware* "*Flawed Ammyy RAT*" dengan metode *Reverse Engineering*".

2.7 Matrik Penelitian Malware

Table 2.6 matrik penelitian

No	Judul Jurnal	Ruanglingkup Penelitian					Penulis
		Analisis			*Reverse Engineering*	*Memory Forensic*	
		Statis	*Dinamis*	*Hybrid*			
1	Investigasi Serangan *Malware* Njrat Pada PC		✓				Devi Rizky Septani
2	*Reverse Enginering Malware Ransomware* Manggunakan Analisis Dinamis		✓		✓		Egi Satria Rusman
3	Metode Klasifikasi dan Analisis Karakteristik Malware Menggunakan Konsep Ontologi	✓	✓				Abdul Haris Muhammad
4	*Malware* Analysis Pada *Windows Operating System* Untuk Mendeteksi *Trojan*	✓	✓				Sabam Chandra Yohanes Hutauruk
5	*Malware Dynamic*		✓				Retno Adenansi
6	Analisis dan Deteksi *Malware* Menggunakan Metode *Malware* Analisis Dinamis dan *Malware* Analisis Statis	✓	✓				Triawan Adi Cahyanto
7	Penggunaan Teknik *Reverse Engineering* pada *Malware* Analisis Untuk Indentifikasi Serangan *Malware*		✓		✓		Heru Ari Nugroho

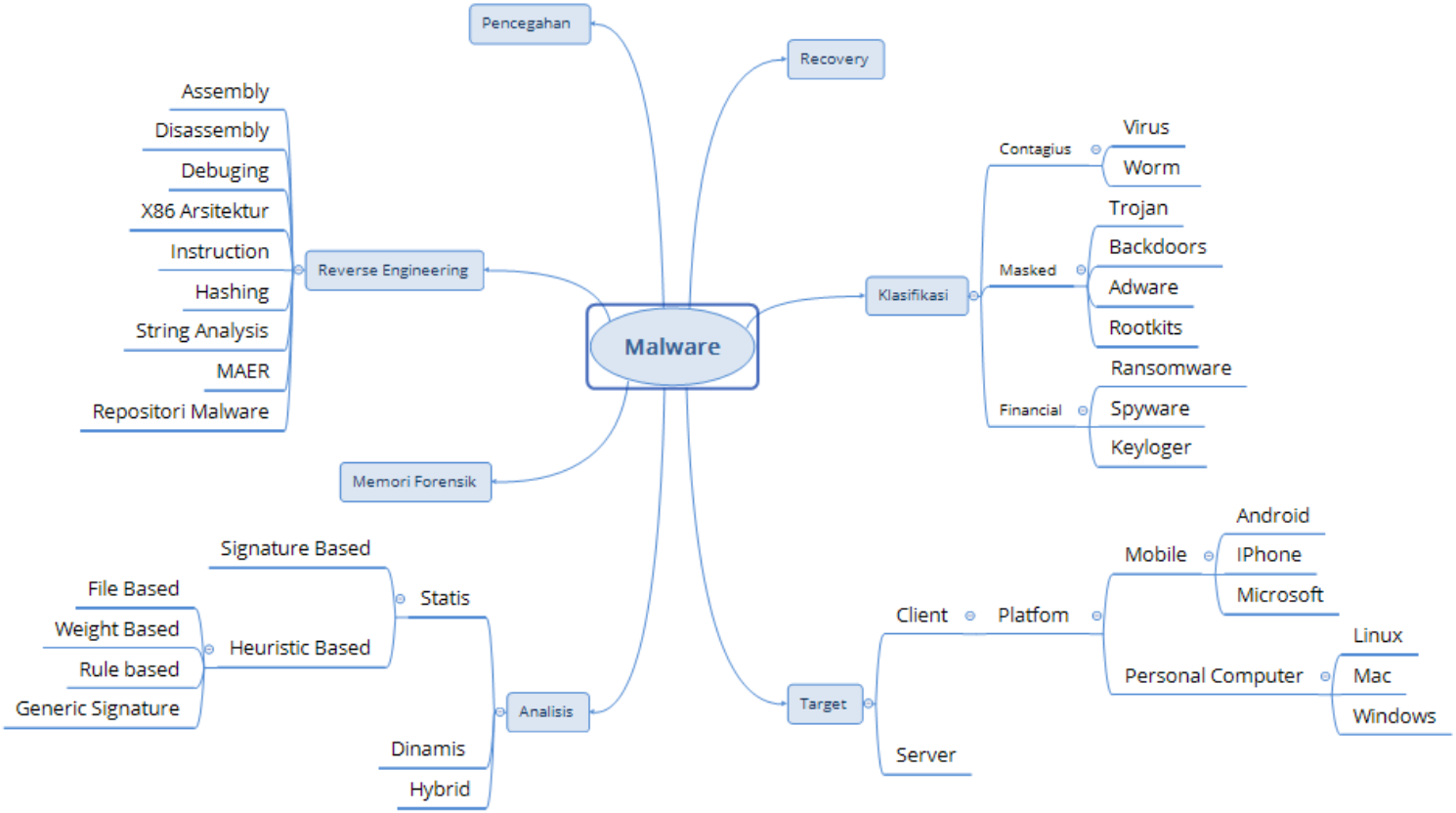
Table 2.6 matrik penelitian (lanjutan)

No	Judul Jurnal	Ruanglingkup Penelitian					Penulis
		Analisis			Reverse Engineering	Memory Forensic	
		Statis	Dinamis	Hybrid			
8	*A Metodology of malware Analysis, tools, and Technique for windows platform – RAT analysis*	✓	✓				Nayan zalavadiya
9	*Malware analysis*	✓	✓				Ujaliben Kalpesh Bavishi
10	*Basic on Malware Analysis, tools and Technique*	✓	✓				Dolly Uppal
11	*Implementation of Malware Analysis using Static and Dynamic Analysis Method*	✓	✓				Syarif Yusirwan S
12	Analisis *Malware Flawed Ammyy RAT* Dengan Metode *Reverse Engineering*		✓		✓		Tesa Pajar Setia

Tabel 2.6 menunjukkan matrik penelitian terkait mengenai malware yang telah dilakukan sebelumnya. Keterbaruan dari penelitian dengan judul "**Analisis *Malware Flawed Ammyy RAT* Dengan Metode *Reverse Engineering***" menggunakan analisis dinamis dengan menambah satu proses yaitu membangun virtual network dan menggunakan *reverse engineering disassembler.*

Peta penelitian

Berikut ini peta peneletian mengenai *malware*:

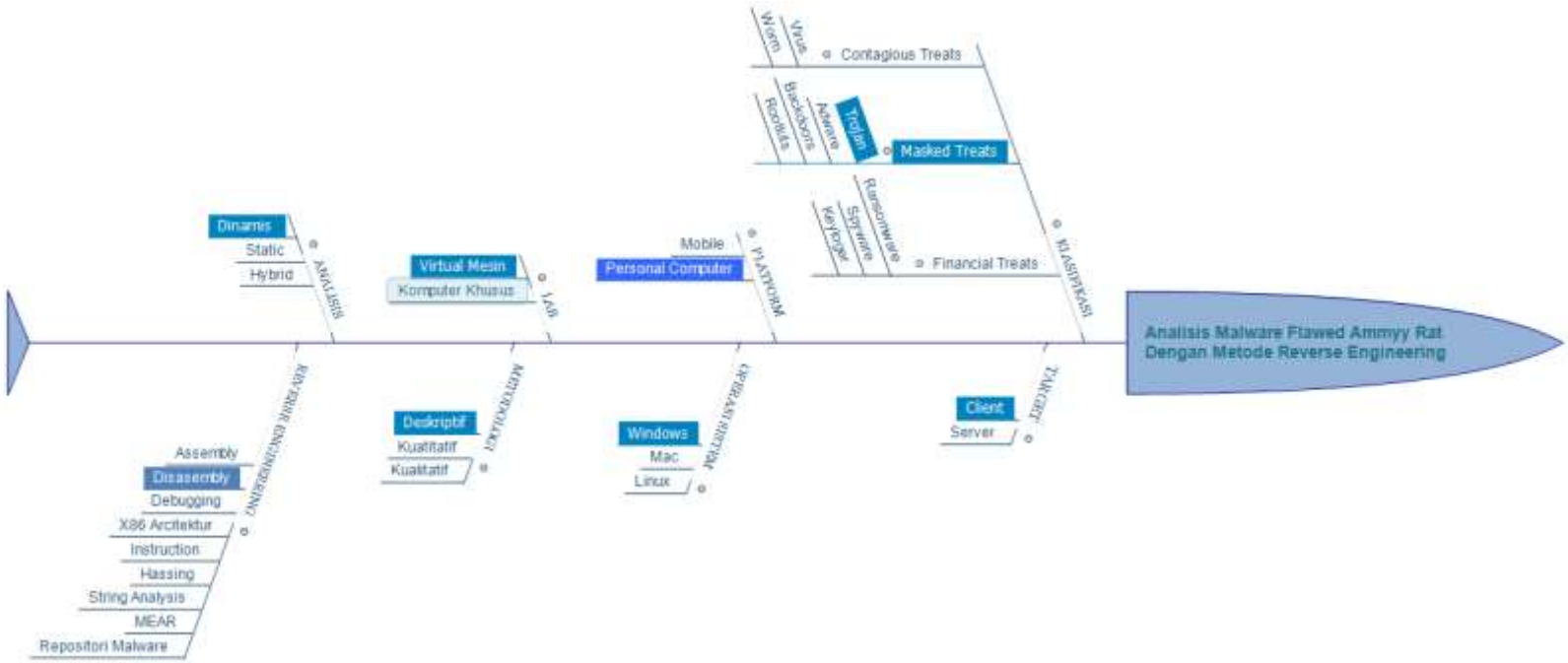


Gambar 2.3 Peta Penelitian Malware

Gambar 2.3 menunjukkan *malware* memliki 7 *sub* topik mengenai *malware*, dimulai dari klasifikasi *malware*, target serangan *malware*, metode analisis *malware*, *memory forensic malware*, *reverse engineering malware*, pencegahan serangan malware dan proses *recovery* setelah terinfeksi *malware*. Penelitian ini berfokus pada proses analisis *malware Flawed Ammyy RAT*, *reverse engineering malware Flawed Ammyy RAT*, pencegahan dari serangan *malware Flawed Ammyy RAT* serta proses *recovery* setelah terinfeksi *malware Flawed Ammyy RAT*.

2.8 Diagram *Fishbone*

Berikut ini diagram fishbone analisis *malware Flawed Ammyy RAT*:



Gambar 2.4 diagram *fishbone*

Gambar 2.4 menunjukkan diagram tulang ikan (*fishbone*) penelitian, diagram ini menggambarkan hal-hal yang terkait dengan penelitian dan rencana *output* penelitian. Klasifikasi *malware* yang akan di teliti adalah *Trojan*. *Sample malware* yang didapat, *malware* melakukan serangan pada *target* posisi sebagai *client*, *platform* yang diserang adalah Personal Computer, sedangkan operasi system yang diserang adalah *Windows*. Laboratorium untuk melakukan analisis *malware* ini menggunakan *virtual* mesin. Penelitian ini menggunaka motodologi deskriptif, sedangkan untuk melakukan analisis *malware* menggunakan analisis dinamis dan *reverse engineering disassembly*.