BAB II

LANDASAN TEORI

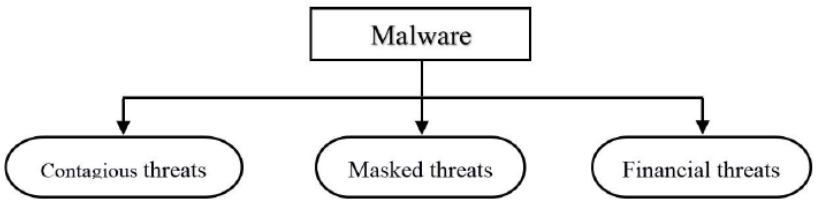
2.1 Teori Pendukung Penelitian

2.1.1 *Malware*

Malware adalah perangkat lunak berbahaya, yang diprogram untuk merusak atau untuk mendapatkan akses ke sistem komputer tanpa sepengetahuan pemilik sistem. *Virus*, *Worms*, Trojan, *Key logger* dan *Spyware* adalah contoh *malware* yang paling banyak digunakan. Istilah, seperti "*worm*", "*virus*", atau "*Trojan horse*" digunakan untuk klasifikasi *malware* yang menunjukkan perilaku jahat serupa (Zalavadiya, 2017).

2.1.2 Klasifikasi *Malware*

Malware dapat diklasifikasikan dalam berbagai kelas dan kategori yang umumnya dikategorikan menurut proses dan reaksi yang dicapai pada sistem yang terinfeksi yang tergantung pada proses perancangan dan pengembangan program jahat. Gambar 2.1 berikut menunjukkan berbagai jenis *malware* (Zalavadiya, 2017).



Gambar 2.1 Klasifikasi *Malware* (Zalavadiya, 2017)

a. *Contagious Threats* (Ancaman yang Menular)

Tabel 2.1 Deskripsi *Contagious Threats* (Zalavadiya, 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

Tabel 2.1 menunjukan *Contagious Threats* (Ancaman yang Menular) yang didalamnya ada *virus* dan *worm. Malware virus dan worm* merupakan kategori pada ancaman yang menular, ini dilihat dari karakteristik dan cara kerja *malware* melakukan infeksi pada sistem.

b. *Masked Threats* (Ancaman Bertopeng)

Tabel 2.2 Deskripsi *Masked Threats* (Zalavadiya, 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	Dipasang melalui eksploitasi perangkat lunak atau *Trojan*	Mencuri kata sandi atau melakukan instal *keylogger*

Tabel 2.2 menunjukan *Masked Threats* (Ancaman Bertopeng). *Malware trojan, backdoor, adware,* dan *rootkits* merupakan kategori pada ancaman

bertopeng, ini dilihat dari karakteristik dan cara kerja *malware* melakukan infeksi pada sistem.

c. *Financial Threats* (Ancaman Keuangan)

Tabel 2.3 Deskripsi *Financial Threats* (Zalavadiya, 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	Dipasang 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*	Dipasang oleh program jahat lain atau ketika seorang pengguna mengujungi.	Menangkap infromasi *sensitive* seperti nama pengguna kata sandi nomor kartu kredit atau rincian perbankan *online*

Tabel 2.3 menunjukan *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 sistem

2.1.3 Trojan Horse

Secara umum *trojan horse* dapat diartikan sebagai sebuah *software* berbahaya (*malware*) yang memiliki kemampuan dalam pengontrolan atau pengaksesan data antar jaringan (Ardiansyah, 2014).

Beberapa jenis *trojan* berdasarkan fungsi dan kemampuannya (S'to, 2010), yaitu:

a. *Trojan Remote Access,* yaitu jenis *trojan* yang bekerja dengan cara membuka sebuah *port* secara diam-diam sehingga *hacker* bisa mengendalikan komputer korban (Ardiansyah, 2014).
b. *Trojan Data-Sending,* yaitu suatu jenis *trojan* yang bertujuan untuk mengirimkan data-data tertentu (*password,* data *credit card,* dan sebagainya) yang berada pada komputer korban ke sebuah *email* khusus yang telah disiapkan (Ardiansyah, 2014).
c. *Trojan Destructive,* yaitu suatu jenis *trojan* yang sangat berbahaya karena jika telah menginfeksi sistem komputer maka *trojan* ini akan menghapus semua file sistem pada komputer korban (seperti file .dll, .ini atau .exe) (Ardiansyah, 2014).
d. *Trojan DoS Attack,* yaitu suatu jenis *trojan* yang memiliki kemampuan untuk menjalankan *Distributed DoS (DDoS)* melalui komputer korban (Ardiansyah, 2014).

e. *Trojan Proxy*, yaitu suatu jenis *trojan* yang berfungsi untuk membuat kom-puter korban menjadi seperti sebuah komputer perantara/*proxy*, sehingga *hacker* dapat menyembunyikan identitas dirinya ketika melakukan kegiatan ilegal menggunakan komputer tersebut (Ardiansyah, 2014).

f. *Trojan FTP*, yaitu sebuah jenis *trojan* yang paling sederhana karena hanya memiliki sebuah fungsi yaitu membuka *port* 21 di komputer korban (Ardiansyah, 2014).

g. *Trojan Software Detection Killers*, yaitu jenis *trojan* yang memiliki kemampuan untuk mendeteksi dan melumpuhkan fungsi antivirus dan *firewall* pada sistem komputer (Ardiansyah, 2014).

2.1.4 *Malware* Zeus

Zeus pertama kali diidentifikasi pada bulan Juli 2007, ketika digunakan untuk mencuri informasi dari departemen perhubungan Amerika Serikat dan menjadi luas pada tahun 2009. Bulan Juli 2009, perusahaan keamanan Prevx menemukan bahwa *Zeus* telah menginfeksi lebih dari 74.000 akun FTP pada website perusahaan seperti Bank of America, NASA, Monster, ABC, Oracle, play.com, Cisco, Amazon, BusinessWeek. Tanggal 14 Juli 2010, perusahaan keamanan Trusteer mengajukan laporan yang menyatakan bahwa kartu kredit lebih dari 15 bank - bank AS yang tidak disebutkan namanya telah terinfeksi. Tanggal 1 Oktober 2010, FBI

mengumumkan telah menangkap jaringan utama internasional dari kejahatan dunia maya yang telah menggunakan *Zeus* untuk meretas komputer Amerika Serikat dan berhasil mencuri sekitar $70 juta. Lebih dari 90 orang ditangkap di Amerika Serikat, Inggris dan Ukraina (Kurniawan, 2014).

Secara umum, *Zeus* bertujuan untuk membuat mesin berperilaku sebagai agen mata-mata dengan tujuan mendapatkan keuntungan finansial. *Malware Zeus* memiliki kemampuan untuk mencatat input yang dimasukkan oleh pengguna serta untuk menangkap dan mengubah data yang ditampilkan ke halaman *web*. Data yang dicuri dapat berisi alamat *email*, kata sandi, akun perbankan *online*, nomor kartu kredit, dan nomor otentikasi transaksi (Binsalleeh, 2010).

Kode sumber *Zeus* bocor pada 2011 dan sejak itu ada banyak varian yang muncul. Fitur-fitur penting dalam varian baru *Zeus* termasuk penambangan bitcoin, perintah *peer-to-peer* dan infrastruktur kontrol, dan menambahkan lapisan enkripsi ke *file* konfigurasi (Mohaisen, 2013).

Malware Zeus menginfeksi sistem dengan menulis salinannya sendiri ke APPDATA folder menggunakan nama *file* yang dibuat secara acak. Data yang dicuri disimpan di bawah direktori yang sama, APPDATA dan dienkripsi. *Zeus* mengirimkan data yang telah dicuri untuk diperintah dan untuk mengendalikan

server, kemudian menghapus salinan lokal. Setelah terinfeksi, *Zeus* menyuntikkan ke proses explorer.exe dan proses sistem yang berjalan lainnya (Mohaisen, 2013).

Zeus menjalankan tugas utamanya dari *explorer.exe* dan mengkomunikasikan ke *server* perintah dan kontrol melalui *explorer*. Varian baru menggunakan *peer-2-peer* terus keluar proses *explorer.exe* tetapi tidak membuat permintaan *HTTP*. *Ze*us mengaitkan beberapa pemrograman aplikasi *Windows* penting untuk menyadap data yang dikirim dari browser dan untuk melakukan modifikasi halaman yang dilihat oleh korban. Misalnya *Zeus* mampu menambahkan bidang dalam formulir web untuk mengumpulkan informasi tambahan dari korban ketika mengunjungi situs perbankan (Mohaisen, 2013).

Setelah *Zeus* menginfeksi sistem dan membangun koneksi dengan *server* perintah dan control (C&C), *Zeus* akan mengunduh versi terbaru dari *file* konfigurasi yang memberi tahu bot situs yang akan dituju. *File* konfigurasi digunakan untuk disimpan dalam APPDATA yang direktori dengan *file* lainnya. Varian terbaru dari *Zeus* telah mengubah area penyimpanan dan metode penyimpanan konfigurasi menjadi terlindungi. *File* konfigurasi dienkripsi dan disimpan dalam registri dengan nama kunci acak. *File* konfigurasi adalah aspek penting dari *Trojan* perbankan *Zeus* dan dapat

mengungkapkan banyak informasi tentang kampanye *Zeus* (Mohaisen, 2013).

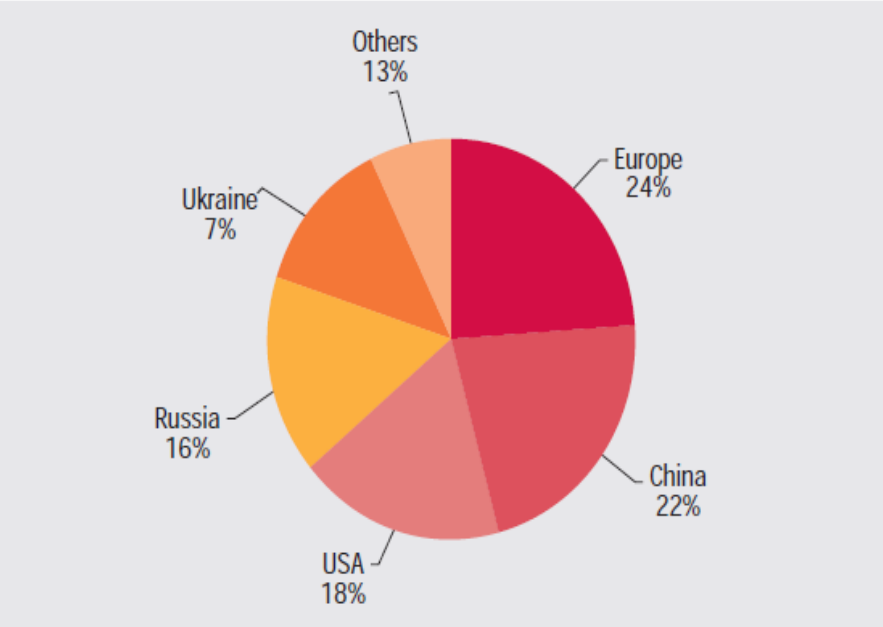
File konfigurasi dapat berisi daftar *server* perintah dan kontrol cadangan, tautan ke versi *Zeus* yang diperbarui, daftar situs web yang ditargetkan, dan daftar *HTML* dan *JavaScript* yang akan disuntikkan ke situs web yang ditargetkan. Informasi *file* konfigurasi penting karena administrator sistem dapat memblokir akses ke semua domain yang digunakan untuk cadangan, entitas yang ditargetkan dapat diberitahu tentang kampanye tertentu yang memengaruhi penggunanya, dan keamanan dapat melacak infrastruktur yang digunakan oleh penyerang (Mohaisen, 2013).

Zeus adalah bagian penting dari *malware* adalah karena itu adalah *trojan* perbankan paling lazim di alam liar. *Zeus* bertanggung jawab atas sebagian besar kejahatan dunia maya yang menargetkan bank dan bisnis kecil, yang menyerukan penyelidikan lebih lanjut untuk melakukan identifikasi sampel *malware* yang termasuk dalam keluarga ini dan menunjukkan perilaku unik baru yang tidak dilihat sebelumnya (Mohaisen, 2013).

a. Skala Infeksi

Banyak sistem yang terinfeksi oleh *zeus*, diperkirakan sekitar 3.6 juta komputer yang terinfeksi di Amerika Serikat saja. Penelitian menunjukkan bahwa sejak April 2010, 88% dari 500 keuntungan perusahaan telah dipengaruhi oleh *malware* ini (Unisys, 2010).

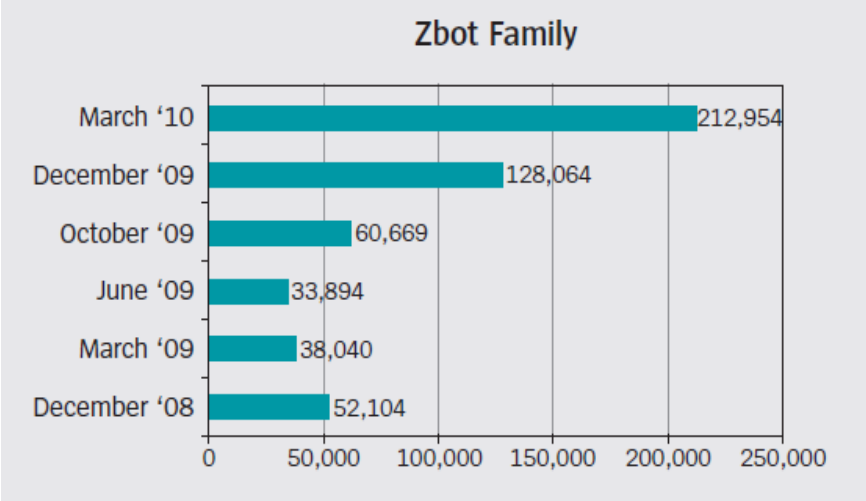
Zeus besar yang baru-baru ini terdeteksi adalah *botnet* yang disebut Kneber. Bulan Februari 2010, perusahaan keamanan korporat yang berbasis di AS, NetWitness, melaporkan deteksi komputer yang terinfeksi *Zeus* di 2.500 organisasi di 196 negara di seluruh dunia. Sebanyak 76.000 komputer yang terinfeksi terdeteksi (Unisys, 2010).



Gambar 2.2 Negara Yang Terinfeksi *Zeus* (Unisys, 2010)

Tanggal 28 Oktober 2009 *Zeus* juga mengirim lebih dari 1,5 juta pesan *phishing* di *Facebook*. Bulan November 2009, *Zeus* menyebar melalui *email* yang mengaku berasal dari Verizon Wireless. Sebanyak sembilan juta *email phising* ini dikirim.

Menurut Pusat Perlindungan *Malware Microsoft*, jumlah infeksi *Zeus* telah meningkat jika dibandingkan dengan tahun lalu. Gambar di bawah ini menggambarkan hal yang sama:



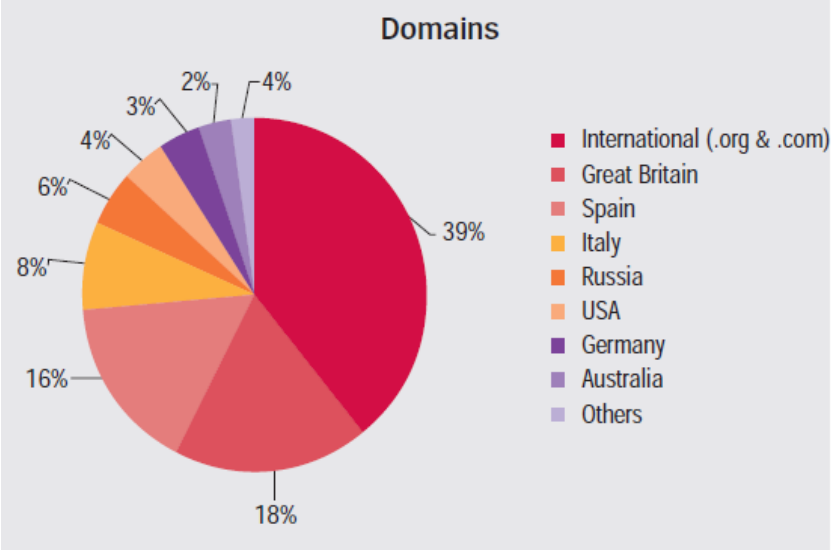
Gambar 2.3 Jumlah Infeksi *Zeus* (Unisys, 2010)

Ada 11 domain internasional yang ditargetkan *Zeus*, 8 bank yang menawarkan layanan internet banking kepada kliennya dan 3 lainnya adalah penyedia layanan internet komersial.

Menjalankan Zeus dari lokasi mana pun sangat mungkin terjadi. Pengguna jahat menempatkan *server* mereka dengan penyedia Eropa, Cina, Amerika Utara, dan Rusia karena mereka menawarkan layanan hosting yang dikembangkan dengan baik. *Zeus* merekam lokasi *host* ketika *bot* memeriksa ke dalam perintah dan *server* kontrol.

Zeus menargetkan domain tingkat atas tertentu, domain yang paling umum dicatat adalah domain internasional (*.org* dan *.com*) yang dimiliki oleh perusahaan multinasional besar.

Bagan berikut dengan jelas menggambarkan domain teratas yang ditargetkan oleh Zeus:



Gambar 2.4 Domain Yang Ditargetkan *Zeus* (Unisys, 2010)

Lima (5) negara korban teratas yang terkena dampak Zeus adalah:

Country Name	% Machines Infected
Egypt	19%
Mexico	15%
Saudi Arabia	13%
Turkey	12%
United States	11%

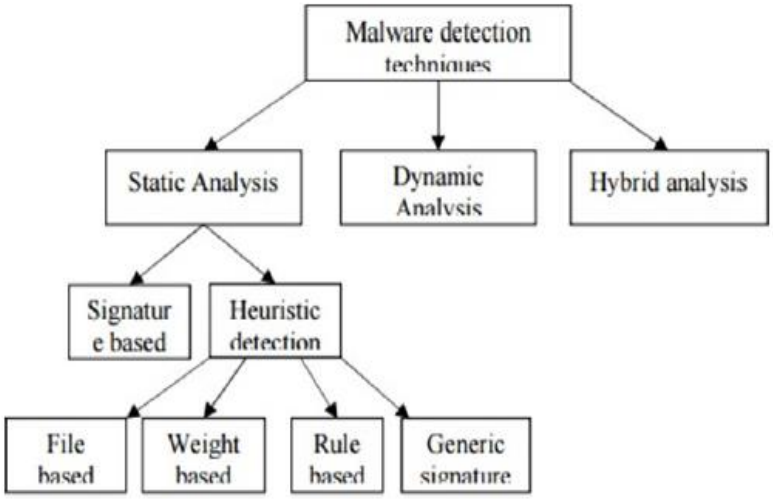
Gambar 2.5 Negara Yang Terkena Dampak *Zeus* (Unisys, 2010)

2.1.5 *Malware* Analysis

Malware analysis adalah kumpulan dari proses penentuan tujuan dan fungsionalitas dari *sample malware* yang diberikan seperti *virus, worm, trojan,* dan sebagainya untuk melakukan deteksi *malicious code.* Baik dengan mengeksekusi *malware* tersebut

(*dynamic analysis)* ataupun dengan pemeriksaan kode program saat sebelum dieksekusi (Hutauruk, 2016).

Tujuan dari *malware analysis* yang dilakukan pada *malware* jenis *trojan* ini adalah untuk menghasilkan data karakteristik *trojan* dan menganalisis siklus hidup masing – masing jenis *trojan.* Melakukan *malware analysis* dilakukan dengan metode penelitian berupa studi literature berupa pendalaman materi yang berhubungan dengan *malware analysis* (Hutauruk, 2016).



Gambar 2.6 Metode *Malware* Analysis (Jerlin, 2015)

a. Analisis *Malware* Static

Malware statis analisis adalah proses menganalisis program dengan memeriksanya. Dikenal sebagai teknik analisis kode. Sebelum program benar-benar dijalankan, informasi statis ditemukan dalam *file* yang dapat dieksekusi termasuk data header dan urutan *byte* yang digunakan untuk menentukan apakah *file* tertentu adalah *file* berbahaya atau tidak (Jerlin, 2015).

Disassembly, menjadi salah satu teknik analisis statis, analisis statis dalam proses ini dilakukan dengan pembongkaran menggunakan alat-alat *disassemble* yang digunakan untuk mendapatkan *file* program bahasa *assembly*. *Opcode* diekstrak sebagai fitur untuk menganalisis perilaku aplikasi secara statis untuk mendeteksi *malware* (Jerlin, 2015).

1. *Signature-based Detection*

   Deteksi berbasis *signature* mempertahankan catatan tanda tangan dan mengidentifikasi *malware* dengan membandingkan atau menyejajarkan pola terhadap database. Sebagian besar alat *antivirus* didasarkan pada teknik pendeteksian ini. Tanda tangan dibuat untuk memeriksa kode yang dibongkar dari biner perangkat lunak perusak. Kode ini kemudian dianalisis dan fitur-fiturnya diekstraksi. Fitur yang diekstraksi digunakan dalam membuat tanda tangan dari keluarga *malware* tertentu (Jerlin, 2015).

   Keuntungan utama dari teknik pendeteksian berbasis *signature* adalah bahwa ia dapat mendeteksi *malware* yang dikenal secara akurat sedangkan sumber daya yang lebih sedikit diperlukan untuk mengungkap *malware* dan terutama berfokus pada tanda tangan serangan sementara kelemahan utamanya adalah karena tidak ada tanda tangan yang tersedia,

tidak dapat mendeteksi kejadian baru yang tidak dikenal dari *malware* (Jerlin, 2015).

2. *Heuristic-based Detection*

Disebut sebagai deteksi berbasis perilaku atau anomali. Tujuan utama dari teknik deteksi ini adalah untuk menganalisis perilaku *malware* selain diketahui atau tidak diketahui. Parameter perilaku mencakup berbagai faktor seperti jenis sumber atau tujuan lampiran, alamat *malware*, dan fitur statistik lainnya yang dapat dihitung (Jerlin, 2015).

Biasanya terjadi pada fase pelatihan dan fase deteksi. Selama fase pelatihan perilaku sistem diamati dengan tidak adanya serangan dan teknik pembelajaran mesin digunakan untuk membuat profil perilaku normal tersebut (Jerlin, 2015).

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, 2014).

b) *Weight based heuristic analysis*

Analisis heuristik berdasarkan *weight* adalah teknik kuno. 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, 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 terdapat *malware* (Uppal, 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, 2014).

b. Analisis *Malware* Dinamis

Analisis *malware* dinamis dikenal sebagai analisis *file* yang terinfeksi selama pelaksanaannya. *File* yang terinfeksi dianalisis dalam lingkungan simulasi, sesuatu seperti mesin *virtual*. Menggunakan alat-alat tertentu seperti *SysAnalyzer*, *Process Explorer*, dan lain - lain. Mengidentifikasi perilaku umum *file* tertentu. Prosesnya, *file* terdeteksi setelah mengeksekusinya di lingkungan yang sebenarnya dan selama pelaksanaan *file*

interaksi sistemnya, perilaku dan efeknya pada sistem diamati (Jerlin, 2015).

Keuntungan dari analisis dinamis adalah bahwa secara akurat menganalisis *malware* yang dikenal maupun yang tidak dikenal, teknik analisis ini lebih memakan waktu. Ini membutuhkan waktu sebanyak untuk mempersiapkan lingkungan untuk analisis *malware* seperti lingkungan mesin *virtual* (Jerlin, 2015).

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, 2014).

2.1.6 Reverse Engineering

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. R.E 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, 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, 2015)

a. *Assembler*

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

b. *Disassembler.*

Disassembler merupakan kebalikan dari proses *assembler*. Proses *disassembler* digunakan dalam teknik *Reverse Engineering* untuk menerjemahkan dari bahasa mesin ke bahasa yang mudah dimengerti manusia, yaitu bahasa *assembler* (Nugroho, 2015).

c. *Debugging.*

Proses *debugging* adalah proses pengujian dari software. 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, 2015).

d. *X86 Arsitektur.*

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

e. *Instruction.*

Instruksi adalah konstruksi yang dibangun dari program *assembly*. *Assembly x86* instruksi terdiri dari nemonic dan nol atau lebih operands (Nugroho, 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 hash pada *sample malware* setelah dilakukan proses analisis (Nugroho, 2015).

g. *String Analysis.*

String atau karakter dalam sebuah program seperti (.,A-) merupakan nilai yang akan dilakukan proses load oleh *sample malware* ketika dieksekusi. Menjadikan dalam proses *Reverse*

Engineering harus dilakukan string analisis untuk mendapatkan bukti kuat dari *sample malware* (Nugroho, 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, 2015).

i. *Repository Malware.*

Repository malware merupakan tempat disimpannya *sample malware* yang telah berhasil melakukan serangan kedalam sistem komputer diamana pun. *Repository malware* dibuat untuk memberikan *sample* kepada seorang *malware* analis untuk melakukan analisa terhadap *malware* yang sudah berhasil melakukan serangan (virusshare) (Nugroho, 2015).

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, 2015).

2.2 Penelitian Terkait Sebelumnya

Tabel 2.4 Literatur Review

No.	Penerbit / Tahun	Judul	Metode	State Of The Art
1.	H. Binsalleeh, T. Ormerod, A. Boukhtouta, P. Sinha, A. Youssef, M. Debbabi, & L. Wang / 2010	On the Analysis of the Zeus Botnet Crimeware Toolkit	• Reverse Engineering	• Menyajikan analisis rekayasa balik terperinci dari perangkat crimeware *Zeus* untuk mengungkap arsitektur yang mendasarinya dan memungkinkan mitigasinya. Merancang alat untuk melakukan pemulihan kunci enkripsi dan ekstraksi informasi konfigurasi dari *executable bot* biner.
2.	Heru Ari Nugroho & Yudi prayudi / 2015	Penggunaan teknik *Reverse Engineering* pada *malware* analisis untuk identifikasi serangan *malware*	• Reverse Engineering	• Hasil yang didapat dari proses *Reverse Engineering* pada *malware* biscuit adalah gambaran bagaimana cara kerja dari *malware* tersebut.
3.	Devi Rizky Septiani, Nur Widiyasono & Husni Mubarok / 2016	Investigasi Serangan *Malware Njrat* Pada PC	Dynamic Analysis	• Cara kerja *malware Njrat* sifatnya sangat berbahaya sehingga *attacker* dapat melakukan hak akses apa saja terhadap PC korban bahkan untuk membobol *web* dan juga *passwordnya* dapat dilakukan dengan adanya serangan *malware* yang terjadi.
4.	Mohammad Abu Qbeitah dan Monther Aldwairi / 2018	Dynamic Malware Analysis of Phishing Emails	• Dinamic Analysis	• Menganalisis 173 *email Phishing* baru-baru dan 45 pesan SPIM untuk mencari *malware* yang berpotensi baru, menyajikan dua sampel malware dan analisis dinamisnya yang komprehensif. • Menangkap sampel *malware* baru, secara dinamis menganalisisnya untuk memahami bagaimana *malware*

Tabel 2.4 Literatur Review (Lanjutan)

				akan berperilaku setelah eksekusi seperti apa perubahan pada *OS*, sistem *file*, *registry*, dan komunikasi jaringan.
5.	Tesa Pajar Setia, Nur Widiyasono & Aldy Putra Aldya / 2018	Reverse Engineering Untuk Analisi *Malware* FLAWED AMM RAT	• Reverse Engineering Dynamic Analysis	• Menunjukan bahwa *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. • *Attacker* terkonesi dengan korban maka *attacker* dengan mudah melakukan *remote control* tanpa sepengetahuan korban.
6.	Santi Nur Apriyani / 2019	Analisis Dan Reverse Engineering Pada Malware Zeus.	• Dynamic Analysis • Reverse Engineering	• Menunjukan hasil proses dari identifikasi *malware zeus* dengan metode analisis dinamis adalah mendapatkan *username* dan *password* yang diinputkan oleh korban. • Percobaan yang dilakukan dengan metode *reverse engineering* menghasilkan struktur dari *malware zeus* secara detail dari sistem operasi malware zeus hingga interaksi antara user dan program.

Tabel 2.4 merupakan hasil dari *literatur review* yang telah dilakukan sebelumnya. Lima (5) penelitian yang telah dilakukan menjelaskan mengenai bagaimana melakukan analisis terhadap *malware*. Satu dari lima (5) penelitian analisis *malware* pada tabel 2.4 mendekati dengan penelitian "**Analisis *Dan Reverse Engineering Pada Malware Zeus"*** ditunjukkan pada tabel 2.4.

Tabel 2.5 Penelitian Terdekat

No.	Penerbit / Tahun	Judul	Metode	State Of The Art
1.	H. Binsalleeh, T. Ormerod, A. Boukhtouta, P. Sinha, A. Youssef, M. Debbabi, & L. Wang / 2010	On the Analysis of the Zeus Botnet Crimeware Toolkit	• Reverse Engineering	• Menyajikan analisis rekayasa balik terperinci dari perangkat crimeware Zeus untuk mengungkap arsitektur yang mendasarinya dan memungkinkan mitigasinya. Pemulihan kunci enkripsi dan ekstraksi informasi konfigurasi dari *executable bot biner*. Selain itu, memberikan rincian untuk struktur pesan jaringan *botnet Zeus*.
2.	Santi Nur Apriyani / 2019	Analisis Dan Reverse Engineering Pada Malware Zeus.	• Analisis Dinamis • Reverse Engineering	• Menunjukan hasil proses dari identifikasi *malware zeus* dengan metode analisis dinamis adalah mendapatkan *username* dan *password* yang diinputkan oleh korban. • Percobaan yang dilakukan dengan metode *reverse engineering* menghasilkan struktur dari *malware zeus* secara detail dari sistem operasi *malware zeus* hingga interaksi antara user dan program.

Tabel 2.5 merupakan tabel penelitian terdekat yang telah dilakukan sebelumnya yang berhubungan dengan analisis *malware*. Penelitian yang telah dilakukan sebagai dasar penelitian terdekat ini melakukan proses *Reverse Engineering* pada *Malware Zeus*. Bulan Juni 2009 bahwa *Zeus* adalah ancaman utama yang layak untuk upaya rekayasa terbalik atau *reverse engineering*.

Faktanya, prediksi ini dikonfirmasi pada Juli 2009 ketika sebuah publikasi keamanan dari Damballa memposisikan *Zeus* sebagai ancaman *botnet* nomor 1 dengan 3,6 juta infeksi di AS saja (sekitar 19% dari basis PC yang dipasang di AS). Diperkirakan juga bahwa *Zeus* bersalah dalam 44% infeksi *malware* perbankan, selanjutnya dilakukan proses *Reverse Engineering* perangkat crimeware *Zeus* untuk mengungkap arsitektur yang mendasarinya dan memungkinkan mengurangi resikonya. Merancang alat untuk melakukan pemulihan kunci enkripsi dan ekstraksi informasi konfigurasi dari *executable* bot biner. Selain itu, memberikan rincian untuk struktur pesan jaringan *botnet Zeus*.

Tabel 2.6 Matriks Penelitian

No	Peneliti/Tahun	Judul	Ruang Lingkup Penelitian								
			Metode		Implementasi				Malware		
			Dinamic	Reverse Engineering	Sistem Operasi	Internet	LAN	Pencegahan	Trojan	Backdoors	Spyware
1.	H. Binsalleeh, T. Ormerod, A. Boukhtouta, P. Sinha, A. Youssef, M. Debbabi, & L. Wang / 2010	On the Analysis of the Zeus Botnet Crimeware Toolkit		√		√			√		
2.	Heru Ari Nugroho & Yudi prayudi / 2015	Penggunaan teknik *Reverse Engineering* pada *malware* analisis untuk indentifikasi serangan *malware*		√	√					√	

Tabel 2.6 Matriks Penelitian (Lanjutan)

3.	Devi Rizky Septiani, Nur Widiyasono & Husni Mubarok / 2016	Investigasi Serangan *Malware Njrat* Pada PC	√		√				√		
4.	Mohammad Abu Qbeitah dan Monther Aldwairi / 2018	Dynamic Malware Analysis of Phishing Emails	√		√						√
5.	Tesa Pajar Setia, Nur Widiyasono & Aldy Putra Aldya / 2018	Reverse Engineering Untuk Analisis *Malware* FLAWED AMM RAT	√	√	√						√
6.	Santi Nur Apriyani / 2019	Analisis Dan Reverse Engineering Pada Malware Zeus.	√	√	√		√	√	√		

Tabel 2.6 menunjukkan matrik penelitian terkait mengenai *malware* analisis yang telah dilakukan sebelumnya. Keterbaruan dari penelitian dengan judul "**Analisis *Dan Reverse Engineering Pada Malware Zeus***" menggunakan metode analisis dinamis yang akan dilakukan pada mode *virtual* dan menggunakan *reverse engineering* sebagai proses dari identifikasi serangan *Malware Zeus* serta melakukan cara pencegahan pada *malware zeus* tersebut.