Analisis dan Deteksi Malware Menggunakan Metode Malware Analisis Dinamis dan Malware Analisis Statis

Triawan Adi Cahyanto[1)], Victor Wahanggara[2)], Darmawan Ramadana[3)]
[1,2,3)]Jurusan Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Jember
Email:[1)]triawanac@unmuhjember.ac.id, [2)]wahanggara.ti.victor@gmail.com, [3)]idar@mail.com

ABSTRAK

Malware merupakan perangkat lunak atau *software* yang diciptakan untuk menyusup atau merusak sistem komputer. Penyebaran *malware* saat ini begitu mudah baik melalui *usb flashdisk*, iklan-iklan tertentu pada *website*, dan media lainnya. Semuanya sangat erat kaitannya dengan tindak kejahatan seperti pencurian *file*, kartu kredit, *internet banking* dan lain sebagainya. Berkaitan dengan hal itu, ada suatu bidang yang menangani tindak kejahatan yaitu forensik digital. Salah satu tahapan dalam forensik digital yaitu melakukan analisis terhadap barang bukti digital, dalam hal ini adalah *malware*. Untuk membuktikan suatu *software* dikatakan *malware* adalah dengan mengetahui cara kerja program tersebut pada sistem komputer. Metode *Malware* Analisis Dinamis dan Statis merupakan kombinasi metode yang sesuai untuk menganalisa cara kerja *malware*. Berdasarkan analisa tentang cara kerja *malware* (poison ivy), dapat disimpulkan bahwa terdapat beberapa *signature*, *filename*, dan *string* yang sudah diteliti ternyata dapat melakukan proses *login* secara *remote* tanpa diketahui oleh pemilik komputer.

***Kata kunci** : Forensik Digital, Malware Analysis, Dynamic Analysis, Static Analysis*

1. PENDAHULUAN

Dalam era teknologi yang semakin berkembang pesat saat ini, komputer digunakan untuk memudahkan pekerjaan manusia, dalam pengoperasiannya ada *software* yang berjalan diatas sistem operasi, dan sangat berperan penting dalam melakukan tugas-tugas yang dikerjakan oleh pengguna. Karena melalui *software* inilah suatu komputer dapat menjalankan perintah sehingga membantu pengguna dalam menyelesaikan pekerjaannya. Namun tidak semua *software* dapat membantu dan memudahkan manusia dalam melakukan pekerjaannya, ada pula jenis *software* yang diciptakan untuk melakukan perusakan atau tindak kejahatan yang dapat merugikan orang lain, software tersebut dikategorikan sebagai *Malicous Software*.

Malicous Software atau yang lebih dikenal sebagai *Malware* merupakan perangkat lunak yang secara eksplisit didesain untuk melakukan aktifitas berbahaya atau perusak perangkat lunak lainnya seperti *Trojan, Virus, Spyware* dan *Exploit* (Kramer & Bradfield, 2010). *Malware* diciptakan dengan maksud tertentu yaitu melakukan aktifitas berbahaya yang berdampak sangat merugikan bagi para korbannya, antara lain seperti penyadapan serta pencurian informasi pribadi, hingga kasus perusakan sistem yang dilakukan oleh penyusup (*Intruder*) terhadap perangkat korban dengan berbagai alasan. Salah satu media yang digunakan oleh *intruder* untuk mengendalikan komputer pengguna secara diam-diam dari jarak jauh adalah *malware poison ivy*, dikenal sebagai "*trojan access remote*" karena dapat memberikan kontrol penuh kepada *intruder* melalui pintu belakang (*backdoor)*. Kemampuan *malware poison ivy* mengadopsi dari software *Remote Administration Tool* (RAT), yaitu termasuk kategori *software* yang baik (legal) yang

dapat melakukan *monitoring* dan pengontrolan secara penuh. Contoh penggunaan *software* RAT ini biasa digunakan oleh seorang pimpinan perusahaan untuk mengontrol perangkat kerja (komputer) karyawannya melalui jaringan jarak jauh. Dengan fitur tersebut tidak jarang *malware poison ivy* dikatakan juga sebagai *Software* RAT yang ilegal (RAT *Malware*) dikarenakan tidak memberikan informasi berupa *notifikasi* saat proses *remote* terhubung (terhubung secara diam-diam), dengan *malware* sebagai medianya maka dalam hal ini merupakan sebuah bukti tindak kejahatan digital yang dilakukan oleh seorang *intruder.*

Forensik Digital merupakan disiplin ilmu yang menerapkan investigasi dan identifikasi dalam menindak kejahatan digital (T. A. Cahyanto & Prayudi, 2014). Salah satu tahapan utama dalam menginvestigasi tindak kejahatan yaitu mengumpulkan barang bukti digital. Untuk menemukan barang bukti digital pada *malware*, dibutuhkan analisis lebih mendetail agar dapat mendeteksi aktifitas sebuah *malware* serta mempelajari bagaimana sebuah *malware* menginfeksi dan berkembang dalam sebuah sistem (T. Cahyanto, 2015; T. A. Cahyanto, Oktavianto, & Royan, 2013). Ada dua tipe analisis dalam melakukan analisis pada *malware* yaitu dengan analisis statis (analisa kode) dan analisis dinamis (Gandotra, Bansal, & Sofat, 2014; Sikorski & Honig, 2013; Tzermias, Sykiotakis, Polychronakis, & Markatos, 2011). Meskipun dari kedua tipe analisis tersebut mempunyai tujuan yang sama yaitu menjelaskan tentang bagaimana sebuah *malware* bekerja namun peralatan, waktu dan kemampuan yang dibutuhkan dalam menganalisa sangatlah berbeda.

Analisis Statis dilakukan dengan membongkar terhadap *source code* dari *malware* lalu mempelajari dan memahami melalui kode tersebut atau dengan kata lain proses analisis tidak memerlukan eksekusi terhadap *malware* (Moser, Kruegel, & Kirda, 2007; Tzermias et al., 2011). Berbeda dengan analisis dinamis yang pada proses analisisnya membutuhkan pengeksekusian terhadap contoh *malware* untuk kemudian dipelajari perilaku yang ditimbulkan oleh *malware* tersebut sehingga dapat diperoleh informasi tentang bagaimana sebuah *malware* tersebut bisa berkembang atau memanipulasi dirinya sendiri, dan pada komponen sistem apa saja *malware* tersebut berkomunikasi (Bayer, Kirda, & Kruegel, 2010; Bayer, Moser, Kruegel, & Kirda, 2006; Education, Science, Sujyothi, & Acharya, 2017; Egele, Scholte, Kirda, & Kruegel, 2012). Harapan setelah proses eksplorasi dilakukan semoga bisa memberikan pembelajaran tentang efek yang ditimbulkan oleh *malware* dan membantu praktisi dalam menemukan barang bukti digital.

2. TINJAUAN PUSTAKA

Poison Ivy RAT (*Remote Access Trojan*)

Poison Ivy RAT merupakan program yang dapat menghubungkan dan melakukan kontrol secara tersembunyi terhadap satu atau lebih perangkat komputer (FireEye, 2014). Aktifitas *Poison Ivy RAT* dilakukan melalui jaringan, baik itu jaringan *local* maupun jaringan *public* sehingga memungkinkan untuk dilakukan pada jarak yang jauh. *Poison Ivy RAT* menggunakan arsitektur *client server*. Dalam hal ini *server* adalah bagian program yang akan ditanamkan (*backdoor*) dan dijalankan pada perangkat korban yang didalamnya telah diberikan beberapa pengaturan seperti alamat *IP* dan *Port* agar dapat menghubungkan diri pada induk programnya (*calling home*).

Induk program yang dimaksud adalah dari sisi *client* yaitu bagian program yang dapat melakukan pengontrolan (perangkat *intruder*). Jika sebuah komputer korban telah terinfeksi oleh program *Poison Ivy RAT* ini maka seorang *intruder* dapat melakukan beberapa pengontrolan penuh antara lain seperti, mengakses *speaker* komputer, mengakses *webcam* untuk merekam *audio* maupun *video*, juga dapat digunakan untuk melakukan pencurian *password* dengan memanfaatkan fitur *Keystroke Logger (KeyLogger)*.

2.1 Metode *Malware* Analisis

Malware Analisis Dinamis

Pada metode ini sebuah *file* yang diperiksa akan diaktifkan dalam sebuah lingkungan yang *safe* baik pada sebuah mesin fisik yang telah disediakan sebagai laboratorium *malware* maupun yang berupa *virtual* (mesin *virtual*) untuk selanjutnya mampu dikumpulkan informasi mengenai dampaknya terhadap komputer ketika *file malware* menjalankan prosesnya. Sehingga dapat diketahui kegiatan apa saja yang dilakukan oleh *malware* saat berhasil menginfeksi sebuah komputer. Tahapan dalam analisis dinamis ini akan memeriksa komputer dengan secara keseluruhan seperti proses yang berjalan di komputer, perubahan *registry*, komunikasi internet dan peristiwa janggal lainnya yang memungkinkan terjadi ketika sebuah komputer telah terinfeksi oleh *malware.*

***Malware* Analisis Statis**

Tidak seperti pada metode *malware* analisis dinamis, dalam metode analisis statis ini *file malware* tidak akan diaktifkan secara langsung melainkan ditelusuri dan diteliti serta dianalisis terhadap kode sumber yang dituliskan didalam program *malware* dengan melakukan tahapan pembedahan terhadap program *malware* tersebut, sehingga informasi yang didapatkan sangatlah lengkap dan bisa memberikan gambaran yang sangat detail tentang mekanisme kerja *malware* tersebut secara keseluruhan. Dalam menggunakan metode *malware* analisis statis ini dituntut mampu memahami bahasa mesin terutama arsitektur sebuah program karena akan sangat membantu dalam menganalisis susunan kode-kode program *malware* terkait dengan mengumpulkan informasi dari perilaku yang ditimbulkan oleh *malware* tersebut.

3. METODE PENELITIAN

Tahapan Metode Malware Analisis Dinamis

Tahapan metode *malware* analisis dinamis dalam penelitian ini sebagai berikut :

1) Membangun Virtual Lab

Dalam menganalisa *malware* diperlukan sebuah lingkungan yang aman (*Virtual Lab*), dimana peneliti dapat dengan bebas melakukan analisa terhadap *malware*, tanpa harus khawatir *malware* tersebut akan menyebar dan menimbulkan kerusakan terhadap komputer. *Virtual Lab* yang dimaksud dalam penelitian ini adalah sebuah mesin virtual yang didalamnya sudah terinstal berbagai macam *tools* yang diperlukan untuk kegiatan analisa. Program untuk mesin *virtual* yang digunakan dalam penelitian ini adalah *Virtualbox*.

Pengaturan pada mesin *virtual* untuk kegiatan menganalisis *malware* meliputi sistem operasi yang digunakan serta seluruh konfigurasinya, termasuk pertimbangan untuk mampu terhubung dengan jaringan serta adanya sambungan dengan perangkat fisik seperti harddisk dan lainnya. Sistem Operasi yang akan digunakan dalam penelitian ini adalah *Windows XP* karena sangat mudah untuk

terinfeksi oleh *malware* sehingga sesuai untuk digunakan dalam kegiatan analisis *malware*. Lingkungan sistem operasi dikonfigurasi sedemikian rupa untuk mengakomodasi kegiatan analisis *malware*. Konfigurasi yang dimaksud adalah pengaturan terhadap sistem operasi yang dilakukan sesuai kebutuhan, dalam hal ini yaitu tidak dipasang program *antivirus* dan juga pertimbangan akan penggunaan *firewall*.

Dengan penggunaan *virtual lab* memungkinkan untuk kegiatan analisis *malware* dilakukan dilingkungan komputer seperti pada keadaan yang nyata namun dengan resiko yang hampir tidak ada karena mesin *virtual* telah diatur untuk tidak memberikan pengaruh terhadap komputer utama.

2) Menjalankan Malware

Dalam tahap ini dilakukan pengujian dengan menjalankan sampel *file malware* (*Poison Ivy*) pada *virtual lab*, sehingga dapat menghasilkan informasi mengenai perilaku apa saja yang dilakukan oleh *malware* terhadap sistem ketika *file* tersebut dijalankan.

3) Analisis Perilaku Malware

Dalam proses analisis akan diperiksa secara keseluruhan proses yang berjalan pada komputer seperti perubahan *registry*, aktivitas komunikasi jaringan dan peristiwa janggal lainnya yang terjadi ketika komputer telah terinfeksi oleh *malware*.

Proses analisis terhadap perubahan pada sistem *registry* menggunakan program pendukung *regshot*, yang mana dengan program *regshot* ini peneliti akan melakukan analisis pada sistem *registry* dengan cara membandingkan *snapshot* dari *registry* sebelum *malware* diaktifkan dan *snapshot* dari *registry* setelah program *malware* diaktifkan sehingga akan dapat diketahui perbedaan dan aktifitas apa saja yang telah dilakukan oleh *malware* terhadap perubahan sistem *registry*. Sedangkan wireshark dalam penelitian ini digunakan untuk menganalisa kinerja jaringan, tujuannya agar didapatkan informasi mengenai kemungkinan adanya indikasi yang ditimbulkan oleh perilaku *malware* terhadap sistem jaringan.

4) Analisis Malware Otomatis (Cuckoo Sandbox)

Untuk lebih menguatkan hasil dari temuan perilaku *malware* sebelumnya dimana file *malware* dijalankan pada *virtual lab,* maka pada tahap ini dilakukan analisis menggunakan program yang dapat melakukan analisis perilaku *malware* secara otomatis yaitu menggunakan *Cuckoo Sandbox,* program tersebut akan menyajikan informasi aktifitas terhadap *malware* yang sedang dianalisis antara lain seperti *file* apa saja yang dibuat *malware, file* apa saja yang dihapus *malware, file* apa saja yang diunduh *malware, a*ktifitas *malware* pada memori, dan trafik jaringan yang diakses *malware*.

Tahapan Metode Malware Analisis Statis

Tahapan metode *malware* analisis statis dalam penelitian ini sebagai berikut :

1) Ekstraksi File Malware

Pada tahap ini dilakukan ekstraksi terhadap file *malware* kedalam bentuk kode *String* menggunakan bantuan *program strings kali linux* (Official Kali Linux Documentation, 2013) untuk kemudian dapat dilakukan analisis terhadap kode-kode tersebut.

2) Analisis Perilaku Kode

Tujuan lebih lanjut dalam penelitian ini juga diharapkan dapat memberikan output berupa hasil pengujian apakah dapat dibuktikan bahwa *file* dari program *poison ivy* merupakan suatu *malware* atau

bukan, untuk itu dibutuhkan sentuhan teknik *Static Malware Analysis* (analisis statik) yang difokuskan pada pencarian dan analisis terhadap kode *string* yang mengandung perilaku ataupun ciri dari program *poison ivy* (Start, 2015).

3) Disassembler

Disassembler adalah program komputer yang dapat melakukan konversi terhadap bahasa mesin menjadi bahasa yang lebih mudah dipahami oleh manusia (Popa, 2012). Dengan *disassemble,* pada penelitian ini akan dilakukan analisis terhadap *malware* dan mencoba untuk memahami *malware* dengan menganalisis bahasa *assembly* dan mengumpulkan informasi dari program *malware* yang dapat digunakan untuk mengidentifikasi komponen maupun karakteristik *malware*.

Tahapan Hasil Analisa dan Pengujian

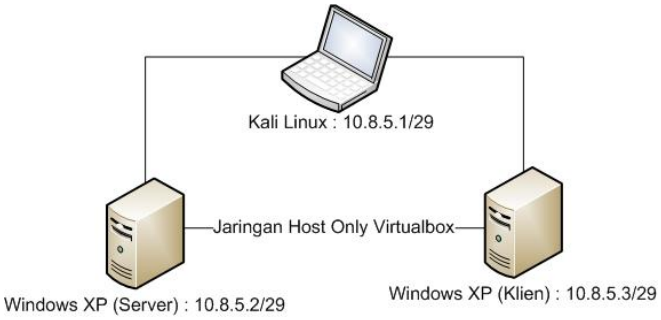
Tahap ini mengumpulkan hasil temuan dari tahapan pengujian dan analisis untuk kemudian dilakukan perbandingan terhadap informasi perilaku *malware*, baik yang didapatkan dengan cara mengeksekusi *malware* secara langsung (Analisis *Malware* Dinamis) maupun yang dilakukan dengan mengamati kode dari *file malware* (Analisis *Malware* Statis). Perbandingan yang dimaksud dalam penelitian ini bukan membandingkan kinerja dari kedua metode yang digunakan, melainkan mencari dan melakukan pembuktian terhadap kemiripan *output* yang dihasilkan oleh kedua metode tersebut sehingga dapat dipastikan kebenaran atas perilaku yang telah ditimbulkan oleh *malware*.

4. HASIL DAN PEMBAHASAN

Dalam menganalisis program *malware*, diperlukan tahap pengujian yang dapat digunakan sebagai acuan dalam menentukan karakteristik dan menggali informasi terkait dari perilaku yang akan ditimbulkan oleh program *malware* tersebut.

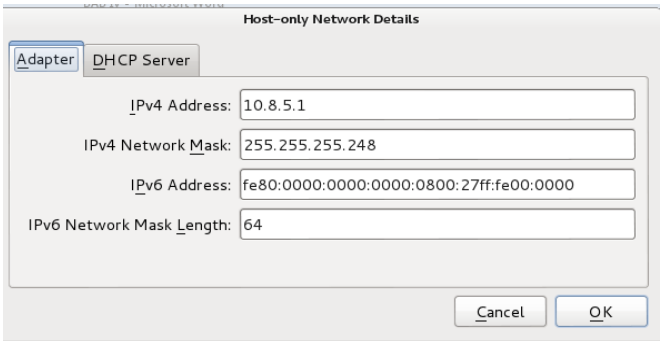
Pengujian dan Analisis Malware Dinamis

Analisis malware dinamis melakukan pengaturan alamat IP jaringan pada *virtual* lab yang akan dibangun. Gambar 1 menggambarkan topologi dalam arsitektur jaringan *virtual* lab.

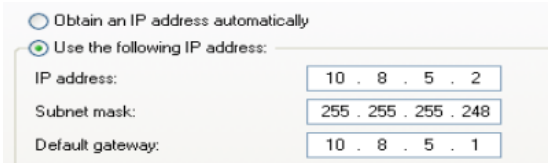


Gambar 1. Topologi Arsitektur Jaringan Virtual Lab

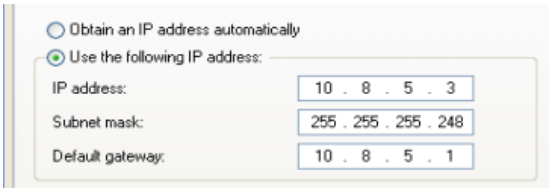
Pengaturan alamat IP pada *interface* kartu jaringan *host only*, yang mana dalam kebutuhan komputer utama (Kali Linux) dapat dilakukan pengisian alamat IP pada *interface vboxnet0*, sedangkan pada tiap *virtual lab* baik *server* maupun *client* dapat dilakukan pada menu *setting*, sub menu *network* dan *interface* diarahkan pada “Adapter1 - Host only adapter”, untuk kemudian dilakukan pengisian alamat IP pada saat *virtual* lab sudah dijalankan.



Gambar 2. Pengaturan Alamat IP *Vboxnet* (*Kali Linux*)



Gambar 3. Pengaturan Alamat IP Komputer Server

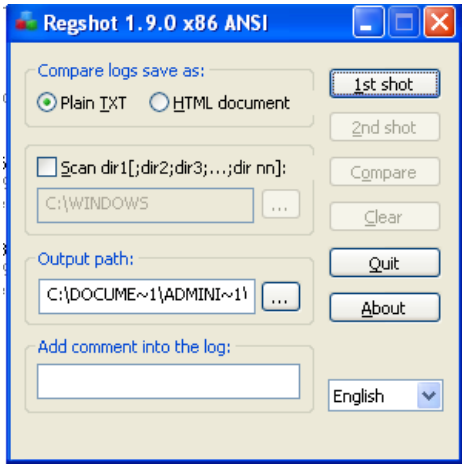


Gambar 4. Pengaturan Alamat IP Komputer Klien

Analisis Perilaku Menggunakan Regshot.

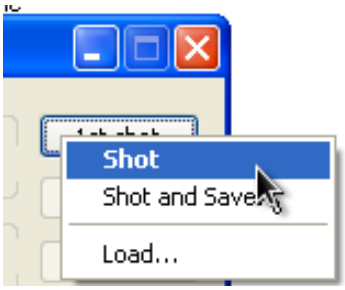
Setelah *Virtual Lab* berhasil dibangun, maka pada tahapan selanjutnya dapat dilakukan analisa langsung terhadap program *malware poison ivy,* pada langkah ini program *malware poison ivy* akan diaktifkan secara langsung pada *virtual lab* sehingga program *malware* akan mencoba untuk menginfeksi sistem. Namun sebagai langkah awal dalam tahap analisis ini diperlukan gambaran dari kondisi sistem pada saat dalam keadaan normal (belum terinfeksi) menggunakan alat pendukung *Regshot* (SourceForge, 2015).

Regshot bekerja dengan cara melakukan *snapshot* pada sistem *Windows* sebanyak dua kali. *Snapshot* yang pertama diambil sebelum *malware* diaktifkan pada sistem dan *snapshot* kedua diambil setelah *malware* diaktifkan dan berhasil menginfeksi sistem. Berikut adalah tampilan aplikasi regshot ketika berjalan pada *Windows XP*.



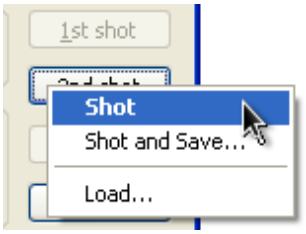
Gambar 5. Aplikasi *Regshot*

Sebelum menjalankan *malware* pada sistem *WindowsXP*, hal yang perlu dilakukan adalah melakukan pengambilan *snapshot* yang pertama dengan cara menekan tombol 1st*shot*, maka *regshot* akan memulai proses snapshot.



Gambar 6. *Menu Regshot* pada 1st shot.

Regshot akan mendata semua *file*, ketika proses *snapshot* yang pertama selesai maka untuk selanjutnya dapat dilakukan langkah mengaktifkan program *malware* sehingga program *malware* tersebut dapat melakukan beberapa perubahan terhadap sistem. Pada saat program *malware* telah melakukan perubahan maka diperlukan untuk mengambil *snapshot* yang kedua untuk mendapatkan informasi sistem apa saja yang telah berubah.



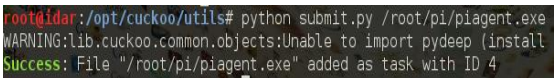
Gambar 7. Pengambilan shot kedua

Setelah pengambilan *shot* pertama dan *shot* kedua pada *regshot* maka dapat dilakukan komparasi data antara kedua *snapshot* yang telah dilakukan pada langkah sebelumnya dengan memanfaatkan *menu compare*.

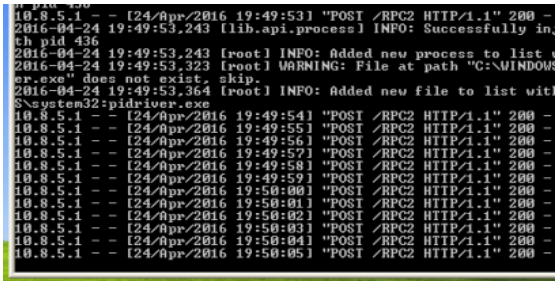
Gambar 8. *MenuCompare* pada *Regshot*.

Analisis Perilaku Menggunakan Cuckoo Sandbox

Dapat dilihat program dari sisi klien (*cuckoo sandbox agent*) mesin *cuckoo sandbox* (Cuckoo Sandbox, 2016)yang sedang melakukan analisis.



Gambar 9. Melakukan eksekusi program *malware* pada mesin *cuckoo sandbox*



Gambar 10. Analisis *Malware Cuckoo Sandbox Agent*

Perintah untuk mengaktifkan *servicebottle server* pada mesin *cuckoo sandbox* sehingga informasi dari hasil analisis dapat disajikan dalam bentuk *visual* berbasis *web*.

Gambar 11. Perintah Mengaktifkan *Bottle Server Cuckoo Sandbox*

Penjelaskan informasi detail *file* dari program *malware* yang telah dianalisis yaitu piagent.exe (*poison ivy*).

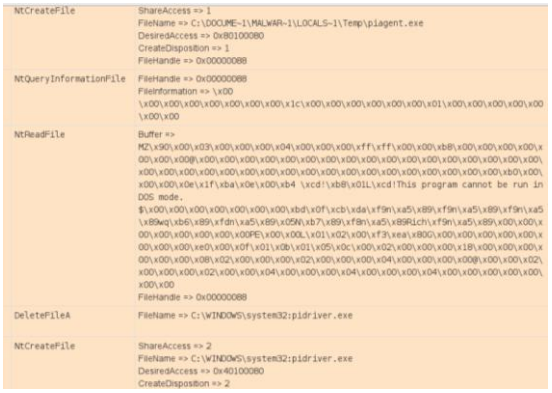
File name	piagent.exe
File size	7168 bytes
File type	PE32 executable (GUI) Intel 80386, for MS Wi
CRC32	2C15AF8B
MD5	e701b565cc14413f36f2a4ece7aae505
SHA1	8e1d35a4e3dec78d13e8059d315343daa01edef3
SHA256	a415c4b5cc25f98599a83fdab4dc276499ce6585401b
SHA512	a4163109b53ecfb0c691924c5537b2037639f5a7894a
Ssdeep	None
PEiD	None matched
Yara	None matched
VirusTotal	Permalink

Gambar 12. Informasi Detail *File* Program PIAGENT.EXE

Penjelasan kesimpulan perilaku yang dilakukan oleh program *malware poison ivy* terhadap mesin *cuckoo sandbox* diantaranya yaitu, membuat 2 *file* baru dengan nama “piagent.exe” dan sebuah *file* dengan nama “pidriver.exe”. Adapula perilaku lainnya program *malware poison ivy* mencoba menandai dirinya ketika aktif dalam memori dengan membuat *mutex* (*Mutual Exclusion*) bernama “**)!VoqA.I4**”. Dan perilaku yang lain dari program *malware poison ivy* yaitu melakukan penambahan *value key* pada “*startup*” dengan tujuan agar program dapat aktif pada saat komputer pertama kali melakukan *booting*.

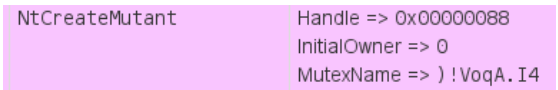
Gambar 13. Kesimpulan Perilaku Dari Program *Malware Poison Ivy* Terhadap Mesin *Cuckoo Sandbox*

Gambar 14 menjelaskan detail informasi yang berhasil dideteksi oleh mesin *cuckoo sandbox* terhadap perilaku program *malware poison ivy* dalam upaya pembuatan *file* "piagent.exe" dan *file* "pidriver.exe"



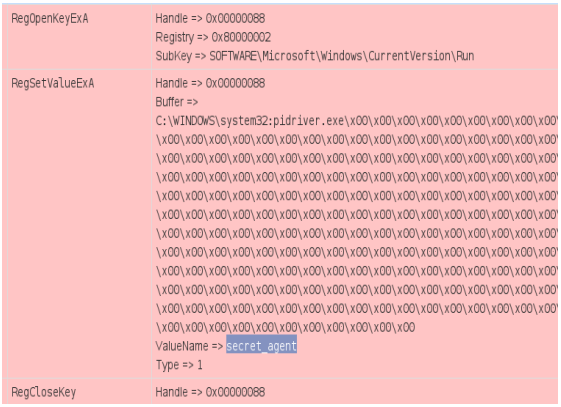
Gambar 14. Upaya pembuatan *File* yang Dilakukan Program *Poison Ivy*

Penjelaskan detail informasi yang berhasil dideteksi oleh mesin *cuckoo sandbox* terhadap perilaku program *malware poison ivy* dalam upaya pembuatan *Mutual Exclusion (Mutex).*



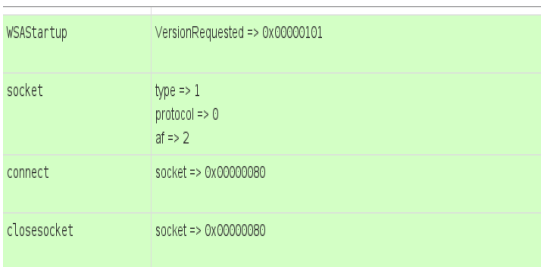
Gambar 15. Upaya Pembuatan *Mutual Exclusion* yang Dilakukan Program *Poison Ivy*

Pada Gambar 15 menjelaskan detail informasi yang berhasil dideteksi oleh mesin *cuckoo sandbox* terhadap perilaku program *malware poison ivy* dalam upaya penambahan *value key registry*, dimana program *malware* berusaha membuka subkey pada "SOFTWARE\Microsoft\Windows\Current Version\Run" dan upaya memberikan nilai terhadap *value* "secret_agent" dengan nilai "C:\Windows\system23:pidriver.exe", sehingga program *malware poison ivy* akan berjalan dalam *startup* komputer.



Gambar 16. Upaya Penambahan dan Perubahan *File Registry*

Gambar 17 menjelaskan detail informasi yang berhasil dideteksi oleh mesin *cuckoo sandbox* terhadap perilaku program *malware poison ivy* dalam upaya melakukan koneksi jaringan. Terlihat dimana program berusaha menyiapkan koneksi dengan memanggil instruksi *socket*.



Gambar 17. Upaya Koneksi Jaringan yang Dilakukan Program *Poison Ivy*

Analisis Paket Jaringan Program Malware Poison Ivy Menggunakan Wireshark.

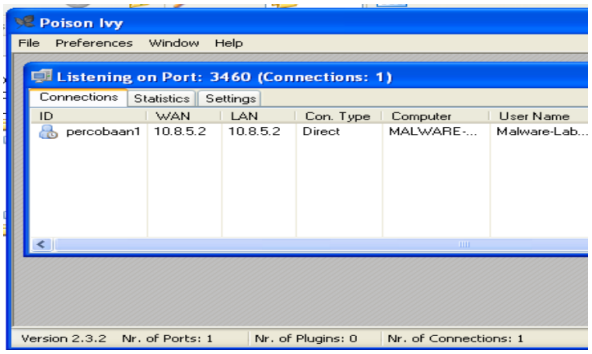
Pada tahap ini akan dilakukan dua kali pengujian langsung dengan

mengaktifkan program *malware poison ivy* terhadap dua perangkat komputer *virtual windows* yang telah dirancang sebelumnya, dimana pada sisi komputer *server* akan ditanamkan program *malware* yang dapat menginfeksi sistem serta menjadi pelayan (*service*) terhadap komputer klien yang melakukan *request,* tentunya pada komputer klien ini telah terinstal program *client malware poison ivy*. Kemudian dilakukan beberapa aktifitas sehingga dapat dianalisis paket data yang berjalan dalam jaringan dengan memanfaatkan program *wireshark* (Wireshark Org, 2016). Pada percobaan 1 akan dilakukan menggunakan *port default* dari program *malware poison ivy* yaitu *port 3460.*

Adapun pengaturan awal pada *wireshark* yaitu pemilihan kartu jaringan pada daftar *interface* program *wireshark*. Kartu jaringan yang akan dianalisis adalah *NIC (Network Interface Card)* dari mesin *virtual* yang terdeteksi dengan nama "*Local Area Connection 2*" seperti digambarkan pada gambar 18.

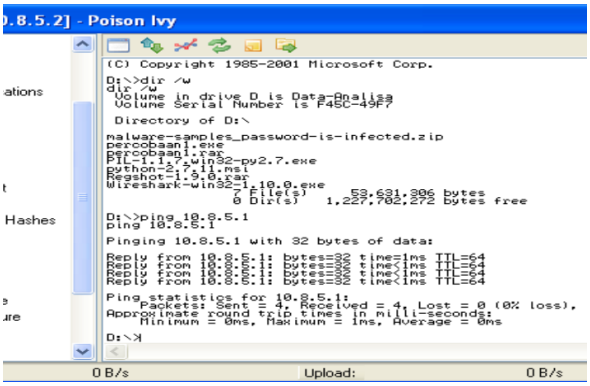
Gambar 18. Daftar Kartu Jaringan *Wireshark*.

Untuk memulai analisis paket jaringan menggunakan *wireshark* dapat digunakan tombol *start* pada *menu "capture > start"*, maka program *wireshark* akan mulai melakukan *sniffing* (merekam aktifitas jaringan) secara *real time*, dengan ini "percobaan1" dimulai. Disisi lain pada gambar 19 menggambarkan ketika komputer *server* (komputer yang tertanam *poison ivy*) berhasil terkoneksi dengan komputer klien (pengontrol) dengan nama id "percobaan1".



Gambar 19. Program *poison ivy* terkoneksi dengan program induk.

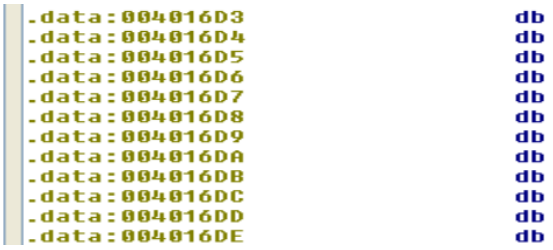
Pada gambar 19 menggambarkan aktifitas pengontrolan program *poison ivy* (*remote shell*) terhadap komputer *server* dengan melakukan *ping* pada alamat IP komputer utama (komputer *kali linux*) yang dilakukan oleh komputer klien agar, program *wireshark* dapat merekam informasi paket data yang dilalui selama aktifitas tersebut berlangsung seperti yang dapat dilihat pada Gambar 20.



Gambar 20. *Remote Shell* Komputer *Server* Oleh Komputer Klien

1) Pengujian dan Analisis Malware Statis

Ditemukan pendefinisian kode *string* "*secret_agent*" pada *offset* 004016D3 sampai *offset* 004016DE.



Gambar 21. Temuan Kode *String* "secret_agent" pada *IDA Pro*

Keterangan konversi dari gambar 21. :

- 5F (Hexa) = 95 (Decimal) = _ (ASCII)
- 61 (Hexa) = 97 (Decimal) = a (ASCII)
- 63 (Hexa) = 99 (Decimal) = c (ASCII)
- 65 (Hexa) = 101 (Decimal) = e (ASCII)
- 67 (Hexa) = 103 (Decimal) = g (ASCII)
- 6E (Hexa) = 110 (Decimal) = n (ASCII)
- 72 (Hexa) = 114 (Decimal) = r (ASCII)
- 73 (Hexa) = 115 (Decimal) = s (ASCII)
- 74 (Hexa) = 116 (Decimal) = t (ASCII)

Hasil Hexa : 73 65 63 72 65 74 5F 61 67 65 6E 74
Hasil Decimal : 115 101 99 114 101 116 95 97 103 101 110 116
Hasil ASCII : secret_agent

2) Hasil Temuan pada Pengujian dan Analisis

Tabel 1 dan Tabel 2 menampilkan hasil temuan secara keseluruhan dari pengujian dan analisis yang telah dilakukan, baik dengan teknik analisis dinamis maupun teknik analisis statis terhadap program *malware poison ivy*. Penyajian hasil temuan dilakukan bertujuan guna mendapatkan kebenaran informasi yang dihasilkan dari kedua teknik atau metode tersebut terkait perilaku program *malware Poison Ivy*.

Tabel 1. Hasil Temuan dari Pengujian dan Analisis Dinamis Program *Malware Poison Ivy*

No	Temuan	Analisis Dinamis		
		Regshot	Cuckoo Sandbox	Wireshark
1.	Penambahanregistry : hklm\software\microsoft\windows\currentversion\run\secret_agent	√	√	–
2.	Penambahan file prefetch : c:\windows\prefetch\piagent.exe-0aebfbee.pf	√	–	–
3.	Penambahan file baru : c:\docume~1\user\locals~1\temp\piagent.exe	–	√	–
4.	Penambahan filebaru : c:\windows\system32:pidriver.exe	–	√	–
5.	Alamat ip program induk (*controller*) :192.168.56.20	–	–	√
6.	Nomor port untukjalur komunikasi :3460	–	–	√
7.	Protokol yangdigunakan dalampengiriman paketdata :tcp (transmission control protocol)	–	–	√
8.	Kode string mutualexclusion(pembuatan mutex) :)!voqa.i4	–	√	–
9.	Kode string(nama identitas/ id) :pi_agent	–	–	–
10.	Kode stringpassword autentikasi :admin	–	–	–

Tabel 2. Hasil Temuan dari Pengujian dan Analisis Statis Program *Malware Poison Ivy*

No	Temuan	Analisis Statis	
		Strings	IDA Pro
1.	Penambahan registry : hklm\software\microsoft\windows\currentversion\run\secret_agent	√	√
2.	Penambahan file prefetch : c:\windows\prefetch\piagent.exe-0aebfbee.pf	–	–
3.	Penambahan file baru : c:\docume~1\user\locals~1\temp\piagent.exe	–	–
4.	Penambahan filebaru : c:\windows\system32:pidriver.exe	√	–
5.	Alamat ip program induk (*controller*) :192.168.56.20	√	√
6.	Nomor port untukjalur komunikasi :3460	–	–
7.	protokol yangdigunakan dalampengiriman paketdata :tcp (transmission control protocol)	–	–
8.	Kode string mutualexclusion(pembuatan mutex) :)!voqa.i4	√	–
9.	Kode string(nama identitas/ id) :pi_agent	√	√
10.	Kode string password autentikasi :admin	√	√

Keterangan : - (lambang *minus*) = Tidak ditemukan.

5. KESIMPULAN DAN SARAN

Kesimpulan

Berdasarkan pengujian dan analisis terhadap program *poison ivy* yang telah dilakukan maka dapat disimpulkan beberapa hal sebagai berikut :

1. Program *Poison Ivy* jelas dapat dikatakan sebagai *malware* karena mempunyai beberapa karakteristik dari program *malware* pada umumnya yaitu melakukan penambahan dan perubahan terhadap sistem (*WindowsRegistry* dan *file prefetch*) sebagaimana ditemukan seperti kode *string* **"secret_agent"** dan **"PIAGENT.EXE-0AEBFBEE.pf"** serta perilaku ketika program *poison ivy* diaktifkan tidak memberikan informasi maupun aktifitas secara kasat mata melainkan dalam perilakunya program *poison ivy* berupaya untuk menghubungkan diri pada program induknya yang dilakukan pada proses *background* (tidak kasat mata). Selain itu dari sisi klien (*controller*) program *poison ivy* dapat melakukan pengontrolan penuh terhadap komputer yang terinfeksi melalui komunikasi jaringan tanpa melakukan prosedur autentikasi secara *legal*.
2. Cara kerja program *poison ivy* dapat dianalisis menggunakan dua metode analisis *malware* yaitu metode analisis *malware* dinamis yang dapat memberikan solusi dalam menganalisis program *malware* yang terkendala pada bagian-bagian kode *signature* bersifat polimorfik maupun yang terenkripsi terkait pencarian perilaku dari program *malware*. Metode yang kedua adalah metode analisis *malware* statis dimana metode ini memungkinkan temuan informasi program *malware* melalui kode-kode *hexa* dan *string* ataupun *binary* yang terkandung didalamnya yang tidak dapat ditemukan jika dilakukan dengan metode analisis *malware* dinamis.

Saran

Beberapa saran yang diusulkan oleh penyusun untuk penelitian lebih lanjut sebagai berikut :

1. Kedua metode yang digunakan dalam penelitian ini, metode analisis *malware* statis merupakan model kajian yang paling sulit dilakukan karena sifatnya yang melibatkan proses melihat dan mempelajari isi program (*white box*) yang sedang dianalisis, untuk itu peneliti menyarankan untuk mempersiapkan strategi yang lebih mendalam pada kajian metode ini khususnya pada sumber daya manusia (SDM) yang harus memiliki pengetahuan dan pengalaman dalam membaca program berbahasa mesin (*assembly language*).
2. *Malware* merupakan topik yang masih sangat terbuka luas maka peneliti, dan juga menyarankan pengembangan teknik analisis program *malware* dengan memanfaatkan sub-teknik analisis statis yang dikenal dengan nama *Reverse Engineering*.

DAFTAR PUSTAKA

Bayer, U., Kirda, E., & Kruegel, C. (2010). Improving the efficiency of dynamic malware analysis. *Proceedings of the 2010 ACM Symposium on Applied Computing - SAC '10*, 1871. http://doi.org/10.1145/1774088.1774484

Bayer, U., Moser, A., Kruegel, C., & Kirda, E. (2006). Dynamic analysis of malicious code. *Journal in Computer Virology*, *2*(1), 67–77. http://doi.org/10.1007/s11416-006-0012-2

Cahyanto, T. (2015). BAUM-WELCH Algorithm Implementation For Knowing Data Characteristics Related Attacks On Web Server Log. *PROCEEDING IC-ITECHS 2014*. Retrieved from http://jurnal.stiki.ac.id/index.php/IC-ITECHS/article/view/131

Cahyanto, T. A., Oktavianto, H., & Royan, A. W. (2013). Analisis dan Implementasi Honeypot Menggunakan Dionaea Sebagai Penunjang Keamanan Jaringan. *JUSTINDO (Jurnal Sistem Dan*

Teknologi Informasi Indonesia), *1*(2), 86–92. Retrieved from http://jurnal.unmuhjember.ac.id/index.php/JUSTINDO/article/view/568

Cahyanto, T. A., & Prayudi, Y. (2014). Investigasi Forensika Pada Log Web Server untuk Menemukan Bukti Digital Terkait dengan Serangan Menggunakan Metode Hidden Markov Models. *Snati*, 15–19. Retrieved from http://jurnal.uii.ac.id/index.php/Snati/article/view/3280

Cuckoo Sandbox. (2016). Automated Malware Analysis - Cuckoo Sandbox. Retrieved July 31, 2017, from https://cuckoosandbox.org/

Education, I. J. M., Science, C., Sujyothi, A., & Acharya, S. (2017). Dynamic Malware Analysis and Detection in Virtual Environment, (March), 48–55. http://doi.org/10.5815/ijmecs.2017.03.06

Egele, M., Scholte, T., Kirda, E., & Kruegel, C. (2012). A survey on automated dynamic malware-analysis techniques and tools. *ACM Computing Surveys*, *44*(2), 1–42. http://doi.org/10.1145/2089125.2089126

FireEye, I. (2014). Poison Ivy: Assessing Damage and Extracting Intelligence, 33. Retrieved from https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/rpt-poison-ivy.pdf

Gandotra, E., Bansal, D., & Sofat, S. (2014). Malware Analysis and Classification: A Survey. *Journal of Information Security*, *5*(2), 56–64. http://doi.org/10.4236/jis.2014.52006

Kramer, S., & Bradfield, J. C. (2010). A general definition of malware. *Journal in Computer Virology*, *6*(2), 105–114. http://doi.org/10.1007/s11416-009-0137-1

Moser, A., Kruegel, C., & Kirda, E. (2007). Limits of static analysis for malware detection. *Proceedings - Annual Computer Security Applications Conference, ACSAC*, 421–430. http://doi.org/10.1109/ACSAC.2007.21

Official Kali Linux Documentation. (2013). Retrieved from https://www.kali.org/kali-linux-documentation/

Popa, M. (2012). Binary Code Disassembly for Reverse Engineering. *Journal of Mobile, Embedded and Distributed Systems*, *IV*(4), 233–248. Retrieved from http://jmeds.eu/index.php/jmeds/article/view/81

Sikorski, M., & Honig, A. (2013). Practical Malware Analysis. *No Starch*, *53*(9), 1689–1699. http://doi.org/10.1017/CBO9781107415324.004

SourceForge. (2015). Regshot download. Retrieved July 31, 2017, from https://sourceforge.net/projects/regshot/

Start, C. (2015). Project 11 : Poison Ivy Rootkit (15 points) What You Need for This Project. Retrieved July 31, 2017, from https://samsclass.info/123/proj10/p11-PI.htm

Tzermias, Z., Sykiotakis, G., Polychronakis, M., & Markatos, E. P. (2011). Combining static and dynamic analysis for the detection of malicious documents. *Proceedings of the Fourth European Workshop on System Security - EUROSEC '11*, 1–6. http://doi.org/10.1145/1972551.1972555

Wireshark Org. (2016). Wireshark · Download. Retrieved July 31, 2017, from https://www.wireshark.org/download.html