LAPORAN PENELITIAN

PERBANDINGAN DAMPAK *MALWARE STEALER REDLINE* DAN *TROJAN ZOMBIEBOY* DENGAN METODE ANALISIS *MALWARE* STATIS DAN DINAMIS

Peneliti:

Said Mahaqil Muhammad

NIM.190705004

Jenis Penelitian	**Penelitian Inter Disipliner**
Bidang Ilmu Kajian	**Cyber Security**
Dosen Peneliti	**Malahayati, M.T**

UNIVERSITAS ISLAM NEGERI AR-RANIRY BANDA ACEH

FAKULTAS SAINS DAN TEKNOLOGI

PRODI TEKNOLOGI INFORMASI

NOVEMBER 2023

LEMBARAN IDENTITAS DAN PENGESAHAN LAPORAN PENELITIAN PUSAT PENELITIAN DAN PENERBITAN LP2M UIN AR-RANIRY TAHUN 2023

1.	a. Judul	:	Perbandingan dampak *malware stealer redline* dan *trojan zombieboy* dengan metode analisis *malware* statis dan dinamis
	b. Jenis Penelitian	:	Penelitian Inter Disipliner
	c. No. Registrasi	:	-
	d. Bidang Ilmu yang diteliti	:	Cyber Security
2.	Peneliti		
	a. Nama Lengkap	:	Said Mahaqil Muhammad
	b. Jenis Kelamin	:	Laki laki
	c. NIM	:	190705004
	d. Fakultas/Prodi	:	Sains danTeknologi/Teknologi Informasi
	e. Anggota Peneliti 1		
	Nama Lengkap	:	Malahayati, M.T
	Jenis Kelamin	:	Perempuan
	Fakultas/Prodi	:	Sains danTeknologi/Teknologi Informasi
	f. Aggota Peneliti 2 *(Jika Ada)*		
	Nama Lengkap	:	Mulkan Fadhli, M.T
	Jenis Kelamin	:	Laki Laki
	Fakultas/Prodi	:	Sains danTeknologi/Teknologi Informasi
3.	Lokasi Kegiatan	:	Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh
4.	Jangka Waktu Pelaksanaan	:	6 (Enam) Bulan
5.	Tahun Pelaksanaan	:	2023
6.	Jumlah Anggaran Biaya	:	-
7.	Sumber Dana	:	Mandiri
8.	Output dan Outcome	:	-

Banda Aceh, 21 November 2023

Mengetahui,
Dosen Pembimbing I

Malahayati, M.T
NIP.198301272015032003

Pelaksana,

Said Mahaqil Muhammad
NIM. 190705004

Menyetujui:

Ketua Prodi. Teknologi Informasi

Ima Dwitawati

NIP. 198210132014032002

ABSTRAK

Nama	: Said Mahaqil Muhammad
NIM	190705004
Program Studi	: Teknologi Informasi
Judul	: Perbandingan Dampak Malware Stealer Redline Dan Trojan Zombieboy Dengan Metode Analisis Malware Statis Dan Dinamis
Jumlah Halaman	: 100 Halaman
Pembimbing I	: Malahayati, M.T
Pembimbing II	: Mulkan Fadhli, S.T., M.T
Kata Kunci	: Stealer Redline, Trojan Zombieboy, Analasis Statis, Analisis Dinamis

Malware stealer redline adalah *malware* yang biasa digunakan oleh penyerang untuk melakukan pencurian data kredential yang tersimpan di dalam perangkat lunak jenis *browser*. *Trojan zombieboy* tergolong ke dalam jenis *downloader* yang dapat melakukan *download* dan melakukan instalasi program lainnya, termasuk *malware* lain terhadap perangkat komputer tanpa disadari. Berdasarkan masalah tersebut, maka diperlukan analisis deteksi malware yang bertujuan untuk melihat malware activity yang dilakukan oleh malware ketika malware dieksekusi pada komputer. Penelitian ini menggunakan *tools byte histogram* dan *pestudio* untuk analisis statis serta *tool cuckoo sandbox* untuk proses analisis dinamis dan dijalankan pada suatu *environment* yang telah dirancang pada *virtual machine*, sehingga tidak ada risiko untuk terinfeksi *malware*. Penelitian ini dilakukan dengan menguji 2 sampel malware yang telah diunduh pada sistem *honeypot* Badan Siber Sandi Negara. Berdasarkan analisis yang dilakukan dengan perangkatlunak *byte histogram* diperoleh informasi tingkat pengacakan data *malware* dan *pestudio* didapatkan informasi umum *malware* seperti nilai *Hash*, *compiler-stamp*,*string*, *import*, *section*, tipe *file*, *library* dan informasi lainnya. Dalam analisis yang dilakukan dengan *tool cuckoo sandbox* terdapat pula hasil analisis kebiasaan dari malware, hasil yang diperoleh menjelaskan perilaku *malware* saat mengenfeksi perangkat komputer. *Malware stealer redline* memiliki perilaku

untuk menyembunyikan diri dengan membuat *file* atau *directory*, mengeksploit alat-alat utilitas *windows*, dan mengeksekusi perintah dan skrip melalui *power shell*, dan perilaku *malware trojan zombieboy* dapat mencuri informasi pribadi pada *browser*, mengambil alih kontrol perangkat, melakukan injeksi kode, dan melakukan upaya mencuri data kredential (*username* dan *password*).

Kata kunci: Stealer Redline, Trojan Zombieboy, Analisis Statis, Analisis Dinamis

KATA PENGANTAR

Alhamdulillahi rabbil ’aalamin, puji dan syukur kepada Allah SWT yang telah melimpahkan rahmat dan karunia-Nya, sehingga rampunglah sudah penulis susun tugas akhir ini, untuk melengkapi syarat-syarat dalam menyelesaikan pendidikan program S1 pada Fakultas Sains dan Teknologi Universitas Islam Negeri Ar-Raniry. Tulisan ini diberi judul “Perbandingan Dampak *Malware Stealer Redline* dan *Trojan Zombieboy* Dengan Metode Analisis *Malware* Statis dan Dinamis”

Dalam proses penyusunan tugas akhir ini, penulis banyak mendapat bimbingan dan arahan dari berbagai pihak, oleh karena itu selayaknya penulis mengucapkan terima kasih dan penghargaan yang setinggi-tingginya kepada:

1. Ayahanda Said Idrus Husein dan ibunda Nurhadisah tercinta yang telah mencurahkan kasih sayang, pengorbanan, doa yang tiada hentinya semenjak penulis dilahirkan hingga hari ini.
2. Saudara penulis Syarifah Aida san Said Muhammad Furqan, yang selalu setia memberikan dukungan dan mendoakan penulis selama penulis menjalani masa kuliah.
3. Ibu Malahayati, M.T dan bapak Mulkan Fadhli, S.T, M.T. selaku pembimbing telah bersedia meluangkan waktu, tenaga, pikiran serta memberikan arahan dan masukan yang sangat berguna dalam menyelesaikan tugas akhir.
4. Ibu Ima Dwitawati, MBA. dan bapak Khairan AR, M. Kom selaku ketua dan sekretaris Program Studi Teknologi Informasi Fakultas Sains dan Teknologi UIN Ar-Raniry.
5. Bapak Muhammad Iman Jaya, S.T selaku Ketua Bidang Persandian Dinas Komunikasi Informatika dan Persandian Aceh.
6. Bapak Bustami, M. Sc selaku Ketua Labaroratorium Program Studi Teknologi Informasi Fakultas Sains dan Teknologi UIN Ar-Raniry.
7. Ibu Cut Ida Rahmadiana, S.Si selaku staf Program Studi Teknologi Informasi Fakultas Sains dan Teknologi UIN Ar-Raniry.
8. Dekan Fakultas SAINTEK UIN Ar-Raniry, Bapak Dr. Ir. M. Dirhamsyah, M.T.

9. Sahabat terbaik saya Putri Nabila yang selalu mendukung dan mendoakan saya serta teman-teman angkatan 2019 telah memberikan masukan dan doanya yang sangat bermanfaat bagi penulis.
10. Akhirnya penulis mengucapkan terima kasih kepada semua pihak yang telah membantu penulis hingga terselesaikannya tugas akhir ini, baik secara langsung maupun tidak langsung yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari sepenuhnya bahwa tugas akhir ini masih jauh dari kesempurnaan, untuk itu penulis menerima saran dan kritikan yang bersifat membangun mutu tulisan ini demi perbaikan di masa yang akan datang. Akhir kata semoga penulisan ini dicatat sebagai amal shalih di sisi Allah SWT, yang bermanfaat bagi penulis dan para pembaca sekalian. Amin Ya Rabbal A'lamin.

Banda Aceh, 5 Desember 2023

Penulis

Said Mahaqil Muhammad

DAFTAR ISI

DAFTAR LAMPIRAN

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Balakang

Keamanan jaringan komputer sebagai bagian dari suatu sistem sangat penting untuk menjaga validitas data dan menjamin ketersediaan layanan bagi penggunanya. Untuk memperkuat keamanan jaringan komputer, sistem deteksi serangan dapat diimplementasikan dalam jaringan komputer untuk mendapatkan informasi anomali dalam jaringan. Tingkat keamanan yang rendah cenderung membuka peluang terjadinya berbagai tindak kriminal di internet, tindakan kriminal dapat dengan mudah dilakukan oleh *attacker* karena sistem keamanan yang diterapkan belum optimal, sehingga menjadi sasaran utama para *hacker* dalam melakukan serangan pada *server* untuk mendapatkan informasi yang diinginkan (Ramadhani et al., 2018).

Pada unggahan *DarkTracer* pada 7 april 2022, *DarkTracer* menyebutkan terdapat 878.319 dengan kredensial 34.714 sundomain pemerintah telah bocor dari pengguna yang terinfeksi *malware Stealer redline* (DarkTracer, 2022). Dari data tersebut, Badan Siber Sandi Negara (BSSN) melaporkan terdapat 233.813 kredensial dari 2.671 subdomain pemerintah Indonesia (go.id) yang mengalami pencurian data karena perangkat pengguna yang terkena oleh *malware stealer redline*. Jumlah subdomain yang terdapat itu tersebar di 490 domain pemerintah. Pada masa pandemi covid-19 BSSN mencatat serangan *cyber crime* sebanyak 88.414.296 sejak 1 Januari sampai 12 april 2020, puncak jumlah serangan terjadi pada 12 Maret 2020 yang mencapai 3.344.470 serangan dan setelah itu jumlah serangan mengalami penurunan yang signifikan di masa belakunya kebijakan*Work From Home* (WFH*).* Dalam masa WFH ini telah banyak terjadi berbagai jenis serangan siber dengan memanfaatkan isu covid-19, jenis serangan yangpaling banyak adalah *trojan activity* sebanyak 56% dan setelah itu disusul dengan aktifitas *information gathering* sebanyak 43% dari total keseluruh serangan, sedangkan sisa 1% serangan merupakan *web application attack*(Madina Nusrat, 2022).

Dampak serangan yang ditimbulkan *malware trojan zombieboy* dan *stealer redline* memiki beberpa perbedaan dari kegunaan *malware* tersebut. *Malware*

stealer redline digunakan oleh *hacker* untuk melakukan penyusupan yang mengambil informasi mengenai korban dari peramban, sistem pesan instan, dan klien protokol *tranfer file*. Target utama *malware stealer redline* adalah kata sandi, informasi kartu kredit, nama pengguna, lokasi, perangkat lunak, dan bahkan konfigurasi perankat keras seperti tata letak *keyboard*, pengaturan *User Account Control* (UAC), dll. Sedangkan dampak yang ditimbulkan *malware trojan zombieboy* sangatlah berbahaya, *malware* ini bisa memindai intranet, eksploitasi kerentanan *EternalBlue*, *DoublePulsar backdoor*, kendali jarak jauh dan pencurian mata uang kripto.

Analisis *malware* secara umum dilakukan menggunakan 3 metode yaitu metode statis, metode dinamis, dan metode *hybrid*. Metode yang digunakan pada penelitian ini adalah metode statis dan dinamis, penggabungan metode ini tidak banyak dilakukan oleh para peneliti karena luasya cakupan dan banyaknya *tool* yang digunakan serta membutuhkan waktu yang tidak sedikit, akan tetapi dilain sisi metode ini menghasilkan analisis yang sangat detil dan menyeluruh. Metode statis dilakukan dengan cara melakukan analisis *malware* secara langsung dengan skenario dalam sistem *virtual* yang telah terinfeksi oleh *malware trojan zombieboy* dan *stealer redline*, sedangkan metode dinamis dilakukan dengan menggunakan *tool cuckoo sandbox* sebagai alat analisis *malware trojan zombieboy* dan *stealer redline*.

Setelah proses analisis *malware* dilakukan diperlukan beberapa tindakan umum yang harus dilakukan. Jika pada proses analisis menunjukkan bahwa ada infeksi di dalam sistem, segera melakukan pembersihan yang komprehensif dengan memindai dan penghapusan *malware* dari semua sistem yang terkana dampak. Selanjutnya melacak aktivitas *malware* untuk memahami lebih dalam bagaimana *malware* bekerja dan apa yang ditargetkan oleh *malware* tersebut. Setelah itu dilakukan langkah untuk peningkatan keamanan sistem untuk menghindari infeksi dari *malware*, dalam hal ini perlu dilakukan pembaruan kebijakan dan praktik keamanan sistem agar *malware* tidak menginfeksi sistem, dan memastikan bahwa sistem dan perangkat lunak perangkat diperbarui secara teratur dan memiliki perlindungan yang memadai.

Oleh karena itu, penelitian ini berjudul "Perbandingan Dampak *Malware Stealer redline* Dan *Trojan zombieboy* Dengan Metode Analisis *Malware* Statis dan Dinamis", fokus pada penelitian ini adalah menganilisis *malware stealerredline* dan *trojan zombieboy* untuk mendapat informasi umum, kerusakan yang ditimbulkan, dan kemampuan cara kerja *malware* serta mengetahui bagaimana melakukan proses analisis *malware* secara statis dan dinamis. Fokus penelitian ini berguna untuk mengantisipasi infeksi jenis serangan *malware* yang sama dari sebelumnya dan menjadi tolak ukur untuk menjaga keamanan perangkat. Dalam penelitian ini menghasilkan pelaporan analisis yang dapat digunakan untuk menentukan tindakan yang harus dilakukan dalam mengevaluasi risiko kelemahankeamanan perangkat.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka rumusan masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana cara melakukan analisis *malware* metode statis menggunakan *tool byte histogram* dan *pestudio*?
2. Bagaimana cara melakukan analisis *malware* metode dinamis dengan menggunakan Cuckoo Sandbox?
3. Bagaimana cara kerja *malware* merusak perangkat dan ancaman yang ditimbulkan *malware*?

1.3 Tujuan Penelitian

Adapun tujuan penelitian ini berdasarkan latar belakang di atas, yaitu:

1. Mengetahui cara melakukan analisis *malware* metode statis menggunakan *tool byte histogram* dan *pestudio*.
2. Mengetahui cara melakukan analisis *malware* metode dinamis dengan Cuckoo Sandbox.
3. Mengetahui cara kerja *malware* dalam merusak perangkat dan mendapatkan informasi ancaman yang ditimbulkan *malware*.

1.4 Batasan Penelitian

Adapun batasan masalah pada penelitian ini adalah:

1. Penelitian ini merujuk pada analisis *malware* secara ststis dan dinamis
2. Melakukan analisis *malware* untuk mendapatkan informasi *malware* dandampak kerusakkan dari *malware*.

1.5 Manfaat Penelitian

Manfaat dari hasil penelitian ini diharapkan dapat memberikan kontribusi kepada beberapa pihak, anatar lain:

1. Bagi penulis, mendapatkan pengetahuan dan pemahaman tentang ilmu menganalisis *malware* serta pola serangan *malware*.
2. Bagi Instansi, sebagai acuan untuk bahan evaluasi analisis *malware* dan pembuatan modul analisis *malware* statis dan dinamis.
3. Bagi Universitas, sebagai kontribusi karya ilmiah dalam disiplin ilmu Teknologi Informasi dan sebagai referensi terhadap penelitian analisis *malware*selanjutnya.

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penelitian terdahulu yang relevan dengan penelitian yang dilakukan penulis sangat dibutuhkan sebagai referensi dalam mengembangkan penelitian yang akan dilakukan penulis. Penelitian terdahulu yang menjadi acuan pada penelitian ini yaitu dengan penggunaan analisis *malware* dinamis untuk proses analisis *malware.*

Pada penelitian "Analisis Deteksi *Malicius Activity* menggunakan Metode Analisis *Malware* Dinamis Berbabis Anomali" (Daniswara et al., 2019).Penelitian ini menganalisa 10 sampel program yang telah diunduh dan dijalankan di *environment,* dari 10 sampel program terdapat 4 yang teridentifikasi sebagai *malware,* dan 6 sampel lain tidak ter-identifikasi karena kurangnya informasi yang didapatkan dan tidak ada aktifitas yang mencurigakan ataupun *registry* yang anomali.

Pada penelitian "Analisis *Ransomware Wannacry* Menggunakan Aplikasi *Cuckoo sandbox*" (Wahyu Wahidin & Sari, 2022). Pada penelitian ini melakukan uji coba pada tiga jenis sampel *virus ransomware* yang sengaja meng-*install* beberapa *malware* di *malware library* yang tersedia pada beberapa website kemudian dijalankan dan dijalankan pada sistem operasi windows *virtual*. Hasil analisis yang diperoleh pada penelitian ini ialah informasi kebiasaan dan perilaku *ransomware* ketika dijalankan pada sistem operasi.

Pada penelitian "Analisis *Malware* Menggunakan Metode *DynamicAnalysis* Pada Jaringan Universitas Sam Ratulangi (virgiawan Arshad Manoppo, 2020)". Penelitian analisis *malware* yang dihasilkan pada penelitian ini menggunakan *cuckoo sandbox* adalah informasi *malware*, karakteristik dari *malware*, behaviour analysis, static analysis dan tingkat maliciousness *malware* berdasarkan hasil yang diperoleh dari *virus* total.

Pada penelitian "Dynamic *Malware Analysis Using Cuckoo sandbox*" (sainadh jamalpur, 2018). pada penelitian anilisis *malware* ini dilakukaninvestigasi perilaku *malware* dengan menjalankan kode berbahaya dari *malware*

dan mengamati perilaku *malware* seperti melakukan pengunduhan *file* dari internet, memodifikasi yang dilakukan pada item registri, menjalankan software, dan melakukan enkripsi setiap *file* pada sistem.

Pada penelitian "Analisis *Malware* Dengan Metode Dinamik Menggunakan Framework *Cuckoo sandbox* (Novansyah & Sutabri, 2023). Pada penelitian ini dilakukan analisis *malware* dengan menggunakan framework *cuckoo sandbox* yang dijalankan di sistem operasi windows 7 dan menggunakan sampel *malware* Kms-R@1n.exe yang diambil di Repositori *Malware* Online. Hasil yang diperoleh analisis *malware* dalam penelitian ini menunjukkan bahwa sampel *malware* Kms-S@1n.exe terdeteksi sebagai jenis *malware* trojan, serta informasi dari *malware*, *karakteristik malware*, *behaviour analysis*, *static analysis*dan tingkat *maliciousness malware.*

Tabel berikut ini adalah perbandingan pemelitian terkait, penjelasan lebih lanjut dapat dilihat pada Tabel 2.1.

Tabel 2. 1 Perbandingan penlitian sejenis

PENELITI	**METODE PENELITIAN**	**JUDUL PENELITIAN**	**HASIL PENELITIAN**
Daniswara, dkk, 2019	Analisis Dinamis Bebasis Anomali	Analisis Deteksi *Malicius Activity* menggunakan Metode Analisis *Malware* Dinamis Berbabis Anomali	Diperoleh 4 sampel *malware* dari 10 sampel *malware* yang dianalisis
Wahyu Wahidin & Sari, 2022	Analisis *Malware* Dinamis	Analisis *Ransomware Wannacry* Menggunakan Aplikasi *Cuckoo sandbox*	Diperoleh informasi perilaku dan kerusakan yang ditimbulakan pada sistem oleh *malware*
virgiawan dkk, 2020	Analisis *Malware* Dinamis	Analisis *Malware* Menggunakan Metode *Dynamic Analysis* Pada Jaringan Universitas Sam Ratulangi	Diperoleh karakteristik dan informasi *malware*
sainadh jamalpur, 2018)	Analisis *Malware* Dinamis	Dynamic *Malware* Analysis Using *Cuckoo sandbox*	Diperoleh karakteristik *malware* dan perubahan yang

			terjadi pada sistem
(Novansyah & Sutabri, 2023)	Analisis *Malware* Dinamik	Analisis *Malware* Dengan Metode Dinamik Menggunakan *Framework Cuckoo sandbox*	Hasil analisis yang diperoleh adalah jenis sampel *malware* yang dipakai ter-identifikasi sebagai jenis *malware* trojan, dan juga diperoleh informasi *malware*, karakteristik *malware*, perilaku *malware* dan behaviour *malware*.

2.2 Jaringan Komputer

Jaringan komputer merupakan jaringan telekomunikasi yang memungkinkan antar komputer untuk saling berkomunikasi dengan bertukar data, jaringan komputer dibangun dengan *hardware* dan *software*. Saat 2 atau komputer saling berkomunikasi atau bertukar data sebenarnya ada bagian dari jaringan komputer yang menjadi pihak yang menerima atau meminta layanan disebut dengan *client* dan yang mengirimkan disebut dengan *server*. Komputer yang saling terhubung ini pun harus mempunyai setidaknya 1 kartu jaringan masing- masing yang kemudian dihubungkan melalui kabel maupun nirkabel sebagai medium transmisi data dan terdapat perangkat lunak sistem operasi jaringan yang akan membantuk sebuah jaringan komputer sederhana (Komputer, 2020).

2.3 Keamanan Komputer

Keamanan komputer adalah keamanan informasi yang diterapkan pada komputer dan jaringan komputer yang bertujuan untuk membantu pengguna mencegah penipuan dan mendeteksi upaya penipuan pada sistem berbasis informasi. Menurut John D. Howard "keamanan komputer adalah tanggapan preventif terhadap serangan oleh pengguna komputer atau akses internet yang tidak bertanggung jawab". Ada empat prinsip dasar yang digunakan dalam mengamankan berbagai arsitektur jaringan. Berikut penjelasan mengenai prinsip-prinsip yang terkandung dalam keamanan jaringan:

- *Rule of least privilage*

  Prinsipnya adalah memberikan akses minimal kepada pengguna. Penggunaan sumber daya (*database*, *disk*, jaringan, dll.) harus dikelola sesuai kebutuhan, misalnya saat pengguna mengubah kueri untuk konten tabel A, aplikasi A tidak boleh diberikan izin penuh (Baca, Tulis, Jalankan) ke tabel A.

- *Defense in depth*

  Prinsip yang memberikan keamanan berlapis pada level atau titik jaringan. Misalnya sebuah *router* dapat mengimplementasikan *firewall, Intrusion Detection System (IDS*) dan jaringan harus memiliki *Authentication dan Authorization. Authentication* adalah kata sandi login yang harus diisiuntuk mendapatkan akses server, sedangkan *Authorization* adalah pembagian izin seperti *Read, Write* dan *Execute* yang harus diterapkan dengan benar. Keamanan harus diterapkan di beberapa titik untuk menjaga tingkat keamanan yang tinggi, jika ada implementasi keamanan yang gagal, masih ada aplikasi keamanan lainnya.

- *Seperation of duties*

  Seperation of duties merupakan pembagian tugas dan wewenang yangbegitu penting, orang yang mengurus bagian ini harus ahli dalam bidangnya. Dari waktu ke waktu harus ada rotasi personel yang hanya mampu menduduki posisi tertentu yang harus dilakukan *Security Policy Review* kembali.

- *Auditing*

  Auditing adalah segala sesuatu yang terjadi pada jaringan, fokusnya pada kejadian-kejadian penting seperti *server error*, *downtime*, *error network* dan lain-lain yang harus direkam untuk dianalisa ulang untuk mencegah kejadian yang sama di kemudian hari. Pada perangkat modern atau sistem operasi *modern*, perekaman otomatis tersedia secara *default* dengan mengaktifkan fitur *Authentication, Authorization, Accouting (AAA)* (linuxhackingid, 2023).

2.3 OSI Layer

OSI layer merupakan singkatan dari *Open System Interconnection* adalah model refernsi/standar komunikasi yang digunakan dalam komunikasi komputer. Dalam model *OSI* memungkinkan pertukaran informasi atau data yang terjadi antara barbagai jenis komunikasi komputer. Pada OSI layer memiliki tujuh layer yang dijabarkan dari layer teratas ke layer terbawah yang terdiri dari *application layer*, *presentation layer*, *session layer*, *tranport layer*, *network layer*, *data link layer*, dan *physical layer*.

a *Application Layer*

Lapisan aplikasi adalah tempat aplikasi jaringan dan protokol lapisan aplikasi berada, lapisan ini berfungsi sebagai antarmuka dengan aplikasi dengan fungsional jaringan dan kemudian membuat pesan-pesan kesalahan.

b *Presentation Layer*

Lapisan presentasi bertugas menentukan format adalah melakukan enkripsi data/ contohnya ketika melakukan *request* halaman web, datanya akan dibentuk dalam format *http-request* dan dienkripsi menjadi *https* menggunakan *SSL*/*TLS* sehingga data pada *web* tersebut memiliki keamanan.

c *Sesion Layer*

Lapisan ini mendefinisikan bagaimana komunikasi dimulai, dikontrol dan dihentikan. Pada session layer bertugas menjaga masing-masing koneksi supaya tetap terhubung dan data masuk tidak bertukar meskipun protokol sama dan masuknya juga bersamaan.

d *Transport Layer*

Lapisan *transport* ini bertugas untuk menyediakan koneksi *reliable* (*TCP* atau *Transmission Control Protocol*) dan *unreliable* (*UDP* atau *User Datagram Protocol*). *TCP* mempunyai koneksi reliable yang artinya koneksinya membutuhkan *acknowledgement*. Arti dari *acknowledgement* yakni indikasi yang melaporkan data diterima oleh penerima adalah data utuh. Di layer ini dapat terjadi kehilangan atau tertinggalnya data sehingga terjadi *error-recovery*, yakni ketika *request web* dan datanya ada yang

tertinggal, layer *transport* akan melakukan *request* ulang sampai datanya utuh untuk kemudian diteruskan ke *session layer*. Di situlah fungsi *acknowledgement*, yang akan melakukan *request* lagi sampai datanya diterima dengan sempurna. Sedangkan *Unreliable* tidak memerlukan *acknowledgement*.

e *Network Layer*

Lapisan ini bertugas melakukan pengalamatan dan melakukan *routing*. Bisa dianalogikan bahwa *layer* ini menentukan kemana data yang dibawa akan dikirim dengan proses *routing*.

f *Data Link Layer*

Lapisan ini bertugas menentukan aturan ketika perangkat mengirim data melalui media transmisinya, aturan disebut enkapsulasi. Perangkat data link layer menghubungkan perangkat dengan media transmisi, Contoh dari perangkat 2 yaitu *switch*, *bridge*, *network interface card* (*NIC)*.

g *Physical Layer*

Lapisan ini untuk mendefinisikan media transmisi yang digunakan pada jaringan komputer, metode pensinyalan, sinkronisasi *bit*, arsitektur jaringan, topologi 4 jaringan, dan pengkabelan. Pada layer ini data yang ditransmisikan dalam bentuk *bit* (WS Ibraheem, 2021).

2.4 Internet Protocol Address

Internet Protocol Address adalah untuk mengidentifikasi *interface* pada *host* suatu mesin atau perangkat, atau bilangan biner dengan ukuran 32 *bit* yang dibagi menjadi 4 bagian, setiap bagian terdiri dari 8 *bit*. Untuk memudahkan manusia membaca alamat *IP*, nama yang digunakan umumnya berupa angkadesimal. Tujuan dari *TCP*/*IP* adalah untuk membuat koneksi antar jaringan yang menyediakan layanan komunikasi antar jaringan yang memiliki banyak entitas fisik dan menghubungkan host yang berada pada jaringan yang berbeda.

2.5 Ethical Hacking

Ethical Hacking adalah kegiatan menembus sistem, jaringan, dan aplikasi dengan mengeksploitasi kerentanan dengan harapan mendapatkan hak akses dari sistem dan data, yang tujuannya adalah untuk membantu perusahaan menguji keamanan sistem dan jaringan mereka. *Ethical hacker* sangat dibutuhkan oleh perusahaan/instansi yang ingin menguji sistem untuk dieksploitasi dan menemukan kerentanan sehingga dapat ditemukan kerentanan yang berbahaya pada sistem (Muhyidin et al., 2022).

Ada 3 tipologi yang populer di kalangan komunitas hacker, yaitu *WhiteHat, BlackHat, dan GreyHat.*

- *WhiteHat*

  Whitehat adalah peretas etis yang melakukan prosedur *ethical hacking* untuk membantu pemilik sistem dalam upaya mereka mendeteksi dan memperbaiki kerentanan dalam sistem keamanan. Whitehat disebutsebagai peretas etis karena whitehat tidak melanggar hukum apa pun dalam melakukan prosedur serangan, meskipun mereka menggunakan banyak alat yang sama dengan peretas topi hitam.

- *BlackHat*

  Blackhat atau sering disebut cracker adalah *hacker* yang melakukan hacking dimotivasi oleh kepentingan pribadi yang mereka peroleh dari pelanggaran sistem komputer secara ilegal, meskipun mereka juga bisa menjadi pembuat kerusakan sosial yang berada di dalamnya untuk sensasi serangan, dendam pribadi, atau untuk mencari popularitas.

- *GreyHat*

  Greyhat dapat memiliki motivasi ideologis yang diterjemahkan menjadi serangan peretasan terhadap posisi politik musuh, kebijakan perusahaan yang tidak mereka setujui, atau bahkan negara-bangsa. Peretas *Greyhat* dapat menjadi topi putih di siang hari dan bekerja untuk organisasi dan pemilik sistem untuk mendeteksi kelemahan dalam sistem dan menguranginya, tetapi terkadang *greyhat* terlibat dalam aktivitas peretasan ideologis untuk memperbaiki kesalahan yang dirasakan(Joana Gaia, 2020).

2.6 Virtualbox

Oracle VirtualBox adalah fitur *virtual*isasi yang dapat dijalankan untuk mengeksekusi sistem operasi tambahan di dalam sistem operasi utama dan dapat digunakan layaknya komputer biasa. *Oracle VirtualBox* adalah salah satu perangkat *virtual*isasi yang lebih kuat untuk sistem operasi pada komputer denganprosesor *Integrated Electronics* (*Intel*) atau *Advanced Micro Devices* (*AMD*). Fitur perangkat lunak *Virtualbox* yang open-source atau sebagai alat gratis tersedia untuk pengguna, dan tersedia untuk pengguna tanpa batasan dasar apapun pada persyaratan jenis lisensi publik umum *GNU*. *Oracle VirtualBox* menghadirkan kemampuan komputer dengan prosesor x86, *VirtualBox* kaya fitur dan kinerja yang kompeten untuk pengguna perusahaan, dapat berjalan di *Windows*, *Mac*, *Linux*, *Solaris OS*. Kelebihan dari *virtualBox* adalah memberikan informasi yang mudah dan cepat antar *Virtual Machine*(*VM*), karena *software open source* cenderung cepat dalam memperbaiki masalah dan menambahkanfitur baru.

2.7 Malware

Malicious software atau disingkat *malware* adalah perangkat lunak yang dirancang untuk merusak sistem dan memanipulasi data tanpa sepengetahuan pemilik perangkat yang terinfeksi, *malware* dapat berupa script, konten aktif dan binari. Pada umumnya *malware* digunakan oleh *hacker* untuk merusak jalannya sebuah sistem operasi, mencuri data pribadi seseorang, melewati kontrol akses dan merusak sistem pada *host*.

Malware memiliki beberapa jenis dan memiliki banyak variasi, berikut beberapa jenis *malware* :

- Virus

  Virus adalah serangkain program yang diciptakan dengan tujuan untuk merusak komputer orang lain.

- *Ransomware*

  Ransomware adalah jenis *malware* yang, ketika diaktifkan, mengunci sistem operasi korban dan *file* sebagai tawanan dan data tidak dapat diakses hingga korban membayar uang tebusan yang telah ditentukan sebelumnya.

- *Rootkit*

  Rootkit adalah jenis *malware* yang memungkinkan peretas melakukan kontrol akses jarak jauh *Remote Access Trojan* (*RAT*) tanpa sepengetahuan korban.

- *Spyware*

  Spyware adalah jenis *malware* yang berguna untuk memata-matai aktivitas sistem korban tanpa adanya kecurigaan dari korban, serta mengambil informasi terkait keuangan seperti transaksi bank yang seharusnya tidak diketahui.

- *Trojan*

  Trojan adalah *malware* yang memiliki karakteristik menyembunyikan *malware* dalam sebuah *file* yang sekilas terlihat aman, ketika *file* tersebut dijalankan/diaktifkan *malware* tersebut akan memberikan akses jarak jauh kepada hacker.

- *Worm*

  Worm adalah *malware* yang menyebar melalui jaringan komputer dengan memanfaatkan celah pada sistem operasi.

- *Downloader*

  Downloader adalah jenis *malware* yang hanya beroperasi untuk *mendownload malware* lainnya

- *Backdoor*

  Backdoor adalah jenis *malware* yang secara otomatis memasang script sendiri di komputer dengan tujuan memberikan akses kepada hacker atau pembuat *malware* (Daniswara dkk, 2019).

2.8 *Tools*

Analisis *malware* adalah kegiatan melakukan pembedahan atau analisis terhadap *malware* untuk mengidentifikasi cara kerja *malware*, fungsional *malware*, serta cara mendeteksi dan mencegah yang paling efektif terhadap *malware*. Untuk melakukan analisis *malware*, ada dua cara untuk menganalisis prosesnya, yaitu analisis statis dan analisis dinamis (Daniswara dkk., 2019).

- Static *malware* analysis adalah melakukan analisis *malware* dengan tidak menjalankan atau tidak mengeksekusi software yang mengandung *malware*. Analisis statik dilakukan dengan menganalisis dengan melihat kode sumber perangkat lunak yang terinfeksi *malware*. Kode sumber dapatdiperoleh dengan memecahkan sampel *malware* kemudian mencartahi pola yang menarik di kode sumber, seperti keberadaan kode lama dan mencurigakan. Ada beberapa *tool*s yang digunakan dalam proses analisis *malware* statis, berikut beberapa *tool*s analisis *malware* statis:

    a. *IDA Pro*

    IDA Pro adalah disassembler dan debugger yang sangat kuat yang digunakan untuk menganalisis kode assembly *malware* dan memahami program berfungsi

    b. *PEview*

    Peviem digunakan untuk menganalisis berkas *Portable Executable* (*PE*) dan mengidentifikasi tanda-tanda *malware*, seperti *packing* atau enkripsi

    c. *Prosess Explorer*

    Prosess Explorer digunakan untuk memonitori dan menganalisis proses yang berjalan di sistem untuk mendeteksi aktivitas yang mencurigakan.

    d. *Wireshark*

    Dalam beberapa kasus analisis, analisis perlu mengamati lalu lintas jaringan untuk mendeteksi aktivitas *malware* yang berkomunikasi melalui jaringan.

    e. *PEStudio*

    Tool PEStudio memeriksa berkas *PE* untuk mendeteksi tanda- tanda keberadaan *malware* atau aktivitas mencurigakan lainnya.

    f. *Byte Histogram*

    Byte Histogran merupakan *tool* yang digunakan untuk mengalisis dan memvisualisasikan distribusi *byte* dalam berkas biner atau teks. Pada *tool byte hitogram* terdapat 2 bagian yaitu warna hijau dan warna merah, pada bagian hijau setiap piksel-kolom sesuai

dengan *byte code* percocokan posisinya dan memvisulisasikan jumlah kemunculan dalam sebuah bar vertikel. Bar hijau yang tinggi pada sisi sebelah kiri memberi informasi bahwa *code byte* 0h memiliki banyak kemunculan dan di sisi paling kanan *code byte* FFh. Bagian merah memiliki akar yang sama seperti bagian hijau, tetapi pada bagian merah diperoleh semua code *byte* yang mungkin dalam urutan yang menurun terkait kemunculannya (Christian Wojner, n.d.).

g. Regshot (Registry Snapshot Tool)

Regshot adalah *tool* yang digunakan untuk membuat *snapshot* atau foto dari database registri windows sebelum dan setelah sesuatu berubah.

h. Hacker process

Proses *hacker* tidak merujuk kepada alat tertentu, melainkan kepada aktivitas atau entitas yang mencoba melakukan peretasan atau akses tidak sah ke sistem atau jaringan komputer.

- Analisis *malware* dinamis mengamati perilaku *malware* selama eksekusi *malware* dan proses infeksi, untuk memperoleh perilaku sistem, pemanggilan fungsi, parameter fungsi, pelacakan aliran informasi, dan pelacakan instruksi (Wahyu Wahidin & Sari, 2022). Ada beberapa *tool* analisis *malware* yang banyak digunakan untuk analisis *malware* seperti *Any.run*, *VirusTotal*, Yara, dan *Cuckoo Sandbox*.

a. Any.run

Any.run merupaka *sandbox* analisis *malware* interaktif. Anyrun menyediakan antarmuka pengguna yang mudah digunakan dengan tampilan yang interaktif dan visualisasi *realtime* dari aktivitas *malware*, memungkinkan pengguna untuk melihat dan memahami dengan lebih baik perilaku *malware*. *Anyrun* sepenuhnya berjalan pada *cloud* yang berarti tidak ada proses instalasi lokal yang diperlukan.

b. *Virustotal*

Virustotal merupakan platform yang menyediakan analisis *file* dan *URL* melalui berbagai mesin *antivirus, sandbox,* dan sumber data lainnya, Ini memberikan visibilitas yang luas dan memungkinkan pengguna untuk melihat hasil dari banyak sumber secara sekaligus. Dalam *virus*total hasil pemindaian yang diperoleh dari beberapa mesin anti*virus* dan *sandboxes* ditampilkan dalam waktu yang sangat singkat.

c. *Cuckoo Sanbbox*

Cuckoo sandbox adalah sistem analisis *malware* otomatis canggih, dengan modul yang sangat luas dan *open source* dengan aplikasi tak terbatas. Sistem analisis ter*virtual*isasi ini dapat memisahkan mesin nyata dari mesin *virtual* dan membuat jaringan terpisah dari jaringan nyata untuk melakukan analisis *malware* dengan lebihaman dan memfasilitasi analitik. Dimana analisis dapat mengamati aktivitas jahat dari *malware* sehingga analis dapat mengetahui cara kerja *malware* secara lebih instan tanpa mengganggu perangkat fisik pengguna. Karena sifat cuckoo *open-source* dan desain modularnya yang ekstensif, para analisis dapat menyesuaikan segala aspek lingkungan analitis, pemerosesan hasil analisis, dan tahapan pelaporan (Claudio Guarnieri, 2019).

2.9 Hash

Hash adalah kode alfanumerik dengan panjang tetap yang diguanakanuntuk mewakili kata, pesan, atau data. Fungsi *Hash* adalah fungsi apapun yang dapat digunakan untuk memetakan data dengan ukuran ar*bit*rer ke nilai ukuran tetap. Fungsi *Hash* yang perlu diketahui adalah *SHA1*, SHA256, dan MD5, berikut penjelasan dari ketiga *Hash* tersebut

1. *SHA1*

SHA1 (Secure *Hash* Algorithm) adalah salah satu fungsi hasg yang nilai *Hash* yang dihasilkann 160 *bit. SHA1* dikembangkan pada tahun 1993 oleh United States *National Scurity Agency* (*NIST*). *SHA*-1 sudah dianggapaman sejak tahun 2005. *Vulnerability* yang ditemukan pada *SHA*-1 adalah

collision attack, namun dengan kecepatan penemuan *collision* yang tinggi. Oleh karena itu, banyak perusahaan teknologi besar yang kemudian menghentikan penggunaan *SHA*-1 dari sistem mereka demi menjaga keamanan data mereka

2. *SHA*-256

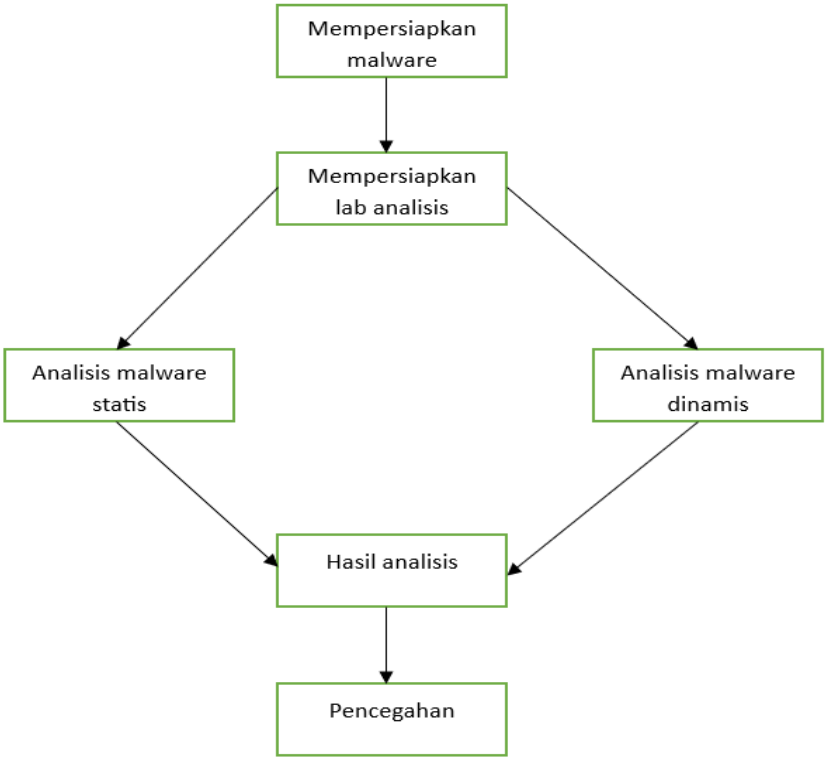
   Selanjutnya, karena masalah keamanan yang ditemukan pada *SHA*-1, dikembangkanlah *SHA*-2 oleh United States *National Security Agency* (*NSA*) pada tahun 2001. *SHA*-2 berbeda secara signifikan dari pendahulunya yaitu *SHA*-1. *SHA*-2 bisa disebut sebagai 'keluarga' karena memiliki fungsi *Hash* turunannya yang memiliki nilai *Hash* yang berbeda- beda; *SHA*-256 adalah keluarga *SHA*-2 yang memiliki nilai *Hash* 256. Saatini, *SHA*-256 adalah satu satu fungsi *Hash* yang masih dianggap paling aman, terbukti hingga saat ini pemerintah Amerika Serikat masih menggunakan *SHA*-256 pada standar keamanannya.

3. *MD5*

   MD5 adalah fungsi *Hash* yang memiliki nilai *Hash* 128 *bit*. *MD5* dikembangkan pada tahun 1991 oleh Ronald Rivest dan memiliki pendahulu *MD5*. Namun, tidak lama kemudian, yaitu pada tahun 1996, ditemukan bahwa *MD5* memiliki vulnerability terhadap collision attack sehingga sudah dianggap tidak aman. Meski begitu, hingga saat ini, *MD5* masih cukup digunakan pada beberapa kasus (Studi Sistem dan Teknologi Informasi, n.d.).

2.10 Kerangka Berpikir Penelitian

Kerangka berpikir adalah kerangka yang menggambarkan alur logika yang digunakan penulis dalam mengembangkan penelitian. Berikut adalah kerangka berpikir pada penelitian ini.



Gambar 2. 1 Bagan Kerangka Berpikir

Berdasarkan bagan kerangka berpikir tahapan yang dilakukan pada penelitian ini yaitu 1). Mempersipkan sampel malware, 2). Mempersiapkan lab analisis, 3.) Analisis malwre statis, 4.) Analisis malware dinamis, 5). Hasil analisis, dan 6). Pencegahan.

Tabel 2. 2 Kerangka berpikir

Tahapan Awal	Analisis *malware* dinamis adalah proses menganalisis *malware* di lingkungan ter*virtual*isasi secara runtime dan juga merupakan bagian penting dari program keamanan apa pun. analisis inidapat membantu mengidentifikasi kode berbahaya yang mungkin ada di sistem dan dapat digunakan untuk menyelidiki dan memahami perilaku *malware*. Banyak kasus kejahatan dunia maya yang terjadi saat ini dan penyebaran *malware* yang mengintai pengguna internet tanpa mengenal tempat dan waktu. Banyak kasus kejahatan dunia maya yang telah terdata oleh BSSN, sehingga diperlukan alternatif untuk mengungkap atau

	mengetahui jenis *malware*, karakteristiknya, dan ancaman yang dapat dilakukan *malware*. Sebelum memulai analisis, penting untuk membuat catatan rinci tentang lingkungan dan perubahan yang dilakukan padanya. Ini akan membantu dalam memahami perilaku *malware*, serta melacak setiap perubahan yang dilakukan. Lingkungan sandbox *virtual* harus dibuat untuk mengisolasi *malware* dari sistem utama. Ini akan membantu mencegah kerusakan yang dapat disebabkan oleh kode berbahaya. Sistem harus dipantau untuk setiap perubahan yang dibuat oleh *malware* seperti mengenkripsi *file*, menjalankan perangkat lunak, dan perilaku lainnya. Analisis perilaku harus dilakukan untuk mengidentifikasi bagaimana *malware* berperilaku di dalam sistem.
Proses	Dalam proses analisis awal, fitur yang perlu disiapkan ialah perangkat lunak pendukung analisis seperti sistem operasi *ISO Windows*, *virtualbox*, dan sampel *malware*. Setelah semua perangkat lunak ter-*install* dan siap dijalankan, perlu dilakukan pengaturan konfigurasi *virtualbox*, langkah ini dilakukan untuk menghindari penyebaran *malware* ke sistem utama dan menjaga agar jaringan tetap tersedia di lingkungan *virtual*. Pada tahap selanjutnya, install perangkat lunak *PeStudio*, *ByteHistogram*, *cuckoo sandbox* sebagai alat analisis statis dan dinamis serta semua lingkungan pendukung lainnya untuk melakukan tahap analisis. Sampel *malware* yang telah disiapkan akan dijalankan ke lingkungan *cuckoo sandbox*, *cuckoo sandbox* secara otomatis menganalisis *malware* dengan cepat dan memberikan informasi terperinci mengenai *malware*. Setelah proses analisis statis dan dinamis telah selesai dijalankan informasi *malware* akan rangkum untuk dipelajari lebih lanjut.
Tahapan akhir	Setelah proses analisis selesai dijalankan, semua informasi yang diperoleh akan dirangkum untuk mengidentifikasi aktivitas yang

	terjadi, perilaku yang dihasilkan, dan tingkat ancaman *malware*. Selain itu, pihaknya juga menentukan penanggulangan terhadap *malware* untuk mengantisipasi serangan *malware* serupa yang mungkin terjadi di masa mendatang. Dengan langkah analisis *malware* ini sangat perlu dilakukan di setiap institusi untuk menjaga perangkat dan meningkatkan keamanan perangkat.

2.12 Hipotesis Penelitian

Adapun hipotesis penelitian ini adalah sebagai berikut:

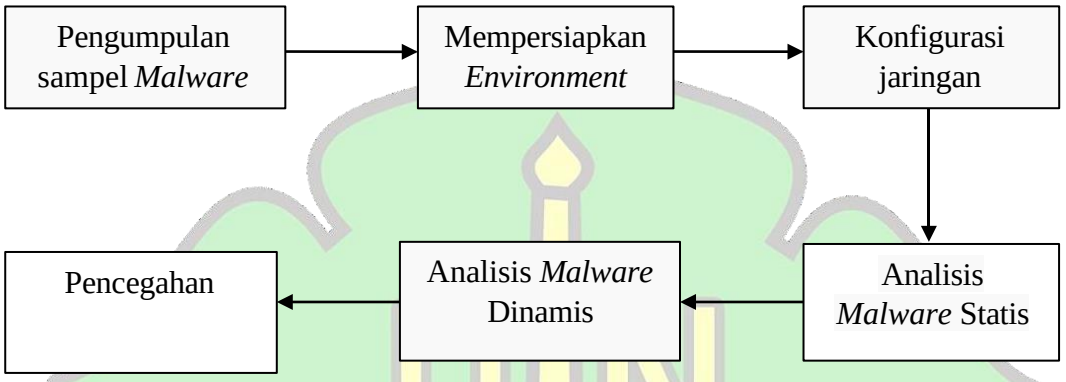
H.0 Analisis *malware* statis dan dinamis dapat menentukan ancaman yang ditimbukan *malware* terhadap sistem.

H.1 Analisis *malware* statis dan dinamis tidak dapat menentukan ancamanyang ditimbukan *malware* terhadap sistem.

BAB III
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Adapun tahapan penelitian ini selengkapnya dapat dilihat pada Gambar 3.1.



Gambar 3. 1 Blok diagram analisis malware

Berdasarkan Gambar 3.1 alur analisis *malware* yaitu: 1) Pengumpulan sampel *malware*, 2) Mempersiapkan *environment*, 3) Konfigurasi jaringan, 4) Analisis *malware* statis, 5) Analisis *malware* dinamis, dan 6) pencegahan, untuk penjelasan selengkapnya dapat lihat dibawah ini

3.1.1 Pengumpulan Sampel *Malware*

Tahapan pertama pada penelitian ini adalah mengumpulkan dan menyiapkan sampel *malware* sebagai bahan utama analisis, sampel *malware* diambil dari serangan *cyber* yang tertangkap pada sistem *honeynet* BSSN yang berada di Diskominsa Aceh. *Malware* yang terperangkap pada sistem *honeynet* akan di-*download* untuk mempersipkan kebutuhan penelitian sebelum melakukan analisis.

3.1.2 Mempersiapkan *Environment*

Tahapan selanjutnya adalah mempersiapkan *environment* analisis*malware*, dalam melakukan analisis *malware* diperlukan persiapan *environment* agar pelaksanaan analisis *malware* dapat berjalan sesuai dengan tujuan dan tidak menyalahi standar yang telah ditetapkan. Berikut ini adalah persiapan *environment* sebelum melakukan analisis *malware*:

1. Instalasi mesin *virtual* untuk laboratorium analisis *malware*.
2. Memastikan mesin *virtual victim* dan *host* berfungsi dengan baik. Mesin *virtual victim* digunakan sebagai mesin *sandbox* untuk menjalankan *malware* yang dianalisis, sedangkan mesin *virtual host* digunakan sebagai mesin yang menyediakan *network* ke mesin *victim.*
3. Menjalankan kedua mesin *virtual* dan pastikan *network* saling terhubung antar mesin.
4. Memastikan kedua mesin telah dilakukan *snapshot* untuk kondisi *clean* dan siap untuk melakukan analisis.

3.1.3 Konfigurasi Jaringan *Virtualbox*

Tahapan selanjutnya adalah mengatur konfigurasi mesin *virtual victim* dan mesin *Host* untuk mengantisipasi penyebaran *malware* yang dapat berpindah dari mesin *virtual* ke mesin fisik. Langkah ini diperlukan untuk melihara *malware* tetap berada di area *virtual* serta menjaga agar *malware* tidak memiliki akses penyebaran melalui koneksi internet dan lalu lintas jaringan tetap tersedia di lingkungan labotarium analisis *malware* yang terisolasi. Berikut adalah beberapa yang perlu diperhatikan menyiapkan lab *virtual*:

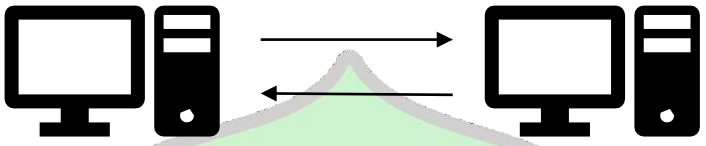
1. Memperbarui mesin *virtual*. hal ini diperlukan untuk kemungkinan adanya *malware* yang dapat mengeksploitasi kerentanan dalam perangkat lunak *virtual*iasi, keluar dari lingkungan *virtual*, dan menginfeksi sistem host.
2. Menerapkan *mode* konfigurasi jaringan *host-only* agar *malware* tidak terhubung ke internet saat analisis *malware*.
3. Tidak menyambungkan media *removeable* yang mungkin akan digunakan pada mesin fisik, seperpi *USB drive*..

a. Arsitektur Lab

Arsitektur lab yang akan digunakan terdiri dari mesin *windows* analisis untuk menjalankan segala proses analisis *malware* statis dan dinamis dalam penelitian ini dan mesin *windows victim* yang berguna untuk mengaktifkan *malware* yang bertujuan melihat aktivitas yang dapat dilakukan malware dan melihat koneksi yang berkomunikasi dengan malware ini.

b. *Virtual* mesin

Gambar 3.2 menunjukkan contoh arsitektur lab sederhana yang akan digunakan. Dalam lab ini, mesin anlisisis akan dikonfigurasi ke alamat IP 192.168.1.10, dan alamat IP mesin *virtual victim* akan dikonfigurasikan ke 10.0.2.15, berikut gambaran mesin *virtual*.



Host : IP. 192.168.1.10 **Mesin *virtual victim* : IP. 10.0.2.15**

Gambar 3. 2 Gambar arsitektur mesin *virtual*

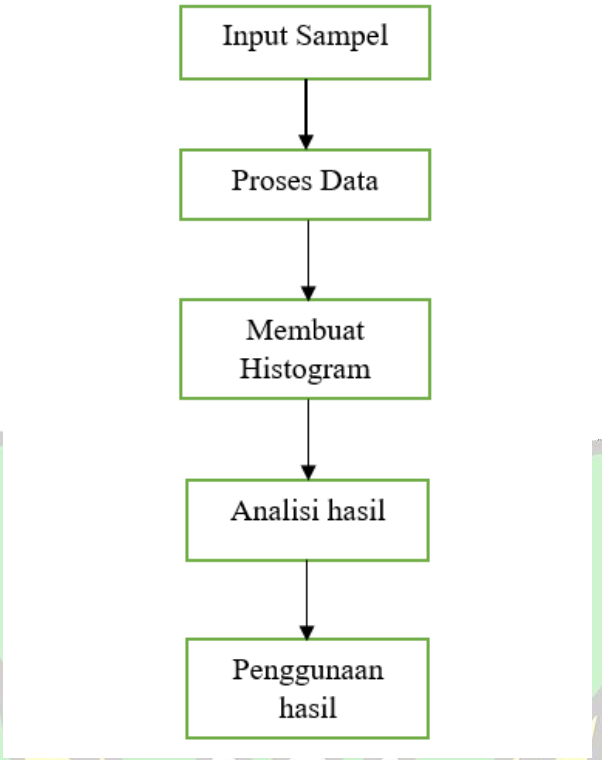
3.1.4 Analisis *Malware* Statis

Analisis *malware* statis ini dilakukan menggunakan *tool PeStudio* dan *Byte Histogram*. *Tool PeStudio* digunakan untuk mendapatkan informasi umum *malware* seperi *Hash*, *libraries*, *string*, *import* dan *signatures* dan *tool ByteHistogram* digunakan untuk menghasilkan grafik dari tingkat pengacakkan data dari *malware*. Sebenarnya masih banyak *tools* analisis statis yang dapat digunakan namun pada penelitian statis hanya menggunakan 2 *tool*s ini sudah sangat mumpuni untuk digunakan dalam proses penelitian yang berguna untuk mendapatkan informasi artefac *malware*. Berikut ini adalah alur cara kerja dari *tool ByteHistogram* dan *Pestudio*.

1. *ByteHistogram*

Adapun cara kerja dari *tool bytehistogram* dapat dilihat pada Gambar 3.3.

- *Input* Data



Gambar 3. 3 cara kerja *Byte*Histogram

Langkah pertama adalah memberikan data yang akan dianalisis. Data ini dapat berupa *file* teks, data biner, atau data dalam format lain.

- Pemrosesan Data

Setelah mendapatkan data, alat "*bytehistogram*" akan memprosesnya *byte* per *byte*. Ini melibatkan membaca data *byte* demi *byte* dan mencatat jumlah kemunculan setiap *byte* atau karakter.

- Membuat *Histogram*

Setiap *byte* atau karakter yang ditemukan akan dicatat dalam histogram. Histogram adalah representasi grafis dari distribusi *byte* dalam data. Ini dapat berbentuk grafik batang, tabel, atau format visual lainnya, yang menunjukkan berapa kali setiap *byte* atau karakter muncul dalam data.

- Analisis Hasil

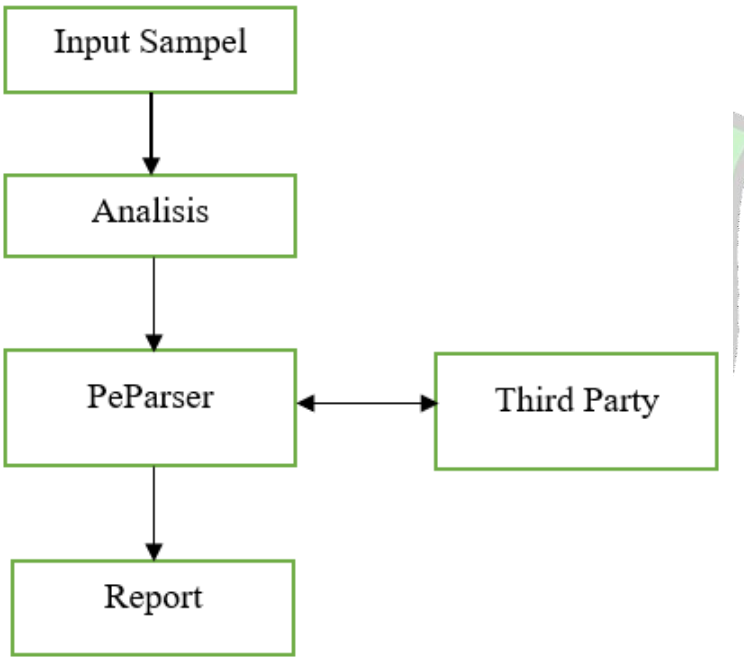
Setelah *histogram* dibuat, pengguna dapat menganalisis distribusi *byte* atau karakter dalam data. Ini bisa memberikan wawasan tentang struktur data, kemungkinan keberadaan pola tertentu, atau keparan karakteristik data.

- Penggunaan Hasil

Hasil analisis histogram *byte* dapat digunakan untuk berbagai tujuan, seperti pemrosesan data lebih lanjut, deteksi anomali, atau pemahaman lebihdalam tentang isi data.

2. *PeStudio*

Adapun cara kerja dari *tool* PeStudio dapat dilihat pada Gambar 3.4.



Gambar 3. 4 Cara kerja PeStudio

- Input Sampel

Pestudio adalah alat analisis *malware* yang digunakan untuk menganalisis berkas eksekusi, seperti berkas dengan ekstensi *.exe*. Alat ini memerlukan berkas yang akan dianalisis sebagai input.

- Analisis

Pestudio melakukan analisis statis pada berkas input. Ini berarti alat ini memeriksa berkas tersebut tanpa menjalankannya. Selama analisis statis, *pestudio*

memeriksa berbagai atribut berkas, seperti informasi metadata, tanda tangan digital, entropi, dan lainnya.

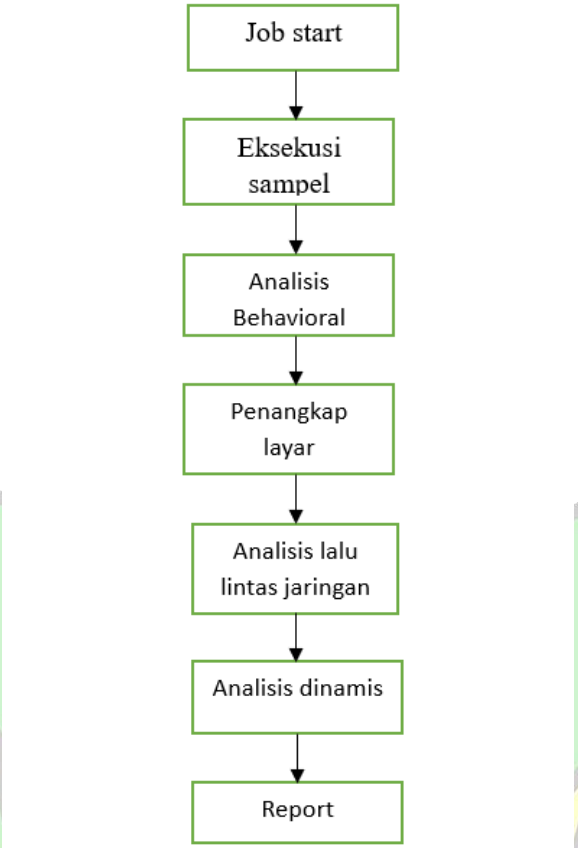
- *PeParser*

PeParser adalah komponen di dalam *PeStudio* yang bertanggung jawab untuk mengurai (parse) *file biner Windows*. Parser ini memiliki fungsi khusus untuk menganalisis struktur internal dari berkas tersebut. Beberapa fungsi utama *PeParser* meliputi:

- Mengidentifikasi Struktur: PeParser dapat mengidentifikasi dan mengurai struktur internal *file*, termasuk bagian seperti *header*, *section headers*, *import*/*export tables*, dan sebagainya.
- Analisis Kode dan Data: Menyelidiki dan mengklasifikasikan bagian-bagian kode dan data dalam *file* biner.
- Deteksi Ancaman Keamanan: Memeriksa berkas untuk tanda-tanda potensial atau karakteristik yang terkait dengan *malware* atau ancaman keamanan lainnya.
- Informasi Eksplisit: Memberikan informasi rinci tentang berkas, seperti versi, dependensi eksternal, dan sebagainya.
- Visualisasi Struktur: Memberikan visualisasi struktur *file* biner untuk mempermudah pemahaman.

3.1.5 Analisis *Malware* Dinamis

Analisis *malware* dinamis ini dilakukan menggunakan *tool Cuckoo sandbox*, alat ini memungkinkan para analisis untuk mengeksekusi sampel *malware* di lingkungan yang terkendali dan memantau perilakunya yang dapat digunakan untuk mengidentifikasi kemampuan dan tujuan *malware*. Berikut ini adalah alur cara kerja dari *cukcoo sandbox* dapat dilihat pada Gambar 3.5.



Gambar 3. 5 cara kerja *cuckoo sandbox*

- Eksekusi sampel

Malware dieksekusi dalam lingkungan terisolasi. *Cuckoo* memantau dan merekam semua aktivitas yang terjadi selama eksekusi berjalan.

- Analisis *Behavioral*

Cuckoo menganalisis perilaku *malware* selama eksekusi, ini mencakup pemantauan aktivitas seperti penciptaan dan penghapusan *file*, perubahan registri, panggilan sistem, dan aktivitas jaringan.

- Penangkapan layar

Cuckoo dapat memotret layar pada titik-titik tertentu selama eksekusi, memberikan pemahaman visual tentang aktivitas yang terjadi.

- Analisis lalu lintas jaringan

Cuckoo menganalisis lalu lintas jaringan yang dihasilkan oleh *malware*. Ini mencakup pemantauan koneksi ke server eksternal, permintaan *HTTP*, dan aktivitas jaringan lainnya.

- Analisis dinamis

Analisis ini mencakup pemantauan tindakan *malware* saat berinteraksi dengan sistem, mencoba mengidentifikasi taktik, teknik, dan prosedur (*TTP*) yang digunakan oleh *malware*.

- *Report*

Setelah analisis selesai, *Cuckoo* menghasilkan laporan yang merinci temuan analisis. Laporan ini dapat mencakup informasi seperti daftar proses, *file* yang dibuat, aktivitas registri, panggilan sistem, dan lalu lintas jaringan.

Penggunaan *tool cuckoo sandbox* yang efektif untuk melakukan analisis *malware* dan memberikan informasi yang sangat komplik dari *malware*. *Cuckoo sandbox* juga dapat memberikan informasi berharga tentang teknik dan taktikyang digunakan oleh penyerang untuk menghindari deteksi. Ada beberapa *tool* analisis *malware* yang banyak digunakan untuk analisis *malware* seperti *Any.run*, *VirusTotal*, *Yara*, dan *Cuckoo Sandbox*. Pada penelitian ini menggunakan cuckoo sandbox sebagai *tool* analisis *malware* dinamis karena *tool* ini memiliki kemampuan yang luar biasa dalam melakukan analisis *malware*. *Cuckoo sandbox* yang merupakan proyek *open source* yang memungkinkan *user* untuk mengunduh, meng-*install* dan mengkonfirmasi sendiri *sandbox*-nya serta memiliki komunitas yang luas yang bisa membantu pengguna untuk mengakses lebih sumber daya, pengetahuan, dan dukungan yang tersedia.

3.1.6 Pencegahan

Langkah selanjutnya adalah melakukan pencegahan untuk mengantisipasi terhadap serangan *malware* yang bisa saja terjadi kepada perangkat. Langkah ini adalah proses untuk mencegah terjadinya injeksi *malware malware* ke dalam perangkat serta untuk memberikan masukkan dalam melindungi dari serangan *malware* lanjutan.

3.2 Alat dan Bahan

Alat yang dibutuhkan untuk melakukan penelitian ini terdiri dari perangkat keras dan perangkat lunak. Analisis alat dan kebutuhan sistem yang diperlukan untuk penelitian ini meliputi:

3.2.1 Perangkat keras

Perangkat keras yang digunakan adalah satu unit leptop Lenovo Ideapad Gaming 3 15ARH05 dengan spesifikasi perangkat keras dalam melakukan analisis *malware* dinamis tercantum pada tabel 3.1.

Tabel 3. 1 Spesifikasi perangkat keras yang digunakan

Komponen	**Spesifikasi**
Processor	AMD Ryzen 5 4600H with Radeon Grapics
RAM	8 GB DDR4 3200 MHz single channel
Strorage	512GB SSD M.2
Graphic card	NVIDIA GeForce GTX 3050i

3.2.2 Perangkat lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah sistem operasi *Microsoft Windows 11 Home Single Language Version 22h2* dan menggunakan beberapa *tools*, yaitu *virtualbox, pestudio, byte histogram, cuckoo sandbox, microsoft windows 10,* dan sampel *malware. Tools* tersebut akan dipakai penulis untuk melakukan proses analisis *malware* statis dan dinamis dapat dilihat pada Tabel 3.2 dan Tabel 3.3.

Tabel 3. 2 perangkat lunak yang digunakan

Perangkat Lunak	**Versi**
Microsoft Windows 11	22h2
Virtualbox	7.0.6
Pestudio	9.20.0
ByteHistogram	1.0.0
Cuckoo Sandbox	2.0.7
Microsoft Windows 10	21H1

Tabel 3. 3 Sampel Malware

Sampel Malware	***Family***
Trojan Zombieboy	*Trojan*
Stealer Redline	*Stealer*

BAB IV
HASIL DAN PEMBAHASAN

Hasil dan pembahasan pada penelitian ini meliputi hasil analisis *malware* Stearler Redline dan *Trojan zombieboy* menggunakan metode analisi *malware* statis dan dinamis. Adapun tahapan yang digunakan yaitu persiapan, hasil analisis *malware* statis, hasil analisis *malware* dinamis, dan penanganan.

4.1 Hasil Analisis Statis

Hasil dari analisis *malware* statis telah dilakukan dan didapatkan dari hasil analisis statis pertama adalah struktur dari *malware stealer redline* dan *trojan zombieboy* seperti:

- Nilai *Hash*
- *Compiler-Stamp*
- *Section*
- Tipe *File*

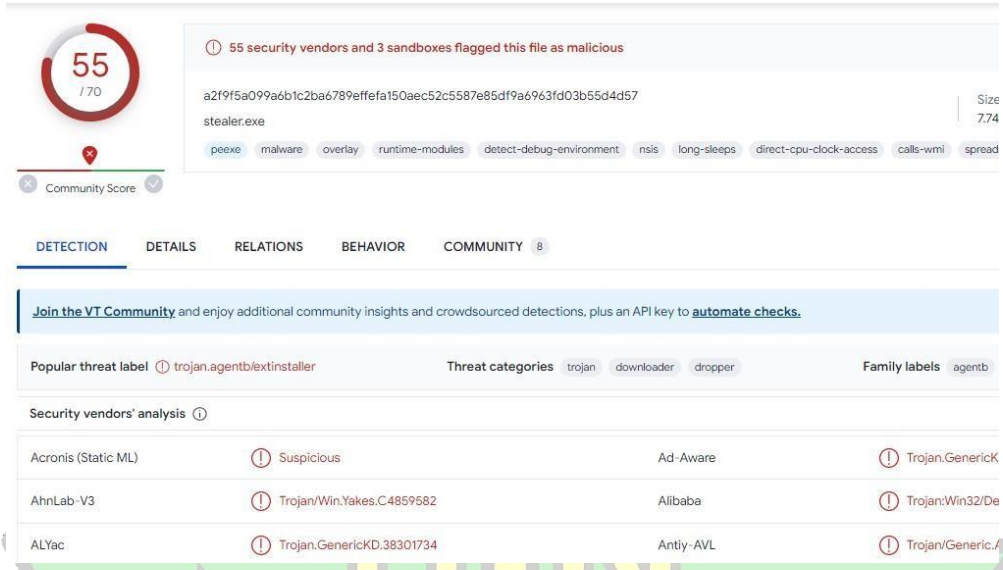
Yang telah didapatkan dengan *tool* seperti yang dijelaskan pada tabel diatas. Lalu setelah tabel ringakasan akan menunjukkan *string*, *import* dan *library* yang digunakan oleh *malware stealer redline* dan trojan zombie boy untuk menginfeksi komputer. Informasi string, *import*, dan libary diperoleh dengan *tool* PEStudio.

Hasil analisis statis dilakukan untuk mengerti apa saja yang dapat dilakukan oleh *malware* sebelum menganalisis *stealer redline* dan trojan zombieboy dengan metode dinamis. Dengan didapatkan gambaran bagaimana malware ini akan berperilaku pada saat proses infeksi.

Sebelum melakukan analisis statis diperlukan pengecekan pada setiap sampel, untuk memastikan bahwa sampel yang dianalisis adalah adalah *file* berbahaya dengan begitu berikut adalah gambar-gamnbar yang menunjukkan bahwa *file* yang dianilisis adalah *file* berbahaya:

A. *Stealer redline*

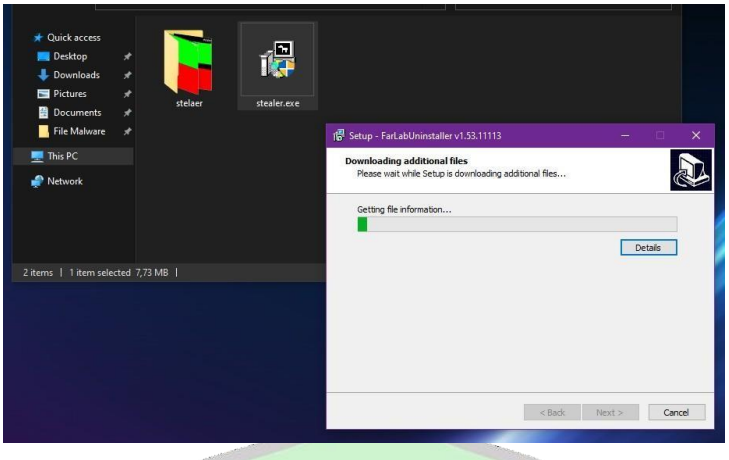
Hasil analisis *virus* total *malware stealer redline* menggunakan nilai *Hash* 0af9c941d86c3914df0d442d51536bd8.



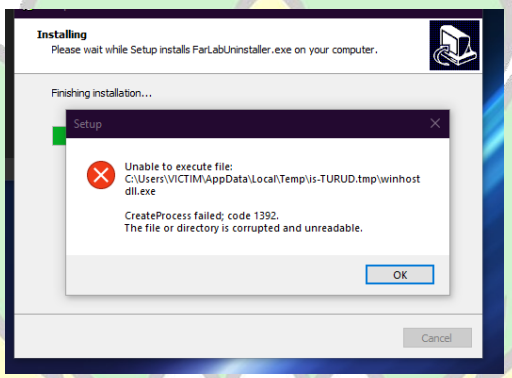
Gambar 4. 1 pengecekkan sampal pada *file stealer redline*

Berdasarkan Gambar 4.1 menunjukkan hasil bahwa bahwa malware *stealer redline* tergolong ke dalam trojan,agentb/extinstaller, beberapa *antivirus* mendeteksi bahwa *malware stealer redline* ke dalam beberapa jenis *malware*, seperti *acronis* mendektis bahwa malware ini (*Suspicious*), *Alibaba* mendeteksi (*Trojan*:*Win*/De), *Ad-Aware* medeteksi (*Trojan*.*Generick*) dan masih banyak lagi antivirus lainnya.

Sebelum melakukan analisis statis dan dinamis peneliti mencoba menjalankan malware *stealer redline* pada operasi sistem *victim* serta menjalankann *tool wireshark* untuk mendapatkan informasi lainnya.

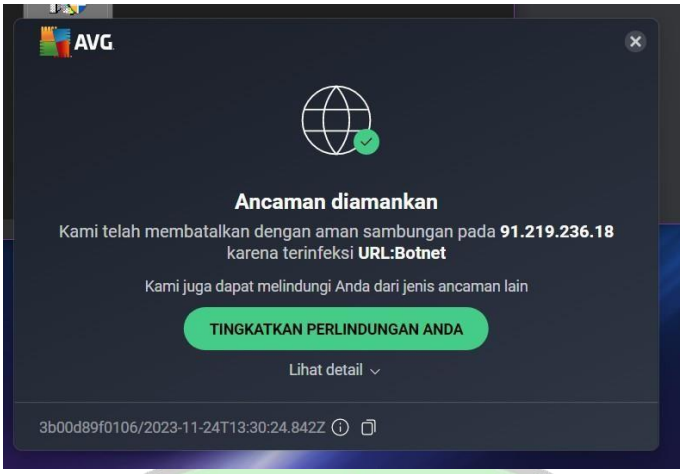


Gambar 4. 2 Menjalankan *malware stealer redline*



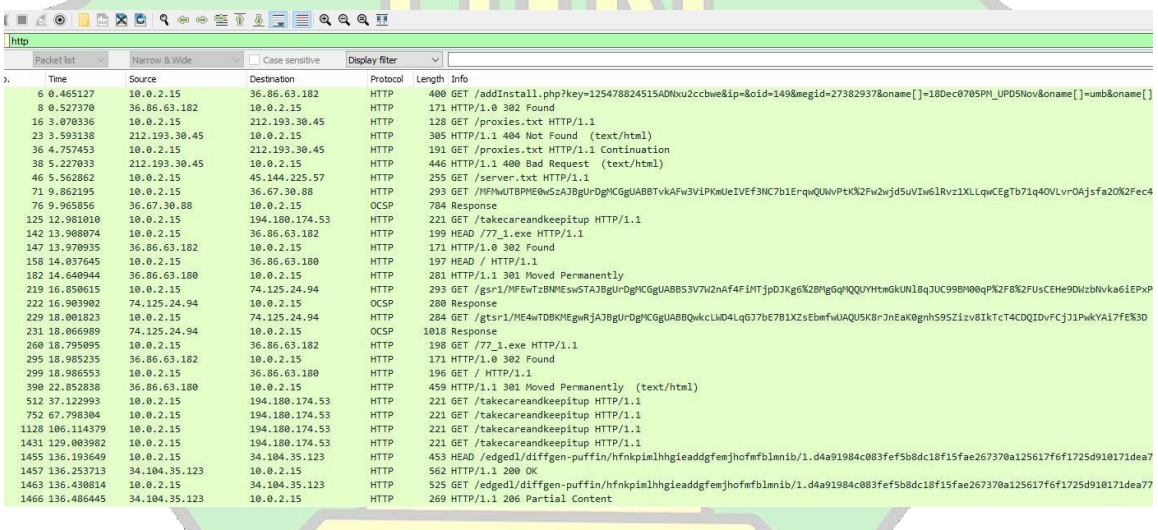
Gambar 4. 3 File stealer redline gagal dijalankan

Pada Gambar 4.2 proses *downloading* sedang berjalan pada operasi sistem *victim*, namun babarapa saat setelah dijalankan *file* ini gagal untuk di-*download* karena *file* tersebut *corrupted*. Pada Gambar 4.4 *antivirus* pada perangkat komputer merespon ancaman yang mungkin terjadi dengan membatalkan sambungan *internet* ke alamat *ip* 91.219.246.18 yang terhubung dengan *malware* ini, antivirus juga mendeteksi bahwa malware ini terinfeksi dengan *Botnet.*

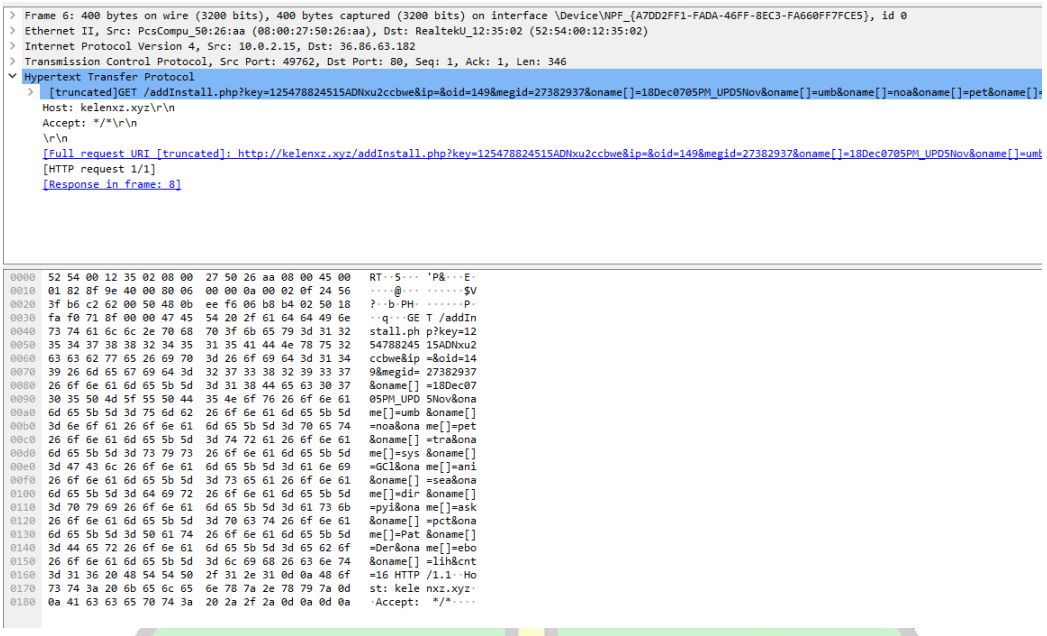


Gambar 4. 4 Respon *antivirus*

Pada tangkapan hasil *wireshark* menunjukkan beberapa alamayt *ip* yang berinteraksi dengan malware *stealer redline*, hasil *capture wireshark* dapar dilihat pada Gambar 4.5.



Gambar 4. 5 capture tool wireshark



Gambar 4. 6 Capture tool wireshark

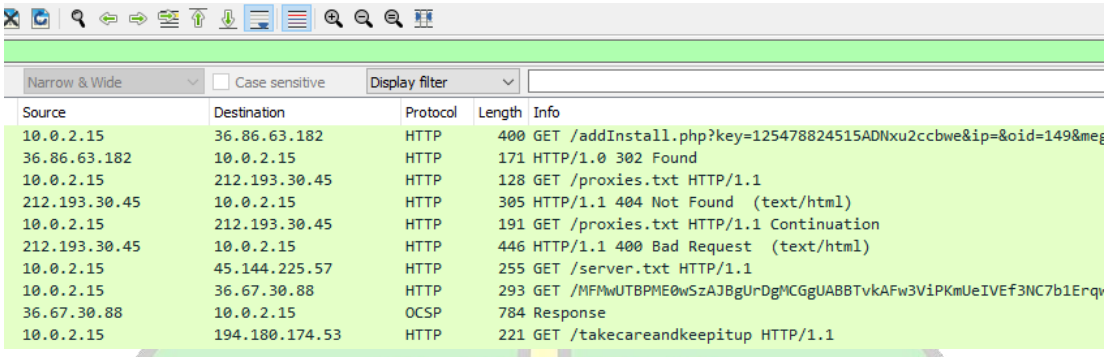
Hasil Gambar 4.5 diperoleh beberapa *ip address* yang melakukan koneksi dengan malware *stealer redline* dan pada Gambar 4.6 memperlihatkan informasi lainnya serta domain dari *ip address* tersebut. Hasil *ip address* yang diperoleh *wireshark* dan deteksi *antivirus* akan dimasukkan ke dalam Tabel 2.1 berikut.

Tabel 4. 1 Ip address yang berkomunikasi dengan *stealer redline*

Nama Domain	Ip Address	Negera
no-hostname.serverastra.com	91.219.236.18	Hungaria
tue.volcanicthree.net	212.193.30.45	Belanda
-	45.144.225.57	United States
-	36.67.30.88	Indonesia
-	194.180.174.53	United States
-	74.125.24.94	United States
-	36.86.63.182	Indonesia

Pada Tabel 4.1 dapat dilihat bahwa *malware* ini melakukan ke beberapa *ip address* dari berbagai negara, dan juga beberapa alamat *domain* yang melakukan komunikasi dengan malware *stealer redline*, untuk informasi *ip address* selengkapnya terlampir dalam lapiran 1.

Pada salah satu proses pengiriman data menuju alamat *ip* 212.193.30.45 ditemukan paket data berisikan alamat *proxy*, yang diindikasikan sebagai daftar alamat *ip* yang digunakan malware *stealer redline* untuk melakukan distribusi data yang berhasil diperoleh dari perangkat korban. *Ip address* yang berisi alamat *proxy* dapat dilihat pada Gambar 4.7.

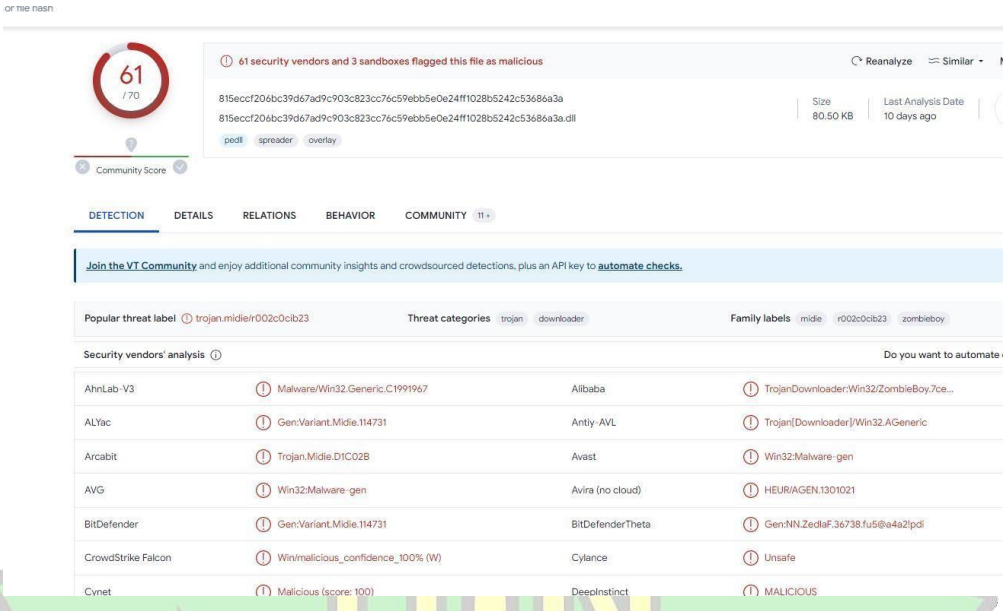


Gambar 4. 7 Alamat *proxy*

Pada Gambar.4.7 dapat dilihat bahwa alamat *ip* 212.193.30.45 melakukan koneksi dengan alamat *proxy*, serta ditemukan alamat *ip* 45.144.225.57 berkomunikasi ke alamat *server*, dan beberapa koneksi lainnya.

B. Trojan zombieboy

Hasil analisis *tool virus* total *malware trojan zombieboy* menggunakan nilai *Hash* 70ccd9220cebb56eaa38b9f1bd1a1cd8.

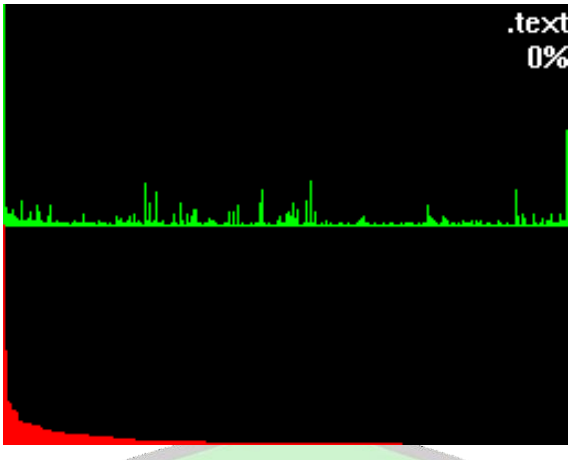


Gambar 4. 8 pengecekkan sampel pada *file trojan zombieboy*

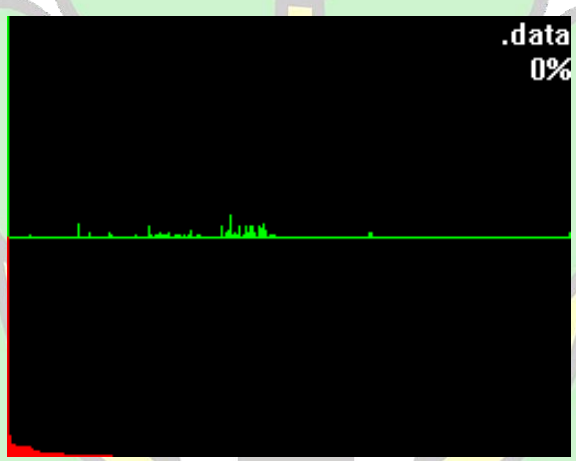
Hasil Gambar 4.2 menunjukkan bahwa *malware trojan zombieboy* tergolong kepada *malware trojan.midie/r002c0cib23*, beberapa *antivirus* juga mendeteksi beberapa jenis *malware* seperti, Alibaba medeteksi (Trojan/Downloader:Win32/Zombieboy.7ce), Antiy-AVL medeteksi (tojan/Downloader/Win32/AGeneric), *AVG* mendeteksi (Win32:malware-gen), dan masih banyak *vendors security* lainnya. Sampel *malware* yang dianalisis terkonfirmasi merupakan *file* yang berbahaya dengan menggunakan *tool virus* total, maka analisis sampel ini dapat dilanjutkan.

4.1.1 Stealer redline

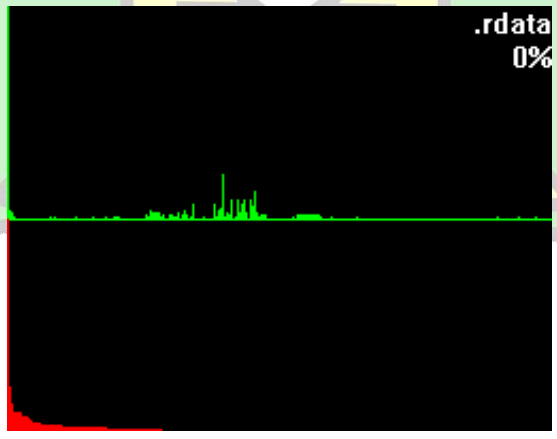
Gambar 4.9 adalah hasil dari *tool byte* histogram sebagai pengecekantingkat pengacakkan data dari *malware* melalui histogram yang dihasilkan. Hasil analisi *tool* histogram dapat dilihat pada gambar berikut ini.



Gambar 4. 9 Hasil *histogram .text*



Gambar 4. 10 Hasil histogram .data



Gambar 4. 11 Hasil *histogram .rdata*

Berdasarkan hasil analisis *histogram* menunjukkan *section .text* 0%, *rdata* 0%, dan *data* 0%.

Gambar 4.12 adalah hasil dari *tool PesStudio* penjelasan singkat tentang struktur dari *file stealer redline*. Seperti nilai *Hash*, *compiler-stamp*, tipe *file*, dan informasi lainnya dapat dilihat pada Gambar 4.12, data ini nantinya akan dipersingkat lagi dalam tabel, yaitu tabel 4.2.

property	value
sha256	A2F9F5A099A6B1C2BA6789EFFEFA150AEC52C5587E85DF9A6963FD03B55D4D57
sha1	86CCEF66BE89113B7DEEF5A09E3354CDD13B0585
md5	0AF9C941D86C3914DF0D442D51536BD8
first-bytes-hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00
first-bytes-text	M Z . . . . . @ . . . . .
file-size	8112022 bytes
entropy	8.000
signature	n/a
tooling	wait...
file-type	executable
cpu	32-bit
subsystem	GUI
file-version	n/a
description	n/a
stamps	
compiler-stamp	Sat Aug 01 02:44:18 2020
debugger-stamp	n/a
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a
file-names	
export	n/a
debug	n/a
version	n/a
manifest	Nullsoft.NSIS.exehead
.NET	n/a

Gambar 4. 12 Ringkasa struktur sampel *file stealer redline*

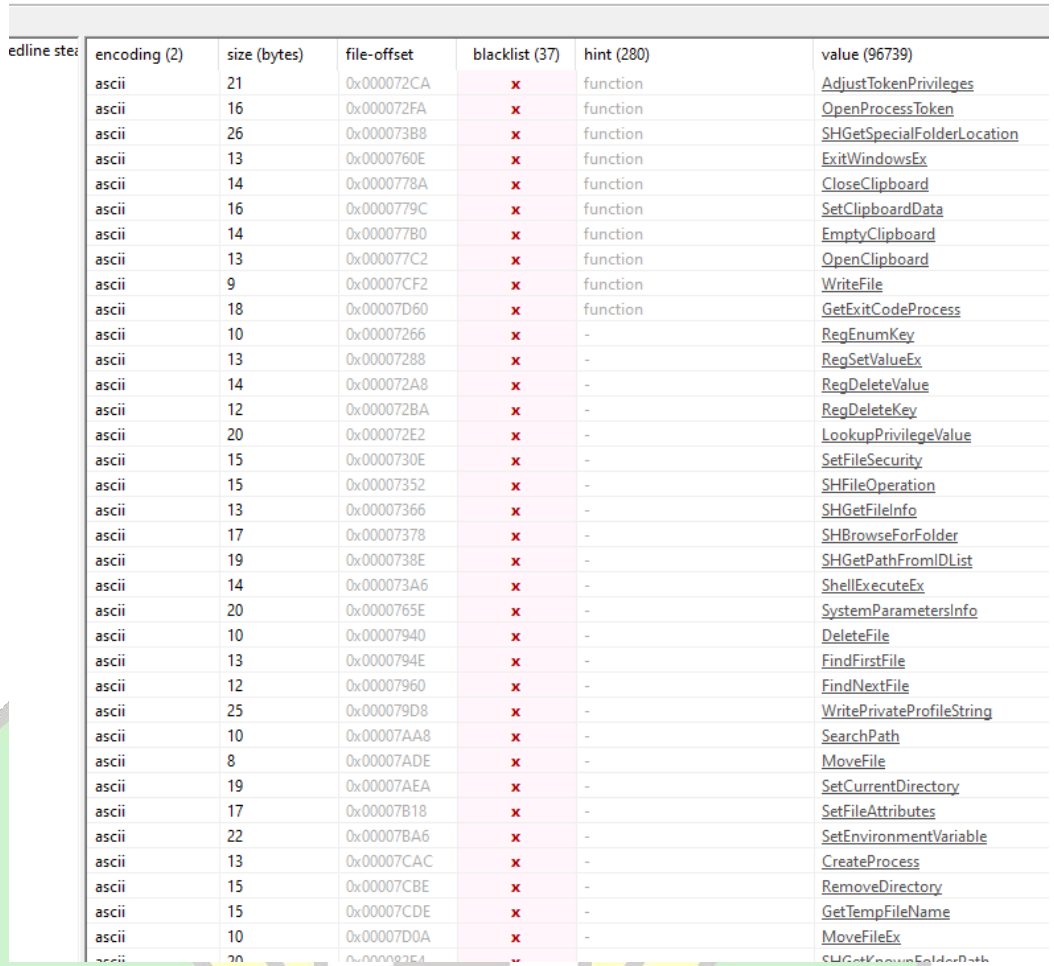
Tabel 4.2 adalah ringkasan dari *malware stealer redline*, dari informasi yang diperoleh dicari lagi informasi-informasi lainnya yang berhubungan dengan *stealer redline* ini dengan menggunakan nilai *Hash* yang tertera pada tabel.

Tabel 4. 2 Ringkasan *stealer redline*

Tools	Variabel	Velue
PeStudio	Nama	*Stealer redline*
	MD5	0AF9C941D86C3914DF0D442D51536BD8
	SHA-1	86CCEF66BE89113B7DEEF5A09E3354CDD13B0585
	SHA256	A2F9F5A099A6B1C2BA6789EFFEFA150AEC52C5587E85DF9A6963FD03B55D4D57
	Compiler-Stamp	Sat Aug 01 02:44:18 2020
	Section	5
	Processor-32*bit*	True
	Excutable	True
ByteHistogram	Ndata	0%
	Rdata	0%
	data	0%
*Virus*Total	Skor	55/70

Informasi yang tertera pada tabel diperoleh dengan menggunakan tiga*tools*, yaitu *PeStudio* untuk menampilkan spesifikasi umum pada *stealer redline* yang dianalisis, seperti kapan *stealer redline* di-compile, *Bity*Histogram untuk mengetahui section dari text, rdate, dan data, dan *virus*total untuk menunjukkan hasil kecocokan *stealer redline* dengan database yang ada pada *virus*total.

Berikut ini adalah tangkapan layar pada Gambar 4.13 dari string yang digunakan oleh *stealer redline*. Data ini akan dijadikan dalam bentuk tabel, namun yang dimasukkan kedalam tabel adalah beberapa string yang dicurigai oleh *PeStudio* saja.



edline stea	encoding (2)	size (bytes)	file-offset	blacklist (37)	hint (280)	value (96739)
	ascii	21	0x000072CA	x	function	AdjustTokenPrivileges
	ascii	16	0x000072FA	x	function	OpenProcessToken
	ascii	26	0x000073B8	x	function	SHGetSpecialFolderLocation
	ascii	13	0x0000760E	x	function	ExitWindowsEx
	ascii	14	0x0000778A	x	function	CloseClipboard
	ascii	16	0x0000779C	x	function	SetClipboardData
	ascii	14	0x000077B0	x	function	EmptyClipboard
	ascii	13	0x000077C2	x	function	OpenClipboard
	ascii	9	0x00007CF2	x	function	WriteFile
	ascii	18	0x00007D60	x	function	GetExitCodeProcess
	ascii	10	0x00007266	x	-	RegEnumKey
	ascii	13	0x00007288	x	-	RegSetValueEx
	ascii	14	0x000072A8	x	-	RegDeleteValue
	ascii	12	0x000072BA	x	-	RegDeleteKey
	ascii	20	0x000072E2	x	-	LookupPrivilegeValue
	ascii	15	0x0000730E	x	-	SetFileSecurity
	ascii	15	0x00007352	x	-	SHFileOperation
	ascii	13	0x00007366	x	-	SHGetFileInfo
	ascii	17	0x00007378	x	-	SHBrowseForFolder
	ascii	19	0x0000738E	x	-	SHGetPathFromIDList
	ascii	14	0x000073A6	x	-	ShellExecuteEx
	ascii	20	0x0000765E	x	-	SystemParametersInfo
	ascii	10	0x00007940	x	-	DeleteFile
	ascii	13	0x0000794E	x	-	FindFirstFile
	ascii	12	0x00007960	x	-	FindNextFile
	ascii	25	0x000079D8	x	-	WritePrivateProfileString
	ascii	10	0x00007AA8	x	-	SearchPath
	ascii	8	0x00007ADE	x	-	MoveFile
	ascii	19	0x00007AEA	x	-	SetCurrentDirectory
	ascii	17	0x00007B18	x	-	SetFileAttributes
	ascii	22	0x00007BA6	x	-	SetEnvironmentVariable
	ascii	13	0x00007CAC	x	-	CreateProcess
	ascii	15	0x00007CBE	x	-	RemoveDirectory
	ascii	15	0x00007CDE	x	-	GetTempFileName
	ascii	10	0x00007D0A	x	-	MoveFileEx

Gambar 4. 13 string yang digunakan *stealer redline*

Dari Gambar 4.13 diambil data yang menurut peneliti dan *PeStudio* berbahaya dan akan dijelaskan lagi dalam bentuk tabel, yang dapat dilihat pada tabel 4.3.

Tabel 4. 3 *String* yang digunakan *stealer redline*

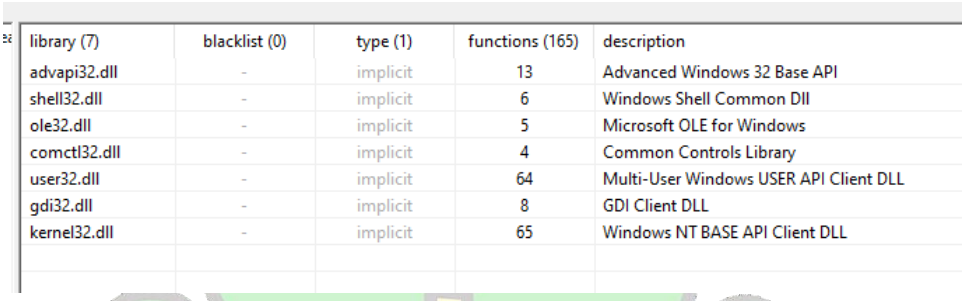
Encoding	Value	Hasil
ascii	AdjustTokenPrivilegen	Mengaktifkan atau menonaktifkan hak istimewa dalam token akses yang ditentukan
ascii	OpenProsessToken	Membuka token akses yang terkait dengan suatu proses
ascii	SHGetSpesialFolderLocation	Mengambil data pointer dari struktur

		ITEMDLIST
ascii	ExitWindowsEx	Menghentikan Proses windows secara keseluruhan
ascii	CreateProcess	Membuat proses baru dan utas utamanya. Proses baru berjalan dalam konteks keamanan proses pemanggilan
ascii	Delete*File*	Menghapus *file* tertentu
ascii	EmptyClipboard	Mengosongkan clipboard
ascii	ShellExecuteEx	Mengoperasikan Shell
ascii	Write*File*	Menulis data ke *file* yang ditentukan atau perankat input/output(I/O)
ascii	GetExitCodeProcess	Mengambil status penghentian dari proses yang ditentukan
ascii	RegEnumKey	Menghitung nilai untuk kunci registri terbuka
ascii	RegSetVelue	Mengatur data dan tipe dari nilai dibawah registri
ascii	RegDeleteVelue	Manghapus nilai dari kunci registri tertentu
ascii	RegDeleteKey	Menghapus kunci dari registry tertentu
ascii	LookupPrivilegeValue	Mengambil pengidentifikasi unik lokal(LUID) yang digunakan pada sistem tertentu untuk secara lokal mewakili nama hak istimewa yang ditentukan

Tabel 4.3 adalah nilai string yang diperoleh dari *malware stealer redline*, *stealer redline* memiliki 96.739 string namun isi dari tabel 4.3 ialah beberapa string yang di-*backlist* oleh aplikasi *PeStudio*, dari informasi di atas dapat diketahui bahwa *stealer redline* dapat membuat proses baru (*CreateProcess*),

mengaktifkan dan menonaktifkan hak istimewa dalam token akses (*AdjustTokenPrivilegen*), dan menghapus kunci dari registry (*RegDeleteKey*).

Berikut adalah tangkapan layar pada gambar 4.14 dari *library* yang digunakan oleh *stealer redline*.



library (7)	blacklist (0)	type (1)	functions (165)	description
advapi32.dll	-	implicit	13	Advanced Windows 32 Base API
shell32.dll	-	implicit	6	Windows Shell Common Dll
ole32.dll	-	implicit	5	Microsoft OLE for Windows
comctl32.dll	-	implicit	4	Common Controls Library
user32.dll	-	implicit	64	Multi-User Windows USER API Client DLL
gdi32.dll	-	implicit	8	GDI Client DLL
kernel32.dll	-	implicit	65	Windows NT BASE API Client DLL

Gambar 4. 14 library yang digunakan *stealer redline*

Tabel 4.4 adalah data yang diambil dari Gambar 4.14 dan dijelaskan kembali dalam bentuk tabel untuk setiap library yang digunakan oleh *stealer redline*.

Tabel 4. 4 Library *yang digunakan stealer redline*

Nama	Jumlah Fungsi	Penjelasan
advapi32.dll	13	Menyediakan akses ke komponen *windows* seperti *service manager* dan *registry*
Shell32.dll	6	*Library* yang berisi *windows shell API* digunakan untuk membuka web dan *file*
Ole32.dll	5	Menyediakan implementasi dari teknologi *OLE(Object Linking Embedding)*
Comct32.dll	4	Menyediakan kontrol umm untuk aplikasi wondows
User32.dll	64	Membuat program untuk menampilkan *GUI*
Gdi32.dll	8	Membuat program dapat mengekspor *GDI*
Karnel32.dll	65	*Karnel32* sangat umum digunakan karena memiliki fungsi penting seperti akses, manipulasi memori, *file* dan *hardware*

Tabel 4.4 menyediakan informasi library yang digunakan oleh stealer redline, dari tabel ini juga dapat melihat perilaku stealer redline melalui library yang digunakan, seperti mengakses registry serta mengakses dan manipulasi memori file dan hardware.

Berikut adalah tangkapan layar pada gambar 4.15 dari *import* yang digunakan oleh *stealer redline*. Nantinya data ini akan dijadikan dalam bentuk tabel, namun yang dimasukkan kedalam tabel adalah beberapa *import* yang dicurigai oleh *PeStudio* saja.

functions (165)	blacklist (35)	type (1)	ordinal (1)	library (7)
RegEnumKeyW	x	implicit	-	advapi32.dll
RegSetValueExW	x	implicit	-	advapi32.dll
RegDeleteValueW	x	implicit	-	advapi32.dll
RegDeleteKeyW	x	implicit	-	advapi32.dll
AdjustTokenPrivileges	x	implicit	-	advapi32.dll
LookupPrivilegeValueW	x	implicit	-	advapi32.dll
OpenProcessToken	x	implicit	-	advapi32.dll
SetFileSecurityW	x	implicit	-	advapi32.dll
SHGetSpecialFolderLocation	x	implicit	-	shell32.dll
SHFileOperationW	x	implicit	-	shell32.dll
SHBrowseForFolderW	x	implicit	-	shell32.dll
SHGetPathFromIDListW	x	implicit	-	shell32.dll
ShellExecuteExW	x	implicit	-	shell32.dll
SHGetFileInfoW	x	implicit	-	shell32.dll
OpenClipboard	x	implicit	-	user32.dll
SetClipboardData	x	implicit	-	user32.dll
CloseClipboard	x	implicit	-	user32.dll
SystemParametersInfoW	x	implicit	-	user32.dll
ExitWindowsEx	x	implicit	-	user32.dll
EmptyClipboard	x	implicit	-	user32.dll
GetExitCodeProcess	x	implicit	-	kernel32.dll
WriteFile	x	implicit	-	kernel32.dll
GetTempFileNameW	x	implicit	-	kernel32.dll
RemoveDirectoryW	x	implicit	-	kernel32.dll
CreateProcessW	x	implicit	-	kernel32.dll
SetEnvironmentVariableW	x	implicit	-	kernel32.dll
SetFileAttributesW	x	implicit	-	kernel32.dll
SetCurrentDirectoryW	x	implicit	-	kernel32.dll
MoveFileW	x	implicit	-	kernel32.dll
SearchPathW	x	implicit	-	kernel32.dll

Gambar 4. 15 ampilan import yang digunakan stealer redline

Tabel 4.5 adalah data yang diambil dari gambar 4.6 dan dijelaskan kedalam bentuk tabel untuk setiap *library* yang digunakan oleh *stealer redline*.

Tabel 4. 5 Import yang digunakan *stealer redline*

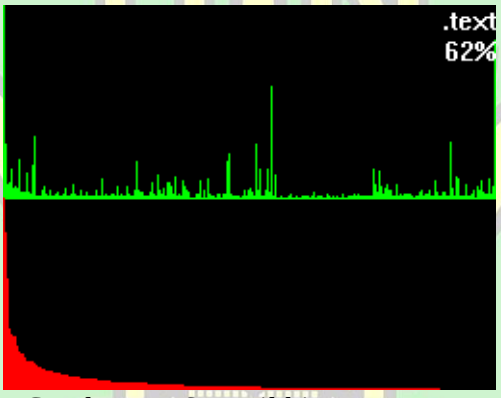
Nama	Kategori	Library	Penjelasan
RegEnumKeyW	implicit	Advapi32.dll	Menghitung sub kunci dari kunci registri terbuka yang ditentukan
RegSetValueExW	implicit	Advapi32.dll	Mangatur data dan tioe dari nilai bawah registri key
RegDeleteValueW	implicit	Advapi32.dll	Mengapus nilai kunci registri tertentu
RegDeleteKeyW	implicit	Advapi32.dll	Menghapus kunci dari registri tertentu
OpenProcessToken	implicit	Advapi32.dll	Membuka token akses yang terkait dengan suatu proses
SH*File*OperationW	implicit	Shell32.dll	Melakukan pengoperasian *file* tertentu
SHBrowserForFolderW	implicit	Shell32.dll	Melakukan pencarian destinasi folder yang dituju
SHGetPathFromDListW	implicit	Shell32.dll	Mengambil alamat destinasi dari daftar IDE sautu objek tertentu
ShelExecuteExW	implicit	Shell32.dll	Mengoperasikan shell
SHGet*File*InfoW	implicit	Shell32.dll	Mengambil informasi dari *file* tertentu
GetExitCodeProcess	implicit	Karnel32.dll	Mengambil status pengentian dari proses yang ditentukan
GetTemp*File*NameW	implicit	Karnel32.dll	Membuat nama untuk *file* sementara
RemoveDirectoryW	implicit	Karnel32.dll	Menghapus direktori kosong yang ada
CreateProcessW	implicit	Karnel32.dll	Membuat proses baru dan utas utamanya

Set*Environment*VariableW	implicit	Karnel32.dll	Atur konten variabel lingkungan yang ditentukan untuk proses saat ini
Set*File*AttributesW	implicit	Karnel32.dll	Menentukan atribut untuk suatu *file* atau direktori

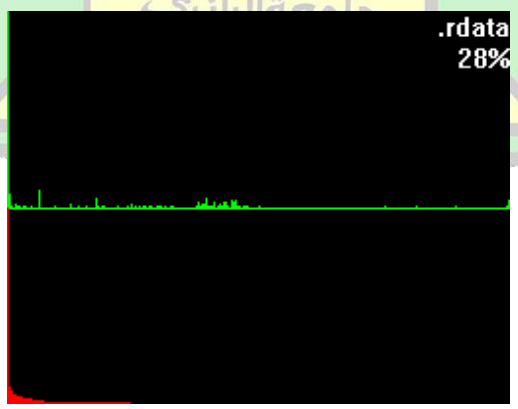
Pada tabel 4.4 disediakan informasi yang di-*import* dari *library*, biasanya yang ditunjukkan pada *string* dan *import* memiliki perbedaan namun pada *stealer redline* memiliki kesamaan yang telah di-*blacklist* oleh *PeStudio*.

4.1.2 Trojan zombieboy

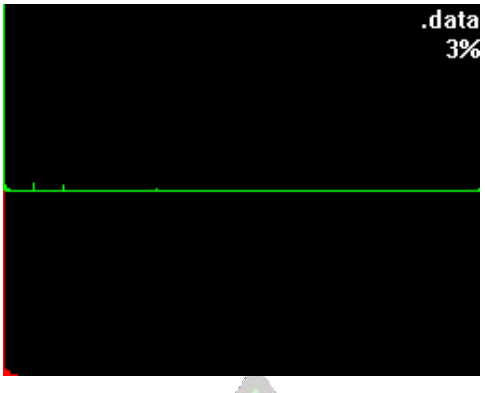
Gambar 4.16, 4.17, dan 4.18 adalah hasil dari *tool byte histogram* sebagai pengecekan tingkat pengacakkan data dari *malware* melalui histogram yang dihasilkan. Hasil analisi *tool histogram* dapat dilihat pada gambar berikut ini.



Gambar 4. 16 *Hasil histogram .text*



Gambar 4. 17 Hasil histogram .rdata



Gambar 4. 18 Hasil histogram .data

Berdasarkan analisis *Byte histogram* menunjukkan nilai *section* .text 62%, rdata 28%, dan data 3%.

Gambar 4.19 adalah gambar dari hasil *tool PeStudio* penjelasan singkat tentang struktur dari *file trojan zombieboy*. Hasil yang diperoleh seperti nilai *Hash*, *compiler-stamp*, tipe *file*, dan hasil lainnya dapat dilihat pada gambar 4.7, data ini nantinya akan dipersingkat lagi dalam bentuk tabel, yaitu pada tabel 4.6.

property	value
md5	70CCD9220CEBB56EAA38B9F1BD1A1CD8
sha1	17EBD69CC7302FE5B44015386054EB87FE73C3CE
sha256	815ECCF206BC39D67AD9C903C823CC76C59EBB5E0E24FF1028B5242C53686A3A
first-bytes-hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00
first-bytes-text	M Z . . . . . @ . . . . .
file-size	82435 (bytes)
entropy	6.340
imphash	FE0DA85AC37FA2E5451E982640B9950C
signature	n/a
entry-point	55 8B EC 83 7D 0C 01 75 05 E8 38 01 00 00 FF 75 10 FF 75 0C FF 75 08 E8 BE FE FF FF 8
file-version	n/a
description	n/a
file-type	**dynamic-link-library**
cpu	**32-bit**
subsystem	GUI
compiler-stamp	0x59115AA9 (Tue May 09 12:59:05 2017)
debugger-stamp	0x59115AA9 (Tue May 09 12:59:05 2017)
resources-stamp	0x00000000 (empty)
import-stamp	0x00000000 (empty)
exports-stamp	n/a

Gambar 4. 19 Ringkasan struktur *file trojan zombieboy*

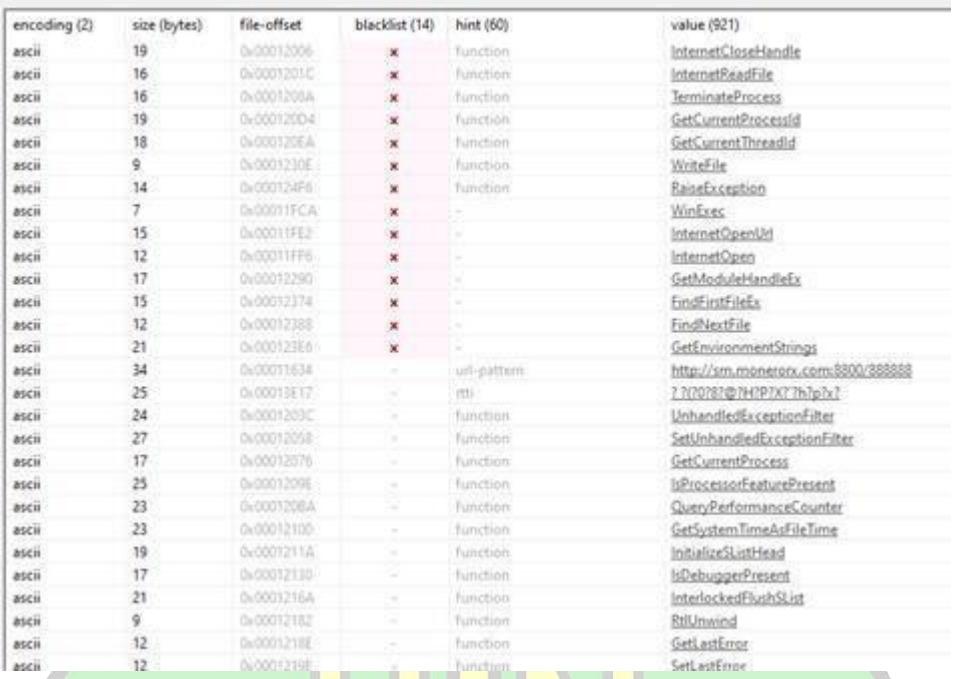
Tabel 4.6 adalah ringkasan dari *trojan zombieboy*, dari informasi ini dapat dicari lagi informasi-informasi lainnya yang berhubungan dengan *trojan zombieboy* ini menggunakan nilai *Hash* tertera pada tabel.

Tabel 4. 6 Ringkasan trojan zombieboy

Tools	**Variabel**	**Value**
PeStudio	Nama	*Trojan zombieboy*
	MD5	70ccd9220cebb56eaa38b9f1bd1a1cd8
	SHA-1	17ebd69cc7302fe5b44015386054eb87fe73c3ce
	SHA256	815eccf206bc39d67ad9c903c823cc76c59ebb5e0e24 ff1 028b5242c53686a3a
	Compiler -Stamp	Tue may 09 12:59:05 2017
	Section	5
	Processor -32*bit*	True
	Excutable	True
ByteHistogra m	Text.	62%,
	Rdata.	28%,
	Data.	3%
*Virus*Total	skor	61/71

Infomasi yang tertera pada tabel diperoleh dengan mengggunakan tigas *tools*, yaitu *PeStudio* untuk menampilkan spesifikasi umum pada *trojan zombieboy* yang dianalisis seperti kapan *trojan zombieboy* di-*compile*, *ByteHistogram* untuk melihat diagram dari *malware*, dan *virus*total untuk menunjukkan hasil kecocokan anatara *trojan zombieboy* dengan database yang ada pada *virus*total.

Berikut adalah tangkapan layar pada gambar 4.20 dari *string* yang digunakan oleh *trojan zombieboy*. Nantinya data ini akan dijadikan dalam tabel, namun yang dimasukkan ke dalam tabel adalah *string* yang dicurigai oleh *PeStudio* saja.



encoding (2)	size (bytes)	file-offset	blacklist (14)	hint (60)	value (921)
ascii	19	0x00012006	x	function	InternetCloseHandle
ascii	16	0x0001201C	x	function	InternetReadFile
ascii	16	0x0001200A	x	function	TerminateProcess
ascii	19	0x000120D4	x	function	GetCurrentProcessId
ascii	18	0x000120EA	x	function	GetCurrentThreadId
ascii	9	0x0001230E	x	function	WriteFile
ascii	14	0x000124F6	x	function	RaiseException
ascii	7	0x00011FCA	x	-	WinExec
ascii	15	0x00011FE2	x	-	InternetOpenUrl
ascii	12	0x00011FF6	x	-	InternetOpen
ascii	17	0x00012290	x	-	GetModuleHandleEx
ascii	15	0x00012374	x	-	FindFirstFileEx
ascii	12	0x00012388	x	-	FindNextFile
ascii	21	0x000123E6	x	-	GetEnvironmentStrings
ascii	34	0x00011634	-	url-pattern	http://sm.minergrx.com:8800/388888
ascii	25	0x00013E17	-	rtti	?.?@@?@?H?P?X?`?h?p?x?
ascii	24	0x0001203C	-	function	UnhandledExceptionFilter
ascii	27	0x00012058	-	function	SetUnhandledExceptionFilter
ascii	17	0x00012076	-	function	GetCurrentProcess
ascii	25	0x0001209E	-	function	IsProcessorFeaturePresent
ascii	23	0x000120B6	-	function	QueryPerformanceCounter
ascii	23	0x00012100	-	function	GetSystemTimeAsFileTime
ascii	19	0x0001211A	-	function	InitializeSListHead
ascii	17	0x00012130	-	function	IsDebuggerPresent
ascii	21	0x0001216A	-	function	InterlockedFlushSList
ascii	9	0x00012182	-	function	RtlUnwind
ascii	12	0x0001218E	-	function	GetLastError
ascii	12	0x0001219E	-	function	SetLastError

Gambar 4. 20 string yang digunakan oleh *trojan zombieboy*

Dari Gambar 4.20 diambil data yang menurut *pestudio* berbahaya dan akan dijelaskan lagi dalam bentuk tabel, yang dapat dilihat pada tabel 4.6.

Tabel 4. 7 string yang digunakan *trojan zombieboy*

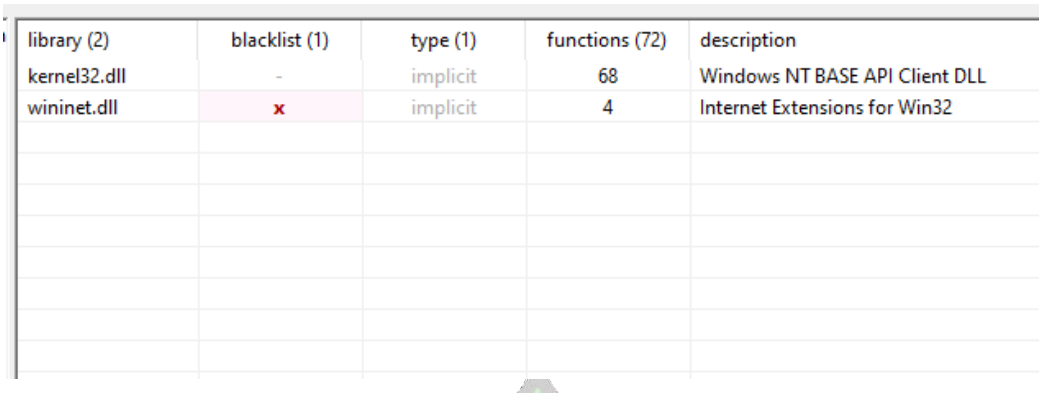
Encoding	Value	Penjelasan
asci	*InternetCloseHandle*	Menutup *handle* internet
asci	*InternetReadFile*	Membaca data yang dibuka oleh fungsi *InternetOpenUrl*
asci	*TerminateProcess*	Menghentikan proses yang ditentukan dan semua yang berurutaan
asci	*GetCurrentProcessId*	Mengambil *ID* pengidentifikasi proses
asci	*GetCurrentThreadId*	Mengambil thread identifier dari calling thread
asci	*WriteFile*	Menulis data pada spesifik *file*
asci	*RaiseException*	*Raise exception* pada calling thread
asci	*WinExec*	Menjalankan aplikasi yang ditentukan
asci	*InternetOpenUrl*	Membuka sumber yang ditentukan oleh *FTP* lengkap atau *URL HTTP*

asci	*GetModuleHandleEx*	Handle modul untuk modul yang ditentukan. Modul harus dimuat oleh proses pemanggilan
asci	*FindFirstFileEx*	Mencari direktori untuk *file* atau subdirektori dengan nama da atribut yang cocok dengan yang ditentukan.
asci	*FindNextFile*	Melanjutkan pencarian *file* dari panggilan sebelumnya ke *FindNextFile*, *FindNextFileEx*, atau fungsi *FindFirstFileTransacted*
asci	*GetEnvironmentStrings*	Mengambil variable *environment* yang sedang berlangsung

Tabel 4.7 adalah nilai *string* yang diperoleh dari *trojan zombieboy*. *Trojan zombieboy* memiliki 912 *string*, namun isi pada tabel 4.7 adalah beberapa *string* yang di-*blacklist* oleh *software pestudio*.

Berdasarkan informasi pada tabel dapat diketahui bahwa *trojan zombieboy* dapat mengotrol komputer dan mengakses *file* di dalam komputer. *Trojan zombieboy* juga dapat menjalankan aplikasi yang ditentukan (*WinExec*), Membaca data yang dibuka oleh fungsi *InternetOpenUrl* (*InternetReadFile*), mengambil variabel *environment* yang sedang berlangsung (*GetEnvironmentStrings*), Mengambil *ID* pengidentifikasi proses (*GetCurrentProcessld*), dan Menutup handle internet (*InternetCloseHandle*).

Berikut adalah tangkapan layar Gambar 4.21 dari *library* yang digunakan oleh *Trojan zombieboy*. Seluruh *library* akann dijelaskan lebih rinci dalam table 4.7



library (2)	blacklist (1)	type (1)	functions (72)	description
kernel32.dll	-	implicit	68	Windows NT BASE API Client DLL
wininet.dll	x	implicit	4	Internet Extensions for Win32

Gambar 4. 21 Library yang digunakan *trojan zombieboy*

Tabel 4.8 adalah data yang diambil dari Gambar 4.21 dan dijelaskan kembali dalam bentuk tabel untuk setiap *library* yang digunakan oleh *trojan zombieboy*.

Tabel 4. 8 Library yang digunakan *trojan zombieboy*

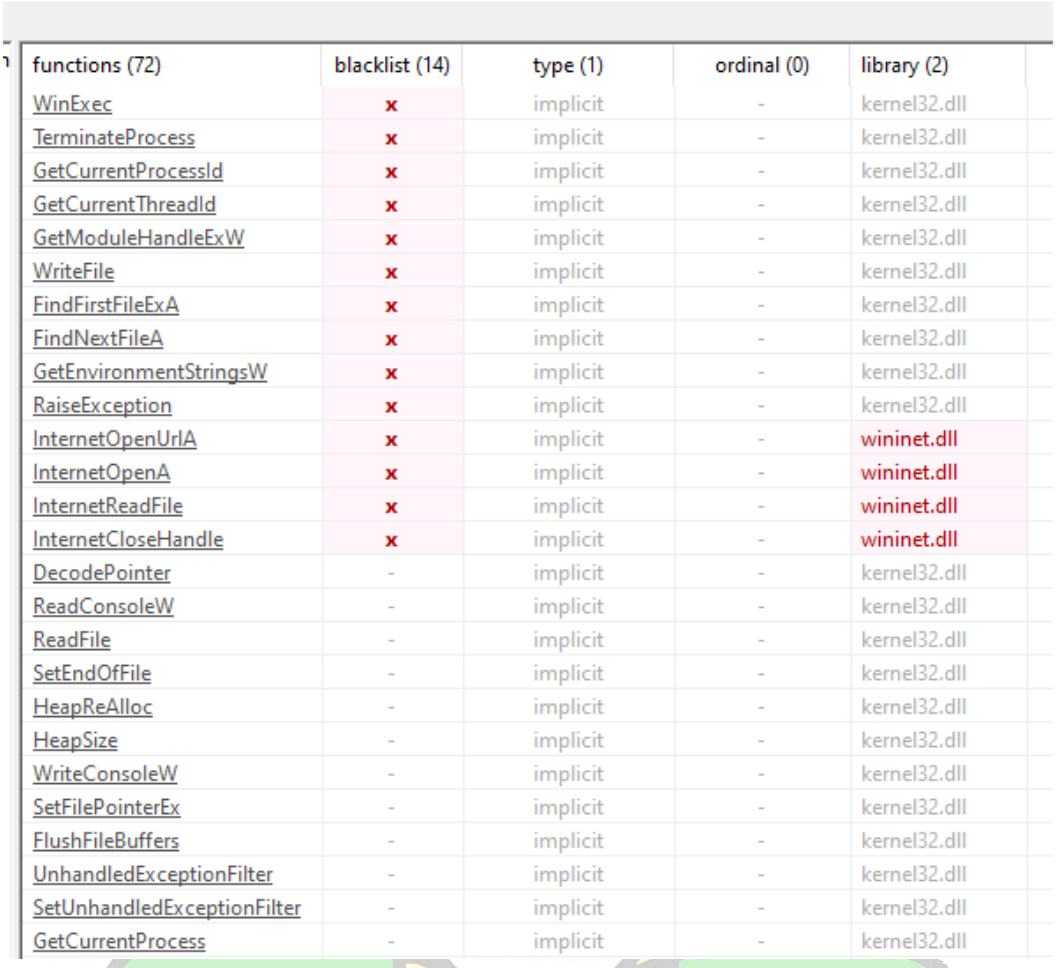
Library	***Import***	**Penjelasan**
Karnel32.dll	68	*Karnel32* sangat umum digunakan karena memiliki fungsi penting seperti akses, manipulasi memory, *file* dan *hardware*
Wininet.dll	4	*Wininet*.ddl memiliki berbagai fungsi internet seperti aplikasi web browser, FTP, dan aplikasi yang memerlukan internet.

Tabel 4.8 adalah *library* yang digunakan oleh *trojan zombieboy*, dari tabel ini juga dapat melihat perilaku *trojan zombieboy* melalui *library* yang digunakan seperti mengakses *registry* serta mengakses dan memanipulasi *memory*, *file* dan *hardware*, serta mencari informasi sensitif di *internet*. Namun *trojan zombieboy* hanya memiliki 2 *library* yang digunakan dengan total penggunaan *import* dari dua *library* sebanyak 72 *import* yang digunakan.

Trojan zombieboy berfokus untuk mengambil akses dari perangkat komputer *software* dan *hardware* yang terdapat di komputer, sehingga setiap software dan hardware di dalam komputer bisa di kendalikan.

Berikut adalah tangkapan layar pada gambar 4.22 dari *import* yang digunakan oleh *trojan zombieboy*. Nantinya data ini akan dijadikan dalam bentuk

tabel, namun yang dimassukan ke dalam tabel adalah *import* yang dicurigai oleh *PeStudio* saja.



functions (72)	blacklist (14)	type (1)	ordinal (0)	library (2)
WinExec	x	implicit	-	kernel32.dll
TerminateProcess	x	implicit	-	kernel32.dll
GetCurrentProcessId	x	implicit	-	kernel32.dll
GetCurrentThreadId	x	implicit	-	kernel32.dll
GetModuleHandleExW	x	implicit	-	kernel32.dll
WriteFile	x	implicit	-	kernel32.dll
FindFirstFileExA	x	implicit	-	kernel32.dll
FindNextFileA	x	implicit	-	kernel32.dll
GetEnvironmentStringsW	x	implicit	-	kernel32.dll
RaiseException	x	implicit	-	kernel32.dll
InternetOpenUrlA	x	implicit	-	wininet.dll
InternetOpenA	x	implicit	-	wininet.dll
InternetReadFile	x	implicit	-	wininet.dll
InternetCloseHandle	x	implicit	-	wininet.dll
DecodePointer	-	implicit	-	kernel32.dll
ReadConsoleW	-	implicit	-	kernel32.dll
ReadFile	-	implicit	-	kernel32.dll
SetEndOfFile	-	implicit	-	kernel32.dll
HeapReAlloc	-	implicit	-	kernel32.dll
HeapSize	-	implicit	-	kernel32.dll
WriteConsoleW	-	implicit	-	kernel32.dll
SetFilePointerEx	-	implicit	-	kernel32.dll
FlushFileBuffers	-	implicit	-	kernel32.dll
UnhandledExceptionFilter	-	implicit	-	kernel32.dll
SetUnhandledExceptionFilter	-	implicit	-	kernel32.dll
GetCurrentProcess	-	implicit	-	kernel32.dll

Gambar 4. 22 Import yang digunakan oleh *trojan zombieboy*

Tabel 4.9 adalah data yang diambil dari Gambar 4.22 dan dijelaskan kembali dalam bentuk tabel untuk setiap *library* yang digunakan oleh *trojan zombieboy*.

Tabel 4. 9 *Import* yang digunakan *trojan zombieboy*

Nama	**Grup**	***Library***	**Penjelasakn**
WinExec	implisit	Karnel32.dll	Menjalankan aplikasi tertentu
TerminateProcess	implisit	Karnel32.dll	Hentikan proses yang ditentukan dan semua yang berurutan
GetCurrentProcessld	implisit	Karnel32.dll	Mengambil *ID* pengidentifikasi proses

GetCurrentThreadId	implisit	Karnel32.dll	Mengambil thread indentifier dari calling
GetmoduleHandleExW	implisit	Karnel32.dll	*Handle module* untuk modul yang ditentukan. Modul harus dimuat oleh proses pemanggilan
WriteFile	implisit	Karnel32.dll	Menulis data pada spesifik *file*
FindFirstFileExA	implisit	Karnel32.dll	Mencari direktori untuk *file* atau subdirektori dengan nama dan atribut yang cocok dengan yang ditentukan
FindNextFileA	implisit	Karnel32.dll	Melanjutkan pencarian *file* dari pemanggilan sebelumnya ke *FindFirstFile*, *FindFirstFileEx*, atau fungsi *FindFirstFileTransacted*.
GetmoduleHandleExW	implisit	Karnel32.dll	Mengambil variabel *environment* yang string
RaiseException	implisit	Karnel32.dll	Raise eception pada calling thread
InternetOpenUrlA	implisit	Wininet.dll	Membuka sumber yang ditentukan oleh *FTP* lengkap atau *URL*, *HTTP*
Internet OpenA	implisit	Wininet.dll	Menginisialisasi penggunaan aplikasi fungsi *WinNet*
InternetReadFile	implisit	Wininet.dll	Membaca data yang dibuka oleh fungsi *InternetOpenUrl*, *FtpOpenFile*, atau *HttpOpenRequest*
InternetCloseHandle	implisit	Wininet.dll	Menutup handle internet

Import yang digunakan oleh *trojan zombieboy* berfokus pada`tabel 4.9 tampilan *trojan zombieboy* pada komputer yang ditunjukkan pada fungsi *WinExec* dimana *trojan zombieboy* dapat menjalankan aplikasi tertentu di dalam komputer, menghentikan proses yang berjalan, mengambil *ID* identitas proses, membaca data yang dibuka, dan menutup handle *internet*.

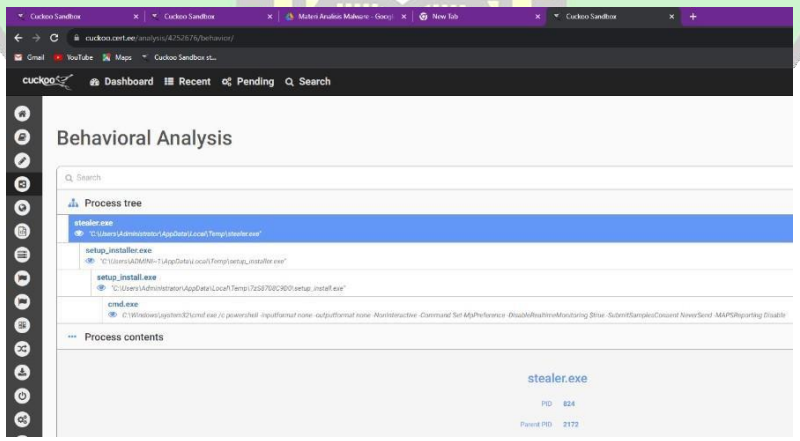
4.2 Hasil Analisis Dinamis

Setalah proses membangun lingkungan untuk proses analisis dan mengonfigurasi *cuckoo* dengan baik, maka *network* yang dibutuhkan *cuckoo* agar dapat menjalankan analisis *malware* sudah siap dianalisis. Sebelumnya telah disiapkan beberapa sampel *malware* yang sudah siap dianalisis, berikut sampel *malware* yang telah disiapkan

- *Stealer redline*
- *Trojan zombieboy*

4.2.1 Stealer redline

Pada Gambar 4.23 adalah hasil analisis *stealer redline*, *stealer redline* memiliki 4 *tree* proses dimna proses dilakukan satu persatu mulai dari *stealer.exe*, lalu *setup_installer.exe*, dilanjutkan *setup_install.exe*, dan *cmd.exe*. Setiap *tree* memiliki proses konten masing-masing dalam melakukan injeksi.



Gambar 4. 23 Hasil analisis dinamis *stealer redline*

Berikut ini adalah isi dari proses *tree* dari *stealer redline*. Proses *tree* terbagi menjadi 9 bagian untuk memisahkan banyak proses yang dilakukan oleh

program yang dianalisis. Berikut adalah Gambar 4.24 *tree behavioral* analisis dinamis *stealer redline*.

Gambar 4. 24 Isi Proses dari *stealer redline*

Setiap proses mengisi bagaian yang berbeda di karenakan setiap proses memiliki tugas masing-masing dalam melakukan infeksi. Misalkan isi proses dari *stealer redline* memiliki 4 bagian proses yaitu pada *file*, proses, *registry*, dan sinkronisasi. Hal ini berarti proses *stealer redline* tidak hanya untuk melakukan proses infeksi namun juga untuk mencari *file* dan mengubah *registry*.

Dapat dilihat pada Gambar 4.25, banyak yang terjadi saat melakukan analisis dinamis salah satunya ialah *malware* dapat menyembunyikan aktivitasnya pada ruang alamat proses yang terinfeksi, *malware* dapat mencoba mengukur jumlah memori yang tersedia di sistem dan *malware* menggunakan *Pem-Packer* untuk menyembunyikan atau mengenkripsi kode mereka.

Gambar 4.25 Signature low *stealer redline*

Pada proses selanjutnya *stealer redline* mencoba menyembunyikan dirinya dengan membuat *file* atau direktori di sistem *file*, *malware* dapat membuat proses-proses mencurigakan di dalam sistem, dan *malware* mencoba untuk

mengeksploitasi alat-alat utilitas *windows* untuk menjalankan atau menyembukan dirinya, proses ini dapat dilihat pada Gambar 4.26.

Gambar *4.26 Signature stealer redline*

Signature terakhir adalah *signature* yang dianggap berbahaya, dapat dilihat pada gambar 4.27 *stealer redline* mencoba mendeteksi apakah sedang dijalankan dalam lingkungan pengembang atau analisis forensik dengan memeriksa keberadaan *debugger*, dan *malware* dapat mengeksekusi perintah dan skrip melalui powershell.

Gambar 4. 27 Signature higt stealer redline

Berikut ini adalah penjelasan dari *API library* dan *function call* yang dimiliki oleh *stealer redline*

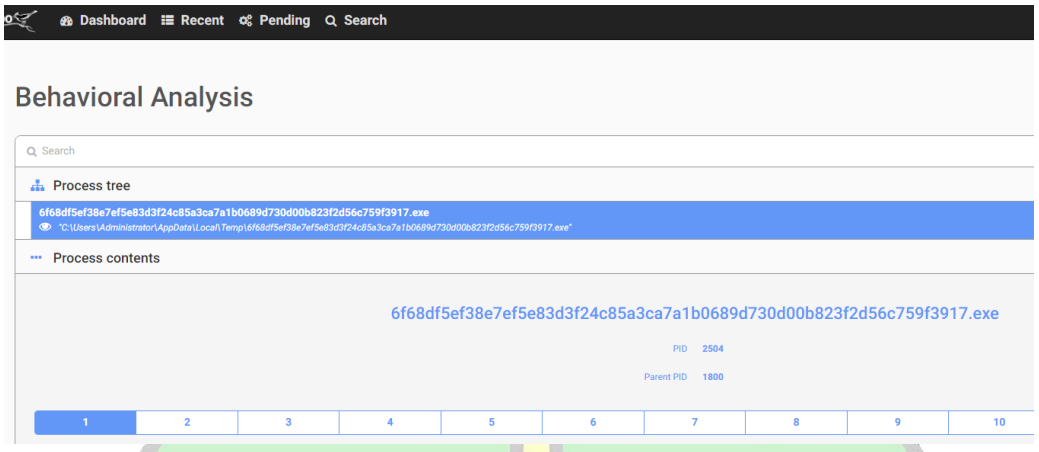
Tabel 4. 10 penjelasan *API Library kernel32.dll* dan *Function stealer redline*

Library KERNEL32.dll		
GetExitCodeProcess	0x408070	Mengambil status penghentian dari proses yang ditentukan
GetModuleHandleA	0x408078	Mengambil handle modul untuk modul yang ditentukan
CreateProcessW	0x4080a0	Membuat proses baru dan utas utamanya. Proses baru berjalan dalam konteks keamanan proses pemanggilan.
GetDiskFreeSpaceW	0x4080b8	Mengambil informasi tentang disk yang ditentukan, termasuk jumlah ruang kosong pada disk.
CopyFileW	0x4080e4	Melakukan penyalinan *file*
GetFullPathNameW	0x408110	Mendapatkan alamat destinasi lengkap
CloseHandle	0x408124	Menutup kendali dari suatu proses
FindClose	0x408164	Menutup kendali pencarian *file*
FindFirstFileW	0x40816c	Melakukan pencarian *file* dengan spesifik nama

Berdasarkan Tabel 4.10 *stealer redline* dapat mengambil status penghentian dari proses (*GetExitCodeProcess*), menutup kendali pencarian *file* (*FindClose*), mencari *file* dengan nama spesifik (*FindFirstFileW*), menyalin *file* (*CopyFileW*), dan masih banyak lagi *function stealer redline* lainnya dapat dilihat pada lampiran 2.

4.2.2 *Trojan zombieboy*

Pada Gambar 4.28 adalah hasil analisi *trojan zombieboy*, *trojan zombieboy* memiliki 1 proses yang di mana proses dilakukan



Gambar 4.28 Hasil analisis dinamis *trojan zombieboy*

Berikut ini isi dari proses tree pada *trojan zombieboy* dan proses tree tersebut terbagi lagi menjadi 9 bagian untuk memecah banyak proses yang dilakukan oleh program yang dianalisis. Berikut adalah laporan proses pada hasil analisis dinamis.

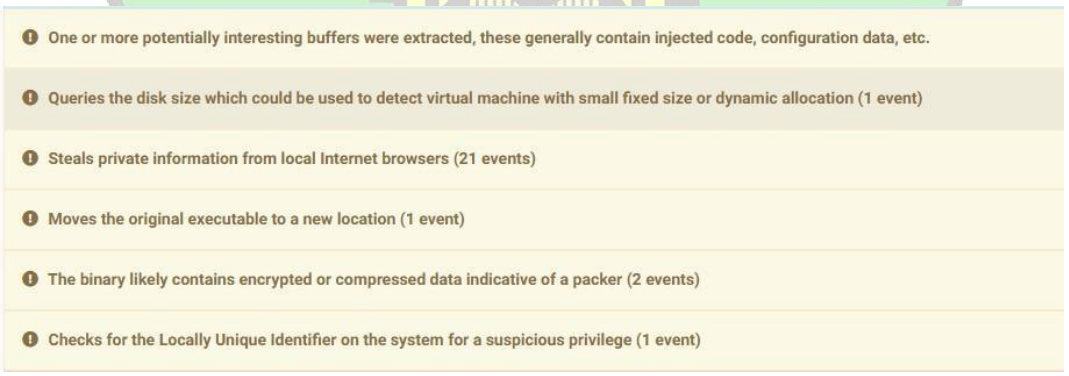
Gambar 4. 29 Isi proses *trojan zombieboy*

Karena proses mengisi bagian yang berbeda dikarenakan setiap proses memiliki tugas masing-masing dalam melakukan infeksi. Isi proses dari *trojan zombieboy* memiliki 3 bagian yaitu pada bagian proses, *file*, dan sinkronisasi. Hal ini berarti proses *trojan zombieboy* dapat melakukan pencarian *file*, isi pada bagian proses yang memiliki perintah *GetSystemTimeAsFileTime* dan *GetSystemInfo*.

Dapat dilihat pada Gambar 4.30 banyak yang terjadi saat melakukan analisis dinamis salah satunya adalah pengecekkan apakah proces yang berjalan sedang di-*debug*, *malware* berusaha mengumpulkan informasi sistem yang dapat digunakan untuk membuat sidik jari sistem, mencari lokasi instalisasi browser pada sistem dan masih banyak lagi proces yang dilakukan *malware* dapat dilihat pada gambar dibawah ini.

Gambar 4. 30 Signature low trojan zombieboy

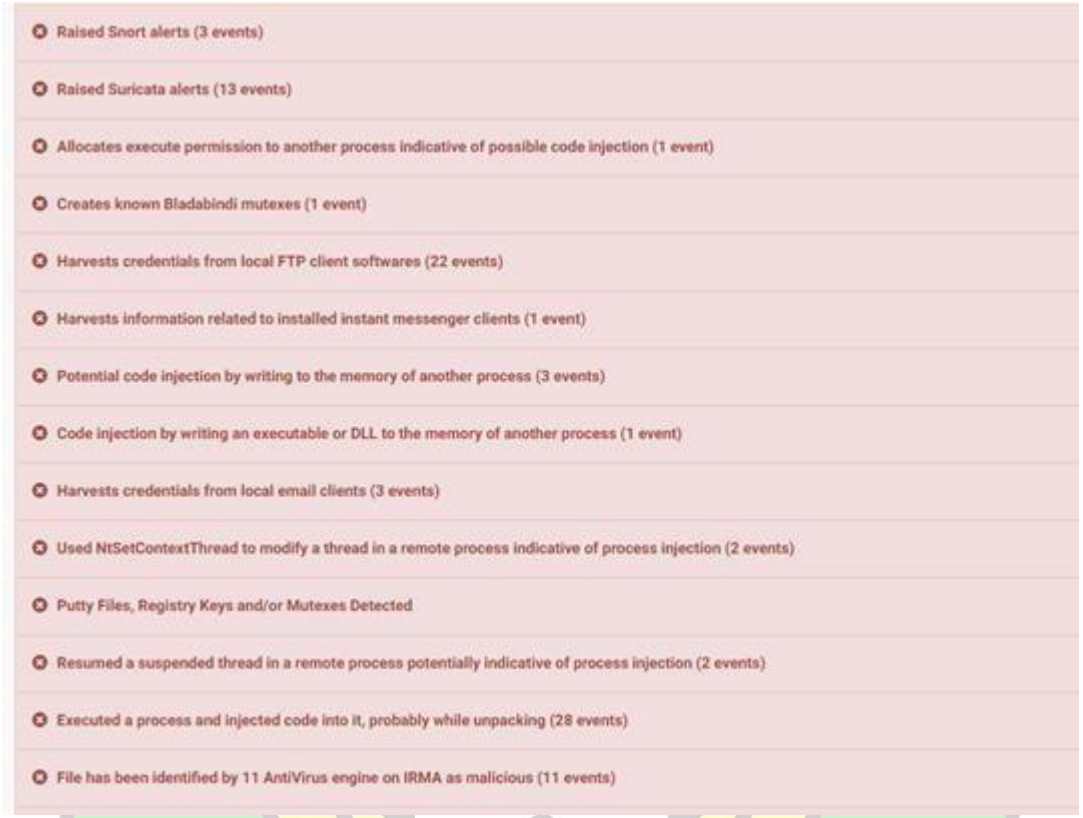
Gambar 4.31 *trojan zombieboy* melakukan ekstraksi pada *buffer* yang berisi data yang menarik atau sensitif, mencuri informasi pribadi pada *browser internet local* (kata sandi, *cookie*, data penjelajahan *web*), *malware* menggunakan *packing* untuk menyembunyikan kode berbahaya, dan masih banyak lagi dapat dilihat seperti Gambar 4.24 dibawah ini.



Gambar 4. 31 Signature medium trojan zombieboy

Pada Gambar 4.32 adalah *signature* yang sangat berbahaya dari *malware trojan zombieboy*, *malware trojan zombieboy* melakukan upaya mengalokasikan izin eksekusi ke proses lain, melakukan upaya untuk mencuri kredensial

(*username* dan *password*), melakukan injeksi kode, *malware* ini mengunakan *Mutex* untuk menghindari konflik antara proses.



Gambar 4. 32 Signature higt trojan zombieboy

Berikut ini adalah penjelasan dari *API library* dan *function call* yang dimiliki oleh *trojan zombieboy*.

Tabel 4. 11 penjelasan *API Library kernel32.dll* dan *Function trojan zombieboy*

Library KERNEL32.dll		
WinExec	0x1000e000	Menjalankan aplikasi yang ditentukan.
CloseHandle	0x1000e0b4	Menutup objek handle yang terbuka
WriteFile	0x1000e0b8	Menulis data pada spesifik *File*
TerminateProcess	0x1000e034	Hentikan proses yang ditentukan dan semua yang berurutan.
GetProcessHeap	0x1000e0f8	*Handle* default heap pada proses panggilan.

Tabel 4. 12 penjelasan *API Library wininet.dll* dan *function trojan zombieboy*

Library WININET.dll		
InternetOpenUrlA	0x1000e114	Membuka sumber yang ditentukan oleh *FTP* lengkap atau *URL HTTP*.
InternetOpenA	0x1000e118	Menginisialisasi penggunaan aplikasi fungsi *WinINet*.
InternetReadFile	0x1000e11c	Membaca data yang dibuka oleh fungsi *InternetOpenUrl*,*FtpOpenFile*, atau *HttpOpenRequest* .
InternetCloseHandle	0x1000e120	Menutup handle Internet.

Berdasarkan Tabel 4.11 *stealer redline* dapat menjalankan aplikasi yang ditentukan (*WinExec*), hentikan proses yang ditentukan dan semua yang berurutan (*TerminateProcess*), mengambil jenis *file* dari *file* yang ditentukan (*GetFileType*) dan masih banyak lagi *function trojan zombieboy* lainnya dapat dilihat pada lampiran 3.

4.3 Pencegahan

Dalam rangka mengantisipasi penyebaran *malware stelaer redline* dan *trojan zombieboy*, berikut ini beberapa rekomendasi pencegahan yang dapat dilakukan:

1) Menggunakan *firewall* untuk memblokir semua koneksi masuk ke layanan yang seharusnya tidak tersedia untuk umum, yang secara *default* menolak semua koneksi yang masuk dan hanya mengizinkan layanan yang digunakan.
2) Melakukan update *Operating System*, Aplikasi/*Software*, *Firmware* dan *Browser* secara berkala untuk meningkatkan keamanan komputer dari kerawanan yang ada.
3) Menggunakan *antivirus* dan perangkat *security* yang *update* dan lakukan scanning anti*virus* baik terhadap *storage* dan *memory* secara berkala.
4) Mematikan fitur *file sharing* jika tidak diperlukan, dan gunakan proteksi kata sandi untuk membatasi akses.

5) Menghindari membuka atau menelusuri situs atau halaman yang tidak jelas dan memiliki reputasi buruk seperti situs bajakan, *keygen*, situs *pornografi*, dsb.

BAB V
PENUTUP

5.1 kesimpulan

Dari analisis yang telah dilakukan diperoleh informasi dari cara *malware* ini melakukan penyerangan, menggunakan fungsi yang bisa mengambil data dan mengontrol perangkat, dan kedua malware ini juga memiliki kemampuan untuk menghindari metode-metode analisis dengan menyembunyikan dirinya. Bedasarkan hasil dan analisis *malware stealer redline* dan *trojan zombieboy*, dapat ditarik kesimpulan sebagai berikut:

1. Proses melakukan analisis *malware* metode statis, pertama menjalankan *tool byte histogram* untuk memperoleh informasi tingkat pengacakan data*malware*, pada penelitian ini tingkat pengacakkan data malware *stealer redline .text* 0%, *.data* 0%, dan *.rdata* 0% dan *trojan zombieboy .text* 62%, *.data* 3% dan *rdata* 28%. Selanjutnya menggunakan *tool pesdtudio* untuk memperoleh informasi spesifik *malware* seperti nilai *Hash*, *MD5*, *SHA*-1, *compiler-stamp*, *string*, *section*, *library*, dan *import* yang digunakan *malware*.
2. Proses analisis *malware* metode dinamis menggunakan *tool cuckoo sandbox*, *malware* tersebut diproses dalam lingkungan *sandbox* yang terisolasi dengan aman dan *cuckoo sandbox* memberikan informasi perilakukam *malware* yang telah dianalisis seperti:
    1) *Stealer redline* menggunakan *Pem-Packer* untuk menyembunyikanatau mengenkripsi kode malware, mengeskploit alat-alat utilitas *windows*, dan mengeksekusi perintah dan skrip melalui *powershell*.
    2) *Trojan zombieboy* mencari informasi instalasi browser pada sistem, menggunakan packing untuk menyembunyikan kode berbahaya, dan mencuri informasi *username* dan *password*.
3. Cara kerja *stealer redline* setelah ketika berhasil menyusupi perangkat korban, *malware* ini menyembunyikan aktivitasnya pada alamat proses yang terinfeksi dengan membuat *file* atau *directory* di dalam sistem, kemudian *malware stealer redline* memcoba mengukur jumlah memori yang tersedia di sistem yang bertujuan untuk mengidentifikasi apakah sistem cukup kuat untuk mengeksekusi beberapa fungsi serta untuk menyesuaikan diri dengan

sumber daya yang tersedia di perangkat, serta menyembunyikan aktivitas dirinya dari perangkat lunak forensik yang bertujuan unutk menghambat proses analisis *malware*, kemudian mengambil fungsi *handel* dari *registry* seperti (menutup handel, menghapus nilai, menghapus nilai kunci, menghapus kunci dari *registry* tertentu) dan melakukan pencurian data sensitif di perangkat berupa *username* dan *password* korban. Berdasarkan informasi yang diperoleh dari analisis metode statis dan dinamis *malware trojan zombieboy* ketika berhasil menginjeksi perangkat, *malware* ini akan memberikan notifikasi kepada *hacker* yang mengendalikannya, kemudian *malware trojan zombieboy* berusaha mengumpulkan informasi sistem yang berguna untuk membuat *system fingerprint* seperti (versi sistem operasi, konfigurasi *hardware*, dan informasi lainnya), *malware* ini mencoba mencuri informasi pribadi dari *browser* yang digunakan, bahkan dapat mengambil alihkontrol pada perangkat lunak dan perangkat keras komputer yang terinfeksi dan dapat dimanipulasikan oleh pernyerang.

Dalam proses analisis *malware* metode statis dan dinamis penting untuk diingat bahwa analisis *malware* adalah sebuah tugas yang berpotensi berbahaya, sehingga dalam proses analisis harus beroperasi dalam lingkungan yang sepenuhnya terisolasi untuk melindungi sistem dan data pada perangkat yang digunakan. Selain itu menjaga rekam jejak dan dokumentasi yang baik sepanjang proses analisis, karena ini akan membantu dalam proses pelacakan, pelaporan, dan perbaikan keamanan pada masa mendatang.

5.2 Saran

Penelitian ini masih banyak terdapat kekurangan karena analisis dilakukan termasuk analsis dasar dalam melakukan analisis pada *malware stealer redline* dan *trojan zombieboy* sehingga membutuhkan penambahan yang lebih baik lagi untuk mengahsilkan laporan analisis yang lebih baik dan dapat dimengerti oleh orang awam. Oleh karena itu, untuk penelitian kedepan disarankan:

1. Mengumpulkan sampel malware dengan famili yang sama untuk memudahkan pemetaan malware
2. Mempelajari lebih dalam tentang malware stealer redline dan *trojan zombieboy* serta teknik analisis *malware*

Saran yang peneliti berikan dikarenakan hasil analisis menunjukkan bahwa penelitian yang dilakukan hanya menyentuh dasar dari malware. Sedangkan terdapat metode analisis lainnya yang dapat mengetahui struktural dan cara kerja *malware stealer redline* dan *trojan zombieboy* dengan lebih dalam.

DAFTAR PUSTAKA

Christian Wojner. (n.d.). *Bytehist*. Retrieved December 20, 2023, from https://cert.at/en/downloads/software/software-*byte*hist

Claudio Guarnieri, A. T. J. B. M. S. (2019, June 19). *cuckoosandbox*. https://cuckoosandbox.org/

Daniswara, D. A., Budiono, A., Almaarif, A., & Kom, S. (2019). Analisis Deteksi Malicious Activity Menggunakan Metode Analisis Malware Dinamis Berbasis Anomaly Detection Analysis of Malicious Activity Using Anomaly-Based Dynamic Malware Analysis Method. *2019, e-Proceeding of Engineering : Vol.6*, *6*(2), 7796–7803. https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/10635

DarkTracer. (2022). *Intelligence Report*. https://twitter.com/stealthmole_int/status/1511982449823449091

Joana Gaia, B. R. (2020). *Hawaii International Conference on System Sciences 2020*. https://scholarspace.manoa.hawaii.edu/handle/10125/64014

Komputer, J. (2020). *Fakultas Komputer INDAH KUSUMA ASTUTI Section 01*. https://scholar.google.com/scholar?hl=id&as_sdt=0%2C5&q=jaringan+komputer+indah+kusuma&btnG=&oq=JARINGAN+KOMPUTER+indah

linuxhackingid. (2023). *Prinsip Keamanan Jaringan*. Www.Linuxhacking.or.Id. https://linuxhacking.or.id/lessons/prinsip-keamanan-jaringan/

Madina Nusrat. (2022, April 11). *BSSN Temukan 490 Domain Pemerintah Jadi Sasaran Pencurian Data*. https://www.kompas.id/baca/hukum/2022/04/11/menginfeksi-perangkat-pengguna-bssn-temukan-460-domain-pemerintah-jadi-sasaran-pencurian-data

Muhyidin, Y., Hafid Totohendarto, M., Undamayanti, E., & Tinggi Teknologi Wastukancana, S. (2022). *Perbandingan Tingkat Keamanan Website Menggunakan Nmap Dan Nikto Dengan Metode Ethical Hacking Comparison of Website Security Levels Using Nmap and Nikto With Ethical Hacking Methods*. https://jurnal.wastukancana.ac.id/index.php/teknologika/article/view/143

Novansyah, H., & Sutabri, T. (2023). ANALISIS MALWARE DENGAN METODE DINAMIK MENGGUNAKAN FRAMEWORK CUCKOO SANDBOX. *Blantika : Multidisciplinary Jornal*, *2*(1). https://blantika.publikasiku.id/

Ramadhani, S., Sultan Syarif Kasim Alamat, U., Koto Kociak Kecamatan Latina Payakumbuh Sumatera Barat, J., Soebrantas Kelurahan Simpang Baru No, J. H., & Tampan, K. (2018). Analisis Sistem Keamanan Web Server Dan Database Server Menggunakan Suricata. *Seminar Nasional Teknologi Informasi Komunikasi Dan Industri*, *0*(0), 308–317. http://ejournal.uin-suska.ac.id/index.php/SNTIKI/article/view/3368

sainadh jamalpur. (2018). *Proceedings of the International Conference on Inventive Communication and Computational Technologies : ICICCT 2018 : 20-21, April 2018*. https://ieeexplore.ieee.org/abstract/document/8473346

Studi Sistem dan Teknologi Informasi, P. (n.d.). *Penggunaan Tiger Hash pada Tanda Tangan Digital Madiha Ainayya Faizzati (18218010)*. https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-

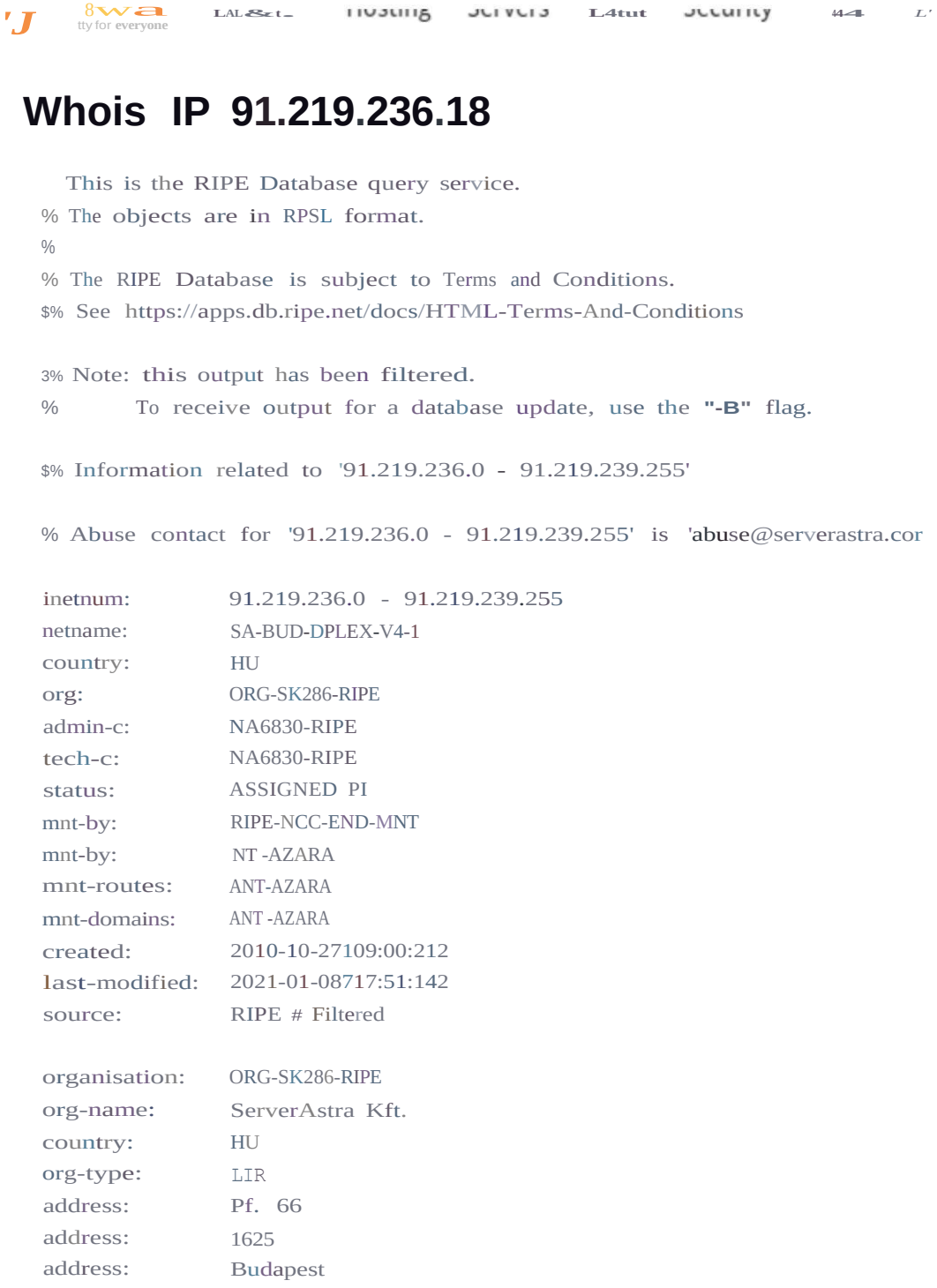
virgiawan Arshad Manoppo. (2020). *jm_elektro,+29567-65818-1-CE*. 181–188. https://ejournal.unsrat.ac.id/index.php/elekdankom/article/view/29567

Wahyu Wahidin, G., & Sari, Z. (2022). Analisis Ransomware Wannacry Menggunakan Aplikasi Cuckoo Sandbox. *REPOSITOR*, *4*(1), 83–94. https://www.sttrcepu.ac.id/jurnal/index.php/simetris/article/view/172

WS Ibraheem, A. M. (2021). *PERT 2 OSI TCP_IP LAYER*.

LAMPIRAN

Lampiran 1 Informasi *IP address* yang melakukan komunikasi



Whois IP 91.219.236.18

This is the RIPE Database query service.
% The objects are in RPSL format.
%
% The RIPE Database is subject to Terms and Conditions.
$% See https://apps.db.ripe.net/docs/HTML-Terms-And-Conditions

3% Note: this output has been filtered.
% To receive output for a database update, use the "-B" flag.

$% Information related to '91.219.236.0 - 91.219.239.255'

% Abuse contact for '91.219.236.0 - 91.219.239.255' is 'abuse@serverastra.cor

inetnum:	91.219.236.0 - 91.219.239.255
netname:	SA-BUD-DPLEX-V4-1
country:	HU
org:	ORG-SK286-RIPE
admin-c:	NA6830-RIPE
tech-c:	NA6830-RIPE
status:	ASSIGNED PI
mnt-by:	RIPE-NCC-END-MNT
mnt-by:	NT -AZARA
mnt-routes:	ANT-AZARA
mnt-domains:	ANT -AZARA
created:	2010-10-27109:00:212
last-modified:	2021-01-08717:51:142
source:	RIPE # Filtered
organisation:	ORG-SK286-RIPE
org-name:	ServerAstra Kft.
country:	HU
org-type:	LIR
address:	Pf. 66
address:	1625
address:	Budapest

Gambar L. 1 whois Ip address 91.219.236.18

```
address:        Budapest
address:        HUNGARY
phone:          +3619990149
admin-c:        NA6830-RIPE
tech-c:         NA6830-RIPE
abuse-c:        AR61142-RIPE
mnt-ref:        MNT-AZARA
mnt-by:         RIPE-NCC-HM-MNT
mnt-by:         MNT-AZARA
created:        2020-10-12T10:33:08Z
last-modified:  2023-06-29T14:40:23Z
source:         RIPE # Filtered

role:           NETOPS
address:        Pf. 66
address:        1625
address:        Budapest
address:        HUNGARY
phone:          +3619990149
nic-hdl:        NA6830-RIPE
mnt-by:         MNT-AZARA
created:        2020-10-12T10:33:08Z
last-modified:  2023-06-29T14:39:33Z
source:         RIPE # Filtered

% Information related to '91.219.236.0/24AS56322'

route:          91.219.236.0/24
origin:         AS56322
mnt-by:         MNT-AZARA
created:        2022-09-08T12:30:47Z
last-modified:  2022-09-08T12:30:47Z
source:         RIPE

% This query was served by the RIPE Database Query Service version 1.108 (ABERDEEN)
```

Gambar L. 2 whois Ip address 91.219.236.18

Whois IP **212.193.30.45**

Uodated 1

$ This is the RIPE Database query service.
% The objects are in RP5L format.
$
% The RIPE Database is subject to Terms and Conditions.
$ gee https://spps.db.ripe.net/daC/HTML -Term-And-Conditions

$ Mote: this output has been filtered.
% To receive output for a database update, use the -s'' flag.

Information related to '212.193.30.0 - 212.193.30 255'

$ Abuse contact for '212.193.30.0 • 212.193.30.255' is 'ripe@interlir.con'

inetnum:	212.193.30.0 - 212.193,30.255
netnare:	DC-DST-IC
org:	0R DHII RIPE
country:	US
din-c:	**8813533-RIP£**
tech-c:	5513533-RIPE
abusec:	**AR6 281-RIP£**
status;	ASSIGNED PA
mnt-by:	interlir-mnt
created:	2023-11-13714:22:152
lest-modified:	2023-11-13714:22 152
	RI 'E

organisation:	0RG DHII RIPE
org-name'	DC Host IC
country:	Us
org-type	0TER
address:	**1309** Coffeen Avenue STE 1200 Sheridan, **wY 82881** United **States**
admin-c:	5513533-RIPE
abuse-c:	D+128-RIPE
mnt ref;	CIKLET-NT
mt-ref_	interlir-mnt
[illegible]	

```
country:        US
org-type:       OTHER
address:        1309 Coffeen Avenue STE 1200 Sheridan, WY 82801 United States
admin-c:        BB13533-RIPE
abuse-c:        DMI20-RIPE
mnt-ref:        CIKLET-MNT
mnt-ref:        interlir-mnt
mnt-ref:        GEO-MNT
mnt-ref:        ADEOX
created:        2022-02-16T07:55:15Z
last-modified:  2023-08-23T13:17:38Z
source:         RIPE # Filtered
mnt-by:         CIKLET-MNT
mnt-by:         DCHost-MNT

person:         Berkay Bulut
address:        1309 Coffeen Avenue STE 1200 Sheridan, WY 82801 United States
phone:          +1-302-208-6020
nic-hdl:        BB13533-RIPE
created:        2022-02-16T07:53:58Z
last-modified:  2023-04-05T02:12:41Z
source:         RIPE
mnt-by:         DCHost-MNT

% Information related to '212.193.30.0/24AS208287'

route:          212.193.30.0/24
origin:         AS208287
mnt-by:         interlir-mnt
created:        2023-11-13T14:23:11Z
last-modified:  2023-11-13T14:23:11Z
source:         RIPE

% This query was served by the RIPE Database Query Service version 1.108 (BUSA)
```

Gambar L. 3 whois Ip address 212.193.30.45

Whois IP 74.125.24.94

```
#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/resources/registry/whois/tou/
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2023, American Registry for Internet Numbers, Ltd.
#


NetRange:       74.125.0.0 - 74.125.255.255
CIDR:           74.125.0.0/16
NetName:        GOOGLE
NetHandle:      NET-74-125-0-0-1
Parent:         NET74 (NET-74-0-0-0-0)
NetType:        Direct Allocation
OriginAS:
Organization:   Google LLC (GOGL)
RegDate:        2007-03-13
Updated:        2012-02-24
Ref:            https://rdap.arin.net/registry/ip/74.125.0.0


OrgName:        Google LLC
OrgId:          GOGL
Address:        1600 Amphitheatre Parkway
City:           Mountain View
StateProv:      CA
PostalCode:     94043
Country:        US
RegDate:        2000-03-30
Updated:        2019-10-31
Comment:        Please note that the recommended way to file abuse complaints
```

```
City:           Mountain View
StateProv:      CA
PostalCode:     94043
Country:        US
RegDate:        2000-03-30
Updated:        2019-10-31
Comment:        Please note that the recommended way to file abuse complaints are located in th
Comment:
Comment:        To report abuse and illegal activity: https://www.google.com/contact/
Comment:
Comment:        For legal requests: http://support.google.com/legal
Comment:
Comment:        Regards,
Comment:        The Google Team
Ref:            https://rdap.arin.net/registry/entity/GOGL


OrgAbuseHandle: ABUSE5250-ARIN
OrgAbuseName:   Abuse
OrgAbusePhone:  +1-650-253-0000
OrgAbuseEmail:  network-abuse@google.com
OrgAbuseRef:    https://rdap.arin.net/registry/entity/ABUSE5250-ARIN

OrgTechHandle: ZG39-ARIN
OrgTechName:   Google LLC
OrgTechPhone:  +1-650-253-0000
OrgTechEmail:  arin-contact@google.com
OrgTechRef:    https://rdap.arin.net/registry/entity/ZG39-ARIN


#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/resources/registry/whois/tou/
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2023, American Registry for Internet Numbers, Ltd.
#
```

Gambar L. 4 Gambar L. 3 whois Ip address 74.125.24.94

Lampiran 2 Hasil *section malware stealer redline*

Tabel L. 1 Hasil *section malware stealer redline*

Library KERNEL32.dll		
GetExitCodeProcess	0x408070	Mengambil status penghentian dari proses yang ditentukan
WaitForSingleObject	0x408074	Menunggu hingga objek yang ditentukan berada dalam status bersinyal atau selang waktu habis.
GetModuleHandleA	0x408078	Mengambil handle modul untuk modul yang ditentukan
GetProcAddress	0x40807c	Mengambil alamat fungsi atau variabel yang diekspor dari pustaka tautan dinamis (*DLL*) yang ditentukan.
GetSystemDirectoryW	0x408080	Mengambil jalur direktori sistem. Direktori sistem berisi *file* sistem seperti pustaka tautan dinamis dan driver.
lstrcatW	0x408084	menambahkan satu string ke string lainnya.
Sleep	0x408088	Menangguhkan eksekusi utas saat ini hingga interval waktu habis berlalu.
lstrcpyA	0x40808c	Menyalin *string* ke *buffer*
WriteFile	0x408090	menulis data ke *file* yang ditentukan atau perangkat input / output (I / O)
GetTempFileNameW	0x408094	Membuat nama untuk *file* sementara
lstrcmpiA	0x408098	Membandingkan dua string karakter. Perbandingannya tidak peka huruf besar/kecil.
RemoveDirectoryW	0x40809c	Menghapus direktori kosong yang ada.
CreateProcessW	0x4080a0	Membuat proses baru dan utas utamanya. Proses baru berjalan dalam konteks keamanan proses pemanggilan.

Library KERNEL32.dll		
CreateDirectoryW	0x4080a4	Membuat direktori baru. Jika sistem *file* yang mendasari mendukung keamanan pada *file* dan direktori, fungsi menerapkan deskriptor keamanan tertentu ke direktori baru.
GetLastError	0x4080a8	Mengambil nilai kode kesalahan terakhir thread panggilan. Kode kesalahan terakhir dipertahankan berdasarkan *per-thread*. Beberapa utas tidak menimpa kode kesalahan terakhir masing-masing.
CreateThread	0x4080ac	Membuat utas untuk dieksekusi dalam ruang alamat *virtual* dari proses panggilan.
GlobalLock	0x4080b0	Mengunci objek memori global dan mengembalikan penunjuk ke *byte* pertama dari blok memori objek.
GlobalUnlock	0x4080b4	Decrements jumlah kunci yang terkait dengan objek memori yang dialokasikan dengan *GMEM_MOVEABLE*. Fungsi ini tidak berpengaruh pada objek memoriyang dialokasikan dengan *GMEM_FIXED*.
GetDiskFreeSpaceW	0x4080b8	Mengambil informasi tentang disk yang ditentukan, termasuk jumlah ruang kosong pada disk.
WideCharToMultiByte	0x4080bc	Memetakan string karakter ke string *UTF*-16 (karakter lebar). String karakter tidak harus dari set karakter multi*byte*.
.lstrcpynW	0x4080c0	Menyalin sejumlah karakter tertentu dari string sumber ke buffer
lstrlenW	0x4080c4	Menentukan panjang string yang ditentukan (tidak termasuk karakter null yang mengakhiri).

Library KERNEL32.dll		
SetErrorMode	0x4080c8	Mengontrol apakah sistem akanmenangani jenis kesalahan serius yang ditentukan atau apakah proses akan menanganinya
GetVersion	0x4080cc	Mengembalikan versi yang dimanifestasikan aplikasi dalam rilis mendatang
GetCommandLineW	0x4080d0	Mengambil string baris perintah untuk proses saat ini
GetTempPathW	0x4080d4	Mengambil jalur direktori yang ditunjuk untuk *file-file* sementara
GetWindowsDirectoryW	0x4080d8	Mengambil jalur direktori *Windows*.
SetEnvironmentVariableW	0x4080dc	Atur konten *variabel* lingkungan yang ditentukan untuk proses saat ini.
ExitProcess	0x4080e0	Mengakhiri proses pemanggilan dan semua utasnya
CopyFileW	0x4080e4	Melakukan penyalinan *file*
GetCurrentProcess	0x4080e8	Mengambil pegangan pseudo untuk proses saat ini.
GetModuleFileNameW	0x4080ec	Mengambil jalur yang memenuhi syarat untuk *file* yang berisi modul yang ditentukan.
GetFileSize	0x4080f0	Mengambil ukuran *file* yang ditentukan, dalam *byte*.
CreateFileW	0x4080f4	Membuat atau membuka *file* atau perangkat I/O.
GetTickCount	0x4080f8	Mengambil jumlah milidetik yang telah berlalu sejak sistem dimulai, hingga 49,7 hari.

Library KERNEL32.dll		
MulDiv	0x4080fc	Mengalikan dua nilai 32-*bit* dan kemudian membagi hasil 64-*bit* dengan nilai 32-*bit* ketiga. Hasil akhir dibulatkan ke bilangan bulat terdekat.
SetFileAttributesW	0x408100	Menentukan atribut untuk suatu *file* atau direktori
GetFileAttributesW	0x408104	Mengambil atribut untuk suatu *file* atau direktori
SetCurrentDirectoryW	0x408108	Menentukan direktori saat ini
MoveFileW	0x40810c	Memindahkan *file* ke destinasi lain
GetFullPathNameW	0x408110	Mendapatkan alamat destinasi lengkap
GetShortPathNameW	0x408114	Mendapatkan alamat destinasi singka
SearchPathW	0x408118	Mencari alamat destinasi
CompareFileTime	0x40811c	Membandingkan dua waktu dari suatu *file*
SetFileTime	0x408120	Menentukan waktu dari suatu *file*
CloseHandle	0x408124	Menutup kendali dari suatu proses
lstrcmpiW	0x408128	Membandingkan dua string karakter. Perbandingannya tidak peka huruf besar/kecil.
lstrcmpW	0x40812c	Membandingkan dua string karakter. Perbandingannya tidak peka huruf besar/kecil.
ExpandEnvironment StringsW	0x408130	Memperluas *string variabel* lingkungan dan menggantinya dengan nilai yang ditentukan untuk pengguna saat ini.
GlobalFree	0x408134	Membebaskan objek memori global yang ditentukan dan membatalkan pegangannya
GlobalAlloc	0x408138	Mengalokasikan jumlah *byte* yang ditentukan dari tumpukan
GetModuleHandleW	0x40813c	Mendapatka kendali dari suatu modul

Library KERNEL32.dll		
LoadLibraryExW	0x408140	Memuat *library* dari suatu proses
MoveFileExW	0x408144	Memindahkan suatu *file* ke destinasi lain
FreeLibrary	0x408148	Membebaskan modul dynamic-link library (*DLL*) yang dimuat dan, jika perlu, mengurangi jumlah referensinya
WritePrivateProfile StringW	0x40814c	Menyalin string ke bagian yang ditentukan dari *file* inisialisasi
RitePrivateProfile StringW	0x408150	Mengambil string dari bagian yang ditentukan dalam *file* inisialisasi.
lstrlenA	0x408154	Menentukan panjang string yang ditentukan (tidak termasuk karakter *null* yang mengakhiri).
MultiByteToWideChar	0x408158	Memetakan string karakter ke string UTF-16 (karakter lebar). String karakter tidak harus dari set karakter multi*byte*
ReadFile	0x40815c	Melakukan pembacaan terhadap suatu *file*
SetFilePointer	0x408160	Menentukan penunjuk *file* yang telah ditentukan
FindClose	0x408164	Menutup kendali pencarian *file*
FindNextFileW	0x408168	Melakukan pencarian *file* dari panggilan sbelumnya
FindFirstFileW	0x40816c	Melakukan pencarian *file* dengan spesifik nama
DeleteFileW	0x408170	Menghapus *file* tertentu

Lampiran 3 Hasil *section malware trojan zombieboy*

Tabel L. 2 Hasil *section malware trojan zombieboy*

Library KERNEL32.dll		
WinExec	0x1000e000	Menjalankan aplikasi yang ditentukan.
DecodePointer	0x1000e004	Menlakukan *decode pointer* yang sebelumnya di-*encode* dengan *EncodePointer*.
ReadConsoleW	0x1000e008	Membaca input karakter dari *buffer input console* dan menghapusnya dari buffer.
ReadFile	0x1000e00c	Membaca data dari *file* atau input/output (I /O) yang ditentukan. Pembacaan terjadi pada posisi yang ditentukan oleh pointer *file* jika didukung oleh perangkat. Fungsi ini dirancang untuk operasi sinkron dan asinkron.
SetEndOfFile	0x1000e010	Menetapkan ukuran *file* fisik untuk *file* yang ditentukan ke posisi saat ini daripointer *file*.
HeapReAlloc	0x1000e014	Alokasi ulang blok memori dari tumpukan. Fungsi ini memungkinkan untuk mengubah ukuran blok memori dan mengubah properti blok memorilainnya.
HeapSize	0x1000e018	Mengambil ukuran blok memori yang dialokasikan dari heap oleh fungsi *HeapAlloc* atau *HeapReAlloc*.
WriteConsoleW	0x1000e01c	Menulis string karakter ke buffer layar konsol yang dimulai pada lokasi kursor saat ini.
SetFilePointerEx	0x1000e020	Memindahkan pointer *file* dari *file* yang ditentukan.
FlushFileBuffers	0x1000e024	Flush buffer dari *file* yang ditentukan dan menyebabkan semua data buffered ditulis ke*file*.

Library KERNEL32.dll		
UnhandledExceptionFilter	0x1000e028	Fungsi yang ditentukan aplikasi yang memberikan pengecualian yang tidak ditangani ke debugger, jika prosesnya sedang di-*debug*. Jika tidak, secaraopsional
SetUnhandledExceptionFilter	0x1000e02c	Mengaktifkan aplikasi untuk menggantikan *handle exception* dari setiap proses thread.
GetCurrentProcess	0x1000e030	Mengambil pseudo *handle* untuk proses yang berlangsung.
TerminateProcess	0x1000e034	Hentikan proses yang ditentukan dan semua yang berurutan.
QueryPerformanceCounter	0x1000e03c	Mengambil nilai saat ini dari performance counter, yang merupakan cap waktu resolusi tinggi (<1us) yang dapat digunakan untuk pengukuran interval waktu.
GetCurrentProcessId	0x1000e040	Mengambil id pengidentifikasi proses
GetCurrentThreadId	0x1000e044	Mengambil thread identifier dari calling thread.
GetSystemTimeAsFileTime	0x1000e048	Mengambil tanggal dan waktu sistem saat ini. Informasi ini dalam format Waktu *UTC*.
InitializeSListHead	0x1000e04c	Menginisialisasi daftar list yang dilinkkan
IsDebuggerPresent	0x1000e050	Menentukan apakah proses panggilan sedang di-*debug* oleh user-mode *debugger*
GetStartupInfoW	0x1000e054	Mengambil kembali isi dari struktur *STARTUPINFO* yang ditentukan saat proses pemanggilan dibuat.

Library KERNEL32.dll		
GetModuleHandleW	0x1000e058.	Handle modul untuk modul yang ditentukan. Modul harus dimuat oleh proses pemanggilan
InterlockedFlushSList	0x1000e05c	Menghapus semua item dari daftar yang ditautkan sendiri. Akses ke daftar disinkronkan pada sistem multiprosesor.
RtlUnwind	0x1000e060	Memulai pelepasan kerangka panggilan prosedur.
GetLastError	0x1000e064	Mengambil nilai thread's *last-error* code value. Beberapa *thread* tidak saling menimpa last-error code terakhir masing-masing.
SetLastError	0x1000e068	Menetapkan last-error code untuk panggilan thread
EnterCriticalSection	0x1000e06c	Menunggu kepemilikan objek bagian kritis yang ditentukan.
LeaveCriticalSection	0x1000e070	Merilis kepemilikan objek bagian kritis yang ditentukan.
DeleteCriticalSection	0x1000e074	Merilis semua sumber daya yang digunakan oleh objek bagian kritis yang tidak dimiliki
InitializeCriticalSectionAndSpinCount	0x1000e078	Menginisialisasi objek bagian kritis dan menetapkan jumlah putaran untuk bagian kritis.
TlsAlloc	0x1000e07c.	Mengalokasikan indeks *thread local storage* (*TLS*)

Library KERNEL32.dll		
TlsGetValue	0x1000e080	Mengambil nilai dalam slot *thread local storage* (*TLS*) panggilan untuk indeks *TLS* yang ditentukan
TlsSetValue	0x1000e084	Menyimpan nilai dalam slot *thread local storage* (*TLS*) untuk indeks *TLS* yang ditentukan.
TlsFree	0x1000e088	Merilis indeks *thread local storage* (*TLS*), sehingga tersedia untuk digunakan kembali.
FreeLibrary	0x1000e08c	Membebaskan modul *dynamic-link library* (*DLL*) yang dimuat serta mengurangi jumlah referensi. Ketika jumlah referensi mencapai nol, modul dilepaskan dari ruang alamat dari proses panggilan dan handle tidak lagi *valid*.
GetProcAddress	0x1000e090	Mengambil alamat fungsi atau variabel yang diekspor dari *dynamic-link library* (*dll*) yang ditentukan.
ExitProcess	0x1000e098	Mengakhiri proses pemanggilan dansemua thread.
GetModuleHandleExW	0x1000e09c	Mengambil handle modul untuk modul yang ditentukan dan menambah jumlah referensi modul
GetModuleFileNameA	0x1000e0a0	Mengambil jalur yang sepenuhnya memenuhi syarat untuk *file* yang berisi modul yang ditentukan. Modul harus dimuat oleh proses yang berlangsung.
MultiByteToWideChar	0x1000e0a4	Memetakan *string karakter* ke *string UTF*-16 (karakter lebar).

Library KERNEL32.dll		
WideCharToMultiByte	0x1000e0a8	Memetakan string *UTF*-16 (karakter llebar) ke string karakter baru. String karakter baru tidak harus dari set karakter multi*byte*.
HeapFree	0x1000e0ac	Membebaskan blok memori yang dialokasikan dari tumpukan oleh fungsi *HeapAlloc* atau *HeapReAlloc*.
HeapAlloc	0x1000e0b0	Mengalokasikan blok memori dari heap. Memori yang dialokasikan tidak dapat dipindahkan
CloseHandle	0x1000e0b4	Menutup objek handle yang terbuka
WriteFile	0x1000e0b8	Menulis data pada spesifik *File*
GetConsoleCP	0x1000e0bc	Mengambil kembali halaman kode input yang digunakan oleh konsol yang terkait dengan proses panggilan.
GetStringTypeW	0x1000e0c4	Mengambil informasi jenis karakter untuk karakter dalam string sumber *Unicode* yang ditentukan.
GetACP	0x1000e0c8	Mengambil pengenal halaman kode *ANSI Windows* untuk sistem operasi.
LCMapStringW	0x1000e0cc	Untuk lokal yang ditentukan oleh pengidentifikasi, memetakan satu stringkarakter input ke yang lain menggunakan transformasi yang ditentukan, atau menghasilkan kunci pengurutan untuk string input.

Library KERNEL32.dll		
FindClose	0x1000e0d0	Menutup handle pencarian *file* yang dibuka oleh fungsi *FindFirstFile,FindFirstFileEx, FindFirstFileNameW,FindFirstFileNameTransactedW,FindFirstFileTransacted,FindFirstStreamTransactedW*, atau *FindFirstStreamW*.
FindFirstFileExA	0x1000e0d4	Mencari direktori untuk *file* atau subdirektori dengan nama dan atribut yang cocok dengan yang ditentukan.
FindNextFileA	0x1000e0d8	Melanjutkan pencarian *file* dari panggilan sebelumnya ke *FindFirstFile,FindFirstFileEx*, atau *fungsiFindFirstFileTransacted*.
IsValidCodePage	0x1000e0dc	Menentukan apakah halaman kode yang ditentukan valid.
GetOEMCP	0x1000e0e0	Mengambil identifikasi halaman kode *Original Equipment Manufacturer* (*OEM*) untuk sistem operasi.
GetCPInfo	0x1000e0e4	Mengambil informasi tentang halaman kode yang diinstal atau tersedia yang valid.
GetCommandLineA	0x1000e0e8	Mengambil string command-line untuk proses yang berlangsung
GetCommandLineW	0x1000e0ec	Mengambil *string* baris perintah untuk proses saat ini.
GetEnvironmentStringsW	0x1000e0f0	Mengambil variabel *environment* yangsedang berlangsung.
FreeEnvironmentStringsW	0x1000e0f4	Membebaskan satu blok *environment* string.

Library KERNEL32.dll		
GetProcessHeap	0x1000e0f8	*Handle* default heap pada proses panggilan.
GetStdHandle	0x1000e0fc	*Handle* ke perangkat standar yang ditentukan (input standar, output standar, atau kesalahan standar).
GetFileType	0x1000e100	Mengambil jenis *file* dari *file* yang ditentukan.
CreateFileW	0x1000e104	Fungsi mengembalikan alih yang dapat digunakan untuk mengakses *file* atau perangkat untuk berbagai jenis I/O tergantung pada *file* atau perangkat dan flag serta atribut yang ditentukan.
SetStdHandle	0x1000e108	Mengatur handle untuk perangkat standar yang ditentukan (input standar, output standar, atau kesalahan standar).
RaiseException	0x1000e10c	*Raise exceptiom* pada *calling thread*.

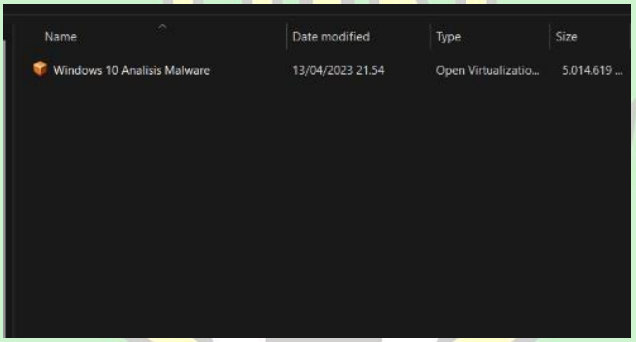
Tabel *L. 3* Hasil *section malware trojan zombieboy Libary Wininet.dll*

Library WININET.dll		
InternetOpenUrlA	0x1000e114	Membuka sumber yang ditentukan oleh *FTP* lengkap atau *URL HTTP*.
InternetOpenA	0x1000e118	Menginisialisasi penggunaan aplikasi fungsi *WinINet*.
InternetReadFile	0x1000e11c	Membaca data yang dibuka oleh fungsi *InternetOpenUrl,FtpOpenFile,* atau *HttpOpenRequest* .

InternetCloseHandle	0x1000e120	Menutup *handle* Internet.

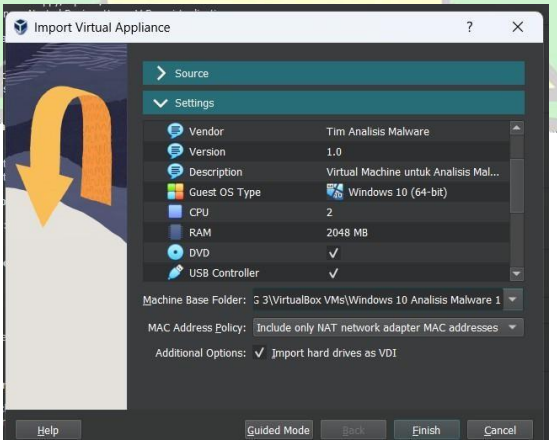
Lampiran 4 *Proses* Installasi Windows 10

1. Langkah pertama dalam melakukan analisis malware perlu dipersiapkan beberapa software yang dibutuhkan seperti operasi sistem, *virtualbox*, sampel *malware* (dapat diperoleh dari malshare, repositori malware, dll), dan *tool* analisis *malware* (*bytehistogram, pestudio* dan *cuckoo sandbox*). Pertama kali yang harus dilakukan adalah installasi operasi sistem *windows* untuk lingkungan analisis malware dan *windows* yang digunakan pada penelitian ini adalah *windows* 10, karena *file* yang digunakan tipe *file windows image*, untuk installasi operasi sistem cukup mudah tanpa harus melakukan proses installasi seperti biasanya. Langkah pertama installasi windows **klik 2** kali pada *file* operasi windows, setelah itu akan di arahkan ke software *virtual*box untuk tahapan finishing installasi.



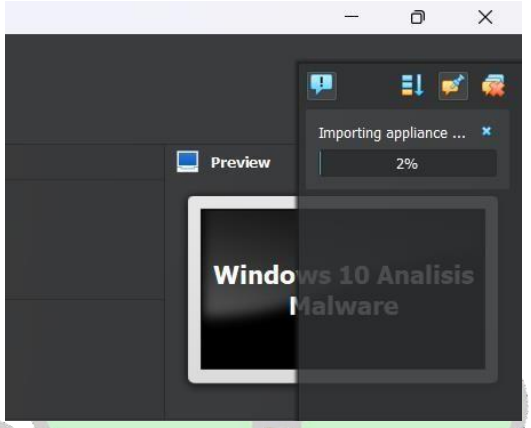
Gambar L. 5 file operasi *windows*

2. Gambar dibawah ini adalah proses finishing installasi windows 10, klik *finish* untuk melanjutkan.



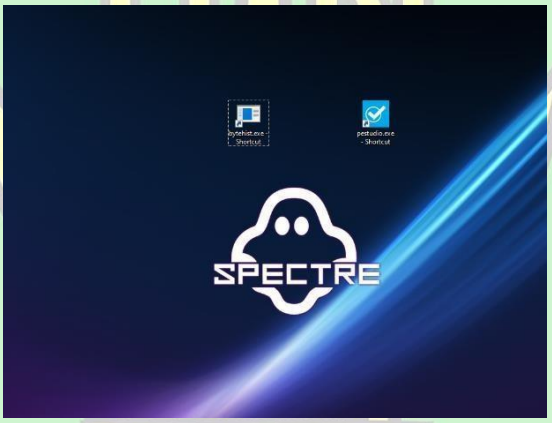
Gambar L. 6 proses installasi *windows*

3. Setelah itu installasi windows akan diproses, untuk proses installasi dapatdilihat pada gambar di bawah ini.



Gambar L. *7* Oroses installasi *windows*

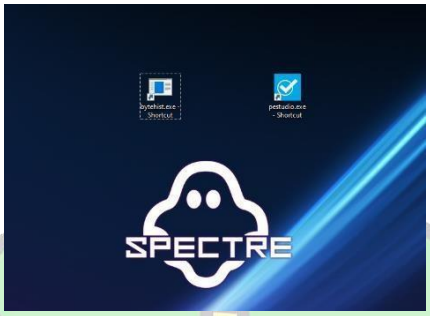
4. Setelah proses *import* selesai, *virtualbox* akan mengarahkan ke operasi *windows* dan telah siap untuk digunakan.



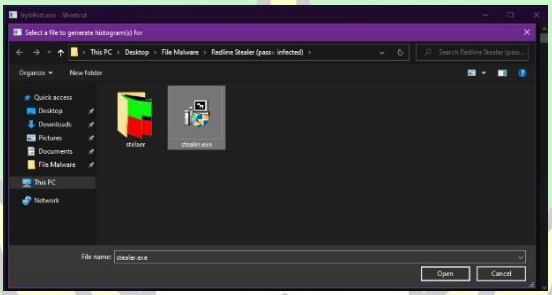
Gambar L. 8 Operasi *windows* selesai di-*instal*

Lampiran 5 Analisis *statis tool byte histogram*

1. Tahapan pertama analisis statis menjalankan *tool byte histogram*, **Klil 2** pada *ikon byte histogram*, selanjutnya akan diarahkan untuk mengambil sampel *malware* yang ingin analisis, "**klik sampel *malware*+*Open*"**.

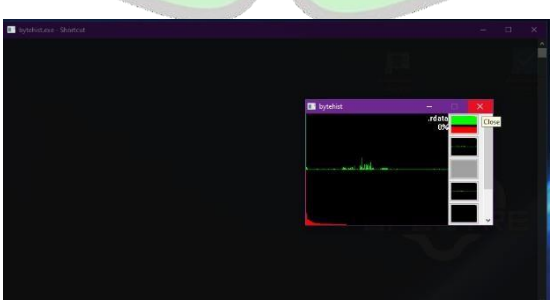


Gambar L. 9 analisis statis

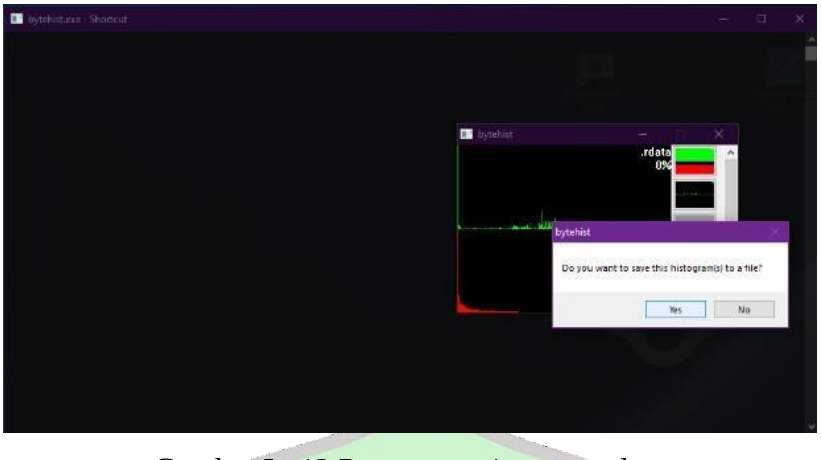


Gambar L. 10 Pilih sampel *malware*

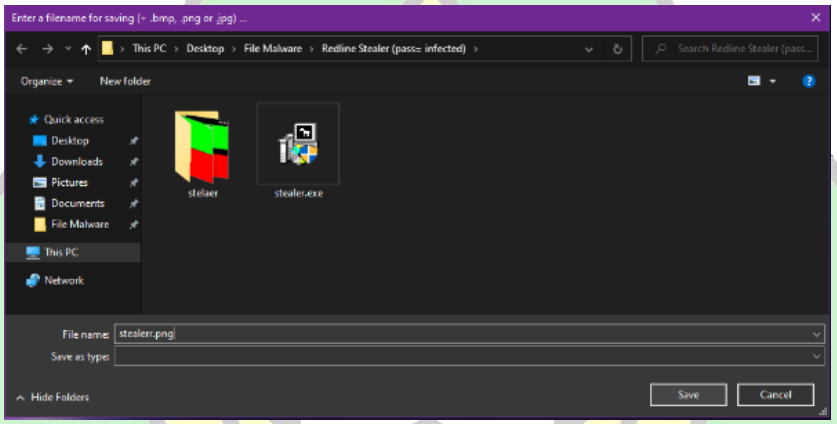
2. Setalah itu akan ditampil hasil pengacakkan data *malware*, ada beberapa macam *type* pengacakkan data yang ditampilkan *byte histogram* yaitu *rdata*, *data*, dan *text*. Cara memnyimpan hasil analisis ini **"Klil *close*+*Yes*"** setelah itu akan diarahkan ke dokumen yang ingin disimpan. Agar data ini tersimpan setiap kali melakukan *save* hasil analisis perlu dituliskan jenis *file* yang ingin disimpan seperti pada Gambar L. 13 dengan menuliskan nama *file* serta jenis *file*-nya "stealerr.png"



Gambar L. 11 Hasil analisis *byte histogram*

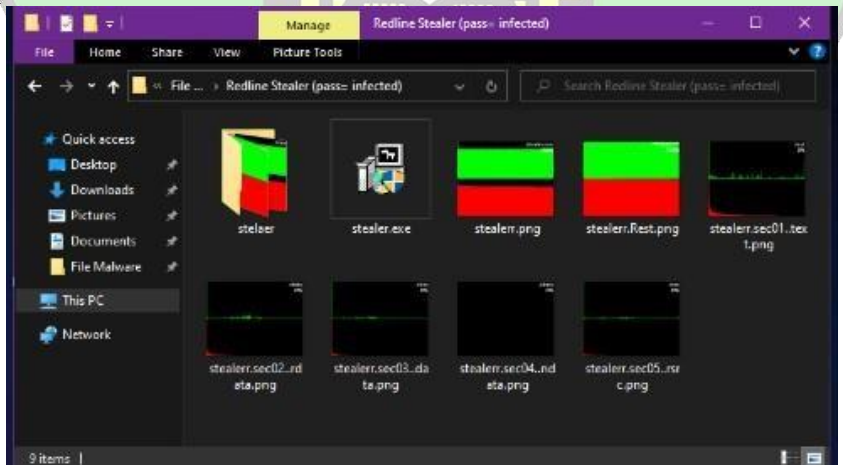


Gambar L. 12 Proses penyimpanan *data*



Gambar L. 13 Proses penyimpanan data

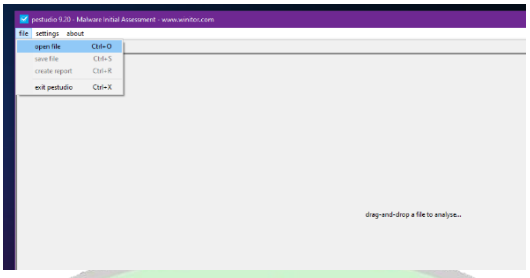
3. Berikut ini adalah hasil analisis dari *tool byte histogram* yang telah tersimpan



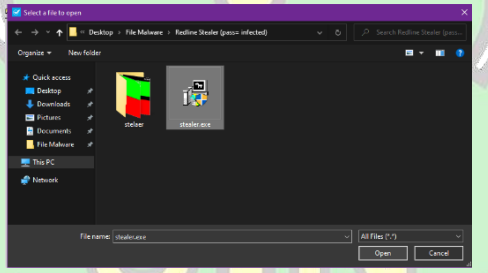
Gambar L. 14 hasil analisis *byte* histogram

Lampiran 6 Analisis *statis tool pestudio*

1. Selanjutnya dilakukan analisis dengan *tool* pestudio, “klik 2” kali pada icon pestudio. “Klik *file*+pilih sampel malware+klik open”.



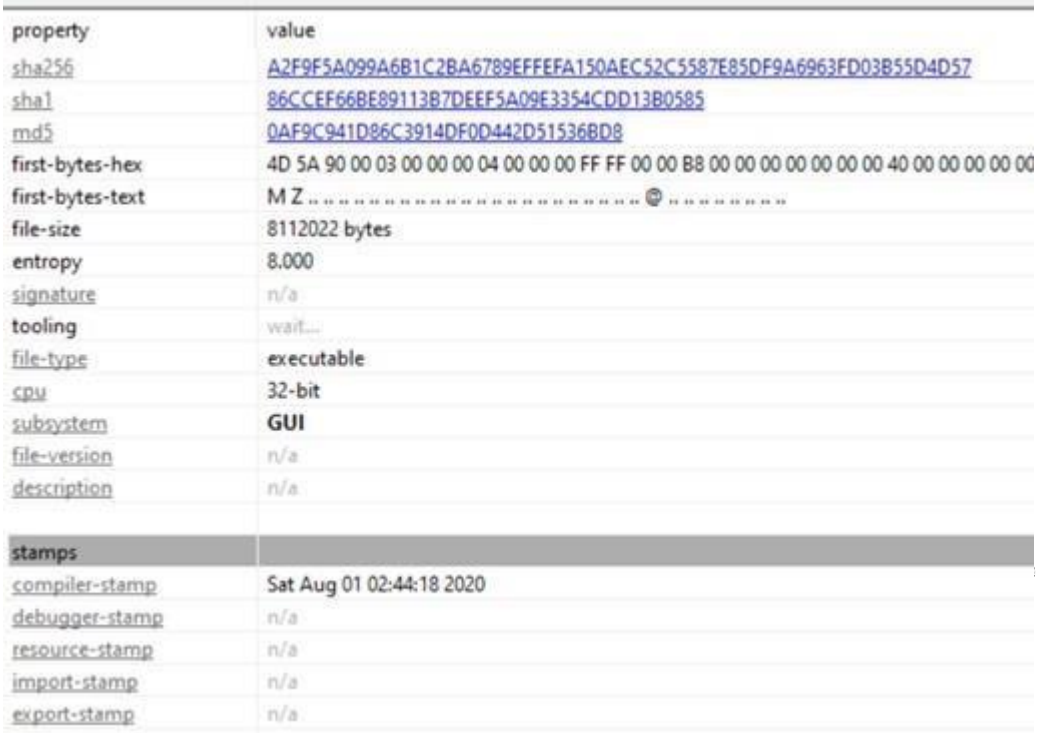
Gambar L. 15 Analisis *tool pestudio*



Gambar L. 16 *Import* sampel

Setelah dilakukan *import* sampel pestudio akan memperoses sampel malware dan informasi yang diperoleh dengan *tool* ini berupa nilai *Hash*, compiler-stamp, *import*, string, section dan informasi informasi lainnya.

2. Gambar di bawah ini menunjukkan informasi umum yang diperoleh dengan menggunakan *tool* pestudio seperti nilai *Hash*, md5, sha256, compiler-stamp, cpu.

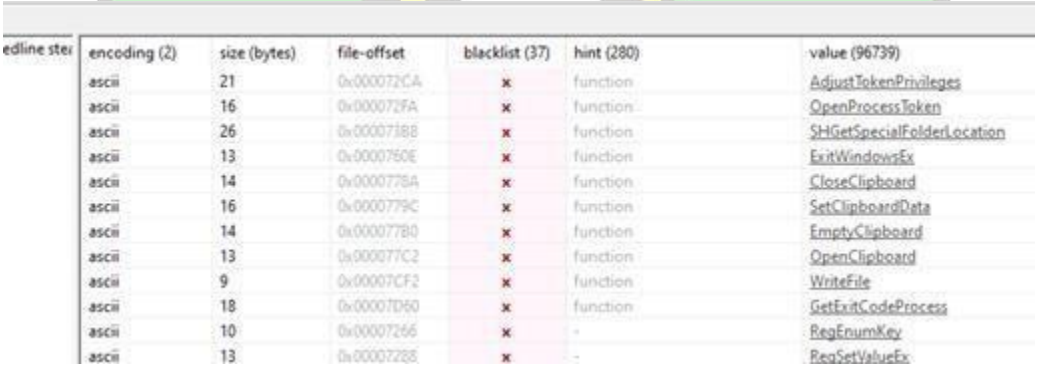


property	value
sha256	A2F9F5A099A6B1C2BA6789EFFEFA150AEC52C5587E85DF9A6963FD03B55D4D57
sha1	86CCEF66BE89113B7DEEF5A09E3354CDD13B0585
md5	0AF9C941D86C3914DF0D442D51536BD8
first-bytes-hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00
first-bytes-text	M Z ..... @ .....
file-size	8112022 bytes
entropy	8.000
signature	n/a
tooling	wait...
file-type	executable
cpu	32-bit
subsystem	GUI
file-version	n/a
description	n/a

stamps	
compiler-stamp	Sat Aug 01 02:44:18 2020
debugger-stamp	n/a
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a

Gambar L. 17 Ringkasan struktur sampel *malware*

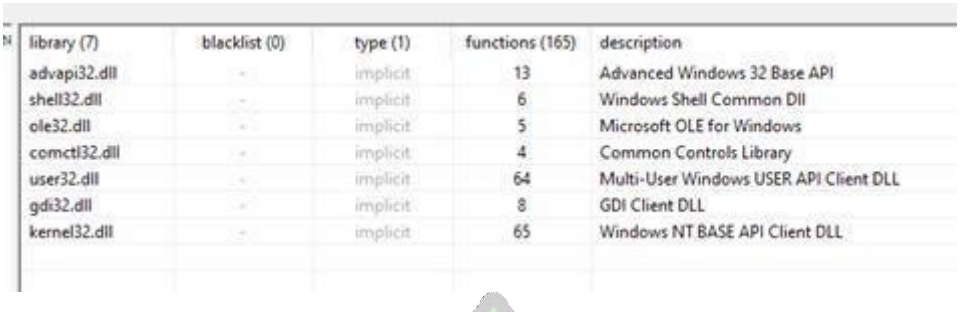
3. Gambar L.17 adalah informasi string yang digunakan malware.



edline ster	encoding (2)	size (bytes)	file-offset	blacklist (37)	hint (280)	value (96739)
	ascii	21	0x000072CA	x	function	AdjustTokenPrivileges
	ascii	16	0x000072FA	x	function	OpenProcessToken
	ascii	26	0x00007388	x	function	SHGetSpecialFolderLocation
	ascii	13	0x0000760E	x	function	ExitWindowsEx
	ascii	14	0x0000778A	x	function	CloseClipboard
	ascii	16	0x0000779C	x	function	SetClipboardData
	ascii	14	0x000077B0	x	function	EmptyClipboard
	ascii	13	0x000077C2	x	function	OpenClipboard
	ascii	9	0x000077CF2	x	function	WriteFile
	ascii	18	0x00007D60	x	function	GetExitCodeProcess
	ascii	10	0x00007266	x	-	RegEnumKey
	ascii	13	0x00007288	x	-	RegSetValueEx

Gambar L. 18 *string* yang digunakan

4. Gambar L.18 adalah informasi library yang digunakan malware



library (7)	blacklist (0)	type (1)	functions (165)	description
advapi32.dll	-	implicit	13	Advanced Windows 32 Base API
shell32.dll	-	implicit	6	Windows Shell Common Dll
ole32.dll	-	implicit	5	Microsoft OLE for Windows
comctl32.dll	-	implicit	4	Common Controls Library
user32.dll	-	implicit	64	Multi-User Windows USER API Client DLL
gdi32.dll	-	implicit	8	GDI Client DLL
kernel32.dll	-	implicit	65	Windows NT BASE API Client DLL

Gambar L. 19 library yang digunakan

5. Gambar L. 19 adalah informasi section yang digunakan malware

functions (165)	blacklist (35)	type (1)	ordinal (1)	library (7)
RegEnumKeyW	×	implicit	-	advapi32.dll
RegSetValueExW	×	implicit	-	advapi32.dll
RegDeleteValueW	×	implicit	-	advapi32.dll
RegDeleteKeyW	×	implicit	-	advapi32.dll
AdjustTokenPrivileges	×	implicit	-	advapi32.dll
LookupPrivilegeValueW	×	implicit	-	advapi32.dll
OpenProcessToken	×	implicit	-	advapi32.dll
SetFileSecurityW	×	implicit	-	advapi32.dll
SHGetSpecialFolderLocation	×	implicit	-	shell32.dll
SHFileOperationW	×	implicit	-	shell32.dll
SHBrowseForFolderW	×	implicit	-	shell32.dll
SHGetPathFromIDListW	×	implicit	-	shell32.dll
ShellExecuteExW	×	implicit	-	shell32.dll

Gambar L. 20 section yang digunakan

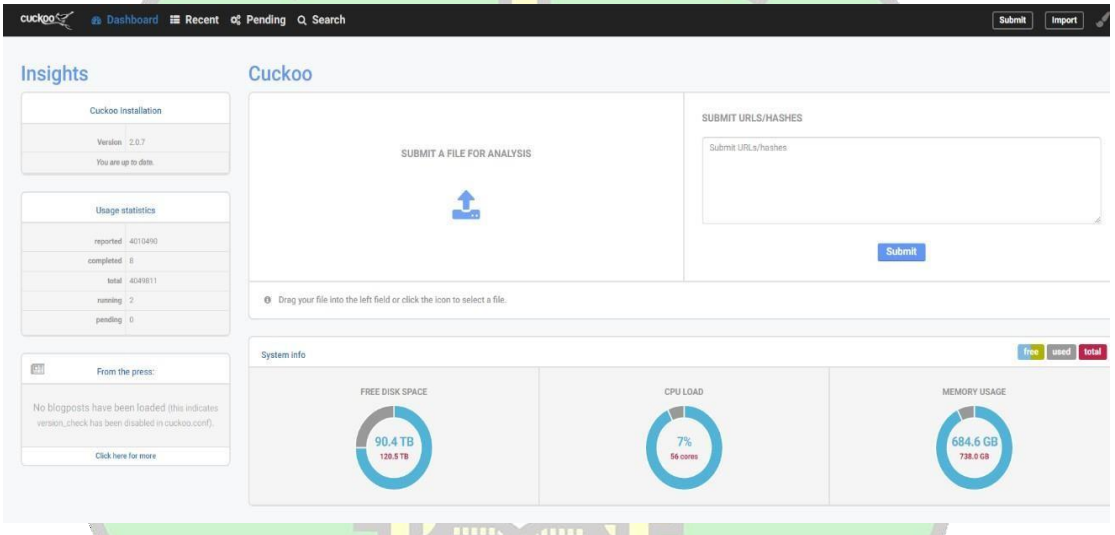
Setelah melakukan analisis statis dengan *tool byte histogram* dan *pestudio*, informasi yang diperolah dikumpulkan untuk mengetahui informasi *malware* yang telah dianalisis serta berguna untuk mendapatkan proses injeksi *malware* dan informasi lainnya.

Lampiran 7 Analisis dinamis *tool Cuckoo Sandbox*

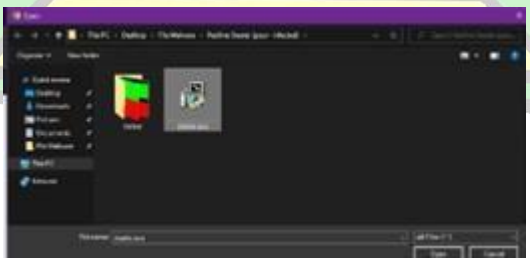
Tahapan analisis dinamis dilakukan pada *website cuckoo* yang dapat diakses melalui link berikut https://cuckoo.cert.ee/ dan tahapan analisisnya sebagai berikut.

1. Masukkan *file malware*

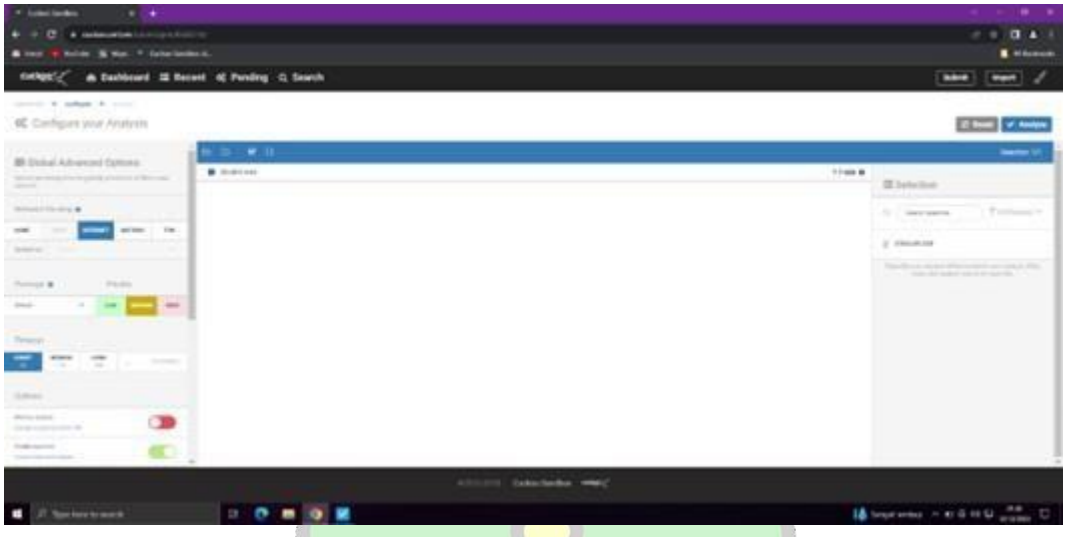
Langkah pertama adalah memasukkan sampel *malware* yang telah disiapkan sebelumnya dengan **"klik *submit file*+pilih sampel *malware*+*Open*"** setalah itu *cuckoo* akan memproses penyiapan *import malware* dan setelah selesai **"Klik analisis+Tunggu Proses Analisis Berjalan+klik *Export*"** proses ini dapat dilihat pada Gambar L 21-23.



Gambar L. 21 Tampilan *Home Website Cuckoo Sandbox*



Gambar L. 22 Masukan sampel



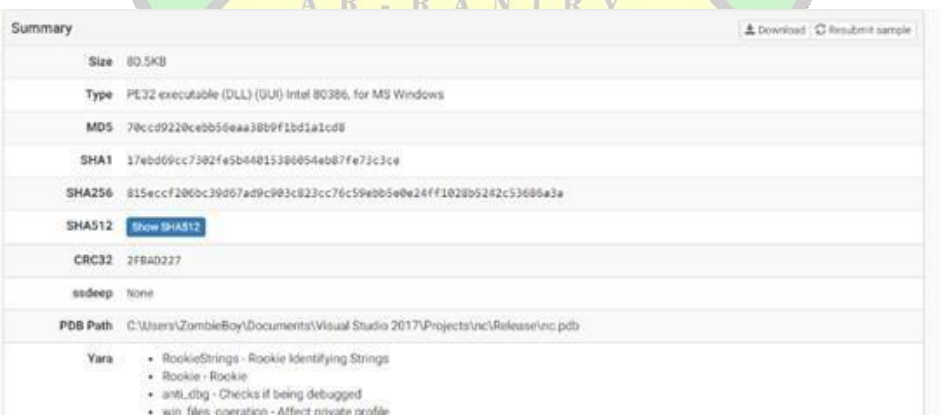
Gambar L. 23 sampel siap untuk dianalisis

Gambar L. 3.24 Proses Analisis berjalan

2. Hasil Analisis dinamis

- summary

  Pada gambar 3.15 manampilkan informasi umum dari *malware,* seperti ukuran *file*, *MD5*, *SHA1*, dan juga path dari *file malware* tersebut. Hasil dapatdilihat pada Gambar L. 24

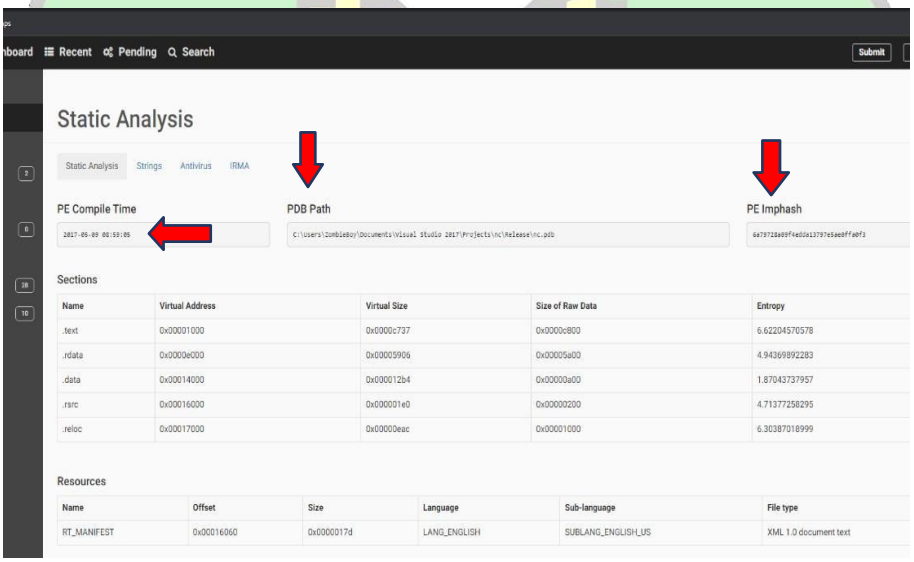


Gambar L. 25 Hasil *Summary* Analisis *Malware Cuckoo Sandbox*

- Berikut adalah informasi siganture terdapat dalam *malware*, *cuckoo* memberikan 3 tingkatan kredintial malware 1). Low (dengan tanda warna biru), 2). *Medium* (dengan tanda warna kuning), dan 3). *Hight* (dengan tanda warna merah) informasi ini dapat diliat pada Gambar L. 25.

Gambar L. 26 Hasil Analisis Perilaku *Malware*

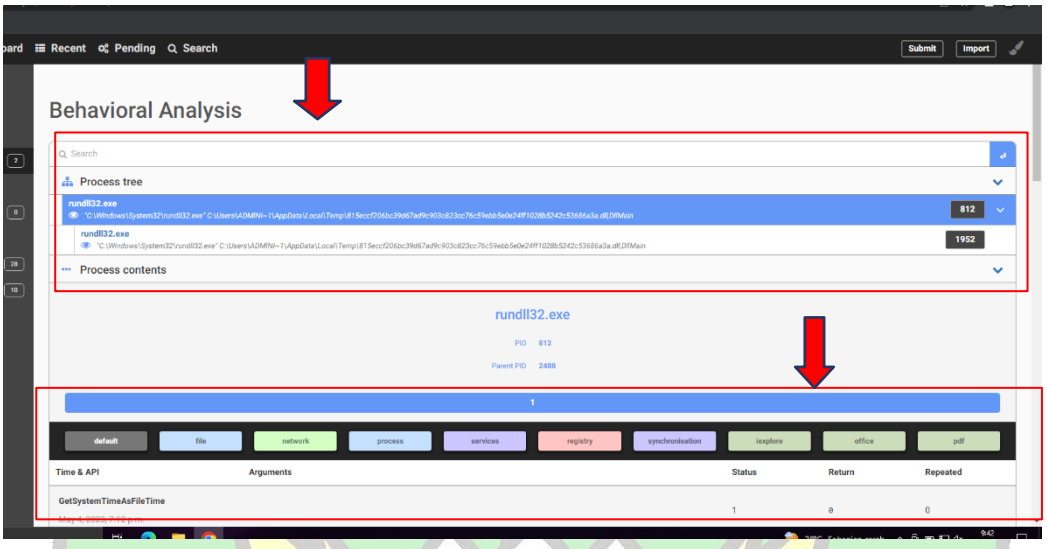
- *Tool cuckoo sandbox* juga memberikan informasi analisis statis, informasi yang diperoleh seperti informasi *malware* dibuat, lokasi path *malware*, *impHash* (*import Hash*), jenis *file*, dan informasi lainnya yang dpat dilihat pada Gambar L. 26.



Gambar L. 27 Hasil Analisis statis *cuckoo*

- Behavioral analisis

  Pada bagian ini *cuckoo* memberikan informasi *Behavioral* analisis *malware*, ini berguna untuk mengetahui proses *tree malware* dan proses ini dibagi 9 bagian untuk memisahkan proses yang dilakukan. Hasil *behavioral* analisis dapat dilihat pada Gambar L. 30.



Gambar L. 3.28 Hasil Analisis *Behavioral*

Setelah selesai melakukan analisis dinamis informasi yang telah diperoleh dikumpulkan untuk mengetahui perilaku *malware* dalam melakukan injeksi pada perangkat. Demikianlah proses analisis *malware* dinamis dengan menggunakan *tool cuckoo sandbox* untuk mengetahui *signature* dari *malware*.