ANALISIS RANSOMWARE MENGGUNAKAN METODE ANALISIS STATIS, DINAMIS, DAN *REVERSE ENGINEERING*.

SKRIPSI

LAYLA ROSYIDAH

2107421023

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK NEGERI JAKARTA
2025

ANALISIS RANSOMWARE MENGGUNAKAN METODE ANALISIS STATIS, DINAMIS, DAN *REVERSE ENGINEERING*

SKRIPSI

Dibuat untuk Melengkapi Syarat-Syarat yang Diperlukan untuk Memperoleh Diploma Empat Politeknik

LAYLA ROSYIDAH

2107421023

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK NEGERI JAKARTA
2025

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

Nama	:	Layla Rosyidah
NIM	:	2107421023
Jurusan/Program Studi	:	Teknik Informatika dan Komputer/Teknik Multimedia dan Jaringan
Judul Skripsi	:	Analisis Ransomware Menggunakan Metode Analisis Statis, Dinamis, dan *Reverse Engineering*.

Menyatakan dengan sebenarnya bahwa skripsiini benar-benar merupakan hasil karya saya sendiri, bebas dari peniruan terhadap karya dari orang lain. Kutipan pendapat dan tulisan orang lain ditunjuk sesuai dengan cara-cara penulisan karya ilmiah yang berlaku.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa dalam skripsi ini terkandung ciri-ciri plagiat dan bentuk-bentuk peniruan lain yang dianggap melanggar peraturan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Depok, 4 Juli 2025

Yang Membuat Pernyataan

Layla Rosyidah

NIM. 2107421023

LEMBAR PENGESAHAN

Skripsi diajukan oleh :

Nama	:	Layla Rosyidah
NIM	:	2107421023
Jurusan/Program Studi	:	Teknik Informatika dan Komputer/Teknik Multimedia dan Jaringan
Judul Skripsi	:	Analisis Ransomware Menggunakan Metode Analisis Statis, Dinamis, dan *Reverse Engineering*.

Telah diuji oleh tim penguji dalam Sidang Skripsi pada hari Rabu, Tanggal 9, Bulan Juli, Tahun 2025 dan dinyatakan **LULUS.**

Disahkan Oleh

Pembimbing I	Iik Muhamad Malik Matin, S.Kom., M.T.	( )
Penguji I	Defiana Arnaldy, S.Tp., M.Si.	( )
Penguji II	Asep Kurniawan, S.Pd., M.Kom.	( )
Penguji III	Fachroni Arbi Murad, S.Kom., M.Kom.	( )

Mengetahui:
Jurusan Teknik Informatika dan Komputer
Ketua

Anita Hidayati, S.Kom., M.Kom.
NIP. 197908032003122003

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Politeknik Negeri Jakarta, saya yang bertanda tangan dibawah ini :

Nama	:	Layla Rosyidah
NIM	:	2107421023
Jurusan/Program Studi	:	Teknik Informatika dan Komputer/Teknik Multimedia dan Jaringan

Demi pengembangan ilmu pengetahuan, menyutujui untuk memberikan kepada Politeknik Negeri Jakarta Hak Bebas Royalti Non-Eksklusif atas karya ilmiah saya yang berjudul :

Analisis Ransomware Menggunakan Metode Analisis Statis, Dinamis, dan Reverse Engineering.

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksklusif ini Politeknik Negeri Jakarta Berhak menyimpan, mengalihmediakan/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta..

Demikian pernyatan ini saya buat dengan sebenarnya.

Depok, 4 Juli 2025

Yang Menyatakan

Layla Rosyidah

NIM. 2107421023

KATA PENGANTAR

Puji dan syukur penulis ucapkan kepada Allah SWT atas berkat dan rahmat-Nya, sehingga penulis mampu menulis dan menyelesaikan skripsi yag berjudul "Analisis Ransomware Menggunakan Metode Analisis Statis, Dinamis, dan *Reverse Engineering*.". Skripsi ini ditulis dalam rangka memenuhi syarat kelulusan di Politeknik Negeri Jakarta.

Dalam proses penulisan skripsi ini, banyak pihak yang telah memberi bantuan, bimbingan, serta dukungan. Oleh karena itu, penulis ingin menyampaikan rasa terima kasih sebesar-besarnya kepada:

1. Jurusan Teknik Informatika dan Komputer, yang telah memberikan seluruh fasilitas terbaik untuk tumbuh dan berkembang.
2. Bapak Iik Muhammad Malik Matin, S.Kom., M.T. sebagai dosen pembimbing yang telah meluangkan waktu untuk senantiasa membimbing, memberi masukan, kritik, saran, dan pengarahan kepada Penulis dalam proses penulisan skripsi ini.
3. Bapak Suratno dan Alm. Ibu Yayah Mujenah, selaku orang tua Penulis, kepada beliau berdualah skripsi ini dipersembahkan. Terima kasih atas segala kasih sayang yang diberikan selama membesarkan Penulis, segala dukungan dan doa yang dipanjatkan dan dukungan untuk kelancaran proses penulisan skripsi ini.
4. Hary Yuliansyah, Mochamad Budi Satria, Puan Melati, Denada Anggia Pradana, dan Shakira Annisa Luthfia, sebagai abang dan kakak yang selalu mendukung, mendengarkan, memberikan ruang untuk Penulis tumbuh menjadi sosok yang akan terus berjuang keras dan menuntaskan harapan terbesar untuk mereka.
5. Hary Alfajri, sebagai sosok yang selalu menyemangati, mendengarkan seluruh keluh kesah, dan menemani Penulis untuk selalu berproses. Terima kasih untuk segala waktu, doa baik, tenaga, tawa, dan kehadiran yang tidak pernah absen disaat Penulis lelah maupun ingin menyerah. Kebersamaan

dan bantuanmu menjadi salah satu hal yang sangat berarti dalam proses menyelasaikan skripsi ini.

6. Yazmin Nur'Aini dan Nurul Aulia Dewi, yang selalu memberikan semangat, energi positif, dan kerap menanyakan kondisi penulis apapun kondisinya.
7. Seluruh sahabat Konjep, Hary Alfajri, Yazmin Nur'Aini, Nurul Aulia Dewi, Sandika Arga Pamungkas, Puguh Muammar Bramantyo, Yusuf Rafif Karback, serta Berlianna Upik N dan M. Daffa Rasyid, yang menjadi tempat berbagi cerita, semangat, dan berjuang bersama hingga akhir. Terima kasih sudah menjadi cerita terbaik yang penulis dapatkan di masa perkuliahan.
8. Teman-teman TMJ 21 yang turut memberikan dukungan selama penulisan skripsi ini.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna. Oleh karena itu, segala bentuk saran dan kritik akan sangat diterima untuk menjadi evaluasi di kemudian hari. Semoga karya ini dapat memberikan manfaat dan dampak positif bagi pembaca dan pihak lain yang membutuhkan.

Depok, 4 Juli 2025

Layla Rosyidah

Analisis Ransomware Menggunakan Metode Analisis Statis, Dinamis, dan Reverse Engineering.

Abstrak

Ransomware merupakan salah satu jenis ancaman siber yang berkembang pesat dan semakin kompleks dari waktu ke waktu. Penelitian ini dilatarbelakangi oleh maraknya munculnya varian ransomware baru dan terbatasnya referensi yang membahas varian tersebut secara teknis mendalam. Selain itu, tidak semua penelitian terdahulu menggunakan kombinasi tiga pendekatan analisis sekaligus, yaitu analisis statis, dinamis, dan *reverse engineering*, untuk memahami struktur dan perilaku ransomware secara menyeluruh. Penelitian ini dilakukan dengan menganalisis tujuh sampel ransomware utama, yaitu LockBit, Akira, BlackCat, WhisperGate, HellCat, Morpheus dan Prince. Analisis statis menggunakan PEStudio dan teknik ekstraksi string untuk melihat struktur internal file, seperti entropi, struktur PE, dan daftar fungsi API. Analisis dinamis dilakukan menggunakan platform Any.Run untuk mengamati perilaku ransomware saat dijalankan, termasuk aktivitas proses, perubahan file dan registry, serta koneksi jaringan. *Reverse engineering* dilakukan menggunakan Ghidra untuk membedah fungsi internal, mendeteksi teknik *obfuscation*, proses dan jenis algoritma enkripsi, dan pemanggilan API secara dinamis. Hasil penelitian menunjukkan bahwa masing-masing ransomware memiliki karakteristik dan teknik yang berbeda. LockBit dan BlackCat menunjukkan struktur kompleks dan teknik evasi tingkat lanjut, sedangkan HellCat memperlihatkan perilaku hybrid sebagai stealer dan ransomware. Morpheus Stealer digunakan sebagai pembanding untuk memperjelas fitur pencurian data pada HellCat, dan sampel Prince dianalisis dengan hasil menunjukkan teknik enkripsi dan evasi tingkat lanjut. Pendekatan gabungan dari ketiga metode analisis terbukti mampu memberikan gambaran yang lebih menyeluruh terhadap teknik, struktur, dan perilaku ransomware modern.

Kata kunci: ransomware, analisis statis, analisis dinamis, *reverse engineering*, PEStudio, Ghidra, Any.Run.

DAFTAR ISI

DAFTAR GAMBAR

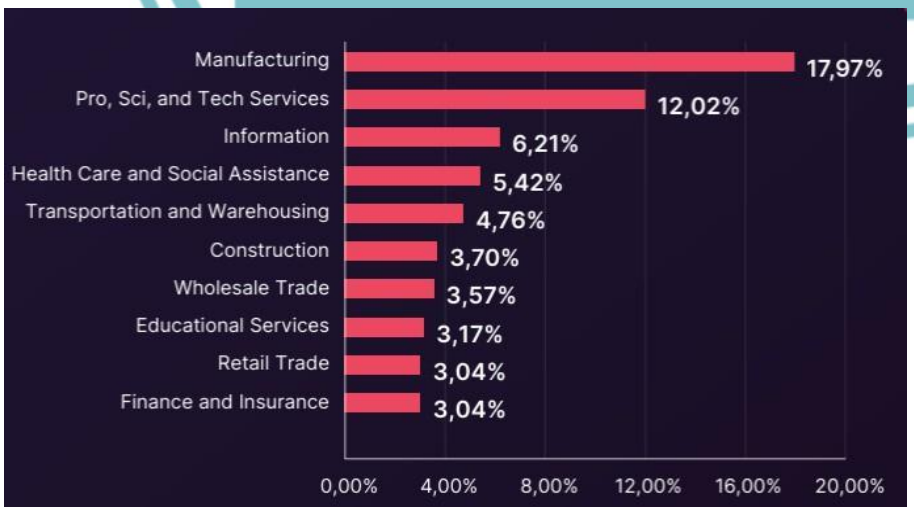
DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam era digital yang semakin maju, teknologi informasi telah menjadi bagian penting dari kehidupan manusia. Penggunaan perangkat komputer dan internet telah menciptakan berbagai kemudahan dalam aktivitas sehari- hari, seperti komunikasi, transaksi keuangan, hingga pengelolaan data. Namun, kemajuan teknologi ini juga diiringi oleh ancaman keamanan siber yang semakin kompleks, salah satunya adalah malware. Malware memiliki karakteristik tertentu, sehingga diklasifikasikan menjadi beberapa jenis, diantaranya *virus, worm, spyware, adware, keylogger, Trojan House, rootkit,* hingga ransomware. (Mylonas dan Gritzalis, 2012).

Diantara beberapa jenis malware, ransomware menjadi jenis malware yang menjadi ancaman paling merugikan. Ransomware bekerja dengan mengenkripsi data korban dan meminta tebusan untuk pemulihannya. Teknik yang digunakan semakin canggih, seperti enkripsi tingkat lanjut, *anti-sandbox*, dan *anti-debugging*. Menurut laporan Cyberint kuartal II tahun 2024, terdapat 1.227 kasus ransomware secara global, meningkat dari kuartal sebelumnya. Indonesia sendiri mencatat 32.803 insiden yang berhasil diblokir, serta menjadi target utama dalam 24 dari 130 serangan global.Seperti yang tertera pada gambar 1.1 mengenai infografis dibawah ini.



Gambar 1. 1 Infografis Sektor Terdampak

Sumber: SOCRadar-Indonesia-Threat-Landscape-Report-2024

Dalam menghadapi ancaman tersebut, analisis malware menjadi penting untuk memahami perilaku dan dampaknya. Pendekatan yang digunakan meliputi analisis statis (mengkaji struktur tanpa eksekusi), analisis dinamis (menjalankan malware dalam lingkungan terisolasi), serta reverse engineering (membongkar kode internal dan teknik tersembunyi). Kombinasi ketiga metode ini mampu mengungkap teknik enkripsi, evasi, dan komunikasi dengan *server Command and Control* (C2) secara lebih dalam.

Berbagai penelitian telah dilakukan untuk memahami karakteristik dan perilaku malware. Nadira (2024) menganalisis malware WannaCry dan beberapa trojan menggunakan pendekatan statis dan dinamis melalui Any.Run dan Flare VM. Namun, penelitian ini belum menyentuh aspek reverse engineering secara mendalam. Cahyadi (2024) menggunakan pendekatan hybrid dan reverse engineering terhadap malware Zeus dengan tools seperti PEStudio, Cutter, dan YARA, namun terbatas pada satu sampel dan tidak membahas teknik enkripsi atau evasi modern. Noor (2024) mengkaji malware Android dengan analisis statis dan reverse engineering untuk mendeteksi pola pencurian finansial, namun tidak mencakup ransomware atau analisis dinamis. David (2021) meneliti backdoor Beast dan Slackbot menggunakan pendekatan statis dan dinamis serta implementasi aplikasi penghapus malware, namun belum mengeksplorasi teknik lanjutan seperti API dynamic resolution atau manipulasi struktur PE. Terakhir, Nicho et al. (2023) menganalisis ransomware WhisperGate dan BlackCat dengan kombinasi analisis statis, dinamis, dan memory analysis, namun cakupannya masih terbatas pada dua varian dan belum menyoroti teknik dari ransomware lain seperti LockBit atau Akira

Berbagai penelitian sebelumnya telah dilakukan, namun sebagian besar hanya berfokus pada satu jenis ransomware, tidak menyertakan analisis menyeluruh, atau belum mengeksplorasi teknik canggih seperti obfuscation dan API dynamic resolution. Oleh karena itu, penelitian ini dilakukan untuk mengisi kekosongan tersebut dengan menganalisis tujuh varian ransomware, yaitu Akira, LockBit,

BlackCat, WhisperGate, HellCat, Morpheus, dan Prince, dengan menggunakan pendekatan gabungan yang bertujuan mengungkap karakteristik teknis, pola enkripsi, serta metode evasi yang digunakan oleh ransomware modern.

1.2 Rumusan Masalah

Berdasarkan uraian dari latar belakang, permasalahan yang dapat dirumuskan adalah mengenai bagaimana melakukan analisis statis, dinamis, dan *reverse engineering* terhadap ransomware Akira, LockBit, BlackCat, WhisperGate, HellCat, Morpheus, dan Prince

1.3 Batasan Masalah

Terdapat beberapa batasan masalah yang disusun guna memfokuskan ruang lingkup penelitian, adalah sebagai berikut:

a. Sampel-sampel yang digunakan adalah Ransomware Akira, LockBit BlackCat, WhisperGate, HellCat, Morpheus, dan Prince, dengan nilai hash SHA-256:
   - Akira : 5DE1061457E759E022FBC9BB02E8726A49D3ED9663FC8A77D83462C69C96AEA8
   - LockBit : F4FB0F2AE098850F2A8FFB771AE4C6C8AAA81144FE53228A2C01DF2D34307053
   - BlackCat : E160B6348F6FBDC444125B865DBD9406D99D4A8C8334C8E682EE4429F813293
   - WhisperGate : 82EF68548D2E2E4B2C41A1C92D06B30B0A433B17B859590989C89D4CE9162033
   - HellCat : B8E71845CC8CCD668A3436D1952A6C57649974BB8399E599DC33AFC4C0843BE7
   - Morpheus : 4B2EDADC8F90E9FCC976F02A9EDA1640CD92C07718C0271842FBD4CA7E2906E2
   - Prince : B7A6EB4DC4C644FE3611B0F4E69202455D61CF426726343B2FFB182C3DEB6CB8

b. Penggunaan PeStudio dan *command* Strings pada terminal untuk melakukan teknik analisis statis.

c. Penggunaan Any.Run sebagai sandbox untuk melakukan teknik analisis dinamis.
d. Penggunaan Ghidra dalam teknik *reverse engineering* untuk meng-decompile dan meng-disassembly source code malware.

1.4 Tujuan dan Manfaat

Adapun tujuan dan manfaat dari penelitian Analisis Ransomware Akira, LockBit BlackCat, WhisperGate, HellCat, Morpheus, dan Prince Menggunakan Metode Analisis Statis, Analisis Dinamis dan *Reverse Engineering,* adalah sebagai berikut:

1.4.1 Tujuan

Tujuan dari penelitian ini adalah untuk melakukan analisis statis, dinamis, dan *reverse engineering* terhadap ransomware Akira, LockBit, BlackCat, WhisperGate, HellCat, Morpheus, dan Prince.

1.4.2 Manfaat

Manfaat yang diharapkan dari penelitian ini adalah sebagai berikut:

a. Meningkatkan kemampuan deteksi awal serangan Ransomware Akira, LockBit BlackCat, WhisperGate, HellCat, Morpheus, dan Prince
b. Memberikan pemahaman lebih mendalam mengenai cara kerja dan karakteristik Ransomware Akira, LockBit BlackCat, WhisperGate, HellCat, Morpheus, dan Prince.
c. Menyediakan referensi bagi penelitian selanjutnya untuk mengidentifikasi ransomware serupa.

1.5 Sistematika Penulisan

Berikut adalah sistematika penulisan yang digunakan dalam membuat laporan penelitian ini:

a. BAB I PENDAHULUAN

Pada bab ini berisi latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, serta sistematika penulisan.

b. BAB II TINJAUAN PUSTAKA

Pada bab ini berisi landasan teori yang digunakan dan menguraikan penelitian-penelitian terkait.

c. BAB III METODE PENELITIAN

Pada bab ini berisi penjelasan mengenai rancangan penelitian, tahapan penelitian, dan skenario analisis yang digunakan.

d. BAB IV HASIL DAN PEMBAHASAN

Pada bab ini berisi analisis kebutuhan sistem, perancangan dan implementasi sistem, serta analisis aturan dan hasil yang diperoleh.

e. BAB V PENUTUP

Pada bab ini berisi kesimpulan dari hasil analisis serta saran-saran untuk penelitian selanjutnya.

BAB II
TINJAUAN PUSTAKA

2.1 Keamanan Informasi

Keamanan informasi merupakan perlindungan terhadap informasi dan elemen-elemen yang mencakup sistem dan perangkat keras yanng digunakan untuk menyimpan, memproses, dan mentransmisikan informasi. Keamanan dicapai melalui penerapan kebijakan, pendidikan, pelatihan, kesadaran, dan teknologi. Model keamanan informasi seringkali didasarkan pada segitiga C.I.A., yang menekankan prinsip pentingnya menjaga kerahasiaan (*Confidentiality*), integritas (*Integrity*), dan ketersediaan (*Availability*) informasi. Ketiga elemen ini memberikan nilai pada informasi bagi sebuah organisasi. (Whitman, 2011)

2.2 Keamanan Siber

Serangan siber merupakan fenomena yang terus berkembang seiring dengan kemajuan teknologi. Serangan ini berfokus pada tiga aspek utama, yaitu manipulasi data, pencurian data, dan penghilangan akses ke suatu sistem, sehingga pengguna lain tidak dapat mengaksesnya. Secara umum, serangan siber dapat diklasifikasikan menjadi dua kategori utama, yaitu serangan yang ditujukan terhadap sistem, perangkat lunak, atau perangkat keras, dan serangan yang menyasar individu atau pengguna dari suatu sistem. Kedua jenis serangan ini berkaitan dengan tiga konsep utama dalam keamanan informasi dan teknologi, yaitu kerahasiaan (*confidentiality*), integritas (*integrity*), dan ketersediaan (*availability*) data maupun sistem digital. (Wahib et al., 2022)

2.3 Malware

Perangkat lunak berbahaya, atau yang dikenal dengan malware, adalah program yang terinstal di sistem komputer tanpa diketahui oleh pengguna atau pemiliknya. Sesuai namanya, malware dapat melakukan tindakan berbahaya seperti mencuri data sensitif, merusak sistem, memperoleh hak akses komputer, dan menjalankan program-program yang ada di komputer tersebut. Setiap perangkat lunak yang dapat menyebabkan kerugian bagi pengguna, komputer, atau jaringan dapat dianggap sebagai malware. (Puji Rahayu, 2021)

Malware dapat diklasifikasikan dalam beberapa kategori. Klasifikasi malware biasanya berdasarkan cara penyebarannya dan respons malware terhadap sistem yang terinfeksi. Berikut adalah tiga kategori utama malware:

1. Ancaman Menular (*Contagious Threats*)

   Malware yang termasuk dalam kategori ini antara lain:

   a. Virus: Malware yang mengambil alih kendali komputer yang terinfeksi secara tidak sah dan menyebabkan kerusakan tanpa diketahui pengguna.
   b. Worm: Malware yang dapat beroperasi secara independen tanpa perlu terhubung ke sistem lain. Kerusakan yang ditimbulkan oleh worm antara lain menghabiskan memori sistem dan menurunkan kinerja jaringan.

2. Ancaman Tersembunyi (*Masked Threats*)

   Malware yang tergolong dalam kategori ini meliputi:

   a. *Trojan*: Malware berbahaya yang tersembunyi dan berpura-pura menjadi program yang sah untuk mengakses sistem secara ilegal. Kerusakan yang ditimbulkan oleh Trojan termasuk pencurian kata sandi, uang digital, perubahan atau penghapusan file, serta pemantauan aktivitas pengguna.
   b. *Backdoor*: Malware yang dapat mengelabui kontrol keamanan dan memberikan akses ilegal kepada penyerang. Dampak yang dapat timbul antara lain pengubahan dan penghapusan file sistem serta pemantauan aktivitas.
   c. *Adware*: Malware yang mengumpulkan informasi tentang kebiasaan browsing pengguna untuk membantu pengiklan menargetkan iklan secara lebih tepat. Hal yang dapat ditimbulkan oleh *adware* termasuk *clickjacking*, *phishing*, dan aktivitas berbahaya lainnya melalui browser.
   d. *Rootkit*: Teknik penyamaran yang dirancang untuk tujuan jahat, dengan kemampuan untuk mencuri kata sandi atau menginstal *keylogger*.

3. Ancaman Finansial (*Finansial Threats*)

   Malware dalam kategori ini mencakup:

a. Ransomware: Malware yang dirancang untuk mengunci akses ke sistem komputer hingga sejumlah uang dibayar. Kerusakan yang ditimbulkan oleh ransomware termasuk pencurian data, enkripsi data korban, dan pembatasan akses ke sistem.
b. *Spyware*: Malware yang memantau aktivitas pengguna tanpa sepengetahuan mereka dan mengirimkan informasi sensitif kepada penyerang. *Spyware* dapat melakukan penyadapan pada jaringan, sertifikat digital, kunci enkripsi, dan data sensitif lainnya.
c. *Keylogger*: Malware yang secara diam-diam merekam setiap ketikan di keyboard korban, yang memungkingkan penyerang untuk mengambil informasi sensitive seperti nama pengguna, kata sandi, nomor kartu kredit, hingga rincian perbankan online.

2.4 Ransomware

Ransomware merupakan jenis perangkat lunak berbahaya (malware) yang dirancang untuk mengenkripsi data pada sistem komputer atau perangkat lainnya, dan kemudian menuntut pembayaran tebusan kepada korban agar data tersebut dapat dikembalikan atau didekripsi. Umumnya, ransomware menyebar melalui tautan atau lampiran berbahaya dalam email, situs web terinfeksi, atau eksploitasi kerentanan dalam sistem. Dampaknya sangat merugikan karena dapat mengganggu operasional bisnis dan menyebabkan kerugian finansial yang besar. (Hartono, 2023)

2.4.1 Akira

Akira ransomware adalah jenis ransomware yang pertama kali muncul pada Maret 2023 dan beroperasi dengan model Ransomware-as-a-Service (RaaS). Hingga tahun 2024, kelompok ini telah menargetkan lebih dari 250 organisasi di berbagai negara dan memperoleh lebih dari 42 juta dolar dari hasil pemerasan. Akira diduga memiliki keterkaitan dengan kelompok Conti yang telah bubar sebelumnya. Teknik serangan yang digunakan meliputi eksploitasi kerentanan pada VPN, brute force terhadap layanan RDP, hingga penyalahgunaan kredensial yang dicuri. Setelah berhasil masuk ke sistem, pelaku akan melakukan pemetaan

jaringan, mencuri data penting, lalu mengenkripsinya dengan algoritma ChaCha20 dan RSA-4096. (Qualys, 2024)

2.4.2 LockBit

LockBit ransomware adalah perangkat jahat yang dirancang untuk memblokir akses pengguna ke sistem komputer, kemudian menuntut uang tebusan untuk mendekripsi data. Ransomware ini beroperasi otomatis dengan memindai target bernilai tinggi, menyebarkan infeksi, dan mengenkripsi semua sistem yang bisa diakses dalam jaringan organisasi atau perusahaan. LockBit mulai dikenal pada September 2019 sebagai varian ".abcd" dan sekarang termasuk dalam model *Ransomware-as-a-Service* (RaaS), di mana pengembangnya menyewakan alat kepada afiliasi yang melakukan serangan, lalu berbagi hasil tebusan ke bagian afiliasi yang bisa mencapai hingga tiga per empat dari total. Serangan tersebut tidak hanya menyebabkan gangguan operasional besar tetapi juga pelanggaran data dengan ancaman publikasi ilegal jika tebusan tidak dibayar. (Kaspersky)

2.4.3 BlackCat

BlackCat ransomware yang juga dikenal sebagai ALPHV atau Noberusadalah malware canggih yang pertama muncul pada November 2021 dan ditulis dengan bahasa pemrograman Rust. Dikembangkan oleh kelompok penjahat siber berbahasa Rusia dengan kaitan teknis ke grup DarkSide dan BlackMatter, BlackCat beroperasi sebagai *Ransomware-as-a-Service* (RaaS), menyediakan alat kepada afiliasi dengan pembagian hasil hingga 80–90 % dari pembayaran tebusan. Selain mengenkripsi file dengan cepat dan tersembunyi, BlackCat juga menerapkan taktik *double* maupun *triple extortion*, yaitu dengan mencuri data, mengenkripsi sistem, serta mengancam serangan DDoS jika korban tidak memenuhi tuntutan tebusan. Grup ini telah menyerang ratusan organisasi di berbagai industri, dan semakin sulit dideteksi karena malware-nya ditulis dalam Rust, bahasa yang jarang digunakan oleh solusi keamanan tradisional. (Akamai)

2.4.4 WhisperGate

WhisperGate adalah varian malware destruktif yang digunakan dalam serangan siber terhadap organisasi di Ukraina pada awal tahun 2022. Malware ini memiliki perilaku mirip ransomware, tetapi tidak menyediakan mekanisme pemulihan data, sehingga termasuk dalam kategori *pseudo-ransomware*. WhisperGate bekerja dalam beberapa tahap: pertama, ia merusak Master Boot Record (MBR) untuk mencegah sistem melakukan booting; kemudian ia menampilkan pesan tebusan palsu; dan selanjutnya mengunduh komponen tambahan yang mengenkripsi atau menghapus file secara permanen. Tujuan utamanya bukan untuk mendapatkan tebusan, melainkan menghancurkan data dan mengganggu operasional korban secara signifikan. Berdasarkan analisis CISA dan FBI, malware ini menunjukkan karakteristik manipulatif dan dirancang untuk menciptakan kesan serangan ransomware padahal sebenarnya bersifat *wiper*, yakni penghancur data permanen (CISA, 2022)

2.4.5 HellCat

Hellcat adalah kelompok ransomware-as-a-service (RaaS) yang mulai muncul pada akhir 2023 hingga awal 2024 dan telah menjadi ancaman signifikan bagi sektor infrastruktur penting, termasuk pemerintahan, pendidikan, dan energi. Kelompok ini menggunakan berbagai vektor serangan seperti spear phishing, eksploitasi aplikasi publik, dan *zero-day* untuk mendapatkan akses awal. Teknik lanjutan mereka mencakup chaîne PowerShell bertahap, bypass AMSI dan reflektif code loading untuk menjalankan malware di memori tanpa menulis ke disk, serta penggunaan SliverC2 dan binary “living off the land” seperti Netcat untuk pergerakan lateral dan ketahanan. Mereka juga menerapkan strategi double extortion, merampok dan mengenkripsi data korban, kemudian mengancam publikasi data sensitif sebagai tekanan tambahan untuk menuntut tebusan. (Broadcom Inc, 2025)

2.4.6 Morpheus

Morpheus adalah kelompok ransomware baru yang mulai terdeteksi pada Desember 2024 dan dikenal dengan strategi pemerasan berbasis kebocoran data melalui *platform data-leak site* (DLS). Kelompok ini pertama kali muncul dengan mengklaim telah mencuri data dari perusahaan farmasi Australia, Arrotex, dan produsen elektronik Jerman, PUS GmbH. Berbeda dari kelompok ransomware tradisional, Morpheus lebih menekankan pada pemerasan berbasis pencurian data tanpa bukti kuat penggunaan enkripsi terhadap sistem korban. Namun, sejumlah peneliti keamanan menemukan bahwa Morpheus kemungkinan menggunakan payload yang memiliki kemiripan struktural dengan ransomware Hellcat, termasuk metode enkripsi AES-256 dan teknik penghindaran deteksi. Hingga kini, aktivitas kelompok ini menunjukkan pola operasi mirip *Ransomware-as-a-Service* (RaaS), meskipun lebih mengarah pada model pemerasan digital berbasis publikasi data daripada enkripsi menyeluruh sistem. (Cyjax, 2025)

2.4.7 Prince

Prince adalah varian ransomware open-source yang ditulis menggunakan bahasa Go dan tersedia di GitHub, menggunakan kombinasi enkripsi ChaCha20 dan ECIES untuk mengamankan file target secara efektif. Meskipun dijual sebagai alat edukasional, peneliti Proofpoint melaporkan penggunaannya dalam kampanye aktif pada September 2024 yang meniru layanan Royal Mail melalui formulir kontak web, menginfeksi sejumlah organisasi di Inggris dan AS. Ransomware ini mengubah ekstensi file menjadi ".womp", menampilkan instruksi pembayaran, namun analisis menunjukkan tidak ada mekanisme validasi pembayaran ataupun kemampuan dekripsi, menjadikannya alat destruktif ketimbang ransomware tradisional. (Proofpoint, 2024)

2.5 Virtual Box

VirtualBox merupakan sebuah hypervisor tipe 2 yang bersifat open source dan dapat digunakan di berbagai platform dengan performa yang tinggi. Awalnya

dikembangkan oleh Sun Microsystems dengan nama Sun VirtualBox, kini produk ini telah menjadi milik Oracle dan tersedia secara gratis di bawah lisensi perangkat lunak *open source* GNU General Public License (GNU GPL) Versi 2. Dengan kemampuan lintas platformnya, VirtualBox dapat beroperasi pada sistem operasi desktop modern seperti Linux, Windows, Mac, atau Solaris. (Dash, 2013)

2.6 Kali Linux

Kali Linux merupakan distribusi Linux berbasis Debian yang dirancang khusus untuk pengujian keamanan dan forensik digital. Distribusi ini telah dilengkapi dengan lebih dari 300 perangkat lunak keamanan, termasuk di antaranya Nmap untuk pemetaan jaringan, Wireshark untuk analisis lalu lintas jaringan, Metasploit Framework untuk eksploitasi kerentanan, hingga John the Ripper untuk pengujian kekuatan sandi. Keberagaman alat yang tersedia dalam Kali Linux menjadikannya platform yang sangat mendukung dalam kegiatan analisis malware secara mendalam. (Gururaj H et al., 2022)

2.7 MalwareBazaar

MalwareBazaar adalah platform repository sampel malware berbasis komunitas yang dikelola oleh abuse.ch bekerja sama dengan Spamhaus. Platform ini menyediakan akses gratis kepada peneliti dan organisasi keamanan untuk mengunggah dan men-download sampel malware terverifikasi tanpa batasan. Berbeda dengan layanan seperti VirusTotal, MalwareBazaar tidak melakukan scanning AV, hanya menerima malware yang sudah dikonfirmasi, dan mengutamakan distribusi data untuk mendukung analisis ancaman berbasis komunitas. (abuse.ch, 2020)

2.8 Analisis Malware

Analisis malware adalah proses yang dilakukan untuk memahami fungsionalitas, asal, dan dampak potensial dari perangkat lunak berbahaya, seperti *virus, worm, trojan horse, rootkit,* atau *backdoor*. Tujuan utama analisis ini adalah untuk mengidentifikasi dan menanggapi ancaman yang ditimbulkan oleh malware terhadap sistem komputer atau jaringan. (Muhammad et al., 2017)

2.8.1 Analisis Statis

Analisis statis merupakan metode yang melibatkan pemeriksaan file eksekusi tanpa menjalankan instruksi yang terkandung di dalamnya. Metode ini dapat mengidentifikasi apakah sebuah file berpotensi berbahaya, memberikan wawasan mengenai fungsionalitasnya, dan mendeteksi dengan pola atau *signature*. (Mylonas, 2012)

Pendekatan metode ini berfokus pada karakteristik statis dari malware, seperti struktur file, metadata, dan elemen lain yang dapat diidentifikasi tanpa risiko infeksi.

2.8.2 Analisis Dinamis

Analisis dinamis malware adalah metode yang digunakan untuk memahami perilaku dan fungsionalitas perangkat lunak berbahaya dengan mengeksekusinya dalam lingkungan yang terkontrol. Tujuan dari pendekatan ini adalah untuk mengidentifikasi jenis malware, cara penyebarannya, serta komponen yang dapat diinfeksinya. (Mylonas, 2012)

2.9 *Reverse Engineering*

Reverse Engineering adalah proses memahami dan menduplikasi produk atau komponen yang sudah ada tanpa bantuan gambar, dokumentasi, atau model komputer. Tujuan utama dari *reverse engineering* adalah untuk mengidentifikasi komponen dan memahami hubungan antar komponen, membuat representasi dari komponen dalam bentuk lain, serta menghasilkan representasi fisik dari sistem. Dalam konteks rekayasa perangkat lunak, *reverse engineering* biasanya berarti kembali melalui siklus pengembangan untuk memahami implementasi dan cara kerja perangkat lunak, dan merupakan fase yang mendahului re-engineering, di mana hasil dari fase implementasi di-*reverse engineer* ke fase analisis. Proses ini melibatkan pemeriksaan tanpa memodifikasi kode sumber, yang membedakannya dari re-engineering yang melibatkan modifikasi. *Reverse engineering* digunakan dalam berbagai bidang teknologi informasi, seperti analisis malware, analisis jaringan, debugging, dan pengembangan prototipe cepat.(Appelbaum, 2012)

2.10 PeStudio

PeStudio merupakan alat analisis statis file eksekusi (*executable file,* seperti file dengan ekstensi .exe) pada sistem operasi Windows. Alat ini dirancang untuk membantu menganalisis file tanpa menjalankannya, sehingga aman digunakan untuk memeriksa malware atau file yang mencurigakan. Dari hasil analisis akan ditampilkan informasi detail tentang struktur file, seperti *Header PE* yang berisikan entry point, arsitektur file, ukuran, serta checksum, dan informasi *Section* yang berisikan tentang bagian-bagian file. (Siddiq et al., 2020)

2.11 Any.Run

Any.Run adalah alat analisis malware interaktif yang memungkinkan pengguna untuk menganalisis perilaku file berbahaya dalam lingkungan sandbox virtual. Alat ini dirancang untuk memberikan pengalaman analisis yang lebih interaktif dibandingkan dengan layanan lain seperti VirusTotal, yang cenderung tidak interaktif. Any.Run memungkinkan pengguna untuk menjalankan dan berinteraksi dengan file atau program dalam waktu nyata, serta mencatat semua aktivitas yang terjadi selama analisis, termasuk proses yang dimulai, koneksi jaringan, dan perubahan sistem lainnya. (Rizqina, 2022)

2.12 Ghidra

Ghidra merupakan perangkat lunak *open* source untuk menjalankan metode *reverse engineering* yang dikembangkan oleh *National Security Agency* (NSA) Amerika Serikat. Ghidra digunakan untuk menganalisis perangkat lunak, membongkar kode biner, dan memahami cara kerja sebuah program, terutama pada kondisi kode sumber tidak tersedia. Ghidra mendukung berbagai arsitektur seperti x86, ARM, MIPS, dan lainnya. Sehingga sering digunakan untuk menganalisis malware, eksploitasi keamanan, atau pembelajaran mengenai cara kerja sistem. Perangkat lunak ini dilengkapi dengan antarmuka grafis, alat *decompiler,* dan kemampuan untuk bekerja secara kolaboratif dengan tim. (A. P. David, 2021)

2.13 VirusTotal

VirusTotal merupakan layanan berbasis web yang digunakan untuk melakukan pemindaian terhadap sampel malware dengan menggunakan berbagai mesin antivirus secara bersamaan. Dalam analisis statis, hasil pemindaian dapat digunakan sebagai pembanding untuk memverifikasi kesesuaian antara karakteristik yang ditemukan pada sampel seperti string value tertentu dengan nilai identifikasi klasifikasi yang diberikan oleh antivirus. (Kurnia Hatika et al., 2019)

2.14 Enkripsi

Enkripsi merupakan proses kriptografi yang digunakan untuk mengubah informasi asli (*plaintext*) menjadi bentuk yang tidak dapat dibaca (*ciphertext*) oleh pihak yang tidak memiliki otorisasi. Proses ini dilakukan dengan menggunakan algoritma dan kunci enkripsi tertentu untuk menjaga kerahasiaan, integritas, dan keamanan data dalam sistem informasi. (Idliman et al., 2025)

2.15 Entropi

Entropi adalah ukuran untuk menghitung sejauh mana data dalam suatu file bersifat acak. Semakin tinggi nilai entropi (maksimal 8.0 untuk file biner), maka semakin besar kemungkinan file tersebut telah dienkripsi atau dikompresi. Dalam konteks ransomware, file yang telah dikunci akan memiliki nilai entropi mendekati 8.0, karena semua data aslinya telah diubah menjadi pola byte yang acak. Oleh sebab itu, analisis entropi sering digunakan sebagai langkah awal untuk mendeteksi keberadaan ransomware secara statis.(Davies et al., 2022)

2.16 Teknik *Obfuscation*

Obfuscation adalah teknik yang digunakan malware untuk menyamarkan kode sehingga sulit dipahami atau dianalisis, sambil tetap mempertahankan fungsi utamanya. Teknik ini mencakup enkripsi payload, penggunaan polymorphic atau *metamorphic engine*, *dead-code insertion*, *renaming identifier*, serta *control-flow flattening*, yang efektif menghindari deteksi statis atau *signature-based detection*. Penggunaannya seperti payload utama dienkripsi dan didekripsi di runtime, serta

struktur kontrol dan nama variabel diacak agar sulit dianalisis manual atau otomatis. (Asghar et al., 2024)

2.17 Teknik Evasi

Evasi adalah teknik malware untuk menghindari deteksi dinamis maupun statis, khususnya oleh antivirus, sandbox, IDS/IPS, atau sistem analisis otomatis. Malware menggunakan teknik seperti *anti-debugging*, anti-VM, *delay execution*, *dynamic code loading*, *process injection* (*process hollowing*, *reflective DLL injection*), serta *fileless execution* dan *living-off-the-land*. Tujuan dari teknik ini agar malware tidak menampilkan payload saat berada di lingkungan teranalisis atau dieksekusi dalam memori tanpa jejak di disk.(Mehmood et al., 2024)

2.16 Penelitian Terdahulu

Tabel 2. 1 Penelitian Terdahulu

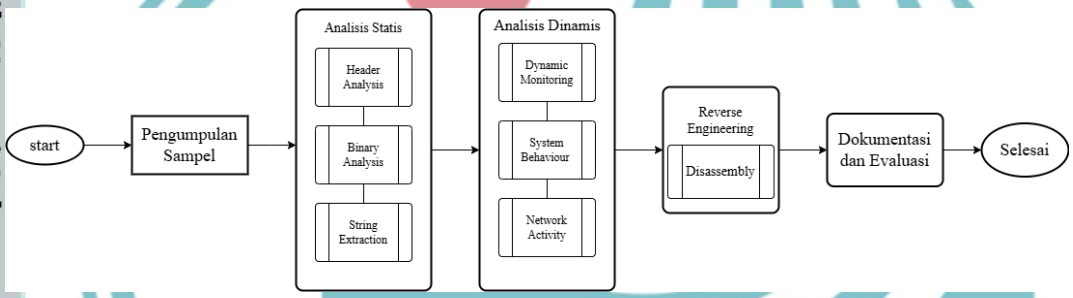
No	Judul Penelitian	Objek	Hasil
1.	Analisis Ransomware Wannacry Dan Trojan Menggunakan Metode Statis Dan Dinamis Berbasis Flare Vm Dan Any Run (Kayla Nadira, 2024)	Ransomware WannaCry dan TrojanRAT	Penelitian hanya menggunakan dua jenis sampel yaitu WannaCry dan TrojanRat. Serta hanya melakukan analisis statis dan dinamis, tidak melakukan *reverse engineering* dan mendalami struktur internal malware
2.	Analisis Statik Dengan Teknik Reverse Engineering Dan Signature Based Detection Pada Perangkat Android (Studi Kasus: Identifikasi Pola Financial Theft	Malware Financial Theft pada file APK	- Tidak mencakup analisis dinamis - Tidak membahas malware jenis ransomware - konteks terbatas hanya pada platform Android

	Dalam File APK) (Nadhilah Noor, 2024)		
3.	Analisis Hybrid Dan Reverse Engineering Untuk Deteksi Malware Zeus Menggunakan Yara Rules (Ivan Harry Cahyadi, 2024)	Malware Zeus	Objek terbatas hanya satu jenis malware (Zeus) dan tidak membandingkan antar ransomware atau menguji variasi teknik enkripsi modern
4.	Analisis Malware Backdoor Menggunakan Metode Analisis Statis Dan Dinamis Pada Malware Analysis Lab Serta Rancang Bangun Aplikasi Untuk Menghapus Malware (Trisya Talia David, 2021)	Malware Backdoor	- Tidak menggunakan pendekatan *reverse engineering* mendalam (seperti decompilation atau analisa fungsi) - Tidak membandingkan malware antar jenis - Fokus hanya pada backdoor, bukan ransomware
5.	*Analyzing Whispergate And Blackcat Malware: Methodology And Threat Perspective.* (Mathew Nicho, 2023)	Ransomware WhisperGate dan BlackCat	- Tidak membandingkan lebih banyak varian ransomware modern - Tidak fokus pada API dynamic resolution, TLS callback, atau struktur PE abnormal

BAB III
METODE PENELITIAN

3.1 Rancangan Penelitian

Pada bab ini menjelaskan lebih rinci mengenai perencanaan dalam penelitian dengan tujuan utama menganalisis malware jenis Ransomware Akira, LockBit BlackCat, WhisperGate, HellCat, Morpheus, dan Prince. Pendekatan yang digunakan melibatkan dua metode utama yaitu analisis statis dan dinamis, serta metode *reverse engineering* untuk proses memahami fungsionalitas, asal, serta dampak potensial yang ditimbulkan dari malware tersebut.



Gambar 3. 1 Rancangan Penelitian

Dalam proses analisis dalam penelitian ini dimulai dengan pengumpulan sampel, di mana file malware diambil untuk dianalisis. Setelah pengumpulan, langkah berikutnya adalah analisis statis oleh PeStudio, yang mencakup beberapa teknik seperti binary analysis, string extraction, dan header analysis. Dalam proses ini binary analysis, berfungsi untuk memeriksa kode biner untuk mengidentifikasi pola dan karakteristik yang menunjukkan perilaku berbahaya, string extraction digunakan untuk mengekstrak string dari file malware, yang dapat memberikan informasi penting tentang fungsi malware, seperti URL atau perintah yang digunakan, dan header analysis dilakukan untuk memahami struktur file malware, termasuk informasi tentang tipe file dan potensi teknik penghindaran deteksi.

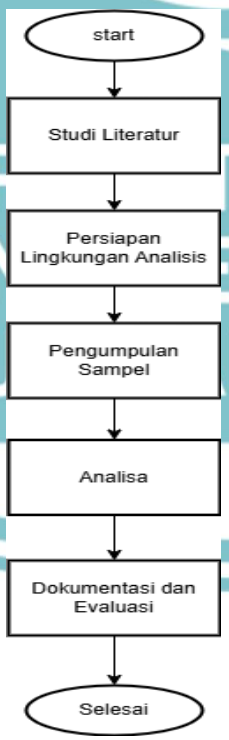
Setelah analisis statis, proses berlanjut ke analisis dinamis. Pada tahap ini, menggunakan tools Any.Run untuk mengeksekusi malware dalam lingkungan terisolasi untuk memantau perilakunya secara real-time. Teknik yang digunakan dalam analisis dinamis meliputi dynamic monitoring, di mana aktivitas sistem

selama eksekusi malware dicatat, serta system behavior analysis dan network analysis yang memantau interaksi malware dengan sistem operasi dan jaringan. Hal ini bertujuan untuk memahami bagaimana Ransomware Akira, LockBit BlackCat, WhisperGate, HellCat, Morpheus, dan Prince beroperasi saat dijalankan, termasuk perubahan yang mereka buat pada file sistem dan komunikasi dengan server command and control.

Selanjutnya, tahap *reverse engineering* dilakukan menggunakan alat seperti decompiler dan debugger untuk menganalisis kode secara mendalam. Proses ini bertujuan untuk memahami logika internal dari ransomware, termasuk algoritma enkripsi, teknik penghindaran dan teknik penyebaran. Dengan reverse engineering, peneliti dapat mengidentifikasi cara ransomware menyebar dan berinteraksi dengan sistem target.

3.2 Tahapan Penelitian

Adapun tahapan yang dilakukan dalam penelitian ini, diantaranya:



Gambar 3. 2 Tahapan Penelitian

Pada gambar 3.2 merupakan tahapan yang akan dilakukan pada penelitian ini yaitu melalaui beberapa tahapan dalam penyelesaian permasalahan yang diangkat, sebagai berikut:

1. Studi Literatur

   Pada tahap ini dilakukan untuk menentukan mengumpulkan referensi terkait analisis terhadap malware atau ransomware dengan metode analisis statis, analisis dinamis, dan reverse engineering. Referensi melibatkan jurnal, buku, situs web, serta sumber informasi relevan untuk membangun dasar teori yang relevan dalam pelaksanaan penelitian ini.
2. Persiapan Lingkungan Analisis

   Tahapan ini merupakan persiapan infrastuktur yang dibutuhkan untuk analisis, seperti instalasi alat-alat yang digunakan. Serta mengatur dan mengonfigurasi VirtualBox agar menjadi lingkungan terisolasi yang aman untuk proses analisis.
3. Pengumpulan Sampel

   Pada tahap ini dilakukan pengumpulan sampel malware yang dibutuhkan melalui repositori malware yang terpercaya seperti TheZoo, VirusTotal, atau Malware Bazaar. Setiap sampel yang dikumpulkan akan diverifikasi untuk memastikan keasliannya sebelum melanjutkan ke tahap analisis lebih lanjut. Pada penelitian ini beberapa sumber keaslian dari informasi sampel didapatkan melalui *advisory* yang dikeluarkan oleh pihak berwengan seperti CISA.
4. Proses Analisis

   Sampel yang sudah diunduh selanjutnya akan diuji degan pendekatan tiga metode:

   1. Analisis Statis

      Melihat struktur file tanpa menjalankan program, termasuk pemeriksaan header file, strings, dan libraries yang digunakan menggunakan PeStudio.
   2. Analisis Dinamis

      Menjalankan malware dalam lingkungan terkontrol (*sandbox*) untuk memantau perilaku dan aktivitasnya dengan menggunakan Any.Run

3. *Reverse Engineering*

   Membongkar kode untuk memahami fungsionalitas detail malware dengan Ghidra.

5. Dokumentasi dan Evaluasi

   Langkah terakhir dalam proses penelitian ini adalah melakukan dokumentasi dan mencatat hasil analisis, termasuk identitas, teknik yang digunakan, proses yang dilakukan, dampak serta potensi migrasi atau pencegahan. Hasil ini akan dievaluasi untuk memberikan rekomendasi penelitian yang relevan

3.3 Objek Penelitian

Objek penelitian utama dalam analisis ini adalah malware Ransomware Akira, LockBit BlackCat, WhisperGate, HellCat, Morpheus, dan Prince. Serta dua sampel tambahan Morpheus dan Prince dengan tujuan sebagai pembanding dan validasi. Sampel-sampel malware ini akan dianalisis menggunakan metode analisis statis, analisis dinamis, serta *reverse engineering*. Sampel malware tersebut didapatkan dengan cara mengunduh pada *TheZoo* pada repositori GitHub, MalwareBazaar, atau situs repositori malware lainnya. Fokus pengimplementasian analisis pada penelitian ini menggunakan Kali Linux dan Windows 10 sebagai lingkungan isolasinya.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Analisis Kebutuhan

4.1.1 Analisis Kebutuhan Tools dan Software

Pada penelitian ini diperlukan kejelian dalam pemilihan tools yang digunakan untuk membantu seluruh proses analisis. Dalam prosesnya, analisis malware dilakukan dengan menggunakan *virtual machine* (VM) yang didasarkan pada pertimbangan keamanan dan fleksibilitas penggunaannya. Dengan menggunakan VM, peneliti mampu menjalankan sampel malware di lingkungan yang terisolasi tanpa mempengaruhi sistem utama (*host)*. Isolasi ini sangat penting untuk mencegah penyebaran malware secara tidak sengaja ke jaringan atau perangkat lain. Berikut tools yang dibutuhkan untuk dapat melakukan penelitian terhadap analisis statis, dinamis, dan *reverse engineering* dalam eksekusi ransomware:

Tabel 4. 1 Analisis Kebutuhan

Tools	Fungsi
Oracle VirtualBox 7.1.10	Virtual Machine
Kali Linux	Sistem operasi VM sebagai tempat analisis dinamis dan *reverse engineering,* diinstal pada VirtualBox
Windows 10	Sistem operasi VM sebagai tempat analisis statis dengan PEStudio
File Ransomware .exe	Sampel ransomware dalam format executable (.exe) yang menjadi objek penelitian.
PeStudio	Tools untuk analisis statis header PE dan struktur internal file executable.
Strings	Tools untuk mengekstraksi string ASCII dan Unicode dari binary ransomware.
Any.Run	Sandbox berbasis cloud untuk

	menganalisis perilaku malware secara dinamis.
Ghidra vers. 11.3.1	Framework reverse engineering untuk melakukan disassembly dan analisis kode biner.

Pada Tabel 4.1, terlihat sejumlah tools yang digunakan untuk membantu melakukan seluruh proses analisis dalam penelitian ini. Penelitian ini dilakukan di dalam Virtual Box sebagai tempat berlangsungnya seluruh analisis dan semua sampel malware dan tools yang digunakan, diunduh dan dijalankan didalam lingkungan *virtual machine*.

Kali Linux berfungsi sebagai sistem operasi utama dalam mesin virtual yang digunakan untuk analisis dinamis dan reverse engineering. Kali Linux dipilih karena menyediakan berbagai tools keamanan dan analisis malware yang lengkap, seperti Ghidra, Wireshark, dan Strings. Sistem operasi ini diinstal pada VirtualBox untuk memastikan keamanan lingkungan analisis. (Gururaj H et al., 2022)

PE Studio merupakan salah satu perangkat lunak yang digunakan untuk menganalisis file *portable executable (PE)* pada sistem operasi Windows. File PE adalah jenis file yang dapat dieksekusi, dan PE Studio memungkinkan pengguna untuk menelusuri berbagai komponen internal dari file tersebut, seperti fungsi impor dan ekspor, pustaka DLL yang digunakan, string, serta elemen lainnya. Dengan menggunakan PE Studio, pengguna dapat mengidentifikasi potensi kode berbahaya, kerentanan keamanan, atau bagian yang mencurigakan dalam sebuah file executable. Alat ini banyak dimanfaatkan oleh analis dan peneliti malware untuk memahami struktur dan perilaku internal dari sebuah program secara lebih mendalam.(Bhardwaj, 2024)

Penggunaan *command* Strings pada terminal digunakan dalam ekstraksi strings. Ekstraksi string merupakan metode untuk mengambil kata dan karakter yang dapat dibaca dari sebuah file yang dicurigai berisi malware. String ini biasanya tersimpan dalam format ASCII (*American Standard Code for Information*

Interchange) maupun Unicode (*Universal Character Encoding*). Informasi yang ditemukan dalam string bisa berupa nama file, nama domain, alamat IP, perintah berbahaya, maupun URL (*Uniform Resource Locator*). (Banik, 2023)

ANY.RUN adalah platform *sandbox* berbasis cloud yang memberikan lingkungan interaktif untuk melakukan analisis dinamis terhadap sampel malware. Dalam penelitian ini, ANY.RUN digunakan sebagai alat utama untuk mengamati perilaku malware saat dieksekusi dalam sistem virtual. Platform ini memungkinkan peneliti untuk mengunggah file berbahaya dan memantau aktivitasnya secara real-time, termasuk proses yang dijalankan, modifikasi pada file dan registry, serta aktivitas jaringan seperti koneksi, permintaan HTTP, dan DNS. (Sato, 2024)

Ghidra adalah tools reverse engineering open-source yang dikembangkan oleh NSA. Dalam penelitian ini, Ghidra digunakan untuk membongkar (disassemble) dan menganalisis kode mesin (assembly) dari ransomware. Tools ini sangat berguna untuk melacak alur eksekusi malware, memahami struktur fungsi, serta mengidentifikasi teknik enkripsi yang digunakan oleh ransomware secara mendalam.

4.1.2 Sampel Ransomware sebagai Objek Penelitian

Dalam penelitian ini, sampel-sampel yang dijadikan objek penelitian didapatkan melalui laman Malware Bazaar serta repositori sampel ransomware untuk penelitian pada GitHub. Untuk menjalankan dan menganalisis, sampel-sampel tersebut yang telah didapatkan diekstraksi dari yang sebelumnya berbentuk file berekstensi Zip. Beberapa sampel yang didapatkan antara lain:

Tabel 4. 2 Sampel Ransomware

No.	Nama Sampel	Nilai Hash SHA256
1.	Akira.exe	5DE1061457E759E022FBC9BB02E8726A49D3ED9663FC8A77D83462C69C96AEA8

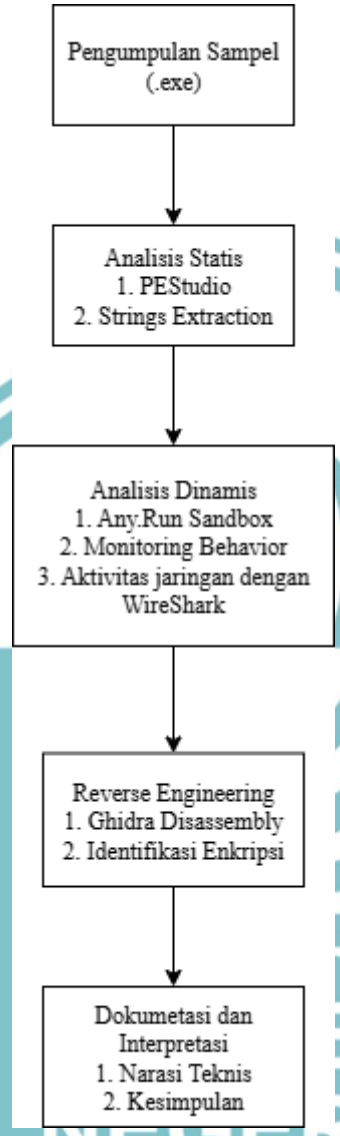
2.	Lockbit.exe	F4FB0F2AE098850F2A8FFB771AE4C6C8 AAA81144FE53228A2C01DF2D34307053
3.	Blackcat.exe	E160B6348F6FBDC444125BB65DBD9460 6D99DA48C8334C8E6B2EE4429F813293
4.	Whispergate.exe	82EF68548D2E2E4B2C41A1C92D06B30B 0A433B17B859590989C89D4CE9162033
5.	Hellcat.exe	B8E71845CC8CCD668A3436D1952A6C57 649974BB8399E599DC33AFC4C0843BE7
6.	Morpheus.exe	4B2EDADC8F90E9FCC976F02A9EDA164 0CD92C07718C0271842FBD4CA7E2906E 2
7.	Prince-Built.exe	B7A6EB4DC4C644FE3611B0F4E6920245 5D61CF426726343B2FFB182C3DEB6CB8

4.2 Perancangan Sistem

Dalam penelitian ini, pendekatan perancangan sistem difokuskan pada proses analisis malware ransomware secara menyeluruh, mulai dari tahap pengumpulan sampel hingga identifikasi fungsi enkripsi. Proses analisis dilakukan dalam tiga pendekatan utama, yaitu:

1. Analisis Statis
2. Analisis Dinamis
3. Reverse Engineering

Ketiga pendekatan ini dirancang agar saling melengkapi, di mana hasil dari satu tahap dapat mendukung atau memperkuat temuan pada tahap lainnya. Secara lebih jelas dijabarkan seperti pada gambar dibawah ini:



Gambar 4. 1 Perancanaan Sistem

Terlihat pada Gambar 4.1, berikut penjelasan detail dari masing masing proses:

1. Pengumpulan Sampel

Tahap awal adalah mengumpulkan sampel ransomware dalam bentuk file .exe. Sampel ini dapat berasal dari sumber repository malware terpercaya seperti VirusShare, MalwareBazaar, atau hasil capture dari sandbox.

2. Analisis Statis

Analisis ini dilakukan tanpa mengeksekusi file ransomware. Tools seperti PEStudio digunakan untuk mengevaluasi struktur internal dari file executable, termasuk import table, header, dan indikasi API mencurigakan. Strings digunakan

untuk mengekstrak string readable dari dalam file yang dapat memberikan petunjuk awal tentang teknik atau data yang digunakan ransomware.

3. Analisis Dinamis

Analisis dinamis dilakukan dengan mengeksekusi ransomware dalam lingkungan virtual yang aman (*sandbox*) menggunakan Any.Run. Tujuannya adalah untuk mengamati perilaku aktual ransomware, seperti aktivitas file system, registry, network, dan proses enkripsi. Hasil pengamatan ini juga digunakan untuk mengidentifikasi command-and-control (C2), mutex, atau teknik penghindaran deteksi.

4. Reverse Engineering

Tahap akhir adalah reverse engineering menggunakan tools Ghidra. Pada tahap ini, ransomware dibongkar ke level assembly dan dilakukan penelusuran terhadap fungsi-fungsi yang terlibat dalam proses enkripsi, manipulasi memori, serta teknik anti-debugging. Fungsi-fungsi penting seperti entry point, *encryption routine*, hingga *dynamic API loading* diidentifikasi secara manual.

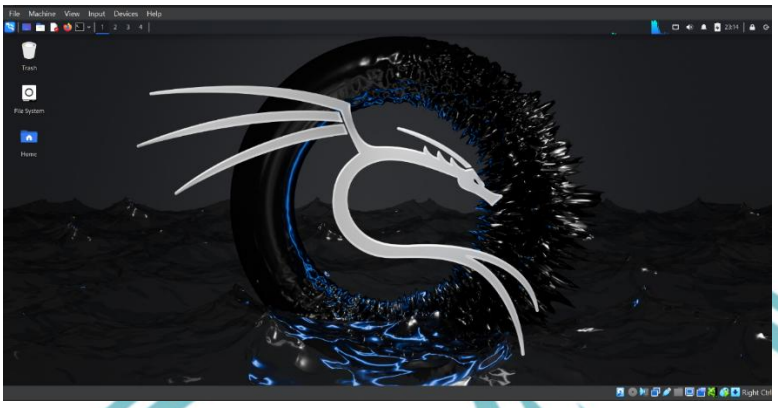
5. Dokumentasi

Semua temuan dari setiap tahapan dianalisis, dibandingkan, dan didokumentasikan dalam bentuk narasi, diagram, serta tabel fungsi untuk memberikan pemahaman menyeluruh terhadap cara kerja ransomware.

4.3 Implementasi Sistem

Implementasi sistem akan dilakukan didalam *virtual machine* menggunakan os Kali Linux dan Windows 10. Dalam menjalankan proses analisis, seluruh tools dan software akan diinstal pada VM. Dengan keterangan, PEStudio akan dijalankan pada Windows, sedangkan Any.Run dan Ghidra akan dijalankan pada Kali Linux.

4.3.1 Kali Linux

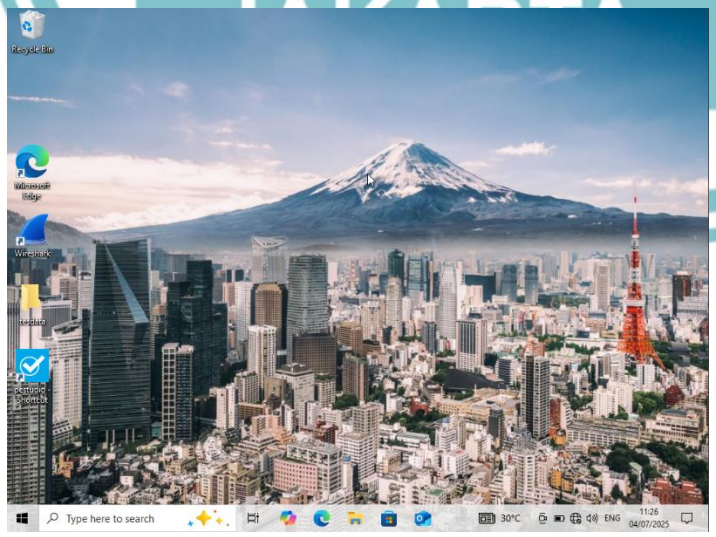


Gambar 4. 2 Tampilan Kali Linux

Terlihat pada Gambar 4.2 merupakan tampilan Kali. Pada tahap implementasi sistem, Kali Linux digunakan sebagai sistem operasi utama untuk melakukan analisis dinamis dan reverse engineering terhadap sampel ransomware. Kali Linux dijalankan di dalam lingkungan virtual melalui Oracle VirtualBox agar proses analisis berjalan secara aman dan terisolasi dari sistem utama.

Setelah instalasi, Kali Linux dikonfigurasi dengan sejumlah tools pendukung seperti hexeditor, objdump, strings, serta Ghidra versi 11.3.1 yang digunakan untuk proses *reverse engineering*. Selain itu, Kali Linux juga digunakan untuk mengakses layanan sandbox berbasis cloud Any.Run melalui web browser bawaan sistem.

4.3.2 Windows 10



Gambar 4. 3 Tampilan Windows

Sistem operasi Windows 10 digunakan dalam tahap implementasi untuk melakukan analisis statis terhadap sampel ransomware. Sistem ini dijalankan pada lingkungan virtual melalui Oracle VirtualBox dan dikonfigurasi tanpa koneksi internet guna mencegah potensi penyebaran malware selama proses analisis.

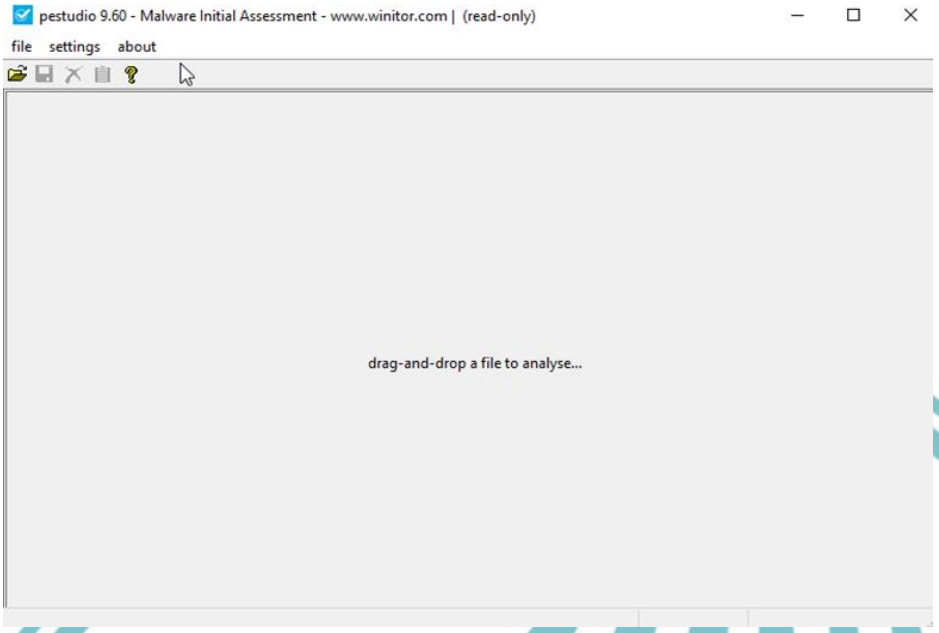
Dalam lingkungan Windows 10 ini, peneliti menggunakan tools utama yaitu PEStudio dan Strings untuk mengekstrak informasi dari file executable (.exe) tanpa mengeksekusinya secara langsung.

- PEStudio digunakan untuk membaca struktur internal file ransomware, seperti header PE, import table, resource section, dan direktori API yang digunakan. Tools ini juga mendeteksi indikator berbahaya (*indicator of compromise*/IOC) dengan memberikan penilaian terhadap fungsi API yang mencurigakan, teknik packing, serta metadata file.
- Strings digunakan untuk mengekstrak string ASCII dan Unicode dari dalam file ransomware. Ekstraksi ini bertujuan untuk mengidentifikasi petunjuk seperti nama file terenkripsi, alamat IP, nama domain, kunci enkripsi, serta potensi ransom note atau pesan tersembunyi lainnya yang disisipkan di dalam binary.

Seluruh proses dilakukan tanpa mengeksekusi file ransomware, sehingga murni berbasis pada analisis struktur dan isi file. Hasil dari analisis ini kemudian digunakan untuk mengarahkan proses lanjutan seperti analisis dinamis atau reverse engineering. Penggunaan Windows 10 dalam konteks ini memungkinkan tools seperti PEStudio berjalan optimal karena tools tersebut dirancang khusus untuk sistem Windows.

4.3.3 Analisis Statis dengan PEStudio dan Ekstraksi String

Sebagai tahapan awal proses analisis ransomware, analisis statis digunakan dalam mengidentifikasi karakteristik tanpa perlu mengeksekusi sampel secara langsung. Pada penelitian ini, analisis statis dilakukan dengan menggunakan dua pendekatan utama, yaitu melalui aplikasi PEStudio dan tool ekstraksi string bawaan sistem operasi.



Gambar 4. 4 Tampilan PEStudio

Pada Gambar 4.4, merupakan tampilan dari PeStudio. Aplikasi tersebut digunakan untuk menganalisis struktur internal file Portable Executable (PE) dari masing masing sampel ransomware. Analisis dilakukan dengan memuat file ke dalam antarmuka aplikasi, yang secara otomatis menghasilkan laporan terperinci terkait atribut file.

Terdapat beberapa atribut penting yang dapat diamati, antara lain:

1. Nilai Hash (MD5, SHA-1, dan SHA-256) untuk mengidentifikasi dan memvalidasi file.
2. Informasi Signature Digital, untuk mengetahui apakah file memiliki tanda tangan resmi dari vendor perangkat lunak, seperti Windows.
3. Struktur Section, untuk mendeteksi anomali seperti adanya section yang tidak lazim dengan nilai persentase entropi yang dapat menjadi indikasi adanya enkripsi atau obfuscated.
4. Daftar fungsi API yang diimpor, sebagai indikasi aktivitas berbahaya seperti VirtualAlloc, WriteProcessMemory, ShellExecuteW, atau fungsi terkait jaringan.
5. DLL atau eksternal library, untuk mengetahui dependensi terhadap komponen Windows lainnya.

```
C:\Users\layla\Downloads\Strings>Strings sampel.exe > hasil_strings.txt
```

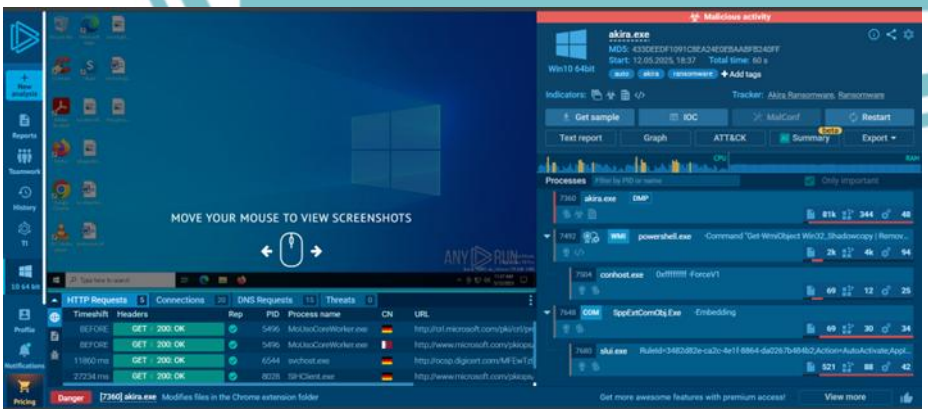
Gambar 4. 5 Command Ekstraksi String

Selain itu, dilakukan juga ekstraksi strings dari setiap sampel ransomware dengan menggunakan perintah pada strings, yang dijalankan melalui terminal seperti pada Gambar 4.5. Proses ini bertujuan untuk mengidentifikasi informasi tersembunyi dalam bentuk teks yang dapat memberikan indikasi awal terhadap perilaku ransomware. Beberapa informasi yang didapatkan dari string yang menjadi perhatian khusus untuk dilakukan analisis lebih lanjut antara lain:

1. Kata kunci seperti “encrypt”, “decrypt”, “ransom”, dan lainnya.
2. Pola strings obfuscation
3. Alamat IP atau URL yang mengarah ke server Command and Control (C2).
4. Jalur direktori atau nama file yang dijadikan target serangan oleh ransomware dalam melakukan enkripsi, seperti Documents, Desktop, Pictures, dan lainnya.

4.3.4 Analisis Dinamis dengan Any.Run

Analisis dinamis merupakan metode pengamatan perilaku ransomware selama proses eksekusi dalam lingkungan yang terisolasi. Pendekatan ini bertujuan untuk memahami dampak nyata dari ransomware terhadap sistem, termasuk modifikasi file, aktivitas jaringan, serta integrasi dengan registry dan proses sistem lainnya.



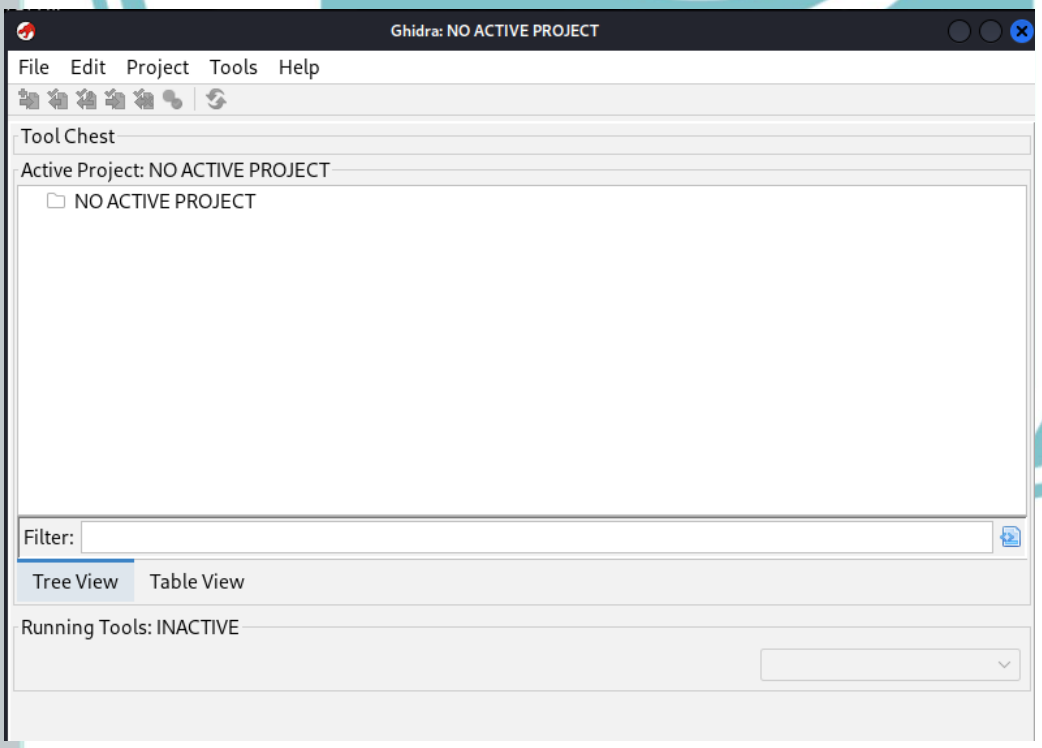
Gambar 4. 6 Tampilan Interface Any.Run

Dalam penelitian ini, analisis dinamis dilakukan menggunakan platform sandbox berbasis cloud, yaitu Any.Run, seperti pada Gambar 4.6. Sampel ransomware diunggah pada laman https://app.any.run/ dan dijalankan dalam lingkungan virtual Kali Linux yang telah dikonfigurasi untuk mengamati perilaku malware secara Real-Time tanpa resiko penyebaran infeksi ke sistem utama.

Terdapat beberapa parameter yang menjadi fokus pengamatan meliputi diantaranya:

1. Proses dan Thread baru yang dibuat selama eksekusi.
2. Perubahan pada sistem file, seperti pembuatan, modifikasi, atau penghapusan file.
3. Perubahan pada registry Windows yang dapat mempengaruhi perilaku sistem atau persistensi ransomware.
4. Aktivitas jaringan, termasuk koneksi keluar ke server Command and Control (C2) atau URL berbahaya

4.3.5 *Reverse Engineering* dengan Ghidra



Gambar 4. 7 Tampilan Interface Ghidra

Ghidra versi 11.3.1 digunakan sebagai tools utama dalam tahap reverse engineering untuk membongkar struktur internal dari sampel ransomware hingga ke level kode mesin (assembly), seperti pada Gambar 4.7. Ghidra dijalankan di dalam sistem operasi Kali Linux, yang sebelumnya telah dikonfigurasi untuk mendukung Java Development Kit (JDK) sebagai dependensi utama agar Ghidra dapat berjalan dengan optimal.

Tahapan implementasi Ghidra meliputi:

1. Pembuatan Project Baru

   Peneliti membuat proyek baru pada Ghidra dan mengimpor file ransomware .exe ke dalam workspace. Ghidra secara otomatis melakukan proses analisis awal (*auto-analysis*), termasuk identifikasi fungsi, struktur memori, dan referensi antar instruksi.

2. Disassembly dan Decompiler View

   Setelah proses analisis selesai, peneliti mengakses view disassembly untuk menelusuri alur instruksi assembly dari ransomware. Ghidra juga menyediakan fitur *decompiler* yang menerjemahkan kode assembly ke dalam bentuk pseudocode C-like, sehingga memudahkan pemahaman logika program.

3. Identifikasi Fungsi Utama

   Dengan bantuan Ghidra, fungsi-fungsi penting dalam ransomware seperti entry point, pembuatan thread, proses enkripsi, serta teknik pengacakan (obfuscation) dapat dianalisis secara manual. Peneliti juga menambahkan label dan komentar pada fungsi yang telah dikenali, sehingga memudahkan pelacakan alur eksekusi.

4. Pelacakan String dan Referensi

   Ghidra digunakan untuk menemukan string internal, pointer, dan referensi ke API yang sebelumnya sudah diidentifikasi melalui analisis statis. Dengan fitur *cross-references*, peneliti dapat melacak di mana dan bagaimana sebuah string atau API digunakan dalam kode.

Implementasi Ghidra ini memberikan pemahaman yang mendalam mengenai struktur dan perilaku ransomware dari sisi teknis, terutama untuk mengenali bagian kode yang bertanggung jawab atas proses enkripsi dan perusakan data. Reverse engineering menggunakan Ghidra juga menjadi tahap kunci dalam mengonfirmasi hasil dari analisis statis maupun dinamis yang telah dilakukan sebelumnya.

4.4 Hasil Pengujian

Berikut hasil pengujian dari implementasi sistem dijabarkan berdasarkan ketiga pendekatan analisis yang digunakan:

4.4.1 Hasil Analisis Statis dengan PEStudio

Berdasarkan parameter pengujian didapatkan hasil dari masing masing sampel, diantaranya:

4.4.1.1 Akira

property	value
file	
file > sha256	5DE1061457E759E022FBC9BB02E8726A49D3ED9663FC8A77D83462C69C96AEA8
file > first 32 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 00
file > first 32 bytes (text)	MZ.....@.....
file > info	size: 1081856 bytes, entropy: 6.439
file > type	executable, 64-bit, GUI
file > version	n/a
file > description	n/a
entry-point > first 32 bytes (hex)	48 83 EC 28 E8 1B 08 00 00 48 83 C4 28 E9 7A FE FF FF CC CC 48 83 EC 28 4D 8B 41 38 48 8B C
entry-point > location	0x0008DD38 (section[.text])
file > signature	Microsoft Linker 14.37 | Visual Studio 2015
stamps	
stamp > compiler	Wed Mar 19 17:53:17 2025 (UTC)
stamp > debug	Wed Mar 19 17:53:17 2025 (UTC)
stamp > resource	n/a
stamp > import	n/a
stamp > export	n/a
names	
file > name	c:\users\layla\downloads\analisis\akira.exe
debug > file	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module > name	n/a
certificate > program-name	n/a

Gambar 4. 8 Property File Akira

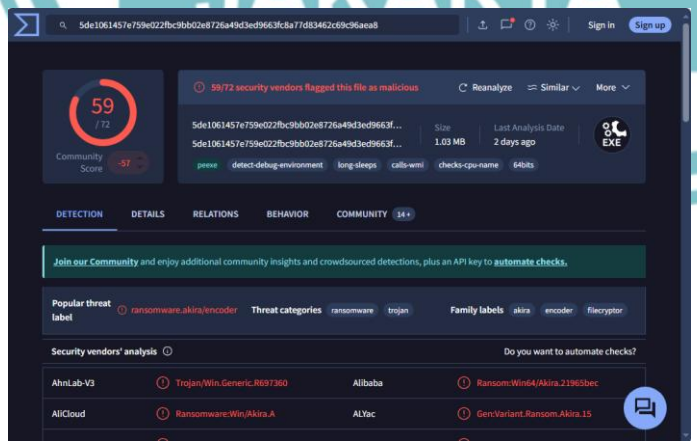
Pada gambar 4.8, merupakan informasi indicator struktur dalam file sampel, yang bila dirincikan seperti tabel 4.3 dibawah ini:

Tabel 4. 3 Informasi File Akira

Atribut	Nilai

Nama Sampel	akira.exe
MD5	433DEEDF1091C8EA24E0EBAABFB240FF
SHA-256	5DE1061457E759E022FBC9BB02E8726A49 D3ED9663FC8A77D83462C69C96AEA8
Entropy	6.439 (tinggi)
File Type	Executable
Ukuran File	1.081.856 byte
First Byte Text	M Z .. .. @ . . . . .
Entry Point	0x0008DD38 (section[.text])
Compiler	Microsoft Linker 14.37 (Visual Studio 2015)
Processor	64-bit, GUI
Skor VirusTotal	59/72

Berdasarkan data pada tabel 4.3, temuan memberikan gambaran teknis mengenai karakteristik file akira.exe. file ini memiliki struktur executable 64-bit dengan antarmuka GUI, serta menunjukkan tidak adanya informasi tanda tangan digital dari vendor resmi, yang menandakan bahwa file tersebut tidak berasal dari vendor perangkat lunak yang resmi dan legal. Selain itu, tingkatan entropi yang tinggi memperkuat indikasi bahwa file telah dimodifikasi, baik melalui teknik packing atau encryption, yang umum digunakan oleh ransomware untuk menghindari proses deteksi (*obfuscation*).



Gambar 4. 9 Nilai File Akira pada VirusTotal

Berdasarkan hasil nilai analisis dari virusTotal yang terlihat pada PeStudio, pada Gambar 4.9, tertera bahwa sampel tersebut menunjukkan skor 59 dari 72 yang mengartikan bahwa file tersebut teridentifikasi sebagai file berbahaya yang dilaporkan oleh sejumlah besar vendor antivirus. Dengan ukuran file seberat 1.03 MB.

library (9)	flag (4)	type (1)	imports (155)	description (4)
KERNEL32.dll	-	Implicit	134	Windows NT BASE API Client
SHELL32.dll	-	Implicit	2	Windows Shell Library
ole32.dll	-	Implicit	4	Microsoft OLE for Windows
OLEAUT32.dll	-	Implicit	4	oleaut32 library
WS2_32.dll	x	Implicit	2	Windows Socket Library
SHLWAPI.dll	-	Implicit	1	Shell Light-weight Utility Library
MPR.dll	x	Implicit	1	Multiple Provider Router Library
WTSAPI32.dll	x	Implicit	2	Windows Remote Desktop Session Host Server SDK APIs
RstrtMgr.DLL	x	Implicit	5	Restart Manager library

Gambar 4. 10 Import Library Akira

Pada Gambar 4.10, menunjukkan bahwa sampel akira.exe mengimpor sejumlah library sistem windows yang umumnya digunakan pada aplikasi, namun pada konteks malware, keberadaan library patut dilakukan analisis lebih lanjut, sebab memberikan gambaran mengenai perilaku program, termasuk kemampuan untuk mengakses sistem file, memanipulasi proses, komunikasi dalam jaringan, hingga melakukan penguncian sistem.

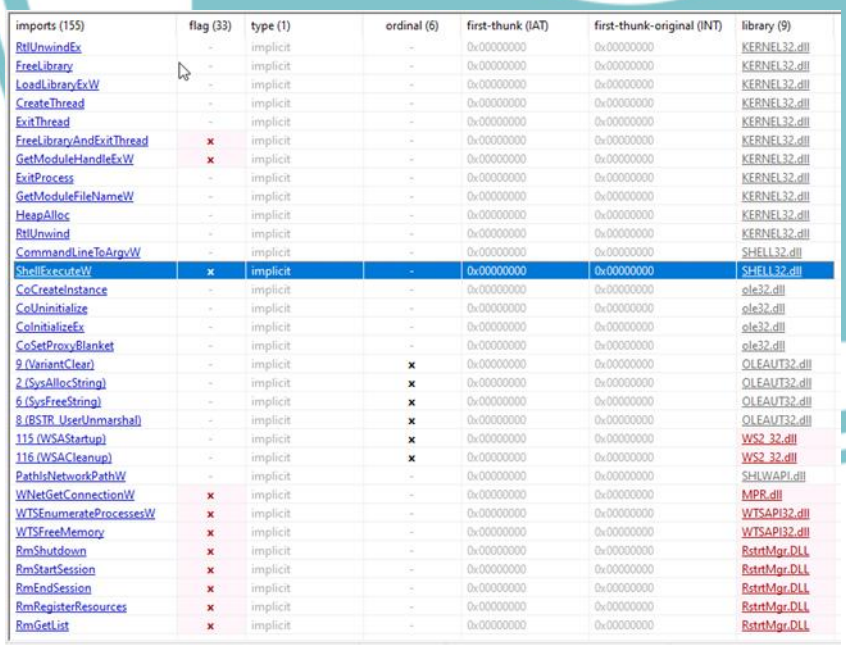
Berdasarkan hasil yang didapatkan, library yang diimpor oleh akira.exe, antara lain:

Tabel 4. 4 Fungsi Library Akira

Library	Fungsi
KERNEL32.dll	Digunakan untuk membuat fungsi. menulis file terenkripsi, menghapus data backup.
SHELL32.dll	Digunakan untuk menjalankan file atau membuka folder dan menampilkan pesan.
ole32.dll	Dipakai untuk menjalankan komponen COM yang terhubung ke sistem atau registry
OLEAUT32.dll	Digunakan ransomware untuk memanfaatkan *scripting* atau otomatisasi Windows
WS_32.dll	Digunakan oleh ransomware untuk komunikasi dengan server *Command and Control* untuk enkripsi kunci

SHLWAPI.dll	Digunakan untuk memanipulasi path file atau nama file sebelum enkripsi
MPR.dll	Digunakan untuk penyebaran ransomware ke jaringan atau mengenkripsi file pada *shared folder*
WTSAPI32.dll	Digunakan untuk memantau dan memanipulasi sesi user, termasuk mematikan sesi yang sedang aktif
RstrtMgr.DLL	Digunakan untuk mematikan layanan yang bisa menghalangi proses enkripsi

Secara keseluruhan, import library yang digunakan menunjukkan bahwa sampel ransomware akira.exe berjalan pada berbagai fungsi inti sistem Windows. Library tersebut tidak hanya digunakan pada proses dasar seperti membaca dan menulis file, tetapi juga untuk mengakses jaringan, mengatur sesi pengguna, hingga memanipulasi antarmuka sistem. Hal ini mengindikasikan bahwa ransomware akira dirancang untuk menjalankan proses enkripsi data, berkomunikasi dengan jaringan luar, dan memaksimalkan kontrol atas lingkungan sistem.



imports (155)	flag (33)	type (1)	ordinal (6)	first-thunk (IAT)	first-thunk-original (INT)	library (9)
RtlUnwindEx	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
FreeLibrary	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
LoadLibraryExW	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
CreateThread	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
ExitThread	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
FreeLibraryAndExitThread	x	implicit	-	0x00000000	0x00000000	KERNEL32.dll
GetModuleHandleExW	x	implicit	-	0x00000000	0x00000000	KERNEL32.dll
ExitProcess	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
GetModuleFileNameW	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
HeapAlloc	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
RtlUnwind	-	implicit	-	0x00000000	0x00000000	KERNEL32.dll
CommandLineToArgvW	-	implicit	-	0x00000000	0x00000000	SHELL32.dll
ShellExecuteW	x	implicit	-	0x00000000	0x00000000	SHELL32.dll
CoCreateInstance	-	implicit	-	0x00000000	0x00000000	ole32.dll
CoUninitialize	-	implicit	-	0x00000000	0x00000000	ole32.dll
CoInitializeEx	-	implicit	-	0x00000000	0x00000000	ole32.dll
CoSetProxyBlanket	-	implicit	-	0x00000000	0x00000000	ole32.dll
9 (VariantClear)	-	implicit	x	0x00000000	0x00000000	OLEAUT32.dll
2 (SysAllocString)	-	implicit	x	0x00000000	0x00000000	OLEAUT32.dll
6 (SysFreeString)	-	implicit	x	0x00000000	0x00000000	OLEAUT32.dll
8 (BSTR_UserUnmarshal)	-	implicit	x	0x00000000	0x00000000	OLEAUT32.dll
115 (WSAStartup)	-	implicit	x	0x00000000	0x00000000	WS2_32.dll
116 (WSACleanup)	-	implicit	x	0x00000000	0x00000000	WS2_32.dll
PathIsNetworkPathW	-	implicit	-	0x00000000	0x00000000	SHLWAPI.dll
WNetGetConnectionW	x	implicit	-	0x00000000	0x00000000	MPR.dll
WTSEnumerateProcessesW	x	implicit	-	0x00000000	0x00000000	WTSAPI32.dll
WTSFreeMemory	x	implicit	-	0x00000000	0x00000000	WTSAPI32.dll
RmShutdown	x	implicit	-	0x00000000	0x00000000	RstrtMgr.DLL
RmStartSession	x	implicit	-	0x00000000	0x00000000	RstrtMgr.DLL
RmEndSession	x	implicit	-	0x00000000	0x00000000	RstrtMgr.DLL
RmRegisterResources	x	implicit	-	0x00000000	0x00000000	RstrtMgr.DLL
RmGetList	x	implicit	-	0x00000000	0x00000000	RstrtMgr.DLL

Gambar 4. 11 String API Akira

Pada gambar 4.11, ditemukan bahwa sampel ransomware akira.exe mengimpor sejumlah besar fungsi API yang merupakan bagian dari library. Fungsi-fungsi

ini memberikan gambaran mengenai cara kerja malware dalam melakukan proses kerjanya. Beberapa fungsi kunci yang diidentifikasi antara lain:

1. Fungsi manipulasi file

   Seperti WriteFile, DeleteFileW, SetFileAttributesW, CreateDirectoryW, FindFirstFile, dan FindNextFileW menunjukkan bahwa sampel ini memiliki kemampuan untuk membuat, menulis, memodifikasi, dan menghapus file. Hal ini menunjukkan adanya proses enkripsi file korban serta penghapusan jejak digital.
2. Fungsi proses dan thread

   Seperti OpenProcess, GetCurrentProcess, GetCurrentProcessId, GetCurrentThreadId, QueueUserAPC, dan FreeLibraryAndExit Thread menunjukkan bahwa adanya mekanisme pengelolaan proses dan thread yang bersifat kompleks, termasuk potensi injeksi kode atau manipulasi proses lain di dalam sistem.
3. Fungsi jaringan

   Seperti WNetGetConnectionW, WSAStartup, dan WSACleanup memiliki indikasi bahwa ransomware memiliki kemampuan untuk terhubung ke jaringan atau melakukan komunikasi data yang digunakan untuk mengirim hasil enkripsi atau menerima perintah dari server Command and Control (C2).
4. Fungsi lingkungan sistem dna enkripsi

   Seperti SetEnvirontmentVariableW, GetEnvirontmentStringsW dan GetLogicalDriveStringW berfungsi untuk mengidentifikasi dan menyesuaikan lingkungan eksekusi ransomware.
5. Fungsi GUI dan konsol

   Seperti GetConsoleScreenBuffetInfo dan RaiseException digunakan untuk memanipulasi atau menghasilkan error yang mempengaruhi perilaku pengguna
6. Fungsi Rm (Restart Manager)

   Seperti RmStartSession, RmShutdown, dan RmRegisterResources digunakan untuk mengelola proses atau aplikasi yang sedang berjalan,

seperti menutup aplikasi yang mengunci file (misal database atau dokumen) sebelum proses enkripsi dimulai.

7. Serta beberapa fungsi seperti VariantClear, SysAllocString, dan BSTR_UserUnmarshal berasal dari library OLEAUT32.dll dan ole32.dll yang biasa digunakan untuk komunikasi antar komponen dalam aplikasi berbasis COM. Hal ini menunjukkan bahwa ransomware memiliki interaksi yang lebih dalam dengan komponen Windows tingkat sistem.

Seperti yang terlihat pada gambar 4.11, beberapa fungsi seperti VariantClear, SysAllocString, SysFreeString, BSTR_UserUnmarshal, WSAStartup, dan WSACleanup ditandai dengan simbol x hitam pada kolom ordinal, yang artinya hal ini mengindikasikan bahwa fungsi tersebut diimpor berdasarkan nomor ordinal. Nomor ordinal merupakan identifikasi numerik unik yang digunakan untuk memanggil fungsi dalam suatu library dinamis (DLL) tanpa menyebutkan nama fungsi secara eksplisit. Teknik impor berdasarkan ordinal ini sering dimanfaatkan oleh malware untuk menyembunyikan nama fungsi yang digunakan dan menghindari deteksi atau obfuscation.

4.4.1.2 LockBit

property	value
file	
file > sha256	F4FB0F2AE098850F2A8FFB771AE4C6C8AAA81144FE53228A2C01DF2D34307053
file > first 32 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 00
file > first 32 bytes (text)	MZ.....@.....
file > info	size: 150016 bytes, entropy: 7.208
file > type	executable, 32-bit, GUI
file > version	n/a
file > description	n/a
entry-point > first 32 bytes (hex)	90 66 66 66 0F 1F 84 00 00 00 00 00 E8 80 FB FF FF 66 66 0F 1F 84 00 00 00 00 00 E8 0D CF FE FF
entry-point > location	0x0001946F (section[.data])
file > signature	Microsoft Linker 14.12
stamps	
stamp > compiler	Tue Sep 13 23:30:57 2022 (UTC)
stamp > debug	Tue Sep 13 23:30:57 2022 (UTC)
stamp > resource	n/a
stamp > import	n/a
stamp > export	n/a
names	
file > name	c:\users\layla\downloads\analisis\lockbit.exe
debug > file	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module > name	n/a
certificate > program-name	n/a

Gambar 4. 12 Property File LockBit

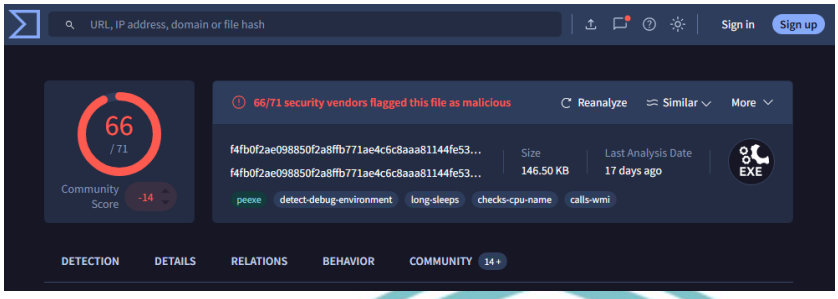
Pada Gambar 4.12, merupakan informasi indicator struktur dalam file sampel, yang bila dirincikan seperti pada tabel 4.5 dibawah ini

Tabel 4. 5 Informasi File LockBit

Atribut	Nilai
Nama Sampel	lockbit.exe
MD5	06F96CB31A2B655835130A09387FB401
SHA-256	F4FB0F2AE098850F2A8FFB771AE4C6C8A AA81144FE53228A2C01DF2D34307053
Entropy	7.208 (tinggi)
File Type	Executable
Ukuran File	150.016 byte
First Byte Tex	M Z . . . . . @ . . . . .
Entry Point	0x0001946F (section .data)
Compiler	Microsoft Linker 14.12
Processor	32-bit, GUI
Skor Nilai VirusTotal	66/71

Berdasarkan tabel 4.5, diketahui bahwa file sampel lockbit.exe menunjukkan sejumlah karakteristik mencurigakan. Nilai entropy yang tinggi sebesar 7.208, mengindikasikan kemungkinan besar adanya mekanisme *packing, obfuscation,* atau enkripsi untuk menyembunyikan kode bahaya dari proses analisis manual maupun otomatis.

Selain itu, posisi *entry point* pada sampel ini terletak pada .data dengan alamat 0x0001946F. Secara umum, eksekusi program normal dimulai dari section .text yang berisi kode mesin. Namun pada sampel ini entry point berada pada .data yang menunjukkan bahwa malware ini memakai teknik anti-analisis seperti *code injection* atau *packed code*.



Gambar 4. 13 Skor VirusTotal LockBit

Berdasarkan Gambar 4.13, merupakan penelusuran sampel pada VirusTotal dan menghasilkan nilai 66/71 yang menunjukkan bahwa 61 dari 71 vendor antivirus melaporkan bahwa file tersebut teridentifikasi sebagai file berbahaya.

property	value	value	value	value
section	section[0]	section[1]	section[2]	section[3]
name	.text	.itext	.rdata	.data
section > sha256	8226ED51E189C255710066C...	FED988545B8FA9D1C0BE54C...	5F8F6E8FBF6A80D1BC3F9A...	A0D0AD408
entropy	6.614	3.039	3.659	7.987
file > ratio (99.32%)	65.19 %	1.02 %	1.02 %	27.30 %
raw-address (begin)	0x00000400	0x00018200	0x00018800	0x00018E00
raw-address (end)	0x00018200	0x00018800	0x00018E00	0x00022E00
raw-size (148992 bytes)	0x00017E00 (97792 bytes)	0x00000600 (1536 bytes)	0x00000600 (1536 bytes)	0x0000A000
virtual-address	0x00001000	0x00019000	0x0001A000	0x0001B000
virtual-size (151687 bytes)	0x00017D46 (97606 bytes)	0x00000569 (1385 bytes)	0x000004B2 (1202 bytes)	0x0000ADC
characteristics	0x60000020	0x60000020	0x40000040	0xC0000040
write	-	-	-	x
execute	x	x	-	-
share	-	-	-	-
self-modifying	-	-	-	-
virtual	-	-	-	-
items				
directory > import	-	-	0x0001A230	-
directory > relocation	-	-	-	-
directory > debug	-	-	0x0001A120	-
directory > import-address	-	-	0x0001A000	-
base-of-code	0x00001000	-	-	-
base-of-data	-	-	0x0001A000	-
entry-point > location	-	0x0001946F	-	-
debug > PGO	-	-	0x0001893C	-

Gambar 4. 14 Section File LockBit

Pada Gambar 4.14, yang merupakan bagian pada section dari sampel, ditemukan sejumlah indikasi mencurigakan yang mengindikasikan aktivitas yang tidak sah. Entry point yang berada pada section .data dengan alamat virtual 0x0001946F, bukan di section .text yang sebagaimana umumnya file executable yang sah. Section .data juga memiliki hak akses *write* dan *execute* secara bersamaan, yang berpotensi digunakan untuk menyimpan serta mengeksekusi *payload* secara langsung dari memori. Selain itu, nilai entropy mencapa 7.987, mendekati nilai maksimum (8.0), yang mengindikasikan

kemungkinan adanya data terenkripsi atau terkompresi di dalamnya. Teknik ini umum digunakan oleh ransomware untuk menghindari deteksi statis dan menyembunyikan kode berbahaya dari analisis awal. Dengan demikian, manipulasi terhadap struktur section ini menunjukkan adanya upaya *stealth* dan teknik *evasion* yang menjadi ciri khas dari malware berjenis ransomware.

imports (25)	flag (0)	type (1)	ordinal (0)	first-thunk (IAT)	first-thunk-original (INT)	librar
SetPixel	-	implicit	-	0x0001A37A	0x0001A37A	gdi32
GetPixel	-	implicit	-	0x0001A33E	0x0001A33E	gdi32
SelectPalette	-	implicit	-	0x0001A36A	0x0001A36A	gdi32
SelectObject	-	implicit	-	0x0001A35A	0x0001A35A	gdi32
GetTextColor	-	implicit	-	0x0001A34A	0x0001A34A	gdi32
BitBlt	-	implicit	-	0x0001A2F0	0x0001A2F0	gdi32
GetDeviceCaps	-	implicit	-	0x0001A32E	0x0001A32E	gdi32
CreateSolidBrush	-	implicit	-	0x0001A31A	0x0001A31A	gdi32
CreateFontW	-	implicit	-	0x0001A30C	0x0001A30C	gdi32
CreateDIBitmap	-	implicit	-	0x0001A2FA	0x0001A2FA	gdi32
LoadMenuW	-	implicit	-	0x0001A3FC	0x0001A3FC	USER
LoadImageW	-	implicit	-	0x0001A3EE	0x0001A3EE	USER
CreateDialogParamW	-	implicit	-	0x0001A390	0x0001A390	USER
CreateWindowExW	-	implicit	-	0x0001A3A6	0x0001A3A6	USER
DefWindowProcW	-	implicit	-	0x0001A3B8	0x0001A3B8	USER
GetDlgItem	-	implicit	-	0x0001A3CA	0x0001A3CA	USER
IsDlgButtonChecked	-	implicit	-	0x0001A3D8	0x0001A3D8	USER
GetLastError	-	implicit	-	0x0001A45C	0x0001A45C	KERN
GetProcAddress	-	implicit	-	0x0001A492	0x0001A492	KERN
GetModuleHandleA	-	implicit	-	0x0001A47E	0x0001A47E	KERN
GetLocaleInfoW	-	implicit	-	0x0001A46C	0x0001A46C	KERN
FreeLibrary	-	implicit	-	0x0001A414	0x0001A414	KERN
GetFileAttributesW	-	implicit	-	0x0001A446	0x0001A446	KERN
GetCommandLineW	-	implicit	-	0x0001A434	0x0001A434	KERN
GetCommandLineA	-	implicit	-	0x0001A422	0x0001A422	KERN

Gambar 4. 15 Import String API LockBit

Hal tersebut diperkuat dengan hasil analisis terhadap tabel impor pada sampel, seperti pada gambar 4.15. Tidak ditemukan adanya indikasi yang mencurigakan seperti pemanggilan API yang umumnya dilakukan pada aktivitas berbahaya, misalnya VirtualAlloc, WriteFile, atau CreateRemoteThread. Serta seluruh impor library yang hanya berkaitan dengan antar muka grafis pengguna (GUI), seperti fungsi gdi32.dll, user32.dll, dan KERNEL32.dll. Ketidakadaan fungsi-fungsi yang mencurigakan inilah yang patut dicurigai sebagai bagian teknik penyamaran (*obfuscation*). Indikator-indikator tersebut mengarah pada dugaan bahwa file tersebut merupakan *stub* atau bagian kecil dari kode yang berfungsi sebagai media eksekusi pertama kali saat file dijalankan atau *custom loader* yang dirancang untuk mengeksekusi payload utama secara dinamis dan menghindari deteksi statis melalui penyembunyian API berbahaya dari tabel impor.

4.4.1.3 BlackCat

property	value
file	
file > sha256	E160B6348F6FBDC444125BB65DBD94606D99DA48C8334C8E6B2EE4429F813293
file > first 32 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 00
file > first 32 bytes (text)	MZ.....@.....
file > info	size: 7312384 bytes, entropy: 6.670
file > type	executable, 32-bit, GUI
file > version	n/a
file > description	n/a
entry-point > first 32 bytes (hex)	C7 05 28 C2 AC 00 01 00 00 00 E9 B1 FC FF FF 90 C7 05 28 C2 AC 00 00 00 00 00 E9 A1 FC FF FF 90
entry-point > location	0x000014B0 (section[.text])
file > signature	Microsoft Linker 2.38
stamps	
stamp > compiler	Sat Jan 14 11:24:51 2023 (UTC)
stamp > debug	n/a
stamp > resource	n/a
stamp > import	n/a
stamp > export	n/a
names	
file > name	c:\users\layla\downloads\analisis\blackcat.exe
debug > file	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module > name	n/a
certificate > program-name	n/a

Gambar 4. 16 Property File BlackCat

Pada gambar 4.16, merupakan informasi indicator struktur dalam file sampel, yang bila dirincikan seperti pada tabel 4.6 dibawah ini:

Tabel 4. 6 Informasi File BlackCat

Atribut	Nilai
Nama Sampel	blackcat.exe
MD5	8F2B7A45A93EE6F4806918AAA99C1B1B
SHA-256	E160B6348F6FBDC444125B865DBD9406D9 9D4A8C8334C8E682EE4429F813293
Entropy	6.670 (tinggi)
File Type	Executable
Ukuran File	7.312.384 bytes
First Byte Tex	M Z .. .. @ . . . . .
Entry Point	0x000014B0 (section [.text])
Compiler	Microsoft Linker 14.12
Processor	32-bit, GUI

Skor Nilai VirusTotal	58/72

Berdasarkan hasil analisis menggunakan *PeStudio* seperti pada Tabel 4.6, sampel malware blackcat.exe menunjukkan karakteristik sebagai file eksekusi bertipe Portable Executable (PE) dengan arsitektur 32-bit berbasis GUI. File ini memiliki ukuran sebesar 7.31 MB (7312384 bytes) dan tingkat entropi sebesar 6.670. Nilai entropi yang tinggi (>6) mengindikasikan kemungkinan kuat bahwa file telah mengalami teknik *packing* atau *obfuscation*, yang umum digunakan oleh malware untuk menghindari deteksi.

File ini ditandatangani dengan linker Microsoft Linker 2.38, dan memiliki titik masuk (*entry point*) yang berada pada offset 0x000014B0 di dalam section [.text]. Pola *bytecode* awal pada entry point menunjukkan adanya instruksi yang mengarah pada eksekusi langsung, dengan kemungkinan modifikasi nilai memori tertentu yang mencurigakan. Tidak terdapat informasi tambahan seperti *debug*, *import table*, atau *resource* yang terlihat, yang dapat mengindikasikan upaya untuk menghilangkan metadata file demi menyulitkan analisis lebih lanjut

property	value	value	value	value
section	section[0]	section[1]	section[2]	section[3]
name	.text	.data	.rdata	.eh_fram
section > sha256	5375796C3107C727AC2CBA...	13FC76EEFD1A1B2409091D7...	0364FDA10EF9436B4E65D13...	4CDBA9AFC6FC136(
entropy	6.533	2.896	6.692	4.511
file > ratio (99.99%)	70.54 %	0.01 %	17.76 %	8.97 %
raw-address (begin)	0x00000400	0x004EB800	0x004EBC00	0x00628C00
raw-address (end)	0x004EB800	0x004EBC00	0x00628C00	0x006C8E00
raw-size (7311360 bytes)	0x004EB400 (5157888 bytes)	0x00000400 (1024 bytes)	0x0013D000 (1298432 bytes)	0x000A0200 (655872
virtual-address	0x00001000	0x004ED000	0x004EE000	0x0062B000
virtual-size (7308780 bytes)	0x004EB214 (5157396 bytes)	0x00000368 (872 bytes)	0x0013CE50 (1298000 bytes)	0x000A0044 (655428
characteristics	0x60000060	0xC0000040	0x40000040	0x40000040
write	-	x	-	-
execute	x	-	-	-
share	-	-	-	-
self-modifying	-	-	-	-
virtual	-	-	-	-
items				
directory > import	-	-	-	-
directory > relocation	-	-	-	-
directory > thread-local-s...	-	-	0x0062ABBC	-
directory > import-address	-	-	-	-
base-of-code	0x00001000	-	-	-
base-of-data	-	0x004ED000	-	-
entry-point > location	0x000014B0	-	-	-
thread-local-storage	0x00487C50	-	-	-
thread-local-storage	0x004E80B0	-	-	-
thread-local-storage	0x004E8060	-	-	-

Gambar 4. 17 Section File BlackCat

Berdasarkan Gambar 4.17, menunjukkan mengenai struktur section dalam file sampel malware BlackCat yang kompleks dan menunjukkan pola distribusi data yang tidak umum untuk aplikasi benign. File ini terdiri dari sembilan section utama, yaitu: .text, .data, .rdata, .eh_fram, .bss, .idata, .CRT, .tls, dan .reloc.

Section .text memiliki proporsi terbesar dari total file, yaitu sebesar 70.54%, dengan tingkat entropi 6.533. Ini menunjukkan bahwa sebagian besar kode program dieksekusi dari bagian ini. Nilai entropi tersebut berada di atas ambang normal, yang mengindikasikan kemungkinan adanya obfuscation atau penggunaan packer dalam section kode. Section ini memiliki hak eksekusi dan menjadi lokasi dari *entry point* file, yakni di offset 0x000014B0, yang menunjuk pada instruksi awal program.

Section .rdata juga menunjukkan entropi tinggi sebesar 6.692, dan berkontribusi 17.76% terhadap ukuran file. Hal ini mengindikasikan bahwa bagian data yang seharusnya bersifat statis atau konstan justru mengandung data terenkripsi atau dikompresi, yang umum ditemukan pada malware. Sebaliknya, section .data dan .CRT memiliki entropi rendah dan ukuran kecil (0.01%), menandakan bahwa data yang disimpan bersifat tetap atau hanya digunakan untuk keperluan inisialisasi.

Section .reloc memiliki entropi sangat tinggi sebesar 6.602, dan ukuran cukup besar (sekitar 2.56%). Hal ini tidak lazim, karena section ini biasanya digunakan untuk informasi relokasi alamat memori dan tidak memuat data terenkripsi. Hal ini mengindikasikan kemungkinan penyalahgunaan struktur PE oleh malware untuk menyembunyikan payload.

Section lainnya seperti .eh_fram, .bss, .idata, .tls, dan .CRT menunjukkan ukuran yang kecil atau bahkan nol (seperti pada .bss), namun tetap mencurigakan karena beberapa memiliki hak akses write dan execute, yang dapat dieksploitasi untuk teknik injeksi atau penyimpanan payload sementara saat *runtime*.

Lebih lanjut, section .idata (import data) dan .reloc menunjukkan karakteristik tidak lazim, seperti ukuran besar dan entropi tinggi, yang memperkuat dugaan penggunaan teknik *anti-analysis* seperti API obfuscation atau dynamic API resolution. Section .tls (Thread Local Storage), meskipun berukuran sangat kecil (8 bytes), berpotensi digunakan untuk teknik seperti TLS callbacks yang umum dalam malware untuk mengeksekusi kode tersembunyi sebelum main() dijalankan.

callback	index
.text:00487C50	0
.text:004E80B0	1
.text:004E8060	2

Gambar 4. 18 Struktur TLS BlackCat

Berdasarkan hasil analisis struktur *Thread Local Storage* (TLS) seperti pada gambar 4.18, ditemukan bahwa file blackcat.exe memiliki tiga buah TLS callback yang masing-masing berada pada alamat berikut:

- .text:00487C50
- .text:004E80B0
- .text:004E8060

TLS (Thread Local Storage) callback adalah mekanisme pada sistem operasi Windows yang memungkinkan eksekusi fungsi tertentu sebelum eksekusi main() atau WinMain() pada saat thread atau proses pertama kali dijalankan. Dalam konteks malware, TLS callback sering dimanfaatkan untuk menjalankan kode berbahaya lebih awal, sebelum kontrol diberikan kepada alur utama program dan sebelum sebagian besar antivirus atau debugger sempat menginisialisasi proses pemantauan.

Adanya lebih dari satu TLS callback, seperti yang ditunjukkan di atas, memperkuat indikasi bahwa malware ini telah didesain untuk menyembunyikan aktivitas berbahaya pada tahap sangat awal dari eksekusi. Seluruh callback berada di dalam section .text, yang merupakan tempat umum

eksekusi kode, namun dalam kasus ini menunjukkan kemungkinan penyusupan kode abnormal di luar fungsi utama aplikasi.

library (13)	flag (4)	type (1)	imports (288)	description (4)
ntdll.dll	-	Implicit	3	NT Layer
advapi32.dll	-	Implicit	46	Advanced Windows 32 Base API
bcrypt.dll	x	Implicit	1	Windows Cryptographic Primitives Library
kernel32.dll	-	Implicit	142	Windows NT BASE API Client
netapi32.dll	x	Implicit	3	Network Win32 Library
ole32.dll	-	Implicit	3	Microsoft OLE for Windows
rstrtmgr.dll	x	Implicit	4	Restart Manager library
shell32.dll	-	Implicit	1	Windows Shell Library
user32.dll	-	Implicit	6	Multi-User Windows USER API Client Library
userenv.dll	-	Implicit	1	User Environment Library
ws2_32.dll	x	Implicit	25	Windows Socket Library
KERNEL32.dll	-	Implicit	8	Windows NT BASE API Client
msvcrt.dll	-	Implicit	45	Microsoft C Runtime Library

Gambar 4. 19 Import API BlackCat

Berdasarkan Gambar 4.19, merupakan struktur *import address table* (IAT) dari file blackcat.exe menunjukkan bahwa sampel ini mengimpor sejumlah besar fungsi dari berbagai pustaka dinamis (*Dynamic Link Library* / DLL) Windows, dengan total lebih dari 280 fungsi dari 13 pustaka utama.

Beberapa pustaka penting yang terdeteksi antara lain:

Tabel 4. 7 Keterangan Import API BlackCat

Library	Jumlah Fungsi	Deskripsi
Kernel32.dll	142	Windows NT BASE API Client sebagai fungsi sistem dasar
Advapi32.dll	46	Advanced Windows API sebagai keamanan, registry, layanan
Msvcrt.dll	45	Microsoft C Runtime sebagai fungsi bahasa C standar
Ws2_32.dll	25	Windows Socket sebagai komunikasi jaringan
Bcrypt.dll	1	Windows Cryptographic API sebagai enkripsi/kriptografi
Ntdll.dll	3	NT Layer sebagai fungsi tingkat rendah sistem Windows
DLL lainnya	<10 Fungsi	Termasuk shell32.dll, ole32.dll,

		netapi32.dll, dan lainnya.

Analisis lebih mendalam mengenai fungsi-fungsi yang diimpor seperti yang terlihat pada Tabel 4.7, mengungkap beberapa temuan string API penting, diantaranya:

1. Fungsi-fungsi Keamanan dan Token Manipulation

   Beberapa API dari advapi32.dll seperti:

   - OpenProcessToken, AdjustTokenPrivileges, DuplicateTokenEx
   - CreateProcessWithTokenW, LogonUserW

   Fungsi-fungsi tersebut sangat mencurigakan karena memungkinkan malware untuk mengakses dan memanipulasi token keamanan milik proses lain, yang dapat digunakan untuk melakukan privilege escalation atau impersonation attack.
2. Manajemen Layanan dan Registry

   Fungsi seperti OpenSCManagerW, CreateServiceW, DeleteService, RegOpenKeyExW, dan RegSetValueExW, menunjukkan kemampuan malware untuk membuat, mengontrol, atau menghapus layanan sistem serta memodifikasi registry. Hal ini biasa digunakan untuk persistensi.
3. Enkripsi dan Kriptografi

   Fungsi BCryptGenRandom dari bcrypt.dll digunakan untuk menghasilkan bilangan acak secara kriptografik. Fungsi tersebut biasanya dimanfaatkan untuk pembuatan kunci enkripsi, nonce, atau obfuscation data.
4. Proses dan Thread Manipulation

   Fungsi seperti CreateProcessW, OpenProcess, TerminateProcess, GetCurrentProcess, dan ExitProcess, digunakan untuk mengontrol atau membuat proses baru, yang umum digunakan dalam process injection, spawn child, atau malware propagation.
5. I/O dan File System

   Adanya fungsi seperti CreateFileW, ReadFile, WriteFile, DeleteFileW, MoveFileExW, menunjukkan kemampuan malware untuk membaca, memodifikasi, dan menghapus file—fitur utama dari ransomware.

6. Fungsi Console dan Debug

   Adanya fungsi seperti GetConsoleMode, SetConsoleTextAttribute, GetCommandLineW, GetEnvironmentVariableW. Meskipun terlihat biasa, dalam konteks malware, ini bisa dimanfaatkan untuk mendeteksi sandbox atau melakukan anti-analysis berbasis lingkungan.

7. Fungsi Jaringan

   Library ws2_32.dll mengindikasikan kemampuan komunikasi jaringan, yang sangat mungkin digunakan untuk berinteraksi dengan server Command and Control (C2) atau eksfiltrasi data.

4.4.1.4 WhisperGate

property	value
file	
file > sha256	82EF68548D2E2E4B2C41A1C92D06B30B0A433B17B859590989C89D4CE9162033
file > first 32 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 00
file > first 32 bytes (text)	MZ.....@.....
file > info	size: 30354 bytes, entropy: 6.073
file > type	executable, 32-bit, GUI
file > version	n/a
file > description	n/a
entry-point > first 32 bytes (hex)	C4 3C C3 90 83 EC 1C C7 04 24 01 00 00 00 FF 15 14 B2 40 00 E8 BB C3 81 3A DE FF F0 E0 4E B7 8D
entry-point > location	0x00001300 (section[.text])
file > signature	Microsoft Linker 2.28
stamps	
stamp > compiler	Tue Oct 15 10:33:10 2024 (UTC)
stamp > debug	n/a
stamp > resource	n/a
stamp > import	n/a
stamp > export	n/a
names	
file > name	c:\users\layla\downloads\analisis\whispergate.exe
debug > file	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module > name	n/a
certificate > program-name	n/a

Gambar 4. 20 Property File WhisperGate

Pada Gambar 4.20, merupakan informasi indikator struktur dalam file sampel, yang bila dirincikan seperti pada tabel 4.8 dibawah ini

Tabel 4. 8 Informasi File WhisperGate

Atribut	Nilai
Nama Sampel	whispergate.exe
MD5	66BFF87A417DFABF5E5C59A2656A19D

	B
SHA-256	82EF68548D2E2E4B2C41A1C92D06B30B0A433B17B859590989C89D4CE9162033
Entropy	6.073 (tinggi)
File Type	Executable
Ukuran File	30.354 byte
First Byte Tex	M Z .. .. @ . . . . .
Entry Point	0x00001300 (section .text)
Compiler	Microsoft Linker 2.28
Processor	32-bit, GUI
Skor Nilai VirusTotal	42/72

Berdasarkan tabel 4.11, diketahui bahwa file sampel whispergate.exe merupakan sebuah file executable (PE) Windows 32-bit dengan antarmuka grafis (GUI). File ini memiliki ukuran sebesar 30.354 byte dengan tingkat entropi sebesar 6.073, yang mengindikasikan bahwa file tersebut memiliki kemungkinan telah dienkripsi atau dipadatkan (*compressed*), suatu karakteristik yang sering ditemukan dalam malware untuk menghindari deteksi oleh sistem keamanan.

property	value	value	value	value
section	section[0]	section[1]	section[2]	section[3]
name	.text	.data	.rdata	.eh_fram
section > sha256	13B17EBD82772AFE9329135...	206F076CF6D931A22716BCB...	ECD6A0BE58C14DAC6856E1...	F0449F8C63F4
entropy	6.430	5.303	4.586	5.084
file > ratio (92.77%)	60.72 %	1.69 %	5.06 %	15.18 %
raw-address (begin)	0x00000400	0x00004C00	0x00004E00	0x00005400
raw-address (end)	0x00004C00	0x00004E00	0x00005400	0x00006600
raw-size (28160 bytes)	0x00004800 (18432 bytes)	0x00000200 (512 bytes)	0x00000600 (1536 bytes)	0x00001200 (4
virtual-address	0x00001000	0x00006000	0x00007000	0x00008000
virtual-size (26188 bytes)	0x000047A4 (18340 bytes)	0x00000028 (40 bytes)	0x00000494 (1172 bytes)	0x000011A8 (4
characteristics	0x60500060	0xC0300040	0x40300040	0x40300040
write	-	x	-	-
execute	x	-	-	-
share	-	-	-	-
self-modifying	-	-	-	-
virtual	-	-	-	-
items				
directory > import	-	-	-	-
directory > thread-local-storage	-	-	-	-
directory > import-address	-	-	-	-
base-of-code	0x00001000	-	-	-
base-of-data	-	0x00006000	-	-
entry-point > location	0x00001300	-	-	-

Gambar 4. 21 Section File WhisperGate

Berdasarkan hasil analisis terhadap struktur section dari sampel whispergate.exe, seperti pada gambar 4.21, ditemukan bahwa file tersebut terdiri dari empat section utama, yaitu .text, .data, .rdata, dan .eh_farm. Section .text yang umumnya digunakan untuk menyimpan kode instruksi program, memiliki tingkat entropi sebesar 6.430, yang mengindikasikan bahwa adanya kemungkinan enkapsulasi atau penyamaran instruksi melalui teknik *packing* atau *obfuscation* ringan.

Obfuscation ringan atau *mild obfuscation* mengacu pada teknik penyamaran kode atau struktur file yang tidak terlalu kompleks, tetapi cukup untuk menghindari deteksi oleh antivirus berbasis *signature,* mempersulit analisis statis sederhana, hingga menyembunyikan informasi langsung seperti string API, alamat impor, atau lokasi fungsi penting.

directory (3/15)	size (bytes)	address (type)	location	stamp
export	0x00000000	-	-	-
import	0x0000079C (1948)	0x0000B000 (virtual)	section:.idata	0x00000000
resource	0x00000000	-	-	-
exception	0x00000000	-	-	-
security	0x00000000	-	-	-
relocation	0x00000000	-	-	-
debug	0x00000000	-	-	-
architecture	0x00000000	-	-	-
global-pointer	0x00000000	-	-	-
thread-local-storage	0x00000018 (24)	0x0000D004 (virtual)	section:.tls	0x00000000
load-configuration	0x00000000	-	-	-
bound-import	0x00000000	-	-	-
import-address	0x00000118 (280)	0x0000B190 (virtual)	section:.idata	0x00000000
delay-loaded	0x00000000	-	-	-
.NET	0x00000000	-	-	-

Gambar 4. 22 Directories File WhisperGate

Pada bagian directories seperti yang yang terlihat pada Gambar 4.22, sampel tersebut mengindikasikan bahwa hanya tiga direktori yang digunakan secara aktif, diantaranya:

1. *Import Table*

   Berukuran 1946 byte dan berlokasi secara virtual di alamat 0x0000B000 dalam section .idata. Tabel ini berfungsi untuk mencantumkan semua fungsi eksternal seperti DLL yang dibutuhkan oleh program saat dijalankan. Dengan adanya *Import Table,* menunjukkan bahwa file sampel ini tidak bisa berjalan sendiri tanpa bantuan dari file lain, khususnya file sistem seperti DLL (*Dynamic Link Libraries*) di Windows. Artinya, program ini membutuhkan fungsi-fungsi dari luar, misalnya kernel32.dll atau user32.dll agar dapat bekerja dengan baik.
2. *Thread Local Storage* (TLS)

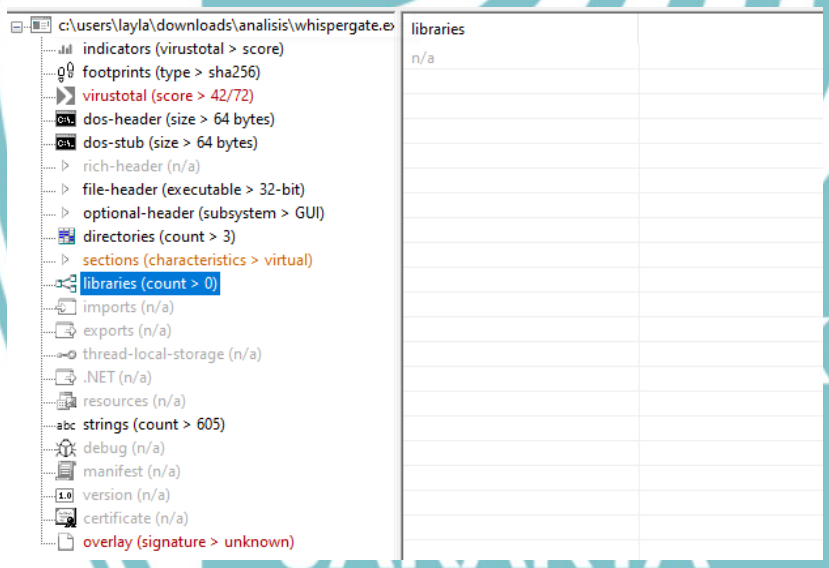
   Direktori ini digunakan ketika program membutuhkan variabel khusus yang hanya berlaku pada satu thread tertentu. Dengan ukuran sebesar 24 byte dan berada di alamat virtual 0x0000D004 dalam section .tls, hal ini dapat mengindikasikan bahwa kemungkinan file ini menggunakan teknik manipulasi thread, yang terkadang digunakan dalam malware untuk menyembunyikan perilaku tertentu atau melakukan *anti-analysis*.
3. *Import Address Table* (IAT)

   Direktori ini berfungsi sebagai pointer ke fungsi yang diimpor. Ukurannya 280 byte dan terletak pada alamat virtual 0x0000B190 juga di dalam section .idata. IAT merupakan bagian penting dalam proses runtime linking,

karena sistem operasi akan mengganti pointer-pointer ini dengan alamat asli fungsi pada saat file dieksekusi.

Direktori lain seperti Export, Resource, Security, dan .Net tidak digunakan sebab memiliki ukuran 0 byte, yang mengindikasikan bahwa file ini tidak mengekspor fungsi yang digunakan oleh file lain, tidak memiliki ikon, string, atau dialog di dalam proses eksekusinya, dan tidak ditandatangani secara digital. Karakteristik tersebut sesuai dengan malware yang dibuat seringan mungkin tanpa banyak metadata yang berguna menghindari deteksi dan meningkatkan kemudahan penyebaran.

Hal tersebut dibuktikan dengan tidak adanya impor library serta string API yang tertera pada Gambar 4.23.



Gambar 4. 23 Libraries Import API WhisperGate

Pada Gambar 4.23, dapat disimpulkan bahwa meskipun file whispergate,exe secara struktural memiliki *import directory*, namun tidak ada informasi terkait library dan import string API yang digunakan. Hal ini membuktikan kuat dugaan indikasi adanya teknik *anti-analysis* atau *obfuscation* yang dilakukan oleh file sampel tersebut.

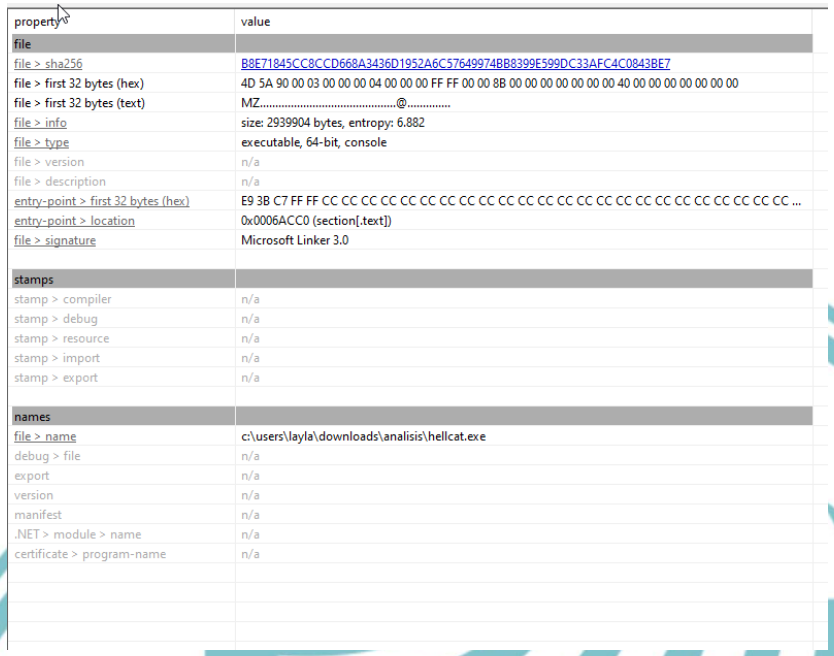
property	value
overlay > sha256	C08F6B5C7FB55081D4CCC2738A1D2C122EB0204226EB10F398A4A1D8309A0...
entropy	0.429
overlay > location	0x00007200 - 0x00007692
size	0x0492 (1170 bytes)
signature	unknown
first 32 bytes (hex)	00 00 53 48 45 4C 4C 33 32 2E 44 4C 4C 00 50 B0 00 00 50 B0 00 00 50 B0 00 00...
first 32 bytes (text)	....SHELL32.DLL..P.....P.....P.....P..
file > ratio	3.85 %

Gambar 4. 24 Overlay WhisperGate

Pada Gambar 4.24, merupakan overlay dari file sampel whispergate.exe. Overlay merupakan bagian dari sebuah file executable (PE) yang berada di luar struktur section standar atau dalam kata lain tidak terdaftar dalam section tabel. Bagian ini tidak dimuat ke dalam memori saat proses dijalankan namun masih bisa diakses sebagai data dari file disk. Overlay tersebut diketahui berukuran sebesar 1170 byte, yang berada pada offset 0x00007200 hingga 0x00007692 dan memiliki entropi sebesar 0.429, yang tergolong rendah. Hal ini menunjukkan bahwa data di dalamnya tidak mengalami proses enkripsi atau pun kompresi. Terlihat hasil pemeriksaan terhadap isi dari overlay memperlihatkan adanya string “SHELL32.DLL” beserta karakter lainnya yang mengindikasikan bahwa bagian ini menyimpan nama dari library (DLL) atau referensi API secara tersembunyi.

Hal ini menunjukkan adanya kemungkinan penerapan teknik penyembunyian API secara manual (*Manual API Resolution*), di mana fungsi-fungsi tidak diekspor melalui import table, tetapi dimuat secara dinamis saat program dijalankan. Selain itu, status signature yang tidak dikenali (unknown) menandakan bahwa overlay tersebut bukan bagian dari digital signature resmi. Dengan demikian, adanya overlay merupakan indikasi dari adanya *obfuscation* ringan yang bertujuan untuk menyulitkan analisis statis dan menghindari deteksi berbasis signature konvensional.

4.4.1.5 Hellcat



property	value
file	
file > sha256	B8E71845CC8CCD668A3436D1952A6C57649974BB8399E599DC33AFC4C0843BE7
file > first 32 bytes (hex)	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 8B 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
file > first 32 bytes (text)	MZ.....@.....
file > info	size: 2939904 bytes, entropy: 6.882
file > type	executable, 64-bit, console
file > version	n/a
file > description	n/a
entry-point > first 32 bytes (hex)	E9 3B C7 FF FF CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC ...
entry-point > location	0x0006ACC0 (section[.text])
file > signature	Microsoft Linker 3.0
stamps	
stamp > compiler	n/a
stamp > debug	n/a
stamp > resource	n/a
stamp > import	n/a
stamp > export	n/a
names	
file > name	c:\users\layla\downloads\analisis\hellcat.exe
debug > file	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module > name	n/a
certificate > program-name	n/a

Gambar 4. 25 Property File HellCat

Pada Gambar 4.25, merupakan informasi indicator struktur dalam file sampel, yang bila dirincikan seperti pada Tabel 4.9 dibawah ini

Tabel 4. 9 Informasi File HellCat

Atribut	Nilai
Nama Sampel	hellcat.exe
MD5	03C2034D91589B84D2F95B5E9408AEDB
SHA-256	B8E71845CC8CCD668A3436D1952A6C57649974BB8399E599DC33AFC4C0843BE7
Entropy	6.882 (tinggi)
File Type	Executable

Ukuran File	2.939.904 byte
First Byte Tex	M Z .. .. @ . . . . .
Entry Point	0x0006ACC0 (section .text)
Compiler	Microsoft Linker 3.0
Processor	64-bit, GUI
Skor Nilai VirusTotal	47/72

Berdasarkan tabel 4.12, file sampel hellcat.exe berukuran sekita 2.9 MB dengan nilai entropi sebesar 6.882. Nilai tersebut tergolong tinggi, yang menunjukkan bahwa file tersebut melakukan teknik penyamaran atau *obfuscation*. File tersebut ditujukan untuk dijalankan pada sistem Windows 64-bit. Hal ini memungkinkan pemrosesan data yang lebih besar dan penggunaan memori di atas 4 GB. Selain itu, file ini juga berjalan melalui Command Line Interface (CLI), bukan melalui antarmuka grafis (GUI). Program dengan jenis ini biasanya digunakan oleh admin sistem, *script automation*, atau dalam konteks malware untuk menghindari deteksi pengguna karena tidak menampilkan sesuatu pada antarmuka.

Selain itu, entry point berada di offset 0x0006ACC0 dalam section .text. Entry point dimulai dengan instruksi E9 (*jump*) , yang menunjukkan *redirect code execution*, sebuah teknik yang digunakan dalam malware untuk melompat ke alamat lain, bukan langsung mengeksekusi instruksi biasa. Teknik ini digunakan malware untuk menyembunyikan *payload* utama, melakukan *unpacking* di memori, dan mengelabui proses analisis statis.

directory (4/15)	size (bytes)	address (type)	location	stamp
export	0x00000000	-	-	-
import	0x00000554 (1364)	0x0033A000 (virtual)	section:.idata	0x00000000
resource	0x00000000	-	-	-
exception	0x00005D54 (23892)	0x00264000 (virtual)	section:.pdata	0x00000000
security	0x00000000	-	-	-
relocation	0x0000429C (17052)	0x0033B000 (virtual)	section:.reloc	0x00000000
debug	0x00000000	-	-	-
architecture	0x00000000	-	-	-
global-pointer	0x00000000	-	-	-
thread-local-storage	0x00000000	-	-	-
load-configuration	0x00000000	-	-	-
bound-import	0x00000000	-	-	-
import-address	0x00000180 (384)	0x001CB140 (virtual)	section:.data	0x00000000
delay-loaded	0x00000000	-	-	-
.NET	0x00000000	-	-	-

Gambar 4. 26 Directory HellCat

Berdasarkan Gambar 4.26, merupakan *Directory table* pada sampel ransomware HellCat. Dari 15 entri hanya 4 entri yang aktif atau terisi, yaitu:

1. Import Directory
   Ukuran sebesar 0x554 (1364 bytes), terletak secara virtual di 0x0033A000, dan dipetakan dalam section:.idata. Direktori ini menunjuk ke tabel fungsi-fungsi API yang diimpor dari library eksternal, menunjukkan ketergantungan malware terhadap fungsi Windows API.
2. Exception Directory
   Dengan ukuran 0x5D54 (23892 bytes), berada di 0x00264000 dan di section:.pdata. Hal ini menunjukkan bahwa malware menggunakan struktur exception handler secara kompleks, yang bisa menjadi bagian dari teknik anti-debugging atau pengacakan alur eksekusi.
3. Relocation Directory
   Memiliki ukuran 0x429C (17052 bytes), berlokasi di 0x0033B000 dalam section:.reloc. Keberadaan direktori ini menunjukkan bahwa file PE dapat dimuat pada alamat memori yang berbeda, mendukung eksekusi dinamis yang fleksibel, menjadikan karakteristik yang umum pada malware yang ingin menghindari deteksi.
4. Import Address Directory
   Berukuran 0x180 (384 bytes), diarahkan ke 0x001CB140 dan berada dalam section:.data. IAT ini berperan penting dalam penautan fungsi saat runtime,

dan dapat dimanipulasi untuk melakukan teknik hooking atau API redirection.

imports (47)	flag (13)	type (1)	ordinal (0)	first-thunk (IAT)	first-thunk-original (INT)	library (
SetConsoleCtrlHandler	x	implicit	-	0x00000000	0x00000000	kernel32
RtlVirtualUnwind	-	implicit	-	0x00000000	0x00000000	kernel32
RtlLookupFunctionEntry	-	implicit	-	0x00000000	0x00000000	kernel32
ResumeThread	-	implicit	-	0x00000000	0x00000000	kernel32
RaiseFailFastException	-	implicit	-	0x00000000	0x00000000	kernel32
PostQueuedCompletionStatus	-	implicit	-	0x00000000	0x00000000	kernel32
LoadLibraryW	-	implicit	-	0x00000000	0x00000000	kernel32
LoadLibraryExW	-	implicit	-	0x00000000	0x00000000	kernel32
SetThreadContext	x	implicit	-	0x00000000	0x00000000	kernel32
GetThreadContext	x	implicit	-	0x00000000	0x00000000	kernel32
GetSystemInfo	-	implicit	-	0x00000000	0x00000000	kernel32
GetSystemDirectoryA	-	implicit	-	0x00000000	0x00000000	kernel32
GetStdHandle	-	implicit	-	0x00000000	0x00000000	kernel32
GetQueuedCompletionStatu...	-	implicit	-	0x00000000	0x00000000	kernel32
GetProcessAffinityMask	-	implicit	-	0x00000000	0x00000000	kernel32
GetProcAddress	-	implicit	-	0x00000000	0x00000000	kernel32
GetErrorMode	-	implicit	-	0x00000000	0x00000000	kernel32
GetEnvironmentStringsW	x	implicit	-	0x00000000	0x00000000	kernel32
GetCurrentThreadId	x	implicit	-	0x00000000	0x00000000	kernel32
GetConsoleMode	-	implicit	-	0x00000000	0x00000000	kernel32
FreeEnvironmentStringsW	-	implicit	-	0x00000000	0x00000000	kernel32
ExitProcess	-	implicit	-	0x00000000	0x00000000	kernel32
DuplicateHandle	-	implicit	-	0x00000000	0x00000000	kernel32
CreateWaitableTimerExW	-	implicit	-	0x00000000	0x00000000	kernel32
CreateThread	-	implicit	-	0x00000000	0x00000000	kernel32
CreateIoCompletionPort	-	implicit	-	0x00000000	0x00000000	kernel32
CreateFileA	-	implicit	-	0x00000000	0x00000000	kernel32
CreateEventA	-	implicit	-	0x00000000	0x00000000	kernel32
CloseHandle	-	implicit	-	0x00000000	0x00000000	kernel32
AddVectoredExceptionHandler	x	implicit	-	0x00000000	0x00000000	kernel32
AddVectoredContinueHandler	x	implicit	-	0x00000000	0x00000000	kernel32

Gambar 4. 27 Import String API HellCat

Berdasarkan Gambar 4.27, merupakan import string API yang digunakan oleh ransomware HellCat. Terdapat total 47 fungsi API yang diimpor, semuanya berasal dari library kernel32.dll. Namun, terdapat 13 diantaranya ditandai dengan flag merah (X) oleh PEStudio, yang mengindikasikan adanya potensi penyalahgunaan fungsi tersebut. Beberapa fungsi yang mencurigakan meliputi:

1. Manupulasi Thread dan Proses

   String seperti CreateThread, ResumeThread, SetThreadContext, GetThreadContext, ExitProcess, GetCurrentThreadId. Fungsi-fungsi ini lazim digunakan untuk membuat dan mengelola thread secara langsung, sebuah pendekatan umum dalam teknik anti-debugging atau pemuatan kode berbahaya secara dinamis.
2. Handling environment

   String seperti GetEnvironmentStringsW dan FreeEnvironmentStringsW digunakan untuk mendapatkan informasi lingkungan proses, seperti

variabel environment, yang dapat digunakan malware untuk mengadaptasi perilakunya tergantung pada konteks sistem.

3. Memory dan Exception handling

   String AddVectoredExceptionHandler, AddVectoredContinueHandler, RtlVirtualUnwind. String-string ini biasanya digunakan untuk manipulasi alur eksekusi dan mendeteksi breakpoint debugger, serta sering digunakan dalam teknik obfuscation atau anti-debugging tingkat lanjut.

4. Interaksi Sistem dan File

   String CreateFileA, CloseHandle, GetSystemInfo, CreateEventA. Fungsi ini sering dipakai dalam ransomware untuk mengakses dan mengenkripsi file korban.

4.4.1.6 Morpheus

property	value
file	
file > sha256	4B2EDADC8F90E9FCC976F02A9EDA1640CD92C07718C0271842FBD4CA7E2906E2
file > first 32 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 00
file > first 32 bytes (text)	MZ.....@.....
file > info	size: 18434 bytes, entropy: 5.220
file > type	executable, 64-bit, GUI
file > version	n/a
file > description	n/a
entry-point > first 32 bytes (hex)	48 81 EC E8 05 00 00 FF 15 3B 11 00 00 48 8D 94 24 D0 00 00 00 48 8B C8 FF 15 02 12 00 00 48 89
entry-point > location	0x00002EF0 (section[.rdata])
file > signature	Microsoft Linker 14.39
stamps	
stamp > compiler	Mon Feb 17 10:11:19 2025 (UTC)
stamp > debug	n/a
stamp > resource	n/a
stamp > import	n/a
stamp > export	n/a
names	
file > name	c:\users\layla\downloads\analisis\morphues.exe
debug > file	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module > name	n/a
certificate > program-name	n/a

Gambar 4. 28 Property File Morpheus

Pada Gambar 4.28, merupakan informasi indicator struktur dalam file sampel, yang bila dirincikan seperti pada tabel 4.10 dibawah ini

Tabel 4. 10 Informasi File Morpheus

Atribut	Nilai
Nama Sampel	Morpheus.exe

MD5	F6536E4E60AB984ED1DC616DD7D673B18A08BA90
SHA-256	4B2EDADC8F90E9FCC976F02A9EDA1640CD9C07718C0271842FBD4CA7E2906E2
Entropy	5.220 (sedang)
File Type	Executable
Ukuran File	18.434 byte
First Byte Tex	M Z .. .. @ . . . . .
Entry Point	0x00002EF0 (section .rdata)
Compiler	Microsoft Linker 14.39
Processor	64-bit, GUI
Skor Nilai VirusTotal	57/71

Berdasarkan tabel 4.10, dapat diketahui bahwa file sampel Morpheus merupakan sebuah file executable 64-bit yang dirancang untuk berinteraksi melalui antarmuka berbasis GUI (*Graphical User Interface*). File berukuran 18.434 byte dengan tingkat entropi sebeser 5.220, mengindikasikan bahwa file tersebut tidak sepenuhnya terkompresi maupun terenkripsi. Tingkat entropi yang cenderung normal, namun tetap memungkinkan terdapat bagian terenkripsi (*obfuscated*).

Informasi pada bagian entry point mengindikasikan bahwa eksekusi program akan dimulai dari section .rdata, pada offset 0x00002EF0. Hal tersebut tidak

lazim, sebab section .rdata umumnya berisi data yang bersifat *read-only*, bukan instruksi yang dapat dieksekusi. Temuan ini menunjukkan adanya anomali atau teknik evasive, seperti *code injection* atau penyamaran struktur PE.

library (3)	flag (1)	type (1)	imports (44)	description (1)
KERNEL32.dll	-	Implicit	33	Windows NT BASE API Client
SHELL32.dll	-	Implicit	2	Windows Shell Library
bcrypt.dll	x	Implicit	9	Windows Cryptographic Primitives Library

Gambar 4. 29 Import API Morpheus

Berdasarkan Gambar 4.29, terlihat bahwa file sampel Morpheus.exe melakukan pemanggilan terhadap tiga library utama Windows, yakni KERNEL32.dll, SHELL32.dll, dan bcrypt.dll. Library KERNEL32.dll dan SHELL32.dll merupakan library yang umum digunakan oleh hampir semua aplikasi Windows. KERNEL32.dll digunakan dalam menangani fungsi inti sistem operasi, seperti manajemen memori, proses, dan file. Sementara SHELL32.dll berperan dalam menunjang proses interaksi aplikasi dengan antarmuka shell pada Windows, seperti pengeksekusian file atau menampilkan GUI.

Namun, keberadaan bcrypt.dll yang digunakan untuk menjalankan operasi kriptografi tingkat rendah, termasuk *hashing* dan enkripsi data. Hal tersebut diindikasikan upaya dalam menyembunyikan atau melindungi payload, komunikasi terenkripsi, atau menyamarkan data berbahaya

imports (44)	flag (14)	type (1)	ordinal (0)	first-thunk (IAT)	first-thunk-original (INT)	libr
InitializeCriticalSection	-	implicit	-	0x00000000	0x00000000	KER
SetErrorMode	-	implicit	-	0x00000000	0x00000000	KER
lstrlenA	-	implicit	-	0x00000000	0x00000000	KER
FindClose	-	implicit	-	0x00000000	0x00000000	KER
WaitForSingleObject	-	implicit	-	0x00000000	0x00000000	KER
CreateFileW	-	implicit	-	0x00000000	0x00000000	KER
lstrcatA	-	implicit	-	0x00000000	0x00000000	KER
ExitThread	-	implicit	-	0x00000000	0x00000000	KER
Sleep	-	implicit	-	0x00000000	0x00000000	KER
GetCurrentThread	x	implicit	-	0x00000000	0x00000000	KER
lstrcpyA	-	implicit	-	0x00000000	0x00000000	KER
CloseHandle	-	implicit	-	0x00000000	0x00000000	KER
GetSystemInfo	-	implicit	-	0x00000000	0x00000000	KER
CreateThread	-	implicit	-	0x00000000	0x00000000	KER
HeapAlloc	-	implicit	-	0x00000000	0x00000000	KER
SetFilePointerEx	-	implicit	-	0x00000000	0x00000000	KER
DeleteCriticalSection	-	implicit	-	0x00000000	0x00000000	KER
ExitProcess	-	implicit	-	0x00000000	0x00000000	KER
GetProcessHeap	-	implicit	-	0x00000000	0x00000000	KER
GetModuleHandleW	-	implicit	-	0x00000000	0x00000000	KER
CommandLineToArgvW	-	implicit	-	0x00000000	0x00000000	SHE
ShellExecuteW	x	implicit	-	0x00000000	0x00000000	SHE
BCryptEncrypt	x	implicit	-	0x00000000	0x00000000	bcr
BCryptDestroyKey	x	implicit	-	0x00000000	0x00000000	bcr
BCryptGetProperty	x	implicit	-	0x00000000	0x00000000	bcr
BCryptOpenAlgorithmProvider	x	implicit	-	0x00000000	0x00000000	bcr
BCryptGenRandom	x	implicit	-	0x00000000	0x00000000	bcr
BCryptCloseAlgorithmProvider	x	implicit	-	0x00000000	0x00000000	bcr
BCryptImportKeyPair	x	implicit	-	0x00000000	0x00000000	bcr
BCryptSetProperty	x	implicit	-	0x00000000	0x00000000	bcr
BCryptGenerateSymmetricKey	x	implicit	-	0x00000000	0x00000000	bcr

Gambar 4. 30 Import String API Morpheus

Berdasarkan hasil analisis terhadap *API Import* pada fila sampel Morpheus.exe yang terlihat pada Gambar 4.30, menunjukkan adanya aktivitas yang mencurigakan melalui pemanggilan beberapa fungsi krusial yang bersumber dari bcrypt.dll. Fungsi-fungsi seperti BCryptEncrypt, BCryprGenerateSymmetricKey, dan BCryptGenRandom merupakan indikator kuat bahwa program ini meliliki kemampuan enkripsi tingkat lanjut.

Fungsi BCryptEncrypt dan BCryprGenerateSymmetricKey mengindikasikan bahwa file ini mengenkripsi data secara ketika sedang dijalankan oleh sistem dan seluruh proses enkripsinya dilakukan langsung di memori (RAM). Hal ini merupakan proses umum yang dilakukan oleh ransomware yang mengenkripsi file korban atau berkomunikasi data dengan server C2 (*Command and Control*). Serta penggunaan BCryptGenRandom yang digunakan untuk pembuatan kunci enkripsi atau nonce (*number used once*), sebuah teknik untuk menjaga keamanan proses enkripsi agar sulit dianalisis secara statis.

Selain itu, fungsi CreateThread dan ShellExecuteW digunakan untuk *code injection* atau pengeksekusian file lain secara tersembunyi, dua jenis teknik yang digunakan malware dalam mengelabui sistem keamanan.

4.4.1.7 Prince

property	value
file	
file > sha256	B7A6EB4DC4C644FE3611B0F4E69202455D61CF426726343B2FFB182C3DEB6CB8
file > first 32 bytes (hex)	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 8B 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
file > first 32 bytes (text)	MZ.....@.....
file > info	size: 2672128 bytes, entropy: 6.257
file > type	executable, 64-bit, GUI
file > version	n/a
file > description	n/a
entry-point > first 32 bytes (hex)	E9 DB C6 FF FF CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC ...
entry-point > location	0x00073960 (section[.text])
file > signature	Microsoft Linker 3.0
stamps	
stamp > compiler	n/a
stamp > debug	n/a
stamp > resource	n/a
stamp > import	n/a
stamp > export	n/a
names	
file > name	c:\users\layla\downloads\prince-ransomware-main\prince-ransomware-main\prince-built.exe
debug > file	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module > name	n/a
certificate > program-name	n/a

Gambar 4. 31 Property File Prince

Pada Gambar 4.31, merupakan informasi indicator struktur dalam file sampel, yang bila dirincikan seperti pada tabel 4.11 dibawah ini

Tabel 4. 11 Informasi File Prince

Atribut	Nilai
Nama Sampel	Prince-Built.exe
MD5	F6536E4E60AB984ED1DC616DD7D673B18A08BA90
SHA-256	B7A6EB4DC4C644FE361180F4E69202455D61CF426726343B2FFB182C3DEB6C8B
Entropy	6.257 (tinggi)
File Type	Executable
Ukuran File	267.128 byte

First Byte Tex	M Z .. .. @ . . . . .
Byte Hex	E9 DB C6 FF CC CC CC …
Entry Point	0x00073960 (section .text)
Compiler	Microsoft Linker 3.0
Processor	64-bit, GUI
Skor Nilai VirusTotal	52/72

Berdasarkan tabel 4.11, diketahui bahwa file sampel Prince-Built.exe teridentifikasi sebagai file eksekusi Windows tipe 64-bit dengan antarmuka grafis (GUI). File berukuran 267.128 byte dengan nilai entropi sebesar 6,257. Nilai entropi yang tinggi mengindikasikan adanya kemungkinan bahwa file telah mengalami proses kompresi, enkripsi, atau penggunaan packer untuk menyembunyikan struktur internalnya.

Hasil analisis juga menunjukkan bahwa entry point file berada pada offset 0x0073960 yang terletak pada section .text. Byte awal pada entry point menunjukkan adanya instruksi E9 (*relative jump*) yang diikuti oleh deretan karakter CC. Dalam arsitektur x86/x64, byte 0xCC merepresentasikan instruksi INT3 (*Interrupt 3*), yang digunakan oleh *debugger* untuk menetapkan *breakpoint* atau titik henti eksekusi program. Saat program mencapai instruksi ini dalam kondisi yang sedang dijalankan melalui *debugger*. maka akan terjadi interrupt yang menghentikan eksekusi.

Adanya instruksi INT3 secara berulang dan dalam jumlah yang signifikan, terutama pada bagian awal kode (entry point), merupakan indikasi kuat terhadap penggunaan teknik anti-debugging. Teknik yang digunakan oleh pengembang malware untuk menyulitkan proses analisis dengan cara

mendeteksi keberadaaan debugger dan memicu *crash* atau perubahan perilaku program jika dijalankan dalam lingkungan analisis.

Selain sebagai teknik anti-analysis, deretan karakter CC juga sering ditemukan sebagai bagian sebagai bagian dari proses *padding* atau *placeholder* yang digunakan oleh beberapa *compiler* atau *packer.* Keberadaan banyaknya karakter CC pada entry point, terutama setelah instruksi jump, memperkuat dugaan bahwa file sampel ini menggunakan metode proteksi seperti *obfuscation* atau *packing,* yang bertujuan untuk menyembunyikan kode utama dari proses analisis statis.

property	value	value	value	value
section	section[0]	section[1]	section[2]	section[3]
name	.text	.rdata	.data	.pdata
section > sha256	993A846FAA54DC1DB87B6F...	3A28E90007F63DE2C4179F3...	5EDC88F65D27F21570F5891...	4DCA0DC4B35D1CEE65A10...
entropy	6.221	5.464	6.790	5.273
file > ratio (99.94%)	43.95 %	47.92 %	6.13 %	1.07 %
raw-address (begin)	0x00000600	0x0011F200	0x00257C00	0x0027FC00
raw-address (end)	0x0011F200	0x00257C00	0x0027FC00	0x00286C00
raw-size (2670592 bytes)	0x0011EC00 (1174528 bytes)	0x00138A00 (1280512 bytes)	0x00028000 (163840 bytes)	0x00007000 (28672 bytes)
virtual-address	0x00001000	0x00120000	0x00259000	0x002CB000
virtual-size (2970719 byt...	0x0011EBB1 (1174449 bytes)	0x001388A8 (1280168 bytes)	0x00071C40 (465984 bytes)	0x00006E28 (28200 bytes)
characteristics	0x60000020	0x40000040	0xC0000040	0x40000040
write	-	-	x	-
execute	x	-	-	-
share	-	-	-	-
self-modifying	-	-	-	-
virtual	-	-	-	-
items				
directory > import	-	-	-	-
directory > exception	-	-	-	0x002CB000
directory > relocation	-	-	-	-
directory > import-addr...	-	-	0x002594C0	-
base-of-code	0x00001000	-	-	-
entry-point > location	0x00073960	-	-	-

Gambar 4. 32 Section Prince

Pada Gambar 4.32, merupakan hasil analisis file sampel pada bagian section. Terlihat bahwa besarnya dominasi antara section .text sebesar 43,95% dan .rdata sebesar 47,92% dari keseluruhan ukuran file. Dengan besarnya porsi ukuran file tersebut menunjukkan adanya teknik *packing* atau *obfuscation*, di mana sebagian besar data program dikemas dan disembunyikan di dalam section .rdata.

Serta tingginya entropi pada section .data sebesar 6.790, yang tergolong tinggi. Hal ini memiliki kemungkinan besar karena file dienkripsi atau dikompresi. Dengan kata lain bahwa ransomware tersebut menyembunyikan konfigurasi

atau payloadnya dalam bentuk terenkripsi dan hanya akan didekripsi saat dijalankan (*runtime*).

imports (46)	flag (13)	type (1)	ordinal (0)	first-thunk (IAT)	first-thunk-original (INT)	library (
WriteFile	x	implicit	-	0x00000000	0x00000000	kernel32
WriteConsoleW	-	implicit	-	0x00000000	0x00000000	kernel32
WerSetFlags	x	implicit	-	0x00000000	0x00000000	kernel32
WerGetFlags	-	implicit	-	0x00000000	0x00000000	kernel32
WaitForMultipleObjects	-	implicit	-	0x00000000	0x00000000	kernel32
WaitForSingleObject	-	implicit	-	0x00000000	0x00000000	kernel32
VirtualQuery	x	implicit	-	0x00000000	0x00000000	kernel32
VirtualFree	-	implicit	-	0x00000000	0x00000000	kernel32
VirtualAlloc	x	implicit	-	0x00000000	0x00000000	kernel32
TlsAlloc	-	implicit	-	0x00000000	0x00000000	kernel32
SwitchToThread	x	implicit	-	0x00000000	0x00000000	kernel32
SuspendThread	x	implicit	-	0x00000000	0x00000000	kernel32
SetWaitableTimer	-	implicit	-	0x00000000	0x00000000	kernel32
SetProcessPriorityBoost	-	implicit	-	0x00000000	0x00000000	kernel32
SetEvent	-	implicit	-	0x00000000	0x00000000	kernel32
SetErrorMode	-	implicit	-	0x00000000	0x00000000	kernel32
SetConsoleCtrlHandler	x	implicit	-	0x00000000	0x00000000	kernel32
RtlVirtualUnwind	-	implicit	-	0x00000000	0x00000000	kernel32
RtlLookupFunctionEntry	-	implicit	-	0x00000000	0x00000000	kernel32
ResumeThread	-	implicit	-	0x00000000	0x00000000	kernel32
RaiseFailFastException	-	implicit	-	0x00000000	0x00000000	kernel32
PostQueuedCompletionStatus	-	implicit	-	0x00000000	0x00000000	kernel32
LoadLibraryW	-	implicit	-	0x00000000	0x00000000	kernel32
LoadLibraryExW	-	implicit	-	0x00000000	0x00000000	kernel32
SetThreadContext	x	implicit	-	0x00000000	0x00000000	kernel32
GetThreadContext	x	implicit	-	0x00000000	0x00000000	kernel32
GetSystemInfo	-	implicit	-	0x00000000	0x00000000	kernel32
GetSystemDirectoryA	-	implicit	-	0x00000000	0x00000000	kernel32
GetStdHandle	-	implicit	-	0x00000000	0x00000000	kernel32
GetQueuedCompletionStatu...	-	implicit	-	0x00000000	0x00000000	kernel32
GetProcessAffinityMask	-	implicit	-	0x00000000	0x00000000	kernel32

Gambar 4. 33 Import Sting API Prince

Hasil analisis terhadap fungsi-fungsi API yang diimpor oleh sampel ransomware prince-built.exe seperti pada Gambar 4.33. Berdasarkan data dari PeStudio, terdapat 46 fungsi API yang diimpor secara eksplisit dari library Windows kernel32.dll. Seluruh fungsi tersebut berperan penting dalam manajemen memori, eksekusi proses, dan manipulasi thread.

Beberapa fungsi seperti VirtualAlloc, VirtualFree, dan VirtualQuery memberikan indikasi bahwa program ini melakukan alokasi memori secara dinamis, suatu teknik yang digunakan oleh malware untuk melakukan unpacking kode atau injeksi payload ke dalam proses yang berjalan. Selain itu, kehadiran WriteFile mengindikasikan adanya aktivitas tulis file, di mana file korban ditimpa atau digantikan oleh versi yang telah dienkripsi.

Fungsi-fungsi manipulasi thread seperti SuspendThread, ResumeThread, SetThreadContext, dan GetThreadContext menunjukkan bahwa ransomware ini kemungkinan besar menerapkan teknik injeksi proses atau process hollowing, di mana kode berbahaya disisipkan ke dalam proses yang sah.

Teknik ini memungkinkan malware menyembunyikan aktivitasnya dari sistem deteksi yang berbasis perilaku.

Selain itu, keberadaan fungsi seperti SetConsoleCtrlHandler dan RaiseFailFastException mengindikasikan upaya dari malware untuk mengontrol proses penghentian aplikasi dan mengganggu proses debugging. Fungsi LoadLibraryW dan GetProcAddress juga memungkinkan program untuk melakukan resolusi API secara dinamis saat runtime, yang merupakan teknik penghindaran deteksi yang umum digunakan oleh malware.

Menariknya, file ini tidak mengimpor fungsi-fungsi yang berkaitan langsung dengan komunikasi jaringan, seperti InternetOpenUrl, WinHttpConnect, atau socket. Hal ini mengindikasikan bahwa ransomware kemungkinan beroperasi secara lokal atau menggunakan teknik alternatif dalam menghubungi server *command and control* (C2), seperti pemanggilan API secara dinamis atau pemanfaatan komponen pihak ketiga.

Secara keseluruhan, temuan ini mendukung dugaan bahwa prince-built.exe adalah varian ransomware dengan kemampuan injeksi proses dan eksekusi multithreaded, serta memanfaatkan teknik anti-analisis untuk menghindari deteksi dan mitigasi.

4.4.2 Hasil Analisis Statis dengan Ekstraksi Strings

4.4.2.1 Stings Akira

Proses ekstraksi string pada sampel ransomware Akira dilakukan menggunakan perintah strings untuk mengidentifikasi string ASCII dan Unicode yang terdapat di dalam file eksekusi. Hasil dari ekstraksi string menunjukkan sejumlah pola dan artefak yang mencerminkan fungsi-fungsi internal ransomware ini.

Tabel 4. 12 String Akira.exe

Kategori	Strings	Analisis
Kriptografi	AES256,	Menunjukkan penggunaan

	HMACSHA256, Encrypt, Decrypt, CryptoStream, SetKey	algoritma kriptografi untuk mengenkripsi file korban.
Anti Analisis	PatchETW, PatchMem, Anti_Analysis, DynamicAPIInvoke, UnhookWindowsHookEx	Teknik penghindaran monitoring dan debugging oleh analis atau antivirus.
Keylogger & HVNC	Keylogger, KeylogConfFile, LowLevelKeyboardProc, HVNC_REPLY_MESSAGE	Indikasi kemampuan keylogging dan kontrol jarak jauh secara tersembunyi. Manajemen Proses
Manajemen Proses	OpenProcess, TerminateProcess, PROCESSENTRY32, CloseHandle, GetProcessById	Digunakan untuk penghentian proses target, seperti antivirus atau backup.
.NET Framework	System.IO, System.Threading, Microsoft.VisualBasic, .NETFramework,Version =v4.0	Malware dibangun dengan bahasa C# dan platform .NET, memanfaatkan library Window
Konfigurasi XML	<assemblyIdentity>, <requestedExecutionLevel level="asInvoker"/>	Manifest file menunjuk ke nama samaran dan hak akses eksekusi minimal.
Identitas Client	Client.exe, Client.Install, Install_Folder	Sebagai indikasi file yang dikemas seperti aplikasi biasa agar tidak mencurigakan.
Sistem Informasi	GetRAM, GetGPU, IsServerOS, GetFileName, HWID	Digunakan untuk mengidentifikasi sistem target secara unik.

Pada tabel 4.12, merupakan hasil analisis yang menunjukkan bahwa ransomware Akira dibangun menggunakan bahasa pemrograman C# dan berbasis pada platform .NET Framework versi 4.0 (v4.0.30319). Hal ini terlihat dari kemunculan banyak namespace seperti System.IO, System.Threading, System.Security.Cryptography, serta deklarasi XML manifest .NET di bagian akhir hasil ekstraksi.

Selain itu adanya keberadaan string-string yang berkaitan dengan kriptografi seperti AES256, HMACSHA256, SHA256Managed, Encrypt, dan Decrypt. Hal ini mengindikasikan bahwa ransomware ini menggunakan algoritma kriptografi tingkat lanjut untuk mengenkripsi data korban dan menjaga integritas file yang disandera. Selain itu, string seperti SetKey, KeySize, dan CryptoStream juga memperkuat temuan bahwa proses enkripsi dilakukan secara tersusun dan terprogram.

Dalam konteks anti-analisis, Akira mengandung banyak string yang menunjukkan penggunaan teknik penghindaran deteksi, seperti PatchETW, PatchMem, Anti_Analysis, UnhookWindowsHookEx, SetWindowsHookEx, dan DynamicAPIInvoke. String ini mengindikasikan bahwa malware memiliki kemampuan untuk menonaktifkan Event Tracing for Windows (ETW), menghindari deteksi dari hook API, serta memanggil fungsi API secara dinamis untuk menyulitkan proses analisis statis.

Lebih lanjut, ditemukan string-string seperti Keylogger, KeylogConfFile, LowLevelKeyboardProc, keylogparam, serta HVNC_REPLY_MESSAGE, yang menunjukkan bahwa ransomware ini tidak hanya mengenkripsi file, tetapi juga memiliki komponen keylogger dan kemungkinan HVNC (Hidden VNC) untuk melakukan pengawasan jarak jauh secara diam-diam pada sistem korban.

String tambahan seperti PROCESSENTRY32, OpenProcess, TerminateProcess, CloseHandle, dan GetProcessById menunjukkan bahwa malware ini dapat memindai dan memanipulasi proses dalam sistem operasi. Hal ini umum dilakukan untuk menonaktifkan proses antivirus, backup, atau layanan lain yang dapat mengganggu proses enkripsi.

Akhirnya, keberadaan nama file seperti Client.exe, Client.Install, serta artefak XML konfigurasi menunjukkan bahwa executable ini dikemas secara rapi agar tampak seperti aplikasi Windows biasa, dengan nama generik dan tidak mencurigakan. Manifest file menunjukkan level eksekusi asInvoker, yang artinya malware ini tidak secara eksplisit membutuhkan hak administrator untuk berjalan, sehingga dapat dijalankan tanpa memicu peringatan keamanan dari sistem.

Secara keseluruhan, hasil ekstraksi string dari sampel Akira mendukung kesimpulan bahwa ransomware ini dibangun dengan menggunakan kombinasi enkripsi kuat, fitur spionase, dan teknik anti-analisis yang kompleks untuk memaksimalkan kerusakan dan menghindari deteksi.

4.4.2.2 Strings LockBit

Hasil ekstraksi strings terhadap file PE sampel ransomware lockbit.exe menunjukkan sejumlah temuan yang relevan, baik dari sisi struktur internal ransomware maupun indikasi aktivitas berbahaya, diantaranya:

1. Identifikasi struktur internal file PE

   Hasil ekstraksi strings menunjukkan adanya segmentasi atau *section* dari file sampel tersebut, diantaranya:

   a. .text : Sebagai tempat penyimpanan kode eksekusi utama
   b. .rdata : Bertujuan untuk memuat string ASCII dan nama-nama fungsi API
   c. .data : Bertujuan untuk menyimpan variable program
   d. .idata, .pdata, .reloc: Bertujuan sebagai media pendukung fungsi impor dan manajemen memori.

2. Deteksi teknik *obfuscation* dan *encoding*

   Hasil ekstraksi strings menunjukkan banyaknya kemunculan pola string yang acak atau tidak terbaca dengan mudah, seperti SQRVW, ^ZY], 5PQB, =HQB, 5lQB. Strings tersebut tidak memiliki makna langsung dalam ASCII dan muncul berulang kali dengan frekuensi kecil. Pola ini menunjukkan adanya teknik *obfuscation*.

Selain itu, ditemukan urutan karakter base64 seperti: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/ yang menunjukkan bahwa ransomware ini menyimpan konfigurasi atau *payload* dalam format *encoded*, yang baru akan didekode saat waktu eksekusi atau *runtime.*

3. Deteksi pemanfaatan Windows API

   Hasil ekstraksi string juga memperlihatkan banyaknya pemanggilan fungsi API Windows, seperti:

   a. LoadLibrary, GetProcAddress : Sebagai indikasi *dynamic function resolution* yang menunjukkan bahwa ransomware tersebut dirancang untuk menghindari deteksi oleh antivirus yang berbasis *signature.*
   b. CreateWindowExW, CreateDialogParamW : Digunakan untuk menampilkan cacatan tebusan.
   c. SetPixel, BitBlt : Sebagai petunjuk adanya akses ke komponen GDI (*Graphics Device Interface*), yang dapat digunakan untuk animasi GUI

4.4.2.3 BlackCat

Hasil ekstraksi strings terhadap file PE sampel ransomware blackcat.exe menunjukkan sejumlah temuan yang relevan, baik dari sisi struktur internal ransomware maupun indikasi aktivitas berbahaya, diantaranya:

1. Identifikasi struktur internal file PE

   Hasil ekstraksi strings menunjukkan adanya segmentasi atau *section* dari file sampel tersebut, diantaranya:

   a. .text : Sebagai tempat penyimpanan kode eksekusi utama
   b. .rdata : Bertujuan untuk memuat string ASCII dan nama-nama fungsi API
   c. .data : Bertujuan untuk menyimpan variabel program
   d. .idata, .bss, .tls, .reloc : Bertujuan sebagai segmen pendukung untuk impor fungsi, alokasi memori, dan inisialisasi *thread.*

2. Deteksi terhadap teknik *Obfuscated*

Tabel 4. 13 Strings Obfuscated BlackCat

No	String Obfuscated	Analisis
1.	Pj+h, j+hl, j(hd	Penggunaan pointer ke fungsi terenkripsi
2.	^_[], D$@P, L$PQ	Hasil dari *encoding* XOR, menunjukkan struktur strings yang tidak wajar
3.	t$hf., D$$;D$ u	Struktur berpola yang kemungkinan format hasil enkripsi blok
4.	SWVP, TXPV, t(W	Kemunculan berulang dan menyerupai tag kontrol internal
5.	DLP, LXVP	Adanya kemungkinan segmentasi memori yang terenkripsi

Pada tabel 4.13 merupakan sebagian besar string yang muncul dalam hasil ekstraksi. Kemunculan berulang string seperti ^_[] dan variasi D$, L$, T$, serta j+h menunjukkan penggunaan obfuscation atau encoding berbasis XOR/custom codec. Hal ini dilakukan untuk menyembunyikan string penting seperti nama file, URL command-and-control (C2), ataupun argumen enkripsi.

3. Indikasi Encoding dan Base64

Terdapat sejumlah pola string dan karakter yang menunjukkan adanya keberadaan encoding seperti base64. Meskipun tidak eksplisit menampilkan tabel karakter base64, namun adanya fragmentasi seperti !"#$%&'()*+,./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~ dan string seperti auth, version,

serta cipher yang mengindikasikan adanya konfigurasi internal yang tersimpan dalam format *encoding*.

4. Indikasi GUI dan Fungsi API Windows

   Berbeda dengan akira dan lockbit yang jelas menampilkan fungsi API, dalam Blackcat string API disamarkan dengan kode obfuscated. Namun beberapa referensi yang menunjukkan hal tersebut masih muncul seperti:

   - Pj+h, Pj7h, j(hd, t$QPRV
   - CreateWindowEx, SetPixel, dan struktur dialog (DP, DtP, L$PQ)

   Hal ini mengindikasikan adanya kemungkinan terhadap pemanfaatan komponen GUI untuk menampilkan antarmuka kepada korban seperti ransom note atau progress enkripsi. Kemunculan fungsi seperti SetPixel atau j(hd juga mengacu adanya kemungkinan yang digunakan untuk teknik visual seperti steganografi atau manipulasi layar.

5. Strings yang berkaitan dengan konfigurasi dan metadata

   Adanya string seperti authhor}1, version}, name1, usage1, shutdown3, file, path3, user, dan host, menunjukkan bahwa ransomware Blackcat memiliki konfigurasi internal yang digunakan untuk menyimpan informasi seperti versi malware, nama pembuat, lokasi file target, serta parameter eksekusi. Kemungkinan besar konfigurasi ini didekode saat runtime.

6. Strings yang berkaitan dengan adanya kemungkinan komunikasi dan variabel status

   Kemunculan strings seperti trueu, false, Onli, Idle, dan TryP menunjukkan bahwa ransomware ini mendukung status atau mode operasi tertentu, seperti aktik atau tidak aktif, idle, atau online. Hal ini biasa dikaitkan dengan Ransomware as a Service (RaaS) yang menguatkan dugaan bahwa Blackcat atau ALPHV adalah ransomware modular yang dapat dikendalikan melalui jarak jauh.

Berdasarkan hasil ekstraksi string, ransomware Blackcat menunjukkan tingkat kompleksitas yang tinggi. Teknik obfuscation yang konsisten, banyak menggunakan karakter acak, dan string konfigurasi menunjukkan bahwa ransomware ini dirancang untuk menghindari deteksi statis, menyimpan parameter konfigurasi internal, adanya kemungkinan memiliki antarmuka pada

korban dan status yang dinamis, serta menggunakan komunikasi yang terenkripsi untuk menjalankan perintah atau menampilkan pesan.

4.4.2.4 WhisperGate

Berdasarkan hasil ekstraksi string terhadap binary pada ransomware WhisperGate, ditemukan beberapa pola yang menunjukkan adanya string-string yang terdeteksi adanya indikasi penggunaan teknik obfuscation, persistence, pemanfaatan API Windows, serta adanya indikasi komunikasi atau payload yang terenkripsi.

Tabel 4. 14 Strings Section PE WhisperGate

No	Section PE	Fungsi
1.	.text	Sebagai instruksi program utama
2.	.data, .rdata	Tempat penyimpanan variabel dan string statis
3.	.bss	Tempat penyimpanan variabel yang tidak terinisialisasi
4.	.tls, .CRT	Thread-local storage dan runtime
5.	.idata	Informasi impor fungsi dari DLL eksternal
6.	.eh_farm	Exception handling frame

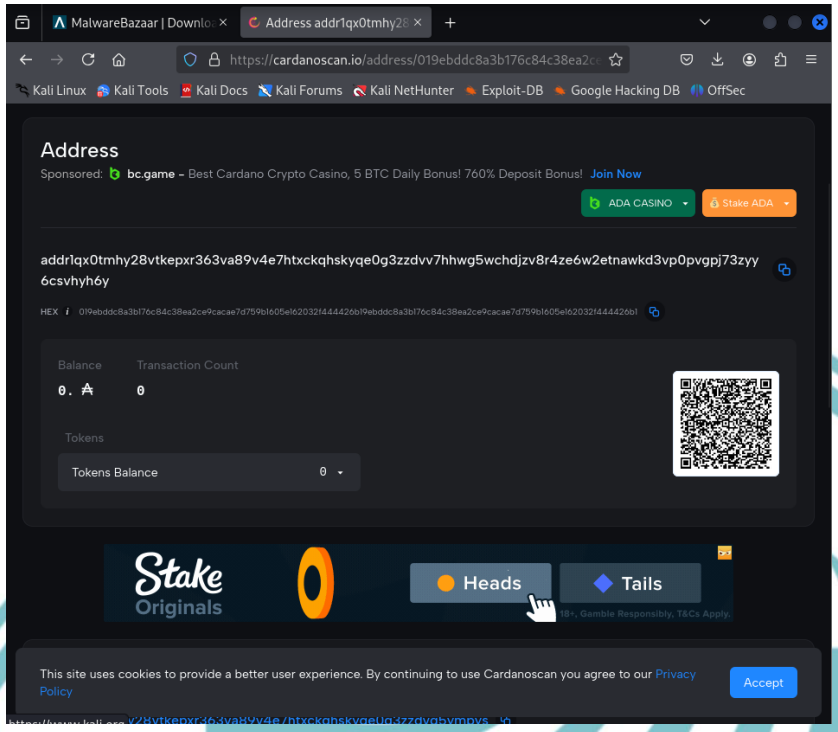
Pada tabel 4.14, menunjukkan struktur internal khas dari format *Portable Executable* (PE) yang ditandai adanya section .text, .data, .rdata, .bss, .idata, dan .tls. Selain itu, adanya keberadaan .eh_farm dan .CRT menunjukkan bahwa malware ini juga memperlihatkan adanya mekanisme penanganan eksepsi (*exception handling mechanism*) yang dalam konteks eksekusi ransomware mengacu pada sistem internal yang digunakan untuk mengantisipasi kesalahan

yang terjadi saat program berjalan. Struktur ini umum ditemukan pada program Windows yang dikompilasi menggunakan *toolchain* berbasis GCC atau MinGW, yang juga diperkuat dengan ditemukannya string *compiler* GCC: (GNU) 6.3.0. Komposisi section ini menegaskan bahwa malware dikompilasi secara utuh dan dirancang agar dapat berjalan stabil di sistem target.

Tabel 4. 15 Strings Obfuscated WhisperGate

No	String	Analisis
1.	D$#WGYZ, D$;VDMJ	Dugaan sebagai hasil *obfuscation*, sebab biasanya berpola D$....
2.	T$=f, [^_], <]t<	Adanya kemungkinan *padding*, *marker control,* atau tag enkripsi
3.	bc1q..., addr1qx...	Menunjukkan alamat kripto (kemungkinan Bitcoin)
4.	service.exe	Nama file *payload* tambahan atau *dropper* yang dijalankan
5.	GCC: (GNU) 6.3.0	Kompiler yang digunakan, kemungkinan dengan MinGW

Berdasarkan tabel 4.15, menunjukkan bahwa sampel ransomware WhisperGate mengindikasikan menggunakan teknik *obfuscation* berbasis skema pengkodean kustom atau XOR. Selain itu, strings seperti service.exe dan %APPDATA% menunjukkan lokasi eksekusi malware, sementara pengulangan string compiler seperti GCC: (GNU) 6.3.0 menandakan bahwa binary dikompilasi dengan kompiler *open-source* yang banyak digunakan dalam malware berbasis Windows.



Gambar 4. 34 String Alamat Kripto WhisperGate

Selain itu, ditemukan string seperti bc1q… dan addr1qx menunjukkan keberadaan alamat kripto yang digunakan untuk menerima pembayaran tebusan, seperti pada gambar 4.34.

Tabel 4. 16 Strings Fungsi API WhisperGate

No	Fungsi API	Tujuan
1.	CopyFileA, DeleteCriticalSection	Mengelola file dan thread
2.	ShellExecuteA, SetClipboardData	Menjalankan perintah dan interaksi dengan *clipboard*
3.	GetCommandLineA, LoadLibraryA	Resolusi dan pemanggilan fungsi secara dinamis
4.	VirtualProtect, VirtualQuery	Manajemen memori tingkat lanjut

5.	Sleep, ExitProcess	Kontrol eksekusi malware

Berdasarkan tabel 4.16, ransomware WhisperGate banyak menggunakan fungsi dari library sistem Windows seperti KERNEL32.dll, USER32.dll, SHELL32.dll, dan msvcrt.dll. Fungsi-fungsi penting seperti VirtualProtect, VirtualQuery, dan LoadLibraryA menunjukkan bahwa malware ini mampu melakukan modifikasi proteksi memori, memuat library tambahan, serta melakukan pemanggilan fungsi secara dinamis. Adanya API ShellExecuteA dan SetClipboardData juga mengindikasikan bahwa malware dapat menjalankan perintah eksternal serta mengakses data pada *clipboard*, yaitu sebuah fitur yang sering digunakan pada *infostealer* atau ransomware yang menyalin teks sensitif.

Tabel 4. 17 Strings Teknik Persistensi WhisperGate

No	String	Fungsi
1.	schtasks /create ... /tr %APPDATA%\service.exe	Menjadwalkan eksekusi harian malware
2.	%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\service.exe	Menempatkan malware pada direktori startup
3.	%s\%s	*Placeholder* konfigurasi *path* atau file
4.	cmd.exe, open	Menunjukkan eksekusi *Shell* pada Windows

Berdasarkan tabel 4.17, WhisperGate mengimplementasikan teknik persistensi dengan menjadwalkan dirinya melalui perintah schtaks, yang dijalankan

melalui cmd.exe. Perintah seperti /C schtasks /create /tn WinApp /tr %APPDATA%\service.exe /st 00:00 /sc daily /ri 1 /f, menunjukkan bahwa malware membuat tugas terjadwal (*Scheduled Task*) untuk mengeksekusi file service.exe setiap harinya tanpa batas tertentu. Lokasi target seperti %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup adalah direktori startup yang sering disalahgunakan malware untuk memastikan dirinya berjalan saat sistem di *reboot*. Teknik ini dikenal sebagai *persistence via registry* dan *task scheduler*.

Dalam banyak kasus, malware memanfaatkan sistem registry Windows sebagai media untuk mempertahankan persistensinya maupun menyimpan informasi konfigurasi. Teknik yang umum digunakan adalah dengan menambahkan entri tertentu pada registry agar malware tersebut dapat dieksekusi secara otomatis setiap kali sistem operasi melakukan proses booting. Struktur registry yang sangat kompleks dan luas, serta terdapat berbagai titik atau jalur yang dapat dieksploitasi oleh malware untuk menyamarkan kehadirannya serta mempertahankan akses secara terus-menerus dalam sistem. (Mylonas & Gritzalis, 2012)

4.4.2.5 Strings HellCat

Hasil ekstraksi strings terhadap file PE sampel ransomware Hellcat.exe menunjukkan sejumlah temuan yang relevan, baik dari sisi struktur internal ransomware maupun indikasi aktivitas berbahaya, diantaranya:

1. Struktur PE

Tabel 4. 18 String Struktur PE HellCat

No	Section PE	Fungsi
1.	.text	Sebagai instruksi program utama
2.	.rdata, .data	Tempat penyimpanan variabel dan string statis

3.	.pdata, .xdata	Terdapat informasi penanganan eksepsi (*Exception Handling*)
4.	.idata	Terdapat daftar fungsi eksternal yang dipanggil
5.	.reloc	Data relokasi yang mendukung eksekusi di alamat dinamis

Berdasarkan tabel 4.18, merupakan struktur Pe dari sampel ransomware Hellcat yang menunjukan file executable yang dikompilasi dengan standar Windows. Section .pdata dan .xdata mengindikasikan mekanisme *exception handling* (penanganan kesalahan pada saat *runtime*), sedangkan adanya .idata yang memuat referensi terhadap fungsi-fungsi eksternal dari sistem operasi. Dan section .reloc memperkuat indikasi bahwa malware ini dapat dieksekusi dalam berbagai alamat memori, menandakan adanya teknik *evasive* terhadap sistem keamanan.

2. Strings *obfuscated* dan *encoding*

Tabel 4. 19 Strings Obfuscates HellCat

No	String *obfuscated*	Indikasi
1.	DPH, T0H, L$xH	String internal dengan struktur berulang (*custom format*)
2.	\$pH, hL, L8H	Pola string *obfuscated* dan merupakan bagian dari tabel *lookup* atau konfigurasi *runtime*
3.	I;f, v-UH, t$PL	Kemungkinan *padding*

		fungsi atau *runtime marker*
4.	Go build ID: ...	Dibangun dengan Go Compiler

Berdasarkan tabel 4.19, analisis string menunjukkan bahwa ransomware Hellcat menggunakan sistem *encoding* khusus yang bertujuan untuk menyembunyikan instruksi khusus dan data internalnya. Pola string unik seperti DPH, T0H, dan L$xh muncul secara berulang dengan variasi yang kecil, menandakan adanya teknik *obfuscation* berbasis skema penamaan, yang dimana malware menggunakan nama-nama variabel, fungsi, label atau string internal yang telah diacak atau disamarkan dengan pola khusus yang hanya dikenali oleh malware itu sendiri.

Selain itu, ditemukan ID hasil kompilasi Go build dengan ID "UmIFugMF3nU6Bc5kkU8V/CxycpfVN0uDHE9r3cGS-/_5k-VZkPGGb_AsZlKNC0/fkixD_PDXGwDsxlq0_d0", yang menandakan bahwa malware ini dikembangkan menggunakan bahasa Golang yang terkenal dengan performa tinggi.

3. Indikasi Fungsi *Runtime* dan *Debugging*

Tabel 4. 20 Strings Fungsi Runtime HellCat

No	String	Fungsi
1.	debugCal, runtime	Referensi ke library runtime Golang
2.	1128, 1512, 1409u	Parameter untuk fungsi debug dan pemrosesan data
3.	error:, panic:	Menandakan penggunaan exception atau handling dalam Go

Berdasarkan tabel 4.20, ditemukan string-string seperti debugCal, runtime., dan error: yang merupakan indikasi bahwa ransomware Hellcat

memanfaatkan library runtime Golang, termasuk fungsi untuk debug dan logging. Hal ini sejalan dengan temuan ID Golang sebelumnya, memperkuat bukti bahwa malware menggunakan bahasa pemrograman modern untuk mendukung stabilitas, error handling, dan modularitas. Penggunaan angka-angka seperti 1512 dan 1409u menunjukkan konfigurasi internal atau skala pemrosesan tertentu.

4. Teknik Persistensi dan Eksekusi

Tabel 4. 21 Strings Teknik Persistensi HellCat

No.	String	Fungsi
1.	service.exe, tPH, LPH	Nama payload dan rutinitas eksekusi
2.	\$hH, tpH, T@H	String kontrol atau instruksi yang dieksekusi saat *booting*
3.	t$0H, L$PL, T$XL	Pola rutin yang berulang dan kemungkinan looping eksekusi

Berdasarkan tabel 4.21, string seperti service.exe mengindikasikan bahwa ransomware Hellcat kemungkinan menjatuhkan (*drop*) file dengan nama tersebut untuk dieksekusi secara berulang. Pola string t$PH, \$hH, dan T$@H menunjukkan struktur pengontrol internal yang digunakan saat proses eksekusi. Hal ini memungkinkan bahwa malware tersebut memiliki loop eksekusi yang dijalankan secara berkala atau berdasarkan pemicu waktu (*timer-based logic*), yang bertujuan untuk menjaga persistensi atau melanjutkan proses enkripsi.

4.4.2.6 Morpheus

Hasil ekstraksi strings terhadap file PE sampel ransomware Morpheus.exe menunjukkan sejumlah temuan yang relevan, baik dari sisi struktur internal ransomware maupun indikasi aktivitas berbahaya, diantaranya:

1. Analisis Ransom Note

```
Your network has been breached and all data were encrypted.
It can be restored to their original state with a decryptor key that
only we have.
1. Do NOT modify encrypted files yourself.
2. Do NOT use third-party software to restore your data.
3. Do NOT hire a recovery company. They can not decrypt without out
private key.
4. Do NOT reboot or turn off storage media.
If you do not contact us within 3 days, or we cannot reach an
agreement, information will either be sold, or shared with the media
We have already downloaded a huge amount of critical data.
Tags of downloaded information:
- Confidential docs
- Finance documents
- Business Plans
- Personal data of employees
- Personal data of clients
- Personal and confidential data of celebrities (Durk Banks, etc.)
- Videoconference depositions
- Various types of records relating to court cases
```

Gambar 4. 35 Strings Ransom Note Morpheus

Catatan tebusan yang ditampilkan oleh ransomware Morpheus seperti pada Gambar 4.35, menunjukkan pola komunikasi yang khas dari skema *double extortion*, yaitu metode pemerasan ganda yang tidak hanya mengenkripsi file korban, tetapi juga mengancam untuk menyebarluaskan data yang telah dicuri. Dalam pesan tersebut, pelaku menyatakan bahwa jaringan korban telah berhasil ditembus dan seluruh data telah dienkripsi. Pemulihan hanya dapat dilakukan menggunakan *decryptor key* yang diklaim hanya dimiliki oleh pelaku.

Dalam catatan tersebut, pelaku juga melampirkan larangan-larangan secara eksplisit kepada korban, seperti tidak memodifikasi file, tidak menggunakan perangkat lunak pihak ketiga, tidak mempekerjakan jasa pemulihan data, dan tidak mematikan perangkat penyimpanan. Instruksi tersebut umum dalam ransomware untuk mencegah kerusakan pada file terenkripsi dan memastikan korban tetap bergantung pada pelaku untuk proses pemulihan.

Lebih lanjut, pelaku memberikan ultimatum dalam waktu tiga hari. Jika korban tidak merespons atau menolak untuk membayar, maka data akan

dijual atau disebarluaskan ke media. Tekanan ini diperkuat dengan pernyataan bahwa mereka telah mengunduh sejumlah besar data penting. Daftar yang disampaikan sangat rinci, termasuk dokumen keuangan, rencana bisnis, data karyawan dan klien, hingga data pribadi selebriti dan dokumen hukum.

Strategi ini tidak hanya bersifat teknis, melainkan juga psikologis, yaitu memanfaatkan rasa takut akan kerugian reputasi dan kebocoran informasi sensitif untuk mendorong pembayaran. Pola ini menunjukkan bahwa ransomware Morpheus adalah bagian dari operasi siber yang terorganisir dan terencana, dengan orientasi finansial yang jelas dan pendekatan pemerasan berbasis eksfiltrasi data.

2. Sumber Pencurian Data

```
Sources of information:
192.168.2.2   (SRV03)
192.168.2.43  (RALLEN03)
192.168.2.92  (DESKTOP-2FL7LP3)
192.168.2.104 (JBOERST03)
192.168.2.114 (EDWARDS01)
192.168.2.163 (SHERNANDEZ03)
```

Gambar 4. 36 Strings Pemetaan Jaringan Morpheus

Bagian "Sources of information" pada catatan tebusan ransomware Morpheus seperti pada gambar 4.36, menampilkan daftar alamat IP internal dan nama host yang menjadi sumber data yang telah dicuri, yang diinterpretasikan seperti pada tabel dibawah ini:

Tabel 4. 22 Strings Proses Pemetaan Jaringan Morpheus

No	IP	Hostname	Interpretasi
1.	192.168.2.2	SRV03	Server internal utama, kemungkinan sebagai pusat file atau domain server

2.	192.168.2.43	RALLEN03	Komputer pengguna bernama R. Allen, workstation karyawan operasional
3.	192.168.2.92	DESKTOP-2FL7LP3	Perangkat klien biasa, kemungkinan staf bagian administratif
4.	192.168.2.104	JBOERST03	Workstation pengguna senior, diduga bagian IT atau pengelola data
5.	192.168.2.114	EDWARDS01	Komputer milik staf penting, kemungkinan manajemen atau HR
6.	192.168.2.163	SHERNANDEZ03	Host korban lainnya, bisa jadi menyimpan dokumen internal atau hukum

Berdasarkan tabel 4.22, terdapat nama-nama host seperti RALLEN03, EDWARDS01, dan SRV03 merujuk pada perangkat milik korban yang berada di dalam jaringan internal organisasi yang sama. Hal ini menunjukkan bahwa ransomware tidak hanya mengenkripsi file pada satu perangkat, tetapi juga melakukan proses pemetaan jaringan internal (*internal network mapping*) untuk menemukan sistem lain yang dapat diakses. Hostname tersebut kemungkinan merupakan nama pengguna atau penanda departemen dalam struktur TI organisasi, seperti SRV03 untuk server utama, atau EDWARDS01 yang merujuk pada staf bernama Edwards.

Informasi ini memperkuat bukti bahwa pelaku memiliki akses lateral (*lateral movement*) dan mampu mengambil data dari berbagai titik dalam jaringan, bukan hanya dari perangkat yang pertama kali terinfeksi. Pendekatan ini merupakan ciri khas dari ransomware modern dengan kemampuan *double extortion*, yang tidak hanya mengunci data lokal, tetapi

juga mengumpulkan dan mengeksfiltrasi informasi strategis dari berbagai sistem yang digunakan oleh korban.

3. Penggunaan String Fungsi Kriptografi

Tabel 4. 23 Strings Kriptografi Morpheus

No	Fungsi BCrypt	Fungsi
1.	BCryptDestroyKey	Menghapus kunci kriptografi dari memori
2.	BCryptGetProperty, BCryptSetProperty	Konfigurasi algoritma kripto
3.	BCryptOpenAlgorithmProvider, BCryptCloseAlgorithmProvider	Membuka dan menutup penyedia algoritma kriptografi
4.	BCryptGenRandom	Menghasilkan angka acak secara kriptografis
5.	BCryptImportKeyPair	Mengimpor pasangan kunci asimetris
6.	BCryptGenerateSymmetricKey	Membuat kunci enkripsi simetris

Berdasarkan tabel 4.23, ransomware Morpheus menggunakan library BCrypt, yaitu komponen dari Windows Cryptography API: Next Generation (CNG). Fungsi-fungsi seperti BCryptGenerateSymmetricKey dan BCryptGenRandom menunjukkan bahwa malware ini memiliki kemampuan untuk melakukan enkripsi tingkat lanjut dengan kunci dinamis, yang memperkuat keabsahan klaim bahwa file korban tidak dapat didekripsi tanpa private key dari pelaku.

4. Strings Fungsi Sistem Windows yang dipanggil

Tabel 4. 24 Strings Fungsi Windows Morpheus

No	API Windows	Tujuan
1.	GetModuleFileNameA, GetModuleHandleW	Untuk menentukan lokasi file malware yang berada dalam sistem
2.	GetLogicalDrives	Mengidentifikasi drive memori aktif untuk enkripsi
3.	SetFilePointerEx	Mengatur posisi baca atau tulis file
4.	WaitForSingleObject, SetThreadPriority	Manajemen eksekusi dan *multitasking* malware
5.	EnterCriticalSection, DeleteCriticalSection	Sinkronisasi thread

Berdasarkan tabel 4.24, Fungsi API seperti GetLogicalDrives dan SetFilePointerEx menandakan bahwa malware aktif melakukan scanning dan manipulasi file sistem. Fungsi threading (EnterCriticalSection, WaitForSingleObject) menunjukkan bahwa ransomware ini berjalan dalam struktur *multi-threaded*, meningkatkan efisiensi proses enkripsi. Hal ini memperkuat kesimpulan bahwa Morpheus dirancang untuk melakukan enkripsi cepat dan paralel dalam skala yang besar.

Berdasarkan hasil analisis dengan ekstraksi strings, dapat ditarik kesimpulan bahwa ransomware Morpheus merupakan varian malware dengan kapabilitas yang tinggi, penggunaan BCrypt untuk menggabungkan teknik enkripsi modern, kemampuan pemetaan jaringan, serta strategi *double extortion*. Adanya keberadaan catatan tembusan yang terstruktur dan penjabaran data eksfiltrasi mengindikasikan bahwa serangan ini tidak dilakukan secara acak, melainkan merupakan bagian dari operasi serangan siber terencana dengan orientasi finansial. Analisis string dan fungsi API menunjukkan pendekatan

secara sistematis, modular, dan efisien yang umum ditemukan pada ransomware tingkat lanjut.

4.4.2.7 Strings Prince

Hasil ekstraksi string pada sampel ransomware Prince menunjukkan bahwa ransomware ini memiliki pola karakteristik yang mirip dengan beberapa ransomware lain, yaitu banyak ditemukannya string yang tidak terbaca secara langsung (*obfuscated*) dan struktur data yang mengarah pada manipulasi sistem file atau memori. Seperti yang dijabarkan pada tabel dibawah ini:

Tabel 4. 25 Strings Prince

No	Temuan	Interpretasi
1.	.text, .data, .rdata, .reloc	Struktur umum pada file PE (*Portable Executable*) pada Windows
2.	Go build ID: ...	Ransomware ditulis dan dikembangkan menggunakan bahasa pemrograman Go (Golang)
3.	Simbol seperti \$pH, L$8H, dll.	Indikasi penggunaan teknik *obfuscation* berbasis penamaan variabel atau instruksi
4.	String debugCal, runtime.	Penggunaan referensi ke library *runtime* dan *debug*
5.	Pola M9,u, v,UH, DXH, dll.	Pola string acak yang berulang yang diduga sebagai bagian dari struktur data yang terenkripsi

Berdasarkan hasil analisis statik terhadap file ransomware Prince pada tabel 4.25, ditemukan sejumlah indikasi bahwa malware ini memiliki struktur file yang sesuai dengan format Portable Executable (PE), yang merupakan standar eksekusi pada sistem operasi Windows. Keberadaan section seperti .text, .rdata,

dan .idata menunjukkan bahwa file ini tidak hanya executable, tetapi juga dapat melakukan interaksi dengan library eksternal dan memori.

Salah satu temuan penting adalah keberadaan string Go build ID dengan identitas "h68odyhTCE8nwWMMwkjW/frQpE1w1g7X3WLI7AzKY/vGAMJx24SeqSNIbjeHmt/o2y4E-8-vd_kyC2OK5Fb", yang menunjukkan bahwa ransomware ini ditulis dan dikompilasi menggunakan bahasa pemrograman Go (Golang).

Selain itu, string-string seperti \$pH, L$8H, t$PH, dan sejenisnya mengindikasikan penggunaan skema penamaan yang tidak umum (*obfuscation*).

Penemuan string seperti debugCal dan runtime. dalam sampel ransomware menunjukkan bahwa file biner masih mengandung referensi terhadap modul internal dan fungsi debug milik Golang. Hal ini mengindikasikan bahwa pengembang malware tidak melakukan proses stripping secara penuh, yaitu tahap optimalisasi di mana simbol debug dan metadata kompilasi dihapus sebelum rilis. Keberadaan elemen-elemen ini dapat disebabkan oleh kelalaian pengembang atau sebagai langkah disengaja untuk menjaga kompatibilitas eksekusi lintas sistem. Secara keseluruhan, temuan-temuan tersebut menunjukkan bahwa Prince adalah ransomware modern yang menggunakan teknik pemrograman dan obfuscation lanjutan.

4.4.3 Hasil Analisis Dinamis dengan *Sandbox* Any.Run

4.4.3.1 Analisis Dinamis Akira

Hasil dari analisis dinamis terhadap sampel ransomware Akira.exe menunjukkan bahwa malware tersebut melakukan sejumlah aktivitas berbahaya, diantaranya:

1. Proses dan *thread* baru yang diciptakan selama eksekusi

Gambar 4. 37 Proses Akira

Pada Gambar 4.37, menunjukkan struktur proses yang terjadi dalam eksekusi sampel akira.exe. *Graph* tersebut memperlihatkan hubungan antar proses dalam bentuk *parent-child*, serta teknik eksekusi yang dilakukan. Berikut penjelasan dari masing-masing proses:

1. Proses utama: Akira.exe

   Proses utama akira.exe merupakan titik awal infeksi terjadi. Malware dijalankan oleh pengguna dan diklasifikasikan sebagai *malicious* oleh sistem deteksi Any.Run. Proses ini bertindak sebagai *dropper* sekaligus payload utama dan bertanggung jawab terhadap seluruh rangkaian aktivitas berbahaya berikutnya.(?)
2. Subproses: powershell.exe via WMI

   Selanjutnya, akira.exe memanggil powershell.exe melalui metode WMI (*Windows Management Instrumentation*) yang merupakan teknik eksekusi secara *stealthy*, dimana teknik tersebut dirancang untuk menghindari deteksi dari antivirus berbasis *signature.* Selanjutnya powershell.eve menjalankan perintah SgeWmiObject Win32_Shadowcopy | Remove-WmiObject untuk menghapus *shadow copy*, sehingga korban tidak dapat memulihkan sistem dengan menggunakan fitur pemulihan yang disediakan oleh Windows.
3. Subproses: conhost.exe

   Proses powershell.exe memunculkan conhost.exe, yang merupakan *Console Host Process* milik Windows. Hal tersebut merupakan proses sah yang biasanya aktif ketika terminal (PowerShell atau CMD) berjalan dalam mode GUI. Keberadaan conhost.exe mengonfirmasi bahwa

perintah powershell benar-benar dijalankan dalam terminal, bukan hanya pemanggilan API.

4. Proses tambahan: sppexcomobj.exe via COM

   Selain menggunakan WMI, akira.exe juga memanggil sppexcomobj.exe melalui COm (*Component Object Model*). sppexxcomobj.exe merupakan bagian dari layanan KMS (*Key Management Service*) yang sering digunakan untuk aktivasi Windows atau Office. Meskipun secara default tidak berbahaya, pemanggilan ini berpotensi digunakan untuk menghindari deteksi atau memanipulasi validasi lisensi sistem.

5. Subproses: slui.exe

   Proses terakhir dari sppexcomobj,exe adalah pemanggilan terhadap slui.exe, yaitu Windows Activation UI. Hal ini kemungkinan digunakan oleh malware untuk memicu sistem validasi ulang lisensi atau memanipulasi antarmuka aktivasi agar pengguna tidak menyadari aktivitas berbahaya yang sedang berlangsung

Dari graph dapat disimpulkan bahwa akira.exe menggunakan kombinasi teknik WMI dan COM-based execution untuk menyebarkan pengaruhnya ke dalam sistem. Teknik ini menunjukkan tingkat yang kompleks dan canggih, dengan tujuan utama menghindari deteksi, mempersulit analisis forensik, dan memperkuat kontrol atas sistem target sebelum enkripsi file dilakukan.

2. Modifikasi File dan Sistem

   Malware akira.exe melakukan banyak modifikasi secara masif pada struktur sistem file, termasuk pembuatan ratusan file berakhiran .akira dan penyebaran file ransom note di berbagai direktori target.

Tabel 4. 26 Modifikasi File Akira

Jenis Aktivitas	File	Keterangan
Pembuatan file ransom	akira_readme.txt	Ditemukan di banyak direktori
Enkripsi file sistem	*.lst.akira, *.bin.akira,	Banyak di temukan di file

	*.log.akira	Adobe dan FileZilla
File PowerShell	__PSScriptPolicyTest_xxx.ps1	Tanda aktivitas PowerShell

Berdasarkan tabel 4.26, malware Akira menciptakan dan memodifikasi lebih dari 300 file, banyak di antaranya memiliki nama asli file sistem yang kemudian diberi ekstensi .akira. Hal ini menandakan file telah terenkripsi.

Gambar 4. 38 Ransom Note Akira

Hasil analisis dinamis juga dapat menunjukkan isi dari file file yang dimodifikasi atau ditambahkan. Salah satunya, seperti pada Gambar 4.38 yang merupakan isi dari file *akira_readme.txt yang berisi ransom note. Di dalam catatan tersebut, terdapat beberapa poin penting yang tercantum dalam catatan tebusan tersebut antara lain:

1. Klaim penghapusan dan pencurian data

   Pelaku menyatakan bahwa infrastruktur internal korban telah dienkripsi dan data penting telah dicuri sebelum proses enkripsi dilakukan.
2. Ancaman publikasi data

Jika korban tidak menuruti ajakan kerjasama, pelaku akan menjual atau membocorkan informasi penting, seperti data keuangan, rahasia dagang, dan database ke berbagai aktor di *dark web*. Sebuah tautan .onion disertakan sebagai alamat blog tempat kebocoran akan dipublikasikan.

3. Ajakan dan instruksi teknik untuk negosiasi

   Korban diminta untuk menggunakan browser TOR dan mengunjungi alamat situs tersembunyi guna memulai proses negosiasi. File ini mencantumkan *unique victim ID* sebagai identitas saat membuka komunikasi dengan pelaku.

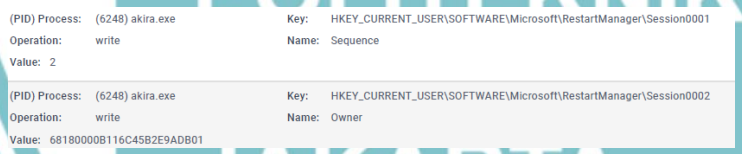
4. Tawaran *decrypt*

   Ransom note tersebut juga menyertakan tawaran untuk melakukan dekripsi terhadap satu file sebagai bentuk hasil negosiasi.

5. Gaya bahasa yang profesional dan persuasive

   Gaya bahasa yang digunakan cenderung formal dan persuasif, dengan argumen bahwa dengan membayar tebusan proses pemulihan dapat lebih hemat biaya dan waktu dibanding pemulihan mandiri. Hal ini menunjukkan pelaku tidak hanya mengandalkan ancaman teknis, tetapi juga mencoba manipulasi psikologis korban.

4. Modifikasi Registry



Gambar 4. 39 Modifikasi Registry Akira

Pada Gambar 4.39, menunjukkan adanya perubahan atau modifikasi signifikan pada Windows registry yang terletak pada subkey RestartManager yang digunakan untuk mengelola state dari sesi yang aktif. Key registry …Session0001 digunakan untuk menyimpan data sesi pengguna dan hash file, sedangkan key registry …Session0002 digunakan untuk menyimpan hash dan *sequence* atau urutan untuk file yang dienkripsi.

Dengan kesimpulan bahwa yang dilakukan ransomware akira.exe menunjukkan adanya penggunaan registry untuk menyimpan atau melacak informasi proses dan file yang dienkripsi.

5. Aktifitas Jaringan

Gambar 4. 40 Aktivitas Jaringan Akira

Pada gambar 4.40, merupakan hasil perekaman file .pcap dari sesi eksekusi, hal ini menunjukkan bahwa akira.exe melakukan beberapa koneksi keluar (*outbound*) ke domain dan IP yang mencurigakan, seperti:

Tabel 4. 27 Domain IP Mencurigakan dari Aktivitas Jaringan Akira

IP tujuan	Port	Protokol	Keterangan
51.124.78.146	443	TLSv1.2	Diduga endpoint server C2
23.216.77.18	80	HTTP	Domain terkait Akamai CDN
crl.microsoft.com	80	HTTP	Validasi sertifikat (bisa jadi dimanipulasi)

Berdasarkan tabel 4.27, menunjukkan IP 51.124.78.146 mengindikasikan menjadi tujuan dari sejumlah sesi komunikasi terenkripsi yang terjadi sesaat setelah ransomware dijalankan. Komunikasi ini menggunakan protokol TLSv1.2 dan membangun sesi melalui port 443, yang memungkinkan lalu lintas tersebut terlihat seperti koneksi HTTPS biasa, sehingga sulit terdeteksi oleh sistem pertahanan jaringan berbasis signature.

Selain itu, ditemukan koneksi ke domain Microsoft (crl.microsoft.com) yang digunakan untuk memvalidasi sertifikat, serta ke CDN milik Akamai (23.216.77.18). Meskipun koneksi ini bisa jadi sah, keberadaannya dalam sesi eksekusi ransomware tetap perlu dicurigai karena berpotensi dimanipulasi untuk menyamarkan komunikasi C2 atau mengunduh payload tambahan.

Lebih lanjut, Wireshark juga mencatat anomali pada sesi TCP, seperti:

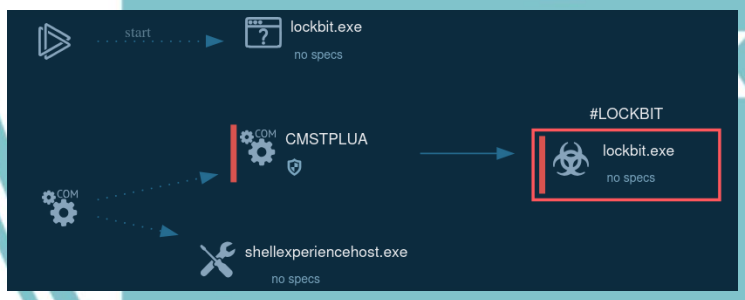
- Out-of-Order packet
- Duplicate ACK

yang bisa menjadi indikasi adanya modifikasi lalu lintas, misalnya upaya penghindaran deteksi atau teknik *evasion* yang dilakukan oleh malware saat membangun sesi komunikasi.

4.4.3.2 Analisis Dinamis LockBit

Hasil dari analisis dinamis terhadap sampel ransomware Lockbit.exe menunjukkan bahwa malware tersebut melakukan sejumlah aktivitas berbahaya, diantaranya:

1. Proses dan *Thread* Baru



Gambar 4. 41 Proses dan Thread Baru LockBit

Pada Gambar 4.41, menunjukkan alur eksekusi proses LockBit dalam bentuk *visual tree*, dengan node dan koneksi antar proses. Berikut penjelasan dari masing-masing proses:

1. Proses utama: lockbit.exe

   Proses lockbit.exe merupakan titik awal dari eksekusi malware, dijalankan secara langsung oleh pengguna (dalam konteks simulasi sandbox ANY.RUN). Pada tahap ini, proses masih berjalan dengan hak

akses pengguna biasa (*medium integrity)* dan belum melakukan aktivitas enkripsi secara langsung. Proses ini berfungsi sebagai *loader* atau penginisiasi tahapan awal serangan, termasuk mempersiapkan lingkungan untuk elevasi hak akses.

2. Eksekusi melalui CMSTPLUA (UAC Bypass)

   Langkah selanjutnya yang terdeteksi dalam behavior graph adalah pemanggilan terhadap CMSTPLUA.exe, sebuah komponen bawaan Windows yang biasa digunakan untuk mengeksekusi perintah administratif dengan perantara *User Account Control* (UAC).

   Pemanfaatan CMSTPLUA.exe menunjukkan bahwa LockBit mencoba melakukan *privilege escalation* (peningkatan hak akses) melalui metode yang bersifat sah namun disalahgunakan, dikenal sebagai UAC bypass. Teknik ini memungkinkan malware untuk mengeksekusi ulang dirinya sendiri dengan otorisasi yang lebih tinggi tanpa menampilkan notifikasi keamanan kepada pengguna.

3. Proses replikasi: lockbit.exe kedua

   Setelah berhasil melalui CMSTPLUA, malware menjalankan kembali lockbit.exe, namun kali ini sebagai proses yang berjalan dengan hak akses administratif penuh. Proses inilah yang teridentifikasi sebagai induk dari seluruh aktivitas enkripsi file, modifikasi registry, penyebaran ransom note, dan upaya persistensi sistem.

   Kondisi *elevated privilege* ini memungkinkan ransomware untuk mengakses direktori sistem, menonaktifkan fitur keamanan, dan membuat perubahan sistem tanpa batasan.

4. Proses tambahan: shellexperiencehost.exe

   Proses tambahan yang juga terdeteksi adalah shellexperiencehost.exe, komponen Windows yang menangani antarmuka pengguna (user interface shell). Meskipun bukan proses utama, keterlibatan proses ini menunjukkan kemungkinan bahwa LockBit juga berinteraksi dengan pengaturan UI atau mengakses profil pengguna, seperti:

- Riwayat folder terbuka
- Lokasi shortcut penting
- Data pengguna aktif di shell session

Aktivitas ini berpotensi mendukung fungsi exfiltrasi informasi pengguna atau memfasilitasi penyebaran ransom note ke lokasi-lokasi folder atau direktori yang kemungkinan besar akan diakses oleh korban.

Dari behavior graph pada sampel ransomware lokbit.exe, dapat disimpulkan bahwa ransomware LockBit menggunakan strategi eksekusi yang berlapis dan tersembunyi, dimulai dari proses awal dengan hak akses biasa, kemudian melakukan bypass terhadap kontrol akses sistem, dan menjalankan kembali proses utama sebagai administrator. Pendekatan ini memungkinkan LockBit untuk beroperasi secara maksimal di dalam sistem tanpa dibatasi oleh level akses pengguna, serta memperkecil kemungkinan deteksi oleh sistem keamanan.

2. Modifikasi File

Berdasarkan observasi terhadap bagian Files Activity pada sesi analisis dengan Any.Run, ransomware LockBit menunjukkan perilaku agresif terhadap sistem file. Berikut ini adalah penjabaran dari jenis-jenis aktivitas file yang dilakukan:

1. Pembuatan Ransom Note

   Ransomware LockBit melakukan pembuatan file ransom note yang dinamai secara acak namun mengikuti pola tertentu, misalnya xEJOHNVZF.README.txt. File ini disisipkan ke berbagai direktori penting dalam sistem, termasuk folder milik pengguna seperti Desktop, Pictures, dan Favorites, serta ke direktori internal seperti AppData\Local\Adobe.

2. Enkripsi file sistem

   LockBit juga melakukan enkripsi pada file konfigurasi dan log sistem, diantaranya:

- File .chk di folder TileDataLayer\Database
- File .log dari aplikasi Outlook di Diagnostik\OUTLOOK
- File konfigurasi .settingcontent-ms di folder Control Panel

Aktivitas ini menandakan bahwa LockBit tidak hanya menyasar data pengguna biasa, tetapi juga menyusupi dan memodifikasi file yang menjadi bagian dari struktur internal dan konfigurasi sistem Windows itu sendiri.

3. Penggunaan Recycle Bin

   Dalam upaya menyamarkan file hasil enkripsi, LockBit memindahkan beberapa file ke direktori Recycle Bin (C:\$Recycle.Bin\...) dan mengganti nama file tersebut dengan string acak. Teknik ini merupakan bentuk penyamaran, agar file yang sebenarnya telah dienkripsi tampak seperti file yang hanya dihapus, sehingga pengguna tidak segera menyadari bahwa file tersebut telah dienkripsi. Contoh dari aktivitas ini adalah file JJJJJJJJJJJ yang ditemukan di dalam C:\$Recycle.Bin\S-1-5-18, yang ketika dianalisis ternyata merupakan file binary hasil enkripsi.

4. Pengubahan Nama dan Ekstensi

   Selain memindahkan file, LockBit juga melakukan modifikasi terhadap file yang ada. Banyak file yang setelah dienkripsi mengalami perubahan struktur, antara lain hilangnya ekstensi asli, penggunaan ekstensi acak seperti .xEJOHNVZF, atau bahkan tidak memiliki ekstensi sama sekali. Akibat dari perubahan ini, file-file tersebut tidak lagi dapat dikenali atau dibuka oleh aplikasi bawaan sistem, yang merupakan ciri khas dari teknik destruktif enkripsi ransomware.

3. Modifikasi Registry

   Selama proses eksekusi ransomware LockBit, terdeteksi beberapa aktivitas yang melibatkan modifikasi entri registry Windows. Salah satu registry yang dimodifikasi adalah HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer, yang berkaitan langsung dengan pengaturan tampilan di File Explorer. Modifikasi pada entri ini berpotensi digunakan untuk

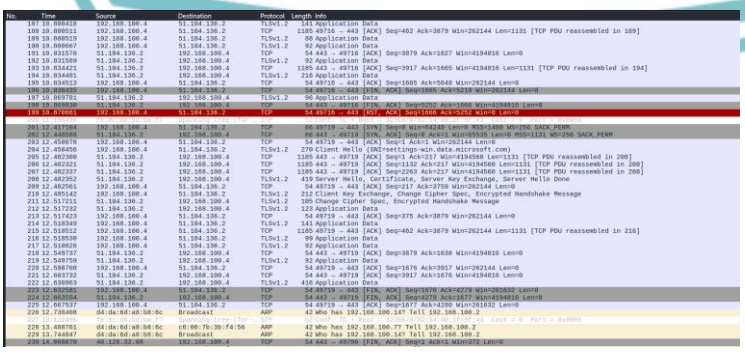
menyembunyikan file yang telah terenkripsi, menyembunyikan ekstensi, atau mencegah file tertentu ditampilkan oleh pengguna.

Selain itu, ditemukan pula aktivitas pada entri \REGISTRY\A\{a78ae51a-3893-71b8-c96c-b547f2d3e112}\LocalState, yang merupakan representasi virtual dari penyimpanan lokal aplikasi di AppData. Penulisan data pada lokasi ini mengindikasikan bahwa malware mungkin menyimpan informasi sementara atau penanda status eksekusi untuk proses berikutnya.

Modifikasi paling menarik terjadi pada HKEY_CLASSES_ROOT\LocalSettings\Software\Microsoft\Windows\CurrentVersion\AppContainer\Storage\microsoft.windows.contentdeliverymanager_cw5n1h2txyewy\Internet Settings\Cache, yang merupakan bagian dari sistem cache pada aplikasi modern. Perubahan pada entri ini mengindikasikan bahwa ransomware berusaha untuk mengakses atau menyimpan data dari browser internal aplikasi, seperti cookie atau riwayat aktivitas pengguna. Hal ini menandakan bahwa LockBit tidak hanya melakukan enkripsi, tetapi juga berpotensi memiliki fungsi sebagai stealer yang mengumpulkan data sensitif.

Secara keseluruhan, perubahan registry yang dilakukan oleh LockBit tidak bersifat mencolok, namun menandakan adanya upaya modifikasi mendalam terhadap sistem, baik untuk menyembunyikan jejak, menciptakan keberlanjutan eksekusi, maupun memanfaatkan data pengguna untuk kepentingan ekstorsi lanjutan.

4. Aktivitas Jaringan



Gambar 4. 42 Aktivitas Jaringan LockBit

Berdasarkan hasil observasi pada lalu lintas jaringan menggunakan Wireshark, seperti pada Gambar 4.42, proses lockbit.exe yang berjalan pada alamat IP lokal 192.168.100.4 diketahui melakukan sejumlah aktivitas jaringan yang bersifat lokal maupun eksternal. Aktivitas ini dimulai dengan koneksi ke alamat IP eksternal 20.73.194.208 melalui port 443, yang menunjukkan penggunaan protokol HTTPS/TLS untuk komunikasi terenkripsi. Koneksi tersebut ditandai dengan paket TCP SYN serta beberapa TCP Retransmission, yang menunjukkan upaya komunikasi berulang dan intensif dengan tujuan tertentu. Pola ini umum terjadi ketika ransomware mencoba menghubungi server Command and Control (C2) untuk mengirimkan kunci enkripsi, laporan status infeksi, atau bahkan menerima instruksi atau payload tambahan.

Selain komunikasi ke luar, LockBit juga menghasilkan lalu lintas broadcast dalam jaringan lokal, yang mencakup protokol berikut:

- NetBIOS Name Service (NBNS), digunakan untuk mendaftarkan nama perangkat (DESKTOP-JGLLJLD) dalam jaringan lokal, yang dapat digunakan untuk enumerasi sistem aktif di dalam jaringan.
- *Address Resolution Protocol* (ARP), digunakan untuk mencari pemilik IP tertentu melalui siaran broadcast.
- ICMPv6 *Neighbor Solicitation* dan *Advertisement*, yang merupakan bagian dari komunikasi dasar jaringan IPv6, menandakan bahwa sistem bekerja dalam mode dual stack (IPv4 dan IPv6).

Meskipun lalu lintas seperti NBNS dan ICMPv6 tergolong sebagai komunikasi jaringan standar, dalam konteks malware, protokol-protokol ini dapat dimanfaatkan untuk enumerasi jaringan internal, deteksi host lain, atau pengukuran respons terhadap broadcast, terutama jika ransomware memiliki komponen penyebaran lateral (meskipun belum teramati secara eksplisit pada sampel ini).

Pada fase lanjutan eksekusi, ransomware LockBit memperlihatkan aktivitas komunikasi jaringan yang sangat aktif dan kompleks, kali ini melibatkan

berbagai alamat IP eksternal dan menggunakan protokol TLSv1.2 secara konsisten. Beberapa koneksi yang tercatat adalah sebagai berikut:

- 4.175.87.197 dengan SNI lscr.update.microsoft.com
- 20.190.160.64 dengan SNI login.live.com
- 172.211.123.249 dengan SNI client.wns.windows.com

Domain pada Server Name Indication (SNI) tersebut merupakan bagian dari ekosistem sah milik Microsoft, sehingga menimbulkan dua kemungkinan yaitu malware menyamarkan koneksi keluar ke server C2 dengan domain impersonation, atau malware memanfaatkan proses validasi TLS oleh sistem untuk menyisipkan data atau melakukan tunneling informasi secara tersembunyi. Ukuran data yang dikirim setelah proses handshake (antara 463 hingga 731 byte Application Data) memperkuat dugaan bahwa komunikasi ini membawa informasi aktual, seperti:

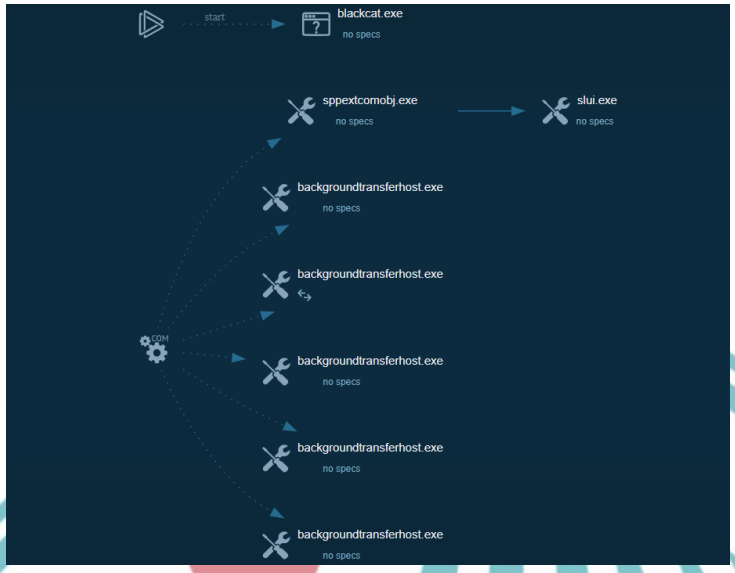
- Kunci enkripsi korban (encryption key)
- Identitas sistem seperti hostname, username, dan versi OS
- Log aktivitas ransomware untuk pelaporan eksekusi ke operator

Dengan demikian, LockBit tidak hanya melakukan serangan lokal berupa enkripsi file, tetapi juga secara aktif berinteraksi dengan jaringan eksternal dan internal, memanfaatkan komunikasi terenkripsi dan protokol jaringan sah untuk menjalankan fungsionalitas Command and Control (C2) sekaligus menghindari deteksi keamanan jaringan.

4.4.3.3 Analisis Dinamis BlackCat

Hasil dari analisis dinamis terhadap sampel ransomware Blackcat.exe menunjukkan bahwa malware tersebut melakukan sejumlah aktivitas berbahaya, diantaranya:

1. *Behaviour Graph* dan Struktur Proses



Gambar 4. 43 Behaviour Graph BlackCat

Pada Gambar 4.43, merupakan *behaviour graph* pada ransomware BlackCat. Proses utama blackcat.exe dijalankan secara manual oleh pengguna dari direktori sementara (AppData\Local\Temp), yang kemudian memulai rantai eksekusi berlapis.

Langkah pertama yang dilakukan oleh blackcat.exe adalah menjalankan SppExtComObj.Exe, sebuah komponen Windows resmi yang biasanya digunakan dalam aktivasi lisensi, namun dalam kasus ini dimanfaatkan sebagai media untuk bypass UAC (*User Account Control*). Proses ini kemudian meneruskan eksekusi ke slui.exe, yang merupakan bagian dari antarmuka aktivasi Windows.

Selain itu, muncul juga beberapa proses bagian dari BackgroundTransferHost.exe, yaitu proses sistem Windows yang biasa digunakan untuk mengelola transfer data latar belakang. Kemunculannya lebih dari lima kali selama eksekusi menunjukkan bahwa proses ini dimanfaatkan oleh ransomware untuk melakukan aktivitas tersembunyi, seperti mengunduh data dari server eksternal, menyimpan konfigurasi, atau mengeksekusi modul tambahan.

Struktur proses ini mencerminkan pola serangan yang bersifat modular, memanfaatkan trusted system binaries, serta menghindari deteksi melalui penggunaan komponen sah dari sistem operasi Windows.

2. Modifikasi File

Berdasarkan hasil analisis, tidak ditemukan file yang terenkripsi secara eksplisit. Namun terdapat lima proses dari BackgroundTransferHost.exe yang lokasi di path C:\Users\admin\AppData\Local\Packages\Microsoft.Windows.ContentDeliveryManager_cw5n1h2txyewy\AC\....down_data, .down_meta, dan .up_meta_secure.

Hal ini mengindikasikan bahwa BackgroundTransferHost.exe mengunduh file melalui ContentDeliveryManager, yaitu komponen Windows yang digunakan untuk distribusi konten seperti *wallpaper* atau notifikasi. Teknik ini menunjukkan upaya malware untuk memanipulasi aktivitas jaringannya dengan menyisipkan file ke dalam direktori sistem yang dianggap sah oleh sistem operasi, sekaligus menghindari deteksi berbasis path atau file name.

3. Perubahan Registry

Hasil analisis ransomware BlackCat ditemukan beberapa perubahan registry, diantaranya:

Tabel 4. 28 Perubahan Registry BlackCat

Path Registry	Operasi	Keterangan
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\Content	Write	Digunakan untuk menyimpan metadata korban
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\Cookies	Write	Digunakan untuk menyisipkan cookie pengguna
HKEY_CLASSES_ROOT \...\Internet	Write	Digunakan untuk mencatat riwayat

Settings\Cache\History		halaman yang pernah dikunjungi pengguna

Berdasarkan tabel 4.28, Selama proses eksekusinya, ransomware BlackCat terdeteksi melakukan modifikasi terhadap beberapa entri registry yang berada dalam jalur Internet Settings\Cache. Perubahan ini dilakukan oleh proses bawaan Windows BackgroundTransferHost.exe, yang disalahgunakan oleh malware.

Tiga registry utama yang dimodifikasi mencakup:

1. Content: digunakan untuk menyimpan metadata dari konten yang diakses atau diunduh.
2. Cookies: menyisipkan atau mengakses data cookie pengguna.
3. History: mencatat riwayat halaman yang telah dikunjungi oleh sistem.

Aktivitas ini mengindikasikan bahwa BlackCat tidak hanya melakukan enkripsi data, tetapi juga kemungkinan memiliki fungsi tambahan sebagai data stealer, khususnya yang menargetkan informasi perilaku pengguna dari browser atau aplikasi modern.

4. Aktivitas Jaringan

Berdasarkan hasil analisis, ransomware BlackCat melakukan aktivitas jaringan, diantaranya:

Tabel 4. 29 Aktivitas Jaringan BlackCat

Proses	IP Tujuan	Domain	Port	Keterangan
BackgroundTransferHost.exe	2.19.96.123	www.bing.com	443	Koneksi TLS Outbound
svchost.exe	20.190.160.5	login.live.com	443	Koneksi TLS, mirip seperti LockBit

backgroundTaskHost.exe	20.223.36.55	arc.msn.com	443	Koneksi outbond Microsoft
SIHClient.exe	52.165.164.15	fe3cr.delivery.mp.microsoft.com	443	Sumber update certificate
svchost.exe	40.113.103.199	client.wns.windows.com	443	Push client untuk windows

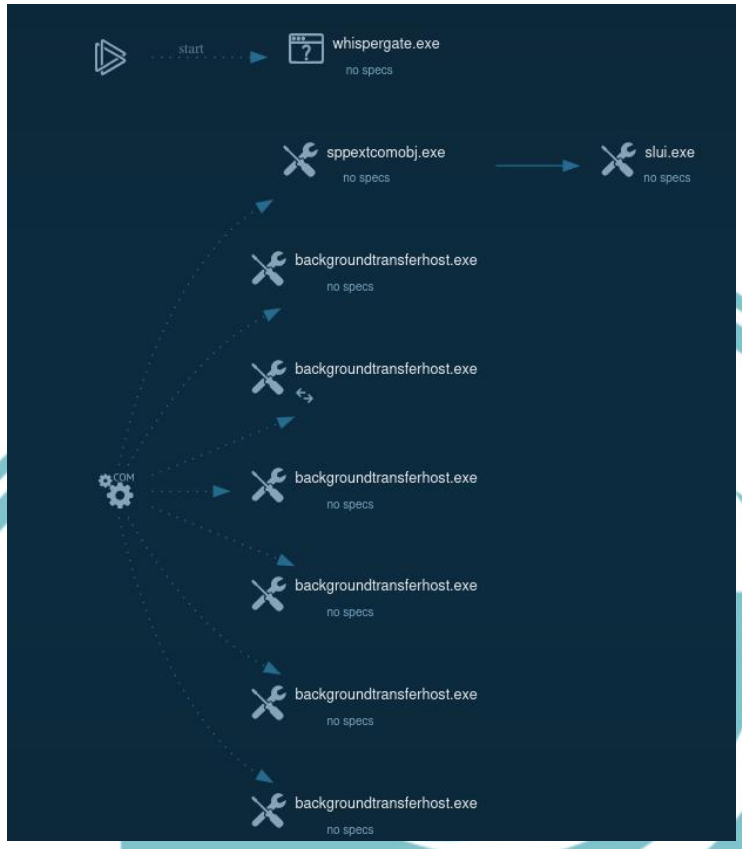
Berdasarkan tabel 4.29, menunjukkan bahwa selama proses eksekusinya, ransomware BlackCat tercatat melakukan sejumlah koneksi outbound menggunakan protokol TLS pada port 443. Koneksi ini dilakukan oleh beberapa proses bawaan Windows seperti svchost.exe, BackgroundTransferHost.exe, backgroundTaskHost.exe, dan SIHClient.exe. Seluruh koneksi ditujukan ke domain milik Microsoft, seperti www.bing.com, login.live.com, arc.msn.com, dan client.wns.windows.com.

Meskipun pada terlihat bahwa domain ini tampak sah, pola komunikasi ini menunjukkan indikasi pemanfaatan domain terpercaya sebagai media untuk menyamarkan aktivitas jaringan malware. Proses BackgroundTransferHost.exe secara khusus menjalankan koneksi TLS outbound ke bing.com, yang dapat dimanfaatkan untuk mengambil file atau payload tambahan secara tersembunyi. Aktivitas ini mencerminkan upaya ransomware untuk menghindari deteksi keamanan jaringan dengan menggunakan jalur komunikasi terenkripsi dan proses yang dipercaya oleh sistem.

4.4.3.4 Analisis Dinamis WhisperGate

Hasil dari analisis dinamis terhadap sampel ransomware whispergate.exe menunjukkan bahwa malware tersebut melakukan sejumlah aktivitas berbahaya, diantaranya:

1. *Behaviour Graph* dan Struktur Proses



Gambar 4. 44 Struktur Proses WhisperGate

Pada Gambar 4.44, *behavior graph* menunjukkan bahwa proses utama whispergate.exe secara langsung memicu sppExtComObj.exe, yang kemudian menjalankan slui.exe sebagai pola yang identik dengan teknik UAC bypass. Setelah itu, malware memicu enam instansi BackgroundTransferHost.exe, sebuah proses bawaan Windows yang biasa digunakan untuk pengunduhan di latar belakang. Banyaknya instansi ini mengindikasikan bahwa WhisperGate menjalankan fungsi komunikasi jaringan, modifikasi registry, atau pengunduhan payload tambahan secara modular dan tersembunyi.

2. Modifikasi File

Tabel 4. 30 Modifikasi File WhisperGate

Lokasi File	Jenis File	Keterangan

AppData\Local\Packages... \ContentDeliveryManager\ AC*.down_data / *.down_meta / *.up_meta_secure	File Unduhan	Disimpan di folder ContentDeliveryManager

Berdasarkan tabel 4.30, menunjukkan bahwa WhisperGate menulis file hasil unduhan ke dalam direktori ContentDeliveryManager, yang biasanya digunakan oleh sistem untuk mengunduh konten resmi. Penggunaan jalur ini bertujuan untuk menyamarkan aktivitas jaringannya agar tidak terdeteksi sebagai aktivitas mencurigakan.

3. Perubahan Registry

Tabel 4. 31 Perubahan Registry WhisperGate

Path Registry	Operasi	Keterangan
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\Content	Write	Digunakan untuk menyimpan metadata korban
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\Cookies	Write	Digunakan untuk menyisipkan cookie pengguna
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\History	Write	Digunakan untuk mencatat riwayat halaman yang pernah dikunjungi pengguna

Berdasarkan tabel 4.31, proses BackgroundTransferHost.exe melakukan penulisan registry yang berkaitan dengan cache browser, termasuk metadata konten, cookies, dan history. Aktivitas ini menunjukkan bahwa WhisperGate memiliki fungsi *profiling* pengguna atau kemampuan pencurian data berbasis aktivitas internet.

4. Aktivitas Jaringan

Tabel 4. 32 Aktivitas Jaringan WhisperGate

Proses	IP Tujuan	Domain	Port	Keterangan
BackgroundTransferHost.exe	23.212.110.171	www.bing.com, ocsp.digicert.com	443	Sertifikat dan unduhan konten
svchost.exe	20.190.160.5	login.live.com	443	Autentikasi akun Microsoft
backgroundTaskHost.exe	20.223.36.55	arc.msn.com	443	Koneksi outbond Microsoft
SIHClient.exe	172.202.163.200	slscr.update.microsoft.com	443	Update root certificate sistem

Berdasarkan tabel 4.32, seluruh koneksi jaringan menggunakan protokol TLS dan diarahkan ke domain sah milik Microsoft. Hal ini menunjukkan bahwa WhisperGate menggunakan teknik penyamaran komunikasi (*living-off-the-land*) agar lalu lintas jaringannya tidak terdeteksi oleh sistem keamanan. Semua koneksi terjadi melalui proses sah seperti svchost.exe dan BackgroundTransferHost.exe, yang memperkuat sifat stealth dari malware ini.

4.4.3.5 Analisis Dinamis HellCat

Hasil dari analisis dinamis terhadap sampel ransomware hellcat.exe menunjukkan bahwa malware tersebut melakukan sejumlah aktivitas berbahaya, diantaranya:

1. Struktur Proses

Tabel 4. 33 Struktur Proses HellCat

PID	Proses	Keterangan
6620	Hellcat.exe	Proses utama malware dan memicu conhost.exe dan tasklist.exe
—--	Tasklist.exe	Dijalankan secara berulang oleh hellcat.exe
7388	SppExtComObj.exe	Disalahgunakan untuk *privilege escalation*
7424	slui.exe	Diluncurkan oleh SppExtComObj.exe
2320	conhost.exe	Dijalankan untuk mendukung proses tasklist.exe

Pada tabel 4.33, menunjukkan bahwa struktur proses pada malware hellcat.exe memperlihatkan karakteristik eksekusi berulang dan intensif. Proses utama hellcat.exe memicu lebih dari 20 instansi tasklist.exe, yang merupakan perintah sistem Windows untuk menampilkan daftar proses yang sedang berjalan. Pola ini mengindikasikan bahwa malware melakukan enumerasi proses secara intensif, kemungkinan besar untuk mengidentifikasi target spesifik, seperti aplikasi browser, password manager, atau proses keamanan yang sedang aktif.

Selain tasklist.exe, terdeteksi juga pemanggilan SppExtComObj.exe—komponen aktivasi Windows yang kemudian memicu slui.exe, menunjukkan adanya upaya eskalasi hak akses melalui teknik UAC bypass, seperti yang juga ditemukan pada malware BlackCat dan WhisperGate.

Proses hellcat.exe juga memicu conhost.exe, yakni proses sah Windows yang digunakan sebagai antarmuka command-line. Kemunculannya menunjukkan bahwa malware menjalankan perintah berbasis terminal,

seperti tasklist.exe, untuk melakukan enumerasi terhadap informasi sistem. Hal ini memperkuat bahwa Hellcat menggunakan teknik native bawaan Windows untuk menyembunyikan aktivitasnya.

Proses enumerasi merupakan tahap awal dalam eksekusi malware yang bertujuan untuk mengumpulkan informasi sistem secara detail. Dalam kasus Hellcat.exe, hal ini ditunjukkan melalui eksekusi berulang terhadap tasklist.exe, yang memungkinkan malware melakukan pemetaan proses aktif pada sistem korban.

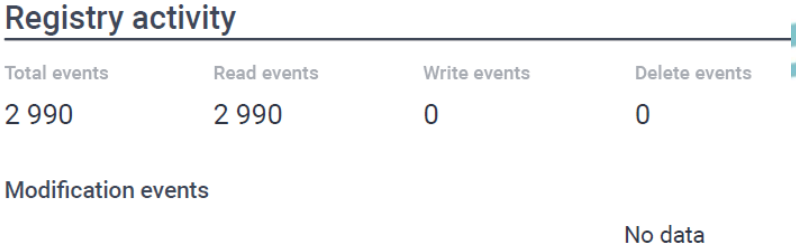
2. Modifikasi File

Berdasarkan hasil analisis, ransomware Hellcat memodifikasi dan menghasilkan 783 file terenkripsi (.HC). 127 file diantaranya merupakan file ransom note README_HELLCAT_.txt yang tersebar di beberapa direktori, seperti:

- AppData\Local\Adobe\Acrobat\DC...
- AppData\Local\Comms\UnistoreDB...
- AppData\Local\Google\Chrome\User Data...
- AppData\Local\FileZilla...

Selain menghasilkan file terenkripsi dan ransom note, malware ini juga mengganti, menimpa, dan menyimpan file sensitif dalam struktur aplikasi seperti Chrome dan FileZilla, hal ini menunjukkan fokus pencurian data pada browser dan protokol FTP.

3. Perubahan Registry



Gambar 4. 45 Perubahan Registry HellCat

Berdasarkan hasil laporan analisis dalam Any.Run, seperti pada Gambar 4.45, tercatat sebanyak 2990 aktivitas, namun tidak ditemukan entri

perubahan registry. Hal ini kemungkinan mengindikasikan bahwa malware hellcat.exe melakukan teknik pencurian data berbasis file dan proses sistem dalam satu eksekusi dan segera mengirim hasil curian keluar dengan cepat. Tidak adanya perubahan registry membuat HellCat tidak membentuk persistensi atau melakukan manipulasi sistem.

4. Aktivitas Jaringan

Tabel 4. 34 Aktivitas Jaringan HellCat

Jenis Aktivitas	Proses	Domain/IP Tujuan	Port/ Protocol	Keterangan
HTTP Request	SIHClient.exe	www.microsoft.com	80 / HTTP	Permintaan CRL
HTTPS Connection	svchost.exe	settings-win.data.microsoft.com	443 / HTTPS	Sambungan sah ke Microsoft
DNS Request	-	ocsp.digicert.com	DNS	Validasi sertifikat digital
DNS Request	-	login.live.com	DNS	Autentikasi akun Microsoft

Berdasarkan tabel 4.34, menunjukkan mengenai hasil aktivitas jaringan pada sampel hellcat.exe tidak menunjukkan adanya komunikasi ke domain atau IP yang tidak dikenal maupun berbahaya. Seluruh aktivitas jaringan yang terekam selama proses eksekusi menunjukkan komunikasi yang bersifat normal dan sah, seperti permintaan sertifikat digital melalui protokol OCSP/CRL, dan koneksi ke domain Microsoft terkait pembaruan sistem atau autentikasi layanan. Proses yang terlibat dalam komunikasi ini meliputi svchost.exe dan SIHClient.exe, yang merupakan komponen sah milik Windows.

Seluruh domain yang diakses, termasuk crl.microsoft.com, login.live.com, serta settings-win.data.microsoft.com, telah diverifikasi dan tercatat sebagai

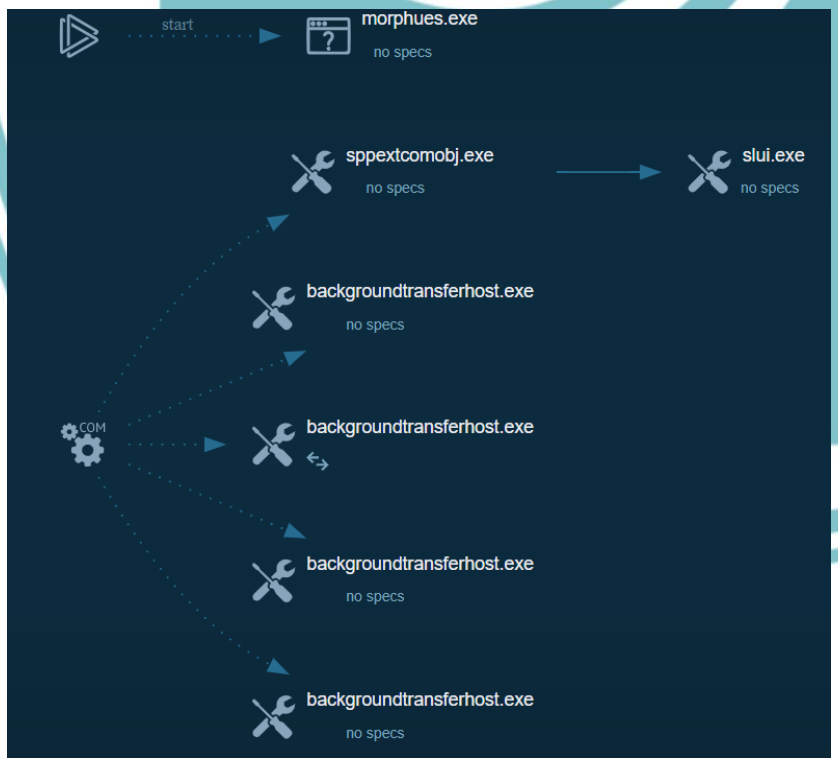
domain yang telah masuk dalam daftar aman (*whitelisted*). Selain itu, permintaan DNS juga tidak menunjukkan adanya upaya penyambungan ke command and control server (C2), yang biasanya menjadi salah satu ciri khas utama aktivitas *malware stealer* seperti Hellcat.

Dengan demikian, berdasarkan data yang tersedia melalui analisis pada Any.Run, tidak ditemukan bukti bahwa hellcat.exe melakukan aktivitas jaringan yang berbahaya dalam environment virtual ini. Namun, hasil ini tidak menutup kemungkinan bahwa *malware* memiliki mekanisme anti-VM atau evasion lainnya.

4.4.3.6 Analisis Dinamis Morpheus

Hasil dari analisis dinamis terhadap sampel ransomware Morpheus.exe menunjukkan bahwa malware tersebut melakukan sejumlah aktivitas berbahaya, diantaranya:

1. Struktur Proses



Gambar 4. 46 Behaviour Graph Morpheus

Berdasarkan Gambar 4.46, merupakan pola eksekusi yang diciptakan selama proses runtime oleh sampel ransomware Morpheus.exe. Proses utama merupakan Morpheus.exe yang bertindak sebagai file pertama dan kemudian memanggil proses-proses lain (*initial dropper* dan *loader*). Setelah berjalan, Morpheus.exe memicu eksekusi sppextcomobj.exe, sebuah file legal Windows yang digunakan untuk aktivasi lisensi, namun proses ini memicu slui.exe sebuah file executable yang tergolong *living off the land Binary* (LOLBins), yang merupakan program bawaan atau sah milik sistem operasi yang disalahgunakan oleh malware untuk menjalankan aktivitas berbahaya.

Selain itu, malware ini juga menjalankan beberapa instance backgroundtransferhost.exe melalui mekanisme *Component Object Model* (COM), yang menunjukkan indikasi penyalahgunaan *Background Intelligent Transfer Service* (BITS) untuk komunikasi atau pengunduhan data secara tersembunyi. Strategi ini mencerminkan teknik *defense evasion* dan *persistence*, di mana malware menyamarkan aktivitasnya melalui executable sah untuk menghindari deteksi.

2. Modifikasi File

Berdasarkan hasil analisis, sampel Morpheus.exe membuat file di dalam sistem, yang terjadi dikarenakan proses yang dilakukan oleh BackgroundTransferHost.exe, diantaranya:

1. ...ContentDeliveryManager...\BackgroundTransferApi*.down_data
2. ...ContentDeliveryManager...\BackgroundTransferApi*.down_meta
3. ...ContentDeliveryManager...\BackgroundTransferApi*.up_meta_secure
4. ...\CryptnetUrlCache\Content…
5. ...\CryptnetUrlCache\MetaData...

Folder BackgroundTransferApi menunjukkan bahwa malware memanfaatkan layanan *Background Intelligent Transfer Service* (BITS) untuk mengunduh atau mengunggah data secara tersembunyi. File yang tersimpan di sini, seperti .down_data dan .down_meta, kemungkinan merupakan file tambahan yang diunduh oleh malware dari sumber eksternal,

yang kemudian digunakan untuk melanjutkan proses infeksi, seperti melakukan enkripsi data, atau menjalin koneksi dengan server *Command and Control* (C2).

Sementara itu, folder CryptnetUrlCache dan MetaData digunakan untuk menyimpan informasi tentang sertifikat digital. Dalam kasus malware, folder ini bisa dimanfaatkan untuk menyimpan file kecil, konfigurasi, atau data hasil pencurian secara tersembunyi. Hal ini dilakukan agar file berbahaya tidak mudah terdeteksi oleh antivirus.

3. Perubahan Registry

Tabel 4. 35 Perubahan Registry Morpheus

Path Registry	Operasi	Keterangan
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\Content	Write	Digunakan untuk menyimpan metadata korban
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\Cookies	Write	Digunakan untuk menyisipkan cookie pengguna
HKEY_CLASSES_ROOT \...\Internet Settings\Cache\History	Write	Digunakan untuk mencatat riwayat halaman yang pernah dikunjungi pengguna

Berdasarkan tabel 4.35, proses BackgroundTransferHost.exe melakukan penulisan registry yang berkaitan dengan cache browser, termasuk metadata konten, cookies, dan history. Hal ini menunjukkan adanya upaya untuk mengakses atau menyimpan sesi aktivitas pengguna, yang menjadi ciri dari *stealer behaviour.*

4. Aktivitas Jaringan

Tabel 4. 36 Aktivitas Jaringan Morpheus

Proses	IP Tujuan	Domain	Port
BackgroundTransferHost	184.30.131.245 92.123.104.53	www.bing.com, ocsp.digicert.com	80 443
svchost.exe	40.113.110.67 20.190.160.22	client.wns.windows.com login.live.com	443
backgroundTaskHost.exe	20.103.156.88	arc.msn.com	443
SIHClient.exe	4.245.163.56	slscr.update.microsoft.com	443

Berdasarkan tabel 4.36, menunjukkan mengenai aktivitas jaringan Morpheus.exe menunjukkan pola koneksi yang sangat mirip dengan malware lainnya, yakni mengarah ke domain-domain sah milik Microsoft. Meskipun domain tersebut bersifat whitelist, komunikasi dilakukan menggunakan protokol TLS/HTTPS yang bisa saja digunakan untuk menyamarkan aktivitas eksfiltrasi atau kontrol dari server C2.

4.4.3.7 Analisis Dinamis Prince

Hasil dari analisis dinamis terhadap sampel ransomware Prince-Built.exe menunjukkan bahwa malware tersebut melakukan sejumlah aktivitas berbahaya, diantaranya:

1. Struktur proses

Gambar 4. 47 Behaviour Graph Prince

Berdasarkan gambar 4.47, merupakan hasil analisis dari malware prince-built.exe yang menunjukkan grafik eksekusi proses (*process tree*) selama proses *runtime*, diantaranya:

1. prince-built.exe (THREAT)

   Merupakan proses utama dari malware yang pertama kali dijalankan. Selain itu file ini merupakan *dropper* sekaligus eksekutor ransomware. Berdasarkan ikon dan label "THREAT", hal ini merupakan file yang dikenali sebagai file berbahaya.
2. powershell.exe (Terdiri dari dua instansi file)

   Malware membuat dua proses PowerShell yang menjalankan skrip secara dinamis. Salah satunya digunakan untuk mengunduh *wallpaper* dan mengatur tampilan desktop, dengan menggunakan API SystemParametersInfo dari USER32.dll. Selain itu PowerShell juga digunakan untuk mengkompilasi file C# saat proses eksekusi sistem
3. conhost.exe

   Merupakan *Console Window Host* yaitu proses sah yang digunakan setiap kali PowerShell dijalankan di mode konsol atau terminal.
4. sppextcomobj.exe memicu slui.exe

   Kedua executable ini merupakan binary sah milik sistem operasi Windows yang tergolong *Living-off-the-Land Binaries* (LOLBins). Pemanfaatan LOLBins ini menunjukkan adanya upaya pengaburan jejak dan penghindaran deteksi, karena malware tidak perlu menulis tool atau executable tambahan yang mencurigakan ke dalam sistem.
5. csc.exe memicu cvtres.exe

   Selanjutnya, hasil kompilasi kode dari csc.exe diteruskan melalui proses cvtres.exe, yang merupakan bagian dari pipeline kompilasi C# .NET. Kombinasi ini mengindikasikan bahwa malware menjalankan teknik *fileless execution*, di mana kode berbahaya dibentuk dan dijalankan langsung dari memori, tanpa perlu menyimpan file executable secara permanen di sistem.

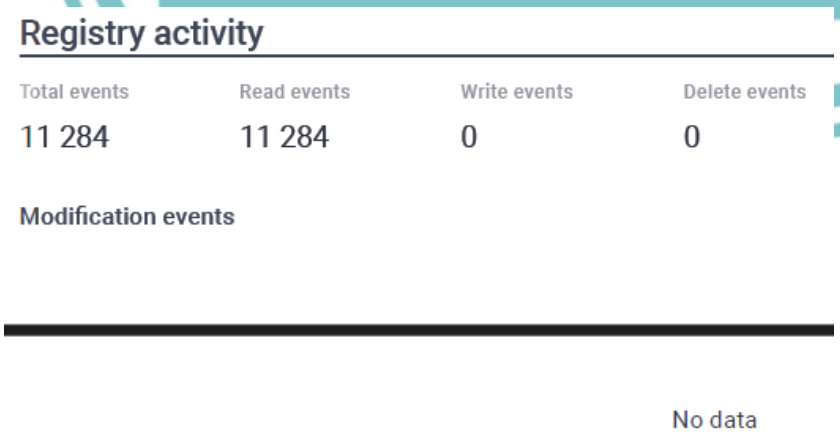
2. Modifikasi File

Tabel 4. 37 Modifikasi File Prince

Perubahan	Path	Keterangan
File Tebusan	C:\Users\admin\Documents\ Decryption Instructions.txt	Disalin dibanyak direktori
File Terenkripsi	availabilityaway.rtf.prince	Indikasi file enkripsi ransomware
File Baru	Wallpaper.png, ma5i1p5m.dll, CSCFBEBC26B...TMP	File unduhan dan kompilasi dinamis (diubah menjadi executable saat runtime)

Berdasarkan tabel 4.37, selama proses eksekusi, malware prince-built.exe menyebabkan perubahan signifikan pada sistem file. File catatan tebusan Decryption Instructions.txt disebarkan ke berbagai direktori pengguna untuk memperjelas pesan ancaman. Selain itu, file-file korban seperti dokumen dan gambar dienkripsi dan diberi ekstensi baru .prince, menandakan telah terjadi proses enkripsi data. Malware juga menghasilkan file baru seperti Wallpaper.png dan file sementara hasil kompilasi C# secara dinamis, yang menunjukkan digunakannya teknik eksekusi *fileless* guna menghindari deteksi.

3. Perubahan Registry



Gambar 4. 48 Perubahan Registry Prince

Berdasarkan hasil analisis aktivitas registry pada Gambar 4.48, menunjukkan bahwa malware prince-built.exe tidak melakukan modifikasi terhadap entri registry, namun tercatat melakukan lebih dari 11.000 aktivitas *read* atau malware hanya melakukan pembacaan terhadap registry dan tidak melakukan perubahan (*write* dan *delete*). Akses ini digunakan untuk mengumpulkan informasi seperti Machine GUID dan konfigurasi jaringan, yang berfungsi sebagai dasar untuk menjalankan proses infeksi secara dinamis dan tersembunyi. Meskipun tidak ada perubahan langsung, aktivitas ini tetap menunjukkan bahwa malware prince-built.exe memiliki karakteristik anti forensik dan menggunakan teknik evasif.

4. Aktivitas Jaringan

Berdasarkan hasil analisis terhadap koneksi jaringan yang dilakukan selama proses eksekusi oleh malware prince-built.exe, diketahui bahwa:

1. Malware mengunduh file dari URL https://i.imgur.com/..
2. Melakukan koneksi ke berbagai domain sah Microsoft (OCSP, CRL, login.live.com) untuk menyamarkan aktivitas (*cover traffic*)
3. Tidak terdeteksi adanya koneksi eksplisit ke server Command and Control (C2), namun dengan adanya temuan bahwa adanya file yang diunduh melalui PowerShell mengindikasikan potensi komunikasi dengan server eksternal.

4.4.4 Hasil *Reverse Engineering*

4.4.4.1 *Reverse Engineering* Akira

Analisis *reverse engineering* pada sampel akira.exe dimulai dengan membongkar struktur biner melalui proses *disassembly.* Melalui analisis ini didapatkan informasi mengenai:

1. Entry Point dan Inisialisasi

Eksekusi dimulai dari fungsi FUN_14004ca30 sebagai titik awal logika internal melalui *command line argument* setelah runtime. Fungsi ini menginisialisasi struktur internal dan menyimpan konfigurasi seperti:

- --encryption_path → direktori target enkripsi
- --encryption_percent → presentase file yang akan dienkripsi

- --exclude, --share_file, -localonly, dan parameter lainnya

  Selain itu, fungsi ini juga:

- Memvalidasi jumlah CPU,
- Mengalokasikan thread untuk melakukan traversing direktori dan pemrosesan file,
- Menyimpan pointer ke objek-objek penting dalam param_1, yaitu struktur memori utama.

2. Persiapan Data Enkripsi

   Fungsi FUN_1400bdb30 bertanggung jawab untuk menelusuri direktori dan file sesuai parameter yang telah diberikan. Dalam fungsi ini, dilakukan:

   - *Filtering* file berdasarkan ekstensi dan direktori yang di-*exclude*.
   - Penyimpanan path file target ke dalam struktur data yang digunakan saat proses enkripsi
   - Penyusunan informasi metadata seperti nama file dan path secara lengkap,
   - Dan validasi nama direktori (menghindari . dan ..).

   Fungsi ini akan terus mengisi buffer data yang digunakan pada proses enkripsi.

3. Fungsi Proses Enkripsi Utama

   Setelah semua file yang ditarget dikumpulkan, proses dilanjutkan ke fungsi FUN_14007b820. Fungsi ini menerima:

   - param_1 = struktur konteks utama
   - param_2 = pointer ke data/file yang akan dienkripsi
   - param_3, param_4 = flag internal
   - param_5–param_7 = parameter tambahan (timestamp, kunci, atau konfigurasi)

   Fungsi ini melakukan:

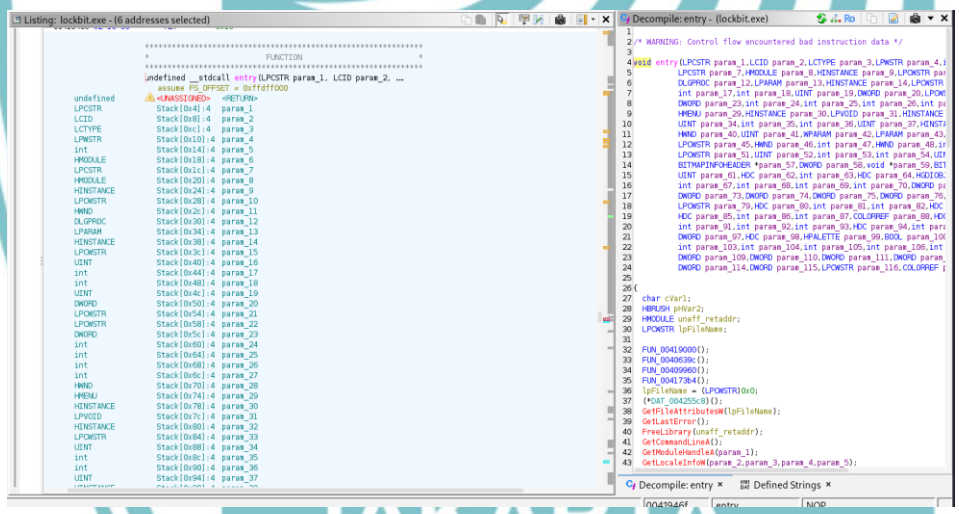
   - Pengecekan validitas slot enkripsi
   - Eksekusi fungsi FUN_14007bab0 untuk mempersiapkan struktur enkripsi
   - Pemanggilan fungsi FUN_14007b0b0 sebagai implementasi utama proses enkripsi.

Fungsi ini memanggil fungsi FUN_14007b0b0 sebagai wrapper enkripsi. Di dalamnya, terdapat pemanggilan ke fungsi FUN_14007c620 dan FUN_140041950, yang menyusun struktur buffer, dan kemudian melakukan enkripsi menggunakan algoritma AES (kemungkinan GCM atau CBC berdasarkan pola alokasi memori, ukuran blok, dan cara manipulasi konten string). Selain itu, file hasil enkripsi disimpan dengan struktur tertentu dan diakhiri dengan *trailing signature* unik sebagai identifikasi.

4.4.4.2 *Reverse Engineering* LockBit

Analisis *reverse engineering* pada sampel blackcat.exe dimulai dengan membongkar struktur biner melalui proses *disassembly.* Melalui analisis ini didapatkan informasi mengenai:

1. Entry Point



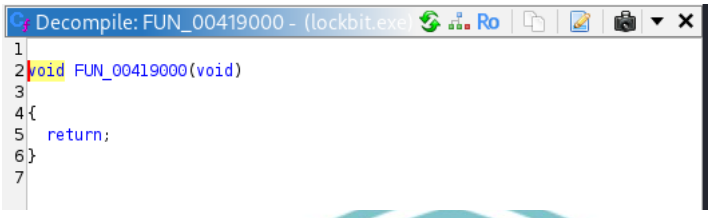
Gambar 4. 49 Entry Point LockBit

Berdasarkan Gambar 4.49, merupakan entry point pada sampel ransomware LockBit. Setelah dianalisis, fungsi ini langsung melakukan inisialisasi dan memanggil beberapa fungsi salah satunya FUN_00419000. Tidak banyak logika pada entry point, perannya lebih sebagai pengarah ke fungsi utama yang menjadi pusat kendali eksekusi LockBit. Seluruh variabel hanya memanggil *import API.*

2. Fungsi Logika

Berdasarkan hasil analisis lanjutan dari entry point, dapat ditelurusi fungsi logika penyusun, diantaranya:

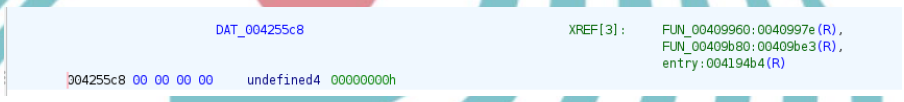
1. FUN_00419000



Gambar 4. 50 Fungsi FUN_00419000 LockBit

Berdasarkan Gambar 4.50, fungsi FUN_00419000 merupakan fungsi dummy atau kosong. Fungsi ini mengindikasikan sebagai bagian dari struktur *obfuscation*.

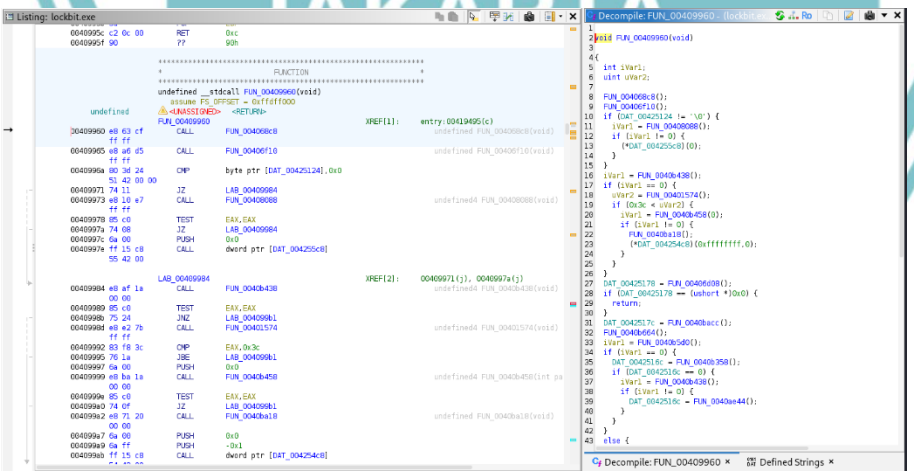
2. DAT_004255C8



Gambar 4. 51 Variabel DAT_004255C8 LockBit

Terlihat pada gamber 4.51, DAT_004255C8 berisikan nilai awal yang kosong (0x00000000), artinya pointer tersebut belum menunjuk ke fungsi manapun saat awal dijalankan. Namun, terdapat tida lokasi dalam kode yang mengakes (membaca) pointer ini, yaitu FUN_00409960, FUN_00409b80, dan entry:004194b4. Sehingga kemungkinan nilai pointer tersebut akan di set oleh fungsi-fungsi tersebut.

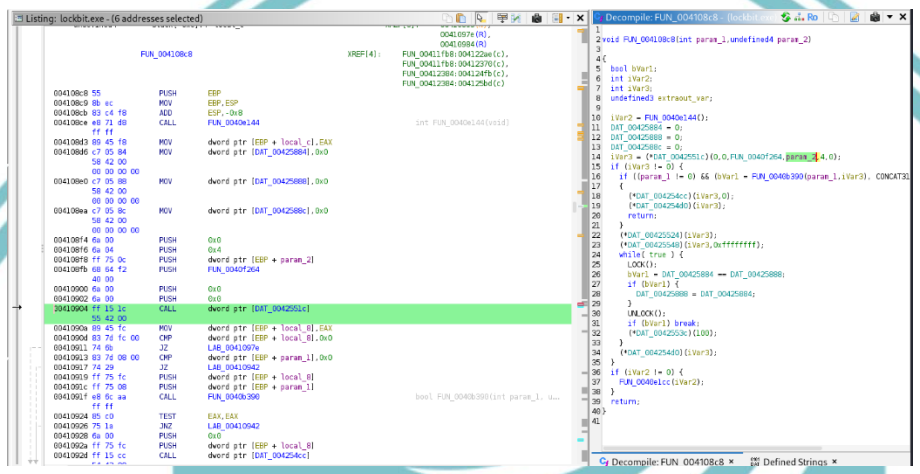
3. FUN_00409960



Gambar 4. 52 Fungsi FUN_00409960 LockBit

Seperti yang terlihat pada Gambar 4.52, fungsi FUN_00409960 merupakan bagian dari *unpacking routine* dari ransomware LockBit yang dimana fungsi ini memanggil banyak fungsi internal yang bertujuan untuk mempersiapkan lingkungan, eksekusi payload via pointer DAT_004255C8, dan fungsi lain yang kemungkinan digunakan sebagai API *Resolving*.

4. FUN_004108C8



Gambar 4. 53 Fungsi FUN_004108c8 LockBit

Berdasarkan Gambar 4.53, fungsi FUN_004108c8 merupakan fungsi yang bertindak sebagai eksekutor thread. Fungsi ini bertanggung jawab menjalankan sebuah thread baru dengan memanfaatkan fungsi callback yaitu FUN_0040F264. Thread tersebut dijalankan secara sinkron, artinya proses utama akan menunggu hingga thread menyelesaikan tugasnya sebelum melanjutkan eksekusi selanjutnya. Hal ini dibuktikan dengan penggunaan dua variabel global shared: DAT_00425884 dan DAT_00425888, yang kemungkinan digunakan sebagai flag status penyelesaian atau komunikasi antar-thread.

Fungsi ini juga melakukan validasi terhadap status thread. Apabila thread gagal dijalankan atau diverifikasi sebagai tidak valid melalui pemanggilan fungsi FUN_0040B390, maka thread tersebut akan dihentikan secara paksa. Validasi ini menunjukkan adanya mekanisme kontrol terhadap eksekusi paralel, yang menjadi penting dalam

konteks ransomware seperti LockBit, karena eksekusi multi-thread sering digunakan untuk mempercepat proses enkripsi terhadap banyak file.

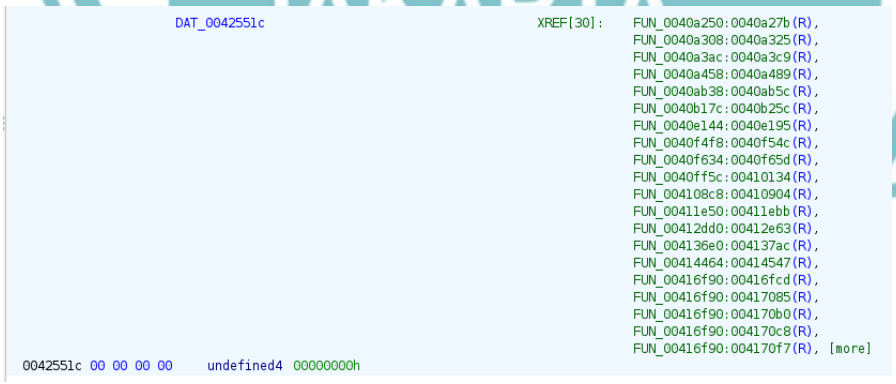
5. FUN_00411FB8

   Fungsi ini berfungsi sebagai loader atau dekriptor terhadap modul internal yang berisi pointer-pointer API atau struktur fungsi. Ia menerima parameter yang diasumsikan mengarah pada blok data terenkripsi, lalu melakukan proses dekripsi internal, kemungkinan menggunakan algoritma XOR atau substitusi sederhana (meskipun belum teridentifikasi jelas karena tidak dilakukan debugging).

   Hasil dekripsi dari fungsi ini adalah sebuah struktur (berbentuk array pointer atau function table) yang kemudian disimpan ke dalam variabel global seperti DAT_004254F0 dan DAT_004254F4. Kedua variabel ini kemudian dimanfaatkan oleh fungsi lain seperti FUN_004108C8 sebagai basis eksekusi terhadap fungsi-fungsi internal secara dinamis.

   Peran fungsi ini sangat penting karena mengaburkan jejak API asli Windows atau fungsi internal ransomware. Dengan menyimpan pointer dalam keadaan terenkripsi, ransomware dapat menghindari deteksi signature statik dan hanya memanggil fungsi sesungguhnya setelah didekripsi.

6. DAT_0042551C



Gambar 4. 54 Variabel DAT_0042551C LockBit

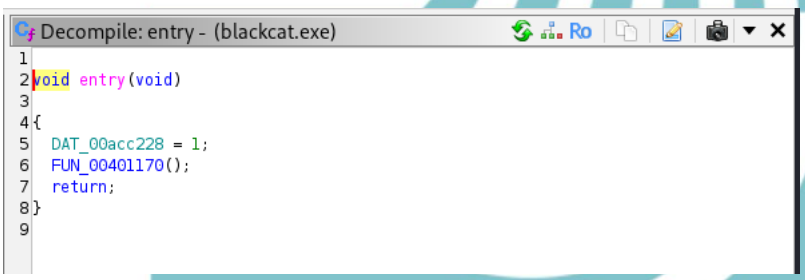
Terlihat pada Gambar 4.54, DAT_0042551C banyak dipanggil oleh beberapa fungsi. Variabel ini merupakan bagian dari struktur

parameter atau konfigurasi yang digunakan di runtime. Sayangnya, karena keterbatasan pada analisis statik dan tidak muncul di memory map, isi aktual dari DAT_0042551C tidak dapat dikonfirmasi secara pasti tanpa analisis dinamis. Namun, dari konteksnya, besar kemungkinan variabel ini digunakan untuk memuat kunci, buffer enkripsi, atau referensi ke resource tersembunyi.

4.4.4.3 *Reverse Engineering* BlackCat

Analisis *reverse engineering* pada sampel blackcat.exe dimulai dengan membongkar struktur biner melalui proses *disassembly.* Melalui analisis ini didapatkan informasi mengenai:

3. Entry Point dan Struktur Awal



Gambar 4. 55 Entry Point BlackCat

Berdasarkan Gambar 4.55, menunjukkan bahwa *entry point* mengarah ke fungsi FUN_00401170(). Fungsi ini bertindak sebagai awal dari proses ekskusi, yang bertanggung jawab menyiapkan lingkungan eksekusi. Fungsi ini memanggil FUN_008e8020, yang bertanggung jawab untuk mempersiapkan struktur eksekusi, memanggil TLS callbacks, hingga inisialisasi mutex yang akan digunakan untuk memastikan bahwa hanya satu instansi malware berjalan dalam satu sistem.

4. *Parsing* Konfigurasi dan *Dispatch* Enkripsi

Selanjutnya, alur program berpindah ke fungsi pemrosesan utama, yang secara bertahap mengarahkan eksekusi menuju fungsi FUN_004016a0. Di dalam fungsi ini, ransomware menjalankan fungsi lain yaitu FUN_00415f50, yang berperan penting sebagai jembatan antara inisialisasi awal dan proses parsing konfigurasi internal.

Fungsi FUN_00415f50 kemudian memanggil fungsi FUN_0041bf70. Fungsi ini merupakan komponen utama yang melakukan parsing terhadap data konfigurasi yang dituliskan dalam format blok kurung kurawal (“{}”). Proses parsing dilakukan secara manual terhadap karakter yang terbaca, dan berdasarkan pola atau nilai-nilai spesifik, program menentukan bagaimana objek tersebut diproses lebih lanjut.

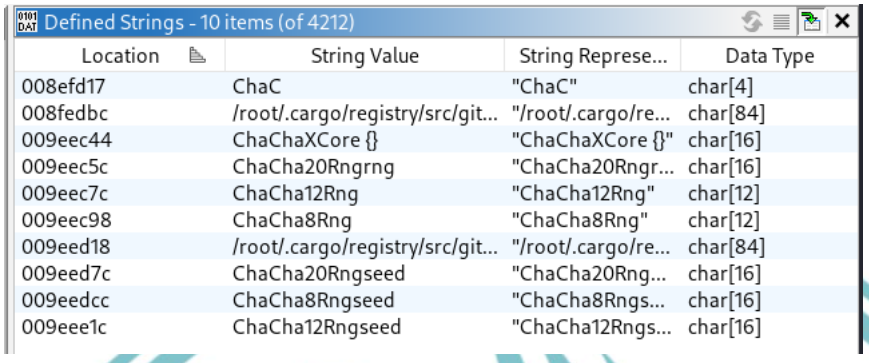
Decompile: FUN_0041bf70 - (blackcat.exe)

```
LAB_0041bff4:
        if (cVar2 != '{') {
          local_20 = 10;
          goto LAB_0041bfc8;
        }
        piVar1 = param_2 + 6;
        *(char *)piVar1 = (char)*piVar1 + -1;
        if ((char)*piVar1 == '\0') {
          local_20 = 0x15;
          goto LAB_0041bfc8;
        }
        param_2[2] = uVar4;
        FUN_004167b0();
        if ((char)local_20 != '\0') goto LAB_0041c016;
        local_1c = FUN_004143f0();
        if (local_1c != 0) goto LAB_0041c016;
        if (local_20._1_1_ == '\0') {
          local_1c = FUN_00417910(param_2);
          if (local_1c != 0) goto LAB_0041c016;
          local_24 = 0;
        }
        else if (local_20._1_1_ == '\x01') {
          local_1c = FUN_00417910(param_2);
          if (local_1c != 0) goto LAB_0041c016;
          local_24 = 1;
        }
        else {
          local_1c = FUN_00417910(param_2);
          if (local_1c != 0) goto LAB_0041c016;
          local_24 = 2;
        }
        *(char *)(param_2 + 6) = (char)param_2[6] + '\x01';
        local_14 = param_2[1];
        uVar4 = param_2[2];
        if (local_14 <= uVar4) goto LAB_0041c0ee;
        local_14 = -local_14;
        goto LAB_0041c0c0;
      }
```

Gambar 4. 56 Fungsi FUN_0041bf70 BlackCat

Selama proses parsing tersebut, ketika ditemukan karakter khusus seperti ‘{’ atau pola tertentu, program akan memanggil fungsi FUN_004167b0. Seperti yang terlihat pada Gambar 4.56. Fungsi ini bertugas untuk memverifikasi dan mengurai struktur dalam objek konfigurasi tersebut. Di dalamnya, program melakukan pemeriksaan terhadap entri-entri konfigurasi, seperti “Best”, “ChaC”, atau “Ae”, yang telah di-obfuscate dan perlu dibuka kuncinya (melalui XOR dan OR terhadap string yang telah dienkripsi sederhana). Proses ini juga mengatur mode operasi dari ransomware,

termasuk parameter enkripsi yang akan digunakan. Fungsi FUN_004167b0 secara tidak langsung berperan dalam inisialisasi komponen enkripsi.



Gambar 4. 57 String Algoritma Enkripsi BlackCat

Berdasarkan gambar 4.57, ditemukan beberapa string yang mengindikasikan penggunaan algoritma ChaCha20 dalam mekanisme enkripsi. String tersebut menunjukkan jalur sumber kode (*source path*) serta identifikasi struktur internal dari pustaka chacha20 dan rand_chacha yang umum digunakan dalam pemrograman bahasa Rust.

Berdasarkan string seperti "ChaChaXCore {}", "ChaCha20Rng", hingga path build "/root/.cargo/registry/src/github.com-1ecc6299db9ec823/chacha20-0.8.2/src/chacha.rs", dapat disimpulkan bahwa binary ransomware ini dikompilasi menggunakan crate pustaka chacha20 versi 0.8.2 dan rand_chacha versi 0.3.1. Crate tersebut menyediakan implementasi dari cipher ChaCha20 dan generator acak berbasis ChaCha.

Dengan demikian, terdapat indikasi kuat bahwa algoritma ChaCha20 digunakan dalam proses:

- Enkripsi file (stream cipher),
- Pembuatan nonce atau IV (Initialization Vector),
- RNG (Random Number Generator) untuk kunci sementara.

Indikasi ini juga diperkuat dengan keberadaan string-string seperti "ChaC", "ChaCha8Rngseed", dan "ChaCha12Rng" yang secara langsung mengacu pada varian dari algoritma ChaCha.

Dengan demikian, dapat disimpulkan bahwa ransomware BlackCat menggunakan pendekatan modular dalam menentukan konfigurasi dan mode enkripsi, dan proses tersebut dimulai dari entry point hingga ke tahap parsing dan pemrosesan konfigurasi melalui serangkaian fungsi penting yang saling terhubung.

4.4.4.4 *Reverse Engineering* WhisperGate

Analisis reverse engineering pada sampel whispergate.exe dimulai dengan membongkar struktur biner melalui proses disassembly. Melalui analisis ini didapatkan informasi mengenai:

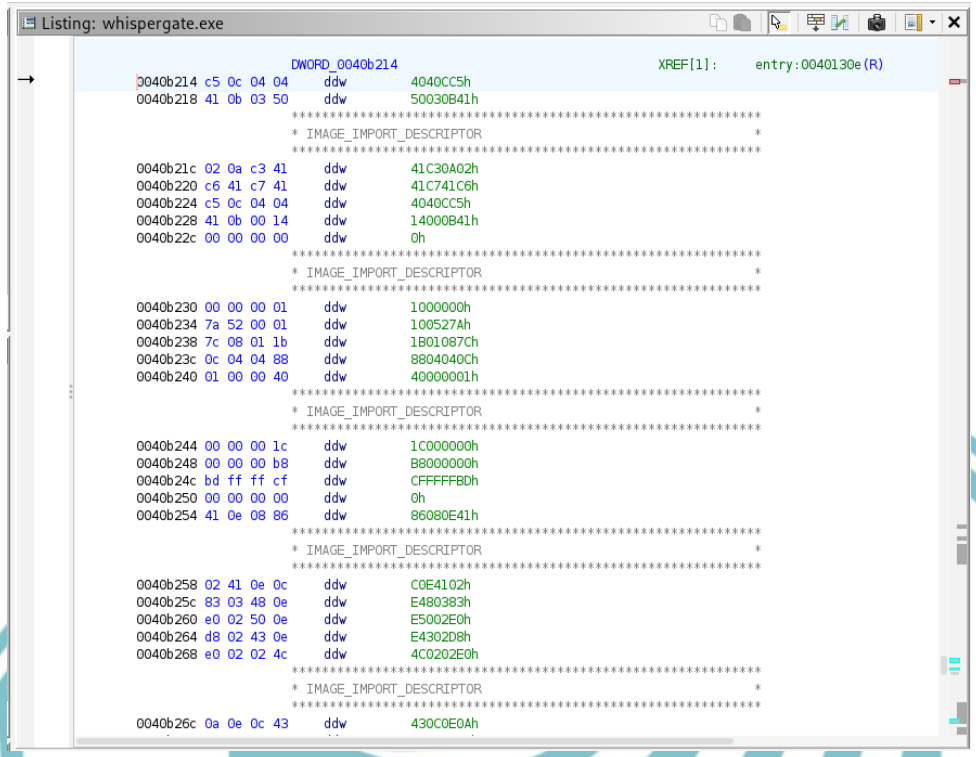
1. *Entry Point*

```
Decompile: entry - (whispergate.exe)
1
2 /* WARNING: Control flow encountered bad instruction data */
3
4 void entry(void)
5
6 {
7   (*(code *)DWORD_0040b214)(1);
8   func_0x3ac1d6d4();
9                     /* WARNING: Bad instruction - Truncating control flow here */
10  halt_baddata();
11 }
12
```

Gambar 4. 58 Entry Point WhisperGate

Berdasarkan Gambar 4.58, yang merupakan entry point pada sampel whispergate.exe. Struktur ini menunjukkan bahwa malware tidak memulai eksekusi dengan cara konvensional seperti pemanggilan main atau WinMain, melainkan langsung melakukan eksekusi terhadap pointer dinamis yang tersimpan pada alamat DWORD_0040b214. Setelah itu, dieksekusi fungsi dengan offset acak dan diakhiri dengan instruksi yang tidak valid (halt_baddata()), yang menjadi indikasi awal teknik obfuscation atau packing.

2. Analisis Pointer Awal



Gambar 4. 59 Pointer DWORD_0040b214 WhisperGate

Terlihat pada Gambar 4.59, yang merupakan pointer awal yang dipanggil dalam entry yaitu DWORD_0040b214. Nilai DWORD_0040b214 adalah 0x4040CC5, yaitu alamat dalam segment .text yang tidak mengarah ke simbol fungsi bernama, melainkan ke blok data yang terdiri dari nilai 4-byte acak (dd). Data ini kemungkinan besar adalah bagian dari tabel fungsi terenkripsi atau payload yang didekripsi saat runtime. Tidak dditemukannya string yang valid maupun referensi langsung ke API Windows, memperkuat dugaan bahwa malware ini menggunakan *runtime loader* atau *decryptor internal*

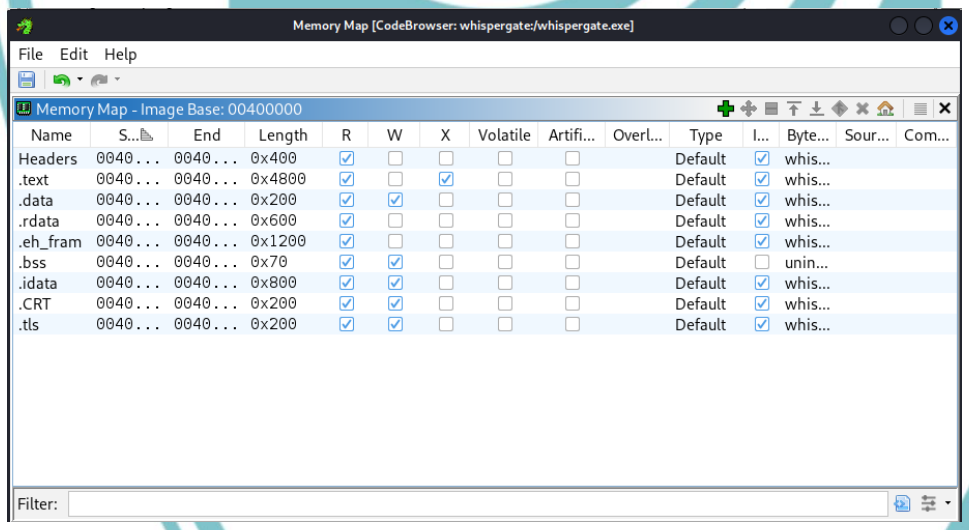
3. Fungsi Loader

```
00401314 e8 bb c3        CALL        SUB_3acld6d4
         81 3a
00401319 de ff           FDIVP       ST7,ST0
0040131b f0              align       align(1)
0040131c e0              ??          E0h
0040131d 4e              ??          4Eh    N
0040131e b7              ??          B7h
0040131f 8d              ??          8Dh
00401320 74              ??          74h    t
00401321 26              ??          26h    &
00401322 00              ??          00h
00401323 8d              ??          8Dh
00401324 bc              ??          BCh
00401325 27              ??          27h    '
00401326 00              ??          00h
00401327 00              ??          00h
00401328 00              ??          00h
00401329 00              ??          00h
0040132a 83              ??          83h
0040132b ec              ??          ECh
```

Gambar 4. 60 Fungsi UndefinedFunction_0040856f WhisperGate

Selain pointer yang sebelumnya, di dalam entry juga dipanggil fungsi UndefinedFunction_0040856f seperti pada Gambar 4.60. Namun, fungsi tersebut tidak bisa dianalisis. Sehingga dugaan kuat bahwa malware ini menulis ulang bagian *executable* saat runtime.

4. Struktur Memori



Gambar 4. 61 Struktur Memori WhisperGate

Berdasarkan Gambar 4.61, merupakan struktur segmen memori dari sampel whispergate.exe. Terlihat bahwa .text memiliki panjang 0x4800 byte dan bersifat executable, sedangkan .rdata dan .idata aktif namun tidak banyak string terdefinisi, menandakan kemungkinan teknik *obfuscation* atau packing. Tampilan ini mendukung temuan bahwa malware

menyembunyikan payload utamanya dan memerlukan analisis dinamis untuk membongkarnya.

4.4.4.5 *Reverse Engineering* HellCat

Analisis reverse engineering pada sampel hellcat.exe dimulai dengan membongkar struktur biner melalui proses disassembly. Melalui analisis ini didapatkan informasi mengenai:

5. *Entry Point*

Gambar 4. 62 Entry Point HellCat

Fungsi _rt0_amd64_windows merupakan entry point utama yang dieksekusi saat program Go pertama kali dijalankan di sistem Windows berbasis arsitektur AMD64. Fungsi ini tertulis dalam bahasa Assembly dan tidak secara langsung ditulis oleh pengembang aplikasi, melainkan menjadi bagian dari runtime bawaan Go. Dalam proses eksekusinya, fungsi ini akan langsung memanggil runtime.rt0_go, yang kemudian akan menginisialisasi runtime Go, termasuk pengelolaan memori, goroutine, dan sistem internal lainnya sebelum akhirnya mengeksekusi fungsi main() dari aplikasi.

6. Proses Eksekusi

   Setelah fungsi entry point berhasil memanggil main.main, ransomware HCLock mulai menjalankan serangkaian prosedur untuk mempersiapkan proses enkripsi. Di antaranya:

   1. Fungsi main.main

      Pada tahap awal, main.main bertanggung jawab untuk menginisialisasi kunci simetris (symKey) dan nonce yang akan digunakan sebagai parameter enkripsi berbasis ChaCha20. Selain itu,

fungsi ini juga akan melakukan enumerasi file pada sistem target, memilih file yang sesuai berdasarkan ekstensi tertentu, dan menyiapkannya untuk proses enkripsi dengan membagi isi file ke dalam unit-unit data yang lebih kecil (*chunk*).

7. Fungsi main.encryptFile

   Selanjutnya, untuk setiap file yang telah dibagi menjadi *chunk*, proses diteruskan ke fungsi main.encryptFile. Fungsi ini membaca isi file secara parsial dan mengirimkan setiap *chunk* ke dalam saluran komunikasi (*channel*) untuk kemudian diproses secara asinkron oleh *goroutine*.

8. Fungsi main.encryptFile.func2

   Selanjutnya fungsi main.encryptFile.func2 berperan sebagai *dispatcher* yang menerima data *chunk* dari *channel* input dan memanggil fungsi main.processChunkOptimized guna melakukan proses enkripsi terhadap chunk tersebut.

9. Fungsi main.processChunkOptimized

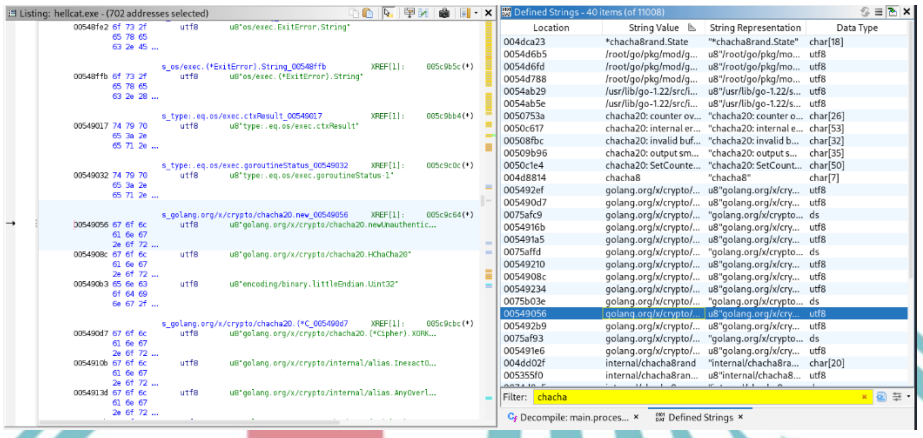
   Fungsi main.processChunkOptimized bertugas untuk memanfaatkan kemampuan paralelisme sistem dengan membagi data chunk menjadi beberapa segmen sesuai jumlah CPU core yang tersedia. Untuk setiap segmen, fungsi ini membuat goroutine baru yang akan menjalankan fungsi main.processChunkOptimized.func1. Tujuan dari pembagian ini adalah untuk mempercepat proses enkripsi dengan menjalankannya secara bersamaan (concurrent).

   Fungsi main.processChunkOptimized.func1 kemudian bertugas sebagai pembungkus (wrapper) yang meneruskan argumen data, batas awal dan akhir segmen, serta parameter enkripsi ke fungsi main.processSegmentOptimized.

10. Fungsi main.processSegmentOptimized

    Fungsi main.processSegmentOptimized menginisialisasi cipher ChaCha20 dengan menggunakan symKey dan nonce, lalu menerapkan operasi XOR antara data asli dengan keystream untuk menghasilkan ciphertext. Seluruh proses ini dilakukan secara in-place, artinya data

yang telah dienkripsi langsung menggantikan data aslinya di dalam memori.



Gambar 4. 63 Algoritma Enkripsi HellCat

Penggunaan algoritma enkripsi ChaCha20 dibuktikan juga melalui pencarian string seperti pada Gambar 4.63.

Dengan demikian, seluruh rangkaian fungsi dari main.main hingga main.processSegmentOptimized membentuk satu jalur proses yang terstruktur untuk mengenkripsi file korban secara paralel dan menyeluruh.

4.4.4.6 *Reverse Engineering* Morpheus

Analisis *reverse engineering* pada sampel Morpheus.exe dimulai dengan membongkar struktur biner melalui proses disassembly. Melalui analisis ini didapatkan informasi mengenai:

1. Inisialisasi dan Persiapan Memori

```
void FUN_140001db0(LPCWSTR param_1)

{
  undefined1 *lpMem;
  LPVOID pvVar1;
  HANDLE *lpHandles;
  HANDLE pvVar2;
  uint local_38;
  uint local_34;

  lpMem = (undefined1 *)HeapAlloc(DAT_140006010,8,0x40);
  if (lpMem != (undefined1 *)0x0) {
    pvVar1 = HeapAlloc(DAT_140006010,8,(ulonglong)DAT_140006038 << 3);
    *(LPVOID *)(lpMem + 0x10) = pvVar1;
    pvVar1 = HeapAlloc(DAT_140006010,8,(ulonglong)DAT_140006038 << 3);
    *(LPVOID *)(lpMem + 8) = pvVar1;
    lpHandles = (HANDLE *)HeapAlloc(DAT_140006010,8,(ulonglong)DAT_140006000 << 3);
    InitializeCriticalSection((LPCRITICAL_SECTION)(lpMem + 0x18));
    pvVar2 = GetCurrentThread();
    SetThreadPriority(pvVar2,2);
    local_38 = 0;
    while (local_38 < DAT_140006000) {
      pvVar2 = CreateThread((LPSECURITY_ATTRIBUTES)0x0,0,FUN_140002ce0,lpMem,0,
(LPDWORD)0x0);
      lpHandles[local_38] = pvVar2;
      if (pvVar2 != (HANDLE)0x0) {
        SetThreadPriority(lpHandles[local_38],2);
        local_38 = local_38 + 1;
      }
    }
    FUN_140002060(param_1,(longlong)lpMem);
    Sleep(2000);
    *lpMem = 1;
    pvVar2 = GetCurrentThread();
    SetThreadPriority(pvVar2,-1);
    WaitForMultipleObjects(DAT_140006000,lpHandles,1,0xffffffff);
    for (local_34 = 0; local_34 < local_38; local_34 = local_34 + 1) {
      if (lpHandles[local_34] != (HANDLE)0x0) {
        CloseHandle(lpHandles[local_34]);
      }
    }
    HeapFree(DAT_140006010,0x10000,lpHandles);
    DeleteCriticalSection((LPCRITICAL_SECTION)(lpMem + 0x18));
    HeapFree(DAT_140006010,0x10000,*(LPVOID *)(lpMem + 8));
    HeapFree(DAT_140006010,0x10000,*(LPVOID *)(lpMem + 0x10));
    HeapFree(DAT_140006010,0x10000,lpMem);
  }
  return;
}
```

Gambar 4. 64 Entry Point Morpheus

Seperti yang tertera pada Gambar 4.64, proses dimulai pada fungsi utama yang bertanggung jawab untuk menyiapkan sumber daya memori, struktur data, dan pembuatan thread. Fungsi FUN_140001db0 mengalokasikan beberapa buffer melalui HeapAlloc, termasuk dua array pointer yang digunakan sebagai antrian kerja (*work queue*) untuk menyimpan daftar file dan ukuran file yang akan dienkripsi. Critical Section diinisialisasi sebagai mekanisme sinkronisasi antar thread. Kemudian, sejumlah thread dibuat menggunakan *CreateThread*, di mana pada masing-masing thread akan menjalankan fungsi enkripsi secara paralel (FUN_140002ce0). Setelah semua thread siap, fungsi FUN_140002060 dipanggil untuk memulai proses traversal direktori.

2. Penentuan Target File

Fungsi FUN_140002060 bertanggung jawab untuk menavigasi direktori dan mengumpulkan file target yang akan dienkripsi. Fungsi ini menggunakan FindFirstFileW dan FindNextFileW untuk mencari semua entri dalam direktori tertentu. Selama proses ini, dilakukan pengecualian terhadap nama file tertentu seperti "." dan "..", serta pengecekan nama "*README*.txt" (file catatan ransom).

Setiap file akan diperiksa ekstensinya. Jika ekstensi file tidak termasuk dalam whitelist (seperti .exe, .dll, dsb) dan juga tidak termasuk dalam blacklist (yang dikecualikan secara eksplisit), maka file tersebut dianggap valid untuk dienkripsi. Informasi tentang file yang valid kemudian dimasukkan ke dalam buffer kerja (work queue) berupa nama path dan ukuran file, dengan akses dilakukan secara eksklusif melalui Critical Section agar tidak terjadi race condition antar thread.

3. Eksekusi Proses Enkripsi Secara Paralel

Setiap thread yang menjalankan fungsi FUN_140002ce0 akan terus melakukan polling terhadap buffer kerja untuk mengambil file yang perlu diproses. Proses ini dilakukan dalam loop tanpa henti, selama penanda penghentian belum diset (flag = 1). Pada setiap iterasi, thread akan mengakses antrian buffer secara sinkron, mengambil maksimal 25 file sekaligus (atau hingga antrian kosong), lalu menjalankan proses enkripsi untuk tiap file dengan memanggil fungsi FUN_140002020.

Fungsi tersebut merupakan wrapper yang kemudian memanggil FUN_1400013f0 untuk melakukan proses enkripsi sesungguhnya. Setelah proses selesai, memori untuk path file akan dibebaskan (HeapFree).

4. Proses Enkripsi File

Proses utama enkripsi dilakukan dalam fungsi FUN_1400013f0. Berikut adalah tahapan enkripsinya:

1. Signature Check

   Fungsi ini memeriksa apakah file sudah memiliki marker khusus di bagian akhir (*trailing signature*). Jika ditemukan signature 13-byte tertentu, file tidak akan dienkripsi ulang.

2. Key Generation:

   Menggunakan BCryptGenRandom untuk menghasilkan random key dan nonce. Kemudian, BCryptGenerateSymmetricKey digunakan untuk membentuk kunci simetris dengan algoritma AES (dikonfirmasi melalui konteks penggunaan Bcrypt API).

3. XOR Obfuscation:

   Sebelum digunakan, key hasil random akan di-XOR dengan pola counter negatif untuk menyamarkan isi.

4. Data Encryption:

   File dibaca dalam blok-blok (max 4 MB). Setiap blok dienkripsi menggunakan BCryptEncrypt dengan mode CBC dan IV (nonce) yang dihasilkan sebelumnya. Data hasil enkripsi akan ditulis kembali ke posisi semula di file.

5. Footer Injection (Trailing Signature):

   Setelah seluruh data terenkripsi, buffer hasil XOR dari key akan ditulis ke akhir file. Lalu disusul dengan metadata tambahan sepanjang 0x49 byte yang mencakup:

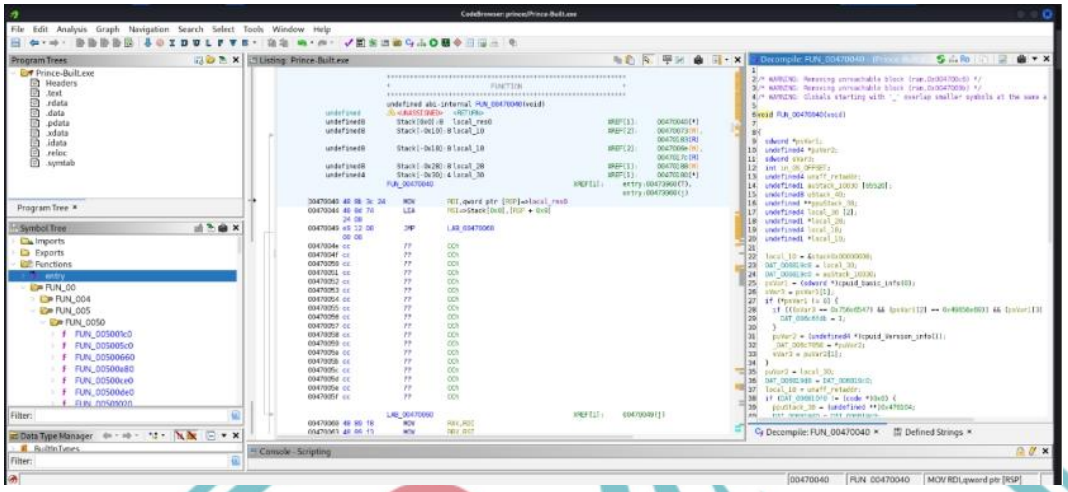
   - Panjang key
   - Nonce
   - Marker signature (13-byte) yang sama seperti signature checker di awal
   - Padding dan struktur pembantu lainnya

Struktur ini memastikan bahwa file yang telah dienkripsi dapat dikenali dengan mudah, serta memungkinkan proses dekripsi di sisi *threat actor* jika memiliki key master.

4.4.4.7 *Reverse Engineering* Prince

Analisis *reverse engineering* pada sampel prince.exe dimulai dengan membongkar struktur biner melalui proses disassembly. Melalui analisis ini didapatkan informasi mengenai:

1. Entry Point



Gambar 4. 65 Entry Point Prince

Berdasarkan Gambar 4.65, ransomware Prince diawali proses eksekusi dari fungsi entry() (FUN_00470040) yang menjalankan beberapa proses sistem awal:

a. Melakukan pengecekan CPUID untuk melihat vendor CPU (Intel).
b. Menyimpan pointer-pointer stack dan alamat penting ke global variabel.
c. Memanggil fungsi seperti FUN_004740a0, FUN_00476fa0, dan FUN_00476f60 sebagai tahap inisialisasi runtime mirip lingkungan Go.
d. FUN_00476fa0 → menuju FUN_0044c940 yang merupakan *sanity check*/validasi terhadap CPU, FPU, dan state memory. Ini sejenis pengecekan apakah lingkungan eksekusi mendukung.

2. Proses Enkripsi

Fungsi utama dalam proses enkripsi adalah FUN_005001c0, yang memanggil FUN_00500660. Di dalam FUN_00500660, proses bercabang bergantung pada kondisi DAT_006c6feb. Bila kondisi terpenuhi, fungsi FUN_00500a80 dipanggil, yang menjadi inti algoritma enkripsi. Langkah penting lainnya dalam enkripsi meliputi:

- FUN_005021c0 dan FUN_00501a60: Proses ini mempersiapkan buffer dan pengaturan internal cipher.
- FUN_004fb660: Melakukan transformasi data menggunakan algoritma berbasis XOR dan/atau AES.

- FUN_00514000 dan FUN_0051a2a0: Berfungsi sebagai transformasi kunci, hashing, atau HMAC.
- FUN_00513d20: Mengatur struktur internal kunci AES dan parameter lainnya, termasuk IV.

3. Algoritma Enkripsi yang Digunakan

   Melalui analisis terhadap string yang ditemukan di biner (seperti crypto/cipher.NewGCMWithNonceSize, crypto/aes.NewCipher, dan golang.org/x/crypto/chacha20.XORKeyStream), diketahui bahwa malware ini menggunakan dua jenis algoritma:

- AES-GCM: Mode authenticated encryption yang populer digunakan untuk keamanan tinggi.
- ChaCha20: Cipher stream modern yang digunakan untuk kecepatan dan keamanan di lingkungan tanpa akselerasi AES.

4. Pembuktian dengan *Source Code* Encrypt.go dan Decrypt.go

   Sebagai pembuktian, file encryption.go dan decryption.go juga dianalisis dan ditemukan kesesuaian fungsi seperti pada tabel 4.38 sebagai berikut:

Tabel 4. 38 Pembuktian Source Code Encrypt.go dan Decrypt.go

Fungsi Reverse (Binary)	Fungsi Source Code (Go)	Penjelasan
FUN_005001c0	encryptFile	Fungsi utama enkripsi file
FUN_00500660	processDataChunks	Proses pengolahan buffer sebelum enkripsi
FUN_00500a80	performEncryption	Inti dari proses enkripsi menggunakan cipher AEAD
FUN_00513d20	deriveKeyFromPassword	Membentuk kunci enkripsi berdasarkan password
FUN_0051a2a0	verifyHMAC / generateHMAC	Proses verifikasi/penambahan

		HMAC
FUN_004fb660	xorBlockTransform (tidak eksplisit)	Transformasi buffer sebelum encryption

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan temuan dari ketiga pendekatan yang digunakan dalam penelitian ini, terdapat beberapa kesimpulan mengenai hasil analisis. Secara statis, sebagian besar menunjukkan entropi tinggi, entry point tidak lazim, dan struktur PE yang dimodifikasi, mengindikasikan teknik *obfuscation* dan evasi. Analisis dinamis mengungkap aktivitas berbahaya seperti enkripsi file, modifikasi registry, koneksi ke server C2, dan pembuatan proses baru. *Reverse engineering* mengidentifikasi teknik lanjutan seperti TLS *callback*, *ordinal import*, dan *dynamic API resolution*. LockBit dan BlackCat fokus pada persistensi dan penyamaran, WhisperGate bersifat destruktif, sementara HellCat bersifat hybrid dengan kemampuan mencuri data. Prince menunjukkan kompleksitas tinggi dengan kombinasi enkripsi AES-GCM dan ChaCha20 serta teknik injection. Pendekatan multi-analisis terbukti efektif dalam mengungkap struktur, perilaku, dan logika internal masing-masing ransomware.

5.2 Saran

Berdasarkan hasil dan kesimpulan yang telah diperoleh, berikut adalah beberapa saran yang dapat diberikan untuk keperluan penelitian selanjutnya:

1. Penelitian ini hanya menggunakan disassembly sebagai teknik yang digunakan dalam *reverse* engineering, namun belum mencakup proses debugging untuk analisis lebih dalam. Sementara penggunaan metode tersebut sangat berguna untuk mengamati jalannya eksekusi program secara langsung, terutama mengidentifikasi fungsi enkripsi, proses *obfuscation*, runtime, dan manipulasi memori.
2. Dalam analisis statis, penelitian ini belum dilakukan eksplorasi kode biner secara langsung melalui hex editor.
3. Penggunaan Any.Run sebagai sandbox cloud masih terbatas pada konfigurasi tertentu. Penggunaan sandbox lain, seperti cuckoo atau falcon memungkinkan peneliti mampu mengatur scenario analisis yang lebih dalam dan flexible

DAFTAR PUSTAKA

Appelbaum, D. (2012). *Reverse engineering: Fundamental analysis for technical analysts*. Wiley Finance.

Cahyadi, A. (2024*). Deteksi malware Zeus menggunakan analisis hybrid dan YARA rules.* Jurnal Teknologi Informasi dan Keamanan Siber, 5(1), 22–31.

David, A. P. (2021). *Ghidra: Software reverse engineering tool by NSA*. Journal of Cybersecurity Tools, 6(3), 45–52.

Mylonas, A., & Gritzalis, D. (2012). *Malware analysis techniques and cybersecurity measures*. Journal of Information Security, 3(4), 245–254.

Nadira, S. (2024). *Analisis malware WannaCry dan Trojan menggunakan metode statis dan dinamis.* Jurnal Keamanan Siber Indonesia, 4(2), 60–72.

Nicho, D., et al. (2023). *Analyzing WhisperGate and BlackCat malware: Methodology and threat perspective*. Journal of Advanced Threat Intelligence, 7(1), 77–90.

Noor, L. (2024). *Identifikasi pola financial theft dalam file APK menggunakan reverse engineering.* Jurnal Rekayasa Perangkat Lunak, 3(2), 90–101.

Peppers, M. (2018). *Building a malware analysis lab*. Syngress.

Puji Rahayu, & Trianto, N. (2021). *Malware dan keamanan sistem informasi*. Andi Publisher.

Rizqina, F. (2022). *ANY.RUN sebagai platform analisis dinamis malware interaktif*. Jurnal Sistem dan Keamanan Siber, 2(1), 15–22.

Sato, H. (2024). *Real-time malware detection with Any.Run*. Proceedings of the Cyber Defense Symposium.

Siddiq, M., et al. (2020). *Static PE analysis using PEStudio for malware detection*. Journal of Applied Cybersecurity.

Wahib, M., et al. (2022). *Ancaman siber dan strategi pertahanan digital di Indonesia*. Jurnal Teknologi dan Keamanan Digital, 3(3), 180–195.

Whitman, M. E., & Mattord, H. J. (2011). *Principles of information security (4th ed.)*. Cengage Learning.

Asghar, H. J., Zhao, B. Z. H., Ikram, M., Nguyen, G., Kaafar, D., Lamont, S., & Coscia, D. (2024). Use of cryptography in malware obfuscation. *Journal of Computer Virology and Hacking Techniques*, *20*(1), 135–152. https://doi.org/10.1007/s11416-023-00504-y

Banik, S. (2023). *STATIC ANALYSIS & IDENTIFICATION OF*. *11*(5), 243–258.

Bhardwaj, S., & Singh Arri, H. (2024). A Comprehensive Study of Efficient and Accurate Malware Detection Using Static and Dynamic Analysis Methods. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4502178

Davies, S. R., Macfarlane, R., & Buchanan, W. J. (2022). Comparison of Entropy Calculation Methods for Ransomware Encrypted File Identification. *Entropy*, *24*(10). https://doi.org/10.3390/e24101503

Gururaj H, L., Soundarya B, C., Janhavi, V., Lakshmi, H., & Prassan Kumar, M. J. (2022). Analysis of Cyber Security Attacks using Kali Linux. *IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics, ICDCECE 2022*, *November*. https://doi.org/10.1109/ICDCECE53908.2022.9793164

Hartono, B. (2023). Ransomware: Memahami Ancaman Keamanan Digital. *Bincang Sains Dan Teknologi*, *2*(02), 55–62. https://doi.org/10.56741/bst.v2i02.353

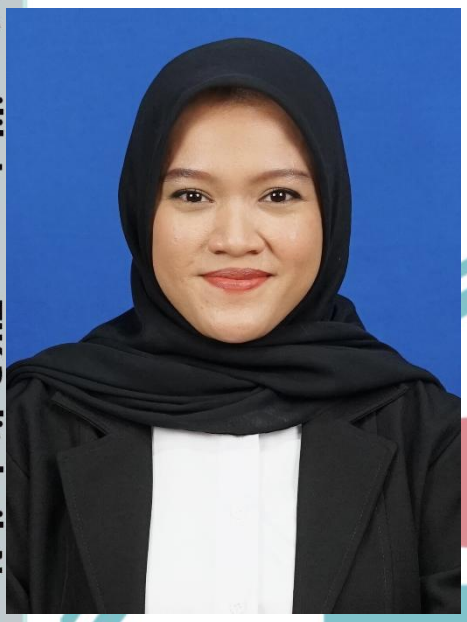
Kurnia Hatika, L., Budiyono, A., & Almaarif, A. (2019). *Analisis Ketepatan Deteksi Malware Pada Software Antivirus Menggunakan Metode Analisis Statis Accuracy Analysis of Malware Detection in Antivirus Software Using Static Analysis Method*. *6*(2), 7812–7819.

Mehmood, R., Ahmed, W., Riaz, R., & Awan, W. H. (2024). *Modern Malware Evasion and Bypass Strategies Against Contemporary Antivirus Software*. *3429*(2).

Mylonas, A., & Gritzalis, D. (2012). Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software. In *Computers & Security* (Vol. 31, Issue 6). No Starch Press. https://doi.org/10.1016/j.cose.2012.05.004

Sato, L. (2024). *Analysing malware evasion techniques Decoding Malicious Camouflage : Analysing Evasion Techniques in Malware Against EDR and AMSI Detection by*. *December*, 0–64. https://doi.org/10.13140/RG.2.2.12937.76641

DAFTAR RIWAYAT HIDUP



Layla Rosyidah

Lahir di Jakarta, 1 Maret 2002, anak ketiga dari tiga bersaudara. Lulus dari SDN 18 Kramat Jati pada tahun 2014, kemudian melanjutkan pendidikan di SMPN 150 Jakarta Timur dan lulus pada tahun 2017, dan selanjutnya di SMAN 93 Jakarta Timur dan lulus pada tahun 2020. Setelah *gap year* selama kurun waktu 1 tahun, tepatnya 2021 melanjutkan studi di Politeknik Negeri Jakarta pada Jurusan Teknik Informatika dan Komputer di Program Studi Teknik Multimedia dan Jurusan.