**JUDUL**

# Analisis Malware Menggunakan *Static Analysis, Dynamic Analysis,* dan *Reverse Engineering* pada *Honeypot*

**SKRIPSI**

**DINA YULIASTI 2207421002**

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

**2026**

**JUDUL**

# Analisis Malware Menggunakan *Static Analysis, Dynamic Analysis,* dan *Reverse Engineering* pada *Honeypot*

**SKRIPSI**

**DINA YULIASTI / 2207421002**

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

**2026**

# SURAT PERNYATAAN BEBAS PLAGIARISME

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

| | |
|---|---|
| Nama | : Dina Yuliasti |
| NIM | : 2207421002 |
| Jurusan/Program Studi | : Teknik Informatika dan Komputer/Teknik Multimedia dan Jaringan |
| Judul Skripsi | : Analisis Malware Menggunakan *static analysis*, *dynamic analysis*, dan *reverse engineering* pada *Honeypot* |

Menyatakan dengan sebenarnya bahwa skripsi ini 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, 2026

Yang membuat pernyataan


Dina Yuliasti

NIM. 2207421002

# LEMBAR PENGESAHAN

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

LEMBAR PENGESAHAN

Skripsi diajukan oleh:

| | |
|---|---|
| Nama | : Dina Yuliasti |
| NIM | : 2207421002 |
| Program Studi | : Teknik Multimedia dan Jaringan |
| Judul Skripsi | : Analisis Malware menggunakan Static Analysis, Dynamic Analysis, dan Reverse Engineering pada Honeypot |

Telah diuji oleh tim penguji dalam Sidang Skripsi pada hari Selasa, Tanggal 13, Bulan Juni, Tahun 2026 dan dinyatakan LULUS.

Disahkan oleh:

| | | |
|---|---|---|
| Pembimbing I | Iik Muhamad Malik Matin, S.Kom., M.T. | ( ) |
| Penguji I | Asep Kurniawan, S.Pd., M.Kom. | ( ) |
| Penguji II | Fachroni Arbi Murad, S.Kom., M.Kom. | ( ) |
| Penguji III | Andika Riyadi Jasril, M.Pd.T | ( ) |

Mengetahui:
Jurusan Teknik Informatika dan Komputer
Ketua

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



Dipindai dengan CamScanner

# KATA PENGANTAR

*Assalamualaikum Wr., Wb.*

Puji dan syukur selalu kepada Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Analisis Malware Menggunakan *Static Analysis*, *Dynamic Analysis*, dan *Reverse Engineering* pada *Honeypot*" dengan baik atas bantuan, motivasi, dan dukungan dari berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. kepada Ibu Dr. Anita Hidayati, S.Kom., M.Kom., selaku Ketua Jurusan, penulis mengucapkan terima kasih.
2. Bapak Iik Muhammad Malik Matin, S.Kom., M.T. selaku pembimbing penulis yang telah banyak membantu, mendukung, dan memberikan masukan serta saran kepada penulis selama mengerjakan skripsi ini hingga selesai.
3. Teristimewa penulis mengucapkan terima kasih yang sebesar-besarnya kepada kedua orang tua tercinta yang telah memberikan banyak dukungan, doa dan cinta tanpa syarat yang diberikan kepada penulis selama proses penyelesaian skripsi.
4. Sahabat dan teman-teman yang telah banyak membantu dan memberikan dukungan dalam pengerjaan skripsi.
5. Seluruh pihak yang telah terlibat, memberikan doa serta dukungan dalam proses penyusunan skripsi ini yang tidak dapat penulis sebutkan satu per satu.
6. Teruntuk Dina Yuliasti, anak perempuan pertama dan harapan orang tua. Terima kasih telah bertahan, tidak menyerah, dan terus berjalan melewati semua tantangan. Perjalanan ini belum selesai, tapi kamu mampu melewatinya. Ingat, kamu berhak bahagia dan bermimpi.

Akhir kata penulis mengucapkan banyak terima kasih kepada semuanya. Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak terdapat kekurangan dan keterbatasan, oleh karena itu penulis memohon maaf atas ketidaksempurnaan ini. Semoga skripsi yang ditulis ini bermanfaat dan menjadi motivasi untuk penelitian selanjutnya dan bagi pembaca.

*Wassalamualaikum Wr., Wb.*

Depok, Juni 2026

Penulis

# SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

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

| | |
|---|---|
| Nama | : Dina Yuliasti |
| NIM | : 2207421002 |
| Jurusan/Program Studi | : Teknik Informatika dan Komputer/Teknik Multimedia dan Jaringan |

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

Analisis Malware menggunakan *Static Analysis*, *Dynamic Analysis*, dan *Reverse Engineering* pada *Honeypot*

Dengan Hak Bebas Royalti NonEksklusif 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 pernyataan ini saya buat dengan sebenarnya.

Depok, 2026

Yang menyatakan


Dina Yuliasti
NIM.2207421002

**Analisis Malware Menggunakan Static Analysis, Dynamic Analysis, dan Reverse Engineering pada Honeypot**

## ABSTRAK

*Malware Windows merupakan ancaman keamanan siber yang terus berkembang dengan varian baru bermunculan setiap harinya, sementara penelitian terdahulu masih terbatas pada satu atau dua metode analisis serta bergantung pada dataset publik yang bersifat statis dan kurang mencerminkan pola serangan terkini. Penelitian ini bertujuan untuk mengembangkan honeypot sebagai sarana pengumpulan sampel malware Windows secara real-time, serta melakukan analisis mendalam terhadap sampel tersebut menggunakan tiga metode terintegrasi, yaitu Static Analysis, Dynamic Analysis, dan Reverse Engineering, guna menghasilkan kompilasi Indicators of Compromise (IoC) yang komprehensif. Studi kasus dilakukan terhadap lima sampel malware yang ditangkap oleh honeypot Dionaea pada platform T-Pot, terdiri atas dua sampel bertipe Launcher dan tiga varian WannaCry, yang sebelumnya diverifikasi menggunakan VirusTotal. Static analysis menggunakan PEStudio, Detect-It-Easy (DIE), dan FLOSS berhasil mengidentifikasi struktur PE, toolchain kompilasi yang identik pada seluruh sampel, evolusi tingkat obfuscation (entropi rendah pada Sampel 1–3 dan tinggi pada Sampel 4–5), serta domain C2 hardcoded yang menyerupai mekanisme kill-switch WannaCry tanpa risiko eksekusi. Dynamic analysis melalui Process Monitor, Process Explorer, Wireshark, dan FakeNet mengungkap perilaku runtime aktual, termasuk network scanning agresif menuju port 445 (SMB), upaya eksploitasi SMBv1/EternalBlue oleh proses mssecsvr.exe, serta pola registry enumeration dan masquerading pada berbagai proses. Reverse Engineering menggunakan Ghidra berhasil membongkar logika dropper dan mekanisme konstruksi path dinamis melalui fungsi ekspor Play Game, yang tidak terlihat pada kedua metode sebelumnya. Hasil penelitian menunjukkan bahwa ketiga metode bersifat saling melengkapi, di mana kombinasi ketiganya menghasilkan cakupan IoC yang jauh lebih komprehensif dibandingkan penggunaan satu atau dua metode saja, sehingga dapat dijadikan acuan deteksi dan respons insiden siber terhadap malware Windows canggih seperti WannaCry.*

**Kata kunci:** *Malware Analysis, Static Analysis, Dynamic Analysis, Reverse Engineering, Honeypot, Indicators of Compromise, Windows Malware, WannaCry.*

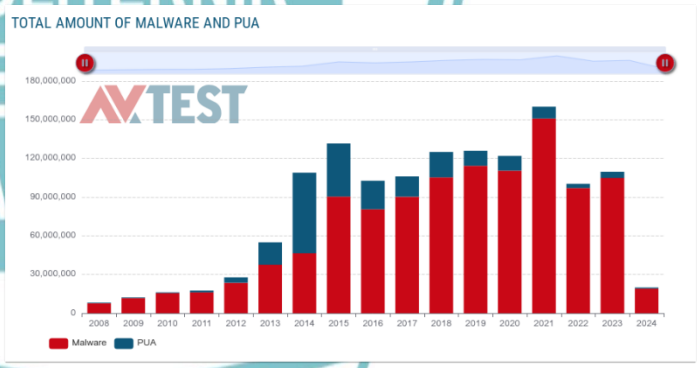
# DAFTAR ISI

# DAFTAR GAMBAR

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Malware merupakan ancaman keamanan siber yang jahat dan paling kritis, yang secara sistematis menargetkan sistem operasi Windows, *platform* yang mendominasi sekitar 73,38% pasar perangkat komputasi global di lingkungan personal, *enterprise*, dan infrastruktur kritis (StatCounter, 2024). Jenis serangan ini mencakup ransomware yang mengenkripsi data berharga secara massal untuk memeras korban (Kaspersky, 2024), trojan downloader yang mengunduh *payload* tambahan secara diam-diam, serta *infostealer* canggih seperti Lumma Stealer yang mencuri kredensial perbankan dan akun korporat, mengakibatkan kerugian ekonomi mencapai ratusan miliar rupiah pada tahun 2023-2024 (Asosiasi Penyelenggara Jasa Internet Indonesia, 2024). Menurut AV-TEST Institute, sebuah lembaga independen yang memantau ancaman siber secara global, lebih dari 450.000 malware dan program tidak diinginkan (Potentially Unwanted Application/PUA) baru terdaftar setiap hari di basis data mereka, dengan jumlah ini terus bertambah karena aktivitas penyerang yang konstan (AV-TEST Institute, 2024). *Platform* Windows tercatat sebagai salah satu target utama malware karena dominasi penggunaannya di lingkungan desktop/laptop serta luasnya serangan yang sering dieksploitasi dalam berbagai teknik malware.




Gambar 1. 1 Statistik Total Malware dan Potentially Unwanted Applications

Pada Gambar menunjukkan perkembangan jumlah total malware dan *Potentially Unwanted Applications* (PUA) secara global yang tercatat oleh AV-TEST

Institute sejak tahun 2008 hingga 2024. Grafik disajikan dalam bentuk batang bertumpuk (stacked bar chart), di mana warna merah merepresentasikan jumlah malware dan warna biru menunjukkan jumlah PUA.

Berdasarkan data yang dipublikasikan oleh analisis antivirus independen di tahun 2025 sekitaran 87,4% dari deteksi malware terjadi pada windows, sedangkan pada mac OS mencatat sekitar 12,6% dari total kasus malware yang terdeteksi.Penelitian sebelumnya menghadapi beberapa tantangan dan keterbatasan yang signifikan. Pertama, berkaitan dengan sumber data, sebagaimana yang telah dilakukan oleh Yusirwan et al. (2020) dan Guo et al. (2023), banyak penelitian sebelumnya bergantung pada dataset publik seperti Virus Share, Malware Bazaar, atau Contagiodump yang bersifat statis dan sudah tersedia secara terbuka, sehingga menimbulkan permasalahan terhadap tingkat kebaruan sampel yang dianalisis. Di sisi lain, penelitian yang dilakukan oleh R.M. Muhammad et al. (2021) dan Holbel et al. (2024) memanfaatkan honeypot, namun fokus penelitian mereka lebih mengarah pada perancangan sistem keamanan jaringan dan proses *threat hunting*, bukan pada analisis teknis mendalam terhadap sampel malware itu sendiri. Kedua, keterbatasan metodologi analisis yang dipakai sebagaimana dilakukan oleh Eliando et al. (2022), Guo et al. (2023), dan Damaševičius et al. (2025). Penelitian terdahulu cenderung melakukan satu jenis analisis saja, baik itu *static analysis*, *dynamic analysis*, maupun *reverse engineering*, tanpa melakukan kombinasi komprehensif dari ketiga metode tersebut untuk membongkar logika internal malware secara mendalam.

*Static analysis* menjadi langkah penting pertama yang memungkinkan ekstraksi informasi struktural dari *Portable Executable* (PE) header tanpa eksekusi berisiko, termasuk analisis *Import Address Table (IAT), section permissions*, overlay data, serta *suspicious strings* yang mengindikasikan C2 domains atau *registry run keys.* Tools seperti PEiD untuk packer identification dan *Strings* untuk *hardcoded IOC extraction* memberikan *baseline analysis* yang cepat namun terbatas pada *surfacelevel artifacts.*

*Dynamic analysis* melengkapi keterbatasannya dengan monitoring perilaku *runtime* dalam lingkungan sandbox terisolasi, mengamati API calls mencurigakan seperti

*CreateRemoteThread, registry* modifications di *HKLM\Software\Microsoft\Windows\CurrentVersion\Run, network beaconing ke external IPs, serta file drops di %AppData% atau %Temp% directories. Platform* tools Process Monitoring dan Process Explorer secara spesifik mencatat *behavioral indicators* yang tak terlihat pada *static examination*, seperti *process injection* dan *privilege escalation attempts.*

*Reverse engineering* melalui *disassembly* dan *decompilation* kode *assembly* merupakan tahap lanjutan analisis untuk mengungkap logika tersembunyi seperti *anti-analysis techniques (timing checks, red pill), custom crypters*, dan *hardcoded C2 infrastructure* yang luput dari dua metode sebelumnya, memerlukan tools profesional seperti Ghidra untuk *control flow reconstruction*.

Penelitian ini mengambil sampel malware terbaru yang dihasilkan langsung dari honeypot tpot, menangkap varian autentik dari penyerang nyata dalam real-time, berbeda dengan penggunaan dataset publik yang bersifat statis seperti VirusShare atau Malware Bazar yang sudah outdated dalam 24-48 jam. Sampel *Honeypot* merepresentasikan *attack patterns* terkini dengan geographical relevance ke Indonesia, memberikan ground truth data untuk validasi ketiga metode analisis secara komprehensif.

Penelitian ini menerapkan analisis sistematis terhadap sampel malware Windows terkini dari honeypot menggunakan tiga metode terintegrasi secara berurutan: *Static Analysis* dengan Detect-It-Easy (DIE), strings/Floss, PEStudio, dan Virustotal untuk *structural profiling* tanpa *execution risk*. *Dynamic Analysis* melalui Process monitor (Procmon), Process explorer, Wireshark, dan juga Fakenet. *Reverse Engineering* menggunakan Ghidra (free) untuk *full disassembly, function identification, string references,* dan *control flow graph reconstruction* guna ekstraksi IOC tingkat tinggi seperti *mutex names, registry persistence paths,* dan *obfuscated C2 communications.*

### 1.2. Perumusan Masalah

Melalui latar belakang tersebut, maka rumusan masalahnya adalah:

1. Bagaimana mengembangkan honeypot untuk mengumpulkan malware?
2. Bagaimana melakukan analisis menggunakan *static analysis*, *dynamic analysis*, dan *reverse engineering* terhadap sampel malware yang dihasilkan dari honeypot.

### 1.3. Batasan Masalah

Batasan masalah ini bertujuan untuk memperjelas ruang lingkup penelitian sehingga pembahasan tidak terlalu luas. Adapun batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Sampel malware terbatas pada varian Windows yang ditangkap honeypot dan dipilih sebanyak 5 sampel
2. Analisis static menggunakan tools DIE, PEStudio, VirusTotal,dan strings/Floss
3. Analisis dinamik menggunakan Procmon, Process Explorer, Wireshark, dan Fakenet
4. *Reverse Analysis* menggunakan tools Ghidra untuk *disassemble* dan *decompilation*.
5. Penelitian ini Tidak membahas pengembangan tools baru atau integrasi dengan SIEM production systems
6. Lingkungan pengujian dibatasi hanya pada windows 11
7. Penelitian ini Tidak membahas *behavioral machine learning classification*
8. Penelitian ini tidak menggunakan YARA rules untuk mendeteksi malware.

### 1.4. Tujuan dan Manfaat

#### 1.4.1. Tujuan

Tujuan dari penelitian ini adalah sebagai berikut:

1. Mengembangkan honeypot untuk mengumpulkan malware

2. Melakukan analisis menggunakan *static analysis, dynamic analysis*, dan *reverse engineering* terhadap sampel malware yang dihasilkan dari honeypot

### 1.4.2. Manfaat

Adapun manfaat dari penelitian ini adalah sebagai berikut.

1. Menyediakan referensi analisis malware berbasis honeypot dengan *dataset real-time* untuk penelitian *cybersecurity* akademik .
2. Memperkaya literatur tentang sinergi *static-dynamic-reverse engineering* pada varian Windows malware terkini.
3. Memberikan studi kasus praktis tools analisis yang direkomendasikan komunitas peneliti malware.
4. Menghasilkan kompilasi IoC *actionable* (*registry keys, mutexes, C2 domains*)
5. Memberikan panduan analisis malware bagi praktisi *cyber security* Indonesia menghadapi ancaman Windows dominan.
6. Mendukung validasi honeypot *deployment* untuk pengumpulan sampel malware berkualitas tinggi di organisasi lokal.
7. Menjadi bahan evaluasi pada sampel malware yang dihasilkan dari honeypot sebagai sumber data analisis malware berbasis serangan nyata.

## 1.5. Sistematika Penulisan

Adapun sistematika penulisan dari laporan ini adalah sebagai berikut

a. BAB I PENDAHULUAN

   Bab I ini menjelaskan latar belakang penelitian, perumusan masalah, batasan, tujuan, dan manfaat penelitian, serta sistematika penulisan.

b. BAB II TINJAUAN PUSTAKA

   Bab II berisikan penjelasan mengenai landasan teori atau kajian ilmu yang berhubungan dengan berbagai pokok pikiran topik penyusunan skripsi ini yang relevan dari sumber yang valid.

c. BAB III METODE PENELITIAN

Bab III menjelaskan rancangan penelitian dan alur sistem analisis malware berbasis honeypot yang diusulkan, sumber data dan sampel yang digunakan, teknik pengumpulan data dari lingkungan honeypot dan lab analisis, teknik analisis data secara kualitatif dan kuantitatif terhadap hasil *static*, *dynamic*, dan *reverse engineering*, serta jadwal pelaksanaan penelitian dan rincian biaya yang diperlukan.

d. BAB IV HASIL DAN PEMBAHASAN

Bab IV menyajikan hasil penelitian dan pembahasan dari analisis malware menggunakan tiga metode terintegrasi. Bab ini terdiri dari sub-sub bab mencakup analisis kebutuhan, perancangan sistem, implementasi sistem, dan pengujian dengan *static analysis*, *dynamic analysis*, dan *reverse engineering* serta evaluasi pengujian berupa kompilasi IoC dari hasil ketiga metode analisis.

e. BAB V PENUTUP

Bab V berisikan kesimpulan atau hasil akhir dari penelitian yang telah dilakukan, serta saran untuk penelitian berikutnya.

# BAB V
# PENUTUP

## 5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan terhadap lima sampel malware Windows yang diperoleh dari honeypot selama 2 bulan menggunakan tiga metode analisis secara berurutan (*static analysis*, *dynamic analysis*, dan *reverse engineering*), menghasilkan kesimpulan sebagai berikut:

1. Proses pengumpulan sampel dan temuan teknis ketiga metode analisis. *Honeypot* Dionaea yang dijalankan dalam platform T-Pot dan dikonfigurasi pada VPS berhasil menangkap sampel malware secara *real-time* dari penyerang nyata yang menargetkan port SMB (445), HTTP (80), dan FTP (21). Sampel yang tertangkap diverifikasi menggunakan VirusTotal, kemudian diseleksi secara *purposive* (keunikan hash, validitas format PE, keberagaman tipe) sehingga diperoleh 5 sampel *representative* yaitu 2 sampel bertipe Launcher (Sampel 1–2) dan 3 varian WannaCry (Sampel 3–5). Ketiga metode analisis yang diterapkan secara berurutan berhasil mengungkap karakteristik malware secara bertahap dan saling melengkapi. *Static analysis* (PEStudio, DIE, FLOSS) mengidentifikasi bahwa kelima sampel dikompilasi dengan *toolchain* identik dan bernama asli "launcher.dll" mengindikasikan satu *builder framework* yang sama dengan entropi rendah pada Sampel 1–3 (4,0–4,3) dan tinggi pada Sampel 4–5 (6,4+, *payload* terenkripsi pada .rsrc), serta *domain C2 hardcoded* yang menyerupai *kill-switch* WannaCry pada Sampel 2 dan 3. *Dynamic analysis* (Procmon, Process Explorer, Wireshark, FakeNet) mengungkap perilaku *runtime* yang tidak terlihat pada static analysis, di antaranya *masquerading* dan *persistence* pada Sampel 1, *registry enumeration* pada Sampel 2, perilaku *worm* paling agresif dengan upaya eksploitasi SMBv1/*EternalBlue* oleh mssecsvr.exe pada Sampel 3 dan 4, serta persiapan eksploitasi SMB melalui *LanmanServer\Parameters* pada Sampel 5. *Reverse engineering* (Ghidra) mengonfirmasi logika internal di baliknya yaitu fungsi ekspor Play Game pada Sampel 3–5 sebagai *entry point payload*, *mekanisme two-stage*

*execution* pada Sampel 4 (konstruksi path mssecsvr.exe via sprintf diikuti eksekusi), pola *dropper/packer FindResource-LoadResource-LockResource-SizeofResource* pada Sampel 1, serta *conditional dispatcher logic* pada fungsi entry Sampel 5 yang memanggil sub-rutin FUN_1000113e sebagai logika inti malware.

2. Komplementaritas ketiga metode analisis. *Static*, *dynamic*, dan *reverse engineering* terbukti bersifat saling melengkapi dan tidak dapat saling menggantikan dalam menghasilkan *Indicator of Compromise* (IoC) yang komprehensif. *Static analysis* unggul dalam kecepatan dan keamanan eksekusi untuk *profiling* awal (IoC berbasis *file/hash*), *dynamic analysis* mengungkap *tradecraft* nyata (IoC jaringan, *registry*, dan proses), sedangkan *reverse engineering* menghasilkan IoC tingkat lanjut (mekanisme enkripsi, anti-debugging, dan logika C2) yang tidak dapat diperoleh dari kedua metode lainnya. Kombinasi ketiganya menghasilkan cakupan IoC yang jauh lebih lengkap dibandingkan penggunaan satu atau dua metode saja, mengonfirmasi bahwa pendekatan *mono-* atau *dual-*metode tidak memadai untuk menganalisis malware canggih seperti WannaCry yang mengimplementasikan *packing*, *anti-debugging*, C2 *terenkripsi*, dan *process injection* secara bersamaan.

## 5.2 Saran

Berdasarkan hasil penelitian dan kesimpulan yang telah dipaparkan, berikut adalah saran yang dapat dijadikan bahan pertimbangan untuk penelitian dan pengembangan selanjutnya:

1. Perluasan sampel dan periode pengumpulan, memperpanjang operasi honeypot menjadi minimal 3 bulan dan menambah sampel menjadi 20–50 untuk hasil *statistik* yang lebih representatif.
2. Integrasi *threat intelligence*, mengintegrasikan IoC (*hash, domain C2, registry key, mutex*) ke platform seperti MISP atau OpenCTI untuk mempercepat respons terhadap varian malware serupa.
3. Pengembangan YARA rules, menyusun dan mengevaluasi YARA rules berbasis IoC hasil penelitian terhadap sampel malware baru dari honeypot.

4. Otomatisasi *pipeline* analisis, mengembangkan scripting (Python/PowerShell) untuk menjalankan tools secara berurutan dan menyusun matriks IoC otomatis, guna meningkatkan skalabilitas dan replikabilitas.
5. Perluasan ke *platform* non-Windows, mengadaptasi metodologi tiga metode ini untuk malware Linux (server/IoT) dan Android (APK), mengingat meningkatnya ancaman pada *platform* tersebut.

# DAFTAR PUSTAKA

AV-TEST Institute. (2025). *Malware statistics*. https://www.av-test.org/en/statistics/malware/

Bombardieri, C., & Capretti, S. (2015). *Honeypot-powered malware reverse engineering (HERESy)* [Preprint]. arXiv. https://arxiv.org/abs/1510.03892

Damaševičius, R., Maskeliūnas, R., Blažauskas, T., & Smuikys, A. (2025). FETA: A systematic and efficient approach for feature extraction in malware analysis. *Journal of Information Security and Applications, 90*, 103965. https://doi.org/10.1016/j.jisa.2025.103965

Eliando, E., Lumenta, A. S. M., & Najoan, X. B. N. (2022). LockBit 2.0 ransomware: Analysis of infection, persistence, and impact. *Cogito Smart Journal*, *8*(2), 356–370. https://cogito.unklab.ac.id/index.php/cogito/article/view/356

Guo, M. I., Anwer, I., Riasat, A., & Kim, S. (2023). Windows malware detection based on static analysis with multiple features. *PeerJ Computer Science, 9*, e1319. https://doi.org/10.7717/peerj-cs.1319

Holbel, T., Combs, J., Graves, J., & Haynes, T. (2024). Utilizing virtualized honeypots for threat hunting, malware analysis and reporting. *Issues in Information Systems*, *25*(1), 265–278. https://iacis.org/iis/2024/1_iis_2024_265-278.pdf

Infosec Institute. (2024). *Static malware analysis tutorial*. https://resources.infosecinstitute.com/topic/static-malware-analysis/

International Journal for Multidisciplinary Research. (2023). Malware analysis and reverse engineering. *IJFMR*, *5*(6). https://www.ijfmr.com/papers/2023/6/10296.pdf

Kaspersky. (2024). *Ransomware: The biggest cyber threat of 2024*. Kaspersky Security Blog. https://www.kaspersky.com/resourcecenter/threats/ransomware

Muhammad, R. M., Stiawan, D., & Idris, M. Y. (2021). Integrated security system implementation for network environment using honeypot. *Journal of Navy and University Studies (JONUNS), 1*(1). https://jonuns.com/index.php/journal/article/view/619

StatCounter GlobalStats. (2024). *Desktop operating system market share worldwide*. https://gs.statcounter.com/os-market-share/desktop/worldwide/

Surfshark. (2025). *Malware cases by operating system.* https://surfshark.com/research/chart/malware-cases-windows-macOS/

YARA Project. (2024). *YARA official documentation*. ReadTheDocs. https://yara.readthedocs.io/

Yusirwan, S., Wahyudi, A., & Sanjaya, R. (2020). Implementation of malware analysis using static and dynamic analysis method. *Proceedings of the 2020 International Conference on Electrical Engineering and Informatics (ICEEI)*. IEEE. https://doi.org/10.1109/ICEEI51358.2020.9254935

National Institute of Standards and Technology. (2021). *Malware incident prevention and handling for desktops and laptops* (NIST Special Publication 800-83 Rev. 1). U.S. Department of Commerce. https://doi.org/10.6028/NIST.SP.800-83r1

Afianian, A., Niksefat, S., Sadeghiyan, B., & Baptiste, D. (2022). Malware dynamic analysis evasion techniques: A survey. *ACM Computing Surveys, 54*(6), 1–35. https://doi.org/10.1145/3365001

Grigorios, L., Apostolidis, K. D., & Kanas, V. G. (2021). Comparative review of malware analysis methodologies. *International Journal of Network Security & Its Applications, 13*(6), 91–104. https://doi.org/10.5121/ijnsa.2021.13608

Fortinet. (2024). *What are indicators of compromise (IoC)?* FortiGuard Glossary. https://www.fortinet.com/resources/cyberglossary/indicators-ofcompromise

Microsoft Security. (2024). *Indicators of compromise (IoCs) in Microsoft Defender*. Microsoft Learn. https://learn.microsoft.com/en-us/defenderendpoint/manage-indicators

Mandiant/Google. (2023). *FLARE FLOSS: FireEye Labs Obfuscated String Solver* (Version 3.0) [Software]. GitHub. https://github.com/mandiant/flare-floss

Microsoft Sysinternals. (2023). *Process Monitor (Procmon) for Windows* [Software]. Microsoft. https://learn.microsoft.com/en-us/sysinternals/downloads/procmon

VirusTotal. (2024). *VirusTotal: Analyze suspicious files, domains, IPs and URLs* [Web service]. Google. https://www.virustotal.com/

Wireshark Foundation. (2024). *Wireshark: Network protocol analyzer* (Version 4.2) [Software]. https://www.wireshark.org/

Avanzato, J. (2025). From CPU spikes to defense: How Varonis prevented a ransomware disaster. Varonis Blog. Retrieved from https://www.varonis.com/blog/varonis-prevents-ransomware-disaster

varo, S., Vila, G., & Pegueroles, J. (2026). Automating the detection of evasive Windows malware: An evaluated YARA rule library for anti-VM and antisandbox techniques. Journal of Cybersecurity and Privacy, 6(2), 69. https://doi.org/10.3390/jcp6020069

Kanj, S., Vila, G., & Pegueroles, J. (2026). Automating the detection of evasive Windows malware: An evaluated YARA rule library for anti-VM and antisandbox techniques. Journal of Cybersecurity and Privacy, 6(2), 69. https://doi.org/10.3390/jcp6020069

Varonis Systems. (2025). From CPU spikes to defense: How Varonis prevented a ransomware disaster. Retrieved from https://www.varonis.com/blog/varonisprevents-ransomware-disaster

Cyber Security News. (2025, February 6). *Ghidra 11.3 released*. Diakses dari https://cybersecuritynews.com/ghidra-11-3-released/

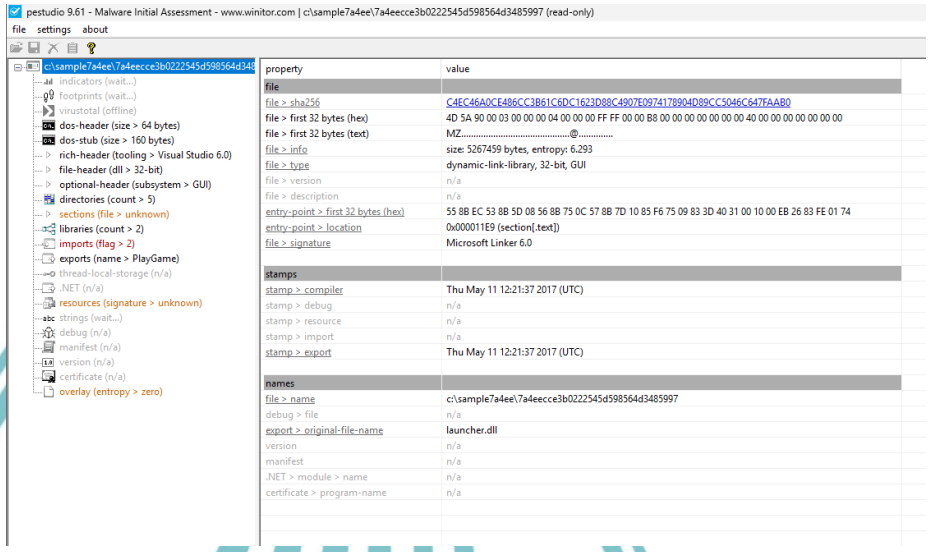
# DAFTAR RIWAYAT HIDUP

Dina Yuliasti, akrab disapa Dina. Lahir di Bekasi, pada tanggal 23 Juli 2003. Penulis memulai Pendidikan formal di SD Negeri 062 Mompang Jae pada tahun 2009, setelah itu melanjutkan Pendidikan di SMP Negeri 1

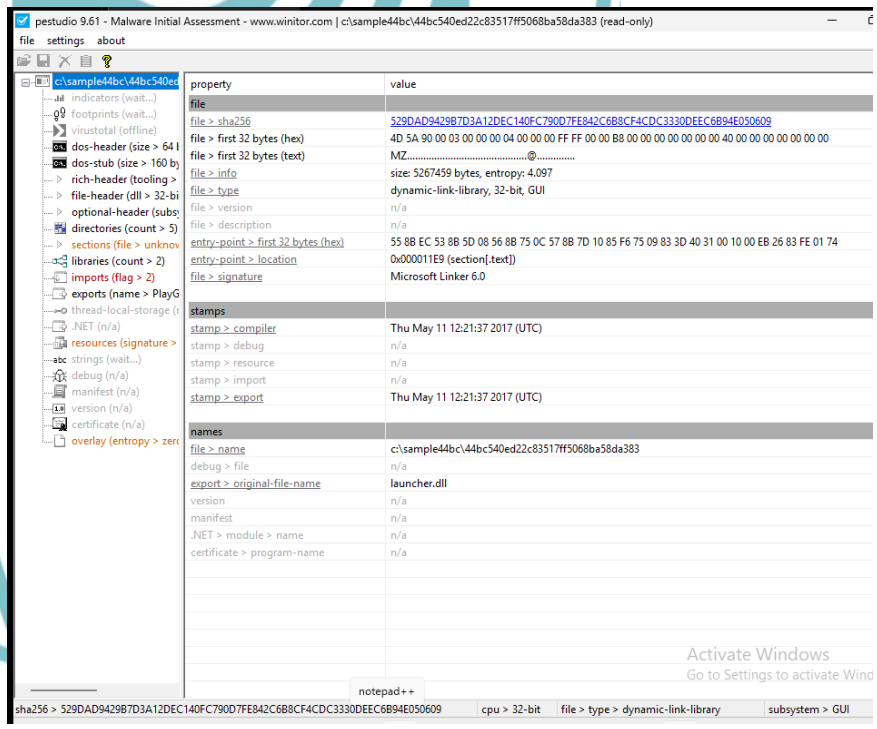
Panyabungan Utara.

pada tahun 2015, Kemudian meneruskan Pendidikan di SMK Negeri 3 Panyabungan pada tahun 2019. Pada tahun 2022 penulis menempuh Pendidikan Diploma IV di Politeknik Negeri Jakarta dengan jurusan Teknik Informatika dan Komputer pada Program Studi Teknik Multimedia dan Jaringan.

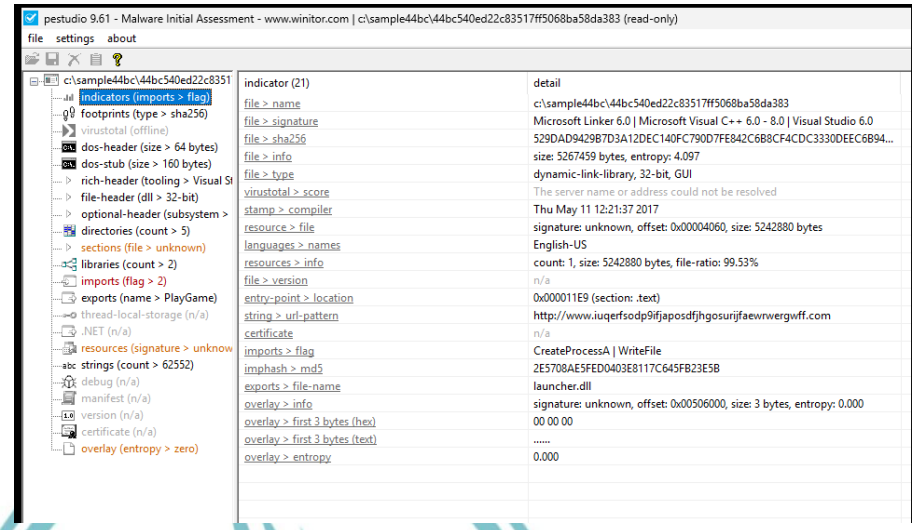
# LAMPIRAN



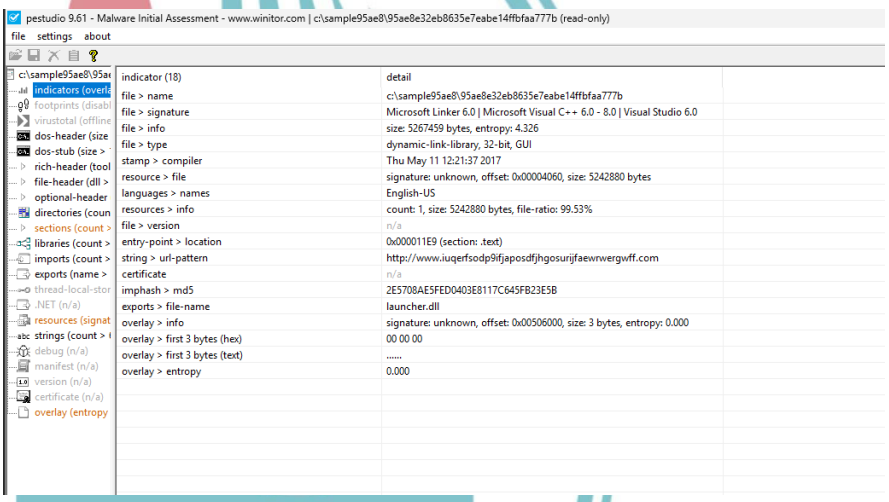

Lampiran 1 Tampilan PEStudio pada tahap analisis sampel 1



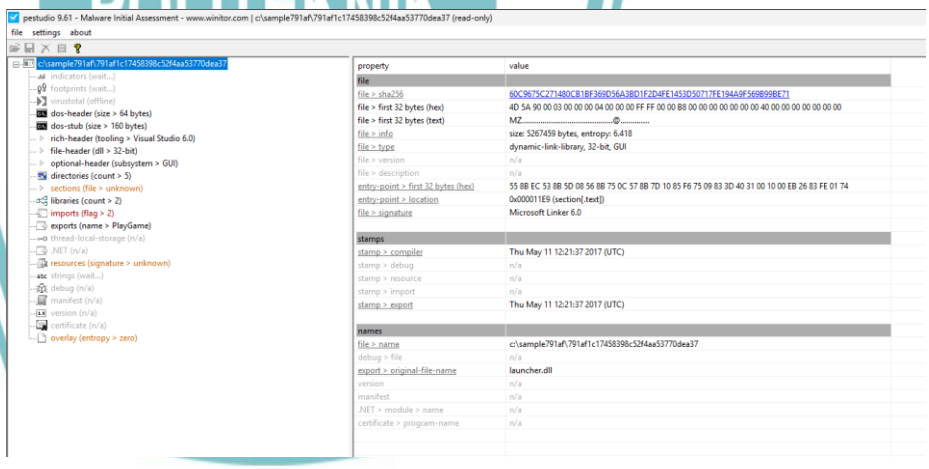

Lampiran 2 Tampilan PEStudio pada tahap analisis sampel 2



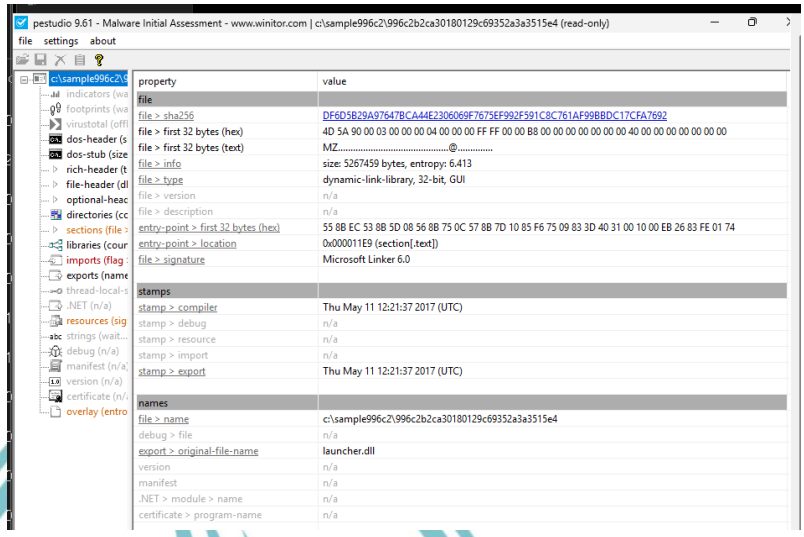

Lampiran 3 . Tampilan jendela indicators pada sampel 2



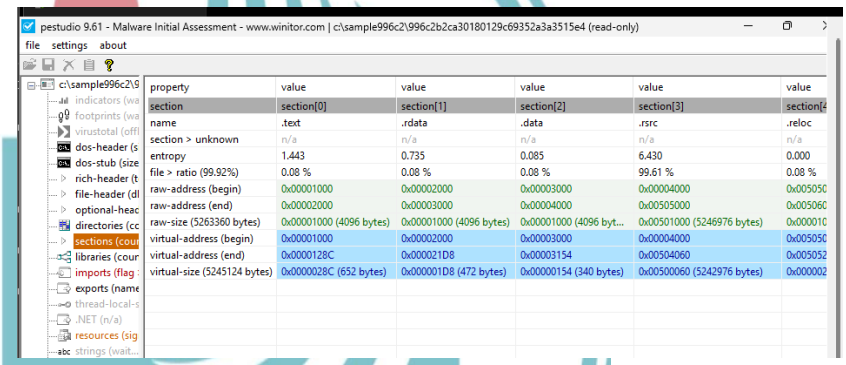

Lampiran 4 Tampilan menu indicators pada sampel 3



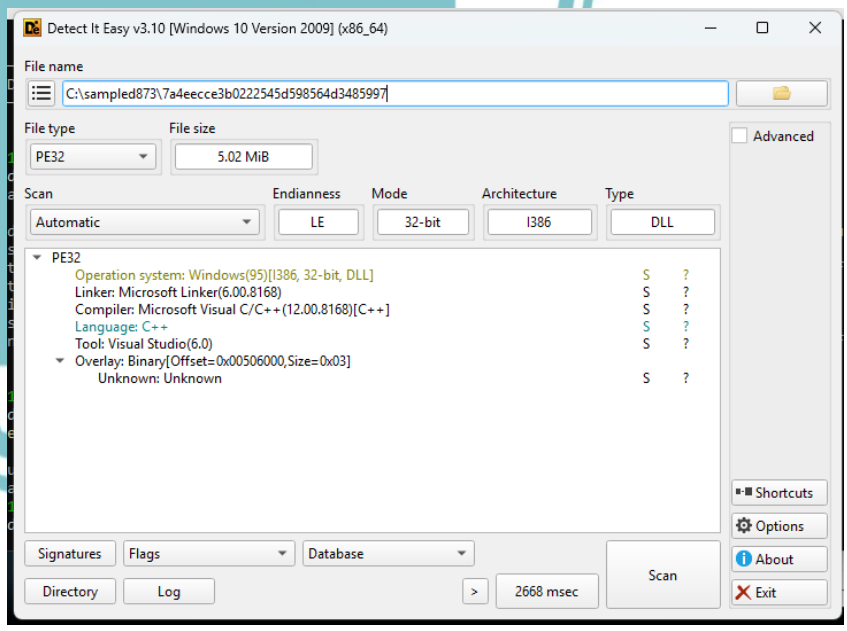

Lampiran 5 Tampilan jendela PEStudio pada sampel 4



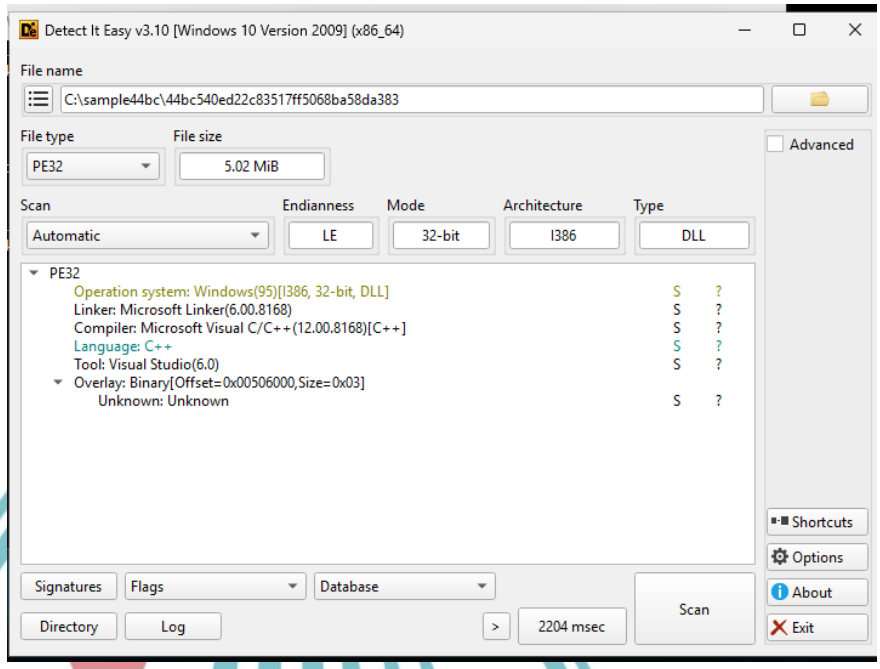

Lampiran 6 Tampilan jendela PEStudio pada sampel 5



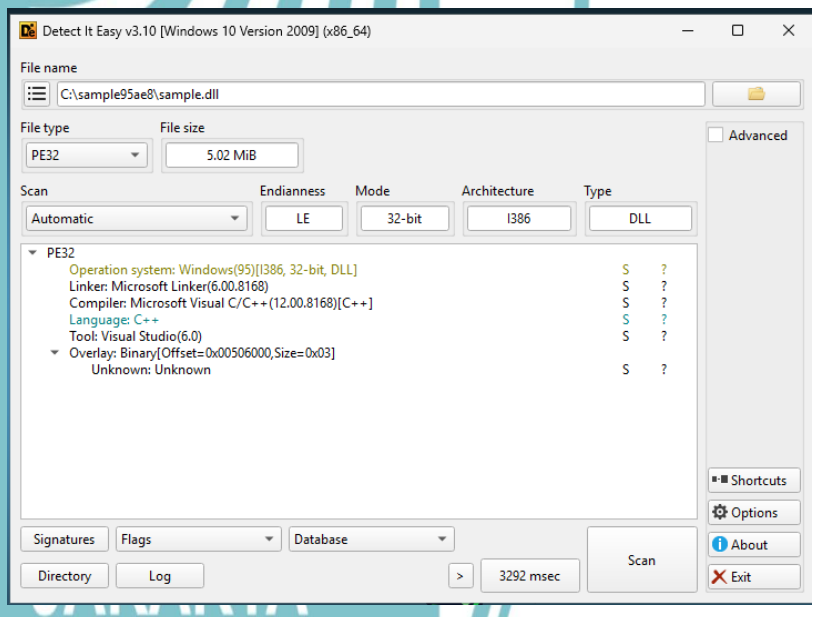

Lampiran 7 Tampilan menu section pada sampel 5



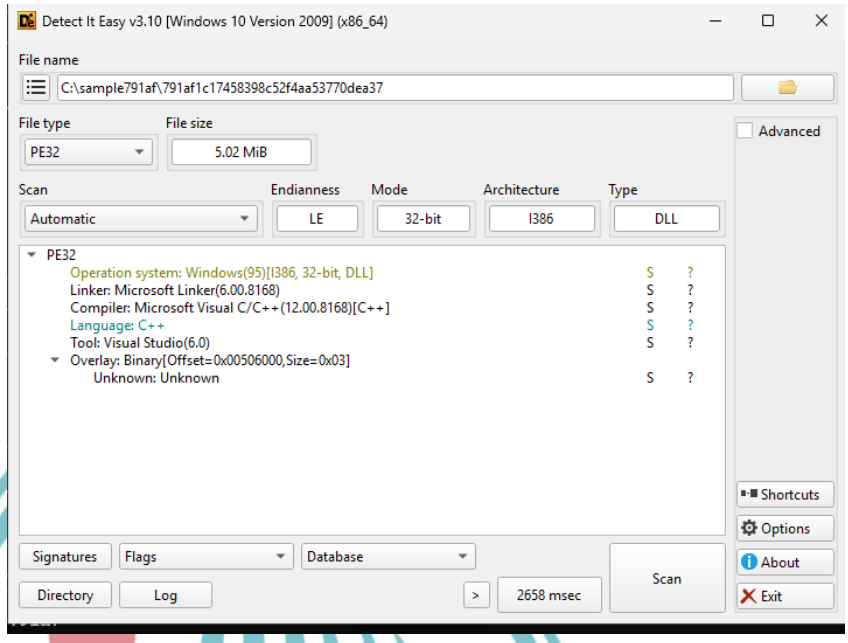

Lampiran 8 Tampilan DIE pada analisis sampel 1



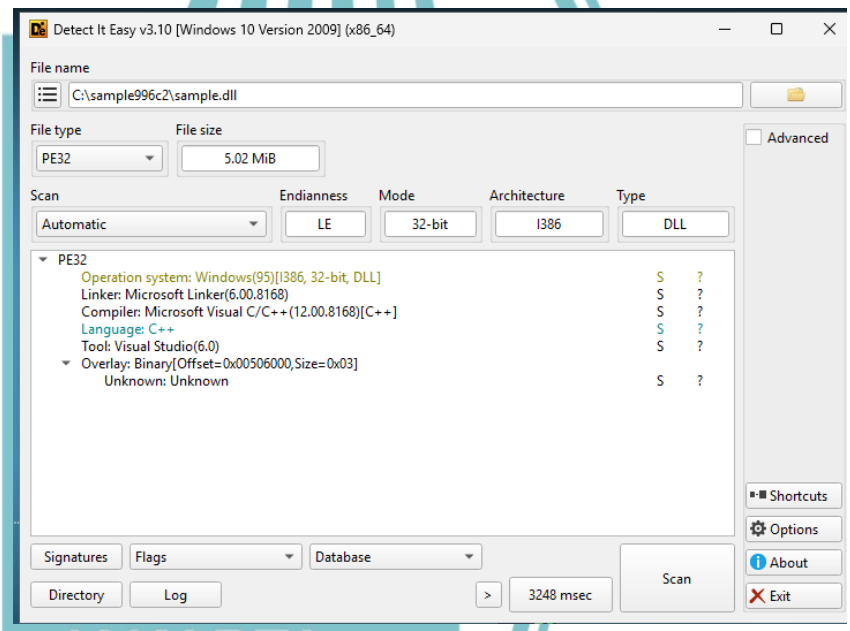

Lampiran 9 Tampilan DIE pada analisis sampel 2




Lampiran 10 Tampilan DIE pada analisis sampel 3



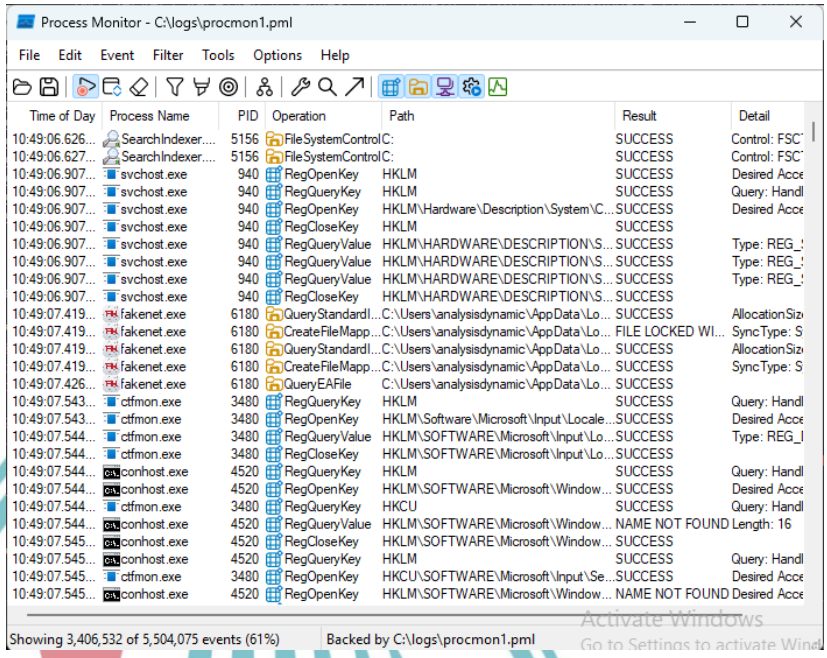

Lampiran 11 Tampilan DIE pada analisis sampel 4



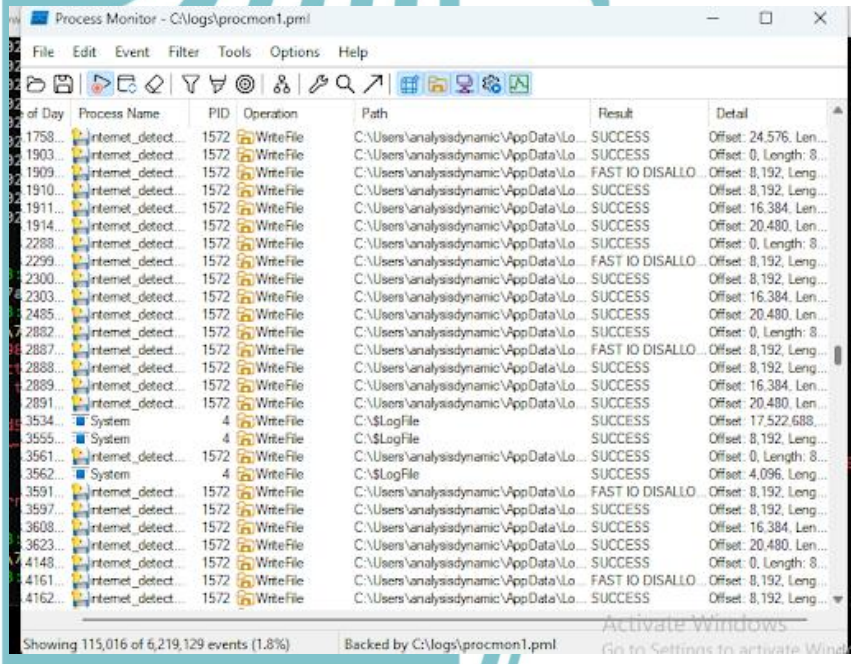

Lampiran 12 Tampilan DIE pada analisis sampel 4

```
PS C:\sample7a4ee > floss .\7a4eecce3b0222545d598564d3485997 > hasil_floss.txt
INFO: floss: extracting static strings
finding decoding function features: 100%|| 7/7 [00:00<00:00, 103.62 functions/s, skipped 2 library functions (28%)]
INFO: floss.stackstrings: extracting stackstrings from 4 functions
extracting stackstrings: 100%|| 4/4 [00:00<00:00, 22.18 functions/s]
INFO: floss.tightstrings: extracting tightstrings from 0 functions...
extracting tightstrings: 0 functions [00:00, ? functions/s]
INFO: floss.string_decoder: decoding strings
emulating function 0x100010ab (call 1/1): 100%|| 4/4 [00:00<00:00, 12.86 functions/s]
INFO: floss: finished execution after 111.95 seconds
INFO: floss: rendering results
```

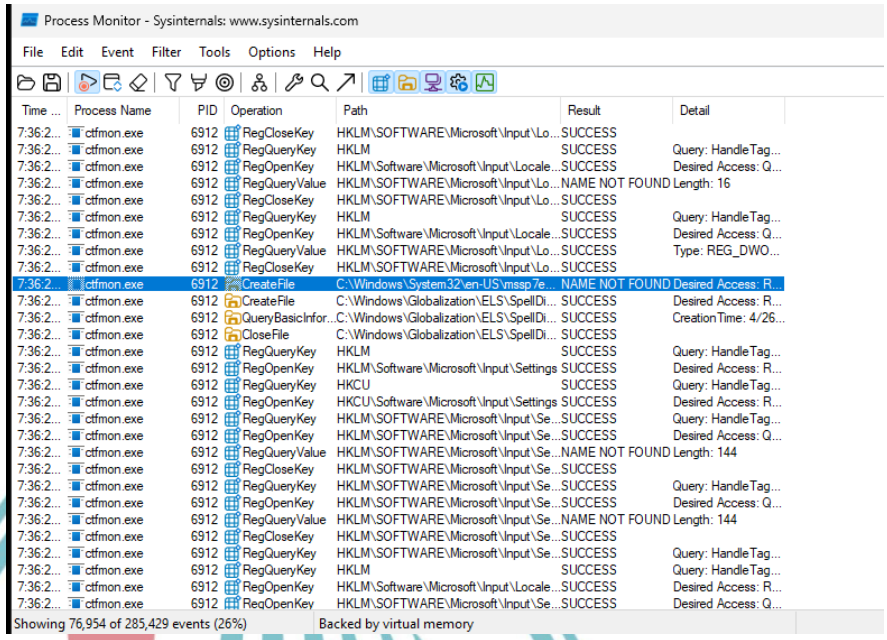
Lampiran 13 Proses ekstraksi dan deskripsi string pada sampel 1



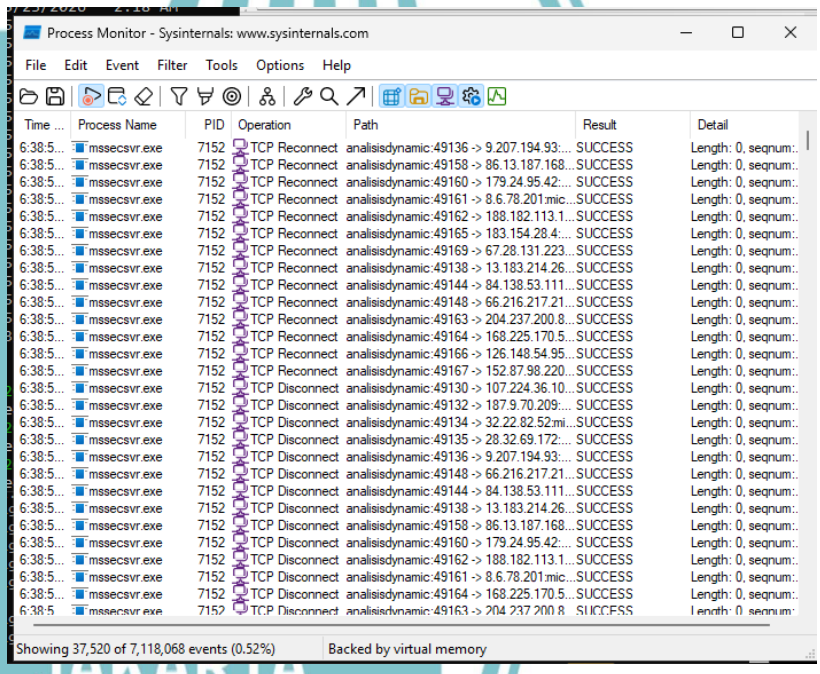

Lampiran 14 Pemantauan aktivitas terhadap sampel 1



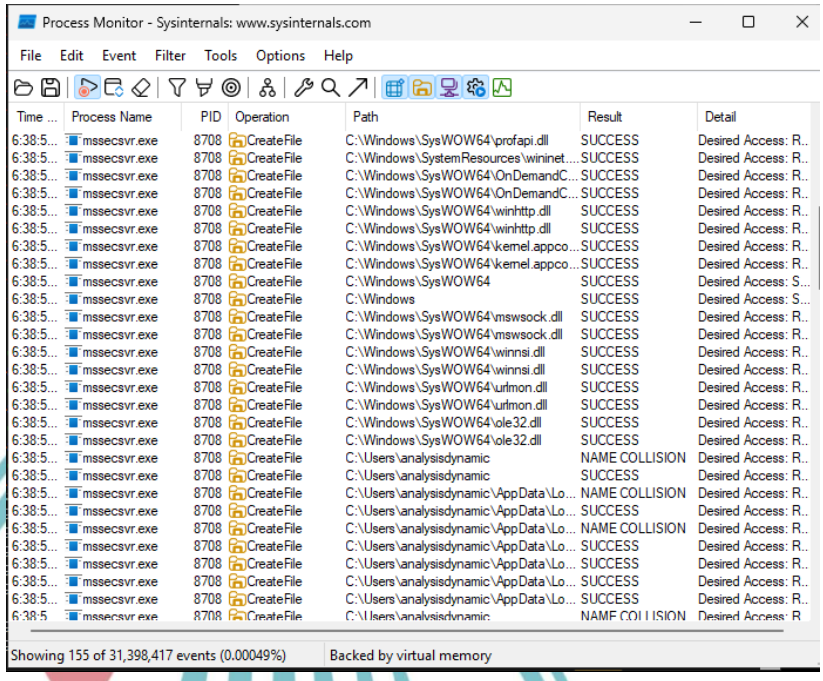

Lampiran 15 Perilaku anomali proses pada sampel 1



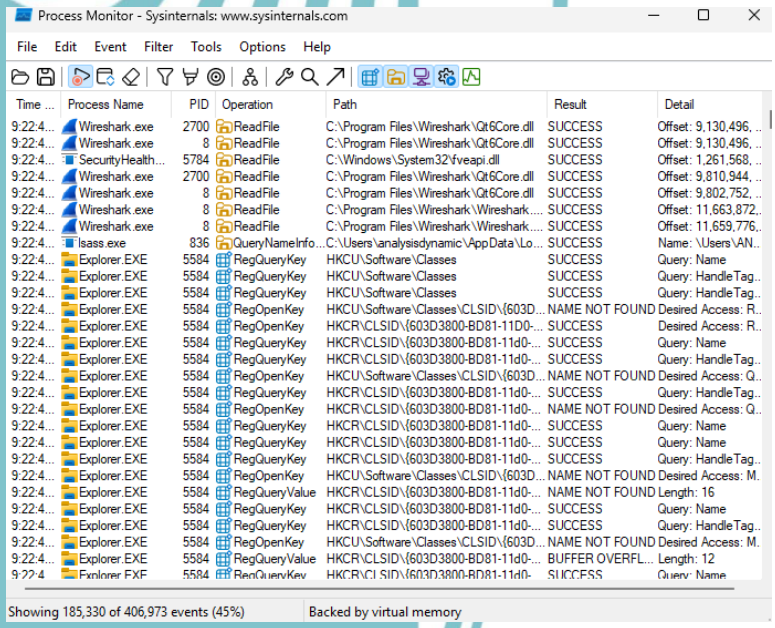

Lampiran 16 Tampilan log aktivitas Procmon pada sampel 2



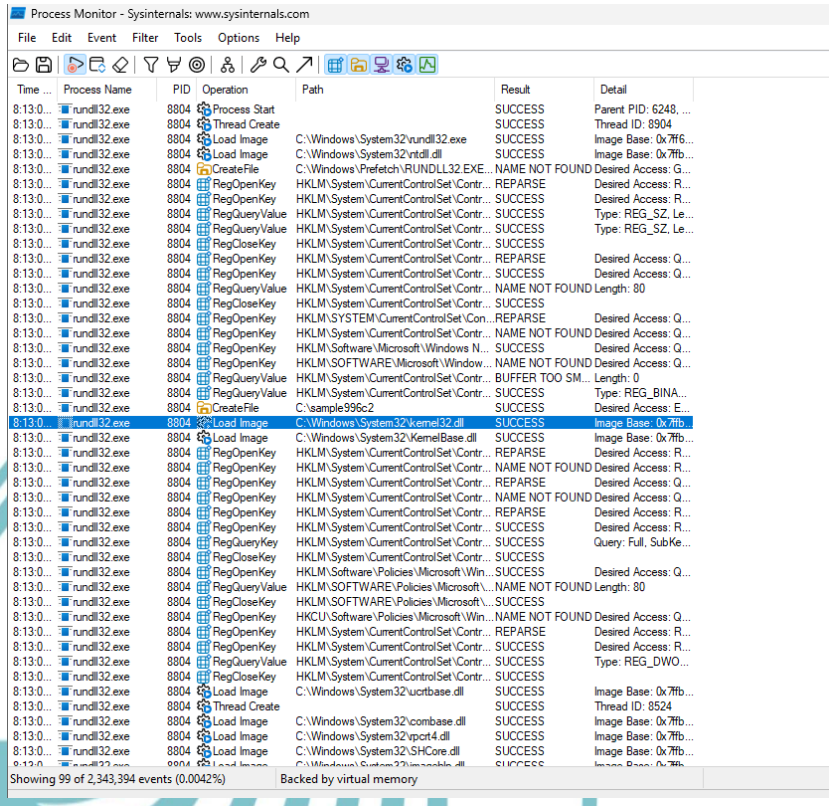

Lampiran 17 Anomali aktivitas jaringan pada sampel 3



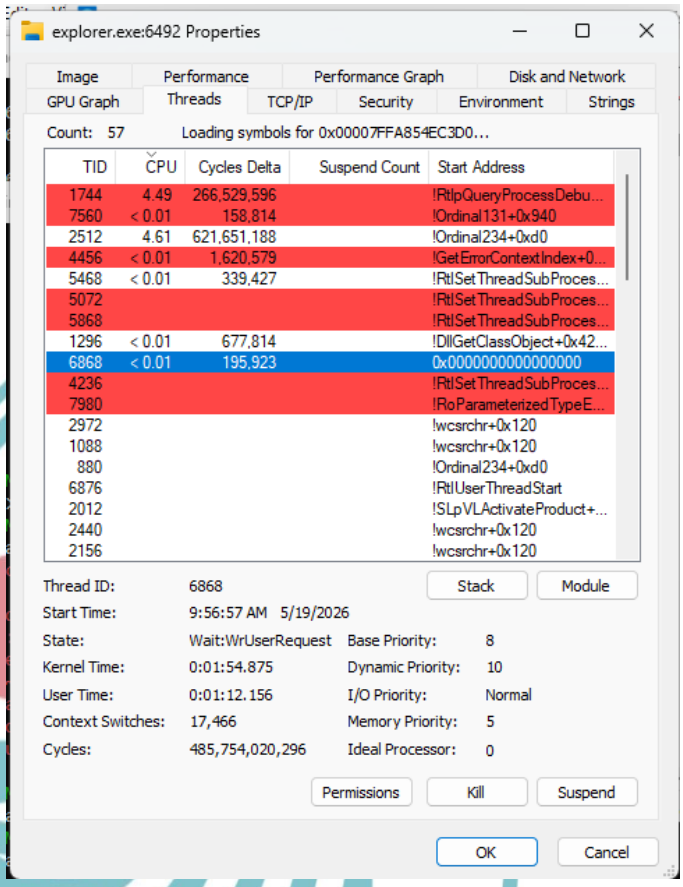

Lampiran 18 Aktivitas file system pada sampel 3



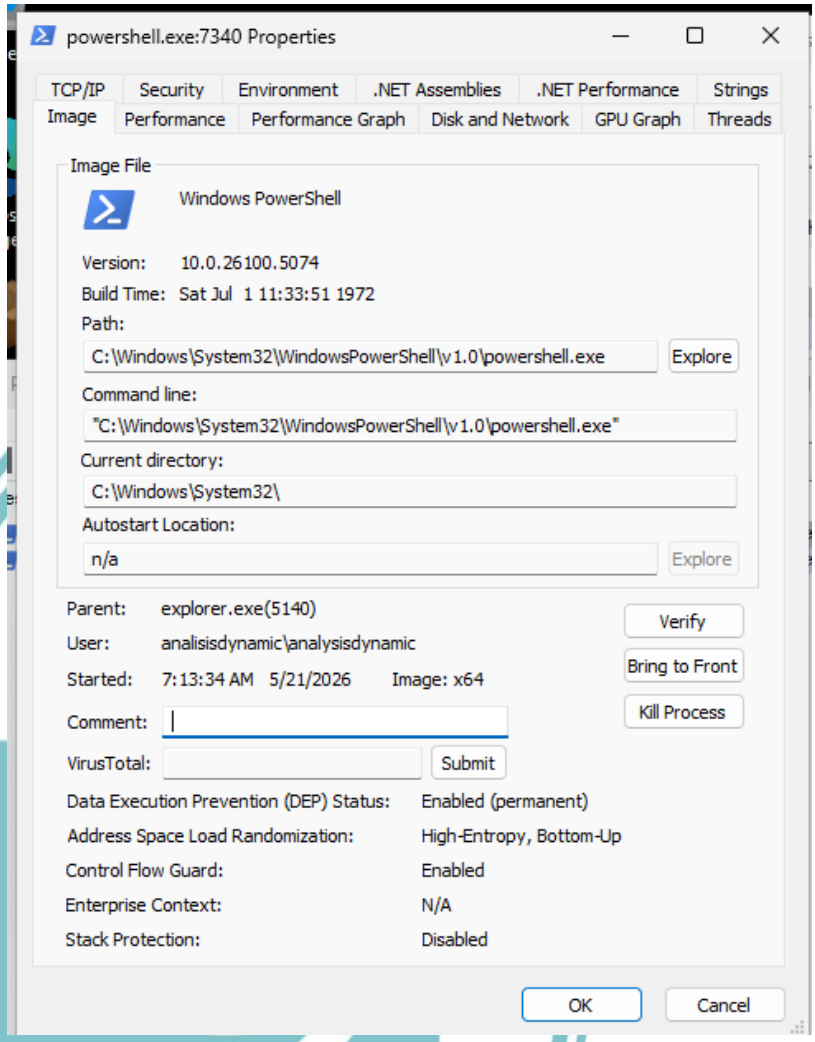

Lampiran 19 Proses monitor pada sampel 4



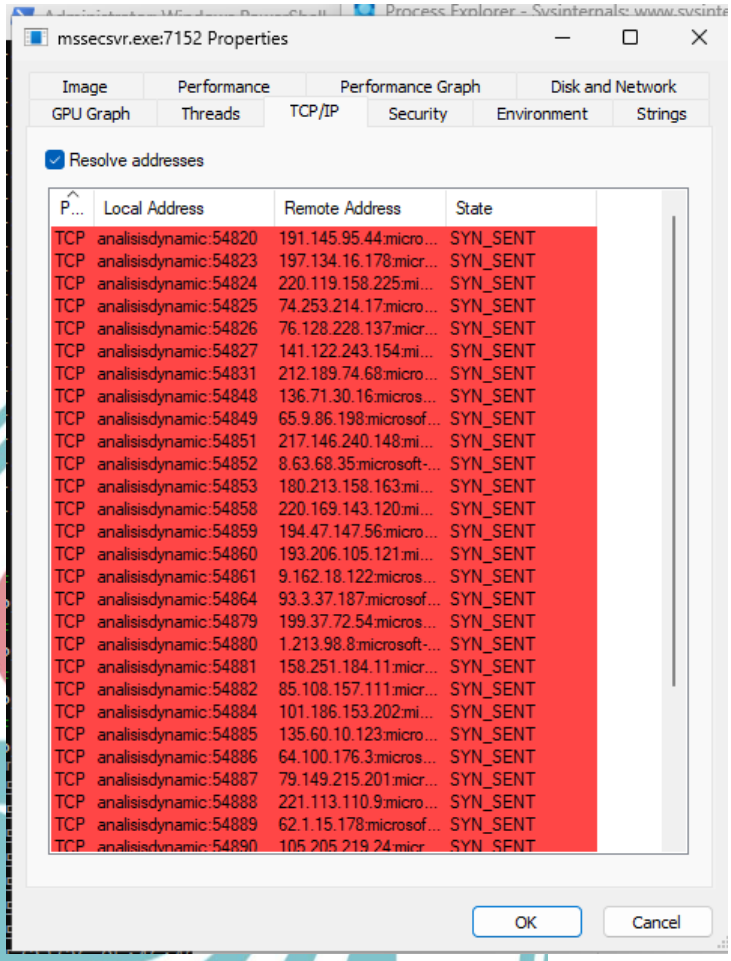

Lampiran 20 Tampilan log Procmon pada sampel 5



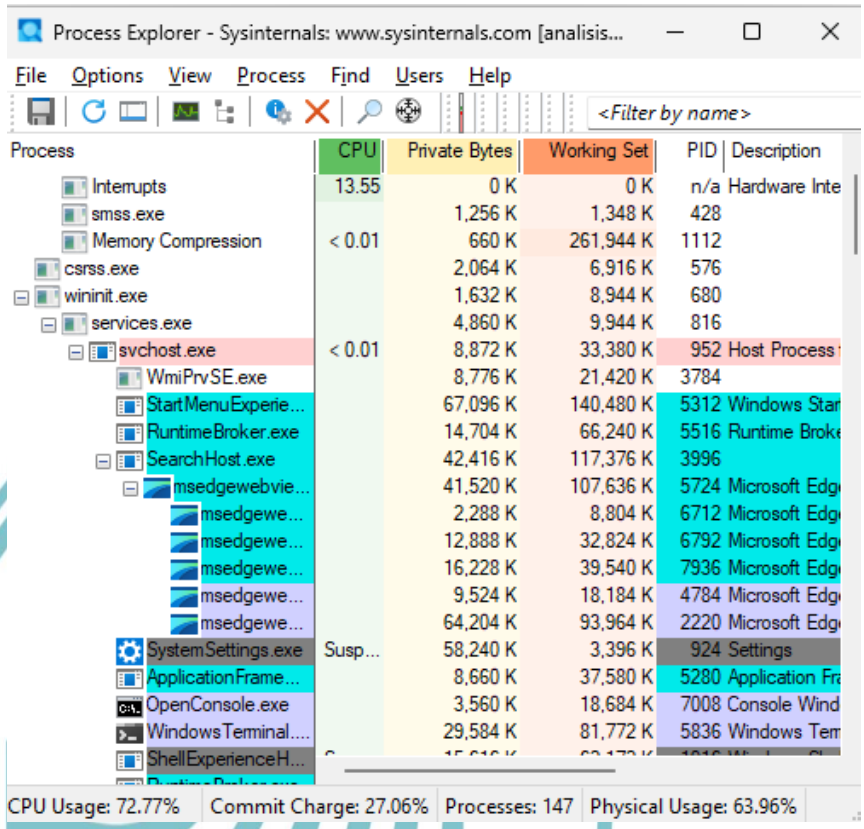

Lampiran 21 Jendela properti Process Explorer pada sampel 1



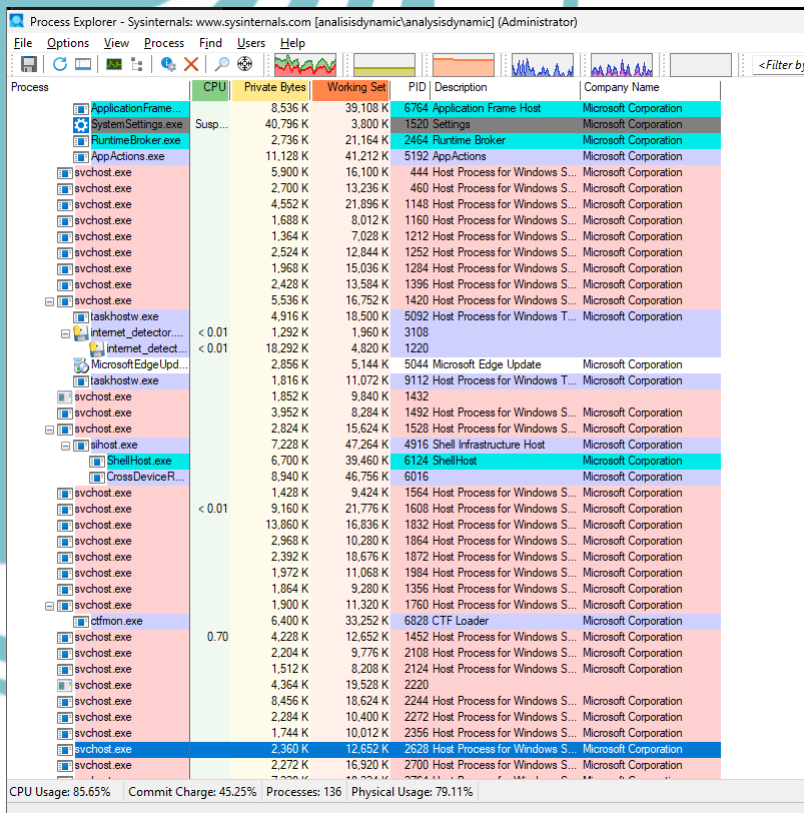

Lampiran 22 Jendela properti Process Explorer pada sampel 2



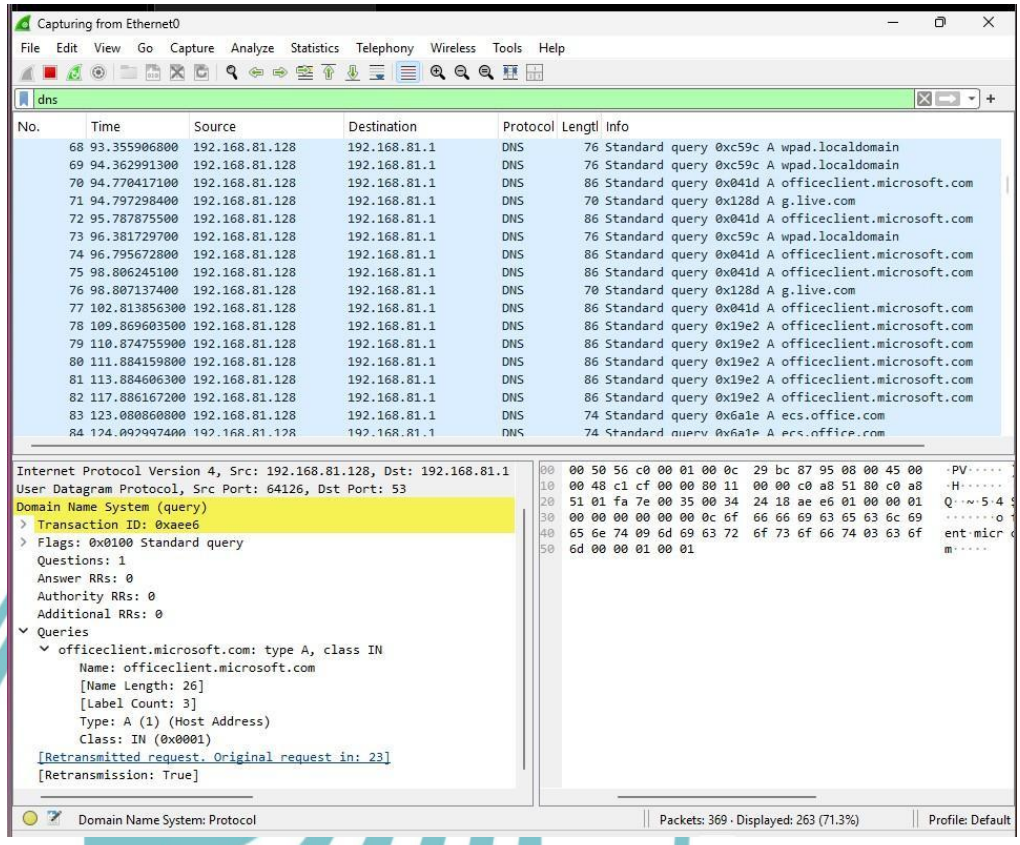

Lampiran 23 Perilaku jaringan pada sampel 3



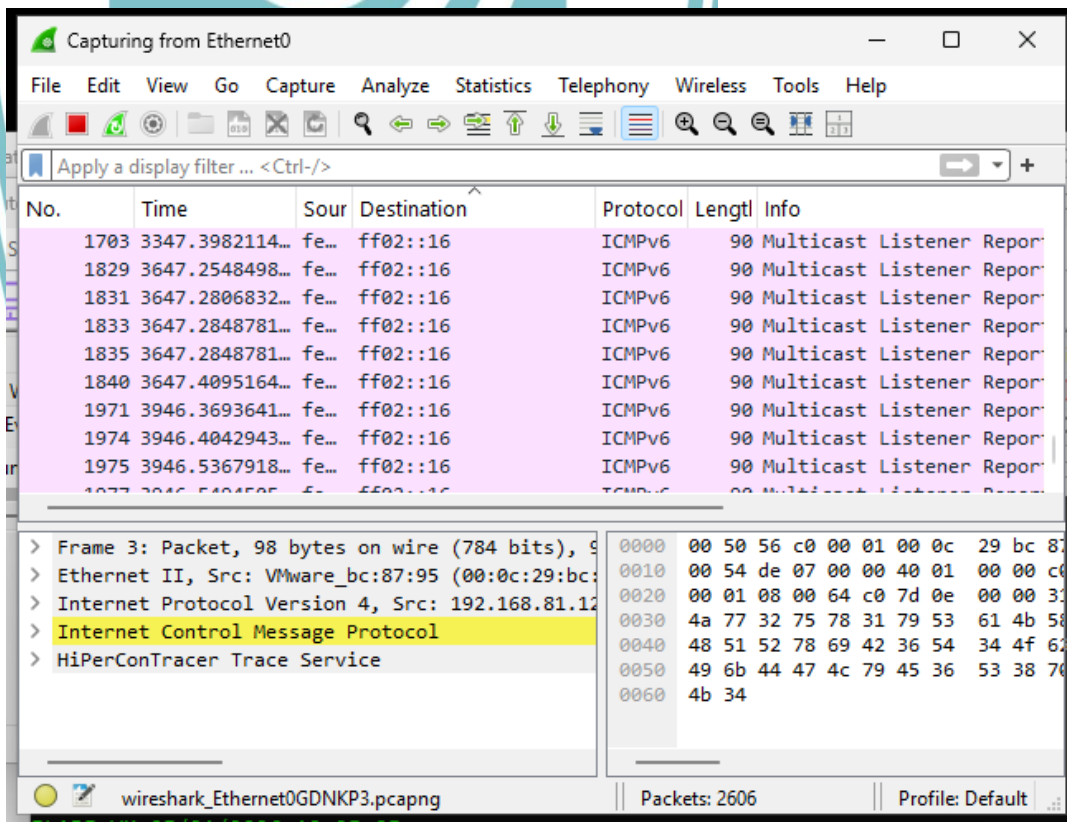

Lampiran 24 Jendela process tree pada sampel 4



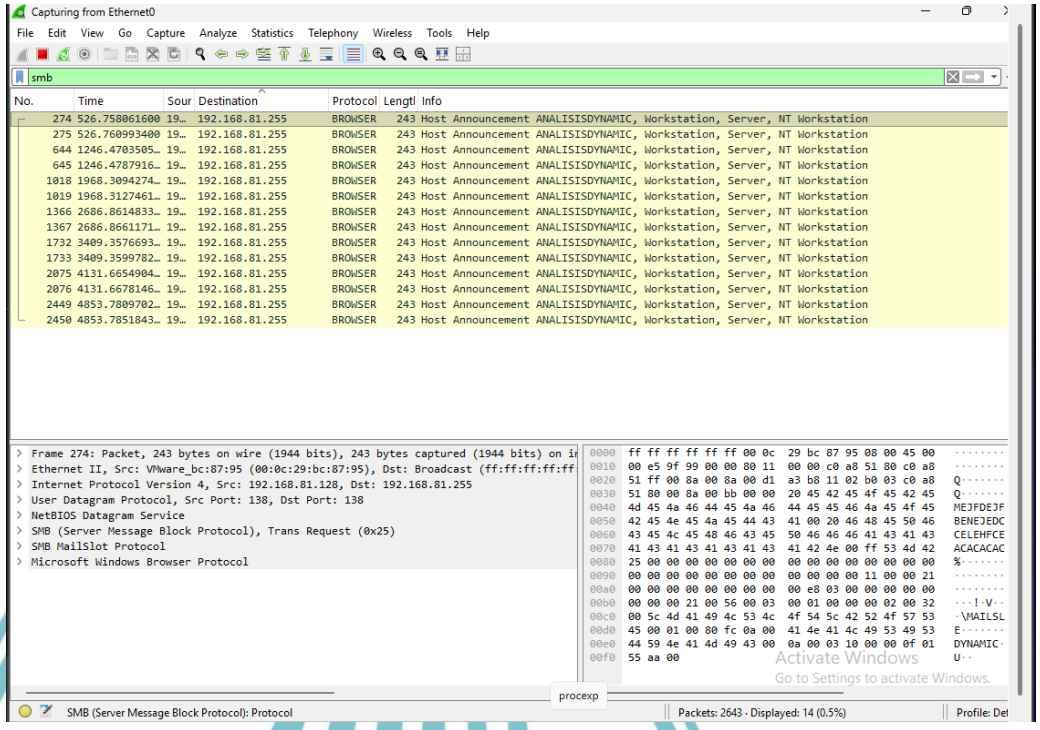

Lampiran 25 Tampilan operasi registri pada sampel 5



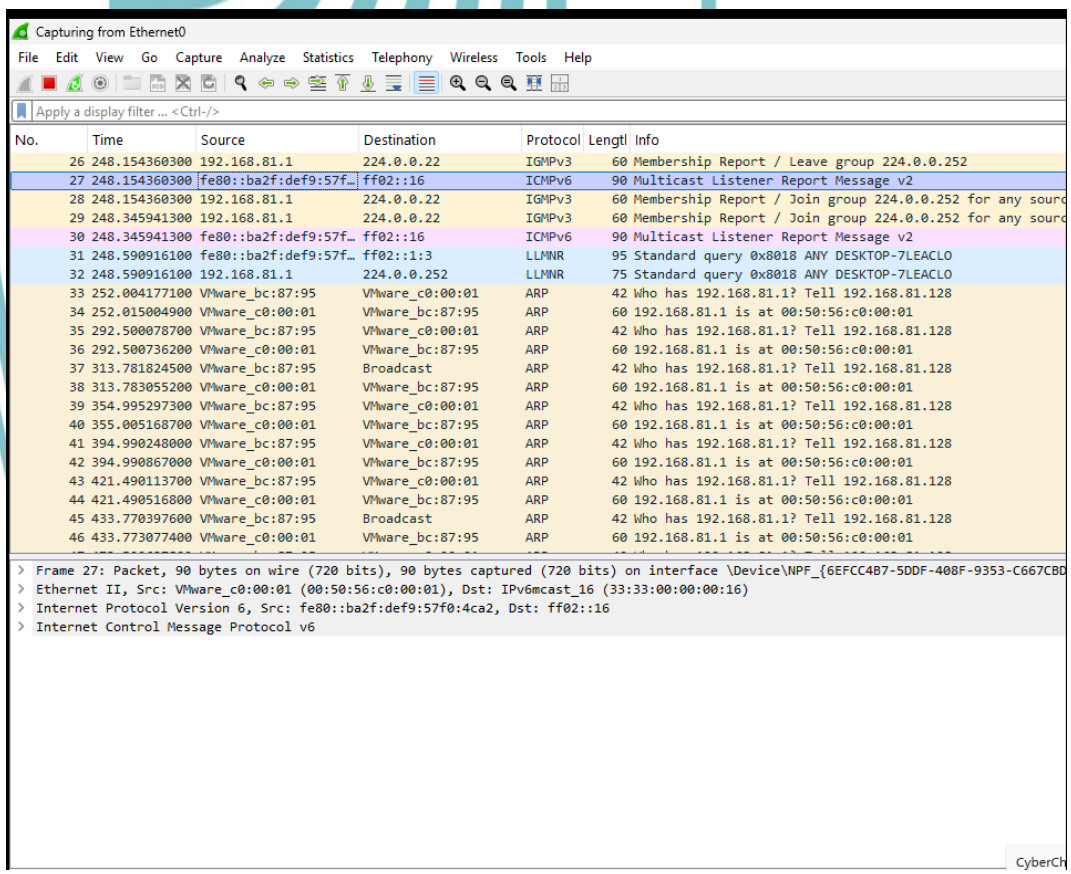

Lampiran 26 Tampilan analisa paket DNS pada sampel 1



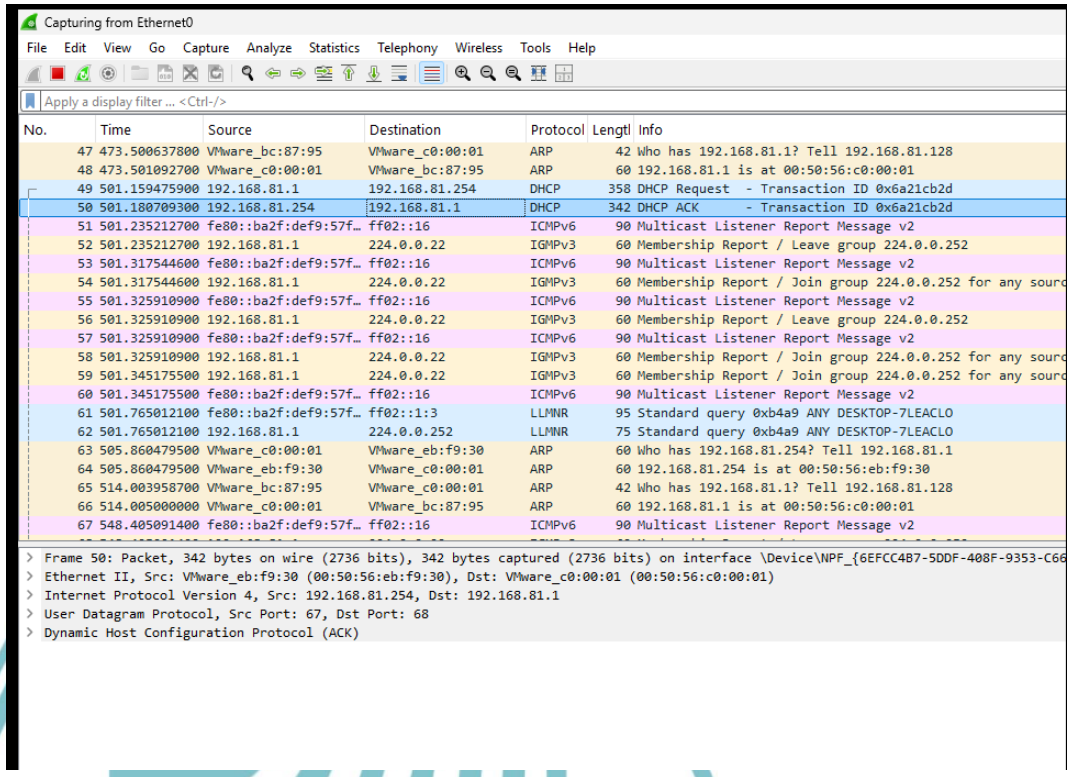

Lampiran 27 Tampilan analisis Wireshark pada sampel 2 (bagian 1)



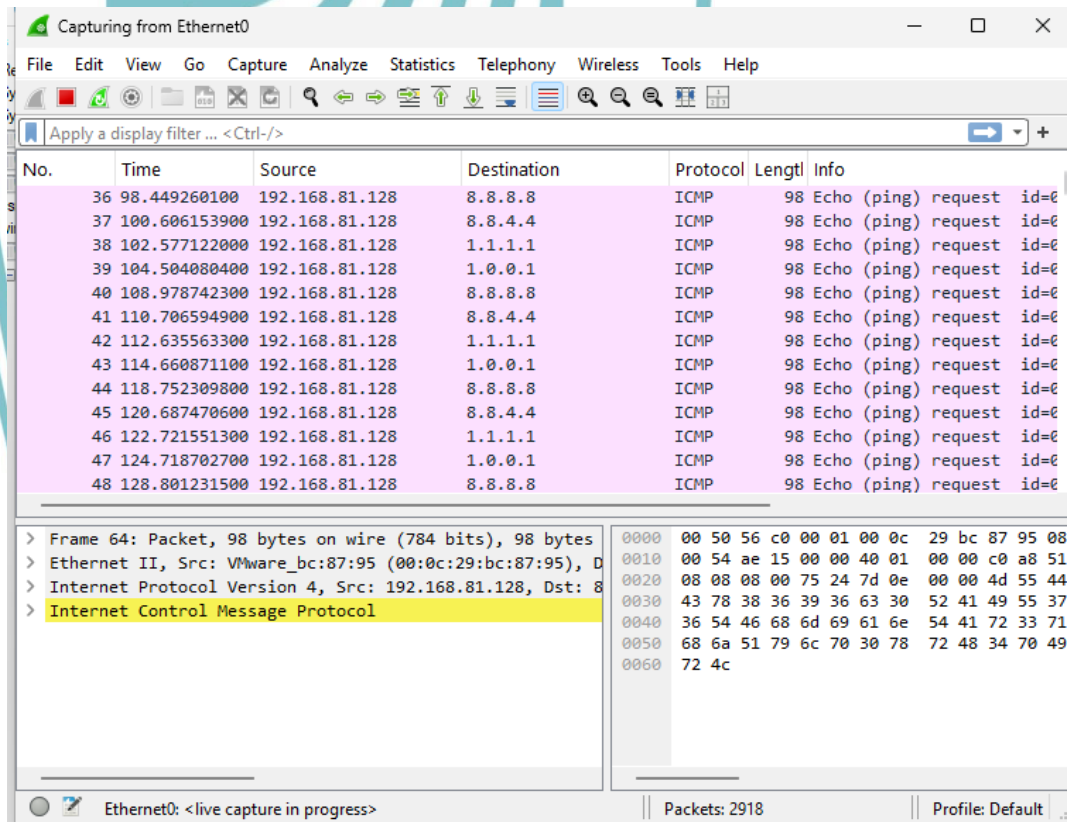

Lampiran 28 Tampilan analisis Wireshark pada sampel 2 (bagian 2)



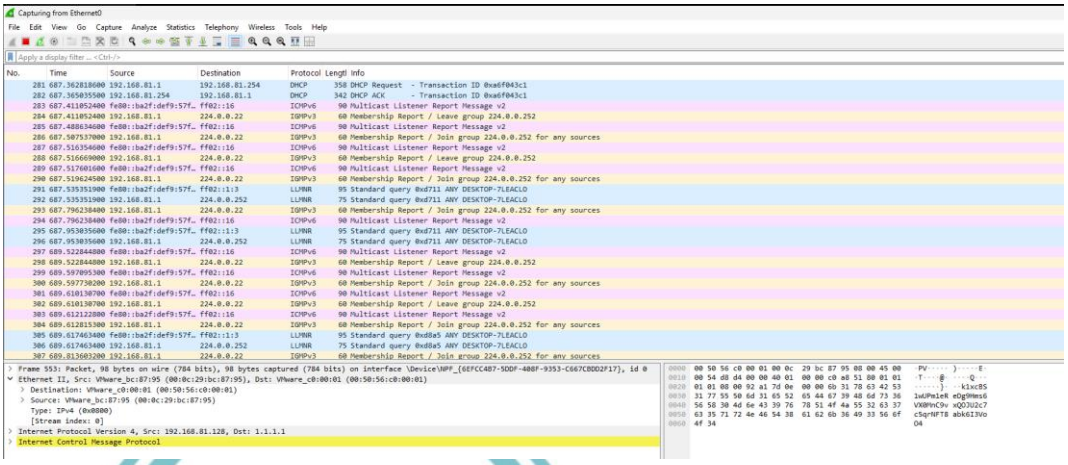

Lampiran 29 Tampilan analisis Wireshark pada sampel 3 (bagian 1)



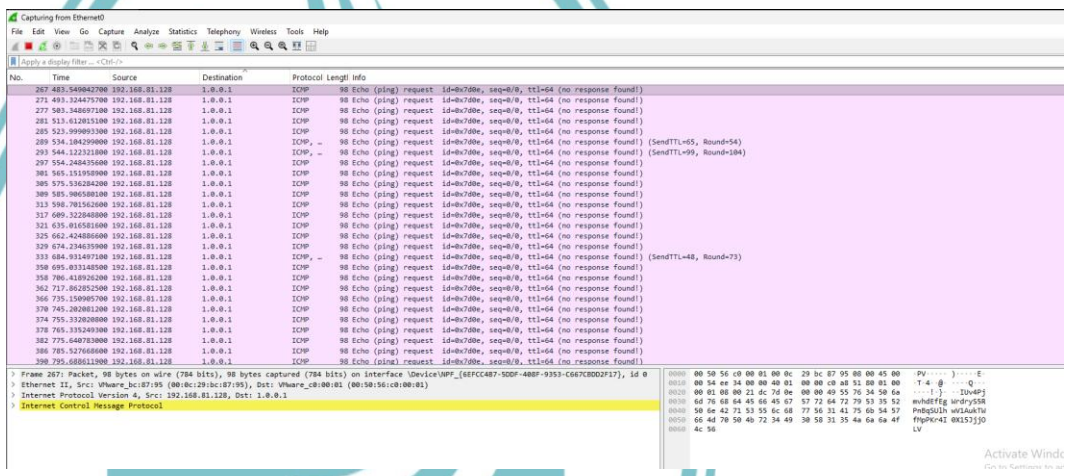

Lampiran 30 Tampilan analisis Wireshark pada sampel 3 (bagian 2)



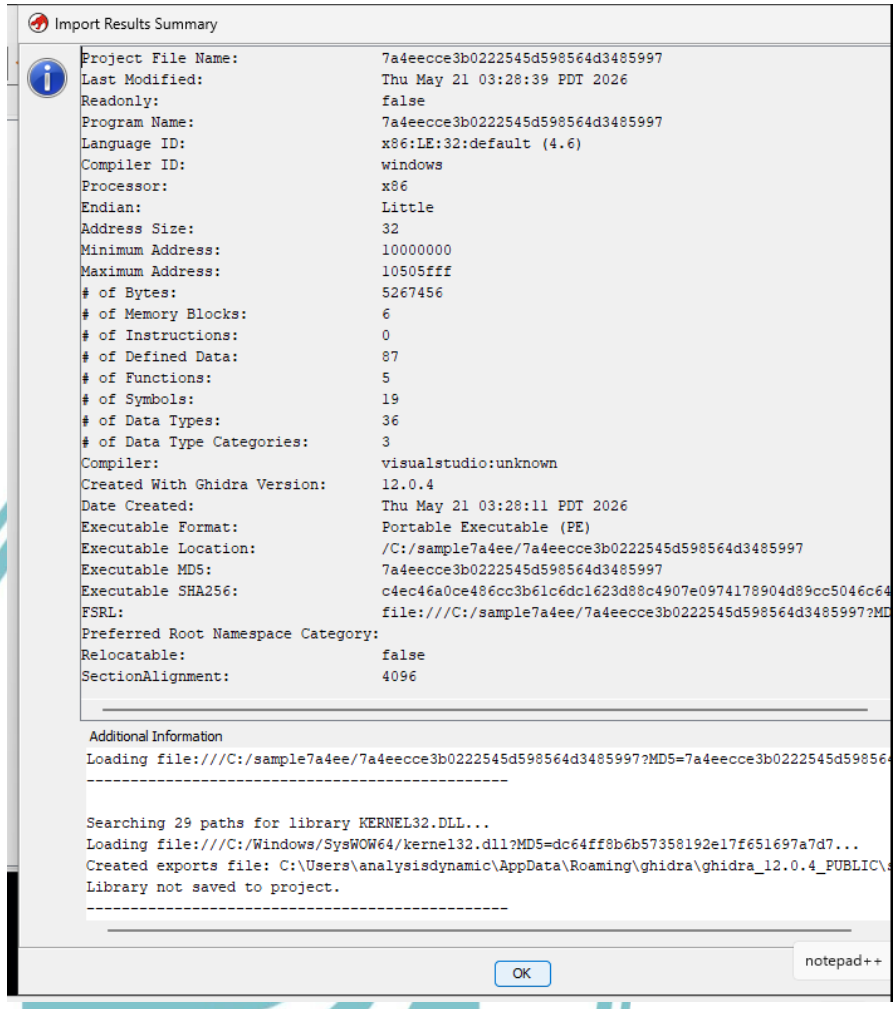

Lampiran 31 Tampilan analisis Wireshark pada sampel 4 (bagian 1)



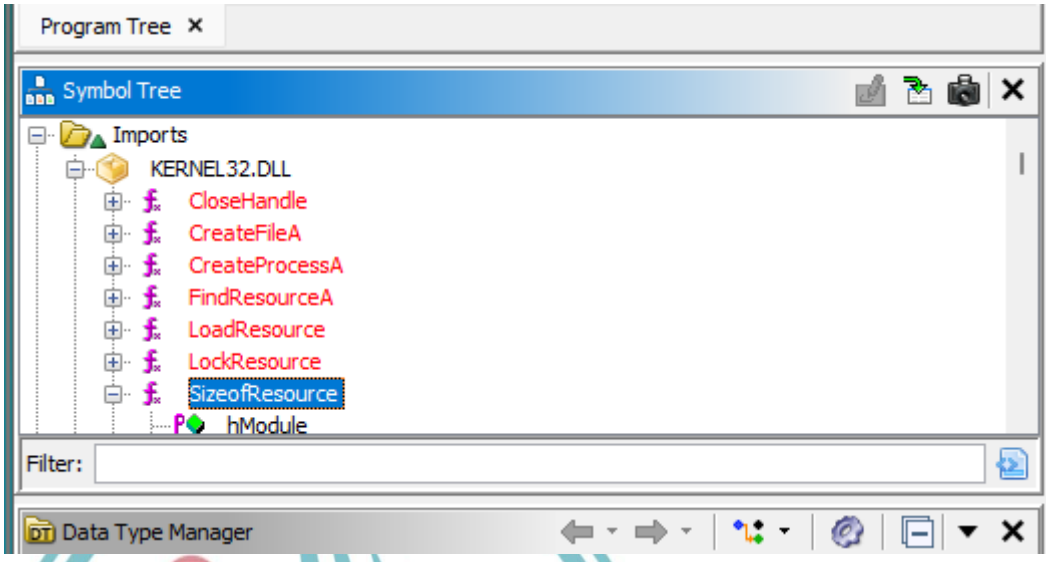
Lampiran 32 Tampilan analisis Wireshark pada sampel 4 (bagian 2)



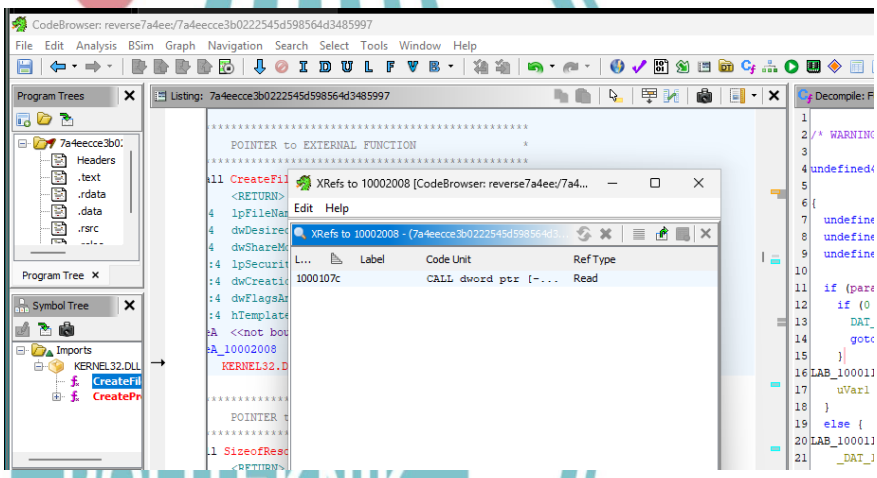
Lampiran 33 Tampilan analisis Wireshark pada sampel 5



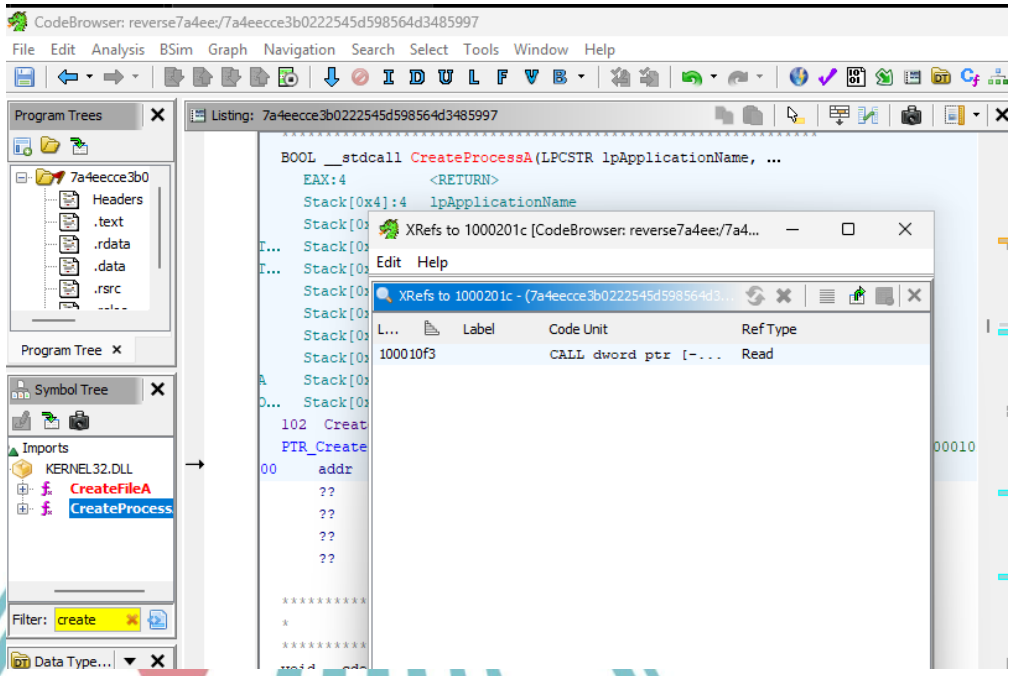

Lampiran 34 Tampilan daftar fungsi pada sampel 1



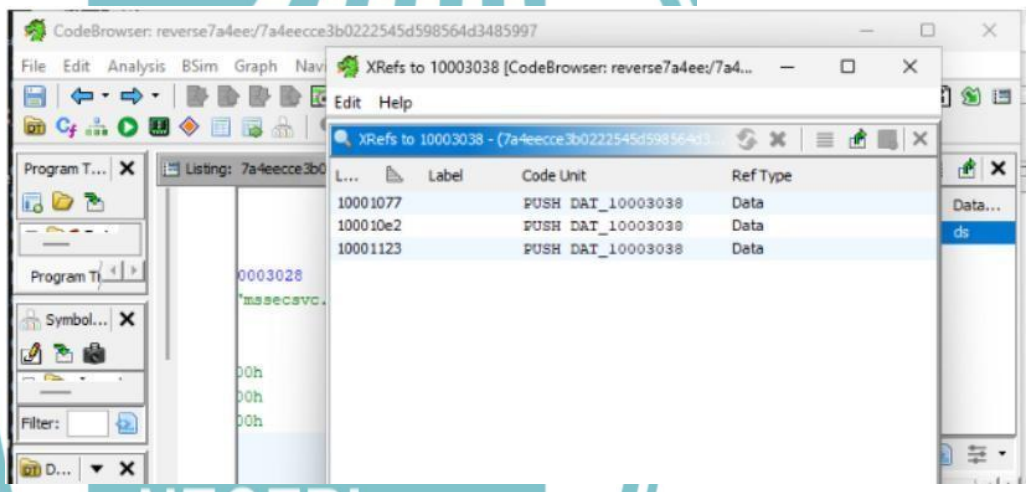

Lampiran 35 Tampilan Symbol Tree pada sampel 1



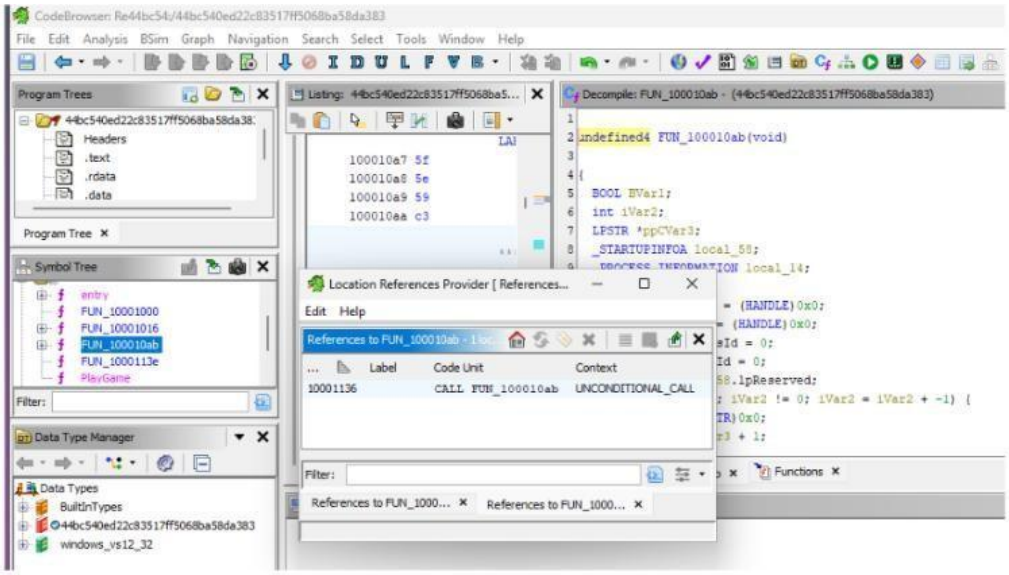

Lampiran 36 Tampilan Cross-Reference (XRefs) pada sampel 1



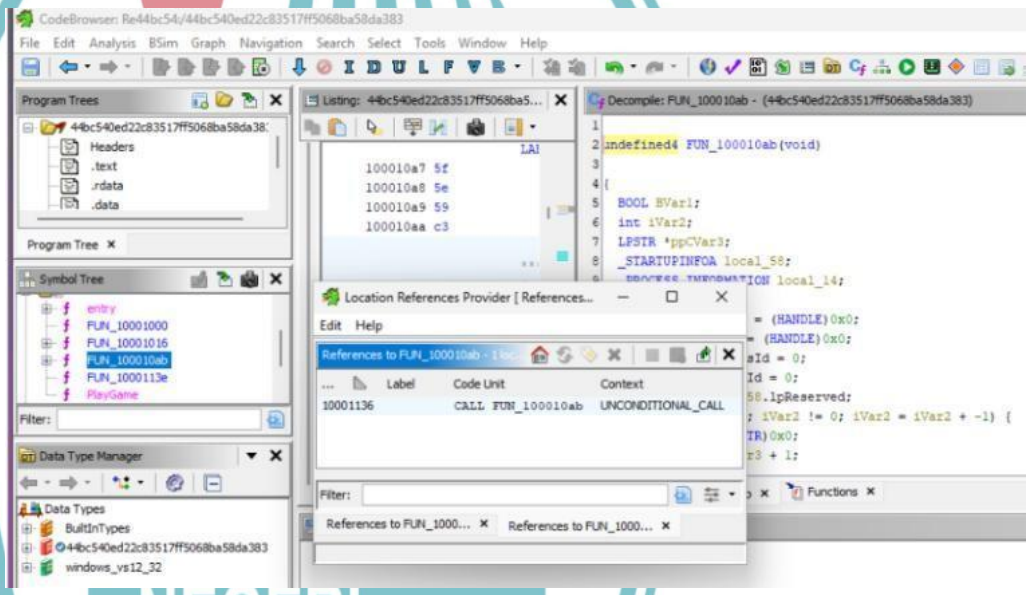

Lampiran 37 Tampilan fungsi CreateProcessA pada sampel 1



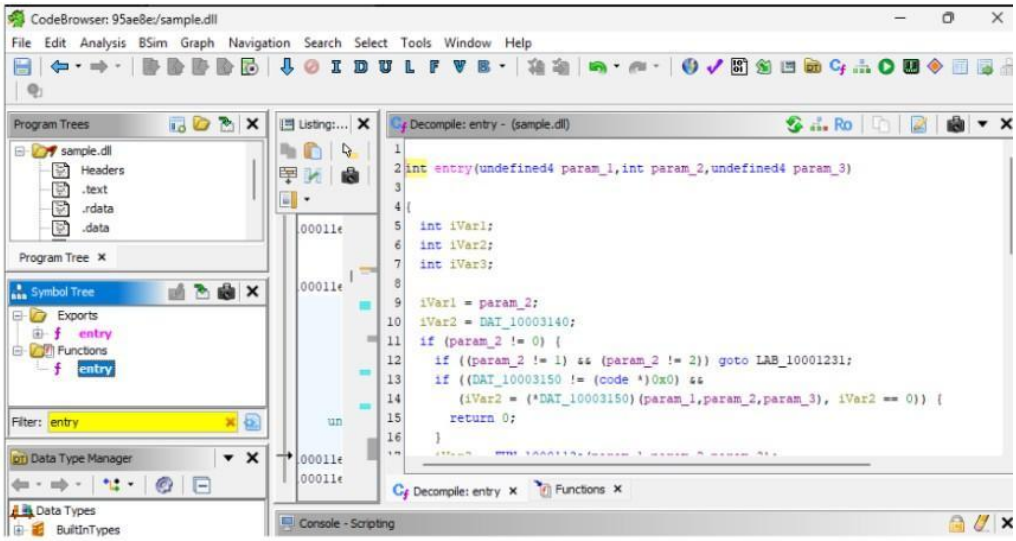

*Lampiran 38 Tampilan DAT_10003038 pada sampel 1*



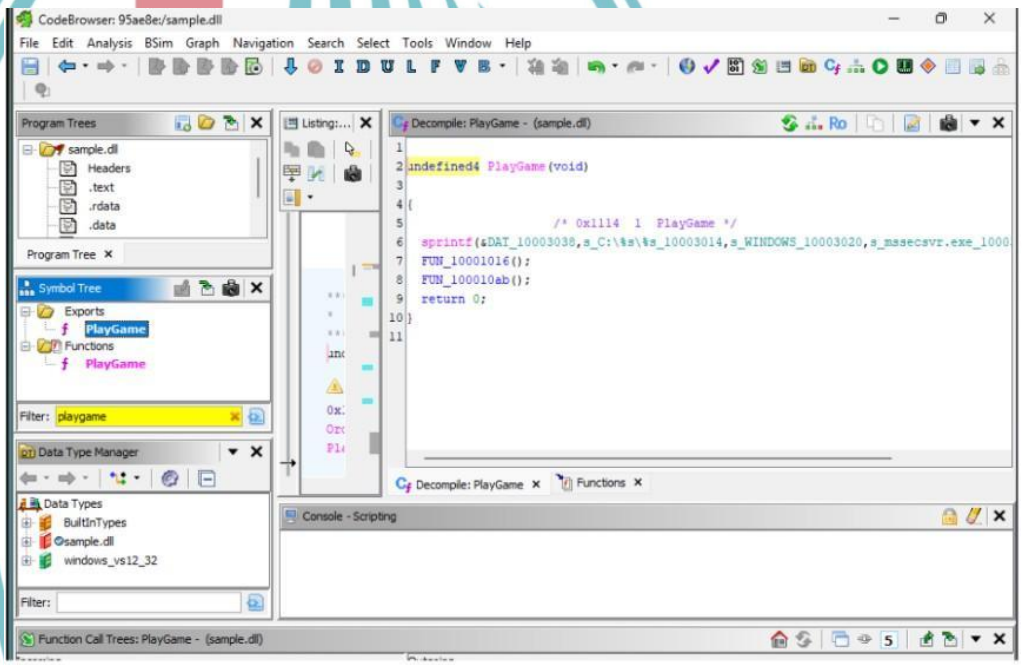
Lampiran 39 Tampilan perintah eksekusi cmd.exe pada sampel 2



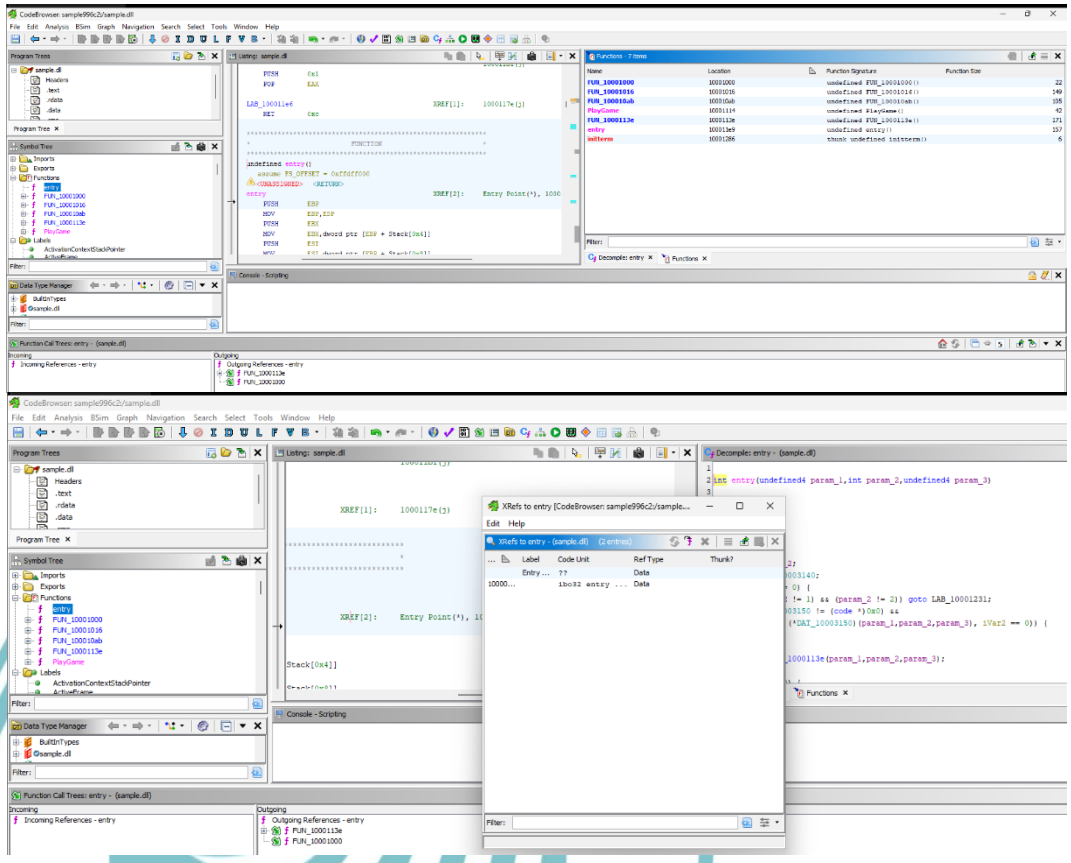
Lampiran 40 mpilan strings parameter layanan pada sampel 2




Lampiran 41 Tampilan decompile pada sampel 3




Lampiran 42 Tampilan decompile pada sampel 4



Lampiran 43 Tampilan Ghidra pada sampel 5