Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

ANALISIS MALWARE DENGAN RUNTIME DAN DYNAMIC ANALYSIS UNTUK IDENTIFIKASI IOC

Eky Januarta[1], Muhamad Azwar[2], Ondi Asroni[3], Husain[4], Lilik Widyawati[5]

[123,]Program Studi Teknologi Informasi, [45,]Program Studi Ilmu Komputer Universitas Bumigora,

Jl. Ismail Marzuki No.22, Universitas Bumigora, Cilinaya, Cakranegara, Mataram 83127
[1] ekyjanuarta@outlook.com, [2] azwar@universitasbumigora.ac.id, [3] ondi@universitasbumigora.ac.id,
[4] husain@universitasbumigora.ac.id,[5] lilikwidya@universitasbumigora.ac.id,

Abstract

The rapid advancement of digital technologies has significantly increased the threat of cyber security risks, particularly through sophisticated and harder-to-detect malware. These attacks, capable of disrupting systems, stealing sensitive data, and gaining control of devices, pose a serious threat requiring effective countermeasures. This research integrates runtime and dynamic analysis within a Sandbox environment to analyze the behavior and digital traces of malware. Experiments were conducted using five malware samples: Trojan, ransomware, spyware, worm, and botnet agents, in a QEMU/KVM-based virtualization environment with a Windows 10 guest operating system. CAPE Sandbox and Volatility Framework were used for automated malware analysis. Identified Indicators of Compromise (IOC) include file details, network activities, registry modifications, dropped files, and malware evasion techniques. The findings demonstrate that combining runtime and dynamic analysis within a Sandbox provides comprehensive, secure results, identifying compromise indicators undetectable by static methods. This approach also enhances mitigation efficiency and strengthens preparedness against cyber threats, contributing to the advancement of malware detection and improving information security systems via automated Sandbox-based digital forensics.

Keywords : *Malware Detection, Runtime Analysis, Dynamic Analysis, Sandbox, Indicators Of Compromise*

Abstrak

Kemajuan teknologi digital yang pesat telah meningkatkan ancaman terhadap keamanan siber, terutama dengan penyebaran malware yang lebih canggih dan sulit dideteksi. Serangan malware ini, yang dapat merusak sistem, mencuri data sensitif, hingga mengendalikan perangkat korban, memerlukan penanganan yang efektif. Penelitian ini menggabungkan metode runtime analysis dan dynamic analysis dalam lingkungan Sandbox untuk menganalisis perilaku dan jejak digital malware. Eksperimen dilakukan dengan lima sampel malware: Trojan, ransomware, spyware, worm, dan botnet agent, dalam lingkungan virtualisasi berbasis QEMU/KVM dengan sistem operasi Windows 10 sebagai guest. CAPE Sandbox dan Volatility Framework digunakan untuk analisis malware otomatis. IOC yang diidentifikasi meliputi detail file, aktivitas jaringan, perubahan registry, file yang dimasukkan (dropped files), serta teknik penghindaran deteksi oleh malware. Hasil penelitian menunjukkan bahwa penggabungan metode runtime dan dynamic analysis dalam lingkungan Sandbox memberikan hasil analisis yang komprehensif dan aman, mengidentifikasi indikator kompromi yang tidak terdeteksi dengan metode statis. Pendekatan ini meningkatkan efisiensi mitigasi dan kesiapsiagaan terhadap ancaman siber, serta memberikan kontribusi dalam pengembangan metode deteksi malware dan penguatan sistem keamanan informasi melalui analisis forensik digital berbasis Sandbox.

Kata Kunci : *Deteksi Malware, Analisis Runtime, Analisis Dinamis, Sandbox, Indikator Kompromi*

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

1. PENDAHULUAN

Malicious software (*malware*) atau perangkat lunak berbahaya adalah program yang dirancang dengan tujuan untuk merusak, mengeksploitasi, atau mengkompromikan sistem. *Malware* mencakup berbagai jenis perangkat lunak seperti virus, *worm*, *logic bomb*, *Trojan horse*, *backdoor*, *mobile code*, dan *multiple-threat malware*. Serangan *malware* dapat mengganggu operasi komputer, mencuri data sensitif, atau memberikan akses tidak sah ke sistem komputer target. Dalam beberapa dekade terakhir, serangan *malware* mengalami peningkatan yang signifikan, terutama dengan munculnya *malware* yang semakin canggih dan sulit terdeteksi. [1]

Data dari laporan *Malware Statistics* & *Trends Report* oleh AV-TEST [2] menunjukkan bahwa lebih dari 450.000 jenis *malware* baru terdaftar setiap hari, yang mengindikasikan adanya lonjakan serangan berbahaya di seluruh dunia . Di Indonesia, laporan Lanskap Keamanan Siber Indonesia 2024 yang dirilis oleh Badan Siber dan Sandi Negara (BSSN) mencatat bahwa pada tahun 2024, *malware* mendominasi kategori trafik anomali dengan 5 dari 10 jenis trafik teratas merupakan serangan *malware* [3]. Fenomena ini menunjukkan bahwa ancaman *malware* menjadi masalah serius dalam keamanan siber, baik di tingkat personal maupun organisasi.

Seiring berjalannya waktu, *malware* terus berkembang dan beradaptasi, menghasilkan berbagai jenis *malware* dengan teknik yang semakin kompleks yang menargetkan berbagai sistem dengan metode yang lebih canggih [4]. Salah satu faktor penting yang mempengaruhi efektivitas serangan *malware* adalah sistem operasi yang digunakan. Berdasarkan data dari *Statcounter Global Stats*, *Windows* tetap menjadi sistem operasi yang paling banyak digunakan di dunia dengan pangsa pasar sebesar 77,12%, dan di Indonesia, pengguna *Windows* mencapai 87,07% [5].

Serangan *malware* yang terus berkembang ini memerlukan penerapan metode analisis yang efektif dan aman untuk menangani insiden-insiden yang terkait dengan ancaman siber. *Malware*, sebagai perangkat lunak berbahaya, dapat merusak operasi komputer, mencuri data sensitif, dan memberi akses tidak sah ke sistem yang diserang. Oleh karena itu, pengenalan *Indicator of Compromise* (IOC) menjadi langkah yang sangat penting dalam mendeteksi dan menangani serangan *malware*. IOC berfungsi sebagai petunjuk yang dapat digunakan untuk mendeteksi adanya kompromi dalam sistem yang diserang, seperti file yang terinfeksi, aktivitas jaringan yang mencurigakan, atau perubahan pada *registry* sistem [6].

Salah satu pendekatan yang sangat efektif untuk mengidentifikasi IOC adalah dengan menggunakan metode *runtime analysis* dan *dynamic analysis* dalam *sandbox environment*. *Sandbox* adalah lingkungan terisolasi yang memungkinkan peneliti untuk mengeksekusi malware dan memantau aktivitasnya tanpa mengganggu sistem utama. Dengan pendekatan ini, peneliti dapat mempelajari perilaku *malware* secara langsung selama eksekusinya, termasuk dampaknya terhadap file, proses, *registry*, dan jaringan [7]. *Runtime analysis* pada dasarnya melibatkan eksekusi program yang dicurigai mengandung *malware* untuk mengamati perilakunya secara langsung dan menganalisis dampak yang ditimbulkan pada sistem. Berbeda dengan *surface analysis* yang hanya mengamati ciri-ciri program tanpa eksekusi, *runtime analysis* memberikan kajian yang lebih mendalam karena memungkinkan pengamatan terhadap aktivitas *malware* secara *real-time*, seperti perubahan *registry*, file, proses, dan komunikasi jaringan [8]. Sebagai contoh, penelitian oleh Panjaitan et al. [9] menggunakan alat seperti *Regshot*, *CaptureBAT*, dan *Noriben Malware Analysis* untuk memantau perubahan *registry* dan aktivitas *malware* dalam sistem *Windows*. Hasil penelitian tersebut menunjukkan bahwa metode ini berhasil mengidentifikasi *malware* berdasarkan perubahan yang terjadi selama *runtime*, seperti penambahan entri pada *registry* dan file yang menggandakan diri ke dalam folder sistem.

Selain itu, *dynamic analysis* memungkinkan pemantauan *real-time* terhadap aktivitas *malware*, seperti perubahan file, aktivitas proses, dan komunikasi jaringan selama eksekusi. Dengan menggunakan alat seperti *Cuckoo Sandbox*, teknik ini dapat memberikan gambaran mendalam mengenai tujuan dan modus operandi *malware*, apakah untuk mencuri data, merusak sistem, atau menyebarkan serangan lebih lanjut. [10] Penelitian oleh dan Wahidin et al. [11] menunjukkan bahwa *dynamic analysis* menggunakan *Cuckoo Sandbox* sangat efektif dalam mengidentifikasi pola-pola enkripsi file, penyebaran *malware*, serta komunikasi *command* and *control* (C2). Hal ini menegaskan pentingnya *dynamic analysis* dalam mendeteksi *malware* yang

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

menggunakan teknik *anti-debugging* dan *obfuscation*, yang sulit terdeteksi menggunakan analisis statis saja.

Dengan demikian, integrasi antara *runtime analysis* dan *dynamic analysis* dalam *sandbox environment* memberikan keunggulan yang lebih kuat dan akurat dalam mengidentifikasi IOC dari *malware*. Pendekatan ini memungkinkan analisis yang lebih mendalam dan aman terhadap perilaku *malware*, serta menyediakan data yang sangat dibutuhkan untuk proses mitigasi dan penanggulangan ancaman siber yang semakin berkembang. Selain itu, metode ini juga memungkinkan identifikasi pola serangan yang lebih kompleks dan memberikan wawasan lebih lanjut dalam menghadapi ancaman siber yang terus berkembang. Penelitian ini bertujuan untuk mengintegrasikan metode *runtime analysis* dan *dynamic analysis* dalam *sandbox environment* untuk mendeteksi dan mengidentifikasi *Indicator of Compromise* (IOC) dari *malware*. Pendekatan ini diharapkan dapat memperkuat sistem deteksi dan mitigasi serangan *malware*, serta memberikan kontribusi signifikan terhadap peningkatan kemampuan sistem keamanan siber secara keseluruhan.

2. TINJAUAN PUSTAKA

2.1. Analisis Malware

Analisis *malware* adalah suatu proses yang bertujuan untuk memeriksa perilaku dan karakteristik dari perangkat lunak berbahaya (malware). Menurut Siddiq et al. [12], analisis merupakan proses penyelidikan yang dilakukan untuk mengungkapkan keadaan suatu peristiwa atau objek yang sedang diteliti, dalam hal ini adalah *malware*. Salah satu teknik yang sering digunakan dalam analisis malware adalah analisis statis dan analisis dinamis. Analisis statis hanya melihat file atau kode *malware* tanpa mengeksekusinya, sedangkan analisis dinamis berfokus pada observasi perilaku *malware* selama eksekusinya .

2.2. Runtime Analysis

Runtime analysis adalah teknik yang digunakan untuk memantau dan menganalisis *malware* selama eksekusinya. Teknik ini sangat berguna untuk mengidentifikasi perubahan yang terjadi pada file dan *registry* serta perilaku *malware* yang dapat mengganggu operasi sistem. [13] Panjaitan et al. [9] menyatakan bahwa *runtime analysis* memungkinkan peneliti untuk mendapatkan informasi yang lebih mendalam tentang bagaimana *malware* memanipulasi atau memodifikasi data dan proses yang berjalan pada sistem. Metode ini memiliki keunggulan dibandingkan analisis statis, karena dapat memantau perubahan yang terjadi secara langsung saat *malware* dijalankan. *Runtime analysis* juga memungkinkan peneliti untuk mengidentifikasi teknik penyamaran yang digunakan *malware* untuk menghindari deteksi. Sebagai contoh, *malware* yang menggandakan diri di folder sistem atau menambah entri *registry* yang mencurigakan dapat terdeteksi melalui pendekatan ini.

2.3. Dynamic Analysis

Dynamic analysis adalah metode yang digunakan untuk memantau aktivitas *malware* dalam waktu nyata selama eksekusi, dengan tujuan untuk mendeteksi perilaku berbahaya yang mungkin tidak terlihat dalam analisis statis. [14] Penelitian oleh Manoppo et al. [8] menunjukkan bahwa *dynamic analysis* sangat efektif untuk mengidentifikasi *malware* yang menggunakan teknik penyembunyian, seperti *obfuscation* dan *anti-debugging*. Dengan menggunakan *Cuckoo Sandbox* dalam analisis dinamis, peneliti dapat mengamati *API calls*, aktivitas memori, serta interaksi *malware* dengan jaringan. Hasil penelitian ini juga menunjukkan bahwa *dynamic analysis* membantu mengidentifikasi *Indicators of Compromise* (IOC) yang menunjukkan adanya serangan yang dilakukan oleh *malware*. Hal ini sangat penting untuk mendeteksi aktivitas berbahaya yang terjadi secara aktif dalam sistem dan mencegah kerusakan lebih lanjut.

2.4. Malware

Malware, singkatan dari "*malicious software*", adalah perangkat lunak yang dirancang untuk merusak, mengakses, atau mengeksploitasi sistem komputer tanpa izin dari pengguna. [15] Menurut Ibrahim et al. [1], malware dapat berupa virus,

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

worm, *Trojan horse*, *ransomware*, *spyware*, dan jenis perangkat lunak berbahaya lainnya. Serangan *malware* dapat mengganggu operasi sistem, mencuri data sensitif, atau memberi akses tidak sah ke perangkat yang diserang. *Malware* umumnya digunakan untuk mencuri informasi pribadi atau melakukan tindakan merusak yang dapat merugikan korban, seperti penghancuran data atau penggunaan perangkat untuk serangan lebih lanjut.

2.5. Indicator of Compromise (IOC)

Indicators of Compromise (IOC) adalah bukti yang menunjukkan bahwa sebuah sistem telah terinfeksi oleh *malware*. IOC dapat berupa alamat IP, file *hash*, nama file, entri *registry*, dan tanda-tanda lain yang dapat diidentifikasi selama analisis *malware* [16]. Menurut Dwi et al [6], pengidentifikasian IOC sangat penting untuk mendeteksi dan merespon serangan *malware* dengan cepat. IOC dapat memberikan informasi yang diperlukan untuk memitigasi ancaman lebih awal dan memulihkan sistem yang terinfeksi. Penggunaan *sandbox environment* seperti *Cuckoo Sandbox* memungkinkan peneliti untuk mengidentifikasi IOC dengan lebih cepat dan akurat, karena alat ini dapat memonitor berbagai aktivitas *malware* secara dinamis dan mengekstrak indikator yang relevan untuk mendeteksi ancaman.

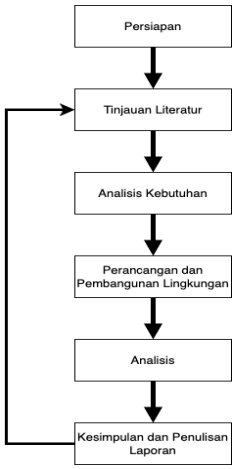
2.6. CAPE Sandbox

CAPE *Sandbox* merupakan sistem analisis *malware* otomatis yang bersifat *open-source*, yang dikembangkan sebagai pengembangan lebih lanjut dari *Cuckoo Sandbox*. Tujuan utama dari CAPE *Sandbox* adalah meningkatkan kemampuan dalam proses *automated malware unpacking* dan ekstraksi konfigurasi *malware*. Sistem ini bekerja dengan menganalisis file berbahaya dan mengumpulkan hasil analisis yang menggambarkan tindakan yang dilakukan oleh *malware* saat dijalankan dalam lingkungan sistem operasi *Windows* yang terisolasi. Arsitektur CAPE *Sandbox* terdiri dari mesin *host* yang bertugas menjalankan komponen inti dari *Sandbox*, serta satu atau lebih mesin *guest* yang digunakan untuk menjalankan sampel *malware* secara terpisah dan aman. [17]

3. METODOLOGI PENELITIAN

3.1. Tahapan Penelitian

Penelitian menggunakan metodologi eksperimen terapan. Tahapan yang ada dalam penelitian dapat ditemukan pada Gambar 1 berikut ini.



Gambar 1. Alur Tahapan Penelitian [18]

1. Persiapan
   Pada tahap awal, peneliti mengidentifikasi latar belakang masalah, rumusan masalah, dan tujuan penelitian. Hal ini penting untuk memahami konteks analisis dan menentukan arah penelitian. Latar belakang menggambarkan masalah, tujuan penelitian menjelaskan hasil yang diinginkan, dan rumusan masalah membantu memfokuskan penelitian.
2. Tinjauan Literatur
   Peneliti meninjau jurnal dan penelitian terkait untuk memahami perkembangan terbaru, metode yang digunakan, serta hasil yang diperoleh oleh peneliti lain. Tinjauan ini membantu mengidentifikasi celah dalam penelitian sebelumnya yang dapat menjadi dasar penelitian ini serta merumuskan hipotesis yang akan diuji.
3. Analisis Kebutuhan
   Tahap ini bertujuan untuk menganalisis kebutuhan penelitian, baik itu kebutuhan lingkungan penelitian maupun sumber daya yang diperlukan. Pada penelitian ini, analisis kebutuhan akan melibatkan identifikasi alat dan

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

lingkungan yang dibutuhkan untuk melakukan analisis malware secara runtime dan dinamis dalam lingkungan *Sandbox*.

4. Perancangan dan Pembangunan Lingkungan

   Pada tahap ini, peneliti menganalisis kebutuhan yang diperlukan untuk menjalankan penelitian, baik itu alat (software), lingkungan, maupun perangkat keras (hardware). Dalam penelitian ini, kebutuhan alat meliputi perangkat lunak dan alat analisis yang digunakan untuk mengidentifikasi dan memantau *malware* dalam lingkungan yang terisolasi (Sandbox environment). Selain itu, perangkat keras yang diperlukan untuk mendukung eksperimen, seperti komputer dengan spesifikasi yang memadai untuk menjalankan analisis *malware* secara *real-time*, juga harus dipertimbangkan. Tahap ini bertujuan untuk memastikan bahwa seluruh sumber daya yang diperlukan untuk penelitian dapat disediakan dan dikonfigurasi dengan tepat guna menghasilkan hasil yang valid.

5. Analisis

   Pada tahap ini, peneliti melakukan analisis data yang diperoleh dari eksperimen di lingkungan *Sandbox*. *Malware* yang dianalisis akan diperiksa untuk mencari IOC, seperti alamat IP, file *hash*, dan indikasi lainnya yang dapat mengungkapkan aktivitas berbahaya. Analisis ini sangat bergantung pada pemrosesan data secara dinamis untuk mengidentifikasi pola dan taktik yang digunakan oleh *malware*.

6. Kesimpulan dan Penulisan Laporan

   Tahap terakhir adalah membuat kesimpulan dari hasil analisis dan menyusun laporan penelitian. Kesimpulan ini akan mencakup temuan-temuan utama tentang bagaimana metode analisis runtime dan dinamis berhasil mengidentifikasi IOC dalam *malware*. Laporan ini juga akan merangkum seluruh proses penelitian, mulai dari desain eksperimen hingga hasil analisis, serta memberikan rekomendasi atau saran untuk penelitian lebih lanjut.

3.2. Pengumpulan Data

Tahap pengumpulan data dalam penelitian ini berfokus pada perolehan sampel *malware*. Pengumpulan sampel dilakukan dengan menggunakan teknik *purposive sampling*, di mana setiap sampel dipilih secara sengaja untuk mewakili jenis *malware* yang telah ditetapkan dalam batasan masalah. Data yang digunakan adalah data sekunder berupa file *malware* yang diperoleh dari repositori terbuka, yaitu Malshare dan Malware Bazaar. [19] Sampel yang digunakan dalam penelitian ini mencakup lima jenis *malware*, sebagaimana disajikan pada Tabel 1.

Tabel 1. Sampel Malware

No	Nama	Jenis Malware
1	Q77FYBluDA.exe	Trojan
2	Wannacry.exe	Ransomware
3	spyware_emotet_	Spyware
4	Xclient.exe	Worm
5	reagentcbotnet.exe	Botnet Agent

3.3. Perancangan Lingkungan Analisis

Perancangan lingkungan analisis bertujuan menciptakan sistem yang aman untuk menjalankan *malware* tanpa risiko terhadap sistem utama. Lingkungan ini dibangun menggunakan server *Microsoft Azure* dengan sistem operasi *Ubuntu 22.04* sebagai *host*. Server memiliki spesifikasi perangkat keras berupa prosesor *AMD EPYC 7763 64-Core, RAM 64 GB,* dan *hard disk 512 GB*, yang mendukung virtualisasi melalui teknologi *AMD-V*. *QEMU/KVM* digunakan sebagai *hypervisor* untuk menjalankan mesin virtual *Windows 10* sebagai *guest*, yang dikonfigurasi dengan alamat IP *192.168.50.50/24*. Mesin virtual *Ubuntu* sebagai *host* memiliki alamat IP *192.168.50.1/24*.

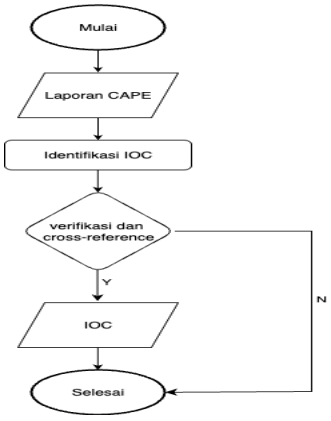
CAPE *Sandbox* diinstal pada *host* untuk melakukan analisis dinamis, sementara *Volatility Framework* digunakan untuk forensik memori. Jaringan dikonfigurasi dalam mode *isolated* dengan akses internet terbatas untuk memantau komunikasi *malware* dengan server eksternal. Mesin virtual *Windows 10* diatur dengan *16 GB RAM, 4 core CPU,* dan *hard disk 70 GB*. Fitur seperti *antivirus*, *firewall*, dan UAC dinonaktifkan untuk kelancaran analisis. *Python 3.11* dan *Pillow* diinstal pada *guest* untuk mendukung eksekusi *agent.py* dari CAPE. Lingkungan ini dilengkapi fitur *snapshot* dan *rollback* untuk menjaga keamanan dan konsistensi selama pengujian.

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

3.4. Analisis

Tahapan analisis dirancang untuk mengamati perilaku *malware* secara mendalam dan menghasilkan *Indicator of Compromise* (IOC) yang dapat digunakan untuk mendeteksi ancaman serupa di masa depan. Proses ini dilakukan oleh peneliti dalam lingkungan *Sandbox* yang telah disiapkan pada server *Microsoft Azure*. Analisis dimulai dengan menjalankan *QEMU/KVM* sebagai *hypervisor* untuk memastikan mesin virtual *Windows 10* siap digunakan. Selanjutnya, layanan CAPE *Sandbox* diaktifkan secara berurutan, mulai dari *cape-rooter* untuk mengatur *routing* jaringan, *cape-web* untuk menyediakan antarmuka pengguna, hingga *cape-processor* untuk memproses data analisis. Sampel *malware*, seperti yang terdaftar pada Tabel 1, dimasukkan melalui antarmuka web CAPE, yang kemudian secara otomatis menjalankan analisis dinamis saat *malware* dieksekusi dalam mesin virtual.

CAPE *Sandbox* merekam berbagai aktivitas *malware*, termasuk perubahan *registry*, komunikasi jaringan dengan server *command* and *control*, dan proses yang dijalankan. Untuk analisis lebih mendalam, *Volatility Framework* digunakan untuk memeriksa memory *dump* yang dihasilkan selama eksekusi, mengungkap data tersembunyi seperti *string* atau informasi sensitif yang mungkin dicuri oleh *malware*. Hasil analisis berupa laporan yang mencakup hash file (MD5, SHA-1, SHA-256), alamat IP, domain, dan perubahan *registry*. Laporan ini kemudian diverifikasi menggunakan *platform threat intelligence* seperti *VirusTotal* dan *SOCRadar* untuk memastikan keakuratan IOC. Proses ini bertujuan untuk memahami perilaku *malware* secara menyeluruh dan mencegah penyebarannya di lingkungan nyata. Alur analisis IOC dari *malware* disajikan pada Gambar 2.



Gambar 2. Alur Analisis IOC Malware

4. HASIL DAN PEMBAHASAN

Hasil analisis mencakup berbagai aktivitas yang terdeteksi saat sampel diuji, seperti komunikasi jaringan, modifikasi *registry*, pembuatan *dropped* files, dan analisis terhadap *memory dump*. Aktivitas-aktivitas ini menghasilkan berbagai IOC yang terdiri dari *hash* file, alamat IP, dan domain DNS yang digunakan untuk mendeteksi keberadaan *malware*. Temuan ini sejalan dengan penelitian sebelumnya oleh Panjaitan et al. [9], yang juga menekankan pentingnya analisis terhadap perubahan registry dan file dalam mendeteksi malware yang tersembunyi, serta mengidentifikasi jejak-jejak yang dapat mengindikasikan kompromi dalam sistem. Selanjutnya, IOC yang ditemukan diverifikasi menggunakan platform *intelligence* ancaman seperti *SOCRadar* dan *VirusTotal* untuk memastikan keakuratannya.

Hasil analisis menunjukkan bahwa perilaku masing-masing *malware* bervariasi, dengan setiap sampel menggunakan teknik yang berbeda untuk menyerang sistem dan menghindari deteksi. **Tabel 2** menyajikan ringkasan aktivitas yang terdeteksi pada setiap sampel *malware*, yang mencakup komunikasi jaringan, pembuatan file, dan modifikasi *registry*. Dari tabel ini, terlihat bahwa *Trojan* (Q77FYBluDA.exe) dan *Botnet Agent* (MouzerBotnet.exe) menghasilkan *dropped files*, sementara *Ransomware*, *Spyware*, dan *Worm* tidak menunjukkan adanya aktivitas tersebut. Semua *malware* terdeteksi melakukan komunikasi jaringan dan memodifikasi *registry*, namun hanya *Botnet Agent* yang terbatas pada komunikasi DNS. Hal ini menunjukkan bahwa beberapa jenis *malware* memanfaatkan komunikasi jaringan secara terbuka, sementara yang lain, seperti *Botnet*, lebih mengandalkan

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

teknik komunikasi yang lebih tersembunyi menggunakan DNS untuk menghindari deteksi. Temuan ini sejalan dengan penelitian oleh Dwi et al. [6], yang juga menunjukkan bahwa *botnet* menggunakan DNS untuk menyembunyikan komunikasi mereka dan menghindari deteksi, serta penelitian oleh Wahidin et al. [11], yang mengidentifikasi bahwa *malware* seperti *trojan* cenderung menghasilkan *dropped files* untuk memperkuat serangan.

Tabel 2. Ringkasan Aktivitas Analisis Malware

Nama	Komunikasi jaringan	Pembuatan File	Modifikasi registry
Q77FYBluDA.exe	Terdeteksi	Terdeteksi	Terdeteksi
Ransomware.exe	Terdeteksi	Tidak Terdeteksi	Terdeteksi
spyware_emotet_	Terdeteksi	Tidak Terdeteksi	Terdeteksi
Xclient.exe	Terdeteksi	Tidak Terdeteksi	Terdeteksi
MouzerBotnet.exe	Terdeteksi	Terdeteksi	Terdeteksi

Detail TTP (taktik, teknik dan prosedur) serangan dirangkum dalam Tabel 3, yang mencakup metode serangan dan teknik penghindaran yang digunakan oleh tiap sampel *malware* sebagai berikut. Hal ini sejalan dengan penelitian oleh Dwi et al. [6], yang juga mengidentifikasi bahwa malware modern menggunakan teknik penghindaran canggih, seperti *anti-debugging* dan *sandbox evasion*, untuk menghindari analisis dan deteksi.

Tabel 3. Metode Serangan Malware

Nama	Metode serangan
Q77FYBluDA.exe	Memeriksa memori, *hostname*, dan *keyboard layout* untuk menyesuaikan serangan; terhubung ke IP 176.46.157.32 dan DNS *boot.net.anydesk.com*; menghasilkan *dropped file* (*ASCII text*, MD5: D17FE0A3F47BE24A6453E9EF58C94641); menggunakan *anti-debugging* (*SetUnhandledExceptionFilter*, *guard pages*), *Sandbox evasion*, dan *disk check*
Ransomware.exe	*Geofencing* melalui pemeriksaan *keyboard layout* dan *locale*; komunikasi dengan domain sah seperti *discord.com* dan *api.ipify.org*; enkripsi file untuk penyanderaan data; akses folder *Public* untuk menyembunyikan file; menggunakan *anti-debugging* (*SetUnhandledExceptionFilter*), *Sandbox evasion*, dan *disk check*
spyware_emotet_	Menyembunyikan komunikasi jaringan melalui DNS-*Over*-HTTPS/TLS ke IP 77.238.212.227 dan 190.101.156.139; memeriksa *keyboard layout*, *locale*, dan pergerakan mouse untuk mendeteksi lingkungan virtual; membaca citra binari untuk menghindari *debugging*; menggunakan *anti-debugging*, *virtual network detection*, dan *Detect Cuckoo*.
Xclient.exe	Menggunakan memori RWX untuk menjalankan kode; komunikasi melalui DNS terenkripsi (*hotels-eq.gl.at.ply.gg*); memicu aturan YARA untuk binari mencurigakan; menggunakan *anti-debugging*, *Sandbox evasion*, dan *virtual network detection*.
MouzerBotnet.exe	Memeriksa informasi sistem dan jaringan; menghasilkan tiga *dropped files* manipulasi proses dan pembuatan jendela tersembunyi; menggunakan *anti-debugging*, *disk check*, dan *Detect Cuckoo*.

Verifikasi IOC dilakukan melalui *SOCRadar* dan *VirusTotal*. Tabel 4 merangkum hasil verifikasi dan *cross-checking* sebagai berikut.

Tabel 4. Ringkasan Aktivitas Analisis Malware

Nama	Parameter	Value	SOC Radar	Virus Total
Q77FYBluDA.exe	MD5	B6988F73D2285E86C7F6508DC292470A	Terdeteksi	Terdeteksi
	SHA256	3D2825C13FC111FC1678C4F93996C7694A8C51562B34C60B112172FEC83FB0EF	Terdeteksi	Terdeteksi
	IP	176.46.157.32	Terdeteksi	Terdeteksi

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

Nama	Parameter	Value	SOC Radar	Virus Total
		9.9.9.9	Tidak Terdeteksi	Tidak Terdeteksi
	DNS	57.128.101.75 - boot.net.anydesk.com	Tidak Terdeteksi	Tidak Terdeteksi
Ransom ware.exe	MD5	9BD7E4E4D5BCBCD7EAB8D9B262BB20D0	Terdeteksi	Terdeteksi
	SHA256	39BCAA56E661AB52B3617151C87C1F6C72040BFB3CCE9FEA924662A209064F33	Terdeteksi	Terdeteksi
	IP	9.9.9.9	Tidak Terdeteksi	Tidak Terdeteksi
	DNS	141.95.145.210 - boot.net.anydesk.com	Tidak Terdeteksi	Tidak Terdeteksi
		104.26.12.205 - api.ipify.org	Tidak Terdeteksi	Terdeteksi
		162.159.128.233 - discord.com	Tidak Terdeteksi	Tidak Terdeteksi
Spyware_emotet.exe	MD5	6E30D8BDA11412A2272A387B549BE17A	Terdeteksi	Terdeteksi
	SHA256	C3AD80D9E8443B1BEAE2DFE76227770B83FA852B9226F91A5628CB06624D8D9C	Terdeteksi	Terdeteksi
	IP	77.238.212.227	Terdeteksi	Terdeteksi
		190.101.156.139	Tidak Terdeteksi	Terdeteksi
		189.223.16.99	Tidak Terdeteksi	Terdeteksi
		217.13.106.14	Terdeteksi	Terdeteksi
		2.45.176.233	Tidak Terdeteksi	Terdeteksi
		193.251.77.110	Terdeteksi	Terdeteksi
		202.134.4.210	Terdeteksi	Terdeteksi
		209.236.123.42	Terdeteksi	Terdeteksi
		169.1.39.242	Tidak Terdeteksi	Terdeteksi
		177.144.130.105	Terdeteksi	Terdeteksi
		138.97.60.141	Terdeteksi	Terdeteksi
		45.46.37.97	Terdeteksi	Terdeteksi
		37.187.161.206	Terdeteksi	Terdeteksi
		94.23.62.116	Terdeteksi	Terdeteksi
		87.230.25.43	Tidak Terdeteksi	Terdeteksi
		70.39.251.94	Terdeteksi	Terdeteksi
		118.69.11.81	Tidak Terdeteksi	Terdeteksi
		190.202.229.74	Terdeteksi	Terdeteksi
		9.9.9.9		
	DNS	195.181.174.173 -	Terdeteksi	Terdeteksi

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

Nama	Parameter	Value	SOC Radar	Virus Total
		boot.net.anydesk.com		
Xclient.exe	MD5	AF3B1168AE1291F3C72557612F425885	Terdeteksi	Terdeteksi
	SHA256	7B1A4F886D5D6BBFE7A359DADB476E62F3B4BAF1C61DE164EF74CF6B97D342C3	Terdeteksi	Terdeteksi
	IP	9.9.9.9	Tidak Terdeteksi	Tidak Terdeteksi
	DNS	147.185.221.28 - hotels-eq.gl.at.ply.gg	Terdeteksi	Terdeteksi
		57.128.101.75 - boot.net.anydesk.com	Terdeteksi	Terdeteksi
MouzerBotnet.exe	MD5	518388EA7CFCE2A0758F610E3EB914BA	Terdeteksi	Terdeteksi
	SHA256	66E6425B04953CD4ADD9B0DD91BC63E37B293E3E65EBC978237747DF6BC99C3A	Terdeteksi	Terdeteksi
	IP	-		
	DNS	57.128.101.74 - boot.net.anydesk.com	Tidak Terdeteksi	Tidak Terdeteksi

Tabel 4 mengonfirmasi validitas IOC yang dihasilkan tahapan analisa sebagai ancaman di *SOCRadar* dan *VirusTotal*. Beberapa IP dan DNS seperti *9.9.9.9* dan *boot.net.anydesk.com* tidak terdeteksi sebagai ancaman.

5. KESIMPULAN DAN SARAN

Penggabungan metode *runtime* dan *dynamic analysis* dalam lingkungan *Sandbox* berhasil mengidentifikasi *Indicator of Compromise* (IOC) dari lima sampel *malware* yang diuji. Kedua metode ini saling melengkapi, di mana *runtime analysis* mendeteksi aktivitas real-time seperti komunikasi jaringan ke IP atau DNS mencurigakan, dan *dynamic analysis* mengungkap jejak digital seperti *dropped files*, modifikasi *registry*, dan teknik *anti-debugging*. *Trojan* dan *Botnet Agent* menghasilkan *dropped files*, sedangkan *Ransomware*, *Spyware*, dan *Worm* berfokus pada aktivitas jaringan dan *registry*. Temuan baru, seperti penggunaan DNS-*Over*-HTTPS/TLS oleh *Spyware*, menunjukkan kompleksitas analisis jaringan. Keunggulan metode ini terletak pada deteksi akurat terhadap perubahan file, *registry*, dan komunikasi jaringan, memberikan wawasan komprehensif tentang perilaku *malware*. Sebagai mitigasi, disarankan menerapkan pemantauan memori untuk *Trojan*, *backup* rutin untuk *Ransomware*, penyaringan DNS untuk *Spyware*, aturan YARA untuk *Worm*, dan pemindaian file tersembunyi untuk *Botnet*.

Berdasarkan hasil penelitian, berikut beberapa saran yang dapat dipertimbangkan:

1. Menambah sampel *malware* dengan jenis dan famili berbeda untuk memperluas cakupan analisis.
2. Mengintegrasikan dengan *static analysis* atau *artificial inteligence* untuk meningkatkan kemampuan deteksi IOC.

6. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih yang sebesar-besarnya kepada Universitas Bumigora atas dukungan dan fasilitas yang diberikan selama proses penelitian ini. Dukungan dari pihak universitas, baik dari segi sumber daya maupun fasilitas, sangat mendukung kelancaran dan keberhasilan penelitian ini.

DAFTAR PUSTAKA:

[1] K. Ibrahim, F. Dewanta, dan N. D. W. Cahyani, “Analisis Perilaku Malware

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 1, Januari 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

Malware Menggunakan Metode Analisis Dinamis," *eProceedings of Engineering*, vol. 10, no. 5, 2023.

[2] "Malware Statistics & Trends Report | AV-TEST." Diakses: 10 Juli 2025. [Daring]. Tersedia pada: https://www.av-test.org/en/statistics/malware/

[3] BSSN, "LANSKAP KEAMANAN SIBER INDONESIA," 2024.

[4] Y. B. Setiadji, D. F. Priambodo, M. Hasbi, dan F. I. Sabila, "Identifikasi Malware Berdasarkan Artefak Registry Windows 10 Menggunakan Regshot dan Cuckoo," *JEPIN (Jurnal Edukasi dan Penelitian Informatika)*, vol. 8, no. 3, hlm. 482–491, 2022.

[5] "Desktop Operating System Market Share Worldwide | Statcounter Global Stats." Diakses: 10 Juli 2025. [Daring]. Tersedia pada: https://gs.statcounter.com/os-market-share/desktop/worldwide

[6] Y. Dwi *dkk.*, "Analisis Malware Menggunakan Metode Analisis Statis dan Dinamis untuk Pembuatan IOC Berdasarkan STIX Versi 2.1," *Jurnal Info Kripto*, vol. 15, hlm. 106–111, Nov 2021.

[7] K. Khalda dan D. K. Wibowo, "Analisis Perilaku Malware Menggunakan Pendekatan Analisis Statis dan Dinamis," *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, vol. 4, no. 1, hlm. 1–8, 2025.

[8] V. A. Manoppo, A. S. M. Lumenta, dan S. D. S. Karouw, "Analisa Malware Menggunakan Metode Dynamic Analysis Pada Jaringan Universitas Sam Ratulangi," *Jurnal Teknik Elektro dan Komputer*, vol. 9, no. 3, hlm. 181–188, 2020.

[9] F. Panjaitan, H. Yudiastuti, M. ulfa, D. Universitas Bina Darma, dan J. Jenderal Ahmad Yani No, "ANALISIS MALWARE DENGAN METODE SURFACE DAN RUNTIME ANALYSIS," *Jurnal Ilmiah MATRIK*, vol. 23, no. 1, 2021.

[10] D. A. Daniswara, A. Budiyono, dan A. Almaarif, "Analisis Deteksi Malicious Activity Menggunakan Metode Analisis Malware Dinamis Berbasis Anomali," *eProceedings of Engineering*, vol. 6, no. 2, 2019.

[11] G. W. Wahidin, S. Syaifuddin, dan Z. Sari, "Analisis Ransomware Wannacry Menggunakan Aplikasi Cuckoo Sandbox," *Jurnal Repositor*, vol. 4, no. 1, 2022.

[12] A. Siddiq, H. Yudiastuti, dan F. Pajaitan, "Analisis Perilaku Malware Dengan Metode Surface Analysis Dan Runtime Analysis," *Journal of Software Engineering Ampera*, vol. 1, no. 3, hlm. 2775–2488, 2020, [Daring]. Tersedia pada: https://journal-computing.org/index.php/journal-sea/index

[13] A. R. Damanik, H. B. Seta, dan T. Theresiawati, "Analisis Trojan Dan Spyware Menggunakan Metode Hybrid Analysis," *Jurnal Ilmiah MATRIK*, vol. 25, no. 1, hlm. 89–97, 2023.

[14] Fahriza, "Analisis Ransomware Secara Statis Dan Dinamis Untuk Pemetaan Evolusi Ransomware," 2022.

[15] Naufal Andrianto Nurfauzi, "Deteksi Serangan Malware Pada Cloud Server Menggunakan Metode Anomaly Based," Universitas Islam Negeri Maulana Malik Ibrahim, MALANG, 2022.

[16] A. Villalón-Huerta, I. Ripoll-Ripoll, dan H. Marco-Gisbert, "Key Requirements for the Detection and Sharing of Behavioral Indicators of Compromise," *Electronics (Basel)*, vol. 11, no. 3, 2022, doi: 10.3390/electronics11030416.

[17] Y. LUOFAN, D.-S. Choi, dan others, "Evasive PDF Malware Detection Based on Deep Learning and CAPE Sandbox," 한국정보과학회 학술발표논문집, hlm. 1004–1006, 2024.

[18] P. D. Sugiyono, *Metode Penelitian Kuantitatif, Kualitatif, dan R&D*, 3rd ed. Bandung: CV Alfabet, 2021.

[19] H. Novansyah dan T. Sutabri, "ANALISIS MALWARE DENGAN METODE DINAMIK MENGGUNAKAN FRAMEWORK CUCKOO SANDBOX," *Blantika : Multidisciplinary Jornal*, vol. 2, no. 1, 2023, [Daring]. Tersedia pada: https://blantika.publikasiku.id/