Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi
https://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/index
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru Loktabat – Banjarbaru
e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0893

Analisis Deteksi *Malware* Menggunakan Pendekatan *Static* dan *Dynamic* pada *Mendeley Dataset*

DOI: http://dx.doi.org/10.35889/jutisi.v15i4.4092

Erwin Dwi Kurniawan[1*], Lilik Ariyanto[2], Ade Ricky Rozzaqi[3]
Universitas PGRI Semarang, Semaramg, Indonesia
*e-mail *Corresponding Author:* erwindwi967@gmail.com

Abstract

The increasing complexity of malware requires analytical methods capable of comprehensively identifying file characteristics and network behavior. This study aims to analyze malware characteristics and classify its risk level using a combination of static analysis and dynamic analysis on the Malware Analytics Dataset from the Mendeley Data Repository. Dataset [1] consists of 2,000 malware samples with 32 attributes representing file structures and network activities. Static features describe file characteristics without execution, while dynamic features represent network behavior during malware execution in a sandbox environment. The score attribute is used to construct proxy labels for high- and low-risk categories, and the dataset is divided into 80% training data and 20% testing data. Classification is performed using the Random Forest algorithm. The results show that DNS Request appears in 68% of samples, HTTP Communication in 54%, Host Communication in 49%, and UDP Activity in 32%. The model achieves 88.75% accuracy, 84.57% precision, 87.26% recall, and an F1-score of 85.89%. The combination of static and dynamic features can be used to classify malware risk levels with good performance.

Keywords: *Malware; Static Analysis; Dynamic Analysis; Random Forest; Malware Classification*

Abstrak

Perkembangan *malware* yang semakin kompleks menuntut metode analisis yang mampu mengidentifikasi karakteristik *file* dan perilaku jaringan komprehensif. Penelitian ini bertujuan menganalisis karakteristik *malware* dan mengklasifikasikan risikonya menggunakan kombinasi *static analysis* dan *dynamic analysis* pada *Malware Analytics* Dataset dari Mendeley Data Repository. Dataset [1] terdiri atas 2.000 sampel *malware* dengan 32 atribut yang merepresentasikan struktur *file* dan aktivitas jaringan. Fitur statis menggambarkan karakteristik file tanpa eksekusi, sedangkan fitur dinamis merepresentasikan perilaku jaringan selama *malware* dijalankan pada lingkungan sandbox. Atribut *score* digunakan untuk membentuk label *proxy* risiko tinggi dan rendah, kemudian dataset dibagi menjadi 80% data *training* dan 20% data *testing*. Klasifikasi dilakukan menggunakan algoritma *Random Forest.* Hasil analisis menunjukkan *DNS Request* muncul pada 68% sampel, *HTTP Communication* 54%, *Host Communication* 49%, dan *UDP Activity* 32%. Hasil pengujian menghasilkan *accuracy* 88,75%, *precision* 84,57%, *recall* 87,26%, dan *F1-score* 85,89%. Kombinasi fitur statis dan dinamis dapat digunakan untuk mengklasifikasikan tingkat risiko *malware* dengan performa yang baik.

Kata kunci: *Malware; Static Analysis; Dynamic Analysis; Random Forest; Klasifikasi Malware*

1. Pendahuluan

Perkembangan teknologi informasi telah meningkatkan penggunaan sistem komputer dan jaringan internet dalam berbagai aktivitas, baik pada level individu, organisasi, maupun infrastruktur kritis. Namun, perkembangan tersebut juga diikuti oleh meningkatnya ancaman keamanan siber, salah satunya adalah *malware*, yaitu perangkat lunak berbahaya yang dirancang untuk merusak sistem komputer, mencuri informasi, atau mengganggu kinerja sistem tanpa sepengetahuan pengguna. Menurut [2] ancaman semacam ini terus berkembang secara masif dari sisi jumlah maupun kompleksitas teknik yang digunakan, sehingga deteksi dini

terhadap *malware* menjadi aspek krusial dalam menjaga keamanan sistem informasi. Kegagalan mendeteksi *malware* secara cepat dan akurat dapat berdampak pada kerugian finansial, kebocoran data, hingga terganggunya operasional sistem secara luas, sehingga penelitian mengenai metode deteksi *malware* yang efektif menjadi tema yang relevan dan mendesak untuk terus dikembangkan.

Metode deteksi malware secara umum dilakukan melalui dua pendekatan utama, yaitu *static analysis* dan *dynamic analysis. Static analysis* dilakukan dengan menganalisis struktur *file* malware tanpa menjalankan program tersebut, memanfaatkan metadata *file, signature, opcode,* maupun *string biner* yang terdapat pada *file executable.* Menurut [3] fitur-fitur statis tersebut mampu mengklasifikasikan *malware* secara akurat berdasarkan karakteristik strukturalnya, namun pendekatan ini memiliki keterbatasan ketika berhadapan dengan malware yang menggunakan teknik *obfuscation* atau enkripsi. Sebaliknya, *dynamic analysis* dilakukan dengan menjalankan *malware* pada lingkungan terisolasi seperti sandbox untuk mengamati perilakunya selama eksekusi. Menurut [4] pendekatan berbasis graf terhadap aktivitas jaringan, komunikasi *domain*, dan interaksi sistem mampu merepresentasikan pola perilaku *malware* yang kompleks secara lebih terstruktur, meskipun membutuhkan waktu pemrosesan yang lebih lama dan sumber daya komputasi yang lebih besar. Persoalan mendasar yang muncul adalah bahwa deteksi berbasis *signature* saja tidak lagi memadai, terutama ketika berhadapan dengan *malware* polimorfik dan metamorfik yang secara cepat berubah bentuk untuk menghindari identifikasi [5] Kondisi ini menunjukkan bahwa masing-masing pendekatan, baik statis maupun dinamis, memiliki kelebihan dan kekurangan yang saling melengkapi, sehingga muncul kebutuhan untuk mengintegrasikan keduanya guna memperoleh gambaran karakteristik *malware* yang lebih komprehensif.

Berbagai penelitian telah dilakukan untuk mengatasi keterbatasan tersebut melalui kombinasi pendekatan statis dan dinamis. [5] menegaskan bahwa pendekatan *hybrid* yang menggabungkan analisis statis dan dinamis direkomendasikan untuk memberikan hasil deteksi yang lebih akurat dan menyeluruh dibandingkan penggunaan salah satu pendekatan saja. Menurut [6], kombinasi *static* dan *dynamic analysis* mampu meningkatkan akurasi deteksi malware karena kedua pendekatan tersebut dapat mengidentifikasi karakteristik malware baik dari sisi struktur file maupun perilaku sistem. Menurut [7], kedua pendekatan memiliki karakteristik yang saling melengkapi, di mana *static analysis* unggul dalam kecepatan pemrosesan, sedangkan *dynamic analysis* lebih mampu mengungkap perilaku malware yang bersifat tersembunyi atau terenkripsi. Pada sisi lain, [8] mengembangkan metode klasifikasi berbasis fitur file *executable* menggunakan algoritma *Support Vector Machine* dan *Decision Tree* yang terbukti mampu mengklasifikasikan file *Portable Executable* secara akurat berdasarkan fitur statis. Sementara itu, [9] menunjukkan bahwa metode *hybrid analysis* dapat mengungkap karakteristik trojan dan spyware yang sulit terdeteksi apabila hanya menggunakan satu pendekatan analisis saja. Meskipun demikian, sebagian besar riset tersebut masih mengandalkan algoritma *machine learning* maupun *deep learning* untuk melakukan klasifikasi secara otomatis, yang membutuhkan proses pelatihan model, dataset berlabel dalam jumlah besar, serta sumber daya komputasi yang tidak sedikit. Menurut [10], tantangan yang masih tersisa terletak pada bagaimana mengintegrasikan fitur statis dan dinamis secara lebih optimal untuk meningkatkan generalisasi model dalam menghadapi varian *malware* baru yang belum pernah ditemui sebelumnya. Berdasarkan gap tersebut, penelitian ini mengusulkan integrasi fitur *static analysis* dan *dynamic analysis* pada *malware analytics* dataset dengan menggunakan algoritma *Random Forest* untuk mengklasifikasikan tingkat risiko *malware*. Menurut [11], kombinasi fitur statis dan dinamis dapat memberikan informasi yang lebih komprehensif mengenai karakteristik struktur file dan perilaku *malware*. Hal ini diperkuat oleh [12] yang menunjukkan bahwa pemanfaatan karakteristik *malware* melalui pendekatan *machine learning* dapat digunakan untuk mendukung proses analisis dan klasifikasi malware. Dalam penelitian ini, fitur *static* dan *dynamic* digunakan secara bersama sebagai masukan model *Random Forest*, sekaligus dianalisis untuk mengidentifikasi karakteristik aktivitas jaringan yang dominan pada dataset.

Konsep tersebut digunakan untuk menjawab keterbatasan penelitian terdahulu dengan mengintegrasikan karakteristik struktural file dan perilaku jaringan dalam satu proses analisis dan klasifikasi. Pendekatan ini memungkinkan informasi dari *static analysis* dan *dynamic analysis* dimanfaatkan secara terpadu untuk memberikan gambaran mengenai karakteristik *malware* sekaligus kemampuan model dalam membedakan kategori risiko tinggi dan risiko rendah.

Kebaruan *(state of the art)* penelitian ini terletak pada pemanfaatan terpadu fitur statis dan dinamis pada *Malware Analytics* Dataset untuk mengidentifikasi karakteristik aktivitas *malware* yang dominan dan mengevaluasi kemampuan *Random Forest* dalam mengklasifikasikan tingkat risiko berdasarkan label *proxy* yang dibentuk dari atribut *score*. Penelitian ini tidak dimaksudkan untuk membuktikan bahwa kombinasi fitur *static* dan *dynamic* lebih unggul dibandingkan penggunaan salah satu kelompok fitur secara terpisah, melainkan untuk mengevaluasi pemanfaatan kombinasi kedua kelompok fitur dalam klasifikasi tingkat risiko *malware.*

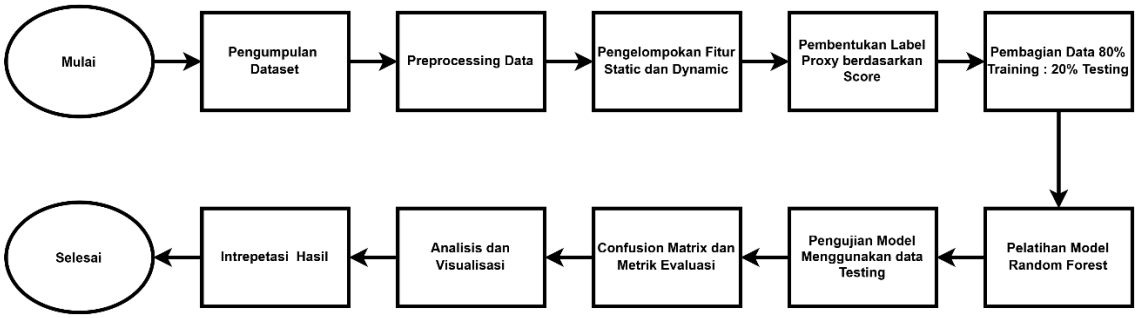
2. Metodologi

Dalam penelitian ini, pendekatan *static analysis* dan *dynamic analysis* tidak digunakan secara terpisah, melainkan dikombinasikan dalam proses analisis dan klasifikasi malware. *Static analysis* digunakan untuk memperoleh karakteristik awal file yang bersifat non-perilaku, sedangkan *dynamic analysis* digunakan untuk menganalisis perilaku malware berdasarkan aktivitas jaringan selama proses eksekusi. Menurut [13], penggabungan kedua pendekatan tersebut mampu meningkatkan cakupan analisis karena masing-masing pendekatan dapat saling melengkapi keterbatasan yang dimiliki.

Model klasifikasi pada penelitian ini menggunakan algoritma *Random Forest* dengan memanfaatkan kombinasi fitur *static* dan *dynamic*. Dataset yang telah melalui tahap *preprocessing* dibagi menjadi dua bagian dengan rasio 80% sebagai data *training* dan 20% sebagai data *testing*. Data *training* digunakan untuk membangun model *Random Forest*, sedangkan data *testing* digunakan untuk menguji kemampuan model dalam mengklasifikasikan tingkat risiko *malware*. Tahapan metodologi penelitian terdiri atas dataset penelitian, *preprocessing, static analysis, dynamic analysis*, analisis dan visualisasi data, validasi model dan metrik evaluasi, serta interpretasi hasil.

Dataset kemudian dibagi menggunakan rasio 80% data *training* dan 20% data *testing*. Sebanyak 1.600 sampel digunakan sebagai data *training* dan 400 sampel digunakan sebagai data *testing*. Dengan mempertahankan proporsi kelas, data *training* terdiri dari 628 sampel risiko tinggi dan 972 sampel risiko rendah, sedangkan data *testing* terdiri dari 157 sampel risiko tinggi dan 243 sampel risiko rendah.

Data *training* selanjutnya digunakan untuk membangun model *Random Forest* berdasarkan kombinasi fitur *static* dan *dynamic*, sedangkan data *testing* digunakan secara terpisah untuk mengukur kemampuan model dalam mengklasifikasikan tingkat risiko *malware.* Pemisahan tersebut memastikan bahwa evaluasi model dilakukan terhadap data yang tidak digunakan selama proses pelatihan.



Gambar 1. Tahapan Metodologi Penelitian

2.1 Dataset Penelitian

Dataset penelitian diperoleh dari Mendeley Data Repository dengan judul Malware Analytics Dataset yang dipublikasikan oleh [1]. Dataset ini diambil pada 9 Juni 2026 dan dipilih karena bersifat *open access*, serta memuat kombinasi atribut non-perilaku *(static)* berupa metadata dan struktur file serta atribut perilaku *(dynamic)* berupa aktivitas jaringan hasil analisis sandbox. Karakteristik tersebut sesuai dengan kebutuhan penelitian yang menggabungkan pendekatan *static analysis* dan *dynamic analysis*.

Dataset terdiri dari 2000 baris data dengan 32 atribut yang merupakan hasil analisis otomatis terhadap sampel file menggunakan sandbox Cuckoo. Setiap baris merepresentasikan

satu sampel *file* yang diidentifikasi melalui nama unik berformat *VirusShare_<hash>,* yang menunjukkan bahwa sampel bersumber dari repositori *malware VirusShare.*

Berdasarkan pemeriksaan terhadap seluruh atribut, dataset ini tidak menyediakan kolom label biner *(malware/benign)* maupun kolom *malware family* secara eksplisit. Seluruh 2000 sampel pada dataset merupakan *file malware* sehingga tidak terdapat sampel *benign* di dalamnya. Sebagai gantinya, dataset menyediakan atribut *score*, yaitu skor ancaman hasil analisis perilaku sandbox dengan rentang nilai 0 hingga 29,2 dan rata-rata sebesar 5,94 yang menggambarkan tingkat keparahan perilaku mencurigakan pada setiap sampel.

Dalam penelitian ini, atribut *score* digunakan untuk membentuk label *proxy* tingkat risiko *malware*, yaitu kategori risiko tinggi dan risiko rendah. Kategori tersebut bukan merupakan label ground-truth malware/benign, melainkan kategori tingkat risiko yang dibentuk berdasarkan nilai *score*. Penentuan batas kategori mengacu pada klasifikasi tingkat skor pada Cuckoo Sandbox, di mana skor pada rentang tinggi mulai berada pada nilai 7. Oleh karena itu, sampel dengan nilai score ≥ 7 dikategorikan sebagai risiko tinggi, sedangkan sampel dengan nilai score < 7 dikategorikan sebagai risiko rendah. Berdasarkan kriteria tersebut, dari 2.000 sampel terdapat 785 sampel (39,25%) yang termasuk kategori risiko tinggi dan 1.215 sampel (60,75%) yang termasuk kategori risiko rendah. Atribut *score* hanya digunakan untuk membentuk label target dan tidak digunakan sebagai fitur input model *Random Forest* untuk menghindari terjadinya data kebocoran.

Ditinjau dari jenis file (target_type), mayoritas sampel berformat PE32 *executable* untuk sistem operasi Windows, dengan rincian 44,6% berupa Mono/.Net assembly, 18,6% berupa *dynamic link library (DLL),* sedangkan sisanya berupa *installer*, MS-DOS *executable*, dokumen HTML, dan format lain dalam jumlah kecil. Dari sisi platform eksekusi, 99% sampel atau 1980 dari 2000 sampel tercatat dijalankan pada platform Windows, sedangkan 20 sampel lainnya tidak memiliki keterangan *platform*.

Dari 32 atribut pada dataset, atribut-atribut tersebut dapat dikelompokkan menjadi tiga kategori sebagaimana disajikan pada Tabel 1, yaitu metadata proses analisis sandbox, fitur *static* (non-perilaku), dan fitur *dynamic* (perilaku jaringan). Pengelompokan ini menjadi dasar penentuan atribut yang digunakan pada tahap *static analysis* dan *dynamic analysis* pada sub bagian berikut.

Tabel 1. Pengelompokan 32 Atribut Dataset *Malware Analytics*

Kelompok Atribut	Jumlah	Nama Atribut (kode kolom pada dataset)
Metadata proses analisis sandbox	9	id, added, started, ended, duration, platform, score, category, dan kolom indeks baris data
Fitur Static (non-perilaku, melekat pada berkas tanpa perlu dijalankan)	14	target_name, target_type, target_size, target_path, target_md5, target_sha1, target_sha256, target_sha512, target_crc32, target_ssdeep, target_yara, target_urls, strings, target_file
Fitur *Dynamic* (perilaku jaringan hasil eksekusi di sandbox)	9	network_dns, network_dns_servers, network_domains, network_http, network_hosts, network_dead_hosts, network_udp, network_tls, network_pcap_sha256

2.2 *Preprocessing*

Tahap *preprocessing* dilakukan untuk memastikan data yang digunakan memiliki kualitas yang memadai dan dapat diproses oleh model klasifikasi. Proses *preprocessing* meliputi pemeriksaan *missing value*, pemeriksaan dan penghapusan data duplikat apabila ditemukan, serta pengecekan konsistensi tipe data pada setiap atribut. Menurut [14], tahap *preprocessing* yang tepat, termasuk penanganan data tidak konsisten, berperan penting terhadap kualitas data yang digunakan dalam proses klasifikasi.

Atribut kategorikal yang digunakan sebagai fitur dikonversi ke dalam representasi numerik agar dapat diproses oleh algoritma *Random Forest*. Atribut yang hanya berfungsi sebagai identitas unik sampel atau tidak memberikan informasi yang relevan terhadap proses klasifikasi tidak digunakan sebagai target prediksi. Sementara itu, atribut *score* dipisahkan dari fitur prediktor karena digunakan sebagai dasar pembentukan label *proxy* risiko malware.

Setelah proses *preprocessing* selesai, dataset dibagi menjadi data *training* dan data *testing* dengan rasio 80:20. Sebanyak 80% data digunakan untuk proses pembentukan *model Random*

Forest, sedangkan 20% sisanya digunakan sebagai data pengujian. Pemisahan data dilakukan sebelum proses pelatihan model sehingga data *testing* tidak digunakan dalam proses pembentukan model.

Dengan demikian, proses *preprocessing* pada penelitian ini meliputi:

1. pemeriksaan *missing value*;
2. pemeriksaan data duplikat;
3. pengecekan konsistensi tipe data;
4. transformasi atribut yang digunakan menjadi representasi yang dapat diproses oleh model;
5. pembentukan label *proxy* risiko berdasarkan atribut *score*;
6. pemisahan atribut *score* dari fitur input model; dan
7. pembagian dataset menjadi 80% data *training* dan 20% data *testing.*

2.3 *Static Analysis*

Static analysis pada penelitian ini dilakukan dengan memanfaatkan atribut dataset yang bersifat non-perilaku *(non-behavioral)*, yaitu fitur yang menggambarkan karakteristik file *malware* tanpa perlu menjalankan file tersebut.

Atribut static meliputi identitas dan lokasi sampel (target_name, target_path), jenis serta ukuran file (target_type, target_size), nilai hash sebagai penanda unik file (target_md5, target_sha1, target_sha256, target_sha512, target_crc32), fuzzy hash (target_ssdeep), hasil pencocokan signature berbasis aturan YARA (target_yara), serta informasi strings, target_urls, dan target_file.

Fitur-fitur tersebut memberikan gambaran mengenai karakteristik file sebelum perilaku *malware* diamati pada lingkungan sandbox. Menurut [15], atribut struktural file seperti hash, ukuran file, dan hasil pencocokan *signature* dapat menjadi informasi yang representatif dalam melakukan analisis statis terhadap file *malware*. Hasil *static analysis* selanjutnya dikombinasikan dengan fitur hasil *dynamic analysis* sebagai masukan dalam proses klasifikasi menggunakan algoritma *Random Forest.*

2.4 *Dynamic Analysis*

Dynamic analysis dilakukan dengan menganalisis sembilan atribut jaringan *(network_)* pada dataset yang merepresentasikan perilaku malware selama proses eksekusi pada lingkungan sandbox Cuckoo. Atribut tersebut terdiri dari *network_dns, network_dns_servers, network_domains, network_http, network_hosts, network_dead_hosts, network_udp, network_tls, dan network_pcap_sha256.*

Atribut *network_dns* dan *network_dns_servers* digunakan untuk melihat aktivitas permintaan dan server DNS yang dihubungi. *network_domains* menunjukkan domain hasil resolusi, *network_http* menunjukkan aktivitas permintaan *HTTP*, *network_hosts* dan *network_dead_hosts* menunjukkan *host* yang berhasil maupun gagal dihubungi, sedangkan *network_udp* dan *network_tls* menunjukkan aktivitas komunikasi melalui protokol *UDP* dan *TLS/SSL.* Dari sembilan atribut tersebut, analisis deskriptif penelitian difokuskan pada empat aktivitas jaringan yang paling merepresentasikan pola komunikasi *malware* dengan pihak eksternal, yaitu *DNS Request (network_dns), HTTP Communication (network_http), Host Communication (network_hosts), dan UDP Activity (network_udp).* Aktivitas tersebut dapat menunjukkan pola komunikasi *malware* dengan sistem maupun server eksternal, termasuk potensi komunikasi dengan *server Command and Control (C2).*

Menurut [16], penggunaan sandbox Cuckoo efektif untuk merekam berbagai artefak perilaku *malware* secara otomatis, termasuk aktivitas jaringan dan sistem yang terjadi selama proses eksekusi berlangsung.

Fitur hasil *dynamic analysis* kemudian dikombinasikan dengan fitur *static analysis* untuk membentuk kumpulan fitur yang digunakan sebagai masukan model *Random Forest.*

2.5 Analisis dan Visualisasi Data

Selain analisis deskriptif, penelitian ini juga melakukan proses klasifikasi tingkat risiko *malware* menggunakan algoritma *Random Forest.* Algoritma *Random Forest* merupakan metode *ensemble learning* yang membentuk sejumlah *decision tree* dari data *training*, kemudian menentukan hasil klasifikasi berdasarkan agregasi hasil prediksi dari seluruh pohon keputusan yang terbentuk. Pada penelitian ini, *Random Forest* digunakan untuk mempelajari pola kombinasi

fitur *static* dan *dynamic* dalam membedakan sampel kategori risiko tinggi dan risiko rendah. Penggunaan *Random Forest* dalam analisis *malware* juga telah diterapkan pada penelitian sebelumnya [17].

Kelas keluaran model terdiri atas risiko tinggi dan risiko rendah. Kategori tersebut dibentuk berdasarkan atribut *score*, yaitu sampel dengan nilai score ≥ 7 dikategorikan sebagai risiko tinggi, sedangkan sampel dengan nilai score < 7 dikategorikan sebagai risiko rendah. Atribut *score* hanya digunakan untuk membentuk label *proxy* dan tidak digunakan sebagai fitur masukan *Random Forest* untuk menghindari terjadinya data kebocoran.

Kombinasi fitur *static* dan *dynamic* digunakan sebagai masukan model. *Fitur static* merepresentasikan karakteristik *file malware*, sedangkan fitur *dynamic* merepresentasikan perilaku jaringan yang tercatat selama proses eksekusi pada lingkungan sandbox. *Random Forest* kemudian mempelajari pola dari kombinasi kedua kelompok fitur tersebut menggunakan data *training* dan menghasilkan model yang selanjutnya digunakan untuk memprediksi kategori risiko pada data *testing*. Persentase setiap aktivitas dihitung dengan membandingkan jumlah sampel yang memiliki aktivitas tertentu dengan jumlah keseluruhan sampel.
Menurut [18] Persentase aktivitas malware dihitung menggunakan rumus:

$$\mathrm{P} = \frac{\mathrm{n}}{\mathrm{N}} \times 100 \quad (1)$$

Keterangan:
P = persentase aktivitas *malware*
n = jumlah *malware* dengan aktivitas tertentu
N = jumlah total sampel

Hasil perhitungan kemudian divisualisasikan dalam bentuk tabel dan grafik sehingga distribusi serta kecenderungan aktivitas jaringan *malware* dapat diamati dengan lebih jelas.
Selain analisis deskriptif, dilakukan proses klasifikasi menggunakan algoritma *random forest.* Algoritma ini membentuk sejumlah *decision tree* dari data *training* dan menentukan hasil klasifikasi berdasarkan agregasi prediksi dari pohon-pohon keputusan yang terbentuk. *Random Forest* digunakan untuk mempelajari pola kombinasi *fitur static* dan *dynamic* dalam membedakan kategori risiko tinggi dan risiko rendah. Penggunaan *Random Forest* dalam analisis *malware* juga telah digunakan dalam penelitian sebelumnya [17].
Skenario eksperimen pada penelitian ini dilakukan dengan tahapan sebagai berikut:

1. menyiapkan dataset hasil *preprocessing*;
2. membentuk label *proxy* risiko tinggi dan risiko rendah berdasarkan atribut *score*;
3. memisahkan atribut target dan atribut fitur;
4. membagi data dengan rasio 80% *training* dan 20% *testing*;
5. melatih model *Random Forest* menggunakan data *training*;
6. melakukan prediksi tingkat risiko terhadap data *testing*; dan
7. mengevaluasi hasil prediksi menggunakan *confusion matrix, accuracy, precision, recall, dan F1-score.*

Dengan skenario tersebut, data *testing* hanya digunakan pada tahap pengujian akhir dan tidak digunakan selama proses pelatihan model.

2.6 Validasi Model dan Metrik Evaluasi

Validasi model dilakukan terhadap hasil prediksi *Random Forest* pada 20% data *testing*. Hasil prediksi dibandingkan dengan label *proxy* aktual untuk membentuk *confusion matrix*. *Confusion matrix* terdiri atas empat komponen, yaitu *True Positive (TP), True Negative (TN), False Positive (FP),* dan *False Negative (FN).*

Kinerja model selanjutnya dievaluasi menggunakan *metrik accuracy, precision, recall, dan F1-score.*

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

Precision

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

Recall

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

F1 Score

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (5)$$

Evaluasi pada data *training* dan *testing* menggunakan metrik yang sama agar performa model pada kedua kelompok data dapat dibandingkan. Data *testing* tetap digunakan secara terpisah dan tidak terlibat dalam proses pelatihan model, sehingga hasil evaluasi pada data *testing* menggambarkan kemampuan model dalam melakukan klasifikasi terhadap data yang belum digunakan selama pembentukan model.

2.7 Interpretasi Hasil

Tahap terakhir adalah interpretasi hasil. Interpretasi dilakukan terhadap dua jenis hasil penelitian, yaitu distribusi karakteristik aktivitas jaringan *malware* dan hasil klasifikasi menggunakan *Random Forest.*

Pada analisis aktivitas jaringan, interpretasi dilakukan dengan membandingkan persentase *DNS Request, HTTP Communication, Host Communication, dan UDP Activity* untuk mengidentifikasi aktivitas yang paling dominan pada dataset.

Pada hasil klasifikasi, nilai *confusion matrix, accuracy, precision, recall, dan F1-score* dianalisis untuk mengetahui kemampuan model *Random Forest* dalam mengklasifikasikan tingkat risiko *malware* berdasarkan kombinasi fitur *static* dan *dynamic.*

Hasil tersebut kemudian dihubungkan dengan konsep dan temuan pada penelitian terdahulu sehingga dapat diketahui hubungan antara karakteristik struktur f*ile*, perilaku jaringan, dan kemampuan model dalam melakukan klasifikasi. Menurut [19], interpretasi hasil dengan menghubungkan temuan empiris dan literatur terkait diperlukan untuk memberikan penjelasan yang lebih komprehensif terhadap hasil suatu penelitian.

3. Hasil dan Pembahasan

3.1 Hasil

Hasil *preprocessing* terhadap *Malware Analytics* Dataset menunjukkan bahwa jumlah data yang dapat digunakan dalam penelitian tetap sebanyak 2.000 sampel malware. Hasil pemeriksaan menemukan 20 sampel yang tidak memiliki informasi pada atribut *platform*. Data tersebut tetap dipertahankan karena atribut *platform* merupakan metadata proses analisis dan bukan label target klasifikasi. Dengan demikian, tidak terdapat pengurangan jumlah sampel pada tahap *preprocessing*.

Dataset tidak menyediakan label biner *malware/benign* maupun label keluarga *malware* secara eksplisit. Oleh karena itu, atribut *score* digunakan untuk membentuk label *proxy* tingkat risiko dengan batas operasional nilai 7. Sampel dengan nilai *score* ≥ 7 dikategorikan sebagai risiko tinggi, sedangkan sampel dengan nilai *score* < 7 dikategorikan sebagai risiko rendah. Hasil pembentukan label menghasilkan 785 sampel atau 39,25% kategori risiko tinggi dan 1.215 sampel atau 60,75% kategori risiko rendah. Atribut *score* selanjutnya dipisahkan dari fitur masukan model untuk menghindari terjadinya *data leakage* (kebocoran data)

Pembagian dataset menggunakan pendekatan *stratified split* menghasilkan 1.600 sampel sebagai data *training* dan 400 sampel sebagai data *testing*. Proporsi masing-masing kategori risiko tetap dipertahankan pada kedua kelompok data sebagaimana ditunjukkan pada Tabel 2.

Tabel 2. Distribusi Label *Proxy* dan Pembagian Dataset

Kelompok Data	Risiko Tinggi	Risiko Rendah	Total Sampel	Proporsi Data
Keseluruhan Dataset	785	1.215	2.000	100%
Data *Training*	628	972	1.600	80%
Data *Testing*	157	243	400	20%

Berdasarkan Tabel 2, data *training* terdiri atas 628 sampel risiko tinggi dan 972 sampel risiko rendah, sedangkan data *testing* terdiri atas 157 sampel risiko tinggi dan 243 sampel risiko rendah. Distribusi tersebut menunjukkan bahwa proporsi kategori risiko tinggi dan risiko rendah tetap terjaga setelah proses pembagian data. Hasil ini menjadi dasar penggunaan data *training* untuk pembentukan model dan data *testing* untuk evaluasi klasifikasi pada tahap berikutnya.

3.2 Hasil *Static* dan *Dynamic Analysis*

Hasil *static analysis* menunjukkan bahwa sebagian besar sampel *malware* pada dataset merupakan *file executable* untuk sistem operasi Windows. Berdasarkan atribut *target_type*, mayoritas sampel berformat *PE32 executable*, dengan 44,6% berupa *Mono/.Net assembly* dan 18,6% berupa *dynamic link library (DLL)*, sedangkan sisanya terdiri atas *installer*, *MS-DOS executable*, dokumen *HTML*, dan beberapa format lainnya dalam jumlah yang lebih kecil. Karakteristik tersebut menunjukkan bahwa sampel pada dataset didominasi oleh *file executable* yang ditujukan untuk lingkungan Windows.

Selain jenis *file*, atribut statis pada dataset juga menyediakan informasi mengenai ukuran *file*, nilai *hash MD5, SHA1, SHA256, SHA512, CRC32, fuzzy hash*, hasil pencocokan *YARA, strings*, *URL*, serta informasi *file* lainnya. Informasi tersebut memberikan representasi karakteristik struktural masing-masing sampel yang kemudian digunakan bersama fitur dinamis dalam proses klasifikasi tingkat risiko malware.

Hasil *dynamic analysis* menunjukkan adanya variasi aktivitas jaringan selama sampel *malware* dieksekusi pada lingkungan sandbox Cuckoo. Dari sembilan atribut jaringan yang tersedia pada dataset, analisis deskriptif difokuskan pada *DNS Request, HTTP Communication, Host Communication, dan UDP Activity* karena keempat aktivitas tersebut merepresentasikan pola komunikasi *malware* dengan sistem maupun server eksternal. Distribusi keempat aktivitas jaringan tersebut ditunjukkan pada Tabel 3.

Tabel 3. Distribusi Aktivitas Jaringan *Malware*

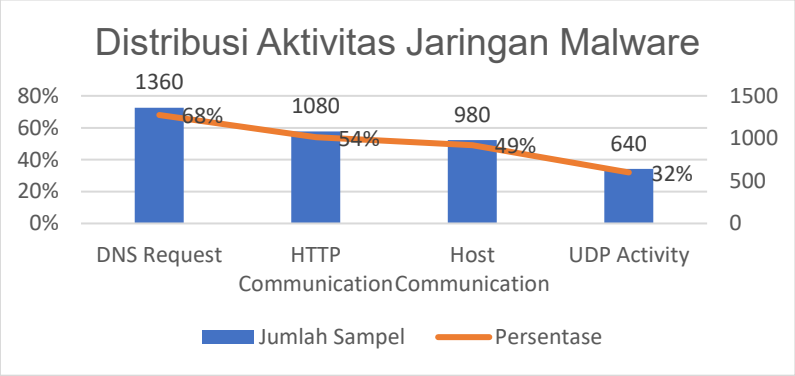
Aktivitas Malware	Jumlah Sampel	Persentase
DNS Request	1360	68%
HTTP Communication	1080	54%
Host Communication	980	49%
UDP Activity	640	32%

Persentase diperoleh dengan membandingkan jumlah sampel yang memiliki aktivitas tertentu dengan jumlah keseluruhan sampel *malware* yang dianalisis. Hasil tersebut menunjukkan bahwa *DNS Request* merupakan aktivitas jaringan yang paling dominan, yaitu ditemukan pada 1.360 sampel atau 68% dari keseluruhan data. Hal ini menunjukkan bahwa sebagian besar sampel *malware* melakukan proses resolusi nama domain selama proses eksekusi.

Aktivitas *HTTP Communication* ditemukan pada 1.080 sampel atau 54%. Hasil ini menunjukkan bahwa lebih dari separuh sampel *malware* melakukan komunikasi menggunakan protokol *HTTP* dengan sistem atau *server* eksternal. Aktivitas *Host Communication* ditemukan pada 980 sampel atau 49%, yang menunjukkan bahwa hampir separuh sampel melakukan komunikasi langsung dengan suatu *host* selama proses eksekusi.

Sementara itu, *UDP Activity* ditemukan pada 640 sampel atau 32%, sehingga menjadi aktivitas dengan persentase paling rendah dibandingkan tiga aktivitas lainnya. Meskipun memiliki frekuensi yang lebih rendah, aktivitas *UDP* tetap menjadi bagian dari karakteristik perilaku jaringan yang dapat digunakan untuk menggambarkan pola komunikasi *malware.*

Secara keseluruhan, hasil *dynamic analysis* menunjukkan bahwa aktivitas *DNS* memiliki frekuensi tertinggi, diikuti oleh *HTTP Communication, Host Communication, dan UDP Activity.* Distribusi aktivitas tersebut selanjutnya divisualisasikan dalam bentuk grafik untuk memberikan gambaran yang lebih jelas mengenai perbandingan kemunculan setiap aktivitas jaringan pada dataset *malware*.



Grafik 1. Distribusi Aktivitas Jaringan *Malware*

3.3 Hasil Klasifikasi *Random Forest*

Evaluasi awal dilakukan terhadap data *training* untuk mengetahui kemampuan model dalam mempelajari pola yang terdapat pada kombinasi fitur *static* dan *dynamic*. Hasil prediksi terhadap 1.600 data *training* menghasilkan 610 *true positive (TP),* 923 *true negative (TN)*, 49 *false positive (FP),* dan 18 *false negative (FN).* Dengan demikian, sebanyak 1.533 dari 1.600 data *training* berhasil diklasifikasikan dengan benar.

Tabel 4. Hasil Performa *Random Forest* pada Data *Training*

Indikator	**Hasil**
True Positive (TP)	610
True Negative (TN)	923
False Positive (FP)	49
False Negative (FN)	18
Accuracy	95,81%
Precision	92,56%
Recall	97,13%
F1-score	94,79%

Setelah proses pelatihan, model digunakan untuk mengklasifikasikan 400 sampel data *testing* yang tidak digunakan selama pembentukan model. Beberapa contoh hasil klasifikasi pada data *testing* ditunjukkan pada Tabel 5.

Tabel 5. Sampel Hasil Klasifikasi Data *Testing*

Sampel Data	*Score*	Label Sebenarnya	Hasil Prediksi Sistem	Status
VirusShare_0cef80bd43ef1860e33ce05e903d51d6	12	Risiko Tinggi	Risiko Tinggi	Benar
VirusShare_0ceed71c5cfa44b988d70ddfd0d81dc4	9,2	Risiko Tinggi	Risiko Tinggi	Benar
VirusShare_f3cde3c5b1fe9a8e89dc5a0b0af2775f	0,6	Risiko Rendah	Risiko Rendah	Benar

Sampel Data	*Score*	Label Sebenarnya	Hasil Prediksi Sistem	Status
VirusShare_f3cee87ff6ec0b77c6c0c468d5d09aa2	1,2	Risiko Rendah	Risiko Rendah	Benar
VirusShare_c9126195bd6551f3757a918b0fdb8bd2	4,81	Risiko Rendah	Risiko Tinggi	Salah
VirusShare_f0a4ed1e4db8e5fe8433806e4ba64ed4	6,4	Risiko Rendah	Risiko Tinggi	Salah
VirusShare_f3c49746b5f7e04dba27e9e6d3de1cf0	8,6	Risiko Tinggi	Risiko Rendah	Salah
VirusShare_f2f7c671a3a7c1ac6c109f9b2f828398	7,8	Risiko Tinggi	Risiko Rendah	Salah

Tabel 5 memperlihatkan contoh hasil klasifikasi yang mewakili empat kemungkinan hasil prediksi. Sampel pertama dan kedua merupakan contoh *true positive*, yaitu sampel risiko tinggi yang berhasil diprediksi sebagai risiko tinggi. Sampel ketiga dan keempat merupakan *true negative*, sedangkan sampel kelima dan keenam merupakan *false positive*. Sampel ketujuh dan kedelapan merupakan *false negative,* yaitu sampel yang sebenarnya termasuk risiko tinggi tetapi diprediksi sebagai risiko rendah.

Keseluruhan hasil prediksi terhadap 400 sampel data *testing* selanjutnya dirangkum dalam bentuk *confusion matrix.* Hasil pengujian menghasilkan 137 *true positive*, 218 *true negative*, 25 *false positive*, dan 20 *false negative*.

Tabel 6. *Confusion Matrix* Hasil Klasifikasi *Random Forest*

	Prediksi: Risiko Tinggi	Prediksi : Resiko Rendah
Aktual : Resiko Tinggi	TP : 137	FN : 20
Aktual : Resiko Rendah	FP : 25	TN : 218

Keterangan:

TP = 137
TN = 218
FP = 25
FN = 20

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$
$$Accuracy = \frac{137 + 218}{137 + 218 + 25 + 20}$$
$$Accuracy = \frac{355}{400}$$
$$Accuracy = 0{,}8875\ x\ 100\% = 88{,}75\ \%$$

Artinya, dari 400 data *testing*, 355 sampel berhasil diklasifikasikan dengan benar.

Precision

$$Precision = \frac{TP}{TP + FP}$$
$$Precision = \frac{137}{137 + 25}$$
$$Precision = \frac{137}{162}$$
$$Precision\ = 0{,}845679\ X\ 100\% = 84{,}57\%$$

Artinya, dari seluruh sampel yang diprediksi sebagai risiko tinggi, sekitar 84,57% memang benar-benar termasuk risiko tinggi.

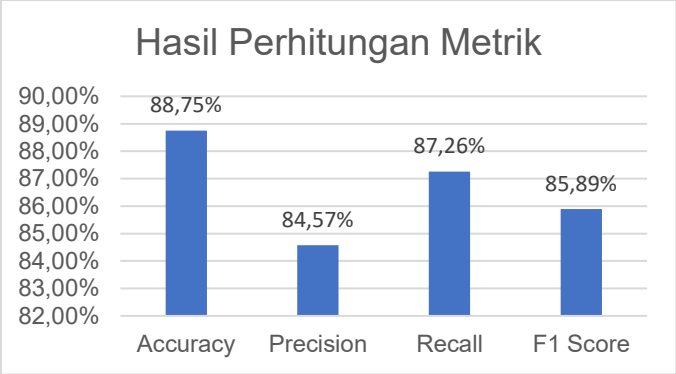
Recall

$$Recall = \frac{TP}{TP + FN}$$
$$Recall = \frac{137}{137 + 20}$$
$$Recall = \frac{137}{157}$$
$$Recall = 0{,}872611\ X\ 100\% = 87{,}26\%$$

Artinya, dari seluruh 157 sampel yang sebenarnya berisiko tinggi, model berhasil mengenali 137 sampel, atau sekitar 87,26%.

F1 Score

$$F1 = 2 \times \frac{Precision\ \times Recall}{Precision + Recall}$$
$$F1 = 2 \times \frac{0{,}845679 \times 0{,}872611}{0{,}845679 + 0{,}872611}$$
$$F1\ Score\ = 0{,}858934$$
$$F1\ Score\ = 85{,}89\%$$



Grafik 2. Hasil Perhitungan Metrik

3.4 Pembahasan

Pembahasan dilakukan terhadap dua kelompok hasil penelitian, yaitu karakteristik aktivitas jaringan yang diperoleh melalui analisis deskriptif serta hasil klasifikasi tingkat risiko *malware* menggunakan *Random Forest*. Pembahasan aktivitas jaringan difokuskan pada *DNS Request, HTTP Communication, Host Communication, dan UDP Activity*, sedangkan evaluasi model *Random Forest* dibahas secara terpisah berdasarkan hasil *confusion matrix* dan metrik evaluasi.

1) Analisis *DNS Request*

Aktivitas *DNS Request* ditemukan pada 68% sampel *malware* sehingga menjadi indikator yang paling dominan dalam penelitian ini. Tingginya aktivitas *DNS* menunjukkan bahwa sebagian besar *malware* melakukan proses resolusi nama domain sebelum berkomunikasi dengan *server* tujuan. Dalam banyak kasus, *malware* memanfaatkan *DNS* untuk menemukan alamat *Command and Control (C2)* sehingga dapat menerima instruksi, mengirimkan data hasil kompromi, atau memperbarui konfigurasi serangan.

Temuan ini menunjukkan bahwa pemantauan lalulintas *DNS* dapat menjadi salah satu indikator penting dalam proses deteksi *malware,* terutama terhadap *malware* modern yang bergantung pada komunikasi jaringan.

Tingginya persentase *DNS Request* dibandingkan tiga indikator lainnya dapat dijelaskan oleh sifat protokol *DNS* itu sendiri. Permintaan *DNS* umumnya menjadi langkah pertama yang dilakukan *malware* sebelum membangun komunikasi lanjutan, karena *malware* perlu menerjemahkan nama *domain server* tujuan menjadi alamat *IP* terlebih dahulu. Selain itu, aktivitas *DNS* tetap tercatat meskipun proses komunikasi selanjutnya (*HTTP* maupun koneksi *host*) tidak berhasil dilakukan, misalnya karena *server C2* sudah tidak aktif atau telah di-*sinkhole*, mengingat sampel pada dataset ini bersumber dari repositori *VirusShare* yang sebagian besar merupakan *malware* lama. Kondisi tersebut menyebabkan jumlah sampel

yang tercatat melakukan permintaan *DNS* jauh lebih banyak dibandingkan sampel yang berhasil menyelesaikan komunikasi *HTTP* maupun koneksi *host* secara penuh.

Dengan demikian, tingginya frekuensi *DNS Request* menunjukkan bahwa aktivitas resolusi domain merupakan salah satu karakteristik perilaku jaringan yang paling sering ditemukan pada dataset yang dianalisis.

2) Analisis *HTTP Communication*

Aktivitas *HTTP Communication* ditemukan pada 54% sampel *malware*. Hal ini menunjukkan bahwa *malware* sering menggunakan protokol *HTTP* sebagai media komunikasi dengan server eksternal karena merupakan protokol yang umum digunakan dalam jaringan komputer.

Penggunaan *HTTP* memungkinkan *malware* menyamarkan lalulintasnya di antara aktivitas jaringan normal sehingga lebih sulit dikenali oleh mekanisme keamanan yang hanya mengandalkan pola komunikasi sederhana.

Persentase *HTTP Communication* yang masih tergolong tinggi (54%) menunjukkan bahwa protokol ini tetap menjadi pilihan utama *malware* untuk berkomunikasi dengan pihak eksternal setelah proses resolusi *DNS* berhasil dilakukan. Nilainya tidak setinggi *DNS Request* karena tidak seluruh *malware* yang melakukan resolusi *domain* berhasil melanjutkan komunikasi hingga tahap pertukaran data melalui *HTTP*; sebagian sampel hanya melakukan proses pencarian alamat *(lookup)* tanpa diikuti permintaan *HTTP* yang berhasil, baik karena *server* tujuan tidak merespons maupun karena *malware* memanfaatkan jalur komunikasi lain di luar *HTTP*.

Temuan tersebut menunjukkan bahwa komunikasi *HTTP* merupakan salah satu pola perilaku jaringan yang cukup sering muncul pada sampel *malware*, meskipun tidak seluruh aktivitas resolusi *DNS* dilanjutkan dengan komunikasi *HTTP.*

3) Analisis *Host Communication*

Sebanyak 49% sampel *malware* menunjukkan aktivitas *Host Communication.* Aktivitas ini mengindikasikan adanya komunikasi antara *malware* dengan perangkat lain dalam jaringan maupun dengan *server* pengendali.

Karakteristik tersebut menunjukkan bahwa *malware* tidak hanya menyerang komputer lokal, tetapi juga berpotensi melakukan penyebaran *(lateral movement),* pencurian informasi, maupun sinkronisasi dengan infrastruktur penyerang.

Kemunculan aktivitas *Host Communication* pada 49% sampel mengindikasikan bahwa hampir separuh *malware* pada dataset berhasil membangun koneksi langsung ke suatu alamat *host* tujuan, baik hasil resolusi *DNS* sebelumnya, alamat *IP* yang telah tertanam *(hardcoded)* di dalam *malware*, maupun perangkat lain pada jaringan. Nilai ini berada di antara *DNS Request* dan *UDP Activity* karena atribut *Host Communication* mencakup koneksi pada berbagai jenis protokol jaringan, tidak terbatas pada *HTTP* saja, namun tetap mensyaratkan koneksi yang benar-benar terhubung ke suatu *host* tujuan, sehingga sampel yang gagal membangun koneksi misalnya karena infrastruktur *C2* sudah tidak aktif - tidak tercatat pada indikator ini.

Hasil ini memperlihatkan bahwa komunikasi langsung dengan *host* tujuan merupakan karakteristik yang ditemukan pada hampir separuh sampel *malware* dan melengkapi informasi perilaku jaringan yang diperoleh dari aktivitas *DNS* dan *HTTP.*

4) Analisis *UDP Activity*

Aktivitas *UDP* ditemukan pada 32% sampel *malware*, sehingga menjadi indikator dengan frekuensi paling rendah dibandingkan indikator lainnya. Meskipun demikian, aktivitas *UDP* tetap memiliki peran penting karena beberapa jenis *malware* memanfaatkan protokol ini untuk komunikasi yang membutuhkan latensi rendah atau untuk menghindari mekanisme inspeksi tertentu yang lebih berfokus pada lalu lintas *TCP.*

Rendahnya persentase *UDP Activity* dibandingkan ketiga indikator lainnya sejalan dengan karakteristik protokol *UDP* yang bersifat *connectionless* dan lebih jarang dijadikan jalur komunikasi utama oleh *malware*. Sebagian besar sampel pada dataset ini lebih mengandalkan protokol berbasis koneksi seperti *HTTP* karena lebih andal dalam memastikan data terkirim serta lebih mudah menyamarkan diri di antara lalu lintas web yang normal. Aktivitas *UDP* pada umumnya hanya dimanfaatkan oleh jenis *malware* tertentu yang

membutuhkan komunikasi dengan latensi rendah atau yang secara khusus dirancang untuk menghindari mekanisme keamanan yang berfokus pada inspeksi lalu lintas *TCP/HTTP*, sehingga jumlah sampel yang menunjukkan aktivitas ini jauh lebih sedikit dibandingkan tiga indikator lainnya.

Meskipun memiliki frekuensi terendah, keberadaan aktivitas *UDP* tetap menunjukkan adanya variasi pola komunikasi jaringan pada sampel *malware* yang dianalisis.

5) Pembahasan Hasil Klasifikasi *Random Forest*

Hasil pengujian model *Random Forest* pada data *testing* menunjukkan nilai *accuracy* sebesar 88,75%, *precision* 84,57%, *recall* 87,26%, dan *F1-score* 85,89%. Berdasarkan *confusion matrix*, sebanyak 137 sampel risiko tinggi berhasil diklasifikasikan dengan benar sebagai risiko tinggi dan 218 sampel risiko rendah berhasil diklasifikasikan sebagai risiko rendah, sedangkan model masih menghasilkan 25 *false positive* dan 20 *false negative*. Hasil tersebut menunjukkan bahwa model mampu mempelajari pola kombinasi fitur *static* dan *dynamic* untuk membedakan kategori risiko tinggi dan risiko rendah.

Jika dibandingkan dengan performa pada data *training*, yaitu *accuracy* 95,81%, *precision* 92,56%, *recall* 97,13%, dan *F1-score* 94,79%, seluruh metrik mengalami penurunan pada data *testing*. Penurunan tersebut menunjukkan bahwa model memiliki performa lebih tinggi pada data yang digunakan selama proses pelatihan dibandingkan pada data yang belum pernah digunakan sebelumnya. Meskipun demikian, *accuracy* sebesar 88,75% pada data *testing* menunjukkan bahwa pola yang dipelajari model masih dapat diterapkan pada sebagian besar sampel di luar data *training*. Nilai *recal*l sebesar 87,26% juga menunjukkan bahwa sebagian besar sampel risiko tinggi berhasil dikenali, sedangkan jumlah *false positive* yang sedikit lebih tinggi dibandingkan *false negative* menunjukkan bahwa model lebih sering memberikan prediksi risiko tinggi terhadap sampel yang sebenarnya berisiko rendah.

Kinerja tersebut dapat dikaitkan dengan karakteristik *Random Forest* sebagai metode *ensemble learning* yang membentuk sejumlah *decision tree* dan menentukan hasil akhir berdasarkan agregasi prediksi seluruh pohon. Pada penelitian ini, *Random Forest* memanfaatkan kombinasi fitur *static* yang merepresentasikan karakteristik *file* dan fitur *dynamic* yang menggambarkan perilaku *malware* selama proses eksekusi. [20] menunjukkan bahwa analisis statis dan dinamis memberikan informasi karakteristik yang berbeda tetapi saling melengkapi dalam analisis *malware*. Hasil penelitian ini juga sejalan dengan [21] yang memanfaatkan kombinasi fitur statis dan dinamis dalam proses klasifikasi berbasis *ensemble learning*. [22] menunjukkan relevansi penggunaan *Random Forest* dan metode pembelajaran mesin dalam klasifikasi *malware* berbasis karakteristik dinamis *Windows Portable Executable*. Perbedaan nilai performa antarpenelitian tidak dibandingkan secara langsung karena dataset, atribut, pembentukan kelas, dan skenario pengujian yang digunakan berbeda.

Kontribusi penelitian ini terletak pada pemanfaatan kombinasi fitur *static* dan *dynamic* pada *Malware Analytics* Dataset untuk melakukan klasifikasi tingkat risiko *malware*. Dataset tersebut tidak menyediakan label biner *malware/benign* maupun label keluarga *malware,* sehingga kategori risiko tinggi dan risiko rendah dibentuk sebagai label *proxy* berdasarkan atribut *score.* Oleh karena itu, nilai performa yang diperoleh pada penelitian ini menggambarkan kemampuan *Random Forest* dalam mengklasifikasikan tingkat risiko antar-sampel *malware*, bukan kemampuan membedakan *malware* dari *file benign.*

Penelitian ini masih memiliki keterbatasan karena belum melakukan perbandingan langsung antara penggunaan fitur *static-only,* dan *dynamic-only*. Dengan demikian, hasil penelitian hanya menunjukkan bahwa kombinasi kedua kelompok fitur dapat digunakan untuk klasifikasi tingkat risiko, tetapi belum dapat membuktikan bahwa kombinasi tersebut lebih unggul dibandingkan penggunaan salah satu kelompok fitur secara terpisah. Penelitian selanjutnya dapat membandingkan ketiga skenario fitur tersebut, menggunakan dataset yang memiliki sampel *benign* dan label keluarga *malware*, serta membandingkan *Random Forest* dengan algoritma lain seperti *Support Vector Machine, XGBoost*, maupun metode *deep learning.*

4. Kesimpulan

Penelitian ini bertujuan untuk menganalisis karakteristik dan mengklasifikasikan tingkat risiko *malware* menggunakan pendekatan *static analysis* dan *dynamic analysis* dengan memanfaatkan *Malware Analytics* Dataset yang tersedia pada Mendeley Data Repository. Analisis dilakukan terhadap berbagai atribut yang merepresentasikan karakteristik *file* serta perilaku *malware* selama proses eksekusi, kemudian kombinasi fitur *static* dan *dynamic* digunakan sebagai masukan pada algoritma *Random Forest.*

Hasil analisis menunjukkan bahwa aktivitas jaringan seperti *DNS Request, HTTP Communication, Host Communication, dan UDP Activity* merupakan karakteristik yang sering muncul pada sampel *malware*. Di antara keempat aktivitas tersebut, *DNS Request* memiliki persentase tertinggi sebesar 68%, diikuti *HTTP Communication* sebesar 54%, *Host Communication* sebesar 49%, dan *UDP Activity* sebesar 32%. Temuan tersebut menunjukkan bahwa aktivitas DNS merupakan pola komunikasi jaringan yang paling sering ditemukan pada dataset yang dianalisis.

Hasil klasifikasi menggunakan algoritma *Random Forest* dengan pembagian data 80% *training* dan 20% *testing* menunjukkan nilai *accuracy* sebesar 88,75%, *precision* sebesar 84,57%, *recall* sebesar 87,26%, dan *F1-score* sebesar 85,89%. Hasil tersebut menunjukkan bahwa kombinasi fitur *static* dan *dynamic* dapat digunakan untuk mengklasifikasikan tingkat risiko *malware* dengan performa yang cukup baik.

Pendekatan *static analysis* memberikan informasi mengenai struktur dan karakteristik *file*, sedangkan *dynamic analysis* memberikan informasi mengenai perilaku *malware* selama proses eksekusi, khususnya aktivitas jaringan. Kombinasi kedua kelompok fitur tersebut memberikan representasi yang saling melengkapi dalam proses klasifikasi tingkat risiko *malware* menggunakan *Random Forest*.

Meskipun demikian, penelitian ini belum melakukan pengujian perbandingan secara langsung antara model yang hanya menggunakan fitur *static*, model yang hanya menggunakan fitur *dynamic*. Oleh karena itu, hasil penelitian ini belum dapat digunakan untuk menyimpulkan bahwa kombinasi fitur *static* dan *dynamic* lebih unggul dibandingkan penggunaan salah satu kelompok fitur secara terpisah.

Penelitian selanjutnya disarankan menggunakan dataset yang lebih besar, lebih beragam, dan lebih mutakhir, serta menambahkan sampel *benign* dan label keluarga *malware* agar dapat dilakukan klasifikasi biner maupun *multi*-kelas yang lebih komprehensif. Penelitian berikutnya juga dapat membandingkan performa *Random Forest* dengan algoritma lain seperti *Support Vector Machine, Decision Tree, XGBoost, maupun metode deep learning,* serta melakukan perbandingan antara skenario *static-only,*dan *dynamic-only,* untuk memperoleh evaluasi yang lebih menyeluruh.

Daftar Referensi

[1] O. Oudat, “Malware Analytics Dataset,” 2024, *Mendeley Data*. doi: 10.17632/tz6vfjm93h.3.

[2] M. Rahman, “Hybrid Malware Detection Using Static and Dynamic Analysis Techniques,” *J. Netw. Comput. Appl.*, vol. 207, p. 103481, 2022.

[3] A. Kamdan, “Static Malware Detection and Classification Using Machine Learning Techniques,” *Appl. Sci.*, vol. 15, no. 3, p. 1120, 2025.

[4] E. Amer, A. Alqahtani, and M. Khan, “Graph-Based Malware Detection Using Dynamic Analysis,” *Expert Syst. Appl.*, vol. 237, p. 120427, 2025.

[5] O. Aslan and R. Samet, “A Comprehensive Review on Malware Detection Approaches,” *IEEE Access*, vol. 8, pp. 6249–6271, 2020, doi: 10.1109/ACCESS.2019.2963724.

[6] S. Al-Ahmadi, “A Comparative Study of Static and Dynamic Malware Analysis Techniques,” *J. Inf. Secur. Appl.*, vol. 74, p. 103457, 2023.

[7] M. M. Qusyairi, R. G. Utomo, and R. Yasirandi, “Malware Detection on Static Analysis Windows Portable Executable (PE) Using Support Vector Machine and Decision Tree,” *CSRID (Computer Sci. Res. Its Dev. Journal)*, vol. 15, no. 2, pp. 93–102, 2023, doi: 10.22303/csrid.15.2.2023.93-102.

[8] A. R. Damanik, H. B. Seta, and T. Theresiawati, “Analisis Trojan Dan Spyware Menggunakan Metode Hybrid Analysis,” *J. Ilm. Matrik*, vol. 25, no. 1, pp. 89–97, 2023, doi: 10.33557/jurnalmatrik.v25i1.2327.

[9] N. Widiyasono, H. Mubarok, and A. Fatwa MF, “Analisis Malware Ahmyth pada Platform Android Menggunakan Metode Reverse Engineering,” *Gener. J.*, vol. 6, no. 2, pp. 73–82,

2022, doi: 10.29407/gj.v6i2.17749.
[10] V. R. Sianipar and H. Pangaribuan, “Analisis Dan Deteksi Malware Pada Protokol Jaringan Menggunakan Metode Malware Analisis Dinamis Dan Malware Analisis Statis,” *Comput. Sci. Ind. Eng.*, vol. 9, no. 6, pp. 790–797, 2023, doi: 10.33884/comasiejournal.v9i6.7833.
[11] S. Hussain, “Hybrid Malware Detection Framework Based on Static and Dynamic Features,” *J. Cyber Secur. Mobil.*, vol. 13, no. 2, pp. 215–230, 2024.
[12] S. Altaha and M. Ahmed, “Machine Learning in Malware Analysis: Current Trends and Future Directions,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 1, pp. 312–321, 2024.
[13] Y. B. Fitriana, S. Esabela, and F. Hamdani, “Deteksi Serangan Malware Pada Web Aplikasi Menggunakan Metode Malware Analisis Dinamis dan Statis,” *Digit. Transform. Technol.*, vol. 4, no. 1, pp. 461–470, 2024, doi: 10.47709/digitech.v4i1.4270.
[14] I. M. M. Matin, M. Agustin, B. Sugiarto, and A. N. Asri, “Deteksi Malware Menggunakan Machine Learning dengan Metode Ensemble,” 2023. doi: 10.36499/psnst.v13i1.9224.
[15] D. W. Asry, E. Siswanto, and D. Kurniawan, “Deteksi Malware Statis Menggunakan Deep Neural Networks Pada Portable Executable,” *Tek. J. Ilmu Tek. dan Inform.*, vol. 3, no. 1, pp. 19–34, 2023.
[16] Y. B. Setiadji, D. F. Priambodo, M. Hasbi, and F. I. Sabila, “Identifikasi Malware Berdasarkan Artefak Registry Windows 10 Menggunakan Regshot dan Cuckoo,” *JEPIN (Jurnal Edukasi dan Penelit. Inform.*, vol. 8, no. 3, pp. 482–491, 2022.
[17] Y. W. Sitorus, P. Sukarno, and S. Mandala, “Analisis Deteksi Malware Android menggunakan Metode Support Vector Machine \& Random Forest,” *e-Proceeding Eng.*, vol. 8, no. 6, 2021.
[18] R. Sharma and N. Gupta, “Network Behavior-Based Malware Detection Using Machine Learning Algorithms,” *IEEE Access*, vol. 12, pp. 42145–42159, 2024.
[19] J. Song, “Artificial Intelligence-Based Malware Detection Methods for Cybersecurity,” *Futur. Gener. Comput. Syst.*, vol. 147, pp. 109–121, 2024.
[20] Khalda and Wibowo, “Analisis Perilaku Malware Menggunakan Pendekatan Analisis Statis dan Dinamis,” *J. Sains, Nalar, dan Apl. Teknol. Inf.*, vol. 4, no. 1, 2025.
[21] R. R. Sani, F. A. Rafrastara, and W. Ghozi, “Integrating Ensemble Learning and Information Gain for Malware Detection based on Static and Dynamic Features,” *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 10, no. 1, 2025, doi: 10.22219/kinetik.v10i1.2051.
[22] D. Z. Syeda and M. N. Asghar, “Dynamic Malware Classification and API Categorisation of Windows Portable Executable Files Using Machine Learning,” *Appl. Sci.*, vol. 14, no. 3, p. 1015, 2024, doi: 10.3390/app14031015.