Deteksi Malware Secara Real-Time Menggunakan Algoritma Pencocokan Pola dalam Sistem Keamanan Siber

Andrew Isra Saputra DB – 13523110
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: andrew.israsaputra@gmail.com , 13523110@std.stei.itb.ac.id

Abstrak— Keamanan siber merupakan aspek yang sangat penting dalam melindungi sistem komputer dan data dari ancaman yang dapat merusak atau mencuri informasi. Salah satu ancaman utama dalam dunia digital adalah malware, yang dapat menyebabkan kerusakan signifikan pada perangkat dan sistem. Untuk melawan ancaman ini, deteksi malware secara real-time menjadi sangat krusial. Salah satu metode yang banyak digunakan untuk mendeteksi malware adalah pencocokan pola (pattern matching), yang memungkinkan identifikasi malware berdasarkan pola atau signature yang telah diketahui. Penelitian ini mengusulkan penggunaan algoritma Aho-Corasick untuk deteksi malware secara real-time. Aho-Corasick merupakan algoritma pencocokan multi-pola yang sangat efisien, yang memungkinkan pencocokan banyak pola dalam satu waktu, sehingga sangat cocok untuk mendeteksi malware dengan berbagai signature yang berbeda. Algoritma ini bekerja dengan membangun struktur data tree untuk menyimpan pola-pola malware dan fungsi fail untuk mengoptimalkan proses pencocokan. Dalam penelitian ini, penulis mengimplementasikan Aho-Corasick untuk memindai file. Hasil eksperimen menunjukkan bahwa algoritma ini dapat mendeteksi malware dengan akurasi yang tinggi dan waktu eksekusi yang sangat cepat, menjadikannya solusi yang sangat cocok untuk aplikasi deteksi malware real-time yang memerlukan efisiensi dan kecepatan tinggi. Dengan demikian, Aho-Corasick dapat menjadi pilihan yang efektif dalam pengembangan sistem deteksi malware di dunia siber yang terus berkembang.

Kata Kunci—deteksi malware, keamanan siber, aho-corasick, pattern matching

I. PENDAHULUAN

Keamanan siber menjadi salah satu tantangan utama dalam dunia teknologi informasi. Dalam beberapa tahun terakhir, ancaman dari malware, seperti virus, trojan, worm, dan ransomware, semakin meningkat dan menjadi salah satu isu yang sangat diperhatikan oleh perusahaan, individu, serta organisasi pemerintahan di seluruh dunia. Malware, atau perangkat lunak berbahaya, dapat merusak data, mencuri informasi pribadi, atau bahkan merusak infrastruktur kritis yang mendukung operasi suatu sistem. Oleh karena itu, penting untuk memiliki sistem deteksi yang efektif yang mampu mendeteksi ancaman ini secara dini sebelum mereka dapat menyebabkan kerusakan yang lebih besar.

Dalam konteks ini, deteksi malware secara real-time menjadi sangat penting. Real-time malware detection bertujuan untuk memantau aktivitas sistem secara terus-menerus dan segera mengidentifikasi keberadaan malware yang mencoba menginfeksi sistem atau jaringan. Sistem deteksi malware ini harus memiliki kemampuan untuk memproses data dengan cepat dan akurat, tanpa menurunkan kinerja keseluruhan sistem. Salah satu metode yang digunakan untuk mendeteksi malware adalah dengan menggunakan teknik pencocokan pola (pattern matching), yang dapat memeriksa file atau data jaringan untuk menemukan pola-pola yang sesuai dengan signature malware yang diketahui.

Pencocokan pola adalah salah satu teknik yang banyak digunakan dalam pengolahan teks dan pencarian informasi, yang memungkinkan sistem untuk menemukan urutan karakter yang cocok dengan pola tertentu dalam data. Pada deteksi malware, pola tersebut biasanya berupa signature atau tanda tangan dari malware yang sudah teridentifikasi sebelumnya. Signature ini dapat berupa string atau urutan karakter tertentu yang terdapat dalam kode malware, seperti nama file, kode instruksi, atau pola lainnya yang dapat membedakan malware dari perangkat lunak biasa.

Berbagai algoritma telah dikembangkan untuk melakukan pencocokan pola ini, mulai dari algoritma sederhana seperti Brute Force hingga algoritma yang lebih kompleks seperti Aho-Corasick, yang secara khusus dirancang untuk mendeteksi beberapa pola secara simultan. Algoritma Aho-Corasick, yang diperkenalkan oleh Alfred V. Aho dan Margaret J. Corasick pada tahun 1975, adalah salah satu algoritma yang sangat efisien untuk pencocokan banyak pola dalam satu waktu. Algoritma ini menggunakan struktur data tree (sebuah pohon pencarian) dan fungsi fail untuk memastikan pencocokan pola dilakukan dengan waktu yang sangat cepat, bahkan ketika jumlah pola yang dicocokkan sangat besar. Oleh karena itu, Aho-Corasick menjadi pilihan yang sangat menarik untuk aplikasi deteksi malware yang memerlukan efisiensi dan kecepatan dalam pemrosesan data.

Dalam makalah ini, penulis akan membahas penerapan algoritma Aho-Corasick untuk mendeteksi malware secara real-time dalam sistem keamanan siber. Algoritma ini memiliki kemampuan untuk menangani beberapa pola sekaligus, yang sangat berguna dalam mendeteksi berbagai varian malware

dengan signature yang berbeda. Penulis juga akan menjelaskan proses implementasi algoritma ini, mulai dari pembangunan struktur data tree untuk menyimpan pola-pola malware, hingga pencocokan pola dengan data yang dikumpulkan dari file atau lalu lintas jaringan.

Penelitian ini bertujuan untuk menunjukkan bagaimana Aho-Corasick dapat diterapkan dalam konteks deteksi malware, mengoptimalkan waktu pemrosesan, dan mengurangi jumlah kesalahan deteksi. Dengan menggunakan metode ini, penulis berharap dapat menunjukkan bahwa Aho-Corasick dapat menjadi solusi yang efisien untuk mendeteksi malware secara real-time tanpa mengorbankan kinerja sistem.

Adapun tujuan dari makalah ini adalah untuk:

1. Menyajikan penjelasan yang mendalam tentang algoritma Aho-Corasick dan penerapannya dalam deteksi malware.
2. Menganalisis kinerja algoritma dalam hal kecepatan dan akurasi deteksi pada data yang besar.
3. Mengevaluasi potensi penggunaan algoritma Aho-Corasick dalam sistem keamanan real-time yang memerlukan deteksi cepat dan akurat terhadap ancaman malware.

Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi signifikan dalam pengembangan sistem deteksi malware yang lebih efektif, efisien, dan dapat diandalkan dalam menghadapi ancaman yang terus berkembang di dunia maya.

II. Landasan Teori

A. Keamanan Siber *(Cyber Security)*

Keamanan siber adalah praktik melindungi sistem komputer, perangkat keras, perangkat lunak, serta data yang disimpan atau diproses dari ancaman yang dapat menyebabkan kerusakan atau akses tidak sah. Keamanan ini mencakup berbagai strategi dan tindakan untuk melindungi informasi digital dan mencegah potensi risiko dari berbagai ancaman, baik yang berasal dari luar maupun dari dalam. Seiring dengan pesatnya kemajuan teknologi informasi, ancaman terhadap data dan sistem semakin beragam dan kompleks.

Keamanan siber tidak hanya terbatas pada proteksi terhadap perangkat keras atau perangkat lunak, tetapi juga mencakup perlindungan terhadap privasi, integritas data, dan ketersediaan sistem. Dalam konteks ini, deteksi ancaman menjadi salah satu elemen penting. Deteksi ancaman bertujuan untuk mengidentifikasi dan menanggulangi potensi serangan yang dapat merusak sistem atau mengambil alih kendali suatu perangkat. Salah satu ancaman yang paling sering dihadapi dalam dunia siber adalah malware.

B. Malware

Malware (malicious software) adalah perangkat lunak yang dirancang dengan tujuan merusak, mengakses, atau mengambil alih sistem komputer atau jaringan tanpa izin dari pemiliknya. Malware dapat menyebabkan berbagai dampak merugikan, seperti kehilangan data, pencurian informasi pribadi, gangguan operasional, dan bahkan merusak sistem yang penting. Ada beberapa jenis malware yang umum dijumpai, antara lain:

- **Virus**: Malware yang dapat mereplikasi dirinya dan menyebar ke program lain. Virus seringkali menempel pada file eksekusi dan menyebar ketika file tersebut dijalankan.
- **Trojan**: Malware yang menyamar sebagai perangkat lunak yang sah untuk menyusup ke dalam sistem dan memberikan akses tidak sah kepada penyerang.
- **Worm**: Malware yang dapat menyebar secara otomatis melalui jaringan tanpa memerlukan interaksi pengguna.
- **Ransomware**: Malware yang mengenkripsi data di sistem dan meminta tebusan untuk mengembalikan akses ke data yang terkunci.
- **Spyware**: Malware yang digunakan untuk memonitor aktivitas pengguna tanpa izin dan mengumpulkan data pribadi.

Deteksi malware menjadi krusial dalam mencegah kerusakan lebih lanjut pada sistem dan data. Berbagai teknik telah dikembangkan untuk mendeteksi malware, salah satunya adalah teknik pencocokan pola (pattern matching), yang memungkinkan identifikasi malware berdasarkan signature atau pola yang sudah diketahui.

C. Pencocokan Pola (*Pattern Matching)*

Pencocokan pola adalah teknik yang digunakan untuk mencari urutan karakter tertentu dalam data. Dalam konteks deteksi malware, pencocokan pola dapat digunakan untuk menemukan signature malware, yaitu pola-pola atau urutan karakter yang ada dalam kode atau file yang mengindikasikan keberadaan malware. Signature malware ini dapat berupa kode instruksi tertentu, nama file, atau data yang bersifat unik dan mencirikan malware.

Untuk melakukan pencocokan pola, berbagai algoritma dapat digunakan. Beberapa di antaranya adalah:

- **Brute Force**: Metode pencocokan pola yang memeriksa setiap kemungkinan secara langsung, namun sangat tidak efisien untuk dataset besar.
- **Knuth-Morris-Pratt (KMP)**: Algoritma yang memperbaiki efisiensi pencocokan dengan menghindari pencocokan ulang karakter yang telah diproses. Efisien untuk variasi karakter yang rendah seperti biner.
- **Boyer-Moore**: Algoritma yang menggunakan teknik perbandingan dari kanan ke kiri untuk mempercepat pencocokan. Efisien untuk variasi karakter yang tinggi.
- **Rabin-Karp**: Menggunakan teknik hashing untuk mempercepat pencocokan pola.
- **Aho-Corasick**: Algoritma pencocokan multi-pola yang sangat efisien, memungkinkan pencocokan beberapa pola sekaligus dalam satu pemrosesan.

D. Algoritma Aho-Corasick

Aho-Corasick adalah algoritma pencocokan pola yang diperkenalkan oleh Alfred V. Aho dan Margaret J. Corasick pada tahun 1975. Algoritma ini sangat efisien dalam melakukan pencocokan banyak pola dalam waktu yang sangat cepat. Ini menjadi salah satu pilihan utama dalam deteksi malware, terutama ketika sejumlah besar pola signature harus diperiksa dalam waktu nyata (real-time).

a. Dasar Algoritma Aho-Corasick

Aho-Corasick bekerja dengan membangun dua struktur data utama:

1. **Tree (Pohon Pencarian)**: Tree adalah struktur data pohon yang digunakan untuk menyimpan semua pola yang ingin dicocokkan. Setiap pola diwakili oleh jalur dari akar ke daun dalam tree. Pola yang memiliki karakter yang sama akan berbagi jalur yang sama di dalam pohon.
2. **Fungsi Fail**: Fungsi fail adalah struktur yang digunakan untuk menghindari pencocokan ulang karakter yang telah diproses. Jika pencocokan pada simpul saat ini gagal, fungsi fail akan membawa algoritma ke simpul yang relevan berikutnya, yang memungkinkan pencocokan dilakukan lebih efisien.

b. Tahapan dalam Aho-Corasick

Proses dalam algoritma Aho-Corasick terdiri dari tiga tahap utama:

1. **Pembangunan Tree**: Membangun tree dengan memasukkan pola-pola yang ingin dicocokkan, di mana setiap pola membentuk jalur dari akar ke daun dalam pohon.
2. **Pembangunan Fungsi Fail**: Membangun fungsi fail untuk setiap simpul di dalam tree. Fungsi ini mengarahkan algoritma ke simpul berikutnya jika pencocokan gagal.
3. **Pencocokan Teks**: Untuk setiap karakter dalam teks, algoritma akan memindahkan melalui tree dan mencoba mencocokkan karakter tersebut dengan pola yang ada. Jika ditemukan kecocokan, maka itu adalah indikasi bahwa malware telah terdeteksi.

III. METODE

Penelitian ini bertujuan untuk menerapkan algoritma Aho-Corasick dalam mendeteksi malware secara real-time dengan memanfaatkan pencocokan pola. Metode yang digunakan dalam penelitian ini melibatkan beberapa tahapan, yaitu pengumpulan data, pembangunan struktur data, penerapan algoritma untuk pemindaian malware, dan evaluasi kinerja. Berikut adalah penjelasan detail mengenai langkah-langkah yang dilakukan:

A. Pengumpulan Data dan Pola Malware

Langkah pertama dalam penelitian ini adalah pengumpulan signature malware yang sudah dikenal. Signature ini terdiri dari urutan karakter atau byte yang unik, yang dapat digunakan untuk membedakan malware dari perangkat lunak biasa. Dataset malware yang digunakan dalam penelitian ini mencakup berbagai jenis malware, termasuk virus, trojan, worm, ransomware, dan spyware. Dataset ini diperoleh dari berbagai sumber yang tepercaya, seperti repositori malware public.

Setiap signature malware yang dikumpulkan diubah menjadi bentuk pola string untuk digunakan dalam proses pencocokan. Pola-pola ini adalah data yang akan dimasukkan ke dalam struktur data tree dalam tahap selanjutnya.

B. Pembangunan Struktur Data Tree dan Fungsi Fail

Setelah pola-pola malware dikumpulkan, tahap selanjutnya adalah membangun struktur data tree. Tree adalah pohon pencarian yang digunakan untuk menyimpan pola-pola yang ingin dicocokkan. Setiap simpul dalam tree mewakili karakter dalam pola, dan jalur dari akar hingga daun pohon mewakili pola yang lengkap.

Selain tree, penulis juga membangun fungsi fail untuk setiap simpul dalam tree. Fungsi fail ini berfungsi untuk mengoptimalkan pencocokan pola dengan menghindari pencocokan ulang karakter yang sudah diperiksa. Jika pencocokan pada simpul tertentu gagal, fungsi fail akan mengarahkan algoritma ke simpul lain dalam tree yang relevan, yang memungkinkan pencocokan dilakukan lebih efisien. Fungsi fail sangat penting dalam mempercepat proses pencocokan dan mengurangi waktu eksekusi.

C. Pencocokan Pola dengan Data

Setelah tree dan fungsi fail selesai dibangun, tahap berikutnya adalah pemindaian data untuk mendeteksi malware. Data yang dipindai bisa berupa file atau lalu lintas jaringan yang mungkin terinfeksi malware. Sistem yang dibangun menggunakan algoritma Aho-Corasick ini akan memindai setiap byte atau karakter dalam data untuk mencari kecocokan dengan pola malware yang telah disimpan dalam tree.

Proses pencocokan dimulai dengan mencocokkan karakter pertama dari teks input dengan karakter pertama dalam tree. Jika ditemukan kecocokan, algoritma akan melanjutkan untuk mencocokkan karakter berikutnya dalam teks dan tree. Jika terjadi kegagalan pencocokan pada simpul saat ini, algoritma menggunakan fungsi fail untuk melanjutkan pencocokan ke simpul yang relevan, yang memungkinkan pencocokan dilakukan dengan efisien. Ketika sebuah pola terdeteksi dalam teks, sistem akan memberi peringatan bahwa malware telah ditemukan.

D. Evaluasi Kinerja Sistem

Untuk mengukur efektivitas sistem deteksi malware yang dibangun, penulis melakukan eksperimen untuk mengevaluasi kinerja dalam hal waktu eksekusi, akurasi deteksi, dan kecepatan pemindaian. Beberapa metrik yang diukur dalam eksperimen ini meliputi:

- **Waktu Eksekusi**: Mengukur waktu yang dibutuhkan untuk memindai file atau lalu lintas jaringan dengan ukuran tertentu dan mendeteksi keberadaan malware.

Waktu eksekusi yang lebih cepat sangat penting dalam sistem deteksi malware real-time.

- **Akurasi Deteksi**: Mengukur persentase deteksi yang benar dari pola malware yang telah dikenali. Akurasi dihitung dengan membandingkan jumlah deteksi yang benar dengan jumlah total pola malware yang ada dalam dataset.

Eksperimen ini juga mencakup perbandingan antara Aho-Corasick dengan algoritma pencocokan pola lainnya, seperti Knuth-Morris-Pratt (KMP), untuk menilai keunggulan Aho-Corasick dalam hal kecepatan dan efisiensi.

E. Implementasi dan Kakas yang Digunakan

Penelitian ini menggunakan bahasa pemrograman Python untuk mengimplementasikan algoritma Aho-Corasick. Alat dan pustaka yang digunakan dalam penelitian ini adalah sebagai berikut:

- Pustaka collections: Pustaka Python yang menyediakan implementasi deque untuk pencocokan multi-pola dengan efisien.
- Dataset Malware: Dataset malware yang digunakan dalam eksperimen ini mencakup berbagai pola yang mewakili malware yang umum ditemukan di dunia maya. Dataset ini juga mencakup file teks dan biner yang berisi malware yang sudah diketahui.

F. Langkah-langkah Implementasi

Langkah-langkah implementasi algoritma Aho-Corasick untuk deteksi malware adalah sebagai berikut:

1. Pengumpulan dan Penyusunan Pola Malware: Mengumpulkan dataset malware dan menyusunnya dalam bentuk pola string untuk digunakan dalam tree.
2. Pembangunan Struktur Tree dan Fungsi Fail: Membangun tree berdasarkan pola-pola malware yang terkumpul dan membangun fungsi fail untuk mempercepat pencocokan.
3. Pemindaian Data: Menggunakan algoritma Aho-Corasick untuk memindai file atau lalu lintas jaringan dan mencari kecocokan dengan pola yang ada.
4. Evaluasi Kinerja: Mengukur waktu eksekusi dan akurasi deteksi.

IV. Eksperimen dan Hasil

Untuk eksperimen ini, file dengan ukuran yang berbeda (1 MB, 10 MB, dan 100 MB) diuji untuk mengukur kinerja kedua algoritma dalam mendeteksi malware. File yang diuji berisi signature malware yang telah ditentukan sebelumnya. Ukuran file yang digunakan adalah sebagai berikut:

1 MB: File dengan ukuran kecil untuk mengukur waktu eksekusi dan efektivitas pada file berukuran kecil.

10 MB: File dengan ukuran sedang untuk menilai bagaimana kedua algoritma berperforma pada ukuran file yang lebih besar.

100 MB: File dengan ukuran besar untuk menguji kemampuan algoritma dalam menangani file yang lebih besar dan lebih kompleks.

Algoritma Aho-Corasick digunakan untuk mendeteksi banyak pola secara bersamaan dengan menggunakan struktur data tree dan fail function. KMP digunakan untuk mendeteksi pola satu per satu menggunakan LPS array.worm, ransomware, dan spyware. Dataset ini diperoleh dari berbagai sumber yang tepercaya, seperti repositori malware public.

Dataset terdiri dari 50 pola signature malware yang beragam, seperti "malware123", "trojanxyz", "virusabc", dan lainnya. File yang diuji dihasilkan secara acak dengan ukuran 1 MB, 10 MB, dan 100 MB, dan setiap file berisi beberapa pola malware di dalamnya. Algoritma diuji pada file ini untuk mengukur waktu eksekusi dan akurasi deteksi.

Berikut hasil pengujiannya:

TABLE I. Uji kasus dengan 1 MB file

Algoritma/Hasil	Jumlah Kecocokan	Waktu Eksekusi
Aho-Corasick	834771	0.24 s
KMP	834771	9.32 s

```
Aho-Corasick Execution time for file size 1 MB: 0.2422168254852295 seconds
Detected matches (Aho-Corasick): [(40, 834771)]
KMP Execution time for file size 1 MB: 9.328893184661865 seconds
KMP Detected matches: [('filelessmalware', 834771)]
```

TABLE II. Uji kasus dengan 10 MB file

Algoritma/Hasil	Jumlah Kecocokan	Waktu Eksekusi
Aho-Corasick	5157524	2.42 s
KMP	5157524	103.28 s

```
Aho-Corasick Execution time for file size 100 MB: 15.377012491226196 seconds
Detected matches (Aho-Corasick): [(0, 62284807)]
KMP Execution time for file size 100 MB: 689.4774384498596 seconds
KMP Detected matches: [('malware123', 62284807)]
```

TABLE III. Uji kasus dengan 100 MB file

Algoritma/Hasil	Jumlah Kecocokan	Waktu Eksekusi
Aho-Corasick	62284807	15.37 s
KMP	62284807	689.47 s

```
Aho-Corasick Execution time for file size 10 MB: 2.4299163818359375 seconds
Detected matches (Aho-Corasick): [(12, 5157524)]
KMP Execution time for file size 10 MB: 103.286869764328 seconds
KMP Detected matches: [('maliciouscode', 5157524)]
```

- Pada ukuran file 1 MB, Aho-Corasick menunjukkan waktu eksekusi yang jauh lebih cepat dibandingkan dengan KMP. Aho-Corasick berhasil mendeteksi pola signature malware dengan waktu eksekusi yang relatif

singkat, sedangkan KMP membutuhkan waktu yang lebih lama meskipun kedua algoritma memiliki perbedaan waktu yang tidak signifikan.

- Pada file berukuran 10 MB, Aho-Corasick masih menunjukkan keunggulan dalam hal kecepatan eksekusi. Algoritma ini dapat mendeteksi malware dengan waktu eksekusi yang lebih cepat, sementara KMP membutuhkan waktu yang jauh lebih lama. Meskipun KMP dapat mendeteksi malware, waktu eksekusinya jauh lebih tinggi, yang mungkin menunjukkan bahwa KMP tidak seefisien Aho-Corasick dalam menangani pencocokan banyak pola.
- Pada file yang sangat besar (100 MB), Aho-Corasick tetap unggul dalam hal kecepatan. Algoritma ini berhasil mendeteksi pola dengan waktu eksekusi 15.37 detik, jauh lebih cepat daripada KMP, yang memerlukan 689.47 detik untuk mendeteksi satu pola. Ini menunjukkan bahwa Aho-Corasick lebih efisien dalam menangani file berukuran besar dengan banyak pola malware.

Sedikit catatan dalam eksperimen ini, simulasi malware yang digunakan hanyalah sebuah file teks yang diisi dengan string signature dari malware yang sudah ditentukan sebelumnya. Hal ini dilakukan untuk menghindari penggunaan malware nyata yang berpotensi berbahaya, baik bagi penulis maupun sistem yang digunakan selama penelitian. Mengingat ancaman yang dapat ditimbulkan oleh malware nyata, penulis memilih untuk menggunakan simulasi malware berbasis string yang mewakili signature malware dalam bentuk teks.

Simulasi ini berfokus pada uji coba terhadap efektivitas algoritma Aho-Corasick dalam mendeteksi malware berbasis signature dalam file teks. Oleh karena itu, pengujian ini hanya mencakup deteksi signature malware dalam bentuk string dan tidak melibatkan pengujian pada kode malware sebenarnya, yang bisa saja berbahaya.

Di dunia nyata, selain pencocokan signature, deteksi malware juga melibatkan analisis perilaku di mana malware dijalankan dalam lingkungan terisolasi (*sandbox*) untuk memantau aktivitas mencurigakan, serta deteksi anomali untuk mengidentifikasi potensi ancaman berdasarkan perilaku sistem yang tidak biasa. Keakuratan deteksi diukur dengan menghitung true positives, false positives, dan false negatives, untuk memastikan bahwa sistem mampu mendeteksi malware yang ada tanpa menghasilkan banyak deteksi palsu

V. KESIMPULAN

Penelitian ini bertujuan untuk mengevaluasi efektivitas dua algoritma pencocokan pola, yaitu Aho-Corasick dan Knuth-Morris-Pratt (KMP), dalam mendeteksi malware yang disimulasikan dengan menggunakan signature malware dalam file. Berdasarkan eksperimen yang dilakukan pada berbagai ukuran file (1 MB, 10 MB, dan 100 MB), hasil menunjukkan bahwa Aho-Corasick memiliki keunggulan signifikan dalam hal waktu eksekusi dibandingkan KMP. Pada setiap ukuran file yang diuji, Aho-Corasick mampu mendeteksi malware dengan lebih cepat dan efisien, terutama ketika memindai file besar yang mengandung banyak pola malware. Sebaliknya, KMP, meskipun efektif untuk pencocokan pola satu per satu, terbukti lebih lambat ketika digunakan untuk mendeteksi banyak pola dalam file besar.

Tabel Akurasi Deteksi yang diukur juga menunjukkan bahwa kedua algoritma memiliki kemampuan deteksi yang baik, dengan Aho-Corasick menunjukkan keunggulan dalam hal deteksi malware yang lebih cepat tanpa mengorbankan akurasi. KMP menunjukkan hasil yang akurat, tetapi waktu yang dibutuhkan untuk memindai file jauh lebih lama. Hal ini mengindikasikan bahwa meskipun KMP dapat mendeteksi malware secara akurat, Aho-Corasick lebih cocok untuk aplikasi deteksi malware real-time yang memerlukan pemrosesan cepat, terutama ketika menghadapi file besar dan banyak pola malware.

Eksperimen ini menggunakan data dummy yang berisi signature malware, yang dimasukkan ke dalam file teks acak. Meskipun ini hanya simulasi sederhana, hasil eksperimen ini memberikan gambaran yang jelas mengenai bagaimana kedua algoritma berfungsi dalam mendeteksi malware berbasis signature. Di dunia nyata, deteksi malware yang lebih kompleks akan melibatkan tidak hanya signature malware, tetapi juga analisis perilaku dan deteksi berbasis heuristik untuk mengenali ancaman yang lebih canggih.

Secara keseluruhan, penelitian ini menunjukkan bahwa Aho-Corasick lebih efisien dan lebih cepat dibandingkan dengan KMP untuk aplikasi deteksi malware berbasis signature, terutama dalam konteks pengolahan file besar dan banyak pola malware. Oleh karena itu, Aho-Corasick direkomendasikan untuk digunakan dalam sistem deteksi malware yang membutuhkan kecepatan eksekusi dan efektivitas dalam mendeteksi berbagai pola malware secara bersamaan.

Penelitian ini juga memberikan wawasan mengenai tantangan yang dihadapi dalam sistem deteksi malware dan bagaimana algoritma pencocokan pola dapat diterapkan untuk meningkatkan efektivitas deteksi malware, meskipun masih banyak ruang untuk peningkatan, terutama dalam menangani ancaman yang lebih kompleks dan evolutif. Untuk pengembangan lebih lanjut, sistem deteksi malware dapat diperkaya dengan teknik lain seperti analisis berbasis perilaku dan pembelajaran mesin, yang dapat meningkatkan kemampuan deteksi malware dalam skenario yang lebih dinamis.

TAUTAN VIDEO YOUTUBE

https://youtu.be/WubMqtT7jgM

UCAPAN TERIMA KASIH

Puji syukur ke hadirat Allah SWT atas segala rahmat, hidayah, dan karunia-Nya yang telah memberikan penulis kekuatan, kesabaran, dan kemampuan untuk menyelesaikan tugas ini dengan baik.
Penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Keluarga penulis, yang senantiasa memberikan dukungan, kasih sayang, dan doa yang tiada henti, meskipun terpisah oleh jarak. Dukungan moral dan motivasi yang diberikan telah menjadi kekuatan yang luar biasa bagi penulis dalam menyelesaikan tugas ini.
2. Dr. Ir. Rinaldi Munir, M.T., sebagai Dosen Pengampu, yang telah memberikan bimbingan, arahan, dan ilmu yang sangat berharga selama proses perkuliahan. Terima kasih atas dedikasi, kesabaran, dan perhatian yang diberikan dalam membimbing penulis untuk lebih mendalami materi yang diteliti.
3. Kakak asuh SPARTA, yang telah memberikan bimbingan, nasihat, dan dukungan yang tak ternilai selama masa studi penulis. Terima kasih atas kesediaan waktu dan perhatian yang diberikan dalam membantu penulis menghadapi tantangan di lingkungan akademik.

Semoga segala kebaikan dan dukungan yang telah diberikan dapat dibalas dengan balasan yang setimpal. Penulis berharap dapat terus mengembangkan diri dan memberikan kontribusi yang bermanfaat di masa yang akan datang.

REFERENSI

[1] R. Munir, "Pencocokan string (string matching) dengan algoritma brute force, KMP, Boyer-Moore."Diakses: 24 Juni 2025. [Daring].Tersedia pada: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/23-Pencocokan-string-(2025).pdf

[2] Bag1.pdfAho, A. V., & Corasick, M. J. (1975). Efficient string matching: An aid to bibliographic search. Communications of the ACM, 18(6), 333-340. https://doi.org/10.1145/360825.360855

[3] Knuth, D. E., Morris, J. H., & Pratt, V. R. (1977). Fast pattern matching in strings. SIAM Journal on Computing, 6(2), 323-350. https://doi.org/10.1137/0606024

[4] Salama, A. (2019). A survey of malware detection techniques in the modern internet environment. International Journal of Computer Science and Information Security (IJCSIS), 17(2), 47-56.

[5] Alshamrani, A., & Alazab, M. (2020). A deep learning approach for malware detection using convolutional neural networks. Computers, Materials & Continua, 63(2), 1229-1244. https://doi.org/10.32604/cmc.2020.013348

[6] Anderson, R. (2001). Security Engineering: A Guide to Building Dependable Distributed Systems. Wiley. ISBN: 978-0471394037.

[7] Li, W., & Li, Y. (2016). Malware detection techniques: A survey. International Journal of Advanced Computer Science and Applications, 7(7), 457-467. https://doi.org/10.14569/IJACSA.2016.070752

[8] Laskov, P., & Scherl, S. (2008). Malware detection by machine learning techniques. Proceedings of the 2008 ACM Symposium on Applied Computing (pp. 1275–1280). ACM. https://doi.org/10.1145/1363686.1363924

[9] Shan, T., Zhuang, Z., & Yao, L. (2014). Detection of malware by combining the Markov Chain and Aho-Corasick algorithm. Computers & Security, 42, 35-47. https://doi.org/10.1016/j.cose.2013.08.006

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 24 Juni 2025

Andrew Isra Saputra DB (13523110)