ANALISIS KETEPATAN DETEKSI *MALWARE* PADA *SOFTWARE ANTIVIRUS* MENGGUNAKAN METODE ANALISIS STATIS

ACCURACY ANALYSIS OF MALWARE DETECTION IN ANTIVIRUS SOFTWARE USING STATIC ANALYSIS METHOD

Leidy Kurnia Hatika[1], Avon Budiyono[2], Ahmad Almaarif[3]

[1,2,3]Prodi S1 Sistem Informasi, Fakultas Rekayasa Industri, Universitas Telkom
[1]leidykurniahatika@student.telkomuniversity.ac.id, [2]avonbudi@telkomuniversity.ac.id,
[3]ahmadalmaarif@telkomuniversity.ac.id

Abstrak

***Malware* merupakan *software* yang melakukan *malicious action* dan dirancang untuk merusak, mengganggu kinerja komputer, mencuri informasi rahasia dan mengendalikan sistem dari jarak jauh tanpa diketahui oleh *user*. Berdasarkan laporan *Cyber Security Business* yang ditulis oleh Steve Morgan pada tahun 2016, menyebutkan bahwa seluruh dunia telah kehilangan data yang sebagian serangannya disebabkan oleh *malware* dan diperkirakan akan merugikan dunia lebih dari US $6 triliun pada tahun 2021. Dengan adanya permasalahan tersebut, maka diciptakan *antivirus* untuk melindungi komputer dari serangan *malware*. *Antivirus* menggunakan *signature* untuk mendeteksi *malware*, tetapi penggunaan *antivirus* untuk mendeteksi serangan *malware* berbeda-beda. Oleh karena itu diperlukan analisis yang bertujuan untuk memahami apa yang dilakukan oleh *malware* serta menganalisis ketepatan deteksi *malware* pada *antivirus*. Pada penelitian ini, analisis dilakukan menggunakan metode *static analysis* dan *scanning* sampel *malware* pada *software antivirus*. Analisis pada *software antivirus* dilakukan dengan cara melakukan *scanning* sampel *malware* pada Virus Total dengan melihat hasil *scanning* terbanyak, lalu dilanjutkan dengan mencari tahu karakteristik dari sampel *malware* yang diuji. Dari pengujian yang dilakukan, hasil yang didapatkan menggunakan metode *static analysis* hanya berfokus pada *value string* yang ditemukan pada sampel *malware* dan dibandingkan dengan karakteristik *malware* berdasarkan hasil *scanning* terbanyak pada *software antivirus*. Hal ini bertujuan untuk melihat apakah hasil deteksi *malware* pada *software antivirus* sesuai dengan hasil analisis yang didapatkan dengan metode *static analysis.* Dari penelitian yang dilakukan, hanya tiga dari sepuluh sampel *malware* yang dapat dianalisis dan diperoleh hasil ketepatan deteksi pada sampel pertama sebagai *spybot* sebesar 100%, sampel kedua sebagai *trojan* sebesar 75% dan sampel ketiga sebagai *trojan* sebesar 87,5%.**

Kata kunci: ***malware, malware analysis, static analysis malware, antivirus.***

Abstract

Malware is software that performs malicious actions and is designed to damage, regulate computers, open confidential information and control the system remotely without being noticed by the user. Based on the Cyber Security Business report written by Steve Morgan in 2016, said that the whole world had lost data, some of which were caused by malware and expected to harm the world more than the US $ 6 trillion in 2021. With these problems, the antivirus was created to protect computers from malware attacks. Antivirus uses signatures to detect malware, but the use of antivirus to detect malware attacks has different results. Therefore the analysis is needed to understand what is done by malware and analyze the accuracy of detection of malware on antivirus. In this research, the analysis was carried out using static analysis methods and scanning malware samples on antivirus software. Analysis of antivirus software is done by scanning malware samples in the Virus Total by seeing the most scanning results, then proceed to find out the characteristics of the tested malware samples. From the tests, the results obtained by using static analysis focuses only on the value's string found in the malware samples and compared with the characteristics of malware based on the most scanning results on antivirus. This is intended to see whether the results of malware detection on software are in accordance with the results of the analysis obtained by static analysis methods. From this research, only three out of ten malware samples can be analyzed and the results of the accuracy of detection in the first sample as Spybot were obtained by 100%, the second sample as a Trojan by 75% and the third sample as a Trojan by 87.5%.

Keywords: malware, malware analysis, static analysis malware, antivirus.

1. Pendahuluan

Malware merupakan *software* yang dirancang untuk merusak komputer, mencuri informasi rahasia dan mengendalikan suatu sistem dari jarak jauh. Serangan *malware* bertujuan untuk membuat komputer korban melakukan instalasi *software* berbahaya tanpa sepengetahuan korban [2]. Ada berbagai macam *malware* yang tersebar pada saat ini, diantaranya yaitu *virus*, *worm*, *trojan*, *rootkit*, *ransomware*, *backdoor* dan lain-lain.

Berdasarkan laporan *Cyber Security Business* yang ditulis oleh Steve Morgan pada tahun 2016, menyebutkan kerusakan yang ditimbulkan *malware* terhadap ekonomi dunia dan aset perusahaan swasta juga meningkat setiap harinya. Melihat kondisi saat ini maka diciptakan *software* untuk melindungi komputer dari serangan *malware* yang biasa dikenal dengan *antivirus*.

Antivirus merupakan sebuah *software* yang berfungsi untuk melindungi perangkat *digital* seperti komputer, *mobile*, dan *pen-drive* dari serangan *malware* [3]. *Antivirus* menggunakan *signature* yang berupa fitur unik dari *malware* yang mirip dengan sidik jari untuk mendeteksi *malware*. *Antivirus* juga menggunakan jenis *scanning* yang berbeda-beda seperti *full scan, quick scan* dan *custom scan* untuk mendeteksi suatu *malware* [1]. Penggunaan *antivirus* untuk mendeteksi *malware* tidak selalu tepat. Ada kondisi dimana *antivirus* salah mendeteksi suatu *file* yang telah terinfeksi *malware*, padahal *file* tersebut tidak terinfeksi *malware*, hal ini disebut dengan *false positive*. Selain itu juga ada kondisi dimana *antivirus* tidak mendeteksi keberadaan *malware* pada suatu *file*, padahal *file* tersebut terinfeksi *malware*, hal ini disebut dengan *false negative* [4].

Untuk memahami apa yang dilakukan oleh *malware*, maka perlu dilakukannya *malware analysis*. *Malware analysis* merupakan salah satu cara penanganan serangan terhadap *malware* dengan cara memahami cara kerja, cara mengidentifikasi, dan cara mengalahkan atau menghilangkan *malware* tersebut, salah satunya yaitu dengan menggunakan metode *static analysis* untuk memudahkan kita dalam mengetahui karakteristik dan perilaku *malware. Static analysis* banyak digunakan dalam menganalisis *malware* yang dapat dilakukan dengan mudah dan cepat menggunakan *static analysis tools* tertentu tanpa harus menjalankan *malware* tersebut. Selain itu, tujuan dilakukannya analisis *malware* adalah untuk memahami bagaimana suatu *malware* bekerja [2]. Pada penelitian ini metode *static analysis* dilakukan dengan cara melihat *value string* yang terdapat pada *malware* dengan tujuan untuk mengetahui bagaimana suatu fungsi dijalankan oleh *malware*.

Metode *static analysis* dan *software antivirus* dapat digunakan dalam menganalisis *malware* untuk melihat karakteristik, perilaku atau fungsi yang dijalankan dari suatu *malware.* Oleh karena itu dalam hal ini penulis akan membandingkan ketepatan deteksi atau *scanning* suatu *malware* menggunakan *softaware antivirus* dan metode *static analysis.*

2. Dasar Teori

2.1 *Malware*

2.1.1 Pengertian *Malware*

Malicious software atau yang biasa dikenal dengan *malware* merupakan sebuah perangkat lunak yang terpasang pada suatu sistem komputer tanpa sepengetahuan oleh *user* atau pemilik sistem tersebut [1]. Sesuai dengan namanya, *malware* dapat melakukan *malicious action* atau tindakan jahat seperti mencuri informasi rahasia, perusakan pada suatu sistem, mendapatkan hak akses suatu komputer dan menjalankan program yang ada pada komputer tersebut. Setiap perangkat lunak yang melakukan sesuatu yang dapat menyebabkan kerugian pada *user*, komputer ataupun jaringan dapat dianggap sebagai *malware* [2].

2.1.2 Klasifikasi *Malware*

Malware dapat diklasifikasikan dalam berbagai kategori. Klasifikasi *malware* umumnya dikategorikan berdasarkan proses propagasi *malware* dan reaksi *malware* pada suatu sistem yang telah terinfeksi. *Malware* diklasifikasikan berdasarkan 3 hal berikut [6]:

1. *Contagios Threats*
   Jenis *malware* yang tergolong kedalam klasifikasi *malware* berdasarkan *contagious threat* diantaranya yaitu:
   a. *Virus*, merupakan salah satu bentuk *malware* yang mengambil alih suatu kontrol pada suatu komputer yang terinfeksi secara tidak sah dan menyebabkan kerusakan tanpa sepengetahuan pengguna.
   b. *Worm*, merupakan *malicious software* yang berdiri sendiri yang dapat beroperasi secara independen dan tidak menghubungkan dirinya untuk melakukan tindakan tertentu. Kerusakan yang disebabkan oleh *worm* yaitu mengkonsumsi sebagian besar memori sistem *resource* dan masalah *network performance*

2. *Masked Threats*
   Jenis *malware* yang tergolong kedalam klasifikasi *malware* berdasarkan *masked threat* diantaranya yaitu:
   a. *Trojan*, merupakan *malware* mematikan yang tersembunyi dan berperilaku sebagai program yang sah untuk mengambil alih kendali suatu komputer atau sistem secara tidak sah. Kerugian yang disebabkan oleh *trojan* yaitu mencuri *password* atau rincian *login*, pencurian uang digital, memodifikasi/menghapus *file*, memantau aktivitas *user.*
   b. *Backdoors*, merupakan *malware* yang dapat menembus suatu *security control* dan memberikan hak akses yang tidak sah kepada *attacker*. Hal yang dapat ditimbulkan oleh *backdoors* yaitu memodifikasi dan menghapus *file* sistem dan memonitor aktivitas sistem.

c. *Adware,* merupakan *malware* yang dapat memberikan informasi kepada pengiklan tentang *browsing habit* pengguna, sehingga memungkinkan pengiklan untuk memberikan penambahan target kepada pengguna. Hal yang dapat ditimbulkan oleh *adware* yaitu *clickjacking*, *phising* atau membuat suatu aktivitas jahat menggunakan *browser*.
d. *Rootkits,* merupakan teknik penyamaran untuk *malware*, pada dasarnya *rootkits* dirancang untuk maksud jahat dari program tersebut. Hal yang dapat ditimbulkan oleh *malware* jenis ini yaitu mencuri *password* atau menginstall *Keyloggers.*

3. *Financial Threats*
Jenis *malware* yang tergolong kedalam klasifikasi *malware* berdasarkan *financial threat* diantaranya yaitu:
a. *Ransomware,* merupakan suatu *software* yang dirancang untuk memblokir akses ke sebuah sistem komputer sampai sejumlah uang dibayarkan. Hal yang dapat ditimbulkan oleh *malware* jenis ini yaitu pencurian data, mengenkripsi data korban serta membatasi *user* untuk mengakses sistem mereka.
b. *Spyware,* merupakan malware yang dapat melacak aktivitas tanpa sepengetahuan pengguna dan mengirimkan informasi sensitif kepada *attacker*. Hal yang dapat dilakukuan oleh *spyware* yaitu melakukan *sniffing* pada *network interface,* sertifikan digital, *encryption keys* dan informasi sensitif lainnya.
c. *Keylogger,* merupakan *malware* yag secara diam-diam merekam *keystrokes.* Hal yang dapat dilakukan oleh *malware* jenis ini yaitu menangkap informasi sensitif seperti *username, password,* nomor kartu kredit atau detail dari suatu perbankan *online*.

2.1.3 Jenis-Jenis *Malware*

Malware dibedakan menjadi beberapa jenis, sebagai berikut [2]:

1. *Backdoor,* merupakan sebuah *malicious code* yang dapat terinstal secara otomatis pada sebuah komputer. yang memungkinkan untuk mendapatkan akses untuk melakukan penyerangan. *Backdoor* memungkinkan *attacker* untuk mengakses komputer yang telah terinfeksi tersebut dari jarak jauh. *Backdoor* memungkinkan *attacker* terhubung ke suatu komputer dengan sedikit atau tidak adanya otentikasi serta menjalankan perintah di sistem lokal pada suatu komputer.
2. *Botnet,* hampir mirip dengan *backdoor*, yaitu *botnet* juga memungkinkan *attacker* untuk dapat mengakses komputer yang sudah terinfeksi *botnet,* tetapi semua komputer yang sudah terinfeksi *botnet* yang sama akan menerima perintah dari *attacker* melalui *single command-and-control server* yaitu *server* yang digunakan sebagai media komunikasi antara *malware* dengan *attacker.*
3. *Downloader,* merupakan *malicious code* yang digunakan untuk mengunduh *malicious code* lainnya. *Downloader* biasanya dipasang oleh *attacker* ketika pertama kali mendapatkan akses ke suatu sistem.
4. *Informatioan-stealing Malware,* merupakan *malware* yang digunakan untuk mengumpulkan informasi dari komputer korban dan akan mengirimkan informasi yang sudah didapat ke *attacker*. *Malware* jenis ini biasanya digunakan untuk mendapatkan akses ke akun *online* seperti *email* atau akun perbankan.
5. *Launcher,* merupakan *malicious program* yang digunakan untuk meluncurkan *malicious program* lainnya. *Launcher* biasanya digunakan untuk menghindari deteksi *antivirus* dan juga digunakan untuk mendapatkan akses sistem yang lebih dalam. Biasanya, *launcher* menggunakan teknik tradisional untuk menjalankan *malicious program* guna memastikan akses tersembunyi pada suatu sistem.
6. *Rootkit,* merupakan *malicious code* yang dirancang untuk menyembunyikan keberadaan *malware* lainnya. *Rootkit* biasanya diintegrasikan dengan *malware* lainnya, seperti *backdoor,* dengan tujuan agar *attacker* dapat mengakses komputer korban dari jarak jauh *(remote).*
7. *Scareware,* merupakan *malware* yang dirancang untuk menakut-nakuti korban sehingga membuat korban menjadi terpaksa untuk membeli sesuatu. Pada umumnya *scareware* memiliki *interface* yang menyerupai *antivirus.*
8. *Spam-sending Malware,* merupakan sebuah *malware* yang dirancang untuk menginfeksi komputer pengguna dan menggunakan komputer tersebut untuk mengirim *spam*.
9. *Worm* atau *virus,* merupakan jenis *malware* yang dapat menggandakan dirinya sendiri dan menginfeksi komputer *user*. *Worm* merupakan program yang berdiri berdiri sendiri dan menjankan programnya tanpa bergantung pada program lain. Sedangkan *virus* menyebar melalui program-program yang telah terinfeksi sebelumnya dan *virus* baru akan aktif apabila program tersebut dijalankan.

2.2 *Malware Analysis*

Malware analysis merupakan adalah seni membedah *malware* untuk memahami bagaimana cara kerja, cara mengidentifikasi, dan cara mengalahkan atau menghilangkan *malware* tersebut [2]. *Malware analysis* juga bisa diartikan sebagai proses untuk menentukan fungsionalitas *malware* dan memberikan solusi untuk mencegah akibat yang ditimbulkan oleh *malware* tersebut [1]. Ada dua metode yang dapat dilakukan untuk

menganalisis *malware*, yaitu *static analysis* dan *dynamic analysis*. Tujuan dari adanya *malware analysis* yaitu untuk mendapatkan pemahaman tentang bagaimana cara kerja, ataupun fungsi bagian tertentu dari suatu *malware* sehingga dapat dibangun pertahanan atau pencegahan agar dapat melindungi jaringan atau suatu perangkat komputer.

2.3 *Static Analysis*

Static analysis merupakan salah satu metode pada analisis *malware* yang dilakukan pada saat sebelum *malware* dieksekusi [5]. Pada *static analysis* juga terdapat teknik yang dapat digunakan untuk menganalisis *malware* yaitu [6]:

1. Teknik *signature based detection*, merupakan teknik yang dilakukan dengan melihat *signature* atau urutan bit yang disuntikkan dalam program aplikasi oleh *malware writer*, yang secara unik mengidentifikasi jenis *malware* tertentu. Untuk mendeteksi *malware* dalam kode, detektor *malware* akan mencari *siganture* yang telah ditentukan sebelumnya dalam kode. Teknik ini juga dikenal sebagai teknik pencocokan pola atau sidik jari.
2. Teknik h*euristic detection*, merupakan teknik yang memungkinkan detektor *malware* untuk mencari perintah yang tidak ada dalam program aplikasi. Teknik ini juga dikenal sebagai teknik proaktif.

Mengestrak informasi dan melakukan analisis *malware* menggunakan metode *static analysis* dapat dilakukan dengan beberapa cara yaitu sebagai berikut [1]:

1. *Antivirus Scanning*, digunakan sebagai salah satu langkah pertama untuk menganaliss *malware* yaitu dengan menggunakan *signature* dari *file* yang menunjukkan kode yang mencurigakan. Hal pertama yang dilakukan oleh *antivirus* yaitu menyimpan *malware signature* yang telah ditetapkan ke dalam *database.* Kedua, pada saat proses *scanning*, *antivirus* akan menemukan *signature* pada *file* yang di *scan* lalu membandingkannya dengan *signature* yang telah tersimpan pada *database*. Jika *signature* ditemukan pada *database* dan ditandai sebagai *malware*, maka *antivirus scanner* akan memberikan notifikasi bahwa *file* tersebut terdeteksi sebagai *malware*. *Antivirus* dapat bekerja dengan cepat dan akurat untuk mendeteksi *malware* yang telah diketahui, tetapi *antivirus* gagal dalam mendeteksi *malware* yang tidak dikenal, karena pada umumnya *malware* yang tidak dikenal merupakan varian *malware* yang dapat mengubah *signature* yang dimilikinyanya untuk membypass *antivirus*.
2. *Hashing*, merupakan metode yang digunakan untuk mendefinisikan *malware*. Pertama, sampel dikirimkan ke *hashing program* dan nilai *hash* akan diekstrak menjadi MD5, SHA-1 atau SHA-256. Setelah itu nilai *hash* hasil ekstraksi akan dicari secara *online* dan dilihat apakah nilai *hash* tersebut mengidentifikasi *malware* atau tidak.
3. *String Analysis*, mencari informasi melalui *string analysis* biasanya dilakukan menggunakan *string extraction tools.* Mencari string dapat memberikan informasi tentang *malware* seperti URL yang terkait dengan *malicious code*, alamat *email* milik *attacker* atau informasi yang berhubungan dengan *malicious password.*
4. *Reverse Compiling*, pada *reverse compiling* membutuhkan versi biner dari sampel *malware* dan menghasilkan instruksi level *assembly*. Menganlisis *malware* menggunakan cara *reverse compiling* memerlukan pemahaman yang mendalam tentang bahasa level *assembly*.

2.4 *Antivirus*

Antivirus merupakan sebuah perangkat lunak yang digunakan untuk mencegah suatu perangkat seperti komputer, ponsel ataupun *pen-drive* agar terhindar dari *malware* [1]. Ada beberapa teknik deteksi *malware* yang umum digunakan oleh *antivirus*, adapun tekniknya yaitu sebagai berikut:

1. *Scanning*, merupakan sebuah proses mencari apakah pada sebuah *file* terdapat *signature malware*. Bila ada sebuah *file* yang dicurigai sebagai *malware*, maka *file* tersebut terlebih dulu dianalisis. *Malware* ini dipelajari perilaku dan karakteristiknya, kemudian dibuat sebuah *signature*. Bila terdapat *signature malware* pada *file* tersebut berarti *file* tersebut telah terinfeksi *malware*, bila tidak maka *file* tersebut bebas dari *malware*.
2. *Static Heuristic*, digunakan untuk *malware* yang belum dikenali karakteristiknya, berbeda dengan teknik *scanning* yang digunakan untuk melakukan deteksi terhadap *malware* yang sudah dikenali karakteristiknya. Teknik ini tidak melakukan pencarian *signature malware*, tapi akan berusaha membuat *signature* baru. Jadi teknik ini berusaha menduplikasi cara seorang analis *malware* dalam mengenali *malware* dari *source codenya*. Umumnya teknik ini akan mencari pada *source code* apakah terdapat aktifitas atau fungsi yang mencurigakan. Contohnya adalah adanya fungsi untuk menggandakan dirinya (*replikasi*), atau fungsi untuk mengelabui *antivirus* (*obfuscation*) dan lain-lain.
3. *Integrity Check*, *malware* yang menginfeksi sebuah *file* umumnya akan melakukan modifikasi pada *file*. Sehingga bila terjadi perubahan pada *file* tersebut tanpa adanya authorisasi yang jelas, maka aktifitas ini akan dicurigai sebagai adanya *malware*. Teknik ini umumnya menggunakan *checksum*, jadi *antivirus* akan

melakukan *checksum* terhadap *file*, kemudian *checksum* ini akan di input ke dalam *database*. Bila *antivirus* melakukan *scanning* maka *checksum* terbaru akan dibandingkan dengan *checksum* yang ada di *database*. Bila terjadi perubahan maka *antivirus* akan memberikan alarm.

2.5 Metodologi Penelitian

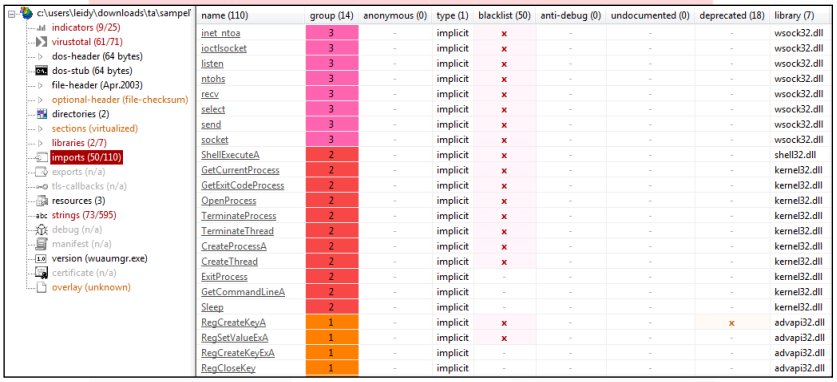
Metodologi penelitian dilakukan menggunakan model konseptual untuk memetakan permasalahan terhadap penelitian yang dilakukan. Permasalahan dari penelitian ini yaitu semakin banyaknya jenis *malware* yang dapat menyerang komputer *user* dan hasil yang tidak tepat dalam deteksi malware menggunakan *software antivirus* sehingga diperlukan analisis *malware* menggunaan metode *static analysis.*

Analisis *malware* dilakukan menggunakan *static analysis tools* dan *scanning* pada *software antivirus* Analisis menggunakan *static analysis tool* dilakukan untuk melihat API *call* dan *library* yang di *import* oleh sampel *malware*. Adapun dasar teori yang dgiunakan yaitu tentang static analysis, behavior atau karakteristik malware dan *value string* yang dihasilkan ketika sampel *malware* dijalankan. Hasil yang didapatkan akan dibandingkan dengan hasil deteksi pada *software antivirus.*

3. Pengujian dan Analisis

3.1 Pengujian Menggunakan PE Studio

PE Studio merupakan salah satu *static analysis tool* untuk melihat API *call* dan *library* yang di *import* oleh sampel *malware*.

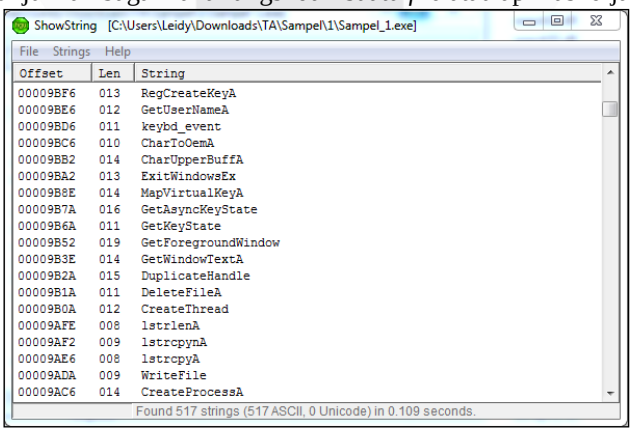


Gambar 1 Pengujian Menggunakan PE Studio

Pada Gambar 1 di atas dapat dilihat API *call* dan *library* apa saja yang di *import* oleh sampel *malware* dan dibedakan berdasarkan warna pada masing-masing API *call group*. Contohnya `GetShellExecuteA` merupakan API *call* yang terdapat pada *library* shell32.dll yang berfungsi untuk melakukan proses eksekusi atau mengekstrak suatu *file* atau direktori. API *call* ini termasuk ke dalam API *call execution group.*

3.2 Pengujian Menggunakan ShowString

ShowString digunakan untuk mendapatkan informasi yang menunjukkan semua *string* pada sebuah *file* atau aplikasi. *String* menunjukkan bagaimana fungsi dari suatu *file* atau aplikasi dijalankan.

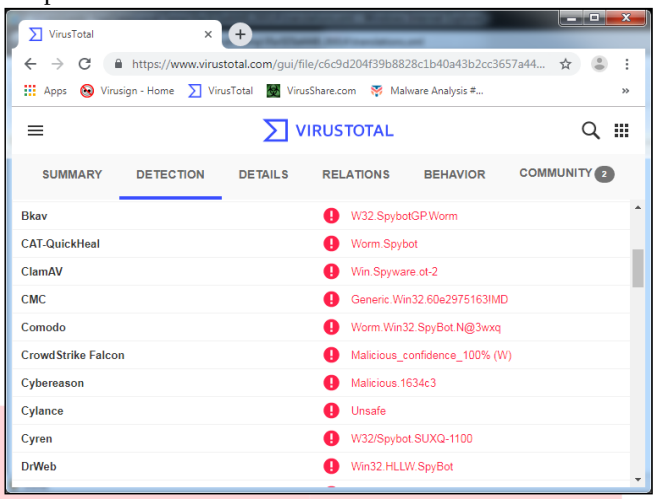


Gambar 2 Pengujan Menggunakan ShowString

Pada Gambar 2 di atas dapat dilihat *string* yang digunakan oleh salah satu sampel *malware*. Pada gambar terdapat value string `keybd_event` merupakan *string* yang berfungsi untuk menyimpan kejadian yang terjadi ketika *user* menekan atau melepas tombol *keyboard*.

3.3 Pengujian Menggunakan Virus Total

Virus Total digunakan untuk melakukan *scanning* terhadap sampel *malware* berdasarkan beberapa *software antivirus* yang ada pada Virus Total.



Gambar 3 Pengujian Menggunakan Virus Total

Pada Gambar 3 diatas dapat dilihat hasil *scanning* salah satu sampel *malware* pada Virus Total. Pada gambar dapat dilihat beberapa *software antivirus* beserta hasil deteksinya.

3.4 Analisis Menggunakan Metode *Static Analysis*

Analisis dilakukan dengan membandingkan hasil deteksi terbanyak yang didapatkan ketika melakukan *scanning malware* pada *software antivirus* dengan *value string* dari sampel *malware*. Pengujian yang dilakukan pada tiga sampel malware pada Virus Total, diperoleh sampel *malware* terdeteksi sebagai *Spybot* dan *Trojan*. Setelah itu dilakukan pencarian karakteriskit dari *Spybot* dan *Trojan*, penelitian terhadap sampel *malware* dilanjutkan dengan mencari *value string* yang sesuai dengan karakteristik yang ada. Tujuan dilakukannya perbandingan terhadapat sampel *malware* yaitu untuk membuktikan ketepatan deteksi antara *software antivirus* dengan metode *static analysis.*

Dari analisis yang dilakukan penulis juga melakukan perbandingan *value string* antara sampel bersih dan sampel yang terkena *malware*. Dari pengujian yang dilakukan diperoleh hasil bahwa terdapat *value string* yang sama-sama dijalankan oleh sampel bersih dan sampel *malware.*

Tabel 1 Perbandingan *Value String* pada Sampel Bersih dan Sampel *Malware*

Value String	Sampel *Malware*				
	Bersih_1	**Bersih_2**	**1**	**2**	**3**
ExitProcess	v	v	v	v	v
TerminateProcess	v	-	v	v	v
killprocess	-	-	v	-	-
TerminateThread	-	-	v	-	-
WNetEnumCachedPasswords	-	-	v	-	-
FindFirstFile	v	v	v	v	v
FindNextFileA	v	-	v	-	
keybd_event	-	-	v	-	-
GetKeyState	-	-	v	-	v
GetAsyncKeyState	-	-	v	-	v
Startkeylogger	-	-	v	-	-
ShellExecuteA	-	-	v	-	v
ShellExecuteEx	-	-	-	-	v
GetCursorPos	-	-	-	v	v
GetAsyncKeyState	-	-	v	v	-
GetKeyState	-	-	v	v	-
VkKeyScanA	-	-	-	-	v
SystemParametersInfo	-	-	-	v	-
RegCreateKeyExA	-	-	v	v	v
RegSetValue	-	-	v	v	v
RegDeleteValueA	-	-	-	v	v
GetCurrentProcess	-	v	v	v	v

GetCurrentProcessId	-	-	-	v	v
CreateProcess	v	v	v	-	v
CloseDesktop	-	-	-	v	v
EnumDesktopWindows	-	-	-	v	-
OpenDesktopA	-	-	-	-	v

Dari hasil perbandingan pada Tabel 1 diatas dapat disimpulkan bahwa terdapat fungsi yang sama-sama dijalankan oleh sampel bersih dan sampel malware yaitu *value string* `ExitProcess`, `TerminateProcess`, `CreateProcess` dan `GetCurrentProcess` merupakan fungsi yang berkaitan dengan proses yang sedang berjalan pada suatu aplikasi atau program. Selain itu juga terdapat *value string* `FindFirstFile` dan `FindNextFile` merupakan fungsi yang berkaitan untuk mencari *file* yang sudah ditentukan secara spesifik. Enam *value string* tersebut tidak dapat disimpulkan sebagai fungsi yang dijalankan hanya pada program yang terdeteksi sebagai *malware* karena value string tersebut juga ditemukan pada sampel bersih.

4. Kesimpulan

Berdasarkan hasil analisis yang didapatkan dari pengujian *malware* menggunakan metode *static analysis* dengan melakukan analisis pada *value string* dan *scanning* pada *software antivirus*, dapat diambil kesimpulan sebagai berikut:

1. Analisis *malware* menggunakan metode *static analysis* dilakukan dengan cara menguji sampel *malware* pada *static analysis tools* yaitu PE Studio dan ShowString. Informasi yang didapatkan dari kedua *tools* tersebut yaitu API *call* dan *library* yang di *import* serta *value string* untuk melihat bagaimana fungsi yang dijalankan oleh sampel *malware.*
2. Analisis *malware* menggunakan *software antivirus* dilakukan dengan cara *scanning* sampel *malware* pada Virus Total yang merupakan kumpulan dari beberapa *software antivirus.* Informasi yang didapatkan yaitu berupa hasil *scanning* dari beberapa *software antivirus* beserta hasil deteksinya. Selanjutnya akan diambil jenis *malware* berdasarkan hasil *scanning* terbanyak pada sampel *malware* yang diuji.
3. Membandingkan ketepatan deteksi antara *software antivirus* dengan metode *static analysis* dilakukan dengan cara mencari karakteristik *malware* yang diperoleh dari hasil *scanning* terbanyak pada Virus Total. Setelah karakteristik *malware* ditemukan, akan dilanjutkan dengan menganalisis *value string* pada sampel *malware* yang sesuai dengan karateristik *malware*. Jika *value string* yang ditemukan mewakili karakteristik *malware,* maka hasil deteksi pada *software antivirus* dapat dikatakan tepat dan sesuai dengan hasil analisis yang dilakukan dengan metode *static analysis*. Dari sepuluh sampel *malware* yang diuji, hanya tiga sampel *malware* yang dapat dianalisis. Untuk sampel pertama yang terdeteksi sebagai *spybot*, semua karakteristiknya dapat dibuktikan dengan *value string* yang dihasilkan. Untuk sampel kedua yang terdeteksi sebagai *trojan*, hanya enam dari delapan karakteristik *trojan* yang dapat dibuktikan dari *value string* yang dihasilkan. Untuk sampel ketiga, hanya tujuh dari depalan karakteristik *trojan* yang dapat dibuktikan dari *value string* yang dihasilkan. Oleh karena itu, dapat disimpulkan ketepatan deteksi *malware* pada *software antivirus* dengan cara membandingkan karakteristik *malware* dari masing-masing sampel program dengan *value string* yang dihasilkan adalah sampel pertama 100% terbukti sebagai *spybot*, sampel kedua 75% terbukti sebagai *trojan* dan sampel ketiga 87,5% terbukti sebagai *trojan*.

5. Saran

Berdasarkan hasil analisis dan pengujian yang telah dilakukan, saran yang dapat diberikan oleh penulis yaitu analisis *malware* menggunakan metode *static analysis* dapat dilakukan selain dengan cara analisis *value string* dan *antivirus scanning*, misalnya dengan cara hasing, reverese compiling dan lain-lain. Selain itu akan lebih baik jika analisis *malware* mengunakan metode *static analysis* juga diiringi dengan menggunakan metode *dynamic analysis* agar pembuktian deteksi terhadap sampel *malware* lebih akurat

Daftar Pustaka:

[1] ASLAN, Ö. (2017). Performance Comparison of Static Malware Analysis Tools Versus Antivirus Scanners To Detect Malware. *International Multidisciplinary Studies Congress (IMSC)*. Diakses pada tanggal 10 September 2018, dari https://www.researchgate.net/publication/321759536_Performance_Comparison_of_Static_Malware_Analysis_Tools_Versus_Antivirus_Scanners_To_Detect_Malware

[2] Sikorski, M., & Honig, A. (2012). *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. *Computers & Security* (Vol. 31). https://doi.org/10.1016/j.cose.2012.05.004

[3] Thomas, R., & Nachamai, M. (2017). Performance Investigation of Antivirus - A Comparative Analysis. *Oriental Journal Of Computer Science And Technology*, *10*(1), 201-206. doi: 10.13005/ojcst/10.01.27

[4] Ismail, J. (2019). Bagaimana cara kerja Antivirus | Jul Ismail. Diakses pada tanggal 18 Novemver 2018, dari https://julismail.staff.telkomuniversity.ac.id/bagaimana-cara-kerja-antivirus/
[5] Uppal, D., Mehra, V., & Verma, V. (2014). Basic survey on Malware Analysis, Tools and Techniques. *International Journal On Computational Science & Applications*, *4*(1), 103-112. doi: 10.5121/ijcsa.2014.4110
[6] Parsons, T. (2009). *A False Positive Prevention Framework for Non-Heuristic Anti-Virus Signatures*. Symantec. Diakses pada tanggal 15 November 2018, pada https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/security-response-false-positive-prevention-framework-09-en.pdf
[7] Pandey, S. K., & Mehtre, B. M. (2014). Performance of Malware Detection Tools: A Comparison. *International Conference on Advanced Communication Control and Computing Technologies*, (978), 1811–1817. Diakses pada tanggal 13 November 2018, pada https://ieeexplore.ieee.org/document/7019422
[8] Yusirwan, S., Prayudi, Y., & Riadi, I. (2015). Implementation of Malware Analysis using Static and Dynamic Analysis Method. *International Journal of Computer Applications*, *117*(6), 11–15. https://doi.org/10.5120/20557-2943
[9] Shahegh, P., Dietz, T., Cukier, M., Algaith, A., Brozik, A., & Gashi, I. (2017). AVAMAT: AntiVirus and Malware Analysis Tool. *International Symposium on Network Computing and Applications*. Diakses pada tanggal 5 Juni 2019, pada https://ieeexplore.ieee.org/document/8171379
[10] Harley, D., & Vibert, R. (2014). *AVIEN Malware Defense Guide for the Enterprise*. Burlington: Elsevier Science.
[11] Egele, M. (n.d.). A Survey on Automated Dynamic Malware Analysis Technique and Tools, *V*, 1–49. Diakses pada tanggal 8 Juni 2019, pada https://www.teliacompany.com/sv/om-foretaget/
[12] Bergeron, J., Debbabi, M., Desharnais, J., Erhioui, M., Lavoie, Y., & Tawbi, N. (2009). Static Detection of Malicious Code in Executable Programs. Diakses pada tanggal 9 Juni 2019, pada https://www.researchgate.net/publication/243766412_Static_Detection_of_Malicious_Code_in_Executable_Programs
[13] Bhojani, N. (2014). Malware Analysis. doi: 10.13140/2.1.4750.6889.
[14] Cybersecurity Ventures. (2017). *2017 Cybercrime Report Cybercrime damages will cost the world $6 trillion annually by 2021*. Herjavec Group. Diakses pada tanggal 12 Juni 2019, pada https://cybersecurityventures.com/2015-wp/wp-content/uploads/2017/10/2017-Cybercrime-Report.pdf
[15] Islam, R., Tian, R., Batten, L., & Versteeg, S. (2010). Classification of Malware Based on String and Function Feature Selection. 2010 Second Cybercrime And Trustworthy Computing Workshop. doi: 10.1109/ctc.2010.11
[16] Kaushal, K., Swadas, P., & Prajapati, N. (2012). Metamorphic Malware Detection Using Statistical Analysis. *International Journal Of Soft Computing And Engineering*, *2*(3), 49-52. Diakses pada tanggal 8 Juni 2019, pada dari https://pdfs.semanticscholar.org/459f/03160713981e7a0e56fc8b99cba3066332fd.pdf
[17] K. A, M. (2018). *Learning Malware Analysis*. Birmingham, UK: Packt Publishing.
[18] Marak, V. (2015). *Windows Malware Analysis Essentials*. Birmingham, UK: Packt Publishing.
[19] Namanya, A., Disso, J., & Awan, I. (2017). Evaluation of Automated Static Analysis Tools For Malware Detection in Portable Executable Files. Diakses pada tanggal 10 Mei 2019, pada https://www.researchgate.net/publication/319719981_Evaluation_of_automated_static_analysis_tools_for_malware_detection_in_Portable_Executable_files
[20] Pektas, A., Acarman, T., Falcone, Y., & Fernandez, J. (2015). Runtime-behavior based malware classification using online machine learning. *2015 World Congress On Internet Security (Worldcis)*. doi: 10.1109/worldcis.2015.7359437
[21] Qin, J., Yan, H., Si, Q., & Yan, F. (2010). A Trojan Horse Detection Technology Based On Behavior Analysis. *Wireless Communications Networking And Mobile Computing*. doi: 10.1109/WICOM.2010.5601305
[22] Ranveer, S., & Hiray, S. (2015). Comparative Analysis of Feature Extraction Methods of Malware Detection. *International Journal Of Computer Applications*, *120*(5), 1-7. doi: 10.5120/21220-3960
[23] Zalavadiya, N., & Sharma, P. (2017). A Methodology of Malware Analysis, Tools and Technique for windows platform – RAT Analysis. International Journal Of Innovative Research In Computer And Communication Engineering, 5(3). Diakses dari http://www.ijircce.com/upload/2017/march/253_A%20Methodology.pdf