ANALISIS DETEKSI *MALICIOUS ACTIVITY* MENGGUNAKAN METODE ANALISIS *MALWARE* DINAMIS BERBASIS ANOMALI

DETECTION ANALYSIS OF MALICIOUS ACTIVITY USING ANOMALY-BASED DYNAMIC MALWARE ANALYSIS METHOD

Damar Auriga Daniswara[1], Avon Budiyono, S.T., M.T[2], Ahmad Almaarif, S.Kom, M.T[3]

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

Abstrak

Perkembangan teknologi yang semakin meningkat memberikan peluang terjadinya *cyber-crime* dengan memanfaatkan penggunaan internet. Salah satu kejahatan *cyber* yang dapat merugikan individu ataupun perusahaan adalah serangan dari *malware*. *Malware* merupakan *software* yang sengaja dirancang untuk pencurian data, manipulasi data serta mendapatkan akses penuh terhadap *host* yang sudah terinfeksi. Menurut situs av-test.org serangan *malware* semakin meningkat hingga 902.82 juta *malware* pada tahun 2019 dan menyebabkan kerugian hingga 11.7 juta dollar Amerika. Berdasarkan masalah tersebut, maka diperlukan analisis deteksi *malware* yang bertujuan untuk melihat *malicious activity* yang dilakukan oleh *malware* ketika *malware* dieksekusi pada komputer. Analisis dilakukan menggunakan metode analisis *malware* dinamis berbasis anomali yaitu dengan menjalankan sampel *malware* pada suatu *environment* yang sudah dirancang pada Virtual Machine. Anomali merupakan sebuah pola yang tidak sesuai dengan pola normal suatu program. Penelitian ini dilakukan dengan menguji 10 sampel *malware* yang diunduh secara random dan dianalisis menggunakan tiga *tools* yang berbeda untuk mendeteksi sampel *malware*. Anomali dari *malware* dilihat dari aktivitas yang mencurigakan, *registry* yang di akses dan API yang dipanggil oleh sampel *malware*. Hasil pengujian menggunakan *tools* akan dicocokkan dengan *malicious activity data set* yang berupa kumpulan API yang dipanggil oleh sampel *malware,* sehingga diperoleh hasil 40% dari total 10 sampel yang diuji terbukti merupakan *malware.*

Kata kunci : *malware,* analisis malware, *cyber-crime,* analisis malware dinamis, anomali

Abstract

With the advance in technology and broader use of internet, comes a greater threat in cyber-crime. One of such that can potentially harm individuals or companies is malware attack. Malware is a software that is thoroughly designed for data theft, data manipulation and full access control of the infected hosts. According to av-test.org, malware attacks have increased to 902.82 million malwares in 2019 and caused losses of up to 11.7 million US dollars. One of the ways to tackle these problems is to design a malware detection analysis which aims to detect malicious activity carried out by malware during the time it is executed. The analysis was carried out using anomaly-based dynamic malware analysis method, namely by running malware samples in an environment that was designed in Virtual Machine. Anomaly is a pattern that is not in accordance with the normal pattern of a program. This study was conducted by testing 10 randomly downloaded malware samples which were then analyzed using three different malware-detection tools. Anomalies from malware are tracked from suspicious activity, registry accessed and API calls by malware samples. The test results are then matched with malicious activity data sets in the form of a collection of APIs called by malware samples, which shows that 40% of the 10 samples tested proven to be malware.

Keywords: *malware, malware analysis, cyber-crime, dynamic malware analysis, anomaly*

1. Pendahuluan

Pada era globalisasi yang sudah sangat maju ini ada banyak ancaman terkait kemajuan teknologi yang sudah pesat ini, salah satu ancaman yang kerap terjadi adalah *cyber-crime*, atau biasa dikenal dengan kejahatan *cyber*. *Cyber-crime* ada banyak jenisnya, namun diantara banyak jenis ada satu yang menjadi ancaman besar karena dampak yang cukup besar bagi individual maupun bagi institusi yaitu *malicious software* atau biasa dikenal dengan *malware. Malware* merupakan *software* atau program perangkat lunak komputer yang digunakan untuk melakukan manipulasi data dan pencurian data tanpa sepengetahuan

pemiliknya[1]. Tujuan dari pembuatan *malware* itu sendiri adalah untuk mendapatkan akses atau kontrol secara penuh terhadap *device* pemiliknya[1]. Serangan *malware* semakin meningkat 5 tahun terakhir ini menurut situs av-test.org pada tahun 2015 jumlah *malware* melingkupi 470.01 juta dan meningkat sampai berjumlah 902.82 juta *malware* pada tahun 2019, dan menurut [14] *malware* dapat merugikan perusahaan hingga 11.7 juta dollar amerika, untuk itu *malware* harus segera ditangani sebelum berdampak lebih besar lagi terhadap individu maupun perusahaan.

Penyebaran *malware* ada banyak bentuknya mulai dari *social-engineering attack* yang memanfaatkan kondisi psikis dari suatu individu, *file download fraud* guna menipu *user* untuk men-*download malware* secara tidak sengaja, dan *email phising* menipu *user* dengan *spam email* yang menarik perhatian *user*. Dampak yang terjadi pada individual maupun institusi adalah gangguan terhadap layanan yang sedang berjalan bahkan dapat menyebabkan penurunan hingga hilangnya *profit* yang didapatkan, karna itu *malware* merupakan ancaman yang sangat berpengaruh terhadap individual dan juga organisasi[1].

Banyak cara penanganan *malware* yang dapat dilakukan namun salah satu cara mengatasinya adalah dengan melakukan *malware analysis*, cara pencegahan ini merupakan cara yang cukup handal dalam mengatasi *malware* karna dapat membaca pola serangan yang tepat jika analisinya juga tepat. Analisis *malware* mempermudah untuk mengenali tanda-tanda suatu *malware* akan melancarkan sebuah serangan, ditambah lagi dengan metode dinamis berbasis *anomaly* yang dapat mempermudah klasifikasi dan pengkategorian *malware*, selain itu keuntungan yang juga didapatkan ketika implementasi metode ini adalah deteksi suatu pola akan terasa lebih mudah dan lebih cepat deteksi nya. Metode anomali banyak digunakan sebagai penaganan yaitu dengan melakukan langkah preventif membaca pola serangan, sehingga serangan dapat diprediksi datangnya dari mana dan seperti apa [2].

Analisis *malware* dengan metode dinamik mengamati perilaku *malware* selama proses eksekusi dan infeksi malware tersebut, guna menentukan perilaku sistem, panggilan fungsi, fungsi parameter yang ada dan juga arus informasi [13]. Pada analisis dinamik menggunakan *environment* buatan seperti *virtual machine* dan juga *sandbox*. Dalam konsep identifikasi *malware* yang belum diketahui, analisis dinamik lebih baik dari pada analisis secara statis, namun analisis secara statis lebih cepat mengidentifikasi *malware* yang sudah diketahui [4].

2. Dasar Teori

2.1 Definisi *Malicious Activity (Malware)*

Malware merupakan sebuah perangkat lunak yang dirancang untuk merusak suatu sistem dan memanipulasi data tanpa sepengetahuan pemilik dari *device* yang terinfeksi, *malware* dapat berbentuk *script*, *active content* dan *binary*[5]. *Malware* pada umumnya digunakan oleh *hacker* atau kriminal lainnya untuk merusak jalannya suatu sistem operasi, mencuri *data* privat seseorang, melakukan *bypass* terhadap *access control* dan merusak sistem pada *host*. [6].

2.2 Jenis-Jenis Malware

Malware dapat dikategorikan menjadi beberapa jenis dan *malware* mempunyai banya variasi juga, antara lain adalah [15] :

a. Adware

Adware mempunya kepanjangan *advertising-supported software*, adalah jenis *malware* yang secara otomatis mengirimkan iklan yang menarik perhatian kepada *user*, seperti contohnya *pop-up* iklan pada *website* tertentu, kadang *adware* juga menawarkan aplikasi yang gratis namun saat di unduh berisikan *malware* lain atau *adware* itu sendiri.

b. Botnet

Bot atau botnet adalah sebuah *software* yang diprogram dan dirancang untung melakukan sebuah aktivitas atau operasi secara otomatis, seperti misalnya pada serangan jaringan *DdoS* (*Distributed Denial of Service*) *botnet* digunakan untuk melakukan *ping* kepada *victim* yang sudah di tentukan, *botnet* dapat beroperasi secara otomatis dan dapat juga di kontrol oleh pihak ketiga.

c. Bug

Bug merupakan sebuah kejanggalan atau kesalahan pada suatu program yang biasanya tercipta karna kesalahan pembuatnya sendiri dalam memasukan barisan kode saat pembuatan sebuah program, namun bug dapat menjadi alasan kenapa sebuah program berjalan tidak sesuai yang diinginkan bahkan saat tidak teridentifkasi pada jangka waktu yang lama juga dapat menyebabkan kelemahan atau celah yang dapat di eksploitasi.

d. Ransomware

Ransomware adalah jenis *malware* yang saat diaktifkan akan mengunci sistem operasi dan data *user* sebagai tawanan dan data tidak akan diberikan kembali sampai tebusan yang sudah ditentukan sudah dibayar. *Malware* ini membatasi hak akses *user* dan mengenkripsi *file* yang ada pada *hard drive*, dan menampilkan pesan yang bertujuan untuk memaksa user membayar tebusan.

e. Rootkit
Rootkit adalah sebuah *malware* yang memungkinkan *hacker* melakukan *remote access control* tanpa terdeteksi *user* aslinya. Ketika *rootkit* diaktifkan akan memungkinkan *hacker* mengeksekusi *file*, mengakses dan mencuri informasi secara diam-diam, karna rootkit sulit untuk terdeteksi maka rootkit sangatlah berbahaya.

f. Spyware
Spyware adalah jenis *malware* yang berfungsi untuk memata-matai aktivitas *user* tanpa *user* itu sendiri mengetahui, *malware* ini bertujuan untuk mengetahui aktivitas *user*, dan memanen informasi terkait finansial seperti misalnya transaksi bank yang sensitif untuk diketahui.

g. *Trojan Horse*
Trojan horse atau biasa disebut *trojan* adalah sebuah *malware* yang mempunyai metode menyembunyikan *malware* nya di dalam file biasa guna menipu *user*, ketika *file* telah ter-*install* maka *malware* yang berada dalam *file* tersebut akan diaktifkan, *malware* ini dapat memberi akses *remote* pada hacker dan memungkinkan pencurian informasi bahkan sampai pencurian uang elektronik

h. Virus
Virus adalah *malware* yang dapat menduplikat dirinya sendiri dan menyebar ke komputer lain ketika telah diaktifkan, *virus* dapat menyebar ke komputer lain dengan menempelkan dirinya ke beberapa program tertentu yang nantinya diaktifkan di komputer lain. *Virus* dapat digunakan untuk mencuri informasi, mencuri uang elektronik, membuat botnet, merusak host dan juga jaringan.

i. Worm
Worm adalah satu diantara *malware* yang paling umum, *worm* menyebar melalui jaringan komputer dengan melakukan eksploitasi terhadap celah yang ada pada sistem operasi. *Worm* pada umumnya menyebabkan rusak nya *host* dengan mengkonsumsi banyak *bandwidth* dan mengisi *web server* sampai *overload*.

j. Launcher
Launcher adalah *malware* yang berfungsi untuk menjalankan *malware* lainnya. Pada umumnya, *launcher* menggunakan teknik non tradisional untuk menjalankan *malware* yang lain untuk memastikan akses tersembunyi atau akses yang lebih baik pada sistem.

k. Information-Stealing Malware
Information-stealing malware adalah *malware* yang mencuri informasi dari komputer korban dan dengan mengirim informasi tersebut langsung kepada *hacker*. Contohnya adalah *sniffer* yang berfungsi untuk melakukan *sniffing* jaringan dan menangkap semua paket yang berlalu-lintas dan *keylogger* yang berfungsi untuk merekam aktivitas *keyboard*. *Malware* ini memiliki tujuan untuk mendapat akses akun *online* seperti *email* dan sistem bank.

l. Downloader
Downloader adalah sebuah *malware* yang hanya beroperasi untuk melakukan *download* tehadap *malware* lainnya. *Downloader* biasanya di-*install* oleh *hacker* untuk mendapatkan *akses* pada suatu sistem.

m. Backdoor
Backdoor adalah sebuah *malware* yang dapat menginstalasi sendiri *script*nya secara otomatis pada suatu komputer. Tujuannya adalah untuk memberi akses pada *hacker* atau pembuat *malware*. *Backdoor* memungkinkan *hacker* untuk melakukan koneksi pada komputer dengan sedikit bahkan tidak ada otentikasi dan mengeksekusi perintah pada suatu sistem komputer.

n. Spam-Sending Malware
Spam-sending malware merupakan sebuah *malware* yang menginfeksi komputer lalu menggunakan komputer tersebut untuk mengirim *spam*.

o. Scareware
Scareware adalah *malware* yang didesain untuk mengintimidasi *user* dan memaksa korban untuk membeli *scareware*. *Scareware* memiliki antarmuka yang mirip dengan *antivirus*. Contohnya adalah menginformasikan *user* bahwa ada virus yang menyerang komputer *user* dan satu-satunya cara untuk melakukan penanggulangan hanya dengan membeli *scareware* tersebut.

2.3 *Malicious Activity pada Malware*

Berikut merupakan rincian proses ketika *malware* telah berhasil diaktifkan, dan apa saja yang aktivitas-aktivitas yang tidak sesuai pada *malware*:

a. *Process Hollowing*
Process hollowing merupakan aktivitas yang diakibatkan oleh *malware*. Saat *malware* melakukan injeksi proses pada memori dan menggantikan dengan kode dari *malware* itu sendiri yang disimpan pada suatu *container* [7].

b. *Create Remote Thread*
Create Remote Thread merupakan aktivitas dimana suatu *malware* melakukan injeksi DLL dengan menggunakan API *call CreateRemoteThread* [8].
c. *Enumerating All Processes*
Enumerating All Processes adalah aktivitas dimana suatu *malware* melakukan perhitungan dan mencari tahu proses apa saja yang berjalan pada suatu komputer yang terinfeksi [8].
d. *Drop Files from PE Resource Section*
Drop Files from PE Resource Section ini adalah aktivitas dari suatu *malware* yang melakukan *drop file* pada komputer yang terinfeksi [8].
e. *IAT Hooking*
IAT Hooking adalah singkatan dari *import address table hooking* merupakan aktivitas yang digunakan untuk mencari informasi mengenai fungsi yang berhubungan dengan *network* yang digunakan oleh suatu aplikasi. IAT *hooking* digunakan *malware* untuk mengambil fungsi data secara bersamaan dari suatu aplikasi menggunakan API *call GetModuleHandle* [9].
f. *Delete Itself*
Delete Itself merupakan *malicious activity* dimana suatu *malware* dapat menghapus diri sendiri setelah menjalankan suatu *script* tertentu [10].
g. *Download & Execute PE Files*
Download & Execute PE Files merupakan *malicious activity* dimana suatu *malware* melakukan pengunduhan *external file* untuk selanjutnya dijalankan pada komputer yang terinfeksi [8].
h. *Bind TCP Port*
Bind TCP Port adalah aktivitas yang dilakukan oleh *malware* untuk membuka suatu *port* tertentu yang dapat diakses dari pihak yang mengetahui. Pada umumnya, *port* yang dibuka merupakan *backdoor* untuk mengakses komputer yang terinfeksi [11].
i. *Capture Network*
Capture Network merupakan aktivitas dimana suatu *malware* melakukan *sniffing* pada komputer yang terinfeksi. *Sniffing* digunakan guna mencari informasi mengenai aktivitas jaringan pada jaringan yang terinfeksi [11].

2.4 *Malware Analysis*

Malware analysis adalah sebuah aktivitas untuk membedah atau menganalisis[5]. *Malware analysis* mengidentifikasi bagaimana cara *malware* tersebut bekerja, fungsionalitas *malware* dan bagaimana cara pendeteksian serta pencegahannya yang paling efektif terhadap suatu *malware* [5]. Untuk melakukan *malware analysis* terdapat dua metode proses analisis yaitu *static analysis* dan *dynamic analysis* [4].

2.4.1 *Dynamic Malware Analysis*

Analisis *malware* dengan metode dinamik mengamati perilaku *malware* selama proses eksekusi dan infeksi malware tersebut, guna menentukan perilaku sistem, panggilan fungsi, fungsi parameter yang ada dan juga arus informasi[13]. Pada analisis dinamik menggunakan *environment* buatan seperti *virtual machine* dan juga *sandbox*. Dalam konsep identifikasi *malware* yang belum diketahui *dynamic analysis* merupakan metode terbaik [4].

2.5 *Metode Anomali*

Anomaly adalah sebuah pola dari kumpulan data yang tidak memiliki pola yang normal yaitu berbeda dengan pola yang lainnya. Pola yang berbeda ini yang disebut dengan anomali[12]. Metode anomali dapat menemukan jenis *malware* baru karena *malware* tersebut tidak berjalan sesuai *malware* yang sudah diketahui sehingga terdeteksi sebagai hal yang unik atau berbeda dengan *malware* yang lainnya. Teknik deteksi anomali menggunakan pola normal sebagai acuan utama, acuan tersebut digunakan untuk mendeteksi apakah sebuah program merupakan *malware* atau jenis *malware* yang baru. Jika ada pola atau perilaku yang melenceng dari acuan tersebut bisa dipastikan kalau program tersebut merupakan *malicious.*

2.6 Alasan Penggunaan Metode

Pada umumnya analisis *malware* terbagi menjadi 2 cara yaitu secara dinamis dan secara statis, metode dinamis merupakan metode yang melakukan eksekusi sampel *malware* pada *virtual machine* sementara metode statis membongkar *source code* sampel *malware* tersebut [16]. Namun diantara kedua metode tersebut analisis *malware* dinamis merupakan metode yang paling efektif dilihat dari API yang dipanggil, aktivitas, dan perubahan *registry* oleh sampel *malware*[17]. Dengan dilandasi hal itu maka penelitian ini akan menggunakan metode analisis dinamis sebagai metode utama dalam melakukan analisis serta deteksi *malware*.

3. Pengujian dan Analisis

3.1. Pengujian

1. Pengujian Menggunakan Process Monitor

Process Monitor adalah *tools* analisis *malware* yang di *install* pada OS Windows, data yang diambil dari Process Monitor adalah *registry*.

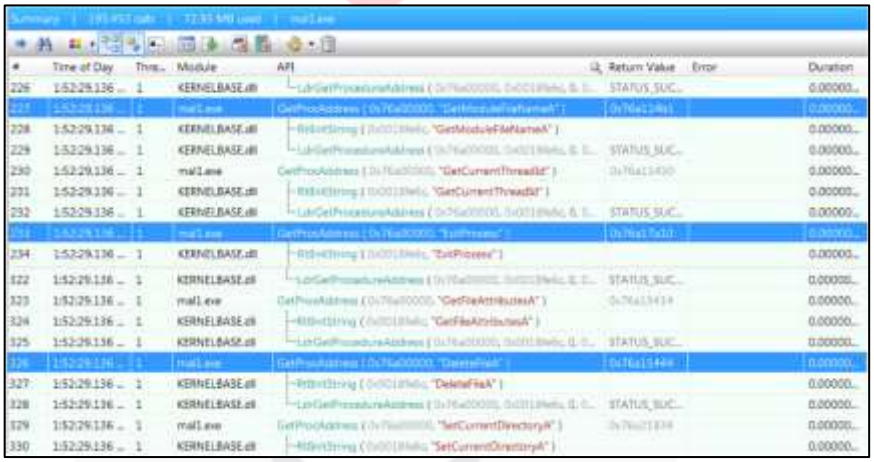
Time of Day	Process Name	PID	Operation	Path
12:00:02.5794078 AM	mal1.exe	3548	CreateFile	C:\Users\Grandine\AppData\Local\Temp\xsvinmat.exe
12:00:02.5794582 AM	mal1.exe	3548	QueryAllInformationFile	C:\Users\Grandine\AppData\Local\Temp\xsvinmat.exe
12:00:02.5794890 AM	mal1.exe	3548	CreateFileMapping	C:\Users\Grandine\AppData\Local\Temp\xsvinmat.exe
12:00:02.5795179 AM	mal1.exe	3548	QueryStandardInformationFile	C:\Users\Grandine\AppData\Local\Temp\xsvinmat.exe
12:00:02.5813984 AM	mal1.exe	3548	CloseFile	C:\Users\Grandine\AppData\Local\Temp\xsvinmat.exe
12:00:02.5814449 AM	mal1.exe	3548	Process Create	C:\Users\Grandine\AppData\Local\Temp\xsvinmat.exe
12:00:02.5814523 AM	xsvinmat.exe	3432	Process Start	
12:00:02.5814576 AM	xsvinmat.exe	3432	Thread Create	

Gambar 1 Hasil Process Monitor

Gambar 1 merupakan hasil dari pengujian salah satu *malware*. Dapat diliat aktivitas *malware* yang membuat file baru dengan operation CreateFile pada direktori C:\Users\Grandine\AppData\Local\Temp\ hal ini tidak wajar karna sampel *malware* membuat file tanpa ijin dari *user*.

2. Pengujian Menggunakan API Monitor

API Monitor adalah *tools* analisis *malware* yang di *install* pada OS Windows, data yang diambil dari API Monitor adalah API *Calls* nya.

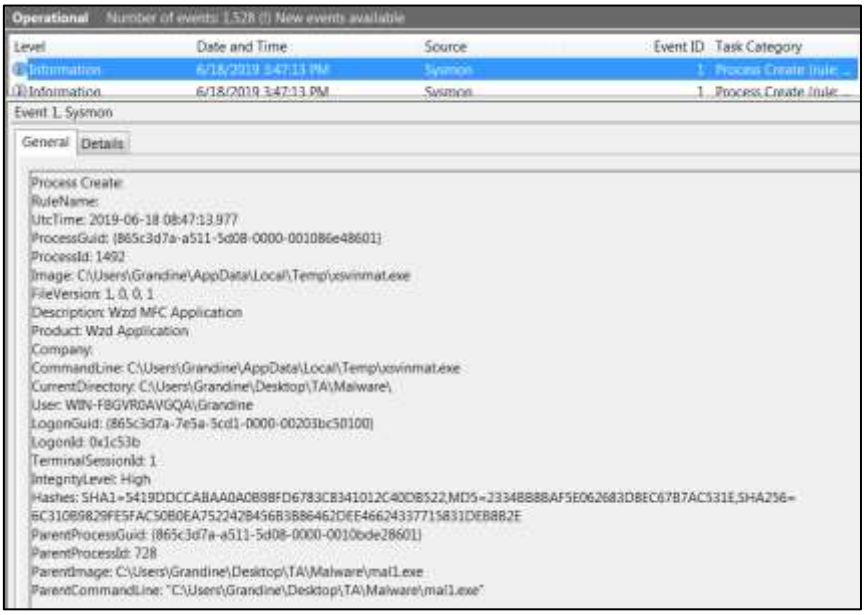


Gambar 2 Hasil API Monitor

Gambar 2 merupakan hasil dari tools API Monitor. Dapat dilihat bahwa terdapat beberapa API yang mencurigakan seperti GetModuleFileNameA, ExitProcess, DeleteFileA jika dicocokan pada *data set* makan API tersebut adalah rincian API untuk *malicious activity delete itself.*

3. Pengujian Menggunakan System Process

System Monitor adalah *tools* analisis *malware* yang di *install* pada OS Windows, data yang diambil adalah aktivitas dari *malware*.



Gambar 3 Hasil System Monitor

Gambar 3 merupakan hasil dari *tools* System monitor, dapat dilihat pada data ParentImage saat sampel *malware* di eksekusi, sampel *malware* membuat *file* baru pada direktori C:\Users\ Grandine\AppData\Local\Temp\ dengan nama xsvinmat.exe hal ini tergolong anomali karna pembuatan *file* tidak dikehendaki oleh *user*.

3.2. Hasil Analisis Sampel Malware

1. Sampel Malware 1

Dari hasil pengujian menunjukan aktivitas yang tergolong anomali, karna sampel *malware* mal1.exe mengakses *registry* HKLM\System\CurrentControlSet\Control\Terminal Server *registry* ini diakses oleh *malware* untuk mengaktifkan Remote Desktop Protocol (RDP), lalu sampel *malware* mal1.exe juga membuat *file* baru pada direktori C:\Users\Grandine\AppData\Local\Temp\ dengan nama xsvinmat.exe, setelah membuat *file* baru dan mengeksekusi *file* tersebut sampel *malware* mal1.exe melakukan koneksi ke internet melalui *port* 49174 ke IP *Address* 195.128.124.140:1356.

Berdasarkan tabel malicious activity datasetsampel *malware* mal1.exe melakukan beberapa aktivitas yang tergolong anomali karna aktivitas yang dilakukan merupakan *malicious activity*, sampel *malware* mal1.exe di identifikasi melakukan aktivitas *Create Remote Thread*, *Drop Files from* PE *Resource Section*, IAT *Hooking*, dan *Delete Itself*. Dengan itu sampel *malware* mal2.exe dinyatakan sebagai *malware.*

2. Sampel Malware 2

Hasil pengujian menunjukan aktivitas dari sampel *malware* mal2.exe yang dapat disebut dengan anomali dikarnakan sampel *malware* mal2.exe mengeksekusi cmd.exe atau Command Prompt secara otomatis hal ini dilakukan untuk melakukan eksekusi terhadap perintah yang sudah ditentukan. Sampel *malware* mal2.exe juga membuat *file* baru pada direktori C:\Users\Grandine\AppData\ Roaming\ATI_Subsystem dengan nama amdocl_as32.exe lalu mengeksekusinya, setelah itu sampel *malware* mal2.exe juga mengakses *registry* HKLM\System\CurrentControlSet\Control\ ComputerName\ActiveComputerName hal ini dilakukan untuk membaca nama komputer atau melihat nama *domain* yang ada pada suatu komputer, hal ini juga memungkinkan *malware* untuk merubah nama *domain* atau nama dari komputer yang terinfeksi.

Berdasarkan hasil pengujian sampel *malware* mal2.exe dinyatakan sebagai *malware* karena melakukan beberapa aktivitas yang tergolong *malicious activity*, hal ini dibuktikan karena saat dilakukannya analisis terdapat API yang memenuhi kriteria *malicious activity*, aktivitas tersebut ialah *Drop Files from* PE *Resource Section* dan *Delete Itself*.

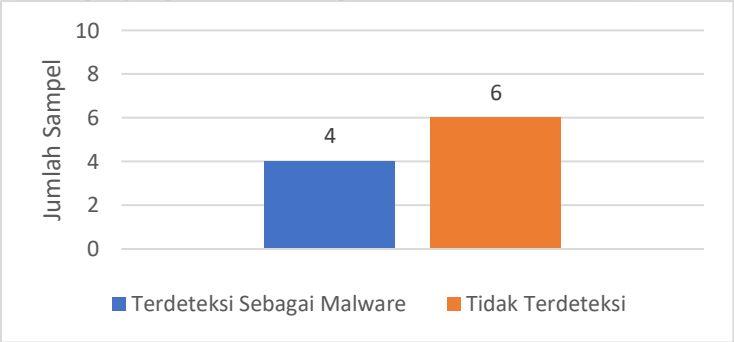
3. Sampel Malware 3

Hasil pengujian menunjukan aktivitas dari sampel *malware* mal3.exe yang dapat disebut anomaly dikarnakan beberapa aktivitas yang berjalan tidak dikehendaki oleh *user*, sampel *malware* mal3.exe membuat *file* pada direktori C:\Users\Grandine\AppData\Roaming\x86_microsoft-windows-n..meworkapi.resources dengan nama NlsLexicons0049.exe setelah diaktifkan *file* baru itu dengan otomatis mengeksekusi rundll32.dll yang berfungsi untuk sebagai *backdoor*, *backdoor* ini memungkinkan *malware* untuk mendapat akses ke komputer dan melakukan aktifitas *hacking* lainnya. Sampel *malware* mal3.exe juga mengakses *registry* HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid hal ini dilakukan *malware* untuk melihat *Globally Uniques Identifier* (GUID) yang digunakan untuk mengidentifkasi GUID dari komputer yang bersifat unik.

Hasil pengujian sampel *malware* mal3.exe terbukti sebagai *malware* karna melakukan beberapa *malicious activity*, hal ini dibuktikan dengan hasil analisis pada tabel diatas. *Malicious activity* yang dilakukan adalah *Drop Files from* PE *Resource Section* dan *Delete Itself*.

3.3. Analisis Hasil Keseluruhan *Tools* Deteksi *Malware*

Setelah dilakukan analisis hasil *dari* tools deteksi *malware* yang dilakukan dapat diketahui bahwa beberapa dari sampel yang di unduh merupakan *malware*.



Gambar 4 Grafik Hasil

Berdasarkan grafik pada Gambar 4 dapat dilihat terdapat total 10 sampel program yang diunduh dan dijalankan pada *environment* untuk mendapatkan hasil deteksi dan analisis nya. Dari 10 sampel ada 4 yang terindikasi sebagai *malware*, dan 6 sampel lainnya tidak terindikasi karna kurang nya informasi yang didapatkan, beberapa sampel tidak terdeteksi karena kurangnya informasi. Pada *tools* Process Monitor sampel yang tidak terindikasi karena tidak ada aktivitas yang mencurigakan maupun *registry* yang juga mencurigakan, pada *tools* API Monitor sampel tidak terindikasi karena API yang dipanggil tidak memenuhi API *malicious activity data set* jadi belum bisa dinyatakan melakukan *malicious activity*, dan pada *tools* System Monitor juga tidak terdeteksi dari *event* yang dihasilkan oleh log System Monitor.

4. Kesimpulan

Berikut adalah kesimpulan dari permasalahan berdasarkan pengujian dan analissi yang dilakukan:

1. Dalam melakukan analisis *malware* dengan metode analisis *malware* berbasis anomali dimulai dengan merancang *environment* yang terisolasi pada virtual machine hal ini dilakukan guna memproteksi OS utama agar tidak terinfeksi *malware* selain itu juga dapat mengembalikan *environment* ke *clean state* sehingga tidak mengganggu aktivitas analisis satu sampel *malware* dengan sampel *malware* lainnya. Setelah itu melakukan eksekusi terhadap sampel *malware* yang telah diunduh, setelah dijalankan maka *tools* yang sudah di instalasi pada *environment* akan merekan seluruh aktivitas sampel *malware* secara *real-time* setelah itu dapat dianalisis dari aktivitas nya. Dengan menggunakan tiga *tools* yang berbeda maka parameter yang diperhatikan juga berbeda pada *tools* Process Monitor yang diperhatikan adalah *operation* dan *registry* nya, pada API Monitor yang diperhatikan adalan API nya dan pada System Monitor yang diperhatikan adalah *event* dari sampel *malware* yang dijalankan. Jika ada aktivitas yang tidak wajar seperti sampel *malware* membuat *file* baru dan menghapus *file* tanpa *user* kehendaki maka sampel *malware* tersebut dapat dipastikan adalah *malware*. Setelah dilakukan pengujian dan analisis, dari 10 sampel yang diunduh secara *random* 40% diantaranya adalah *malware* yang aktif dan dapat menginfeksi komputer *user*. Sampel *malware* melakukan aktivitas *spawn process*. Proses ini berakibat fatal karena program yang terus membuat dan mengeksekusi program baru dan berdampak pada pengurangan kinerja sistem bahkan dapat mengakibatkan komputer *user* mati total.

2. Setelah dilakukannya pengujian dan memperoleh hasil analisis cara deteksi *malware* menggunakan *malicous activity* membutuhkan *data set* yang berguna untuk mengkategorikan aktivitas yang tergolong *malicious* atau anomali. Pada penelitian ini yang diperhatikan adalah proses yang berjalan, akses *registry* dan API *calls*. Lalu hasilnya bisa dibandingkan dengan *malicious activity data set* jika memenuhi kriterianya maka sampel *malware* bisa dibuktikan sebagai *malware*.
3. Untuk membuktikan sebuah *software* merupakan sebuah *malware* adalah dengan memperhatikan aktivitas dari *malware*, *registry* yang diakses, dan dari API *calls* nya, jika terjadi aktivitas yang tergolong anomali seperti melakukan *spawn process*, membuat *file* baru, mengahpus *file* secara otomatis, dan mengeksekusi sebuah aplikasi secara otomatis maka *software* tersebut bisa dipastikan sebuah *malware*.

Daftar Pustaka:

[1] Zeltser, L. (2016). OUCH! What is Malware?. *The SANS Institute*. [online] Available at: https://www.sans.org/sites/default/files/2017-12/OUCH-201603_en_0.pdf [Diakses 21 Jun. 2019].

[2] Bromiley, M. (2019). Keys to Effective Anomaly Detection. *SANS Institute*. [online] Available at: https://www.sans.org/reading-room/whitepapers/hackers/paper/37362 [Diakses 16 Mar. 2019].

[3] Uppal, D., Mehra, V., & Verma, V. (2014). International Journal on Computational Science & Applications. Basic Survey on Malware Analysis, Tools and Techniques,4(1), 103-112. Diakses pada 5 April, 2019, diambil dari https://www.semanticscholar.org/paper/Basic-survey-on-Malware-Analysis,-Tools-and-Uppal-Mehra/b25479a230816e8566df0937d9ff9265754f826d

[4] Aslan, O., & Samet, R. (2017). Investigation of Possibilities to Detect Malware Using Existing Tools. 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA). Diakses pada 10 Juni, 2019, diambil dari https://ieeexplore.ieee.org/document/8308437.

[5] Sikorski, M., & Honig, A. (2012). Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software. Diakses pada 25 September, 2018, diambil dari https://repo.zenk-security.com/Virus-Infections-Detections-Preventions/Practical_Malware_Analysis.pdf.

[6] Nate, L. (2012, Oktober 15). Common Malware Types: Cybersecurity 101. Diakses pada 7 Desember 2018, dari http://blog.veracode.com/2012/10/commonmalware-types-cybersecurity-101/

[7] Fortuna, Andrea. (2017, Oktober 9). *Understanding Process Hollowing*. Diakses pada 9 Oktober 2018, dari https://www.andreafortuna.org/cybersecurity/understanding-process-hollowing/.

[8] Microsoft. (2018) Windows API *Index*. Diakses pada 9 Oktober 2018, dari https://docs.microsoft.com/en-us/windows/desktop/apiindex/windows-api-list

[9] Tigzy. (2014, Oktober 15). *Import Address Table* (IAT). Diakses pada 9 Oktober 2018, dari https://www.adlice.com/userland-rootkits-part-1-iat-hooks/.

[10] Greenberg, Adam. (2015, Maret 11). *Self-deleteing Malware Targets Home Routers to Gather Information.* Diakses pada 9 Oktober 2018, dari https://www.scmagazine.com/malware-that-connects-to-home-routers-deletesitself-without-a-trace/article/536538/.

[11] Anupama, Bindu. (2007, November 6). *Know Your TCP System call sequences.* Diakses pada 9 Oktober 2018, dari https://www.ibm.com/developerworks/aix/library/au-tcpsystemcalls/index.html

[12] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection. *ACM Computing Surveys*, *41*(3), 1–58. https://doi.org/10.1145/1541880.1541882

[13] Uppal, D., Mehra, V. and Verma, V. (2014). Basic survey on Malware Analysis, Tools and Techniques. *International Journal on Computational Science & Applications*, 4(1), pp.103-112.

[14] Inside Out Security. (2019). *60 Must-Know Cybersecurity Statistics for 2019*. [online] Available at: https://www.varonis.com/blog/cybersecurity-statistics/ [Accessed 7 Oct. 2018].

[15] Nate, L. (2012, Oktober 15). Common Malware Types: Cybersecurity 101. Diakses pada 7 Desember 2018, dari http://blog.veracode.com/2012/10/commonmalware-types-cybersecurity-101/

[16] Cahyanto, T., Wahanggara, V., & Ramadana, D. (2017). Analisis dan Deteksi Malware Menggunakan Metode Malware Analisis Dinamis dan Malware Analisis Statis. *Jurnal Sistem & Teknologi Informasi Indonesia*, *2*(1). doi: 10.32528/justindo.v2i1.1037

[17] Damodaran, A., Troia, F., Visaggio, C., Austin, T., & Stamp, M. (2015). A comparison of static, dynamic, and hybrid analysis for malware detection. *Journal of Computer Virology and Hacking Techniques*, *13*(1), 1-12. doi: 10.1007/s11416-015-0261-z