ANALISIS *MALWARE* DENGAN TEKNIK *STATIC ANALYSIS*

Irman Hariman[1], Azhar Syams[2]

Email : irmanhariman@gmail.com[1]
de2n_iir@yahoo.com[2]

Abstrak

Berbagai masalah yang dapat menyebabkan suatu file rusak dan tidak dapat dipergunakan kembali adalah ketika file terindikasi dan terinfeksi malware, yang terdiri dari virus, trojan atau worm. Kesulitan dalam melakukan pendetekdian bahkan perbaikan oleh suatu antivirus kerap terjadi, hal ini dikarenakan perkembangan malware jauh didepan antivirus atau antimalware. Analisis malware yang dilakukan dalam penelitian ini akan memberikan pengetahuan yang sangat penting untuk mengetahui karakter dan perilaku malware yang menginfeksi suatu file, bahkan akibat yang ditimbulkannya.

Dalam jurnal ini akan dibahas analisis yang bersifat static karena penggunaan tools yang digunakan sebagai alat bantu untuk mengetahui*behavior* dari suatu malware. Pada akhirnya hasil analisis dapat dijadikan sebagai sebuah referensi dalam membangun algoritma antimalware yang kemudian dapat dibangun menjadi antimalware (antivirus, antitrojan, antiworm) dengan bahasa pemrograman yang sesuai.

Kata Kunci : Malware, Antivirus, Virus, Trojan dan Worm

1. Pendahuluan

Dalam perkembangan teknologi, khususnya teknologi informasi, kehandalan suatu aplikasi mutlak dibutuhkan untuk memenuhi kebutuhan dari pengguna itu sendiri. Sebuah sistem operasi yang telah terinfeksi *malware* (malicious software) dapat mengalami kerusakan dan lebih rentan terhadap gangguan penyusup, *virus* bahkan pencurian data-data penting.

Malware memiliki karakteristik dalam berbagai bentuk yaitu virus, worm dan trojan, namun demikian ketiga bentuk tersebut sangat mengganggu sistem komputer. Sesuai dengan namanya, *malicious software* komputer adalah program yang dapat menginfeksi suatu aplikasi atau dokumen yang telah tersimpan dalam media penyimpanan dan sistem yang dapat menjalankan sistem komputer.

Malicious software dapat berkembang biak menjadi banyak dan memanipulasi makin banyak pula suatu aplikasi dan data pada suatu komputer. Kemampuan manipulasi inilah yang menjadi sistem pertahanan dari suatu *malicious software* agar tidak terdeteksi oleh pengguna dan bahkan antivirus. Dalam kasus ini menggunakan contoh *malicious software* berjenis *WORM*. *Malicious software* berjenis *WORM* ini menginfeksi file sistem komputer dengan kemampuan menginfeksi folder dan menyembunyikan folder serta menduplikasi *malware* dengan icon tertentu.

Berdasarkan dari latar belakang diatas maka dapat dirumuskan masalah sebagai berikut :

a. Bagaimana menganalisis sebuah file yang diduga *malware worm*.
b. Bagaimana mencegah *malware* untuk berkembang biak dalam sistem komputer.
c. Membaca identitas dari file yang diduga *malware* dengan pembuatan aplikasi anti*malware*.
d. Bagaimana membuat sebuah anti*malware* sederhana dengan teknik *Heuristic Scan*, S*ignature Checksum*, dan *Executable Icon Binary*.

Dalam pembahasan analisis*malware* dan pembuatan anti*malware* terdapat beberapa pembatasan masalah, yaitu :

a. Analisis file yang diduga *malware* menggunakan metodologi static malware analysis.
b. Analisis file yang diduga *malware* yang telah diketahui varian *malware* tersebut.
c. Pembuatan anti*malware* hanya mampu mendeteksi dan menghapus *malware* yang signature CRC-nya sudah dimasukkan dalam kode program.

2. Metodologi

Jenis penelitian yang dilakukan selama proses analisis ini adalah penelitian kuantitatif, yaitu dengan mengumpulkan berbagai jenis malware yang ditemukan dalam kehidupan sehari-hari, kemudian ditentukan malware apa yang paling banyak ditemui dan menjumpai banyak keluhan dari narasumber. Penelitian lebih mendalam dilakukan dalam laboratorium pribadi. Malware yang menjadi objek penelitian analisis statik ini adalah worm berjenis W32.Generic. jenis ini didapat dengan membandingkan karakteristik dan hasil *scan* padaantivirus yang ada di pasaran.

Perbandingan ini dilakukan untuk menguji analisis antivirus dalam negeri dan luar negeri. W32.Generic mempunyai karakteristik infeksi file, registry, dan duplikasi worm pada setiap drive yang terdeteksi oleh worm ini. Secara garis besar jenis worm akan mempunyai karakteristik yang hampir sama, kecuali pada tingkat bahaya dari jenis worm tersebut.

3. Analisis Statik

Analisis statik adalah sebuah proses awal dalam penentuan file terduga *malware* sehingga didapatkan ketetapan analisis bahwa file yang terduga menjadi file yang mempunyai kode berbahaya atau disebut *malware*. Tahap yang dilakukan adalah sebagai berikut :

1. Dalam analisis statik, file yang dianalisis akan mengalami *dissasembly unpacking*, dan perbandingan file terduga dan file yang masih sehat. Penjelasan dibawah ini adalah proses yang akan berlangsung dalam analisis file terduga virus.

2. Analisis hexadecimal digunakan untuk mengetahui jenis dari file yang akan dianalisis. File yang dapat dieksekusi mempunyai kode hexadecimal 4D 5A, yaitu file Part Executable (PE). Pemakaian tools yang dipakai adalah IDAPro Free, nOllyDbg, dan HxD.

3. Analisis ini bertujuan untuk mendapatkan identitas dari sebuah file. CRC32 adalah pengenal sebuah file yang telah mengalami transmisi data dari asal file menuju tujuan file. CRC32 ini merupakan salah satu fungsi hash yang dikembangkan untuk mendeteksi kerusakan data dalam proses transmisi atau penyimpanan.

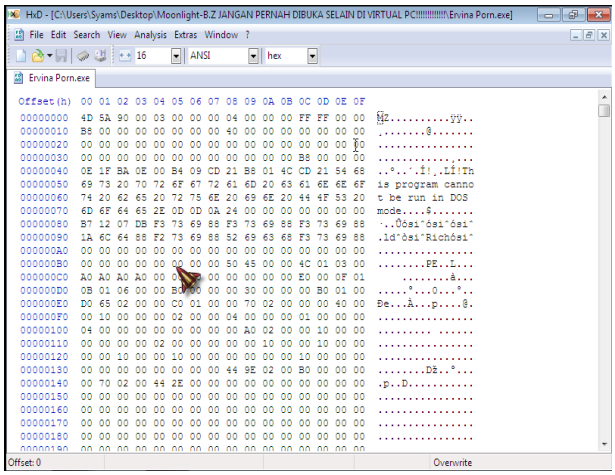
4. Pembahasan

Pada berbagai proses yang terjadi dalam analisis statik untuk menganalisis sebuah file yang terduga *malware* sehingga didapat kesimpulan akan bersih atau tidaknya file tersebut dari kode berbahaya yang dapat menyebabkan instabilitas sistem file pada komputer, mengalami proses yang lama dan membutuhkan kesabaran. Dokumentasi yang berkelanjutan diperlukan dalam analisis statik, untuk memantau perubahan dari *Header File, chekcksum,* dan mutasi file yang terinfeksi.

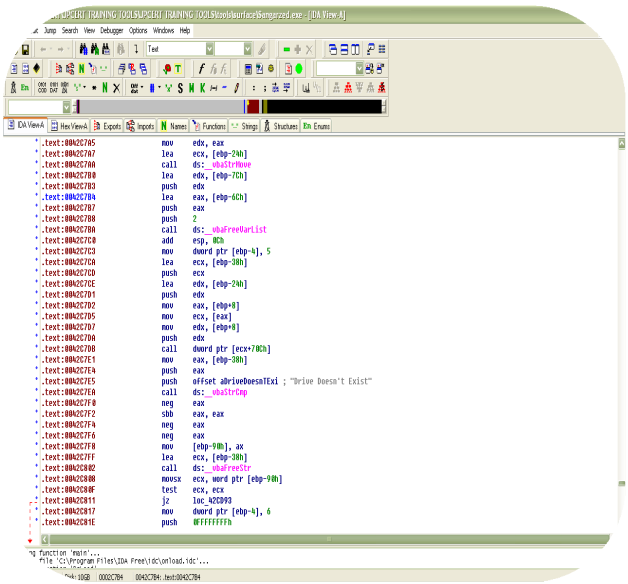
Untuk dapat menyimpulkan file yang terduga virus,harus melewati beberapa tahapan dan proses yang tidak sederhana. E3Sebuah file yang terduga mengandung *malware* akan di*disassemble* atau pemecahan file. Dalam proses ini, analisis membandingkan 4 buah *malware*dimana 2 diantaranya adalah *malware* experimental buatan sendiri, sedangkan sebuah virus lainnya adalah *malware* yang telah beredar di masyarakat. Dalam proses tersebut file terduga *malware*akan dianalisis cara kerjanya dalam lingkungan independen untuk mengetahui tingkat infeksi dan penyebaran melalui pemeriksaan registry. Inilah yang disebut proses Analisis Statik.

Pada gambar dibawah*header file* yang dianalisis tersebut memiliki *header file* 4D5A, yaitu *header file*Part *Executable* (PE). Kasus *malware* yang beredar di masyarakat, terjadi karena pengguna komputer lengah dengan ikon sebuah file. Ikon sebuah file kadang menipu sehingga ketika file tersebut jalankan maka file *malware* akan menyebar dan berkembang biak dalam sistem komputer tersebut. Dalam pemeriksaan kode hexa dari sebuah *malware* perlu diperhatikan penempatan alamat atau register oleh *malware*, dengan menggunakan tools IDAPro Free dan HxD.

Analisis *Disassembly* adalah proses analisis untuk mengetahui pengalamatan register oleh sebuah file. Untuk mendapat gambaran tentang proses input, proses dan output yang dilakukan oleh sebuah file. Dengan menggunakan tools IDAPro Free, nOllyDbg, dan FileInsight.

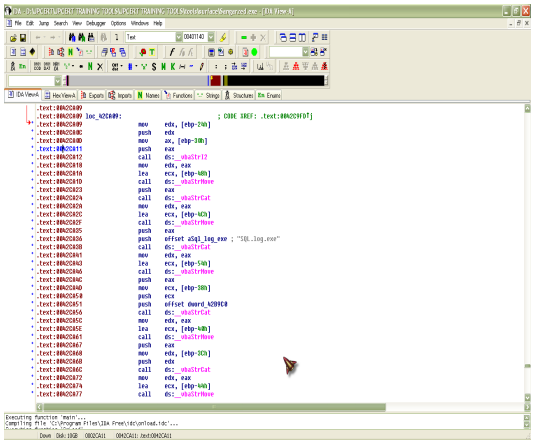


Gambar 1. Hasil proses disassemble



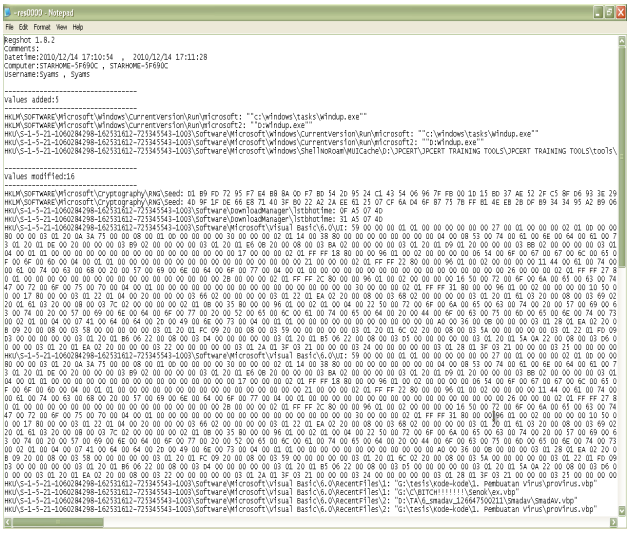
Gambar 2. Hasil pengaksesan file berekstensi .dll

Dari gambar diatas dapat disimpulkan bahwa malware yang dianalisis dengan menggunakan alat input dan output (I/O), diantaranya yang tergambar dari proses dalam IDAPro Free adalah penduplikasian pada alat penyimpan data atau drive. Proses yang tergambar diatas, malware tersebut menggunakan *error handler*dengan cara memberikan peringatan apabila drive penyimpan data atau partisi tidak ada dalam sistem komputer tersebut.
Lain halnya dengan proses dibawah ini :



Gambar 3. Proses peletakan duplikasi virus

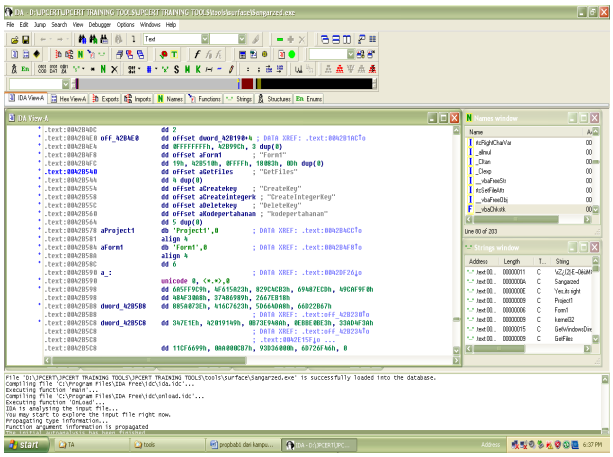
Proses dalam gambar diatas adalah proses peletakan duplikasi virus yang bernama “SQL.log.exe” dalam folder atau drive penyimpanan data suatu sistem komputer. Dimana virus memanfaatkan API Windows yang terdapat dalam MS Visual Basic untuk mempermudah dalam mengakses sistem dari Windows. Setelah virus menggandakan diri dengan file SQL.log.exe, virus akan memanggil file tersebut pada saat booting Windows dan mencari drive dan file yang belum terinfeksi.



Gambar 4. Peletakkan registry untuk memanggil file duplikasi virus

Peletakkan registry untuk memanggil file duplikasi virus menurut analisis yang dilakukan oleh penulis terletak pada :

1. HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\Folder\Hidden\SHOWALL\CheckedValue", 1.
2. HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\Folder\Hidden\SHOWALL\DefaultValue", 1.
3. "HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\Folder\Hidden\NOHIDDEN\CheckedValue", 2.
4. HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\NoFolderOptions", 1.
5. "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System\DisableRegistryTools", 1.
6. "HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\HideFileExt", 1.
7. "HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\HideFileExt", 1.
8. HKLM\Software\Microsoft\Windows\CurrentVersion\Run\TestingVirus\SQL.log.exe.

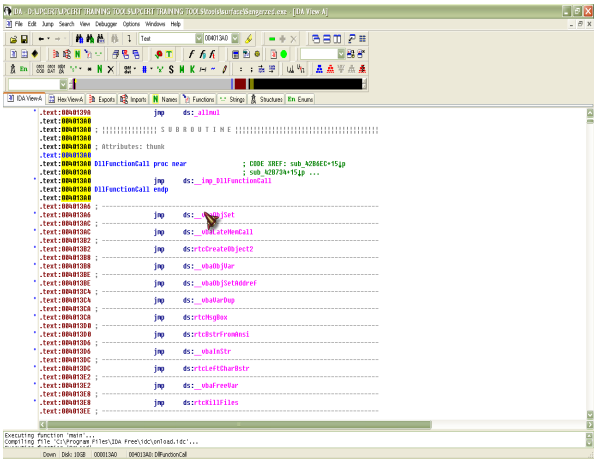


Gambar 5. File virus akan mengakses WSCRIPT.EXE

Pada gambar layar diatas dapat dilihat file virus akan mengakses WSCRIPT.EXE untuk menuliskan key pada registry Windows. WSCRIPT adalah Windows Script, dimana file ini akan menulis script yang telah diperintahkan oleh pengguna, aplikasi, dan program komputer. Pemanfaatan WSCRIPT.EXE dan WSCRIPT SHELL oleh virus akan memudahkan virus untuk menulis key pada registry, pembuatan file duplikasi, dan modifikasi registry.

Pemanggilan Dynamic Link Library juga dilakukan oleh virus yang tengah dianalisis. Pemanggilan file DLL memungkinkan virus untuk mengumpulkan data tentang sistem yang berguna untuk penduplikasian dan modifikasi registry Windows. Pemanfaatan DLL hampir sama dengan pemanfaatan WSCRIPT.EXE oleh virus. Pada beberapa kasus virus WSCRIPT.exe bawaan dari Windows dengan WSCRIPT.exe dari virus itu sendiri. Seperti pada *worm* Heart dan variannya.

Hal ini memungkinkan pengguna komputer tidak dapat me-*recovery* registry dengan *recovery* registry buatan sendiri dengan ekstensi .inf. Atau dengan kata lain meskipun registry editor dapat terbuka oleh pengguna maka pengguna komputer tidak dapat merubah struktur dari registry tersebut



Gambar 6. Hasil Pemanggilan Dynamic Link Library

CRC menghasilkan suatu checksum yaitu suatu nilai dihasilkan dari fungsi hash-nya, dimana nilai inilah yang nantinya digunakan untuk mendeteksi error pada transmisi atau penyimpanan data. Nilai CRC dihitung dan digabungkan sebelum dilakukan transmisi data atau penyimpanan, dan kemudian penerima akan melakukan verifikasi apakah data yang diterima tidak mengalami perubahan atau kerusakan. CRC32. CRC32 juga melambangkan panjang checksum dalam bit.

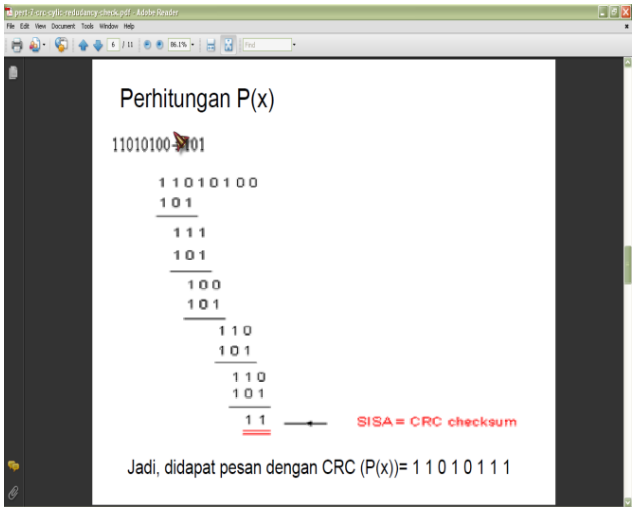
Bentuk CRC yang disediakan untuk algoritma sesuai dengan ide pembagian ”polynomial”. Dan hal ini digunakan untuk memperhitungkan checksum yang sama dari seluruh algoritma CRC. Algoritma CRC adalah cara yang bagus dan teruji untuk pengecekan byte dalam jumlah besar dari suatu file yang telah termodifikasi mau tidak. Algoritma ini mencari lewat seluruh jumlah byte dan menghasilkan angka 32 bit untuk menggambarkan isi file. Dan sangat kecil sekali kemungkinan dua stream dari byte yang berbeda mempunyai CRC yang sama. Algoritma CRC32 dapat diandalkan juga untuk memeriksa error yang terjadi dalam urutan byte.

A. Metode CRC

1. Data diperlakukan sebagai bilangan biner (0 & 1)
2. Bilangan ini dibagi dengan bilangan biner lainnya yang disebut polinomial.
3. Hasil sisa pembagian ini merupakan checksum CRC, yang akan ditambahkan pada pesan yang akan ditransmisikan.
4. Receiver akan membagi pesan (termasukCRC yang dihitung) dengan polynomial yang sama dengan polinomial yang digunakan oleh transmitter.
5. Jika sisa pembagian yang dilakukan oleh receiver ini sama dengan sisa pembagian yang dilakukan oleh transmitter, maka transmisi dapat dikatakan berhasil.

B. Perhitungan untuk mendapatkan CRC

Contoh :
Pesan = 110101 (yang akan ditransmisikan)
Polinomial = 101 (divisor)



Gambar 7. Perhitungan untuk mendapatkan CRC

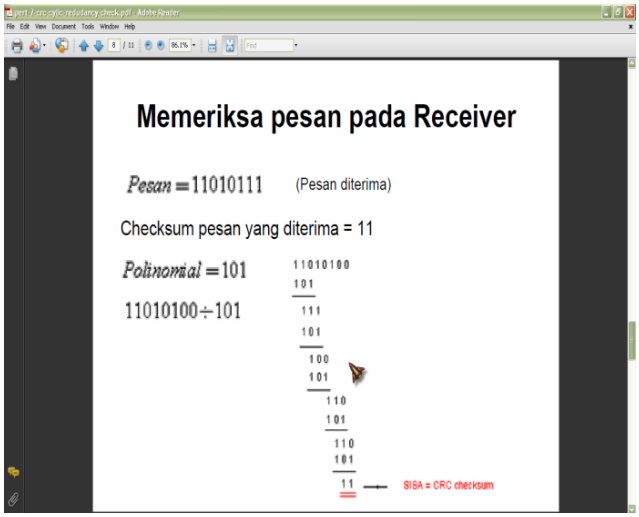
1. Pesan akan ditambahkan dengan bit nol sebanyak lebar bit polinomial. Dalam hal ini, lebar bit polynomial adalah 2, maka pesan akan ditambahkan dengan 00 menjadi 11010100, dan akan dibagi dengan polinomial.
2. Pembagian ini sama saja dengan men-XOR-kan semua bit yang dibagi dengan bit pembagi.
3. Perhitungan P(x) Jadi, didapat pesan dengan CRC (P(x))= 1 1 0 1 0 1 1 1

C. Memeriksa pesan pada Receiver

Hal ini dapat dilakukan dengan dua cara, yaitu:

Cara 1:

1. Pertama receiver akan memisahkan pesan dan checksum, kemudian akan menghitung checksum untuk pesan (setelah menambahkan bit nol sebanyak lebar bit polinomial).
2. Lalu receiver akan membandingkan kedua checksum tersebut (yang diterima dan yang dihitung).
3. Jika kedua checksum tersebut sama besar, maka tidak terjadi error selama transmisi

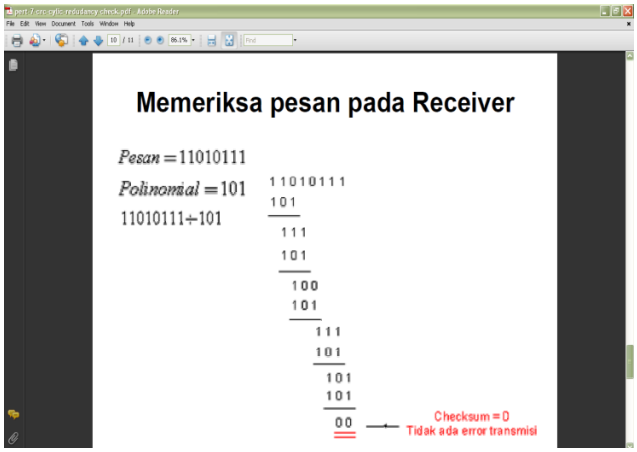


Gambar 8. Pemeriksaan pesan pada receiver (1)

(Pesan diterima) Checksum pesan yang diterima= 11

Cara 2 :

1. receiver akan menghitung checksum untuk keseluruhan pesan.
2. kemudian memeriksa apakah hasilnya sama dengan nol(berarti tidak terjadi error selama transmisi).



Gambar 9.Pemeriksaan pesan pada receiver (1)

1. Pesan : 100110001101
2. Divisor : 11001

Checksum MD5 sering digunakan untuk identifikasi sebuah file. Checksum MD5 atau juga yang sering disebut Cheksum MD5 adalah alternative identifikasi file yang diduga virus selain CRC32 yang sering kali mampu dikelabui apabila program virus tersebut di-*package* ulang dengan *packager*, seperti UPX dimana file asli dan file yang telah di-package akan mempunyai CRC yang berbeda.

Apabila sebuah file mengalami proses *packing* maka dengan sendirinya file itu akan mengalami enkripsi, dan akan sulit untuk menganalisis file tersebut. Cara yang dapat dilakukan untuk menganalisis file tersebut adalah meng-*unpack* file tersebut. *Unpack* sebuah file akan melepaskan enkripsi dari badan file dan analis bisa menemukan badan file secara utuh. Tools yang digunakan adalah CFF Explorer. Tools ini akan meng-*unpack* file sehingga kita bisa menemukan *header file* dari file tersebut. Hal ini memudahkan analis untuk menentukan jenis file.

Cheksum MD5 adalah cheksum yang stabil sehingga cheksum ini akan digunakan dalam identifikasi malware pada aplikasi anti*malware*. Cheksum dari CRC32 dan MD5 akan dimasukkan dalam *database* anti*malware* untuk pengenalan dalam proses *scanning.*

5. Penutup

Demikian proses yang dilakukan selama tahap analisis malware dengan cara statik. Perlu manjadi manjadi catatan penting disini adalah penggunaan berbagai macam perangkat lunak yang mendukung proses analisis secara statik. Setiap tahahap yang dilakukan mulai dari proses disassemble, kemudian pemeriksaan header pada hexadesimal, pemeriksaan checksum sampai pemeriksaan MD5 diharapkan akan menghasikan suatu deskripsi yang lengkap mengenai karakteristik suatu malware, sehingga sebagai pengguna komputer akan lebih peduli dan berhati-hati.

6. Pustaka

Zeltser Lenny, Introduction To Malware Analysis, SANS Institute, 2009

Dr. Masato Terada, Anti-Malware Engineering WorkShop, SGU 2009

Cristodorescu, Mihai, Malware Detection, Springer 2007

Aycock, John, Computer Viruses and Malware, Advances In Information Security, Springer 2006

Armknecht, Randy, Malware Analysis, October 2005 – http://www.rarmknecht.com

Brian Carrier, File System Forensic Analysis, Addison Wesley Professional, 2005

William G Eckert, Introduction To Forensic Sciences 2nd, CRC Press New York 1996