BAB IV
IMPLEMENTASI DAN PEMBAHASAN SISTEM

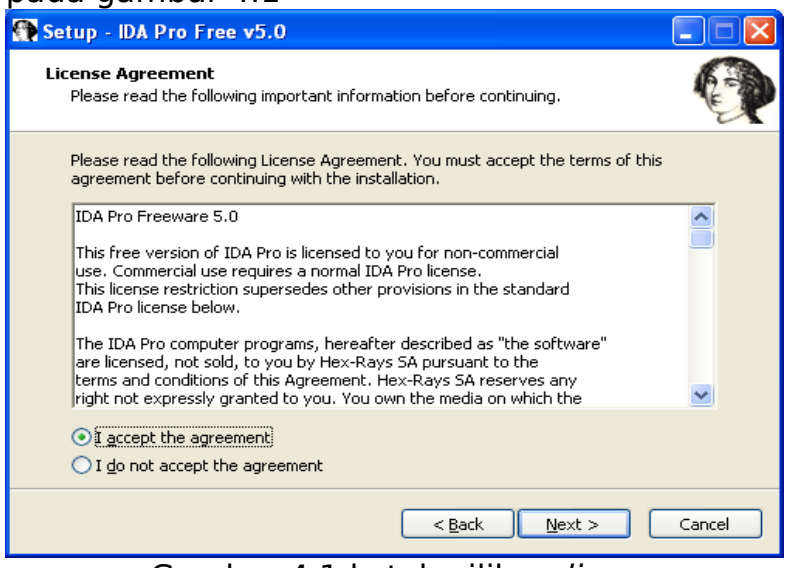
4.1. Implementasi

4.1.1. Menginstal *Tools* Analisis *Malware* pada SO *Windows*

Beberapa *tools* yang akan diinstall untuk proses analisis *Malware* antara lain:

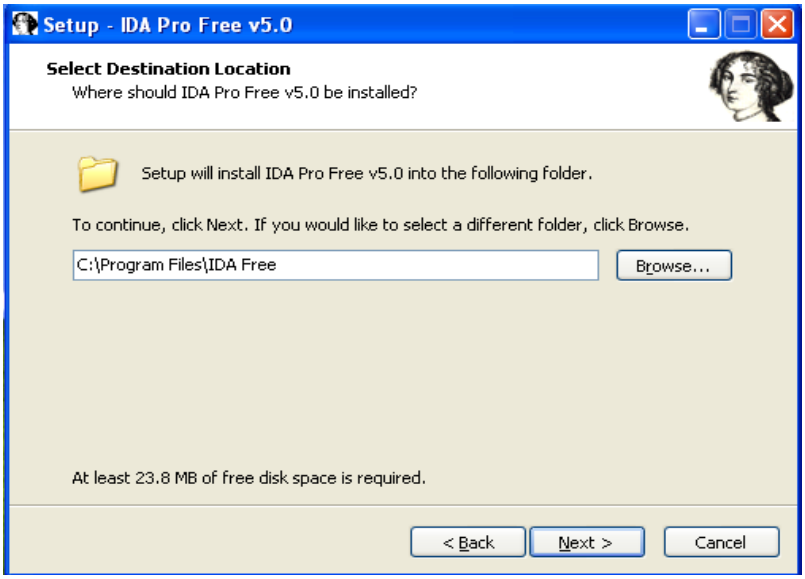
a. Instalasi *Ida-pro Disassembly*

1. Klik *setupIda-pro.exe* kemudian akan tampil *splash screen*, kamudian klik *next*.
2. Pilih *I accept the agreement*, lalu klik *next* seperti pada gambar 4.1



Gambar 4.1 kotak pilihan *license*

3. Masukan lokasi dimana *file – file Ida-pro* akan diinstal. *Ida-pro* secara default akan terinstall pada folder C:\Program *File*s\IDA Free, klik *next* untuk melanjutkan seperti ditunjukkan pada gambar 4.2.

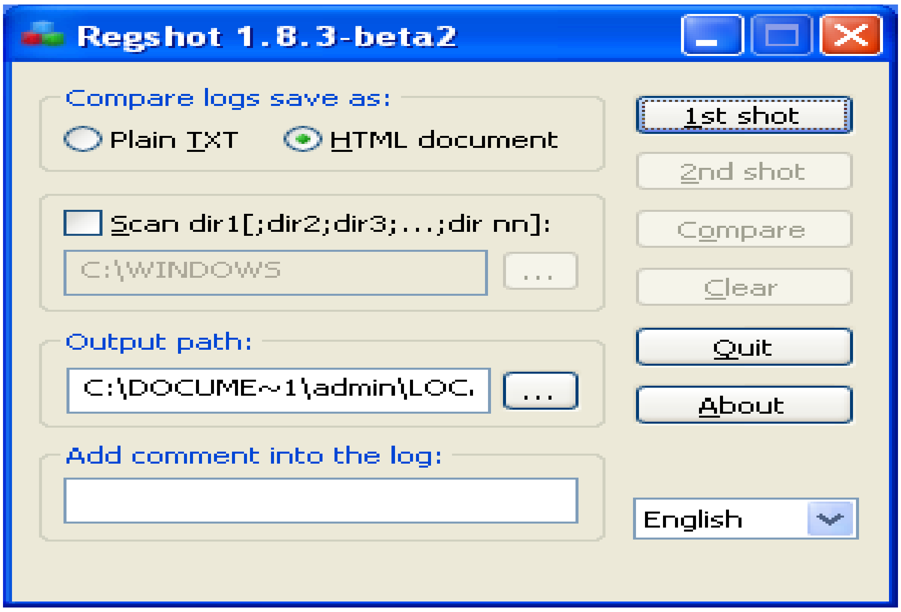


Gambar 4.2 kotak pilihan lokasi install

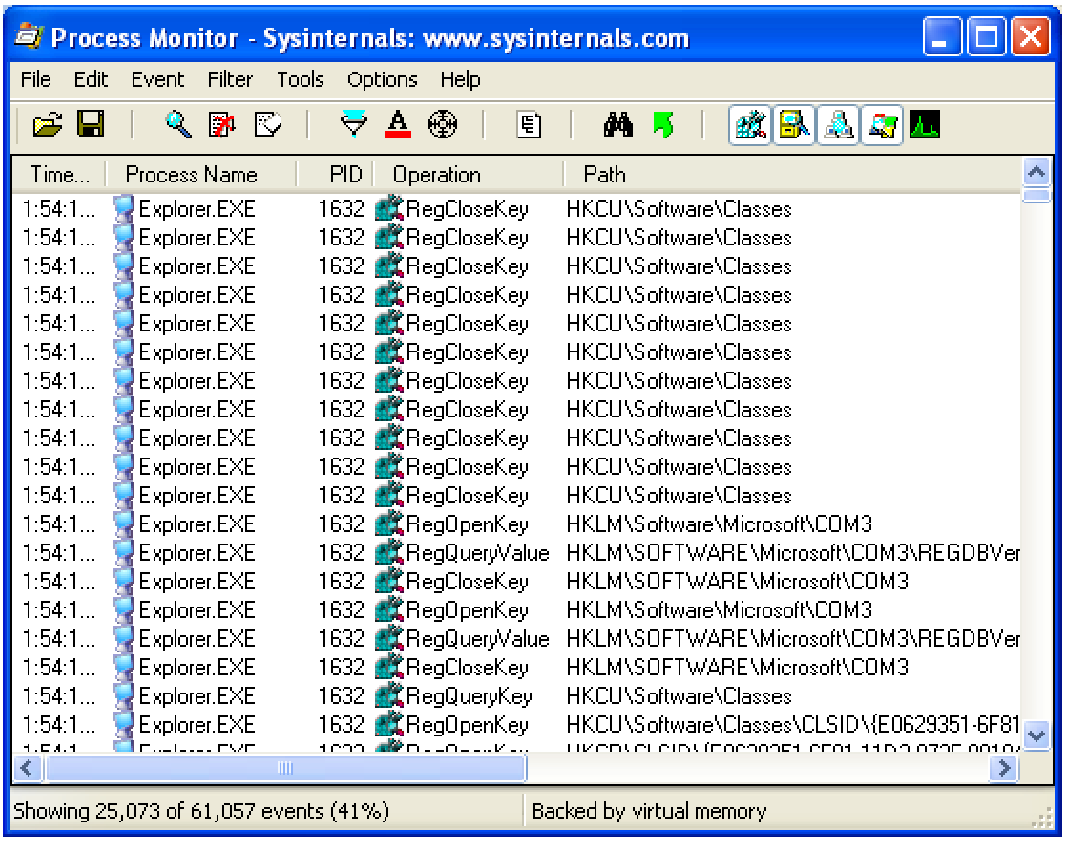
4. Beri centang pada *checkbox Create a desktop icon*, klik *next*
5. Klik install, untuk mulai menginstal *Ida-pro*.

b. Instalasi *RegShot*

Regshot merupakan aplikasi *fortable*, dan tidak membutuhkan proses instalasi seperti yang dilakukan pada *tool Ida-pro*. Untuk menjalankan regshot klik *file Regshot.exe.* Tampilan regshot ditunjukkan pada gambar 4.3.



Gambar 4.3 tampilan *tool* regshot



Gambar 4.4 tampilan tool process monitor

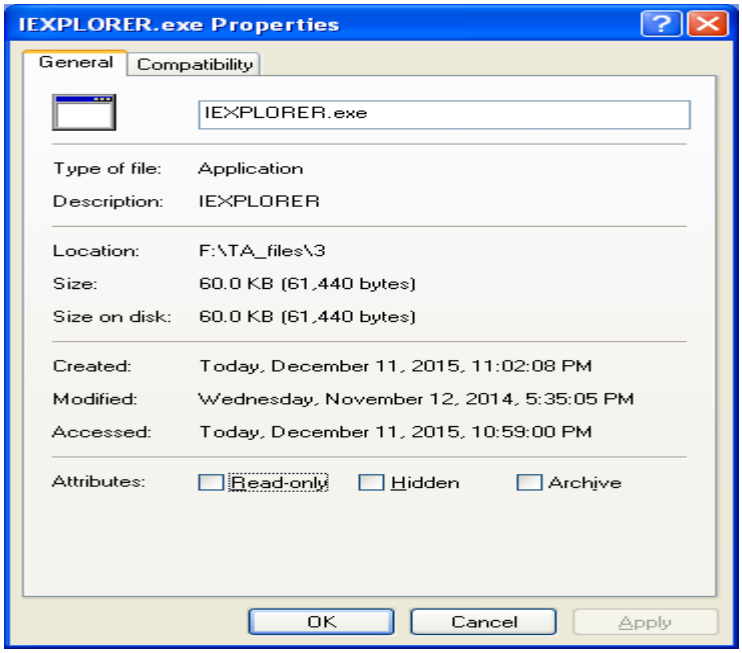
c. Process Monitor

Process Monitor merupakan aplikasi *fortable*, dan tidak membutuhkan proses instalasi seperti yang dilakukan pada *tool Ida-pro*. Untuk menjalankan regshot klik *file procmon.exe.* Tampilan *process monitor* ditunjukkan pada gambar 4.4.

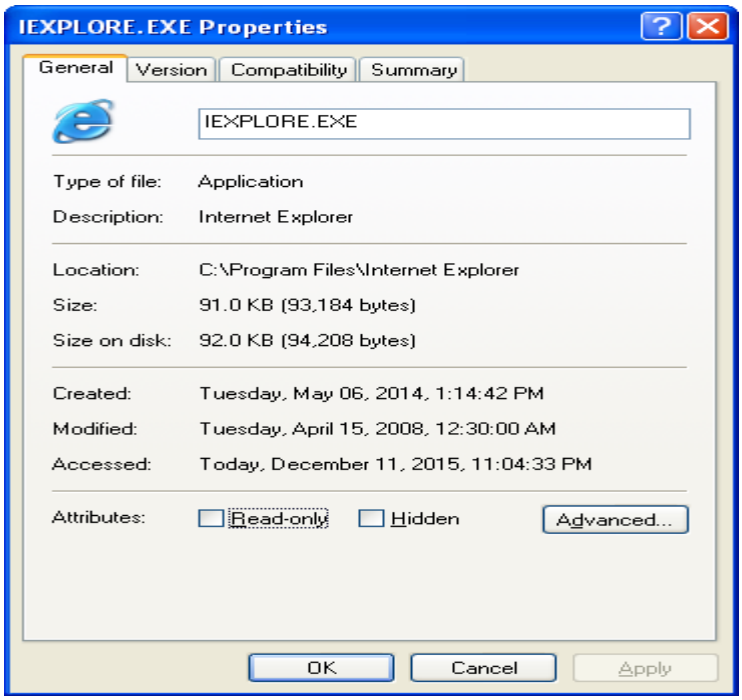
4.1.2. **Mencari *file Malware Trojan***

Mencari *file Malware Trojan* dapat dilakukan dengan mengikuti langkah – langkah seperti di bawah ini :

1. Jika nama *file Malware* menyerupai nama *file* program yang ada pada sistem operasi, maka *file* yang dicurigai harus dibandingkan dengan program asli dengan melakukan langkah – langkah sebagai berikut :
   a. Melihat *properties* dari kedua *file,* seperti pada gambar 4.5 dan 4.6.



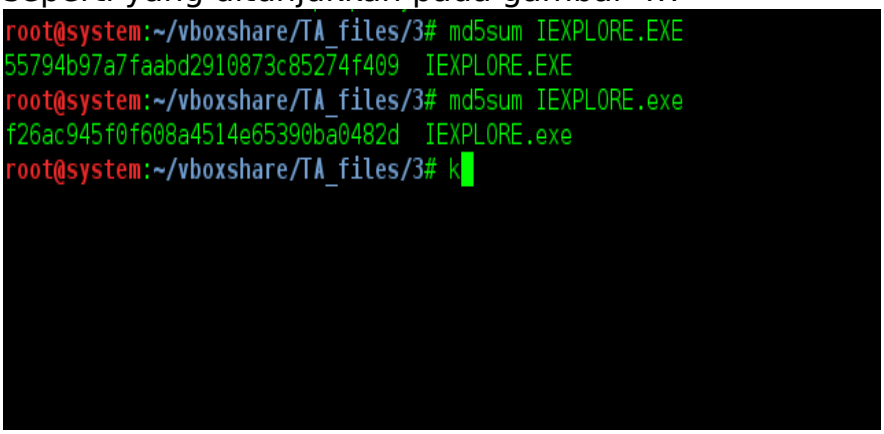
Gambar 4.5 hasil *properties* dari *file* program yang dicurigai



Gambar 4.6 hasil *properties* dari *file* program yang asli

b. Menggunakan *md5 checksum*

Untuk membandingkan struktur *file* dari *file* program yang asli dengan *file* program yang dicurigai maka digunakan *tool md5 checksum*, seperti yang ditunjukkan pada gambar 4.7



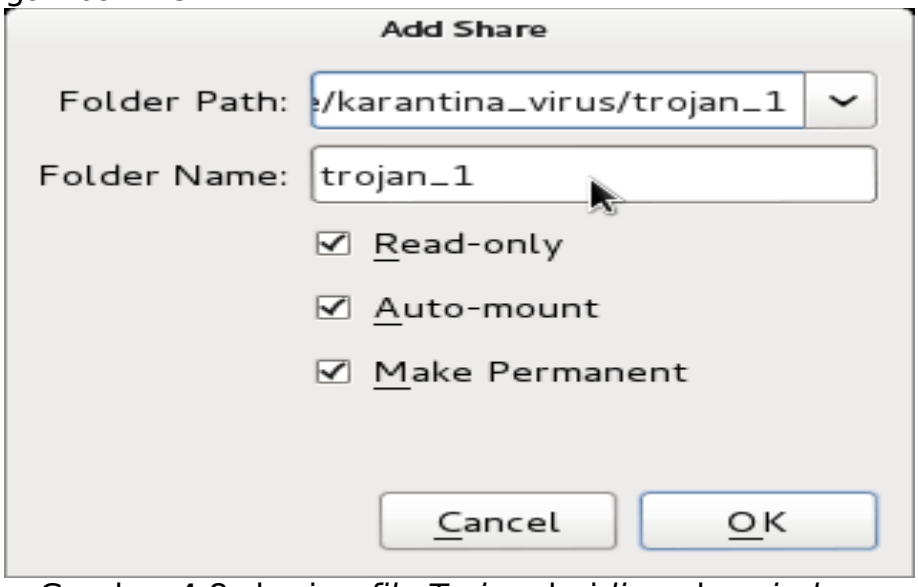
Gambar 4.7 membandingkan *file* menggunakan md5sum

Dari kedua hasil perbandingan seperti pada gambar 4.7, membuktikan bahwa *file* dengan nama *IEXPLORE.exe* adalah bukan *file* yang asli dan tidak lain adalah sebuah *file Malware* (*Trojan*)

4.1.3. **Melakukan karantina terhadap *file Trojan***

Karantina dapat dilakukan pada sistem operasi yang tidak kompatibel dengan *file Trojan*. Karena pada penelitian ini objek *Malware Trojan* yang akan dianalisis hanya komptibel pada sistem operasi *windows*, maka karantina dapat dilakukan pada sistem operasi selain *windows*, pada penelitian ini digunakan sistem operasi *linux*. Proses sharing file dari linux ke windows

menggunakan virtualbox share ditunjukkan pada gambar 4.8.



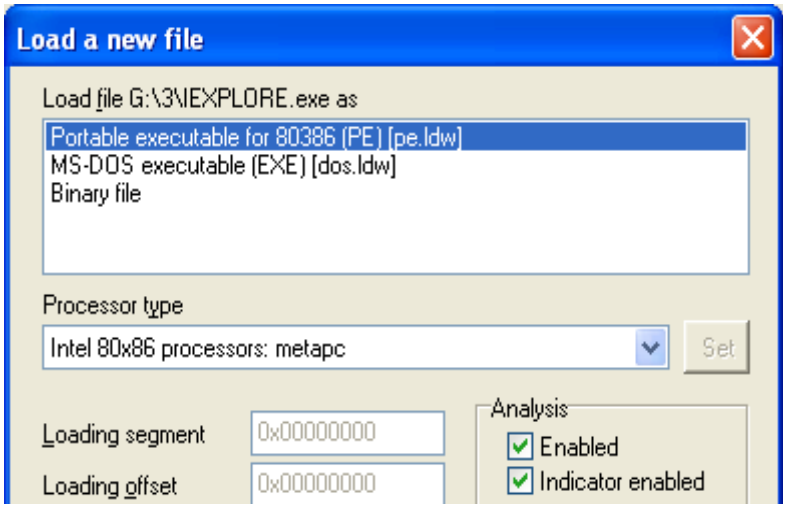
Gambar 4.8 sharing *file Trojan* dari *linux* ke *windows*

4.2. **Pembahasan Sistem**

4.2.1. **Analisis statis**

Analisis statis dilakukan dengan cara *Disassembly file* binari *Trojan.* Proses *disassembly* dilakukan untuk bisa mengetahui struktur program tersebut, sehingga cara kerja Trojan tersebut dapat diketahui. Adapun tahap yang dilakukan adalah :

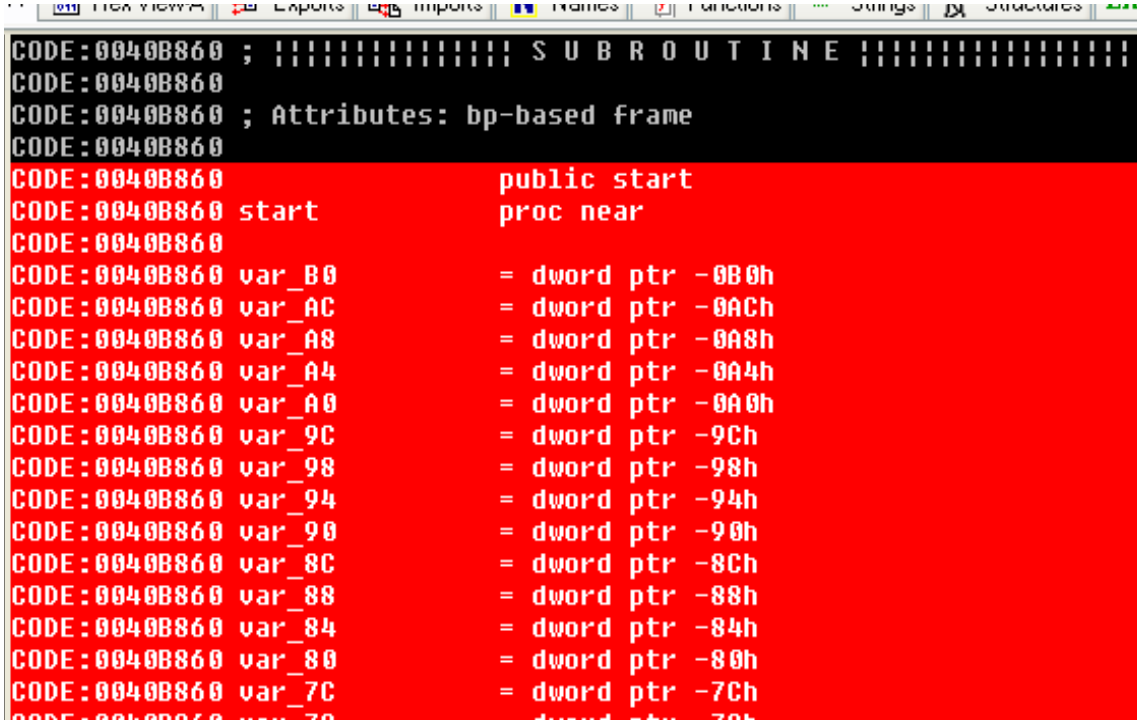
a. Loading file Trojan ke dalam tool ida-pro disassembly seperti ditujukkan pada gambar 4.9.



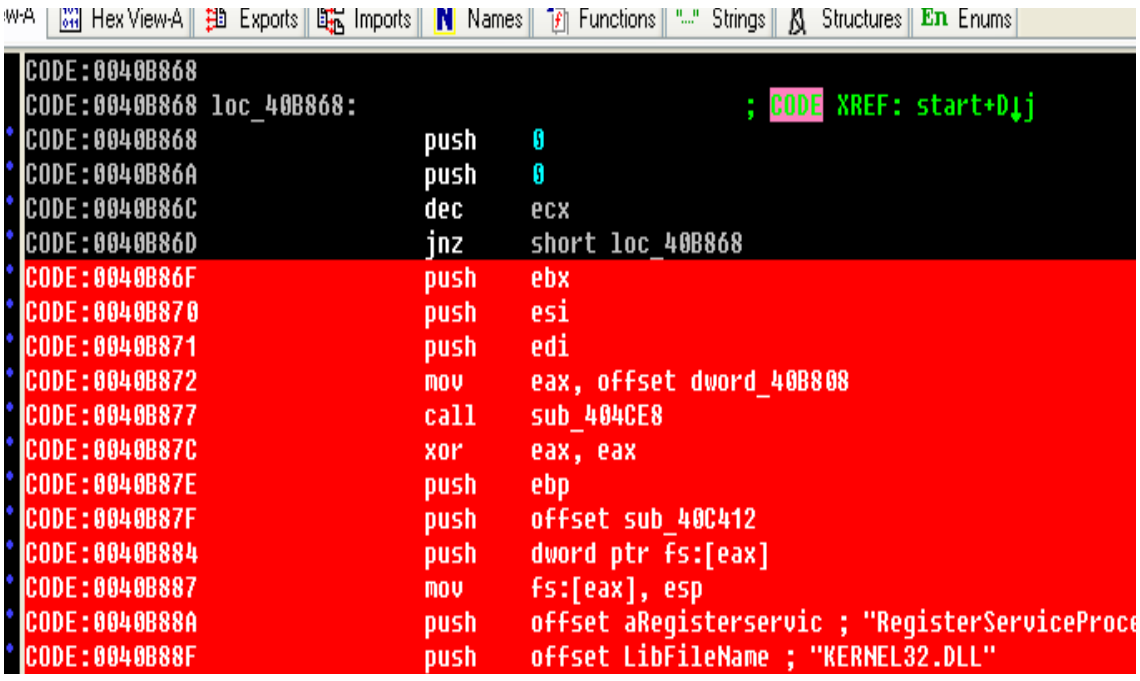
Gambar 4.9 proses *load file Trojan IEXPLORE.exe*

Gambar 4.9 menujukkan bahwa *type file Trojan IEXPLORE.exe* adalah *Portable executable for 80386*. Dan tipe PE 32.

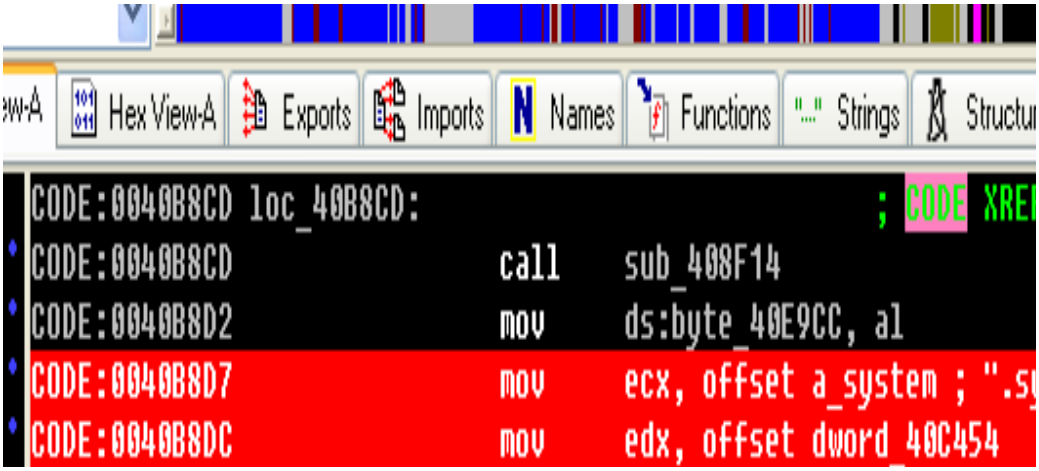
Gambar 4.10, 4.11 dan 4.12 menunjukkan kode *assembly* dari *file Trojan IEXPLORE.exe* setelah di-*load* ke dalam *tool Ida-pro disassembly*. Kode *assembly* dapat ditampilkan dalam bentuk struktur atau sesuai alur eksekusi program, dan dapat juga ditampilkan dalam bentuk baris kode.



Gambar 4.10 kode *assembly* dari file *IEXPLORE.exe*



Gambar 4.11 kode *assembly* dari file *IEXPLORE.exe*



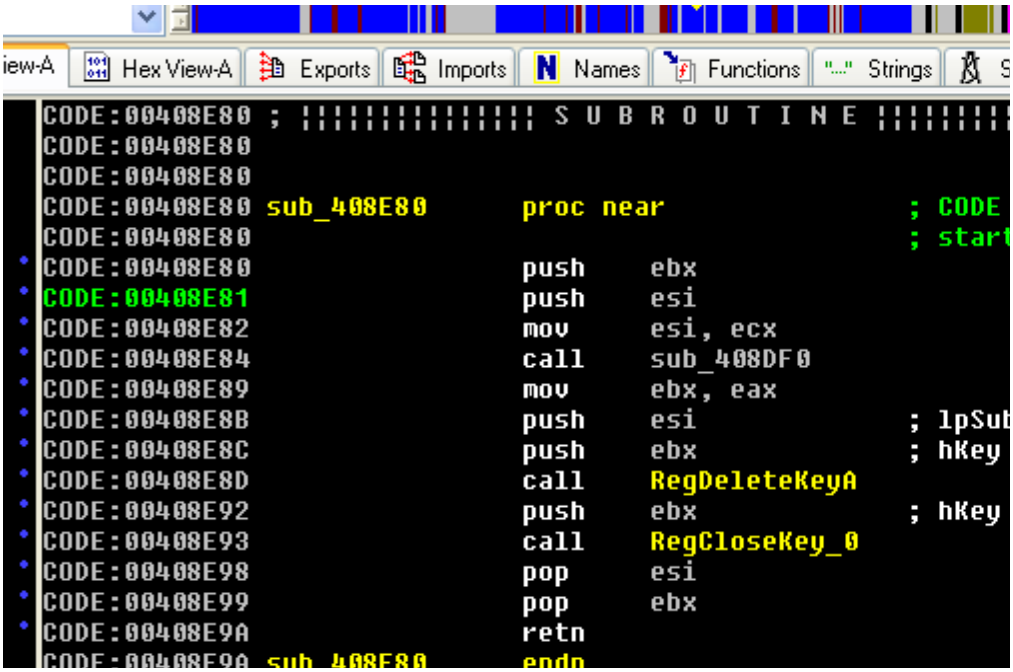
Gambar 4.12 kode *assembly* dari file *IEXPLORE.exe*

Setelah kode *assembly* didapatkan maka dilakukan analisis terhadap kode tersebut. Pada gambar 4.10, 4.11 dan 4.12 terlihat aksi – aksi yang dilakukan oleh *Trojan* saat pertama kali berjalan diantaranya:

1. Pada baris kode dengan alamat CODE:0040B860, seperti pada gambar 4.10, program pertama kali dijalankan dengan memanggil fungsi *start*, lalu baris berikutnya berisi variabel lokal yang dimiliki oleh program. Fungsi start sama dengan fungsi main pada program *java* dan *c++*.
2. Pada baris kode dengan alamat CODE:0040B86F hingga alamat CODE:0040B894 seperti pada gambar 4.11, terlihat program memanggil *library* milik *system* operasi *windows* yaitu *kernel32.dll* menggunakan fungsi *LoadLibraryA*. Library kernel32.dll adalah library milik

system operasi windows, library ini menyediakan fungsi API untuk manajemen memory, proses, input output file dan lain-lain.

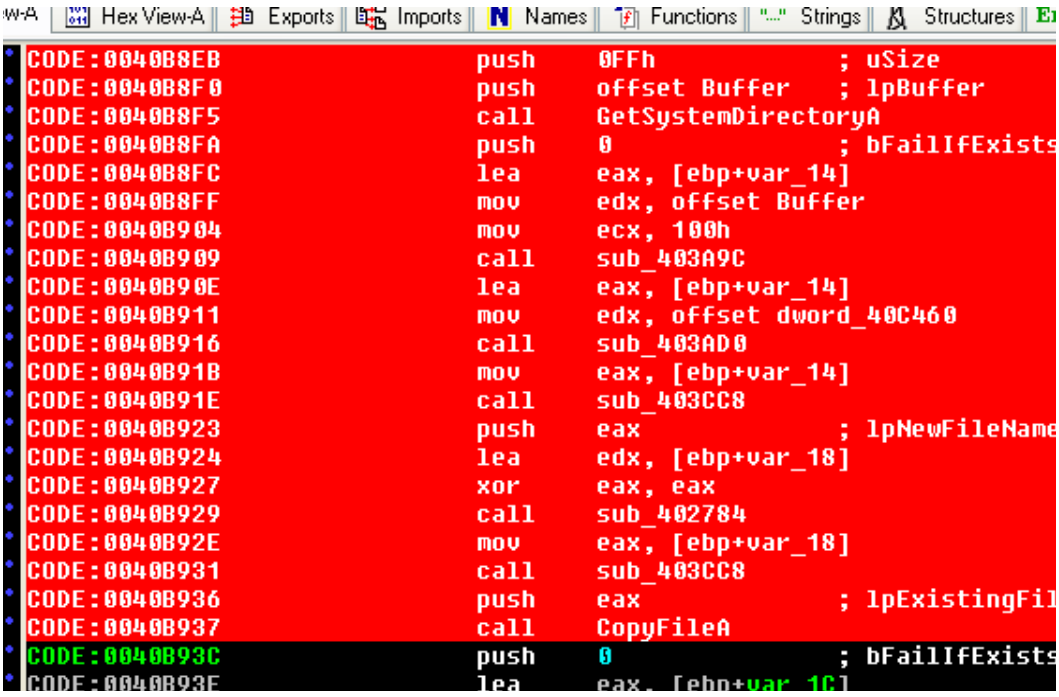
3. Pada baris kode dengan alamat CODE:0040B8D7 sampai dengan alamat CODE:0040B8E6 program melakukan pemanggilan fungsi sub_408E80 menggunakan perintah *assembly* yaitu perintah *call*. Fungsi sub_408E80 menerima tiga buah parameter ketika dipanggil. Baris kode *assembly* dari fungsi sub_408E80 ditunjukkan pada gambar 4.12 .



Gambar 4.13 kode assembly dari fungsi sub_408E80

Kode yang terdapat pada fungsi *sub_408E80* yang ditunjukkan oleh gambar 4.13, *Trojan* akan menghapus sebuah kunci *registry* pada *windows*. Jika dilihat dari

parameter yang diterima oleh fungsi *sub_408E80*, pada parameter pertama yaitu pada alamat *CODE:0040B8E1* dengan perintah *mov eax* 80000000h, dimana nilai 80000000h merupakan konstanta dari string *'HKEY_CLASSES_ROOT'*. sehingga dapat diketahui bahwa *Trojan* akan menghapus sebuah kunci *registry* denga nama *.system* di dalam *registry 'HKEY_CLASSES_ROOT'*.



Gambar 4.14 kode assembly Trojan IEXPLORE.exe

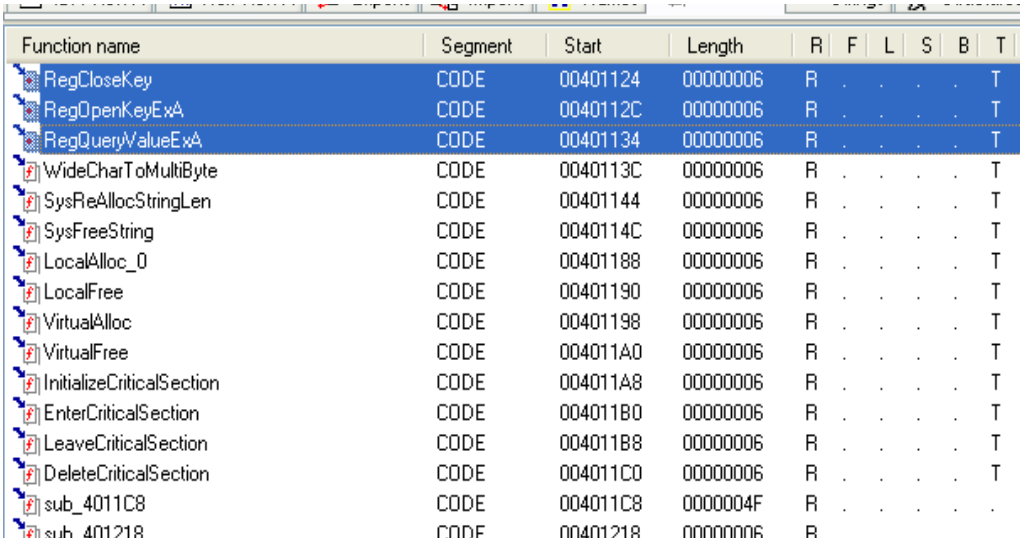
4. Seperti pada gambar 4.14, baris kode dengan alamat CODE:0040B8EB sampai dengan alamat CODE:0040B937, *Trojan* menggunakan fungsi *API* dari *windows* untuk menggandakan dirinya ke dalam *directory root system* operasi. fungsi – fungsi *API* yang digunakan adalah

- *GetSystemDirectoryA* adalah fungsi *API* untuk mengetahui *root* dari *system*, pada system operasi *windows root system* berada pada *directory 'C:\windows\system32\'.*
- *CopyFileA* adalah fungsi *API* untuk menggandakan file *Trojan* ke dalam file dengan nama yang berbeda.

Nama-nama file yang merupakan salinan dari *Trojan IEXPLORE.exe* adalah *SVCHOST.exe* dan *IEXPLORE.exe* yang akan tersimpan pada *directory 'C:\windows\system32\'.*

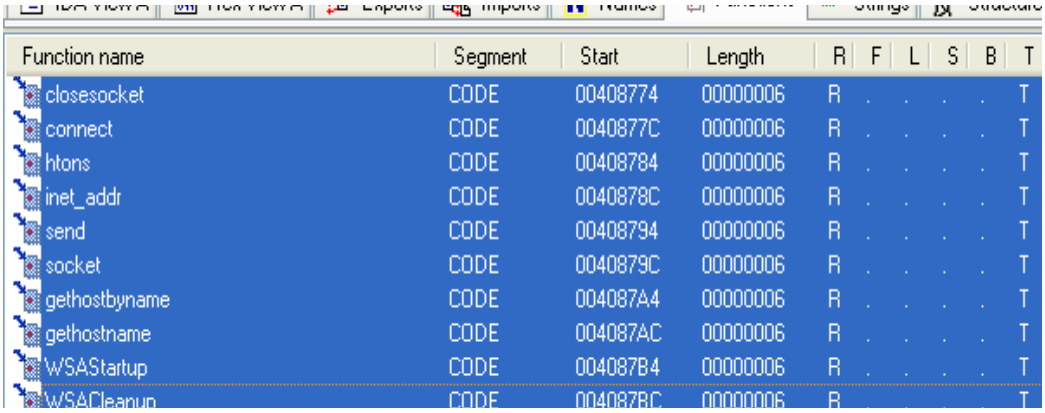
b. Analisis fungsi

Analisis fungsi ditujukan untuk mempercepat proses analisis, dikarenakan banyaknya fungsi fungsi yang terdapat di dalam program yang sekiranya tidak perlu untuk dianalisis. Untuk menemukan fungsi – fungsi apa saja yang terdapat di dalam *Trojan IEXPLORE.exe*, maka digunakan *tool function* milik *Ida-pro*. daftar fungsi – fungsi yang terdapat di dalam *Trojan IEXPLORE.exe* ditujukkan pada gambar 4.15 dan 4.16.



Function name	Segment	Start	Length	R	F	L	S	B	T
RegCloseKey	CODE	00401124	00000006	R	.	.	.	.	T
RegOpenKeyExA	CODE	0040112C	00000006	R	.	.	.	.	T
RegQueryValueExA	CODE	00401134	00000006	R	.	.	.	.	T
WideCharToMultiByte	CODE	0040113C	00000006	R	.	.	.	.	T
SysReAllocStringLen	CODE	00401144	00000006	R	.	.	.	.	T
SysFreeString	CODE	0040114C	00000006	R	.	.	.	.	T
LocalAlloc_0	CODE	00401188	00000006	R	.	.	.	.	T
LocalFree	CODE	00401190	00000006	R	.	.	.	.	T
VirtualAlloc	CODE	00401198	00000006	R	.	.	.	.	T
VirtualFree	CODE	004011A0	00000006	R	.	.	.	.	T
InitializeCriticalSection	CODE	004011A8	00000006	R	.	.	.	.	T
EnterCriticalSection	CODE	004011B0	00000006	R	.	.	.	.	T
LeaveCriticalSection	CODE	004011B8	00000006	R	.	.	.	.	T
DeleteCriticalSection	CODE	004011C0	00000006	R	.	.	.	.	T
sub_4011C8	CODE	004011C8	0000004F	R	.	.	.	.	.
sub_401218	CODE	00401218	00000006	R					

Gambar 4.15 fungsi yang terdapat di dalam *Trojan*



Function name	Segment	Start	Length	R	F	L	S	B	T
closesocket	CODE	00408774	00000006	R	.	.	.	.	T
connect	CODE	0040877C	00000006	R	.	.	.	.	T
htons	CODE	00408784	00000006	R	.	.	.	.	T
inet_addr	CODE	0040878C	00000006	R	.	.	.	.	T
send	CODE	00408794	00000006	R	.	.	.	.	T
socket	CODE	0040879C	00000006	R	.	.	.	.	T
gethostbyname	CODE	004087A4	00000006	R	.	.	.	.	T
gethostname	CODE	004087AC	00000006	R	.	.	.	.	T
WSAStartup	CODE	004087B4	00000006	R	.	.	.	.	T
WSACleanup	CODE	004087BC	00000006	R					T

Gambar 4.16 fungsi yang terdapat di dalam *Trojan*

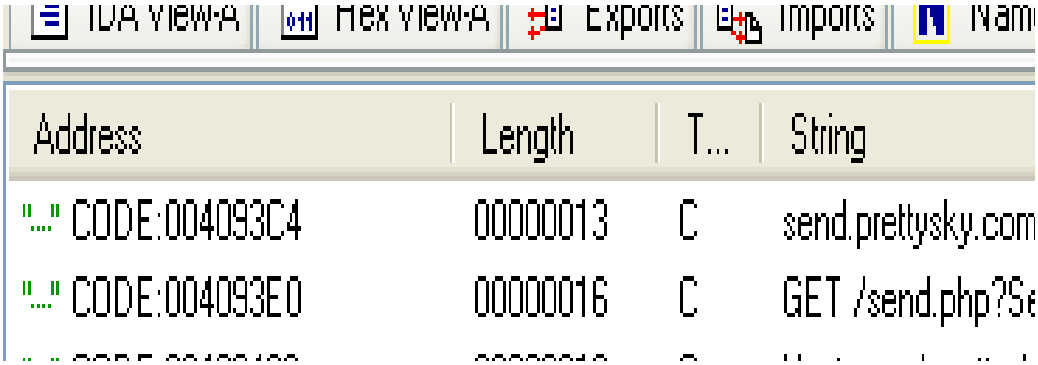
Gambar 4.15 dan 4.6, menunjukkan beberapa fungsi yang digunakan oleh *Trojan* adalah fungsi *API* yang disediakan oleh *windows*. Diantaranya :

1. Fungsi *RegCloseKey*
2. Fungsi *RegOpenKey*
3. Fungsi *RegQueryValueExA*
4. Fungsi *socket*
5. Fungsi *connect*
6. Fungsi *gethostbyname*

fungsi *API* di atas adalah fungsi yang disediakan oleh *library system* operasi *windows* untuk melakukan manipulasi *registry* dan koneksi ke jaringan. Untuk mempelajari fungsi di atas, maka dapat dibaca manual yang disediakan oleh *Microsoft*.

c. Analisis *string* yang terdapat di dalam *Trojan IEXPLORE.exe*

Analisis *string* dilakukan untuk mencari *string-string* yang terdapat di dalam program *Trojan IEXPLORE.exe.* karena string adalah *type* data yang dapat berisi suatu informasi.



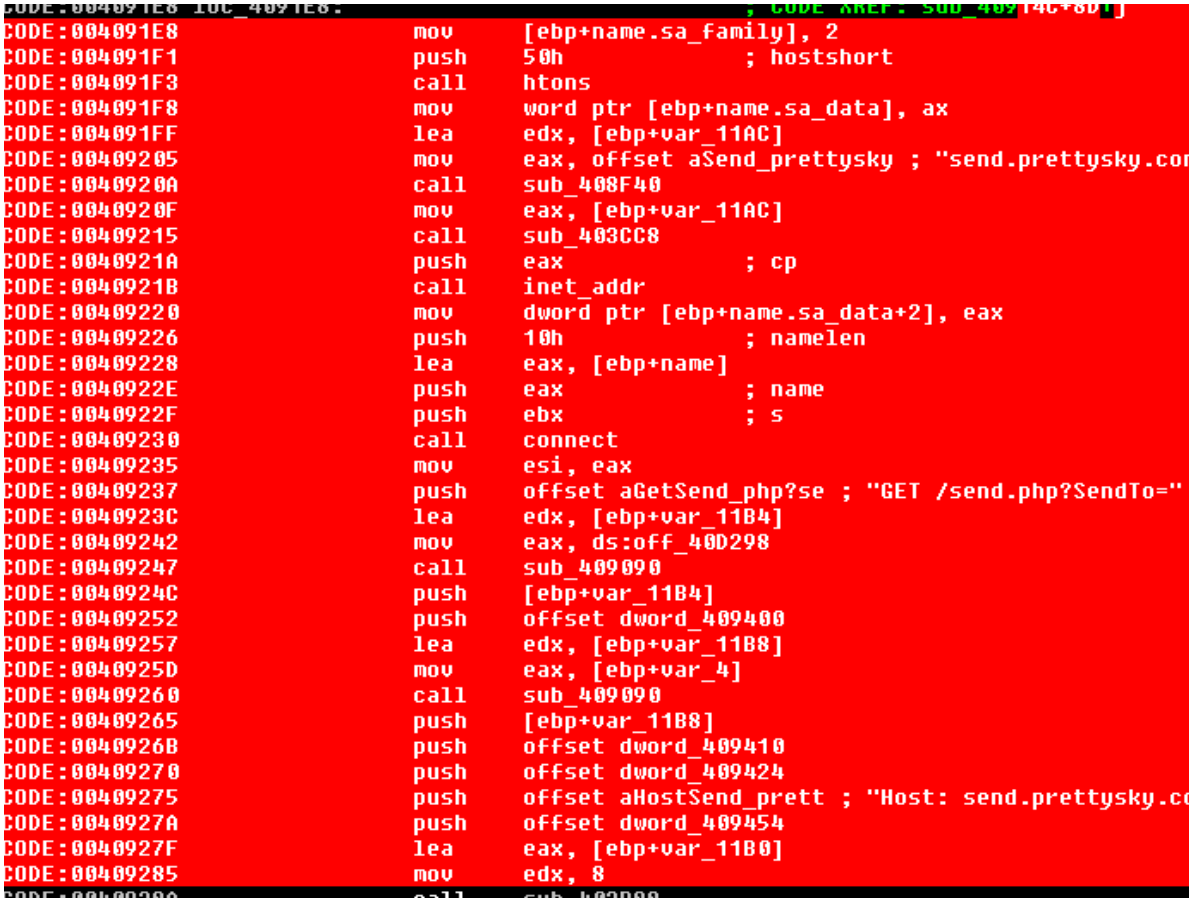
Gambar 4.17 string yang terdapat di dalam *Trojan*

Seperti pada gambar 4.17, terdapat *string* yang berisi informasi nama domain, *string* ditunjukkan pada tabel 4.1:

Tabel 4.1 tabel daftar string

Alamat	String
CODE:004093C4	Send.prettysky.com
CODE:004093E0	GET /send.php?SendTo=
CODE:00409430	Host : send.prettysky.com
CODE:00409464	Info.prettysky.com
CODE:00409480	Host : info.prettysky.com

Dari daftar *string* pada tabel 4.1, menunjukkan kemungkinan *Trojan IEXPLORE.exe* mencoba melakukan koneksi ke jaringan luar, karena string di atas menunjukkan sebuah nama domain. Gambar 4.18 menunjukkan kode *assembly* secara lengkap ketika *string* digunakan oleh sebuah fungsi di dalam program.



Gambar 4.18 fungsi yang menggunakan string *prettysky.com*

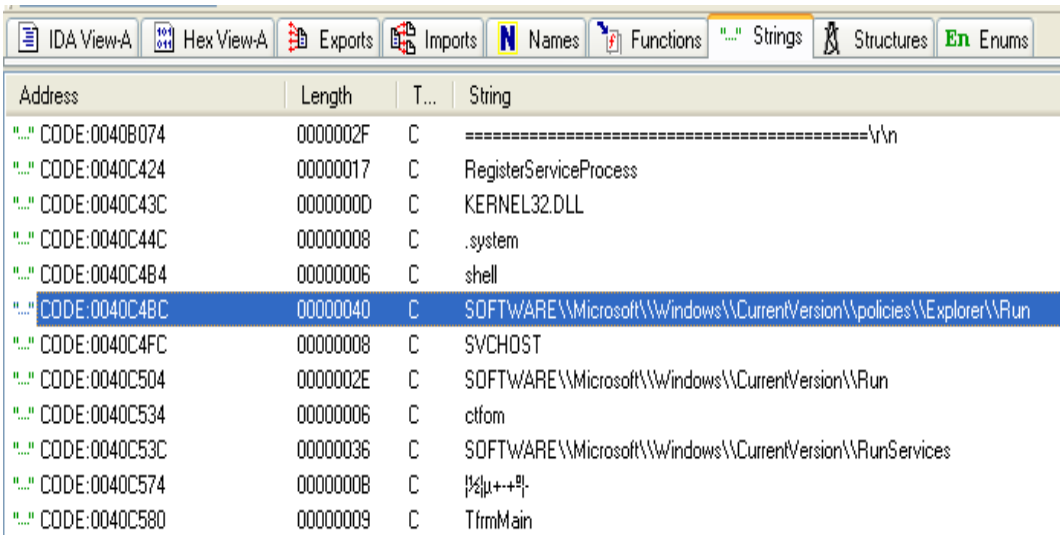
Dari kode *assembly* yang ditunjukkan oleh gambar 4.18, yaitu pada alamat CODE:004091E8 sampai dengan alamat CODE:00409258, beberapa fungsi milik *windows API* yang digunakan oleh program ditunjukkan pada table 4.2:

Tabel 4.2 tabel *windows api*

Fungsi Win32 API	Library
WSAStartup	Wsock32
Socket	Wsock32
htons	Wsock32
Inet_addr	Wsock32

Berdasarkan table 4.2, dapat disimpulkan *Trojan IEXPLORE.exe* mencoba melakukan koneksi ke alamat *'prettysky.com',* dan memanggil url *send.prettysky.com/send.php* yang menggunakan fungsi *GET* untuk mengirim suatu informasi.

Pada gambar 4.19 terdapat *string* yang berisi sebuah informasi alamat registry.

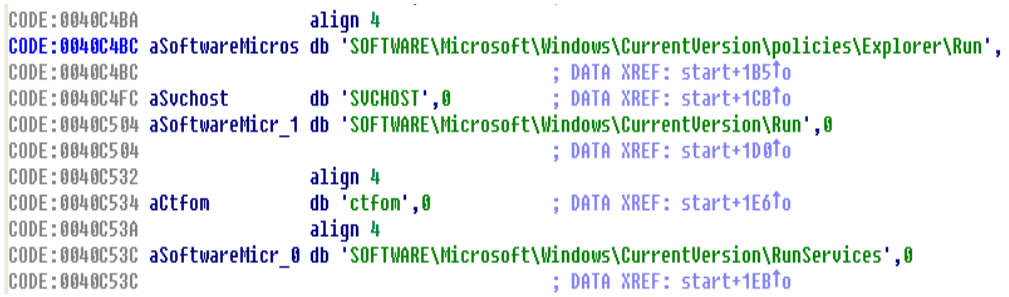


Gambar 4.19 *string* yang terdapat di dalam *Trojan*

Tabel 4.3 tabel string

Alamat	String
CODE:0040C4B4	Shell
CODE:0040C4BC	SOFTWARE\Microsoft\Windows\CurrentVersion\policies\Explorer\Run
CODE:0040C4FC	SVCHOST
CODE:0040C504	SOFTWARE\Microsoft\Windows\CurrentVersion\Run
CODE:0040C534	Ctfom
CODE:0040C53C	SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Services

Proses analisis dilanjutkan dengan mengklik salah satu string yang ada, gambar 4.20 menujukkan string tersebut di inisialisasi kedalam variabel yaitu pada alamat CODE:0040C48C sampai dengan CODE:0040C53C



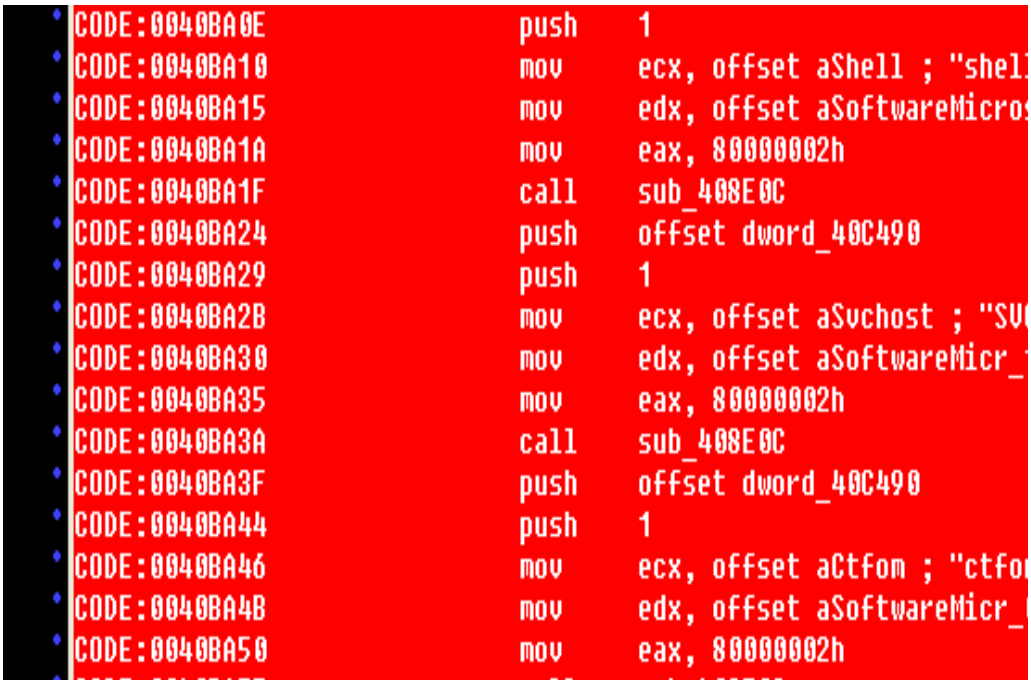
Gambar 4.20 code *assembly* dari string

Jika proses di konversi ke dalam bahasa pemrograman level atas yaitu bahasa c++ maka akan tampak seperti pada gambar 4.21

```
ar aSoftwareMicros[] =
OFTWARE\Microsoft\Windows\CurrentVersion\policies\Explorer
n";

ar aSvchost[] = "SVCHOST";

ar aSoftwareMicr_1[] =
OFTWARE\Microsoft\Windows\CurrentVersion\Run";

ar aCtfom[] = "ctfom";

ar aSoftwareMicr_0[] =
OFTWARE\Microsoft\Windows\CurrentVersion\RunServices";
```

Gambar 4.21 kode dalam bentuk bahasa c++

kode program pada gambar 4.21 menunjukkan kemungkinan *Trojan IEXPLORE.exe* mencoba melakukan manipulasi ke dalam *registry windows* ketika sedang di jalankan. Gambar 4.22 menunjukkan proses ketika *string* digunakan oleh sebuah fungsi di dalam program.



Gambar 4.22 string dipanggil oleh fungsi

Pada gambar 4.21 yaitu pada baris kode dengan alamat CODE:0040BA09 sampai dengan CODE:0040BA55 merupakan proses ketika *string* tersebut dipanggil oleh fungsi yaitu fungsi *sub_408E0C.* fungsi *sub_408E0C* menerima 5 buah parameter diantaranya:

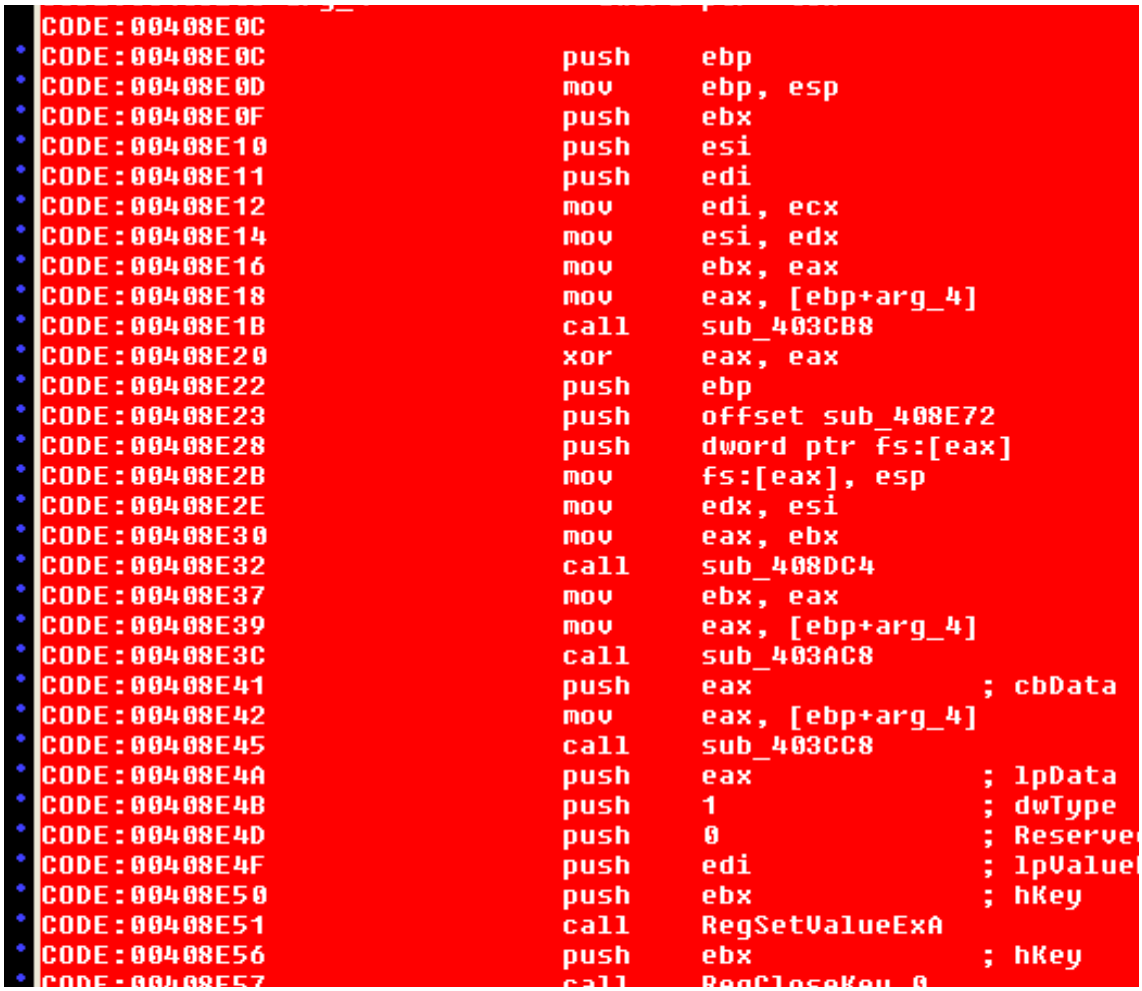
1. *Registry* yang dituju.
2. Alamat sub *registry* yang dituju untuk membuat nilai *registry*. Alamat sub *registry* ditunjukkan pada tabel 4.3.
3. Nama dari nilai *registry* yang dibuat, Seperti yang ditunjukkan pada gambar 4.21 yaitu *aShell, aSvchost, aCtfmon.*
4. Nilai 1
5. Data yang akan dimasukan ke dalam *registry*. Data ditunjukkan pada tabel 4.4

Tabel 4.4 tabel variabel dan data

Nama Variabel	Data
dword_40C4A4	IEXPLORE.EXE
dword_40C490	SVCHOST.EXE

Setelah fungsi sub_408E0C menerima semua parameter, lalu fungsi akan dieksekusi. Fungsi sub_408E0C akan melakukan manipulasi nilai pada sebuah registry menggunakan fungsi *API* milik *windows* yaitu *regCreateKey* dan *regSetValue*, fungsi sub_408E0C akan membuat nilai

berdasarkan parameter yang diterima. Kode *assembly* dari fungsi *sub_408E0C* ditujukkan pada gambar 4.23.



Gambar 4.23 kode *assembly* dari fungsi *sub_408E0C*

Kode *assembly* dari fungsi *sub_408E0C* seperti ditujukkan pada gambar 4.23 dapat diterjamahkan ke dalam bahasa pemrograman level atas seperti c++. Seperti ditunjukkan pada gambar 4.24.

5.cpp

```
#include <windows.h>
#include <iostream>
void _fastcall sub_408E0C(LPCTSTR aSoftwareMicros, LPCSTR aShell, int angka, LPCSTR val )
    {
        HKEY hKey;
        DWORD dwDisposition;
        LONG res = RegCreateKeyEx(HKEY_LOCAL_MACHINE, aSoftwareMicros, 0,NULL,0,KEY_ALL_ACCESS,NULL, &hKey,&dwDisposition);
        if(dwDisposition != REG_CREATED_NEW_KEY && dwDisposition != REG_OPENED_EXISTING_KEY){
            std::cout<<"terjadi error saat membuat key\n";
        }
        else{
            std::cout<<"key berhasil dibuat\n";
            LONG resp = RegSetValueEx(hKey, aShell, 0, REG_SZ, (BYTE*)val, strlen(val));
            if(resp == ERROR_SUCCESS){
                std::cout<< "berhasil mengisi nilai pada key";
            }
            else {
                std::cout<< "terjadi error saat mengisi nilai ke key";
            }
            RegCloseKey(hKey);
            }
        }
int main(){
    char aShell[] = "shell";
    char aSoftwareMicros[] = "SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\policies\\Explorer\\Run";
    CHAR iex[] = "IEXPLORER.EXE";
    sub_408E0C(aSoftwareMicros, aShell, 1, iex);
}
```

Gambar 4.25 konversi fungsi sub_408E0C ke c++

Dari hasil analisis menggunakan teknik *disassembly* di atas, dapat disimpulkan bahwa *Trojan IEXPLORE.exe* akan menggandakan diri ke dalam *directory root windows* dengan nama *SVCHOST.exe* dan *IEXPLORE.exe* kemudian mencoba melakukan koneksi ke domain *prettysky.com* dan melakukan manipulasi *registry*.

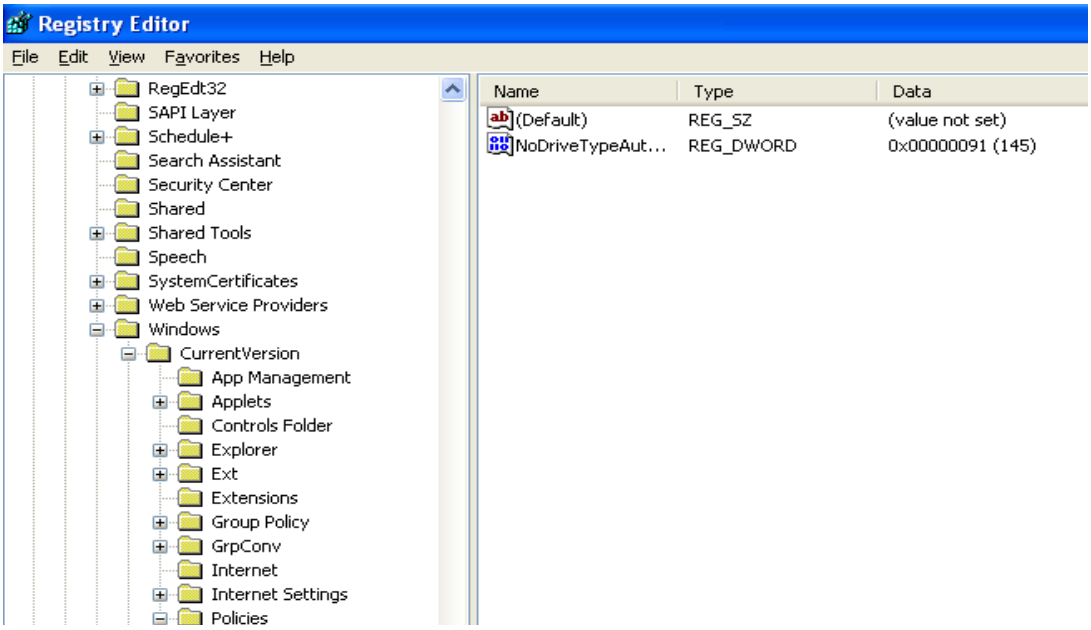
4.2.2. **Analisis Dinamis**

Proses analisis dinamis dilakukan untuk mengetahui tingkah laku *Trojan* ketika sedang berjalan pada lingkungan *system* operasi, analisis dinamis akan dibagi menjadi dua proses yaitu :

1. Proses sebelum eksekusi *Trojan IEXPLORE.exe*

Pada proses analisis statis yang dilakukan sebelumnya, menunjukkan bahwa *Malware Trojan* mencoba melakukan aksi manipulasi pada *registry*, yaitu menambahkan *key shell*, *svchost*, dan *ctfmon* pada alamat *registry* “SOFTWARE\Microsoft\Windows\CurrentVersion\policies\Explorer” dan “SOFTWARE\Microsoft\Windows\CurrentVersion\policies\Explorer\Run”.

Kedaan registry pada alamat tersebut ditjunjukkan pada gambar 4.26.



Gambar 4.26 Keadaan registry sebelum eksekusi *Trojan*

Beberapa tools yang harus dijalankan pada proses sebelum eksekusi diantaranya:

a. Buat *md5sum file Trojan IEXPLORE.exe*

Gambar 4.27 menunjukkan proses pembuatan *md5sum* terhadap *file Trojan IEXPLORE.exe*. *file md5* yang tersimpan akan dicek setelah *Trojan* selesai dieksekusi, untuk mengetahui struktur *file* apakah masih valid atau tidak.

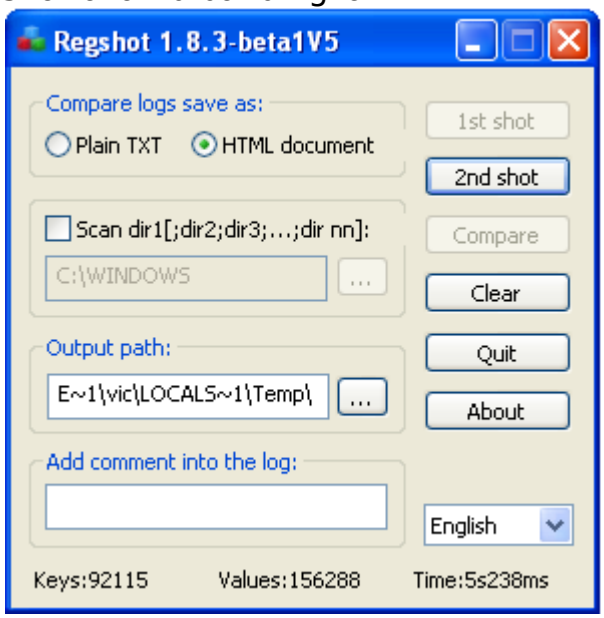
```
root@system:~/vboxshare/karantina_virus/trojan_1# ls -l
total 156
-rw------- 1 root root 61440 Nov 12  2014 IEXPLORE.exe
-rw-r--r-- 1 root root 93184 Apr 15  2008 IEXPLORE.EXE
-rw-r--r-- 1 root root    47 Dec 23 11:07 IEXPLORE.exe.md5
root@system:~/vboxshare/karantina_virus/trojan_1# md5sum IEXPLORE.exe > IEXPLORE.exe.md5
root@system:~/vboxshare/karantina_virus/trojan_1# md5sum -c IEXPLORE.exe.md5
IEXPLORE.exe: OK
root@system:~/vboxshare/karantina_virus/trojan_1#
```

Gambar 4.27 pembuatan *file md5* dari *IEXPLORE.exe*

b. Menjalan *RegShot* sebelum eksekusi

Gambar 4.28 menujukkan kondisi *registry windows* setelah di *shot* pertama kali dan akan menunggu *shot* yang

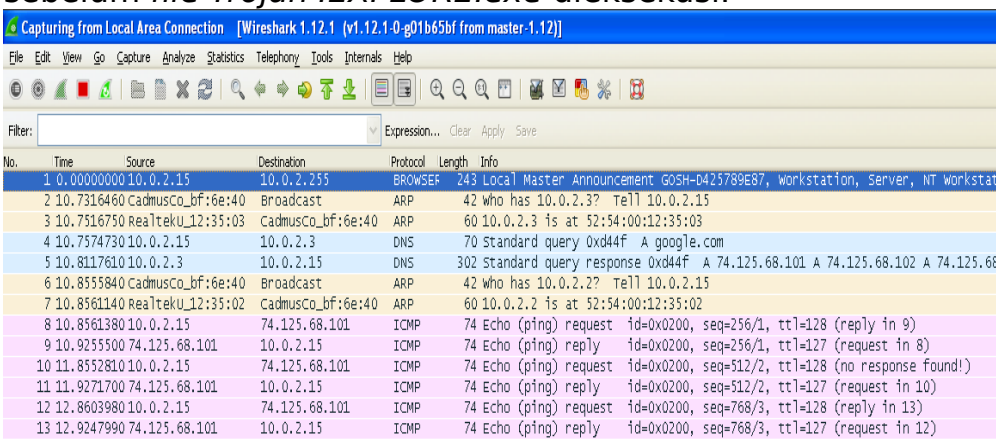
ke dua, yaitu setelah eksekusi *file Trojan*, lalu kemudian kedua hasil *shot* akan dibandingkan.



Gambar 4.28 *Regshot* sebelum eksekusi

c. Menjalankan *wireshark*

Wireshark berguna untuk memantau lalulintas jaringan pada computer. Gambar 4.29 menujukkan kondisi jaringan sebelum *file Trojan IEXPLORE.exe* dieksekusi.



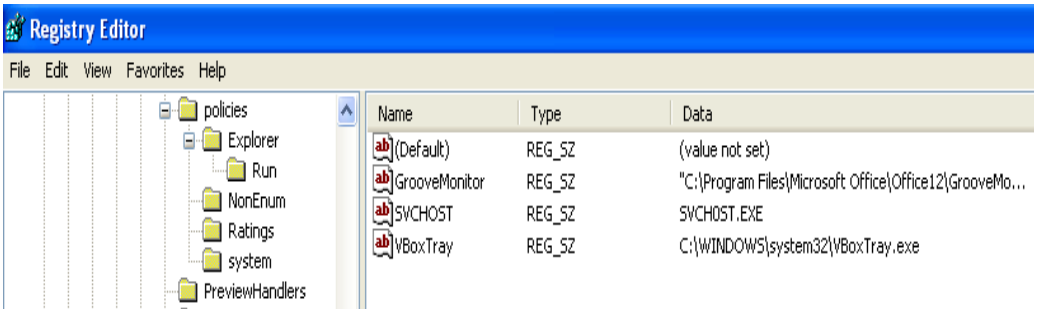
Gambar 4.29 *capture* paket dengan *wireshark*

2. Proses setelah eksekusi trojan *IEXPLORE.exe*

Setelah *Trojan* tereksekusi, terlihat key baru berhasil ditambahkan pada registry dengan alamat yang ditunjukkan

pada gambar 4.26, yaitu pada proses sebelum eksekusi. Gambar 4.30 dan 4.31 menunjukkan keadaan registry setelah proses eksekusi *Trojan*.

Gambar 4.30 key registry yang ditambahkan



Gambar 4.31 key registry yang ditambahkan

Pada gambar 4.30 dan 4.31 menunjukkan penambahan *key* baru pada *registry*, yaitu *key shell*, dan *SVCHOST* dengan nilai *IEXPLORE.exe* dan *SVCHOST.exe* yang tidak lain adalah file *Trojan* yang ter-*copy* ke dalam folder *system32* pada saat *Trojan* dieksekusi. *Key run* pada *registry* merupakan *key* yang digunakan untuk menjalankan suatu file atau program secara otomatis ketika *system* pertama kali dijalankan, sehingga dapat disimpulkan *Trojan IEXPLORE.exe* dan *SVCHOST.exe* akan secara otomatis berjalan ketika komputer dinyalakan.

Beberapa *tools* yang dijalankan untuk mengecek keadaan *system* setelah proses eksekusi sebagai berikut:

1. Validasi *file* menggunakan *md5sum*

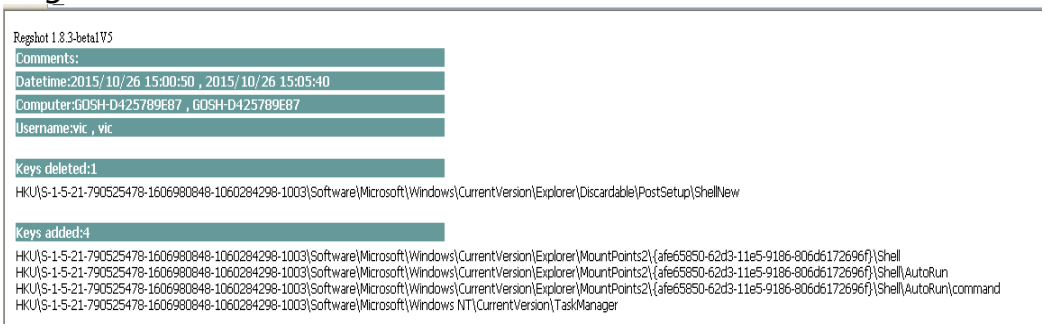
Gambar 4.32 menunjukkan kondisi struktur *file* dari *Trojan IEXPLORE.exe* setelah dieksekusi.

```
root@system:~/vboxshare/karantina_virus/trojan_1# ls -l
total 156
-rw------- 1 root root 61440 Nov 12  2014 IEXPLORE.exe
-rw-r--r-- 1 root root 93184 Apr 15  2008 IEXPLORE.EXE
-rw-r--r-- 1 root root    47 Dec 23 11:07 IEXPLORE.exe.md5
root@system:~/vboxshare/karantina_virus/trojan_1# md5sum IEXPLORE.exe > IEXPLORE.exe.md5
root@system:~/vboxshare/karantina_virus/trojan_1# md5sum -c IEXPLORE.exe.md5
IEXPLORE.exe: OK
root@system:~/vboxshare/karantina_virus/trojan_1#
```

Gambar 4.32 validasi *file* dengan *md5sum* setelah eksekusi

Berdasarkan gambar 4.32. menandakan bahwa *Trojan IEXPLORE.exe* tidak melakukan manipulasi pada struktur binernya.

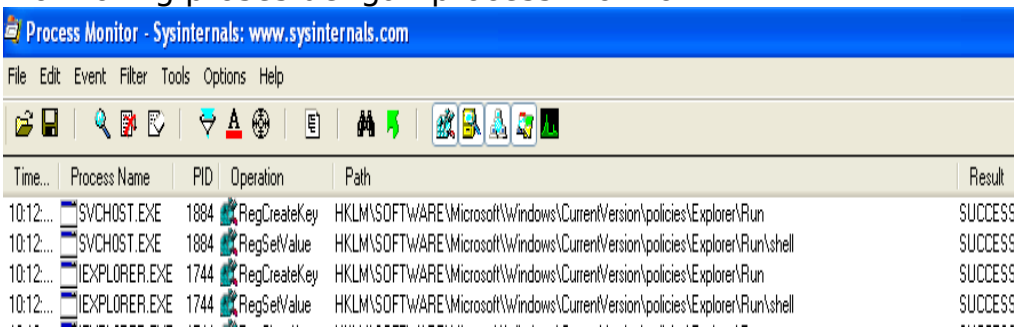
1. *Regshot* setelah eksekusi



Gambar 4.33 hasil compare regshot sebelum dan sesudah eksekusi

Gambar 4.33 adalah hasil *compare* dari *registry windows* sebelum dan sesudah eksekusi, tampak bahwa beberapa *key registry* baru yang dibuat.

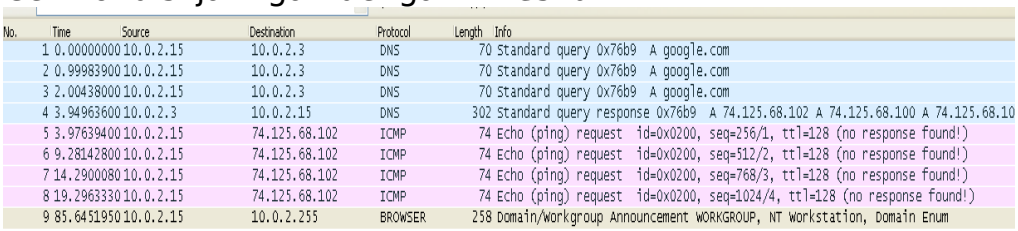
2. Monitoring proses dengan process monitor



Gambar 4.34 *monitoring* proses dengan *process monitor*

Dari pantauan proses yang berjalan menggunakan *process monitor* seperti gambar 4.34, tampak beberapa proses dengan nama *SVCHOST.EXE* dan *IEXPLORER.EXE* melakukan operasi pembuatan *key* pada *registry*. Keadaan ini sesuai dengan informasi yang ditemukan pada proses analisis statis sebelumnya.

3. Cek kondisi jaringan dengan *wireshark*



Gambar 4.35 pemantauan jaringan menggunakan *wireshark*

Gambar 4.35 menujukkan bahwa tidak ada aktifitas pada jaringan yang mencoba mengkases ke alamat *prettysky.com* atau membuka suatu *port*.

Pada proses analisis dinamis dapat disimpulkan bahwa aktifitas Trojan IEXPLORE.exe ketika dijalankan pada lingkungan system operasi, tidak jauh berbeda dengan

informasi yang ditemukan ketika proses analisis statis, hanya saja tidak ditemukan aktifitas koneksi ke domain ‘prettysky.com’ seperti yang ditemukan pada proses analisis statis. Hal ini memungkinkan fungsi koneksi tidak tereksekusi dengan baik ketika program dijalankan.