
악성코드 분석 보고서

2022.08

<<목 차>>

Part1. 기초 분석

- 개요 ----- 3
- VirusTotal 기초 분석 결과 ----- 4
- 기초 파일 분석 결과 ----- 5

Part2. 정적 분석

- EXE 파일 정보 ----- 8
- 텍스트 정보 ----- 10
- PE 구조 분석 ----- 11

Part3. 동적 분석

- OllyDbg 를 통한 동적 분석 ----- 15
- IDA Pro 를 통한 동적 분석 ----- 17

Part4. 결론 및 대응 방안

- 결론 ----- 24
- 대응 방안 ----- 25

Part1. 기초 분석

1. 개요

■ 악성코드 개념

이 파일의 악성코드는 트로이목마 악성코드이다. 트로이목마 악성코드는 겉보기에 정상적인 프로그램으로 보이지만, 실행하면 악성코드가 실행된다.

■ 트로이목마 특징

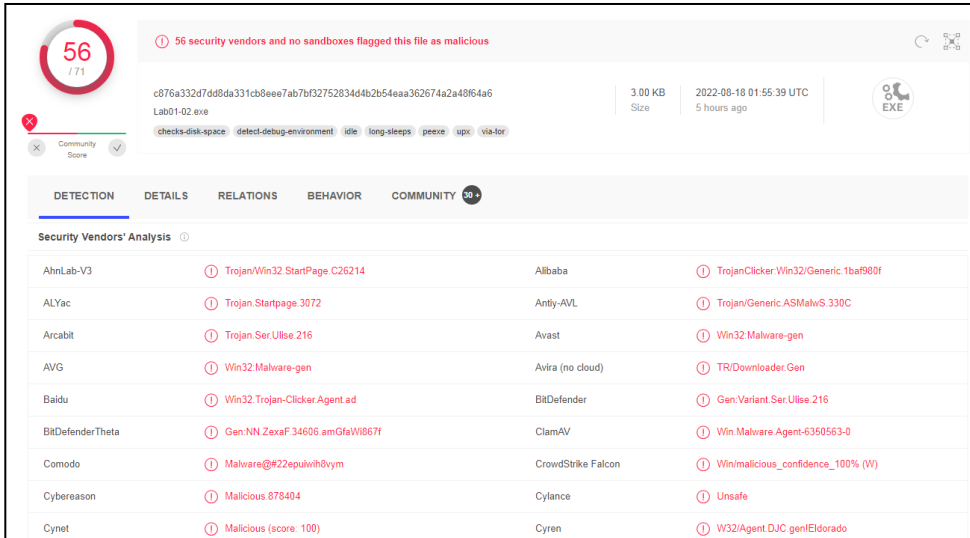
- 정보유출 및 제어권 획득이 목적이므로 컴퓨터 사용자가 트로이 목마의 침투 및 활동을 인식하지 못하도록 설계
- 바이러스와 같이 복제 기능이 없으며, 웜과 같이 증식 능력이 없는 악성 프로그램
- 자신을 복사하지 않고, 다른 파일을 감염시키지 않으므로, 트로이 목마 역할을 하는 파일만 삭제하면 치료 가능

■ 주요 감염 경로

- P2P 사이트: 유틸리티 프로그램이나 게임 등으로 가장하여 프로그램 내에 숨겨 있음
- Email: 첨부파일에 숨겨 있는 경우가 많음. 은행이나 휴대폰 사용 명세서 등과 같이 실행을 유도하는 메일로 감염되는 경우가 많음

2. VirusTotal 기초 분석 결과

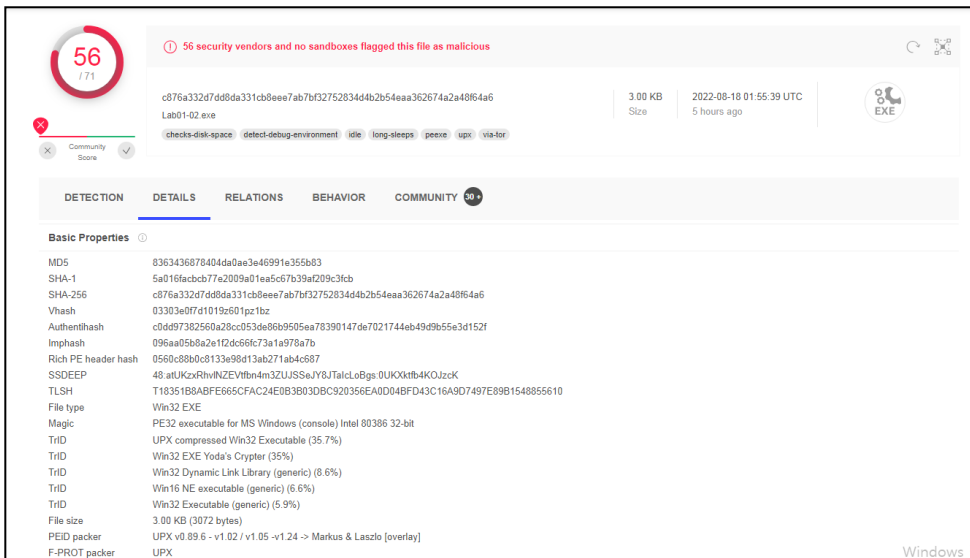
■ VirusTotal 사이트를 통한 악성코드 및 파일 정보 확인



The screenshot shows the VirusTotal detection results for a file named 'Lab01-02.exe'. The file has a size of 3.00 KB and was uploaded 5 hours ago. It has a community score of 56/71, indicating it is likely malicious. The detection results show that 56 security vendors and no sandboxes flagged this file as malicious. The file is identified as a Win32 executable (EXE) and is associated with the TrojanClicker-Win32/Generic.1ba9f90f.

Security Vendor	Detection
AhnLab-V3	Trojan.Win32.StartPage.C26214
ALYac	Trojan.Startpage.3072
Arcabit	Trojan.Ser.Ulisse.216
AVG	Win32.Malware-gen
Baidu	Win32.Trojan-Clicker.Agent.ad
BitDefender Theta	Gen.NN.ZexaF.34606.amGfaW/867f
Comodo	Malware.@#22epulwib8vym
Cybereason	Malicious.875404
Cynet	Malicious (score: 100)
Alibaba	TrojanClicker.Win32/Generic.1ba9f90f
Anfly-AVL	Trojan/Generic.ASMalwS.330C
Avast	Win32.Malware-gen
Avira (no cloud)	TR/Downloader.Gen
BitDefender	Gen.Variant.Ser.Ulisse.216
ClamAV	Win.Malware.Agent-6350563-0
CrowdStrike Falcon	Win/malicious_confidence_100% (W)
Cylance	Unsafe
Cyren	W32/Agent.DJC.gen/Eldorado

[그림 1-1] VirusTotal 의 detection 에 표시된 악성코드 정보



The screenshot shows the VirusTotal details for the file 'Lab01-02.exe'. The file is identified as a Win32 executable (EXE) and is associated with the TrojanClicker-Win32/Generic.1ba9f90f. The details section provides information about the file's basic properties, including its MD5, SHA-1, SHA-256, Vhash, Authenthash, Imphash, Rich PE header hash, SDEEP, TLSH, File type, Magic, TriD, File size, PEID packer, and F-PROT packer.

Property	Value
MD5	8363436878404da0ae3e46991e355b83
SHA-1	5a016facbcb77e2099a01ea5c67b39a209c3fcb
SHA-256	c876a332d7dd8da331cb8eee7ab7bf327528344db2b54aaa362674a2a48f64a6
Vhash	03303e07d10192601pz1bz
Authenthash	c0d97382560a28cc053de86b9505ea78390147de7021744eb49db55e3d152f
Imphash	096aa05b8a2e1f2dc66f73a1a978a7b
Rich PE header hash	0560c88b0c8133e98d13ab271ab4c687
SDEEP	48:atUKxRhVnZEVItn4m3ZUJSs6Jy8JTaLc0Bgs.0UKX0ttb4KOJzcK
TLSH	T1835188ABFE665CFAC24E0B38030BC920356EA0D048FD43C16A9D7497E89B1548855610
File type	Win32 EXE
Magic	PE32 executable for MS Windows (console) Intel 80386 32-bit
TriD	UPX compressed Win32 Executable (35.7%)
TriD	Win32 EXE Yoda's Crypter (35%)
TriD	Win32 Dynamic Link Library (generic) (8.6%)
TriD	Win16 NE executable (generic) (6.6%)
TriD	Win32 Executable (generic) (5.9%)
File size	3.00 KB (3072 bytes)
PEID packer	UPX v0.89.6 - v1.02 / v1.05 - v1.24 -> Markus & Laszlo [jovray]
F-PROT packer	UPX

[그림 1-2] VirusTotal 의 details 에 표시된 프로그램 정보

3. 기초 파일 분석

■ 기초 파일 분석 종합

구분	내용
파일명	Lab01-02.exe
파일 버전	N/A
파일 크기	3072 bytes
제작 시기	2019/08/08
파일 타입	Win32.exe
MD5	8363436878404DA0AE3E46991E355B83
SHA-1	5A016FACBCB77E2009A01EA5C67B39AF209C3FCB
포함 함수	KERNEL32.dll / MSVCRT.dll / ADVAPI32.dll / WININET.dll
악성코드 분류	트로이목마
패킹 여부	Exeinfo/VirusTotal 에서 UPX 패킹되어 있다고 출력
출처	실습 파일으로 MegaIT 에서 제공 받음

[표 1-2] 악성코드 파일 분석 종합 축약표

2. VirusTotal 기초 분석 결과

■ VirusTotal 사이트를 통한 악성코드 및 파일 분석

구분	내용
악성코드 탐지 백신	56 / 71 (71개 백신 중 56개 탐지)
분석 일시	2022/08/18
악성코드 분류	트로이목마

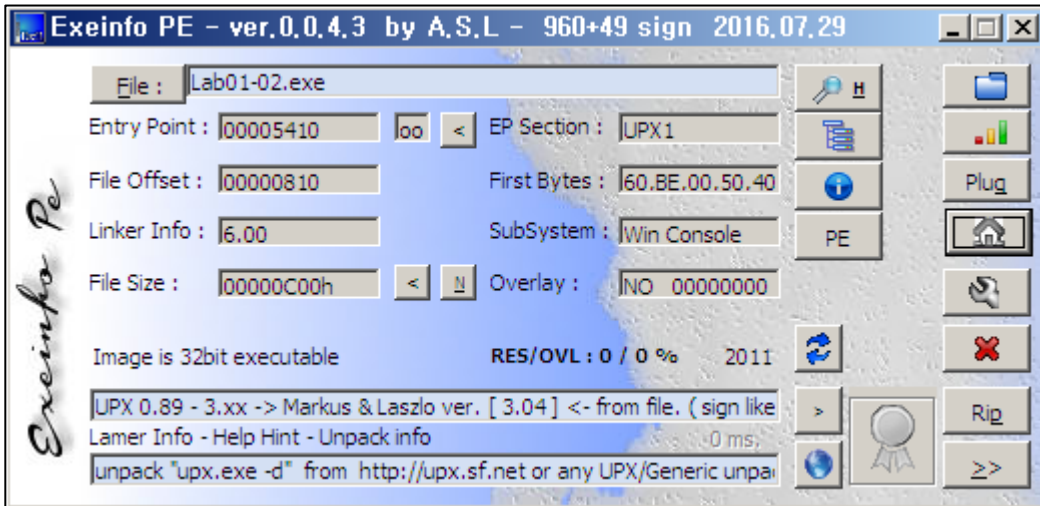
[표 1-1] VirusTotal 를 통한 기초 분석 결과 축약표

- 악성코드 Lab01-02.exe. 를 본격적으로 분석하기 전 기초 분석한 결과 다수의 백신으로부터 Trojan이 많이 검색되는 것으로 보아 트로이목마일 가능성이 높다고 판단

Part2. 정적 분석

1. EXE 파일 정보

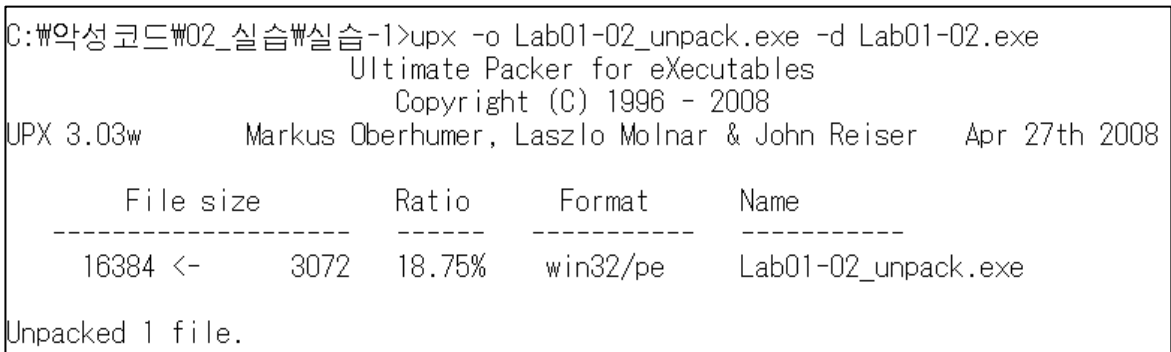
■ Lab01-02.exe 파일 Exeinfo 을 통한 파일 정보 확인



[그림 2-1] Exeinfo PE 에 악성코드를 업로드한 결과

- Exeinfo: EXE 파일 정보를 보여주는 정적 분석 도구
- UPX 를 통해 패킹되어 있는 파일임을 알 수 있음 → 언패킹 필요

■ Lab01-02.exe 언패킹

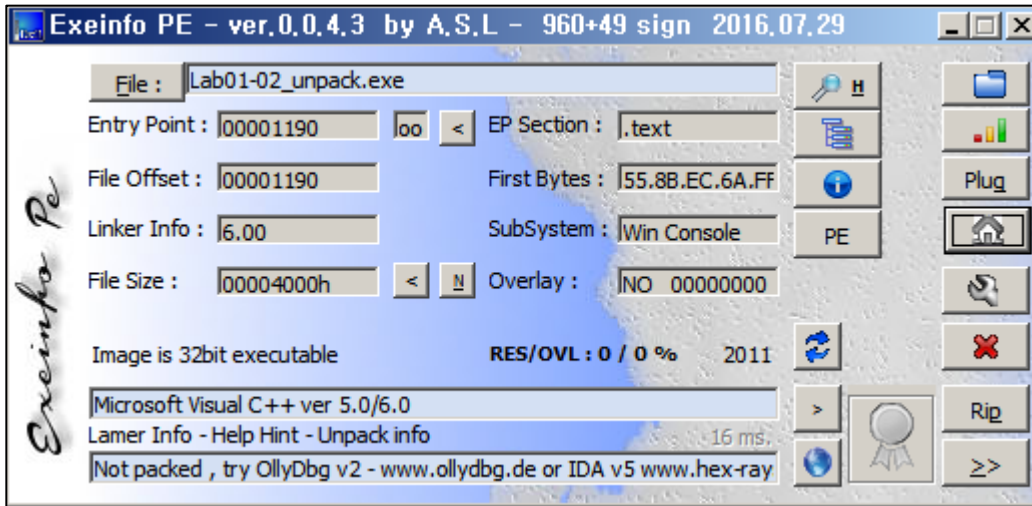


[그림 2-2] upx 명령어를 통한 언패킹

- 명령어: upx -o Lab01-02_unpack.exe -d Lab01-02.exe

1. EXE 파일 정보

■ Lab01-02_unpack.exe 파일 Exeinfo 을 통한 파일 정보 확인

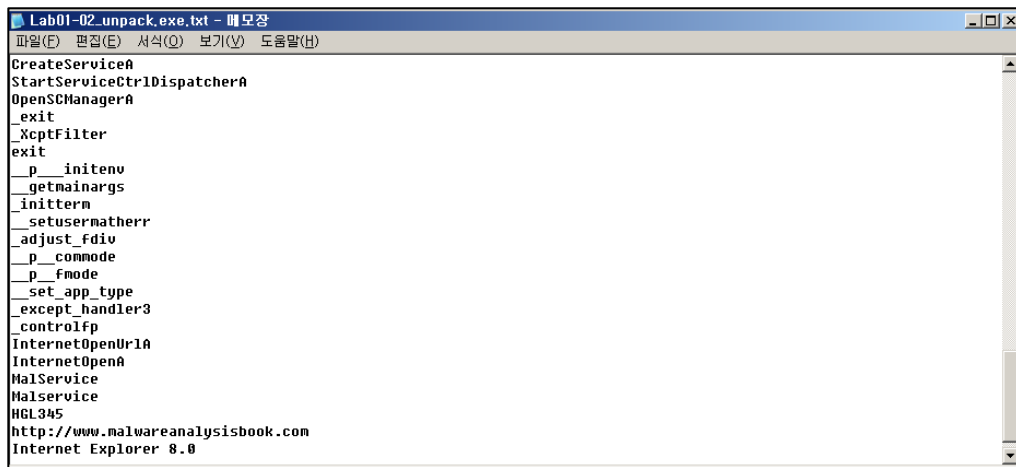


[그림 2-3] Exeinfo PE 에 악성코드를 업로드한 결과

- 패킹(Packing): 실행압축이라는 뜻으로, 실행파일 (PE 파일) 을 압축하였으나, 일반 프로그램처럼 실행 가능, 데이터 보호나 프로그램 크기를 줄이기 등의 목적을 위해 실행
- 언패킹(Unpacking): 패킹된 파일의 압축을 푸는 행위
- 패킹되어 있는 파일인 경우 문자열이 암호화되어 있기 때문에 문자열 분석을 하기 위해서는 언패킹 필요

2. 텍스트 정보

■ strings 명령어 통한 파일 텍스트 정보 확인



```
Lab01-02_unpack.exe.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
CreateServiceA
StartServiceCtrlDispatcherA
OpenSCManagerA
_exit
_XcptFilter
exit
_p__initenv
_getmainargs
_initterm
_setusermatherr
_adjust_fdiv
_p__commode
_p__fmode
_set_app_type
_except_handler3
_controlfp
InternetOpenUrlA
InternetOpenA
MalService
MalService
HGL345
http://www.malwareanalysisbook.com
Internet Explorer 8.0
```

[그림 2-4] strings 명령어를 통해 텍스트 정보 얻은 결과

- 명령어: strings1 Lab01-02_unpack.exe > Lab01-02_unpack.exe.txt
- MalService, Internet Explorer 8.0, <http://www.malwareanalysisbook.com> 등의 인터넷 접속이 의심되는 텍스트 발견

3. PE 구조 분석

■ PVIEW 로 확인한 DLL 함수

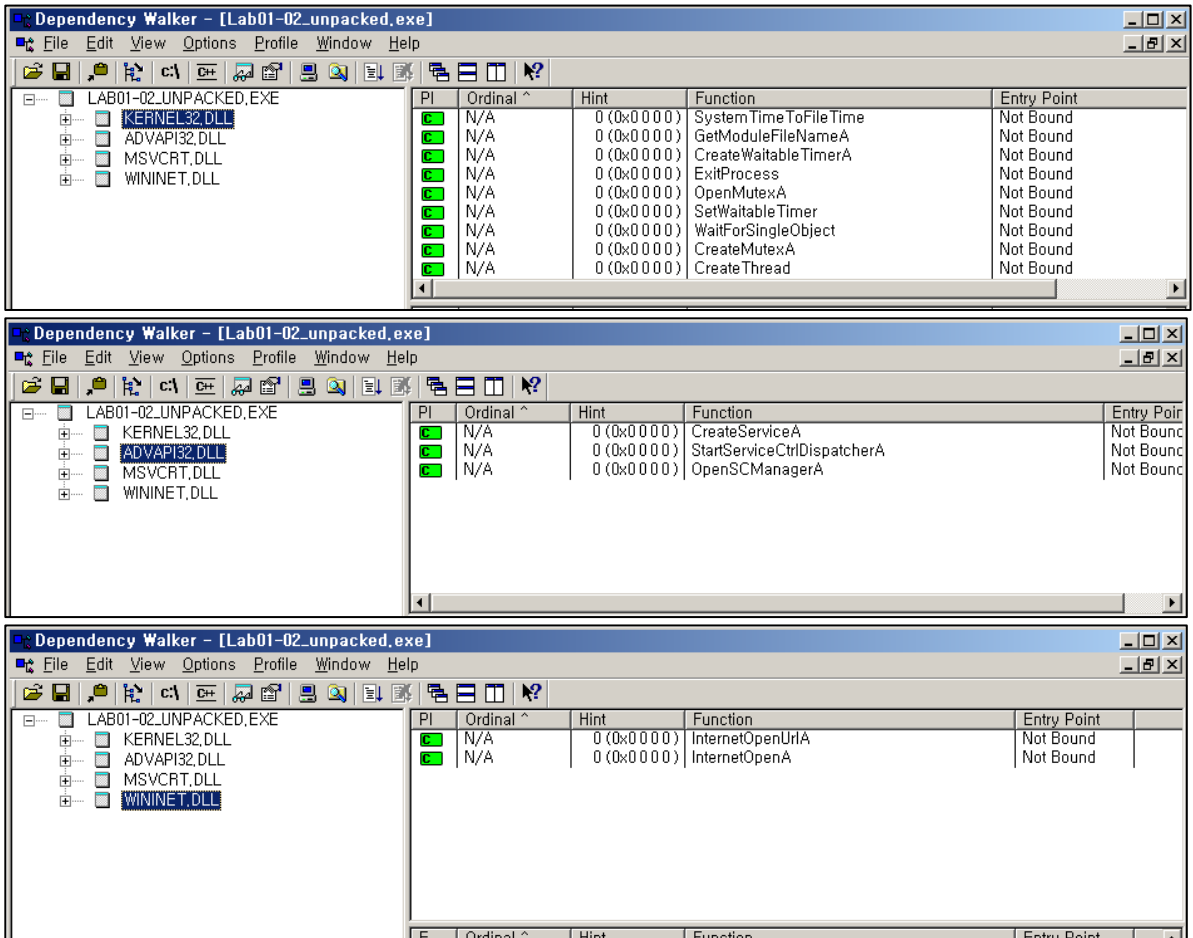
pFile	Data	Description	Value
00002000	0000223C	Hint/Name RVA	0000 CreateServiceA
00002004	0000224C	Hint/Name RVA	0000 StartServiceCtrlDispatcherA
00002008	0000226A	Hint/Name RVA	0000 OpenSCManagerA
0000200C	00000000	End of Imports	ADVAPI32.dll
00002010	0000219E	Hint/Name RVA	0000 SystemTimeToFileTime
00002014	000021B4	Hint/Name RVA	0000 GetModuleFileNameA
00002018	000021C8	Hint/Name RVA	0000 CreateWaitableTimerA
0000201C	000021DE	Hint/Name RVA	0000 ExitProcess
00002020	000021EC	Hint/Name RVA	0000 OpenMutexA
00002024	000021F8	Hint/Name RVA	0000 SetWaitableTimer
00002028	0000220A	Hint/Name RVA	0000 WaitForSingleObject
0000202C	00002220	Hint/Name RVA	0000 CreateMutexA
00002030	0000222E	Hint/Name RVA	0000 CreateThread
00002034	00000000	End of Imports	KERNEL32.dll
00002038	0000227A	Hint/Name RVA	0000 _exit
0000203C	00002282	Hint/Name RVA	0000 _XcptFilter
00002040	00002290	Hint/Name RVA	0000 _exit
00002044	00002296	Hint/Name RVA	0000 _p__initenv
00002048	000022A6	Hint/Name RVA	0000 _getmainargs
0000204C	000022B6	Hint/Name RVA	0000 _initterm
00002050	000022C2	Hint/Name RVA	0000 _setusermatherr
00002054	000022D4	Hint/Name RVA	0000 _adjust_fdiv
00002058	000022E2	Hint/Name RVA	0000 _p__commode
0000205C	000022F0	Hint/Name RVA	0000 _p__fmode
00002060	000022FC	Hint/Name RVA	0000 _set_app_type
00002064	0000230C	Hint/Name RVA	0000 _except_handler3
00002068	0000231E	Hint/Name RVA	0000 _controlfp
0000206C	00000000	End of Imports	MSVCRT.dll
00002070	0000232A	Hint/Name RVA	0000 InternetOpenUrlA
00002074	0000233C	Hint/Name RVA	0000 InternetOpenA
00002078	00000000	End of Imports	WININET.dll

[그림 2-5] IAT(Import Address Table) 에서 확인한 DLL 함수 결과

- PVIEW: PE 구조체를 분석하는 정적분석 툴
- Import Address Table 에서 다음과 같은 DLL 함수 확인 가능 (KERNEL32.dll / MSVCRT.dll / ADVAPI32.dll / WININET.dll)
- Image File Header 에서 생성시기 확인 가능 - 2011/01/19

3. PE 구조 분석

■ Dependency Walker 로 확인한 DLL 함수



[그림 2-6] Dependency Walker 에서 확인한 DLL 함수 결과

- Dependency Walker: DLL 을 분석하는 도구로, 실행파일에 동적으로 링크되는 함수 확인 가능
- 의심스러운 API (ADVAPI32.dll / WININET.dll)
 - * CreateServiceA: 윈도우 서비스 생성
 - * InternetOpenA, InternetOpenUrlA: Winnet 함수 초기화 및 URL 오픈

3. PE 구조 분석

■ 그 외 프로그램에 사용되는 API 검색 (MSDN 검색 결과)

이름	설명
InternetOpenA	응용 프로그램의 Win 사용 초기화
InternetOpenUrlA	FTP 또는 HTTP URL로 지정된 리소스 오픈
CreateMutexA	mutex 개체를 만들거나 오픈
CreateServiceA	서비스개체를 생성하여 지정된 SCM데이터 베이스에 추가
OpenSCManagerA	SCM에 대한 연결을 확립하고 SCM데이터 베이스 오픈
StartServiceCtrlDispatcherA	서비스 프로그램의 주 함수를 SCM에 연결 시키고 컨트롤 디스패처스레드를 시작

[표 2-1] 프로그램에 사용되는 API 설명

Part3. 동적 분석

1. OllyDbg 를 통한 동적 분석

■ 의심 API 분석

Found intermodular calls		
Address	Disassembly	Destination
00401028	CALL DWORD PTR DS: [<&ADVAPI32.StartServ	ADVAPI32.StartServiceCtrlDispatcherA
00401052	CALL DWORD PTR DS: [<&KERNEL32.OpenMutexA	kernel32.OpenMutexA
0040105E	CALL DWORD PTR DS: [<&KERNEL32.ExitProces	kernel32.ExitProcess
0040106E	CALL DWORD PTR DS: [<&KERNEL32.CreateMutex	kernel32.CreateMutexA
0040107A	CALL DWORD PTR DS: [<&ADVAPI32.OpenSCMan	ADVAPI32.OpenSCManagerA
0040108E	CALL DWORD PTR DS: [<&KERNEL32.GetModuleF	kernel32.GetModuleFileNameA
00401086	CALL DWORD PTR DS: [<&ADVAPI32.CreateServ	ADVAPI32.CreateServiceA
004010DF	CALL DWORD PTR DS: [<&KERNEL32.SystemTime	kernel32.SystemTimeToFileTime
004010EB	CALL DWORD PTR DS: [<&KERNEL32.CreateWait	kernel32.CreateWaitableTimerA
00401101	CALL DWORD PTR DS: [<&KERNEL32.SetWaitab	kernel32.SetWaitableTimer
0040110A	CALL DWORD PTR DS: [<&KERNEL32.WaitForSi	kernel32.WaitForSingleObject
0040115F	CALL DWORD PTR DS: [<&WININET.InternetOp	WININET.InternetOpenA (Initial CPU selection)
00401190	PUSH EBP	
004011BC	CALL DWORD PTR DS: [<&MSVCRT.__set_app_ty	MSVCRT.__set_app_type

[그림 3-1] OllyDbg 에서 Search for – All intermodular calls 결과

- OllyDbg: 바이너리 코드 분석을 위한 디버거
- 의심이 갔었던 CreateServiceA, InternetOpenA API 확인 가능

■ CreateServiceA

00401082	. 8D4424 1C	LEA EAX, DWORD PTR SS: [ESP+1C]	
00401086	. 68 E8030000	PUSH 3E8	
00401088	. 50	PUSH EAX	
0040108C	. 6A 00	PUSH 0	
0040108E	. FF15 14204000	CALL DWORD PTR DS: [<&KERNEL32.GetModuleFileNameA]	GetModuleFileNameA
00401094	. 6A 00	PUSH 0	PathBuffer
00401096	. 6A 00	PUSH 0	hModule = NULL
00401098	. 6A 00	PUSH 0	hModule = NULL
0040109A	. 6A 00	PUSH 0	hModule = NULL
0040109C	. 8D4C24 2C	LEA ECX, DWORD PTR SS: [ESP+2C]	hModule = NULL
004010A0	. 6A 00	PUSH 0	hModule = NULL
004010A2	. 51	PUSH ECX	hModule = NULL
004010A3	. 6A 00	PUSH 0	hModule = NULL
004010A5	. 6A 02	PUSH 2	hModule = NULL
004010A7	. 6A 10	PUSH 10	hModule = NULL
004010A9	. 6A 02	PUSH 2	hModule = NULL
004010AB	. 68 1C304000	PUSH Lab01-02.0040301C	hModule = NULL
004010B0	. 68 1C304000	PUSH Lab01-02.0040301C	hModule = NULL
004010B5	. 56	PUSH ESI	hModule = NULL
004010B6	. FF15 00204000	CALL DWORD PTR DS: [<&ADVAPI32.CreateServiceA]	hModule = NULL

[그림 3-2] CreateServiceA 분석

- Malservice 라는 이름으로 서비스를 생성
- StartType: Service_Auto_Start 설정 → 자동 실행
- ServiceType: Service_Win32_Own_Process 설정 → 자체 프로세스로 실행

1. OllyDbg 를 통한 동적 분석

■ InternetOpenA

0040115A	. 68 54304000	PUSH Lab01-02.00403054	ASCII "Internet Explorer 8.0"
0040115F	. FF15 74204000	CALL DWORD PTR DS:[<&WININET	WININET.InternetOpenA
00401165	. 8B3D 70204000	MOV EDI,DWORD PTR DS:[<&WINI	WININET.InternetOpenUrIA
0040116B	. 8BF0	MOV ESI,EAX	
0040116D	> 6A 00	PUSH 0	
0040116F	. 68 00000080	PUSH 80000000	
00401174	. 6A 00	PUSH 0	
00401176	. 6A 00	PUSH 0	
00401178	. 68 30304000	PUSH Lab01-02.00403030	ASCII "http://www.malwareanalysisbook.c
0040117D	. 56	PUSH ESI	
0040117E	. FFD7	CALL EDI	
00401180	. ^EB EB	JMP SHORT Lab01-02.0040116D	

[그림 3-3] InternetOpenA 분석

- Internet Explorer 8.0 으로 세션 연결
- InternetOpenUrlA 으로 <http://www.malwareanalysisbook.com> 연결 시도

※ CreateServiceA 파라미터

파라미터	이름	설명
StartType	Service_Boot_Start(0x00)	시스템 로더에 의해 기동
	Service_System_Start(0x01)	함수에 의해 시작
	Service_Auto_Start(0x02)	자동 시작
	Service_Demand_Start(0x03)	SCM 요청에 의해 시작
	Service_Disabled(0x04)	시작할 수 없는 서비스
ServiceType	Service_Kernel_Driver(0x01)	드라이버 서비스
	Service_File_System_Driver(0x02)	파일 시스템 드라이버
	Service_Adapter(0x04)	예약 완료
	Service_Recognizer_Driver(0x08)	예약 완료
	Service_Win32_Own_Porcess(0x10)	자체 프로세스로 실행

[표 3-1] StartType, ServiceType 서비스 설명

2. IDA Pro 를 통한 동적 분석

■ StartServiceCtrlDispatcher

```
ServiceStartTable= SERVICE_TABLE_ENTRYA ptr -10h
var_8= dword ptr -8
var_4= dword ptr -4
argc= dword ptr 4
argv= dword ptr 8
envp= dword ptr 0Ch

sub     esp, 10h
lea     eax, [esp+10h+ServiceStartTable]
mov     [esp+10h+ServiceStartTable.lpServiceName], offset aMalService ; "MalService"
push    eax                ; lpServiceStartTable
mov     [esp+14h+ServiceStartTable.lpServiceProc], offset sub_401040
mov     [esp+14h+var_8], 0
mov     [esp+14h+var_4], 0
call    ds:StartServiceCtrlDispatcherA
push    0
push    0
call    sub_401040
add     esp, 18h
retn
_main endp
```

[그림 3-4] StartServiceCtrlDispatcher 분석

- ServiceStartTable 구조체를 보면 서비스 이름은 MalService, 주요 서비스는 **sub_401040** 임을 확인 가능
- Main 함수를 보면 StartServiceCtrlDispatcherA 를 호출하는 것을 확인할 수 있으며, 이 파일은 MalService라는 서비스를 생성하여 시스템이 시작 할 때마다 실행이 될 것이라고 유추 가능

2. IDA Pro 를 통한 동적 분석

■ OpenMutexA / CreateMutexA

```
sub_401040 proc near
SystemTime= SYSTEMTIME ptr -400h
FileTime= _FILETIME ptr -3F0h
Filename= byte ptr -3E8h

sub     esp, 400h
push    offset Name      ; "HGL345"
push    0                ; bInheritHandle
push    1F0001h          ; dwDesiredAccess
call    ds:OpenMutexA
test    eax, eax
jz      short loc_401064
```

```
loc_401064:
push    esi
push    offset Name      ; "HGL345"
push    0                ; bInitialOwner
push    0                ; lpMutexAttributes
call    ds:CreateMutexA
```

[그림 3-5] CreateMutexA 분석

- OpenMutexA 함수 호출 후 반환 값이 0 으로 확인 되면 loc_401064 로 점프 하여 HGL345 라는 이름의 뮤텍스 생성
- 시스템에서 악성코드 실행파일이 하나만 동작하게끔 설계 되어 있다고 예측 가능

2. IDA Pro 를 통한 동적 분석

■ OpenSCManagerA / GetModuleFileNameA

```

push    3                ; dwDesiredAccess
push    0                ; lpDatabaseName
push    0                ; lpMachineName
call    ds:OpenSCManagerA
mov     esi, eax
lea     eax, [esp+404h+Filename]
push    3E8h             ; nSize
push    eax              ; lpFilename
push    0                ; hModule
call    ds:GetModuleFileNameA
push    0                ; lpPassword
push    0                ; lpServiceStartName
push    0                ; lpDependencies
push    0                ; lpdwTagId
lea     ecx, [esp+414h+Filename]
push    0                ; lpLoadOrderGroup
push    ecx              ; lpBinaryPathName
push    0                ; dwErrorControl
push    2                ; dwStartType
push    10h              ; dwServiceType
push    2                ; dwDesiredAccess
push    offset DisplayName ; "Malservice"
push    offset DisplayName ; "Malservice"
push    esi              ; hSCManager
call    ds:CreateServiceA
    
```

[그림 3-6] OpenSCManagerA / GetModuleFileNameA API 분석

- OpenSCManagerA: 프로그램이 서비스를 추가하거나 수정할 수 있는 서비스 제어 관리자 핸들 반환
- dwServiceType (0x10) 는 Service_Win32_Own_Process 이며, 자체 프로세스 실행을 의미
- dwStartType (0x02) 는 Service_Auto_Start 이며, 시스템 시작 시 서비스가 자동으로 실행됨을 의미

2. IDA Pro 를 통한 동적 분석

■ SystemTimeToFileTime / CreateWaitableTimeA / SetWaitableTimer / WaitForSingleObject

```

lea     eax, [esp+404h+FileTime]
mov     dword ptr [esp+404h+SystemTime.wYear], edx
lea     ecx, [esp+404h+SystemTime]
mov     dword ptr [esp+404h+SystemTime.wDayOfWeek], edx
push    eax                ; lpFileTime
mov     dword ptr [esp+408h+SystemTime.wHour], edx
push    ecx                ; lpSystemTime
mov     dword ptr [esp+40Ch+SystemTime.wSecond], edx
mov     [esp+40Ch+SystemTime.wYear], 834h
call    ds:SystemTimeToFileTime
push    0                  ; lpTimerName
push    0                  ; bManualReset
push    0                  ; lpTimerAttributes
call    ds:CreateWaitableTimerA
push    0                  ; fResume
push    0                  ; lpArgToCompletionRoutine
push    0                  ; pfnCompletionRoutine
lea     edx, [esp+410h+FileTime]
mov     esi, eax
push    0                  ; lPeriod
push    edx                ; lpDueTime
push    esi                ; hTimer
call    ds:SetWaitableTimer
push    0FFFFFFFFh         ; dwMilliseconds
push    esi                ; hHandle
call    ds:WaitForSingleObject

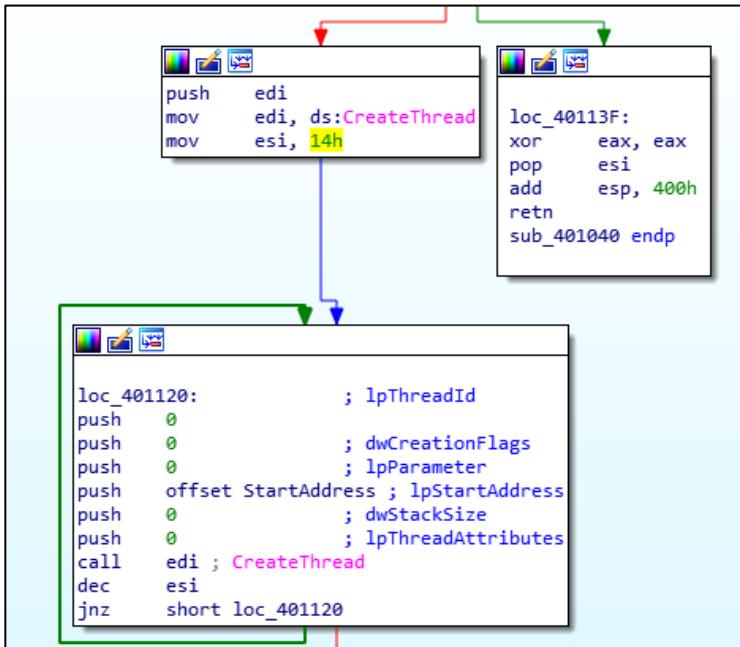
```

[그림 3-7] SystemTimeToFileTime / CreateWaitableTimeA / SetWaitableTimer / WaitForSingleObject 분석

- SystemTime 의 wYear 을 0x834 으로 설정함 → 2100년
- SystemTimeToFileTime: 시스템 타임을 파일 타임 포맷으로 전환을 의미
- CreateWaitableTimeA: waitable 타이머 객체 생성
- SetWaitableTimer: waitable 타이머 객체에 알람 시간 설정
- WaitForSingleObject: 파일 타임에 설정한 대로 2100년 1월 1일 프로그램 재개

2. IDA Pro 를 통한 동적 분석

■ CreateThread



[그림 3-8] CreateThread 분석

- 반복문을 통해 20개의 스레드 생성
- 생성된 스레드는 시작 주소로 lpStartAddress 사용
- 2100년 1월 1일이 되면 실행

2. IDA Pro 를 통한 동적 분석

■ StartAddress



[그림 3-9] StartAddress 분석

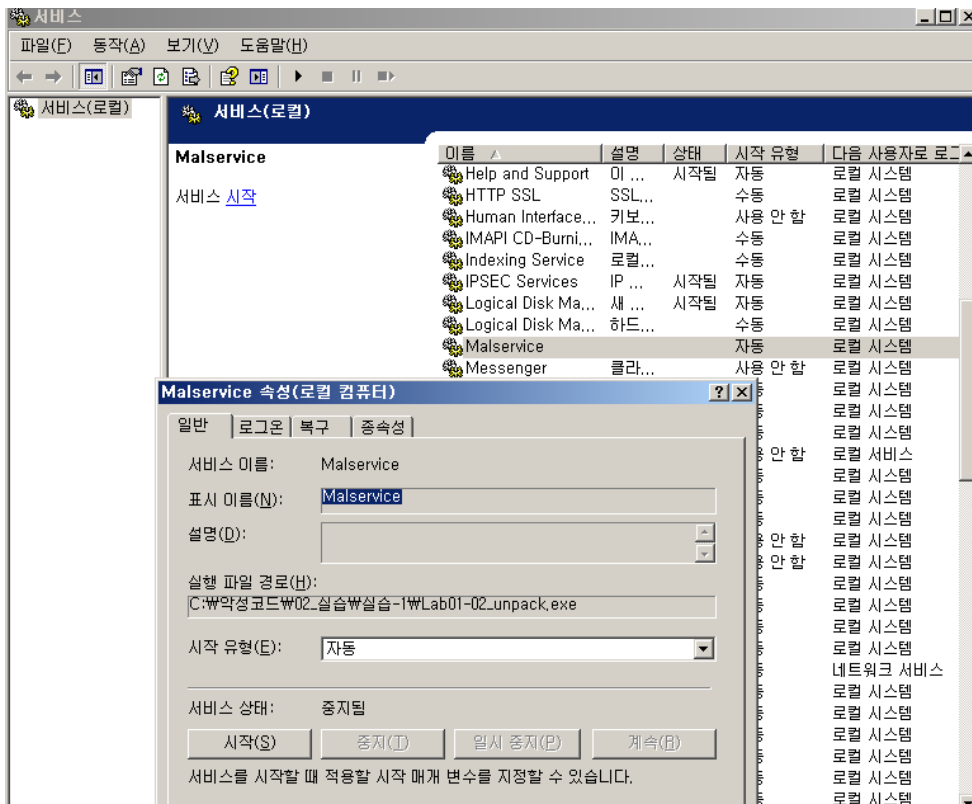
- Internet Explorer 8.0 을 이용하여 <http://www.malwareanalysisbook.com> 을 오픈하는 것을 계속 반복
- 20개의 스레드가 같은 동작을 실행하므로, 트래픽 부하가 일어날 수 있음

Part4. 결론 및 대응 방안

1. 결론

■ 분석을 통한 악성코드 특징

- 패킹이 되어 있어서 언패킹 하지 않으면, 파일의 정보를 알 수 없음
- Malservice 라는 서비스를 만들어 컴퓨터가 재시작할 때 마다 실행
- HGL345 라는 이름의 뮤텍스를 오픈해서 실행 파일 사본이 하나만 동작
- IE 8.0 으로 <http://www.malwareanalysisbook.com> 으로 접속 시도
- 2100년 1월 1일이 되면 20개의 스레드를 동작시킴



[그림 4-1] 서비스에 등록된 Malservice

