‘윈도우 긴급업데이트 설치 패키지’로 위장
‘샌드박스 탐지 우회기법’ 적용

매그니베르 랜섬웨어 변종

요약

1. **불법 다운로드, 광고 사이트로 위장**한 악성코드 유포사이트를 통해 배포
2. **윈도우 긴급업데이트 설치 패키지로 위장** →피해자는 의심없이 다운로드 및 실행
3. 샌드박스 탐지기능 포함 → **APT솔루션으로는 탐지불가**
4. 감염 후 **해커와 연락이 두절되어 데이터 복구조차 불가능**
5. 매그니베르는 케르베르 랜섬웨어의 후속버전으로
   현재까지 매그니베르 자체에서 변형된 8가지 변종 발견/공개

대응 방안

1. Privacy-i EDR과 같은 EDR 솔루션의 **‘행위기반 탐지엔진’으로 취약점 실행 차단**
   : 일반 Anti-Virus 솔루션에서도 대부분 차단 가능하나 최신 업데이트 필요
2. 악성코드 주요 감염경로인 **P2P, 음란, 도박, 불법광고 사이트 연결차단**
3. 메일 내용과 보내는이 계정에 연관성이 없거나, 문법적으로 어색하고,
   신뢰할 수 없는 링크 또는 첨부파일 클릭을 유도하는 메일은 실행 금지
4. 비정상적인 프로세스 행위는 **실시간으로 모니터링**
5. 내부 데이터 보호를 위해 **업무망 망분리 수행**
6. OS 및 소프트웨어 보안 업데이트를 항상 최신형상으로 유지

목차

1. 개요

1.1 배경

매그니베르(Mangniber) 랜섬웨어는 2017년 등장한 'Cerber 랜섬웨어'의 후속 버전이다.
올해 다시 활발히 활동 중인 매그니베르(Magniber) 랜섬웨어는 여덟 가지 변종을 만들어냈다.
이로 인해 안티바이러스 제품이 동작 중임에도 불구하고 감염사례가 발생하고 있으며,
해커와 연락이 닿지 않아 암호화 파일복원이 불가능한 상황도 지속적으로 발생하고 있다.

매그니베르(Magniber) 랜섬웨어는 교묘하게 위장한 불법광고 및 피싱 사이트 등을 활용한다.
이들은 스크립트, 설치파일 또는 제어판 파일형태로 배포되며
긴급 업데이트를 가장한 파일명으로 피해자를 속인다.

[그림 1]매그니베르(Magniber) 랜섬웨어 감염 화면

최근에는 광고 또는 불법 사이트로 위장한 사이트에서 긴급 업데이트 MSI 설치파일형태로 유포됐다.
매그니베르(Magniber) 랜섬웨어는 복수 프로세스에 무차별 인젝션을 수행하여 피해자 PC를 암호화하고 있다.

본 보고서는 매그니베르(Magniber) 랜섬웨어 주요행위 동작방식에 대한 분석내용과 함께
Privacy-i EDR 솔루션의 대응정보를 담고 있다.

2. 정보

2.1 파일 정보

Name	SYSTEM.Critical.Upgrade.Win10.0.eaa89fa55d9e.msi
Type	Windows MSI 설치 파일
Behavior	Ransomware
SHA-256	a1c813e83640de7ba811264cf980cfd258c884415171b45175cf481a3a944834

[표 1] MSI 형태의 Magniber 랜섬웨어 변종

Name	SYSTEM.Critical.Upgrade.Win10.0.a6fbae094d484.wsf
Type	Windows WSF 스크립트 파일
Behavior	Ransomware
SHA-256	1719cf6341b7ef28d39ec21c046b0a7adaad97add8622831a5a16f96651f1c5a

[표 2] WSF 형태의 Magniber 랜섬웨어 변종

Name	Antivirus_Upgrade_Cloud.a57a068b68ae8ed.jse
Type	Windows JSE 스크립트 파일
Behavior	Ransomware
SHA-256	f41ec94f9d0c7480df2196b3fc5493599d50de222d2c903b173db3e7caff8747

[표 3] JSE 형태의 Magniber 랜섬웨어 변종

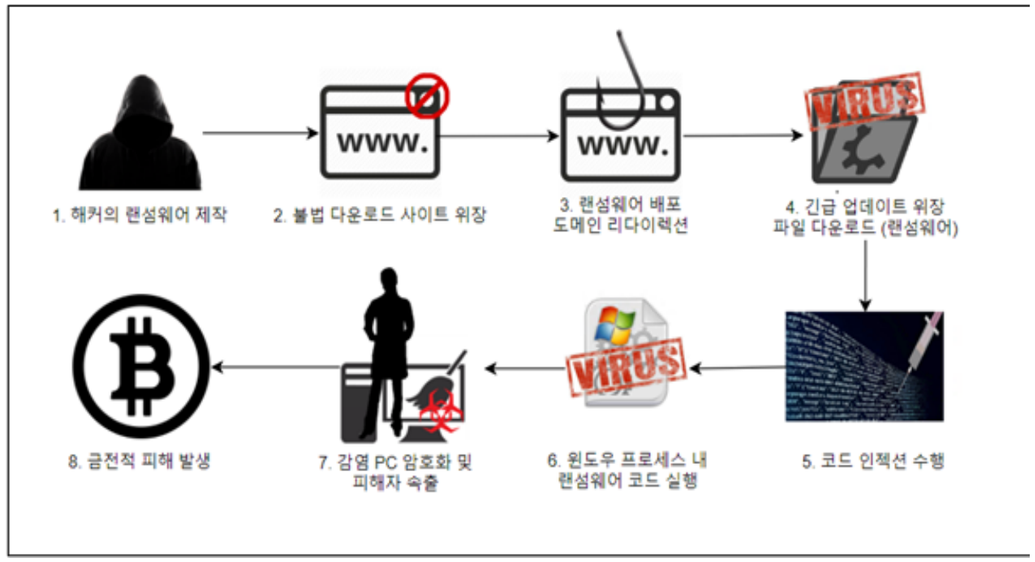
Name	Antivirus.Upgrade.Database.Cloud.js
Type	Windows JS 스크립트 파일
Behavior	Ransomware
SHA-256	831f88fcd634385833fe84b4e4d88d03432f255a818e8d353c0c4de0a0f8ead4

[표 4] JS 형태의 Magniber 랜섬웨어 변종

Name	MS.Upgrade.Database.Cloud.cpl
Type	Windows CPL 제어판 파일
Behavior	Ransomware
SHA-256	0e2ceee00815b899f750fd5013b3b839c6d62a946b6b305afc19dda85f6f6e52

[표 5] CPL 형태의 Magniber 랜섬웨어 변종

2.2 Magniber 랜섬웨어 정보

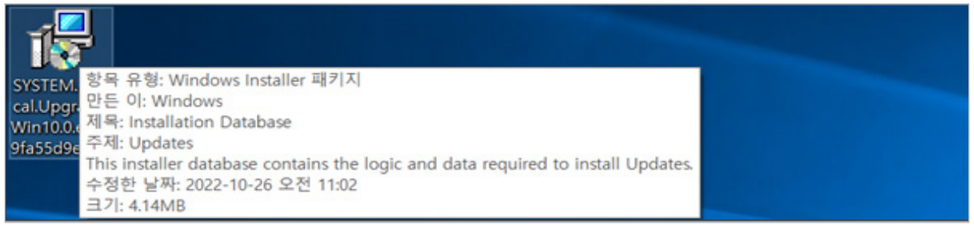


[그림 2] 매그니베르(Magniber) 랜섬웨어 공격흐름

①~②	**해커의 랜섬웨어 제작 및 유포** 제작한 랜섬웨어는 불법 다운로드 또는 광고 사이트로 위장한 유포 사이트를 통해 배포된다.
③~④	**도메인 리다이렉션** 유포 사이트에 접속한 피해자들은 긴급 업데이트로 위장한 랜섬웨어 파일을 다운로드한다.
⑤	**코드 인젝션 수행** 긴급 업데이트 파일로 위장한 랜섬웨어 악성코드는 피해자들에 의해 직접 실행되어 다른 정상 윈도우 프로세스에 코드 인젝션을 수행한다.
⑥	**윈도우 프로세스를 통한 랜섬웨어 감염** 정상 윈도우 프로세스에 인젝션된 코드를 실행하여 랜섬웨어의 암호화 행위를 수행한다.
⑦~⑧	**피해자 속출 및 금전적 피해 발생** 무차별적인 감염을 유발하는 매그니베르(Magniber) 랜섬웨어로 인한 피해가 발생한다.

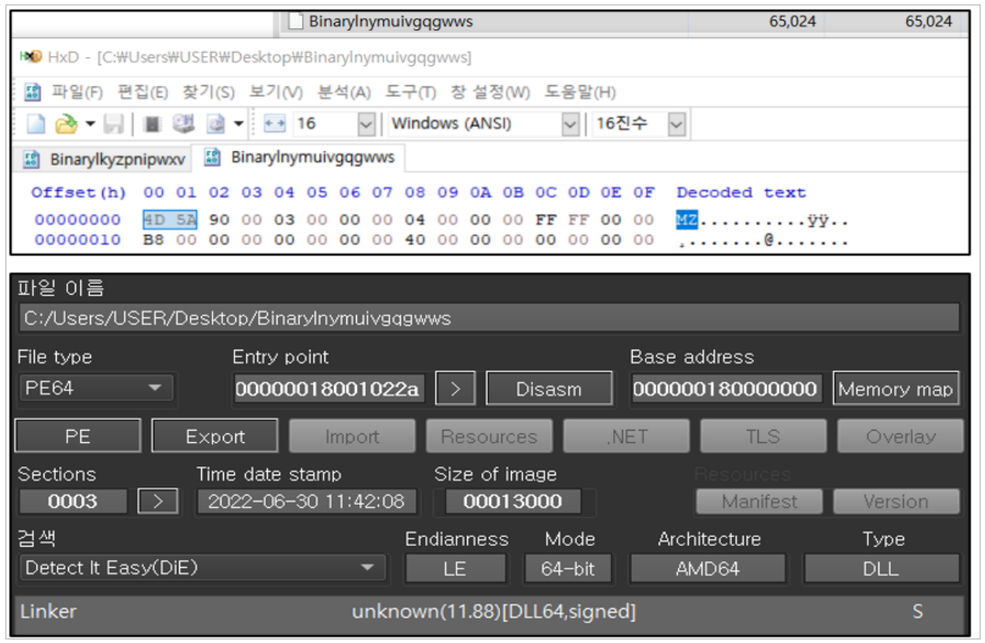
3. 분석

3.1 위장 및 탐지 우회



[그림 3] 매그니베르(Magniber) 랜섬웨어의 위장

본 보고서에서 분석한 매그니베르(Magniber) 랜섬웨어 변종은
사용자가 의심없이 실행하도록 '윈도우 긴급 업데이트 설치 패키지 형태'로 위장했다.



[그림 4] 매그니베르(Magniber) 랜섬웨어의 탐지 우회

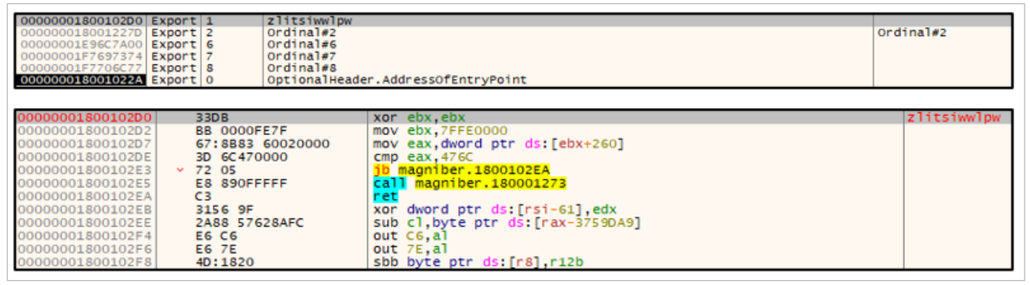
윈도우 설치 패키지로 위장한 파일 내부를 확인하면,
매그니베르(Magniber) 랜섬웨어가 DLL 형태로 포함되어 있는 것을 확인할 수 있다.
파일을 실행하면 위처럼 DLL 형태의 매그니베르(Magniber) 랜섬웨어가 실행된다.

3.2 윈도우 프로세스를 통한 실행 (MSI Installer)

[그림 5] MSI Installer를 통해 1차 실행을 수행한 매그니베르(Magniber) 랜섬웨어

윈도우 설치 패키지 형태로 위장한 매그니베르(Magniber) 랜섬웨어는
윈도우 인스톨러인 Msiexec.exe 유틸리티를 통해 실행된다.
해당 과정이 수행되면 랜섬웨어 DLL이 실행된다.

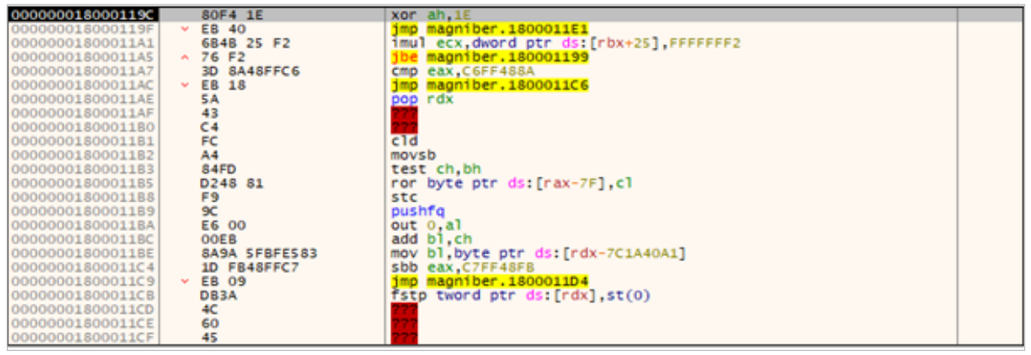
3.3 Export 함수를 통한 실행 방지



[그림 6] MSI Installer를 통해 1차 실행을 수행한 매그니베르(Magniber) 랜섬웨어

매그니베르(Magniber) 랜섬웨어 DLL은 MSI Installer를 통해 실행된다.
이 외의 경우 Export 함수를 통해 암호화행위 실행을 차단한다.
즉, MSI Installer에서 실행되지 않으면 보안솔루션의 탐지를 피할 수 있다.
또한 해당 Export 함수 이름은 버전 별로 상이하다. 이는 시그니처 기반 탐지를 우회하기 위함이다.

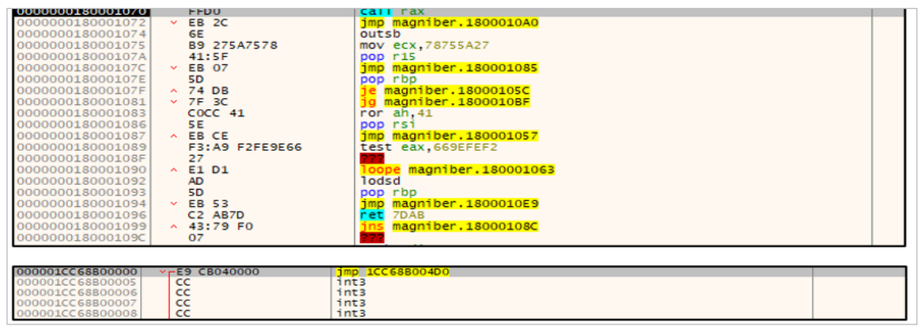
3.4 코드 난독화



[그림 7] 코드 난독화

매그니베르(Magniber) 랜섬웨어의 코드는 쉘코드와 어셈블리 코드를 이용한 난독화로 이루어져 있다.
코드 난독화는 코드의 패턴 등을 통해 악성코드를 판단하는 휴리스틱 기반의 안티바이러스를 우회한다.
이를 통해 악성코드 분석가의 탐지 분석에 어려움을 준다.

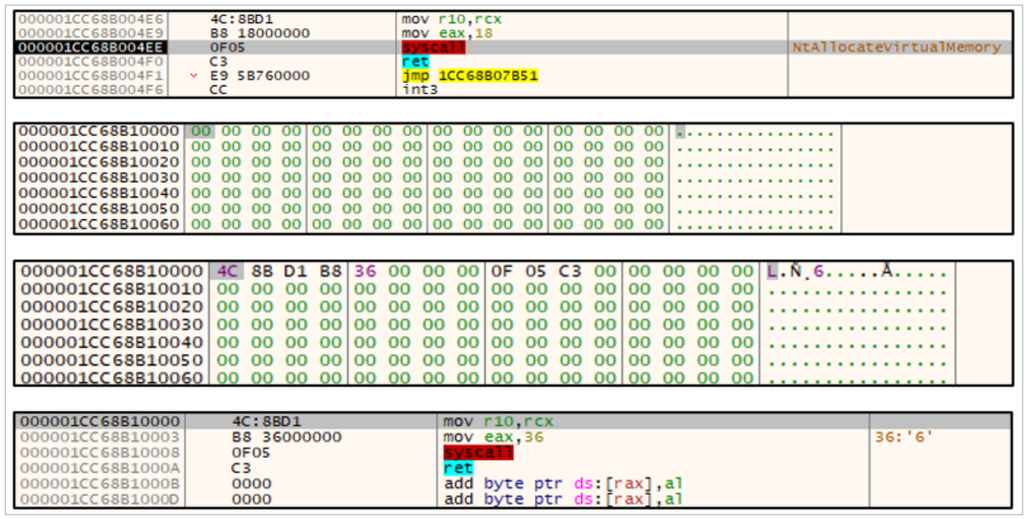
3.5 코드 난독화 해제



[그림 8] 코드 난독화 해제

매그니베르(Magniber) 랜섬웨어의 코드는 난독화가 해제되며,
실제 악성 행위를 수행하는 메모리 주소로 이동하는 코드가 호출된다.
JMP 명령을 수행하는 메모리 주소를 RAX 레지스터 내에 담고,
이를 호출하여 실제 악성 행위를 수행하는 JMP 명령을 수행하게 된다.

3.6 임의의 코드 실행을 위한 메모리 할당



[그림 9] 코드 실행을 위한 메모리 할당

NtAllocateVirtualMemory API를 호출하고 공간을 할당한 후,
0x000001CC68B10000 메모리 주소에 코드를 적재한다.
해당 주소에는 이후 임의의 코드 실행을 위한 SYSCALL 명령을 확인할 수 있다.
이를 통해 Native API를 직접 실행할 수 있게 되어,
API 후킹을 기반으로 악성코드를 탐지하는 보안 제품을 우회할 수 있다.

3.7 프로세스 목록 획득

```
000001CC68B10000   4C:8BD1          mov r10,rcx
000001CC68B10003   B8 36000000      mov eax,36                  36:'6'
000001CC68B10008   0F05             syscall                     NtQuerySystemInformation
000001CC68B1000A   C3               ret
000001CC68B1000B   0000             add byte ptr ds:[rax],al
000001CC68B1000D   0000             add byte ptr ds:[rax],al
000001CC68B1000F   0000             add byte ptr ds:[rax],al
000001CC68B10011   0000             add byte ptr ds:[rax],al
```

```
typedef struct _SYSTEM_PROCESSES
{ // System Information Class 5

    ULONG           NextEntryDelta;
    ULONG           ThreadCount;
    ULONG           Reserved1[6];
    LARGE_INTEGER   CreateTime;
    LARGE_INTEGER   UserTime;
    LARGE_INTEGER   KernelTime;
    UNICODE_STRING  ProcessName;
    ULONG           BasePriority;
    ULONG           ProcessId;
    ULONG           InheritedFromProcessId;
    ULONG           HandleCount;
    ULONG           Reserved2[2];
} SYSTEM_PROCESSES, * PSYSTEM_PROCESSES;
```

[그림 10] 프로세스 목록 획득

NtQuerySystemInformation API를 이전의 임의 코드 실행을 위해 메모리 할당했던 주소에서 실행한다.
이 때, 인자로 SystemProcessInformation(0x5)를 적재하여 호출하는데,
이를 통해 현재 실행중인 프로세스 목록을 확인할 수 있다.
이는 이후 프로세스들의 핸들 획득에 사용된다.

3.8 유휴 시간 대기 및 프로세스 핸들 획득

```
000001CC68B60000   4C:8BD1          mov r10,rcx
000001CC68B60003   B8 34000000      mov eax,34                  34:'4'
000001CC68B60008   0F05             syscall                     NtDelayExecution
000001CC68B6000A   C3               ret
000001CC68B6000B   0000             add byte ptr ds:[rax],al
000001CC68B6000D   0000             add byte ptr ds:[rax],al
000001CC68B6000F   0000             add byte ptr ds:[rax],al
000001CC68B60011   0000             add byte ptr ds:[rax],al
000001CC68B60013   0000             add byte ptr ds:[rax],al

000001CC68B70000   4C:8BD1          mov r10,rcx
000001CC68B70003   B8 26000000      mov eax,26                  26:'&'
000001CC68B70008   0F05             syscall                     NtOpenProcess
000001CC68B7000A   C3               ret
000001CC68B7000B   0000             add byte ptr ds:[rax],al
000001CC68B7000D   0000             add byte ptr ds:[rax],al
000001CC68B7000F   0000             add byte ptr ds:[rax],al
```

[그림 11] 유휴 시간 대기 및 프로세스 핸들 획득

NtDelayExecution API를 호출하여 잠시 유휴시간을 주어 실행을 대기한다.
이는 샌드박스 기반 탐지를 회피하기 위함으로,
일정시간 동안 움직임을 정지하여 샌드박스에서 탐지되지 않도록 우회하는 기법이다.
이와 더불어 NtOpenProcess API를 호출하여 현재 실행중인 프로세스들에 대해 핸들을 획득한다.
이는 추후 랜섬 행위를 수행하는 코드를 인젝션하기 위해
접근이 가능한 프로세스 목록을 획득하는 행위이다.

3.9 대상 프로세스 정보 획득

```
000001CC6AD50000   4C:8BD1          mov r10,rcx
000001CC6AD50003   B8 19000000      mov eax,19
000001CC6AD50008   0F05             syscall                     NtQueryInformationProcess
000001CC6AD5000A   C3               ret
000001CC6AD5000B   0000             add byte ptr ds:[rax],al
000001CC6AD5000D   0000             add byte ptr ds:[rax],al
000001CC6AD5000F   0000             add byte ptr ds:[rax],al
000001CC6AD50011   0000             add byte ptr ds:[rax],al
000001CC6AD50013   0000             add byte ptr ds:[rax],al
000001CC6AD50015   0000             add byte ptr ds:[rax],al
000001CC6AD50017   0000             add byte ptr ds:[rax],al
000001CC6AD50019   0000             add byte ptr ds:[rax],al
```

[그림 12] 대상 프로세스 정보 획득

NtQueryInformationProcess API를 ProcessWow64Information(0x1A)를 인자로 주어 실행한다.
이를 통해 대상 프로세스가 WOW64 환경에서 실행중인지 여부를 확인한다.

3.10 코드 인젝션과 탐지 회피

```
000001CC68B004E6   4C:8BD1          mov r10,rcx
000001CC68B004E9   B8 18000000      mov eax,18
000001CC68B004EE   0F05             syscall                     NtAllocateVirtualMemory
000001CC68B004F0   C3               ret
000001CC68B004F1   E9 5B760000      jmp 1CC68B07B51
000001CC68B004F6   CC               int3

000001CC6AD80000   4C:8BD1          mov r10,rcx
000001CC6AD80003   B8 3A000000      mov eax,3A                  3A:':'
000001CC6AD80008   0F05             syscall                     NtWriteVirtualMemory
000001CC6AD8000A   C3               ret
000001CC6AD8000B   0000             add byte ptr ds:[rax],al
000001CC6AD8000D   0000             add byte ptr ds:[rax],al
000001CC6AD8000F   0000             add byte ptr ds:[rax],al
000001CC6AD80011   0000             add byte ptr ds:[rax],al
000001CC6AD80013   0000             add byte ptr ds:[rax],al

000001CC68B004F1   E9 5B 76 00 00 CC CC CC CC CC CC CC CC CC CC CC   é[v..ÌÌÌÌÌÌÌÌÌÌ
000001CC68B00501   48 89 5C 24 08 48 89 6C 24 10 48 89 74 24 18 48   H.\$.H.l$.H.t$.H
000001CC68B00511   89 7C 24 20 65 48 8B 04 25 60 00 00 00 8B E9 48   .|$ eH..%`....éH
000001CC68B00521   8B 50 18 48 8B 72 10 4C 8B 5E 30 4D 85 DB 0F 84   .P.H.r.L.^0M.Û..
000001CC68B00531   BB 00 00 00 49 63 43 3C 48 8B 56 58 48 8B 4E 60   »...IcC<H.VXH.N`
000001CC68B00541   46 8B 8C 18 88 00 00 00 48 8B 36 45 33 C0 45 85   F.......H.6E3ÀE.
000001CC68B00551   C9 0F 84 8B 00 00 00 48 C1 EA 10 0F B7 C2 85 C0   É.....HÁê..·Â.À
000001CC68B00561   74 2B 8B D0 66 66 66 66 0F 1F 84 00 00 00 00 00   t+.Ðffff........
```

[그림 13] 코드 인젝션과 탐지 회피

이전의 WOW64 환경 여부에 따라 대상 프로세스에 대해 NtAllocateVirtualMemory API를 호출하여
대상 프로세스 내 메모리를 할당하고, NtWriteVirtualMemory API를 호출하여 코드를 인젝션한다.
이 때, 대상 프로세스에 인젝션 되는 코드는 랜섬 행위를 수행하는 코드이다.

3.11 메모리 속성 변경

```
000001CC6ADA0000   4C:8BD1          mov r10,rcx
000001CC6ADA0003   B8 50000000      mov eax,50                  50:'P'
000001CC6ADA0008   0F05             syscall                     NtProtectVirtualMemory
000001CC6ADA000A   C3               ret
000001CC6ADA000B   0000             add byte ptr ds:[rax],al
000001CC6ADA000D   0000             add byte ptr ds:[rax],al
```

[그림 14] 메모리 속성 변경

이전의 코드 삽입 행위 후, 해당 코드를 실행할 수 있도록
NtProtectVirtualMemory API를 호출하여 메모리 속성을 변경시켜준다.
이 때 변경하는 속성은 PAGE_EXECUTE_READWRITE(0x40)이 사용된다.
이를 통해 해당 메모리에 인젝션을 시킨 랜섬 행위 코드를 실행할 수 있도록 변경한다.

3.12 원격 스레드 생성 및 실행

```
000001CC6ADC0000   4C:8BD1          mov r10,rcx
000001CC6ADC0003   B8 C1000000      mov eax,C1
000001CC6ADC0008   0F05             syscall                     NtCreateThreadEx
000001CC6ADC000A   C3               ret
000001CC6ADC000B   0000             add byte ptr ds:[rax],al
000001CC6ADC000D   0000             add byte ptr ds:[rax],al
000001CC6ADC000F   0000             add byte ptr ds:[rax],al
000001CC6ADC0011   0000             add byte ptr ds:[rax],al
000001CC6ADC0013   0000             add byte ptr ds:[rax],al
000001CC6ADC0015   0000             add byte ptr ds:[rax],al

000001CC6ADE0000   4C:8BD1          mov r10,rcx
000001CC6ADE0003   B8 52000000      mov eax,52                  52:'R'
000001CC6ADE0008   0F05             syscall                     NtResumeThread
000001CC6ADE000A   C3               ret
000001CC6ADE000B   0000             add byte ptr ds:[rax],al
000001CC6ADE000D   0000             add byte ptr ds:[rax],al
```

[그림 15] 원격 스레드 생성 및 실행

코드를 삽입하고 메모리 속성을 변경한 뒤, NtCreateThreadEx API를 호출하여 원격 스레드를 생성한다.
해당 스레드는 삽입된 코드를 실행시켜주는 매개로서,
코드가 삽입된 대상 프로세스가 랜섬 행위를 하는 랜섬웨어로서 동작하게 한다.
이를 위해 NtResumeThread API를 호출하여 원격 스레드를 실행한다.

3.13 랜섬 행위 수행

[그림 16] 원격 스레드에 의한 랜섬 행위 수행

빠른 암호화 행위를 수행하여 기존 보안 제품의 탐지를 우회하기위해
PC에서 실행 중인 다수의 정상 프로세스에 코드 인젝션을 하고
원격 스레드를 실행하여 랜섬행위를 수행한다.
실제 암호화가 시작된 후 모든 행위가 끝나고 바탕화면 변경까지는 긴 시간이 필요하지 않으며,
정상적인 윈도우 프로세스에서 수행되므로 보안 제품은 탐지가 어렵다.

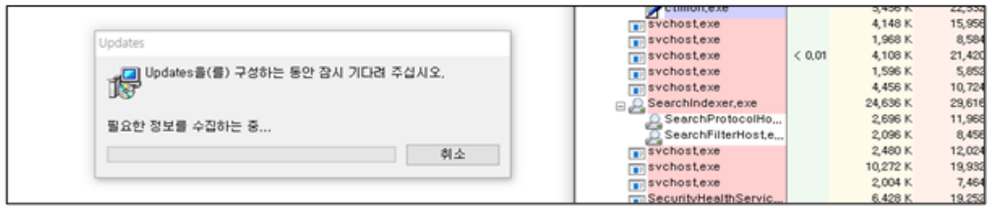
4. Privacy-i EDR 대응

4.1 취약점 공격 방지 기능을 통한 탐지 및 차단

[그림 17] 원격 스레드에 의한 랜섬 행위 수행

본 보고서에서 서술한 매그니베르(Magniber) 랜섬웨어의 프로세스 인젝션 행위는
Privacy-i EDR의 취약점 공격방지 기능으로 탐지 및 차단할 수 있다.

4.2 매그니베르(Magniber) 변종 대응



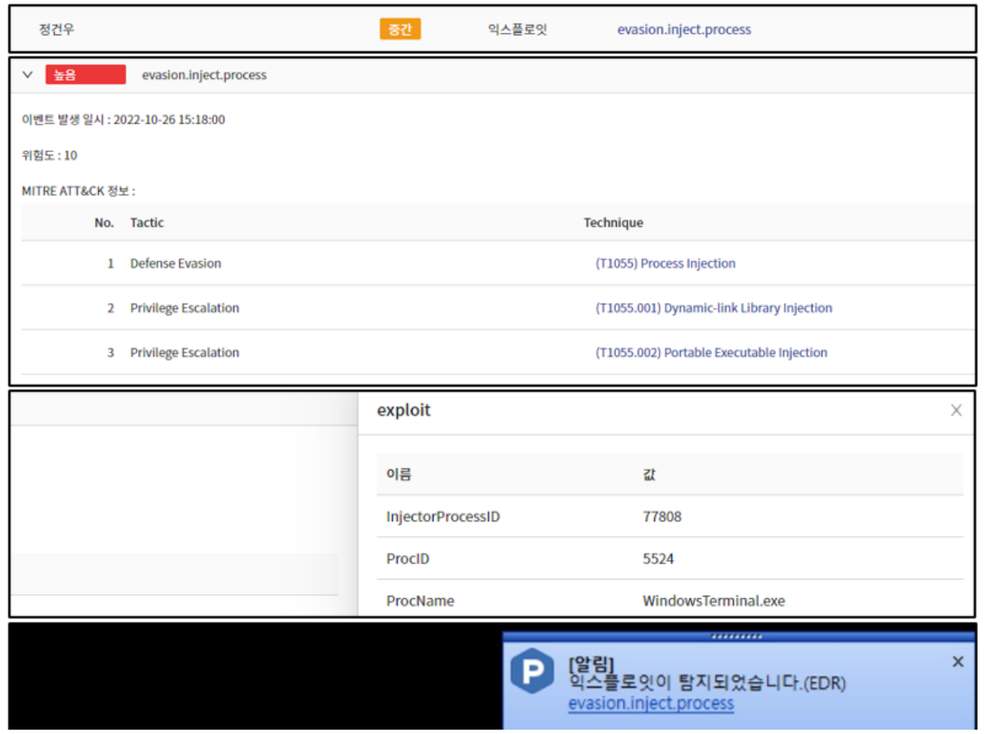
[그림 18] 인젝션 행위를 차단하여 인스톨러의 진행이 중단된 모습

매그니베르(Magniber) 랜섬웨어는 MSI, WSF, JS, JSE, CPL 등의 다양한 형식으로 유포되고 있다.
변종들의 주요 행위는 인젝션 후 랜섬 행위를 수행하는 것인데,
해당 악성 행위는 Privacy-i EDR의 취약점 공격방지기능으로 모두 탐지 및 차단이 가능하다.

[그림 19] 매그니베르(Magniber) 랜섬웨어 변종

5. Privacy-i EDR 탐지 정보

5.1 탐지 정보 (evasion.inject.process)



[그림 20] Privacy-i EDR 랜섬웨어 탐지 정보

프로세스의 취약점 실시간 분석을 통해 매그니베르(Magniber) 랜섬웨어를 탐지하였으며,
대상 프로세스에 코드 인젝션을 수행하는 익스플로잇 행위를 파악하고 차단하였다.

6. 대응

1. Privacy-i EDR과 같은 EDR 솔루션의 **'행위기반 탐지엔진'으로 취약점 실행 차단**
   : 일반 Anti-Virus 솔루션에서도 대부분 차단 가능하나 최신 업데이트 필요
2. 악성코드 주요 감염경로인 **P2P, 음란, 도박, 불법광고 사이트 연결차단**
3. 메일 내용과 보내는이 계정에 연관성이 없거나, 문법적으로 어색하고,
   신뢰할 수 없는 링크 또는 첨부파일 클릭을 유도하는 메일은 실행 금지
4. 비정상적인 프로세스 행위는 **실시간으로 모니터링**
5. 내부 데이터 보호를 위해 **업무망 망분리 수행**
6. OS 및 소프트웨어 보안 업데이트를 항상 최신형상으로 유지

궁금하신 점이나 문의사항은 malware@somansa.com 으로 문의주십시오