윈도우 소프트웨어 버그헌팅

취약점 분석 팁을 곁들인

화이트햇 스쿨 1기 강찬송, 김민서

POC SECURITY

목차

01

프로젝트 소개

02

취약점 분석 방법론

03

취약점 분석 과정

04

취약점 제보 절차

POC SECURITY

발표자 소개

강찬송

화이트햇 스쿨 1기 수료생

명지전문대학 졸업

김민서

화이트햇 스쿨 1기 수료생

서울대학교 3학년 재학

프로젝트의 필요성

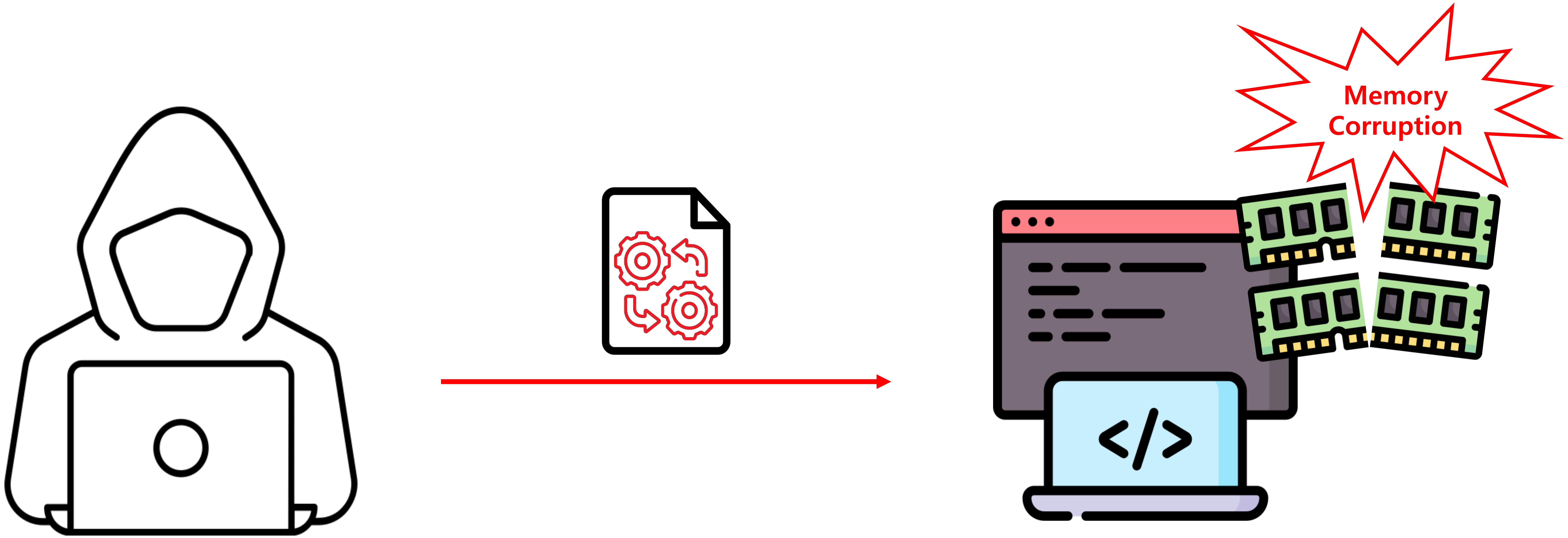
지식 향상

- 취약점 탐색 능력 강화
- 리버싱, 시스템 관련 지식 증진

사회 기여

- 보안 문제에 대한 인식을 강화하는 데 기여
- 궁극적으로 기업과 개인 사용자 모두에게 더욱 강력하고 안전한 디지털 환경을 구축하는데 기여

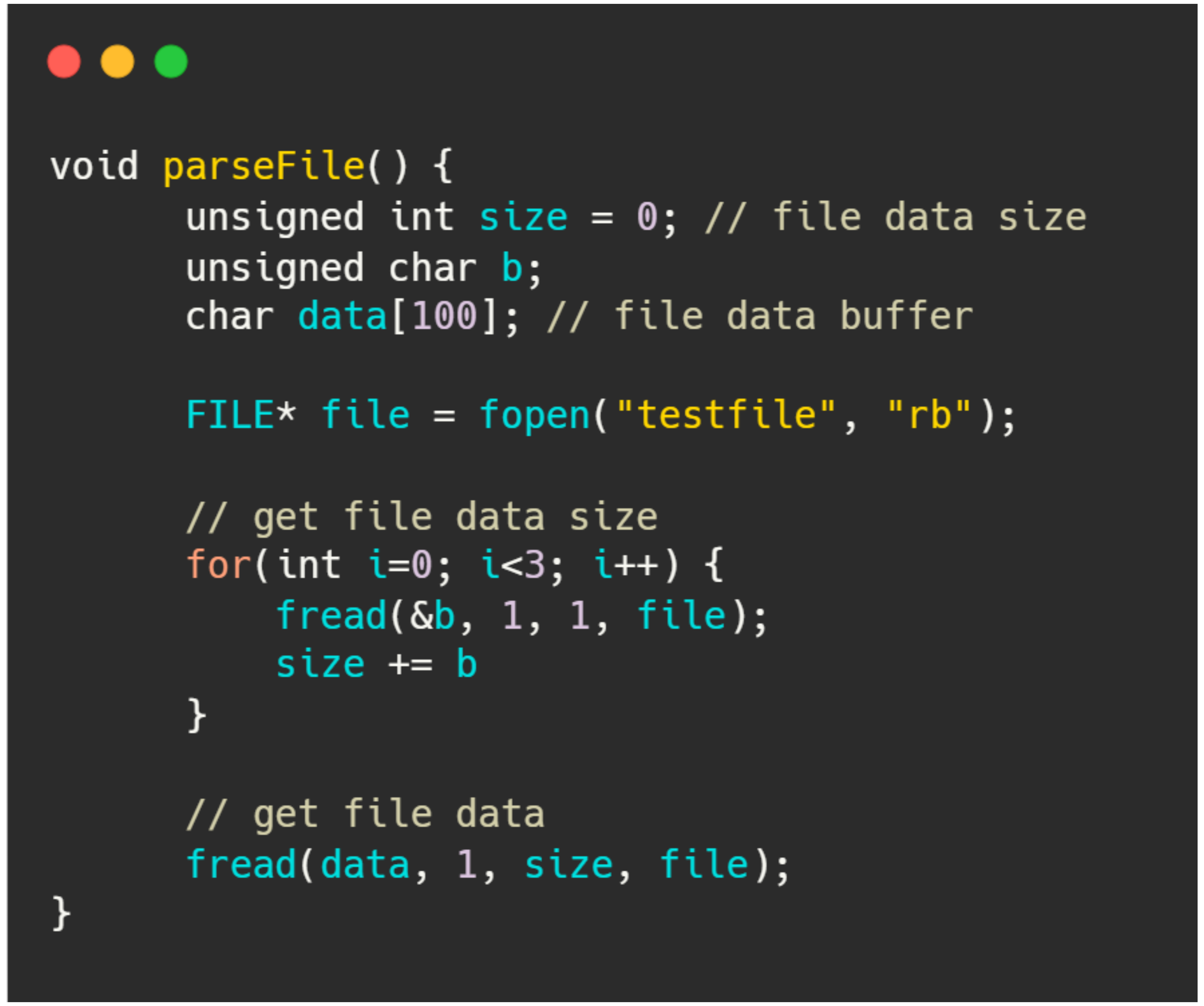
프로젝트 분석 대상



- 상용 소프트웨어 실행 시 메모리에서 발생할 수 있는 메모리 커럽션 취약점이 탐색 대상
- Stack overflow, Heap overflow, Out-of-bounds read/write, Integer overflow, Integer issue 등의 취약점

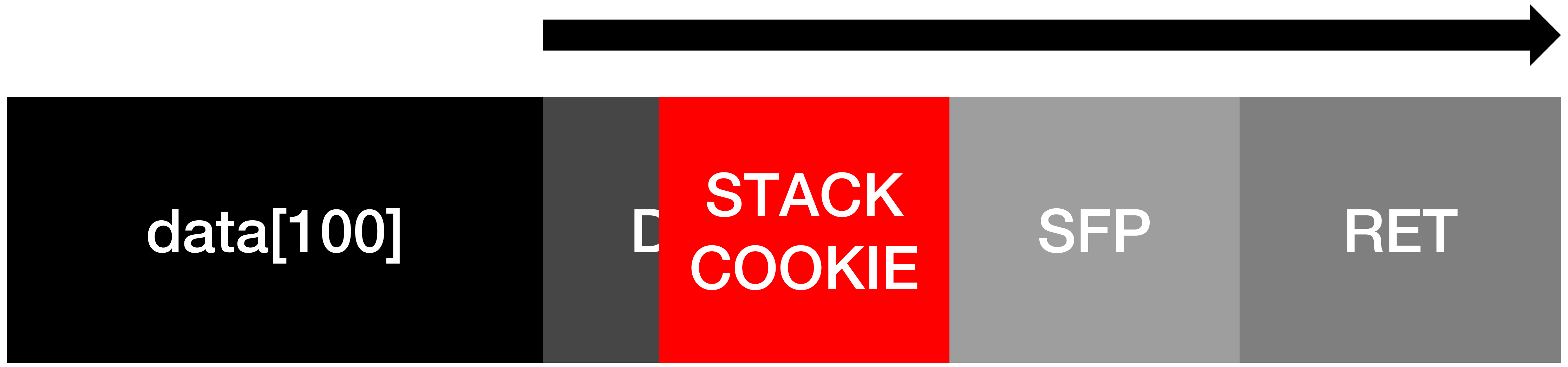
취약점 유형 1

Buffer Overflow



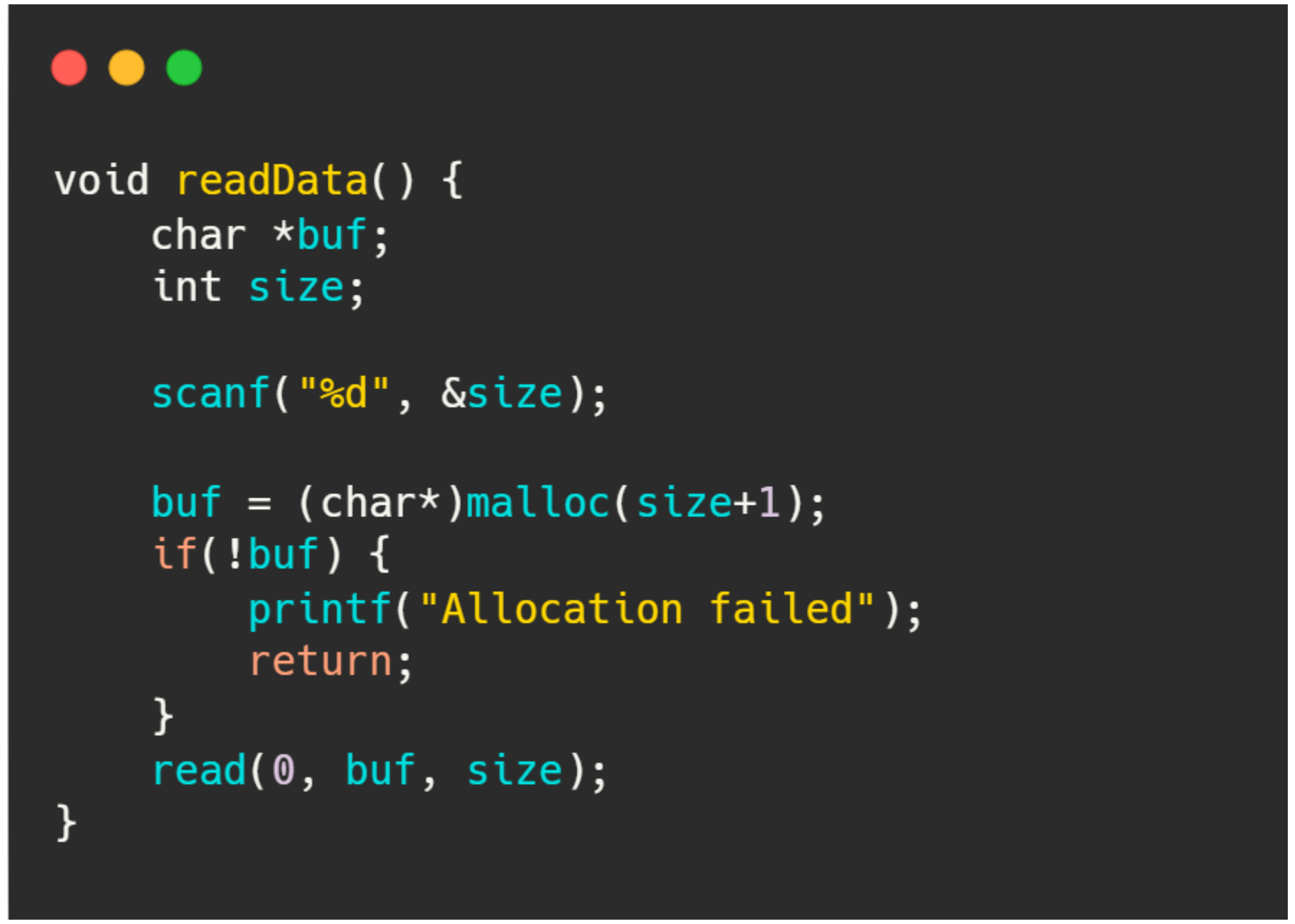
```
void parseFile() {
        unsigned int size = 0; // file data size
        unsigned char b;
        char data[100]; // file data buffer

        FILE* file = fopen("testfile", "rb");

        // get file data size
        for(int i=0; i<3; i++) {
            fread(&b, 1, 1, file);
            size += b
        }

        // get file data
        fread(data, 1, size, file);
}
```

- 데이터의 길이에 대한 불명확한 정의
- Stack / Heap 영역에서의 덮어쓰기
- strcpy, scanf, fread …

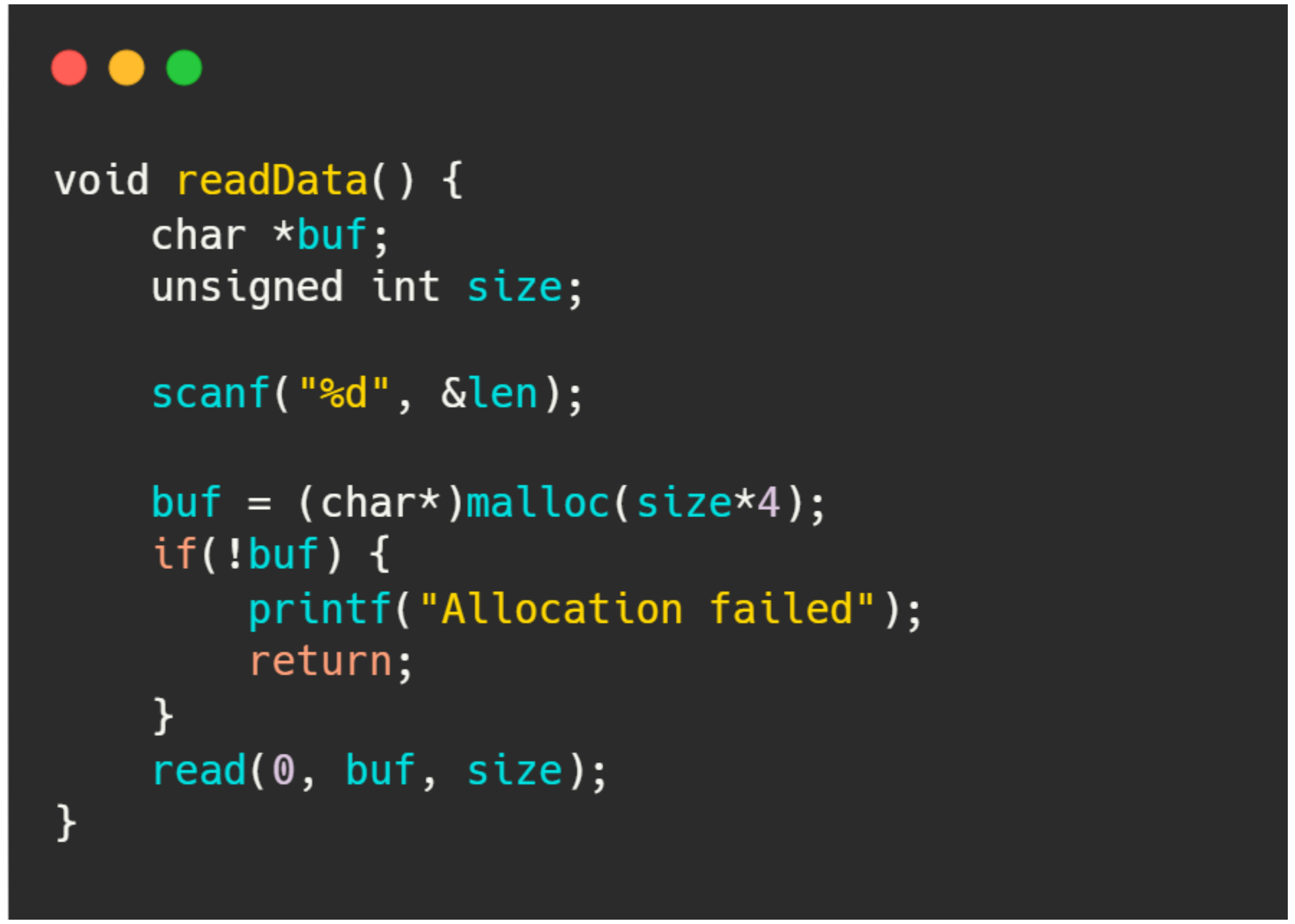


취약점 유형 2

Integer Issue



```
void readData() {
    char *buf;
    int size;

    scanf("%d", &size);

    buf = (char*)malloc(size+1);
    if(!buf) {
        printf("Allocation failed");
        return;
    }
    read(0, buf, size);
}
```



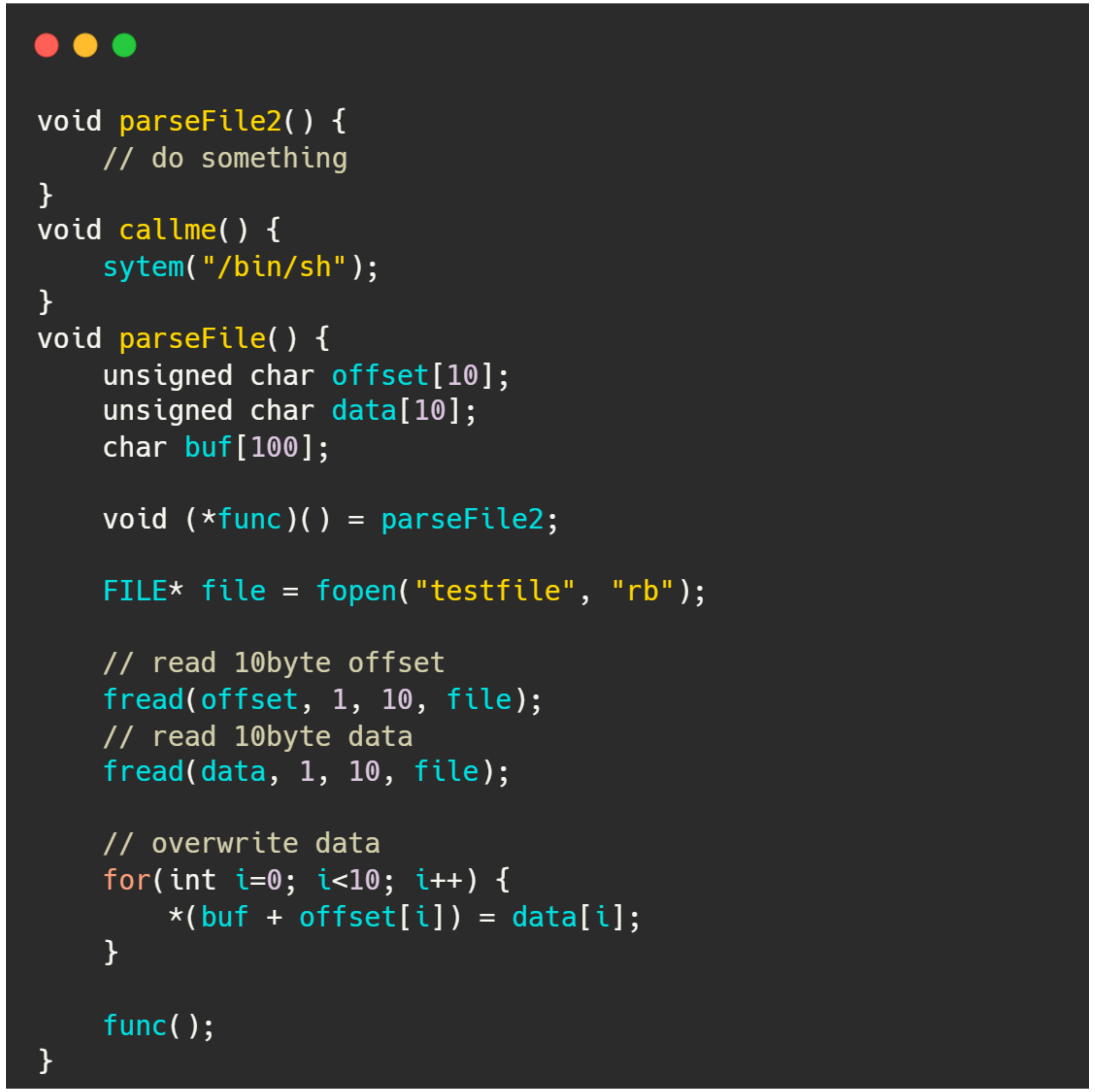
```
void readData() {
    char *buf;
    unsigned int size;

    scanf("%d", &len);

    buf = (char*)malloc(size*4);
    if(!buf) {
        printf("Allocation failed");
        return;
    }
    read(0, buf, size);
}
```

- int ⇔ size_t
- -1 ⇔ 0xFFFFFFFF

```
  0x40000001
×          4
────────────
0x🟥00000004
```

취약점 유형 3

Out of Bounds Read/Write



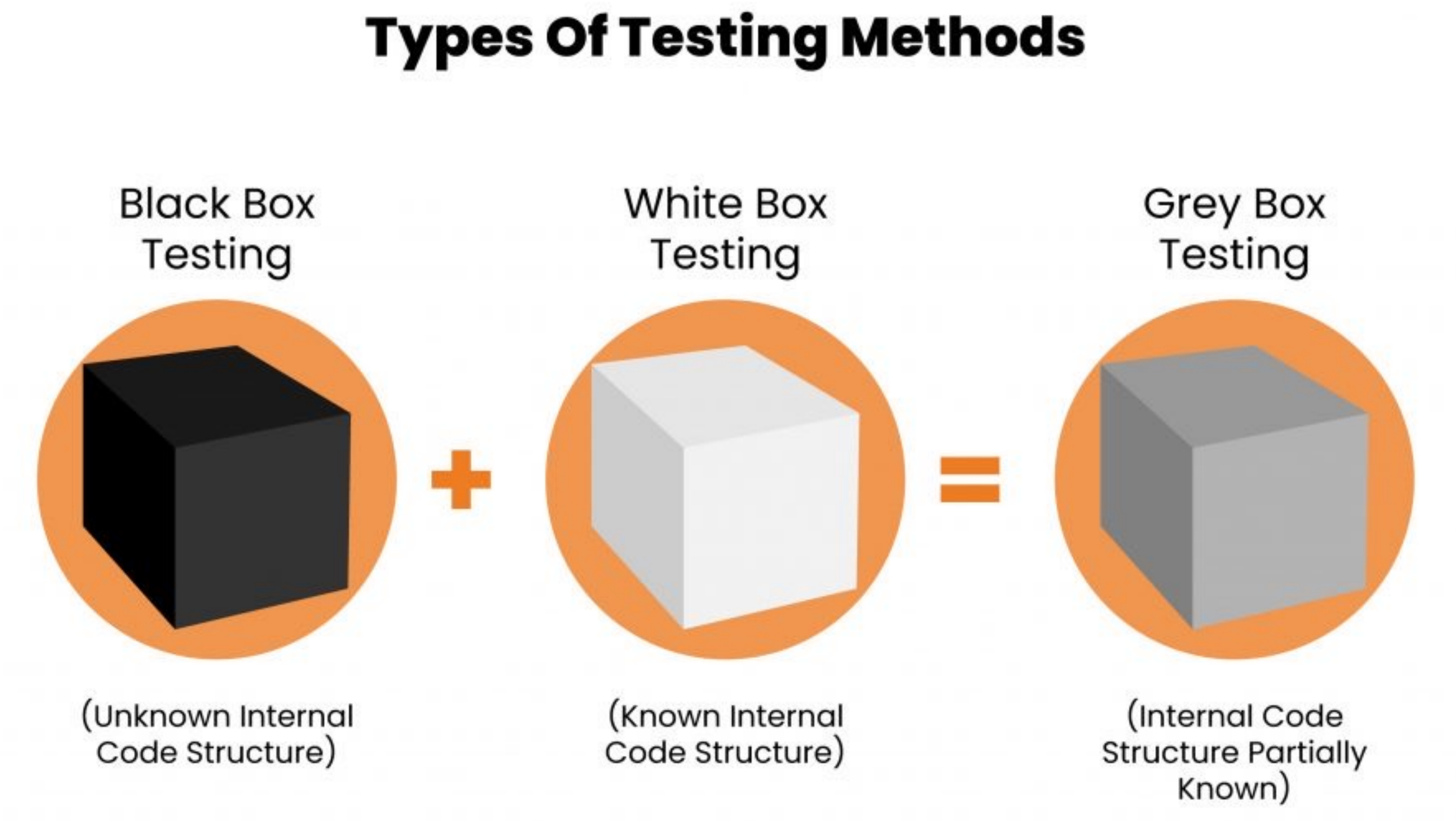
```c
void parseFile2() {
    // do something
}
void callme() {
    sytem("/bin/sh");
}
void parseFile() {
    unsigned char offset[10];
    unsigned char data[10];
    char buf[100];

    void (*func)() = parseFile2;

    FILE* file = fopen("testfile", "rb");

    // read 10byte offset
    fread(offset, 1, 10, file);
    // read 10byte data
    fread(data, 1, 10, file);

    // overwrite data
    for(int i=0; i<10; i++) {
        *(buf + offset[i]) = data[i];
    }

    func();
}
```

- 배열의 임의 인덱스에 접근할 수 있어 발생하는 취약점
- 인덱스 값 음수이거나 배열의 길이를 벗어날 때
- 임의 주소 읽기 및 쓰기 가능함

취약점 분석 방법론

취약점 분석 방법론

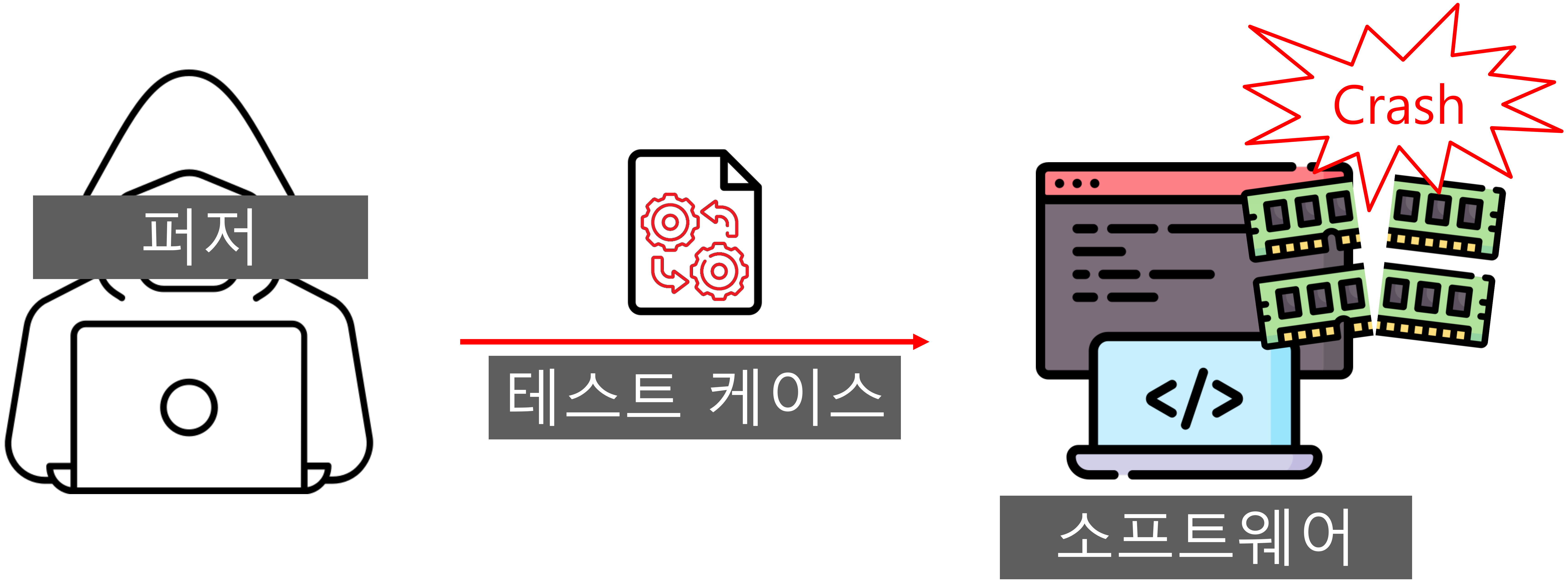
블랙박스, 화이트박스, 그레이박스


Types Of Testing Methods
Black Box Testing
White Box Testing
Grey Box Testing
(Unknown Internal Code Structure)
(Known Internal Code Structure)
(Internal Code Structure Partially Known)

https://kratikal.com/blog/types-of-testing-techniques-black-white-and-grey-box/

취약점 분석 방법론

퍼징



- 소프트웨어에 랜덤한 데이터 넣었을 때 발생하는 크래시를 분석하여 취약점을 찾아내는 것
- 모든 취약점을 다 찾을 순 없고, 오탐일 가능성도 존재

취약점 분석 방법론

Dumb 퍼저 vs WinAFL 퍼저

Dumb 퍼저

- 랜덤하게 데이터 주입
- 프로그램 동작 분석 불필요
- 쉽게 구현 가능
- 복잡한 버그 찾기 어려움
- 비효율적임

VS

WinAFL 퍼저

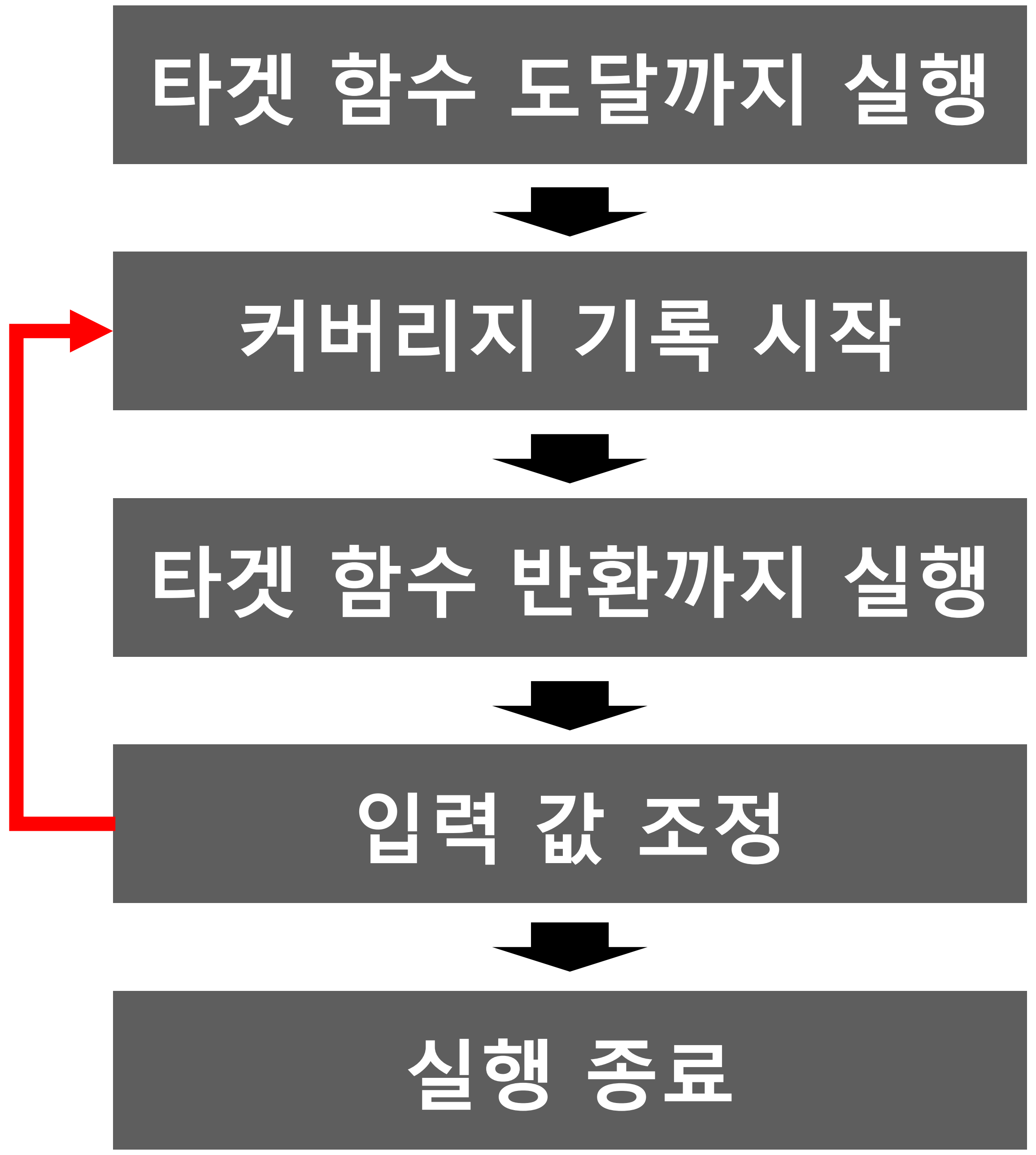
- 코드 커버리지 측정
- 뮤테이션 기반 퍼징
- 프로그램 동작에 대한 분석 필요
- 복잡한 버그 발견 가능성 높음
- 하네스를 활용해 효율적인 퍼징 가능

1) **코드 커버리지**: 테스트 케이스가 프로그램의 어떤 코드 부분을 실행했는지를 측정하는 지표

2) **뮤테이션**: 특정 알고리즘을 이용해서 기존의 입력 데이터를 변형하여 새로운 테스트 케이스를 생성하는 과정

취약점 분석 방법론

WinAFL 동작 과정



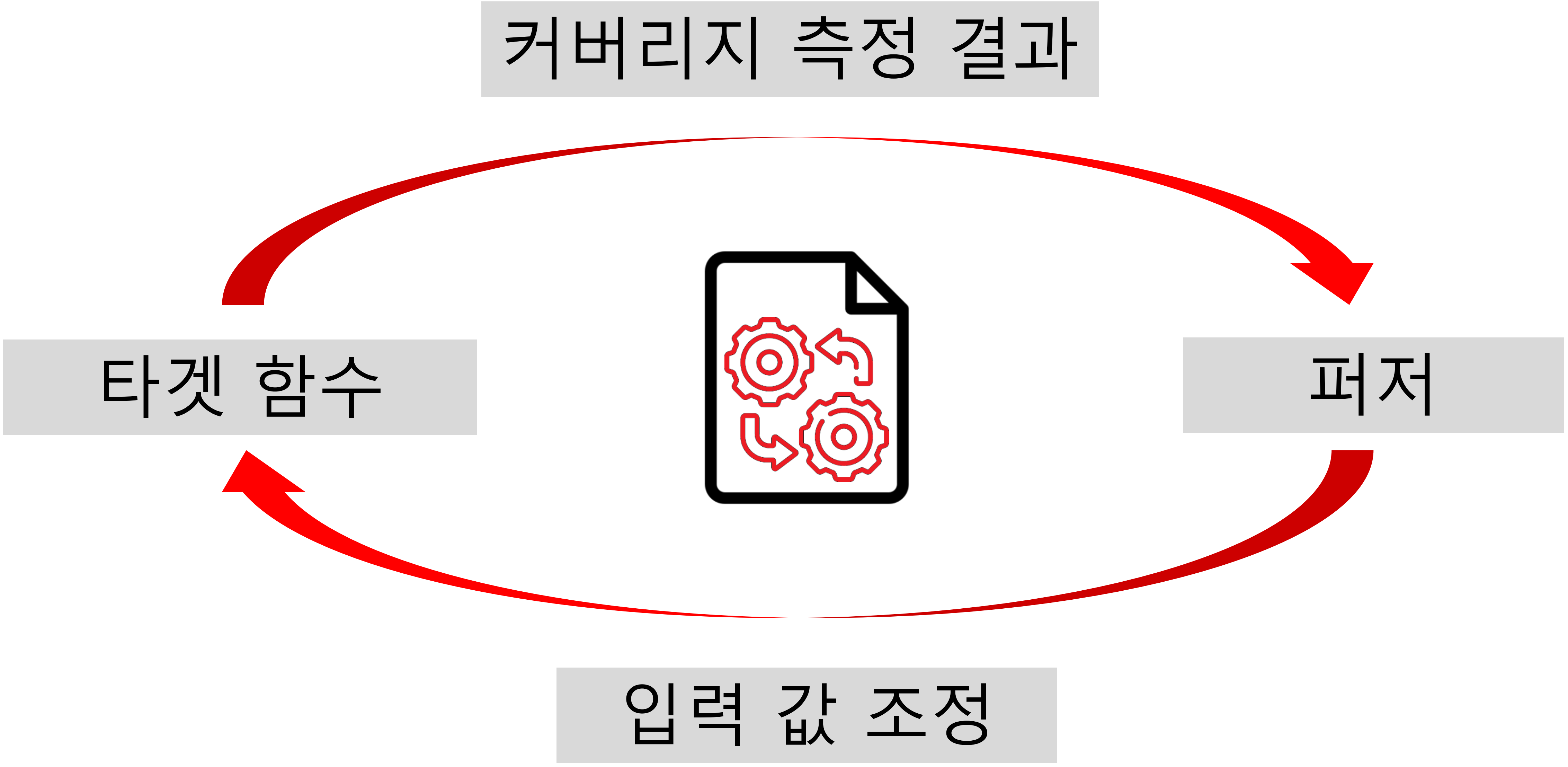
타겟 함수 도달까지 실행

커버리지 기록 시작

타겟 함수 반환까지 실행

입력 값 조정

실행 종료



커버리지 측정 결과

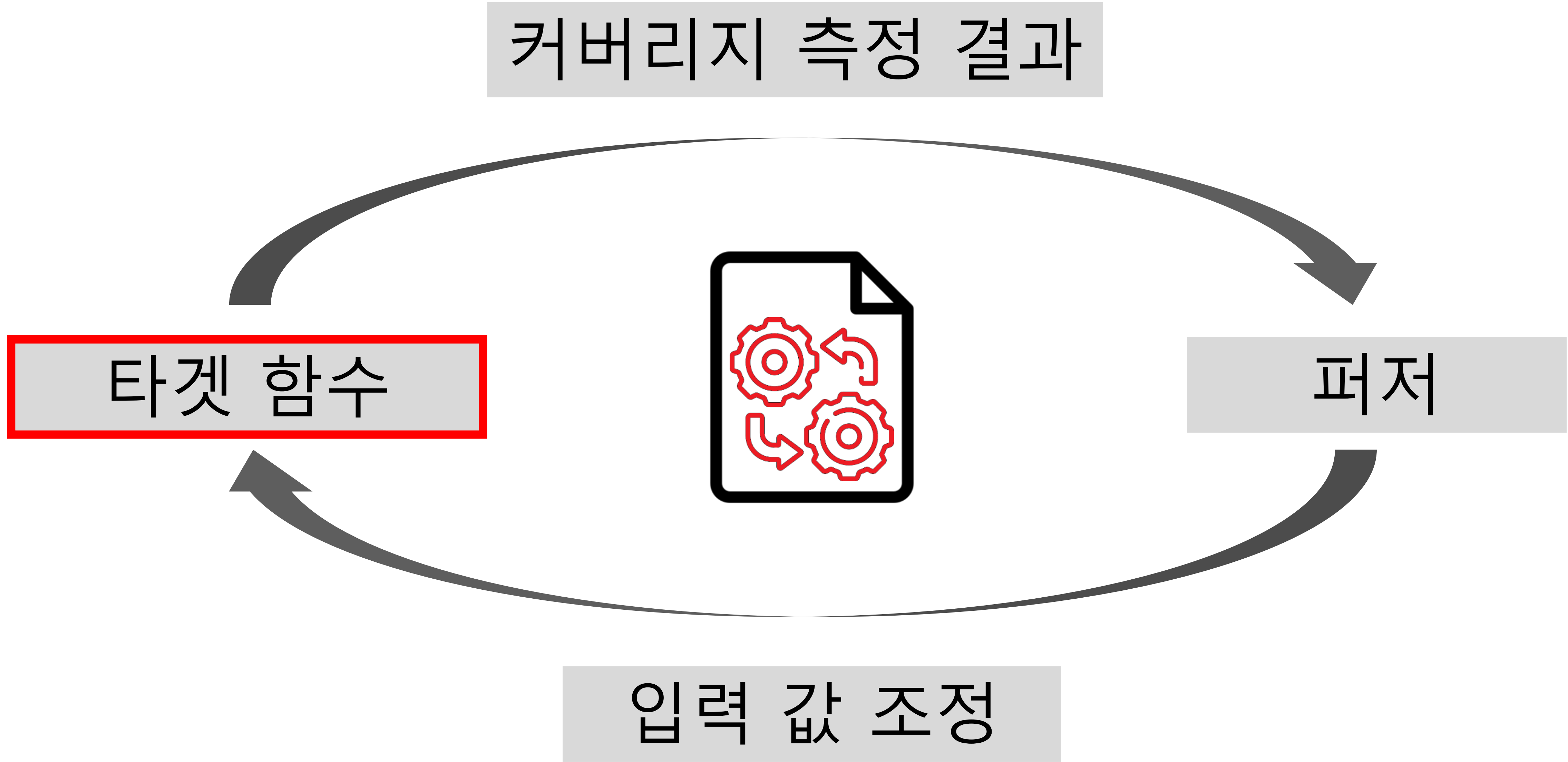
타겟 함수

퍼저

입력 값 조정

취약점 분석 방법론

WinAFL 타겟 함수 선정

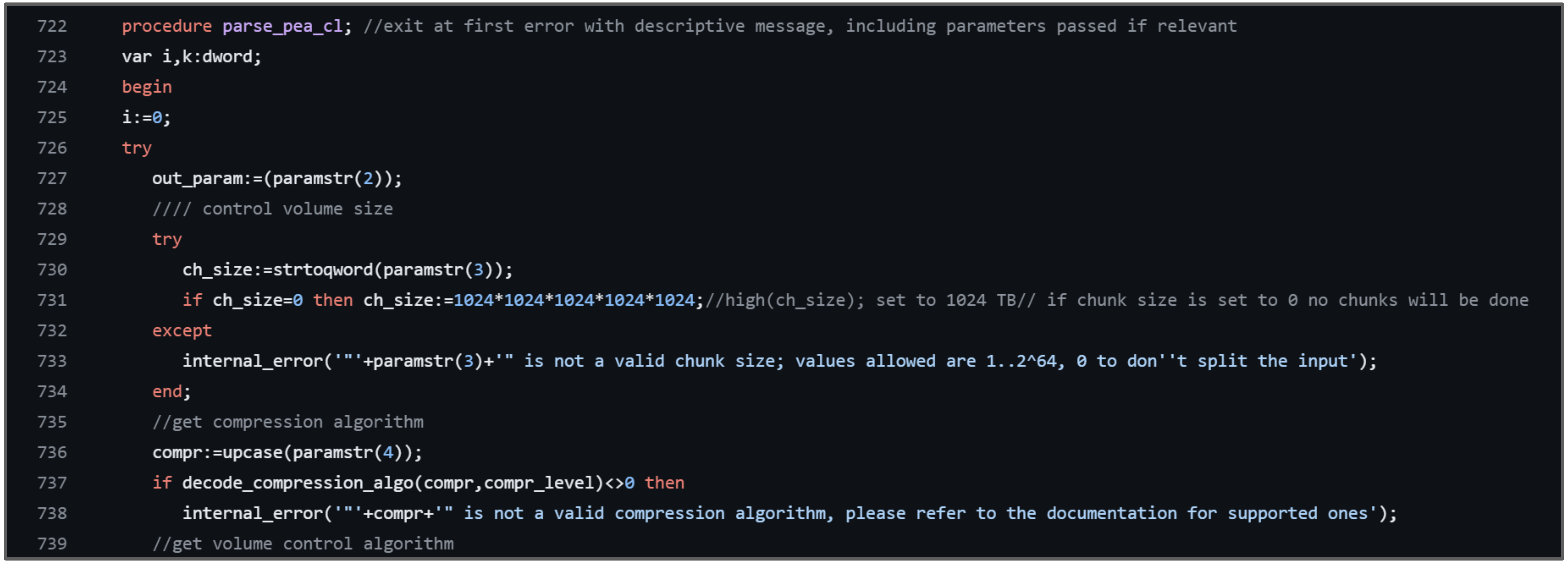


타겟 함수 선정 기준

- 파일을 열고 파싱을 수행해야 함
- 정상적으로 반환되어야 함
- 함수 종료 후 파일을 수정할 수 있어야 함

취약점 분석 방법론

소스 코드 오디팅

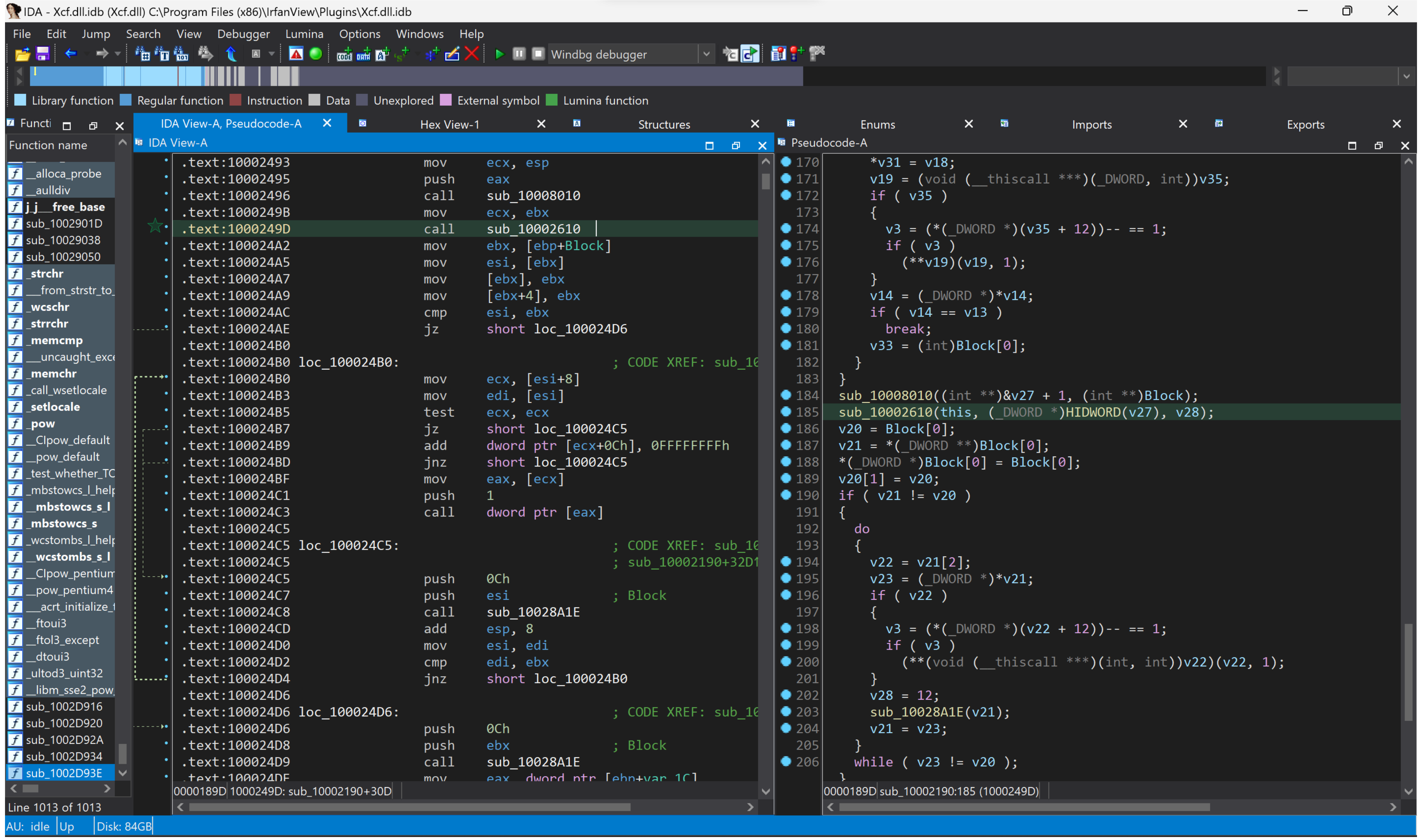


```
procedure parse_pea_cl; //exit at first error with descriptive message, including parameters passed if relevant
var i,k:dword;
begin
i:=0;
try
   out_param:=(paramstr(2));
   //// control volume size
   try
      ch_size:=strtoqword(paramstr(3));
      if ch_size=0 then ch_size:=1024*1024*1024*1024*1024;//high(ch_size); set to 1024 TB// if chunk size is set to 0 no chunks will be done
   except
      internal_error('"'+paramstr(3)+'" is not a valid chunk size; values allowed are 1..2^64, 0 to don''t split the input');
   end;
   //get compression algorithm
   compr:=upcase(paramstr(4));
   if decode_compression_algo(compr,compr_level)<>0 then
      internal_error('"'+compr+'" is not a valid compression algorithm, please refer to the documentation for supported ones');
   //get volume control algorithm
```

오픈 소스 소프트웨어의 소스 코드를 분석하여 취약점 탐색

취약점 분석 방법론

리버스 엔지니어링



```
__alloca_probe
__aulldiv
j_j__free_base
sub_1002901D
sub_10029038
sub_10029050
_strchr
___from_strstr_to_
_wcschr
_strrchr
_memcmp
___uncaught_exce
_memchr
_call_wsetlocale
_setlocale
_pow
__Clpow_default
__pow_default
_test_whether_TC
_mbstowcs_l_help
__mbstowcs_s_l
_mbstowcs_s
_wcstombs_l_help
__wcstombs_s_l
__Clpow_pentium
__pow_pentium4
___acrt_initialize_t
__ftoui3
__ftol3_except
__dtoui3
_ultod3_uint32
__libm_sse2_pow_
sub_1002D916
sub_1002D920
sub_1002D92A
sub_1002D934
sub_1002D93E
```

```
.text:10002493                 mov     ecx, esp
.text:10002495                 push    eax
.text:10002496                 call    sub_10008010
.text:1000249B                 mov     ecx, ebx
.text:1000249D                 call    sub_10002610
.text:100024A2                 mov     ebx, [ebp+Block]
.text:100024A5                 mov     esi, [ebx]
.text:100024A7                 mov     [ebx], ebx
.text:100024A9                 mov     [ebx+4], ebx
.text:100024AC                 cmp     esi, ebx
.text:100024AE                 jz      short loc_100024D6
.text:100024B0
.text:100024B0 loc_100024B0:                           ; CODE XREF: sub_10
.text:100024B0                 mov     ecx, [esi+8]
.text:100024B3                 mov     edi, [esi]
.text:100024B5                 test    ecx, ecx
.text:100024B7                 jz      short loc_100024C5
.text:100024B9                 add     dword ptr [ecx+0Ch], 0FFFFFFFFh
.text:100024BD                 jnz     short loc_100024C5
.text:100024BF                 mov     eax, [ecx]
.text:100024C1                 push    1
.text:100024C3                 call    dword ptr [eax]
.text:100024C5
.text:100024C5 loc_100024C5:                           ; CODE XREF: sub_10
.text:100024C5                                         ; sub_10002190+32D1
.text:100024C5                 push    0Ch
.text:100024C7                 push    esi             ; Block
.text:100024C8                 call    sub_10028A1E
.text:100024CD                 add     esp, 8
.text:100024D0                 mov     esi, edi
.text:100024D2                 cmp     edi, ebx
.text:100024D4                 jnz     short loc_100024B0
.text:100024D6
.text:100024D6 loc_100024D6:                           ; CODE XREF: sub_10
.text:100024D6                 push    0Ch
.text:100024D8                 push    ebx             ; Block
.text:100024D9                 call    sub_10028A1E
.text:100024DF                 mov     eax, dword ptr [ebp+var_1C]
```

0000189D 1000249D: sub_10002190+30D

```
      *v31 = v18;
      v19 = (void (__thiscall ***)(_DWORD, int))v35;
      if ( v35 )
      {
        v3 = (*(_DWORD *)(v35 + 12))-- == 1;
        if ( v3 )
          (**v19)(v19, 1);
      }
      v14 = (_DWORD *)*v14;
      if ( v14 == v13 )
        break;
      v33 = (int)Block[0];
    }
  }
  sub_10008010((int **)&v27 + 1, (int **)Block);
  sub_10002610(this, (_DWORD *)HIDWORD(v27), v28);
  v20 = Block[0];
  v21 = *(_DWORD **)Block[0];
  *(_DWORD *)Block[0] = Block[0];
  v20[1] = v20;
  if ( v21 != v20 )
  {
    do
    {
      v22 = v21[2];
      v23 = (_DWORD *)*v21;
      if ( v22 )
      {
        v3 = (*(_DWORD *)(v22 + 12))-- == 1;
        if ( v3 )
          (**(void (__thiscall ***)(int, int))v22)(v22, 1);
      }
      v28 = 12;
      sub_10028A1E(v21);
      v21 = v23;
    }
    while ( v23 != v20 );
  }
```

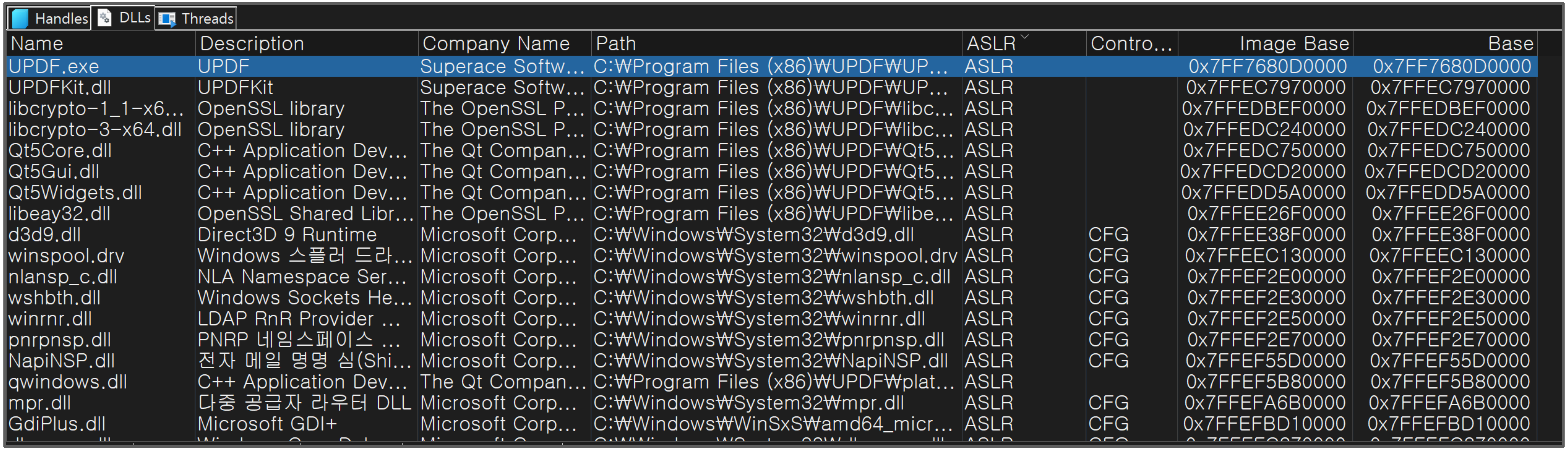
0000189D sub_10002190:185 (1000249D)

WinDbg, x64Dbg 등의 툴을 사용해 프로그램의 동작(함수) 분석

취약점 분석 과정

취약점 분석 과정

보호 기법 확인 (procexp)

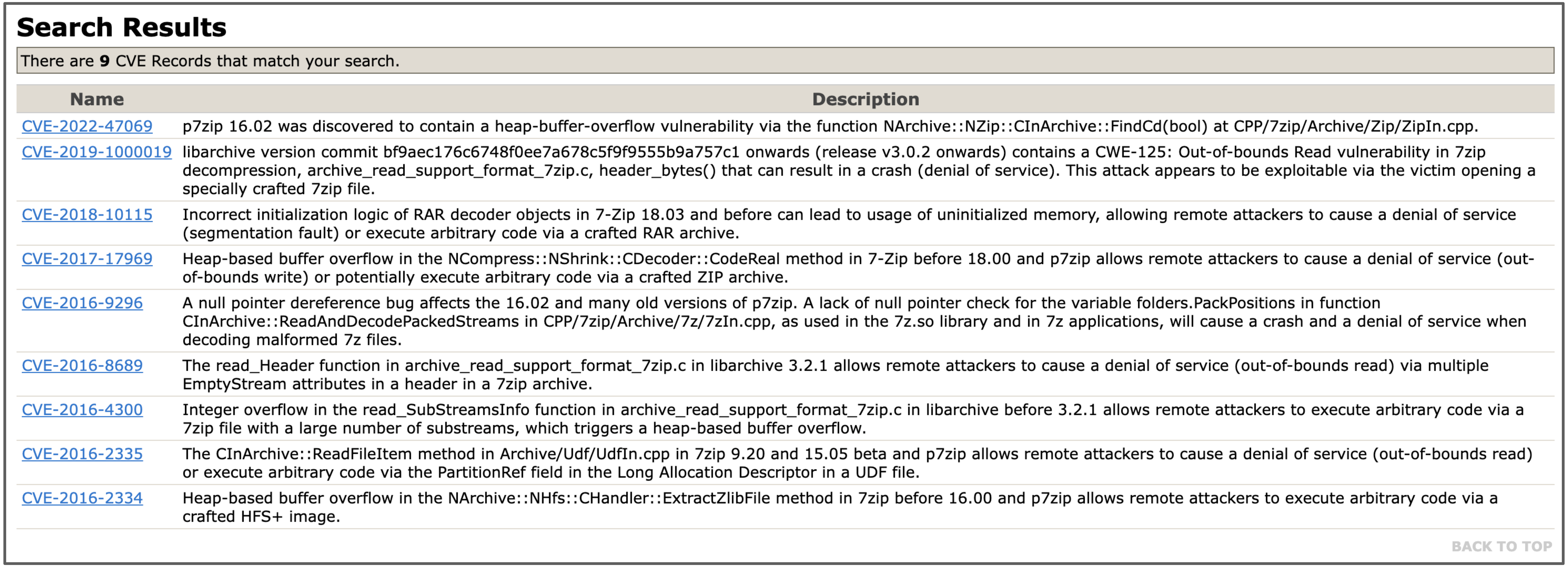


Name	Description	Company Name	Path	ASLR	Contro...	Image Base	Base
UPDF.exe	UPDF	Superace Softw...	C:₩Program Files (x86)₩UPDF₩UP...	ASLR		0x7FF7680D0000	0x7FF7680D0000
UPDFKit.dll	UPDFKit	Superace Softw...	C:₩Program Files (x86)₩UPDF₩UP...	ASLR		0x7FFEC7970000	0x7FFEC7970000
libcrypto-1_1-x6...	OpenSSL library	The OpenSSL P...	C:₩Program Files (x86)₩UPDF₩libc...	ASLR		0x7FFEDBEF0000	0x7FFEDBEF0000
libcrypto-3-x64.dll	OpenSSL library	The OpenSSL P...	C:₩Program Files (x86)₩UPDF₩libc...	ASLR		0x7FFEDC240000	0x7FFEDC240000
Qt5Core.dll	C++ Application Dev...	The Qt Compan...	C:₩Program Files (x86)₩UPDF₩Qt5...	ASLR		0x7FFEDC750000	0x7FFEDC750000
Qt5Gui.dll	C++ Application Dev...	The Qt Compan...	C:₩Program Files (x86)₩UPDF₩Qt5...	ASLR		0x7FFEDCD20000	0x7FFEDCD20000
Qt5Widgets.dll	C++ Application Dev...	The Qt Compan...	C:₩Program Files (x86)₩UPDF₩Qt5...	ASLR		0x7FFEDD5A0000	0x7FFEDD5A0000
libeay32.dll	OpenSSL Shared Libr...	The OpenSSL P...	C:₩Program Files (x86)₩UPDF₩libe...	ASLR		0x7FFEE26F0000	0x7FFEE26F0000
d3d9.dll	Direct3D 9 Runtime	Microsoft Corp...	C:₩Windows₩System32₩d3d9.dll	ASLR	CFG	0x7FFEE38F0000	0x7FFEE38F0000
winspool.drv	Windows 스풀러 드라...	Microsoft Corp...	C:₩Windows₩System32₩winspool.drv	ASLR	CFG	0x7FFEEC130000	0x7FFEEC130000
nlansp_c.dll	NLA Namespace Ser...	Microsoft Corp...	C:₩Windows₩System32₩nlansp_c.dll	ASLR	CFG	0x7FFEF2E00000	0x7FFEF2E00000
wshbth.dll	Windows Sockets He...	Microsoft Corp...	C:₩Windows₩System32₩wshbth.dll	ASLR	CFG	0x7FFEF2E30000	0x7FFEF2E30000
winrnr.dll	LDAP RnR Provider ...	Microsoft Corp...	C:₩Windows₩System32₩winrnr.dll	ASLR	CFG	0x7FFEF2E50000	0x7FFEF2E50000
pnrpnsp.dll	PNRP 네임스페이스 ...	Microsoft Corp...	C:₩Windows₩System32₩pnrpnsp.dll	ASLR	CFG	0x7FFEF2E70000	0x7FFEF2E70000
NapiNSP.dll	전자 메일 명명 심(Shi...	Microsoft Corp...	C:₩Windows₩System32₩NapiNSP.dll	ASLR	CFG	0x7FFEF55D0000	0x7FFEF55D0000
qwindows.dll	C++ Application Dev...	The Qt Compan...	C:₩Program Files (x86)₩UPDF₩plat...	ASLR		0x7FFEF5B80000	0x7FFEF5B80000
mpr.dll	다중 공급자 라우터 DLL	Microsoft Corp...	C:₩Windows₩System32₩mpr.dll	ASLR	CFG	0x7FFEFA6B0000	0x7FFEFA6B0000
GdiPlus.dll	Microsoft GDI+	Microsoft Corp...	C:₩Windows₩WinSxS₩amd64_micr...	ASLR	CFG	0x7FFEFBD10000	0x7FFEFBD10000

https://learn.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite

취약점 분석 과정

CVE 확인 (MITRE)



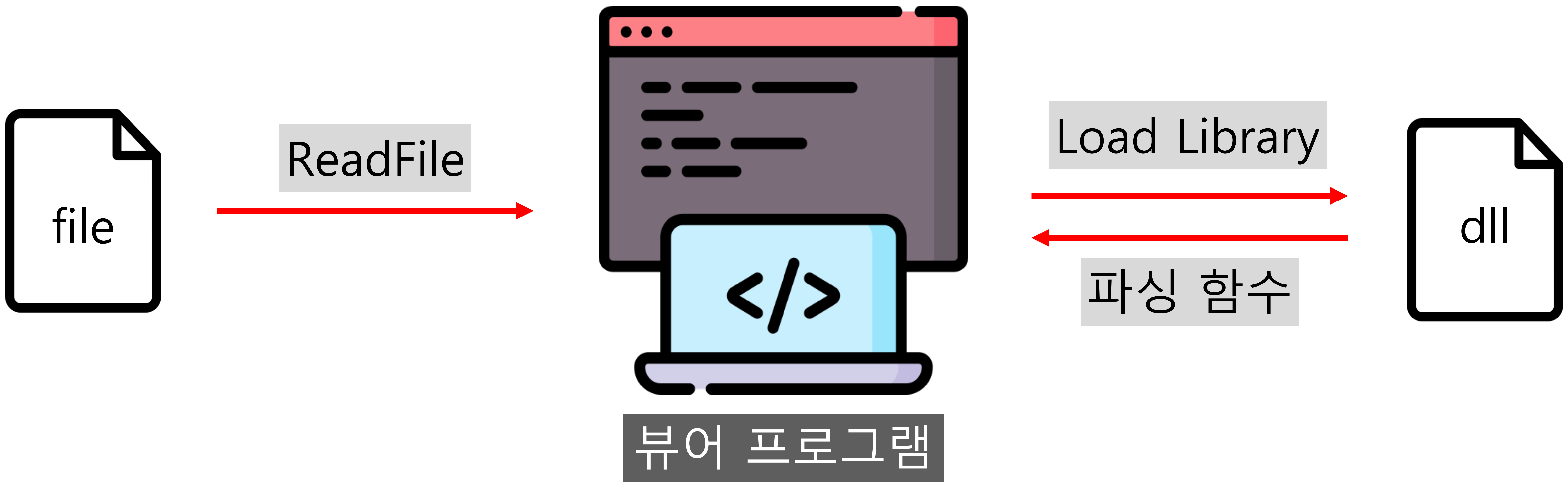
Search Results

There are **9** CVE Records that match your search.

Name	Description
CVE-2022-47069	p7zip 16.02 was discovered to contain a heap-buffer-overflow vulnerability via the function NArchive::NZip::CInArchive::FindCd(bool) at CPP/7zip/Archive/Zip/ZipIn.cpp.
CVE-2019-1000019	libarchive version commit bf9aec176c6748f0ee7a678c5f9f9555b9a757c1 onwards (release v3.0.2 onwards) contains a CWE-125: Out-of-bounds Read vulnerability in 7zip decompression, archive_read_support_format_7zip.c, header_bytes() that can result in a crash (denial of service). This attack appears to be exploitable via the victim opening a specially crafted 7zip file.
CVE-2018-10115	Incorrect initialization logic of RAR decoder objects in 7-Zip 18.03 and before can lead to usage of uninitialized memory, allowing remote attackers to cause a denial of service (segmentation fault) or execute arbitrary code via a crafted RAR archive.
CVE-2017-17969	Heap-based buffer overflow in the NCompress::NShrink::CDecoder::CodeReal method in 7-Zip before 18.00 and p7zip allows remote attackers to cause a denial of service (out-of-bounds write) or potentially execute arbitrary code via a crafted ZIP archive.
CVE-2016-9296	A null pointer dereference bug affects the 16.02 and many old versions of p7zip. A lack of null pointer check for the variable folders.PackPositions in function CInArchive::ReadAndDecodePackedStreams in CPP/7zip/Archive/7z/7zIn.cpp, as used in the 7z.so library and in 7z applications, will cause a crash and a denial of service when decoding malformed 7z files.
CVE-2016-8689	The read_Header function in archive_read_support_format_7zip.c in libarchive 3.2.1 allows remote attackers to cause a denial of service (out-of-bounds read) via multiple EmptyStream attributes in a header in a 7zip archive.
CVE-2016-4300	Integer overflow in the read_SubStreamsInfo function in archive_read_support_format_7zip.c in libarchive before 3.2.1 allows remote attackers to execute arbitrary code via a 7zip file with a large number of substreams, which triggers a heap-based buffer overflow.
CVE-2016-2335	The CInArchive::ReadFileItem method in Archive/Udf/UdfIn.cpp in 7zip 9.20 and 15.05 beta and p7zip allows remote attackers to cause a denial of service (out-of-bounds read) or execute arbitrary code via the PartitionRef field in the Long Allocation Descriptor in a UDF file.
CVE-2016-2334	Heap-based buffer overflow in the NArchive::NHfs::CHandler::ExtractZlibFile method in 7zip before 16.00 and p7zip allows remote attackers to execute arbitrary code via a crafted HFS+ image.

취약점 분석 과정

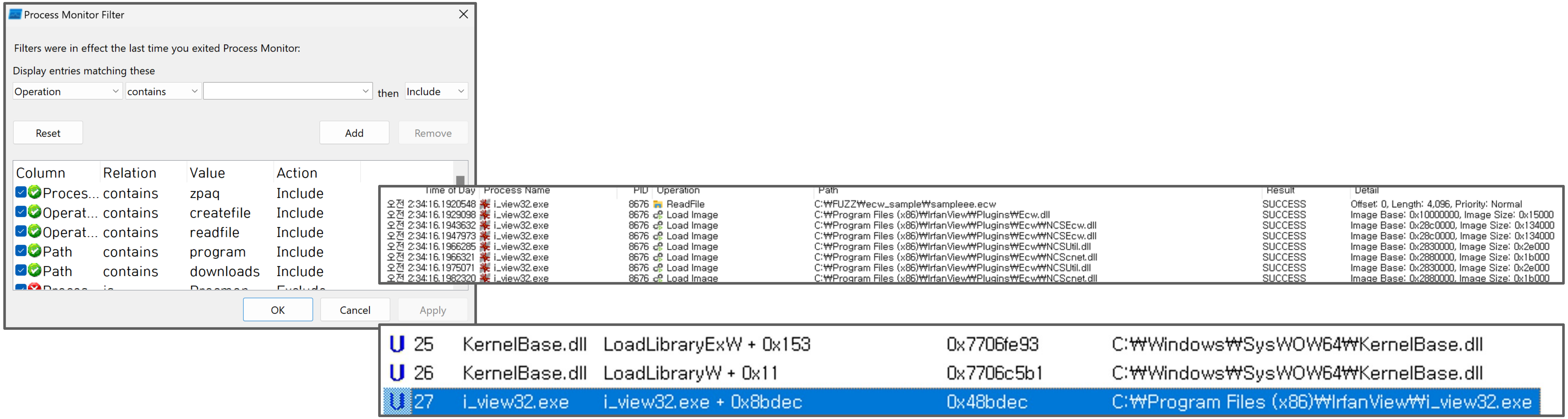
뷰어 프로그램



- 뷰어 프로그램은 여러 가지 확장자를 열어볼 수 있음
- 확장자별 dll의 함수를 불러와 파싱 수행

취약점 분석 과정

API 호출 지점 파악



Process Monitor Filter

Filters were in effect the last time you exited Process Monitor:

Display entries matching these

Operation | contains | | then Include

Column	Relation	Value	Action
☑ Proces...	contains	zpaq	Include
☑ Operat...	contains	createfile	Include
☑ Operat...	contains	readfile	Include
☑ Path	contains	program	Include
☑ Path	contains	downloads	Include
☑ Proces...	is	Procmon	Exclude

```
      Time of Day  Process Name   PID  Operation   Path                                                         Result   Detail
오전 2:34:16.1920548  i_view32.exe  8676  ReadFile    C:₩FUZZ₩ecw_sample₩sampleee.ecw                              SUCCESS  Offset: 0, Length: 4,096, Priority: Normal
오전 2:34:16.1929098  i_view32.exe  8676  Load Image  C:₩Program Files (x86)₩IrfanView₩Plugins₩Ecw.dll              SUCCESS  Image Base: 0x10000000, Image Size: 0x15000
오전 2:34:16.1943632  i_view32.exe  8676  Load Image  C:₩Program Files (x86)₩IrfanView₩Plugins₩Ecw₩NCSEcw.dll       SUCCESS  Image Base: 0x28c0000, Image Size: 0x134000
오전 2:34:16.1947973  i_view32.exe  8676  Load Image  C:₩Program Files (x86)₩IrfanView₩Plugins₩Ecw₩NCSEcw.dll       SUCCESS  Image Base: 0x28c0000, Image Size: 0x134000
오전 2:34:16.1966285  i_view32.exe  8676  Load Image  C:₩Program Files (x86)₩IrfanView₩Plugins₩Ecw₩NCSUtil.dll      SUCCESS  Image Base: 0x2830000, Image Size: 0x2e000
오전 2:34:16.1966321  i_view32.exe  8676  Load Image  C:₩Program Files (x86)₩IrfanView₩Plugins₩Ecw₩NCScnet.dll      SUCCESS  Image Base: 0x2880000, Image Size: 0x1b000
오전 2:34:16.1975071  i_view32.exe  8676  Load Image  C:₩Program Files (x86)₩IrfanView₩Plugins₩Ecw₩NCSUtil.dll      SUCCESS  Image Base: 0x2830000, Image Size: 0x2e000
오전 2:34:16.1982320  i_view32.exe  8676  Load Image  C:₩Program Files (x86)₩IrfanView₩Plugins₩Ecw₩NCScnet.dll      SUCCESS  Image Base: 0x2880000, Image Size: 0x1b000
```

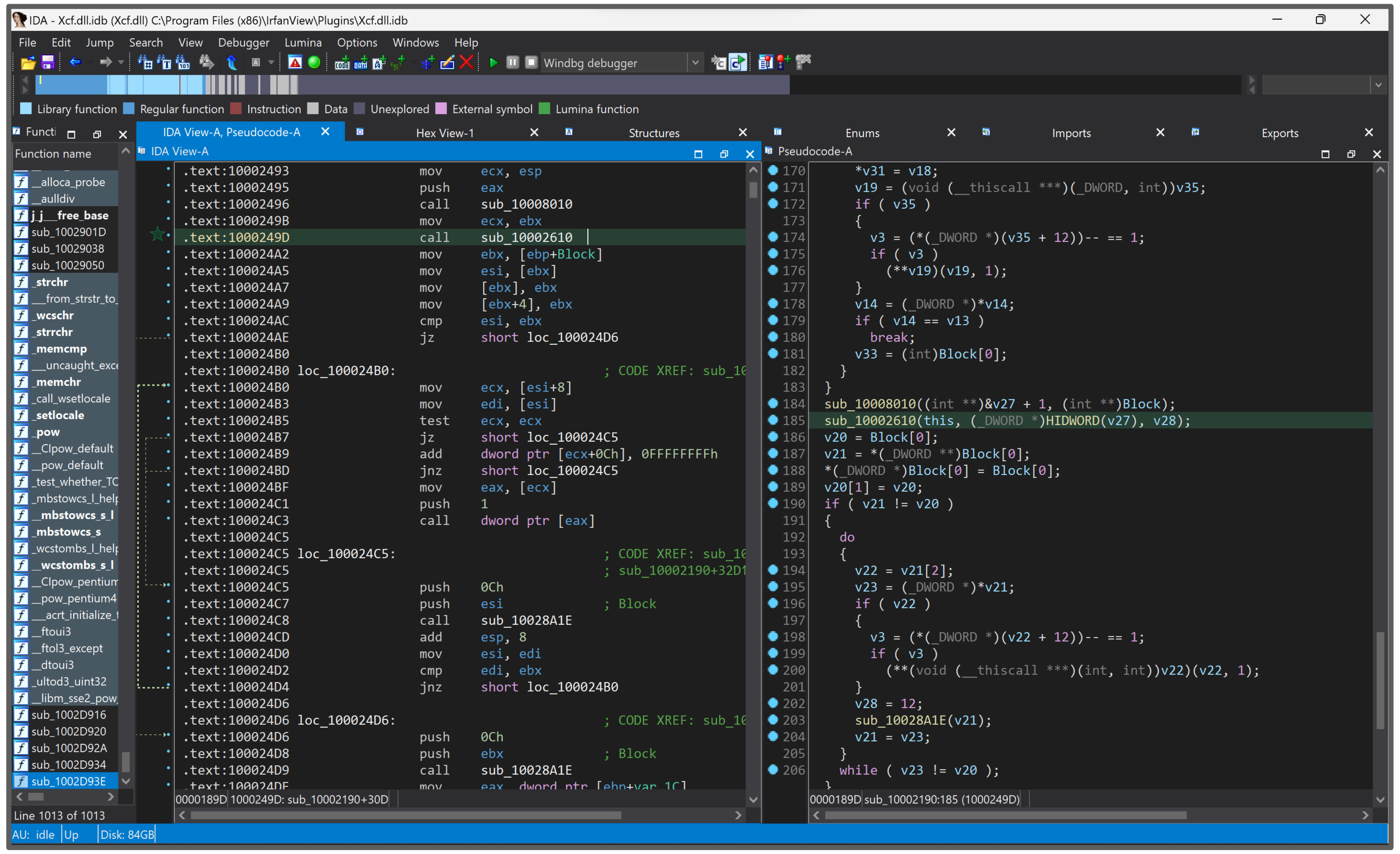
```
U 25  KernelBase.dll  LoadLibraryExW + 0x153   0x7706fe93  C:₩Windows₩SysWOW64₩KernelBase.dll
U 26  KernelBase.dll  LoadLibraryW + 0x11      0x7706c5b1  C:₩Windows₩SysWOW64₩KernelBase.dll
U 27  i_view32.exe    i_view32.exe + 0x8bdec   0x48bdec    C:₩Program Files (x86)₩IrfanView₩i_view32.exe
```

https://learn.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite

ReadFile, Image Load 등으로 필터링하여 확인

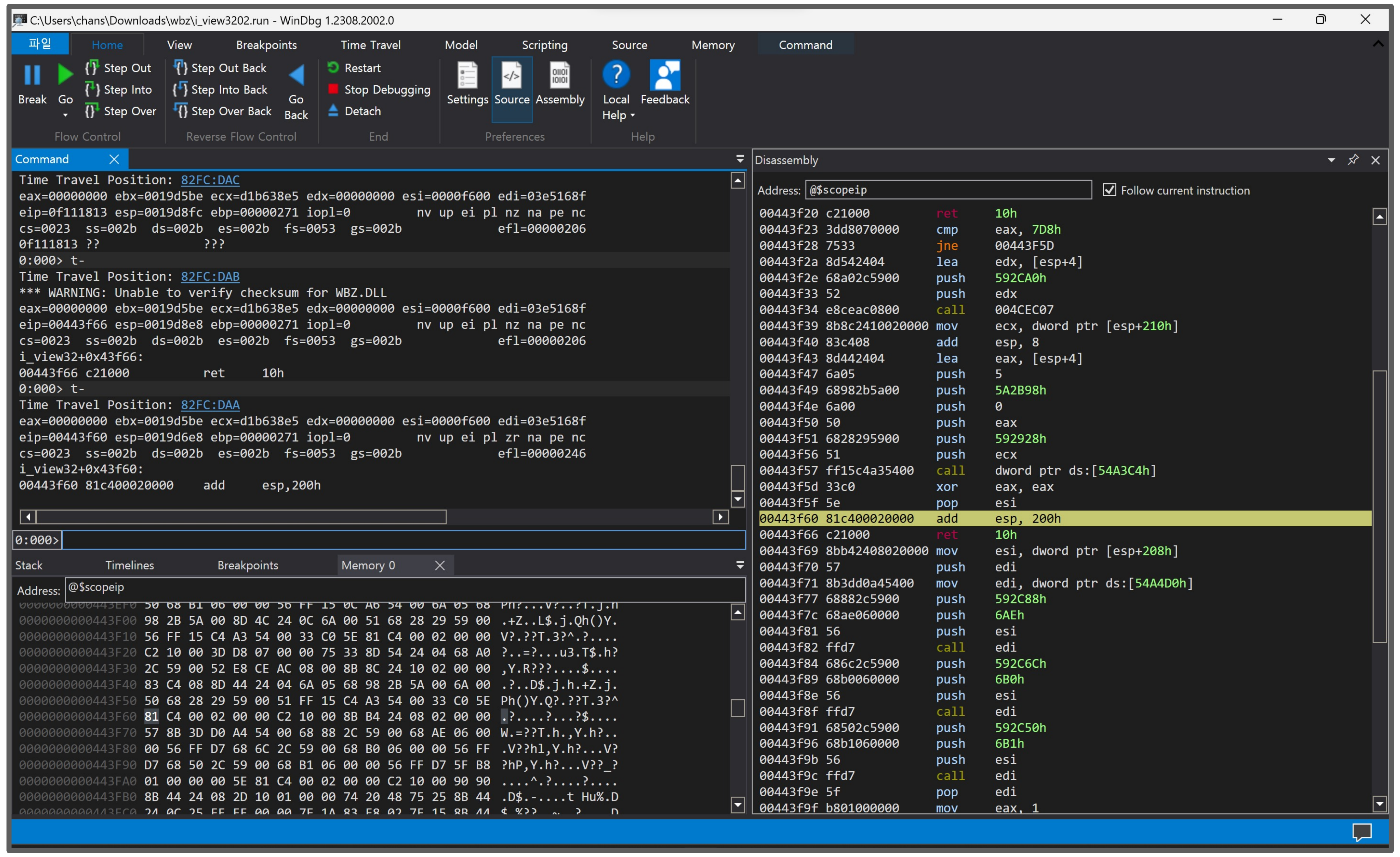
취약점 분석 과정

함수 인자 분석



```
.text:10002493                 mov     ecx, esp
.text:10002495                 push    eax
.text:10002496                 call    sub_10008010
.text:1000249B                 mov     ecx, ebx
.text:1000249D                 call    sub_10002610
.text:100024A2                 mov     ebx, [ebp+Block]
.text:100024A5                 mov     esi, [ebx]
.text:100024A7                 mov     [ebx], ebx
.text:100024A9                 mov     [ebx+4], ebx
.text:100024AC                 cmp     esi, ebx
.text:100024AE                 jz      short loc_100024D6
.text:100024B0
.text:100024B0 loc_100024B0:                           ; CODE XREF: sub_10
.text:100024B0                 mov     ecx, [esi+8]
.text:100024B3                 mov     edi, [esi]
.text:100024B5                 test    ecx, ecx
.text:100024B7                 jz      short loc_100024C5
.text:100024B9                 add     dword ptr [ecx+0Ch], 0FFFFFFFFh
.text:100024BD                 jnz     short loc_100024C5
.text:100024BF                 mov     eax, [ecx]
.text:100024C1                 push    1
.text:100024C3                 call    dword ptr [eax]
.text:100024C5
.text:100024C5 loc_100024C5:                           ; CODE XREF: sub_10
.text:100024C5                                         ; sub_10002190+32D1
.text:100024C5                 push    0Ch
.text:100024C7                 push    esi             ; Block
.text:100024C8                 call    sub_10028A1E
.text:100024CD                 add     esp, 8
.text:100024D0                 mov     esi, edi
.text:100024D2                 cmp     edi, ebx
.text:100024D4                 jnz     short loc_100024B0
.text:100024D6
.text:100024D6 loc_100024D6:                           ; CODE XREF: sub_10
.text:100024D6                 push    0Ch
.text:100024D8                 push    ebx             ; Block
.text:100024D9                 call    sub_10028A1E
0000189D 1000249D: sub_10002190+30D
```

```
      *v31 = v18;
      v19 = (void (__thiscall ***)(_DWORD, int))v35;
      if ( v35 )
      {
        v3 = (*(_DWORD *)(v35 + 12))-- == 1;
        if ( v3 )
          (**v19)(v19, 1);
      }
      v14 = (_DWORD *)*v14;
      if ( v14 == v13 )
        break;
      v33 = (int)Block[0];
    }
  }
  sub_10008010((int **)&v27 + 1, (int **)Block);
  sub_10002610(this, (_DWORD *)HIDWORD(v27), v28);
  v20 = Block[0];
  v21 = *(_DWORD **)Block[0];
  *(_DWORD *)Block[0] = Block[0];
  v20[1] = v20;
  if ( v21 != v20 )
  {
    do
    {
      v22 = v21[2];
      v23 = (_DWORD *)*v21;
      if ( v22 )
      {
        v3 = (*(_DWORD *)(v22 + 12))-- == 1;
        if ( v3 )
          (**(void (__thiscall ***)(int, int))v22)(v22, 1);
      }
      v28 = 12;
      sub_10028A1E(v21);
      v21 = v23;
    }
    while ( v23 != v20 );
0000189D sub_10002190:185 (1000249D)
```



```
Time Travel Position: 82FC:DAC
eax=00000000 ebx=0019d5be ecx=d1b638e5 edx=00000000 esi=0000f600 edi=03e5168f
eip=0f111813 esp=0019d8fc ebp=00000271 iopl=0         nv up ei pl nz na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000206
0f111813 ??              ???
0:000> t-
Time Travel Position: 82FC:DAB
*** WARNING: Unable to verify checksum for WBZ.DLL
eax=00000000 ebx=0019d5be ecx=d1b638e5 edx=00000000 esi=0000f600 edi=03e5168f
eip=00443f66 esp=0019d8e8 ebp=00000271 iopl=0         nv up ei pl nz na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000206
i_view32+0x43f66:
00443f66 c21000          ret     10h
0:000> t-
Time Travel Position: 82FC:DAA
eax=00000000 ebx=0019d5be ecx=d1b638e5 edx=00000000 esi=0000f600 edi=03e5168f
eip=00443f60 esp=0019d6e8 ebp=00000271 iopl=0         nv up ei pl zr na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000246
i_view32+0x43f60:
00443f60 81c400020000    add     esp,200h
0:000>
```

Address: @$scopeip

```
0000000000443F00 98 2B 5A 00 8D 4C 24 0C 6A 00 51 68 28 29 59 00 .+Z..L$.j.Qh()Y.
0000000000443F10 56 FF 15 C4 A3 54 00 33 C0 5E 81 C4 00 02 00 00 V?.??T.3?^.?....
0000000000443F20 C2 10 00 3D D8 07 00 00 75 33 8D 54 24 04 68 A0 ?..=?...u3.T$.h?
0000000000443F30 2C 59 00 52 E8 CE AC 08 00 8B 8C 24 10 02 00 00 ,Y.R???....$....
0000000000443F40 83 C4 08 8D 44 24 04 6A 05 68 98 2B 5A 00 6A 00 .?..D$.j.h.+Z.j.
0000000000443F50 50 68 28 29 59 00 51 FF 15 C4 A3 54 00 33 C0 5E Ph()Y.Q?.??T.3?^
0000000000443F60 81 C4 00 02 00 00 C2 10 00 8B B4 24 08 02 00 00 .?....?...?$....
0000000000443F70 57 8B 3D D0 A4 54 00 68 88 2C 59 00 68 AE 06 00 W.=??T.h.,Y.h?..
0000000000443F80 00 56 FF D7 68 6C 2C 59 00 68 B0 06 00 00 56 FF .V??hl,Y.h?...V?
0000000000443F90 D7 68 50 2C 59 00 68 B1 06 00 00 56 FF D7 5F B8 ?hP,Y.h?...V??_?
0000000000443FA0 01 00 00 00 5E 81 C4 00 02 00 00 C2 10 00 90 90 ....^.?.....?....
0000000000443FB0 8B 44 24 08 2D 10 01 00 00 74 20 48 75 25 8B 44 .D$.-....t Hu%.D
```

Address: @$scopeip ☑ Follow current instruction

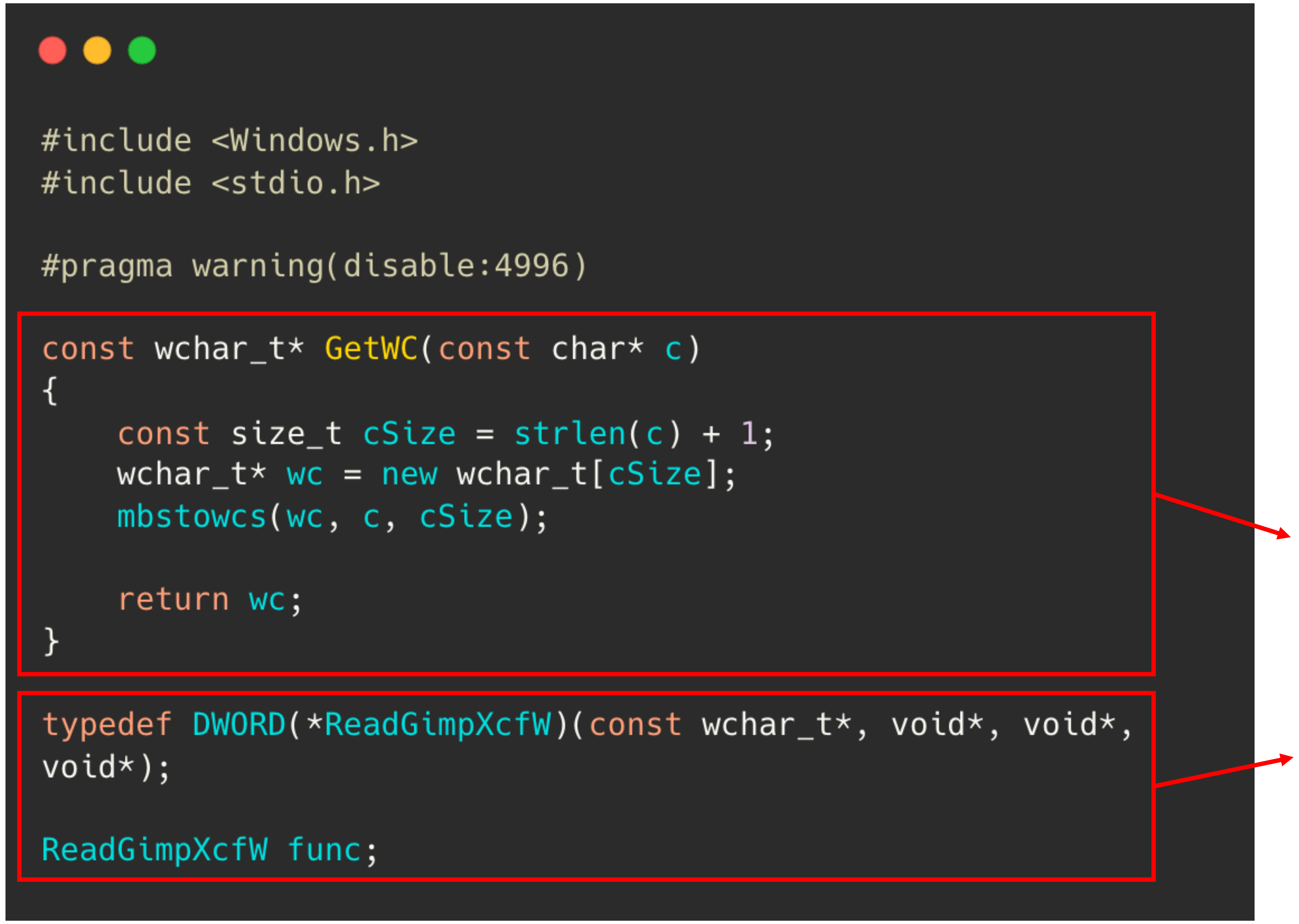
```
00443f20 c21000          ret     10h
00443f23 3dd8070000      cmp     eax, 7D8h
00443f28 7533            jne     00443F5D
00443f2a 8d542404        lea     edx, [esp+4]
00443f2e 68a02c5900      push    592CA0h
00443f33 52              push    edx
00443f34 e8ceac0800      call    004CEC07
00443f39 8b8c2410020000  mov     ecx, dword ptr [esp+210h]
00443f40 83c408          add     esp, 8
00443f43 8d442404        lea     eax, [esp+4]
00443f47 6a05            push    5
00443f49 68982b5a00      push    5A2B98h
00443f4e 6a00            push    0
00443f50 50              push    eax
00443f51 6828295900      push    592928h
00443f56 51              push    ecx
00443f57 ff15c4a35400    call    dword ptr ds:[54A3C4h]
00443f5d 33c0            xor     eax, eax
00443f5f 5e              pop     esi
00443f60 81c400020000    add     esp, 200h
00443f66 c21000          ret     10h
00443f69 8bb42408020000  mov     esi, dword ptr [esp+208h]
00443f70 57              push    edi
00443f71 8b3dd0a45400    mov     edi, dword ptr ds:[54A4D0h]
00443f77 68882c5900      push    592C88h
00443f7c 68ae060000      push    6AEh
00443f81 56              push    esi
00443f82 ffd7            call    edi
00443f84 686c2c5900      push    592C6Ch
00443f89 68b0060000      push    6B0h
00443f8e 56              push    esi
00443f8f ffd7            call    edi
00443f91 68502c5900      push    592C50h
00443f96 68b1060000      push    6B1h
00443f9b 56              push    esi
00443f9c ffd7            call    edi
00443f9e 5f              pop     edi
00443f9f b801000000      mov     eax, 1
```

정적 분석 (IDA)

동적 분석 (WinDbg)

취약점 분석 과정

하네스 작성



```
#include <Windows.h>
#include <stdio.h>

#pragma warning(disable:4996)

const wchar_t* GetWC(const char* c)
{
    const size_t cSize = strlen(c) + 1;
    wchar_t* wc = new wchar_t[cSize];
    mbstowcs(wc, c, cSize);

    return wc;
}

typedef DWORD(*ReadGimpXcfW)(const wchar_t*, void*, void*,
void*);

ReadGimpXcfW func;
```

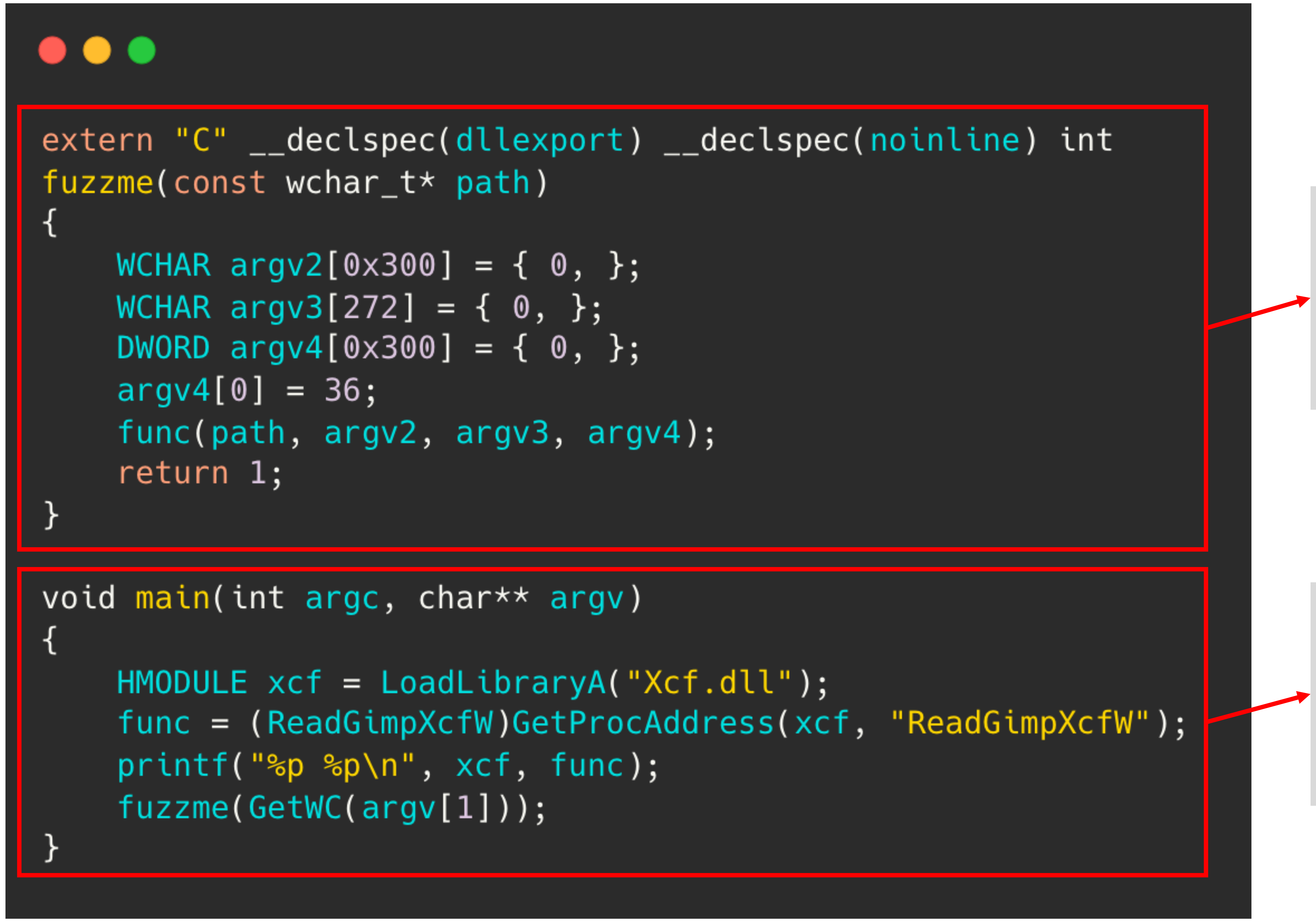
- 타겟 함수만 가져와 퍼징 수행하기 위해 제작
- WinAFL은 하네스의 타겟 함수에만 집중
- 함수 인자 분석 필요

- char 타입 문자열을 wchar 타입으로 변환

- 라이브러리 함수의 인자 타입을 기반으로 하여 함수 포인터 정의

취약점 분석 과정

하네스 작성



```
extern "C" __declspec(dllexport) __declspec(noinline) int
fuzzme(const wchar_t* path)
{
    WCHAR argv2[0x300] = { 0, };
    WCHAR argv3[272] = { 0, };
    DWORD argv4[0x300] = { 0, };
    argv4[0] = 36;
    func(path, argv2, argv3, argv4);
    return 1;
}
```

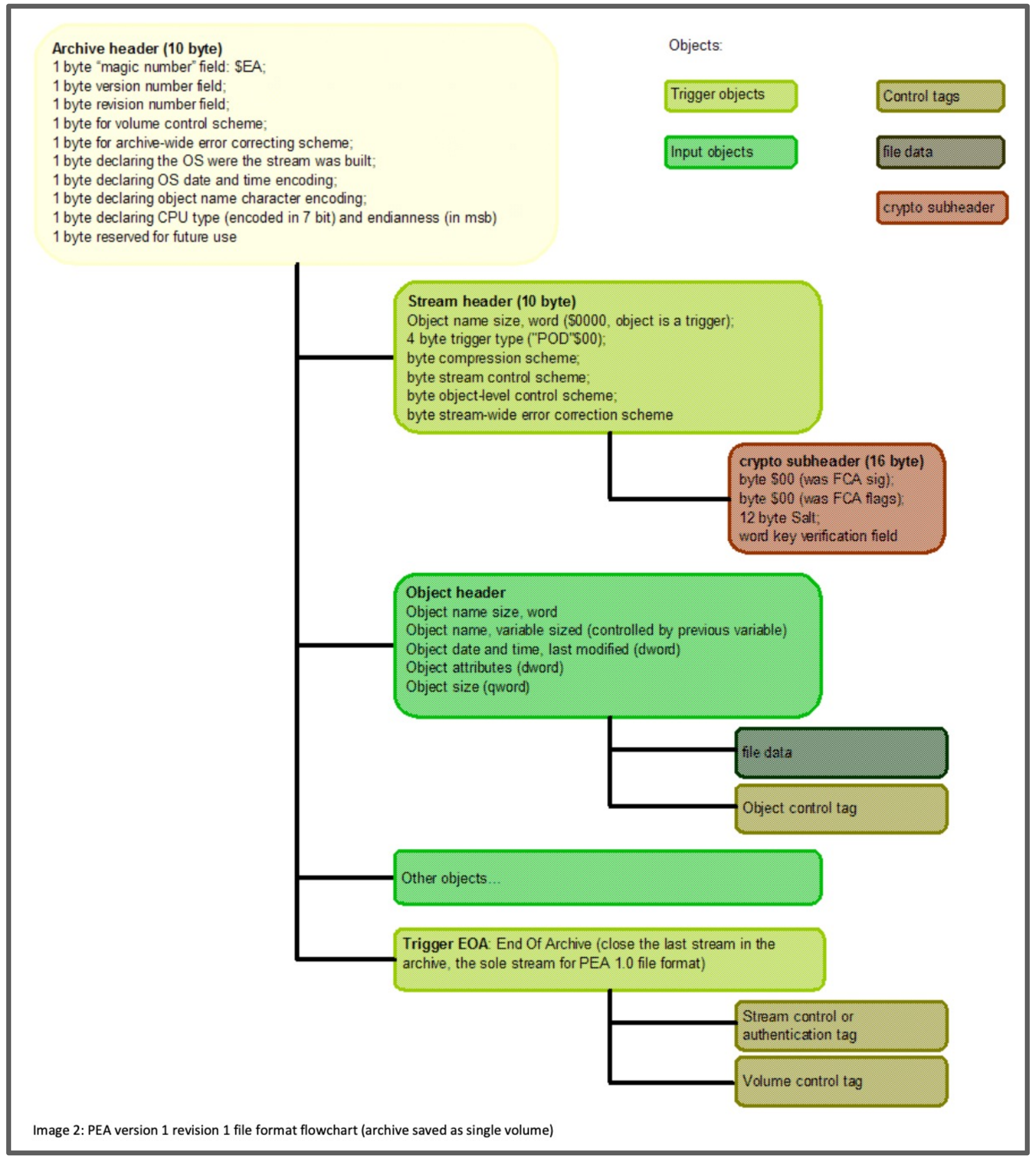
```
void main(int argc, char** argv)
{
    HMODULE xcf = LoadLibraryA("Xcf.dll");
    func = (ReadGimpXcfW)GetProcAddress(xcf, "ReadGimpXcfW");
    printf("%p %p\n", xcf, func);
    fuzzme(GetWC(argv[1]));
}
```

- 실질적인 퍼징의 대상이 되는 함수
- 타겟 함수의 인자들을 설정한 뒤 호출

- fuzzme 함수를 도와주는 함수
- dll의 타겟 함수를 가져와 함수 포인터 변수에 저장

취약점 분석 과정

시드 파일 확보



Archive header (10 byte)
1 byte "magic number" field: $EA;
1 byte version number field;
1 byte revision number field;
1 byte for volume control scheme;
1 byte for archive-wide error correcting scheme;
1 byte declaring the OS were the stream was built;
1 byte declaring OS date and time encoding;
1 byte declaring object name character encoding;
1 byte declaring CPU type (encoded in 7 bit) and endianness (in msb)
1 byte reserved for future use

Objects:
Trigger objects
Input objects
Control tags
file data
crypto subheader

Stream header (10 byte)
Object name size, word ($0000, object is a trigger);
4 byte trigger type ("POD"$00);
byte compression scheme;
byte stream control scheme;
byte object-level control scheme;
byte stream-wide error correction scheme

crypto subheader (16 byte)
byte $00 (was FCA sig);
byte $00 (was FCA flags);
12 byte Salt;
word key verification field

Object header
Object name size, word
Object name, variable sized (controlled by previous variable)
Object date and time, last modified (dword)
Object attributes (dword)
Object size (qword)

file data

Object control tag

Other objects...

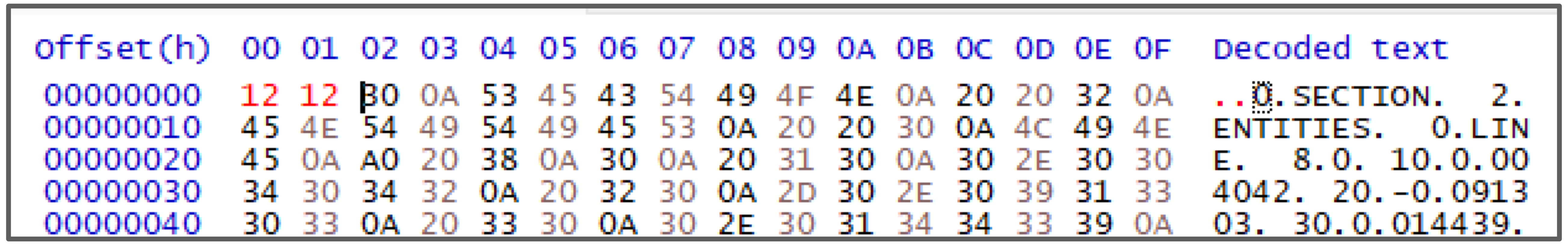
Trigger EOA: End Of Archive (close the last stream in the archive, the sole stream for PEA 1.0 file format)

Stream control or authentication tag

Volume control tag

Image 2: PEA version 1 revision 1 file format flowchart (archive saved as single volume)

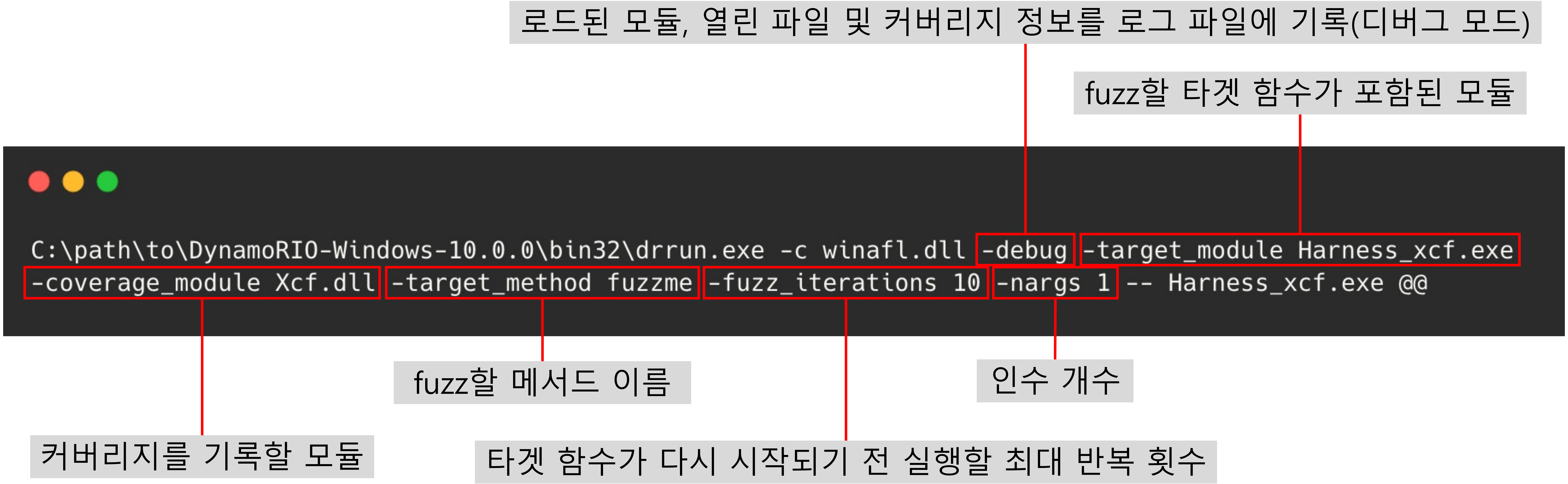
- 샘플 파일 찾기
  - http://fileformats.archiveteam.org/wiki/Main_Page
- 자체 제작
  - 파일 구조에 대한 문서를 참고해 직접 제작
  - 기존 파일 수정



```
offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   12 12 30 0A 53 45 43 54 49 4F 4E 0A 20 20 32 0A  ..0.SECTION.  2.
00000010   45 4E 54 49 54 49 45 53 0A 20 20 30 0A 4C 49 4E  ENTITIES.  0.LIN
00000020   45 0A A0 20 38 0A 30 0A 20 31 30 0A 30 2E 30 30  E. 8.0. 10.0.00
00000030   34 30 34 32 0A 20 32 30 0A 2D 30 2E 30 39 31 33  4042. 20.-0.0913
00000040   30 33 0A 20 33 30 0A 30 2E 30 31 34 34 33 39 0A  03. 30.0.014439.
```

취약점 분석 과정

실행 전 커버리지 확인



```
C:\path\to\DynamoRIO-Windows-10.0.0\bin32\drrun.exe -c winafl.dll -debug -target_module Harness_xcf.exe
-coverage_module Xcf.dll -target_method fuzzme -fuzz_iterations 10 -nargs 1 -- Harness_xcf.exe @@
```

로드된 모듈, 열린 파일 및 커버리지 정보를 로그 파일에 기록(디버그 모드)

fuzz할 타겟 함수가 포함된 모듈

커버리지를 기록할 모듈

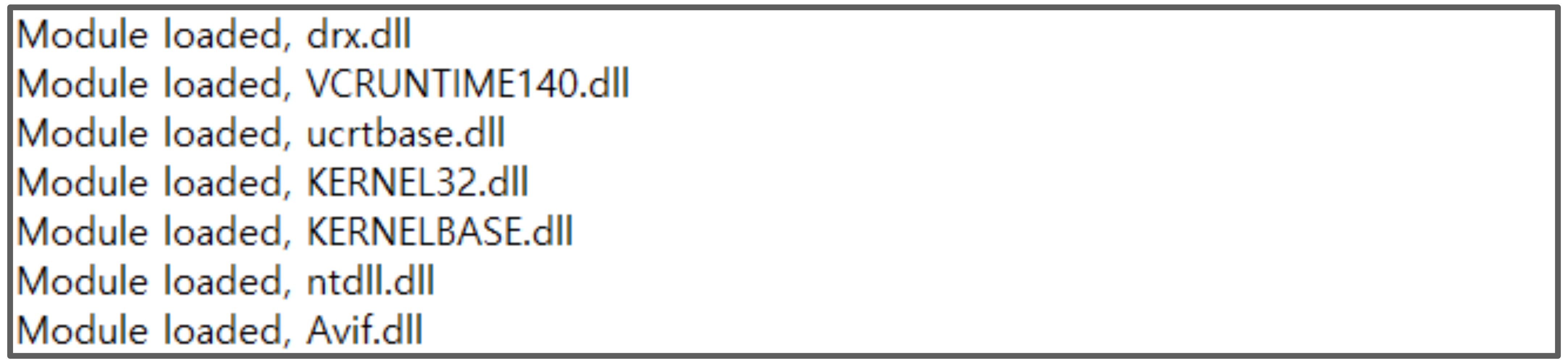
fuzz할 메서드 이름

타겟 함수가 다시 시작되기 전 실행할 최대 반복 횟수

인수 개수

취약점 분석 과정

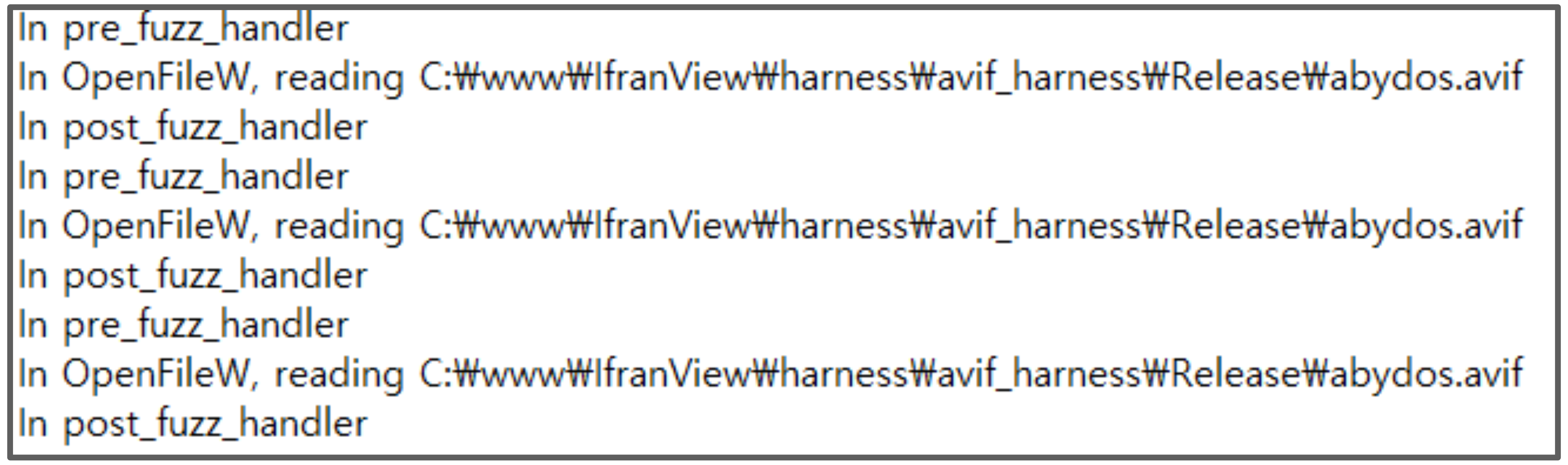
실행 전 커버리지 확인



```
Module loaded, drx.dll
Module loaded, VCRUNTIME140.dll
Module loaded, ucrtbase.dll
Module loaded, KERNEL32.dll
Module loaded, KERNELBASE.dll
Module loaded, ntdll.dll
Module loaded, Avif.dll
```

Dynamorio가 실행 시작하면서 DLL들을 메모리에 로드하는 과정 확인

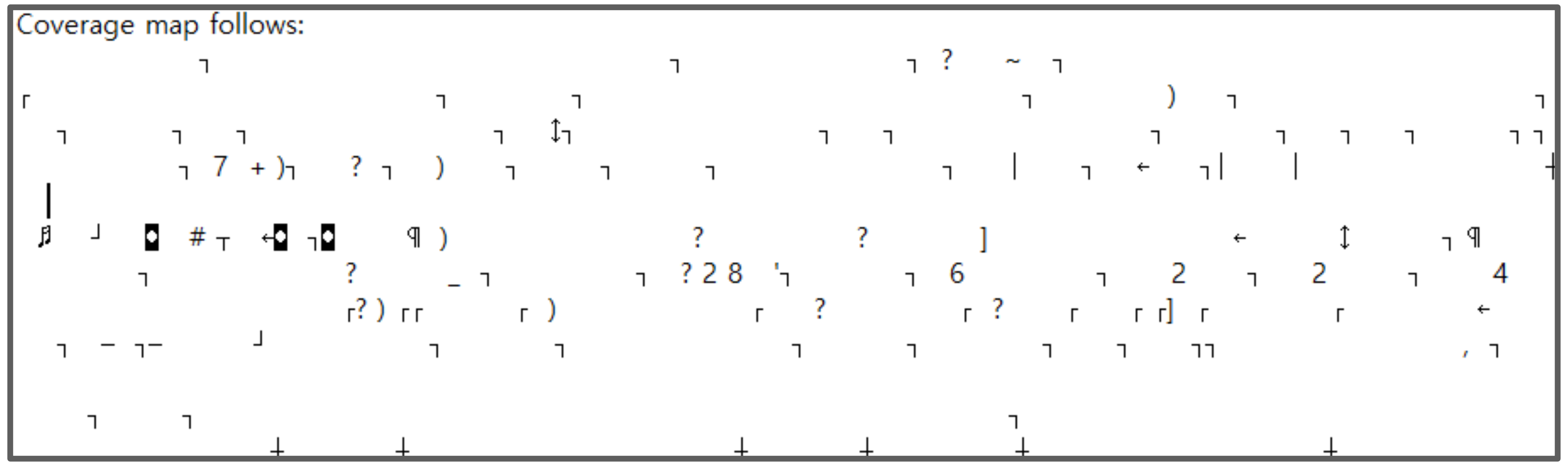
(타겟 DLL명과 로드될 때 인식되는 DLL명이 다른 경우 존재)



```
In pre_fuzz_handler
In OpenFileW, reading C:₩www₩IfranView₩harness₩avif_harness₩Release₩abydos.avif
In post_fuzz_handler
In pre_fuzz_handler
In OpenFileW, reading C:₩www₩IfranView₩harness₩avif_harness₩Release₩abydos.avif
In post_fuzz_handler
In pre_fuzz_handler
In OpenFileW, reading C:₩www₩IfranView₩harness₩avif_harness₩Release₩abydos.avif
In post_fuzz_handler
```

퍼징 전처리, 시드 파일 테스트, 퍼징 후처리 반복 과정 확인

(하네스 잘못 작성 시 시드 파일 읽기 실패)



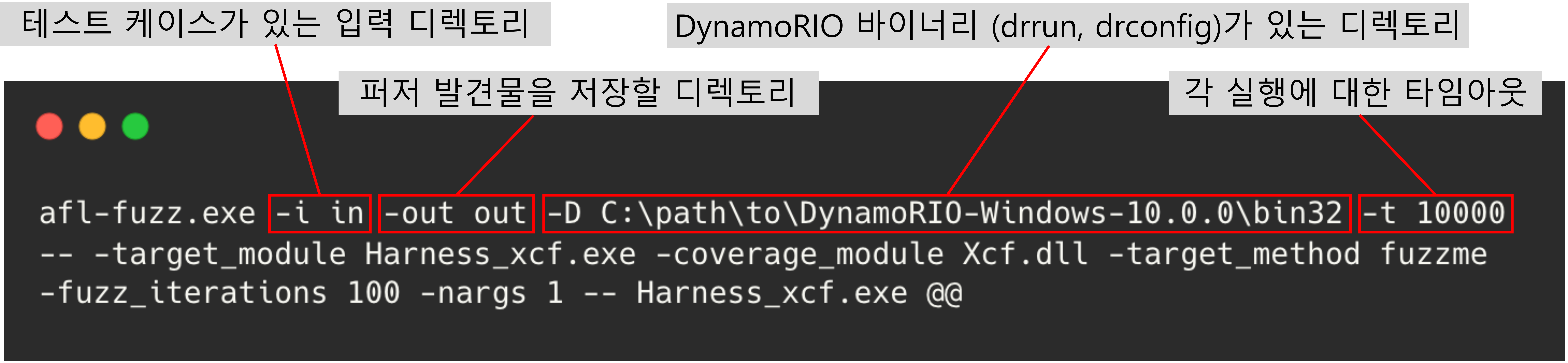
```
Coverage map follows:
              ┐                              ┐                 ┐ ?    ~  ┐
┌                            ┐        ┐                           ┐        )   ┐                     ┐
  ┐        ┐    ┐                  ┐   ↕┐                ┐    ┐                  ┐        ┐    ┐    ┐       ┐┐
           ┐ 7  + )┐    ? ┐   )    ┐      ┐       ┐                 ┐    |    ┐   ←    ┐|     |            ┼
 |
♬   ┘   ◘  # ┬  ←◘ ┐◘      ¶ )                    ?             ?         ]                   ←        ↕       ┐¶
        ┐                 ?        _ ┐            ┐ ?28  '┐          ┐ 6            ┐    2    ┐    2     ┐     4
                          ┌?) ┌┌       ┌ )                ┌    ?            ┌ ?     ┌     ┌┌] ┌         ┌          ←
  ┐  ─ ┐─      ┘              ┐        ┐                ┐       ┐          ┐     ┐    ┐┐                 , ┐

    ┐      ┐                                                       ┐
                   ┼        ┼                        ┼        ┼           ┼                    ┼
```

타겟 모듈(dll)의 코드 커버리지 맵 확인

(내부 함수 진입 실패 시 커버리지 측정 안 됨)

취약점 분석 과정

WinAFL 퍼징



테스트 케이스가 있는 입력 디렉토리

DynamoRIO 바이너리 (drrun, drconfig)가 있는 디렉토리

퍼저 발견물을 저장할 디렉토리

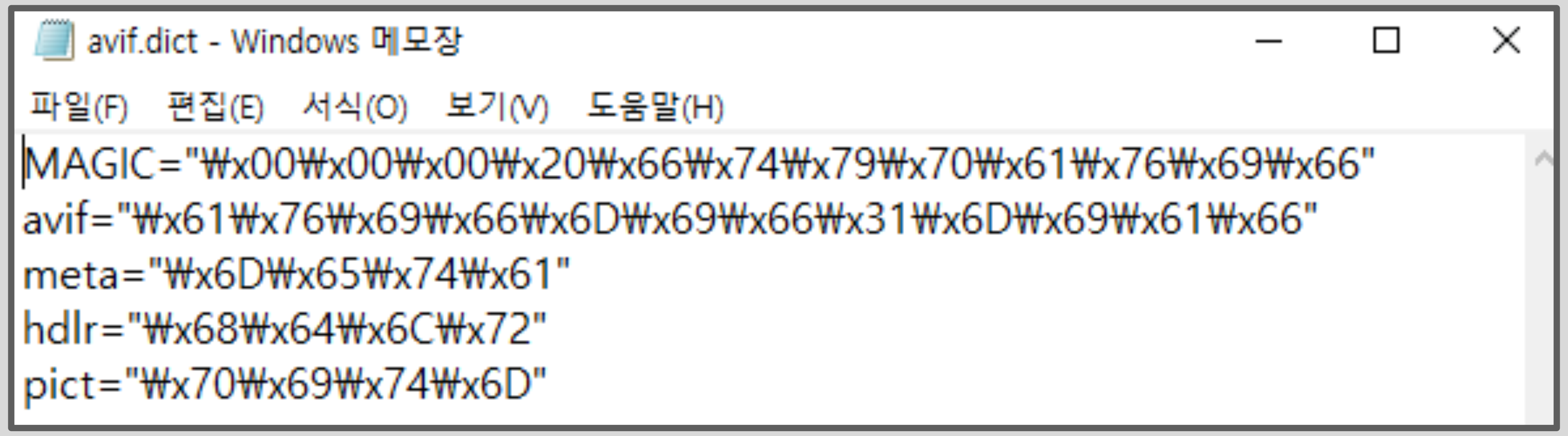
각 실행에 대한 타임아웃

```
afl-fuzz.exe -i in -out out -D C:\path\to\DynamoRIO-Windows-10.0.0\bin32 -t 10000
-- -target_module Harness_xcf.exe -coverage_module Xcf.dll -target_method fuzzme
-fuzz_iterations 100 -nargs 1 -- Harness_xcf.exe @@
```

[옵션]

-M [fuzzer명] / -S [fuzzer명] : Master-Slave 병렬 퍼징

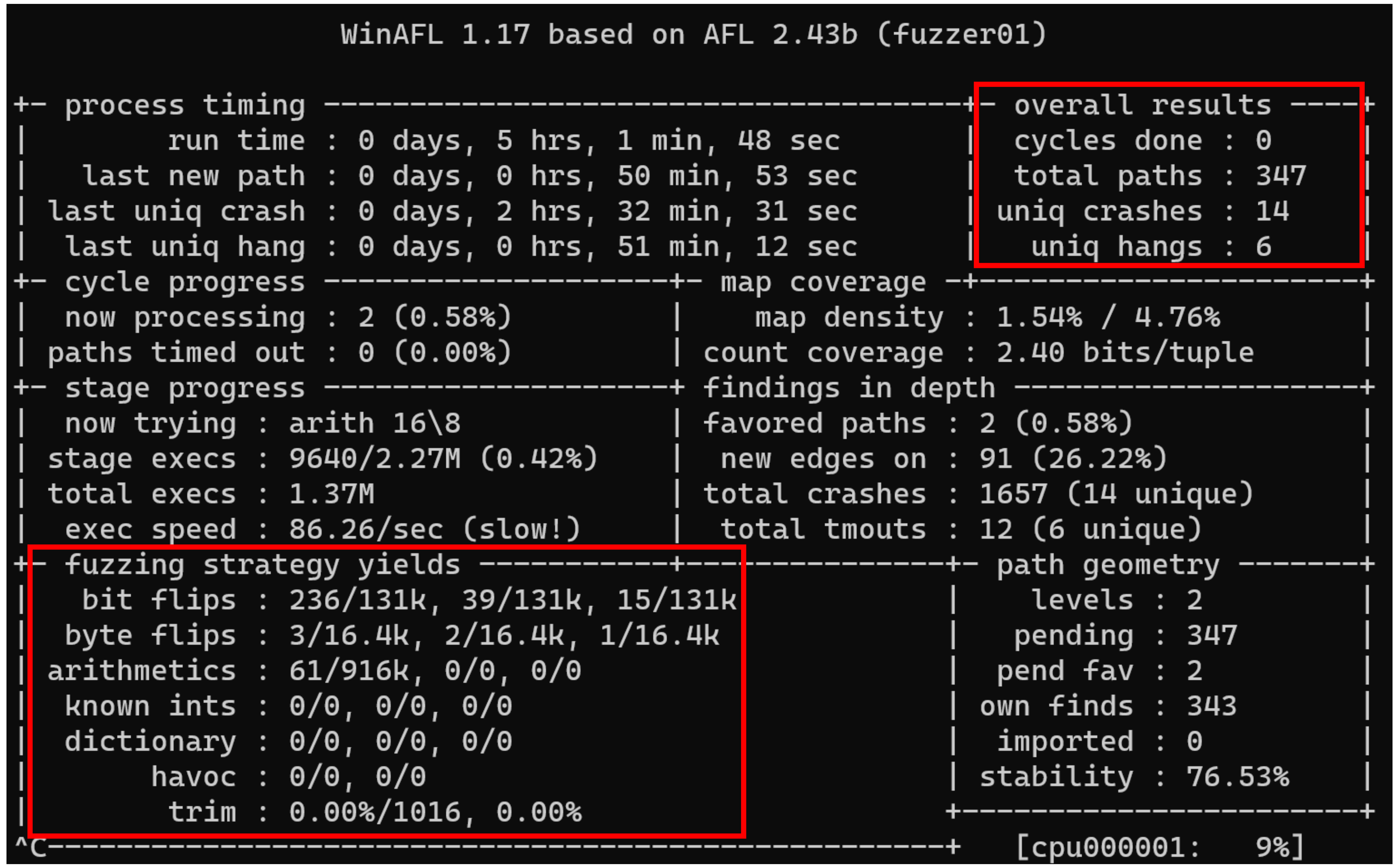
-x [dictionary 파일 위치] : 뮤테이션에 활용할 토큰 목록 제시


avif.dict - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

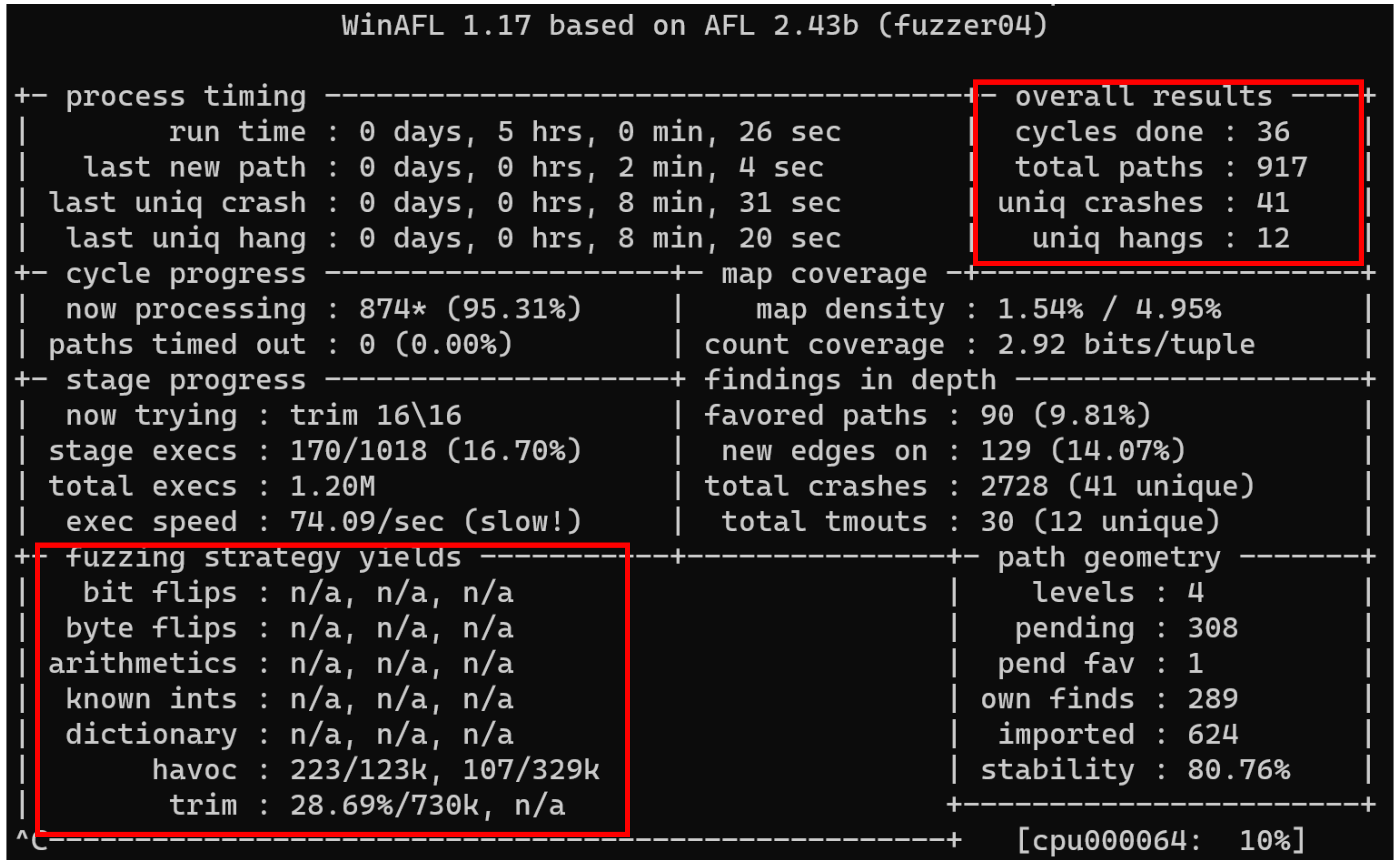
```
MAGIC="₩x00₩x00₩x00₩x20₩x66₩x74₩x79₩x70₩x61₩x76₩x69₩x66"
avif="₩x61₩x76₩x69₩x66₩x6D₩x69₩x66₩x31₩x6D₩x69₩x61₩x66"
meta="₩x6D₩x65₩x74₩x61"
hdlr="₩x68₩x64₩x6C₩x72"
pict="₩x70₩x69₩x74₩x6D"
```

취약점 분석 과정

WinAFL 퍼징



```
                WinAFL 1.17 based on AFL 2.43b (fuzzer01)

+- process timing -------------------------------------+- overall results ----+
|        run time : 0 days, 5 hrs, 1 min, 48 sec       |  cycles done : 0     |
|   last new path : 0 days, 0 hrs, 50 min, 53 sec      |  total paths : 347   |
| last uniq crash : 0 days, 2 hrs, 32 min, 31 sec      | uniq crashes : 14    |
|  last uniq hang : 0 days, 0 hrs, 51 min, 12 sec      |   uniq hangs : 6     |
+- cycle progress --------------------+- map coverage -+----------------------+
|  now processing : 2 (0.58%)         |    map density : 1.54% / 4.76%        |
| paths timed out : 0 (0.00%)         | count coverage : 2.40 bits/tuple      |
+- stage progress --------------------+ findings in depth --------------------+
|  now trying : arith 16\8            | favored paths : 2 (0.58%)             |
| stage execs : 9640/2.27M (0.42%)    |  new edges on : 91 (26.22%)           |
| total execs : 1.37M                 | total crashes : 1657 (14 unique)      |
|  exec speed : 86.26/sec (slow!)     |  total tmouts : 12 (6 unique)         |
+- fuzzing strategy yields -----------+---------------+- path geometry -------+
|   bit flips : 236/131k, 39/131k, 15/131k            |    levels : 2         |
|  byte flips : 3/16.4k, 2/16.4k, 1/16.4k             |   pending : 347       |
| arithmetics : 61/916k, 0/0, 0/0                     |  pend fav : 2         |
|  known ints : 0/0, 0/0, 0/0                         | own finds : 343       |
|  dictionary : 0/0, 0/0, 0/0                         |  imported : 0         |
|       havoc : 0/0, 0/0                              | stability : 76.53%    |
|        trim : 0.00%/1016, 0.00%                     +-----------------------+
^C----------------------------------------------------+   [cpu000001:    9%]
```

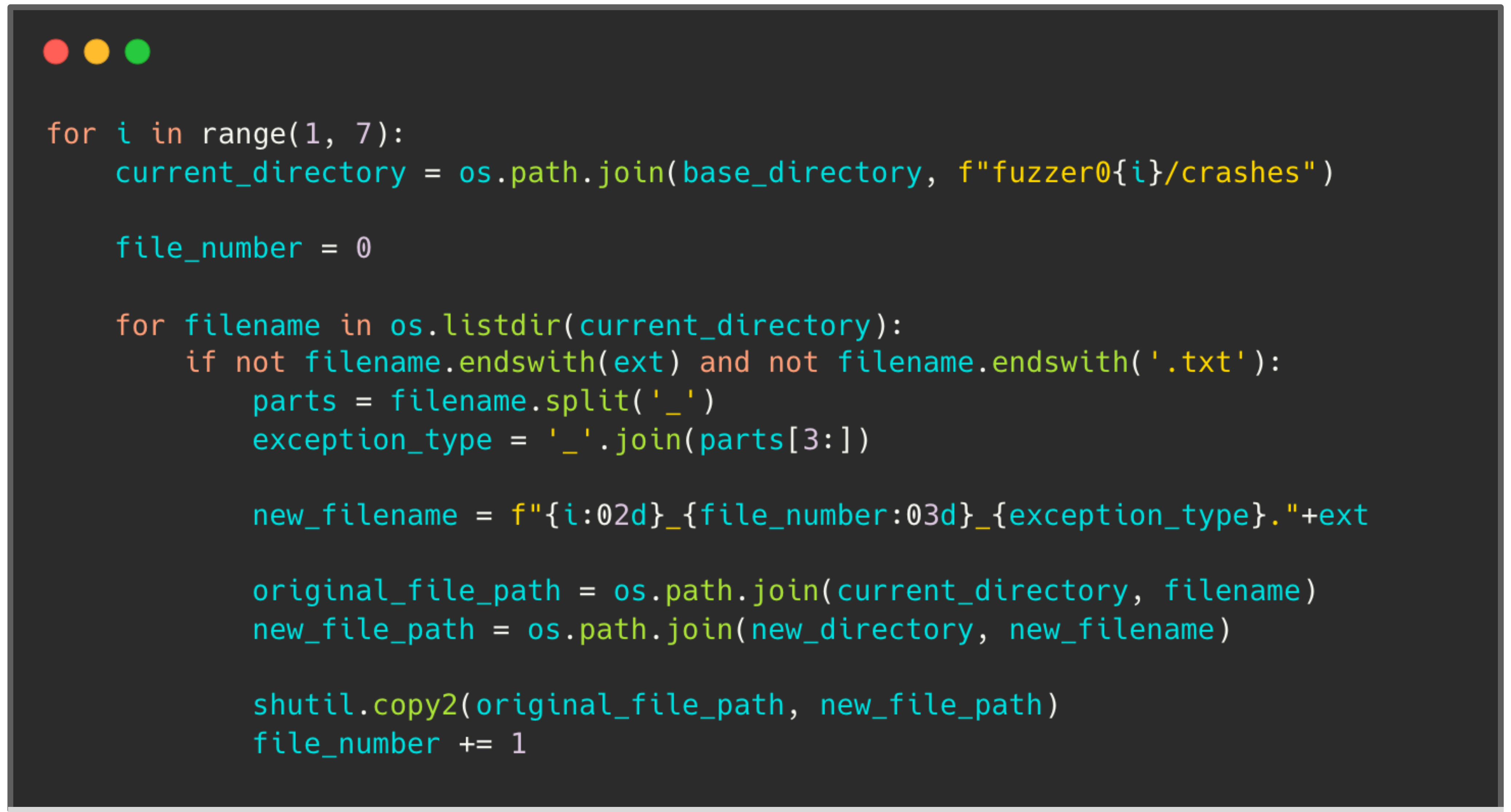


```
                WinAFL 1.17 based on AFL 2.43b (fuzzer04)

+- process timing -------------------------------------+- overall results ----+
|        run time : 0 days, 5 hrs, 0 min, 26 sec       |  cycles done : 36    |
|   last new path : 0 days, 0 hrs, 2 min, 4 sec        |  total paths : 917   |
| last uniq crash : 0 days, 0 hrs, 8 min, 31 sec       | uniq crashes : 41    |
|  last uniq hang : 0 days, 0 hrs, 8 min, 20 sec       |   uniq hangs : 12    |
+- cycle progress --------------------+- map coverage -+----------------------+
|  now processing : 874* (95.31%)     |    map density : 1.54% / 4.95%        |
| paths timed out : 0 (0.00%)         | count coverage : 2.92 bits/tuple      |
+- stage progress --------------------+ findings in depth --------------------+
|  now trying : trim 16\16            | favored paths : 90 (9.81%)            |
| stage execs : 170/1018 (16.70%)     |  new edges on : 129 (14.07%)          |
| total execs : 1.20M                 | total crashes : 2728 (41 unique)      |
|  exec speed : 74.09/sec (slow!)     |  total tmouts : 30 (12 unique)        |
+- fuzzing strategy yields -----------+---------------+- path geometry -------+
|   bit flips : n/a, n/a, n/a                         |    levels : 4         |
|  byte flips : n/a, n/a, n/a                         |   pending : 308       |
| arithmetics : n/a, n/a, n/a                         |  pend fav : 1         |
|  known ints : n/a, n/a, n/a                         | own finds : 289       |
|  dictionary : n/a, n/a, n/a                         |  imported : 624       |
|       havoc : 223/123k, 107/329k                    | stability : 80.76%    |
|        trim : 28.69%/730k, n/a                      +-----------------------+
^C----------------------------------------------------+   [cpu000064:   10%]
```

- 마스터-슬레이브 모드로 병렬 퍼징 수행
- 슬레이브는 뮤테이션 전략을 순서대로가 아닌 랜덤하게 수행
- 각 퍼저별로 유니크 크래시가 나옴

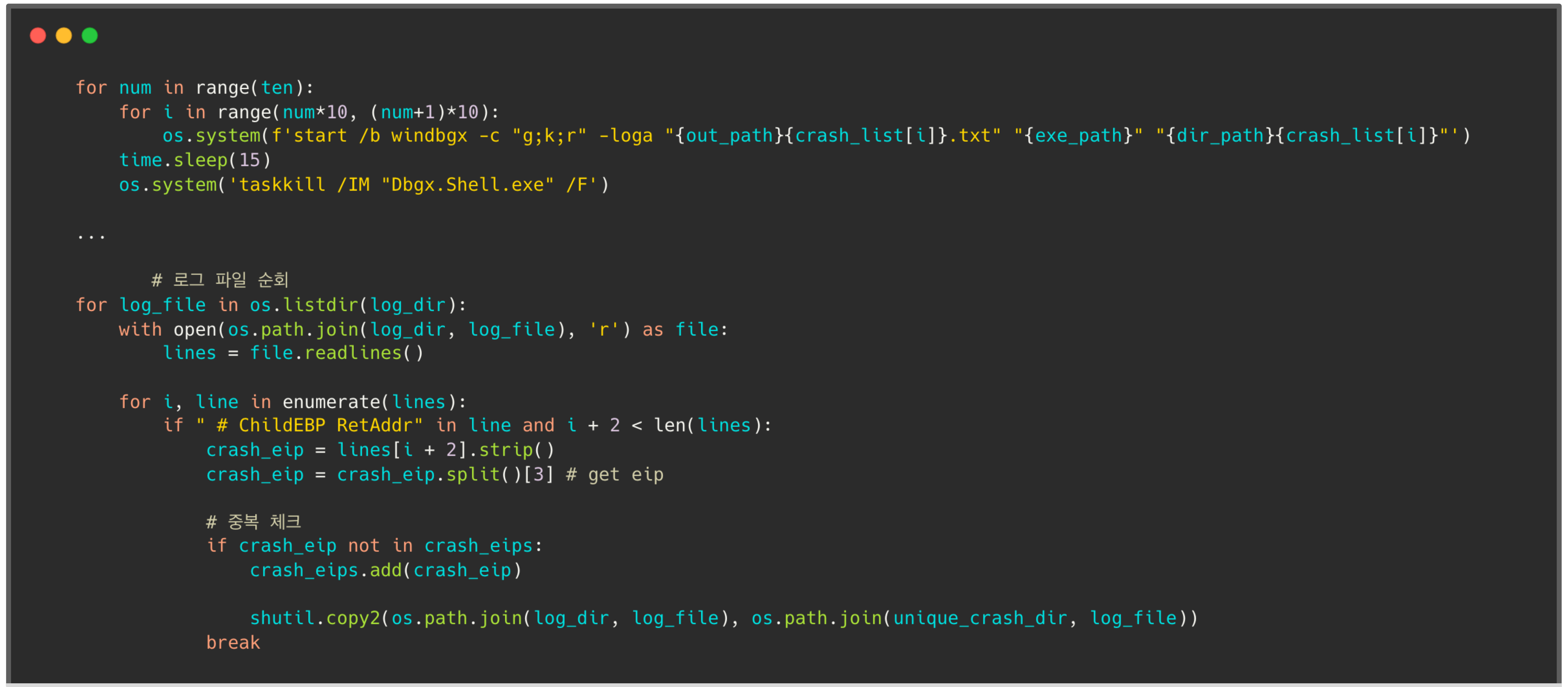
취약점 분석 과정

크래시 취합&분류



```
for i in range(1, 7):
    current_directory = os.path.join(base_directory, f"fuzzer0{i}/crashes")

    file_number = 0

    for filename in os.listdir(current_directory):
        if not filename.endswith(ext) and not filename.endswith('.txt'):
            parts = filename.split('_')
            exception_type = '_'.join(parts[3:])

            new_filename = f"{i:02d}_{file_number:03d}_{exception_type}."+ext

            original_file_path = os.path.join(current_directory, filename)
            new_file_path = os.path.join(new_directory, new_filename)

            shutil.copy2(original_file_path, new_file_path)
            file_number += 1
```

- 퍼저별로 흩어진 크래시들을 한 곳에 취합
- Renaming + 적절한 확장자 추가

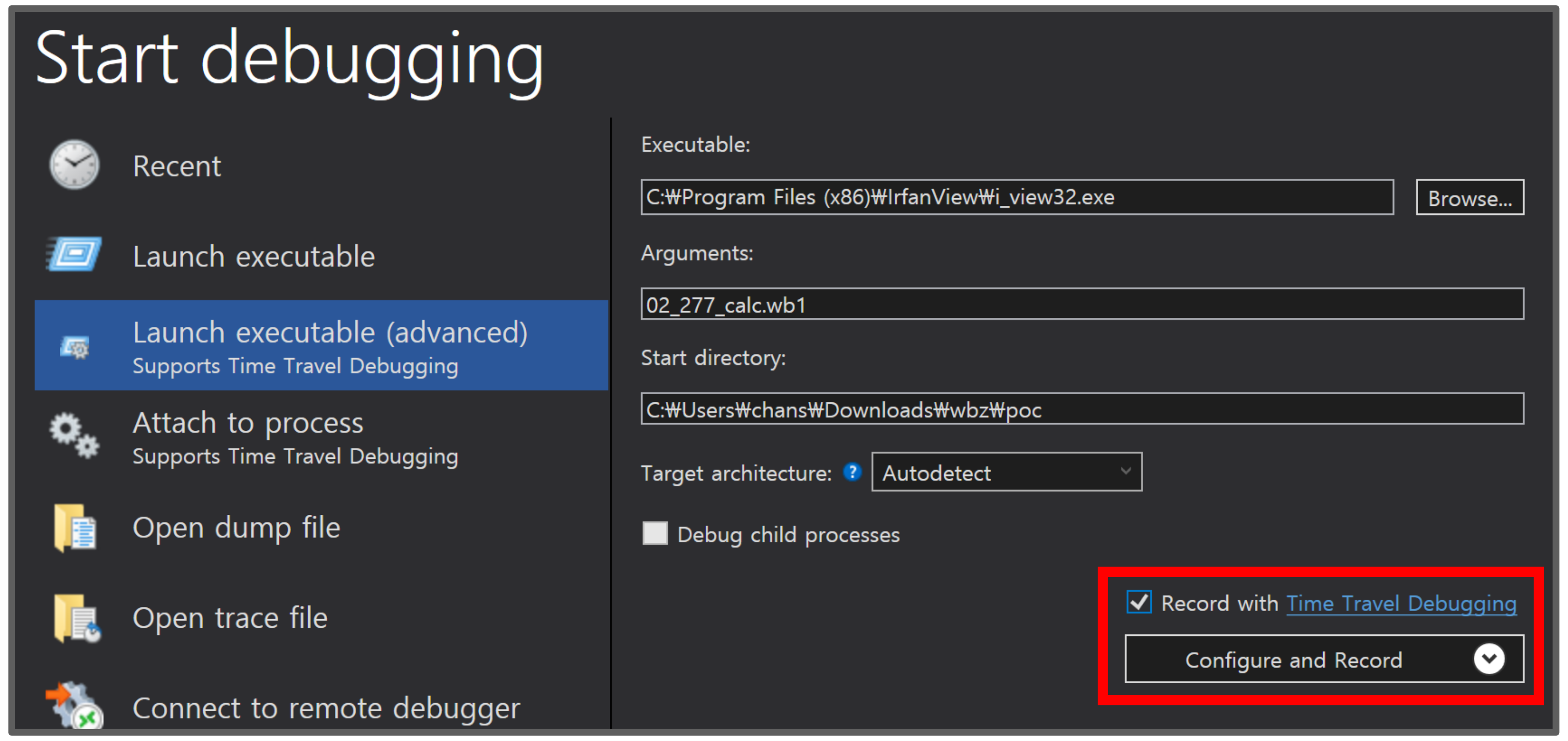


```
for num in range(ten):
    for i in range(num*10, (num+1)*10):
        os.system(f'start /b windbgx -c "g;k;r" -loga "{out_path}{crash_list[i]}.txt" "{exe_path}" "{dir_path}{crash_list[i]}"')
    time.sleep(15)
    os.system('taskkill /IM "Dbgx.Shell.exe" /F')

...

        # 로그 파일 순회
for log_file in os.listdir(log_dir):
    with open(os.path.join(log_dir, log_file), 'r') as file:
        lines = file.readlines()

    for i, line in enumerate(lines):
        if " # ChildEBP RetAddr" in line and i + 2 < len(lines):
            crash_eip = lines[i + 2].strip()
            crash_eip = crash_eip.split()[3] # get eip

            # 중복 체크
            if crash_eip not in crash_eips:
                crash_eips.add(crash_eip)

                shutil.copy2(os.path.join(log_dir, log_file), os.path.join(unique_crash_dir, log_file))
            break
```

- 크래시 재현 여부 확인, 레지스터 및 콜 스택 정보 수집
- eip 레지스터 기준으로 중복 크래시 재분류

취약점 분석 과정

WinDbg + TTD(Time Travel debugging)



Start debugging

- Recent
- Launch executable
- Launch executable (advanced)
  Supports Time Travel Debugging
- Attach to process
  Supports Time Travel Debugging
- Open dump file
- Open trace file
- Connect to remote debugger

Executable: C:₩Program Files (x86)₩IrfanView₩i_view32.exe [Browse...]

Arguments: 02_277_calc.wb1

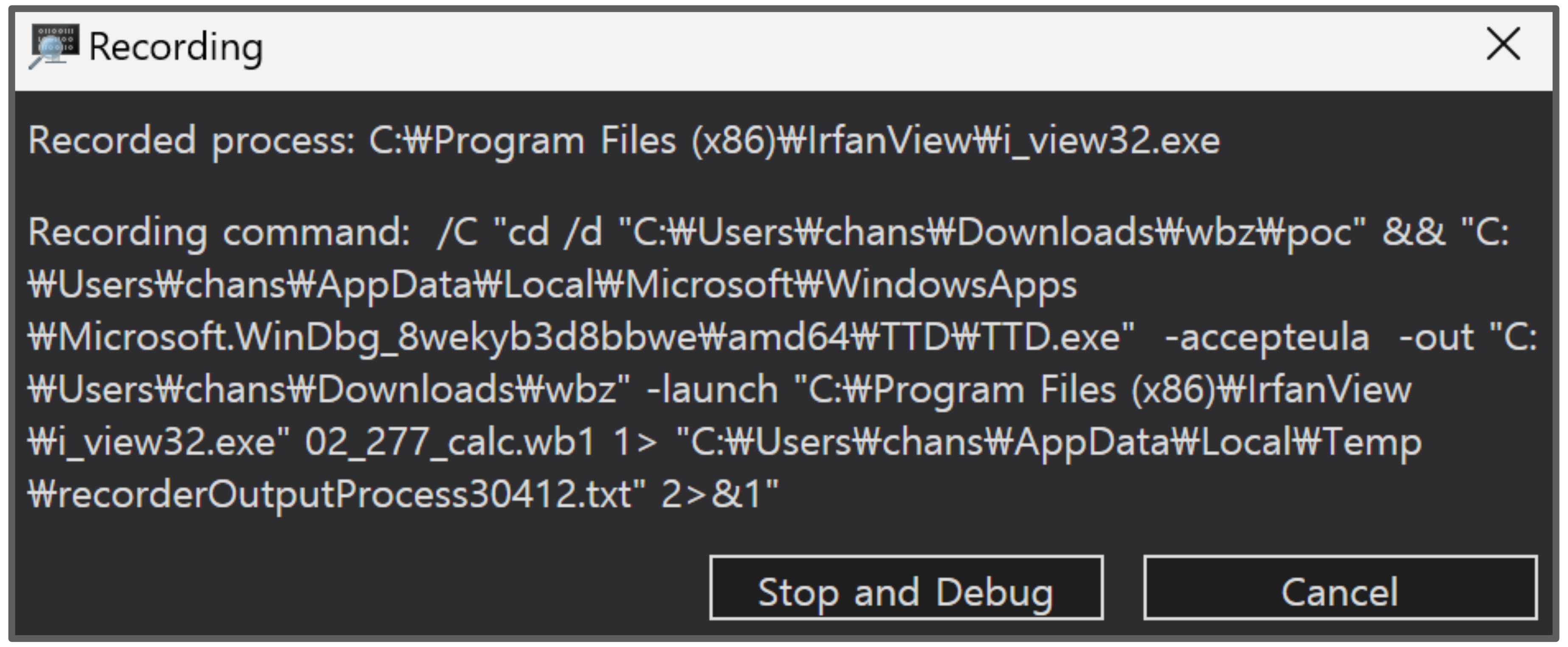
Start directory: C:₩Users₩chans₩Downloads₩wbz₩poc

Target architecture: Autodetect

□ Debug child processes

☑ Record with Time Travel Debugging

[Configure and Record]

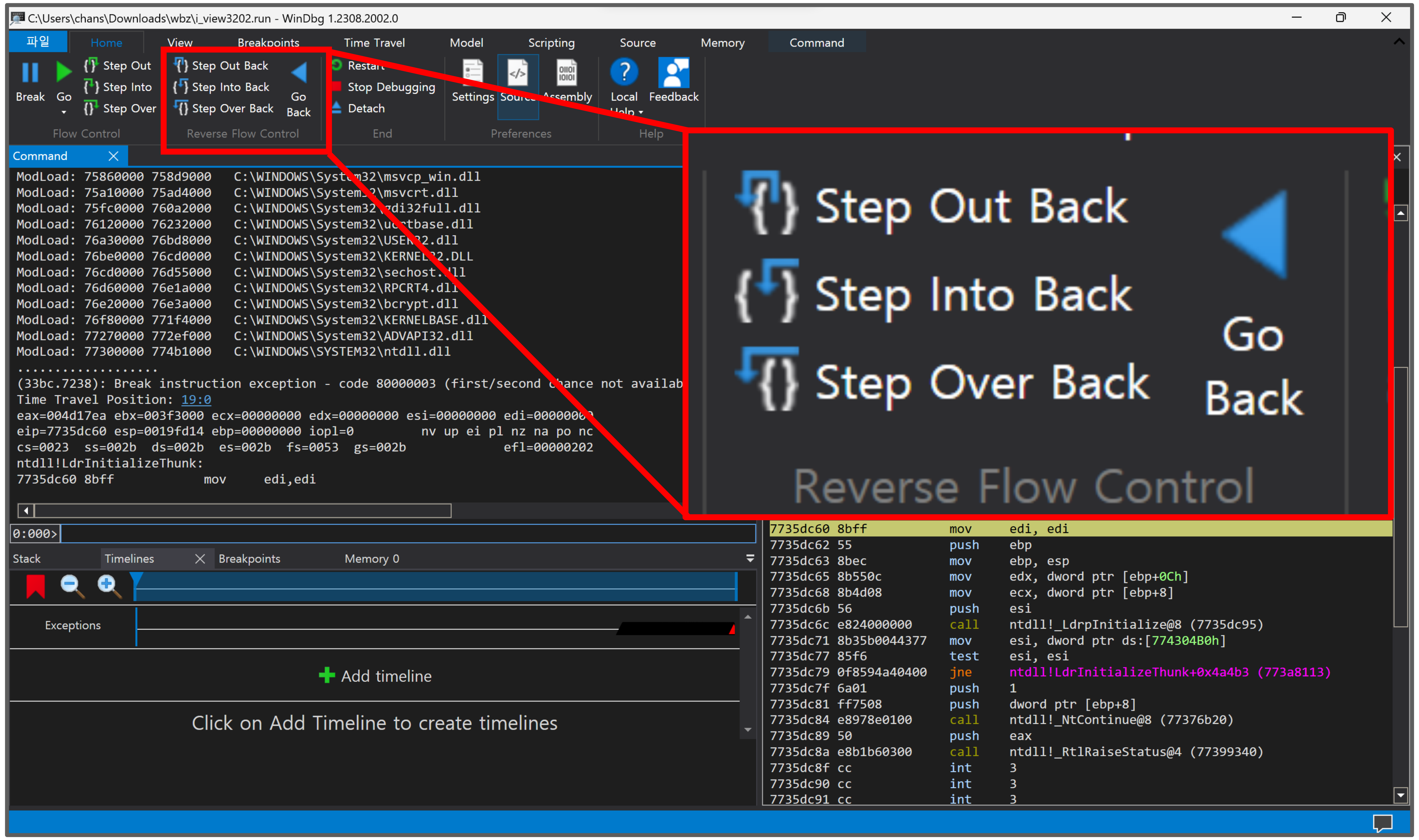


Recording

Recorded process: C:₩Program Files (x86)₩IrfanView₩i_view32.exe

Recording command: /C "cd /d "C:₩Users₩chans₩Downloads₩wbz₩poc" && "C:₩Users₩chans₩AppData₩Local₩Microsoft₩WindowsApps₩Microsoft.WinDbg_8wekyb3d8bbwe₩amd64₩TTD₩TTD.exe" -accepteula -out "C:₩Users₩chans₩Downloads₩wbz" -launch "C:₩Program Files (x86)₩IrfanView₩i_view32.exe" 02_277_calc.wb1 1> "C:₩Users₩chans₩AppData₩Local₩Temp₩recorderOutputProcess30412.txt" 2>&1"

[Stop and Debug] [Cancel]



C:\Users\chans\Downloads\wbz\i_view3202.run - WinDbg 1.2308.2002.0

```
ModLoad: 75860000 758d9000   C:\WINDOWS\System32\msvcp_win.dll
ModLoad: 75a10000 75ad4000   C:\WINDOWS\System32\msvcrt.dll
ModLoad: 75fc0000 760a2000   C:\WINDOWS\System32\gdi32full.dll
ModLoad: 76120000 76232000   C:\WINDOWS\System32\ucrtbase.dll
ModLoad: 76a30000 76bd8000   C:\WINDOWS\System32\USER32.dll
ModLoad: 76be0000 76cd0000   C:\WINDOWS\System32\KERNEL32.DLL
ModLoad: 76cd0000 76d55000   C:\WINDOWS\System32\sechost.dll
ModLoad: 76d60000 76e1a000   C:\WINDOWS\System32\RPCRT4.dll
ModLoad: 76e20000 76e3a000   C:\WINDOWS\System32\bcrypt.dll
ModLoad: 76f80000 771f4000   C:\WINDOWS\System32\KERNELBASE.dll
ModLoad: 77270000 772ef000   C:\WINDOWS\System32\ADVAPI32.dll
ModLoad: 77300000 774b1000   C:\WINDOWS\SYSTEM32\ntdll.dll
...................
(33bc.7238): Break instruction exception - code 80000003 (first/second chance not availab
Time Travel Position: 19:0
eax=004d17ea ebx=003f3000 ecx=00000000 edx=00000000 esi=00000000 edi=00000000
eip=7735dc60 esp=0019fd14 ebp=00000000 iopl=0         nv up ei pl nz na po nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000202
ntdll!LdrInitializeThunk:
7735dc60 8bff            mov     edi,edi
```

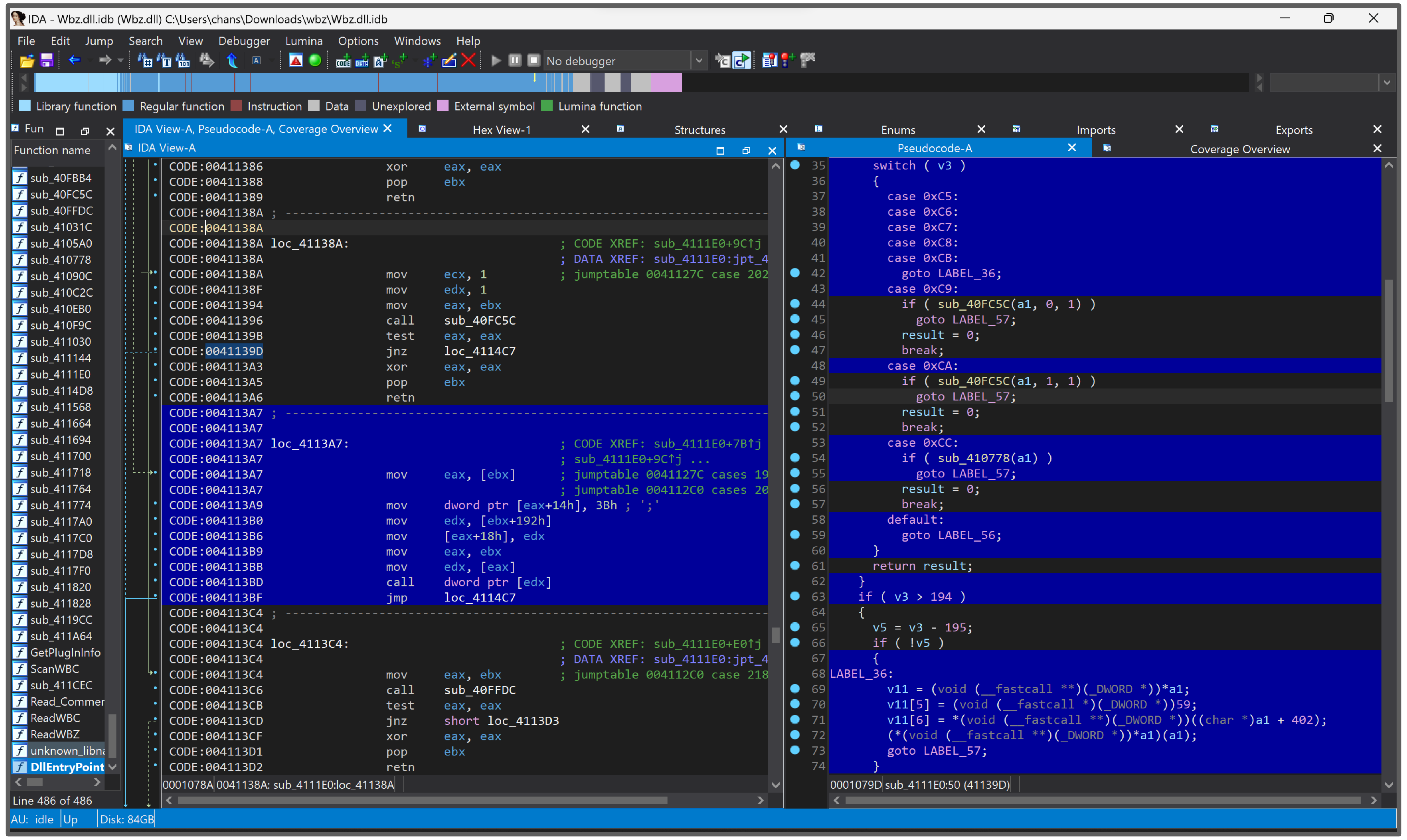
```
0:000>
```

```
7735dc60 8bff            mov     edi, edi
7735dc62 55              push    ebp
7735dc63 8bec            mov     ebp, esp
7735dc65 8b550c          mov     edx, dword ptr [ebp+0Ch]
7735dc68 8b4d08          mov     ecx, dword ptr [ebp+8]
7735dc6b 56              push    esi
7735dc6c e824000000      call    ntdll!_LdrpInitialize@8 (7735dc95)
7735dc71 8b35b0044377    mov     esi, dword ptr ds:[774304B0h]
7735dc77 85f6            test    esi, esi
7735dc79 0f8594a40400    jne     ntdll!LdrInitializeThunk+0x4a4b3 (773a8113)
7735dc7f 6a01            push    1
7735dc81 ff7508          push    dword ptr [ebp+8]
7735dc84 e8978e0100      call    ntdll!_NtContinue@8 (77376b20)
7735dc89 50              push    eax
7735dc8a e8b1b60300      call    ntdll!_RtlRaiseStatus@4 (77399340)
7735dc8f cc              int     3
7735dc90 cc              int     3
7735dc91 cc              int     3
```

취약점 분석 과정

IDA + Lighthouse

```
C:\path\to\DynamoRIO-Windows-10.0.0\bin32\drrun.exe -t drcov -- "C:\path\to\binary.exe" crash_file
```



```
CODE:00411386                 xor     eax, eax
CODE:00411388                 pop     ebx
CODE:00411389                 retn
CODE:0041138A ; ---------------------------------------------------------------
CODE:0041138A
CODE:0041138A loc_41138A:                             ; CODE XREF: sub_4111E0+9C↑j
CODE:0041138A                                         ; DATA XREF: sub_4111E0:jpt_4
CODE:0041138A                 mov     ecx, 1          ; jumptable 0041127C case 202
CODE:0041138F                 mov     edx, 1
CODE:00411394                 mov     eax, ebx
CODE:00411396                 call    sub_40FC5C
CODE:0041139B                 test    eax, eax
CODE:0041139D                 jnz     loc_4114C7
CODE:004113A3                 xor     eax, eax
CODE:004113A5                 pop     ebx
CODE:004113A6                 retn
CODE:004113A7 ; ---------------------------------------------------------------
CODE:004113A7
CODE:004113A7 loc_4113A7:                             ; CODE XREF: sub_4111E0+7B↑j
CODE:004113A7                                         ; sub_4111E0+9C↑j ...
CODE:004113A7                 mov     eax, [ebx]      ; jumptable 0041127C cases 19
CODE:004113A7                                         ; jumptable 004112C0 cases 20
CODE:004113A9                 mov     dword ptr [eax+14h], 3Bh ; ';'
CODE:004113B0                 mov     edx, [ebx+192h]
CODE:004113B6                 mov     [eax+18h], edx
CODE:004113B9                 mov     eax, ebx
CODE:004113BB                 mov     edx, [eax]
CODE:004113BD                 call    dword ptr [edx]
CODE:004113BF                 jmp     loc_4114C7
CODE:004113C4 ; ---------------------------------------------------------------
CODE:004113C4
CODE:004113C4 loc_4113C4:                             ; CODE XREF: sub_4111E0+E0↑j
CODE:004113C4                                         ; DATA XREF: sub_4111E0:jpt_4
CODE:004113C4                 mov     eax, ebx        ; jumptable 004112C0 case 218
CODE:004113C6                 call    sub_40FFDC
CODE:004113CB                 test    eax, eax
CODE:004113CD                 jnz     short loc_4113D3
CODE:004113CF                 xor     eax, eax
CODE:004113D1                 pop     ebx
CODE:004113D2                 retn
0001078A 0041138A: sub_4111E0:loc_41138A
```

```
    switch ( v3 )
    {
      case 0xC5:
      case 0xC6:
      case 0xC7:
      case 0xC8:
      case 0xCB:
        goto LABEL_36;
      case 0xC9:
        if ( sub_40FC5C(a1, 0, 1) )
          goto LABEL_57;
        result = 0;
        break;
      case 0xCA:
        if ( sub_40FC5C(a1, 1, 1) )
          goto LABEL_57;
        result = 0;
        break;
      case 0xCC:
        if ( sub_410778(a1) )
          goto LABEL_57;
        result = 0;
        break;
      default:
        goto LABEL_56;
    }
    return result;
  }
  if ( v3 > 194 )
  {
    v5 = v3 - 195;
    if ( !v5 )
    {
LABEL_36:
      v11 = (void (__fastcall **)(_DWORD *))*a1;
      v11[5] = (void (__fastcall *)(_DWORD *))59;
      v11[6] = *(void (__fastcall **)(_DWORD *))((char *)a1 + 402);
      (*(void (__fastcall **)(_DWORD *))*a1)(a1);
      goto LABEL_57;
    }
0001079D sub_4111E0:50 (41139D)
```

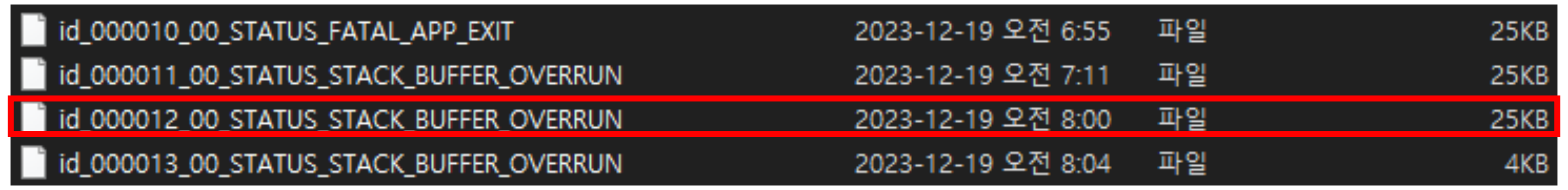
- 코드 커버리지를 시각적으로 제공
- drrun.exe의 옵션을 사용해서 커버리지 로그 파일 추출
- drcov 버전이 3인 경우 2로 낮춰줘야 라이트하우스에서 파싱 가능

https://github.com/gaasedelen/lighthouse

https://gist.github.com/wumb0/de671cc5051353fd32af4aecc811a282

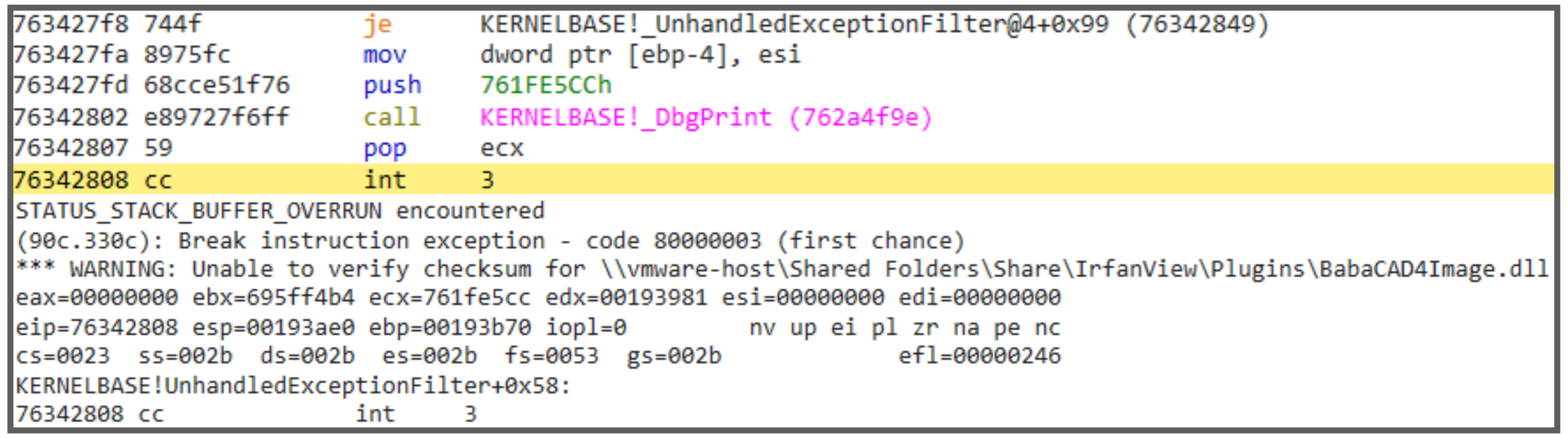
취약점 분석 과정

실전!



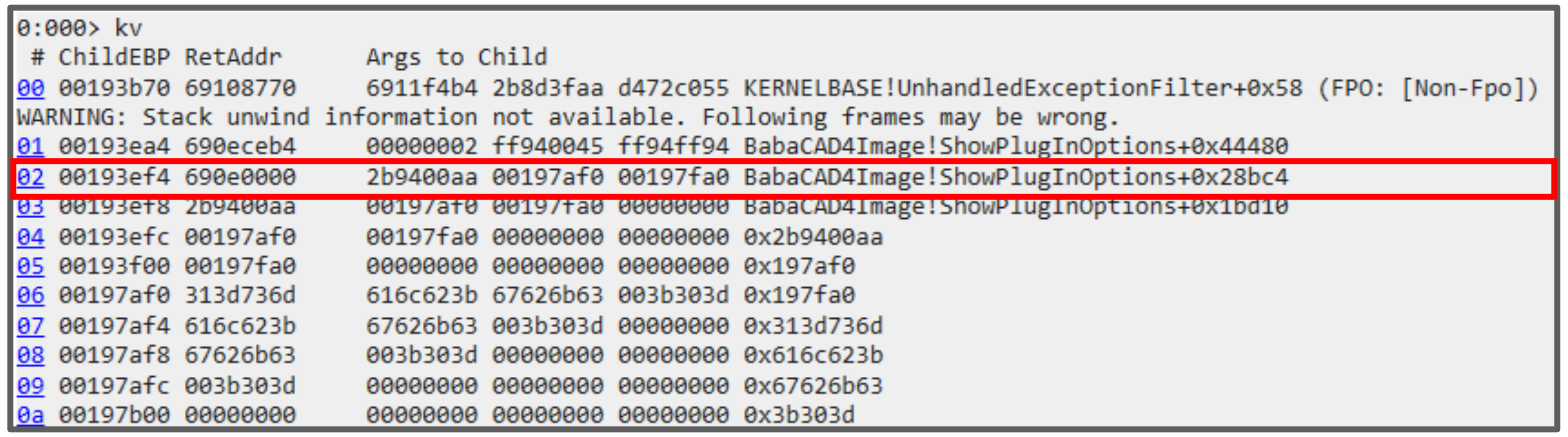
id_000010_00_STATUS_FATAL_APP_EXIT	2023-12-19 오전 6:55	파일	25KB
id_000011_00_STATUS_STACK_BUFFER_OVERRUN	2023-12-19 오전 7:11	파일	25KB
id_000012_00_STATUS_STACK_BUFFER_OVERRUN	2023-12-19 오전 8:00	파일	25KB
id_000013_00_STATUS_STACK_BUFFER_OVERRUN	2023-12-19 오전 8:04	파일	4KB

① 분석할 크래시 하나 선택



```
763427f8 744f            je      KERNELBASE!_UnhandledExceptionFilter@4+0x99 (76342849)
763427fa 8975fc          mov     dword ptr [ebp-4], esi
763427fd 68cce51f76      push    761FE5CCh
76342802 e89727f6ff      call    KERNELBASE!_DbgPrint (762a4f9e)
76342807 59              pop     ecx
76342808 cc              int     3
STATUS_STACK_BUFFER_OVERRUN encountered
(90c.330c): Break instruction exception - code 80000003 (first chance)
*** WARNING: Unable to verify checksum for \\vmware-host\Shared Folders\Share\IrfanView\Plugins\BabaCAD4Image.dll
eax=00000000 ebx=695ff4b4 ecx=761fe5cc edx=00193981 esi=00000000 edi=00000000
eip=76342808 esp=00193ae0 ebp=00193b70 iopl=0         nv up ei pl zr na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000246
KERNELBASE!UnhandledExceptionFilter+0x58:
76342808 cc              int     3
```

② Windbg 붙여 실행중인 프로그램으로 크래시 파일 열기,
프로그램 터지는 시점 분석



```
0:000> kv
 # ChildEBP RetAddr      Args to Child
00 00193b70 69108770     6911f4b4 2b8d3faa d472c055 KERNELBASE!UnhandledExceptionFilter+0x58 (FPO: [Non-Fpo])
WARNING: Stack unwind information not available. Following frames may be wrong.
01 00193ea4 690eceb4     00000002 ff940045 ff94ff94 BabaCAD4Image!ShowPlugInOptions+0x44480
02 00193ef4 690e0000     2b9400aa 00197af0 00197fa0 BabaCAD4Image!ShowPlugInOptions+0x28bc4
03 00193ef8 2b9400aa     00197af0 00197fa0 00000000 BabaCAD4Image!ShowPlugInOptions+0x1bd10
04 00193efc 00197af0     00197fa0 00000000 00000000 0x2b9400aa
05 00193f00 00197fa0     00000000 00000000 00000000 0x197af0
06 00197af0 313d736d     616c623b 67626b63 003b303d 0x197fa0
07 00197af4 616c623b     67626b63 003b303d 00000000 0x313d736d
08 00197af8 67626b63     003b303d 00000000 00000000 0x616c623b
09 00197afc 003b303d     00000000 00000000 00000000 0x67626b63
0a 00197b00 00000000     00000000 00000000 00000000 0x3b303d
```

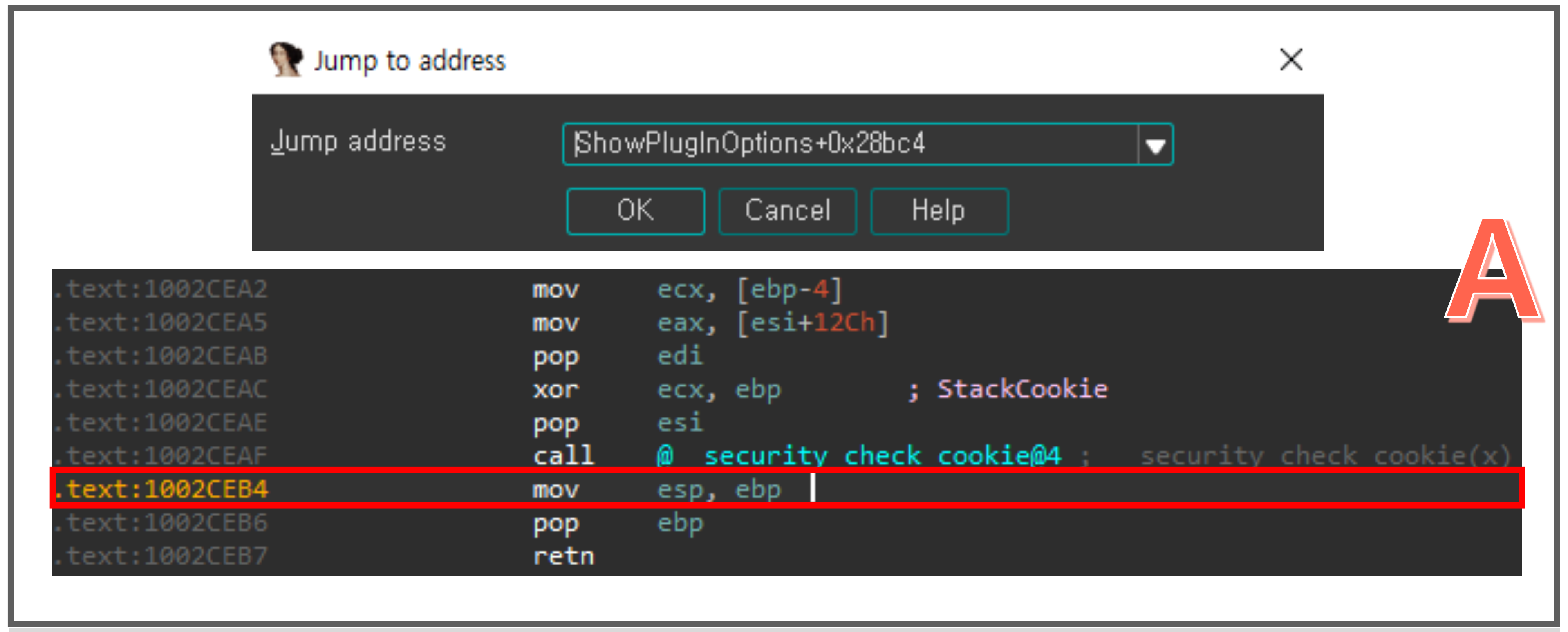
③ 콜스택 분석

Buffer Overflow, 콜스택 망가짐

POC SECURITY

취약점 분석 과정

실전!

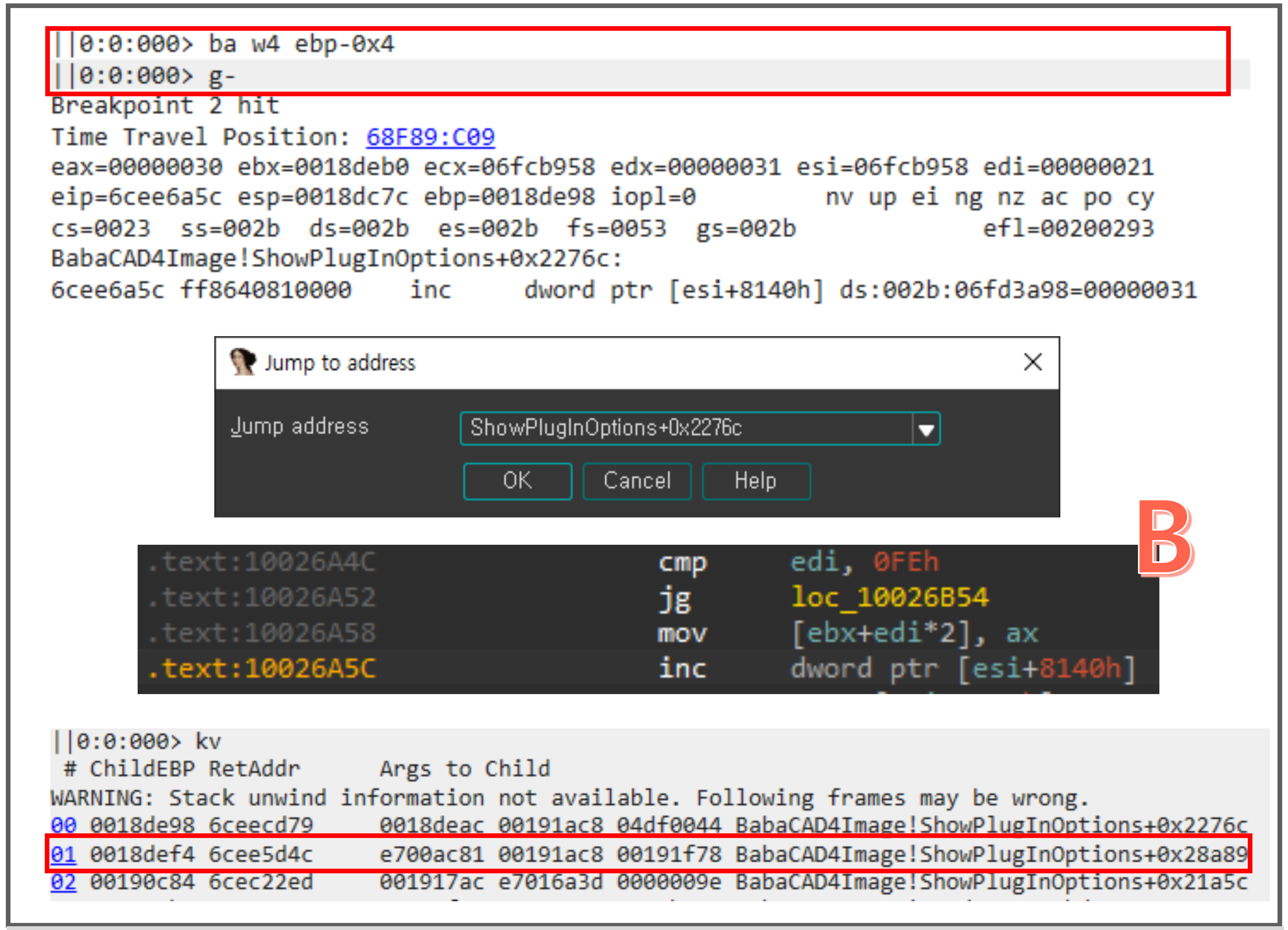


Jump address: ShowPlugInOptions+0x28bc4

```
.text:1002CEA2                 mov     ecx, [ebp-4]
.text:1002CEA5                 mov     eax, [esi+12Ch]
.text:1002CEAB                 pop     edi
.text:1002CEAC                 xor     ecx, ebp        ; StackCookie
.text:1002CEAE                 pop     esi
.text:1002CEAF                 call    @  security check cookie@4 ;   security check cookie(x)
.text:1002CEB4                 mov     esp, ebp
.text:1002CEB6                 pop     ebp
.text:1002CEB7                 retn
```

④ 콜스택 tracing → Canary CHECK 부분 확인

WHEN??



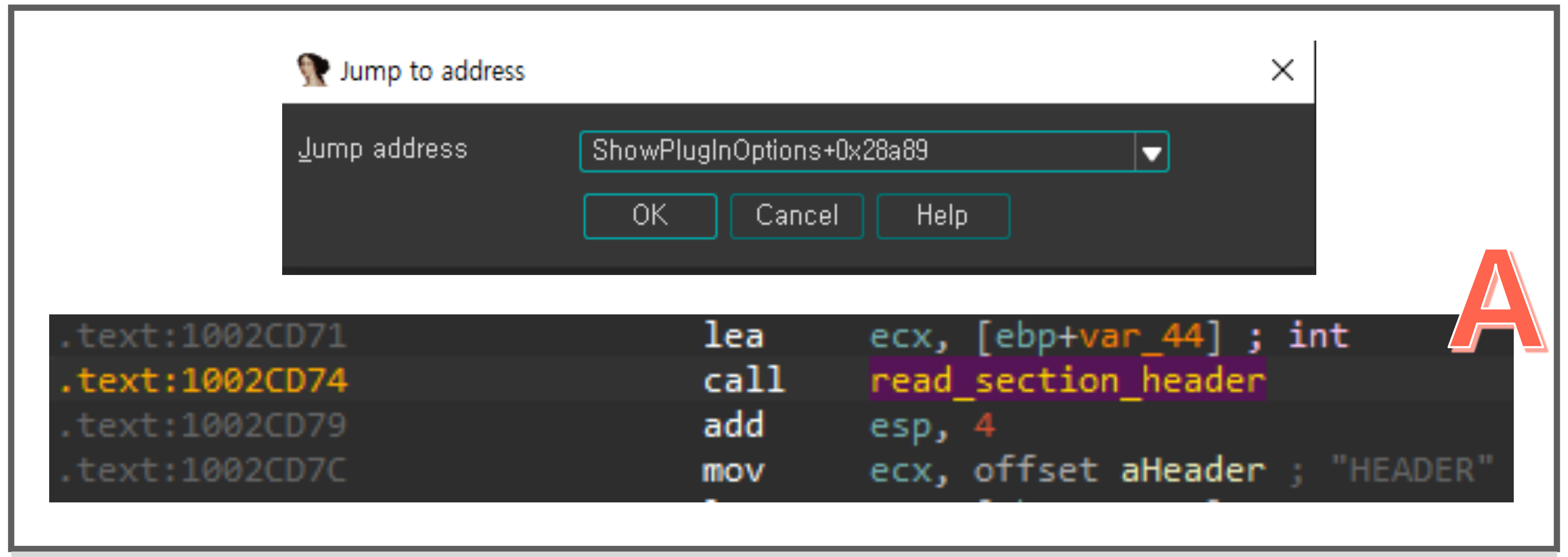
```
||0:0:000> ba w4 ebp-0x4
||0:0:000> g-
Breakpoint 2 hit
Time Travel Position: 68F89:C09
eax=00000030 ebx=0018deb0 ecx=06fcb958 edx=00000031 esi=06fcb958 edi=00000021
eip=6cee6a5c esp=0018dc7c ebp=0018de98 iopl=0         nv up ei ng nz ac po cy
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00200293
BabaCAD4Image!ShowPlugInOptions+0x2276c:
6cee6a5c ff8640810000    inc     dword ptr [esi+8140h] ds:002b:06fd3a98=00000031
```

Jump address: ShowPlugInOptions+0x2276c

```
.text:10026A4C                 cmp     edi, 0FEh
.text:10026A52                 jg      loc_10026B54
.text:10026A58                 mov     [ebx+edi*2], ax
.text:10026A5C                 inc     dword ptr [esi+8140h]
```

```
||0:0:000> kv
 # ChildEBP RetAddr      Args to Child
WARNING: Stack unwind information not available. Following frames may be wrong.
00 0018de98 6ceecd79     0018deac 00191ac8 04df0044 BabaCAD4Image!ShowPlugInOptions+0x2276c
01 0018def4 6cee5d4c     e700ac81 00191ac8 00191f78 BabaCAD4Image!ShowPlugInOptions+0x28a89
02 00190c84 6cec22ed     001917ac e7016a3d 0000009e BabaCAD4Image!ShowPlugInOptions+0x21a5c
```

⑤ Canary 덮이는 부분 확인, 그 지점에서의 콜스택 확인



Jump address: ShowPlugInOptions+0x28a89

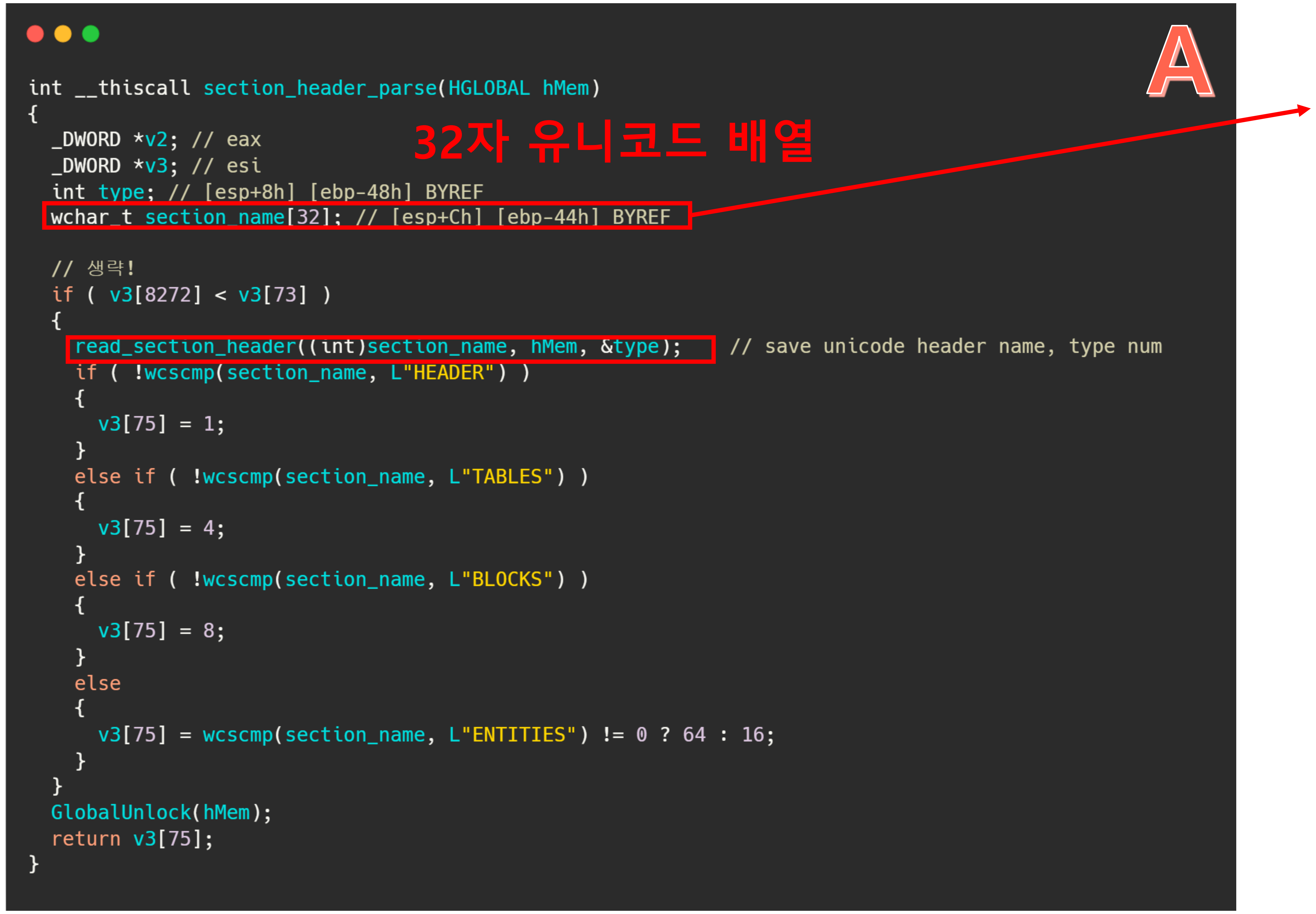
```
.text:1002CD71                 lea     ecx, [ebp+var_44] ; int
.text:1002CD74                 call    read_section_header
.text:1002CD79                 add     esp, 4
.text:1002CD7C                 mov     ecx, offset aHeader ; "HEADER"
```

⑥ 콜스택 tracing → 함수 간 호출 순서 파악

A → B → A

취약점 분석 과정

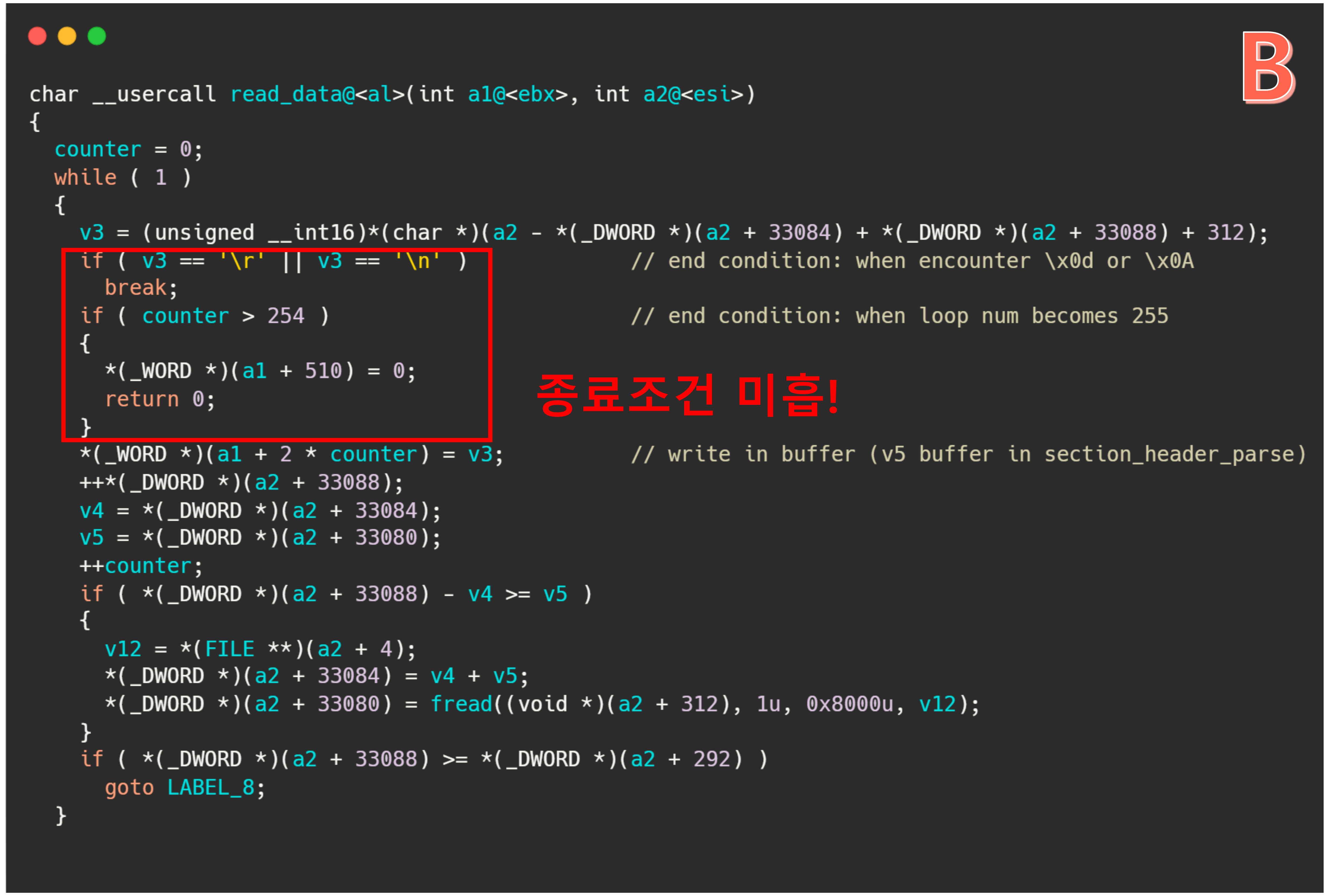
실전!



```
int __thiscall section_header_parse(HGLOBAL hMem)
{
  _DWORD *v2; // eax
  _DWORD *v3; // esi
  int type; // [esp+8h] [ebp-48h] BYREF
  wchar_t section_name[32]; // [esp+Ch] [ebp-44h] BYREF

  // 생략!
  if ( v3[8272] < v3[73] )
  {
    read_section_header((int)section_name, hMem, &type);  // save unicode header name, type num
    if ( !wcscmp(section_name, L"HEADER") )
    {
      v3[75] = 1;
    }
    else if ( !wcscmp(section_name, L"TABLES") )
    {
      v3[75] = 4;
    }
    else if ( !wcscmp(section_name, L"BLOCKS") )
    {
      v3[75] = 8;
    }
    else
    {
      v3[75] = wcscmp(section_name, L"ENTITIES") != 0 ? 64 : 16;
    }
  }
  GlobalUnlock(hMem);
  return v3[75];
}
```

A

32자 유니코드 배열



```
char __usercall read_data@<al>(int a1@<ebx>, int a2@<esi>)
{
  counter = 0;
  while ( 1 )
  {
    v3 = (unsigned __int16)*(char *)(a2 - *(_DWORD *)(a2 + 33084) + *(_DWORD *)(a2 + 33088) + 312);
    if ( v3 == '\r' || v3 == '\n' )                  // end condition: when encounter \x0d or \x0A
      break;
    if ( counter > 254 )                             // end condition: when loop num becomes 255
    {
      *(_WORD *)(a1 + 510) = 0;
      return 0;
    }
    *(_WORD *)(a1 + 2 * counter) = v3;               // write in buffer (v5 buffer in section_header_parse)
    ++*(_DWORD *)(a2 + 33088);
    v4 = *(_DWORD *)(a2 + 33084);
    v5 = *(_DWORD *)(a2 + 33080);
    ++counter;
    if ( *(_DWORD *)(a2 + 33088) - v4 >= v5 )
    {
      v12 = *(FILE **)(a2 + 4);
      *(_DWORD *)(a2 + 33084) = v4 + v5;
      *(_DWORD *)(a2 + 33080) = fread((void *)(a2 + 312), 1u, 0x8000u, v12);
    }
    if ( *(_DWORD *)(a2 + 33088) >= *(_DWORD *)(a2 + 292) )
      goto LABEL_8;
  }
```

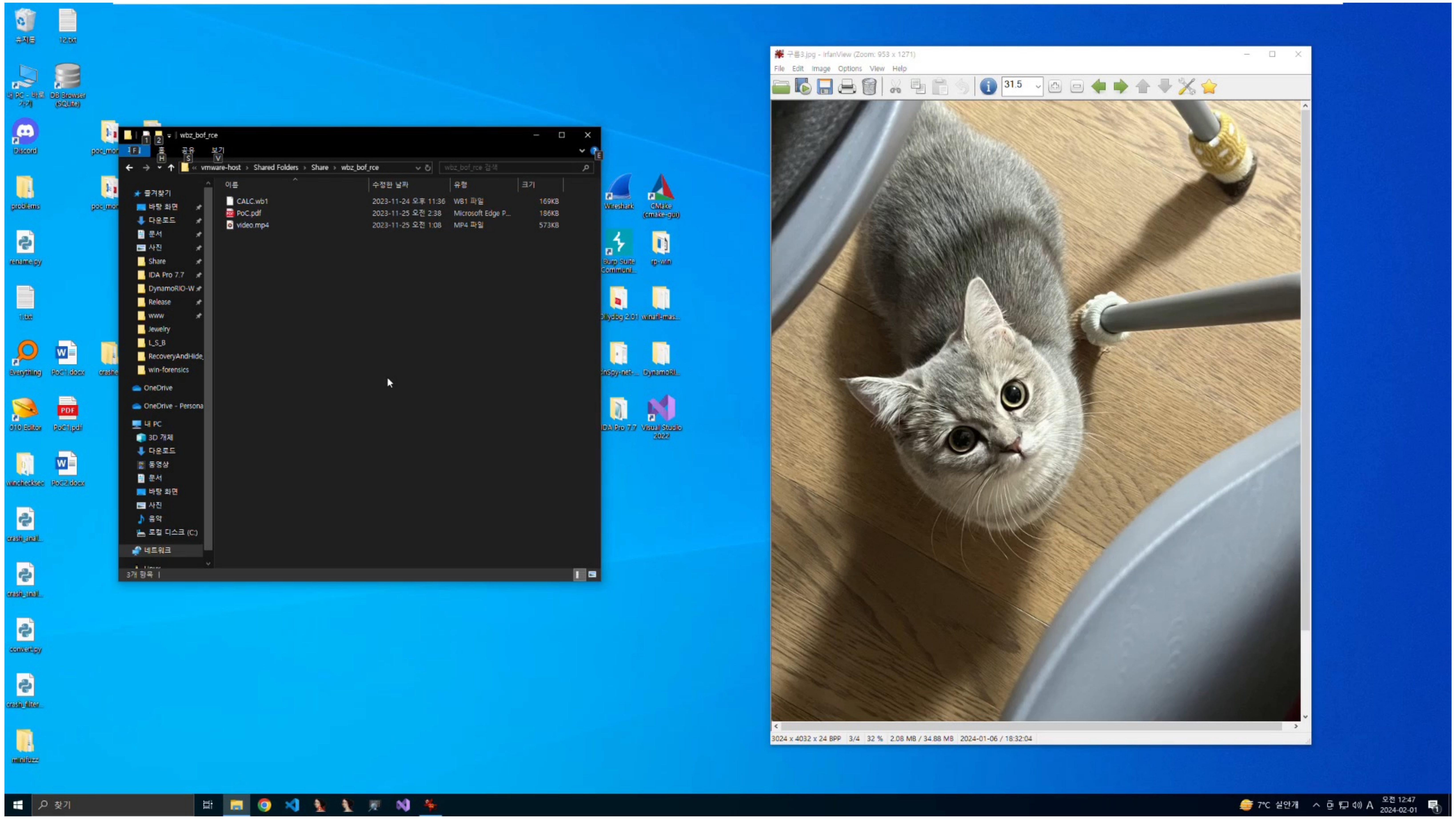
B

종료조건 미흡!

⑦ IDA 이용해 Root Cause 분석

취약점 분석 과정

익스플로잇 데모 영상

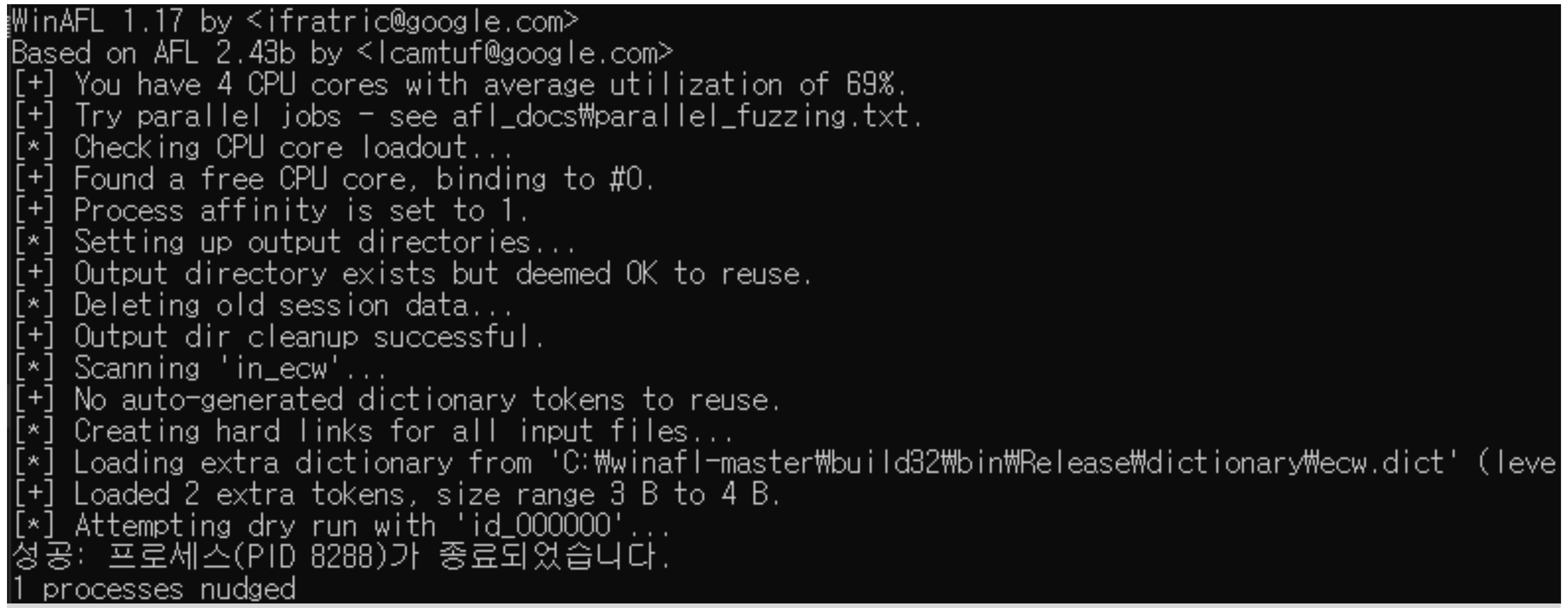


vmware-host › Shared Folders › Share › wbz_bof_rce

이름	수정한 날짜	유형	크기
CALC.wb1	2023-11-24 오후 11:36	WB1 파일	169KB
PoC.pdf	2023-11-25 오전 2:38	Microsoft Edge P...	186KB
video.mp4	2023-11-25 오전 1:08	MP4 파일	573KB

WinAFL의 한계

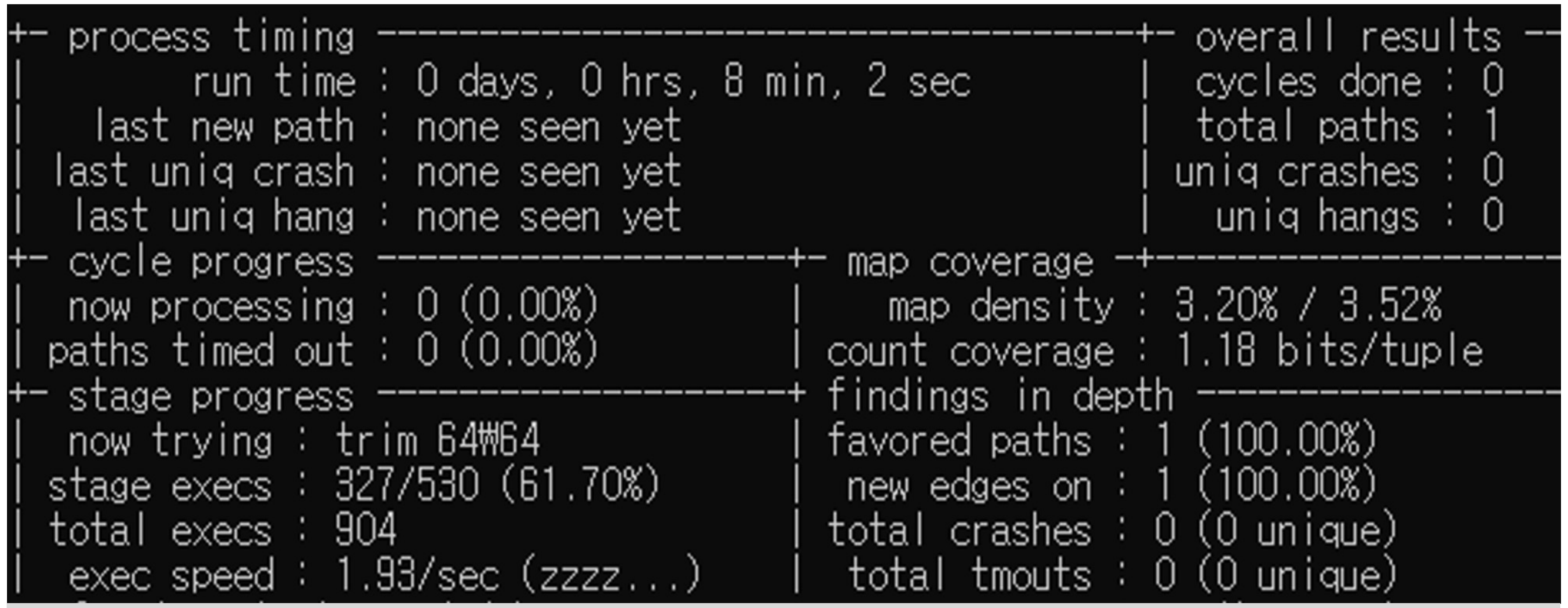
WinAFL 퍼징은 무적이 아닙니다



```
WinAFL 1.17 by <ifratric@google.com>
Based on AFL 2.43b by <lcamtuf@google.com>
[+] You have 4 CPU cores with average utilization of 69%.
[+] Try parallel jobs - see afl_docs₩parallel_fuzzing.txt.
[*] Checking CPU core loadout...
[+] Found a free CPU core, binding to #0.
[+] Process affinity is set to 1.
[*] Setting up output directories...
[+] Output directory exists but deemed OK to reuse.
[*] Deleting old session data...
[+] Output dir cleanup successful.
[*] Scanning 'in_ecw'...
[+] No auto-generated dictionary tokens to reuse.
[*] Creating hard links for all input files...
[*] Loading extra dictionary from 'C:₩winafl-master₩build32₩bin₩Release₩dictionary₩ecw.dict' (leve
[+] Loaded 2 extra tokens, size range 3 B to 4 B.
[*] Attempting dry run with 'id_000000'...
성공: 프로세스(PID 8288)가 종료되었습니다.
1 processes nudged
```

상세 조건 맞춰주지 않으면

Dry run 뜨면서 퍼저 강제 종료



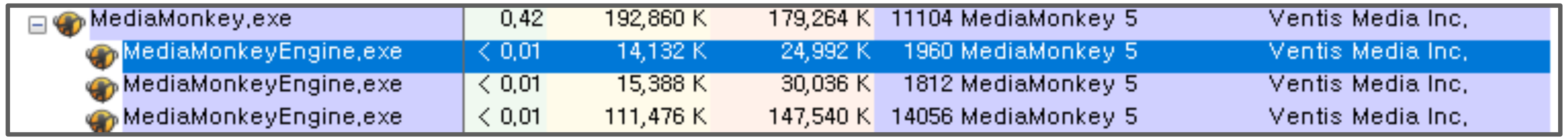
```
+- process timing ------------------------------------+- overall results --
|        run time : 0 days, 0 hrs, 8 min, 2 sec      |  cycles done : 0
|   last new path : none seen yet                    |  total paths : 1
| last uniq crash : none seen yet                    | uniq crashes : 0
|  last uniq hang : none seen yet                    |   uniq hangs : 0
+- cycle progress --------------------+- map coverage -+--------------------
|  now processing : 0 (0.00%)          |    map density : 3.20% / 3.52%
| paths timed out : 0 (0.00%)          | count coverage : 1.18 bits/tuple
+- stage progress --------------------+ findings in depth ------------------
|  now trying : trim 64₩64             | favored paths : 1 (100.00%)
| stage execs : 327/530 (61.70%)       |  new edges on : 1 (100.00%)
| total execs : 904                    | total crashes : 0 (0 unique)
|  exec speed : 1.93/sec (zzzz...)     |  total tmouts : 0 (0 unique)
```

GUI 렌더링, 복잡한 초기화 작업 등의

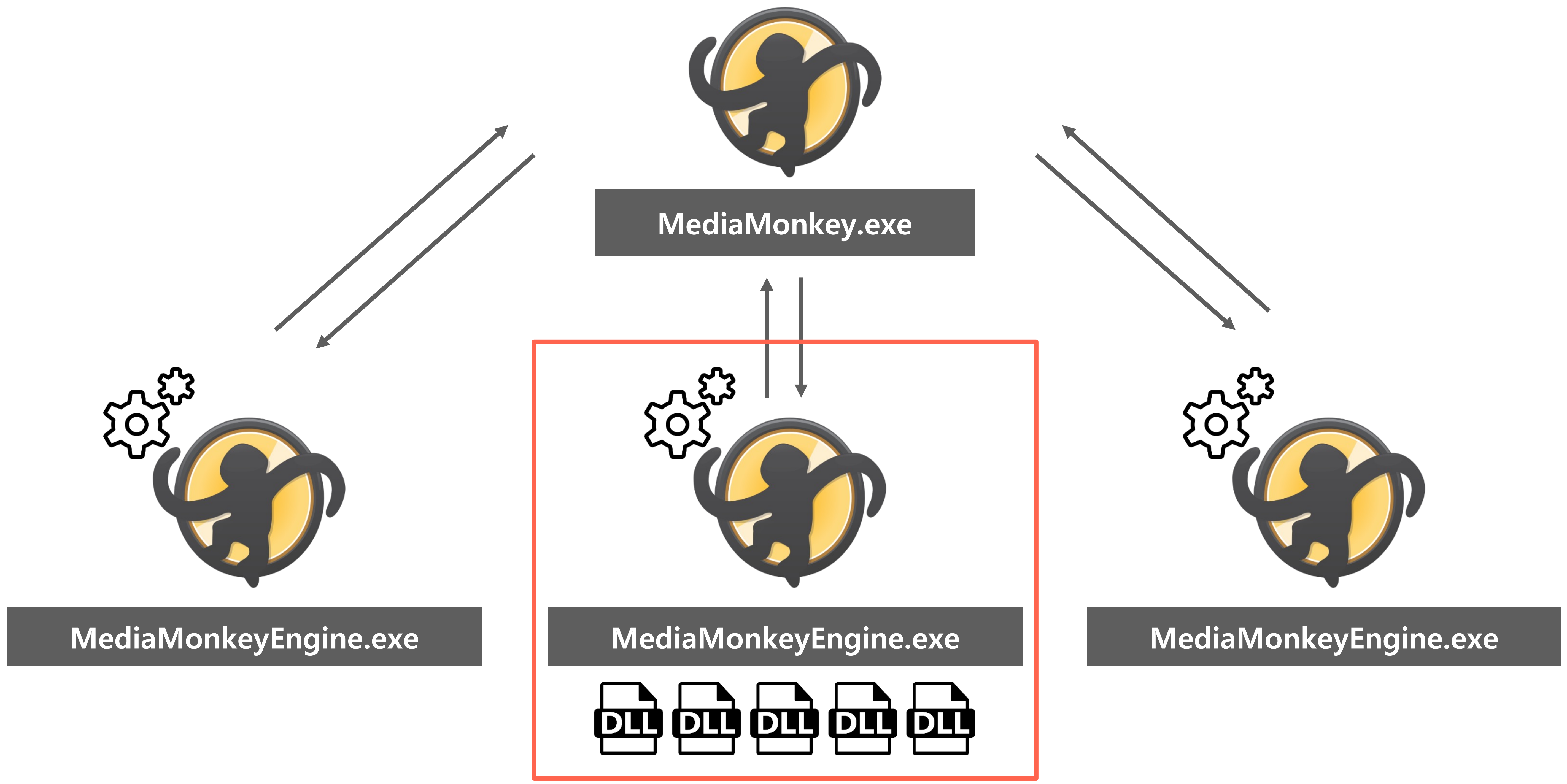
요인으로 인한 심각한 속도 저하

WinAFL의 한계

WinAFL 퍼징은 무적이 아닙니다



```
⊟ MediaMonkey.exe              0.42   192,860 K  179,264 K  11104 MediaMonkey 5    Ventis Media Inc.
    MediaMonkeyEngine.exe    < 0.01    14,132 K   24,992 K   1960 MediaMonkey 5    Ventis Media Inc.
    MediaMonkeyEngine.exe    < 0.01    15,388 K   30,036 K   1812 MediaMonkey 5    Ventis Media Inc.
    MediaMonkeyEngine.exe    < 0.01   111,476 K  147,540 K  14056 MediaMonkey 5    Ventis Media Inc.
```



1. 상위 프로세스 퍼징이 효과적이지 않다면?

2. 단독으로 실행할 수 없다면?

3. 패치해도 프로세스 간 종속성 때문에 퍼징에 실패한다면?

스냅샷(인메모리) 퍼징

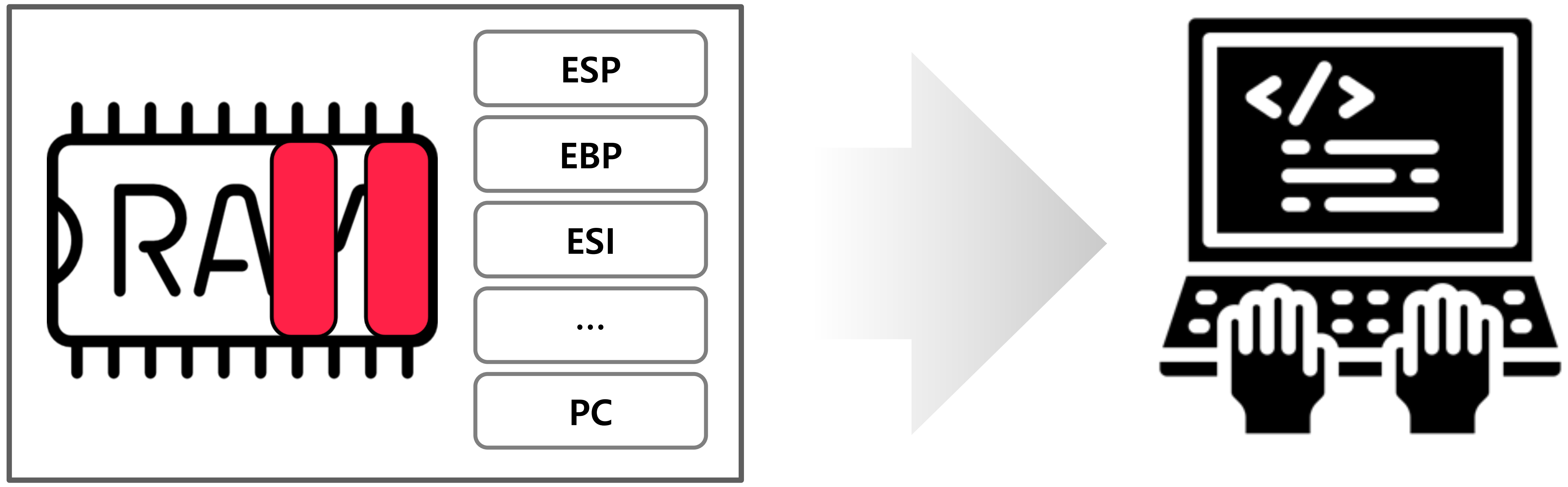
WTF Fuzzer

PrtScr
SysRq

스냅샷(Snapshot)?

순간을 재빠르게 포착한 사진.

특정 시점에 데이터 저장 장치의 상태를 별도의 파일로 저장하는 것.

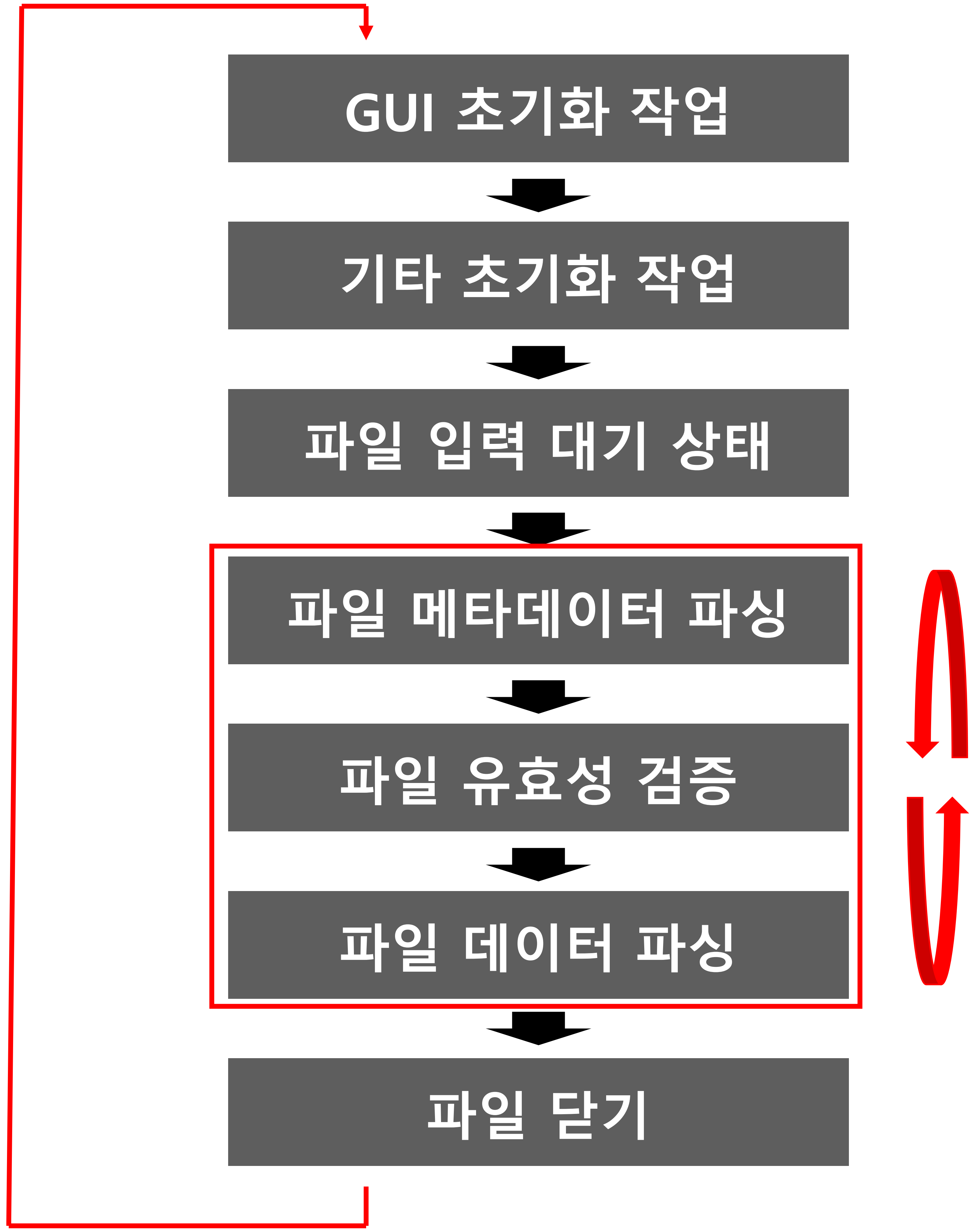


- 프로그램을 특정 시점까지 실행시킨 후 메모리 상태를 스냅샷 찍음
- 변형된 입력을 반복적으로 주입하여 취약점 탐지

POC SECURITY

스냅샷(인메모리) 퍼징

WTF Fuzzer

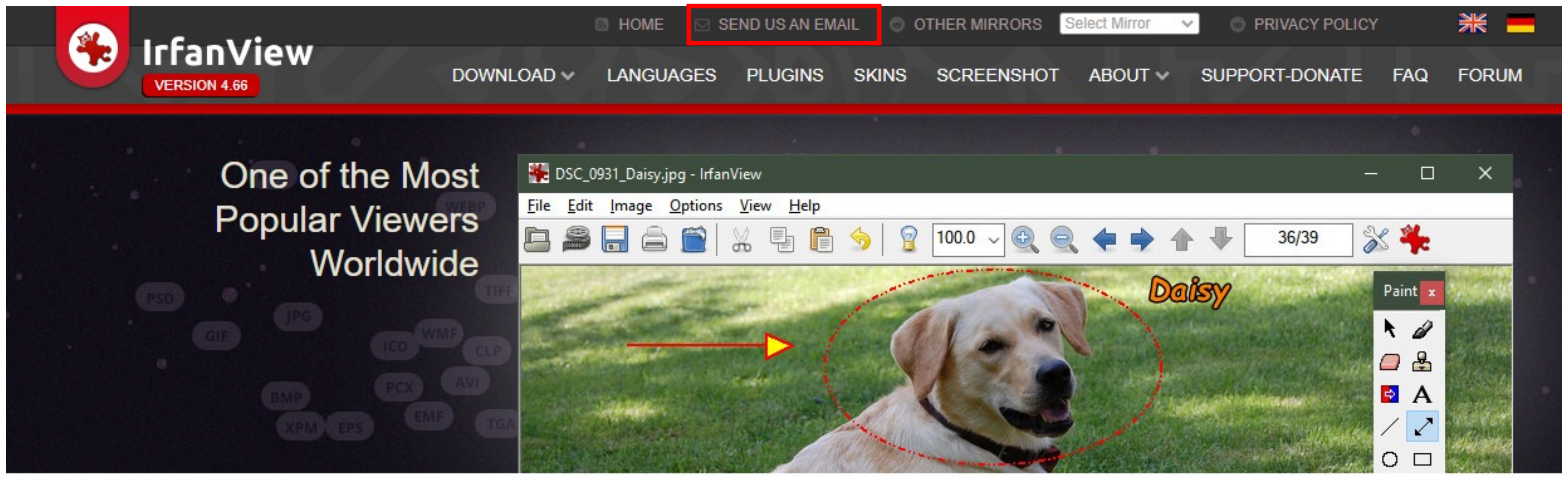


- 프로세스 생성하고 입력값 받기까지의 상당한 오버헤드
- **속도 향상**: 복잡한 초기화 과정 건너뛰고 분석 지점부터 테스트 케이스 주입 가능
- **분석의 집중화**: 관심 있는 특정 동작 코드에 집중

취약점 제보 절차

취약점 제보 절차

1. 벤더사 직접 제보



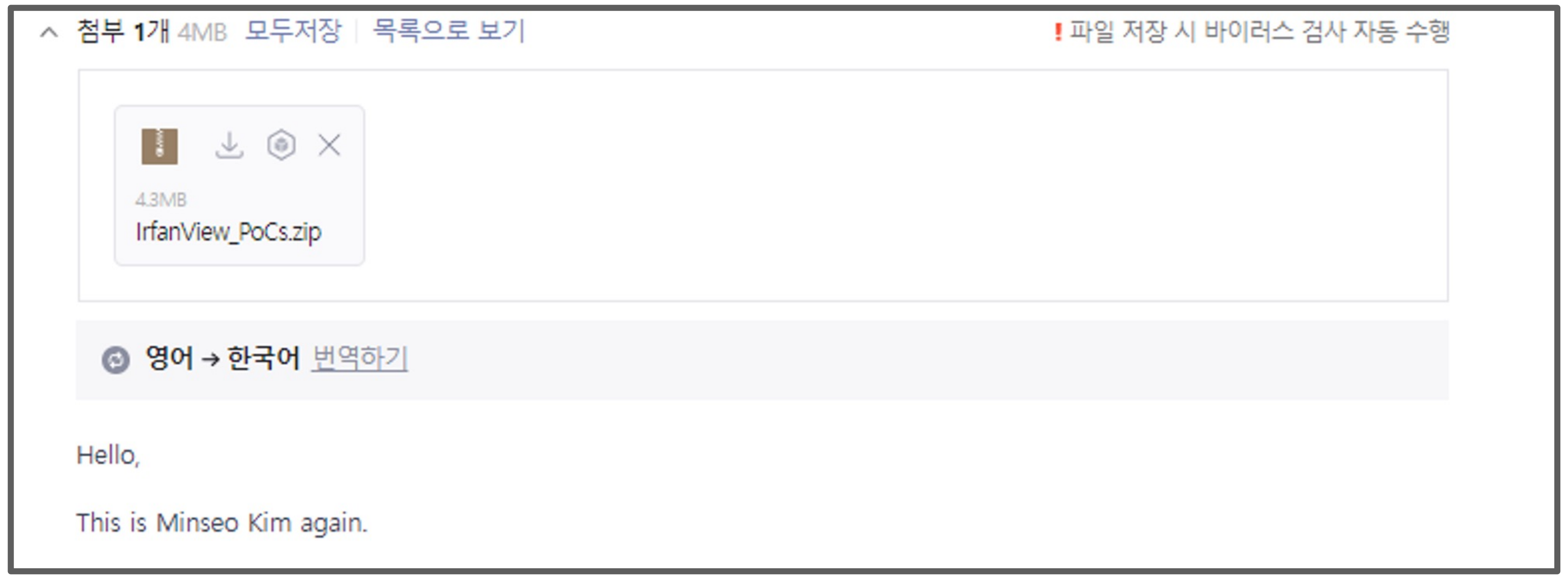
① 벤더사 이메일 주소 확인

Stack-based Buffer Overflow Vulnerability in the Latest Version of 32bit IrfanView Plugin

② PoC 파일 및 분석 보고서 준비

취약점 제보 절차

1. 벤더사 직접 제보



첨부 1개 4MB 모두저장 | 목록으로 보기 ! 파일 저장 시 바이러스 검사 자동 수행

4.3MB
IrfanView_PoCs.zip

영어 → 한국어 번역하기

Hello,

This is Minseo Kim again.

③ 개발자에게 이메일로

PoC 파일 및 보고서 전달

④ 패치 완료 후

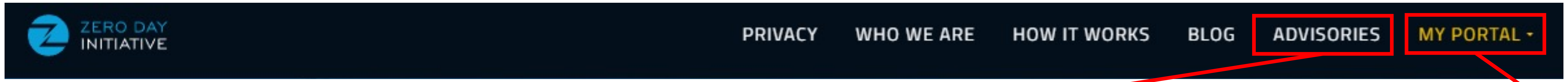
개발자의 CVE 신청

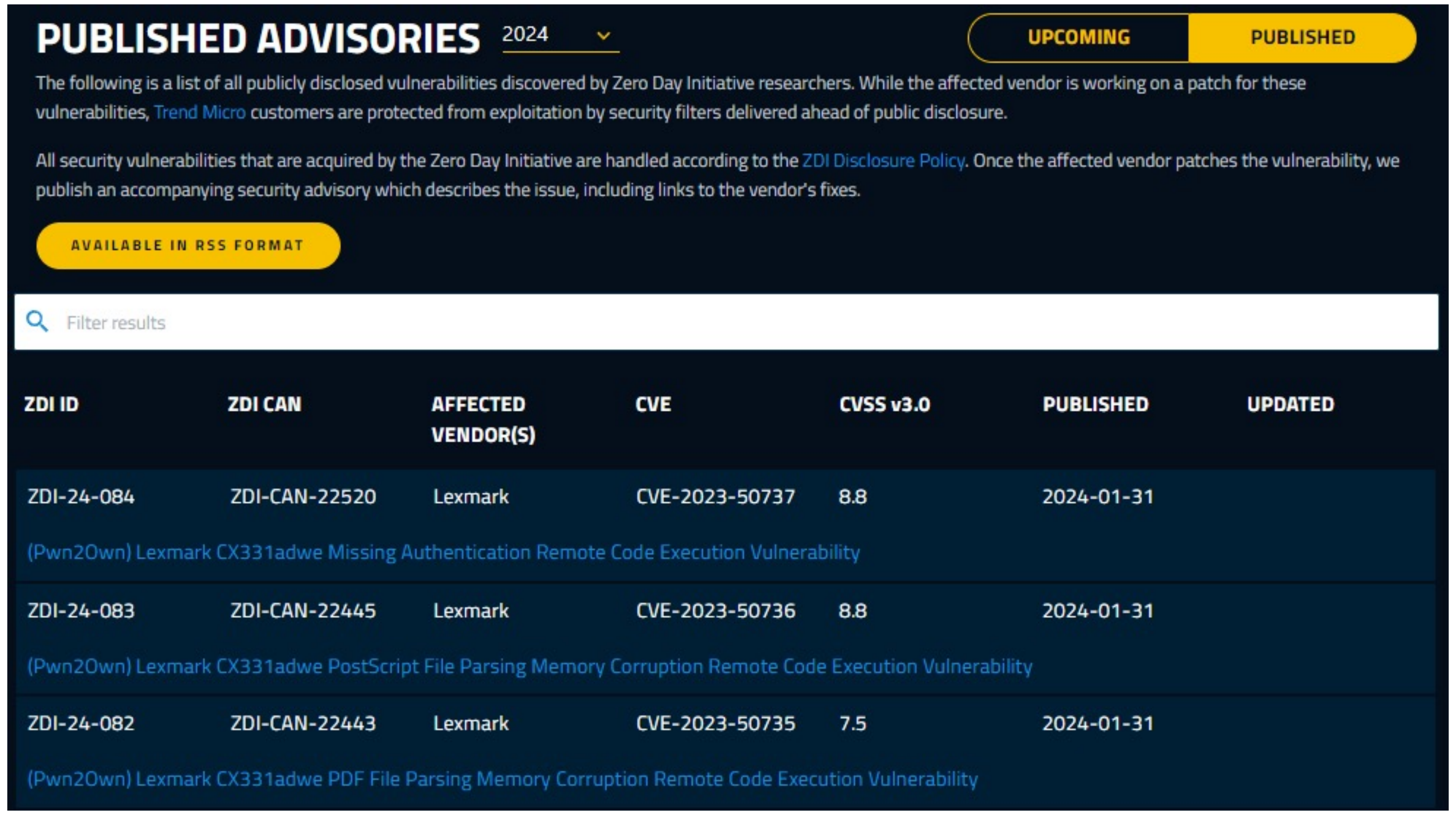
④ 패치 거부

CVE 신청하지 않음

취약점 제보 절차

2. Zero Day Initiative(ZDI) 제보



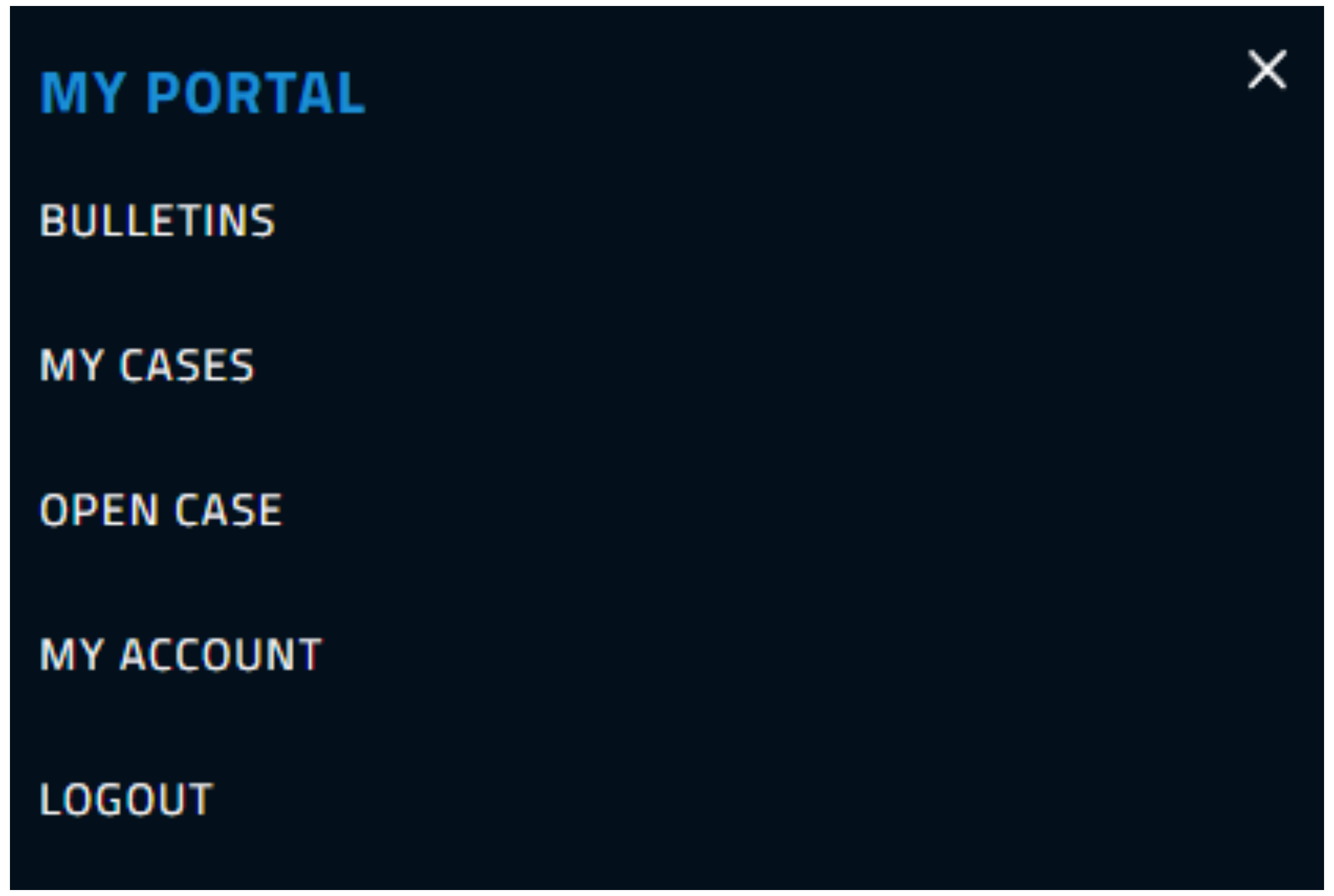


PUBLISHED ADVISORIES

The following is a list of all publicly disclosed vulnerabilities discovered by Zero Day Initiative researchers. While the affected vendor is working on a patch for these vulnerabilities, Trend Micro customers are protected from exploitation by security filters delivered ahead of public disclosure.

All security vulnerabilities that are acquired by the Zero Day Initiative are handled according to the ZDI Disclosure Policy. Once the affected vendor patches the vulnerability, we publish an accompanying security advisory which describes the issue, including links to the vendor's fixes.

ZDI ID	ZDI CAN	AFFECTED VENDOR(S)	CVE	CVSS v3.0	PUBLISHED	UPDATED
ZDI-24-084	ZDI-CAN-22520	Lexmark	CVE-2023-50737	8.8	2024-01-31	
(Pwn2Own) Lexmark CX331adwe Missing Authentication Remote Code Execution Vulnerability						
ZDI-24-083	ZDI-CAN-22445	Lexmark	CVE-2023-50736	8.8	2024-01-31	
(Pwn2Own) Lexmark CX331adwe PostScript File Parsing Memory Corruption Remote Code Execution Vulnerability						
ZDI-24-082	ZDI-CAN-22443	Lexmark	CVE-2023-50735	7.5	2024-01-31	
(Pwn2Own) Lexmark CX331adwe PDF File Parsing Memory Corruption Remote Code Execution Vulnerability						

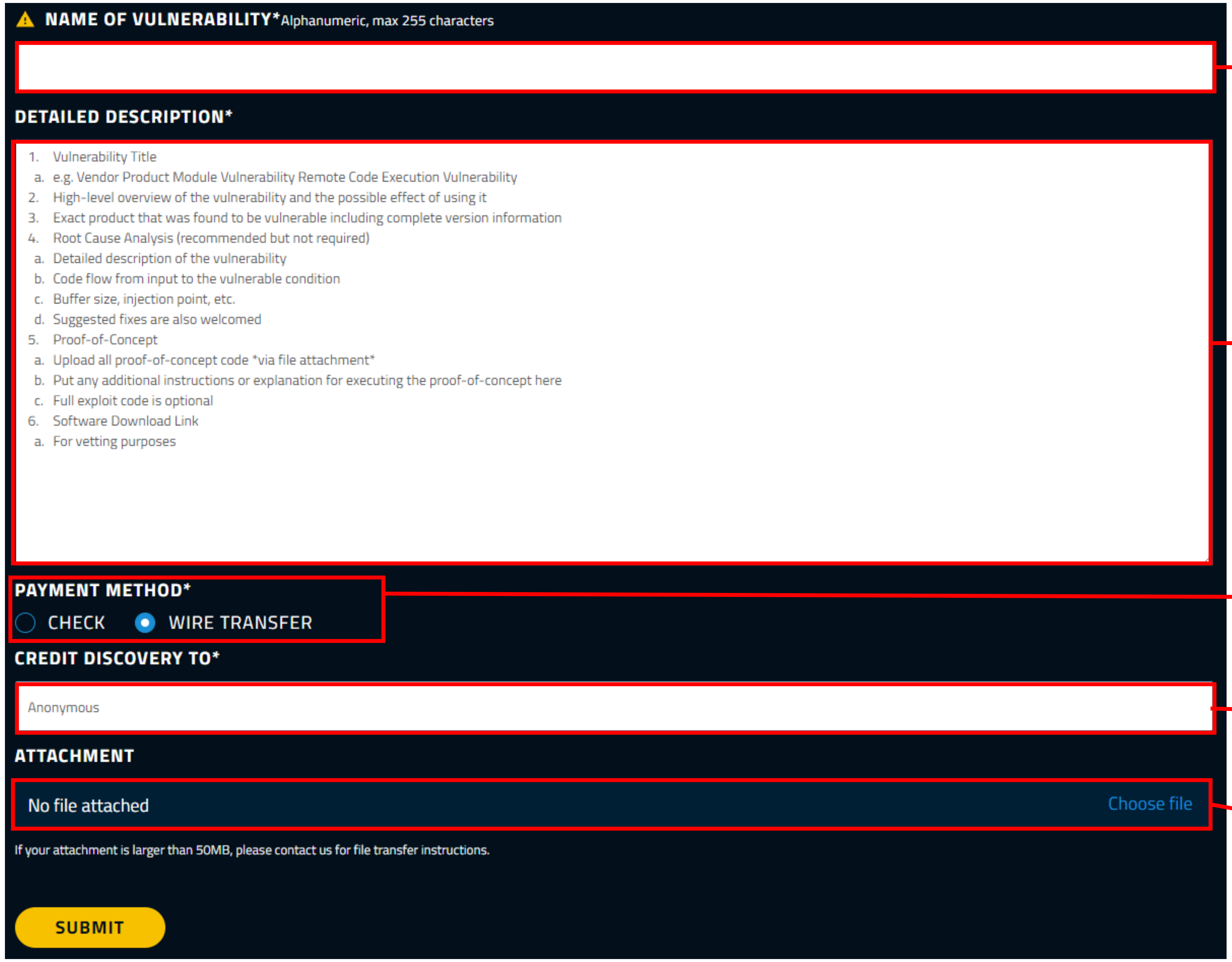


ZDI 발행 목록: UPCOMING vs. PUBLISHED

계정 관리: 진행 상황 확인, 제보, 개인정보 설정 등

취약점 제보 절차

2. Zero Day Initiative(ZDI) 제보



⚠ NAME OF VULNERABILITY*Alphanumeric, max 255 characters

DETAILED DESCRIPTION*

1. Vulnerability Title
   a. e.g. Vendor Product Module Vulnerability Remote Code Execution Vulnerability
2. High-level overview of the vulnerability and the possible effect of using it
3. Exact product that was found to be vulnerable including complete version information
4. Root Cause Analysis (recommended but not required)
   a. Detailed description of the vulnerability
   b. Code flow from input to the vulnerable condition
   c. Buffer size, injection point, etc.
   d. Suggested fixes are also welcomed
5. Proof-of-Concept
   a. Upload all proof-of-concept code *via file attachment*
   b. Put any additional instructions or explanation for executing the proof-of-concept here
   c. Full exploit code is optional
6. Software Download Link
   a. For vetting purposes

PAYMENT METHOD*

○ CHECK ● WIRE TRANSFER

CREDIT DISCOVERY TO*

Anonymous

ATTACHMENT

No file attached Choose file

If your attachment is larger than 50MB, please contact us for file transfer instructions.

SUBMIT

제목: PUBLISHED ADVISORIES 참고하여 작성

본문: 취약점 제목, 취약점에 대한 요약 및 잠재적인 영향, 취약점이 발생한 제품의 상세 버전, 취약점 원인 분석, 패치 방향, PoC 등을 영어로 설명

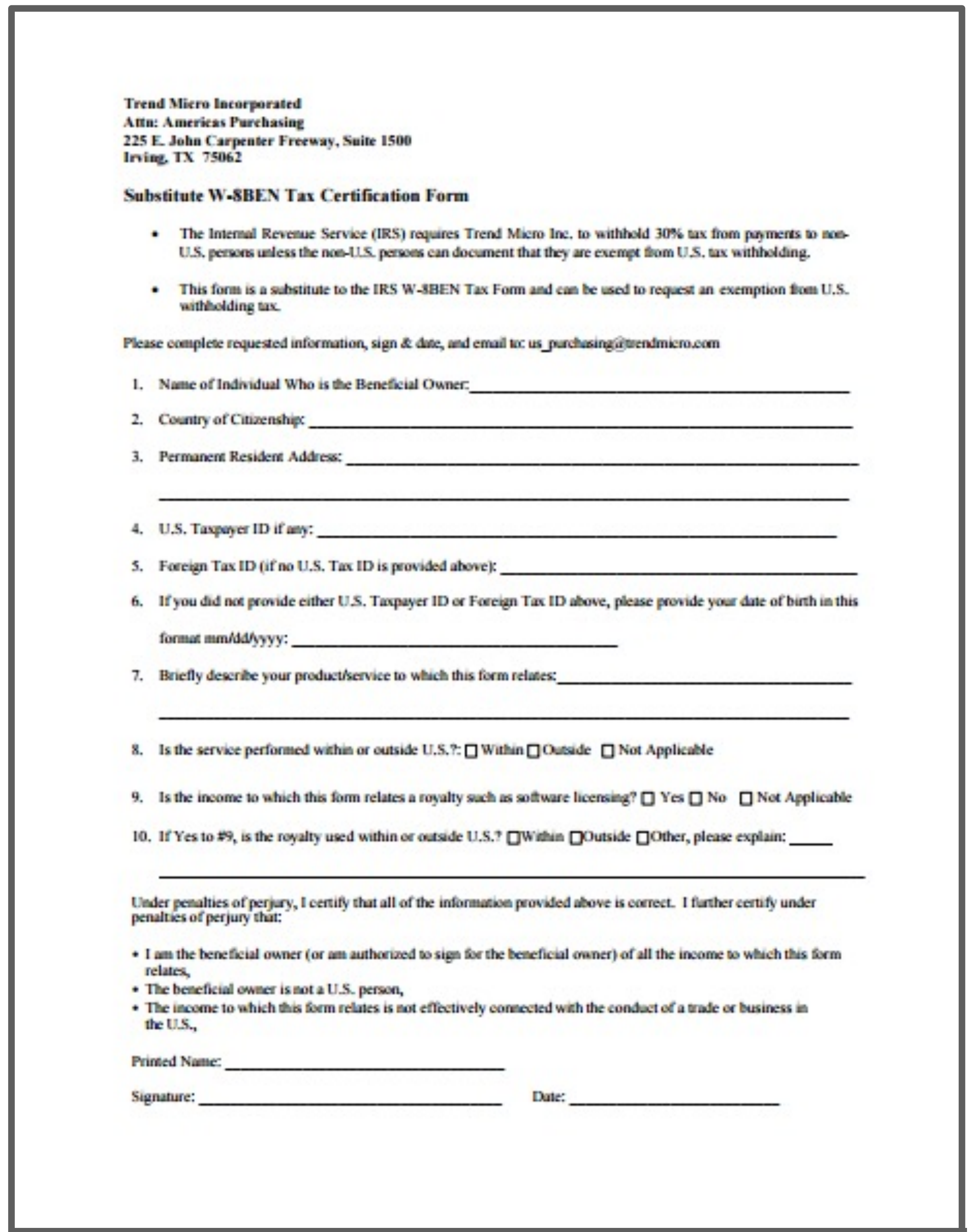
지불 방법: 바운티 수령 방법 (WIRE TRANSFER 추천)

기여자: ZDI 및 CVE 발급 시 들어갈 이름

파일 첨부: 분석 보고서, PoC 파일, 영상 자료 등

취약점 제보 절차

2. Zero Day Initiative(ZDI) 제보



Trend Micro Incorporated
Attn: Americas Purchasing
225 E. John Carpenter Freeway, Suite 1500
Irving, TX 75062

Substitute W-8BEN Tax Certification Form

- The Internal Revenue Service (IRS) requires Trend Micro Inc. to withhold 30% tax from payments to non-U.S. persons unless the non-U.S. persons can document that they are exempt from U.S. tax withholding.
- This form is a substitute to the IRS W-8BEN Tax Form and can be used to request an exemption from U.S. withholding tax.

Please complete requested information, sign & date, and email to: us_purchasing@trendmicro.com

1. Name of Individual Who is the Beneficial Owner:________________
2. Country of Citizenship: ________________
3. Permanent Resident Address: ________________
____
4. U.S. Taxpayer ID if any: ________________
5. Foreign Tax ID (if no U.S. Tax ID is provided above): ________________
6. If you did not provide either U.S. Taxpayer ID or Foreign Tax ID above, please provide your date of birth in this format mm/dd/yyyy: ________________
7. Briefly describe your product/service to which this form relates:________________
____
8. Is the service performed within or outside U.S.?: □ Within □ Outside □ Not Applicable
9. Is the income to which this form relates a royalty such as software licensing? □ Yes □ No □ Not Applicable
10. If Yes to #9, is the royalty used within or outside U.S.? □Within □Outside □Other, please explain: _____
____

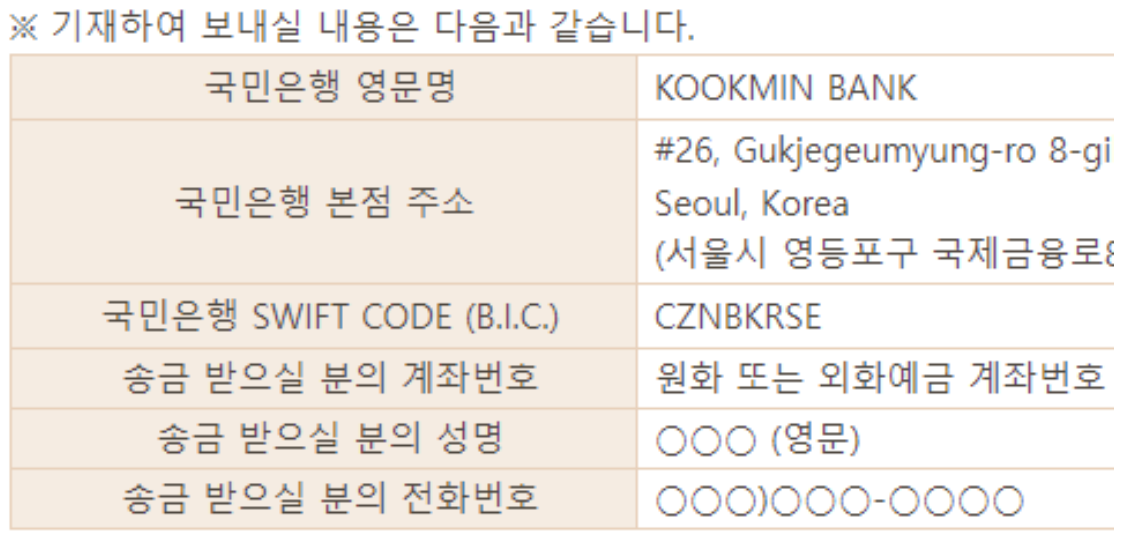
Under penalties of perjury, I certify that all of the information provided above is correct. I further certify under penalties of perjury that:

- I am the beneficial owner (or am authorized to sign for the beneficial owner) of all the income to which this form relates,
- The beneficial owner is not a U.S. person,
- The income to which this form relates is not effectively connected with the conduct of a trade or business in the U.S.,

Printed Name: ________________

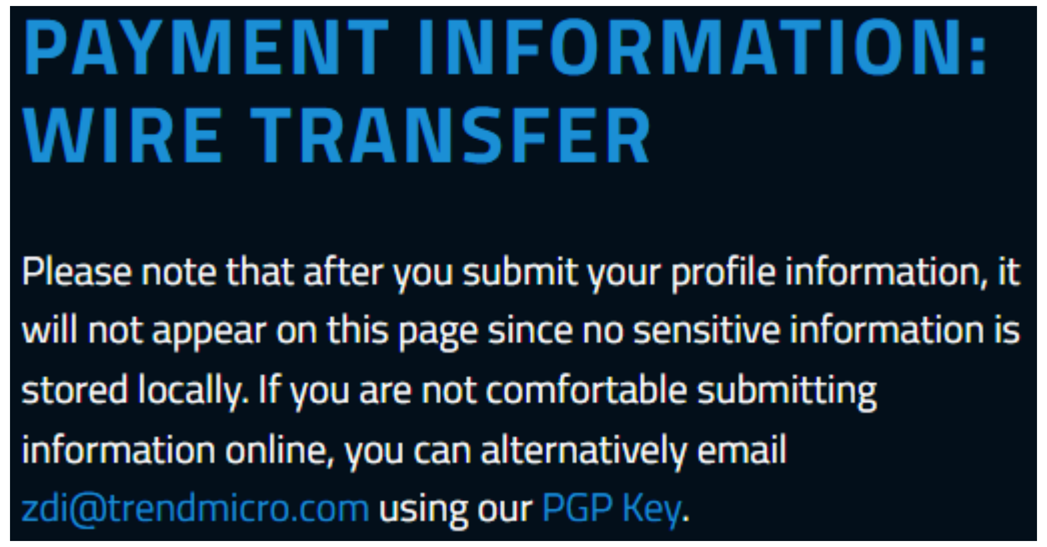
Signature: ________________ Date: ________________

W8-BEN 문서



※ 기재하여 보내실 내용은 다음과 같습니다.

국민은행 영문명	KOOKMIN BANK
국민은행 본점 주소	#26, Gukjegeumyung-ro 8-gi Seoul, Korea (서울시 영등포구 국제금융로
국민은행 SWIFT CODE (B.I.C.)	CZNBKRSE
송금 받으실 분의 계좌번호	원화 또는 외화예금 계좌번호
송금 받으실 분의 성명	○○○ (영문)
송금 받으실 분의 전화번호	○○○)○○○-○○○○



PAYMENT INFORMATION: WIRE TRANSFER

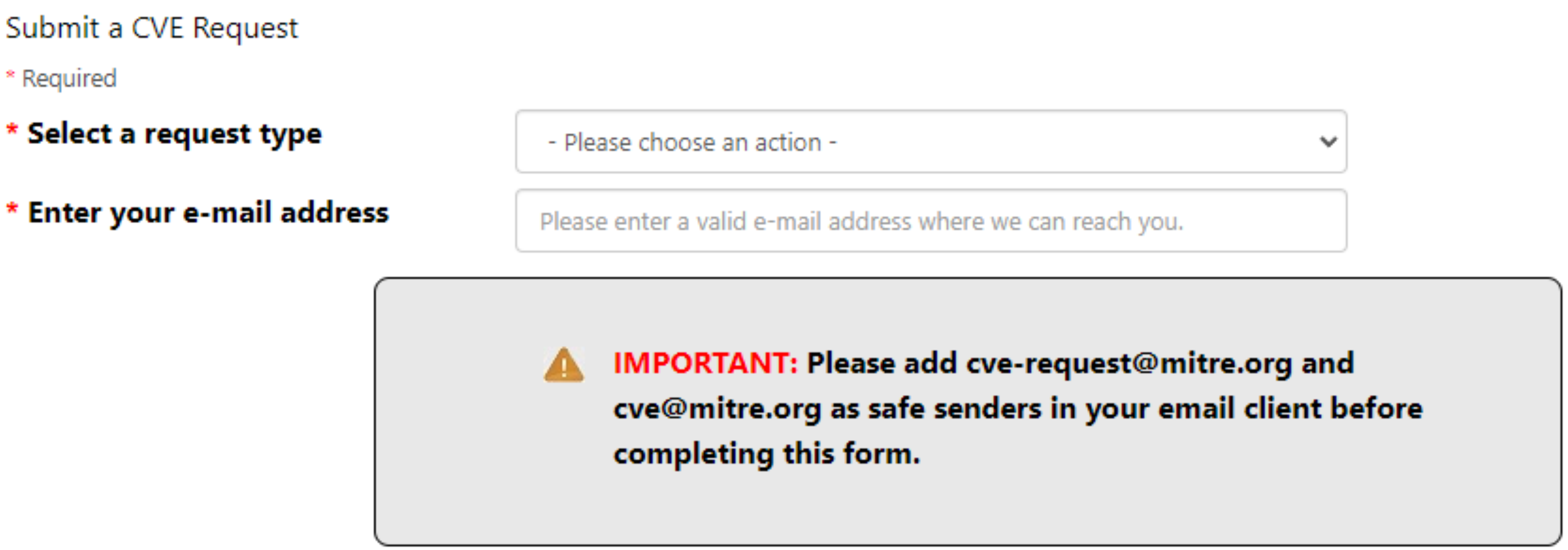
Please note that after you submit your profile information, it will not appear on this page since no sensitive information is stored locally. If you are not comfortable submitting information online, you can alternatively email zdi@trendmicro.com using our PGP Key.

Wire Transfer 정보

신분증/여권 사본

취약점 제보 절차

3. CNE 기관에 직접 제보



Submit a CVE Request

* Required

* **Select a request type** - Please choose an action -

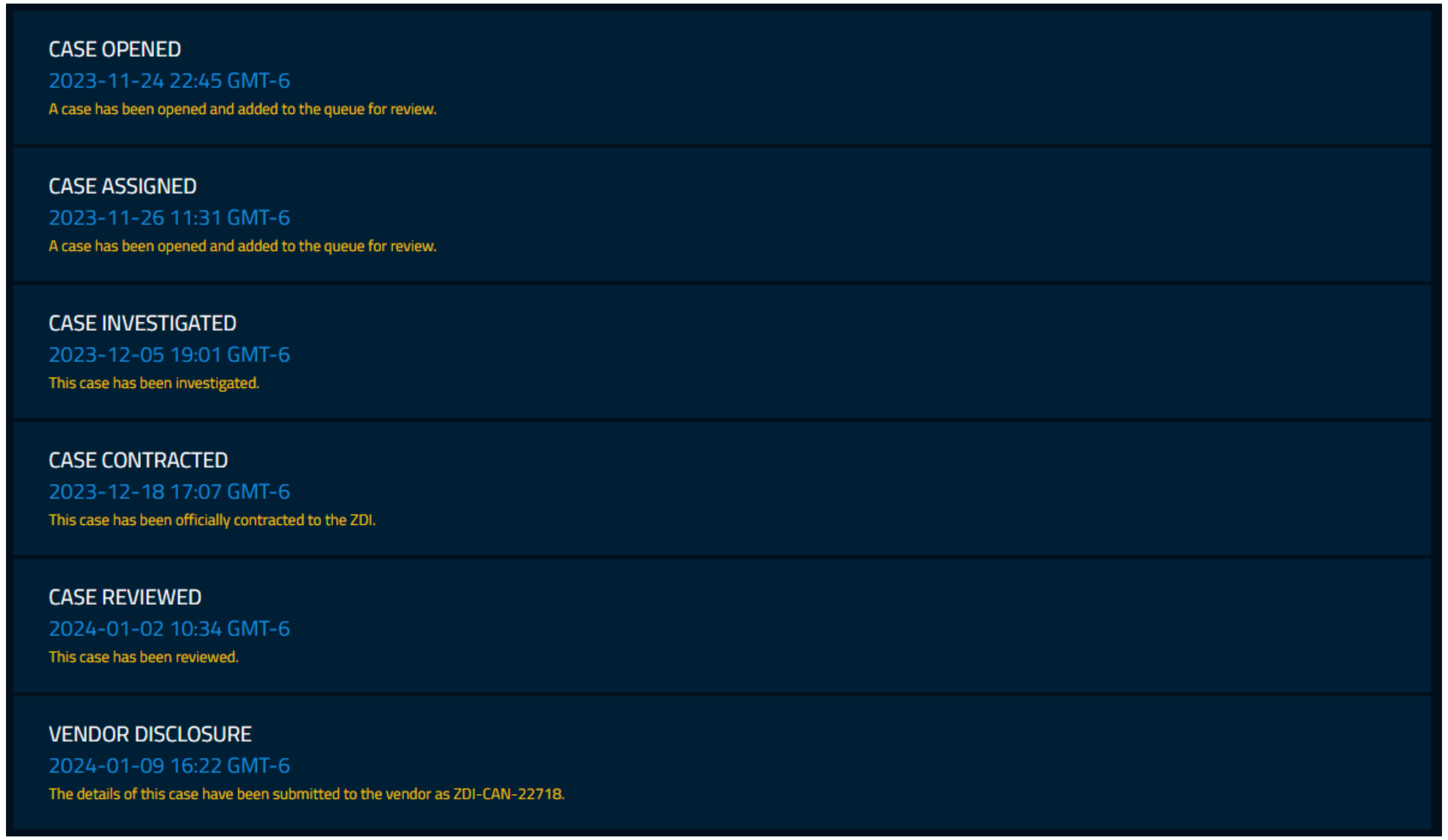
* **Enter your e-mail address** Please enter a valid e-mail address where we can reach you.

⚠ **IMPORTANT: Please add cve-request@mitre.org and cve@mitre.org as safe senders in your email client before completing this form.**

https://cve.mitre.org/

취약점 제보 결과

프로젝트 결과



CASE OPENED
2023-11-24 22:45 GMT-6
A case has been opened and added to the queue for review.

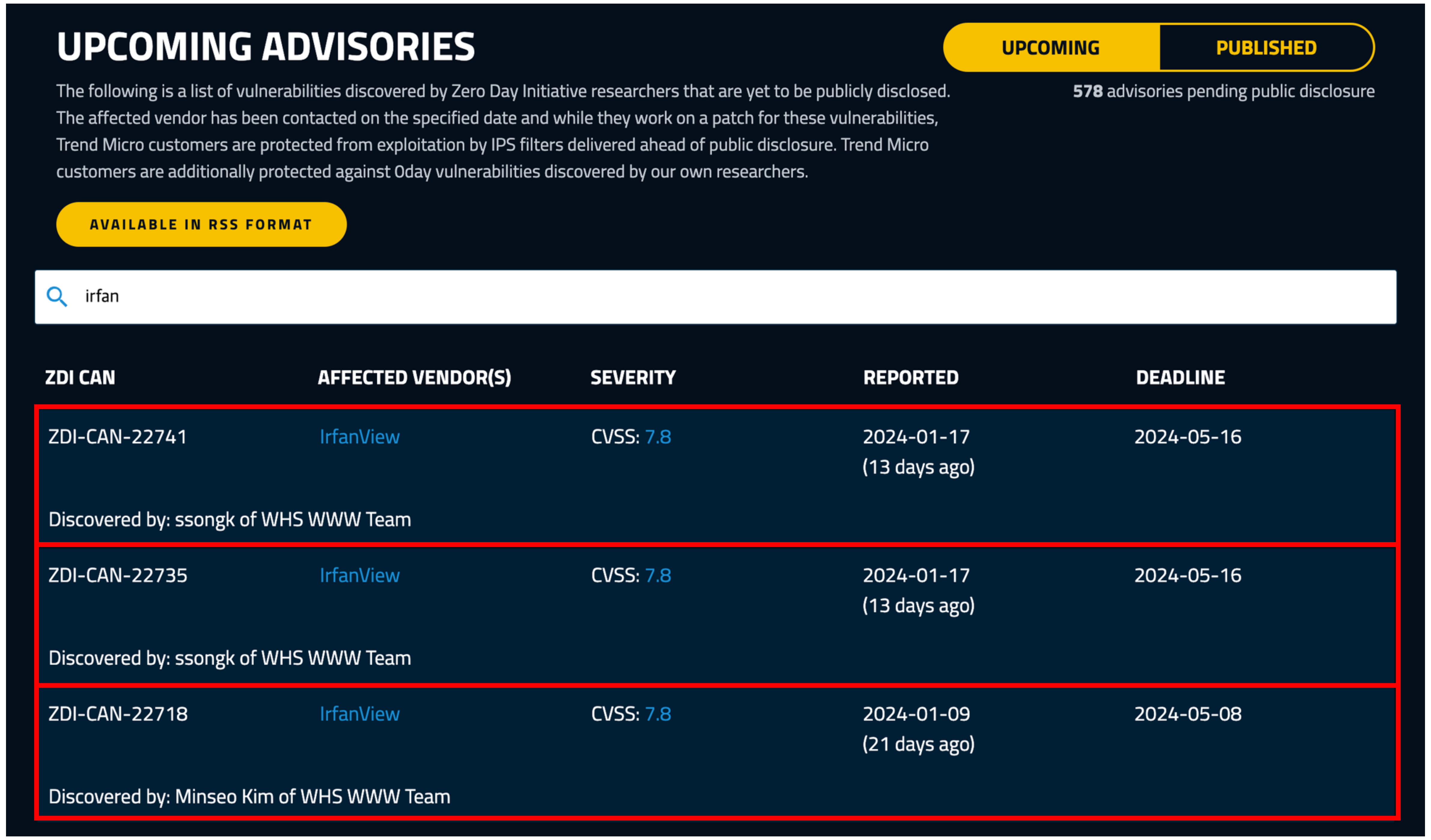
CASE ASSIGNED
2023-11-26 11:31 GMT-6
A case has been opened and added to the queue for review.

CASE INVESTIGATED
2023-12-05 19:01 GMT-6
This case has been investigated.

CASE CONTRACTED
2023-12-18 17:07 GMT-6
This case has been officially contracted to the ZDI.

CASE REVIEWED
2024-01-02 10:34 GMT-6
This case has been reviewed.

VENDOR DISCLOSURE
2024-01-09 16:22 GMT-6
The details of this case have been submitted to the vendor as ZDI-CAN-22718.



UPCOMING ADVISORIES

The following is a list of vulnerabilities discovered by Zero Day Initiative researchers that are yet to be publicly disclosed. The affected vendor has been contacted on the specified date and while they work on a patch for these vulnerabilities, Trend Micro customers are protected from exploitation by IPS filters delivered ahead of public disclosure. Trend Micro customers are additionally protected against 0day vulnerabilities discovered by our own researchers.

578 advisories pending public disclosure

irfan

ZDI CAN	AFFECTED VENDOR(S)	SEVERITY	REPORTED	DEADLINE
ZDI-CAN-22741	IrfanView	CVSS: 7.8	2024-01-17 (13 days ago)	2024-05-16
Discovered by: ssongk of WHS WWW Team				
ZDI-CAN-22735	IrfanView	CVSS: 7.8	2024-01-17 (13 days ago)	2024-05-16
Discovered by: ssongk of WHS WWW Team				
ZDI-CAN-22718	IrfanView	CVSS: 7.8	2024-01-09 (21 days ago)	2024-05-08
Discovered by: Minseo Kim of WHS WWW Team				

감사합니다.

QnA

화이트햇 스쿨 1기 강찬송(ssong_k), 김민서(__yeonyeon)

POC SECURITY