
hooking & visualization

Jaeyong Kim (BlueH4G at gmail dot com)
2013 CodeEngn Conference 09

AGENDA

1. Introduce
 2. about this presentation
 3. why did i do it?
 4. what is hooking?
 5. what to do with hooking?
 6. Demo
 7. QnA
-

who is me?

김재용 26세 (xx 염색체)

이글루시큐리티 & B10S & Hackerschool WG

<http://wargame.kr>

blueh4g at gmail dotcom

about this presentation



why did i do it?



why did i do it?

branch: 트래킹-기능-추가 **fuzzer** / +

트래킹 실패시 파일 삭제 ...

cdpython authored 11 months ago

Requirement	간단한 오류 수정
sample	크래시 트래킹 기능 추가
tmp	크래시 트래킹 기능 추가
README.md	트래킹 무한루프 카운트 감소
fuzzer.py	트래킹 실패시 파일 삭제

pydbg 를 이용한 커스텀 퍼저



Software Engineering
Carnegie Mellon

HOME | Software Assurance | Secure Systems | Orga

ERT Failure Observation Engine

downloads

Carnegie Mellon 의 FOE



기타 등등....

why did i do it?

Immunity Debugger - my_rev_tcp.exe

File View Debug Plugins ImmLib Options Window Help Jobs

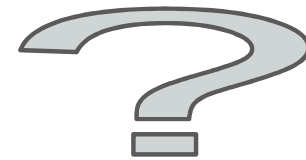
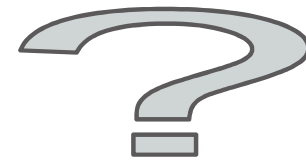
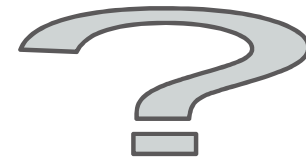
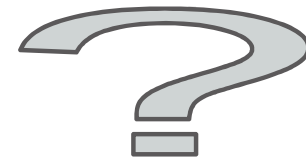
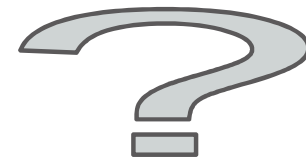
CDLL - main thread module ntdll

M Memory map

Address	Size	Owner	Section	Contains	Type	Access	Initial	Mapped as
00010000	00010000				Map	RW	RW	
00020000	00010000				Map	RW	RW	
00040000	00001000				Imag	R	RWE	
00089000	00007000				Priv	??? Gua	RW	
00280000	00001000				Priv	??? Gua	RW	
0028E000	00002000			stack of ma	Priv	RW Gua	RW	
00290000	00004000				Map	R	R	
002A0000	00001000				Priv	RW	RW	
002B0000	00067000				Map	R	R	\Device\HarddiskVolume2
00360000	00003000				Priv	RW	RW	
00400000	00001000	my_rev_t		PE header	Imag	R	RWE	
00401000	00001000	my_rev_t	.text	code	Imag	R E	RWE	
00402000	00001000	my_rev_t	.data	data	Imag	RW Cop	RWE	
00403000	00001000	my_rev_t	.rdata		Imag	R	RWE	
00404000	00001000	my_rev_t	.bss		Imag	RW Cop	RWE	

D Dump - my_rev_t:.data 00402000..00402FFF

Address	Size	Owner	Section	Contains	Type	Access	Initial	Mapped as
00402000	FC E8 89 00 00 00 60 89 E5 31 D2 64 8B 52 30 8B							ARé... 'eh10d0R00
00402010	52 0C 88 52 14 88 72 28 0F 87 4A 26 31 FF 31 C0							R.örRlor(*EJ&1 1^
00402020	AC 3C 61 7C 02 2C 20 C1 CF 0D 01 C7 E2 F0 52 57							C<a!0, +B.0ã0-RW
00402030	8B 52 10 8B 42 3C 01 D0 8B 40 78 85 C0 74 4A 01							ôR>ôB<ôôô&ad^tJ0
00402040	D0 50 8B 48 18 8B 58 20 01 D3 E3 3C 49 8B 34 8B							ôPôH^ôX ô&K^Iô4ô
00402050	01 D6 31 FF 31 C0 AC C1 CF 0D 01 C7 38 E0 75 F4							0i1 1^C^B.0ã0du~
00402060	03 7D F8 3B 7D 24 75 E2 8B 58 24 01 D3 66 8B							^?%;)su0XôX\$0&fô
00402070	0C 4B 8B 58 1C 01 D3 8B 04 8B 01 D0 89 44 24 24							.KôXL0&ô^ôôô&D\$
00402080	5B 5B 61 59 5A 51 FF E0 58 5F 5A 8B 12 EB 86 5D							[[aVZQ ôX_Zô^Uô]
00402090	68 33 32 00 00 68 77 73 32 5F 54 68 4C 77 26 07							h32..hws2_ThLw&.
004020A0	FF 05 B8 90 01 00 00 29 C4 54 50 68 29 80 68 00							NSE0..)-TPH)Ck.
004020B0	FF 05 50 50 50 50 50 68 EA 0F DF E0 FF							NPPPPôPôPhô^ô
004020C0	D5 89 C7 68 C0 A8 28 C8 68 02 00 00 50 89 E6 6A							Rêãh^E(^h0..P&3j
004020D0	10 56 57 68 9A 74 61 FF D5 68 63 6D 64 00 89							^UWhôata Rhomd.ê
004020E0	E3 57 57 57 31 F6 6A 12 59 56 E2 FD 66 C7 44 24							RWUW1+j^VUô^f&D\$
004020F0	3C 01 01 8D 44 24 10 C6 00 44 54 50 56 56 56 46							<002D\$^A.DTPUUV
00402100	56 4E 56 56 53 68 79 CC 3F 86 FF D5 89 E0 4E							UNUUSUhyIf?ô RêôN
00402110	56 46 FF 30 68 08 87 1D 60 FF D5 B8 F0 B5 A2 56							UF ôh^c+^~ô^du
00402120	68 A6 95 8D 90 FF D5 3C 06 7C 0A 80 FB E0 75 05							h2L2L R<+!.Cûôu\$
00402130	BB 47 13 72 6F 6A 00 53 FF D5 00 00 00 00 00 00							gG!!roj.S R.....

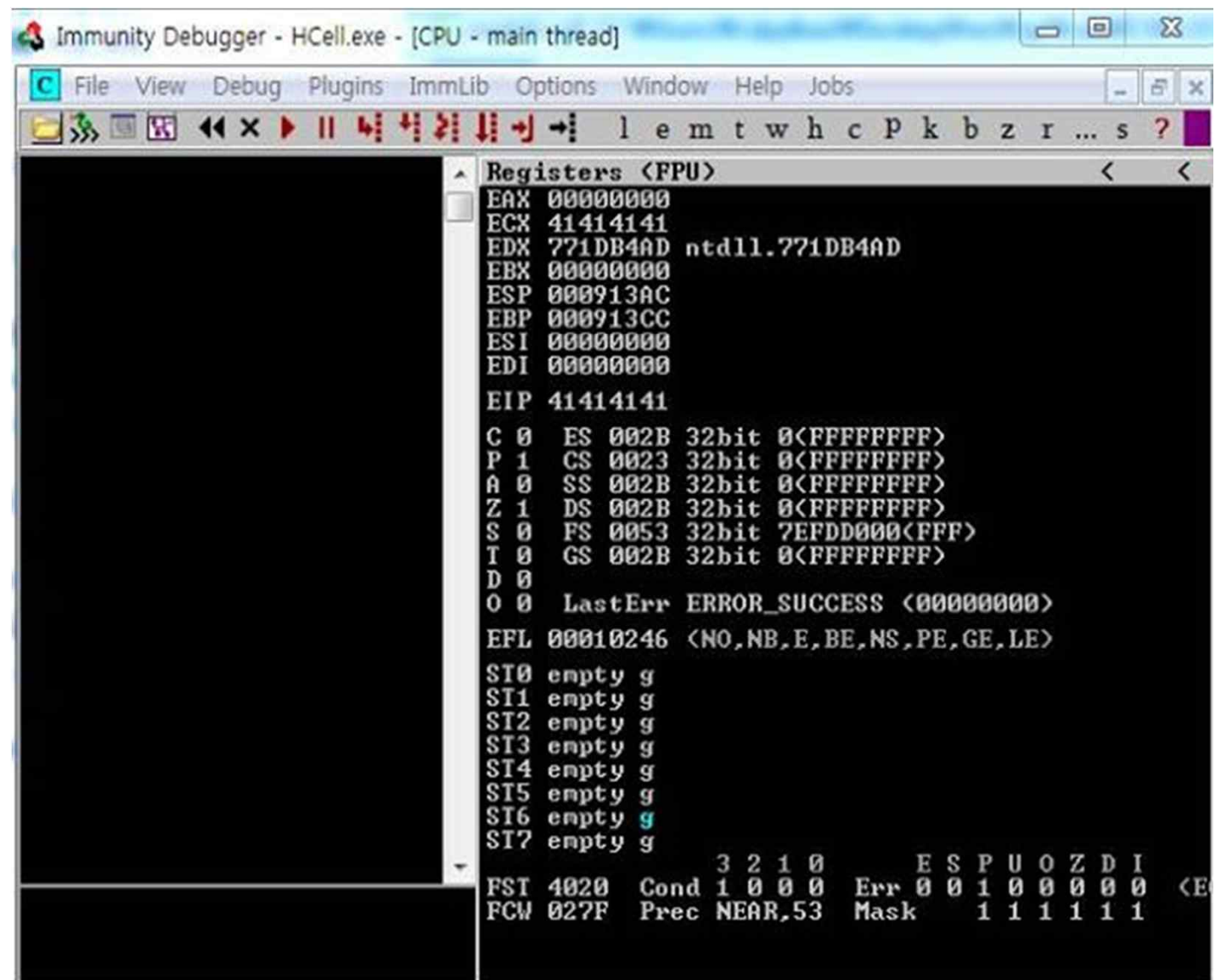


why did i do it?

EIP
41414141
1

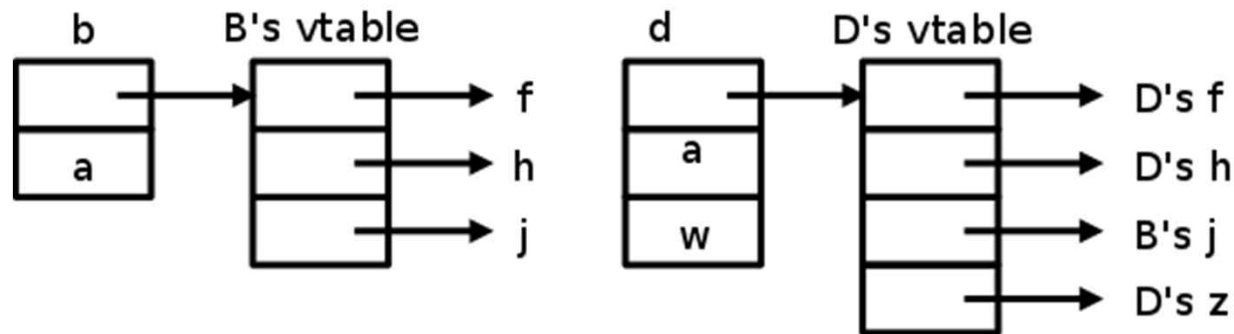
?????

did you
dream
about
dragon?



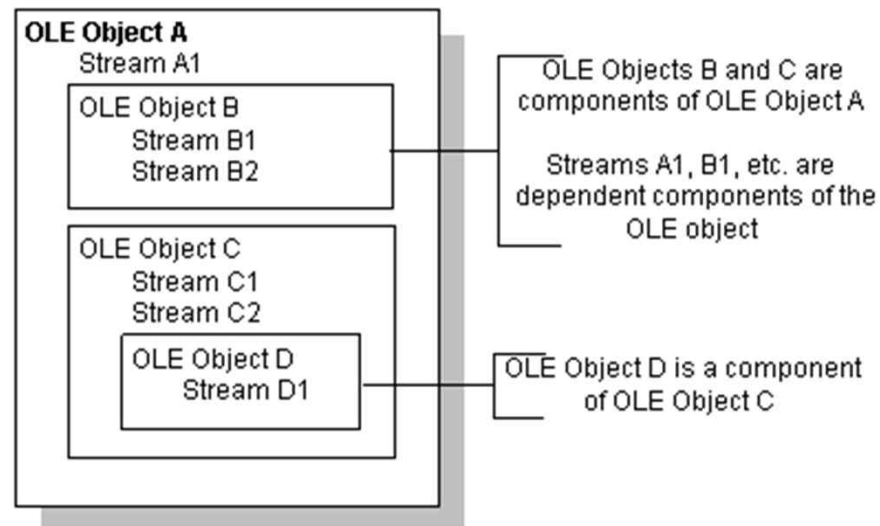
```
Immunity Debugger - HCell.exe - [CPU - main thread]
File View Debug Plugins ImmLib Options Window Help Jobs
l e m t w h c P k b z r ... s ?
Registers (FPU)
EAX 00000000
ECX 41414141
EDX 771DB4AD ntdll.771DB4AD
EBX 00000000
ESP 000913AC
EBP 000913CC
ESI 00000000
EDI 00000000
EIP 41414141
C 0 ES 002B 32bit 0<FFFFFFFF>
P 1 CS 0023 32bit 0<FFFFFFFF>
A 0 SS 002B 32bit 0<FFFFFFFF>
Z 1 DS 002B 32bit 0<FFFFFFFF>
S 0 FS 0053 32bit 7EFDD000<FFF>
T 0 GS 002B 32bit 0<FFFFFFFF>
D 0
O 0 LastErr ERROR_SUCCESS <00000000>
EFL 00010246 <NO,NB,E,BE,NS,PE,GE,LE>
ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
ST5 empty g
ST6 empty g
ST7 empty g
FST 4020 Cond 1 0 0 0 Err 0 0 1 0 0 0 0 0 <E
FCW 027F Prec NEAR,53 Mask 1 1 1 1 1 1
```


why did i do it?

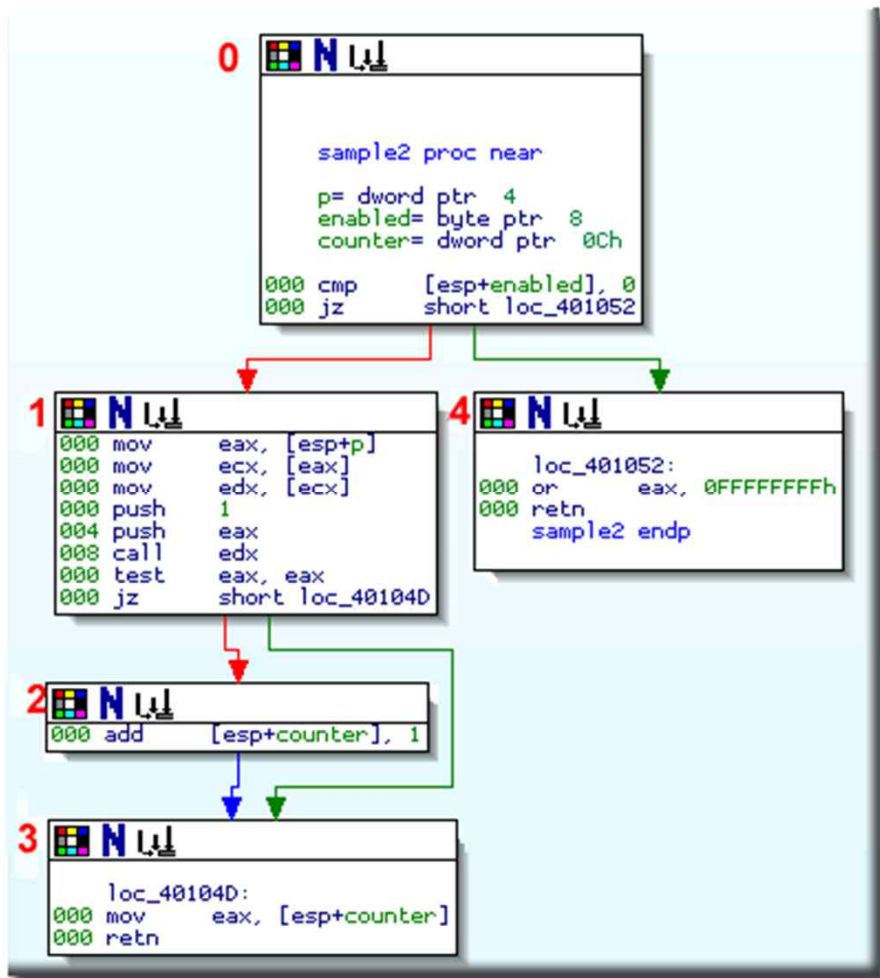


vtable!

OLE Structure!



what is hooking?



I want to know flow
application flow!

Basic block?

or... other?

what is hooking?

2:16:18.831 PM	1	MSVCR110.dll	LeaveCriticalSection (0x5c049d80)	
2:16:18.831 PM	1	MSVCR110.dll	EnterCriticalSection (0x5c049d68)	
2:16:18.831 PM	1	MSVCR110.dll	EnterCriticalSection (0x00627064)	
2:16:18.831 PM	1	MSVCR110.dll	GetConsoleMode (0x00000003, 0x0026fc28)	TRUE
2:16:18.831 PM	1	MSVCR110.dll	ReadFile (0x00000003, 0x5c047280, 0x00001000, 0x0026fc34, NULL)	TRUE
2:16:18.831 PM	1	kernel32.dll	RtlEnterCriticalSection (0x76250120)	STATUS_SUCCESS
2:16:18.831 PM	1	kernel32.dll	memcpy (0x0026fa68, 0x762507c0, 0x00000014)	0x0026fa68
2:16:18.831 PM	1	kernel32.dll	RtlLeaveCriticalSection (0x76250120)	STATUS_SUCCESS
2:16:24.348 PM	1	MSVCR110.dll	LeaveCriticalSection (0x00627064)	
2:16:24.348 PM	1	MSVCR110.dll	LeaveCriticalSection (0x5c049d68)	
2:16:24.348 PM	1	MSVCR110.dll	GetLastError ()	ERROR_SUCCESS
2:16:24.348 PM	1	MSVCR110.dll	FlsGetValue (0x00000004)	0x00626c90
2:16:24.348 PM	1	MSVCR110.dll	SetLastError (ERROR_SUCCESS)	
2:16:24.348 PM	1	MSVCR110.dll	GetLastError ()	ERROR_SUCCESS
2:16:24.348 PM	1	MSVCR110.dll	FlsGetValue (0x00000004)	0x00626c90

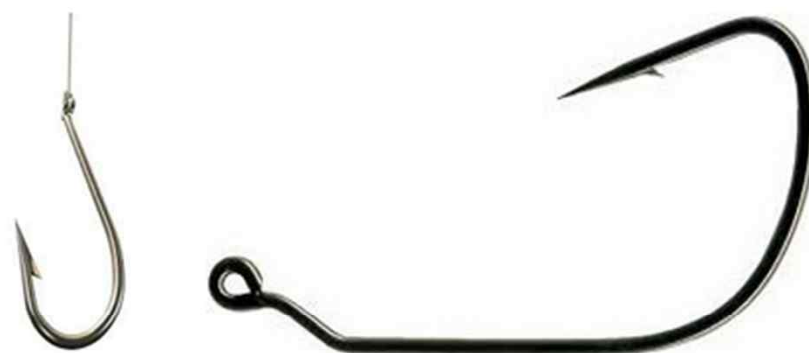
WinAPI - Windows Application Programming Interface

윈도우에서 사용되는 모든 어플리케이션은 winapi를 사용한다.

모든 WinAPI에 후킹을 걸어두고 flow 를 tracing 한다면?

what to do with hooking?

What is hooking?



후킹

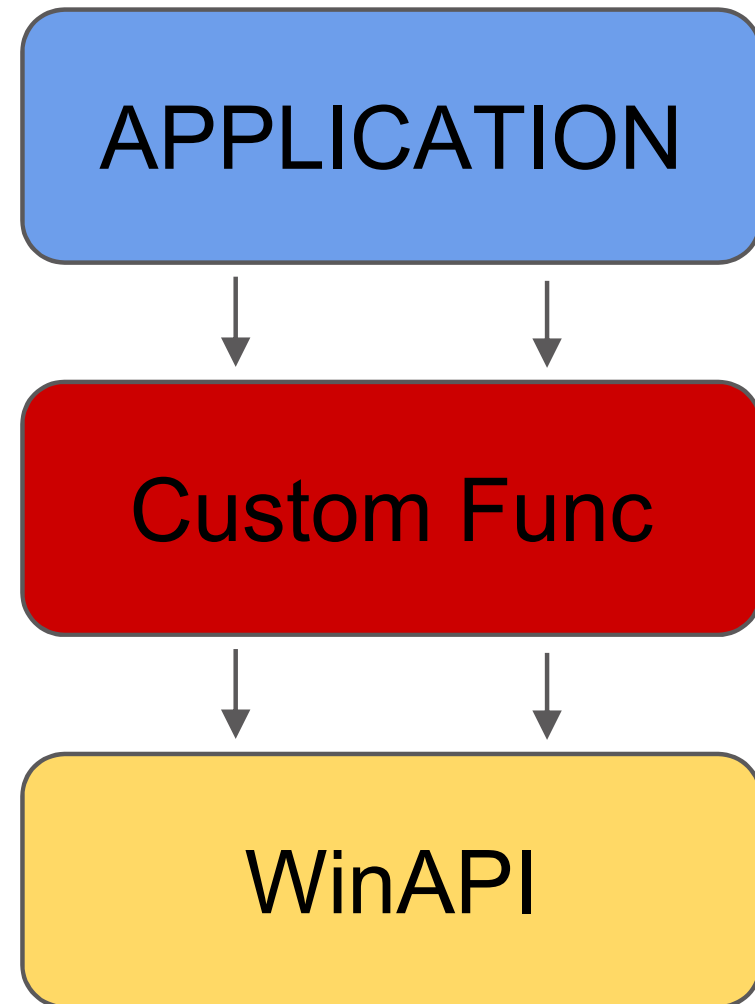
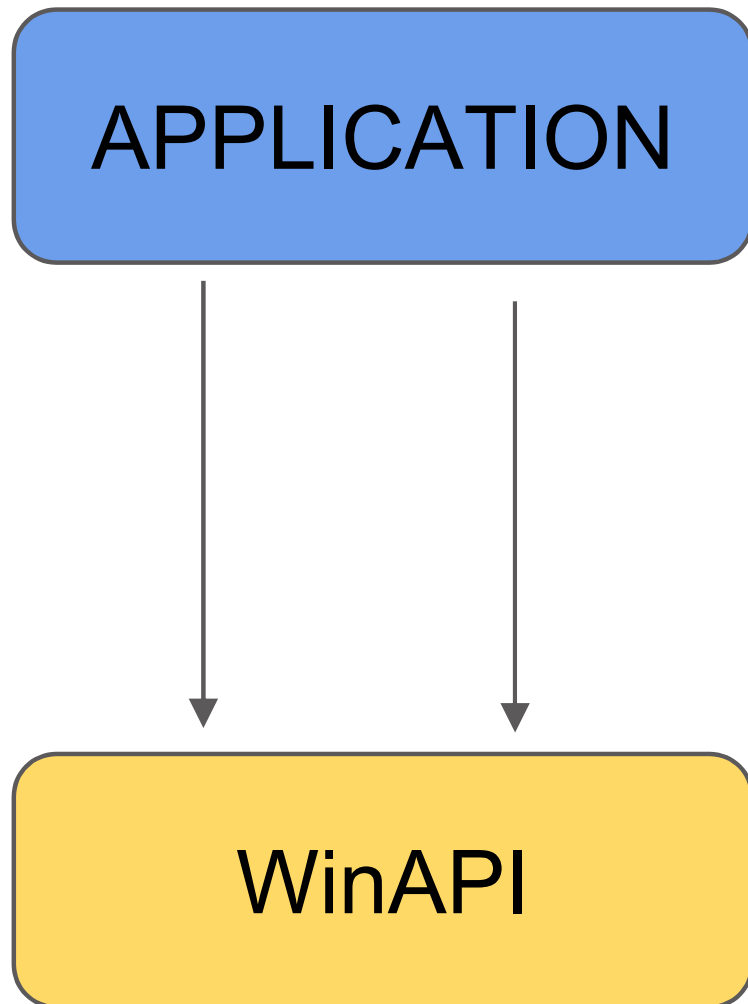
위키백과, 우리 모두의 백과사전.

후킹(영어: hooking)은 소프트웨어 공학 용어로, 운영 체제나 응용 소프트웨어 등의 각종 컴퓨터 프로그램에서 소프트웨어 구성 요소 간에 발생하는 함수 호출, 메시지, 이벤트 등을 중간에서 바꾸거나 가로채는 명령, 방법, 기술이나 행위를 말한다. 이때 이러한 간섭된 함수 호출, 이벤트 또는 메시지를 처리하는 코드를 후크(영어: hook)라고 한다.

크래킹(불법적인 해킹)을 할 때 크래킹 대상 컴퓨터의 메모리 정보, 키보드 입력 정보 등을 빼돌리기 위해서 사용되기도 한다.

예를 들어 특정한 API를 후킹하게 되면 해당 API의 리턴값을 조작하는 등의 동작을 수행할 수 있다.

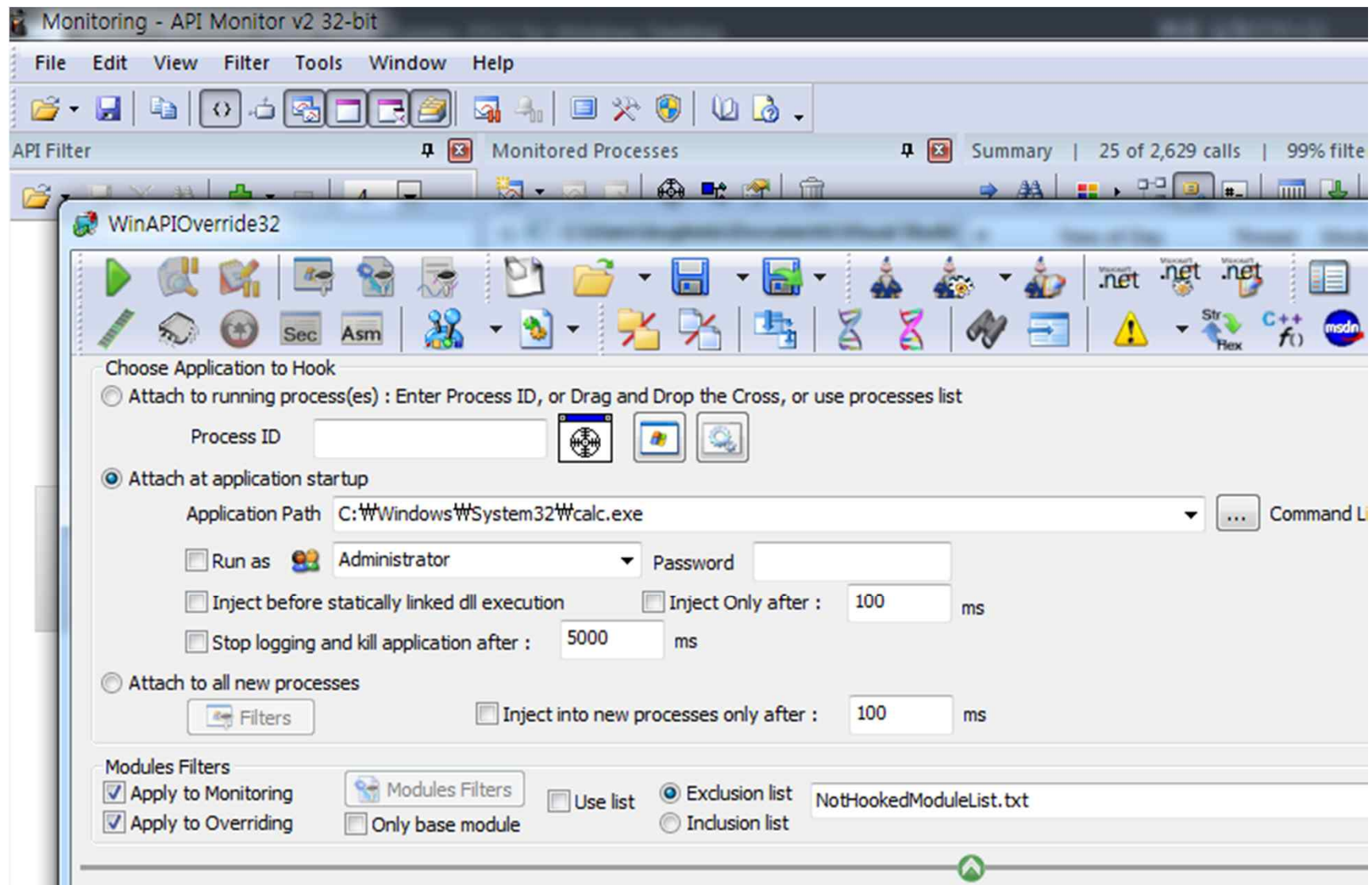
what to do with hooking?



so, what?

1. Application Flow Analysis
 2. WinAPI Parameter, return value monitoring
 3. Crash & Original sample diffing (in App)
 4. Call Stack log of WinAPI
 5. memcpy, heapalloc etc.. API tagging
 6. invalid param & invalid ret tagging
 7. basic rule is readability
-

hooking tools



hooking tools

WinAPIOverride32/64

- Opensource (Thx!)
- jacquelin.potier.free.fr/winapioverride32/

API Monitor v2 32/64

- not opensource (but free)
 - www.rohitab.com
-

Demo

Demo

another episode..

1. RtlWriteDecodedUcsDataIntoSmartLBlobUcsWritingContext
2. full GUI interface?

QnA

Question
&
Answer
...?

질문은 없는걸로...



thx

이후에도 궁금한점이 있으시면 메일 보내주세요 :D

blueh4g [at] gmail {dot} com