Module 2

Typical goals of malware and their implementations

https://github.com/hasherezade/malware_training_vol1

Hooking

Hooking: the idea

- Hooking means intercepting the original execution of the function with a custom code
- Goal: to create a **proxy** through which the input/output of the called function bypasses
- Possible watching and/or interference in the input/output of the function

Hooking: the idea

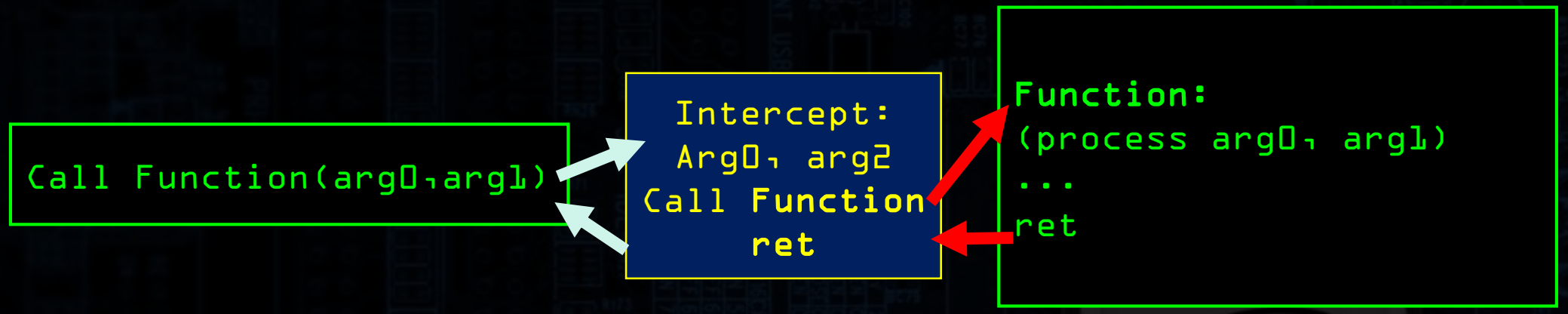
- Calling the function with no hook:

```
Call Function(arg0,arg1)
```

```
Function:
(process arg0, arg1)
...
ret
```

Hooking: the idea

- Calling the hooked function: the high-level goals



Hooking: who?

Hooking is used for intercepting and modifying API calls

- By malware: i.e. spying on data
- By Anti-malware: monitoring execution
- Compatibility patches (Operating System level) - i.e. shimming engine
- Extending functionality of the API

Hooking in malware

- Sample purposes of hooks used by malware:
  - Hiding presence in the system (rootkit component)
  - Sniffing executions of APIs (spyware)
  - Doing defined actions on the event of some API being called (i.e. propagation to a newly created processes, screenshot on click)
  - Redirection to a local proxy (in Banking Trojans)

Hooking: how?

There are various, more or less documented methods of hooking. Examples:

- Kernel Mode (*will not be covered in this course)
- User Mode:
  - SetWindowsEx etc. – monitoring system events
  - Windows subclassing – intercepting GUI components
  - Inline/IAT/EAT Hooking – general API hooking

Monitoring system events

- Windows allows for monitoring certain events, such as:
  - WH_CALLWNDPROC – monitor messages sent to a window
  - WH_KEYBOARD
  - WH_KEYBOARD_LL
  - etc.
- The hook can be set via `SetWindowsHookEx`
- This type of hooks are often used by keyloggers

Monitoring system events

- Example: Remcos RAT

```
sub_405CC7(v6, v7, v8, v9);
std::basic_string<char,std::char_traits<char>,std::allo
  &v6,
  "Online Keylogger Started",
  &v11);
std::basic_string<char,std::char_traits<char>,std::allo
  &v2,
  "[INFO]",
  &v10);
sub_411AE4(v2, v3, v4, v5, v6);
if ( !*((_BYTE *)v1 + 60) )
{
  sub_40504A(v1);
  if ( !*(_DWORD *)v1 )
    CreateThread(0, 0, keylogging_thread, v1, 0, 0);
```

```
cmp     [esi], edi
jnz     short loc_4051F4

push    edi                 ; dwThreadId
push    edi                 ; hmod
push    offset to_get_keys  ; lpfn
push    13                  ; idHook
call    ds:SetWindowsHookExA
mov     [esi], eax          ; 13 -> WH_KEYBOARD_LL
```

https://www.virustotal.com/gui/file/47593a26ec7a9e791bb1c94f4c4d56deaae25f37b7f77b0a44dc93ef0bca91fd

Monitoring system events

- Example: Remcos RAT

```
cmp     [esi], edi
jnz     short loc_4051F4

push    edi                 ; dwThreadId
push    edi                 ; hmod
push    offset to_get_keys ; lpfn
push    13                  ; idHook
call    ds:SetWindowsHookExA
mov     [esi], eax          ; 13 -> WH_KEYBOARD_LL
```

```
1  LRESULT __stdcall get_keys_state(int a1, int nCode, WPARAM wParam, LPARAM lParam)
2  {
3    qmemcpy((void *)(a1 + 64), (const void *)lParam, 0x14u);
4    if ( !nCode )
5    {
6      switch ( wParam )
7      {
8        case 0x100u:
9          *(_BYTE *)(a1 + 44) = GetKeyState(20) && GetKeyState(20) != 0xFF80u;
10         sub_406A9B(a1);
11         sub_406ABF(a1);
12         process_keys(a1);
13         if ( !*(_BYTE *)(a1 + 47) )
14           sub_406846(a1);
15         *(_BYTE *)(a1 + 47) = 0;
16         break;
17       case 0x101u:
18         sub_406AAD(a1);
19         sub_406AD1(a1);
20         sub_406A55(a1);
21         break;
22       case 0x104u:
23         sub_4069C5(a1);
24         break;
25     }
26   }
27   return CallNextHookEx(*(HHOOK *)a1, nCode, wParam, lParam);
28 }
```

Windows subclassing

- This type of hooking can be applied on GUI components
- Window subclassing was created to extend functionality of the GUI controls
- You can set a new procedure that intercepts the messages of the GUI controls
- Related APIs:
  - SetWindowLong, SetWindowLongPtr (the old approach: ComCtl32.dll < 6)
  - SetWindowSubclass/RemoveWindowSubclass, SetProp/GetProp (the new approach: ComCtl>=6)
- Subclassed window gets a new property in: UxSubclassInfo or CC32SubclassInfo (depending on the API version)

https://docs.microsoft.com/en-us/windows/win32/controls/subclassing-overview

Windows subclassing

- Windows subclassing can also be used by malware
- Example: subclassing the Tray Window in order to execute the injected code

https://github.com/hasherezade/demos/blob/master/inject_shellcode/src/window_long_inject.cpp

General API Hooking

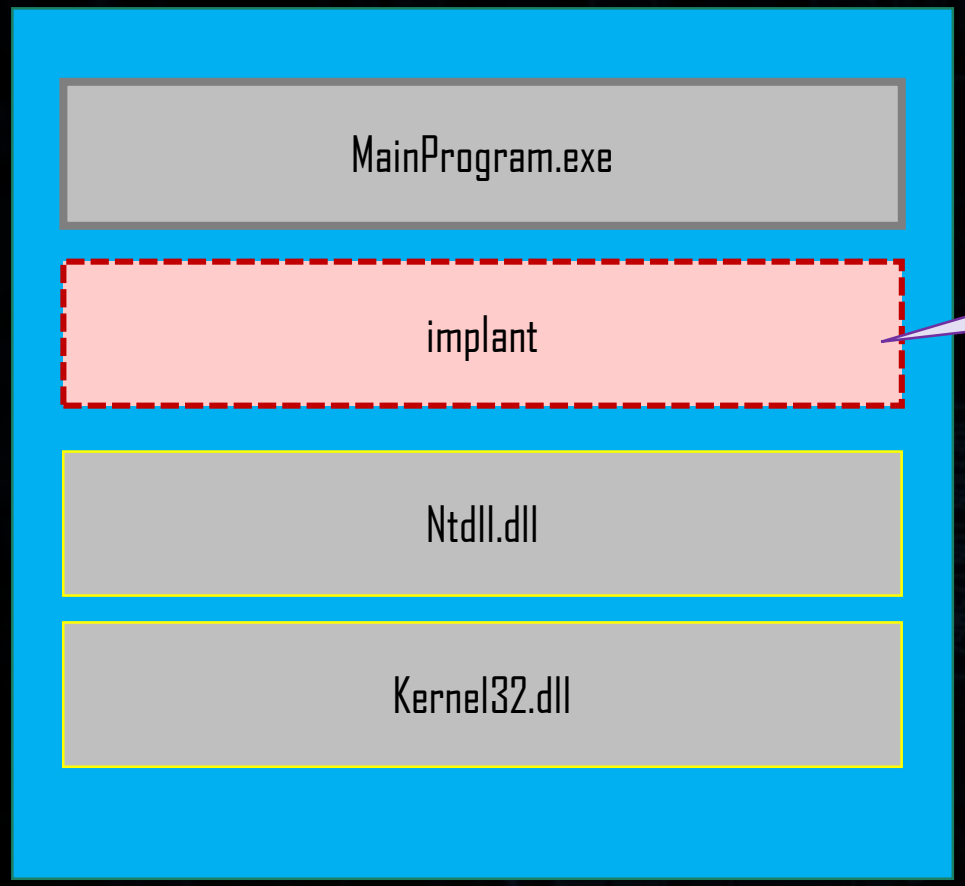
- Most common and powerful, as it helps to intercept any API
- Types of userland hooks:
  - Inline hooks (the most common)
  - IAT Hooks
  - EAT Hooks

API Hooking: the idea

Hooking API of a foreign process requires:

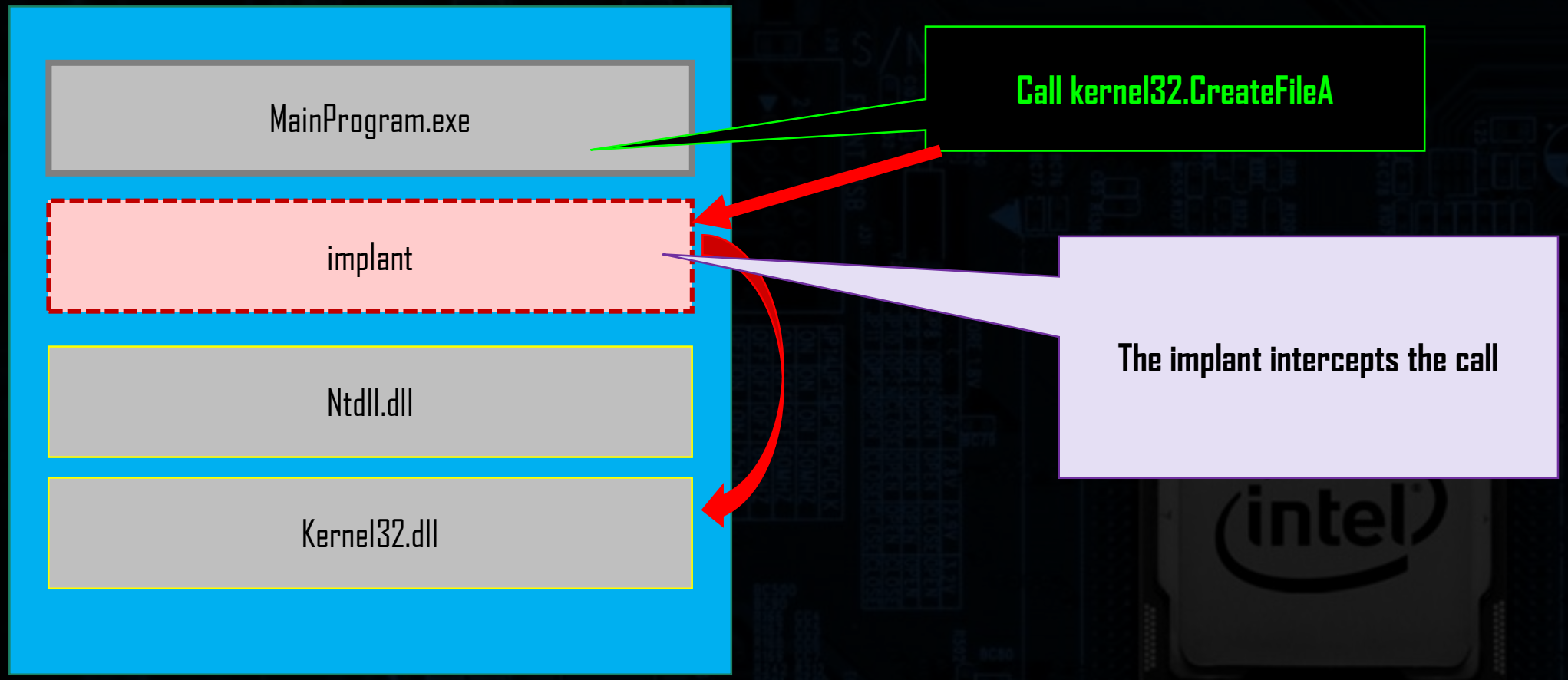
1. Implanting your code into the target process
2. Redirecting the original call, so that it will pass through the implant

Implanting a foreign code



Any code that was added to the original process. It can be a PE (DLL, EXE), or a shellcode

Implanting a foreign code



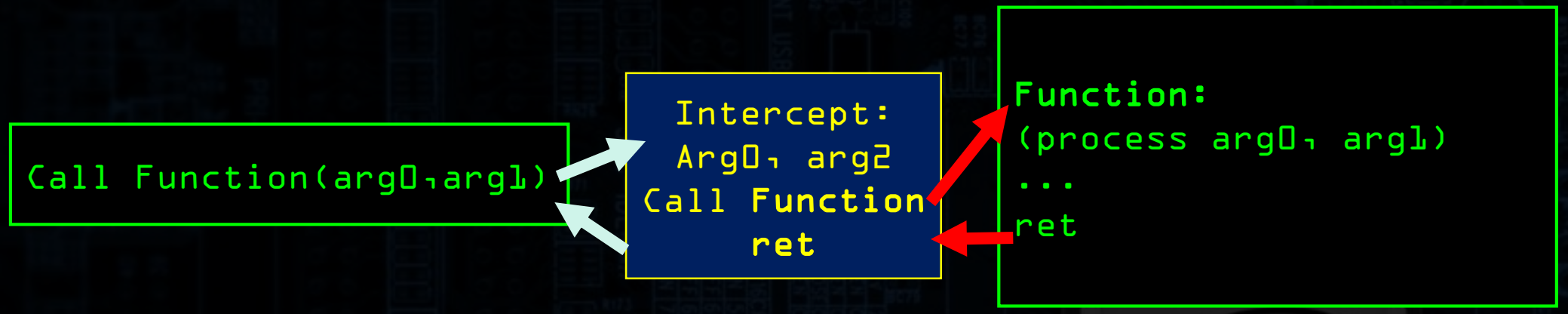
IAT Hooking

IAT Hooking

- In case of IAT hooks, the address in the Import Table is altered
- IAT hooks are often used by Windows compatibility patches, shims
- Not as often (but sometimes) used by malware

IAT Hooking: idea

- In case of IAT Hooking we can really implement it in this simple way: by replacing the address via which the function is called in the IAT



IAT Hooking

The address filled in IAT leads to User32.dll (as the table points)

USER32.dll [21 entries]

Call via	Name	Ordinal	Original Thunl	Thunk
4060	SetDlgItemTextW	-	4362	764EF550
4064	GetDlgItemTextW	-	4374	764EEE
4068	MessageBoxW	-	4386	10001
406C	CreateWindowExW	-	4394	764F19A
4070	GetDesktopWindow	-	43A6	764F9610
4074	CharUpperW	-	43BA	76504B70
4078	LoadIconW	-	43C8	764FDE10
407C	RegisterClassExW	-	43D4	764F1660

Base address	Type	Size	Protect...	Use
0x764c0000	Image	1,6...	WCX	C:\Windows\SysWOW64\user32.dll
0x764c0000	Image: Commit	4 kB	R	C:\Windows\SysWOW64\user32.dll
0x764c1000	Image: Commit	192 kB	RX	C:\Windows\SysWOW64\user32.dll
0x764f1000	Image: Commit	4 kB	RWX	C:\Windows\SysWOW64\user32.dll
0x764f2000	Image: Commit	72 kB	RX	C:\Windows\SysWOW64\user32.dll
0x76504000	Image: Commit	4 kB	RWX	C:\Windows\SysWOW64\user32.dll
0x76505000	Image: Commit	380 kB	RX	C:\Windows\SysWOW64\us 32.dll
0x76564000	Image: Commit	8 kB	RW	C:\Windows\Sys W6 32.dll
0x76566000	Image: Commit	972 kB	R	C:\W
0x76660000	Image	56 kB	WCX	
0x766e0000	Image	2,5...	WCX	
0x76ad0000	Image	72 kB	WCX	

original

IAT Hooking

The address filled in IAT leads to a different module

USER32.dll [21 entries]

Call via	Name	Ordinal	Original Thunl	Thunk
4060	SetDlgItemTextW	-	4362	764EF550
4064	GetDlgItemTextW	-	4374	764EEE90
4068	MessageBoxW	-	4386	1000100C
406C	CreateWindowExW	-	4394	764F19A0
4070	GetDesktopWindow	-	43A6	764F9610
4074	CharUpperW	-	43BA	76504B70
4078	LoadIconW	-	43C8	764FDE10
407C	RegisterClassExW	-	43D4	764F1660

Base address	Type	Size	Protect...	Use
0x1fd0000	Private	64 kB	RW	Heap 32-bit (ID 3)
0x1fe0000	Mapped	892 kB	R	
0x20c0000	Mapped	3,2...	R	C:\Windows\Globalization\Sorting\SortDefault.nls
0x2b60000	Mapped	18,...	R	C:\Windows\Fonts\StaticCache.dat
0x10000000	Image	20 kB	WCX	C:\Users\IEUser\Desktop\Hooked\NagMeNot.dll
0x10000000	Image: Commit	4 kB	R	C:\Users\IEUser\Desktop\Hooked\NagMeNot.dll
0x10001000	Image: Commit	4 kB	RX	C:\Users\IEUser\Desktop\Hooked\NagMeNot.dll
0x10002000	Image: Commit	4 kB	R	C:\Users\IEUser\Desktop\Hooked\NagMeNot.dll
0x10003000	Image: Commit	4 kB	WC	C:\Users\IEUser\Desktop\Hooked\NagMeNot.dll
0x10004000	Image: Commit	4 kB	R	C:\Users\IEUser\Desktop\Hooked\NagMeNot.dll
0x72910000	Image	152 kB	WCX	C:\Wi

hooked

IAT Hooking - the pros

- IAT hooking is much easier to implement than inline hooking
- The original DLL is unaltered, so we can call the functions from it via the intercepting function directly – no need for the trampoline

IAT Hooking - the cons

- IAT hooking can intercept only the functions that are called via import table
  - Cannot hook lower level functions that are called underneath
- Cannot set hooks globally for the process – each module importing the function has to be hooked separately

IAT hooking detection

- IAT Hooking is detected i.e. by PE-sieve/HollowsHunter

```
Pe-sieve.exe /pid <my_pid> /iat
Hollows_hunter.exe /iat
```

400000.hooked.exe.iat_hooks.txt

```
1  4068;user32.MessageBoxW #533->nagmenot.MBox #1;10000000+100c;0
2
```

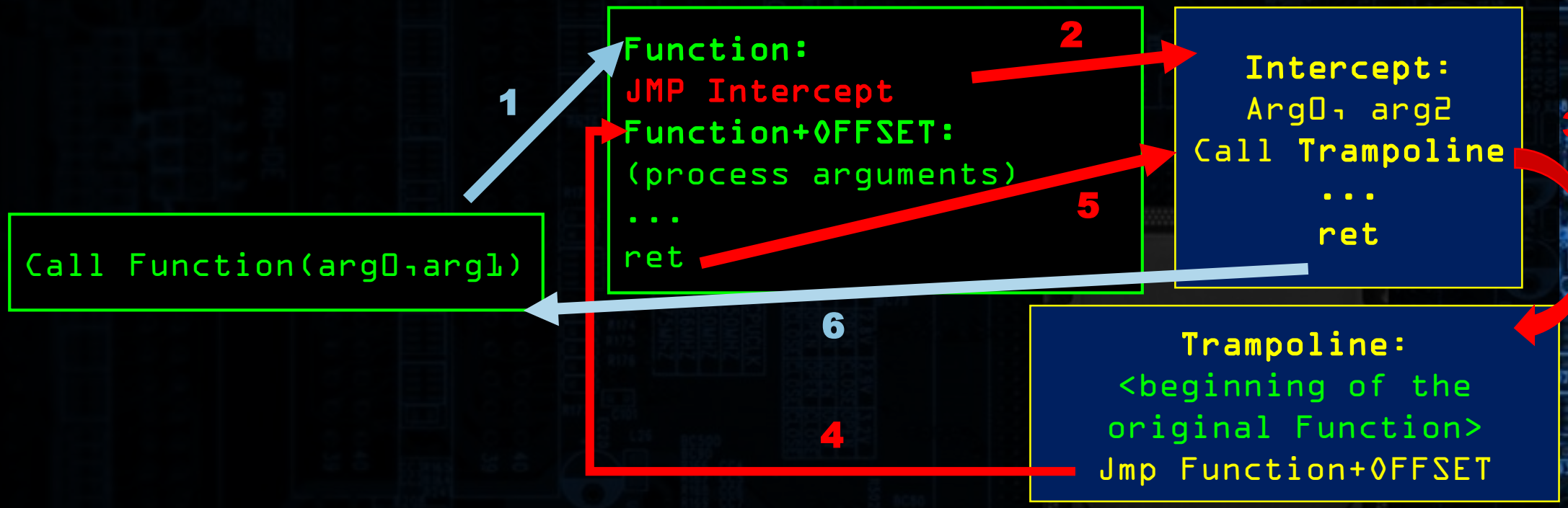
Inline Hooking

Inline Hooking

- In case of Inline hooks, the beginning of the original function is altered
- Inline hooks may also be used in legitimate applications
- Extremely often used in malware

Inline Hooking: idea

- In case of Inline Hooking we need to overwrite the beginning of the function: so, calling the original one gets more complicated...



Inline Hooking: example

- Example of an inline hook installed by a malware in function `CertGetCertificateChain`

```
76036CCE   90            nop
76036CCF   8BFF          mov edi,edi                          CertGetCertificateChain
76036CD1   55            push ebp
76036CD2   8BEC          mov ebp,esp
76036CD4   51            push ecx
76036CD5   51            push ecx
76036CD6   53            push ebx
76036CD7   56            push esi
76036CD8   57            push edi
76036CD9   8B7D 08       mov edi,dword ptr ss:[ebp+8]
76036CDC   8D45 FC       lea eax,dword ptr ss:[ebp-4]
76036CDF   33DB          xor ebx,ebx
76036CE1   50            push eax
76036CE2   897D 08       mov dword ptr ss:[ebp+8],edi
76036CE5   895D FC       mov dword ptr ss:[ebp-4],ebx
76036CE8   E8 A8000000   call crypt32.76036D95
```

Inline Hooking: example

- Example of an inline hook installed by a malware in function `CertGetCertificateChain`

```
76036CCE    90              nop
76036CCF  ^ E9 6179178A     jmp <hook_CertGetCertificateChain>      CertGetCertificateChain
76036CD4    51              push ecx                                hook_trampoline1_target
76036CD5    51              push ecx
76036CD6    53              push ebx
76036CD7    56              push esi
76036CD8    57              push edi
76036CD9    8B7D 08         mov edi,dword ptr ss:[ebp+8]
76036CDC    8D45 FC         lea eax,dword ptr ss:[ebp-4]
76036CDF    33DB            xor ebx,ebx
76036CE1    50              push eax
76036CE2    897D 08         mov dword ptr ss:[ebp+8],edi
76036CE5    895D FC         mov dword ptr ss:[ebp-4],ebx
76036CE8    E8 A8000000     call crypt32.76036D95
```

Original: CertGetCertificateChain

```
76036CCE   90                 nop
76036CCF   8BFF               mov edi,edi                    CertGetCertificateChain
76036CD1   55                 push ebp
76036CD2   8BEC               mov ebp,esp
76036CD4   51                 push ecx
76036CD5   51                 push ecx
76036CD6   53                 push ebx
76036CD7   56                 push esi
76036CD8   57                 push edi
76036CD9   8B7D 08            mov edi,dword ptr ss:[ebp+8]
76036CDC   8D45 FC            lea eax,dword ptr ss:[ebp-4]
76036CDF   33DB               xor ebx,ebx
76036CE1   50                 push eax
76036CE2   897D 08            mov dword ptr ss:[ebp+8],edi
76036CE5   895D FC            mov dword ptr ss:[ebp-4],ebx
76036CE8   E8 A8000000        call crypt32.76036D95
                   ...
76036D56   5F                 pop edi
76036D57   5E                 pop esi
76036D58   8BC3               mov eax,ebx
76036D5A   5B                 pop ebx
759A6D5B   C9                 leave
759A6D5C   C2 2000            ret 20                         end of CertGetCertificateChain
```

```
call crypt32.CertGetCertificateChain
```

Hooked: CertGetCertificateChain

```
76036CCE    90              nop
76036CCF  ^ E9 6179178A     jmp <hook_CertGetCertificateChain>   CertGetCertificateChain
76036CD4    51              push ecx                             hook_trampoline1_target
76036CD5    51              push ecx
76036CD6    53              push ebx
76036CD7    56              push esi
76036CD8    57              push edi
76036CD9    8B7D 08         mov edi,dword ptr ss:[ebp+8]
76036CDC    8D45 FC         lea eax,dword ptr ss:[ebp-4]
76036CDF    33DB            xor ebx,ebx
76036CE1    50              push eax
76036CE2    897D 08         mov dword ptr ss:[ebp+8],edi
76036CE5    895D FC         mov dword ptr ss:[ebp-4],ebx
76036CE8    E8 A800000      call crypt32.76036D95
                            ...
76036D56    5F              pop edi
76036D57    5E              pop esi
76036D58    8BC3            mov eax,ebx
76036D5A    5B              pop ebx
759A6D5B    C9              leave
759A6D5C    C2 2000         ret 20                               end of CertGetCertificateChain
```

```
call crypt32.CertGetCertificateChain
```

```
00500FA0
 mov edi,edi
 push ebp
 mov ebp,esp
 jmp crypt32.76036CD4
```

```
crypt32.76036CD4
```

```
<hook_CertGetCertificateChain>
 push ebp ; hook_CertGetCertificateChain
 mov ebp,esp
 push ebx
 push edi
 push esi
 mov esi,dword ptr ss:[ebp+24]
 push esi
 push dword ptr ss:[ebp+20]
 push dword ptr ss:[ebp+1C]
 push dword ptr ss:[ebp+18]
 push dword ptr ss:[ebp+14]
 push dword ptr ss:[ebp+10]
 push dword ptr ss:[ebp+C]
 push dword ptr ss:[ebp+8]
 call dword ptr ds:[2219C8] ; to_trampoline1
 xor edi,edi
 push edi
 push eax
 call 20F6C0
 add esp,8
 mov ebx,eax
 and bl,1
 push edi
 push esi
 call 212400
 add esp,8
 mov cl,1
 test al,cl
 jne 1AE69C
```

```
001AE67B
 mov esi,dword ptr ds:[esi]
 push 0
 push esi
 call 2125F0
 add esp,8
 test al,1
 jne 1AE69C
```

```
001AE68C
 mov bl,1
 mov dword ptr ds:[esi+4],0
 mov dword ptr ds:[esi+8],800
```

```
001AE69C
 movzx eax,bl
 pop esi
 pop edi
 pop ebx
 pop ebp
 ret 20
```

Hooked: CertGetCertificateChain

```
76036CCE    90                pop
76036CCF    E9 6179178A       jmp <hook_CertGetCertificateChain>   CertGetCertificateChain
76036CD4    51                push ecx                             hook_trampoline1_target
76036CD5    51                push ecx
76036CD6    53                push ebx
76036CD7    56                push esi
76036CD8    57                push edi
76036CD9    8B7D 08           mov edi,dword ptr ss:[ebp+8]
76036CDC    8D45 FC           lea eax,dword ptr ss:[ebp-4]
76036CDF    33DB              xor ebx,ebx
76036CE1    50                push eax
76036CE2    897D 08           mov dword ptr ss:[ebp+8],edi
76036CE5    895D FC           mov dword ptr ss:[ebp-4],ebx
76036CE8    E8 A8000000       call crypt32.76036D95
...
76036D56    5F                pop edi
76036D57    5E                pop esi
76036D58    8BC3              mov eax,ebx
76036D5A    5B                pop ebx
759A6D5B    C9                leave
759A6D5C    C2 2000           ret 20                               end of CertGetCertificateChain
```

call crypt32.CertGetCertificateChain

```
00500FA0
 mov edi,edi
 push ebp
 mov ebp,esp
 jmp crypt32.76036CD4
```

crypt32.76036CD4

```
<hook_CertGetCertificateChain>
 push ebp ; hook_CertGetCertificateChain
 mov ebp,esp
 push ebx
 push edi
 push esi
 mov esi,dword ptr ss:[ebp+24]
 push esi
 push dword ptr ss:[ebp+20]
 push dword ptr ss:[ebp+1C]
 push dword ptr ss:[ebp+18]
 push dword ptr ss:[ebp+14]
 push dword ptr ss:[ebp+10]
 push dword ptr ss:[ebp+C]
 push dword ptr ss:[ebp+8]
 call dword ptr ds:[2219C8] ; to_trampoline1
 xor edi,edi
 push edi
 push eax
 call 20F6C0
 add esp,8
 mov ebx,eax
 and bl,1
 push edi
 push esi
 call 212400
 add esp,8
 mov cl,1
 test al,cl
 jne 1AE69C
```

```
001AE67B
 mov esi,dword ptr ds:[esi]
 push 0
 push esi
 call 2125F0
 add esp,8
 test al,1
 jne 1AE69C
```

```
001AE68C
 mov bl,1
 mov dword ptr ds:[esi+4],0
 mov dword ptr ds:[esi+8],800
```

```
001AE69C
 movzx eax,bl
 pop esi
 pop edi
 pop ebx
 pop ebp
 ret 20
```

Hooked: CertGetCertificateChain

```
76036CCE    90                  nop
76036CCF    E9 6179178A         jmp <hook_CertGetCertificateChain>   CertGetCertificateChain
76036CD4    51                  push ecx                             hook_trampoline1_target
76036CD5    51                  push ecx
76036CD6    53                  push ebx
76036CD7    56                  push esi
76036CD8    57                  push edi
76036CD9    8B7D 08             mov edi,dword ptr ss:[ebp+8]
76036CDC    8D45 FC             lea eax,dword ptr ss:[ebp-4]
76036CDF    33DB                xor ebx,ebx
76036CE1    50                  push eax
76036CE2    897D 08             mov dword ptr ss:[ebp+8],edi
76036CE5    895D FC             mov dword ptr ss:[ebp-4],ebx
76036CE8    E8 A8000000         call crypt32.76036D95
...
76036D56    5F                  pop edi
76036D57    5E                  pop esi
76036D58    8BC3                mov eax,ebx
76036D5A    5B                  pop ebx
759A6D5B    C9                  leave
759A6D5C    C2 2000             ret 20                               end of CertGetCertificateChain
```

```
call crypt32.CertGetCertificateChain
```

```
00500FA0
  mov edi,edi
  push ebp
  mov ebp,esp
  jmp crypt32.76036CD4
```

```
crypt32.76036CD4
```

```
<hook_CertGetCertificateChain>
 push ebp ; hook_CertGetCertificateChain
 mov ebp,esp
 push ebx
 push edi
 push esi
 mov esi,dword ptr ss:[ebp+24]
 push esi
 push dword ptr ss:[ebp+20]
 push dword ptr ss:[ebp+1C]
 push dword ptr ss:[ebp+18]
 push dword ptr ss:[ebp+14]
 push dword ptr ss:[ebp+10]
 push dword ptr ss:[ebp+C]
 push dword ptr ss:[ebp+8]
 call dword ptr ds:[2219C8] ; to_trampoline1
 xor edi,edi
 push edi
 push eax
 call 20F6C0
 add esp,8
 mov ebx,eax
 and bl,1
 push edi
 push esi
 call 212400
 add esp,8
 mov cl,1
 test al,cl
 jne 1AE69C
```

```
001AE67B
 mov esi,dword ptr ds:[esi]
 push 0
 push esi
 call 2125F0
 add esp,8
 test al,1
 jne 1AE69C
```

```
001AE68C
 mov bl,1
 mov dword ptr ds:[esi+4],0
 mov dword ptr ds:[esi+8],800
```

```
001AE69C
 movzx eax,bl
 pop esi
 pop edi
 pop ebx
 pop ebp
 ret 20
```

Hooked: CertGetCertificateChain

```
76036CCE    90              nop
76036CCF  ^ E9 6179178A     jmp <hook_CertGetCertificateChain>   CertGetCertificateChain
76036CD4    51              push ecx                             hook_trampoline1_target
76036CD5    51              push ecx
76036CD6    53              push ebx
76036CD7    56              push esi
76036CD8    57              push edi
76036CD9    8B7D 08         mov edi,dword ptr ss:[ebp+8]
76036CDC    8D45 FC         lea eax,dword ptr ss:[ebp-4]
76036CDF    33DB            xor ebx,ebx
76036CE1    50              push eax
76036CE2    897D 08         mov dword ptr ss:[ebp+8],edi
76036CE5    895D FC         mov dword ptr ss:[ebp-4],ebx
76036CE8    E8 A8000000     call crypt32.76036D95
                            ...
76036D56    5F              pop edi
76036D57    5E              pop esi
76036D58    8BC3            mov eax,ebx
76036D5A    5B              pop ebx
759A6D5B    C9              leave
759A6D5C    C2 2000         ret 20                               end of CertGetCertificateChain
```

```
call crypt32.CertGetCertificateChain
```

```
00500FA0
 mov edi,edi
 push ebp
 mov ebp,esp
 jmp crypt32.76036CD4
```

```
crypt32.76036CD4
```

```
<hook_CertGetCertificateChain>
 push ebp ; hook_CertGetCertificateChain
 mov ebp,esp
 push ebx
 push edi
 push esi
 mov esi,dword ptr ss:[ebp+24]
 push esi
 push dword ptr ss:[ebp+20]
 push dword ptr ss:[ebp+1C]
 push dword ptr ss:[ebp+18]
 push dword ptr ss:[ebp+14]
 push dword ptr ss:[ebp+10]
 push dword ptr ss:[ebp+C]
 push dword ptr ss:[ebp+8]
 call dword ptr ds:[2219C8] ; to_trampoline1
 xor edi,edi
 push edi
 push eax
 call 20F6C0
 add esp,8
 mov ebx,eax
 and bl,1
 push edi
 push esi
 call 212400
 add esp,8
 mov cl,1
 test al,cl
 jne 1AE69C
```

```
001AE67B
 mov esi,dword ptr ds:[esi]
 push 0
 push esi
 call 2125F0
 add esp,8
 test al,1
 jne 1AE69C
```

```
001AE68C
 mov bl,1
 mov dword ptr ds:[esi+4],0
 mov dword ptr ds:[esi+8],800
```

```
001AE69C
 movzx eax,bl
 pop esi
 pop edi
 pop ebx
 pop ebp
 ret 20
```

Hooked: CertGetCertificateChain

```
76036CCE   90                 pop
76036CCF   E9 6179178A        jmp <hook_CertGetCertificateChain>   CertGetCertificateChain
76036CD4   51                 push ecx                             hook_trampoline1_target
76036CD5   51                 push ecx
76036CD6   53                 push ebx
76036CD7   56                 push esi
76036CD8   57                 push edi
76036CD9   8B7D 08            mov edi,dword ptr ss:[ebp+8]
76036CDC   8D45 FC            lea eax,dword ptr ss:[ebp-4]
76036CDF   33DB               xor ebx,ebx
76036CE1   50                 push eax
76036CE2   897D 08            mov dword ptr ss:[ebp+8],edi
76036CE5   895D FC            mov dword ptr ss:[ebp-4],ebx
76036CE8   E8 A8000000        call crypt32.76036D95
...
76036D56   5F                 pop edi
76036D57   5E                 pop esi
76036D58   8BC3               mov eax,ebx
76036D5A   5B                 pop ebx
759A6D5B   C9                 leave
759A6D5C   C2 2000            ret 20                               end of CertGetCertificateChain
```

call crypt32.CertGetCertificateChain

```
00500FA0
mov edi,edi
push ebp
mov ebp,esp
jmp crypt32.76036CD4
```

crypt32.76036CD4

```
<hook_CertGetCertificateChain>
push ebp ; hook_CertGetCertificateChain
mov ebp,esp
push ebx
push edi
push esi
mov esi,dword ptr ss:[ebp+24]
push esi
push dword ptr ss:[ebp+20]
push dword ptr ss:[ebp+1C]
push dword ptr ss:[ebp+18]
push dword ptr ss:[ebp+14]
push dword ptr ss:[ebp+10]
push dword ptr ss:[ebp+C]
push dword ptr ss:[ebp+8]
call dword ptr ds:[2219C8] ; to_trampoline1
xor edi,edi
push edi
push eax
call 20F6C0
add esp,8
mov ebx,eax
and bl,1
push edi
push esi
call 212400
add esp,8
mov cl,1
test al,cl
jne 1AE69C
```

```
001AE67B
mov esi,dword ptr ds:[esi]
push 0
push esi
call 2125F0
add esp,8
test al,1
jne 1AE69C
```

```
001AE68C
mov bl,1
mov dword ptr ds:[esi+4],0
mov dword ptr ds:[esi+8],800
```

```
001AE69C
movzx eax,bl
pop esi
pop edi
pop ebx
pop ebp
ret 20
```

Hooked: CertGetCertificateChain

```
76036CCE    90              nop
76036CCF    E9 6179178A     jmp <hook_CertGetCertificateChain>   CertGetCertificateChain
76036CD4    51              push ecx                             hook_trampoline1_target
76036CD5    51              push ecx
76036CD6    53              push ebx
76036CD7    56              push esi
76036CD8    57              push edi
76036CD9    8B7D 08         mov edi,dword ptr ss:[ebp+8]
76036CDC    8D45 FC         lea eax,dword ptr ss:[ebp-4]
76036CDF    33DB            xor ebx,ebx
76036CE1    50              push eax
76036CE2    897D 08         mov dword ptr ss:[ebp+8],edi
76036CE5    895D FC         mov dword ptr ss:[ebp-4],ebx
76036CE8    E8 A8000000     call crypt32.76036D95
...
76036D56    5F              pop edi
76036D57    5E              pop esi
76036D58    8BC3            mov eax,ebx
76036D5A    5B              pop ebx
759A6D5B    C9              leave
759A6D5C    C2 2000         ret 20                               end of CertGetCertificateChain
```

```
call crypt32.CertGetCertificateChain
```

```
00500FA0
 mov edi,edi
 push ebp
 mov ebp,esp
 jmp crypt32.76036CD4
```

```
crypt32.76036CD4
```

```
<hook_CertGetCertificateChain>
 push ebp ; hook_CertGetCertificateChain
 mov ebp,esp
 push ebx
 push edi
 push esi
 mov esi,dword ptr ss:[ebp+24]
 push esi
 push dword ptr ss:[ebp+20]
 push dword ptr ss:[ebp+1C]
 push dword ptr ss:[ebp+18]
 push dword ptr ss:[ebp+14]
 push dword ptr ss:[ebp+10]
 push dword ptr ss:[ebp+C]
 push dword ptr ss:[ebp+8]
 call dword ptr ds:[2219C8] ; to_trampoline1
 xor edi,edi
 push edi
 push eax
 call 20F6C0
 add esp,8
 mov ebx,eax
 and bl,1
 push edi
 push esi
 call 212400
 add esp,8
 mov cl,1
 test al,cl
 jne 1AE69C
```

```
001AE67B
 mov esi,dword ptr ds:[esi]
 push 0
 push esi
 call 2125F0
 add esp,8
 test al,1
 jne 1AE69C
```

```
001AE68C
 mov bl,1
 mov dword ptr ds:[esi+4],0
 mov dword ptr ds:[esi+8],800
```

```
001AE69C
 movzx eax,bl
 pop esi
 pop edi
 pop ebx
 pop ebp
 ret 20
```

Inline Hooking: Hotpatching

- Inline hooking is officially supported
- The hotpatching support can significantly simplify the operation of setting the inline hook
- If the application is hotpatchable, then just before the prolog we can find: additional instructions:
  - MOV EDI, EDI, and 5 NOPs

```
4BDF3   5D              POP EBP
4BDF4   90              NOP
4BDF5   90              NOP
4BDF6   90              NOP
4BDF7   90              NOP
4BDF8   90              NOP
4BDF9   8BFF            MOV EDI, EDI                    support hotpatching
4BDFB   55              PUSH EBP
4BDFC   8BEC            MOV EBP, ESP
4BDFE   81EC38040000    SUB ESP, 0X438
4BE04   A18060EA77      MOV EAX, [0X77EA6080]           'N'
4BE09   33C5            XOR EAX, EBP
```

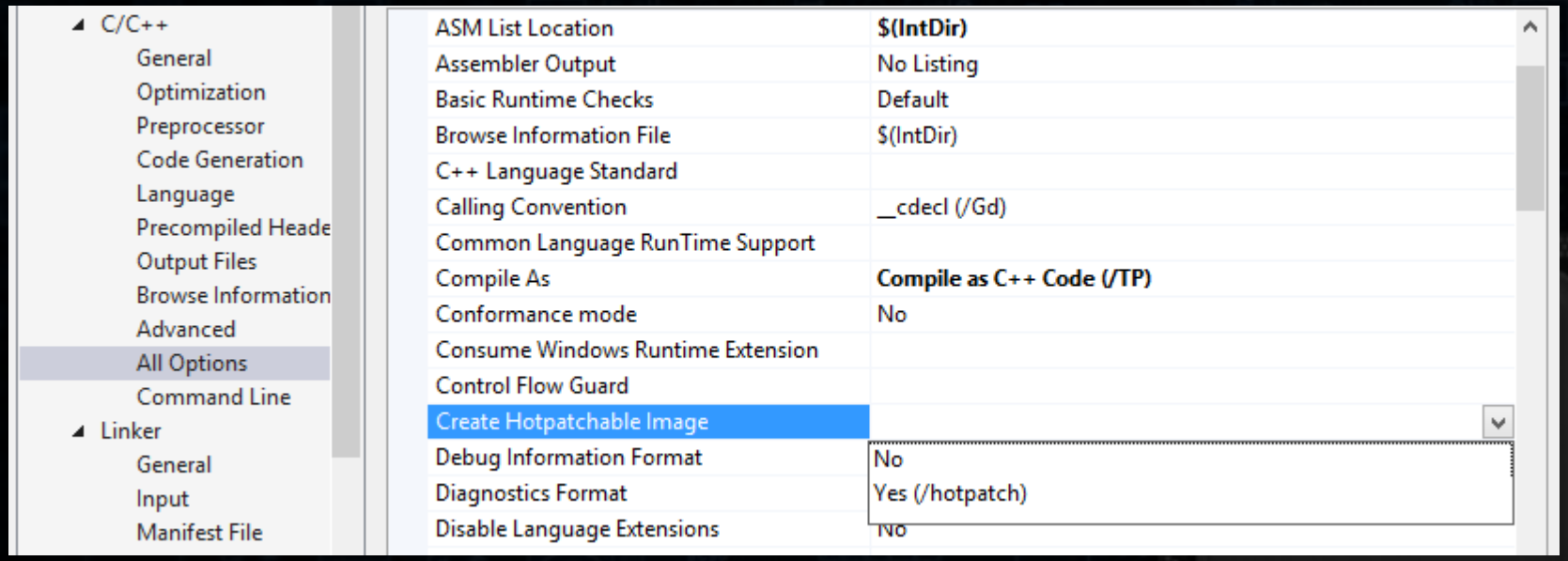
Inline Hooking: Hotpatching

- MOV EDI,EDI -> 2 BYTEs : can be filled with a short jump
- 5 NOPS -> 5 BYTEs : can be filled with a CALL

```
4BDF3   5D              POP EBP
4BDF4   E812000000      CALL 0X77E2BE0B
4BDF9   EBF9            JMP SHORT 0X77E2BDF4        patch created
4BDFB   55              PUSH EBP
4BDFC   8BEC            MOV EBP, ESP
4BDFE   81EC38400000    SUB ESP, 0X438
4BE04   A18060EA77      MOV EAX, DWORD PTR [0X77EA6080]
4BE09   33C5            XOR EAX, EBP
4BE0B   8945FC          MOV DWORD PTR [EBP - 4], EAX
4BE0E   8B4510          MOV EAX, DWORD PTR [EBP + 0X10]
4BE11   8985E0FBFFFF    MOV DWORD PTR [EBP - 0X420], EAX
4BE17   64A118000000    MOV EAX, DWORD PTR FS:[0X18]
```

Inline Hooking: Hotpatching

- Hotpatching support can be enabled in the compiler options:



Inline hooking: common steps

1. `GetProcAddress`(<function_to_be_hooked>)
2. `VirtualAlloc`: alloc executable memory for the trampoline
3. Write the trampoline: copy the beginning of the function to be hooked, and the relevant address (common opcode: `0xE9` : JMP)
4. `VirtualProtect` – make the area to be hooked writable
5. Write the hook (common opcode: `0xE9` : JMP)
6. `VirtualProtect` – set the previous access

Inline Hooking - the pros

- The hook works no matter which way the function was called
- Hook once, execute by all the modules loaded in the process

Inline Hooking - the cons

- We need to overwrite the beginning of the function, which means:
  - Parsing assembly is required (in order not to corrupt any instructions, and make a safe return)
  - Additional space must be used for the trampoline (where the original beginning of the function will be copied, allowing to call the original version of the function)
  - Making a stable hooking engine requires solving the concurrency issues: the function that we are just hooking may be called from another thread

Inline hooking detection

- Inline Hooking is detected i.e. by PE-sieve/HollowsHunter

```
Pe-sieve.exe /pid <my_pid> (detects inline hooks by default)
Hollows_hunter.exe /hooks (hook detection can be enabled by /hooks)
```

75ab0000.user32.dll.tag

```
1  ec7c;CreateWindowExW->402551[400000+2551:hooked.exe:0];5
2  1e981;CharUpperW->4017be[400000+17be:hooked.exe:0];5
3
```

Exercise 1

- The sample hooked application:
  - https://drive.google.com/file/d/1CJL4tLInbaMj-nC9Mw7BOqc9KhNZGTH1/view?usp=sharing
- Run the crackme that has both inline hooks, and IAT hooks installed
- Scan the application by PE-sieve
- Analyze the reports, and see what can we learn about the hooks

Exercise 2

- Sphinx Zbot
  - 52ca91f7e8c0ffac9ceaefef894e19b09aed662e
- This malware installs variety of inline hooks in available applications
- Scan the system with Hollows Hunter to grab the hook reports
- Examine the hooks
- Compare them with the sourcecode of the classic Zeus – find all the hooks that overlap in both