

Simple registration checks in unpacked applications potassium of ARTeam

Version 1.0 – 13th December 2005

1.	Abstract	2
2.	Where to begin, what and how to find, what to do	3
3.	References	.18
4.	Conclusions	.18
5.	Greetings	.18

Keywords

Reverse code engineering (RCE), cracking, registration checks.



1. Abstract

When constructing commercial software with the intent to make money from it, it becomes crucial to incorporate some kind of protection scheme into the product. Thus, controlling who is able to run the application and who is not. Since programmers are just as human as you and me, they often make it as simple as possible for themselves. It is very common that they use one specially designated pointer that holds the information regarding registration, eg registered = 1 (you have purchased the application) or register = 0 (you have not...). Debugging the code of a certain application and look how it flows (interactive) depending on the value of the designated pointer and conditional jumps often gives you information of what value to put into the designated pointer. In some cases, disassembly of the target exe into a static text file (deadlist) will do the trick.

The purpose with this document is simply to give some very important information to the crowd of newbies on the scene. Since most of the software available on the internet today is packed or/and encrypted, unpacking becomes a more and more important skill. However, if you lack the knowledge of the basic reversing you are guaranteed to fail in your quest.

Note:

This document is ONLY for educational purposes, if you intend to use your newly acquired knowledge for criminal acts that is **your** decision. ViceVersa Pro Build 2.0.0.9 herein called "the target" is simply an example and the theory of this document works on many other targets. Do not use this tutorial to crack this application if you intend to use it. As always, if you like an application and find it useful, you should **buy** it!



2. Where to begin, what and how to find, what to do

First of all, to get started, you need some basic knowledge of the assembler language and some terminology. This you can (if not acquired earlier) get by reading previous materials written by various authors. *The Lazy Beginning Reverser's Guide to Windows Assembly* by Vortex168 $(2002)^1$ is highly recommended since it is short, easily understood and very informative. Due to lack of time I will not provide a terminology list, the search engines on the net can provide you with that information.

Then, as any craftsman of any kind, you will need TOOLS. The more you got, the better. Recommended must-haves are: Ollydbg & Imprec with a bunch of plug-ins and of course, Lord PE, PEiD and Soft-ICE. Classics like these also comes in handy from time to time: W32Dasm, APIS32 and Hiew.

Some download links :

http://www.ollydbg.de/odbg110.zip http://www.exetools.com/ - Here you can find some candy http://www.tgrmn.com/web/downloads/vvpro.exe - ViceVersa Pro

Now you have to acquire a target to practice on, that preferable isn't packed. You can verify this (at least in most cases) using the tool PEiD. Take note of the contents in figure 2.1, PEid reports that this application is written in M\$ Visual C++ 7.0 and in this particular case, not packed.

💯 PEiD v0.93	
File: C:\Application X.exe	
Entrypoint: 00136758 EP Set	ction: .text >
File Offset: 00136758 First E	Bytes: 6A,60,68,10 >
Linker Info: 7,10 Subsy	vstem: Win32 GUI >
Microsoft Visual C++ 7.0 [Debug]	
Multi Scan Task Viewer Options	<u>A</u> bout E <u>x</u> it
🗐 Stay on top	»» ->

Figure 2.1 Checking with PEiD



Next, we start the target application and look for certain clues that might be useful to us.

🖻 ViceVersa Pro - (Unregistered)								
<u>F</u> ile ⊻iew	<u>T</u> ools <u>H</u> elp							
Source: Target:								
<u>Multiple Sou</u> <u>View Profile</u> <u>Change Pro</u>	urces/Targets <u>s Settings</u> ofile Settings	\$						
This is day 4	7 of your 30 day evaluation period.							

Figure 2.2 Mainwindow of the target application

Okay, here we have a nice lead, the string "- (Unregistered)". It sound reasonable that the string could alternatively be "- (Registered)", which is what we want to accomplish, right?

Close the target. From here we can take two alternative routes:

- 1. Disassemble the exe into a deadlist
- 2. Use an interactive debugger like Ollydbg (or others like SoftIce, IDA etc.)

Since this tutorial is at the newbie level I will present you both methods. There are some tutorials about the usage of Olly available on <u>http://tutorials.accessroot.com</u>. I recommend you to read them first and/or consult the help file included with Olly.



Method 1 – Deadlist disassembly using W32Dasm 8.93

First, make a copy of the target exe and open the copy in W32Dasm.

🗷 URSoft W32Dasm Ver 8.93 Program Disassembler/Debugger	X
Disassembler Project Debug Search Goto Execute Text Eunctions HexData Refs Help	
Disassembly of File: C:\Program\ViceVersa Pro 2\Copy of ViceVersa.exe Code Offset = 00001000, Code Size = 00188000 Data Offset = 00188000, Data Size = 00007000	
Number of Objects = 0004 (dec), Imagebase = 00400000h	
Object01: .text EVA: 00001000 Offset: 00001000 Size: 00188000 Flags: 60000020 Object02: .rdata XVA: 00189000 Offset: 00189000 Size: 0005F000 Flags: 4000040 Object03: .data KVA: 00188000 Offset: 00188000 Size: 00007000 Flags: C000040 Object04: .rsrc RVA: 00200000 Offset: 001EF000 Size: 000028000 Flags: 4000040	

There Are No Menu Resources in This Application	
Number of Dialogs = 57 (decimal) Name: DialogID_0003, # of Controls=000, Caption: "UV", ClassName: "" Name: DialogID_0022, # of Controls=000, Caption: "'8", ClassName: "" Name: DialogID_0022, # of Controls=000, Caption: "eN", ClassName: "" Name: DialogID_0022, # of Controls=000, Caption: "A", ClassName: "" Name: DialogID_0024, # of Controls=000, Caption: "A)", ClassName: "" Name: DialogID_0024, # of Controls=000, Caption: "A)", ClassName: ""	
Name: DialogID_0026, # of Control=000, Caption:'\o', ClassName:'' Name: DialogID_0026, # of Control=200, Caption:'l4', ClassName:'' 001 - ControlID:FFFF, Control Class:'' Control Text:'' 002 - ControlID:0062, Control Class:''Control Text:'' 003 - ControlID:0062, Control Class:'BUTTON' Control Text:'No confirmations (overwrite, read-only, errors, etc.)' 004 - ControlID:0000, Control Class:'BUTTON' Control Text:''Copy file permissions and security attributes (NTFS only)'	
005 - ControlID:0000, Control Class:"BUTION" Control Text:"Copy folder permissions and security attributes (NTFS only)" 006 - ControlID:0000, Control Class:"BUTION" Control Text:"Synchronize files even if timestamp changes after initial compariso 007 - ControlID:0000, Control Class:"BUTION" Control Text:"Copy directly to destination file (do not copy to temporary file)" 008 - ControlID:0000, Control Class:"BUTION" Control Text:"Try to copy files that are in use by other applications" 009 - ControlID:0000, Control Class:"BUTION" Control Text:"Try to copy files that are in use by other applications"	m
005 - Controllo SFFF, Control Class: " Control Text: " 011 - Controllo SFFF, Control Class: " Control Text: " 012 - Controllo SFFF, Control Class: " Control Text: " 013 - Controllo SFFF, Control Class: " Control Text: "	
005 - ControllD:5000, Control Class: Borrow Control Text: "Newsy verify successful copy operation by therming file Cxt 011 - ControllD:FFFF, Control Class: "Control Text: " 012 - ControllD:FFFF, Control Class: "Control Text: "(Max)" 013 - ControllD:FFFF, Control Class: "Control Text: "	
005 - ControlD.SOGO, Control Class: "Softon text: "Aways verify successful copy operation by theoring file Crc 011 - ControlD:FFFF, Control Class:" Control Text:"" 012 - ControlD:0000, Control Class:"Control Text:" 013 - ControlD:FFFF, Control Class:"Control Text:" 014 - ControlD:0000, Control Class:"COMBOBOX" Control Text:" 015 - ControlD:0000, Control Class:"COMBOBOX" Control Text:"	
<pre>005 - ControlD.SUGO, Control liss: Boilow Control Text: "Always Verify Successful copy operation by thetaring file Cat 010 - ControlD:FFFF, Control liss:" Control Text:"" 012 - ControlD:0000, Control liss:" Control Text:"" 013 - ControlD:FFFF, Control liss:" Control Text:"" 014 - ControlD:0000, Control liss:" Control Text:"" 015 - ControlD:0000, Control liss:"BUTTOW Control Text:"Plush the buffer every write"</pre>	
<pre>005 - ControlD.5000, Control liss: Bollow Control Text: "" 011 - ControlD:FFFF, Control liss: " Control Text: "" 012 - ControlD:0000, Control liss: "STATIC" Control Text: "" 013 - ControlD:FFFF, Control liss: " Control Text: "" 014 - ControlD:0000, Control liss: " Control Text: "" 015 - ControlD:0000, Control liss: "BUTTON" Control Text: "" 016 - ControlD:0000, Control liss: "BUTTON" Control Text: "Plush the buffer every write" 016 - ControlD:0000, Control liss: "BUTTON" Control Text: "Plush the buffer every write" 017 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 018 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 019 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 010 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 018 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 019 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 019 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 019 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 019 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write" 019 - ControlD:0000, Control Liss: "BUTTON" Control Text: "Plush the buffer every write"</pre>	>

Figure 2.3 W32Dasm

Press the search button and enter the text "Unregistered" and search for it.

Text {Unr	egistered} Not Fo	und	? 🗙
<u>S</u> ök efter:	Unregistered		Sök <u>n</u> ästa
<u> </u>	jemener/VERSALER	Sökriktning © <u>U</u> pp © Ne <u>d</u>	Avbryt

Figure 2.4 Searching for strings with W32Dasm.

Hmmm.. Nothing found?! That's odd. Or is it?

No, not really. Strings are usually stored as ASCII or Unicode. And W32Dasm do not always reference to strings. The string "Hey Man!" would look like this in ASCII: **48 65 79 20 4D 61 6E 21**. But if it was Unicode things it would look like this: **00 48 00 65 00 79 00 20 00 4D 00 61 00 6E 00 21**.

This difference is crucial when searching for strings, in this case the string is stored as ASCII.



Any hexeditor will do the job here. I will use Hiew.

GN ?	???????	???????	????									_ [⊐ ×
MZÉ his \$	VICE prog	VE~1.] ◆ Fam_ca £òìîï	EXE C Annot I 9	↓FRO be ru	PE Ø 0 n in [³ð⊔	0000000 DOS mod) 0- le.F	P 	2	2846720 	Hiew 6. ☺ ♬▼ ♬ ¶ू "î®	82 (c)SE o=!©©I ì¶ <mark>∭ ∵</mark> î∦	EN L=! T
×s ¶	• • • DM▲	e •	₩ ₩	PE LO Ç,	∳"´fû ► Ч	dy∭=L³ C @ ù↑ ⊨ .text	-::9 ó ж6 ▶ ► ýq1		C1 -‼ ▶ ↑ ►	Nd∭=:00= Xg‼ ►	u¶ ® Ric ! ► É ¶	hI4 N ■ † @ ► M▲ †© `.rdata	- Ɇ Τμ
s s	LFo ASC H	rward II: U ex: 59	∕Fu] nregi 5 6E	11]= istered 72 65	67 69	73 74	65 72	65 64					×
1		2	3		4	5	6		7	8	9	10	

Figure 2.5 Searching for strings with Hiew.

Open the target exe in Hiew and search for the text "Unregistered" and you will end up here.

ex ????????????????????????????????????			_ 🗆 🗙
VICEUE~1.EXE	↓FRO PE.00598F58	284672	20 Hiew 6.82 (c)SEN
.00598EA0: 64 00	00 00-54 69 6D 65-73	74 61 6D-70 20 45 7	2 d Timestamp Er
.00598EB0: 72 6F	72 00-47 65 74 44-69	73 6B 46-72 65 65 5	3 ror GetDiskFreeS
.00598EC0: 70 61	63 65-45 78 41 00-47	65 74 44-69 73 6B 4	5 paceExA GetDiskF
.00598ED0: 72 65	65 53-70 61 63 65-45	78 57 00-4B 00 45 0	3 reeSpaceExW K E
.00598EE0: 52 00	4E 00-45 00 4C 00-33	00 32 00-2E 00 44 0	0 RNEL32.D
.00598EF0: 4C 00	4C 00-00 00 00 00-5B	45 52 52-4F 52 5D Ø	<u>a l</u> l [e <u>r</u> ror]
.00598F00: 0A 00	00 00-0D 00 00 00-0D	00 0A 00-00 00 00 0	
.00598F10: 4D 00	65 00-73 00 73 00-61	00 67 00-65 00 42 0	Ø MessageB
.00598F20: 6F 00	78 00-00 00 00 00-5B	49 4E 46-4F 5D 20 0	0 o x [INF0]
.00598F30: 5B 45	52 52-4F 52 5D 20-00	00 00 00-5B 57 41 5	2 [ERROR] [WAR
.00598F40: 4E 49	4E 47-5D 20 00 00-20	2D 20 28-52 65 67 6	9 NING] - (Regi
.00598F50: 73 74	65 72-65 64 29 00-20	2D 20 28-55 6E 72 6	5 stered) - (Unre
.00598F60: 67 69	73 74-65 72 65 64-29	00 00 00-20 2D 20 21	a gistered) - *
.00598F70: 41 55	54 4F-53 59 4E 43-2A	00 00 00-5D 00 00 0	a Autosync∗ 1
.00598F80: 20 2A	00 00-20 2D 20 5B-00	00 00 00-20 3A 20 0	∂ × _−[:
.00598F90: 55 00	6E 00-6B 00 6E 00-6P	00 77 00-6E 00 20 0	<u>0</u> Unknown
.00598FA0: 45 00	72 00-72 00 6F 00-72	00 2E 00-00 00 00 0	d Error.
.00598FB0: 5C 00	56 00-69 00 63 00-65	00 56 00-65 00 72 0	d 🔪 ViceVer
.00598FC0: 73 00	61 00-20 00 50 00-72	00 6F 00-5C 00 4D 0	d sa Pro∖M
.00598FD0: 65 00	73 00-73 00 61 00-67	00 65 00-42 00 6F 0	d essageBo
.00598FE0: 78 00	NN NN-63 NN 34 NN-20	00 00 00-53 48 47 6	o x c∶∖ SHGe
.00598FF0: 74 53	70 65-63 69 61 6C-46	6F 6C 64-65 72 50 6	tSpecialFolderPa
.00599000: 74 68	41 00-53 48 47 65-74	53 70 65-63 69 61 6	C thA SHGetSpecial
MIGIODAL 2Filblk 3	GryBIK 4ReLoad 5	6 ZDirect 8X	lat y <u>10</u> Leave

Figure 2.6 Found it!

Do you recognize the string from the application mainwindow? (fig 2.2) Well, there it is, starting at offset 00598f58. This is what we will search for in W32Dasm.



So, switch back to W32Dasm. Search for the text "00598f58" and will land here :

🚾 URSoft W32Dasm Ver 8.93 Program	Disassembler/Debugger	
Disassembler Project Debug Search Goto	Execute Text Eunctions HexData Refs Help	
	I Call Ref Inc Exp Data Code Menu DLG Stra B B B B B B B B B B B B B B B B B B B	
:0045AFA5 8B4004	mov eax, dword ptr [eax+04]	~
:0045AFA8 8B7C2410	mov edi, dword ptr [esp+10]	
:0045AFAC 57	push edi	
:0045AFAD 8BC8	mov ecx, eax	
:0045AFAF E87C42FEFF	call 0043F230	
:0045AFB4 K8071BFFFF	call UU44CACU	
:0045AFB9 8400 -0045AFBD 05462400	test al, al	
-00454FBB 00402400	ine 00451RD2	
-0045AFC1 68588F5900	nush 00598858	
:0045AFC6 EBOF	jmp 0045AFD7	
	,	
* Referenced by a (U)nconditiona :0045AD67(C), :0045AD70(C), :00	l or (C)onditional Jump at Addresses: 45AEA3(C), :0045AEAC(C)	
:0045AFC8 6857000780	push 80070057	
:0045AFCD E82E61FAFF	call 00401100	
* Referenced by a (U)nconditiona :0045AFBF(C) 	l or (C)onditional Jump at Address:	
:0045AFD2 68488F5900	push 00598F48	
* Referenced by a (U)nconditiona :0045AFC6(U) 	l or (C)onditional Jump at Address:	
:0045AFD7 E86491FAFF	call 00404140	
:0045AFDC 8D442408	lea eax, dword ptr [esp+08]	
:0045AFE0 50	push eax	
:0045AFE1 B924005F00	mov ecx, 005F0024	
:0045AFE6 E8656FFAFF	call 00401F50	
:0045AFEB 8D4C2410	lea ecx, dword ptr [esp+10]	
:0045AFEF 51	push ecx	
:0045AFF0 B928005F00	mov ecx, 005F0028	
:0045AFF5 K8566FFAFF	call 00401F50	
-0045AFFA 88742408	mov esi, dword ptr (esp+08)	
-00458002 56	much eci	
:0045B003 K8D9CC0F00	call 00557CE1	
:0045B008 8D47F0	lea eax, dword ptr [edi-10]	
:0045B00B C644242400	mov [esp+24], 00	~
<		>
Line:151260 Pg 1759 and 1760 of 8322. Code I	Data @:0045AFC1 @Offset 0005AFC1h in File:C:\Program\ViceVersa Pro 2\Copy of ViceVersa.exe	

Figure 2.7 Searching for text pointer"00598f58"

This is also the one and ONLY occurrence of the instruction *push 00598f58* and therefore this place is the place of interest. See the rows a little bit up ? A call to 0044CAC0, test al, al and then jne 0045AFD2.

This code tells me that (probably) the value maintained in al (the low byte of eax) controls whether the text "- (Unregistered)" or "- (Registered)" should be displayed in the window caption. Check your Hiew 16 (\$10) bytes before the string "- (Unregistered)" and you will see the string "- (Registered)". The OP code 75 does not mean 'jump if not equal' as W32Dasm says, in fact it means 'jump if not zero'². So, if al is non zero the cpu jumps to 0045AFD2! Lets assume that the preceding call to 0044CAC0 stores or returns something interesting in the al register.

Mark the row at 0045AFB4 and press the "Call" button located in the toolbar.



Then you will end up here:

🚾 URSoft W32Dasm Ver 8.93 Progr	am Disassembler/Debugger	
Disassembler Project Debug Search Go	oto <u>E</u> xecute Text Eunctions Hex <u>D</u> ata <u>R</u> efs <u>H</u> elp	
	Ret Call Ret Inc. Exp. Data Code Manu DLG Str.	
		<u>^</u>
:0044CABC CC	int 03	
:0044CABD CC	int 03	
:0044CABE CC	int 03	
:0044CABF CC	int 03	
* Referenced by a CALL at Addr	esses:	
:0044051F , :004405B3 , :	004407D9 ,:0044552F ,:004476A1	-
:0044F686 ,:0045094C ,:	00450A8E , :00450AAE , :00450E03	
1:00450EA0 , :0045AFB4		
:0044CACO AOD4FB5E00	mov al, byte ptr [005EFED4]	
:0044CAC5 C3	ret	
:0044CAC6 CC	int 03	
:0044CAC7 CC	int 03	
:0044CAC8 CC	int 03	
:0044CAC9 CC	int 03	
:0044CACA CC	int 03	
:0044CACB CC	int 03	
:0044CACC CC	int 03	
:0044CACD CC	int 03	
:0044CACE CC	int 03	
:0044CACF CC	int 03	
* Referenced by a (U)nconditio	nal or (C)onditional Jump at Address:	
:0044CC54(U)		
1		
:0044CAD0 56	push esi	
:0044CAD1 57	push edi	
:0044CAD2 887C240C	mov edi, dword ptr (esp+ut)	
-0044CAD6 6BF1 -0044CAD8 9806	mov est, ecx	
-0044CAD0 0000	mov eax, dword per (esr) mov eax dword ntr [esr-10]	
-0044CADD 8B11	mov edx dword per [eax 10]	
:0044CADF 83E810	sub eax, 00000010	
:0044CAE2 6A01	push 0000001	
:0044CAE4 57	push edi	1
:0044CAE5 50	push eax	×
		>
Line:125635 Pg 1461 and 1462 of 8322 Co	de Data @:0044CAC0 @Offset 0004CAC0h in File:C:\Program\ViceVersa Pro 2\Copy of ViceVersa.exe	

Figure 2.8 Finding the pointer [005EFBD4]

Ahaa! The byte value stored in the pointer [005EFBD4] is moved into al. This is it!!

Now we have to find the location in the code where the registration info is verified and the non zero value is put into [005EFBD4]. Since this is probably accomplished by moving a value into the pointer we will now search for "[005EFBD4]," Also this one occurs only once in this particular target. But one should always keep an open mind; programmers are sometime sneaky and puts in several registration checks. ;-)



This is what you will find:

🚾 URSoft W32Dasm Ver 8.93 Program	Disassembler/Debugger	
Disassembler Project Debug Search Goto	Execute Text Functions HexData Refs Help	
	Call Ret Imp Exp Data Code Menu DLS Stra	
:0044F2DF 50	push eax	<u>^</u>
:0044F2E0 E81BE9FFFF	call 0044DC00	
:0044F2E5 8BCC	mov ecx, esp	
:0044F2E7 8964240C	mov dword ptr [esp+0C], esp	
:0044FZKB 51	push ecx	
-0044F2KC C74424IC00000000 -0044F2K4 F0F7F5FFF	mov [esp+ic], 00000000	
-0044F2F4 B8B/BDFFFF -0044F2F9 83C404	edd esn 00000004	
:0044F2FC C7442418FFFFFFFF	mov [esp+18]. FFFFFFF	_
:0044F304 E887FDFFFF	call 0044F090	
:0044F309 8B4C2410	mov ecx, dword ptr [esp+10]	
:0044F30D A2D4FB5E00	mov byte ptr [005EFBD4], al	
:0044F312 64890D0000000	mov dword ptr fs:[00000000], ecx	
:0044F319 83C41C	add esp, 0000001C	
:0044F31C C3	ret	
-0044701D CC	int 00	
-0044F31D CC	int 03	
-0044F31F CC	int 03	
* Referenced by a CALL at Address	:	
:0044F440		
1		
:0044F320 B80C100000	mov eax, 0000100C	
:0044F325 E8768B0E00	call 00537EA0	
:0044F32A A188D25E00	mov eax, dword ptr [005ED288]	
:0044F32F 89842408100000	mov dword ptr [esp+00001008], eax	
:0044F336 6800080000	push 00000800	
:0044F33B 8D442408	lea eax, dword ptr [esp+08]	
-0044F33F 50 -0044F340 C744240900000000	push eax	
-0044F348 FF1558F65F00	call dword ntr [005KK658]	
:0044F34E 85C0	test eax. eax	
:0044F350 7515	jne 0044F367	
* Referenced by a (U)nconditional	or (C)onditional Jump at Address:	
:0044F377(C)		
:0044F352 33C0	xor eax, eax	
:0044F354 8B8C2408100000	mov ecx, dword ptr [esp+00001008]	×
<u>S</u>)		>
Line:130786 Pg 1521 and 1522 of 8322 Code D	ata @:0044F304 @Offset 0004F304h in File:C:\Program\ViceVersa Pro 2\Copy of ViceVersa.exe	

Figure 2.9 Locating the registered check call

There you are! Follow the call at 0044F304 (call 0044F090). Now scroll down a bit. A call can return to the caller or jump on to another subroutine, in this case the call is ended by a RET without a preceding push and thus returning to the caller.



See that line at 0044F132 (xor al, al) ? An eXclusive OR (XOR) is made with al on al. If a value is XOR:ed with it self it is **always** zero. Hence, this is leaving al = 0! Changing this to '*mov al*, 01' is a suitable solution since it also is 2 OP codes long (B0 01).

💯 URSoft W32Dasm Ver 8.93 Program	Disassembler/Debugger	
Disassembler Project Debug Search Goto	Execute Text Eunctions HexData Refs Help	
C	Call Ret Imp Exp Data Code Men DLG Str.	
:0044FOFC 83CAFF	or edx, FFFFFFFF	<u>~</u>
:0044FOFF FO	lock	
:0044F100 0FC111	xadd dword ptr [ecx], edx	
:0044F103 4A	dec edx	
:0044F104 85D2	test edx, edx	
-0044F106 /F08 -0044F100 0D00	jg 00447110 men egy dverd ptr (eey)	
-0044F108 8B11	mov edv. dword ptr [eav]	
-0044F10C 50	much eax	
:0044F10D FF5204	call [edx+04]	
* Referenced by a (U)nconditional :0044F106(C) -0044F110 SD47F0	or (C)onditional Jump at Address:	
-0044F110 6D47F0 -0044F113 C744243CFFFFFFFFFFF	mor (esp+3C) REFERE	
:0044F11B 8D480C	lea ecx. dword ptr [eax+0C]	
:0044F11E 83CAFF	or edx. FFFFFFFF	
:0044F121 F0	lock	
:0044F122 OFC111	xadd dword ptr [ecx], edx	
:0044F125 4A	dec edx	
:0044F126 85D2	test edx, edx	
:0044F128 7F08	jg 0044F132	
:0044F12A 8B08	mov ecx, dword ptr [eax]	
:0044F12C 8B11	mov edx, dword ptr [ecx]	
:0044F12E 50	push eax	
:0044F12F FF5204	call [edx+04]	
* Referenced by a (U)nconditional :0044F128(C) 	or (C)onditional Jump at Address:	
:0044F132 32C0	xor al, al	
:0044F134 8B4C2434	mov ecx, dword ptr [esp+34]	
:0044F138 64890D00000000	mov dword ptr fs:[00000000], ecx	
:0044F13F 5F	pop edi	
:0044F140 5E	pop esi	
:0044F141 83C438	add esp, 00000038	
:0044F144 C3	ret	
<		>
Line:130560 Pg 1519 of 8322 Code Data @:004	4F10D @Offset 0004F10Dh in File:C:\Program\ViceVersa Pro 2\Copy of ViceVersa.exe	
E: 210 P : (1 : (

Figure 2.10 Browsing the registered check call



EX ???????????? ????????????????????????				- 🗆 ×
UICEUE~1.EXE ↓FRO	PE.0044F132 a32		- 2846720 Hiev	/ 6.82 (c)SEN
.0044F132: 32C0		xor	al,al	
.0044F134: 8B4C2434		mov	ecx,[esp][34]	1
.0044F138: 64890D000000	00	mov	fs:[00000000]	,ecx
.0044F13F: 5F		pop	edi	
.0044F140: 5E		pop	esi	
.0044F141: 83C438		add	esp,038 ;"8"	
.0044F144: C3		retn		
.0044F145: 6A01		քացի	001	
.0044F147: 8D4C244C		lea	ecx,[esp][4C]	
.0044F14B: E8F02AFBFF		call	.000401C40	↑ (1)
.0044F150: 50		քսոհ	eax	
.0044F151: 8D44241C		lea	eax,[esp][1C]	1
.0044F155: 50		քսչի	eax	
.0044F156: E8A5E9FFFF		call	.00044DB00	1 (2)
.0044F15B: 83C408		add	esp,008 ;" <mark>e</mark> "	
.0044F15E: 83781810		cmp	d,[eax][18],	310 ;"►"
.0044F162: C644243C02		mov	b,[esp][3C],[162 ; "O"
.0044F167: 7205		jb	.00044F16E	
.0044F169: 8B4004		mov	eax,[eax][04]	
.0044F16C: EB03		jmps	.00044F171	
.0044F16E: 83C004		add	eax,004 ;"•"	
.0044F171: 50		push	eax	
.0044F172: 8D4C2414		lea	ecx, [esp][14.	
MIGlobal 2FilBlk 3	4ReLoad 50rd0ff	61byte	Direct SXlat	Auto 10Leave

Figure 2.11 Locating the registered check call in Hiew

Lets find this place in Hiew. Open the original target exe press F5 and enter the location 0044F132.

ex ????????????????????????????????????					- 🗆 ×
UICEUE~1.EXE ↓FWO	PE 0004	F134 a32 <ed< td=""><td>litor></td><td>2846720 Hiew 6</td><td>.82 (c)SEN</td></ed<>	litor>	2846720 Hiew 6	.82 (c)SEN
0004F132: B001		mov		al,001 ;"©"	
■ 0004F134: 8B4C2434		mov		ecx,[esp][34]	
📓 0004F138: 64890D0000000	0	mov		fs:[00000000],e	CX
0004F13F: 5F		pop		edi	
0004F140: 5E		pop		esi	
0004F141: 83C438		add		esp, <mark>038</mark> ;"8"	
0004F144: C3		retr			
0004F145: 6A01		pusł	1	001	
◎ 0004F147: 8D4C244C		lea		ecx,[esp][4C]	
0004F14B: E8F02AFBFF		call		000001C40	
📓 0004F150: 50		pusl	1	eax	
0004F151: 8D44241C		lea		eax,[esp][1C]	
0004F155: 50		pusl	1	eax	
0004F156: E8A5E9FFFF		call		00004DB00 _	
0004F15B: 83C408		add		esp,008 ;" <mark>-</mark> "	
📓 0004F15E: 83781810		cmp		d,[eax][18],010	-;* ™ **
0004F162: C644243C02		mov		b,[esp][3C],002	;"8"
0004F167: 7205		jb		00004F16E	
0004F169: 8B4004		mov		eax,[eax][04]	
₩ 0004F16C: EB03		jmps		00004F171	
0004F16E: 83C004		add		eax,004 ;"+"	
0004F171: 50		pusl	1	eax	
0004F172: 8D4C2414		lea		ecx,[esp][14]	
1 2 3	4	5 6	7	8 9	10

Figure 2.12 Patching the application in Hiew

Then press F3 to edit, B0 01. Press F9 to save the changes. F10 to quit.

Run the target application, and.... Look! It's registered!

🖻 ViceVe	rsa Pro - (Registered)	
<u>F</u> ile ⊻iew	<u>I</u> ools <u>H</u> elp	
Source: Target:		·
Multiple Sou View Profile Change Pro	urces/Targets <u>Settings</u> <u>Settings</u> Compare	¥

Figure 2.13 Task complete!



Method 2 – Interactive debugger (Ollydbg)

Launch Olly and open the target application.

C CPU - main thread, module ViceVers						
Liber 2012 PUSH 04 PUSH 01	Backup Copy Copy Copy Copy Copy Copy Copy Cop	Registers (M1X) EAX 0000000 ECX 0012FFB0 ECX 0012FFB0 EDX 7FFDF000 EDX 7FFDF000 EDX 7FFDF000 EDX 001638E EDI 00036758 ViceVers.Modu C 0 ES 0023 32bit 0(FFFFFFF F 1 CS 0018 32bit 0(FFFFFFF F 1 CS 0018 32bit 0(FFFFFFF F 1 CS 0018 32bit 0(FFFFFFF G 0 FS 0038 32bit 0(FFFFFFF G 0 FS 0038 32bit 0(FFFFFFF C 0 FS 0038 32bit 0(FFFFFFF F 1 CS 0018 32bit 0(FFFFFFF F 1 CS 0018 32bit 0(FFFFFFF G 0 FS 0038 32bit 0(FFFFFFF G 0 FS 0038 32bit 0(FFFFFFF G 0 FS 0038 32bit 0(FFFFFFF F 0 0000246 (NO,NB,E,BE,NS, M10 07705 E010 0012 D030 FL 00000 0477 0000 0000 M1X 0040 6070 0012 D030 M1X 0040 6070 0012 D030 M1X 0040 6000 0000 0000 M1X 0040 A74B 0012 D020 M14 0000 0000 0000 0000 M14 0000 0000 0000 0000				
Bessize Imp Shight Uievers Bessize 005367F6 > 8389 84000000 CHP DwORD PTR DS: EECX+843,0E Address Hex dump DwORD PTR DS: EECX+843,0E 0055807F8 > 8389 84000000 CHP DwORD PTR DS: EECX+843,0E 00558070 00 00 00 00 00 C1 D 00558070 00 00 00 00 C1 D S 00 00558070 00 00 00 C1 D S 00 S 00 S 00558040 00 00 00 C1 D S 00 S 00 D D S S 00 S 00 S S 00 S S S S S S S S S S S S S S S S S S S <t< td=""><td>Search for Name (does) in current model Find references to Name in all modules View Command Copy to executable Sequence of commands Analysis Constant Bookmark Binary string Dump debugged process All intermodular calls</td><td>Ctrl+F Ctrl+S Ctrl+B</td></t<>	Search for Name (does) in current model Find references to Name in all modules View Command Copy to executable Sequence of commands Analysis Constant Bookmark Binary string Dump debugged process All intermodular calls	Ctrl+F Ctrl+S Ctrl+B				
06C22000 08062000 Teerapoo : idat. 06C2000 08062000 Teerapoo : edat. 06C2000 08062000 Teerapoo : rsrc 5D581000 08070000 CONCTL32 : text 5D624000 08081000 CONCTL32 : rsrc 5D624000 08081000 CONCTL32 : rsrc SF060000 08081000 SF06000 08081000 SF950000 98081000 SF050000 : rsrc SF060000 98081000 : rsrc : rslo	Generate XRefs From Graph Generate XRefs To Graph Generate Call Stack Graph All commands All sequences All constants All switches All switches All switches All referenced text strings Vertice Rule Priv Rule	E Ex Base				

Figure 2.14 Locating strings in Olly

Start with a 'Search for .. All referenced text strings'

Move the marking to the top of the line and right click and choose 'Search for text'. Enter the text "- (Unregistered)", if not sure of upper/lower case untick the Case sensitive box.



R Text strings referenced in ViceVers:.text]
Address Disassemblu	Text string	
99492D02 PUSH Ilications 995999D9	UNICODE "Invisit d DateTime"	
00402545 PUSH Ilications 00509000	UNICODE "Invalid DateTime"	
00402400 PUSH Higoland 00509000		2
00403408 PUSH Licellers 005890F0	UNICODE "Settings"	
0040390F MOLL FOX Ilicellers 00599034	UNICODE "Include"	
00403816 MOULEDX Hicelers 00589024	UNICODE "Evolude"	
00403B68 PUSH LiceVers, 00589D44	ASCIL "Select filter to remove from list."	
00403D02 PUSH UiceVers,00589D68	ASCII "Select filter to copy."	
00403DC7 PUSH WiceVers 00589D80	ASCII "Select filter to move up."	
00403E87 PUSH ViceVers,00589D9C	ASCIL "Select filter to move down."	
00403EE3 PUSH ViceVers,00589E14	UNICODE "Test File (*.tst)(*.tst)!"	
00403EE2 PUSH ViceVers,00589E0C	UNICODE "tst"	
00403E23 MOU DWORD PTR DS:[E0X+301.ViceVers.0058	UNICODE "Select text file to load"	
00403E91 PUSH ViceVers,00589DB8	ASCIL "Error loading values from file."	
00404013 PUSH ViceVers,00589E14	UNICODE "Test File (*.tst)(*.tst)!"	
0040401F PUSH ViceVers.00589E0C		
00404050 MOV DWORD PTR DS: [EAX+] Enter tout to an	arch for	
004040BE PUSH ViceVers.00589E48 Linter text to Se		
004042D9 PUSH ViceVers.00589EC4		
004043D7 MOU EAX.ViceVers.00589		
004043DE MOU EAX. ViceVers. 005891 Unregistered	T	
0040446B PUSH ViceVers.00589E9C		
00404519 PUSH ViceVers.00589ECC		
00404751 PUSH ViceVers.00589F08		
0040479A MOU DWORD PTR SS: [ESP+: Lase sensitive		
004047D5 MOV DWORD PTR SS:[ESP+]		
004047FD MOV DWORD PTR SS: [ESP+] C Entire second		
0040481A PUSH ViceVers.00589D04		
0040481F PUSH ViceVers.00589CF0	OK Cancel	
0040691A PUSH ViceVers.0058A2C0	OK Califer	
00406935 PUSH ViceVers.0058A2A8		
0040693A PUSH ViceVers.00589CF0	UNICODE "Settings"	
004069C2 PUSH ViceVers.0058A2A8	UNICODE "archviewdlg"	
004069C7 PUSH ViceVers.00589CF0	UNICODE "Settings"	
00406A61 PUSH ViceVers.0058A2C0	UNICODE "%4,%4,%4,%4,%4,%4"	
00406AB4 PUSH ViceVers.0058A2E4	UNICODE "arcfilelist"	
00406AB9 PUSH ViceVers.00589CF0	UNICODE "Settings"	
00406C00 PUSH ViceVers.0058A2FC	UNICODE " -"	
00407838 MOV DWORD PTR SS:[ESP+3C],ViceVers.0058	UNICODE "properties"	
004078E9 PUSH ViceVers.0058A320	ASCII "::/archiveviewer.htm"	
00407F4B PUSH ViceVers.0058A3E8	UNICODE " "	
00407FA2 MOV DWORD PTR SS:[ESP+34],ViceVers.0058	UNICODE "Orig. Name"	
00407FCA MOV DWORD PTR SS:[ESP+34],ViceVers.0058	UNICODE "Orig. Timestamp"	
00407FE7 MOV DWORD PTR SS:[ESP+28],ViceVers.0058	UNICODE "Archive Status"	
0040801A MOV DWORD PTR SS:[ESP+34],ViceVers.0058	UNICODE "When"	
00408042 MOU DWORD PTR SS:[ESP+34],ViceVers.0058	UNICODE "Size on Disk"	
0040806F MOU DWORD PTR SS:[ESP+34],ViceVers.0058	UNICODE "File Name on Disk"	
00408084 PUSH ViceVers.005882E4	UNICUDE "arc+itelist" and	

Figure 2.15 Locating strings in Olly

Press OK and you land here:

R Text strings referenced in ViceVers:.text			
Address Disassembly	Text string		
0045829D PUSH ViceVers.00598F30	ASCII "[ERROR] "		
0045A46D PUSH ViceVers.00598F3C	ASCII "[WARNING] "		
0045A63A PUSH ViceVers.00598F10	UNICODE "MessageBox"		
0045A6D0 PUSH ViceVers.00598F10	UNICODE "MessageBox"		
0045HD79 PUSH ViceVers.00598F84	HSUII " - L"		
0045HEIZ FUSH VICEVERS.00596F80			
00458E4E PUSH ViceVers,00598E80	ASCII " *"		
0045AF92 PUSH ViceVers.00598F6C	ASCII " - *AUTOSYNC*"		
0045AFC1 PUSH ViceVers.00598F58	ASCII " - (Unregistered)"		
0045AFD2 PUSH ViceVers.00598F48	ASCII " - (Registered)"	Follow in Disassembler	Enter
0045B35H PUSH ViceVers.00598F90	UNICUDE "Unknown Error."		
00455368 FUSH ViceVers.00578F86	HSUII : UNICODE "cofturnes"	Counch for book	
0045B498 PUSH ViceVers,00594620	UNICODE "\ViceVersa Pro\MessageF	Search for text	
0045859E PUSH ViceVers.0059901C	UNICODE "shell32.dll"	Search next	Ctrl+L
0045B5C8 PUSH ViceVers.00599004	ASCII "SHGetSpecialFolderPathW"		
0045B5D4 PUSH ViceVers.00598FEC	ASCII "SHGetSpecialFolderPathA"	To and a local state	50
00458600 PUSH ViceVers.00598FE4	UNICODE "c:\"	l oggle breakpoint	F2
004587E3 PUSH ViceVers,00599064	ASCIL "SHGetFolderPathw"	Conditional breakpoint	Shift+F2
0045B803 PUSH ViceVers.00599050	ASCII "SHGetFolderPathA"		
0045B81A PUSH ViceVers.00599034	UNICODE "SHFolder.dll"	Conditional log breakpoint	Shift+F4
0045B82B PUSH ViceVers.00599064	ASCII "SHGetFolderPathW" -	het the set is the first of the	
0045B837 PUSH ViceVers.00599050	ASCII "SHGetFolderPathA"	Set breakpoint on every command	
0045B7FH FUSH VICEVERS.00576FE4	UNICODE "NN"		
0045C3B4 PUSH ViceVers.00598CFC	UNICODE "NN"	 Set log breakpoint on every command 	
0045DC09 PUSH ViceVers.0059907C	ASCII "dll"		
0045DE93 PUSH ViceVers.0059908C	ASCII "exe"	Copy to clipboard	•
0045DEB9 PUSH ViceVers.00599088	ASCII "ico"	Copy to clipboard	
0045DED5 PUSH ViceVers.00599084	HSCII "Ink"	Appearance	•
AGAEESAD DUSU HigoNove AGEGGEDO	HSUIT "CUP"		
0045E324 PUSH ViceVers,00599588	UNICODE "hiddenautoesec"		
0045E334 PUSH ViceVers.00599598	UNICODE "dialogautosync"		
0045E344 PUSH ViceVers.00599578	UNICODE "dialogautoexec"		
0045E354 PUSH ViceVers.00599564	UNICODE "autoclose"		
0045E364 PUSH_ViceVers.0059954C	UNICODE "autocompare"		
0045E30D FUSH Highland 00E99E24	UNICODE "autosync"		
0045E3AB PUSH ViceVers.0059620C	UNICODE "settings"		
0045E3EC PUSH ViceVers.00599510	UNICODE "priority:"		
0045E436 PUSH ViceVers.00599504	UNICODE "wait"		
0045E45D PUSH ViceVers.005994F4	UNICODE "source:"		
0045E4H9 PU5H VICEVers.005994E4 004EE4EE DUSH UiceVers.00E994D4	UNICODE "incoub"		
0045E523 PUSH ViceVers.00599404	UNICODE "comptype:"		~
0045E525 FUSH VICEVERS.00599400	UNICODE "COMPTYPE:"		

Figure 2.16 Following strings into code in Olly

When you have found the string right-click and chose 'Follow in Disassembler'.



Recognize this from earlier?

C CPU -	main thread, m	odule ViceVers		
04454792 04454745 04454745 04454745 04454745 04454745 04454745 04454745 04454745 04454745 0445475 04	- 68 6C8F5900 - 804C24 9C - 81 9091FHFF - 88 9091FHFF - 88 97C24 10 - 885C8 - 88 7C24 10 - 885C8 - 88 7C42FEFF - 84C0 - 88 97C34 08 - 75 11 - 68 53875900 - 68 53875900 - 68 4855900 - 88 2661FHFF - 88 924005F00 - 88 92424 08 - 887424 08 - 887444 08 - 887444 08 - 887444 08 - 887444 08 - 887444 08 - 887444 08 -	PUSH UiceUers.00598F6C LEA ECK.000RD PTR SS:[ESP+C] CALL UiceUers.00404140 CALL UiceUers.00459510 MOV EDX.000RD PTR DS:[ENX+4] MOV EDX.000RD PTR DS:[ESP+10] PUSH EDI MOV ECX.EAX CALL UiceUers.0043F230 CALL UiceUers.0043F230 CALL UiceUers.0043F230 CALL UiceUers.0043FF02 PUSH UiceUers.0045AFD2 PUSH UiceUers.00459F48 CALL UiceUers.00459F48 CALL UiceUers.00459F48 CALL UiceUers.00459F50 LEA ECX.000RD PTR SS:[ESP+8] HOS ECX.UiceUers.005F024 MOU ECX.UiceUers.005F028 CALL UiceUers.00557CE1 LEA ECX.000RD PTR DS:[ED1-10] MOU ECX.UiceUers.004557CE1 LEA ECX.000RD PTR DS:[ED1-10] MOU ECX.UiceUERS.00557CE1 LEA ECX.000RD PTR DS:[ED1-10] MOU DWORD PTR SS:[ESP+24],-1 0	ASCII " - *AUTOSYNC*"	Registers (NMX) EAX 00000000 EAX 00000000 EDX 7C500E94 ntdll.KiFastSys ESY 0012FFC4 EBX 0012FFC4 EDI 00000000 EIN 0000000 EIN 00000000 EIN 00000000 EIN 00000000 EIN 00000000 EIN 00000000 EIN 00000000 EIN 000000000 EFL 000000246 MM0 7703 E010 0012 D08C MM1 00000 00000 00400 00000 00400 MM1 00000 00000 00400 00000 00400 MM1 00000 00000 00400 00000 MM1 00000 00000 00400 00000 MM1 00000 00000 00400 00000 MM1 000000 00400 00000 00400 MM1 00000 00000 00400 00000 MM2 00000 00400 00000 00400 MM3 0044 6C77 0012 D08C MM1 00000 00000 00400 00000 MM1 00000 00000 00400 00000 MM2 00400 00000 00400 00000 MM3 0044 6C77 0012 D08C MM1 00000 00000 00000 00000
HODRESS 005E8000 005E8010 005E8020 005E8030	nex dump 00 00 00 00 C3 D 8D 3B 55 00 A2 7 A7 BB 56 00 C0 B FC 7B 56 00 C0 7	1 53 00 5A 95 56 00 66 95 56 00 1 8 58 00 C4 78 58 00 DA 78 58 00 1 0 56 00 35 C9 56 00 66 78 58 00 1 0 56 00 35 C9 56 00 66 78 58 00 1 0 612 0 58 00 30 78 58 00 59 70 58 00 1 0 612	FFC8 00000000 FFCC 0016038E FFC0 7FD7000 FFD4 30548038 FFD4 30548038	
005E8050	F0 70 58 00 10 7		PFDC 81857280	<u>×</u>

Figure 2.17 Finding the call to the pointer in Olly

Put a breakpoint (red) on the call to 0044CAC0 by marking the line and pressing F2, run the application by pressing F9. Press 'Ok, Evaluate' and Olly will break on the breakpoint, here:

CPU - main thread, mo	dule ViceVers	
0044CAC0 r\$ A0 D4FB5E00 N	10V AL,BYTE PTR DS:[5EFBD4]	Registers (FPU
00444CRC5 L. C3 F 0044CRC6 CC 1 0044CRC7 CC 1 0044CRC8 CC 1 0044CRC9 CC 1	ETN INT3 INT3 INT3 INT3 INT3 INT3 INT3	EAX 00000001 ECX 00000800 EDX 7C90EB94 n EBX 00000000 ESP 00128014 EBP 00120F34 EST 77D3C6BC U EDI 00F2C2D0 U
0044CHCU CC [] 0044CACE CC []	INTS	EIP 0044CAC0 U
0044CACF CC 1 0044CAD1 > 56 F 0044CAD1 - 57 0044CAD2 - 887C24 0C 1 0044CAD3 - 88F1 1 0044CAD3 - 8806 1 0044CAD3 - 8848 F0 1 0044CAD1 - 8811 1	INTS USH ESI UVSH EDI 10V EDI,DWORD PTR SS:[ESP+C] 10V ESI,ECX 10V EAX,DWORD PTR DS:[ESI] 10V ECX,DWORD PTR DS:[EAX-10] 10V ECX,DWORD PTR DS:[ECX]	C 0 ES 0023 3 P 1 CS 001B 3 A 0 SS 0023 3 Z 1 DS 0023 3 S 0 FS 003B 3 T 0 GS 0000 N D 0
0044CADF . 83E8 10	SUB EAX, 10	EEL 00000246 (
0044CRE4 . 57 0044CRE5 . 50		ST0 empty 8.06 ST1 empty -6.3
DS:[005EFBD4]=00 AL=01 Local calls from 0044051F,	004405B3, 004407D9, 0044552F, 004476A1, 0044F686, 0045094C, 00450A8E, 004	ST2 empty -UNO ST3 empty 1.10 5 ST4 empty 9.59 ST5 empty 1.00
Address Hex dump	0012B014 0045AFB9 RETURN to ViceVers.0045AFB9 from 0012B014 0045AFB9 RETURN to ViceVers.0045AFB9 from	ViceVers.0044 🔨
005E8000 00 00 00 00 00 C3 D1 005E8010 8D 3B 55 00 A2 7B 005E8020 A7 BB 56 00 C0 BD 005E8030 FC 7B 58 00 20 70	53 00 54 55 00 56 00 258 00 77D3C6EC USER32.RedrawWindow 58 00 C4 78 58 00 100 128010 00128000 00128010 00128010	v

Figure 2.18 Pointer Found



	Backup Copy Binary Assemble Label Comment Breakpoint Hit trace Run trace	> Space ; ; >		
	Thread	•		
	Follow in Dump View call tree	Ctrl+K	Selection Memory address	
	Search for Find references to View Copy to executable Analysis	+ + + + +		
	Bookmark Dump debugged process Generate Function Flowchart Generate XRefs From Graph Generate XRefs To Graph Generate Call Stack Graph	•		
CPU - main thread, module ViceVers	Appearance	•		
004440ACS C3 RETN 00444CACS CC INT3 00444CACS CC INT3 00444CACS CC INT3 00444CACS CC INT3 0044CACS CC INT3 0044CACB S6 PUSH ESI 0044CAD0 56 PUSH EDI 0044CAD1 57 PUSH EDI 0044CAD2 88F1 MOV ESI,ECX 0044CAD3 8866 MOV ECX,DWORD PT 0044CAD4 8848 F0 MOV ECX,DWORD PT 0044CAD5 83E3 10 SUB EAX,10 0044CAD7 83E3 10	DS: [5EFBD4] R SS: [ESP+C] R DS: [ESI] R DS: [EAX-10] R DS: [ECX] P Command	line		Registers (FPU EAX 00000001 ECX 00000001 ECX 000000001 EDX 7C90EB94 n ESP 0012D014 ESP 0012D034 ESP 0012D04 ESP 003B 3 C 0 ESP 003B 3 S 0 FS 003B 3 T 0 GS 0000 N D 0 O 0 LastErr E
0044CAE5 50 PUSH EAX DS:[005EFBD4]=00		une		
HL-01 Coal calls from 0044051F, 004405B3, 00440 Address Hex dump 005EFBD4 00 00 00 50 60 00 <th>209,0044552f 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280</th> <th></th> <th></th> <th>×</th>	209,0044552f 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280 00 001280			×

Figure 2.19 Locating the pointer in memory

Right click on the line 0044CAC0 and chose 'Follow in dump.. Memory address'.



C CPU - mi	Backup Copy Binary) 				
2044CRC5 0044CRC5 0044CRC6 0044CRC7 0044CRC9 0044CRC9 0044CRC8 0044CRC8 0044CRC8 0044CRC8 0044CRC8	Label : Breakpoint Search for Find references C View executable file) Itrl+R	ESEFBD43 Memory, on access Memory, on write Hardware, on access Hardware, on write	Byte Word		Registers (FPU) EAX 00000001 ECX 00008BBD EDX 7C90EB94 nt ESX 0000000 ESP 001228014 EBN 0012DF34 ESI 77D3C6EC US EDI 0022C140 UN
0044CACE 0044CAD0 > 0044CAD0 > 0044CAD1 . 0044CAD2 . 0044CAD6 . 0044CAD6 . 0044CAD6 . 0044CAD6 . 0044CAD6 . 0044CAD7 .	Copy to executable file Go to V Hex Text Short	> > >	Hardware, on execution SS:[ESP+C] DS:[ESI] DS:[EAX-10] DS:[ECX]	Dword		EIP 0044CAC0 Vi C 0 ES 0023 32 P 1 CS 001B 32 A 0 SS 0023 32 Z 1 DS 0023 32 S 0 FS 003B 32 T 0 GS 0000 NU D 0 O 0 LastErr ER
0044CAE2 0044CAE4 0044CAE5 0044CAE5 0044CAE6 DS: [005EFBD AL=01 Local calls	Long Float Disassemble Special Appearance	+ + +	10044552F, 004476A1	, 0044F686, 0045094C, 00450A8E B9∣RETURN to ViceVer⊆,0045AFB	2, 00450	EFL 00000246 (N STO empty -UNOR STI empty +UNOR ST2 empty +UNOR ST3 empty +UNOR ST4 empty -UNOR ST5 empty 1.000 ST6 empty 1.000 ST6 empty 1.000
005EFBD4 00 005EFBE4 00 005EFBF4 00 005EFBF4 00 005EFC04 7C 005EFC14 00	80 50 50 50 10 57 50 50 60 60 60 60 60 60 60 60 60 60 60 60 60 60 60 60 13 58 60 60 60 60 60 60 60 60 60 60 60 60 60	00 00 0 00 00 0 00 00 0 01 15 0 00 00 0	00128018 00E298 0012801C 77D3C6 00128020 00E2C1 00128024 77D384 00128024 77D384 00128028 00E2C1	F8 BC USER32.RedrawWindow 40 UNICODE "ViceVersa Pro" A9 RETURN to USER32.77D3B4A9 40 UNICODE "ViceVersa Pro"		

Figure 2.20 Setting a hardware breakpoint on pointer

And set a hardware breakpoint on access, byte. press ctrl+F2 to restart the application. Then F9 to run again.

C CPU - main thread, module ViceVers	
0044F2E5 88CC MOU ECX,ESP 0044F2E7 896424 0C MOU DWORD PTR SS: [ESP+C], ESP 0044F2E6 51 PUSH ECX 0044F2E7 51 PUSH ECX 0044F2E7 C74424 10 000 DWORD PTR SS: [ESP+1C], 0 0044F2F7 83C4 04 PDD ESP,4 0044F2F7 83C4 04 PDD UNORD PTR SS: [ESP+18],-1 0044F2F7 83C4 04 PDD ESP,4 0044F2F7 83C4 04 PDD UNORD PTR SS: [ESP+18],-1 0044F2F6 C74424 18 FFFI MOU DWORD PTR SS: [ESP+18],-1 0044F2F6 884C24 10 MOU ECX,DWORD PTR SS: [ESP+18] 0044F309 884C24 10 MOU ECX,DWORD PTR SS: [ESP+18] 0044F309 .884C24 10 MOU DUORD PTR SS: [ESP+18] 0044F319 .64:8900 00000 MOU DUORD PTR SS: [6],ECX 0044F310 .64:8900 00000 MOU ESP,1C .63 .64:8900 00000 MOU ESP,1C .64:8900 MOU EAX, 100C ESP 760537EA0 0044F31F CC INT3 0044F325 FS 76880E00 CALL ViceVers.00537EA0 0044F326 .88 0C10000 MOU EAX, 100CD PTR SS: [ESD+1008], EAX	Registers (FPU) EAX 005FBB00 VI ECX 0012FE80 EDX 00012FE80 EDX 0000039C EBX FFFFFFF ESP 00020000 EIP 000400001 EIP 00044F312 Vi C 0 ES 0023 32 P 0 CS 001B 32 P 0 CS 001B 32 P 0 FS 0023 82 C 0 ES 0023 32 S 0 FS 003B 32 T 0 GS 0000 NU D 0 LastErr ER EFL 00000202 (N ST0 empty -1.62 ST12 empty 0.0 ST2 empty 0.0
0044+090=010evers.0044+090	ST3 empty -3.02 ST4 empty -UNOR ST5 empty +UNOR ST6 empty 1.000
Address Hex dump 005E8000 00	<u>^</u>
005E8020 A7 BB 56 00 H2 7B 58 00 35 C9 56 00 E 0012B878 0012B86C 005E8020 A7 BB 56 00 C0 BD 56 00 35 C9 56 00 E 0012B87C 0012E88C 005E8030 FC 7B 58 00 20 70 58 00 30 70 58 00 € 0012B87C 0012FE80 Pointer to next SEH record 005E8040 70 70 58 00 90 70 58 00 B0 70 58 00 € 0012B880 005768E8 SE handler	

Figure 2.21 Finding the registered check call

Olly then breaks here (0044F312) scroll up a little and you will recognize the call to 0044F090! Set a breakpoint on that line (0044F304) and restart (ctrl-F2) and run again (F9).



When Olly breaks this time you should press F7 (step into the call) and you end up on 0044F304. Continue to execute the code step by step for a while by pressing F8.

C CPU - main thread, module ViceVers	
0044F0FC . 83CA FF OR EDX,FFFFFFFF 0044F0FC . F9:0FC111 LOCK XADD DWORD PTR DS:LECX],EDX 0044F104 . 85D2 TEST EDX,EDX 0044F108 . 8808 MOU ECX,DWORD PTR DS:LECX1 0044F108 . 8808 MOU ECX,DWORD PTR DS:LECX1 0044F100 . FF52 04 CALL DWORD PTR DS:LEDX+41 0044F110 . 8047 F0 LEA EAX,DWORD PTR DS:LEDX+11 0044F110 . 8047 F0 LEA EX,DWORD PTR DS:LECX1,-101 0044F111 . C74424 3C FFFI MOU DWORD PTR DS:LECX1,EDX	Registers (FPU) EAX 005FFB8S Uid EAX 005FFB8S Uid EDX 005FFB8S Uid EDX 00000039C EBX FFFFFFF ESP 0012B828 EBP 0000000 ESP 005FFB8S Uid EDI 005FFB8S Uid EDI 005FFB8S Uid EIP 0044F132 Uid C 0 ES 0023 32t A 0 SS 0023 32t Z 0 DS 0023 32t S 0 DS 0023 32t S 0 DS 0023 32t G 0 DS 0023 32t S 0 DS 0020 NUL D 0 LastErr ERF EFL 00000206 (NU S 0 empty -3.473 S 1 empty -3.748 S 1 empty -3.748 S 14 empty -0.068
	ST5 empty +UNOR ST6 empty 1.000
Address Hex dump 00128328 00000001 00000001 005E58000 00	m ViceVers.004BF4

Figure 2.22 Tracing in the registered check call

Here is where al is set to 0! Press space to assemble.

CPU - main thread, module ViceVers	
0044F0FC . 83CA FF OR EDX, FFFFFFFFF OR EDX, FFFFFFFFF 0044F103 .44 DEC EDX 0044F104 .85D2 TEST EDX, EDX 0044F106 .85D2 TEST EDX, EDX 0044F106 .85D2 TEST EDX, EDX 0044F106 .8808 MOU ECX, DWORD PTR 05: (ECX), EDX 0044F106 .8811 MOU ECX, DWORD PTR 04 0044F106 .550 PUSH ERX 0044F107 .552 04 CALL DWORD PTR 04 0044F110 > 8047 F0 LEA EAX, DWORD PTR 04 0044F110 > 8047 F0 LEA EAX, DWORD PTR 04 0044F110 > 8047 F0 LEA ECX, DWORD PTR 05 0044F110 > 8047 F0 LEA ECX, DWORD PTR 05 0044F110 > 8047 F0 LEA ECX, DWORD PTR 05 0044F118 .074424 3C FFF OR EDX, FFFFFFFF 0044F112 .776 08 JG SHORT UlceVers 0044F125 .44 DEC EDX DEC EDX 0044F126 .8808 MOU ECX, DWORD PTR 05: LECX1 0044F127 .8808 MOU ECX, DWORD PTR 05: LECX1 0044F126 .8809 DEX DWORD PTR D5:	 ▲ Registers (FPU) ► EAX 005FB8SC UIG ECX 005FB8SC UIG ECX 005FB898 UIG EDX 0000039C ► ESP 00128238 ► ESP 00128238 ► ESP 00128238 ► ESP 00165E858 UIG EDI 005E5E852 UIG EDI 0054FB89C UIG EIP 0044F132 UIG C 0 ES 0023 321 P 1 CS 0018 321 A 0 SS 0023 321 S 0 FS 0038 321 C 0 0 LastErr ERF EFL 00000206 (MU ST0 empty -1.623 ST1 empty -3.474 ST1 empty -3.474 ST4 empty -3.744 ST4 empty -3.744 ST4 empty -3.744 ► ST6 empty 1.0000
Hoaress Hex dump Monoress Monoress	m ViceVers.004BF4

Figure 2.23 Patching with Olly

Apply the change by pressing Assemble and then cancel. Right-click and choose 'Copy to executable.. All modifications'. And then 'Copy All'. Then a new window appear, right-click on that window and then choose 'Save file'.



D File C:\Program\ViceVersa Pro 2\ViceVersa.exe				
0004F132 B0 01 0004F134 8B4C24 34 0004F138 64:890D 0000000	MOV AL,1 MOV ECX,DWORD PTR S MOV DWORD PTR FS:10	S:[ESP+34]],ECX		<u>^</u>
004F140 5E 004F140 5E 0004F144 83C4 38 0004F144 C3 0004F144 C3 0004F145 6A 01 0004F145 8D4C24 4C 0004F148 E8 F02AFBFF (0004F150 50 0004F150 8D4424 1C	POP EDI ADD ESP,38 RETN PUSH 1 LEA ECX,DWORD PTR : CALL 00001C40 PUSH EAX LEA EAX.DWORD PTR :	Backup Copy Binary Assemble		
0004F155 50 0004F156 E8 A5E9FFFF 0004F156 83C4 08 0004F15E 8378 18 10 0004F152 C64424 3C 02	PUSH EAX CALL 0004DB00 ADD ESP,8 CMP DWORD PTR DS:[I MOV BYTE PTR SS:[E: DV DVDT PTR SS:[E:	Search for Save file Go to offset	Ctrl+G	
0004F167 ×72 05 0004F169 8B40 04 0004F16C ×EB 03	MOV EAX, DWORD PTR I JMP SHORT 0004F171	View image in Disasser	nbler	~
		Hex Text Short Long Float V Disassemble Special	+ + + + +	
		Appearance	•	

Figure 2.24 Saving the modified exe

Name the file to whatever you want, close Olly and you're done!

3. References

[1] Vortex 168, *The Lazy Beginning Reverser's Guide to Windows Assembly*, 2002
[2] Intel® Architecture Software Developer's Manual, Volume 2A, page 3-486, 2005

4. Conclusions

Using a single pointer dedicated for registration check is vulnerable indeed and should, if possible, be avoided. The use of an interactive debugger gives you the benefit of actually seeing the real values of pointers and registers as you go along, but is often more complex to handle and demands a great deal of knowledge to be used efficiently. The deadlist method is pretty straight forward but can sometimes be hard to follow due to the fact that you do not know the current state of registers and pointers.

5. Greetings

Fly out to all members of the victorious ARTeam and to all other whom in some way contribute to the scene.

Best regards, merry Christmas and happy reversing...

potassium / ARTeam

http://cracking.accessroot.com