

Forum: [Hack Forums](#)

Thread: [how to crack any soft +pic = a full totutrial to crack any soft too easy](#)

the_metalisme - August 16, 2009, 7:27 pm

Intro:

All the tools you will need can be found online:
pm to give you the link or just download it here
<http://#.com/files/268231152/hw32v601.exe>

Let us first get set up and ready to crack. I like to open the folder with my target in it. In this case c:\program files\@stake\ and make a quick backup that we can work on without fear of screwing up the file. I named mine lc52.exe and I will refer to it as such through the tutorial.

We will begin again by opening LC52.exe in PEID to see if it is packed or protected. The result: Microsoft Visual C++ 7.0. Good, we will not have to worry about unpacking this file before reversing it.

Note: I used three seperate computers to create this tutorial. This is why my serial changes halfway through. It is also why my Addresses in Olly change. If your Addresses in Olly do not match mine, it is not a problem. The code will be the same.

Body:

Knowing that we will not need to unpack the executable lets us do a little examination of the protection scheme. Open up LC52.exe. It goes directly into the nag screen. From the previous tutorial we know that it is checking for a registration key. Press Register and we are presented with a new window holding a unique serial number and a box asking for our unlock code. Enter anything in as the unlock code and press Okay. We get a message box saying "You have entered an invalid code. Please try again.". Remeber to write this message down as we will be searching for it later. Press Okay and the program returns to the window asking for our unlock code.

Searching For A Serial:

Rather than try and patch the registration of this target, we are going to try and find a valid serial number. Close LC52.exe and open Ollydbg. In Olly, go to File, and open LC52.exe. It will take a second as Olly analyzes the code. We are going to start by looking for the string from the error message we recieved. To do this, right click on the code window and go to Search For, select All Referenced Text Strings.

[img]http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut1.gif[/img]

A new window will open up with all the referenced strings in the executable. Scroll to the top of the page and select the first string. Right click and choose Search For Text. In the dialog box enter "you have entered" without the quotes. Make sure Case Sensitive is unchecked. You should end up here.

[img]http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut22.gif[/img]

Double click the selected line and you will be taken to this code in the program:

[img]http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut11.gif[/img]

Scroll up a little and we can see that there are three different versions of the program you can register as. Basic, Professional, and Administrator. We see that there is a TEST EAX, EAX before each registration confirmation. The program first tests if you entered the Basic unlock code, then if you entered the Professional unlock code and, finally if you entered the Administrator code. We are

going to try and register the Administrator version. Scroll back down until your Screen matches what is shown above. If we look closely we can see where the test is made to call the messagebox. 0041082A . 85C0 TEST EAX,EAX followed by a push and then a Jump if Not Zero, JNZ 4108B2. 004108B2 is where our invalid message is created. Let's find out what is going on when the test is made. Select 0041082A . 85C0 TEST EAX,EAX and press F2. This will set a breakpoint on the test.

Press the Run button. The LC5 nag screen pops up, press the Register button. In the window asking for our unlock code enter any serial. Let's try 1234567890. Press Okay. Olly breaks on our TEST EAX, EAX. First thing you notice is the registers on the right hand side. There are a string of numbers and letters in ECX and EDI that look suspiciously like an unlock code.

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut4.gif>[/img]

Was it that easy? Well, try them and find out. Open the original LC5.exe and use the numbers to try and register. This is new, a license agreement opens up. After pressing Okay on the License Agreement, we are presented with the message that we have successfully registered the Administrator version of LC5.

It was that easy!

So, are we done?

No!

Before we continue, return the program to the state of unregistered. Go to the START MENU in Windows and select the RUN command. Type in REGEDIT and press Okay. In REGEDIT's left window, click on the [+] in front of HKEY_CURRENT_USER. Then click on the [+] in front of SOFTWARE. Click on the [+] in front of @STAKE and then the [+] in front of LC5. Select the Registration folder. In the right hand window you will see Unlock Code. Double click it, and erase the Value Data in the new window. Press Okay and close REGEDIT.

Creating An Internal Keygen:

Because the unlock code for this program is computed based on the unique serial of our computer, anyone who wants to register would have to find the unlock code for themselves. We are going to solve that problem by using the program as a personal keygen. This is not often considered good reversing because you do not understand how the key is generated. However, as a beginner it can be an valuable learning experience.

Before we begin let's evaluate what we need to do to create an internal keygen.

1. We need to find where the unlock code is created and stored in memory
2. We need to find some way to display the unlock code to the user
3. We need to let the user know that they are seeing the unlock code

We already know where to find the unlock code so we need to solve how to display the code. A good idea would be, rather than displaying an error message, we display the unlock code. Let's reexamine the code where we see the test and where we see the "You have entered an invalid..." string pushed for the message box.

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut5.gif>[/img]

What we want to do is edit the code. Instead of pushing the error message onto the stack we can push our unlock code. There are a few things to consider when editing code. We need to make sure that what we change is not vital to the program's function. We are also restrained by how many bytes we can use to write new code. We can only overwrite current code. We cannot add our own. (There are ways to add your own code but it is more advanced than this tutorial, search for "code caves" if interested) The other thing we need to consider is the stack. The stack is a designated part of memory where you can save values and instructions to. The stack is true to it's name, values are stacked onto it as you would stack boxes. The first one on is at the bottom and the last one is on

the top. Instructions are then removed from the stack in the order of top to bottom. Last one on is the first one off.

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut6.gif>[img]

You can view the stack in Ollydbg at the bottom right of the window. When modifying code we must make sure that nothing extra gets added or removed from the stack. Otherwise our program will cease to function.

Now that we know what we need to consider, let's move onto how to implement the unlock code into the messagebox. We know where the error message is pushed onto the stack so let's start by changing the code to push our code instead. We can see our code in both the ECX and EDX registers. However before we get to the PUSH, EAX is zeroed out and our code is overwritten in memory. So let's address that issue first. We only need our code in one register so we can keep XOR EAX, EAX. To stop our code from being overwritten we need to change MOV DWORD PTR SS:[ESP+14],ECX, MOV DWORD PTR SS:[ESP+1C],ECX and, MOV BYTE PTR SS:[ESP+24],CL. Select the first one with your mouse and right click. Choose Assemble from the menu. This will bring up a box where you can edit the code.

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut7.gif>[img]

We want to remove this code altogether, so in the box type NOP and make sure that the Fill With NOP's is selected. NOP stands for No Operation and is written in hex as 90. When the program runs it will just pass over these bytes. Press the Assemble button and your code is edited with NOP's. Do the same for the other two lines that overwrite our unlock code. It should end up looking like this:

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut8.gif>[img]

Bazerkin - August 16, 2009, 7:48 pm

Nice tutorial, but useless for login-based software that requires a login based in a MYSQL table.

the_metalisme - August 16, 2009, 8:01 pm

Okay, we have now successfully stopped our code from being overwritten. Press RUN and make sure that the program still works. Excellent, we still see our error message box pop up. Press Okay on the error box, and Okay on the Registration box so we will return to our breakpoint in Olly. At our breakpoint, begin pressing the Step Over button . Keep a watch on the registers where our serial is stored. As we get to the PUSH EBX, we see that our serial is still in register EDX. Keep stepping until you get to the NOP just above the PUSH lc5.004D62D4.

We are now going to make our message box display the unlock code. Select PUSH lc5.004D62D4 and select Assemble. When the box comes up, enter PUSH EDX and press Assemble. Now press the Run button and...

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut9.gif>[img]

It works! We are now halfway finished. Before we continue we want to make sure we save our changes to a file. Return to the breakpoint in Olly and right click anywhere in the code section. Select Copy to Executable, and then All Modifications. A box pops up asking us if we want to copy the selection, choose Copy All. A new window will open with the program in it and our changes included. Right click and select Save File. Save it as LC5Modified.exe. You can now close Olly, though we will use it again in a few seconds.

Modifying the Internal Keygen:

Because we made the keygen, we know that the messagebox is now displaying an unlock code. A normal person would not know what these letters and numbers represent. What we are going to do next is change the title of the window to inform the user that they are seeing the unlock code for the Administrator version.

A Quick Overview of the MessageBox Function:

The MessageBox function is an internal function in Windows. This means that a programmer can write a program and, instead of having to code his/her own message box, they can use the one included with Windows.

The MessageBox function is called like any other function. When it is used you must pass it variables called Arguments.

The Arguments for MessageBox are HWND hWnd: handle of owner window, LPCTSTR lpText: address of text in message box, LPCTSTR lpCaption: address of title of message box, and UINT uType: style of message box.

Knowing how a MessageBox is created gives us the knowledge of how to manipulate it. (For more information on internal functions of Windows research Win32 API). What we want to do is find where the program calls MessageBox and where the Arguments are passed.

Open your new LC5modified.exe up in Ollydbg. Right click in the code area and choose Search For, select All Intermodular Calls. A new window opens up with all the functions called by the program. We will sort these by Destination by clicking the Destination bar at the top. As you can see they are now in alphabetical order. Scroll down until you find MessageBoxA. Select one of the MessageBoxA lines and right click. Now select Set Breakpoint On Every Call To MessageBoxA:

[img]http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut10.gif[/img]

Press the Run button to start the program. At the nag screen go through the registration process again. Remember to enter an invalid code. When you press Okay, to try and register with your code, you should break here:

[img]http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut11.gif[/img]

If you take a look at the Stack you can see the Arguments that were PUSHed for this MessageBox. The first one pushed on is at the bottom of the stack. So the first Argument is Window Style, the second is Title, the third is the Text, and the last is the Owner Window.

Our objective is to let the user know that the code they are seeing is the Unlock Code. A good place to do so would be in the Title of the message box. Let's examine how we are going to change the Title.

We need to find a place in code to create our new text string. Because we cannot write new code we are going to have to overwrite old code. We then need to change the code for the MessageBox call to PUSH our string as the Title Argument. Let's begin by creating a new string.

Because we have already modified this program, we know of a string that is not being used: "You have entered an invalid code. Please try again". If we refer back to the original code of the serial test we can see this string being pushed onto the stack: PUSH lc5.004D63D4. We now know that the string is located at 004D63D4. We will change it later, let's first try and get the program to actually push this value in.

Looking at the code we see that this part of the code jumps to MessageBoxA.

74 05 | JE SHORT lc5modif.004BD576 : Jumps to GetModuleFileNameA

8B79 4C | MOV EDI,DWORD PTR DS:[ECX+4C] : Moves ECX+43 into EDI. EDI is where the Title is stored.

EB 1A | JMP SHORT lc5modif.004BD590 : Jumps to the MessageBoxA call

In the Arguments for MessageBoxA we see EDI is pushed as the Title. And the value for Title is moved into EDI just before the call to the MessageBox. We need to change the program to move our string into EDI rather than ECX+43. I am going to show you the very ugly, unaccepted, however it works, way to change the program.

Begin by selecting MOV EDI,DWORD PTR DS:[ECX+4C], select Assemble (or press spacebar),

and in the box type MOV EDI, 4D63D4. Press Assemble. Do you see why this method is ugly and unaccepted? Because our code to move the string into EDI is longer than the original code, we have just overwritten our jump to MessageBox. "Well, just write it underneath". That would seem the logical option, but we will be destroying the function call to GetModuleFileNameA. Let's find out how many times GetModuleFileName is called before we do any overwriting. Select the first line of the function: 68 04010000 PUSH 104. At the bottom of the code window we see: Jump from 004BD56F. There is only one jump to this function and it is directly above our jump to MessageBoxA. Set a breakpoint on JE SHORT lc5modif.005BD576 and press Run. Go through the registration process and when we reach our breakpoint (do not step past it or the program will quit) we see in the bottom code window that the Jump is not even taken. If we insert our new jump ovetop of PUSH 104 the program will call a MessageBox instead of GetModuleFileName but we do not even see the program calling GetModuleFileName anyway so let's try it out.

Select PUSH 104 and choose Assemble. Enter JMP SHORT 4BD590 (Your number may need to be different, check the Address of PUSH EBX, that is the Address you want to jump to). Your code should look like this:

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut8.gif>[/img]

Go ahead and press Run to get to the MessageBoxA breakpoint. Press Run again and let's see if it worked.

It worked! Go through the registration process a few times to make sure that the program does not crash. For future information: The correct way to modify this program would be to jump to a code cave, move our serial into EDI and jump back to the MessageBox function. You can try it out on your own if you are interested, I may cover code caves in another tutorial.

Now we want to save our modifications before we do anything else. Just as before, select Copy to Executable, and then All Modifications. A box pops up asking us if we want to copy the selction, choose Copy All. A new window will open with the program in it and our changes included. Right click and select Save File. Save it as LC5Final.exe. You can close Olly, we are almost done. The last thing we need to do is change the text string "You have entered..."

Changing The Text String:

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut13.gif>[/img]

We want to change the text into something informative. To do this Open LC5Final.exe in Hex Workshop. You can do so by right clicking the actual file and choosing Hex Edit with Hex Workshop. With LC5final open, right click in Hex Workshop and choose Find. Select Text String from the drop down box and enter "You have entered" without the quotes. Make sure Match Case is unchecked and press Okay.

You will end up at 000D63D4. Sound familiar, that is the same value you pushed into EDI just a few minutes ago. We have found the correct string.

We can only overwrite the current string, we cannot use any more or any less space. By placing your cursor just before the Y in the right hand window you can type just as you would in any text editor. Experiment around until you find something that will completely overwrite the string. Keep in mind the string includes a period at the end. Here is my modified string:

[img]<http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut14.gif>[/img]

Backup LC5final.exe and then save the newly modified program. Close Hex Workshop and try out LC5Final. Go through the registration process and... Success! We now have a Title in our MessageBox that will let the user know what they are seeing.

Creating a Patch:

The final thing we are going to cover is how to create a patch. What happens when we reverse a

program and want to share that new reversed program with other people? We can zip up the executable and send it to them, however some programs are very big. We can write a tutorial on it, however that takes time and assumes reversing knowledge of the other party. Or we can quickly create a small program that will automatically apply the changes we made to the file. That is what we are going to do here.

There are two ways to make a patch. Code it completely yourself, or use a program that will help you create it. There are many different patching programs to choose from. I am going to use Diablos2002 Universal Patcher, or DUP. It is newer, easy to use, and allows you to customize how and what you want to

patch.http://i562.photobucket.com/albums/ss64/themetalisme/Ollytut15.gif

Begin by opening up DUP. You start in the Patcher Settings. Click the browse button next to the Target Filename box and browse to lc5.exe. The other boxes are optional, under Release Info you may want to give a little description on how to use the patch. You can customize the About Box and the Icon and add your own skin as well as other options, by pressing the Options button.

Once you have filled in the boxes, choose the Offset Patch tab at the top. Select, for the Original File, LC5.exe and, for the Patched File, LC5Final.exe. Press the Compare button and it will create a list of the different bytes between the two. Last thing you need to do is press the Create Patch button and choose a name for your patch.

You are done! Backup LC5.exe and run the patch. Then try registering with an invalid serial... It works! You have successfully modified a program to create an display a valid serial. You also learned how to create a patch so you can apply the modifications on other computers. You can now apply the techniques you learned to other reversing projects.

Conclusion:

I used this particular program purely as a demonstration for internal keygens and patching. If like the program and are going to use it please purchase it.

Thanks to all the people who take time to write tutorials. Without the teaching's of others we would all lack knowledge. Thanks to Exetools, Woodmann, and ARTeam for being a great place of learning.

Thanks also to The Codebreakers Journal

i hope you enjoy cracking

<http://#.com/files/268231152/hw32v601.exe>

and for who they have a doubt in me this is a full pack tools for crack

[A.I.O. - Crack Searching Shareware Hacking.rar](#)

Bazerkin - August 16, 2009, 8:11 pm

Ha, when I posted you didn't have the full tutorial up. Wow this is very nice! Thanks man!

the_metalisme - August 16, 2009, 8:20 pm

well i have more comming in few minute a hole package for cracking man enjoy it am uploading it

J4P4NM4N - August 16, 2009, 8:22 pm

Awesome tut, man! I just downloaded your program and found no viruses w/ AVG or ClamWin (for those that are curious). Will mess around with it tomorrow.

noob2001204 - August 16, 2009, 8:23 pm

does this crack any kind of software?

thanks!

the_metalisme - August 16, 2009, 8:27 pm

name of some helpfull tools

The Tools:

OlllyDbg 1.09d, HexWorkshop 4.1, Diablos2002 Universal Patcher

The Protection:

Serial Protection

Other Information:

This is a tutorial to introduce the beginner to internal keygens and introductory patching. This will take you through the steps of finding the serial and constructing an internal keygen and creating a patch for the original file.

the_metalisme - August 16, 2009, 8:32 pm

what you mean i didn't undersatnd you if you can explain in french

J4P4NM4N - August 16, 2009, 8:36 pm

(Aug 16, 2009 08:29 PM)cofusion Wrote: [\[CITE\]](#) by reading that little bottom paragraph you can see that he coppied and pasted this.

Well obviously I didn't care enough to actually read the entire thing =P Plus to me it doesn't matter if he copied/pasted . . . info went to my eyeballs. That's pretty much all that matters at this point. My "thanks" is generally just a courtesy and not actually authentic.

the_metalisme - August 16, 2009, 8:41 pm

well

first in my country its 04.49 i worked on this for more than 10 hours

second it doent matter the source what matter is the result lol

andd of course i write this

and for who they have a doubt in me this is a full pack tools for crack

[A.I.O. - Crack Searching__Shareware_Hacking.rar](#)

the_metalisme - August 16, 2009, 8:59 pm

well am not here for your toys next time ill report you

maxiaccurus - February 14, 2020, 3:17 am

Has anyone confirmed if this worked?

I would love to know