Cracking With +Vip-Vop

Cracking Winrar 2.04

http://www.creabel.com/softronic

Tools Needed:
W32Dasm
Softice
Your favorite hex-editor

To start off open up WinRAR95.exe in W32dasm. Go to refs, then choose string references to look for interesting strings. Right away you'll see evaluation copy and registered version, but searching for those strings won't get you anywhere fast. So scroll down a little bit more until you get to string number 51, "Available in registered version only". This sounds like a good string to check for, so double click on it. You'll stop in the following code:

```
* Referenced by a (U)nconditional or (C)onditional Jump at Address:
|:00413655(C)

:00413659 833D5857420000        cmp dword ptr [00425758], 00000000
:00413660 7534                  jne 00413696

* Possible Reference to String Resource ID=00048: "Normal"

:00413662 6A30                  push 00000030

* Possible Reference to String Resource ID=00026: "Warning"

:00413664 6A1A                  push 0000001A
:00413666 E84D680000            call 00419EB8
:0041366B 59                    pop ecx
:0041366C 50                    push eax

* Possible Reference to String Resource ID=00051: "Available in registered version only"

:0041366D 6A33                  push 00000033
:0041366F E844680000            call 00419EB8
:00413674 59                    pop ecx
:00413675 50                    push eax
```

```
:00413676 53                    push ebx

* Reference To: USER32.MessageBoxA, Ord:0000h
                                 |
:00413677 E8F60E0100            Call 00424572
:0041367C 6A00                  push 00000000

* Possible Reference to Dialog: GENERALCFGDLG, CONTROL_ID:0089, "Put authenticity &verification

* Possible Reference to String Resource ID=00137: "Comment broken"

:0041367E 6889000000            push 00000089
:00413683 53                    push ebx

* Reference To: USER32.CheckDlgButton, Ord:0000h

:00413684 E81D0E0100            Call 004244A6
:00413689 6A00                  push 00000000
```

Now that's a lot of code, but look near the bottom. Notice the reference to authenticity verification? If you open up WinRAR, go to "Options", then "Interface", then "General", you'll see a check box for put authenticity verification. If you try and check it though a nag box pops up with the words (guess) "Available in registered version only". So lets look just at the nag box in the code above:

```
:00413659 833D5857420000        cmp dword ptr [00425758], 00000000
:00413660 7534                  jne 00413696

* Possible Reference to String Resource ID=00048: "Normal"

:00413662 6A30                  push 00000030

* Possible Reference to String Resource ID=00026: "Warning"

:00413664 6A1A                  push 0000001A
:00413666 E84D680000            call 00419EB8
:0041366B 59                    pop ecx
:0041366C 50                    push eax

* Possible Reference to String Resource ID=00051: "Available in registered version only"

:0041366D 6A33                  push 00000033
:0041366F E844680000            call 00419EB8
:00413674 59                    pop ecx
:00413675 50                    push eax
:00413676 53                    push ebx
```

```
* Reference To: USER32.MessageBoxA, Ord:0000h

:00413677 E8F60E0100          Call 00424572
```

See how it compares [00425758] to 0, and if its not equal (meaning its most likely equal to 1), it jumps over the nag box, to the code which has calls like CheckDlgButton which you can guess checks the box for authenticity verification. So now all we have to do is see when [00425758] is set to anything, and make sure its set to 1 instead of 0. This is another one of those things where [00425758] is probably a boolean variable in the author's source, name registered or isreg or something like that. So search through your listing of the program for [00425758]. You will get a couple of false stops where its comparing [00425758] to something, but ignore them all until you reach code where its value is being altered. You will end up at this code:

```
  :0040A55C FF35F0BE4200          push dword ptr [0042BEF0]
:0040A562 E809270100            call 0041CC70
:0040A567 83C40C                add esp, 0000000C
:0040A56A 85C0                  test eax, eax
:0040A56C 0F94C1                sete cl
:0040A56F 83E101                and ecx, 00000001
:0040A572 890D58574200          mov dword ptr [00425758], ecx
:0040A578 FF3538574200          push dword ptr [00425738]
:0040A57E E845870100            call 00422CC8
:0040A583 59                  pop ecx
:0040A584 33C0                  xor eax, eax
:0040A586 A338574200            mov dword ptr [00425738], eax
:0040A58B A158574200            mov eax, dword ptr [00425758]
```

What does that do? well lets take an in depth look at the lines where ecx is moved into [00425758].

:0040A562 E809270100 call 0041CC70

A call that we can almost guarentee modifies eax (see below for importance of eax)

:0040A567 83C40C add esp, 0000000C

Just correcting the stack (hint: don't mess up any instructions dealing with esp, it will most likely screw up the program)

:0040A56A 85C0 test eax, eax

If eax is 0, it will set the equal flag, otherwise it won't

:0040A56C 0F94C1 sete cl

Set if equale cl. This means if the equal flag is set (from the previous test eax,eax call) the cl register (the lower part of cx a.k.a ecx) will be set (made equal to 1)

:0040A56F 83E101 and ecx, 00000001

And the value of ecx by 1. In order to see how this effects it go to your start menu, click on programs, then accessories, then calculator. Goto view then choose scientific. Now see how the "dec" spot is checked? Check off "hex" instead. Now lets test some possible values of ecx. Lets try 0. Press 0, then and, then 1. It is equal to 0. Now lets say ecx is equal to 1. press 1, then and, then 1. Its now equal to 1.

:0040A572 890D58574200 mov dword ptr [00425758], ecx

Now the value of ecx is moved into our isreg flag. Remember if [00425758] is equal to 0 then we get nagged, so we want it to equal 1.

So now that we know we want ecx to equal 1, we know ecx must already be 1 when it is ANDed by 1. And in order for it to be 1, the test eax,eax must set the equal flag so the "sete cl" will make ecx 1. What does that mean? It means that call 0041CC70 should make eax equal to 0 so the test eax,eax will set the equal flag. If this is kind of confusing look at the code one last time, keeping in mind we want ecx to be 1:

```
0:040A562 E809270100        call 0041CC70; needs to make eax 0
:0040A567 83C40C            add esp, 0000000C;messing with the stack
:0040A56A 85C0              test eax, eax; if eax is 0, equal flag is set
:0040A56C 0F94C1            sete cl; if equal flag is set, ecx is 1
:0040A56F 83E101            and ecx, 00000001; if ecx is 1, it stays 1
:0040A572 890D58574200      mov dword ptr [00425758], ecx; setting isreg
```

So what should we do? Lets go to the location called in W32dsm, and check out how eax is set. So go to :0041CC70 and heres what you see:

```
* Referenced by a CALL at Addresses:
|:00409DE5   , :00409E20   , :00409E5B   , :0040A562   , :0040AECB
|:00417A80   , :0041E876
|
:0041CC70 55                push ebp
:0041CC71 8BEC              mov ebp, esp
:0041CC73 53                push ebx
:0041CC74 8B4D10            mov ecx, dword ptr [ebp+10]
:0041CC77 8B4508            mov eax, dword ptr [ebp+08]
:0041CC7A 8B550C            mov edx, dword ptr [ebp+0C]
:0041CC7D 85C9              test ecx, ecx
:0041CC7F 751A              jne 0041CC9B
:0041CC81 33C0              xor eax, eax
:0041CC83 5B                pop ebx
:0041CC84 5D                pop ebp
:0041CC85 C3                ret
```

There we go, now the program is cracked. See how its called many times? That means this is most likely the main registration procedure. See the xor eax,eax on line :0041cc81? That will make eax equal to 0, exactly what we want. But there is that jne 0041cc9b right before it. Well if we never jump then eax will always be xor'ed by itself (any time you XOR a number by itself it will equal 0) so lets just get rid of that jump. So simply changing the bytes "751A" to "9090" (two NOPs), will crack this program. So open up WinRAR95.exe in your hex editor and search and replace the following bytes:

SEARCH: 85C9751A33C0
REPLACE: 85C9909033C0

We have to include those extra bytes so we don't replace the wrong 751A in the exe file. Now save WinRAR95.exe and start it up. What the hell it says its still an evaluation copy! Well a lot of program wont execute the registration procedure unless there is something for where it stores your name and reg number. In some kind of pseudo code it might look like this:

```
temp=GetRegName
temp2=GetRegNum
if temp = 0 then goto bad
if temp2 = 0 then goto bad
isreg = checkNameandNum(temp,temp2)
goto end
bad:
isreg = false
end:
```

Get it? Well lets make sure there is some info in where it stores our reg name and number. Well it stores its settings in rar.ini in its directory, but instead of manually adding our name lets have the program do it for us. Start WinRAR, go to "Options", then choose "Registration". Enter whatever you want for your name and reg number, then click Ok. It thanks you for registering and all that crap. Now close the program and restart it. It worked! its registered. How did I know that it would let us register with any name and num? well remember all the times the reg procedure we cracked was called? I took a guess and assumes the Reg part of the program used that procedure too. So there we go, WinRAR is cracked.

Written by : +Vip-Vop
Published by: Splatter Industries,Inc.