Tutor2:

In this tutorial we are going to start taking a look at registration possibilities...
We will again be looking at another program I coded.

Step 1 : Run the program

You will see a nice dino :-) Nothing says you are unregistered ... or tells you you are registered... Well .. check out the help menu, ah... a register option :-) Select it ... and you will see another dialog box pop up ... In this one you can enter your name and serial ....
Do so, enter your name and a random serial and press ok... hmmm.. it tells you that you entered a wrong serial .... It also asks you to reenter it please ....

Step 2: Disassemble the program.

When you disassemble this program, and look in the string references, (Refs->String Data References) you will see a few strings about registration... Some of em you already have seen... The one that complains about a wrong serial name combination... But you see that there is a possibility to get a Thank you message :-)

Double click on the "you entered an invalid serial" string ... and you will land right in the middle of the registration routine of the program.

```
////////////////////// Code snip ////////////////////////////
 ADDRESS MACHINE CODE             ASSEMBLER INSTRUCTIONS

:004013B7       55                                  push ebp
:004013B8       89E5                          mov ebp, esp
:004013BA       833D0020400000                      cmp dword ptr [00402000], 00000000
:004013C1       7416                                je 004013D9
:004013C3       6A40                          push 00000040

* Possible StringData Ref from Data Obj ->"Registered"
                                    |
:004013C5       68E8404000                          push 004040E8

* Possible StringData Ref from Data Obj ->"Thank you for registering ..."
                                    |
:004013CA       68F3404000                    push 004040F3
:004013CF       FF7508                              push [ebp+08]

* Reference To: USER32.MessageBoxA, Ord:018Ah
                                    |
:004013D2       E821030000                Call 004016F8
:004013D7       EB14                                jmp 004013ED

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
|:004013C1(C)
|
:004013D9       6A10                          push 00000010

* Possible StringData Ref from Data Obj ->"Error in Entry "
                                    |
:004013DB       686E404000                    push 0040406E

* Possible StringData Ref from Data Obj ->"You entered an invalid serial "
                                        ->"/ name combination ...please check "
                                        ->"your entry thoroughly and try "
                                        ->"to reenter"
                                    |
```

```
:004013E0       687E404000                      push 0040407E
:004013E5       FF7508                      push [ebp+08]

* Reference To: USER32.MessageBoxA, Ord:018Ah
                                 |
:004013E8       E80B030000                      Call 004016F8

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
|:004013D7(U)
|
:004013ED       5D                          pop ebp
:004013EE       C3                          ret
////////////////////// Code snip ////////////////////////////
```

If you quickly look at this piece of disassembly, you will easily understand what
is going on ... A part of this code is able to tell you that you are registered, and thanks you for it.
The other part can tell you that you entered a wrong number / name combination.
In some pseudo-code it would look like this :

If correctserial then tell em thank you
else tell em wrong serial

The c-code looked as following :

```
////////////////////// Code snip ////////////////////////////
void DoFakeCheckRoutine(HWND hDlg)
{
  if (registered ) MessageBox(hDlg, "Thank you for registering ...\r\n"
                                    "We will be thankfull to you for the rest of our lives !",\
                                 "Registered", MB_ICONINFORMATION);
        else MessageBox(hDlg,     "You entered an invalid serial / name combination ..."
                                 "please check your entry thoroughly and try to reenter",
                                 "Error in Entry ", MB_ICONERROR );
}
////////////////////// Code snip ////////////////////////////
```

No big deal right ? the thing is we have to find the place where the decision is made... That is luckily
not that different either ...

```
////////////////////// Code snip ////////////////////////////
ADDRESS         MACHINE CODE                ASSEMBLER INSTRUCTIONS
:004013BA       833D0020400000                      cmp dword ptr [00402000], 00000000
:004013C1       7416                                je 004013D9
:004013C3       6A40                        push 00000040

* Possible StringData Ref from Data Obj ->"Registered"
                                 |
:004013C5       68E8404000                          push 004040E8

* Possible StringData Ref from Data Obj ->"Thank you for registering ..."
////////////////////// Code snip ////////////////////////////
```

The je 004013D9 at address 4013C1 is the instruction that controls the flow of the program. If you manipulate this instruction, you will have control over the code :-)
And now in simple English : If you make sure the processor never executes the je instruction, you will be

registered :-)
to prevent this je instruction from occuring, the best way is to nop it.

Step 3: Open the program in Hiew and Patch it:

Drag and drop a copy of your tutor2.exe on top of hiew, so that it opens in hiew. Now go stand on address 4013C1 in w32dasm. Look at the statusbar. It says : data @L004013C1 @Offset 00007C1h in File Tutorial2.exe
Switch over to Hiew. Get into decode mode by pressing Enter twice. Now press F5 and enter 7C1 and you will land right on top of the je instruction.
Modify the je instruction by first pressing F3 then F2 and then reassembling 2 NOP instructions. Press Esc once and then F9 to save your changes. Verify that your changes are correct ... Hiew should look like this :

```
////////////////////// Code snip ////////////////////////////
ADDRESS:MACHINE CODE                ASSEMBLER INSTRUCTIONS

000013C1: 90                                          nop
000013C2: 90                                          nop
000013C3: 6A40                                        push        040
000013C5: 68E8404000                                  push        0004040E8
000013CA: 68F3404000                                  push        0004040F3
000013CF: FF7508                                      push        d,[ebp][00008]
000013D2: E821030000                                  call       .0000016F8     -------- (1)
000013D7: EB14                                        jmps        .0000013ED     -------- (2)
////////////////////// Code snip ////////////////////////////
```

Then press F10 to exit hiew.

Step 4: Verify your crack:

Run the program and go through help -> register and enter any name and serial ... and press enter ...
Look what happens :-)) And enjoy your power ... Enjoy the power to control !

Extra:

Do you think these first two programs I coded and we cracked are too easy ? They are nothing compared to real life programs ??
Believe me, if you have understood the last two tutorials, and successfully followed every step, you can crack 60% of
all shareware applications availble on the internet ... All you need is : a little practice ...
Don't you believe me ? Read the Next tutorials... We will be cracking a few shareware apps... And you will see that they are very much like the first two apps we cracked :-)