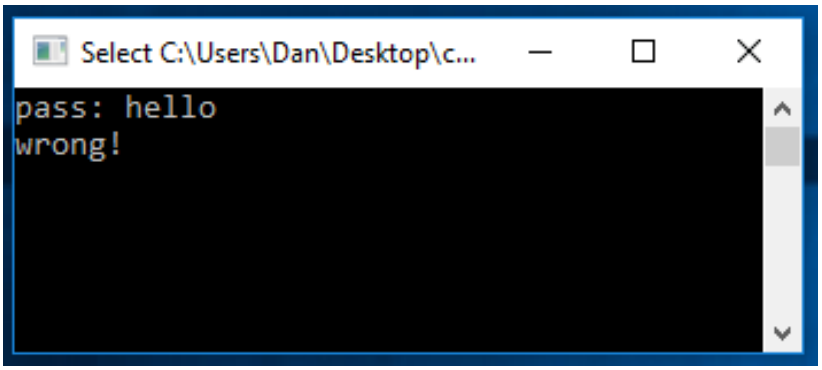
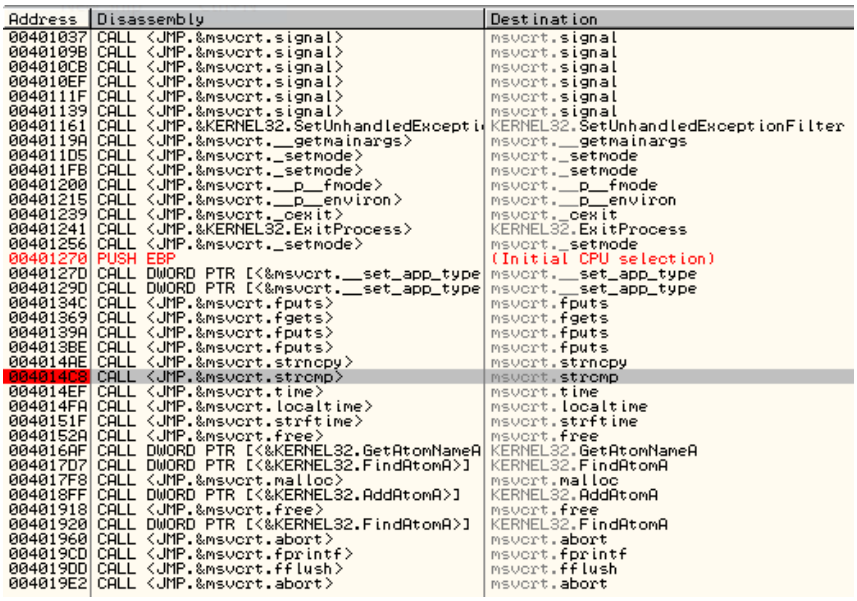
Final Turn In (extended): Crack Me Daniel Hoynoski
Friday, May 4th, 2018 **"dailycracking"** Net ID: hoynosk2

1. No rules were given with this Crack-Me, so I will first open it up and see what the objectives are:

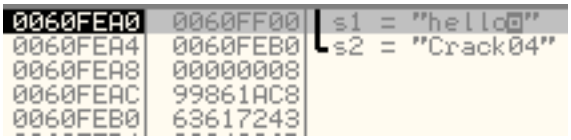


2. I enter the password "hello" and am greeted with the bad message. I pop the binary into OllyDBG and look at the intermodular calls:

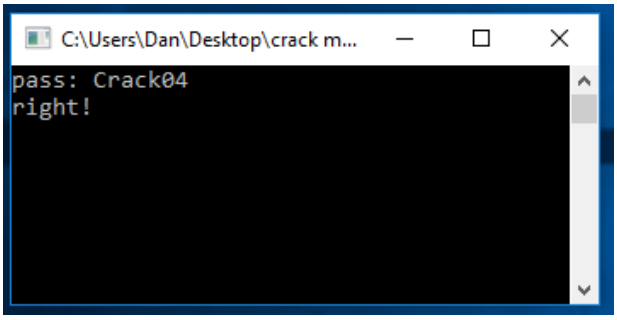


Address	Disassembly	Destination
00401037	CALL <JMP.&msvcrt.signal>	msvcrt.signal
0040109B	CALL <JMP.&msvcrt.signal>	msvcrt.signal
004010CB	CALL <JMP.&msvcrt.signal>	msvcrt.signal
004010EF	CALL <JMP.&msvcrt.signal>	msvcrt.signal
0040111F	CALL <JMP.&msvcrt.signal>	msvcrt.signal
00401139	CALL <JMP.&msvcrt.signal>	msvcrt.signal
00401161	CALL <JMP.&KERNEL32.SetUnhandledExcepti	KERNEL32.SetUnhandledExceptionFilter
0040119A	CALL <JMP.&msvcrt.__getmainargs>	msvcrt.__getmainargs
004011D5	CALL <JMP.&msvcrt._setmode>	msvcrt._setmode
004011FB	CALL <JMP.&msvcrt._setmode>	msvcrt._setmode
00401200	CALL <JMP.&msvcrt.__p__fmode>	msvcrt.__p__fmode
00401215	CALL <JMP.&msvcrt.__p__environ>	msvcrt.__p__environ
00401239	CALL <JMP.&msvcrt._cexit>	msvcrt._cexit
00401241	CALL <JMP.&KERNEL32.ExitProcess>	KERNEL32.ExitProcess
00401256	CALL <JMP.&msvcrt._setmode>	msvcrt._setmode
00401270	PUSH EBP	(Initial CPU selection)
0040127D	CALL DWORD PTR [<&msvcrt.__set_app_type	msvcrt.__set_app_type
0040129D	CALL DWORD PTR [<&msvcrt.__set_app_type	msvcrt.__set_app_type
0040134C	CALL <JMP.&msvcrt.fputs>	msvcrt.fputs
00401369	CALL <JMP.&msvcrt.fgets>	msvcrt.fgets
0040139A	CALL <JMP.&msvcrt.fputs>	msvcrt.fputs
004013BE	CALL <JMP.&msvcrt.fputs>	msvcrt.fputs
004014AE	CALL <JMP.&msvcrt.strncpy>	msvcrt.strncpy
004014C8	CALL <JMP.&msvcrt.strcmp>	msvcrt.strcmp
004014EF	CALL <JMP.&msvcrt.time>	msvcrt.time
004014FA	CALL <JMP.&msvcrt.localtime>	msvcrt.localtime
0040151F	CALL <JMP.&msvcrt.strftime>	msvcrt.strftime
0040152A	CALL <JMP.&msvcrt.free>	msvcrt.free
004016AF	CALL DWORD PTR [<&KERNEL32.GetAtomNameA	KERNEL32.GetAtomNameA
004017D7	CALL DWORD PTR [<&KERNEL32.FindAtomA>]	KERNEL32.FindAtomA
004017F8	CALL <JMP.&msvcrt.malloc>	msvcrt.malloc
004018FF	CALL DWORD PTR [<&KERNEL32.AddAtomA>]	KERNEL32.AddAtomA
00401918	CALL <JMP.&msvcrt.free>	msvcrt.free
00401920	CALL DWORD PTR [<&KERNEL32.FindAtomA>]	KERNEL32.FindAtomA
00401960	CALL <JMP.&msvcrt.abort>	msvcrt.abort
004019CD	CALL <JMP.&msvcrt.fprintf>	msvcrt.fprintf
004019DD	CALL <JMP.&msvcrt.fflush>	msvcrt.fflush
004019E2	CALL <JMP.&msvcrt.abort>	msvcrt.abort

3. I notice right away that the "strcmp" or string copy function is used within this binary. My suspicions are that my entered password is compared against the true password. I sent a break point and enter in a dummy password to see if the breakpoint was triggered.
4. The breakpoint is triggered and I noticed that before the strcmp function is called, the arguments on the stack are as follows:



5. My suspicions are that the true password is "Crack04." Let's try entering in that password:



Finished.

Final Remarks:

This Crack-Me was pretty easy. A good first step into the Crack-Me game and I did enjoy doing this one a lot.