R·I·T	Rochester Institute of Technology Golisano College of Computing and Information Sciences Department of Computing Security

Professor Jonathan S. Weissman CSEC 466 Debugging 101

Name: Smayan Daruka
Date: 10/23/18

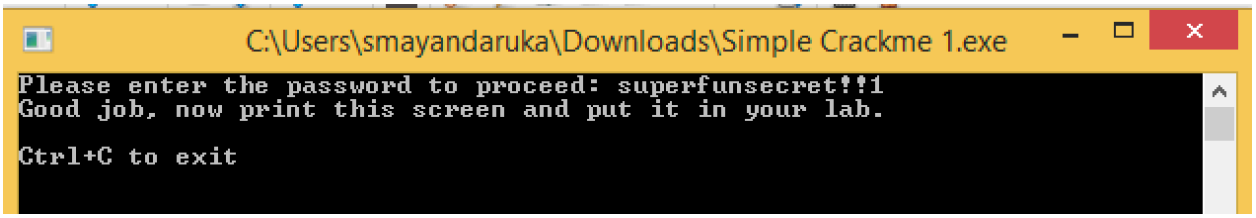
Requirements:

- Lab executables
- x32dbg

Challenges:

Use x32dbg to find the password for the console application entitled 'Simple Crackme 1'. Select 'Run' (press F9) to start the program. Review the program and figure out where to set breakpoints at strategic locations to crack the password that the application is looking for.

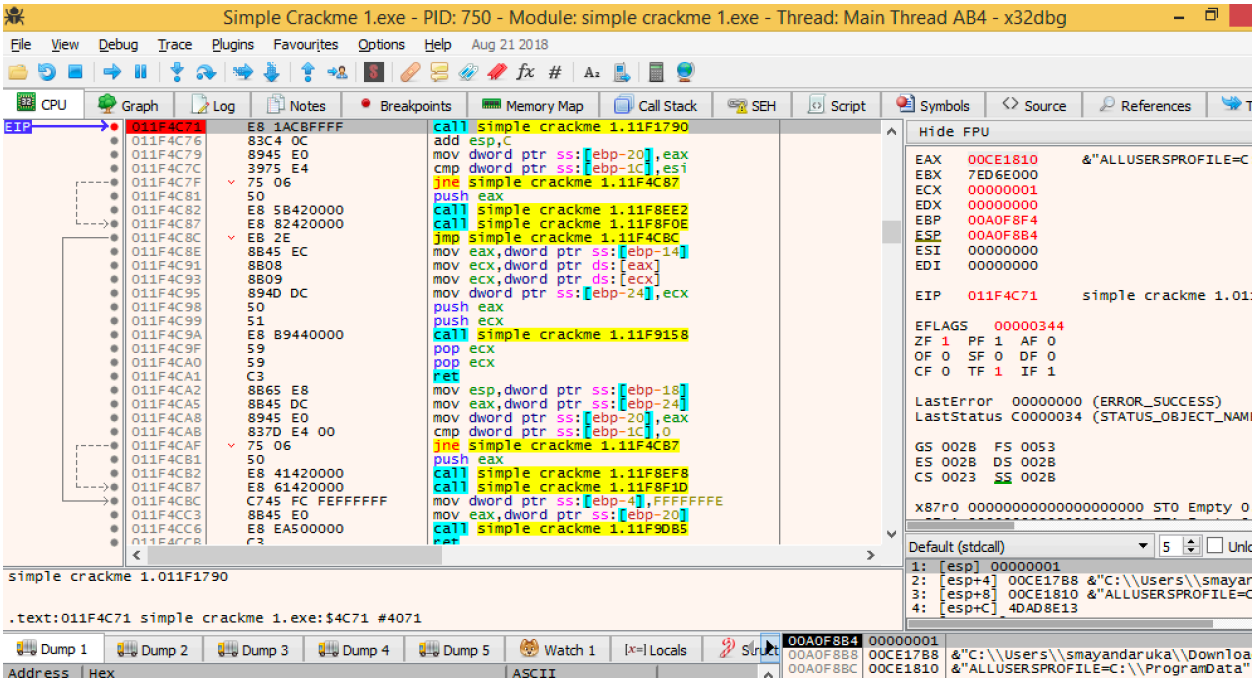
1. Paste the screenshot of the success window below:



2. Explain your methodology in how you were able to locate the password ***using x32dbg***.

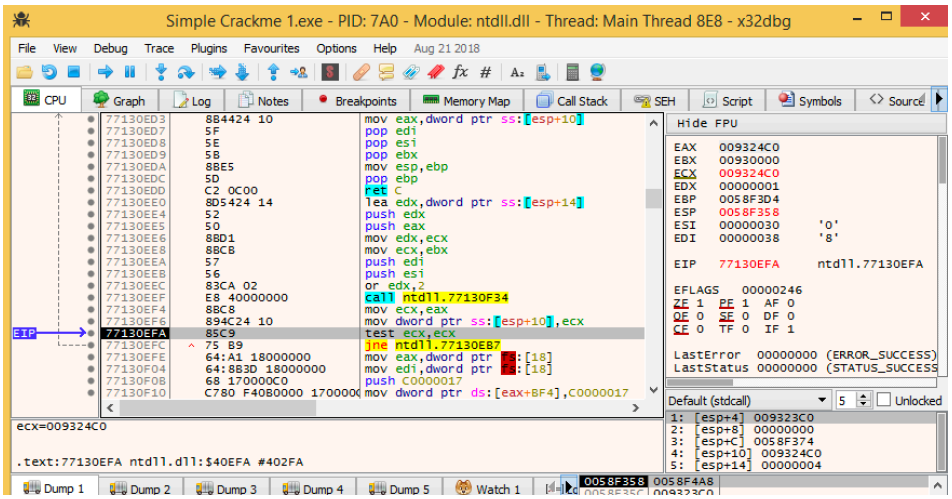
I looked at each comparison that was being made to understand the characters that it's being compared to. I set multiple breakpoints in the program to make it easy to skip over some instructions. I also looked at all the strings that were in the program using x32dbg and was able to successfully find the password after all this.

3. Create a breakpoint near the password comparison routine, and then take a screenshot of the routine in x32dbg.



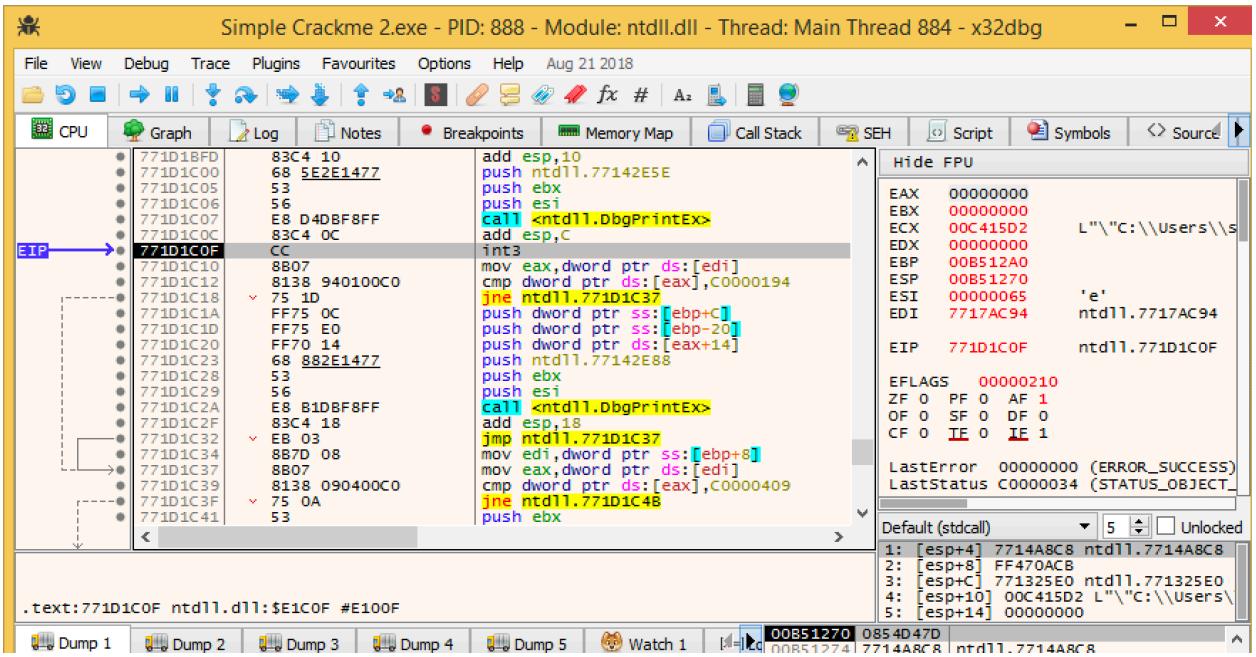
4. How could you use x32dbg to trick the application into thinking it received the right password (regardless of input)? Explain how in as detailed a manner as possible and provide screenshots to support your answer.

The easiest way to accomplish this is by changing the instruction that prompts for user input to NOP so the whole password verification is bypassed. I was able to find the actual instruction which prints whether the password verification succeeded or not to the console and bypassed its previous verification.



Use x32dbg to understand how keys are generated for the console application entitled 'Simple Crackme 2'.

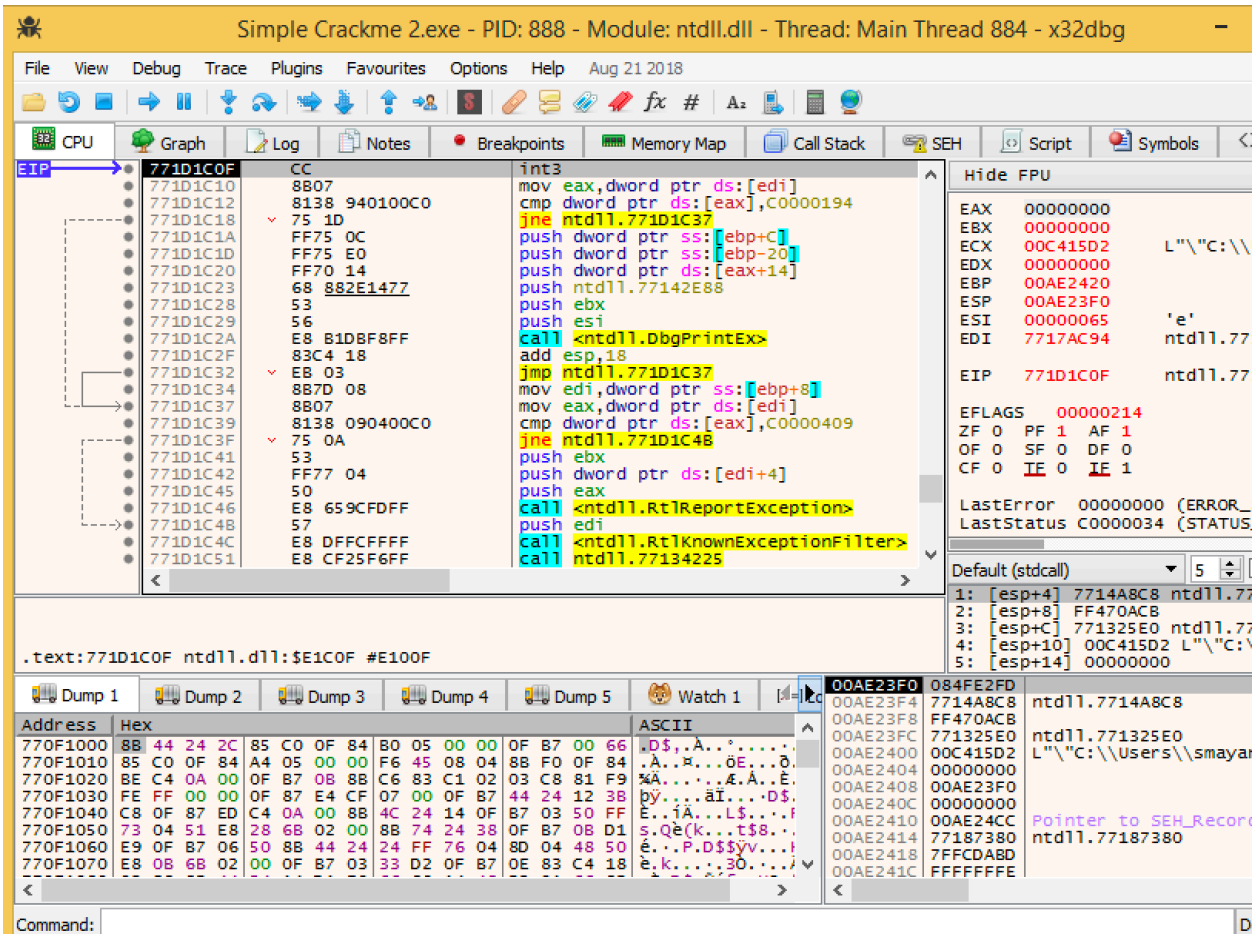
5. Fully explain how the key is generated using the input you provided. Please provide as thorough of an explanation as possible citing resources such as MSDN to support your claims.



This is the main computation that is being done to calculate the key. Keys are generated using a secret value or a license key seed. There is a content packager that generates a key at the time it is encrypted. The key ID is included in the header of the file.
https://msdn.microsoft.com/en-us/library/windows/desktop/bb614583(v=vs.85).aspx

6. Explain how you might use the debugger to manipulate the outcome of the computed result (be creative).
We can use the debugger to see every calculation done to compute the key and thus can manipulate it as we wish. We can see the outcome of every instruction executed and with the ability to manipulate instructions during runtime, we can change instructions to see the effect our manipulations will have on the output.

7. Try your answer to the previous question to make the output the number 1337. Paste a screenshot of the console window demonstrating this. Also, paste a screenshot of the changes you made in x32dbg to make this happen.



Use x32dbg and documentation available on MSDN to find the APIs responsible for writing output to the command prompt.

8. What's the API used to do this?
The API used is WriteFile.