Harder Crackmes

Joshua Reynolds
NMSU Reverse Engineering
Spring 2024

Crackme1 - Rock-Paper-Scissors-Cracker-CrayCray

Identify your source and sink. Where are you trying to get to?

As you walk through each function, keep track of the rules your keygen must follow to be valid.

For example, you might find rules like these:

> char 17 XOR char 12 must equal “x”
>
> char 0 must be even when read as an integer

Test your rules, see how far you get based on the print statements

Remember that a good keygen produces valid keys, where possible. Meaning that it should not just output one static string that works.

Crackme 2 - Written in C++

This crackme is written in C++

The decompilation may look more confusing than a C program

Stream operators in C++ are actually functions

The equivalent of a StringBuilder in Java is stringstream

Watch out that they can convert types (as in ATOI) at the same time as they consume from streams like cin, which wraps **stdin**

Pay attention to the control flow, and use the error messages to check how far you've come.

Crackme 3 - Two interacting inputs

This crackme asks for two inputs, a “pseudo” and a “clef”

The correct “clef” depends on the chosen “pseudo”.

Your keygen must produce a new pseudo and working clef for that pseudo every time it is run for full credit.

I suggest using GDB to help you step through and see how each character of the “clef” is checked.

You can use addresses in Ghidra or IDA to add breakpoints in GDB since there are no debugging symbols to help you.

You can run a program in GDB with the command: **run [arg1] [arg2]**

Crackme 4 - Windows

Be sure to read the README. The keygen, once you figure it out, is relatively simple. So, you also need to explain *why* it works and how you figured that out.

This program has a GUI. If you follow it from the entry point it creates window components, and then waits till the window is closed.

To find what is done with the input, you will have to find another function that is called as the handler when a key is submitted using the GUI.

Another clue that can be helpful is using CFFExplorer to look at the resources.

Crackme 5 - A Crackme that Fights Back

Before running this crackme, make a backup – it deletes itself!

You are allowed to patch this crackme, and include in your report how you changed it.

In Ghidra, for example, you can right click on assembly instructions and “patch” them into something else. Then you can export the program in binary format, preserving your changes.

The keygen for this crackme depends on the PID number of the crackme when it’s running. You can either patch this out, or have your keygen program run the crackme, check its PID, and then generate a correct key.

Let me know in your report which route you took. If you patched it out, describe how.