

Forum: [Hack Forums](#)

Thread: [Finding address of function to patch](#)

thewhitewhale - October 29, 2018, 9:38 am

Hi guys,

I have a basic understanding of assembly/C/C++ from university courses, but I haven't learned to apply them to reverse engineering yet. I was hoping I could get pointed in the right direction as to the general workflow of finding the address of a function to patch - assuming I know the function's name from the source code of a similar project, what it does, and that the actual patch is brain-dead easy (just overwrite the function call with NOP once I find it). I'm pretty lost because I don't even know which tool I should use or even what search terms I should try with Google.

Or maybe my Google-fu's just not up to snuff.

CyberStrike - October 29, 2018, 11:15 am

I am sure that much more experienced members will comment as well but at least a couple of resources to help

<https://marcoramilli.blogspot.com/2011/01/how-to-patch-binary-with-ida-pro.html>

<https://oalabs.openanalysis.net/2017/10/22/quick-and-dirty-binary-patching-with-ida-and-a-hex-editor/>

Recently I ran into malware which had one function calling to various other functions to do some checks so checking the xrefs (cross references) should be done since you are changing the execution flow of the program.

Twistd - October 29, 2018, 12:51 pm

Ollydbg, Ida pro, x64dbg will be your best friend.

Infamia - October 30, 2018, 7:17 am

if you know where the function is called, i.e. within the main entry point, simply just look at all the called functions; and check whether the function's disassembly matches up with what you have in mind.

if the program wasn't compiled with debugging symbols of any sort, and the functions are still intact with any unrelated library functions still named accordingly (import segment intact), then it should be fairly easy relative to the size of the codebase.