

Forum:[Hack Forums](#)

Thread: [Obfuscating C++ / Development](#)

Z3phyru5 - December 4, 2019, 11:13 am

Hello, HF people,

This is my first post here, I hope I'm not breaking any rules.

So, as the thread title might suggest, I'm working on my own Crypter (Original, I know). I took the project in order to learn C++ and expand my knowledge into the world of Malware development as currently, I know very little of it. My Crypter right now is functional, it does the traditional RunPE as well as Process Hollowing. I intend to code it in a way that it would be able to run all the different injection techniques.

However, I was very keen on testing it out and the detection rates are crazy, 14/70 in Virus Total. Yeah, I know I should not be testing there, however, I'm not developing it with any malicious intent and I was not able to locate any other good scanners besides HybridAnalysis, which, also, is a culprit of distribution. Maybe you could recommend some as offtopic?

Anyways, due to detection rates I started looking into Obfuscation of the code and I do not seem to find any good information on it. I scoured these forums but most topics are for .NET apps and they usually end up using ConfuserEx, however, nothing for pure C++, so I thought maybe you guys could help me out with that? I was not able to find what I am looking for on GitHub as well.

Any articles, tools, Git repos, source codes or general knowledge/tips on obfuscation development would be greatly appreciated. Also, all the tips for catching detection and eliminating it would be great as well!

As I understand the main reason the detections are happening is due to visible Windows API calls. What is the way about hiding those?

Best regards!

SIM-3602340 - December 4, 2019, 11:25 am

How are you calling the windows api right now? Are you loading the DLLs at runtime, are you linking them to the binary, etc. ?

P.S. I am working on a crypter too, for educational purposes ofc, we can exchange ideas if you want

Z3phyru5 - December 5, 2019, 3:21 am

Correct me if I did not understand your question properly, I'm very new to C++ still. As of right now, the only .DLL that I explicitly load is the ntdll.dll for the "NtUnmapViewOfSection" function. This is being included at runtime. Besides that every other WinAPI call comes from including "Windows.h" to the project.

SIM-3602340 - December 5, 2019, 6:58 am

(Dec 5, 2019 04:21 AM)Z3phyru5 Wrote: [\[CITE\]](#) Besides that every other WinAPI call comes from including "Windows.h" to the project.

By doing this, all the APIs you are calling this way will be visible in the executable imports.

The problem with this is that those imports are visible even with a basic static analysis, so without running the program it's possible to know which apis the program will be calling during runtime.

Try to call all the APIs by getting their addresses at runtime (for example with LoadLibrary/GetModuleHandle in combination with GetProcAddress), but this won't totally hide the calls because 1) The dll is still mapped in your process 2) A lot of runtime scanners can hook the function calls.

Z3phyru5 - December 5, 2019, 10:02 am

Oh, oooh. Now that you mention it, I have seen a very interesting repository which might be related to it.

Could this actually help with the process of hooking the required modules?

Sorry for the weird link, I cannot post those until I "level up".

[github\(dot\)com/JustasMasiulis/lazy_importer](https://github.com/JustasMasiulis/lazy_importer)

Dark Spider - December 5, 2019, 10:10 am

(Dec 5, 2019 11:02 AM)Z3phyru5 Wrote: [\[CITE\]](#) Oh, oooh. Now that you mention it, I have seen a very interesting repository which might be related to it.

Could this actually help with the process of hooking the required modules?

Sorry for the weird link, I cannot post those until I "level up".

[github\(dot\)com/JustasMasiulis/lazy_importer](https://github.com/JustasMasiulis/lazy_importer)

this looks very interesting ?

PHP Code

<?php

```
LI_FN(OutputDebugStringA)("hello world");
```

```
LI_FN(VirtualProtect).in(LI_MODULE("kernel32.dll").cached());
```

Z3phyru5 - December 5, 2019, 10:47 am

Do you reckon this could be the way to go?

SIM-3602340 - December 5, 2019, 10:59 am

Try it out, shouldn't be so hard

Z3phyru5 - December 5, 2019, 11:42 am

Yeah, true that. Anyways, I'll let you know if it turns out into something nice! Cheers!

__arkr - December 6, 2019, 11:29 pm

Great post so far, I might just add this mind-map of different protection options, [lurumdare \(dot\) github \(dot\) io/ideas/](#)