

Forum:[Hack Forums](#)

Thread: [Alternatives to WriteProcessMemory to make Process Hollowing FUD \(or alternatives\)](#)

hansbald - May 19, 2018, 6:57 am

Hello,

I am pretty much stuck at this problem. My crypter stub is scantime FUD, runtime I get a IPD.Generic detection.

By commenting out every line of my injection routine I noticed the problem is WriteProcessMemory.

I just can't seem to bypass it. I already load the Function dynamically, I tried delays, restructuring code flow, doing something else before calling WriteProcessMemory, always the same detection at that one line.

Is there something I'm missing? Is Proccess Hollowing (aka runpe) even used today? If not what is? What can I research to solve my problems?

I dont want to copy paste code, I just want some pointers were to look :)

Thanks alot!

narc0tic - May 19, 2018, 7:19 am

Loading dynamically and delays will not do anything.

You need to bypass the hooks that AV's put onto your process.

You can do this by unhooking the functions or using ntdll functions by manually mapping it or using direct syscalls.

hansbald - May 19, 2018, 7:38 am

(May 19, 2018 07:19 AM)narc0tic Wrote: [\[CITE\]](#) Loading dynamically and delays will not do anything.

You need to bypass the hooks that AV's put onto your process.

You can do this by unhooking the functions or using ntdll functions by manually mapping it or using direct syscalls.

By manually mapping do You mean mapping them to your own Adresse space temporarily?

gigajew - May 19, 2018, 9:32 am

Look into ntdll WriteVirtualMemory

spiderfruit - May 19, 2018, 10:00 am

(May 19, 2018 09:32 AM)gigajew Wrote: [\[CITE\]](#) Look into ntdll WriteVirtualMemory
ntwritevirtualmemory is just a wrapper for writeprocessmemory

(May 19, 2018 06:57 AM)hansbald Wrote: [\[CITE\]](#) runtime I get a IPD.Generic detection.
you mean idp generic?

anways calling wpm dynamically won't help you on runtime as some some anti-viruses implement callbacks/hooks on these api calls. meaning they will be alerted even if you use getprocaddr :p

hansbald - May 19, 2018, 12:33 pm

(May 19, 2018 10:00 AM)spiderfruit Wrote: [\[CITE\]](#) ntwritevirtualmemory is just a wrapper for writeprocessmemoryYeah I know thats why I didnt even bother with that ^^

(May 19, 2018 10:00 AM)spiderfruit Wrote: [\[CITE\]](#) you mean idp generic?
anways calling wpm dynamically won't help you on runtime as some some anti-viruses implement callbacks/hooks on these api calls. meaning they will be alerted even if you use getprocaddr :pYes
IDP Generic sorry
I am currently trying to load a copy of ntdll.dll into my own process, without LoadLibrary and GetProcAddress, but its rather hard I must say

Hoodie+ - May 19, 2018, 3:43 pm

There are a few tricks to achieve WriteProcessMemory using other WinAPI functions, look it up online.

hansbald - May 20, 2018, 11:44 am

Well it seems like I suck too much at the moment, I just cant get it to work, the DLL seems to load fine, and it seems to get the Address of the function just fine but then it just crashes, debugger isnt helping since it cant see the loaded DLL ^^ I will rethink it and try again

chrizator - May 21, 2018, 6:10 am

You can even load Kernel32 (Kernelbase) dynamically in order to call GetModuleHandle() or LoadLibrary(). This is usually done by taking advantage of the fact that the PEB (Process Environment Block), created with each process, holds a list of modules, loaded in the address space of the process. You can take a look at the _PEB structure yourself to learn more about it.

Furthermore, you can implement GetProcAddress() yourself (there is probably a lot of source code on the web).

Those two techniques combined can result in the AV being unable to hook your API calls.

Hoodie+ - May 21, 2018, 3:55 pm

(May 21, 2018 06:10 AM)chrizator Wrote: [\[CITE\]](#) You can even load Kernel32 (Kernelbase) dynamically in order to call GetModuleHandle() or LoadLibrary(). This is usually done by taking advantage of the fact that the PEB (Process Environment Block), created with each process, holds a list of modules, loaded in the address space of the process. You can take a look at the _PEB structure yourself to learn more about it.

Furthermore, you can implement GetProcAddress() yourself (there is probably a lot of source code on the web).

Those two techniques combined can result in the AV being unable to hook your API calls.
Inline hooking maybe, syscall hooking no...

At the end of the day, you're calling the same function that will get you detected.

spiderfruit - May 22, 2018, 12:26 am

(May 21, 2018 06:10 AM)chrizator Wrote: [\[CITE\]](#) You can even load Kernel32 (Kernelbase) dynamically in order to call GetModuleHandle() or LoadLibrary(). This is usually done by taking advantage of the fact that the PEB (Process Environment Block), created with each process, holds a list of modules, loaded in the address space of the process. You can take a look at the _PEB structure yourself to learn more about it.

Furthermore, you can implement GetProcAddress() yourself (there is probably a lot of source code on the web).

Those two techniques combined can result in the AV being unable to hook your API calls.
how dumb do you think AV's are..?

while you could lurk through IMAGE_DATA_DIRECTORY and write custom GPA it won't help

much.

these days anti-viruses use lots of techniques to detect viruses using heuristics etc..

they use global hooks from kernel land (ring 0) and monitor any suspicious activity (e.g.

unauthorized process using openprocess/read and write to memory/createfile/send/recv..) and in most cases they will block these calls before your software will execute them.

@op

it's not as easy as you think, bypassing runtime.. your best bet on avoiding detection is- just don't tamper with it meaning write code which AV cannot/will not pick.

fyi, I am not saying there's no alternative way to achieve what you want. of course there's it's just requires more knowledge/researching

HiddenEye - May 26, 2018, 4:29 am

interesting topic.

OP your best bet it is more knowledge/researching with trial and error

e.g i did a test with public runpe code= 2009 and got it FUD with 3 lines of code and little modification

XMR Miner - May 29, 2018, 7:50 pm

Like someone else said, your best bet to bypass user-mode hooking (WriteProcessMemory detection) is using ntdll functions from manually mapped ntdll or using direct syscalls. An alternative to manually mapping ntdll from disk is copying it in memory from the knowndlls section.