

Forum: [Hack Forums](#)

Thread: [\[c++\] unhook wininet](#)

neonbytes - March 15, 2024, 12:18 pm

hey guys

on windows c++, any idea on how to:

a. unhook wininet dll functions

or

b. use syscalls to call wininet dll counterparts on the kernel [which i don't know how to find]

or any other tip to make wininet http requests stealthier to evade avs?

thanks in advance

an4l 1st - March 15, 2024, 2:52 pm

(Mar 15, 2024 12:18 PM)neonbytes Wrote: [\[CITE\]](#) hey guys

on windows c++, any idea on how to:

a. unhook wininet dll functions

or

b. use syscalls to call wininet dll counterparts on the kernel [which i don't know how to find]

or any other tip to make wininet http requests stealthier to evade avs?

thanks in advance

look, most popular antiviruses like avast and windows defender use kernel mode drivers to monitor the network...

so if you are transferring data without encryption, unhooking wininet won't really do much for you...

also notice that antiviruses like avast, will install a root authority certificate on your machine and perform a mitm attack to decrypt https communication...

so if you want to keep your communication safe, you need to implement an additional layer of encryption to your data... because you can't rely on ssl/tls

wininet won't have syscalls because it is not located on ntdll.dll and if you are going to go down that path to try to use native functions, you are better using some custom communication library like boost best (asio)

<https://github.com/boostorg/beast>

but that is quite hard to implement and make it work... at least I think so... if anyone has an easy to implement solution let us know...

in the end of the day you are better using some raw sockets with aes... to send http data... it is simple but quite effective...

also relying on domains is quite of a problem since dns functions are really simple to block/blacklist for antiviruses ...

dga can be effective.. also other ways of resolving dns like dns over https can help a lot..
that is why I just use python... anyway, I am happy the russians don't understand my point... so I can just keep using it without being annoyed so much by security researchers...

I guess the good thing of being ahead of the curve is that even if you clearly state the reasons why a certain solution is better, people still won't accept it..
it is just like when I said 10 years ago to my colleagues that anyone who bought btc was gonna get rich... because clearly the decentralization and the deflationary currency was a better solution to money...

I am getting off topic already... kek.. that is the price of getting old... I am always trying to look at the bigger picture..

neonbytes - March 15, 2024, 3:36 pm

(Mar 15, 2024 02:52 PM)an4l 1st Wrote: [\[CITE\]](#) look, most popular antiviruses like avast and windows defender use kernel mode drivers to monitor the network...

so if you are transferring data without encryption, unhooking wininet won't really do much for you... also notice that antiviruses like avast, will install a root authority certificate on your machine and perform a mitm attack to decrypt https communication...

so if you want to keep your communication safe, you need to implement an additional layer of encryption to your data... because you can't rely on ssl/tls

wininet won't have syscalls because it is not located on ntdll.dll and if you are going to go down that path to try to use native functions, you are better using some custom communication library like boost best (asio)

<https://github.com/boostorg/beast>

but that is quite hard to implement and make it work... at least I think so... if anyone has an easy to implement solution let us know...

in the end of the day you are better using some raw sockets with aes... to send http data... it is simple but quite effective...

also relying on domains is quite of a problem since dns functions are really simple to block/blacklist for antiviruses ...

dga can be effective.. also other ways of resolving dns like dns over https can help a lot..
that is why I just use python... anyway, I am happy the russians don't understand my point... so I can just keep using it without being annoyed so much by security researchers...

I guess the good thing of being ahead of the curve is that even if you clearly state the reasons why a certain solution is better, people still won't accept it..
it is just like when I said 10 years ago to my colleagues that anyone who bought btc was gonna get rich... because clearly the decentralization and the deflationary currency was a better solution to money...

I am getting off topic already... kek.. that is the price of getting old... I am always trying to look at the bigger picture..
thank you for the extended reply

"look, most popular antiviruses like avast and windows defender use kernel mode drivers to monitor the network...

so if you are transferring data without encryption, unhooking wininet won't really do much for you...

also notice that antiviruses like avast, will install a root authority certificate on your machine and perform a mitm attack to decrypt https communication...

so if you want to keep your communication safe, you need to implement an additional layer of encryption to your data... because you can't rely on ssl/tls"

i was mostly looking at how to hide/obfuscate/avoid the winapi calls, but this info is very helpful

"wininet won't have syscalls because it is not located on ntdll.dll and if you are going to go down that path to try to use native functions, you are better using some custom communication library like boost asio

<https://github.com/boostorg/beast>"

will give it a check

"in the end of the day you are better using some raw sockets with aes... to send http data... it is simple but quite effective..."

i could try it out and do some a/b testing

do you think switching to libcurl (on windows) can help? i guess it uses raw sockets under the hood

DarkVision - March 15, 2024, 4:36 pm

A bit OT, but since you mentioned libcurl, let me recommend that you give the WinHttp API a try.

It offers a variety of options for carrying out HTTP and web server related tasks and it can be linked both statically (with a .lib file) and dynamically (with a .dll file), and of course by mapping it manually if necessary.

Phantom_NiNja - March 16, 2024, 12:24 am

(Mar 15, 2024 04:36 PM)DarkVision Wrote: [\[CITE\]](#) A bit OT, but since you mentioned libcurl, let me recommend that you give the WinHttp API a try.

It offers a variety of options for carrying out HTTP and web server related tasks and it can be linked both statically (with a .lib file) and dynamically (with a .dll file), and of course by mapping it manually if necessary.

What is winhttp? Is it part of windows?

Hackers - March 16, 2024, 6:29 am

(Mar 15, 2024 12:18 PM)neonbytes Wrote: [\[CITE\]](#) on windows c++, any idea on how to:

a. unhook wininet dll functions

or

b. use syscalls to call wininet dll counterparts on the kernel [which i don't know how to find]

This doesn't answer your question fully but is a good start:

<https://hackforums.net/showthread.php?tid=6266052>

Here is something I made to unload unwanted dlls, maybe useful:

<https://github.com/LNodesL/Detect-Uknown-DLL-in-Runtime-CPP-Single-File-Header-Include>

(Mar 16, 2024 12:24 AM)randommanty89 Wrote: [\[CITE\]](#) What is winhttp? Is it part of windows?

Here is an example of sending a POST request to a website with WinHTTP , and yes - part of Windows.

If you combine my methods in my first thread you can get a good start going for this :) Just convert to GET request instead of POST probably if you are fetching data - up to you.

```

Code #include <windows.h>
#include <winhttp.h>
#include <iostream>
#pragma comment(lib, "winhttp.lib")
int main() {
    HINTERNET hSession = WinHttpOpen(L"Send Post Request/1.0",
    WINHTTP_ACCESS_TYPE_DEFAULT_PROXY, WINHTTP_NO_PROXY_NAME,
    WINHTTP_NO_PROXY_BYPASS, 0);
    HINTERNET hConnect = WinHttpConnect(hSession,
    L"your-website.com",INTERNET_DEFAULT_HTTPS_PORT, 0);
    HINTERNET hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/post/receive.php",
    NULL, WINHTTP_NO_REFERER, WINHTTP_DEFAULT_ACCEPT_TYPES,
    WINHTTP_FLAG_SECURE);
    std::wstring headers = L"Content-Type: application/x-www-form-urlencoded";
    std::string postData = "something=goes&here=booya";
    BOOL bResults = WinHttpSendRequest(hRequest, headers.c_str(), headers.length(),
    (LPVOID)postData.c_str(), postData.length(), postData.length(), 0);
    if (bResults) bResults = WinHttpReceiveResponse(hRequest, NULL);
    if (hRequest) WinHttpCloseHandle(hRequest);
    if (hConnect) WinHttpCloseHandle(hConnect);
    if (hSession) WinHttpCloseHandle(hSession);
    return 0;
}

```

an4l 1st - March 16, 2024, 7:55 am

(Mar 16, 2024 06:29 AM)nodes Wrote: [\[CITE\]](#) This doesn't answer your question fully but is a good start:

<https://hackforums.net/showthread.php?tid=6266052>

Here is something I made to unload unwanted dlls, maybe useful:

<https://github.com/LNodesL/Detect-Uknown-DLL-in-Runtime-CPP-Single-File-Header-Include>

Here is an example of sending a POST request to a website with WinHTTP , and yes - part of Windows.

If you combine my methods in my first thread you can get a good start going for this :) Just convert to GET request instead of POST probably if you are fetching data - up to you.

```

Code #include <windows.h>
#include <winhttp.h>
#include <iostream>
#pragma comment(lib, "winhttp.lib")
int main() {
    HINTERNET hSession = WinHttpOpen(L"Send Post Request/1.0",
    WINHTTP_ACCESS_TYPE_DEFAULT_PROXY, WINHTTP_NO_PROXY_NAME,
    WINHTTP_NO_PROXY_BYPASS, 0);
    HINTERNET hConnect = WinHttpConnect(hSession,
    L"your-website.com",INTERNET_DEFAULT_HTTPS_PORT, 0);
    HINTERNET hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/post/receive.php",
    NULL, WINHTTP_NO_REFERER, WINHTTP_DEFAULT_ACCEPT_TYPES,
    WINHTTP_FLAG_SECURE);
    std::wstring headers = L"Content-Type: application/x-www-form-urlencoded";

```

```

std::string postData = "something=goes&here=booya";
BOOL bResults = WinHttpRequest(hRequest, headers.c_str(), headers.length(),
(LPVOID)postData.c_str(), postData.length(), postData.length(), 0);
if (bResults) bResults = WinHttpReceiveResponse(hRequest, NULL);
if (hRequest) WinHttpCloseHandle(hRequest);
if (hConnect) WinHttpCloseHandle(hConnect);
if (hSession) WinHttpCloseHandle(hSession);
return 0;
}

```

this has turned up to be a very interesting thread... thank you op and nodes :)

Hackers - March 16, 2024, 8:43 am

(Mar 16, 2024 07:55 AM)an4l 1st Wrote: [\[CITE\]](#) this has turned up to be a very interesting thread... thank you op and nodes :)

Glad you enjoy that. I'm still working on things, currently using this very efficient method of using vs code's built in searcher for type definitions to port them from a core file, to my own file. Still working down the last of the PE methods :)

For something like winhttp you could search for functions & types to copy them over to your custom solution:

Just keep following the types until you have all :)

This is what lets you swap from the full library loading in and instead it uses your own solution to interact with core windows system through your own headers:

```

Code // #include <anything.h>
#include "custom-anything.h"

```

neonbytes - March 18, 2024, 3:10 pm

(Mar 15, 2024 04:36 PM)DarkVision Wrote: [\[CITE\]](#) A bit OT, but since you mentioned libcurl, let me recommend that you give the WinHttp API a try.

It offers a variety of options for carrying out HTTP and web server related tasks and it can be linked both statically (with a .lib file) and dynamically (with a .dll file), and of course by mapping it manually if necessary.

thanks for the info

i'm aware of WinHttp, but from what i read it was more difficult to setup, so i ended up with WinInet i should review it in greater detail, maybe i'm missing something

(Mar 16, 2024 06:29 AM)nodes Wrote: [\[CITE\]](#) This doesn't answer your question fully but is a good start:

<https://hackforums.net/showthread.php?tid=6266052>

Here is something I made to unload unwanted dlls, maybe useful:

<https://github.com/LNodesL/Detect-Uknown-DLL-in-Runtime-CPP-Single-File-Header-Include>

Here is an example of sending a POST request to a website with WinHTTP , and yes - part of Windows.

If you combine my methods in my first thread you can get a good start going for this :) Just convert to GET request instead of POST probably if you are fetching data - up to you.

```
Code #include <windows.h>
#include <winhttp.h>
#include <iostream>
#pragma comment(lib, "winhttp.lib")
int main() {
    HINTERNET hSession = WinHttpOpen(L"Send Post Request/1.0",
    WINHTTP_ACCESS_TYPE_DEFAULT_PROXY, WINHTTP_NO_PROXY_NAME,
    WINHTTP_NO_PROXY_BYPASS, 0);
    HINTERNET hConnect = WinHttpConnect(hSession,
    L"your-website.com",INTERNET_DEFAULT_HTTPS_PORT, 0);
    HINTERNET hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/post/receive.php",
    NULL, WINHTTP_NO_REFERER, WINHTTP_DEFAULT_ACCEPT_TYPES,
    WINHTTP_FLAG_SECURE);
    std::wstring headers = L"Content-Type: application/x-www-form-urlencoded";
    std::string postData = "something=goes&here=booya";
    BOOL bResults = WinHttpSendRequest(hRequest, headers.c_str(), headers.length(),
    (LPVOID)postData.c_str(), postData.length(), postData.length(), 0);
    if (bResults) bResults = WinHttpReceiveResponse(hRequest, NULL);
    if (hRequest) WinHttpCloseHandle(hRequest);
    if (hConnect) WinHttpCloseHandle(hConnect);
    if (hSession) WinHttpCloseHandle(hSession);
    return 0;
}
thank you for your reply and example
```