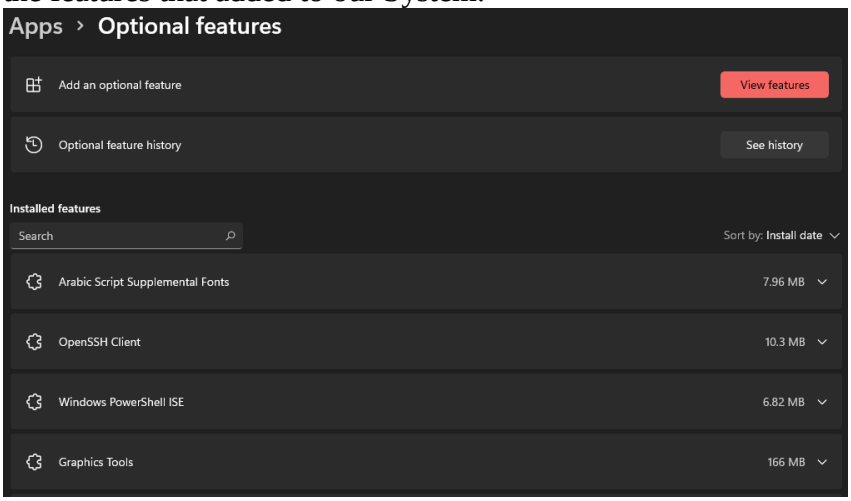
Bypassing UAC "User Access Control" for Windows 10 and Above Using FODHelper

Introduction
Windows OS has many techniques to bypass the UAC window some of them are been patched after updates and some still working. I will discuss how we can use a component called FODHelper to bypass the UAC window and run anything as an administrator.

What is windows UAC?
UAC stands for "User Account Control" which is a fundamental component in the window used for security purposes to let the user accept or decline running executable as administrator.

What is FODHelper?
FODHelper stands for "Feature on Demand Helper" it is one of the Windows OS features used to add features for windows at any time. That includes Language resources like voice recognition and handwriting or Print Management features, Graphic tools. We can use it if you go to "Windows Settings" and then click on "Apps" after that select "Optional features". We can see the features that added to our System.

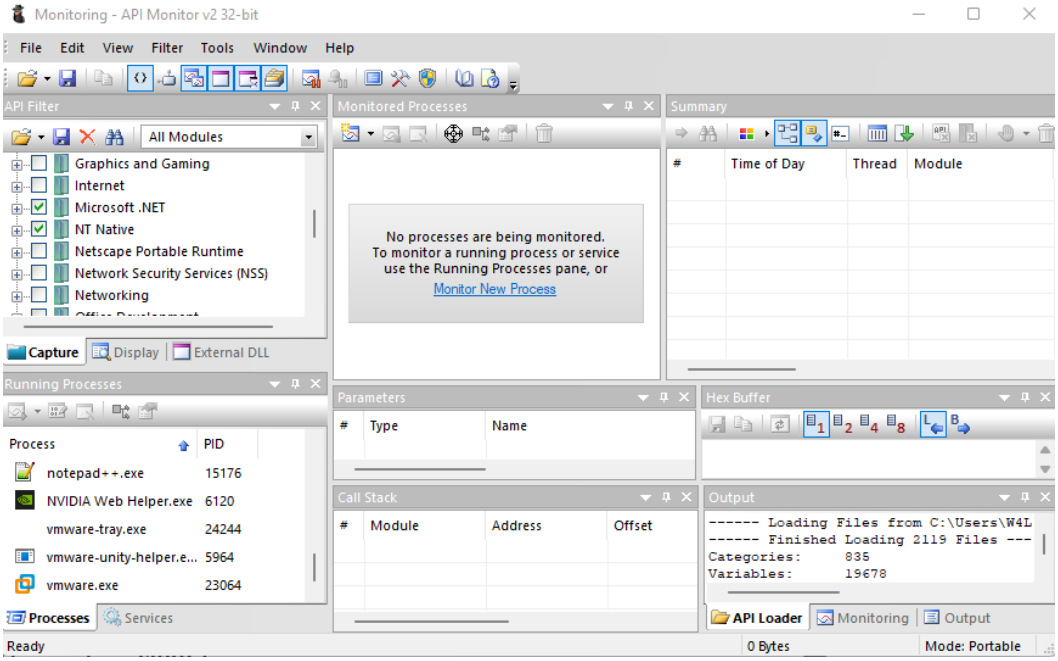


Why and how we choose FODHelper.
For example, if we try to run CMD "Command Prompt", we can see this window pop up:

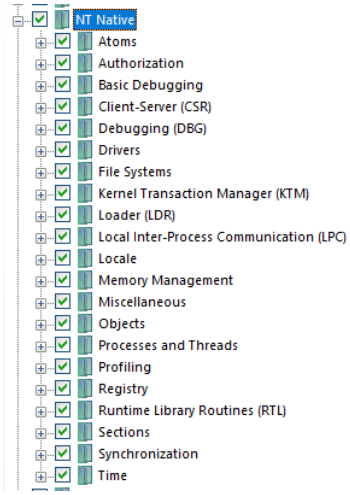
But this window is not shown when we try to open "*C:\Windows\System32\fodhelper.exe*" Because of that, we will see how we can exploit FODHelper to open our program as administrator without user acceptance. Therefore, we try to find if there was a way that we can let FODHelper run our program with FODHelper privileges.

What are the steps we should take to know how the FODHelper works?
We will use a program called "Api Monitor" to monitor FODHelper behaviour and know the path that is used inside FODHelper and the registry keys etc.

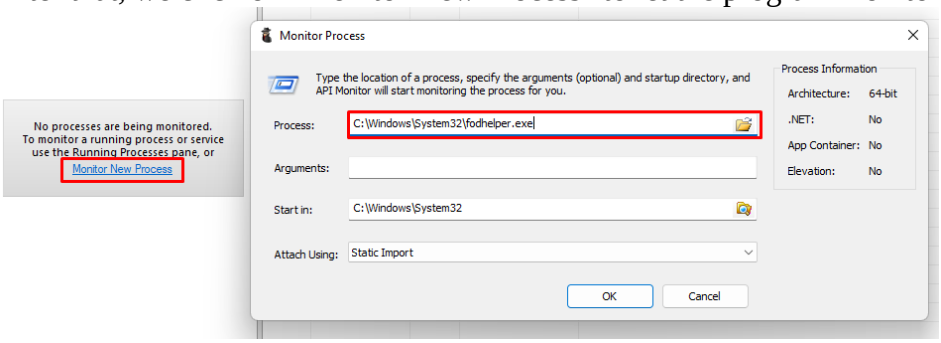
How to use API monitor?
We can download it from this website: *http://www.rohitab.com/apimonitor*
Then after installing it, we run it as administrator then we will see this window



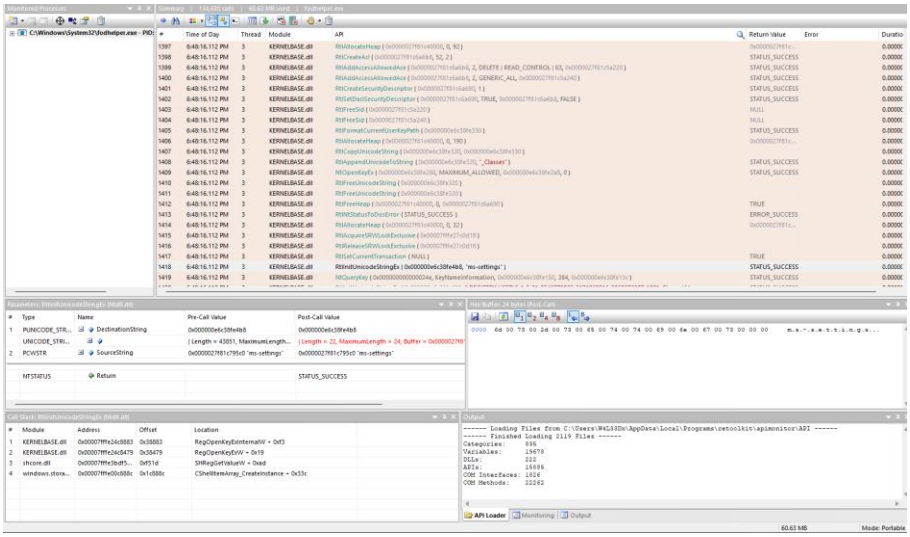
We should select the Modules that we want the program to monitor.
Select the NT Native module because it has most of the things we need as shown:



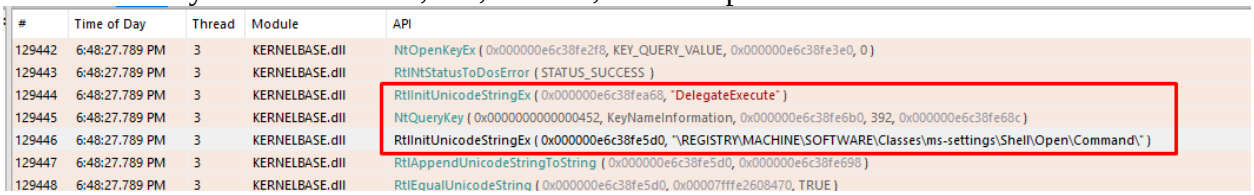
After that, we click on "Monitor New Process" to let the program monitor FODHelper:



Then click ok to see the results:

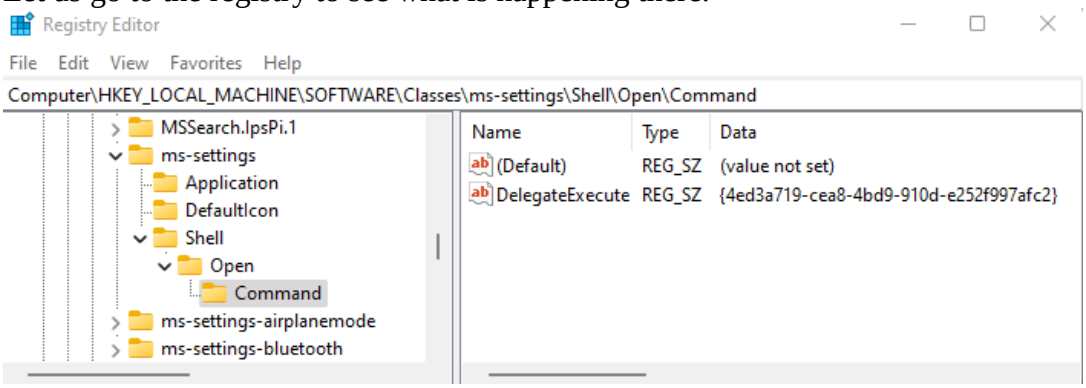


We are now searching for anything that has been running using FODHelper.
We can use keywords like "shell, run, execute, start and process" to make the search easier.

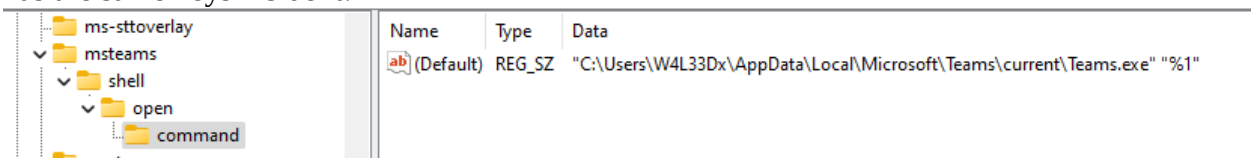


#	Time of Day	Thread	Module	API
129442	6:48:27.789 PM	3	KERNELBASE.dll	NtOpenKeyEx (0x000000e6c38fe2f8, KEY_QUERY_VALUE, 0x000000e6c38fe3e0, 0)
129443	6:48:27.789 PM	3	KERNELBASE.dll	RtlNtStatusToDosError (STATUS_SUCCESS)
129444	6:48:27.789 PM	3	KERNELBASE.dll	RtlInitUnicodeStringEx (0x000000e6c38fea68, "DelegateExecute")
129445	6:48:27.789 PM	3	KERNELBASE.dll	NtQueryKey (0x0000000000000452, KeyNameInformation, 0x000000e6c38fe6b0, 392, 0x000000e6c38fe68c)
129446	6:48:27.789 PM	3	KERNELBASE.dll	RtlInitUnicodeStringEx (0x000000e6c38fe5d0, "\REGISTRY\MACHINE\SOFTWARE\Classes\ms-settings\Shell\Open\Command\")
129447	6:48:27.789 PM	3	KERNELBASE.dll	RtlAppendUnicodeStringToString (0x000000e6c38fe5d0, 0x000000e6c38fe698)
129448	6:48:27.789 PM	3	KERNELBASE.dll	RtlEqualUnicodeString (0x000000e6c38fe5d0, 0x00007fffe2608470, TRUE)

We found this registry path: "*\REGISTRY\MACHINE\SOFTWARE\Classes\ms-settings\Shell\Open\Command*"
And above it key called "*DelegateExecute*"
Let us go to the registry to see what is happening there:

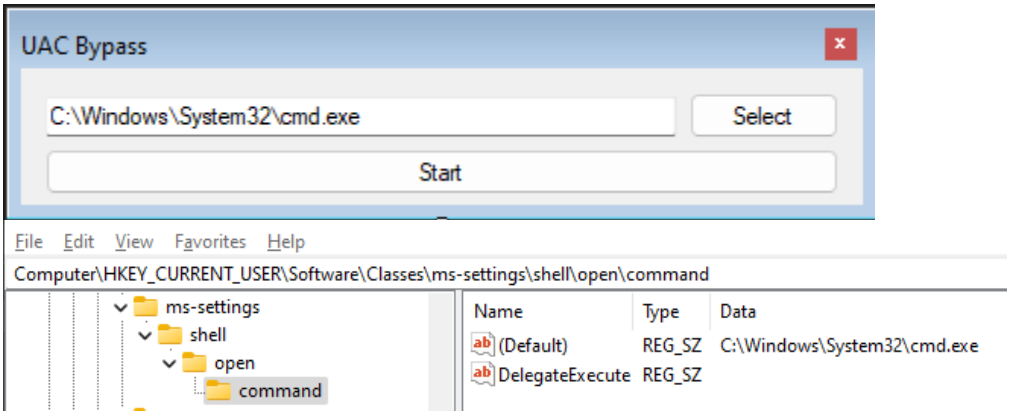


As we see, there are two values inside the command key. The next step is to let see if there is another key have the same name in the registry. For example, I found a key for Microsoft Teams has the same keys inside it:

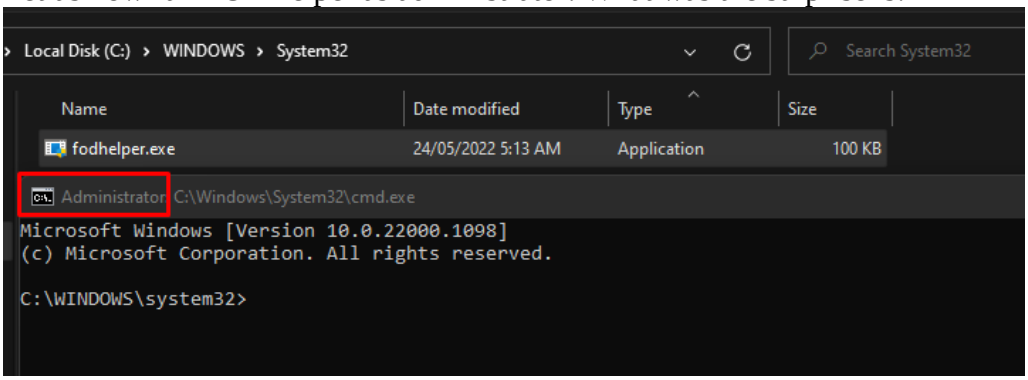


As we see, it uses the default string for the path of the executable file for teams.
What we can do now? Let us try to edit the value of the FODHelper registry to see if we can run our program using it.
I made a small script in VB.NET to create a new value of this key:

```
"Dim registryKey As RegistryKey = Registry.CurrentUser.CreateSubKey("SOFTWARE\Classes\ms-settings\shell\open\command")
registryKey.SetValue("", "C:\Windows\System32\cmd.exe", RegistryValueKind.String)
registryKey.SetValue("DelegateExecute", "", RegistryValueKind.String)"
```



Let us now run FODHelper as administrator. What was the surprise is:



So now, we know that we can run anything using this. However, for cleaning purposes, we need to delete the keys from the registry because every time we run FODHelper or the windows it runs it we will see the CMD appear to us and we do not want that to happen.
I made a full exploit in VB.NET to run any program as an administrator using this vulnerability.

GitHub Full POC:
https://github.com/W4L33Dx/FODHelper-UAC-Bypass

Conclusion

This is just one way to bypass the UAC window as we said before there are too many techniques to bypass. From this, explain we can try to use the same method to find any other component that has the same feature by using the steps we learned:

1- Run the program from System32 folder as administrator to see if the UAC window pops up or not.
2- Try to monitor the process to see if was there anything running using the program.
3- Search using keywords about the important thing that we want.
4- Try to edit the values or pass a path to any command to see if it works.
5- Make a small script to not repeat the steps each time you want to bypass.
6- Clean up any waste after performing the process to ensure the stability of the system.

References

https://attack.mitre.org/tactics/TA0004/

https://attack.mitre.org/techniques/T1548/002/

https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/features-on-demand-v2--capabilities?view=windows-11

https://learn.microsoft.com/en-us/windows/security/identity-protection/user-account-control/how-user-account-control-works

https://learn.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-rtlinitunicodestring

http://www.rohitab.com/apimonitor

Author:
Waleed Alharbi
University of Hafr Al-Batin
24\10\2022