National College of Ireland

Configuration Manual

MSc Research Project
Programme Name: EDR Silencers: Prevent, Detect and Remediate

Arghadeep Lahiri
Student ID: 23403888

School of Computing
National College of Ireland

Supervisor: Kamil Mahajan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Arghadeep Lahiri
Student ID:	23403888
Programme:	Msc Cyber Security **Year:** 2025
Module:	Practicum
Lecturer:	Kamil Mahajan
Submission Due Date:	11th December 2025
Project Title:	EDR Silencers: Prevent, Detect and Remediate
Word Count:	1446 **Page Count:** 7

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Arghadeep Lahiri

Date: 11th December 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Arghadeep Lahiri
Student ID: 23403888

1 Introduction:

This manual lays out the steps you need to install, set up, and run the WFP Block Rule Monitoring System that we developed for this project. The app works by spotting misconfigurations and any possibly harmful changes to firewall rules. It does this through checking out the Windows Filtering Platform filters and pinpointing block rules tied to specific executables. The system starts with a baseline of apps that are already blocked. Then it checks later setups against that baseline to catch any odd changes.

The guide here targets academic supervisors, system admins, and examiners. They might want to roll out or check the system in a safe test setup. That could be something like a virtual machine with Windows 10 or Windows 11 running.

2 System Requirements:

The following system requirements are necessary for the reproduction.

2.1 Operating System Requirements:
Windows 10 x 64 or later with minimum 4 gb RAM for faster processing of C++ code.

2.2Software requirements:

- Visual Studio 2022/ Visual Studio 2026. We used 2026 for this project.
- Windows SDK version 10.0.26100.0
- C++ Development tools installed.
- Full administrator privileges for firewall and BFE access.

2.3 Windows Libraries used:

- fwpuclnt.lib
- ole32.lib
- advapi32.lib

These libraries are automatically installed with Windows SDK kit.

3 Project Component:

3.1 MonitorBlockedAppsWithBaseline.exe: It is created using Visual Studio 2026 which creates a baseline of present filters in blocked state. Running it again in a loop will monitor the WFP filters for any new rule config with keyword “Block”. Using BFE API it enumerates

active WFP filters and detect filters that block network access by application path. Output is group wise summarized per application.

3.2 Baseline.txt file: The BlockedAppsBaseline.txt outputs the present filters in the system with a block rule attached to it. The location is generally the E:\Project folder which can be changed according to user preferences. In this case the location was E:\PROJECT\WFPBase+Diff\MonitorBlockedAppsWithBaseline.cpp\x64\Debug\BlockedAppsBaseline.txt.It stores unique application paths for all blocked executables discovered during baseline creation. Ensure that all operation products are installed and then the baseline is created.

4 Installation and Build Instructions:

In Visual Studio 2026 create a new project and name it MonitorBlockessApps or as you prefer and import the WFPHealthMonitor.cpp submitted with the project that contains the C++ code to build the app. From the build option select Build solution to make the application. Ensure under project **properties** you have **Linker**>**Input** has **fwpuclnt.lib;ole32.lib; advapi32.lib** added under **additional dependencies**. The result produces WFPHealthMonitor.exe created in the working directory of the project.

Please note: Compile the code with elevated privileges

5 Configuration and Usage:

5.1 Baseline creation:
Before any further testing is done, a baseline needs to be created with will have the necessary list of all active blocked filters. User needs to ensure they have all important apps installed before creating the baseline.
From an elevated powershell prompt navigate to the location where the application exists. For e.g: E:\PROJECT\WFPHealthMonitor\ WFPHealthMonitor \x64\Debug as shown in the screenshot.

```
PS C:\Windows\system32> cd E:\PROJECT\WFPHealthMonitor\WFPHealthMonitor\x64\Debug\
PS E:\PROJECT\WFPHealthMonitor\WFPHealthMonitor\x64\Debug>
```

Then from the same elevated command prompt run the below

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

PS C:\Windows\system32> cd E:\PROJECT\WFPHealthMonitor\WFPHealthMonitor\x64\Debug\
PS E:\PROJECT\WFPHealthMonitor\WFPHealthMonitor\x64\Debug> .\WFPHealthMonitor.exe --create-baseline
```

This will create the base line by scanning the active WFP filters and extracting executable with block rules. Writes the unque application path into the baseline file and presents a grouped summary for verifcation efficiency.

```
2025-12-11 00:15:01 [INFO] Enumerated 24 BLOCK filters with ALE_APP_ID conditions.
2025-12-11 00:15:01 [INFO] Creating baseline file: BlockedAppsBaseline.txt
2025-12-11 00:15:01 [INFO] Baseline written with 8 unique blocked application paths to BlockedAppsBaseline.txt
```

The output will have an output similar to: "*Baseline written with X blocked application paths*".

5.2 Detection Mode:

After baseline creation normal executions performs a comparison check with the baseline file. Depending on if any new block rule was created or not the application lodges output in 2 situations:
If no new configuration detected, it says ***"[OK] No new applications with BLOCK rules compared to baseline."***. Simultaneously if there is a new filter detected it would alert as

```
[ALERT] New applications with BLOCK rules detected since baseline:
*** NEW BLOCKED APP *** : \Device\HarddiskVolume4\Program Files\App\app.exe
```

Group summary always prints app summary with it's application path and number of block rules.

6 Automation:

The bare minimum automation can be achieved using a scheduled task that could run every day for a certain number of times in 24 hours. This will enable continuous monitoring of the WFP filters stays active. From Windows run open taskschd.msc to open the task scheduler window. Select "Create Task" > General and then select "Run whether user is logged on or not" and "Run with highest privileges". Triggers tab> Daily or every X minutes. Action tab > "Program: MonitorBlockedAppsWithBaseline.exe" ; Start-in:<folder location to the app>"

Once done, an automatic check will be scheduled to monitor for any misconfigured WFP filter with block rule associated to it and alert the user.
Simultaneously create a powershell script named WFPAlertpopup.ps1that will continuously monitor the event viewer application logs with source=WFPMonitor. Automate the process via task scheduler too.
More instructions will follow in the tutorial video of the project.

7 Validation:

To validate how the application is working, user can create a new firewall rule from an elevated powershell window:
"netsh advfirewall firewall add rule name="Block Test App" dir=out action=block program="C:\Windows\System32\Notepad.exe" enable=yes"

The re-run "WFPHealthMonitor.exe" app and it shall output something like this:

*** NEW BLOCKED APP *** :
\Device\HarddiskVolumeX\Windows\System32\notepad.exe

8 Limitations:

Local firewall rules maybe ignored if AllowLocalFirewallRules = False & AllowLocalPolicyMerge = False.
Central policy via GPO or Security solution may override user rules. WFP Operational Log does not reliably log firewall enforcement for all rules hence it becomes necessaryto monitor

the firewall log also. Detection occurs based on a missing or unexpected block rule configuration, rather than working on packet analysis.

9 Security recommendations:

To maintain integrity of the baseline file, store in a protected location with limited access to users. Push the alerts to Windows logs and restrict write permission to System or Administrators. In future consider centralising the baseline file via SIEM forwarding.

References

1. H. C. Junior, "HookChain: A new perspective for Bypassing EDR Solutions," *arXiv.org*, Apr. 04, 2024. https://arxiv.org/abs/2404.16856
2. Chatzoglou, E., Karopoulos, G., Kambourakis, G. & Tsiatsikas, Z., "Bypassing antivirus detection: old-school malware, new tricks," *arXiv preprint arXiv:2305.04149*, May 2023
3. G. Karantzas and C. Patsakis, "An Empirical Assessment of Endpoint Detection and Response Systems against Advanced Persistent Threats Attack Vectors," *Journal of Cybersecurity and Privacy*, vol. 1, no. 3, pp. 387–421, Jul. 2021, doi: 10.3390/jcp1030021.
4. M. E. Ahmed, H. Kim, S. Camtepe, and S. Nepal, "Peeler: Profiling Kernel-Level Events to detect Ransomware," *arXiv.org*, Jan. 29, 2021. https://arxiv.org/abs/2101.12434#:~:text=In%20this%20paper%2C%20we%20present%20a%20novel%20ransomware,generic%20characteristics%20of%20ransomware%20depicted%20at%20the%20kernel-level.
5. A. J. Rose, "ScriptBlock Smuggling: Uncovering stealthy evasion techniques in PowerShell and .NET environments," *AFIT Scholar*. https://scholar.afit.edu/facpub/1519/
6. F. Yang, Y. Han, Y. Ding, Q. Tan, and Z. Xu, "A flexible approach for cyber threat hunting based on kernel audit records," *Cybersecurity*, vol. 5, no. 1, Jun. 2022, doi: 10.1186/s42400-022-00111-2.
7. M. Landauer, L. Alton, M. Lindorfer, F. Skopik, M. Wurzenberger, and W. Hotwagner, "Trace of the Times: Rootkit Detection through Temporal Anomalies in Kernel Activity," *arXiv.org*, Mar. 04, 2025. https://arxiv.org/abs/2503.02402#:~:text=The%20framework%20outlined%20in%20this%20paper%20injects%20probes,and%20uses%20statistical%20tests%20to%20detect%20time%20shifts.
8. R. Uetz *et al.*, *You Cannot Escape me: Detecting evasions of SIEM rules in enterprise networks*. 2024. [Online]. Available: https://www.usenix.org/system/files/usenixsecurity24-uetz.pdf
9. K. M. Shulika, D. S. Balagura, and Z. M. Sydorenko, "Analysis of methods for bypassing modern EDR endpoint protection systems," *Radiotekhnika*, vol. 2, no. 217, pp. 64–68, Jun. 2024.
10. "PAD: towards principled Adversarial Malware Detection against evasion attacks," *IEEE Journals & Magazine | IEEE Xplore*, Apr. 01, 2024. https://ieeexplore.ieee.org/document/10097506

11. Y. Sao and Sk. S. Ali, “Security analysis of scan obfuscation techniques,” *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 2842–2855, Jan. 2023, doi: 10.1109/tifs.2023.3265815.
12. D. C. D’Elia, E. Coppa, F. Palmaro, and L. Cavallaro, “On the Dissection of Evasive Malware,” *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2750–2765, Jan. 2020, doi: 10.1109/tifs.2020.2976559.
13. M. Hand, “Evading EDR,” *O’Reilly Online Learning*. https://learning.oreilly.com/library/view/evading-edr/9781098168742/xhtml/chapter1.xhtml#h-7
14. M. Graeber and L. Christensen, “Subverting sysmon: application of a formalized security product evasion methodology,” conference-proceeding, 2018. [Online]. Available: https://specterops.io/wp-content/uploads/sites/3/2022/06/Subverting_Sysmon.pdf
15. K. Elastic, “Potential Evasion via Windows Filtering Platform,” Elastic Security — Prebuilt detection rule, 2025. Available: https://www.elastic.co/docs/reference/security/prebuilt-rules/rules/windows/defense_evasion_windows_filtering_platform
16. R. Ben Yizhak, *#NoFilter: Abusing Windows Filtering Platform for privilege escalation*. 2021. [Online]. Available: https://media.defcon.org/DEF%20CON%2031/DEF%20CON%2031%20presentations/Ron%20Ben-Yizhak%20-%20NoFilter%20Abusing%20Windows%20Filtering%20Platform%20for%20privilege%20escalation.pdf
17. EliotSeattle, “Allocated Filter Altitudes - Windows drivers,” *Microsoft Learn*. https://learn.microsoft.com/en-us/windows-hardware/drivers/ifs/allocated-altitudes
18. Stevewhims, “Windows Filtering Platform - Win32 apps,” *Microsoft Learn*. https://learn.microsoft.com/en-us/windows/win32/fwp/windows-filtering-platform-start-page
19. H. S. Mahajan, “Filtering engine model for VIMNet,” M.S. thesis, The University of Toledo, Toledo, OH, 2023.
20. https://static.googleusercontent.com//42189.pdfmedia/research.google.com/en//pubs/archive/42189.pdf
21. Ciholas P, Such JM, Marnerides AK, Green B, Zhang J, Roedig U. Fast and Furious: Outrunning Windows Kernel Notification Routines from User-Mode. Detection of Intrusions and Malware, and Vulnerability Assessment. 2020 Jun 11;12223:67–88. doi: 10.1007/978-3-030-52683-2_4. PMCID: PMC7338165.
22. T. Imbert and Synacktiv, “Windows Kernel Security: A Deep Dive into Two Exploits Demonstrated at Pwn2Own,” Jul. 2023. [Online]. Available: https://conference.hitb.org/hitbsecconf2023hkt/materials/D2T1%20-%20Windows%20Kernel%20Security%20-%20A%20Deep%20Dive%20into%20Two%20Exploits%20Demonstrated%20at%20Pwn2Own%20-%20Thomas%20Imbert.pdf
23. K. Pundeer, “Host-Based Malware Analysis,” M.S. thesis, University of Alberta, Edmonton, AB, 2020.
24. Pat_H/to/File, “Getting more out of the Windows Filtering ETW Events,” *pat_h/to/file*, Oct. 31, 2020. https://blog.tofile.dev/2020/10/31/wfp.html

25. T. Strohmann, “DSR Approach Models | Design Science Research,” Design Science Research, Oct. 17, 2024. /https://design-science-research.de/en/course/dsr-questions/vorgehensmodelle/
26. Stevewhims, “Task Scheduler for developers - Win32 apps,” Microsoft Learn. https://learn.microsoft.com/en-us/windows/win32/taskschd/task-scheduler-start-page
27. K. Parveen, “Advanced Techniques of Malware Evasion and Bypass in the Age of Antivirus,” *International Journal for Electronic Crime Investigation*, vol. 8, no. 3, Sep. 2024
28. Decoding Malicious Camouflage: Analysing Evasion Techniques in Malware Against EDR and AMSI Detection,” *ResearchGate*, 2025.
29. J. Heibrink s1040183, “Antivirus and EDR bypasses for initial access.” [Online]. Available: https://www.cs.ru.nl/bachelors-theses/2023/Jorn_Heibrink___1040183___Antivirus_and_EDR_bypasses_for_initial_access.pdf