Automated Testing Framework for Malware Detection in Microsoft Defender for Endpoint

Graduate

Philipp Hutter

Initial Situation: Microsoft Defender for Endpoint (MDE) is an important tool to protect computers and networks from cyber attacks. However, it functions largely as a Black Box EDR, an endpoint detection and response system whose internal detection logic is not visible to users. MDE's detection rules are continuously updated in the background through cloud services, but there is no version history or official documentation explaining what has changed. As a result, security teams often don't know if MDE is still recognizing certain threats or if its detection capabilities have silently changed over time. Without a clear and structured way to test and confirm what MDE is detecting, organizations risk having a false sense of security.

Approach: To address this problem, this thesis introduces an automated testing system that uses real malware samples to evaluate MDE's behavior. The system runs these samples inside isolated virtual machines, ensuring they don't affect the host computer. Microsoft's Hyper-V technology is used to create clean and secure environments for each test. Once a sample is run, the system checks how MDE reacts by using its official cloud interface (API), which provides details about what was detected and how.

One special feature of the system is a comparison method based on the Levenshtein distance, a string metric for measuring the difference between two strings. This is used to compare MDE's alert titles against a reference list. Since these titles can be dynamically generated and may change slightly over time, such as differences in wording, formatting, or plurality, this approach helps avoid false positives when analyzing detection results. It ensures that minor variations do not lead to incorrect conclusions about changes in MDE's behavior. All tests can be configured easily, either through a menu or with a configuration file, and the results can be stored for future comparison. While the system doesn't yet include automated long-term tracking, it lays the groundwork for such features to be added later.

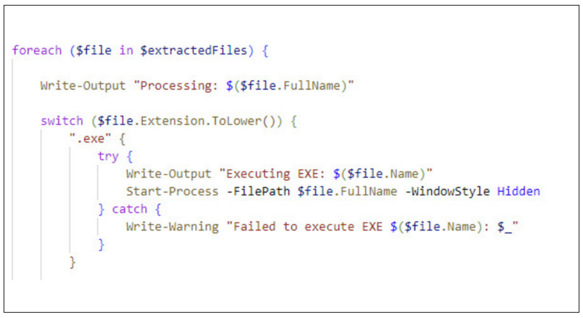
Conclusion: This project successfully delivered a reliable framework for testing Microsoft Defender for Endpoint using real malware samples in isolated virtual environments. By automating execution, log retrieval, and report generation, it gives security teams a practical way to monitor MDE's behavior, verify detections, and spot unexpected changes.

A key limitation is that MDE does not always return consistent results for the same sample. Its cloud-based analysis and environment-dependent behavior can lead to varying detections, making interpretation more complex.

Despite this, the framework fills an important visibility gap and lays the foundation for continuous validation. With future features like scheduled retesting or broader malware sources, it could become a powerful tool for ongoing endpoint security assurance.

Black-Box EDR
Own presentment

Logic to execute .exe files inside the VM
Own presentment

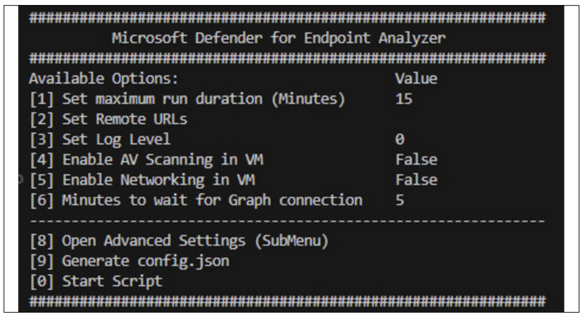


```
foreach ($file in $extractedFiles) {

    Write-Output "Processing: $($file.FullName)"

    switch ($file.Extension.ToLower()) {
        ".exe" {
            try {
                Write-Output "Executing EXE: $($file.Name)"
                Start-Process -FilePath $file.FullName -WindowStyle Hidden
            } catch {
                Write-Warning "Failed to execute EXE $($file.Name): $_"
            }
        }
```

Part of the configuration menu
Own presentment



```
###########################################################
         Microsoft Defender for Endpoint Analyzer
###########################################################
Available Options:                              Value
[1] Set maximum run duration (Minutes)          15
[2] Set Remote URLs
[3] Set Log Level                               0
[4] Enable AV Scanning in VM                    False
[5] Enable Networking in VM                     False
[6] Minutes to wait for Graph connection        5
-----------------------------------------------------------
[8] Open Advanced Settings (SubMenu)
[9] Generate config.json
[0] Start Script
###########################################################
```

Advisor
Cyrill Brunschwiler

Co-Examiner
Dr. Benjamin Fehrensen, Bern, BE

Subject Area
Networks, Security & Cloud Infrastructure