USENIX Security '25 Artifact Appendix – EvilEDR: Repurposing EDR as an Offensive Tool

Kotaiba Alachkar
Delft University of Technology

Dirk Gaastra
Independent Researcher

Eduardo Barbaro
Delft University of Technology

Michel van Eeten
Delft University of Technology

Yury Zhauniarovich
Delft University of Technology

A Artifact Appendix

A.1 Abstract

We provide artifacts to detect EvilEDR, as described in our study. These include Sigma rules in a generic format, which can be adapted for various security tools, along with platform-specific detection queries tailored to the Endpoint Detection and Response (EDR) solutions covered in this research, including Microsoft Defender for Endpoint (MDE) using Kusto Query Language (KQL), Elastic EDR using Elastic Query Language (EQL), Sophos EDR using SQL-based queries, and Trend Micro EDR using search queries.

For environments without EDR or Security Information and Event Management (SIEM) solutions, we provide PowerShell scripts that can be executed out of the box on any Windows 11 machine. These scripts detect unauthorized EDR drivers and processes directly from the local system without requiring privileged access or external dependencies.

Our artifacts offer practical and reproducible implementations, aligning with our "Open Science" commitment. This enables the scientific community to validate our findings and helps enterprises detect EvilEDR effectively. For example, MDE users can directly apply the KQL queries to monitor for unauthorized EDR activity, while PowerShell scripts provide a quick and flexible alternative for manual checks.

A.2 Description & Requirements

To recreate the same experimental setup and evaluate the artifact functionality, no specific hardware requirements are needed. The evaluation requires Windows 11 (tested on version 22H2) and trial licenses for the EDR solutions covered in our study. The objective is to test whether the provided detection rules (tailored to different EDR platforms) can successfully identify the presence of EvilEDR, an unauthorized EDR deployed and controlled by an attacker.

The process is as follows: install Windows 11, deploy one of the supported EDRs as the Enterprise EDR, and deploy a different EDR from another vendor to act as EvilEDR. Then, import the detection rule corresponding to the Enterprise EDR into its management console. These rules identify EvilEDR drivers and/or processes using vendor-specific detection queries. For environments without an EDR or SIEM, we provide PowerShell scripts that can be executed directly on the machine. These scripts detect the presence of known EDR presence and provide a viable alternative for EvilEDR detection.

It is important to note that we included Sigma rules in a generic format, which can be mapped to other detection formats as needed. It is up to the evaluators to test and adapt them based on their specific security tools and environment. Additionally, the Ludus testbed configuration file is provided to replicate our test environment; however, evaluators may verify and modify it as needed for their testing setup.

A.2.1 Security, privacy, and ethical concerns

Our artifact does not pose any security, privacy, or ethical risks to evaluators. The detection rules and scripts provided are designed solely for identifying EvilEDR activity and do not perform any destructive actions, disable security mechanisms, or alter system configurations.

The PowerShell scripts execute locally without requiring administrative privileges, and all detection is performed in a read-only manner. No persistent changes are made to the operating system, installed software, or user data.

A.2.2 How to access

Our artifact repository is hosted on the Zenodo platform and is publicly accessible through the following permanent URL: https://doi.org/10.5281/zenodo.15116409

A.2.3 Hardware dependencies

None.

A.2.4 Software dependencies

Our artifact requires a Windows 11 operating system for evaluation. We tested it on Windows 11 version 22H2, but any recent version should work. Additionally, the evaluation requires access to trial licenses for the following EDR solutions, which offer publicly available trial versions: MDE[1], Elastic EDR[2], Sophos EDR[3], and Trend Micro EDR[4].

Evaluators must obtain and install these solutions independently using the links provided. Trial versions were used during testing and provided full support for running the detection rules and queries included in the artifact. No proprietary software is included in the artifact package.

A.2.5 Benchmarks

Our artifact does not require predefined datasets, models, or workloads for evaluation. Instead, the detection rules and scripts are designed to identify the presence of EvilEDR on a machine. When evaluating detection rules for a specific vendor, the corresponding EDR must be deployed as the Enterprise EDR, and the queries are used to detect the presence of EvilEDR installed from a different vendor. If evaluators are testing the PowerShell scripts, it is sufficient to have EvilEDR deployed on the system and execute the scripts, which will list any detected drivers or processes.

A.3 Set-up

Evaluators must prepare their environment by obtaining trial licenses, installing the EDR solutions, and then running the detection artifacts. Trial licenses can be obtained directly from the vendors using the links listed in the Software Dependencies section.

A.3.1 Installation

Evaluators must set up a Windows 11 test environment and deploy two different EDR solutions: one as the Enterprise EDR and another as EvilEDR (from a different vendor). After deployment, they must import the detection rule specific to the Enterprise EDR into its management console:

- **MDE:** `MDE_EvilEDR_drivers.kql` and `MDE_EvilEDR_processes.kql`
- **Elastic:** `Elastic_EvilEDR_drivers.eql` and `Elastic_EvilEDR_processes.eql`
- **Sophos:** `Sophos_EvilEDR_drivers.txt` and `Sophos_EvilEDR_processes.txt`
- **Trend Micro:** `TrendMicro_EvilEDR_drivers.txt` and `TrendMicro_EvilEDR_processes.txt`

[1] https://learn.microsoft.com/en-us/defender-endpoint/defender-endpoint-trial-user-guide

[2] https://www.elastic.co/endpoint-detection-response

[3] https://www.sophos.com/en-us/products/endpoint-antivirus/free-trial

[4] https://www.trendmicro.com/en_hk/business/products/trials.html

To reduce false positives, evaluators should modify the detection rules to exclude drivers and processes belonging to the Enterprise EDR. If testing the PowerShell scripts, enable script execution using:

```
Set-ExecutionPolicy Unrestricted -Scope Process
```

A.3.2 Basic Test

Evaluators must run the detection rule in the Enterprise EDR management console. If EvilEDR is present, the query will return a list of detected drivers or processes; if not – it should return an empty result. For PowerShell-based detection, evaluators should run *Invoke-EDRDriverChecker.ps1* and *Invoke-EDRProcessChecker.ps1*. If EvilEDR is found, the script will list detected components; otherwise, the output will show “No EvilEDR detected.”

A.4 Evaluation workflow

This section outlines the steps required to evaluate the artifact and validate Section 6 (Defense) of our research. The evaluation involves setting up a controlled test environment, deploying EvilEDR, applying detection rules using the Enterprise EDR's management console, and verifying their effectiveness. It also includes testing PowerShell-based detection methods for environments without an EDR or SIEM.

A.4.1 Major Claims

(C1): *Enterprise EDR solutions can detect the presence of EvilEDR when our detection rules are applied. This is validated through experiment (E1), where the detection rules queries successfully identify EvilEDR drivers and processes across different vendor combinations. This is described in Section 6.1 (Enterprise-Level Defenses).*

(C2): *Our PowerShell scripts provide an effective alternative to identify EvilEDR in environments without an Enterprise EDR or SIEM. This is demonstrated in experiment (E2), where running `Invoke-EDRDriverChecker.ps1` and `Invoke-EDRProcessChecker.ps1` on a test machine lists detected EvilEDR drivers and processes. This is also described in Section 6.1 (Enterprise-Level Defenses).*

A.4.2 Experiments

This section outlines the experiments required to evaluate the artifact and validate Section 6 (Defense) of our paper. Each experiment describes how to configure the test environment, execute the detection methods, and interpret the results.

(E1): *Enterprise EDR Detection [3 human-hours]: Evaluates the ability of an Enterprise EDR to detect EvilEDR* *using the provided detection rules. This experiment validates (C1).*

Preparation: Install Windows 11 and deploy two different EDR solutions, one as the Enterprise EDR (e.g., MDE) and another as EvilEDR (e.g., Elastic). Import the detection rules specific to the Enterprise EDR into its management console:

- **MDE:** `MDE_EvilEDR_drivers.kql`, `MDE_EvilEDR_processes.kql`
- **Elastic:** `Elastic_EvilEDR_drivers.eql`, `Elastic_EvilEDR_processes.eql`
- **Sophos:** `Sophos_EvilEDR_drivers.txt`, `Sophos_EvilEDR_processes.txt`
- **Trend Micro:** `TrendMicro_EvilEDR_drivers.txt`, `TrendMicro_EvilEDR_processes.txt`

Repeat this process with different combinations of Enterprise EDR and EvilEDR (e.g., Elastic as Enterprise EDR and Sophos as EvilEDR, Sophos as Enterprise EDR and Trend Micro as EvilEDR) to ensure cross-compatibility testing.

Execution: Run the detection rule in the Enterprise EDR's query interface. Perform this step for each EDR pairing tested.

Results: If EvilEDR is present, the query should return a list of detected drivers or processes. Otherwise, the result should be empty. Confirm detection works consistently across different EDR combinations.

(E2): *PowerShell-Based Detection [0.5 human-hours]: Evaluates the ability to detect EvilEDR in environments without an Enterprise EDR or SIEM. This experiment validates (C2).*

Preparation: Install Windows 11 and deploy an EvilEDR (no Enterprise EDR is required). Place the following scripts in a working directory: `Invoke-EDRDriverChecker.ps1`, `Invoke-EDRProcessChecker.ps1`

Execution: Open PowerShell and run both scripts using the following commands:

```
.\Invoke-EDRDriverChecker.ps1
.\Invoke-EDRProcessChecker.ps1
```

Results: If EvilEDR is present, the scripts will list detected drivers and processes, including file paths. If no unauthorized EDR is found, the output will be: `No EvilEDR detected.`

A.5 Notes on Reusability

To reuse the detection artifacts in practical scenarios, evaluators must tune the detection rules to exclude *authorized* or *enterprise-approved* EDR processes and drivers. The provided rules are designed to identify all known EDR components, including both enterprise-deployed EDRs and unauthorized EvilEDR instances. To reduce false positives, particularly in environments with multiple active EDRs, enterprises should adapt the detection rules by explicitly excluding trusted drivers and processes while keeping detections for unauthorized instances enabled. Additionally, the PowerShell-based detection scripts are lightweight, self-contained, and can be executed without elevated privileges. They can be integrated into endpoint management solutions (e.g., Microsoft Intune) to support automated scanning, alerting, and response workflows. This enables security teams to scale EvilEDR detection across large environments and automate what would otherwise be manual investigations.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at `https://secartifacts.github.io/usenixsec2025/`.