PROCWATCH SENTINEL

Host-Based Watcher for Windows Process, Service, and Persistence Abuse

Domain: Cybersecurity / Blue Team / Web Security
Type: Practical Security Project
Submitted by: *Chirayu Paliwal*
Duration: Internship Practical Assignment
Tools & Technologies: Python, PowerShell, Visual Studio Code

1. INTRODUCTION

Modern cyberattacks rarely begin with dramatic explosions. They start quietly—through a suspicious process spawn, an abused system utility, or a malicious service registered for persistence.

Traditional antivirus tools often miss these early-stage indicators because they focus on known malware signatures rather than behavior.

PROCWATCH SENTINEL is a lightweight, behavior-focused Windows monitoring agent designed to detect suspicious process activity, service abuse, and persistence mechanisms in real time.

The project simulates how an entry-level SOC analyst or blue team tool would observe, log, score, and explain potentially malicious activity on a Windows host.

2. OBJECTIVE OF THE PROJECT

- To monitor running processes and Windows services programmatically.
- To analyze parent–child process relationships for abnormal execution chains.
- To detect suspicious persistence mechanisms (services, registry run keys, startup folders).
- To apply rule-based risk scoring aligned with real SOC logic.
- To map detected behaviors to MITRE ATT&CK techniques.
- To generate a structured, analyst-readable detection report.

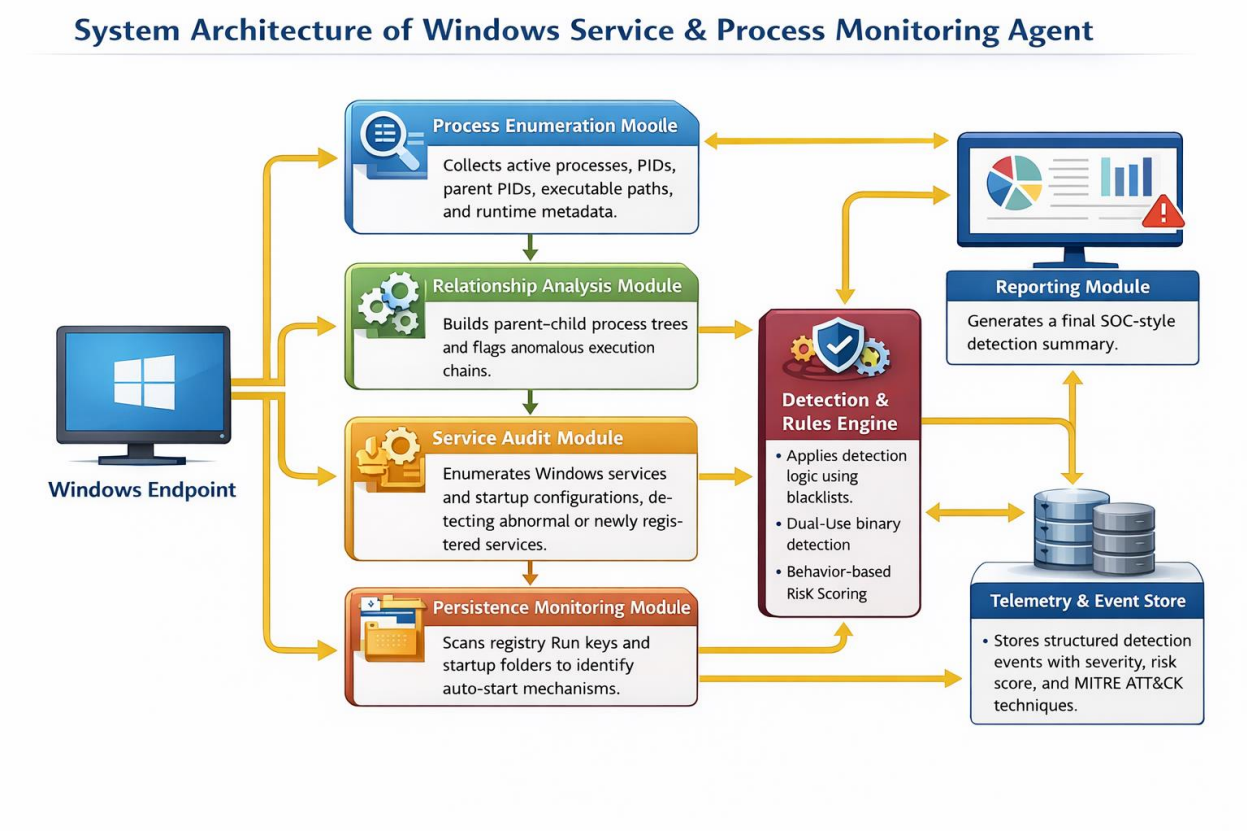
2.1. PROBLEM STATEMENT

The project starts from a simple headache: Windows has lots of built-in features that are useful for admins — things like PowerShell, scheduled services, and auto-run keys — but attackers can quietly abuse those same tools. They can run bad commands without obvious malware, stay on the machine after reboots, and hide in what looks like normal activity.

Beginner security tools often stumble: either they scream about everything (false alarms everywhere) or they flag suspicious behavior without explaining why it matters. That leaves learners staring at alerts without really learning.

This project steps into that gap. It focuses on detecting these subtle abuses while also teaching why they're dangerous. The goal isn't just "red light, green light," but helping people understand how attackers piggyback on legitimate Windows features — turning the operating system itself into their toolbox.

3. ARCHITECTURE



PROCWATCH SENTINEL uses a modular pipeline design similar to tools used in real security operations centers. It begins by collecting data across the system, including active processes, services, registry keys, and startup programs. This gives it a broad baseline of system activity.

In the analysis stage, it builds relationships between processes, tracking which ones spawn others and how they interact. It then applies rules and heuristic logic to spot behavior that either matches known threats or simply looks suspicious.

The detection layer assigns severity and risk scores so analysts know what to prioritize. It also maps behaviors to MITRE ATT&CK techniques, which helps describe exactly what type of attack pattern may be unfolding.

Finally, PROCWATCH SENTINEL logs everything in structured form and produces a unified detection report. Because each layer works independently, the system stays flexible, scalable, and realistic compared to enterprise security tools.

3.1 Project Structure & Module Organization

**Workflow of the PROCWATCH SENTINEL detection framework.*

4. METHODOLOGY

Step 1: Process Monitoring

- Enumerated all active processes.
- Collected PID, parent PID, executable path, and binary name.
- Flagged dual-use binaries frequently abused by attackers.

Step 2: Relationship Analysis

- Built parent–child process trees.
- Detected suspicious execution chains (e.g., Office → PowerShell).

Step 3: Service Auditing

- Enumerated installed Windows services.
- Identify unusual or newly registered services.
- Flagged services linked to persistence behavior.

Step 4: Persistence Detection

- Scanned registry Run keys.
- Inspected startup folders.
- Detected auto-start mechanisms.

Step 5: Rule-Based Detection

- Applied blacklist and heuristic rules.
- Assigned severity levels (LOW / MEDIUM / HIGH).
- Calculated risk scores based on behavior patterns.

Step 6: MITRE ATT&CK Mapping

- Linked detections to techniques such as:
- Command and Scripting Interpreter (T1059).
- Boot or Logon AutoStart Execution (T1547).

Step 7: Reporting

- Generated a structured final detection report.
- Summarized events, severities, and attack techniques.

5. TOOLS & TECHNOLOGIES USED

Tool / Technology	Purpose
Programming & OS	Python 3.14.2 Windows Operating System
Libraries & Concepts	Process enumeration libraries Windows service inspection Registry access JSON-based rule and policy files
Security Framework	MITRE ATT&CK (for tactic and technique mapping)
Development Environment	Visual Studio Code PowerShell (for execution and validation)

6. Threat Model and Assumptions

PROCWATCH SENTINEL is designed around a realistic post-compromise threat model.

Attacker Assumptions

- Attacker has local user or limited administrative access
- Uses living-off-the-land binaries (LOLBins)
- Attempts persistence through services or startup mechanisms
- Avoids deploying obvious malware

Defender Assumptions

- No kernel or memory-level visibility
- Limited to user-space telemetry
- Focused on detection and analysis, not prevention

This threat model aligns with common enterprise breach scenarios and red team techniques.

7. Detection Logic Rationale

PROCWATCH SENTINEL does not treat system utilities as inherently malicious. Instead, it evaluates **the context**.

PROCWATCH SENTINEL does not label binaries as “malicious” by default. Instead, it evaluates context and behavior.

- Dual-use binaries are treated as risk indicators, not threats by default.
- Parent–child relationships provide execution context.
- Persistence mechanisms increase overall risk score.

This mirrors detection logic used in real EDR and SOC environments.

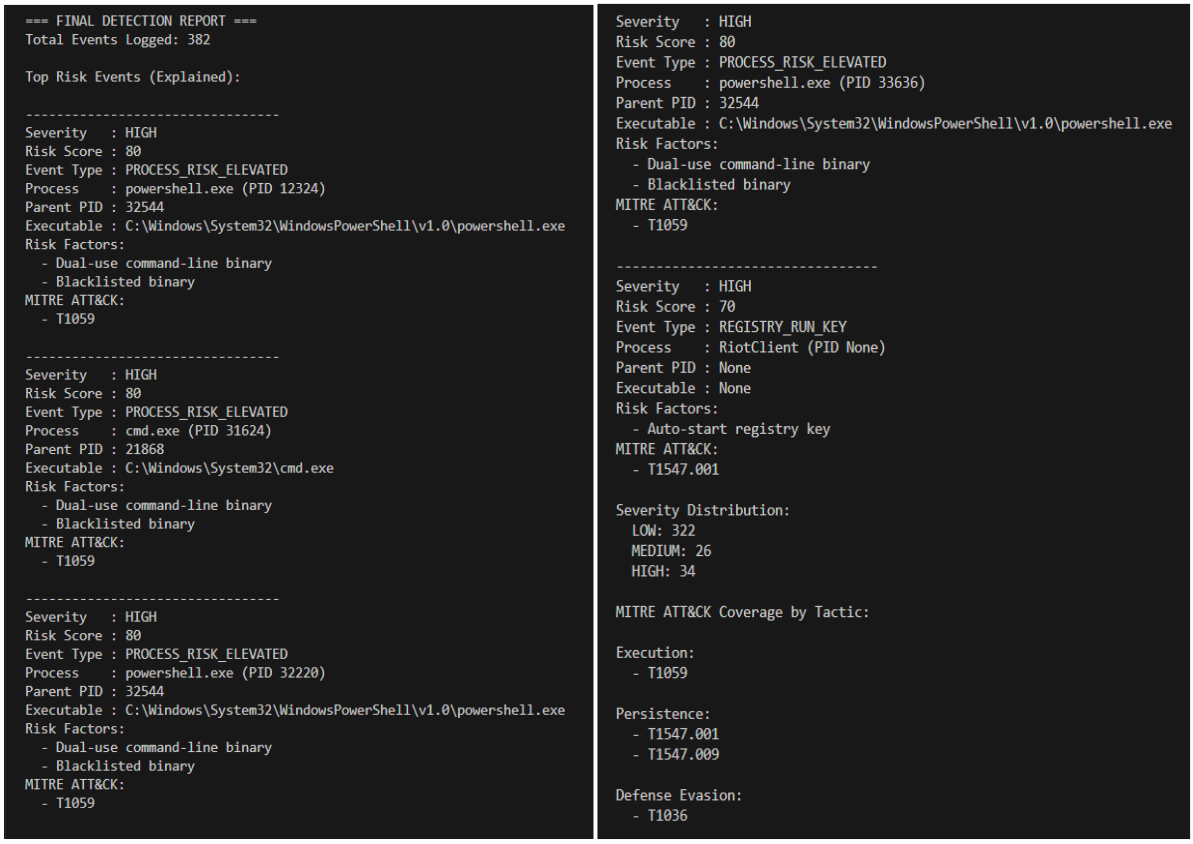
8. Risk Scoring Philosophy

Risk scoring is based on a weighted model considering:

- Type of executable
- Execution context
- Presence of persistence mechanisms
- Known malicious indicators

This allows analysts to prioritize alerts based on likelihood and impact.

9. Observations



```
=== FINAL DETECTION REPORT ===
Total Events Logged: 382

Top Risk Events (Explained):

--------------------------------
Severity   : HIGH
Risk Score : 80
Event Type : PROCESS_RISK_ELEVATED
Process    : powershell.exe (PID 12324)
Parent PID : 32544
Executable : C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Risk Factors:
  - Dual-use command-line binary
  - Blacklisted binary
MITRE ATT&CK:
  - T1059

--------------------------------
Severity   : HIGH
Risk Score : 80
Event Type : PROCESS_RISK_ELEVATED
Process    : cmd.exe (PID 31624)
Parent PID : 21868
Executable : C:\Windows\System32\cmd.exe
Risk Factors:
  - Dual-use command-line binary
  - Blacklisted binary
MITRE ATT&CK:
  - T1059

--------------------------------
Severity   : HIGH
Risk Score : 80
Event Type : PROCESS_RISK_ELEVATED
Process    : powershell.exe (PID 32220)
Parent PID : 32544
Executable : C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Risk Factors:
  - Dual-use command-line binary
  - Blacklisted binary
MITRE ATT&CK:
  - T1059
```

```
Severity   : HIGH
Risk Score : 80
Event Type : PROCESS_RISK_ELEVATED
Process    : powershell.exe (PID 33636)
Parent PID : 32544
Executable : C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Risk Factors:
  - Dual-use command-line binary
  - Blacklisted binary
MITRE ATT&CK:
  - T1059

--------------------------------
Severity   : HIGH
Risk Score : 70
Event Type : REGISTRY_RUN_KEY
Process    : RiotClient (PID None)
Parent PID : None
Executable : None
Risk Factors:
  - Auto-start registry key
MITRE ATT&CK:
  - T1547.001

Severity Distribution:
  LOW: 322
  MEDIUM: 26
  HIGH: 34

MITRE ATT&CK Coverage by Tactic:

Execution:
  - T1059

Persistence:
  - T1547.001
  - T1547.009

Defense Evasion:
  - T1036
```

10. RESULTS

The system successfully detected high-risk processes such as:

- powershell.exe
- cmd.exe

Identified suspicious service and persistence behaviors

Generated SOC-style output explaining:

- What happened
- Why it is risky
- How it maps to attacker techniques

The report format closely resembles early-stage EDR or SIEM alert summaries.

11. False Positives and Analyst Triage

Expected False Positives

- IT administrators legitimately using PowerShell.
- Security tools running command-line utilities.
- Legitimate services registered by software updates.

How PROCWATCH SENTINEL Handles This

- Severity and risk scoring instead of binary decisions.
- Contextual details included in reports.
- MITRE ATT&CK mapping to guide analyst judgment.

12. CONCLUSION

Through the development of PROCWATCH SENTINEL, I gained a clear understanding of how host-based detection works in real-world Windows environments. This project helped me move beyond theoretical cybersecurity concepts and apply them in a practical, analyst-focused manner.

I learned how attackers abuse legitimate Windows utilities such as command-line and scripting tools, and why these binaries cannot be treated as malicious on their own. By analyzing parent–child process relationships, I understood how execution context plays a critical role in distinguishing normal system activity from suspicious behavior.

Working on service auditing and persistence detection taught me how attackers maintain long-term access using Windows services, registry Run keys, and startup mechanisms. I also learned the importance of correlating multiple weak signals rather than relying on a single indicator to determine risk.

Implementing rule-based detection and risk scoring helped me understand how SOC tools prioritize alerts instead of making binary decisions. Mapping detections to the MITRE ATT&CK framework strengthened my ability to relate low-level system activity to high-level attacker tactics and techniques.

Overall, this project significantly improved my blue team thinking. I learned how to design explainable security detections, analyze system behavior from a defender's perspective, and generate reports that support analyst decision-making rather than overwhelming it. PROCWATCH SENTINEL has given me practical experience that directly aligns with real-world SOC operations and endpoint security monitoring.