Sentry: Advanced Process Monitoring & System Activity Tracker

Alternative Learning Assessment

Presented to the Faculty of the

College of Information and Communications Technology

West Visayas State University

La Paz, Iloilo City

In Fulfillment

Of the Requirements for the Subject

CCS 236: Operating Systems

by

Domenic Taganahan

BSCS 4B AI

to

Orlando Cabillos

Instructor

December 01, 2025

Table of Contents

I. OVERVIEW

In the domain of contemporary operating systems, the effective management of computational resources is of critical importance to ensure system stability, responsiveness, and optimal performance. Modern computing environments demand tools that not only present resource utilization data but also enable precise and timely interventions to address performance bottlenecks or system anomalies. Although Windows provides native graphical utilities, such as Task Manager and Resource Monitor, which serve as accessible interfaces for monitoring system processes and resource consumption, these tools primarily prioritize user accessibility and simplicity. As a result, they often lack the level of detailed control and rapid responsiveness required by advanced users or system administrators engaged in complex troubleshooting and performance tuning tasks.

Moreover, graphical utilities tend to impose additional overhead on the system, particularly noticeable when the system is operating under heavy load or experiencing resource constraints. This overhead can inadvertently exacerbate performance issues instead of alleviating them, limiting the effectiveness of these tools in critical scenarios. Consequently, there is a distinct need for lightweight, efficient solutions that minimize resource consumption while providing rich, real-time monitoring capabilities.

Sentry is a command line interface tool developed to fulfill this need by offering a high-performance alternative to traditional graphical monitoring utilities. Designed with an emphasis on minimal resource usage and rapid data processing, Sentry bridges the gap between standard monitoring tools and the requirements of advanced system administration. It functions not only as an effective real-time process monitor, capable of

delivering immediate insights into system behavior, but also as a comprehensive system activity tracker that aggregates and presents detailed information about system operations over time. This dual functionality empowers users to conduct in-depth analysis and make informed decisions to optimize system performance and maintain operational integrity under diverse workloads.

II. PURPOSE

The development of Sentry addresses several significant limitations inherent in conventional Graphical User Interface system tools.

First, in terms of accessibility and responsiveness, graphical tools frequently exhibit sluggish launch times, particularly during instances of system unresponsiveness or freezes. In contrast, Sentry operates instantly within a terminal environment, imposing minimal overhead on system resources and thereby ensuring rapid deployment even under adverse conditions.

Second, regarding advanced system control, standard monitoring utilities often restrict users from altering specific process attributes essential for fine-grained management. Sentry introduces enhanced capabilities, informally referred to as "God Mode," which enable users to suspend individual threads in memory, thus effectively freezing processes, bind processes to designated central processing unit cores to optimize performance through affinity settings, and dynamically adjust process priority classes to better allocate system resources in real time.

Third, in the context of boot performance management, while native tools such as Task Manager provide visibility into startup applications, they do not permit immediate and precise removal of registry entries that contribute to system bloat and extended boot times. Sentry addresses this gap by offering a direct interface to critical sections of the Windows Registry, including the current user and local machine hives, facilitating the

permanent elimination of unwanted startup configurations that degrade system initialization performance.

Fourth, with respect to activity tracking and behavioral analysis, typical system monitors present only instantaneous snapshots of process activity, lacking historical context or behavioral insights. Sentry incorporates an Activity Logger that captures and records process behaviors over time, enabling longitudinal analysis. Moreover, it employs heuristic algorithms to detect potentially suspicious processes, such as unsigned executables operating from temporary directories, which are frequently indicative of malware or unauthorized scripts.

By translating complex Windows application programming interface calls into an interactive text-based user interface, Sentry exemplifies the capacity of PowerShell beyond its conventional role as a scripting language, establishing it as a versatile platform for the development of robust and sophisticated system utilities.

III. KEY FEATURES

- **Deep Process Control**: Includes "Freeze" (**Suspend**) and "Thaw" (**Resume**) functionality, Priority Class management, and CPU Affinity (Core Latching).
- **Startup App Manager:** Scans HKCU and HKLM registry hives to list startup programs and allows for permanent removal. It also retrieves the "Last BIOS Boot Time" from the Windows Event Log.
- **Live Dashboard (HUD):** A real-time, auto-refreshing Heads-Up Display showing CPU/RAM vitals and top consumers.
- **"Shady" Process Scanner**: Heuristic scanning to identify potential malware based on unsigned code, suspicious file paths (AppData/Temp), and hidden windows.
- **Activity Logger**: Snapshots active processes to a local JSON database to track usage history and average session durations.
- **Network Sentinel**: Filters active TCP/UDP connections to identify applications establishing external connections.

IV. USER INTERFACE OVERVIEW

Sentry utilizes a high-contrast Text-based User Interface (TUI) designed for readability and rapid keyboard navigation. The interface is divided into modular screens, each providing specific system insights or control options.

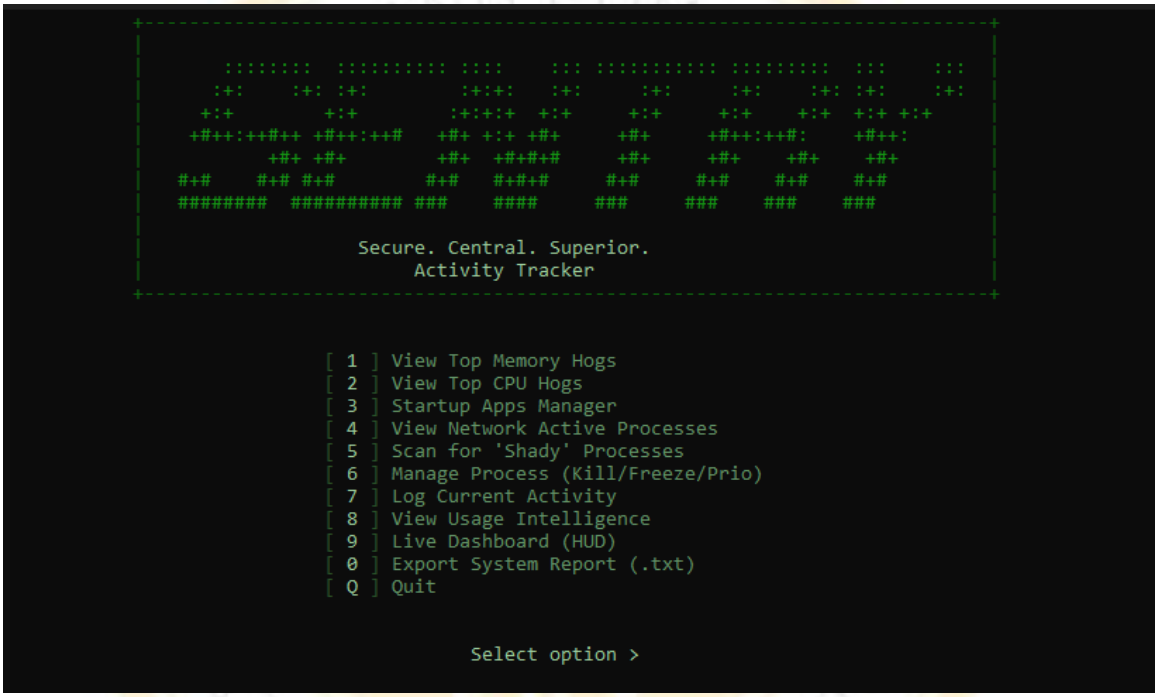

Figure 1: Main Menu Dashboard

The entry point of the application features a stylized ASCII header ("Secure. Central. Superior.") and displays the title "Activity Tracker". The user is presented with the main navigation loop where they can select tools by entering the corresponding number. Note the error handling message ("Invalid selection") displayed in red when incorrect input is detected.

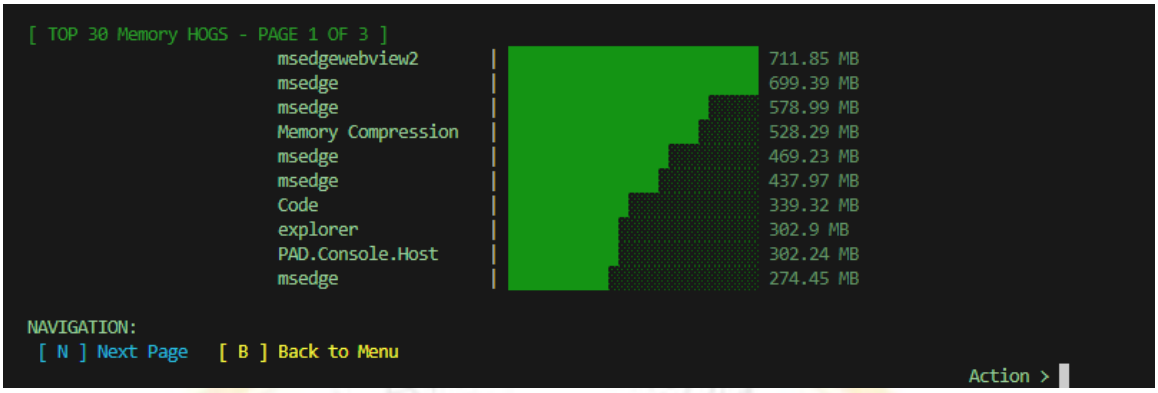

Figure 2: Memory Hogs Visualizer

This screen displays the top consumers of system RAM. Sentry visualizes memory usage using green ASCII bar charts for instant data interpretation. The interface supports pagination (Page 1 of 3), allowing users to browse through the top 30 processes using [N] for Next Page and [P] for Previous Page without cluttering the view.

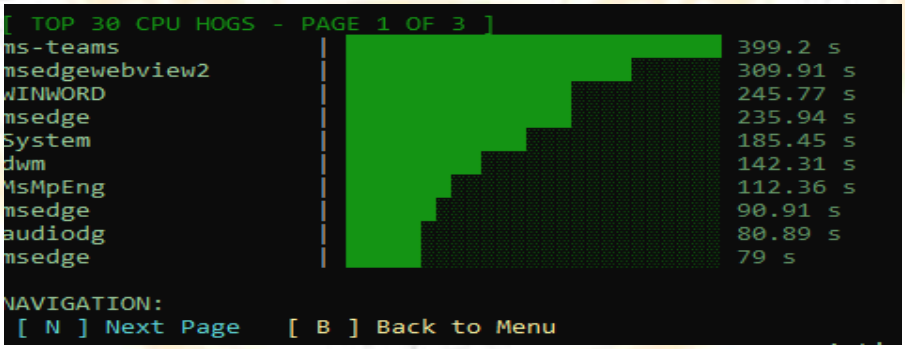

Figure 3: CPU Hogs Visualizer

Similar to the memory visualizer, this screen ranks processes by Processor Time (CPU). It displays the process name alongside the exact CPU time in seconds. The visual bars help identify processes that are "pegging" the CPU, such **as ms-teams or msedgewebview2** in this example.

Figure 4: Startup Apps Manager

This module provides a detailed list of applications configured to launch at boot. It displays the **Last BIOS Boot Time** (in seconds) at the top. The list categorizes entries by Registry Hive ([CurrentUser] vs [LocalMachine]) and displays the full file path. Users can select a number to permanently remove an entry or press B to return to the menu.

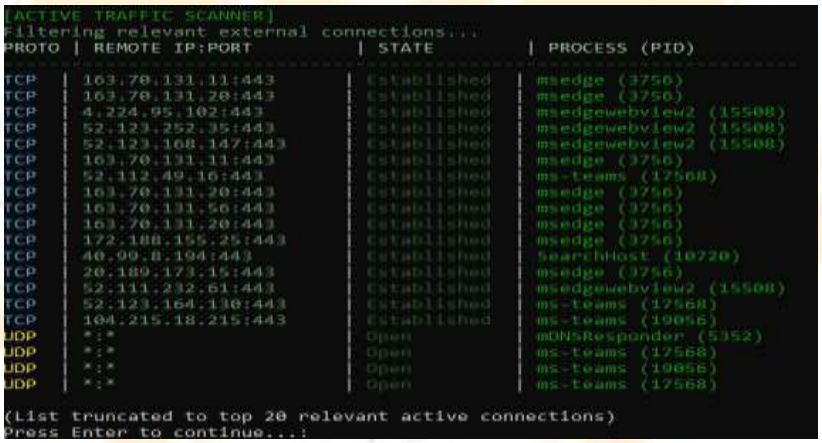

Figure 5: Active Traffic Scanner

The Network Sentinel view filters active connections to show external traffic. It organizes data into columns: Protocol (TCP/UDP), Remote IP:Port, Connection State (Established/Open), and the Owning Process with its PID. This allows for the immediate identification of applications communicating with external servers.

Figure 6: Shady Process Scanner

This security feature uses heuristics to flag suspicious activity. The output lists the Process Name, the Reason for flagging (e.g., [Suspicious Path], [Unsigned], or [Hidden PowerShell]), and the executable path. This helps identify malware running from temporary folders or unsigned scripts masquerading as system files.

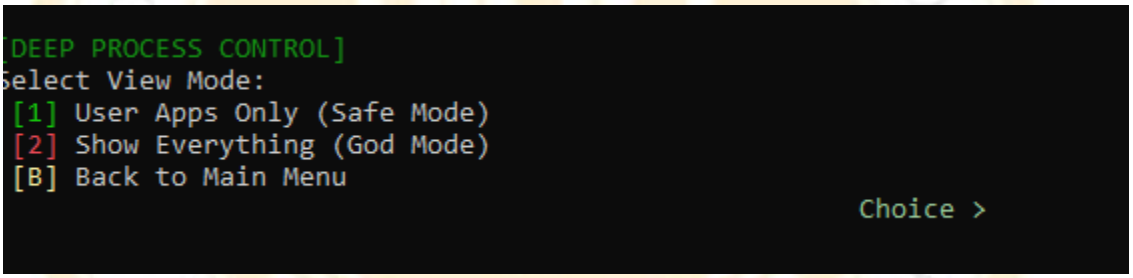

Figure 7: Deep Process Control

View Mode Upon selecting the Process Manager, the user is prompted to choose a "View Mode".

- [1] **User Apps Only (Safe Mode**): Filters out critical OS processes to prevent accidental system crashes.
- [2] **Show Everything (God Mode):** Grants access to all running processes, including kernel-level tasks.

Figure 8: Deep Process Control

Process List (Safe Mode) In "Safe Mode," Sentry lists standard user applications (e.g., ms-teams, WINWORD, Code) sorted by CPU usage. The user can select a target by typing the number or the application name to access the Kill/Freeze/Affinity sub-menu.

Figure 9: Deep Process Control

Process List (God Mode) In "God Mode," the list expands to include critical Windows processes such as System, dwm (Desktop Window Manager), MsMpEng (Defender), and audiodg. This mode is intended for advanced troubleshooting where system services may need to be managed directly.

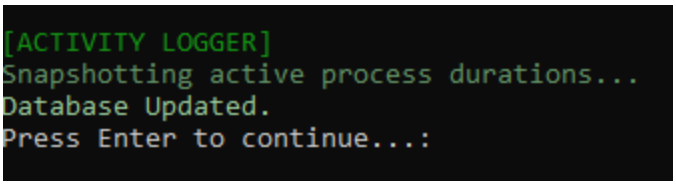

Figure 10: Activity Logger (Snapshot Tool)

This function acts as the system's "Black Box" recorder. When selected (Option 7), Sentry takes an instantaneous "snapshot" of all active processes and their duration, appending this data to a local JSON database. This does not display data immediately; instead, it updates the historical record, which allows the **Usage Intelligence** module (Option 8) to calculate long-term statistics and average session time later.

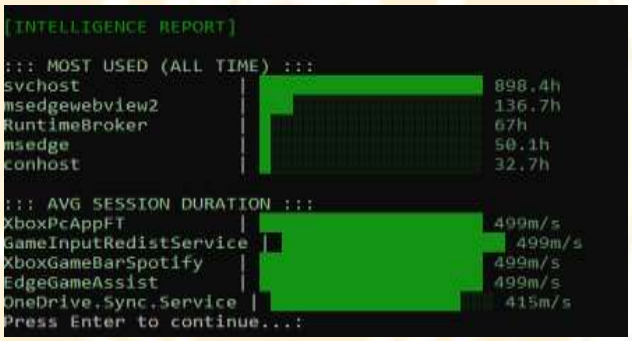

Figure 11: Usage Intelligence Report

Accessed via **Option 8**, this screen visualizes the data collected by the Activity Logger. It is divided into two analytical sections:

1. **Most Used (All Time):** Displays which processes have been active for the longest cumulative duration (e.g., svchost running for 898.4 hours).
2. **Avg Session Duration:** Calculates how long specific applications run on average before closing, helping users identify their most "sticky" or frequently used apps.

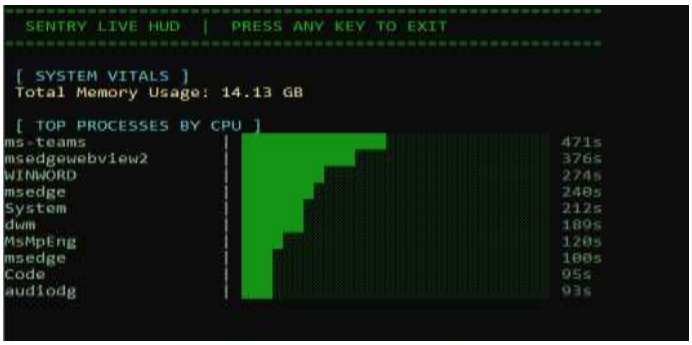

Figure 12: Live Dashboard (HUD)

This is the "Head-Up Display" (Option 9), designed for a secondary monitor or quick status checks. Unlike the interactive menus, this screen refreshes automatically every 1.2 seconds. It displays high-level System Vitals (Total Memory Usage) and a real-time leaderboard of the Top Processes by CPU usage, rendered with stable ASCII bars to prevent visual jitter.

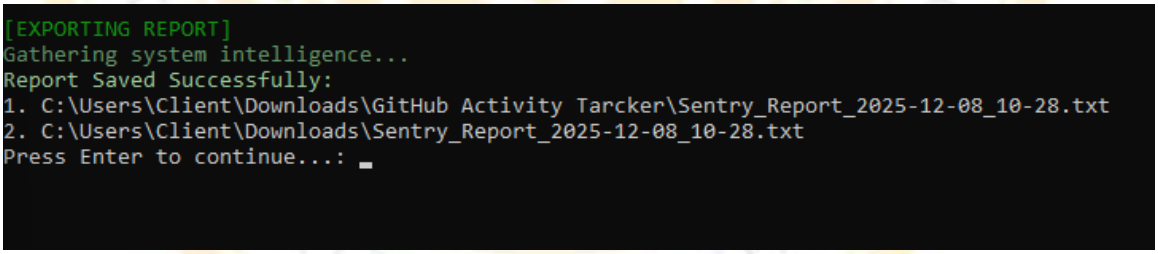

Figure 13. Export File System Report

When the user selects **Option 0**, Sentry aggregates all current system data (CPU hogs, RAM usage, Network connections, and "Shady" processes) into a timestamped text file. The interface confirms the operation and provides the full file paths for the report, which is saved in two locations simultaneously for convenience: the script's root directory and the user's **Downloads** folder.

V. Usage

5.1 Prerequisites

Before using Sentry, ensure your environment meets these requirements:

- Windows 10 or 11

  (Required for specific Registry paths and Event Log IDs).

- PowerShell 5.1 or Higher

  (Pre-installed on modern Windows).

- Administrator Privileges

  CRITICAL: Sentry must be run as Administrator to access System Registry keys (Startup Apps), manage Services, and inject C# code for process suspension.

- Execution Policy

  Must be set to allow scripts (e.g., RemoteSigned or Bypass).

5.2 Main Menu Options

Resource Visualizers (Options 1 & 2):

- **View Top Memory/CPU Hogs:** Instantly sorts processes by resource consumption.
- **Visualization:** Uses graphical ASCII bars to identify outliers.
- **Paginated:** Browse top 30 processes across multiple pages.

System Management (Options 3 & 6):

- **Startup Apps Manager:** Displays "Last BIOS Boot Time" and allows permanent deletion of Registry run keys to speed up booting.
- **Manage Process (Deep Control):** The core "Killer" suite. Allows Killing, freezing (Suspend), Resuming, Priority setting, and CPU Core Latching (Affinity).

Security & Network (Options 4 & 5):

- **Network Active Processes:** Filters for applications currently sending/receiving data to external IPs (TCP/UDP).
- **Shady Process Scanner:** Heuristic scan for unsigned apps, hidden windows, or executables running from Temp/AppData folders.

Intelligence & Logging (Options 7, 8, 9):

- **Log Current Activity:** Takes a snapshot of current processes to build a history database.
- **View Usage Intelligence:** Calculates average session durations and most frequently used apps based on logs.
- **Live Dashboard (HUD):** Auto-refreshing monitor for a secondary screen.

Reporting (Option 0):

- **Export System Report:** Generates a timestamped .txt file containing all current system metrics.

5.3 Navigation Controls

- **[0-9]** - Select Main Menu options.
- **[N]** - Next Page (Available in Memory/CPU views).
- **[P]** - Previous Page (Available in Memory/CPU views).
- **[B]** - Back to Main Menu (Used in sub-menus).
- **[Q]** - Quit application.
- **[Enter]** - Confirm selection or pause scrolling.

5.4 Example Workflows

Workflow 1: Deep Process Control (The "Freeze" Method)

Use this when a game or render is consuming too much CPU, but you don't want to close it.

1. Launch Sentry as Administrator.
2. Select **[6] Manage Process**.
3. Select **[2] Show Everything (God Mode)**.
4. Type the process name (e.g., Valorant or Blender).
5. Select **[2] Freeze/Resume** -> **[1] Freeze**.
6. *Result:* The app is suspended in RAM (0% CPU). You can now perform other tasks.
7. To restore, return to the menu and select **[2] Unfreeze**.

Workflow 2: Optimizing Boot Speed

Use this to remove "bloatware" that slows down your PC start-up.

1. Launch Sentry.
2. Select **[3] Startup Apps Manager**.
3. Observe the **Last BIOS Boot Time** at the top.
4. Review the list of apps (e.g., OneDrive, Teams, Steam).
5. Enter the number of the app you want to remove.
6. Type YES to confirm.
7. *Result:* The Registry key is permanently deleted.

Workflow 3: Security Audit

Use this if your PC feels sluggish or behaves strangely.

1. Select **[5] Scan for 'Shady' Processes**.
2. Review the output for "Hidden PowerShell" or "Unsigned" warnings.
3. If a suspicious PID is found, copy the ID.
4. Go to **[4] View Network Active Processes** to see if that PID is connecting to the internet.
5. Go to **[6] Manage Process** -> **[1] Kill** to terminate it instantly.

5.5 File Structure

Sentry is a portable application requiring no installation, but it generates local data files.

Sentry/

├── *Sentry.ps1 (The core script application)*

├── *sentry_activity_log.json (database of process history)*

└── *Sentry_Report_YYYY-MM-DD.txt (Reports generated via Option 0)*

5.6 Recommended Setup: Activity Tracking Strategy

To get the most out of the **Usage Intelligence (Option 8)** feature, Sentry needs data. It does not run in the background service by default to save resources.

Recommended Routine:

1. **Launch Sentry** periodically during your work session.
2. Hit **[7] Log Current Activity**.
3. This takes a split-second snapshot of what is running.
4. **Repeat** this a few times a day.

Why these matters:

- **Option 8** calculates averages based on these snapshots.
- The more snapshots you take over time, the more accurate your "Average Session Duration" stats will become.
- This creates a local database of your computing habits without sending data to the cloud.

VI. Implementation

Sentry is written in **PowerShell 5.1+** and utilizes .NET framework classes and Win32 API calls for advanced functionality.

6.1 Script Architecture

Core Script (Sentry.ps1): Unlike multi-file solutions, Sentry is compiled into a monolithic script for portability. It follows a procedural execution flow:

1. **Environment Setup:** Configures Console Encoding to UTF-8 and injects C# definitions.
2. **Function Definitions:** Loads all logic modules into memory.
3. **Main Execution Loop:** Initiates the interactive do...while dashboard loop.

Core Functions:

- **Initialize-NativeMethods:** (Implicit) Compiles C# code to access ntdll.dll for process suspension.
- **Show-Header:** Renders the responsive "Matrix-style" ASCII banner.
- **Get-ResourceHogs:** Fetches Get-Process, sorts by CPU/RAM, and renders paginated bar charts.
- **Manage-StartupApps:** Queries Registry hives and Event Logs for boot management.
- **Manage-Process:** The "Killer" module containing logic for Killing, Freezing, and Affinity setting.
- **Analyze-ShadyProcs:** Runs heuristic checks against active process paths and signatures.
- **Start-LiveDashboard:** A self-contained loop for the auto-refreshing HUD.
- **Export-Report:** Aggregates all data points into a formatted text file.
-

6.2 Native Methods and PowerShell Integration

P/Invoke (Platform Invocation): Standard PowerShell cmdlets (Stop-Process) cannot pause a running application. Sentry overcomes this limitation by embedding C# code directly within the script using the Add-Type cmdlet.

Native Function Import:

- **Library:** ntdll.dll (Windows Native API)

- **Methods Imported:**
  - NtSuspendProcess: Halts the execution threads of a target process.
  - NtResumeProcess: Restores execution threads.

Why this approach:

- **Granular Control:** Allows Sentry to manipulate memory handles that are usually restricted to the OS kernel.
- **Portability:** By embedding the C# definition in the string $MethodDefinition, the script requires no external .dll files to be distributed with it. It compiles the necessary code at runtime.

6.3 Data Acquisition and Monitoring

Real-Time Metrics:

- **Source:** System.Diagnostics.Process .NET class.
- **Data Points:** Working Set (RAM), Processor Time, PID, Priority Class.
- **Startup Data:** Queries HKCU:\Software\Microsoft\Windows\CurrentVersion\Run and HKLM equivalents using Get-ItemProperty.

Boot Performance:

- **Source:** Windows Event Log (Diagnostics-Performance).
- **Filter:** EventID 100 (Boot Performance Monitoring).
- **Calculation:** Converts the stored MainPathBootTime from milliseconds to seconds.

6.4 Search Algorithms (Heuristics)

Shady Process Logic: The Analyze-ShadyProcs function utilizes a heuristic filtering algorithm to identify potential threats without a virus signature database.

Detection Criteria:

1. **Path Analysis:** Flags executables running from volatile directories (AppData, Temp) often used by malware droppers.
2. **Signature Verification:** Runs Get-AuthenticodeSignature on the executable. If the status is not "Valid," it is flagged.
3. **Window Obfuscation:** Detects powershell.exe instances running with Hidden window styles, a common tactic for malicious scripts.

6.5 History Management

Activity Logger: Sentry maintains a local database to track long-term usage statistics.

Storage Structure:

- **Format:** JSON (sentry_activity_log.json).
- **Fields:** Date, ProcessName, Duration (Minutes).
- **Logic:** The Update-ActivityLog function reads the existing JSON, appends the current snapshot of active processes, and writes it back to disk.

Data Trimming: To prevent the JSON file from growing indefinitely and slowing down the script, the array is strictly limited to the last **5,000 entries**. Oldest entries are automatically discarded when this limit is reached.

6.6 Performance Optimization

What Sentry does to stay fast:

- **ASCII Rendering:** Uses text-based characters for graphs instead of heavy GDI+ or WPF graphics, ensuring near-zero GPU usage.
- **Critical Filtering:** The "Safe Mode" view pre-filters the process list to exclude hundreds of system drivers (svchost, csrss), reducing the rendering load.
- **Manual Garbage Collection:** Variables containing heavy process objects are overwritten in each loop iteration to manage memory footprint.
- **Pagination:** Options 1 and 2 limit the display to 10 items per page, preventing console buffer lag associated with printing hundreds of lines at once.

6.7 Error Handling

How Sentry handles problems:

- **Privilege Checks:** Registry and Service management functions are wrapped in try...catch blocks to detect "Access Denied" errors, prompting the user to run as Administrator.
- **Null Validation:** Checks if a process still exists before attempting to Kill/Freeze it (handling race conditions where a process closes while the menu is open).
- **Input Sanitization:** The Get-SafeInput function trims whitespace and enforces character limits to prevent buffer overflow or script injection attempts via the prompt.

VII. Why Sentry?

Sentry provides a robust, power-user alternative to standard Windows system utilities for users who require granular control over their operating environment. Unlike the default Task Manager, which prioritizes user-friendly safety and often hides critical details, Sentry puts the user in complete command of process execution and system startup behavior. It is lightweight, portable, and requires no installation—just a single PowerShell script that can be deployed instantly on any machine.

Sentry is particularly useful for gamers needing to maximize CPU resources, developers debugging application states, and system administrators auditing security without third-party bloatware. The tool is not meant to replace Task Manager for casual monitoring, but rather serves as a specialized "Killer" application for deep performance tuning, boot optimization, and troubleshooting unresponsive tasks.

Key Reasons to Use Sentry

- **Deep System Control** You decide how applications run. Sentry exposes kernel-level controls like **Process Suspension (Freezing)**, **CPU Core Latching (Affinity)**, and **Priority Class** modification—features often buried or inaccessible in standard tools.
- **Portable & Zero-Installation** Consisting of a single script file (Sentry.ps1), it requires no setup wizards or registry clutter. It can be carried on a USB drive and run on any Windows 10/11 system with Administrator rights instantly.

- **Boot Performance Mastery** While standard tools allow you to "disable" startup apps, Sentry allows for the surgical **deletion** of Registry run keys (HKCU/HKLM), permanently removing bloatware that slows down your BIOS boot time.
- **Activity Intelligence** Sentry doesn't just show you what is happening *now*; it maintains a local history of your process activity. This allows you to view **Average Session Durations** and identify which applications are consuming the most time over days or weeks.
- **Security Auditing** The integrated **"Shady" Process Scanner** uses heuristics to instantly flag unsigned executables or programs running from volatile directories (Temp, AppData), providing a quick security health check without a full antivirus scan.
- **Complete Privacy** No cloud connectivity, no telemetry, and no background services sending data to external servers. All activity logs (.json) and system reports (.txt) are generated and stored locally on your machine.
- **Visual Clarity** The Text-based User Interface (TUI) uses high-contrast ASCII bar charts to visualize CPU and RAM usage, making it easier to spot resource hogs at a glance compared to dense numerical tables.

VIII. Potential Enhancements

Sentry's current architecture provides robust functionality for real-time monitoring and manual deep process control. However, several enhancements could significantly expand its capabilities from a "monitoring tool" to a fully automated "system administrator." One major improvement would be implementing **remote management**, allowing Sentry to monitor processes across a network using PowerShell Remoting (Invoke-Command). Additionally, integrating **automated thresholds** could transform Sentry into a true background guardian that takes action without user intervention. Below are the most impactful enhancements that could be implemented in future versions:

- **Remote System Monitoring**

  Implement PowerShell Remoting to allow Sentry to connect to, monitor, and manage processes on other computers within the same local network, turning it into a centralized administration console.

- **Automated "Sentinel" Mode**

  Introduce a background service mode where Sentry monitors specific thresholds (e.g., "If CPU usage > 90% for 5 minutes") and automatically throttles or terminates the offending process based on user-defined rules.

- **Network Geo-Location**

  Integrate a lightweight API call (like ip-api.com) into the Active Traffic Scanner to resolve remote IP addresses to physical Countries and Cities, helping users instantly identify suspicious foreign connections.

- **Service Dependency Visualization**

  Enhance the Service Manager to display a "Dependency Tree" before stopping a service, warning the user if stopping "Service A" will accidentally crash "Service B" and "Service C."

- **Enhanced Data Exporting**

  Expand the Activity Logger and Reporting modules to export data in .CSV or .HTML formats (with charts), allowing for easier analysis of long-term usage trends in external tools like Excel or PowerBI.