WinChangeMonitor

Background

In modern computing environments, system changes introduced by third-party executables and scripts are both common and consequential. These modifications affect files, folders, and registry entries, often without transparency, especially when the underlying code is closed-source. Existing monitoring tools such as Microsoft Sysinternals' Process Monitor or various open-source alternatives demonstrate the importance of tracking system changes, but they frequently fall short in two key areas: accessibility and portability. Many require expert knowledge to interpret low-level outputs, and most are tied to specific operating system versions, limiting their use in specialized environments such as Windows Embedded or Windows IoT.

Academic research in digital forensics and system auditing highlights the critical role of capturing registry and file system changes in supporting incident response, malware detection, and compliance audits. Yet, there remains a gap between highly technical forensic-grade tools and practical, user-friendly solutions for everyday use.

WinChangeMonitor aims to fill this gap by combining the rigor of forensic monitoring with an intuitive interface that broadens accessibility for both technical and non-technical users. Unlike existing tools, it emphasizes:

- Ease of use through clear, human-readable reporting.
- Cross-version compatibility, including specialized Windows environments.
- Actionable insight, reducing guesswork and risk when running unverified or legacy software.

By addressing these shortcomings, WinChangeMonitor positions itself not just as another monitoring tool, but as a practical bridge between complex system auditing solutions and the needs of everyday developers, security analysts, and administrators.

Project Goals and Objectives

Modern Windows systems are constantly modified by third-party executables and scripts, many of which are closed-source and offer little visibility into the changes they perform. For developers, security analysts, and system administrators, this poses a serious problem because they frequently have to make educated guesses about how such programs impact files, folders, and registry entries. These opaque modifications increase security risks, complicate troubleshooting, and hinder system compatibility testing.

The goal of this project is to offer a dependable method for recording and examining every system-level modification made by outside applications. The tool guarantees transparency and lowers risks when working with unverified or legacy software by tracking changes in real time and displaying them in an easy-to-use interface.

The specific objectives of this project are listed as follow:

- Automating file system and registry change detection.
- Producing thorough, readable reports of system changes/adjustments.
- Ensuring compatibility across multiple Windows environments.
- Offering a user-friendly interface for effective reporting and monitoring.

This project intends to empower stakeholders by eliminating uncertainty around third-party executables and enabling the safe documentation, replication, and portability of system modifications across different

Windows versions (e.g., Windows Embedded or Windows IoT). This prolongs the lifecycle and usability of crucial software in a variety of settings while also improving security and dependability.

Major Functionality

The project's core functionality revolves around the performance of an inventory capture before and after a third party executable or script is run followed by a report highlighting the differences. These steps are summarized as follows:

1. ***Baseline Inventory Creation***: The program records relevant system information, including registry entries, file attributes, and directory structures.
2. ***User Execution***: The user runs the third-party executable or script.
3. ***Post-execution Inventory***: Once execution is finished, a second inventory is performed.
4. ***Difference Analysis***: The two inventories are compared, and all detected changes are reported.
5. ***Reporting and Exporting***: Reports can be exported for documentation, auditing, or migration needs, and the results are displayed in an orderly, user-friendly manner.

Technologies

The project will leverage the following technologies to ensure robustness and usability:

- ***C# and .NET Framework 4.8***: This is the core programming environment that will be used for developing the WinForms Application.
- ***WinForms***: It provides a user-friendly graphical interface for configuration, execution, and visualization of results.
- ***Git***: It will be used with GitHub for software development collaboration and version tracking/control across the team.
- ***Windows APIs***: They are utilized to access useful system functions for file, directory, and registry monitoring.
- ***JSON/XML***: They will be used for exporting inventory data and reports in a structured, portable format.
- ***SQL***: A database might be needed for storing system inventory information.

Project Roles and Responsibilities

Project manager will be responsible for setting up GitHub for code collaboration as well as JIRA for project management and to ensure all tasks are completed before each sprint deadline. JIRA will serve as our primary tool for logging tasks in each sprint as we practice the Agile methodology. The software developers will be responsible for writing a significant portion of the code and assigning sub-tasks to the rest of the team depending on project needs. UX designer will work closely with the software developers to build the interface which will display the final outputs and will later be presented at the end of the project. The quality engineer will ensure proper code practices are maintained across the project. The quality engineer in collaboration with the developers, will write the necessary tests (unit, contract, integration, etc.) or documentation to maintain code functionality. Each team member is given a primary role in addition to contributing duties that support the primary roles of others in order to guarantee equitable and balanced work distribution. This approach avoids concentrating the workload on one person while encouraging accountability, cooperation, and cross-functional learning. The only exception is the position of Project Manager/Configuration Management, which is given to just one person in order to preserve coordination and clear leadership. Lastly, in order to ensure that every team member contributes to both the content and the review process, all reports produced as part of project management will be created collaboratively.

The table below illustrates the distribution of roles and responsibilities within the team:

	Princely Oseji	***Jeff Rose***	***Yeryoung Kim***	***Anjian Chen***	***Yu Wu***
Project Manager/Configuration Management	Primary	-	-	-	-
Software Development	Contributing	Primary	Contributing	Contributing	Primary
UX Designer	Contributing	Contributing	Primary	Contributing	Contributing
Quality Engineer	Contributing	Contributing	Contributing	Primary	Contributing