Carnegie Mellon University
Software Engineering Institute

Do You Know What Your Software Is Actually Doing?

Use Silent Sentinel to Evaluate Software Before Release or Deployment

BEFORE INSTALLING TRUSTED SOFTWARE, SYSTEM OWNERS MUST ASSESS THE RISK it presents and the impact it will have on the environment where it will be installed. We at the Software Engineering Institute (SEI) designed Silent Sentinel to streamline and automate this analysis to help teams evaluate the behavior of the following: software, development frameworks, application programming interfaces (APIs), or libraries under consideration in a development project.

By automating the analysis process and producing consistent output for analysis, Silent Sentinel helps teams answer questions like the following:

- How much memory and disk space should we provision for this application? How many CPU cores?
- What files should we expect the software to add, modify, or delete?
- How many processes does the software create when it runs? What are they?
- What is the baseline for network communication?

Streamlining Quality Assurance

Most organizations require some level of quality assurance (QA) testing before an application can be deployed into a production environment. Some organizations require a deeper level of testing and more in-depth data, such as a risk analysis assessment, a list of dependencies for the software, and information about the supply chain. The aim of a risk analysis assessment is to go beyond what a software application is supposed to do and evaluate how that application may affect the computer system or environment where it is deployed.

To perform a risk analysis assessment, testers evaluate different aspects of the tool's execution relating to functional, operational, and security metrics. When this is done manually, the evaluation results are highly dependent on the tester's skill, and results can be inconsistent, both in the amount of effort required to complete the tests and in the accuracy of test results. Inconsistent results can negatively affect risk analysis, and poor or incomplete risk analysis can negatively affect the assessment's accuracy.

By automating such testing, Silent Sentinel creates a unified set of data that teams can reference and update over time to not only support risk assessment activities but also to serve as a baseline for evaluating future proposed changes.

Assessing Frameworks

Software developers with code security requirements must often incorporate one or more development frameworks into code to minimize development time. Silent Sentinel enables developers to compare different frameworks by providing objective metric data that they can analyze. This data helps developers determine whether the desired performance and functionality requirements for their use case are met.

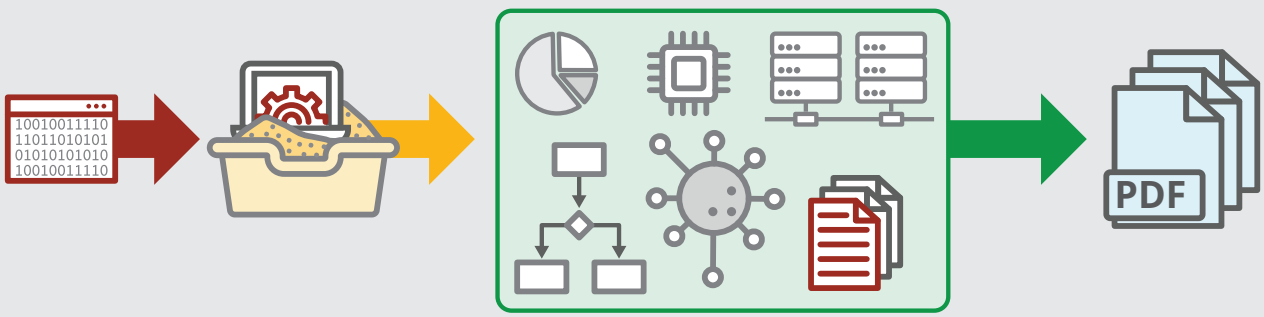
Interpreting the Results

When evaluating the static characteristics and the dynamic impact of software, many factors come into play. Silent Sentinel comes with an Interpretation Guide to help end users understand the objective data collected and compiled into the report.

This guide presents example questions users may wish to contemplate and commentary to help users understand the report when testing and evaluating software. For each section of the report, the Interpretation Guide also provides (1) a description of why the information in that section is useful for detecting unwanted behavior and (2) a sample of report output data.

How It Works

Silent Sentinel works by following this process:



It runs a battery of tests on the software in a Linux-based containerized sandbox environment.

It collects and stores the test results.

It synthesizes the results into a report for review.

During the automated tests, Silent Sentinel collects a wealth of information, including the following:

- processes
- system calls
- changes to files
- memory usage
- CPU usage
- virus scan results
- packet captures
- network configurations

Silent Sentinel then makes this information available to system owners to help them understand how the software will affect their systems. This information also provides data to help teams establish a baseline of expected behavior that subsequently helps them detect any future anomalous or unwanted behavior.

Supported Systems

Silent Sentinel easily integrates into a program's development pipeline. It is currently supported on Linux systems with ARM or x86 architectures. Silent Sentinel's sandbox is a Linux-based containerized environment. It supports multiple base container images, including Debian, Rocky Linux, Alpine, and Arch Linux.

Ready to Learn More About Silent Sentinel?

Download the code from GitHub and try it in your environment: https://github.com/cmu-sei/silentsentinel.

For more information, contact us at silent-sentinel@sei.cmu.edu.

About the SEI

Always focused on the future, the Software Engineering Institute (SEI) advances software as a strategic advantage for national security. We lead research and direct transition of software engineering, cybersecurity, and artificial intelligence technologies at the intersection of academia, industry, and government. We serve the nation as a federally funded research and development center (FFRDC) sponsored by the U.S. Department of Defense (DoD) and are based at Carnegie Mellon University, a global research university annually rated among the best for its programs in computer science and engineering.

Contact Us

CARNEGIE MELLON UNIVERSITY
SOFTWARE ENGINEERING INSTITUTE
4500 FIFTH AVENUE; PITTSBURGH, PA 15213-2612

sei.cmu.edu
412.268.5800 | 888.201.4479
info@sei.cmu.edu