BinStop
(An binary executable behaviour analysis tool)

Synopsis submitted to
Shri Ramdeobaba College of Engineering & Management, Nagpur
in partial fulfillment of requirement for the award of the degree of

Bachelor of Technology (B.Tech)

In

COMPUTER SCIENCE AND ENGINEERING (Cyber Security)

By

Aditya Bahe (28)
Ayush Paranjale (25)
Gunangi Bhagat (3)
Himanshu Pawar (43)
Prabhu Kalantri (55)

Guide

Prof. Koushik Roy

RCOEM
Shri Ramdeobaba College of Engineering and Management, Nagpur

Department of Computer Science and Engineering – Cyber Security
Shri Ramdeobaba College of Engineering & Management, Nagpur 440 013
(An Autonomous Institute affiliated to Rashtrasant Tukdoji Maharaj Nagpur University Nagpur)

July 2025

PROBLEM STATEMENT

Knowing how an application interacts with the operating system is essential in domains such as security behavior analysis. Applications use system calls (syscalls) to ask the operating system for services, but it can be difficult to manually track and analyze these syscalls in large, complex systems. The depth and interactivity required for completely understanding the behaviors and performance impacts of these syscalls are often not present in current tools, which results in inefficient analysis and pushed back debugging.

PROBLEM DESCRIPTION :

Debugging, performance optimisation, and security analysis today's software development all depend on understanding of how an application interacts with the operating system in order to request different services from the OS, like reading files, allocating memory, or interacting with hardware, applications mainly rely on system calls, or syscalls. However, manually monitoring and deciphering these syscalls becomes a difficult and time-consuming task in large and complex applications. Debugging and optimisation are inefficient because current tools often lack the depth and automation needed to fully analyse these interactions. Furthermore, these tools might not be able to identify unusual activity or possible security-threats. The goal of this project is to create a syscall tracer that will record, log, and examine syscalls made by executable programs By adding thorough context to the logs. The tracer will assist developers and security experts in learning more about application workflows, identifying inefficiencies, and detecting malicious activity by using tools such as Large Language Models (LLMs) for deeper insights. The goal of this tool is to provide a more thorough, effective, and perceptive method of examining how applications communicate with the operating system which will ultimately enhance security monitoring, performance tuning, and debugging

PROJECT OBJECTIVES :

- *Intercept and Trace Syscalls*: Record and intercept system calls from programs, noting important information like timestamps, parameters, and return values.
- *Improve Log Context*: For better analysis, add more context to syscall logs, such as function names, source locations, and expected behaviors.
- *Examine Syscall Patterns*: Use syscall sequence analysis to find performance snags, redundancies, and odd behaviors.
- *Leverage LLM for Insights*: From enriched syscall logs, extract high-level insights, workflows, and intent by using Large Language Models (LLM).
- *Create Reports*: Create actionable, readable reports that highlight security threats, performance problems, and syscall behavior.
- *Support Debugging and Optimization*: Help developers find and address problems relating to faulty or inefficient syscalls, thereby improving system performance.
- *Improve Security Monitoring*: Look for strange syscall patterns that might point to malicious activity or security flaws.
- *Enable Reverse Engineering*: By tracking syscalls and discovering operational intent, this technique assists in the analysis and understanding of application workflow

METHODOLOGY :

- Syscall Tracing: When the application or EXE file launches, the syscall tracer is launched. Using hooking functions to intercept system calls while they are being executed, the tracer (shown as the Syscall Tracer Core in the flow) records syscalls made by the application.

- Logging and Database Storage: After being intercepted, the syscalls are recorded and kept in a database for later use. Function parameters, return values, and timestamps are among the crucial information found in logs. To add context, these logs are enhanced with extra descriptive data (such as function names and expected behavior).
- Log Enhancement: After undergoing additional processing to include thorough descriptions, the logs are saved in the database for later retrieval and deeper insights.
- LLM Analysis: To produce high-level insights, identify patterns, and find any problems —such as inefficiencies or questionable activity—the LLM examines the enriched logs. Rebuilding workflows and comprehending the purpose of the syscalls are aided by this step.
- Report Generation: The results of the log analysis and LLM insights are compiled in a final report. This covers the behavior or workflow of the application, identified problems, performance snags, and possible security threats.

TECHNOLOGY :

- The Detours Library is used in C++ to hook and intercept system calls.
- *SQLite/MySQL*: Used to store and handle enriched data and captured system call logs.
- *Custom Parsing Logic*: To add more context to logs, like function names and expected behavior.
- *OpenAI GPT (LLM)*: For syscall intent comprehension, insight generation, and enriched log analysis.
- *Python and C++*: Python for log analysis and report generation, and C++ for syscall tracing.
- *SQL*: Used to manage and query database syscall data.

FUNTIONAL SPECIFICATION :

- Syscall logs are kept in a database and improved with information about expected behaviors, function names, and source code locations.
- The enriched logs are analyzed by a Large Language Model (LLM) to find patterns and workflows as well as problems like performance difficulties or security threats.
- Based on the analysis, the tool produces thorough reports that point out inefficiencies and security issues.
- By identifying challenging syscalls, developers can use the insights to debug problems and improve system performance.
- The program keeps an eye on syscall activity to spot odd or unauthorized activity, helping in spotting possible security risks.

Approved by:

Head of Department	**Guide**
Cyber Security	Prof. Koushik Roy