Malware Analysis

File Actions & Events

15 February, 2024

Team Members

Xavier Christian

Raven Mott

Jihad Turner

Introduction

Our project aims to analyze, strategize, and implement techniques to to review contents of a file to understand if it has malicious intents when executed on a host.

Background Project

Current scenario:

Malware analysis tasks may be timely when reviewing a file type and discovering its true actions in it’s malicious intent when executed on a system.

Methodology

Data Collection:

- Utilize multiple systems to compute different tasks
- Based those actions, logs from the events and actions of the file will be gathered

Analyze Data

- Data will then be analyzed by AI via API calls
- Based on the response a documentation of the response will be created and sent to the controller

Testing and Refinement:

- The documentation will be gathered and sent back to the client for review

Architecture

Client machines

Client machines upload files that may be deemed suspicious.

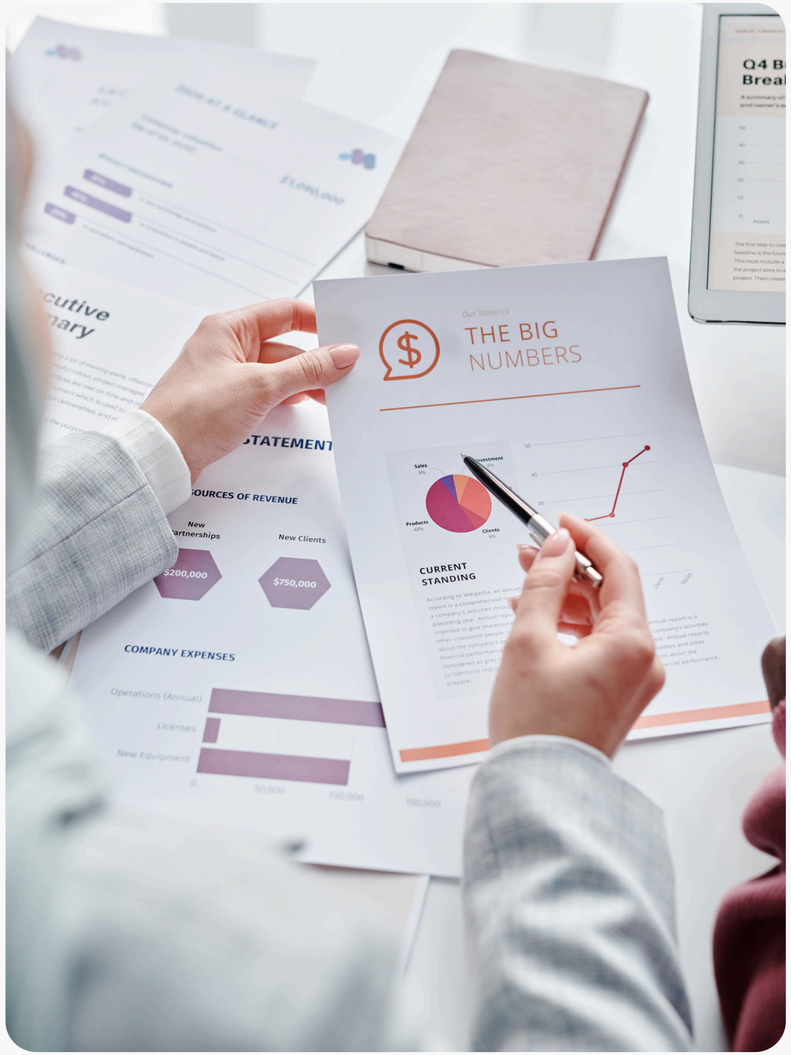
Controller

The controller server oversees the distribution of workloads.

Worker nodes

Worker nodes, which are independent machines, analyze various aspects of the files, including hash verification, dynamic behavior analysis, and reverse engineering.

Expected Outcomes



Node 1:

- Upload a file to a docker image and review the actions of the file based on sysmon logs
- Logs those actions and package them into a document.

Node 2

- Upload that file to file analyzers for hashes to review potential threats from the file
- Document the actions of that file if hash is similar

Node 3:

- Dump the file into a dissembler and understand malicious actions of the file through tools like objdump and others.

Optimizing Performance with Multiprocessing & Multithreading

Malware analysis tasks are CPU-intensive, so we leverage multiprocessing and multithreading:

- Multiprocessing → Handles parallel execution across CPU cores.
- Multithreading → Optimizes I/O-bound operations (e.g., file reading, network communication).

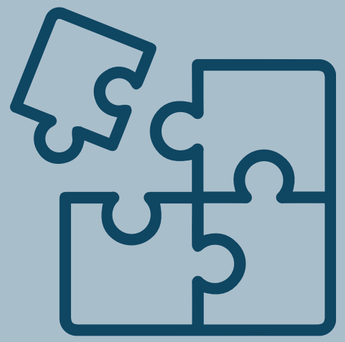
Example Implementation:

- Each worker node spawns multiple processes (one per CPU core).
- Inside each process, threads handle different analysis task concurently

Project Objectives

Analysis Phase

- Analyze time performance, accuracy, and analysis behavior.



Strategy Development

- Leveraging dockers and threads to optimize actions.
- Part out tasks to improve time

Implementation Plan

- Understand malware analysis and complications
- Developing a docker image for analysis

Conclusion

By implementing a well-researched and innovative malware analysis strategy we hope to decrease time consumption in analyzing files for malicious intent when executed on a system.

Thank you