SOFTWARE REQUIREMENTS SPECIFICATION

for

Malware Detection Automation System

Prepared by:

IIIT ALLAHABAD TEAM

Team Members:

Swaroop Dora, Piyush Priyadarshi, Avanish Gurjar, Ajay Kumar, Gaurav Singh Painkara

Submitted to:

Punjab National Bank

February 27, 2025

Contents

1 Introduction

1.1 Purpose

The purpose of the Malware Detection Automation System is to provide an automated, robust solution for detecting and analyzing malware threats through in-depth static and dynamic analysis. This system aims to enhance cybersecurity by identifying malicious files, monitoring their behavior, and mitigating risks efficiently. It employs advanced techniques such as AI-powered static code inspection, behavioral anomaly detection in a sandbox environment, and network threat intelligence to ensure comprehensive threat protection. The ultimate goal is to safeguard systems and networks by preventing malware execution, quarantining threats, and notifying users of potential risks.

1.2 Intended Audience and Reading Suggestions

This SRS is intended for a diverse group of stakeholders involved in the development, deployment, and use of the Malware Detection Automation System, including:

- Software Developers: Responsible for implementing the system's features and algorithms.
- System Architects: Tasked with designing the system's modular structure and integration points.
- Cybersecurity Analysts: Users who configure the system and interpret its analysis results.
- System Administrators: Personnel responsible for installation, maintenance, and updates.
- Project Managers and Stakeholders: Individuals overseeing project scope, timelines, and resource allocation.

1.3 Project Scope

The Malware Detection Automation System is designed to automate the detection and analysis of malware files through a combination of static and dynamic methods. The scope includes:

- Static Analysis: Examining malware files without execution, using techniques like malicious signature detection and file structure analysis.

- Dynamic Analysis: Observing malware behavior in a secure sandbox environment to identify suspicious activities.
- Network Threat Intelligence: Monitoring network traffic to detect potential threats.
- AI and Machine Learning Integration: Enhancing detection accuracy and adapting to evolving threats.
- Threat Response: Taking actions such as preventing execution, quarantining files, or notifying users based on analysis outcomes.

The system is limited to automated processes and does not include manual threat analysis or unrelated cybersecurity functions outside the described workflow.

1.4 References

This SRS references the following documents, standards, and resources:

1. National Institute of Standards and Technology (NIST) Special Publication 800-83: "Guide to Malware Incident Prevention and Handling for Desktops and Laptops"
2. OWASP Automated Threat Handbook
3. ISO/IEC 27001:2013 - Information Security Management
4. MITRE ATT&CK Framework
5. Punjab National Bank's Cybersecurity Standards and Compliance Requirements
6. CERT Malware Analysis Fundamentals
7. SANS Institute's Windows Forensic Analysis

1.5 Definitions and Acronyms

- **APT (Advanced Persistent Threat):** Stealthy threat actors who gain and maintain unauthorized access to systems over extended periods.
- **ASCI (AI-Powered Static Code Inspection):** Component that performs static analysis using artificial intelligence techniques.
- **ATL (Adaptive Threat Learning):** System capability to improve detection through continuous learning from new threats.
- **BAD (Behavioral Anomaly Detection):** Component that identifies abnormal behaviors during dynamic analysis.
- **C2 (Command and Control):** Communication channels used by attackers to control compromised systems.

- **IOC (Indicator of Compromise):** Forensic evidence indicating a security breach.
- **IQS (Intelligent Quarantine System):** Component responsible for isolating detected malware.
- **ML (Machine Learning):** Artificial intelligence techniques enabling systems to learn from data.
- **NTI (Network Threat Intelligence):** Component monitoring network traffic for suspicious activities.
- **STP (Statistical Threat Profiling):** Component using statistical methods to identify threat patterns.
- **SOC (Security Operations Center):** Facility housing security analysts who monitor and defend against cybersecurity threats.
- **SRS (Software Requirements Specification):** Document specifying requirements for a software system.
- **VM (Virtual Machine):** Isolated software environment used for secure analysis.
- **YARA:** Pattern matching tool used for malware identification and classification.

2 Overall Description

2.1 Product Perspective

The Malware Detection Automation System is a standalone cybersecurity solution that integrates seamlessly into existing IT and network security infrastructures. It serves as a critical layer of defense by automating malware detection and analysis, complementing tools like firewalls and antivirus software. The system's workflow begins with network monitoring, progresses through detailed file analysis, and culminates in automated threat mitigation, positioning it as an advanced, proactive component of a layered security strategy.

2.2 User Classes and Characteristics

The system supports the following user classes:

- Cybersecurity Analysts: Technical experts who configure analysis parameters, review results (e.g., threat notifications), and respond to detected malware. They require a deep understanding of malware behavior and system outputs.
- System Administrators: IT professionals responsible for deploying, maintaining, and updating the system. They ensure compatibility with the operating environment and handle system performance.
- End Users: Individuals or organizations protected by the system, who benefit from its automated threat mitigation without direct interaction.

Each class interacts with the system at different levels, from configuration and maintenance to indirect reliance on its protective capabilities.

2.3 Product Functions

The Malware Detection Automation System performs the following key functions:

2.3.1 Network Traffic Investigation

Network Traffic Investigation

- Monitors network traffic in real-time using deep packet inspection
- Identifies suspicious communication patterns and potential C2 traffic
- Detects anomalous data transfers and exfiltration attempts
- Recognizes network-based attack signatures and exploits
- Correlates traffic with known malicious IPs, domains, and URLs
- Extracts files from network streams for further analysis

2.3.2 Static Code Inspection

Static Code Inspection

- Performs hash-based analysis to detect known malware signatures
- Extracts and analyzes strings using FLOSS to identify obfuscated or encoded data
- Examines imported DLLs, functions, and API calls using PEview and PESTudio
- Analyzes PE file structure, headers, sections, and embedded resources for anomalies
- Detects suspicious patterns in import tables, relocations, and entry points
- Identifies anti-analysis techniques such as packing, encryption, and code injection
- Utilizes static indicators like hardcoded IP addresses, URLs, and registry modifications

2.3.3 Dynamic Behavior Analysis

Dynamic Behavior Analysis

- Performs multi-level hash-based analysis to detect known malware signatures
- Extracts and analyzes strings using FLOSS to identify obfuscated or encoded data
- Examines imported DLLs, functions, and API calls using PEview and PESTudio
- Analyzes PE file structure, headers, sections, and embedded resources for anomalies
- Detects suspicious patterns in import tables, relocations, and entry points
- Identifies anti-analysis techniques such as packing, encryption, and code injection
- Utilizes static indicators like hardcoded IP addresses, URLs, and registry modifications
- Monitors child processes created by malware to detect process injection or evasion techniques
- Tracks changes and modifications in the Windows registry to identify persistence mechanisms

These functions work together to provide a comprehensive, automated malware defense system.

2.4 Operating Environment

The system operates within a networked IT environment, requiring:

- Hardware: The system is designed to normally run on a local server, but for better performance, it will shift to cloud-based computing when processing large datasets and running AI algorithms.
- Software: Compatibility with network monitoring tools, sandbox environments, and various operating systems (e.g., Windows, Linux).
- Network: Access to real-time traffic data for threat intelligence and a secure, isolated sandbox for dynamic analysis.
- File Support: Ability to handle diverse file types commonly associated with malware (e.g., executables, scripts, documents).

The system must integrate with existing security infrastructure and scale to handle high volumes of analysis requests.

2.5 Design

The system's design is modular and sequential, as depicted in the workflow:

- Input Module: Accepts malware files or network traffic data for analysis.
- Analysis Engine: Combines static (e.g., signature detection, file structure analysis) and dynamic (e.g., sandbox behavior monitoring) analysis, leveraging AI and machine learning.
- Decision Engine: Processes analysis results, using decision points (e.g., "Malicious Detected?") to determine outcomes.
- Response Module: Executes actions like file quarantine, execution prevention, or user notification.
- Learning Module: Incorporates adaptive learning to refine detection capabilities based on new threats.

This modular architecture ensures scalability, flexibility, and ease of maintenance.

3 System Features

3.1 Description and Priority

The system includes the following features, prioritized based on their criticality:

1. Static Analysis (High Priority): Examines files without execution to detect known threats, forming the first line of defense.
2. Dynamic Analysis (High Priority): Monitors file behavior in a sandbox, critical for identifying unknown or sophisticated malware.
3. Network Threat Intelligence (Medium Priority): Enhances detection by monitoring network activities, complementing file-based analysis.
4. AI and Machine Learning Integration (Medium Priority): Improves detection accuracy and adaptability to new threats over time.
5. Threat Response and Notification (Low Priority): Provides actionable outcomes and user feedback, supporting operational awareness.

3.2 System Architecture Diagram

3.3 Functional Requirements

The system must meet the following functional requirements:

- FR1: Accept malware files or network traffic data as input for analysis.
- FR2: Perform static analysis to detect known malicious signatures and analyze file structures.
- FR3: Conduct dynamic analysis in a sandbox environment to observe and identify suspicious behaviors.
- FR4: Integrate AI and machine learning to enhance detection accuracy and adapt to evolving threats.
- FR5: Evaluate analysis results and execute decisions, such as preventing execution, quarantining files, or allowing safe operations.
- FR6: Notify users of detected threats with detailed reports, including analysis outcomes and recommended actions.

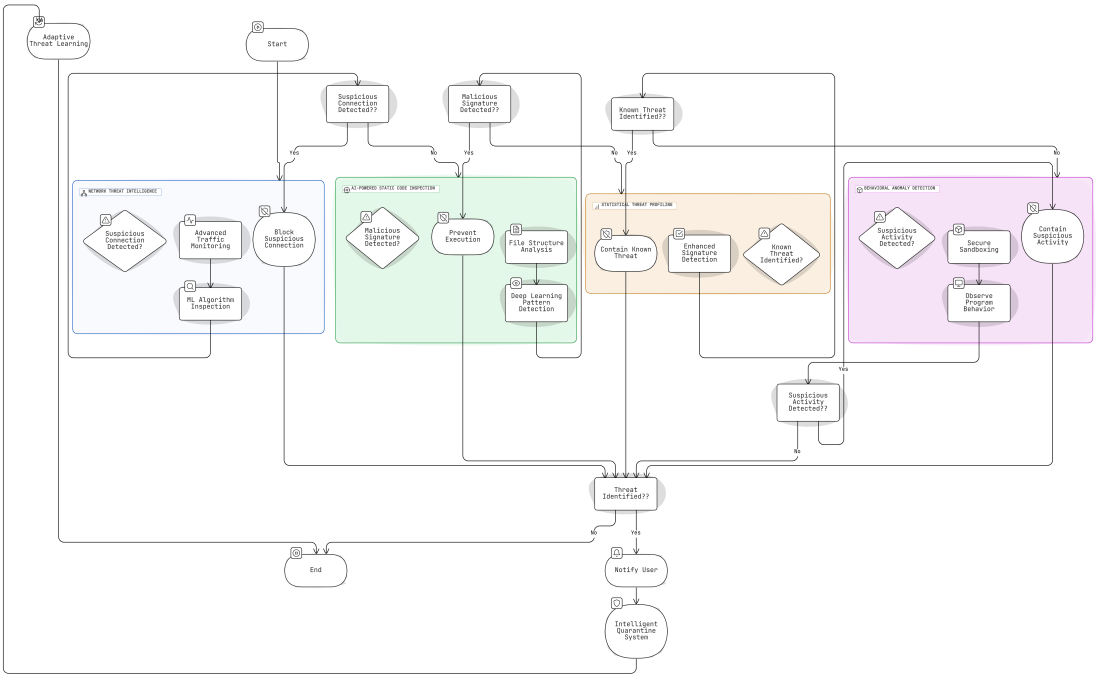

Figure 3.1: System Architecture Diagram

4 Nonfunctional Requirements

4.0.1 Software Requirements

The system is designed to run on multiple platforms and requires the following software:

Supported Operating Systems

- **Enterprise Servers:**
  - Sun Solaris 14
  - IBM AIX
  - Windows Server 2012, 2016, 2019
  - RedHat Linux
  - Ubuntu
- **Endpoint Devices:**
  - Windows 11, 10, 8, XP
  - macOS

Malware Detection Software

- Behavior-based malware detection engine
- Signature-based threat analysis
- AI-powered anomaly detection
- Sandboxing for suspicious files

Security Tools and Libraries

- **Antivirus & EDR Solutions:** Windows Defender ATP, CrowdStrike, McAfee
- **Network Security:** IDS/IPS (Intrusion Detection/Prevention Systems)
- **SIEM (Security Information & Event Management):** Splunk, IBM QRadar

Development and Deployment Frameworks

- **Programming Languages:** Python, html, css, javascript
- **Machine Learning Frameworks:** PyTorch, Pytorch, Flask, Scikit Learn.
- **Cloud & Virtualization Support:** VMware.

4.1 Performance Requirements

- The system must complete analysis of a single malware file within 5 minutes to ensure timely threat mitigation.
- Network threat intelligence must process traffic data in real-time, with low latency.

4.2 Security Requirements

- The system must ensure the confidentiality and integrity of analyzed data, preventing unauthorized access or leakage.
- The sandbox environment must be fully isolated from the host system to prevent malware escape during dynamic analysis.
- All system components must resist tampering or reverse-engineering to maintain operational security.

4.3 Software Quality Attributes

- Usability: Provide an intuitive interface for analysts and administrators, with clear reports and configuration options.
- Maintainability: Allow easy updates to threat signatures, algorithms, and system components to address new malware variants.
- Scalability: Support increasing workloads (e.g., more files or traffic) through modular design and resource allocation.

4.4 Business Rules

- The system must comply with data protection regulations (e.g., GDPR, HIPAA) when processing sensitive files or network data.
- It must prioritize high-risk threats (e.g., ransomware, trojans) over low-risk anomalies, optimizing resource use.
- Adaptive learning updates must be validated before deployment to ensure system stability and accuracy.

5 Other Requirements

The Malware Detection Automation System requires ongoing maintenance to adapt to new threats and technological advancements. Regular updates to algorithms, threat signatures, and system components are essential for maintaining effectiveness. Additionally, the system should be designed with future scalability in mind to accommodate growing data volumes and evolving cybersecurity needs.