****

**School of Computer Science**

**University of Petroleum & Energy Studies  
Bidholi, Via Prem Nagar, Dehradun, Uttarakhand  
November– 2024**

**IP Vulnerability Tracker**

***Project System Requirements Specification***

***Report***

*submitted in fulfillment of the   
requirements for the award of the degree of*

***under the guidance of  
Dr. Akashdeep Bhardwaj***

|  |  |  |
| --- | --- | --- |
| **Specialization** | **SAP ID** | **Name** |
| B.Tech-CSE (CSF) | 500106943 | Shreyanshi Dobhal |
| B.Tech-CSE (CSF) | 500106533 | Akshat Negi |
| B.Tech-CSE(CSF) | 500106703 | Arush Dubey |
| B.Tech-CSE(CSF) | 500101975 | Ameya Taneja |

# System Requirements Specification (SRS)

## 1. Introduction

### 1.1 Purpose

The purpose of this SRS document is to define the requirements for the **Vulnerability Tracker** system, which aims to analyze Internet Protocol (IP) data for detecting and mitigating security threats within a network.

### 1.2 Scope

The Vulnerability Tracker will provide a comprehensive analysis of IP traffic, integrating multiple scanning tools and APIs to identify vulnerabilities and improve network security. It will support real-time monitoring and provide insights into various security metrics.

### 1.3 Definitions, Acronyms, and Abbreviations

* **IP**: Internet Protocol
* **API**: Application Programming Interface
* **DDoS**: Distributed Denial of Service
* **SSL**: Secure Sockets Layer
* **DNS**: Domain Name System
* **SRS**: System Requirements Specification

## 2. Overall Description

### 2.1 Product Perspective

The Vulnerability Tracker will consist of:

* A frontend interface built with ReactJS.
* A backend developed with Node.js and Express.
* A MongoDB database for data storage.
* Integration with external APIs (Shodan, Whois, Rapid API) for data retrieval and analysis.

### 2.2 Product Functions

The system will include the following functionalities:

* Real-time threat detection and monitoring.
* Data collection and analysis from multiple APIs.
* User interface for visualizing security metrics.
* Reporting and logging of security incidents.

### 2.3 User Classes and Characteristics

* **System Administrators**: Responsible for configuring and maintaining the system.
* **Network Security Analysts**: Utilize the system for monitoring network traffic and analyzing threats.
* **End Users**: Use the system to view reports and insights regarding network security.

### 2.4 Operating Environment

The system will operate in a cloud-based environment and should be compatible with:

* Major web browsers (Chrome, Firefox, Safari).
* Windows, macOS, and Linux operating systems.

### 2.5 Design and Implementation Constraints

* Compliance with data protection regulations
* Performance must be optimized for large-scale networks.

## 3. Functional Requirements

### 3.1 User Interface Requirements

* The system shall provide an interactive dashboard displaying real-time metrics related to IP security.
* Users shall be able to navigate between different sections, such as IP Information, SSL Chain, DNS Records, and Security Alerts.

### 3.2 Data Processing Requirements

* The system shall analyze incoming IP traffic data in real time.
* The system shall retrieve data from external APIs and aggregate it for analysis.

### 3.3 Security Requirements

* The system shall implement authentication mechanisms for user access.
* The system shall log all security incidents and user activities for auditing purposes.

### 3.4 Reporting Requirements

* The system shall generate reports summarizing security incidents and trends.
* Users shall have the option to export reports in PDF and CSV formats.

## 4. Non-Functional Requirements

### 4.1 Performance Requirements

* The system shall handle up to 10,000 concurrent users without performance degradation.
* Response time for data retrieval and analysis shall not exceed 2 seconds.

### 4.2 Reliability Requirements

* The system shall have an uptime of 99.9%.
* Data backup and recovery processes shall be implemented to prevent data loss.

### 4.3 Usability Requirements

* The system shall have a user-friendly interface with clear navigation.
* User documentation and help resources shall be provided.

### 4.4 Scalability Requirements

* The system shall be designed to scale horizontally to accommodate increased data loads and user traffic.

## 5. System Architecture

### 5.1 Overview

The architecture of the Vulnerability Tracker will consist of a client-server model with distinct layers for presentation, application logic, and data storage.

### 5.2 Components

* Frontend: Developed using ReactJS, responsible for user interaction.
* Backend: Handling API requests and business logic.
* Database (Future Work): MongoDB for storing collected data and logs.

## 6. Future Enhancements

* Integration of Web Vulnerability Exploitation Tools: Incorporate tools for actively testing website resilience against various attack vectors, enhancing the overall security posture.
* Comprehensive Database Implementation: Develop a robust database schema to store all data retrieved from APIs. This will allow for efficient historical comparisons of security metrics, enabling the system to identify trends, anomalies, and potential vulnerabilities over time.
* Data Versioning: Implement versioning for stored data to track changes and facilitate comparison with past results, improving the accuracy of vulnerability assessments.
* Enhanced Reporting Features: Add advanced reporting functionalities that leverage the stored historical data for in-depth analysis and visualizations of security trends.