SentinelShield: Advanced Intrusion Detection & Web Protection System – Practical Work Documentation

1. INTRODUCTION

This document provides detailed practical work documentation for the project SentinelShield.

This system acts as a simplified but realistic Intrusion Detection and Web Protection System, helping

students understand how threat detection, request inspection, and alert generation work in real-world

cybersecurity environments.

The project simulates the behavior of a lightweight Web Application Firewall by inspecting incoming requests,

detecting malicious intent, monitoring abusive traffic, analyzing logs, and generating alerts. This documentation

explains each component in-depth, focusing on concepts, workflows, and expected student activities during practical work.

2. PROJECT SCOPE AND PRACTICAL OBJECTIVES

The practical work aims to ensure the student develops a hands-on understanding of web security mechanisms.

By the end of the practical sessions, the student should be able to:

1. Understand how modern WAFs detect threats using patterns, signatures, and rule engines.

2. Analyze HTTP requests from a security perspective.

3. Identify common web attacks using manual and automated testing techniques.

4. Document findings and generate test reports.

5. Create meaningful analysis logs and summaries.

6. Understand the workflow of attack detection → decision → logging → alerting → dashboarding.

7. Demonstrate how rate limiting and traffic behavior analysis help prevent brute-force or flooding attacks.

This project is designed to emulate a realistic work environment where students configure tools, analyze data,

interpret results, and produce documentation similar to real cybersecurity operations.

3. PRACTICAL CONCEPTS AND BACKGROUND

This section provides detailed explanations of the core concepts students will interact with during the practical work.

A. HTTP Request Inspection

Students learn how incoming requests contain headers, parameters, and body content that may reveal signs of an attack.

They are expected to break down and interpret sample HTTP requests and identify suspicious elements such as encoded payloads,

unexpected characters, or command patterns.

B. Attack Signature Identification

Students examine how typical signatures for SQL Injection, XSS, LFI, directory traversal, and command injection look.

They analyze example payloads and understand what makes a request malicious.

C. Behavior Monitoring and Rate Limiting

Students observe how repeated identical or patterned requests indicate automated scanning or brute-force activity.

They understand concepts like request thresholds, time windows, and IP-based abuse tracking.

D. Log Analysis

Students manually inspect log entries and categorize alerts.

They learn how timestamps, IP addresses, and detection labels contribute to incident understanding.

E. Dashboard Interpretation

Students work with static or dynamic dashboards summarizing events.

They learn to read charts, identify trends, and interpret real-time attack metrics.

4. PRACTICAL WORKFLOW FOR STUDENTS

This section describes the workflow students must follow while performing the practical tasks.

Step 1: Understanding the System Architecture

Students begin by drawing or reviewing the architecture diagram to understand how the WAF, Analyzer, Logging System,

and Dashboard interact.

Step 2: Reviewing Rule Definitions

Students examine predefined attack signatures and learn how they represent malicious behaviors.

Step 3: Simulating HTTP Requests

Using tools such as curl, browser-based forms, or a simple test application, students manually submit requests.

These include:

- Normal, harmless requests

- Malicious test payloads

- Repeated requests to simulate brute-force behavior

Step 4: Observing Detection Behavior

Students compare normal vs. malicious request outcomes. They document:

- Which requests were blocked

- What detection category triggered the alert

- How the system responded

Step 5: Log File Examination

Students open the log files and interpret entries. They must identify patterns, suspicious IP activity, and frequency of attacks.

Step 6: Reporting and Analysis

Students prepare a practical report including:

- Attack attempts submitted

- Detection results

- Analysis of security logs

- Observed patterns

- Suggested improvements to rules

This process mirrors real-world SOC and application security workflows.

5. DETAILED SYSTEM WORKING (NON-TECHNICAL EXPLANATION)

The following describes each system component in a detailed, practical manner suitable for student learning.

A. Request Processing

When a request arrives, the system interprets every part of it: URL, parameters, headers, and optional body.

It checks for unusual strings, suspicious filenames, encoded characters, or command keywords.

B. Rule-Based Detection

Each rule reflects a known attack or technique.

When the system finds content matching these patterns, it marks the request as suspicious or malicious.

C. Traffic Monitoring

The system keeps track of how many requests each IP sends in a certain period.

If an IP crosses the allowed threshold, it is considered abusive.

D. Alert Decisions

Once a request is marked as malicious or abusive, the system follows a decision sequence:

- The request may be blocked.

- The event is logged.

- The IP may be temporarily or permanently flagged.

- An alert is generated for dashboard visibility.

E. Logging and Dashboard Visibility

Every blocked or flagged request appears in the log.

The dashboard summarizes key insights such as attack counts, types, and IP details.

Students analyze these outputs during practical sessions.

6. PRACTICAL OUTPUT EXPECTATIONS

The practical work produces several observable outputs:

1. Attempted attack requests and their detection results.

2. System messages indicating whether each request was allowed or blocked.

3. Log entries containing timestamps, IP addresses, and detection categories.

4. A summary table showing:

- Number of malicious requests detected
- Distribution by attack category
- IP addresses repeatedly flagged

5. Student-created interpretation notes explaining:

- Why certain attacks were detected
- How behavior analysis contributed to rate-limiting decisions
- Security recommendations based on observed patterns

These outputs form the basis of the student's practical submission.

7. STUDENT ASSIGNMENT REQUIREMENTS

Students must complete and submit the following components:

1. A detailed practical journal containing:

- Purpose of the experiment
- Tools used
- Step-by-step execution
- Observations with screenshots
- Interpretation of logs

2. A final report summarizing:

- Total attacks performed
- Detection accuracy
- False positives and false negatives

- Improvements they suggest

3. Architecture and workflow diagrams drawn manually or digitally.

4. Dashboard screenshots or recreated summary tables.

5. A reflection section explaining what they learned about:

- Web application security
- Signature-based detection
- Threat behavior analysis

8. CONCLUSION

This practical work equips students with in-depth knowledge of intrusion detection and

web application protection principles. It provides hands-on experience with interpreting requests,

recognizing attack patterns, performing behavior-based analysis, and understanding the role of monitoring

in cybersecurity operations.

By completing SentinelShield, students gain practical and analytical skills essential for careers in

Cybersecurity, SOC operations, penetration testing, and secure application development.