



SOFTWARE DESIGN SPECIFICATION

CYBERSECURITY   
  
Web Service Open to Malicious Attack

|  |  |  |  |
| --- | --- | --- | --- |
| **Created By:** | Gaurav Jaywant Moynak | **Approved By:** | Harshada Topale |
| **Created On:** | 28/05/2025 | **Approved On:** |  |

# INDEX

1. PURPOSE ..................................................................................................1

2. PROJECT SCOPE ........................................................................................1

3. SYSTEM OVERVIEW ....................................................................................1

4. DESIGN CONSIDERATIONS .........................................................................1

4.1 Requirements .......................................................................................1

4.2 Assumptions ........................................................................................1

4.3 Dependencies .......................................................................................1

5. SYSTEM ARCHITECTURE .............................................................................2

5.1 Architectural Strategies ........................................................................2

5.2 Structure & Relationships .....................................................................2

6. DETAILED DESCRIPTION OF COMPONENTS .................................................3

7. INTEGRATION ............................................................................................3

8. APPENDICES ...............................................................................................3

8.1 Appendix A – Detailed Description of Components ................................3

# **PURPOSE**

This document is created based on the requirement specification document. The purpose of this Software Design Specification (SDS) document is to describe the structure and components of the VAPT project. Though no application was developed, this document breaks down the testing components and their implementation strategies. It also supports verification and validation of all testing phases.

# **PROJECT SCOPE**

The scope of the "Web Service Open to Malicious Attack" project includes performing a structured Vulnerability Assessment and Penetration Testing (VAPT) on the public-facing Bitrix24 site: https://ccgac.bitrix24.site. The assessment identifies vulnerabilities using both automated and manual testing techniques. It aims to uncover potential security issues, demonstrate exploitation with PoC, and provide remediation strategies.

# **SYSTEM OVERVIEW**

The system under assessment is a cloud-based platform hosted on Bitrix24 that includes document sharing and CRM functionalities. The assessment is black-box, focusing only on externally visible components such as the website interface, URLs, HTTP headers, accessible endpoints, and public forms.

# **DESIGN CONSIDERATIONS**

## Requirements

Test platform must be live and reachable externally.

Tools: Nmap, Nikto, Burp Suite, curl, ApacheBench, and Python for steganography.

System must respond to basic web requests and support standard HTTP headers.

## Assumptions

Website remains stable during the assessment period.

DNS records and hosting configuration are not altered during testing.

Auth system is not accessible (handled externally by Bitrix).

## Dependencies

Bitrix24 cloud infrastructure availability.

No internal access to application source code or server logs.

All tools are to be used in ethical, non-intrusive manner.

# **SYSTEM ARCHITECTURE**

The system architecture refers to the conceptual layout of how various test components were structured and related in this security assessment. While the system was treated as a black box, the external interaction and component relationships can still be logically mapped.

**5.1 Architectural Strategies**:

* **Modular Testing Structure**: Recon, scanning, exploitation, and reporting phases were treated as distinct test modules.
* **Layered Approach**: Initial reconnaissance laid the groundwork for focused vulnerability testing.
* **Tool Diversity**: Usage of multiple tools per phase to maximize detection (e.g., Nmap + Nikto for scanning).
* **Passive-to-Active Flow**: Started with non-intrusive (WHOIS, DNS) and moved to active methods (Burp, ApacheBench).
* **Automation Augmented with Manual Efforts**: Automated results were validated manually for false positives

## Structure & Relationships

* **User Layer**: Tester (Intern) using browser or CLI tools
* **Communication Layer**: Internet with encrypted (HTTPS) communication using TLS protocols
* **Application Layer**: Bitrix24 Web Interface (Static/Interactive Elements)
* **Data Layer (Observed)**: HTML, HTTP Headers, Cookies, Page content

**Logical Flow**:

1. User initiates DNS/WHOIS lookup and mapping
2. Executes scans via Nmap/Nikto to detect server setup, open ports, headers
3. Manual analysis of responses using Burp Suite for tampering/XSS attempts
4. Performance tested with ApacheBench under low-medium load
5. Proof of Concept (PoC) for steganography executed using Python
6. Results validated and compiled into report

# **DETAILED DESCRIPTION OF COMPONENTS**

See Appendix A below.

# **INTEGRATIONS**

There was no direct integration with internal or external APIs, devices, or databases during this VAPT project. Testing involved passive scanning and interaction with public web components only. No authentication or login bypass integration was attempted due to scope constraints.

# **APPENDICES**

## Appendix A – Detailed Description of Components

**Identification:** Reconnaissance Module  
**Type:** Module  
**Purpose:** Gather open-source and technical data about the web service  
**Subordinates:** DNS lookup, WHOIS, Traceroute, Google Dorking  
**Dependencies:** Availability of internet and DNS resolution  
**Interfaces:** Command line, browser tools  
**Resources:** Internet access, terminal  
**Processing:** Collect IP/domain data and related technologies  
**Data:** DNS records, IP addresses, hostnames

**Identification:** Vulnerability Scanning Module  
**Type:** Module  
**Purpose:** Discover known vulnerabilities using tools  
**Subordinates:** Nikto, Nmap, Burp Suite  
**Dependencies:** Network access, web server response  
**Interfaces:** CLI/GUI interfaces  
**Resources:** Burp Suite, Nikto, Nmap  
**Processing:** Scan and parse responses  
**Data:** Headers, cookies, response codes

**Identification:** Steganography Component  
**Type:** Python Script  
**Purpose:** Demonstrate data hiding using image encoding  
**Subordinates:** Python, Pillow/Stegano  
**Dependencies:** PNG image, message  
**Interfaces:** Script console  
**Resources:** Pillow, input image  
**Processing:** Encode message into image and decode back  
**Data:** Image files and text conten