UNDP Mauritius - MauStat Platform
Security Risk Assessment Audit Report

Version 1.0

10 October 2024

Confidential

Status: ***FINAL***

1 Contents

1 Contents 2

2 Overview 3

2.1 Copyright 3

2.2 Confidentiality 3

2.3 Document History 3

3 Executive Summary 4

4 Key Findings 5

4.1 IT Security Governance 5

4.2 Risk Management 5

4.3 Data Security 5

4.4 Access Control and Privilege Management 5

4.5 Network Security 6

4.6 Secure Software Development Lifecycle (SDLC) 6

4.7 Logging and Monitoring 7

4.8 Incident Response 7

4.9 Environment Segregation 7

5 General Recommendations 8

2 Overview

2.1 Copyright

2.2 Confidentiality

This document contains information regarding and is **Confidential**.

2.3 Document History

Version	Day	Person	Action
0.1	1 Oct 2024	Alessio D'AMICO and Amged Mulla	First draft
1.0	15 Oct 2024	Alessio D'AMICO and Amged Mulla	Final

3 Executive Summary

The UNDP Mauritius engaged UNICC to carry out a comprehensive security assessment of the MauStats platform prior to its release into production. The primary objective of this assessment was to ensure that the necessary security controls were in place and to offer recommendations where improvements were required, especially before transitioning to Phase 2 and full production.

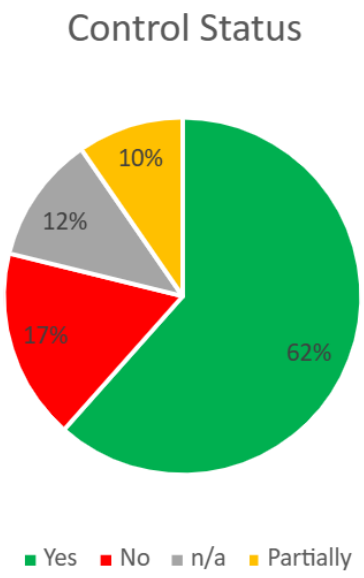
This project is expected to undergo further enhancements, with two additional implementation rounds planned for completion.

The security assessment was conducted through a series of interviews with key stakeholders, including representatives from ITM, SM, and GOC. These interviews were supplemented by a review of relevant documentation and records. Participants in the interviews included members from various teams, such as:

- Vichittra Purdassee (UNDP Project Manager)
- Rajesh Ballchand and Shashi Bullywon (GOC Representatives)
- Sameer Thapar, Riya Arora, and Kamaljit (ITM)
- Shariff, Harsha (SM)

As a result of interviewing several key stakeholders from ITM, SM, and GOC, as well as a comprehensive analysis of the documents they provided, the implementation of the controls within the defined scope can be summarized as follows:

Self-assessment Implementation Status	Control Count	Control Count (in %)
Compliant	32	62%
Partially Compliant	5	10%
Not Applicable	6	12%
Non-Compliant	9	17%



The review revealed that the MauStats solution has implemented adequate security measures for most controls; however, a significant number of control gaps were identified that fall short of industry best practices.

In addition, several findings, along with corresponding recommendations, were uncovered during the assessment. These will be outlined in detail in the following chapters.

4 Key Findings

4.1 IT Security Governance

- Lack of formalized support contracts with ITM. The absence of Service Level Agreements (SLAs) for security updates, incident response, and maintenance can lead to delays in deploying critical security patches. This can lead to a risk of data breach.

Recommendation:

- Establish formal contracts with clear SLAs for security updates, incident response and platform maintenance. Ensure vendor accountability for timely patching and support.

4.2 Risk Management

- No regular penetration testing or automated vulnerability assessments. The platform lacks a structured process for identifying and mitigating vulnerabilities.

Recommendation:

- Implement a formal risk management framework, including regular **penetration testing** and **vulnerability scanning** of both the infrastructure and the application code.

4.3 Data Security

- The MongoDB database is not fully encrypted at rest. Data encryption is critical for protecting sensitive data, especially in case of data breaches.
- Real data is being used in staging environments without proper encryption or anonymization. This violates the principles of data protection and increases the risk of data leaks.

Recommendations:

- Implement **full-database encryption** using AES-256 encryption for all sensitive data stored in the database.
- Enforce the use of **anonymized** or **synthetic data** in non-production environments. If real data must be used for testing, ensure it is fully encrypted and access is restricted.
- Implement a separate test environment which should only contain test or anonymized data (no real data should be used)

4.4 Access Control and Privilege Management

- Both **root** and **admin accounts** are actively used in both staging and production environment across multiple teams (SM, GOC, and ITM). The use of these accounts without **multi-factor authentication (MFA)** poses a significant risk of unauthorized access.

- While some **role-based access controls (RBAC)** are in place, there is a lack of proper segregation between internal users and external users (ITM), resulting in over-privileged accounts.
- Multi-Factor Authentication (MFA) is currently not implemented to secure access to the application

Recommendations:

- Implement **RBAC (Role Based Access Control)** to assign permission based on the principle of least-**privilege** access. Regularly review user roles to minimize the risk of over-privileged access and lateral movement within the network.
- Implement and enforce **Multi-Factor Authentication (MFA)** for all administrative, privileged, and VPN accounts to reduce the risk of account compromise and unauthorized access.
- Implement and enforce **Multi-Factor Authentication (MFA)** to secure access to the application
- Set up logging and monitoring to **track all actions** performed by admin and root users. This includes changes made to configurations, installation of new software, and system reboots.
- Introduce **Just-in-Time (JIT) privileged access**, which grants elevated permissions only for the duration of a task, rather than providing persistent access. This limits the time during which elevated privileges are available and reduces the potential attack window.
- Utilize **Privileged access management (PAM)** solution for controlling access for privileged users.

4.5 Network Security

- The platform lacks a **Web Application Firewall (WAF)**. This leaves the application vulnerable to common web-based attacks, such as **SQL Injection** and **Cross-Site Scripting (XSS)**.
- The VPN uses **IP-based whitelisting** and is protected only by credentials without the use of **Multi Factor Authentication (MFA)**, which increases the risk of credential theft leaving the platform vulnerable to unauthorized access.

Recommendations:

- Deploy a **Web Application Firewall (WAF)** to protect against application-layer attacks and integrate it with the existing intrusion detection/prevention systems (IPS). The integration will ensure protection at both the network and application levels, significantly reducing the risk of a breach.
- Implement MFA **for VPN access** and enforce strong encryption protocols like IPSec or SSL to protect remote access.
- Ensure that **VPN logs** are forwarded to the central **Security Information and Event Management (SIEM)** system to allow for real-time monitoring of connections, suspicious activity, or unauthorized access attempts.

4.6 Secure Software Development Lifecycle (SDLC)

- **Peer code reviews** are the current process for ensuring code security. There are no automated tools for **static** or **dynamic code analysis** as part of the CI/CD pipeline.
- No formal **secure coding standards** are enforced, increasing the likelihood of introducing vulnerabilities into the platform (e.g., **SQL Injection** and **XSS**).

- No **automated vulnerability scanning** is integrated into the CI/CD pipeline, leaving code vulnerabilities undetected before deployment.

Recommendations:

- Develop and enforce secure coding guidelines based on **OWASP** standards.
- Train developers on secure coding practices and regularly review the code for vulnerabilities.
- Integrate and automate **Static Application Security Testing (SAST)** and **Dynamic Application Security Testing (DAST)** into the CI/CD pipeline to detect vulnerabilities early in the development cycle. The scan should be carried out before deploying any new code to the production environment.

4.7 Logging and Monitoring

- There is an issue with centralized **log collection** configuration. While the logs are generated by the different application components and systems, these logs are forwarded to GOC **SIEM (QRadar)** and are not actively monitored.

Recommendation:

- Ensure that **log collection** of security logs from the Operating system, Application and Database servers is correctly configured and ingested into the **SIEM (QRadar)**. This will allow for proactive monitoring of security events and timely detection of potential attacks.
- GOC to implement use cases to detect suspicious activities.

4.8 Incident Response

- No formal **incident response plan** for handling application-specific vulnerabilities like SQL injection and XSS.

Recommendations:

- Develop a comprehensive **incident response plan** that includes procedures for detecting, responding to, and recovering from application-layer attacks. Regularly test the plan through incident response simulations.
- Ensure that all security incidents are logged, escalated, and resolved in coordination with GOC and the CERT team.

4.9 Environment Segregation

- There is no Test environment available. Only Production and Staging environment which includes partial live data.

Recommendations:

- Create a dedicated Test environment that is fully segregated from both the Staging and Production environments. The Test environment should be strictly isolated and must not contain any live data to ensure secure and controlled testing conditions

5 General Recommendations

- Engage an independent entity to conduct a penetration test on the application before it goes live. UNICC can assist with this process.
- Establish regular audits of ITM operations to ensure ongoing compliance and performance. UNICC can facilitate this.
- Perform an independent audit of the security configurations for the container and Docker image to ensure robust protection.
- Consult UNICC for a review of the Service Level Agreement (SLA) and the security provisions in the ITM contract once finalized.