Authorization Testing

1. Introduction

Authorization testing is the process of verifying that the application correctly enforces access controls, ensuring that users can only access the data and functionality that they are explicitly permitted to. This includes testing for vulnerabilities such as directory traversal, privilege escalation, and insecure direct object references (IDOR). A failure in authorization can lead to unauthorized access to sensitive data and functionality, and is often a critical vulnerability.

2. CVSS Score Summary

Test Case	Vulnerability	CVSS v3.1 Score	CVSS Vector	Severity
OTG-AUTHZ-001	Testing Directory Traversal/File Include	0.0	N/A	Informational
OTG-AUTHZ-002	Testing for Bypassing Authorization Schema	0.0	N/A	Informational
OTG-AUTHZ-003	Testing for Privilege Escalation	0.0	N/A	Informational
OTG-AUTHZ-004	Testing for Insecure Direct Object References	0.0	N/A	Informational

3. Summary of Findings

No authorization vulnerabilities were identified during this assessment. The tests for directory traversal, bypassing authorization schema, privilege escalation, and insecure direct object references did not reveal any weaknesses in the application's access control mechanisms.

The primary challenge in this testing category was the inability to log in with low-privilege credentials, which is a prerequisite for many of the authorization tests. As a result, the tests could not be performed as intended, and the assessment of the application's authorization controls is incomplete. It is recommended that a valid low-privilege user account be created to allow for a more thorough evaluation of the application's authorization scheme.

4. General Recommendation/Remediation

- **Implement Proper Access Control:** Ensure that all sensitive functionality and data is protected by a robust access control mechanism. This includes verifying that the user is authenticated and has the necessary permissions to access the requested resource.
- **Use the Principle of Least Privilege:** Users should only be granted the minimum level of access required to perform their job functions.
- **Avoid Insecure Direct Object References:** Instead of using direct object references in URLs, use indirect references that are mapped to the user's session. This will prevent attackers from being able to access other users' data by simply changing a parameter in the URL.
- **Implement a Low-Privilege User Account for Testing:** To allow for a more thorough assessment of the application's authorization controls, a low-privilege user account should be created for testing purposes.