Overpass — Application Security Assessment

Vulnerability Class: Authentication Bypass (Broken Authentication)
Risk Rating: Critical

> **Scope / Authorization:**
> This assessment was performed in a controlled, authorized lab environment and did not involve testing any real-world production systems.

Table of Contents

Executive Summary

An application security assessment was performed against the Overpass web application, focusing on its authentication and session handling mechanisms. The application relies on a custom authentication design that fails to enforce proper server-side validation of authentication state.

The assessment identified a critical authentication bypass vulnerability that allows an unauthenticated external attacker to gain unauthorized access to protected application resources. This flaw undermines the application's core trust model and enables account takeover and downstream system compromise.

Severity: Critical
Business Risk: Complete loss of authentication integrity, leading to unauthorized access, data exposure, and potential full application compromise.

---

1 Application Overview

The Overpass application is a self-hosted password management system designed to allow users to store, retrieve, and manage credentials through a web-based interface and supporting backend services.

The application implements a **custom-built authentication mechanism** rather than leveraging a mature authentication framework. Authentication state is managed using application-generated tokens intended to represent logged-in users.

Authentication Mechanism

- Custom token-based authentication
- Authentication state derived from client-supplied data
- Limited server-side validation of token integrity

Assets Protected

- User credentials and stored secrets

- Authenticated session state
- Backend system access tied to authenticated users
- Internal application logic and configuration files

The application assumes authenticated sessions are trustworthy and does not sufficiently re-validate authentication state server-side.

---

2 Threat Model (CRITICAL)

Attacker

- Unauthenticated external user
- No valid credentials
- No prior system access

Attacker Goal

- Bypass authentication controls
- Gain unauthorized access to protected resources
- Perform account takeover and escalate privileges

Broken Trust Assumptions

- Client-side authentication data is trusted
- Authentication tokens are not cryptographically verified server-side
- Session integrity is assumed rather than enforced

This creates a **broken trust boundary**, where untrusted client input directly influences authentication and authorization decisions.

---

3 Attack Hypotheses

During initial reconnaissance, a privileged application endpoint appeared exposed prior to authentication and was protected by custom login logic rather than a standardized authentication framework.

Based on these observations, the following hypotheses guided testing:

1. Custom authentication logic may allow token forgery or manipulation.
2. Client-controlled authentication artifacts may be trusted without integrity checks.
3. Sensitive authentication secrets may be stored insecurely, enabling secondary compromise.
4. Session state may not be strongly bound to server-side verification, allowing unauthorized access.

These hypotheses were formed **before exploitation** and focused testing on authentication, authorization, and session-management controls.

---

4 Vulnerability Analysis (No Payloads)

Vulnerability Description

The application suffers from **broken authentication design**, allowing authentication bypass due to improper trust in client-supplied authentication data.

Root Cause

- Authentication tokens are accepted without strong cryptographic verification
- Token origin and integrity are not enforced
- Authorization decisions rely on attacker-controlled data

Design Failures

- Lack of server-side verification
- Improper session management
- Broken trust boundary between client and server

Standard authentication controls expected in production systems were missing or incorrectly implemented.

---

5 Exploitation (Minimal & Clean)

Affected Component: authentication cookie handling and session validation logic.

Exploitation was achieved by manipulating the application's client-controlled authentication state.

The application relies on a cookie-based mechanism to determine whether a user is authenticated. By manually supplying a cookie value normally issued after login, the server **trusted the presence of the cookie instead of validating it server-side**, including its integrity, origin, or associated user role.

As a result, the application treated the request as authenticated and granted access to privileged functionality **without valid credentials**, confirming a complete authentication bypass.

---

6 Evidence (Sanitized)

- Screenshot demonstrating access to protected administrative resources without login
- Screenshot confirming elevated application privileges after bypass

(All sensitive information has been redacted.)

---

7 Impact (Business Risk)

This vulnerability enables:

- Unauthorized access to protected application resources
- Exposure of stored credentials and sensitive data
- Account takeover and privilege escalation
- Complete compromise of the application's authentication trust model

In a real-world deployment, this could lead to data breaches, loss of user trust, regulatory violations, and full application compromise.

---

8 Remediation (Concrete)

Design-Level Fixes

- Replace custom authentication with a proven authentication framework or identity provider
- Eliminate trust in client-controlled authentication state

Technical Controls

- Enforce server-side session validation on every request
- Implement cryptographically signed tokens (HMAC or asymmetric signing)
- Validate token integrity, expiration, and issuer
- Store secrets securely using environment variables or a secrets vault
- Apply least-privilege access controls

Standards Reference

- OWASP Top 10 – Identification & Authentication Failures
- OWASP ASVS v4 —
  - V2: Authentication
  - V3: Session Management

Remediation requires **architectural redesign**, not superficial patching.

---

9 Lessons Learned

Missed Early Signal

Custom authentication logic should immediately trigger focused security testing.

What to Test First Next Time

- Authentication flows
- Token structure and integrity
- Client–server trust boundaries

Real-World Mapping

This maps to real-world breaches involving token forgery, broken session handling, and trusting client-side authorization data.

Overall Assessment: The authentication mechanism requires architectural redesign to ensure trust boundaries are enforced and session integrity is guaranteed.