# PENETRATION TEST REPORT

**Client:** Global Corp (Simulation)
**Date:** November 21, 2025
**Auditor:** Shonel Geri, Security Engineer

---

## 1. Executive Summary

This document presents the findings of a security assessment conducted on Global Corp's web infrastructure and external network perimeter. The objective was to identify security vulnerabilities that could compromise the confidentiality, integrity, and availability of business data.

**Assessment Result: CRITICAL**
The assessment identified critical vulnerabilities allowing for **Remote Code Execution (RCE)** and unauthorized administrative access via **Authentication Bypass**. Immediate remediation is recommended.

---

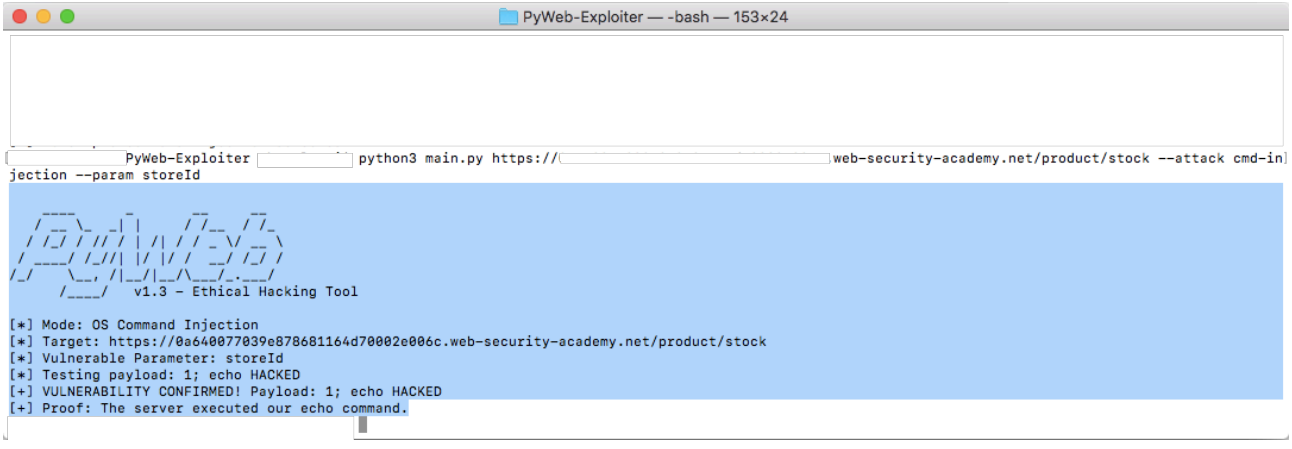
## 2. Technical Findings

### Finding #1: OS Command Injection (RCE)

- **Severity:** **CRITICAL (CVSS 9.8)**
- **Target:** Product Stock Check API
- **Tool Used:** PyWeb-Exploiter (Custom Module)

**Description:**
The application fails to properly sanitize user input in the storeId parameter. By injecting shell metacharacters (|), it was possible to execute arbitrary system commands on the host server.

**Proof of Concept:**
The custom tool injected the payload 1|echo HACKED. The server responded with the string "HACKED", confirming code execution.



**Remediation:**
Implement strict input validation and avoid passing user input directly to system shells. Use parameterized APIs instead of os.system() calls.

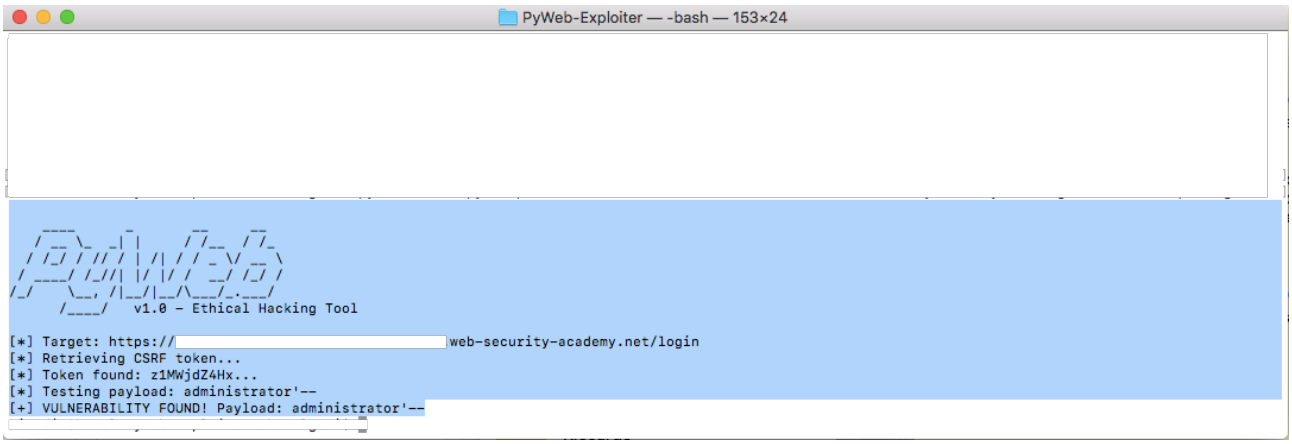
---

## Finding #2: SQL Injection & Authentication Bypass

- **Severity:** **HIGH (CVSS 8.1)**
- **Target:** Administrative Login Portal
- **Tool Used:** PyWeb-Exploiter (Custom Module)

**Description:**
The login mechanism is vulnerable to SQL Injection. It was possible to manipulate the backend query to ignore the password check, allowing unauthorized access as the administrator user. The attack also successfully bypassed the anti-CSRF token protection.

**Proof of Concept:**
Payload used: administrator'-- injected into the username field.




**Remediation:**
Use Prepared Statements (Parameterized Queries) for all database interactions.

---

## Finding #3: Exposed SSH Service (Information Disclosure)

- **Severity:** **LOW / INFORMATIONAL**
- **Target:** External Perimeter IP
- **Tool Used:** Net-Audit (Network Scanner)

**Description:**
A port scan revealed that Port 22 (SSH) is open and exposing the service version banner (OpenSSH 6.6.1p1). This information can be used by attackers to identify specific exploits for this version.

**Proof of Concept:**

```
[                                  ]$ python3 scanner.py scanme.nmap.org --ports 1000 --threads 100
[*] Starting fast scan on target: 45.33.32.156
[*] Scanning first 1000 ports with 100 threads...
------------------------------------------------
[+] Port 22 is OPEN --> SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.13
[+] Port 80 is OPEN
------------------------------------------------
[*] Scan completed in 0:00:03.455044
```

**Remediation:**
Restrict access to Port 22 to trusted IP addresses only (VPN/Firewall) and disable the service banner display in the SSH configuration.

---

# 3. Methodology

The assessment was performed using a "Grey Box" approach. Custom automated tooling developed in Python (PyWeb-Exploiter, Net-Audit) was utilized to minimize noise and target specific vulnerability classes (OWASP Top 10).

---