**Guardian Store Vulnerability Report**

**Table of Contents**

[**Abstract** 3](#_Toc147591487)

[**Overview** 4](#_Toc147591488)

[**Introduction** 4](#_Toc147591489)

[**Vulnerabilities Identified** 4](#_Toc147591490)

[**Chosen Vulnerability:** Broken Authentication 5](#_Toc147591491)

[**Introduction** 5](#_Toc147591492)

[**Research** 5](#_Toc147591493)

[**Exploitation** 5](#_Toc147591494)

[**Patch** 5](#_Toc147591495)

[**Security Opportunities** 6](#_Toc147591496)

[**Defence-In-Depth** 6](#_Toc147591497)

[Definition 6](#_Toc147591498)

[Opportunities 6](#_Toc147591499)

[**Secure Software Development Mitigations** 6](#_Toc147591500)

[Definition 6](#_Toc147591501)

[Opportunities 6](#_Toc147591502)

[**Conclusion** 7](#_Toc147591503)

[**Bibliography & References** 8](#_Toc147591504)

# **Abstract**

# **Overview**

## **Introduction**

Guardian has contacted us to help penetration test GuardianStore, identify vulnerabilities and attempt to resolve them.

## **Vulnerabilities Identified**

The following vulnerabilities have been identified:

* Broken authentication
* Broken authorization
* Business logic flaws
* Cross-site request forgery
* Cross-site scripting
* Cryptography issues
* Insecure file upload
* NoSQL injection
* Sensitive data exposure
* SQL injection

# **Chosen Vulnerability:** Broken Authorization

## **Introduction**

The chosen vulnerability for this penetration report is Broken Authorization, also known as broken access control. This is a major/critical flaw which could compromise the account security and confidentiality of clients and admins alike. Being a surprisingly common vulnerability, finding a fix for this issue will mean similar cases and issues can also be resolved.

## 

## **Research**

Broken

## **Exploitation**

## **Patch**

To mitigate against this brute-force authentication vulnerability we should implement Multi-Factor Authentication as well as an account lockout and unlock mechanism.   
Multi-Factor Authentication would involve implanting an email or sms authentication request, requiring the user to validate their connection using another service. This mitigates the attack as third-parties would be required to have access to this authentication source in order to get access to the account.  
A Lockout and Unlock mechanism would prevent against brute force attacks using password and account keys. This would be accomplished by putting a login attempt limit which if excided would lockout the user preventing logins until they respond to an automated email informing the user of the security threat and asking if they would like to go through the process of unlocking their account. This would slow down or protect against these types of attacks.

# **Security Opportunities**

System & Application

Network

Physical

## **Defence-In-Depth**

### Definition

Defence in depth is a multi-layered security control methodology devised by the National Security Agency (NSA), and is a globally prevalent standard, for cyber security and data protection. It’s aim is to supply redundancy, procedural, technical, personal, and physical security.

Defence-In-Depth is often segmented into physical, technical, and administrative controls.  
**Physical:** physical controls, defending a system i.e., guards, locks, fences, security, etc.  
**Technical:** technical controls, securing a system i.e., Firewalls, Authentication, etc.  
**Administrative:** administrative controls, protecting a system i.e., policies, procedures, etc.

The layers that make up Defence-In-depth are System & Application, Network, and Physical. This includes but isn’t limited to the following methods:

|  |  |  |
| --- | --- | --- |
| **System and application:**  Antivirus software Multi-factor authentication  Encryption Hashing passwords Vulnerability scanners Timed access control Internet Security Awareness | **Network:**  Firewalls (hardware or software) Demilitarized zones (DMZ) Virtual private network (VPN) | **Physical:**  Biometrics Data-centric security Physical security |

### Opportunities

**Technical:** System, Application and NetworkImplementing Multi-factor Authentication would be a great opportunity to fix our broken authentication & authorisation.Encryption and password hashing should also be look into to secure our backend data and mitigate man-in-the-middle attack scouting.

**Administrative:** System, Application and Physical

An Administrative Defence-In-Depth opportunity would be teaching our administrators Internet Security Awareness. By learning common tactics, threats and red flags for social engineering or other attack methods, we can mitigate their effectiveness at tricking our staff into compromising system security. Intern further strengthening our system.

**Physical:** Network and Physical

To Physically protect our system we would implement firewalls between our servers on the Network layer and on the Physical layer we should implement site security protocols with guards and surveillance to ensure no physical harm or improper access can happen on our system

## **Secure Software Development Mitigations**

### Definition

Secure Software Development Principles which make recommendations on how to strengthen our cyber security and mitigate against attacks or intrusions..

**Principles:**

* Least Privilege

Users have the minimum privilege required for proper function.

* Fail-Safe Defaults

Defaults should be created that maintain security, in case of failure.

* Economy of Mechanism

Simple security design and controls, the simpler the more stable.

* Complete Mediation

Access must be authorised before granting access.

* Open Design

Security can’t depend on secrecy of design or implementation.

* Separation of Privilege

Security checks should require multiple conditions to be met. More secure than one condition.

* Least Common Mechanism

Minimise commonalities and dependencies between users.

* Psychological Acceptability

Security controls should be easy-to-use, understand, and control, rather than bypass.

### Opportunities

# **Conclusion**

By implementing \_\_ we can fix the broken authentication. Though making use of opportunities presented to us by Defence-In-Depth methods and Secure Software Development Mitigation Techniques we can further defend our system from future attacks.

# **Bibliography & References**

[1] OWASP. (2021). Secure Product Design - OWASP Cheat Sheet Series. [online] Available at: [https://cheatsheetseries.owasp.org/cheatsheets/Secure\_Product\_Design\_Cheat\_Sheet.  
html#2-the-principle-of-defense-in-depth](https://cheatsheetseries.owasp.org/cheatsheets/Secure_Product_Design_Cheat_Sheet.html#2-the-principle-of-defense-in-depth).

[2] NSA .(2012). Defence In Depth. [online] Available at: [https://web.archive.org/web/  
20121002051613/https://www.nsa.gov/ia/\_files/support/defenseindepth.pdf.](https://web.archive.org/web/20121002051613/https://www.nsa.gov/ia/_files/support/defenseindepth.pdf.)

[3] A., Barrett. (2015). Some principles of secure design – Designing Secure System module Autumn 2015. [online] Available at: <https://docplayer.net/17241198-Some-principles-of-secure-design-designing-secure-systems-module-autumn-2015.html>

[4] OWASP. (2021). A01 Broken Access Control - OWASP Top 10:2021. [online] owasp.org. Available at: <https://owasp.org/Top10/A01_2021-Broken_Access_Control/>.