Udajuicer:
Threat Report

Mohammad Eissa Alsow

13, Nov

Purpose of this Report:

This is a threat model report for **Udajuicer**. The report will describe the threats facing Udajuicer. The model will cover the following:

- Threat Assessment
  - Scoping out Asset Inventory
  - Architecture Audit
  - Threat Model Diagram
  - Threats to the Organization
  - Identifying Threat Actors
- Vulnerability Analysis
- Risk Analysis
- Mitigation Plan

Section 1

Threat Assessment

1.1: Asset Inventory

Components and Functions

- ***Web server:*** *Receive clients requests.*
- ***Web application:*** *Retrieve data if neede from database server.*
- ***Database server:*** *Stores the data and orgnaize it.*

1.1: Asset Inventory

Explanation of How A Request Goes from Client to Server

First of all the client will open our web page, at this time the client is asking for the web page components. so he will send a request through internet to our web server, then web server will translate the request to binary and send it to the application server which will retrieve needed data from the database server, then data will go back to client in the same order.

1.2 Architecture Audit

Flaws

- ***There wasn’t a Firewall, which mean all traffic will pass to the server.***
- ***The network wasn’t segmented, all servers were in one LAN which is easier in lateral movement.***
- ***There wasn’t any component to deal with heave traffic or DDOS attacks such as a Load balancer or a CDN.***

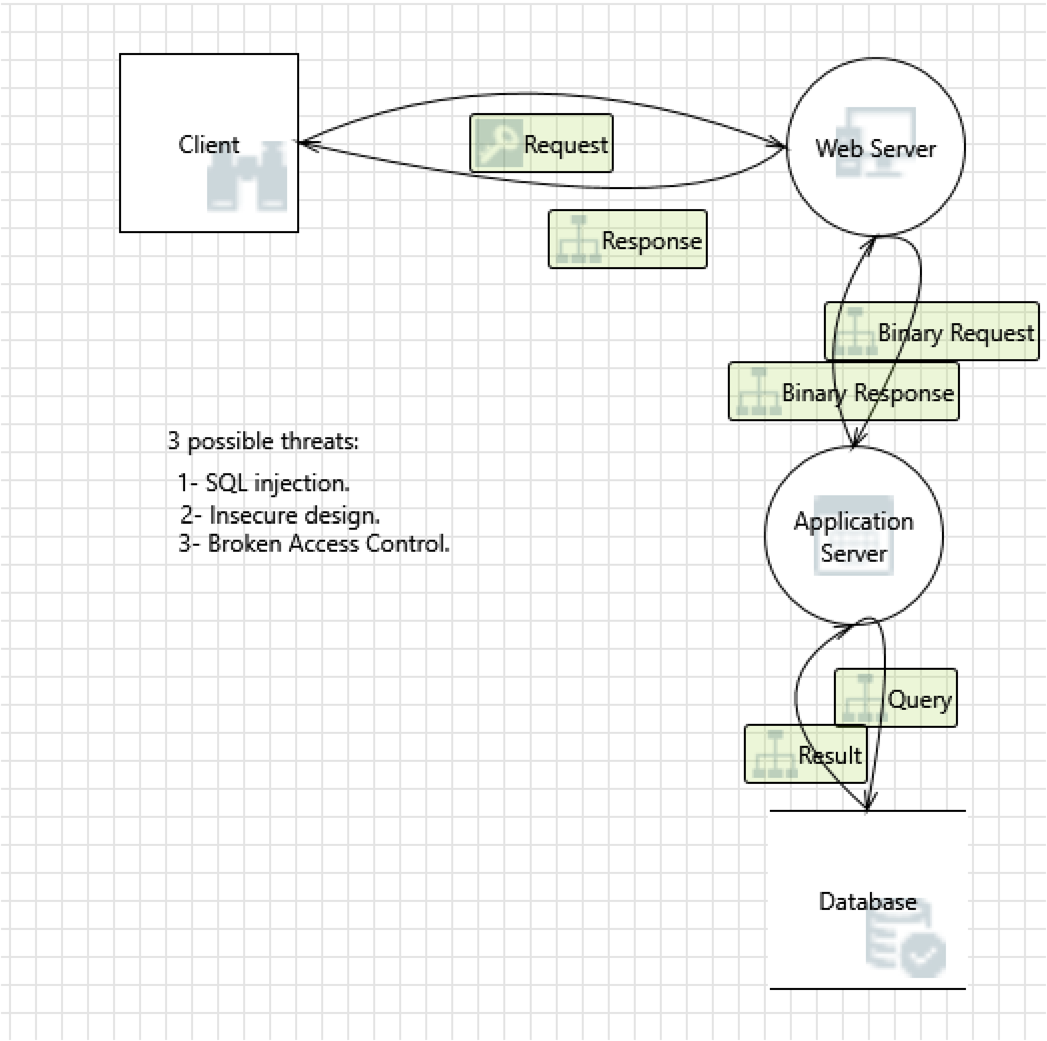
1.3 Threat Model Diagram

Using OWASP Threat Dragon, build a diagram showing the flow of data in the Juice Shop application and identify 3 possible threats to the Juice Shop. Make sure to include the following components:

- **Client**
- **Web Server**
- **Application Server**
- **Database**

1.3 Threat Model Diagram

Insert threat Model Diagram Here:



3 possible threats:

1- SQL injection.
2- Insecure design.
3- Broken Access Control.

1.4 Threat Analysis

What Type of Attack Caused the Crash?

It seems to be a DDOS attack.

What in the Logs Proves Your Theory?

A lots of requests at the same time without receiveing response.

1.5 Threat Actor Analysis

Who is the Most Likely Threat Actor?

It seems to be a Script Kiddies.

What Proves Your Theory?

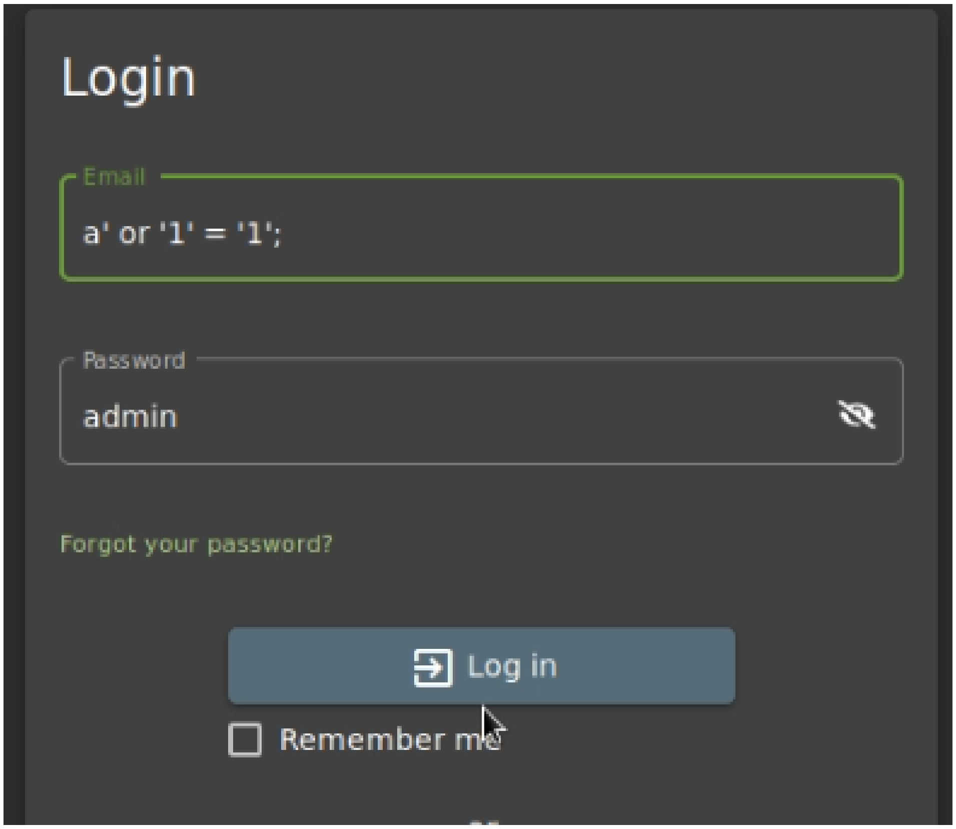
Because there wasn’t anything stolen, and the attack seems to be for fun without any intention. Additionally the web server was the only affected part there wasn’t a modify in data, data breach or anything else.

Section 2

Vulnerability Analysis

2.1 SQL Injection

Insert Screenshot of Your Commands Here:

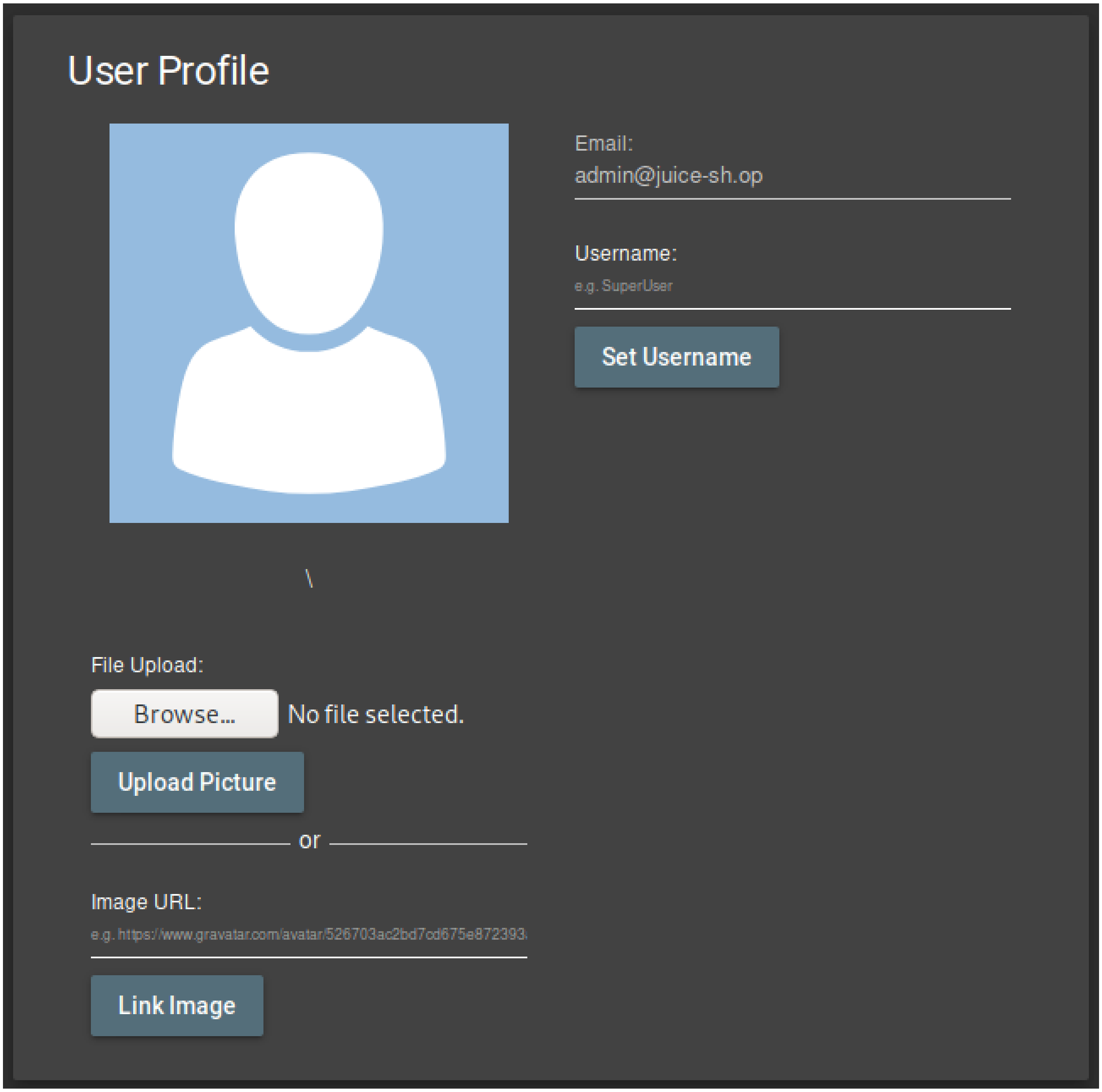


Email: a' or '1' = '1';

Password: admin

2.1 SQL Injection

Insert Screenshot of Account Settings Showing You as Admin Here:



User Profile

Email: admin@juice-sh.op

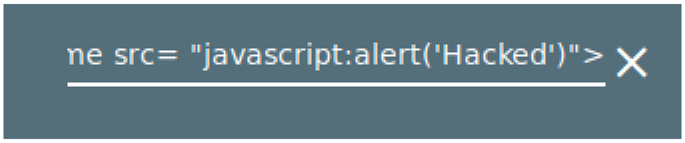
Username: e.g. SuperUser

File Upload: No file selected.

Image URL: e.g. https://www.gravatar.com/avatar/526703ac2bd7cd675e872393

2.2 XSS

Insert Screenshot of Your Commands Here:

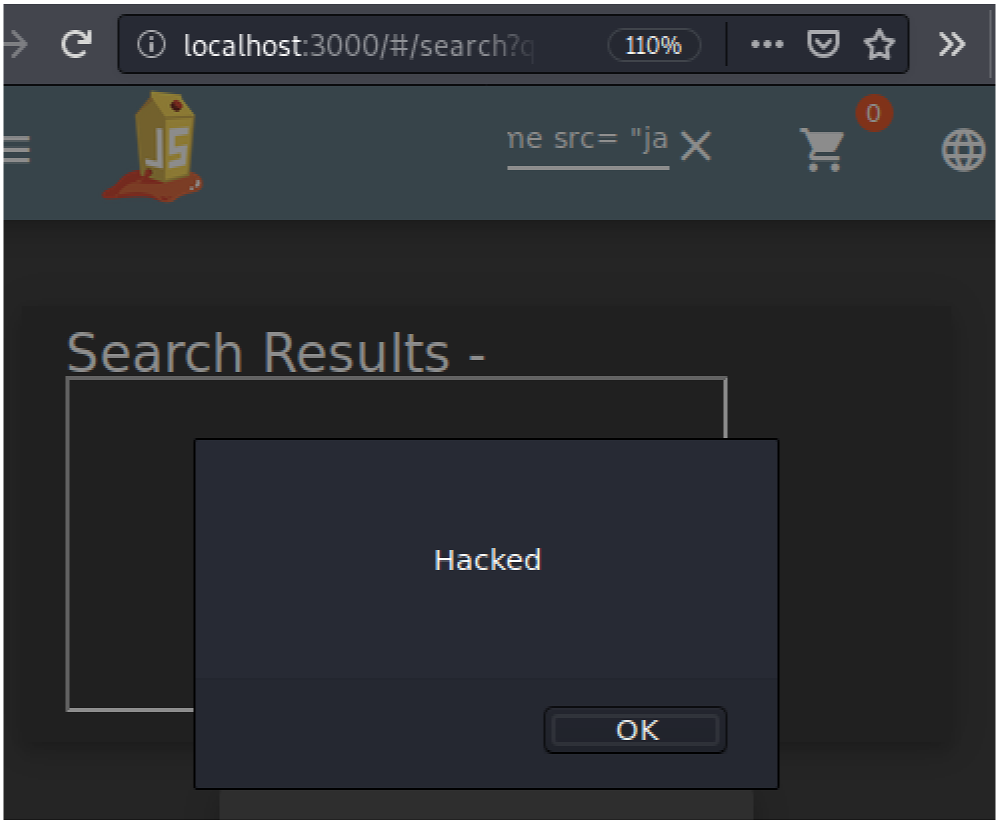


```
ne src= "javascript:alert('Hacked')">
```

```
<iframe src="javascript:alert('Hacked')">
```

2.2 XSS

Insert Screenshot of `alert()` popup saying "Hacked!" Here:

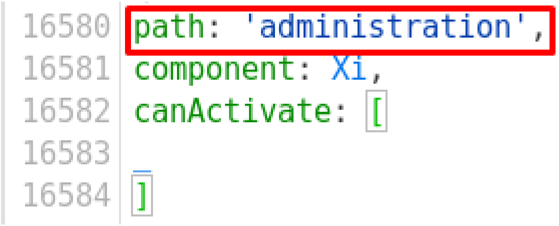


Hacked

Optional Task:

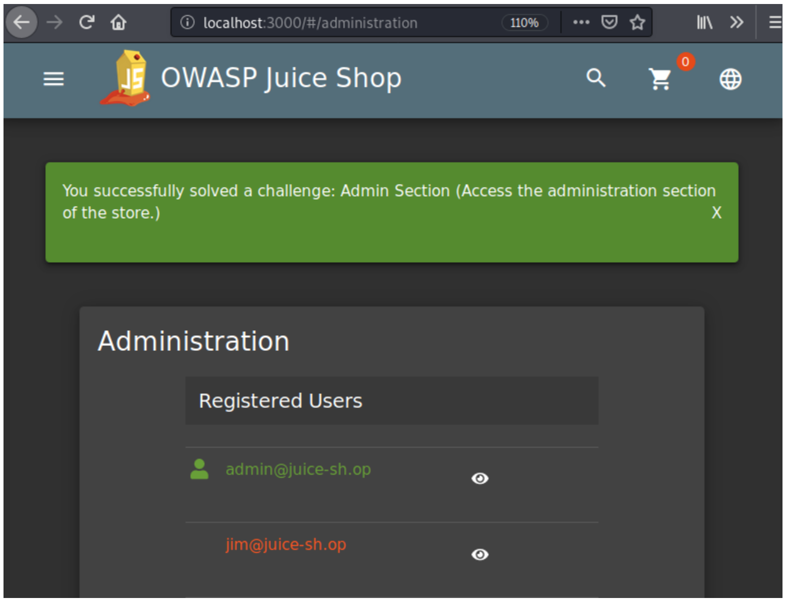
Extra Vulnerabilities

- *Broken Access Control*



```
path: 'administration',
component: Xi,
canActivate: [

]
```



You successfully solved a challenge: Admin Section (Access the administration section of the store.)

X

Administration

Registered Users

admin@juice-sh.op

jim@juice-sh.op

Section 3

Risk Analysis

3.1 Scoring Risks

Risk	**Score** ***(1 is most dangerous, 4 is least dangerous)***
Broken Access Control	3
SQL Injection	2
XSS Vulnerability	4
DDOS attack	1

3.2 Risk Rationale

Why Did You Choose That Ranking?

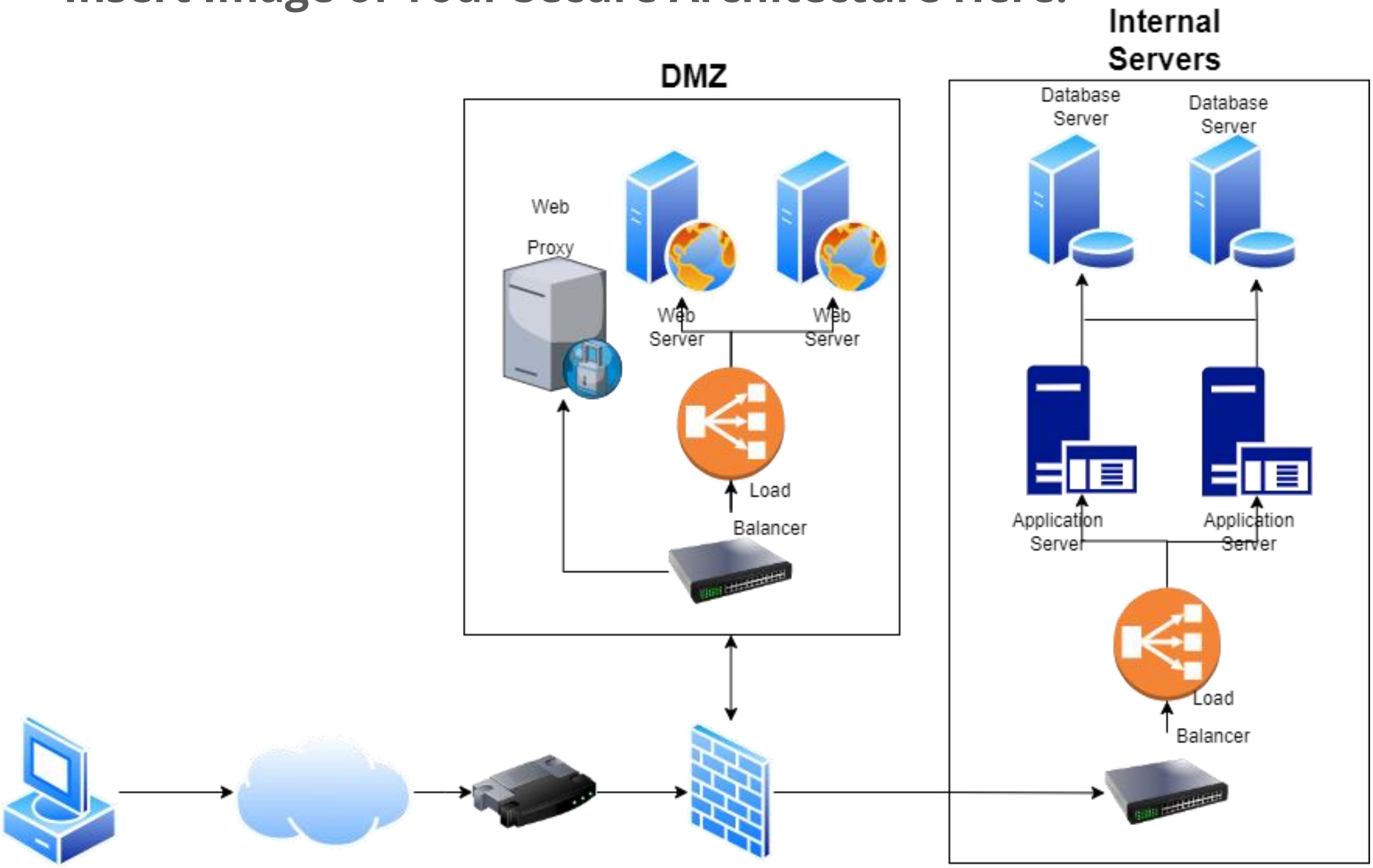
- DDOS attack cause the server to crash.
- SQL Injection allowed me to log in as admin and it could cause a data breach from the database server.
- Using Broken access control I managed to reach admin's section.
- XSS Vulnerability can turn the website into a malicious page, which will lead to Reputational Damage.

Section 4

Mitigation Plan

4.1 Secure Architecture

Insert Image of Your Secure Architecture Here:



4.2 Mystery Attack Mitigation

What is Your Mitigation Plan?

DDOS attack can be mitigated by implementing a secure architecture by using a Content Delivery Network (CDN), Load Balancers, and Firewalls. CDN can help by redistributing this traffic and ensuring it doesn't reach your origin servers. Load Balancer automatically scales to absorb the additional traffic when these types of attacks are detected. Firewall can help if a single IP sent a lot of packets it blocks the IP.

4.3 SQL Injection Mitigation

What is Your Mitigation Plan?

SQL INJECTIONS can be mitigated by Input Sanitization, Input Validation, Prepared Statements with Parameterized Queries, and Escaping. This strategies will help us to make sure input will be correct and will not include suspicious character.

4.4 XSS Mitigation

What is Your Mitigation Plan?

In XSS mitigation plan we can use Sanitizing User Input, Validating User Input, and Escaping. Sanitization will help in clearing input. Validating will make sure input is same as we want. Escaping encode so they don't get interpreted.