Udajuicer:
Threat Report

Mariam AlHugail

15/10/2022

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

A web server is a machine that houses the web server software and the data that make up a website, such as the HTML code, the CSS stylesheets, and the JavaScript code. A web server establishes a connection to the Internet and facilitates the physical data exchange with other web-connected devices.

- ### Application Server

An application server is a software framework that offers all the resources necessary to design and execute both web-based and enterprise-based applications. It sits between the principal web server tier and the back end tier of a database and serves dynamic content. It works best for distributing dynamic material and moving apps between devices.

- ### Database

Databases are collections of structured data that are often kept electronically in computer systems and are arranged in a logical fashion.

1.1: Asset Inventory

Explanation of How A Request Goes from Client to Server

- *The client's device is connected to the Internet, and the browser asks for a URL, in this example, the URL for Udajuicer.*
- *Through an ISP, this request is made to the general internet. Using a DNS lookup, the URL is converted to the corresponding IP address. The web server hosting this website has the following network address: Udajuicer.*
- *The requested file is typically fetched by the web server and returned. The web server knows to forward the request to the application server, which will handle the script code, if the requested file in our case is a script file.*
- *The application server locates the file containing the source code and runs the script program after receiving an HTTP Request from the web server.*
- *In order to request data that is stored and arranged by the database server, the software makes reference to a connection to a database and sends a query.*
- *In order to request data that is stored and arranged by the database server, the software makes reference to a connection to a database and sends a query.*
- *The files requested by the query are downloaded by the database server.*
- *The database server returns to the application server the information that was requested.*
- *If necessary, the application server merges files before returning the result as an HTTP Response to the web server.*
- *The web server replies with HTML text to the connecting client.*
- *The connection is cut off after the data is returned to the client's device over the Internet.*
- *The generated HTML code is read by the device browser and displayed on the screen.*

1.2 Architecture Audit

Flaws

The following issues have been found:

- *There is no firewall or IDS in the architecture. A crucial security foundation is provided by firewalls. An intrusion detection system (IDS) is a device that scans network traffic for any suspicious behavior and sends out alarms when it does.*
- *Improved error handling would stop an attacker from getting data from mistakes. The website can display a generic error web page rather than displaying the error.*
- *Input validation is not present. Before being transmitted, data must be inputted and verified.*
- *Absence of stored parameterized procedures (SQL group)*
- *All users have unrestricted access to all server files, including administration and ftp. File server access exploitation can be avoided by limiting user groups' or individual users' access to files and folders.*
- *Adding load balancing and enough bandwidth.*
- *Fault tolerance via backups of the database.*

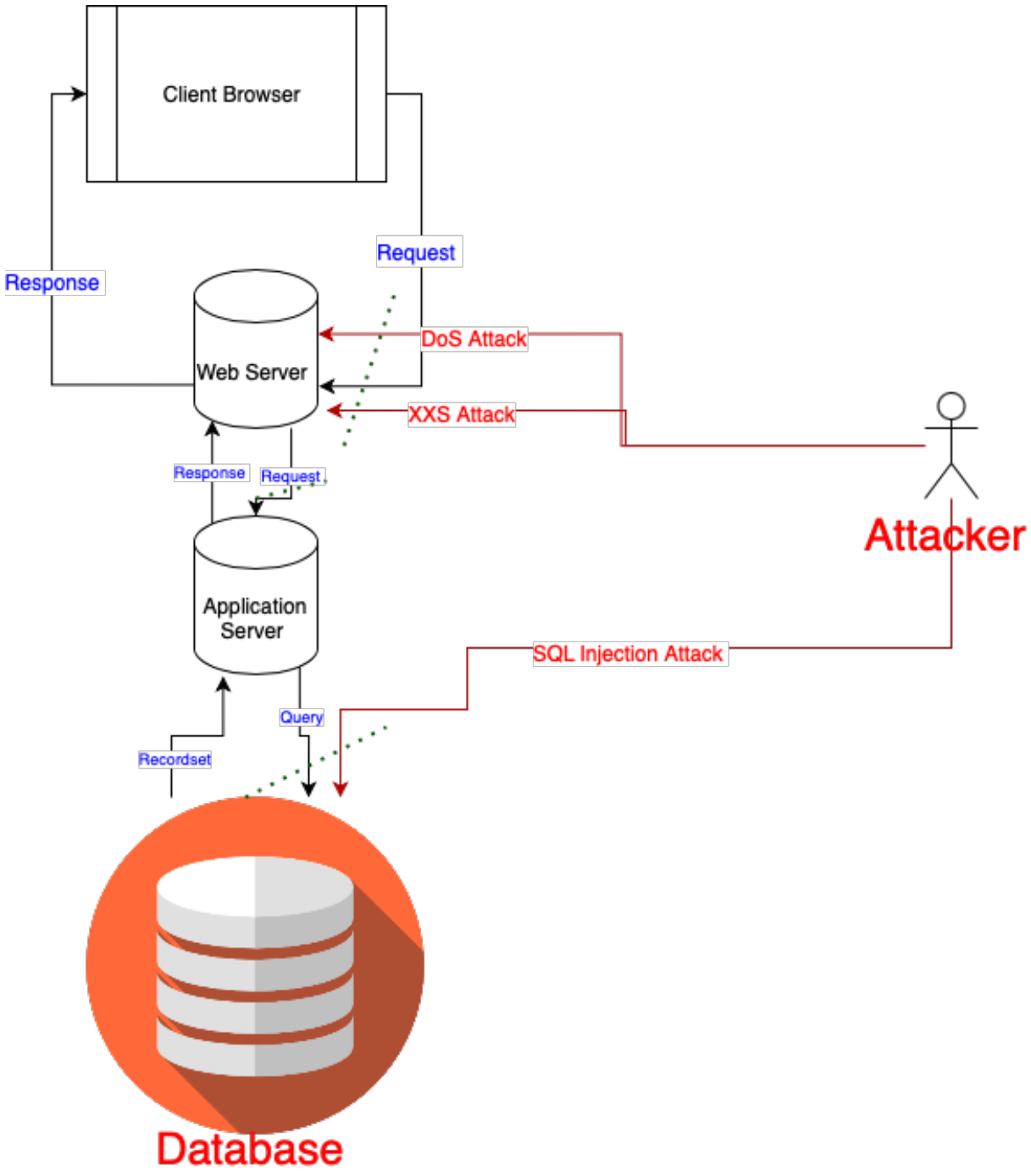
1.3 Threat Model Diagram

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 hreat Model Diagram Here:



1.4 Threat Analysis

What Type of Attack Caused the Crash?

The Udajuicer store was likely brought down by an HTTP flood / DDoS attack. HTTP flood is a sort of Distributed Denial of Service (DDoS) attack in which the attacker uses seemingly valid HTTP GET or POST requests to assault a web server or application. HTTP flood assaults are volumetric attacks that frequently employ a botnet, or a network of Internet-connected machines.

What in the Logs Proves Your Theory?

The log file indicates several requests from different IP addresses at the same time, which leads me to believe it was a DDoS attempt. It also includes "GET" entries. HTTP clients, such as a web browser, communicate with applications or servers by issuing HTTP requests, which can be one of two types: GET or POST. The attacker's goal with HTTP Flood attacks is to flood the server or application with as many requests as possible that are each as processing heavy as possible.

1.5 Threat Actor Analysis

Who is the Most Likely Threat Actor?

A "Script Kiddie" is the most likely threat actor in my opinion. I believe the attacks were carried out by an unskilled attacker who used a Distributed Denial of Service attack, despite the fact that the application architecture would provide for easier access to sensitive data.

What Proves Your Theory?

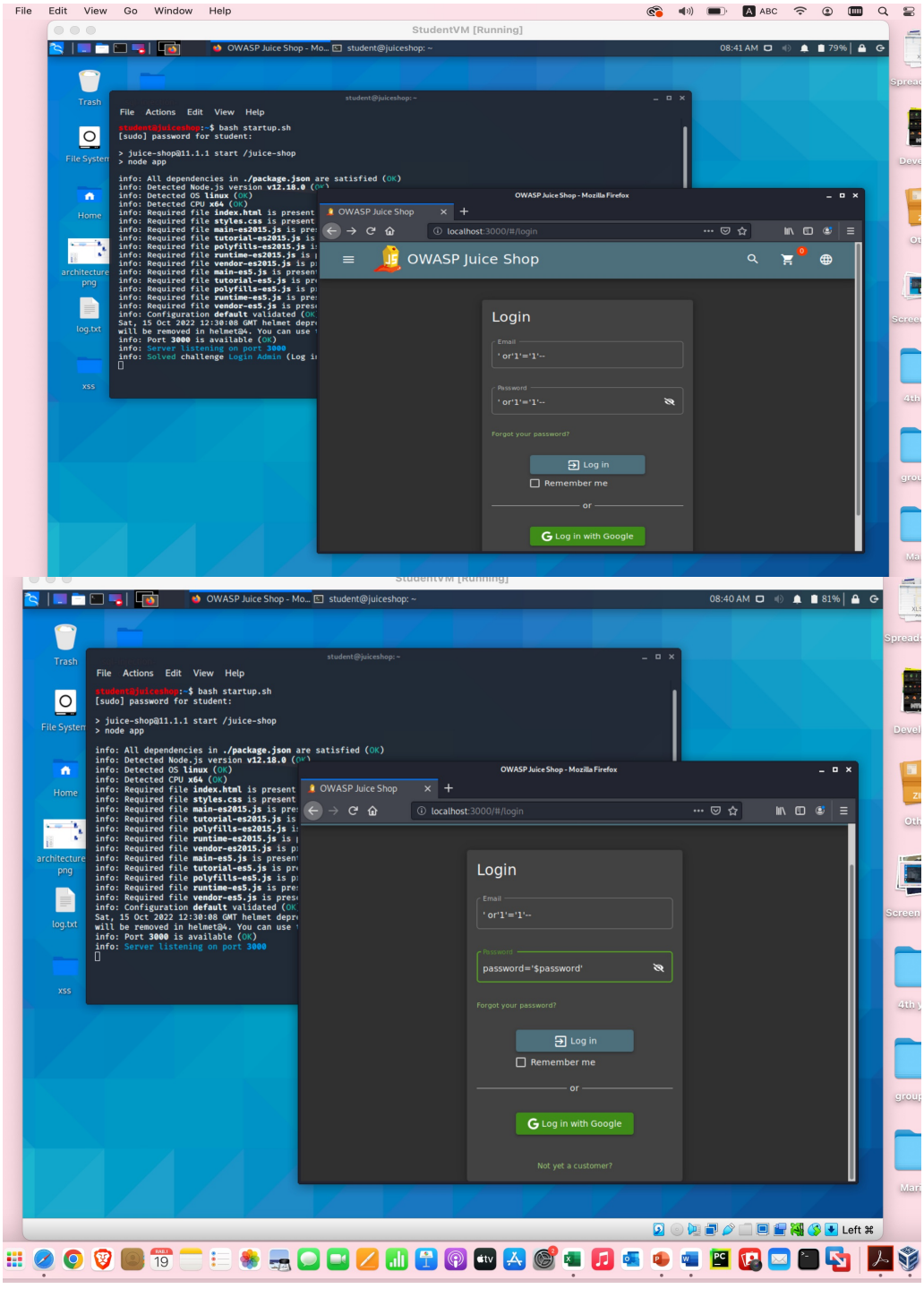
An expert hacker will most likely try another method of gaining access that leaves less trace and is faster, such as SQL injections. After the attack, an experienced hacker may destroy any proof. In my perspective, we may dismiss organized crime, APT groups, and state-sponsored attacks because the shop does not have enough valuable information for these types of organizations. Insiders would most likely have access to credentials but would have no idea how to enter the system or employ hacking tools.

Section 2

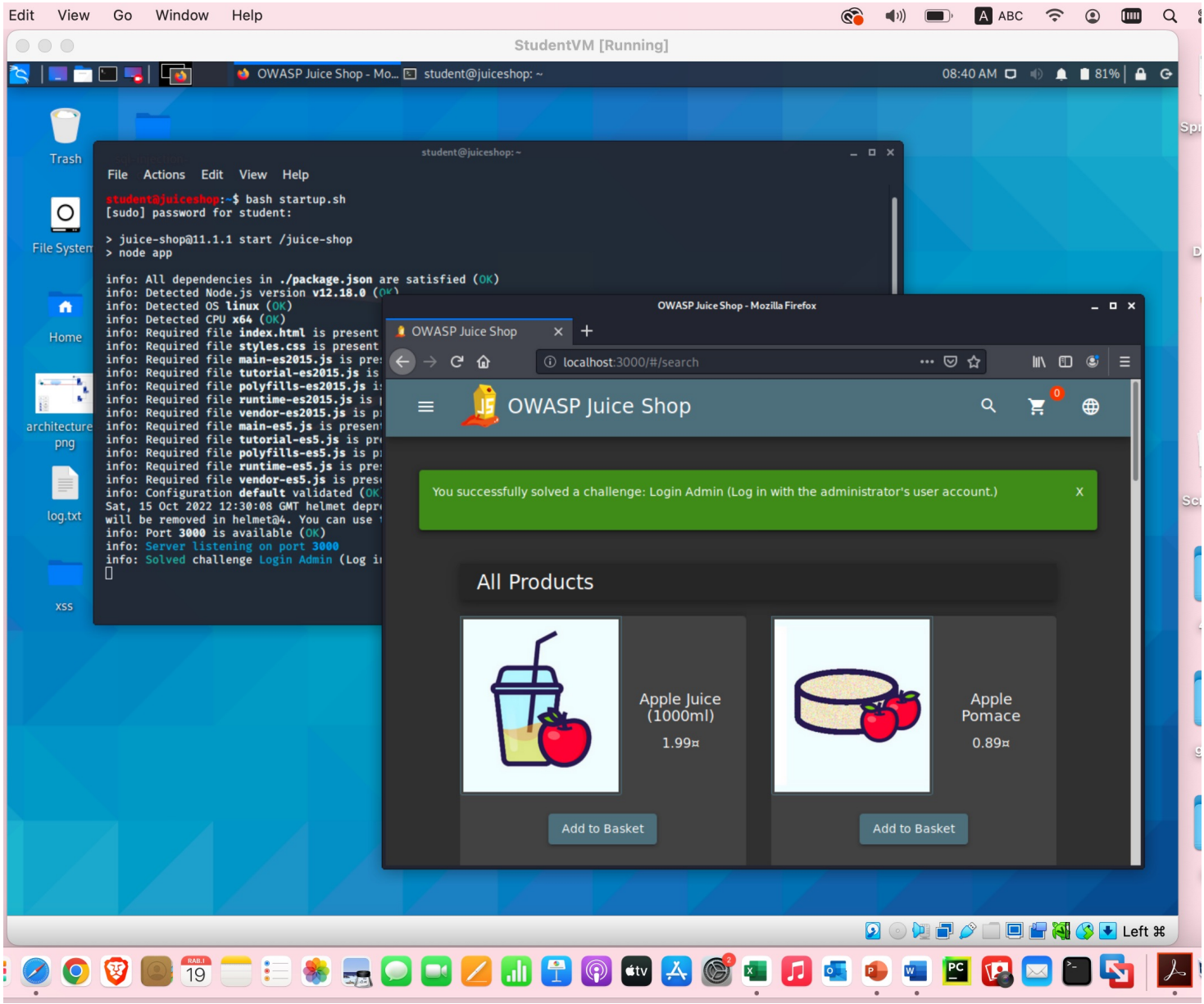
Vulnerability Analysis

2.1 SQL Injection

Insert Screenshot of Your Commands Here:

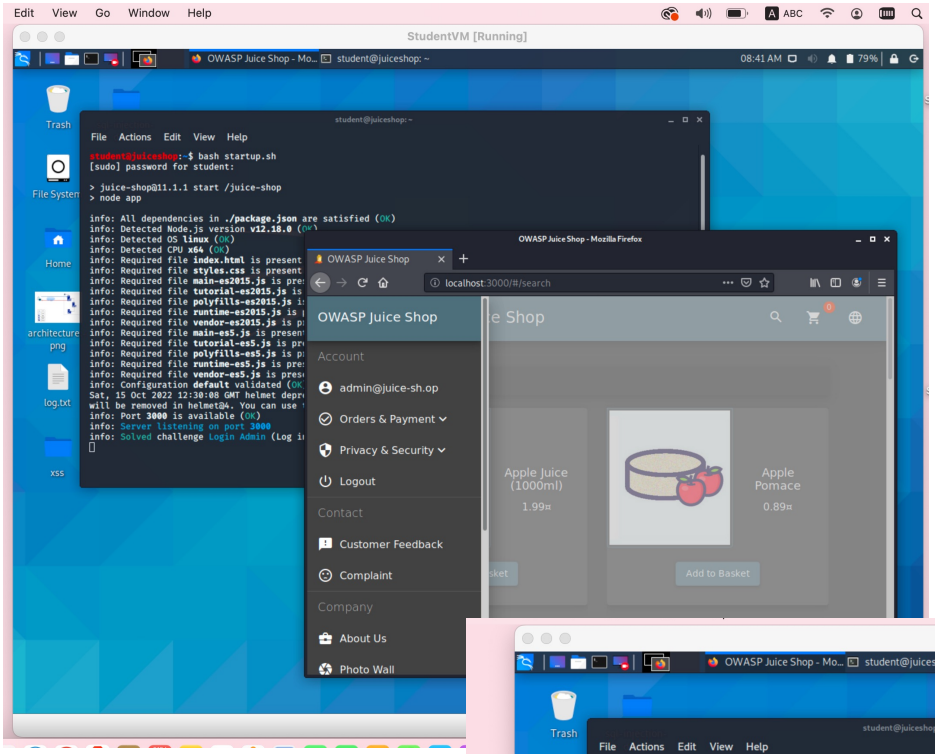


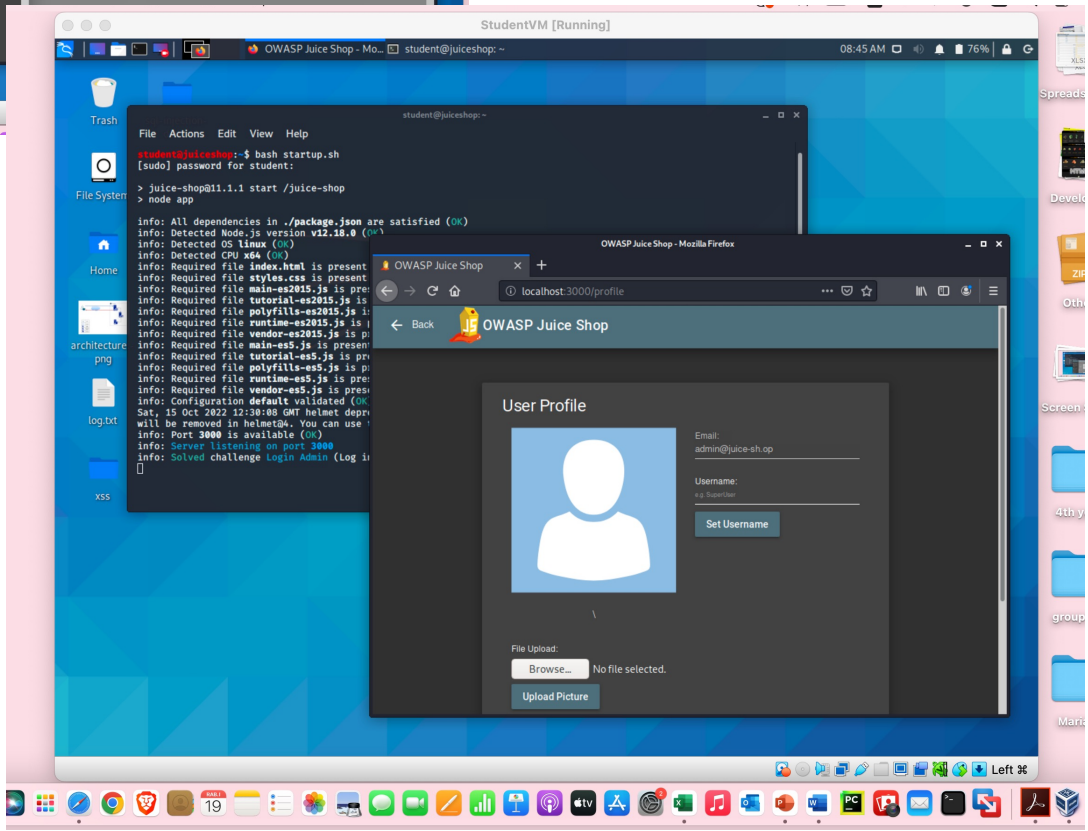
The screenshots below demonstrate that I was successful in logging into the admin account.



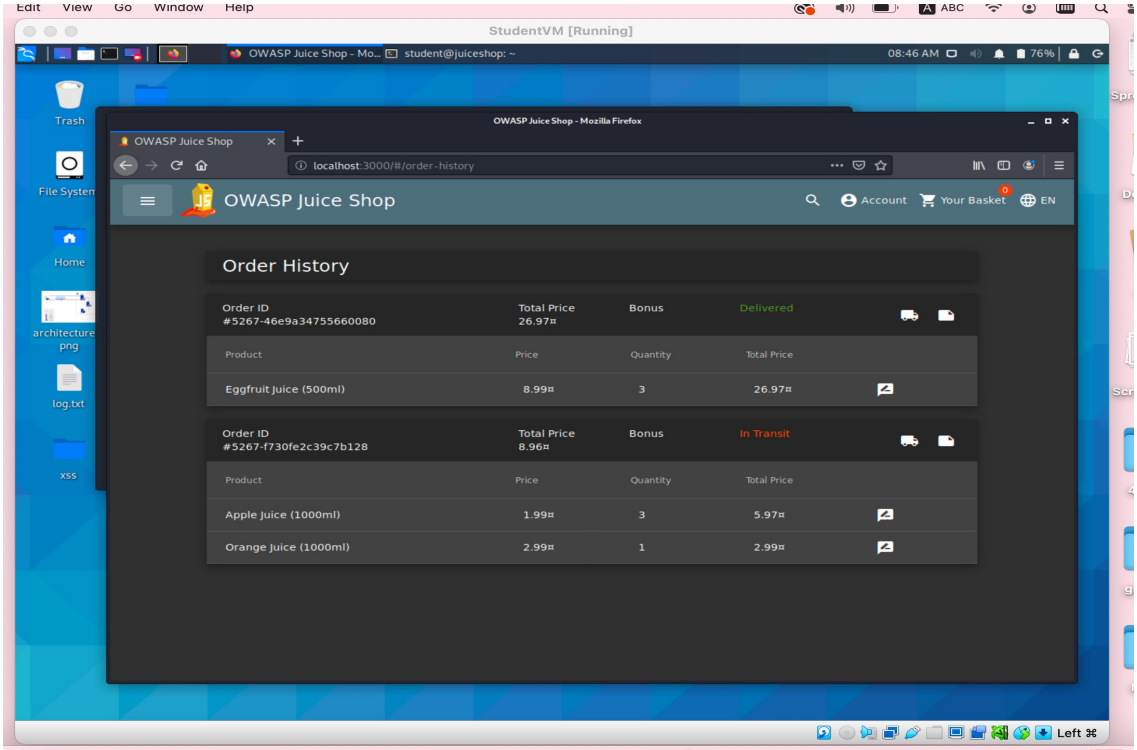
2.1 SQL Injection

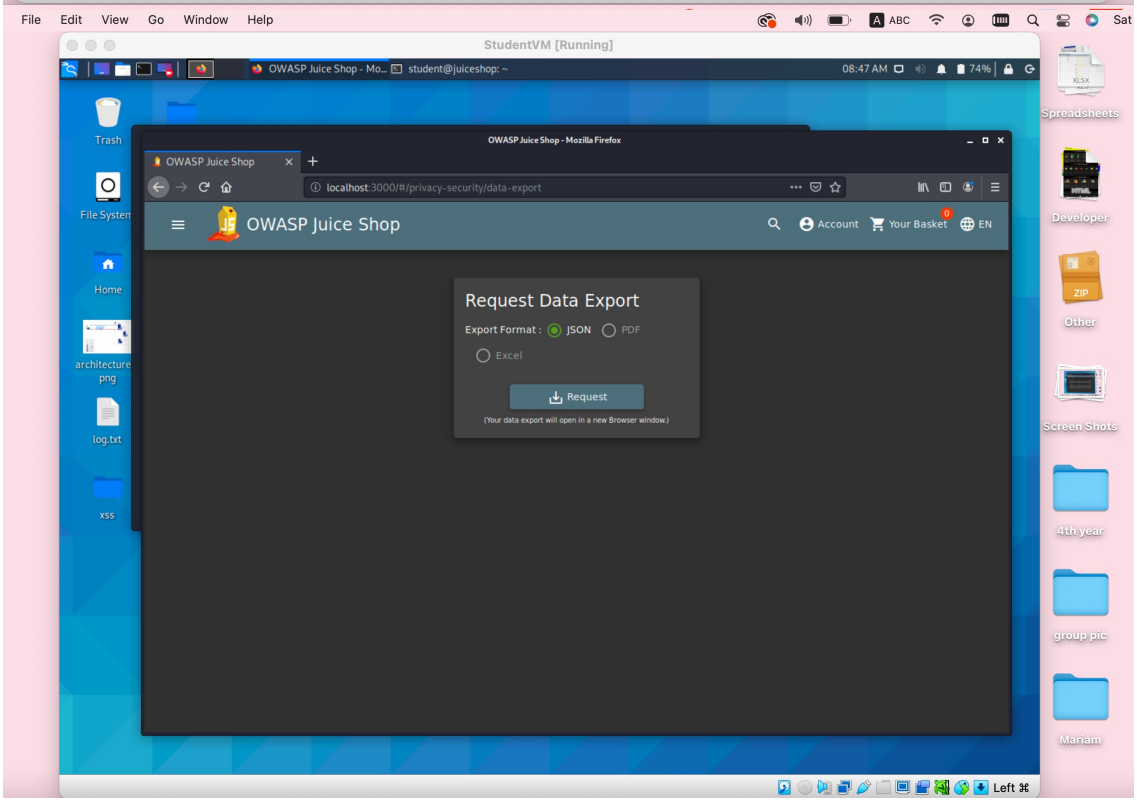
Insert Screenshot of Account Settings Showing You as Admin Here:





I now have full access to the website admin profiles, as seen in the screenshots below.

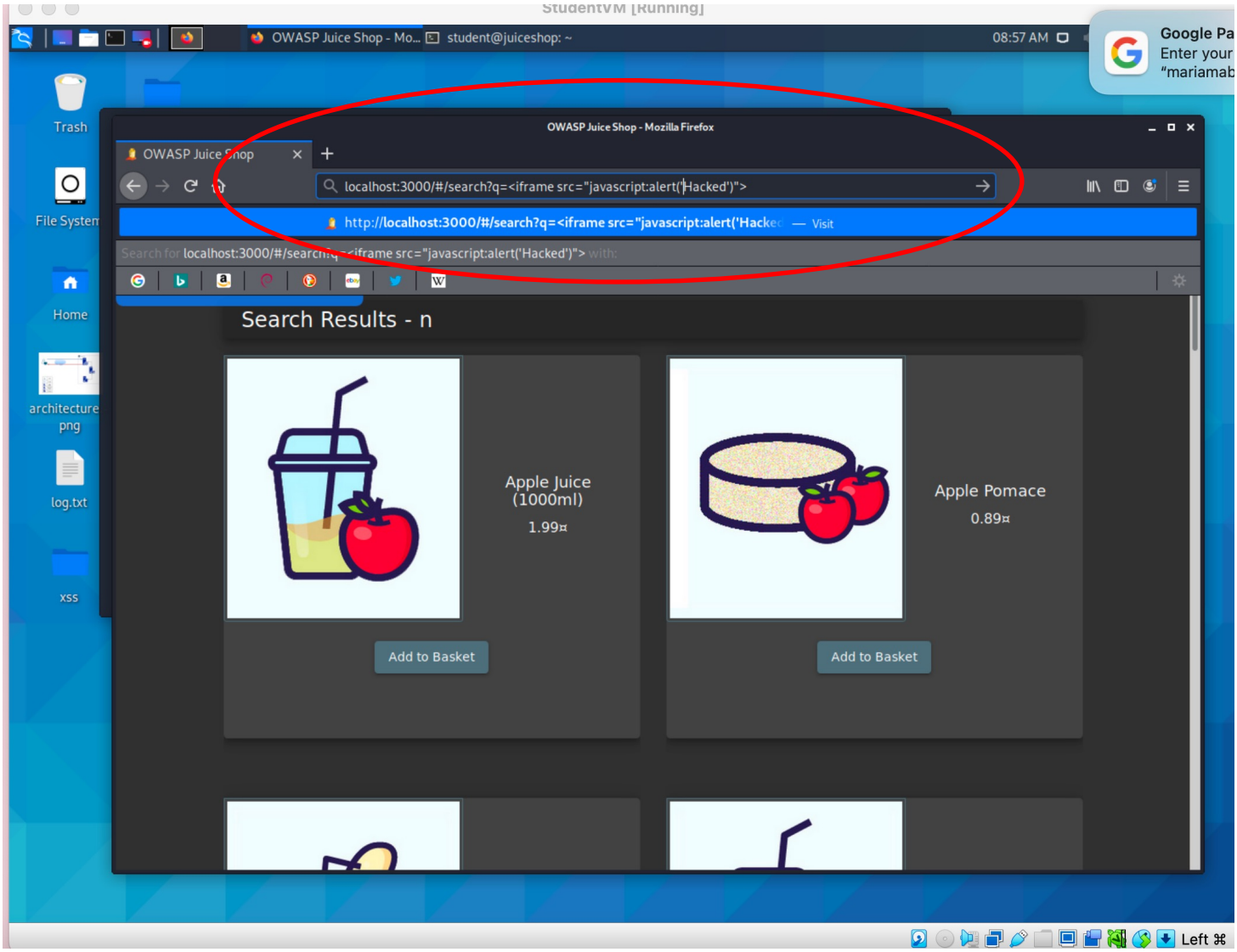




2.2 XSS

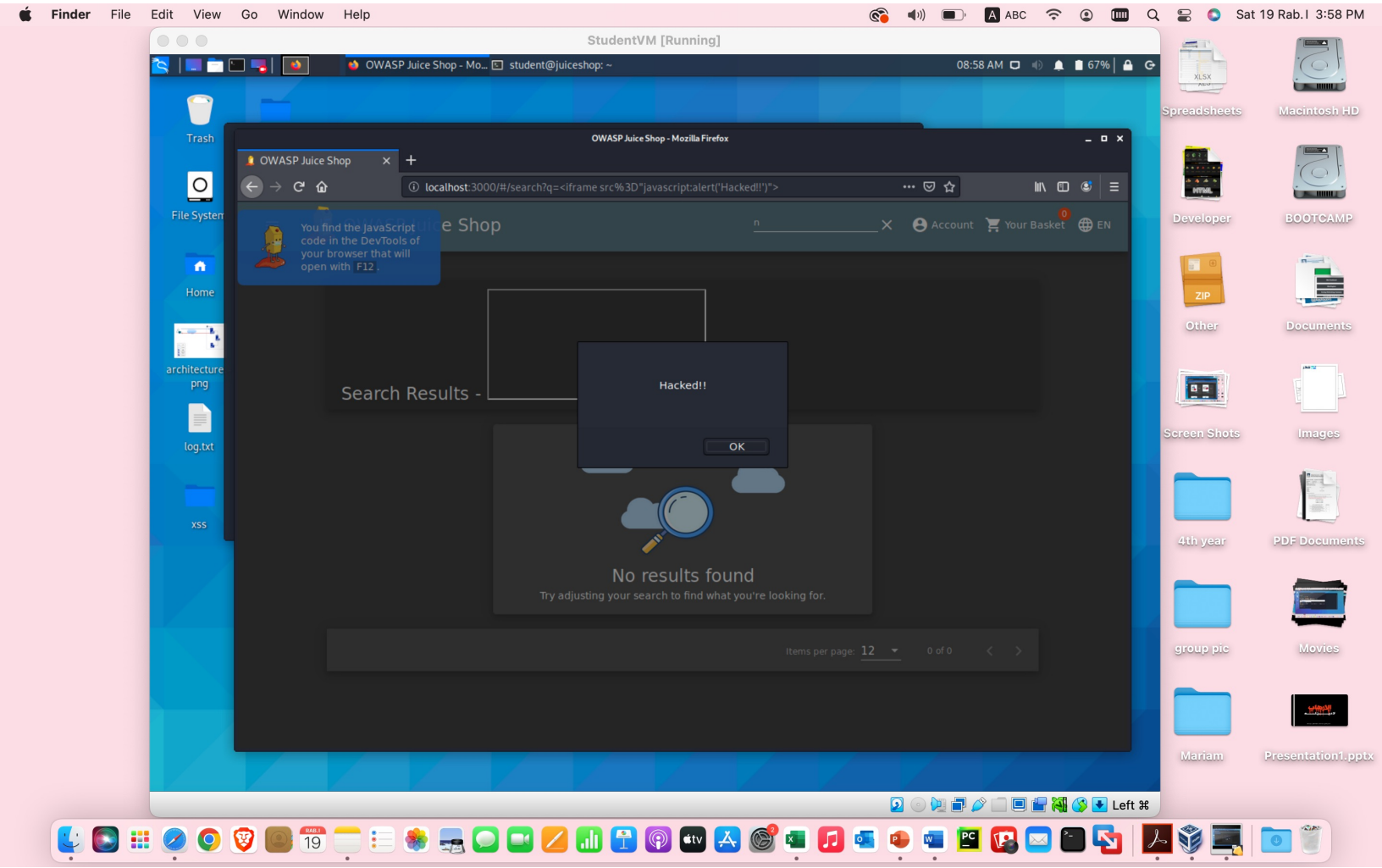
Insert Screenshot of Your Commands Here:

The command: <iframe src="javascript:alert('Hacked!!')">



2.2 XSS

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



Section 3

Risk Analysis

3.1 Scoring Risks

Risk	**Score** ***(1 is most dangerous, 4 is least dangerous)***
DDoS / HTTP flood	1
Insecure Architecture	3
SQL Injection	2
XSS Vulnerability	4

3.2 Risk Rationale

Why Did You Choose That Ranking?

Distributed Denial of Service/The HTTP flood attack comes in first because the most serious threat is always the one that is currently active. Injections are ranked second. The most frequent application vulnerabilities according to OWASP's top 10 list are injections, which are followed by security configuration errors and Cross-Site Scripting (XSS).

Section 4

Mitigation Plan

4.1 Secure Architecture

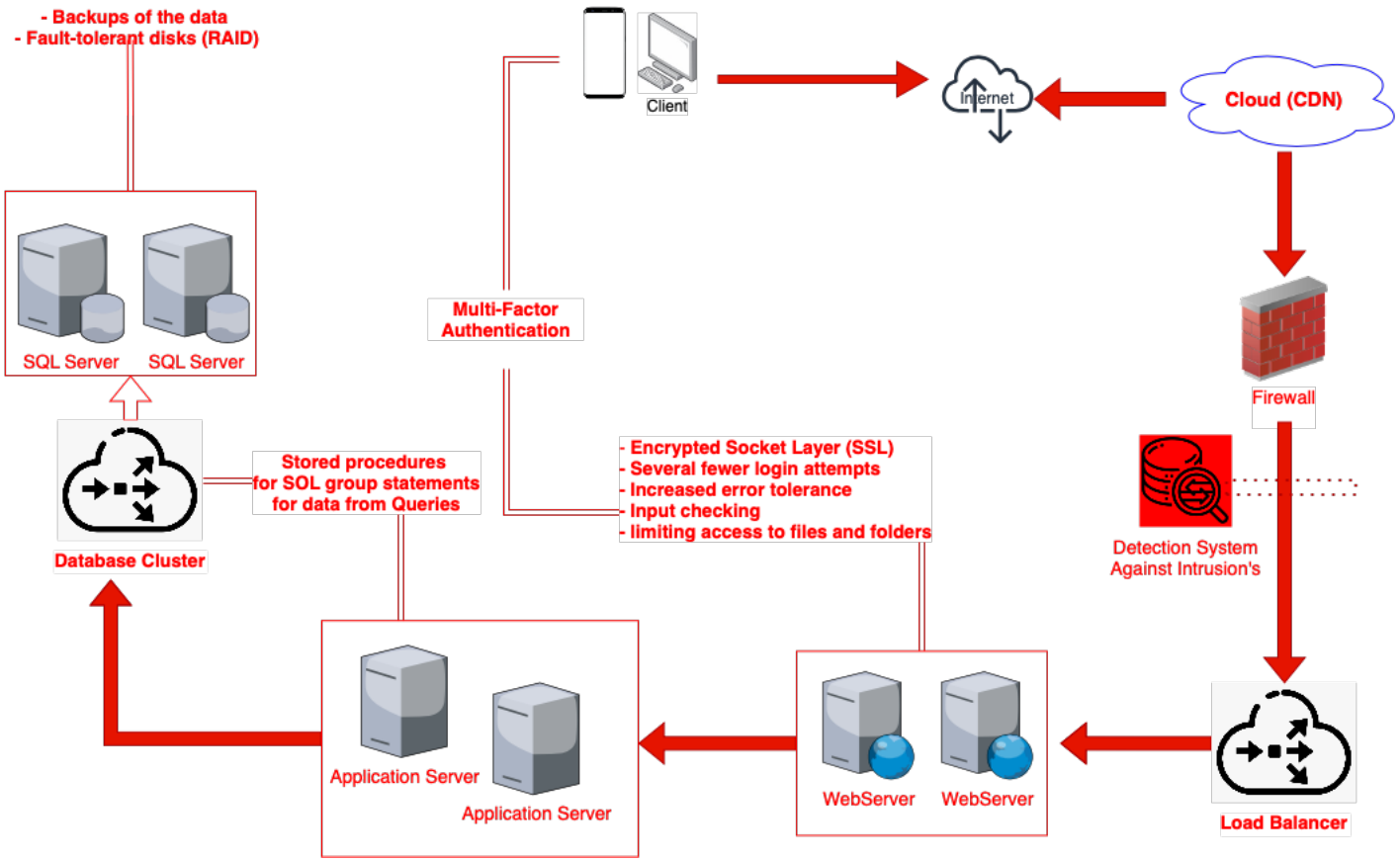
What is Your Mitigation Plan?

We introduce Multifactor Authentication (MFA), which offers two or more verification factors to access each user account, as part of the secure architecture. The firewall protects the network from malicious or unnecessary network traffic and deters outside cyber attackers. The intrusion detection system (IDS) monitors network traffic for suspicious activity and issues alerts when such activity is discovered.

• To split up network traffic among several servers, load balancing will be used.

• Frequent backups of your database will guarantee fault tolerance. Backup components in fault-tolerant systems replace failing ones automatically to prevent service interruptions.

• On edge servers, the Content Delivery Network (CDN) stores content



4.2 Mystery Attack Mitigation

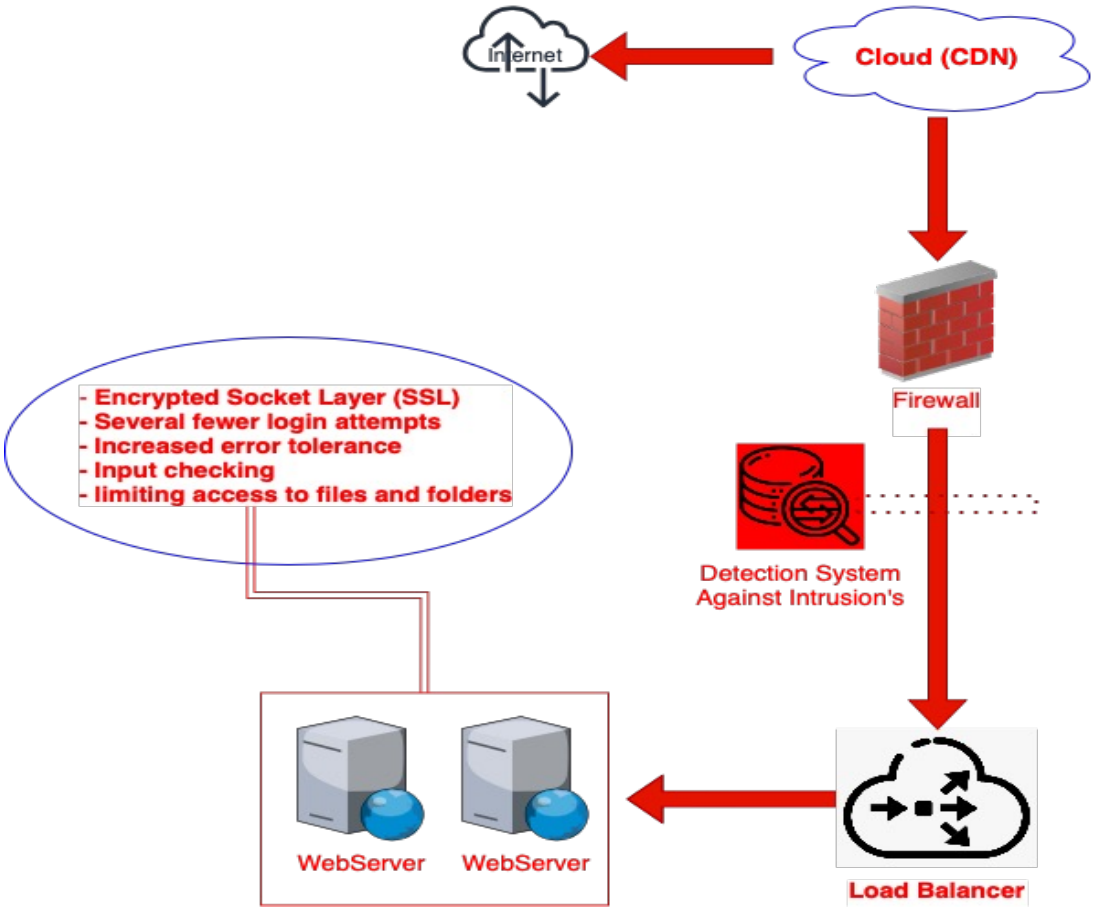
What is Your Mitigation Plan?

To protect web servers from HTTP floods and DDoS, the following techniques can be used:

• A load balancer to spread network traffic among several servers and prevent overload

• A firewall with rate limitations.

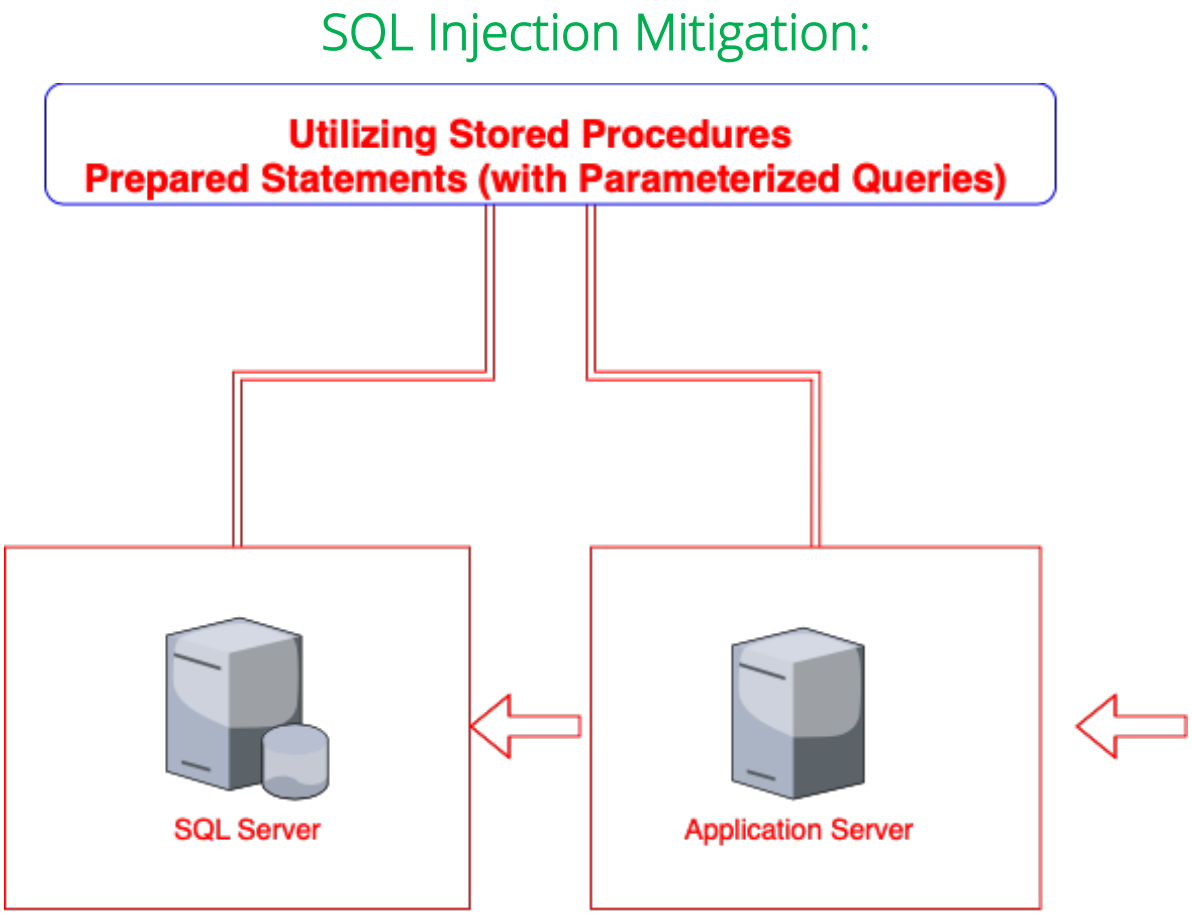
• Implementing Content Delivery Network DDoS-optimized firewalls can detect incomplete connections and flush them from the system when they reach a particular threshold When a data attack occurs, CDN configures cached content on the edge server to handle requests, while IDS keeps track of traffic patterns.



4.3 SQL Injection Mitigation

What is Your Mitigation Plan?

The following techniques can be used to safeguard web servers against SQL Injections: Utilizing Stored Procedures or Prepared Statements (with Parameterized Queries.



4.4 XSS Mitigation

What is Your Mitigation Plan?

The following techniques can be used to lessen XSS attacks:

• User input cleaning: removing the user's input.

• Validation of input. user inputs that have been screened to remove the malicious command sequence. the process of encoding. It is necessary to encrypt all inputs, including special characters, using the appropriate HTML or URL codes.