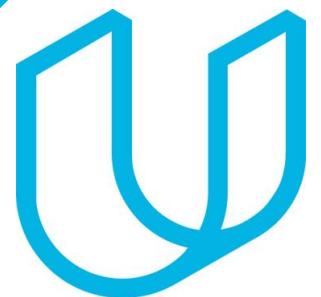


Udajuicer: Threat Report



Shrividya Ranjani Kaliyur
NarayanaPrasad
02/14/2021



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

- **Client Browser:** *Allows the client to connect to a webserver and read the webcontents*
- **Webserver:** *Stores, processes and delivers web content to the clients*
- **Application Server:** *Installs, operates and hosts applications and services required for the end users*
- **Database:** *Organizes, integrates, separates and controls data*

1.1: Asset Inventory

Explanation of How A Request Goes from Client to Server

When a client wants to view some content on the webpage, the client must request for it. The browser sends a HTTPS request to the webserver. The webserver sends a binary request to the application server. The application server requests pages that are stored in the disk, if static webpages are requested. It sends a query call to the database, if data is requested.

The database returns the application server a query response. Then the application server sends the binary response to the webserver, the webserver sends the requested content to the client. The content is sent to the client in the form of an HTTP response.

1.2 Architecture Audit

Flaws

- *Encryption is not applied*
- *Load balancers are not being used to increase the availability of the website*
- *CDN is not being used, so there is no way to handle single point of failure*
- *Network firewalls are not being used*
- *Network segmentation is not applied*

1.3 Threat Model Diagram

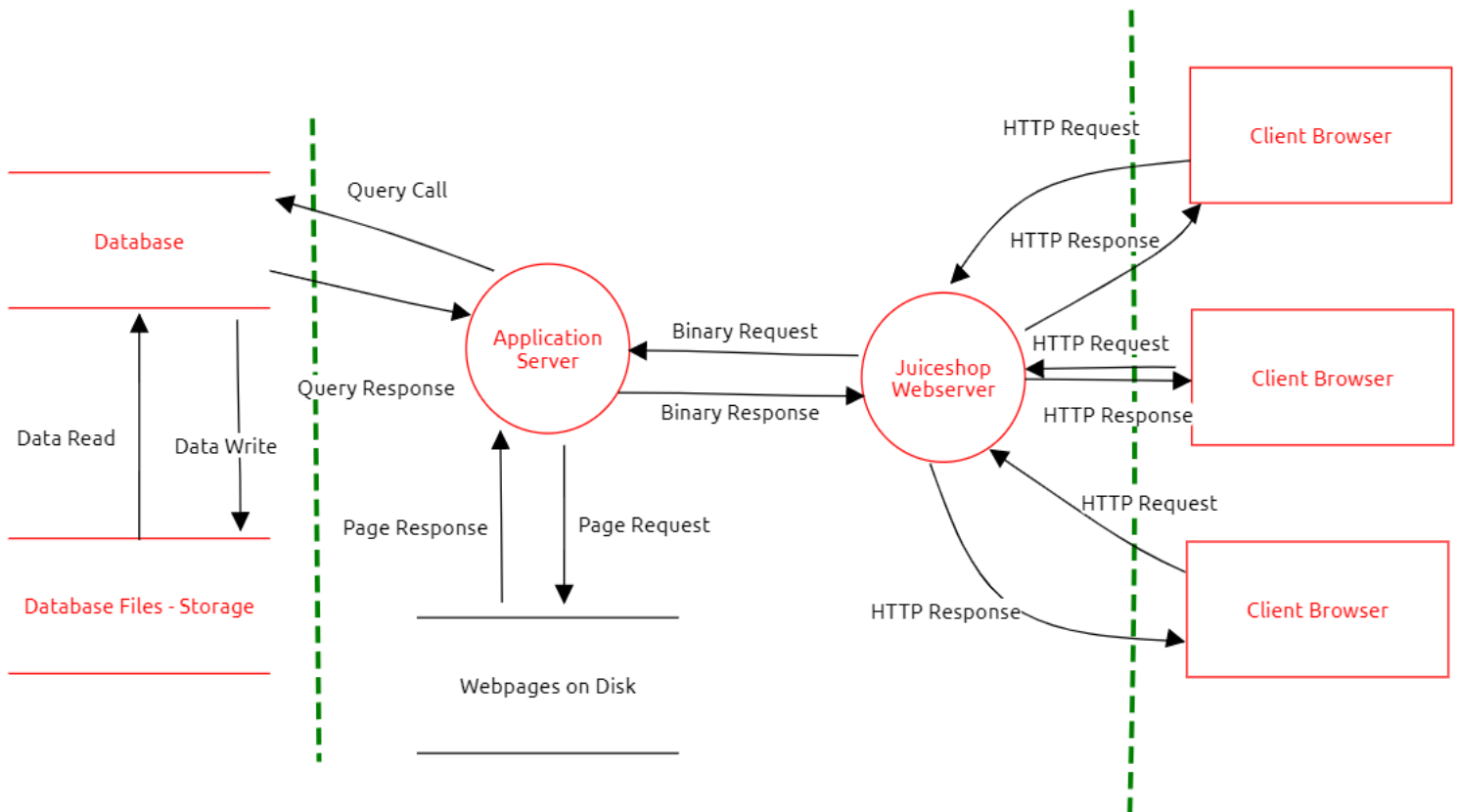
Using OWASP Threat Dragon, build a diagram showing the flow of data in the Juice Shop application and identify 3 possible threats to the Juice Shop. Make sure to include the following components:

- Client
- Web Server
- Application Server
- Database

Threats Identified:

1. *Injection attack*
2. *Cross Site Scripting (XSS)*
3. *Sensitive data exposure*

1.3 Threat Model Diagram



1.4 Threat Analysis

What Type of Attack Caused the Crash?

Distributed Denial of Service (DDoS) attack.

What in the Logs Proves Your Theory?

DDoS attack uses multiple compromised computers to send requests to the server at once, so that there is a denial of service. The log shows us that multiple IP addresses are trying to login to the website at once. This proves that multiple computers were used to send a login request to the server to cause denial of service.

1.5 Threat Actor Analysis

Who is the Most Likely Threat Actor?

A Script Kiddie

What Proves Your Theory?

The threat actor could be a hacktivist but, hacktivists are mission-oriented with social or political ideology. Attacking Udajuicer cannot be subject to a social or political ideology attack.

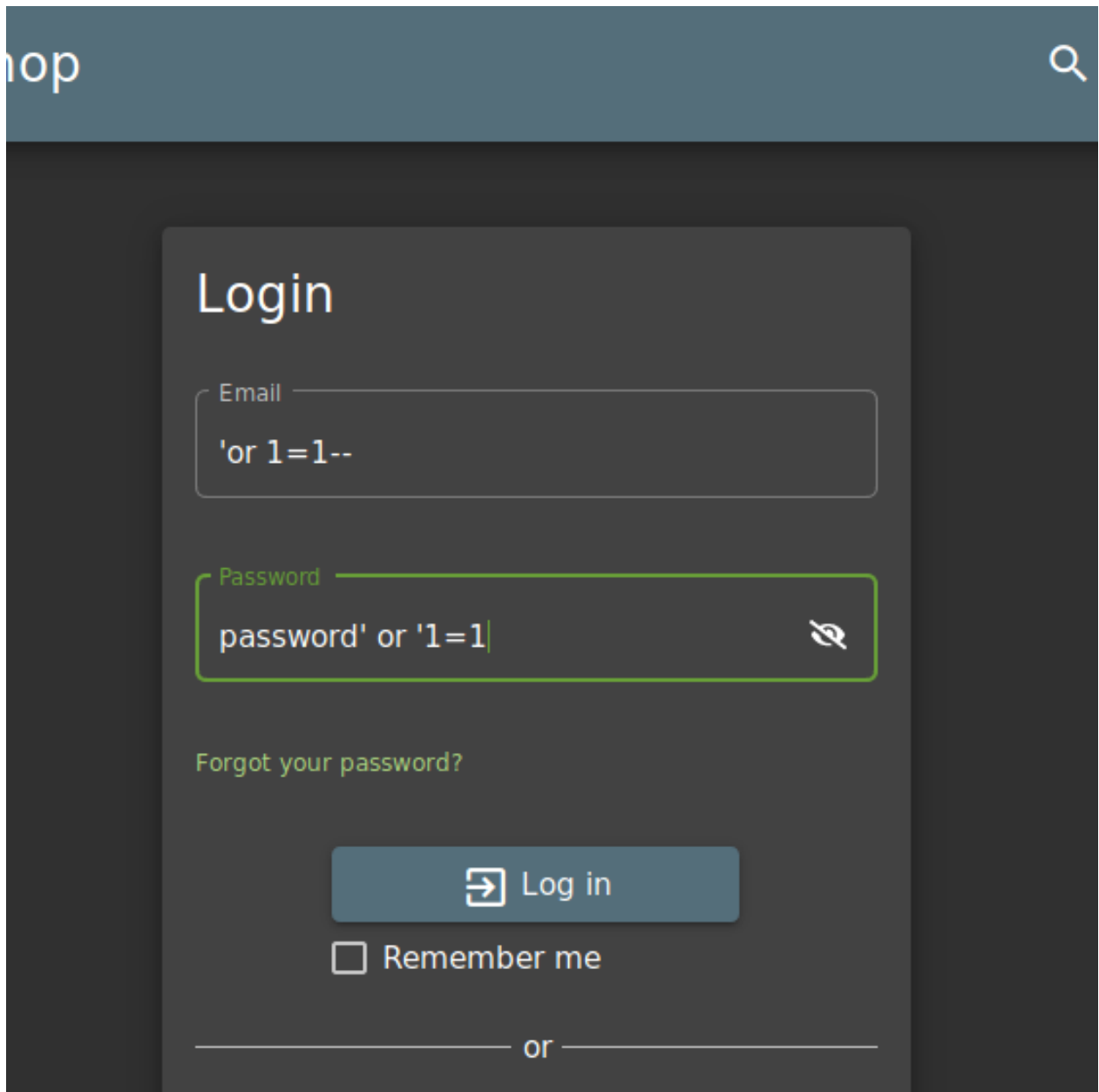
There are scripts available that can be used to execute DDoS attack. Script Kiddies are threat actors who do not have much experience and use already available scripts to perform attacks and cause disturbance. This proves that the act was performed by a Script Kiddie.



Section 2

Vulnerability Analysis

2.1 SQL Injection: Command



The image shows a login form with a dark background. At the top, there is a header bar with the word "Shop" on the left and a magnifying glass icon on the right. The login form itself is a light gray box in the center. It has a title "Login" at the top. Below the title, there are two input fields. The first field is labeled "Email" and contains the text "'or 1=1--". The second field is labeled "Password" and contains the text "password' or '1=1|". To the right of the password field is a small icon of a crossed-out eye. Below the password field, there is a link that says "Forgot your password?". At the bottom of the form, there is a "Log in" button with a right-pointing arrow icon. Below the button is a checkbox labeled "Remember me". At the very bottom, there is a horizontal line with the word "or" in the center, indicating an alternative login method.

Shop

Login


Email

'or 1=1--

Password

password' or '1=1|

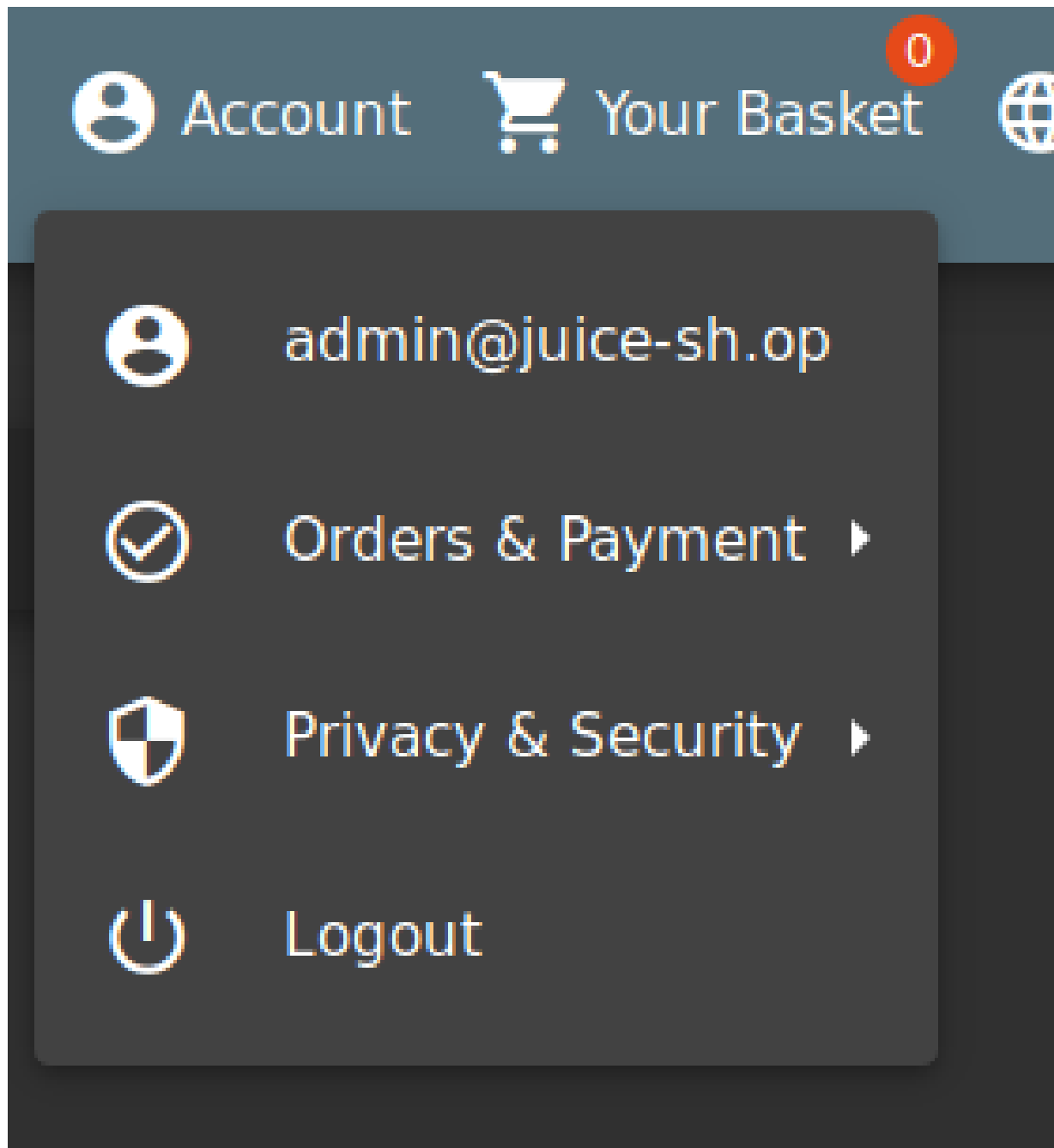
[Forgot your password?](#)

 Log in

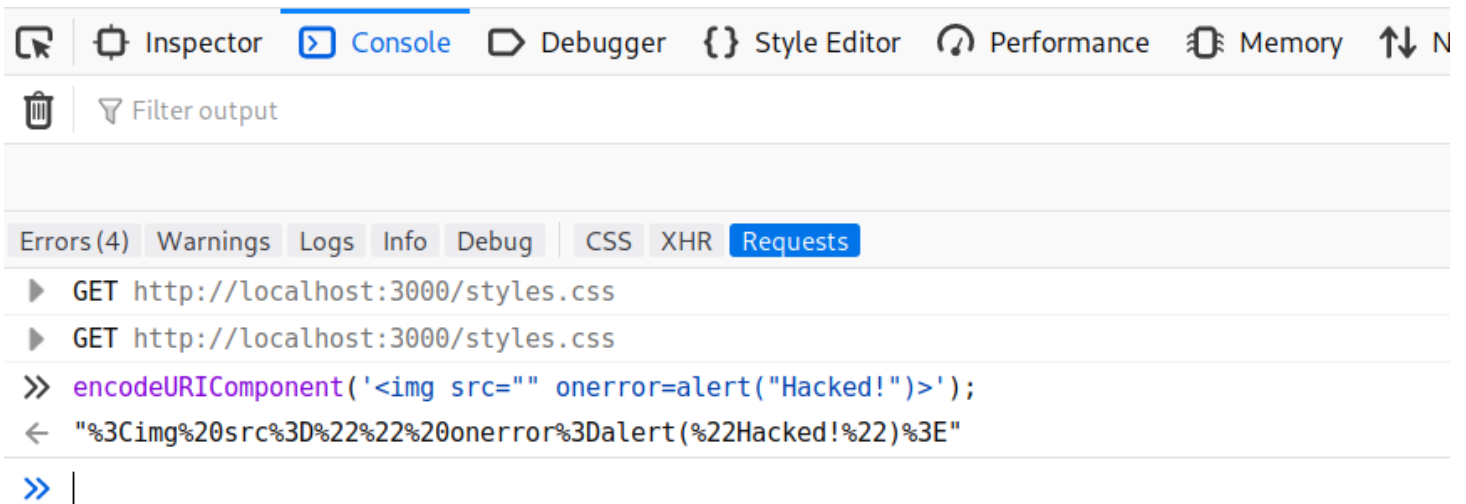
☐ Remember me

or

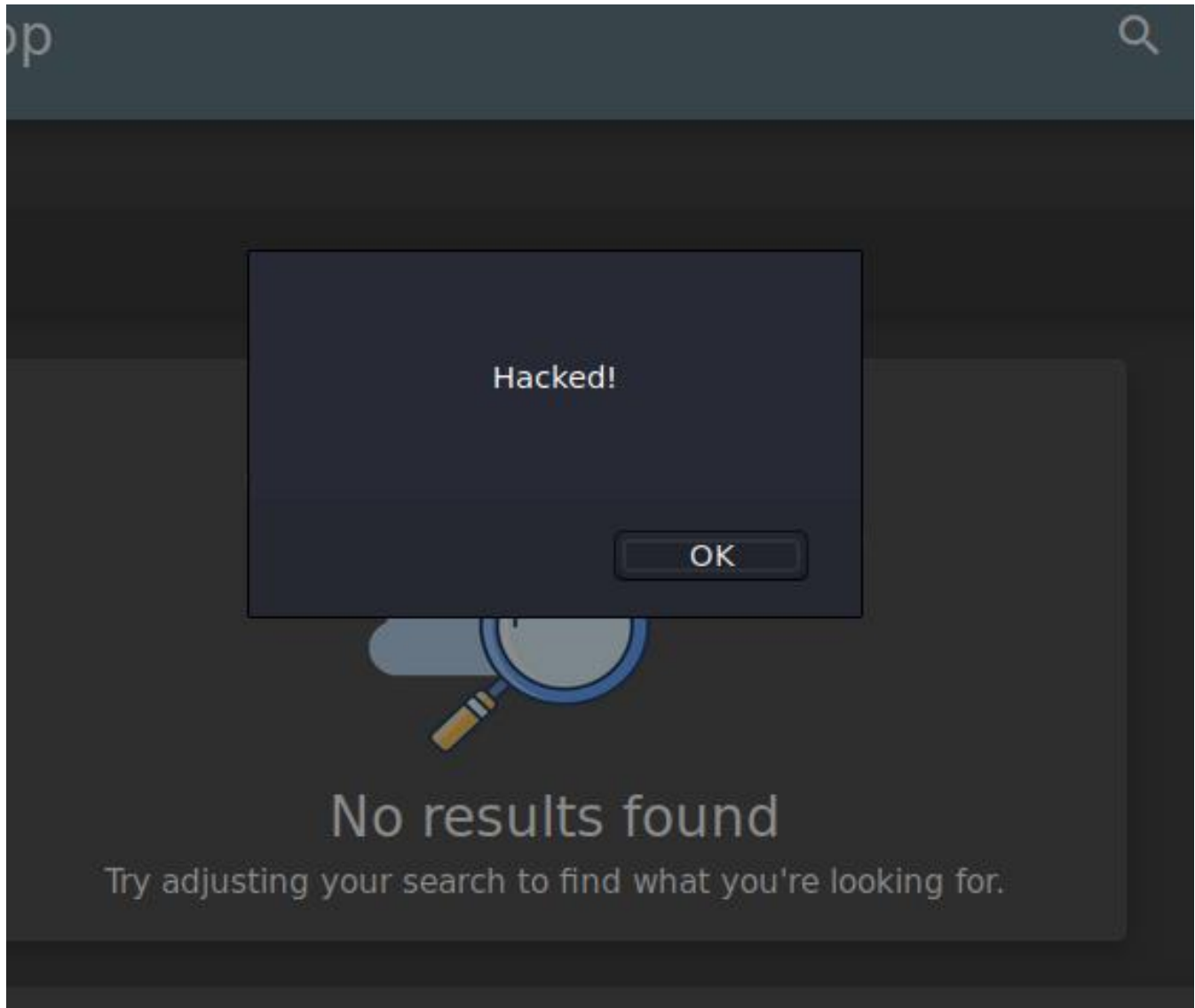
2.1 SQL Injection: Result



2.2 XSS: Command



2.2 XSS: Result



Optional Task:

Extra Vulnerabilities

- *[Vulnerability 1 Here]*
- *[Vulnerability 2 Here]*
- *[Add more vulnerabilities as necessary]*



Section 3

Risk Analysis

3.1 Scoring Risks

| Risk | Score <i>(1 is most dangerous, 4 is least dangerous)</i> |
|-------------------------------|---|
| Distributed Denial of Service | 1 |
| Insecure Architecture | 1 |
| SQL Injection | 2 |
| XSS Vulnerability | 2 |

3.2 Risk Rationale

Why Did You Choose That Ranking?

Distributed Denial of Service – Score-1:

Distributed Denial of Service can lead to invisibility of the website on the internet, cut off interaction with customers. The attacker can send requests, saturate open ports so that authorized users cannot connect. This attack causes unusually slow network performance. The attack can last for hours, days or even longer. The cost of getting out of this situation is very high.

Insecure architecture – Score-1:

Now a days, attackers just wait for a chance to exploit a system and gain access to all the resources and information. If we have an insecure architecture the attacker has a golden chance to find the issue and use it to their advantage. Insecure architecture can lead to Sensitive Information Disclosure, Spoofing, Tampering and DoS threats.

SQL Injection – Score-2:

When an attacker is successful in injection of an SQL command on the webpage, he/she can gain access to any information present on the application. Tables can be removed or modified. Sometimes an attacker might even be able to execute operating system commands.

Cross-Site scripting – Score-2:

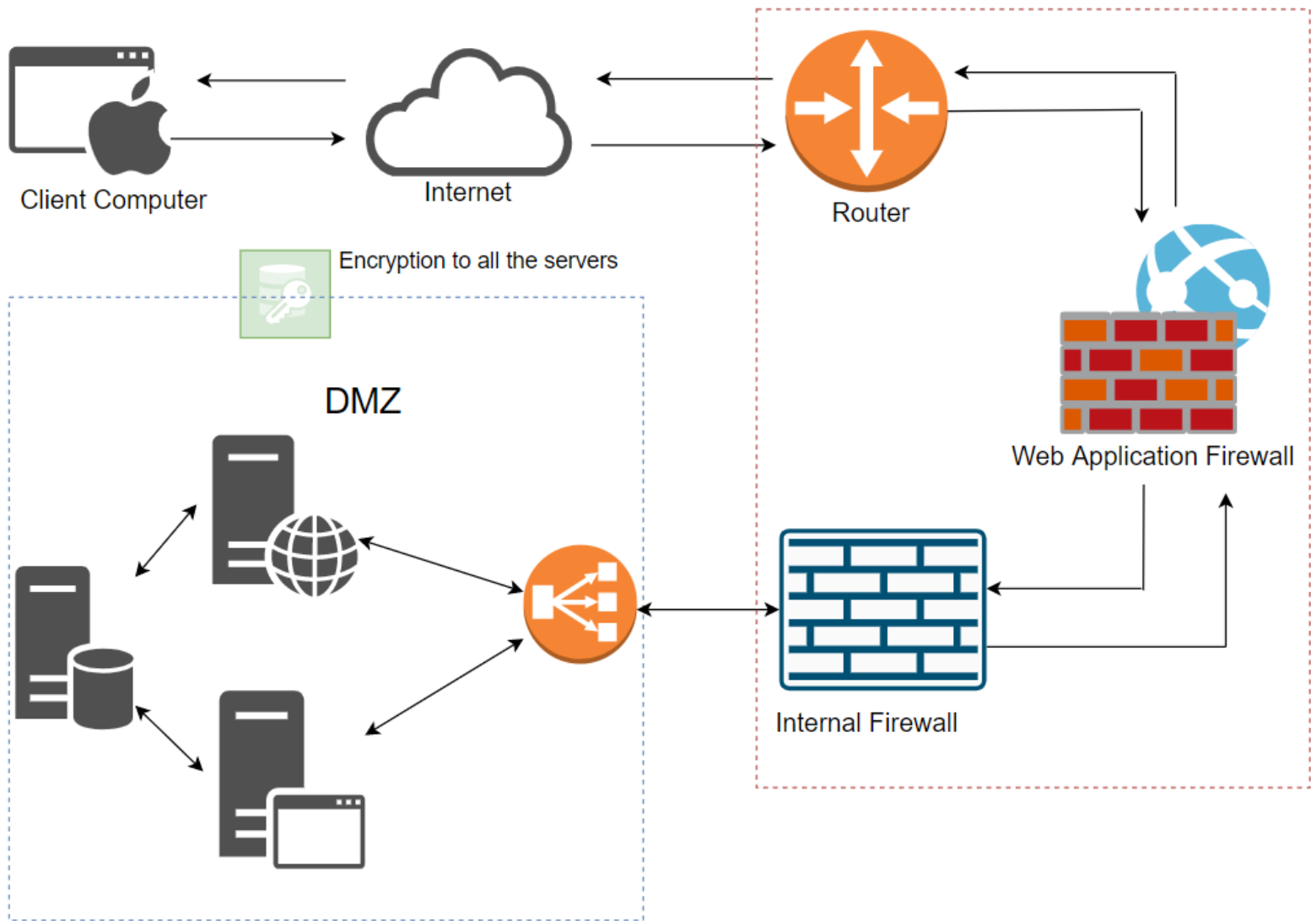
Cross site scripting vulnerability leads to dangerous situations where the attacker gains full control on the victim's browser and can use it for their advantage. The attacker can also hijack the user account by stealing the session cookies and gain access to the user's credentials. This vulnerability also leads to data leakage.



Section 4

Mitigation Plan

4.1 Secure Architecture



4.2 Mystery Attack Mitigation

What is Your Mitigation Plan?

- *Form a secure architecture, use CDN, Load Balancers and Firewalls.*
- *Partition critical online services from other online services.*
- *Have a static website ready and have it available to the users. Static websites require less processing and bandwidth to provide facilities to the users.*

4.3 SQL Injection Mitigation

What is Your Mitigation Plan?

- *User input must not be trusted. Proper user input sanitization and validation must be done before accepting the input on both client side and server side.*
- *Escape special characters in the given input.*
- *Use prepared and parameterized SQL statements.*
- *Use stored procedures instead of writing the queries over and over.*
- *Avoid connecting the application to the database using an account which has root access.*
- *Having an application firewall can help identify SQL injection attacks and block them.*

4.4 XSS Mitigation

What is Your Mitigation Plan?

- *Never trust user input.*
- *Have proper sanitization and validation of user input on both client side and server side.*
- *Remove or encode all the special HTML characters.*
- *Use appropriate response headers.*
- *Have Content Security Policy ready to reduce the severity of an XSS vulnerability.*