Web Application Security Audit Report

For

Exchange Web Application

18th December 2017

Submitted By

Prolitus Technologies Private Ltd.

A-83, 1st FLOOR, SECTOR-2,

NOIDA 201301, UTTAR PRADESH, INDIA

MOB: +91 7042001174, 1 800 806 0712, +91 0120-6517462

Email: info@avyaan.com

AVYAAN
WE PROTECT DATA

Table of Content

1.0 Web Application Security Audit Report for Exchange Web Application 5

1.1 Warning 5

1.2 DISCLAIMER 6

1.3 DOCUMENT DETAILS 7

1.4 Version History Information 7

2.0 INTRODUCTION 8

2.1 ASSESSMENT OBJECTIVE 8

2.2 SCOPE 8

3.0 TERMS, DEFINITIONS AND LEGENDS 9

3.1 Vulnerability Table 9

3.2 Tile of Findings - A short title that describe the findings. 10

4.0 METHODOLOGY 11

4.1 Information Gathering 11

4.2 Test Application 12

4.3 Classify Finding According to the Risk Level 12

4.4 Report Writing 12

5.0 EXECUTIVE SUMMARY 13

5.1 Some of the Major Findings are 13

5.2 Graph Representations of the Vulnerabilities 14

5.3 Summary of the Key Observation 15

6.0 DETAILED REPORT AND RECOMMENDATIONS 18

6.1 User Enumeration 18

6.1.1 In Forgot Password 19

6.1.2 In Admin Login 20

6.1.3 In Customer Login 22

6.1.4 In Registration 23

6.1.5 In Email Verify 24

6.1.6 In 2FA Generation 25
6.1.7 In Signup 2FA Code 26
6.2 Application Logic Bypass (Register as Admin) 27
6.3 2FA Bypass at Login 29
6.4 Insecure Direct Object Reference in Email Verify 31
6.5 Insecure Direct Object Reference in 2FA Generation 33
6.6 Improper Session Management 35
6.7 Improper Server Side Validation 36
6.8 No Captcha on Registration 38
6.9 Multiple Time Registration of Customer 40
6.10 Error Message Display 42
6.11 SQL Injection 44
6.12 Insecure Direct Object Reference in Password Reset 48
6.13 Improper Password Policy Implementation 50
6.14 Privilege Escalation 53
6.14.1 In Get New Customer API 54
6.14.2 In Get All Customer List API 56
6.14.3 In Edit Customer Profile API 58
6.14.4 In Get Currency List API 58
6.14.5 In Edit Currency API 58
6.14.6 In Update Currency API 58
6.14.7 In Add Currency API 59
6.14.8 In Upload Currency Icon API 59
6.14.9 In Get All Trade Currency Pairs API 59
6.14.10 In Update Default Currency Pairs API 59
6.14.11 In Save Currency Pair API 60
6.14.12 In Delete Currency Pair API 60
6.14.13 In Get Commissions API 60
6.14.14 In Get Trade Limits API 60

6.14.15 In Get All Country List API 61
6.14.16 In Add Country API 61
6.14.17 In Edit Country API 61
6.14.18 In Update Country API 62
6.14.19 In Delete Country API 62
6.14.20 In Get All State List API 62
6.14.21 In Delete Country API 63
6.14.22 In Add State API 63
6.14.23 In Edit State API 63
6.14.24 In Update State API 64
6.14.25 In Delete State API 64
6.14.26 In Get All City List API 64
6.14.27 In Get State By Id API 65
6.14.28 In Add City API 65
6.14.29 In Edit City API 65
6.14.30 In Update City API 66
6.14.31 In Delete City API 66
6.14.32 In Get Active Currency Icons API 66
6.14.33 In Get Trade Currency Pairs API 67
6.14.34 In Get States By Country Id API 67
6.15 Malicious File Upload 68
6.15.1 In Uploading Currency Icon 69

1.0 Web Application Security Audit Report for Exchange Web Application

1.1 Warning

THIS DOCUMENT, AND ALL ACCOMPANYING MATERIALS, MAY CONTAIN INFORMATION THAT COULD SEVERELY DAMAGE OR IMPACT THE INTEGRITY AND SECURITY OF THE ORGANIZATION IS DISCLOSED PUBLICLY. THIS DOCUMENT, AND ALL ACCOMPANYING MATERIALS, SHOULD BE SAFEGUARDED AT ALL TIMES AND MAINTAINED IN A SECURE AREA WHEN NOT IN USE. AVYAAN ASSUMES NO RESPONSIBILITY OR LIABILITY FOR THE SECURITY OF THIS DOCUMENT OR ANY ACCOMPANYING MATERIALS AFTER DELIVERY TO THE ORGANIZATION NAMED HEREIN. IT IS THE ORGANIZATION'S RESPONSIBILITY TO SAFEGUARD THIS MATERIAL AFTER DELIVERY.

THIS REPORT CONTAINS PROPRIETARY INFORMATION THAT IS NOT TO BE SHARED, COPIED, DISCLOSED OR OTHERWISE DIVULGED WITHOUT THE EXPRESS WRITTEN CONSENT OF AVYAAN OR THEIR DESIGNATED REPRESENTATIVE. USE OF THIS REPORTING FORMAT BY OTHER THAN AVYAAN OR ITS SUBSIDIARIES IS STRICTLY PROHIBITED AND MAY BE PROSECUTED TO THE FULLEST EXTENT OF THE LAW.

1.2 DISCLAIMER

THE RECOMMENDATIONS CONTAINED IN THIS REPORT ARE BASED ON INDUSTRY STANDARD "BEST PRACTICES". BEST PRACTICES ARE, BY NECESSITY, GENERIC IN NATURE AND MAY NOT TAKE INTO ACCOUNT EXACERBATING OR MITIGATING CIRCUMSTANCES. THESE RECOMMENDATIONS, EVEN IF CORRECTLY APPLIED, MAY CAUSE CONFLICTS IN THE OPERATING SYSTEM OR INSTALLED APPLICATIONS. ANY RECOMMENDED CHANGES TO THE OPERATING SYSTEM OR INSTALLED APPLICATION SHOULD FIRST BE EVALUATED IN A NON-PRODUCTION ENVIRONMENT BEFORE BEING DEPLOYED IN YOUR PRODUCTION NETWORK.

1.3 DOCUMENT DETAILS

Document Title	Web Application Security Assessment Report
Company	Avyaan
Client	Exchange Web Application
URL	https://exchange-dev.sofodev.co/
Classification	Confidential
Document Type	Report
Version	1.0
Submission Date	18th December 2017
Author	Himanshu Sharma
Reviewed By	Nikhil Kumar
Approved By	Nikhil Kumar
Engineering head's contact	security@prolitus.com

1.4 Version History Information

Date	Version	Author	Comments
18 December 2017	**v1.0**	Himanshu Sharma	Vulnerabilities Reported

2.0 INTRODUCTION

The purpose of the security assessment was to establish a baseline of information that could be obtained about the application and assets. Specifically, we performed procedures to obtain an understanding, and assess, the potential vulnerabilities associated with the web applications available for access via the Internet. Mentioned below are our findings of the application security assessment.

2.1 ASSESSMENT OBJECTIVE

The Objective of this engagement was to

- Identify and assess security flaws in web application according to industry principal security standards.
- Provide recommendations for mitigation of risk(s) emerged during the identified vulnerabilities.

2.2 SCOPE

Avyaan performed the vulnerability tests for the below stated specific scope.

No	Audit Scope	Description
1.	https://exchange-dev.sofodev.co	

3.0 TERMS, DEFINITIONS AND LEGENDS

This section describes the format in which the identified vulnerabilities are reported in the later section of the report. "Vulnerability Table" shown below is used to provide the details of the vulnerability, its impact and the recommendations.

3.1 Vulnerability Table

No	
Vulnerability Name	
Severity	
Description	
Recommendation	
Affected URL	
Screen Shot	

- **Title of the Vulnerability –** A short title that describes the vulnerability.
- **Risk Level –** It describes the risk level. The title bar of each vulnerability table is color coded for quick identifications of the severity level of the vulnerabilities.
- **Description –** It provides a brief description of the vulnerability.
- **Impact –** Describes the probable impact if the vulnerability is successfully exploited.
- **Recommendations –** Provide the recommendations to fix the vulnerability.
- **Affected URL –** Provides the list of the affected URL's.
- **Screen Shot**- Give the Screen Shot of the Vulnerability

3.2 Tile of Findings - A short title that describe the findings.

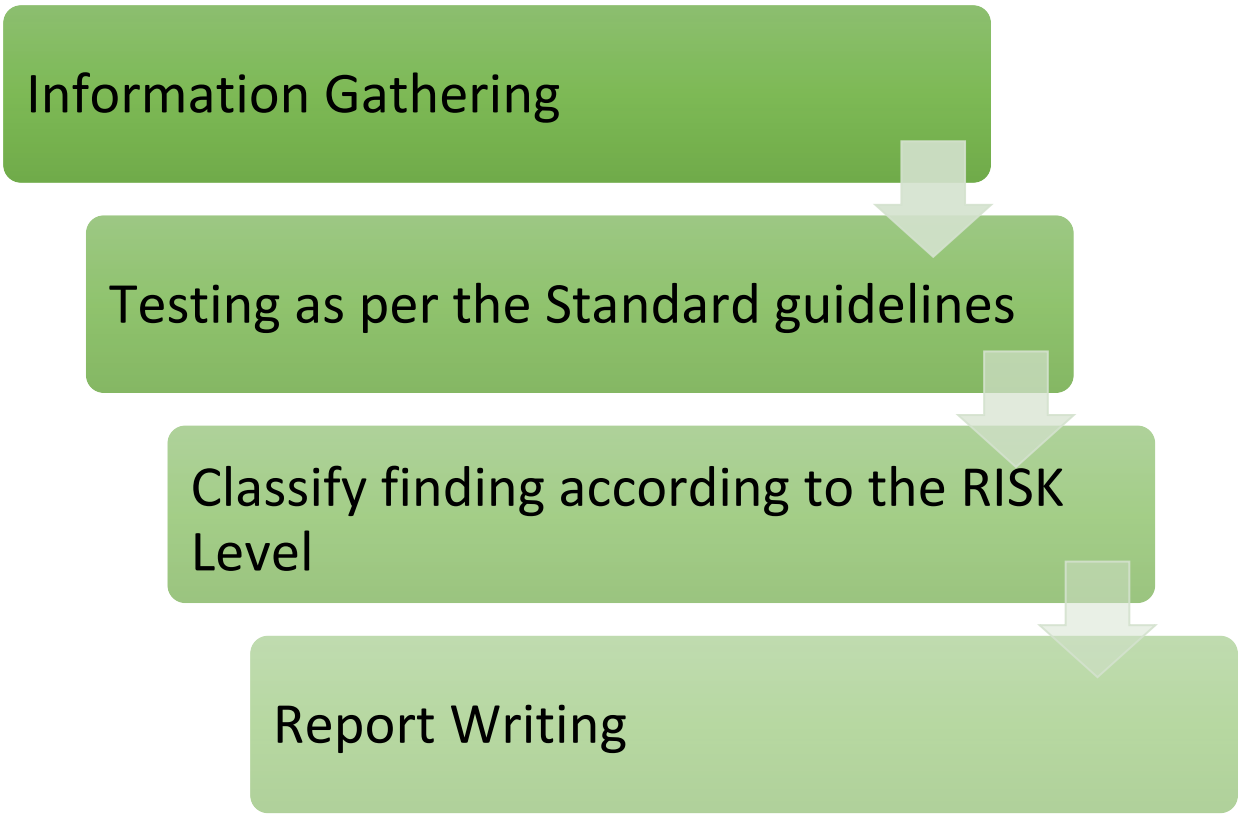
The title bar of each finding is colour coded for a quick identification of the Risk Level. Title bar colour codes are as follows:

Risk Exposure	Description
Critical	Intruders can easily gain control of system and network. This needs immediate attention.
High	Intruders can possibly gain control of the host, or there may be potential leakage of highly sensitive information. This should be addressed as soon as possible.
Medium	Intruders may be able to collect sensitive information from the host, such as the precise version of software installed. With this information, intruders can easily exploit known vulnerabilities specific to software versions. Address this the next time you perform a minor reconfiguration of the host.
Low	Intruders can collect information about the host (open ports, services, etc.) and may be able to use this information to find other vulnerabilities. Address this the next time you perform a major reconfiguration of the host.

4.0 METHODOLOGY

4.1 Information Gathering

One of the first steps the methodology applied in Web Application Security Testing is explained in the diagram below.



Of this test is to identify the Web application environment, including the scripting language and Web server software in use, and the operating system of the target server. However, this step is generally omitted if the testing is limited to just the web application and not the host.

4.2 Test Application

While testing the application, we follow but are not limited to the OWASP standards. The top 10 vulnerabilities are tested for static and dynamic web sites. Our testing is done manually as well as using tools. An indicative list of tools is given in the section below.

4.3 Classify Finding According to the Risk Level

After an exhaustive testing, the findings are compiled and classified according to a Risk Level of High, Medium or Low depending on the harm they may cause to the Web Application, server or to the network.

4.4 Report Writing

It is the outcome of the security audit in which a report is created highlighting the findings together with details for each finding, POC and recommendation.

5.0 EXECUTIVE SUMMARY

The purpose of reporting existing security loopholes in the web application and also to provide with recommendation to rectify the problems. This Web Application Security Assessment Report assesses the use of resources and controls to eliminate and/or manage vulnerabilities that are exploitable by threats internal and external client's infrastructure. The scope of this security assessment effort was limited to the security controls applicable to the client's system environment.

The methodology used to conduct this security assessment is qualitative, and no attempt was made to determine any annual loss expectancies, asset cost projections, or cost-effectiveness of security safeguard recommendations. The Approach uses OWASP, WASC and SANS and other industry best practices that are used industry-wide by security and audit professionals.

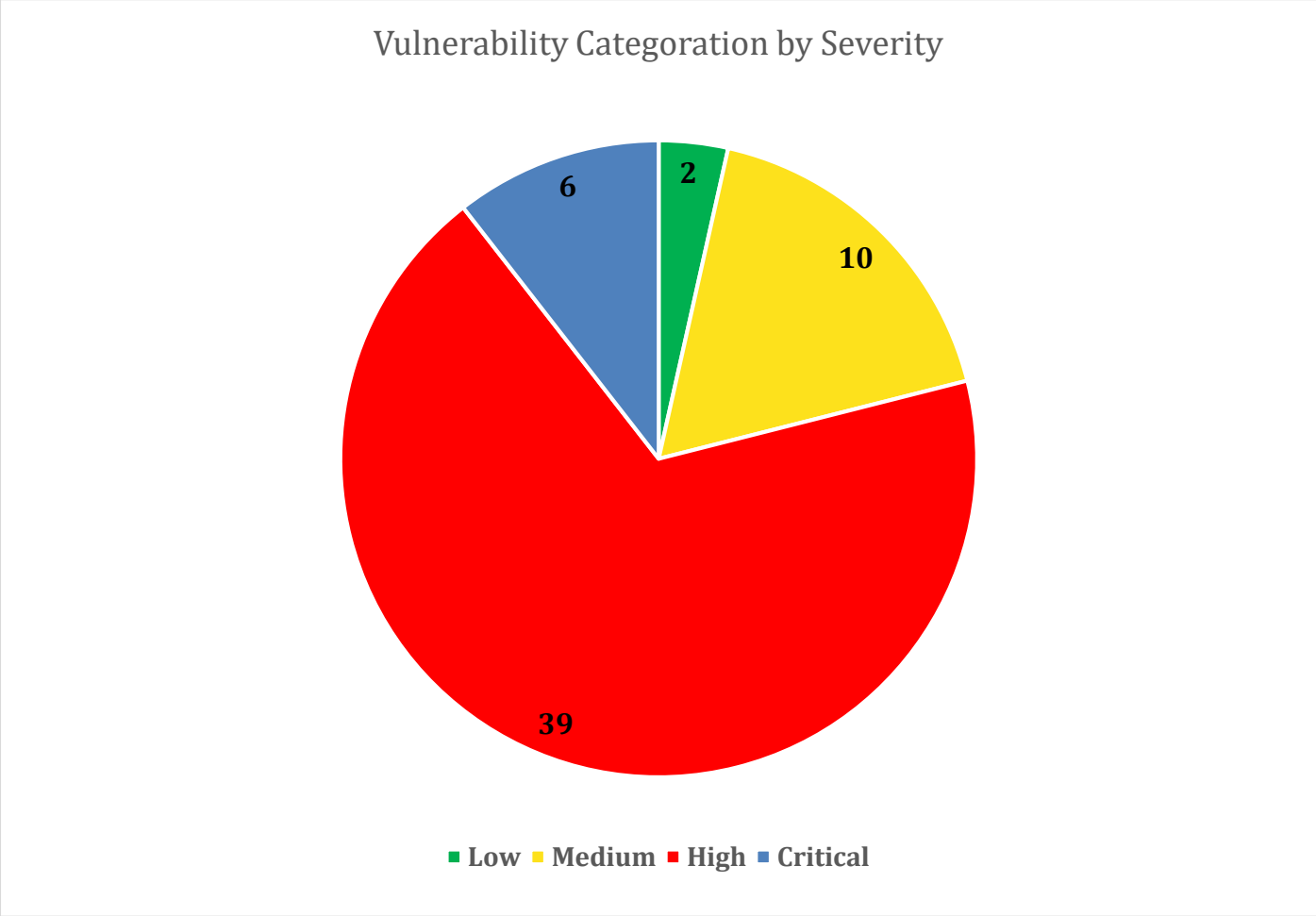
The overall client's web application security categorization is rated as High in accordance with industry standard. If the safeguards recommended in this security assessment are not implemented, the result could be modification or destruction of data, disclosure of sensitive information, or denial of service to the users who require the information on a frequent basis.

5.1 Some of the Major Findings are

- It was found that Access Management Controls were not properly implemented in the application. We have successfully downloaded internal files without any authentication.
- No CSRF Protection is Implemented in the Application.
- Enumeration of Credentials can be easily done by any malicious user.

5.2 Graph Representations of the Vulnerabilities

The following graph presents the total number of vulnerability and their severity levels.



Severity	No. of Risks
Low	2
Medium	10
High	39
Critical	6

No. of vulnerabilities by severity

5.3 Summary of the Key Observation

Following table presents a list of identified vulnerabilities and their severity level with their current status:

S No	Vulnerability Name	Severity
6.1	User Enumeration	MEDIUM
6.1.1	In Forgot Password	MEDIUM
6.1.2	In Admin Login	MEDIUM
6.1.3	In Customer Login	MEDIUM
6.1.4	In Registration	MEDIUM
6.1.5	In Email Verify	MEDIUM
6.1.6	In 2FA Generation	MEDIUM
6.1.7	In Signup 2FA Code	MEDIUM
6.2	Application Logic Bypass (Register as Admin)	CRITICAL
6.3	2FA Bypass at Login	HIGH
6.4	Insecure Direct Object Reference in Email Verify	CRITICAL
6.5	Insecure Direct Object Reference in 2FA Generation	CRITICAL
6.6	Improper Session Management	MEDIUM
6.7	Improper Server Side Validation	MEDIUM
6.8	No Captcha on Registration	LOW
6.9	Multiple Time Registration of Customer	LOW
6.1	Error Message Display	HIGH
6.11	SQL Injection	HIGH
6.12	Insecure Direct Object Reference in Password Reset	CRITICAL
6.13	Improper Password Policy Implementation	HIGH
6.14	Privilege Escalation	HIGH
6.14.1	In Get New Customer API	HIGH

6.14.2	In Get All Customer List API	HIGH
6.14.3	In Edit Customer Profile API	HIGH
6.14.4	In Get Currency List API	HIGH
6.14.5	In Edit Currency API	HIGH
6.14.6	In Update Currency API	HIGH
6.14.7	In Add Currency API	HIGH
6.14.8	In Upload Currency Icon API	HIGH
6.14.9	In Get All Trade Currency Pairs API	HIGH
6.14.10	In Update Default Currency Pairs API	HIGH
6.14.11	In Save Currency Pair API	HIGH
6.14.12	In Delete Currency Pair API	HIGH
6.14.13	In Get Commissions API	HIGH
6.14.14	In Get Trade Limits API	HIGH
6.14.15	In Get All Country List API	LOW
6.14.16	In Add Country API	HIGH
6.14.17	In Edit Country API	LOW
6.14.18	In Update Country API	HIGH
6.14.19	In Delete Country API	HIGH
6.14.20	In Get All State List API	LOW
6.14.21	In Delete Country API	HIGH
6.14.22	In Add State API	HIGH
6.14.23	In Edit State API	LOW
6.14.24	In Update State API	HIGH
6.14.25	In Delete State API	HIGH
6.14.26	In Get All City List API	LOW
6.14.27	In Get State By Id API	LOW

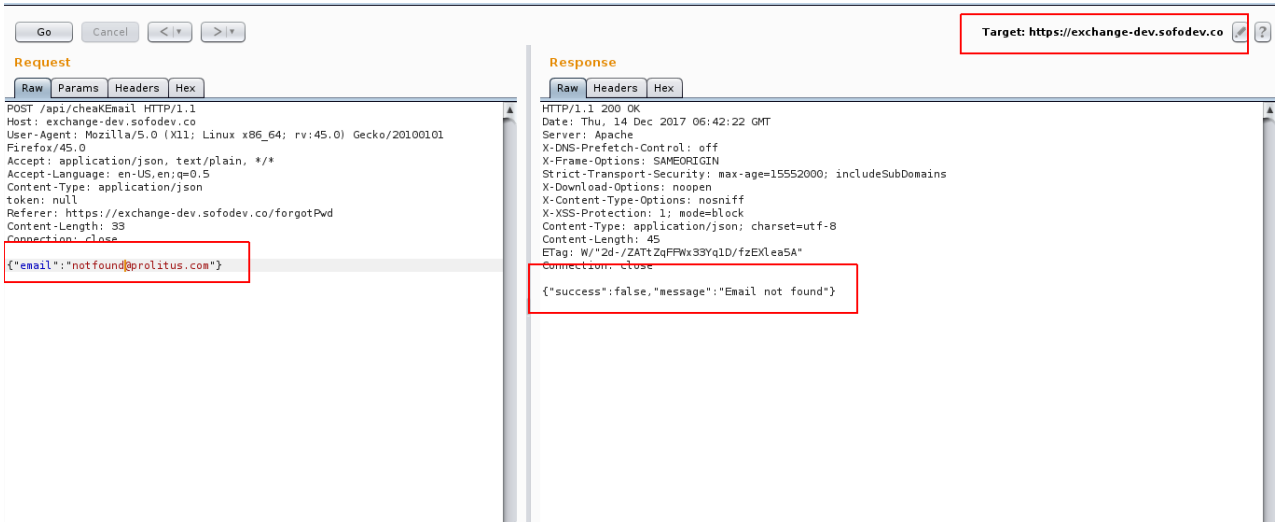
6.14.28	In Add City API	HIGH
6.14.29	In Edit City API	LOW
6.14.30	In Update City API	HIGH
6.14.31	In Delete City API	HIGH
6.14.32	In Get Active Currency Icons API	LOW
6.14.33	In Get Trade Currency Pairs API	LOW
6.14.34	In Get States by Country Id API	LOW
6.15	Malicious File Upload	CRITICAL
6.15.1	In Uploading Currency Icon	CRITICAL

6.0 DETAILED REPORT AND RECOMMENDATIONS

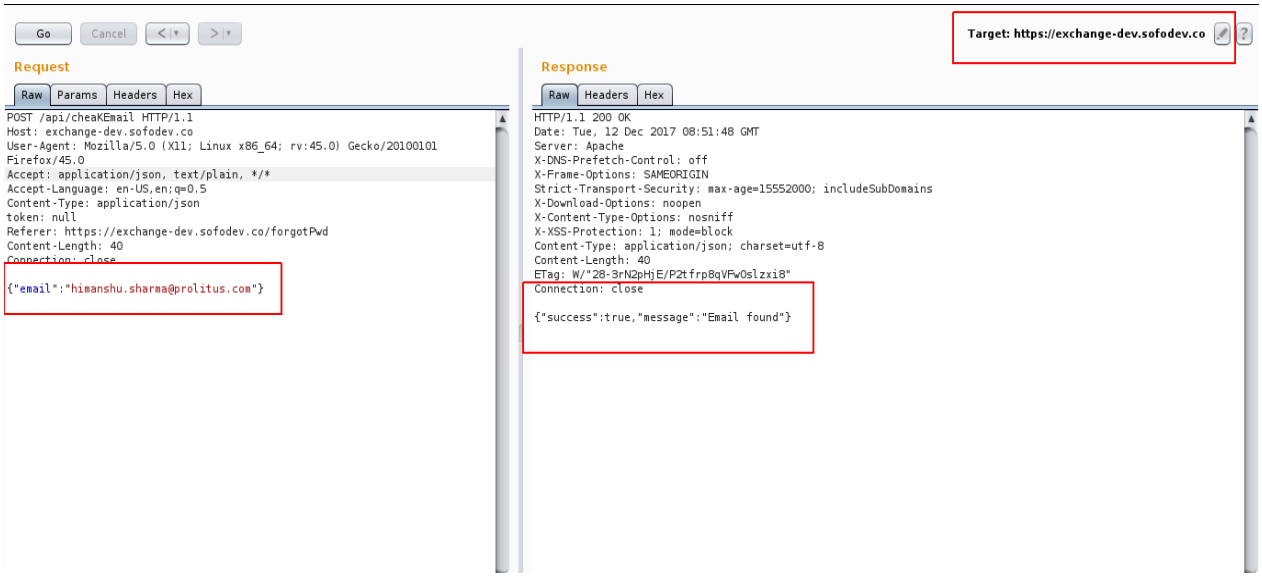
6.1 User Enumeration	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
Description	This Vulnerability Arises when Application responds with a message that symbolizes that we have used a valid credentials. For instance, we use an email which is known to us and is valid and get response and comparing it with invalid number and can differentiate according to response.
Recommendation	Avoid to send messages to client side that reveals whether the credentials used is valid or not.

6.1.1 In Forgot Password	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
URL	https://exchange-dev.sofodev.co/api/cheaKEmail

Screenshot 1: Response "Email not found" revealing we have not used valid email:

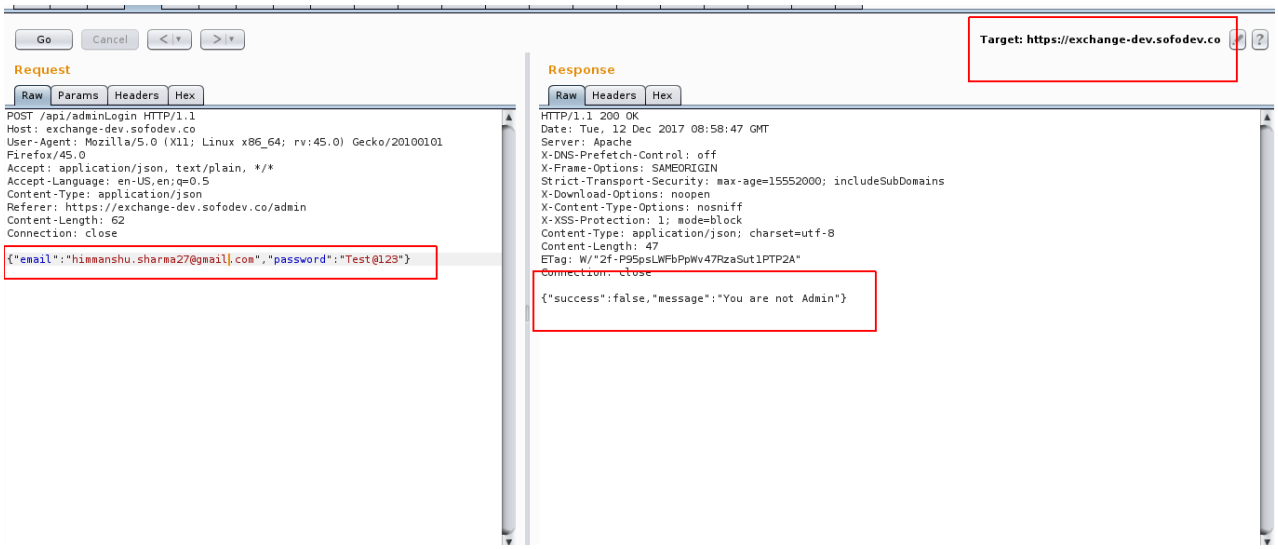


Screenshot 2: Response "Email found" revealing we have used valid email:



6.1.2 In Admin Login	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
URL	https://exchange-dev.sofodev.co/api/adminLogin

Screenshot 1: Response "You are not admin" revealing we have not used valid admin email:



Screenshot 2: Response "Password not match" revealing we have used valid admin email:

Go Cancel < > Target: https://exchange-dev.sofodev.co

Request

Raw Params Headers Hex

```
POST /api/adminLogin HTTP/1.1
Host: exchange-dev.sofodev.co
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101
Firefox/45.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Referer: https://exchange-dev.sofodev.co/admin
Content-Length: 59
Connection: close

{"email":"himanshu.sharma@prolitus.com","password":"T@123"}
```

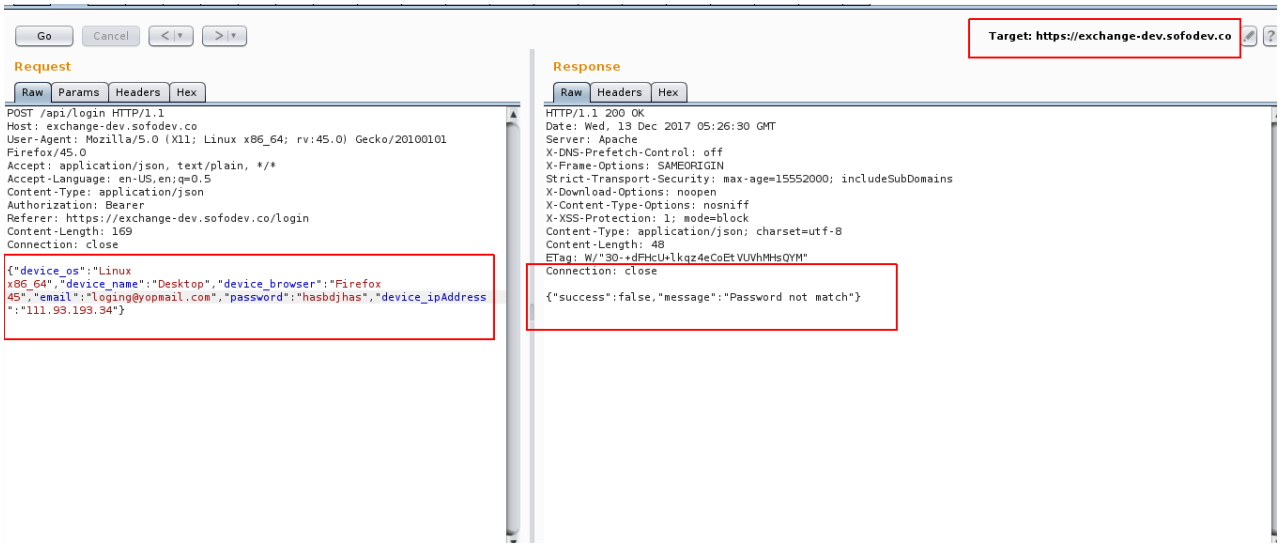
Response

Raw Headers Hex

```
HTTP/1.1 200 OK
Date: Thu, 14 Dec 2017 06:50:23 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 48
ETag: W/"30-22K98m9u2YySL1x+epwNWiX4jP4"
Connection: close

{"success":false,"message":"password not match"}
```

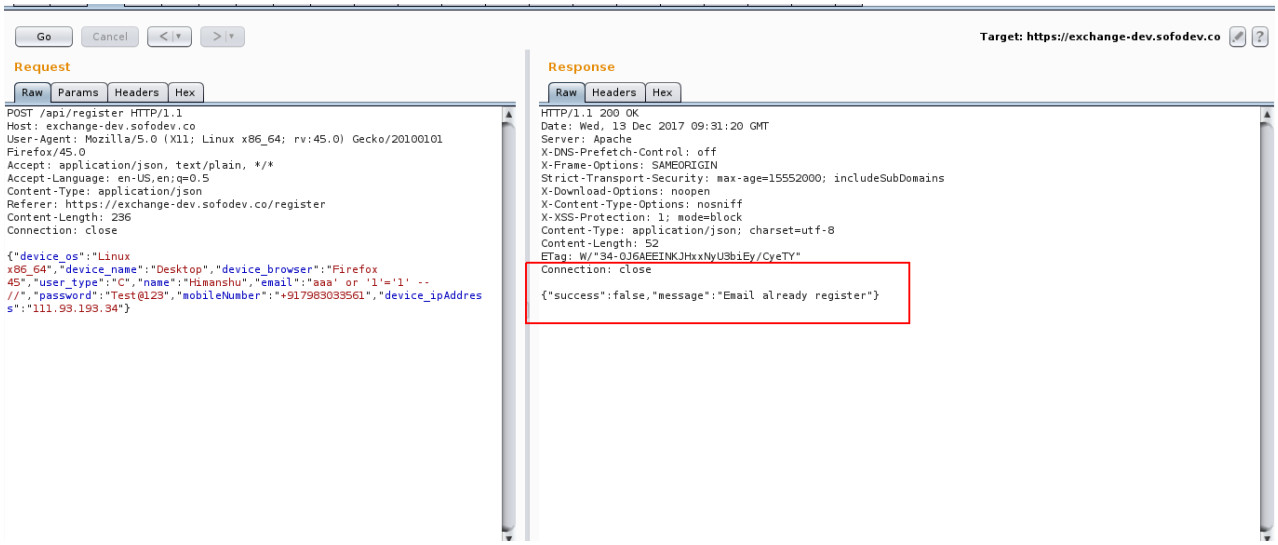
6.1.3 In Customer Login	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
URL	https://exchange-dev.sofodev.co/api/login

Screenshot 1: Response "Password not match" revealing we have a valid email:



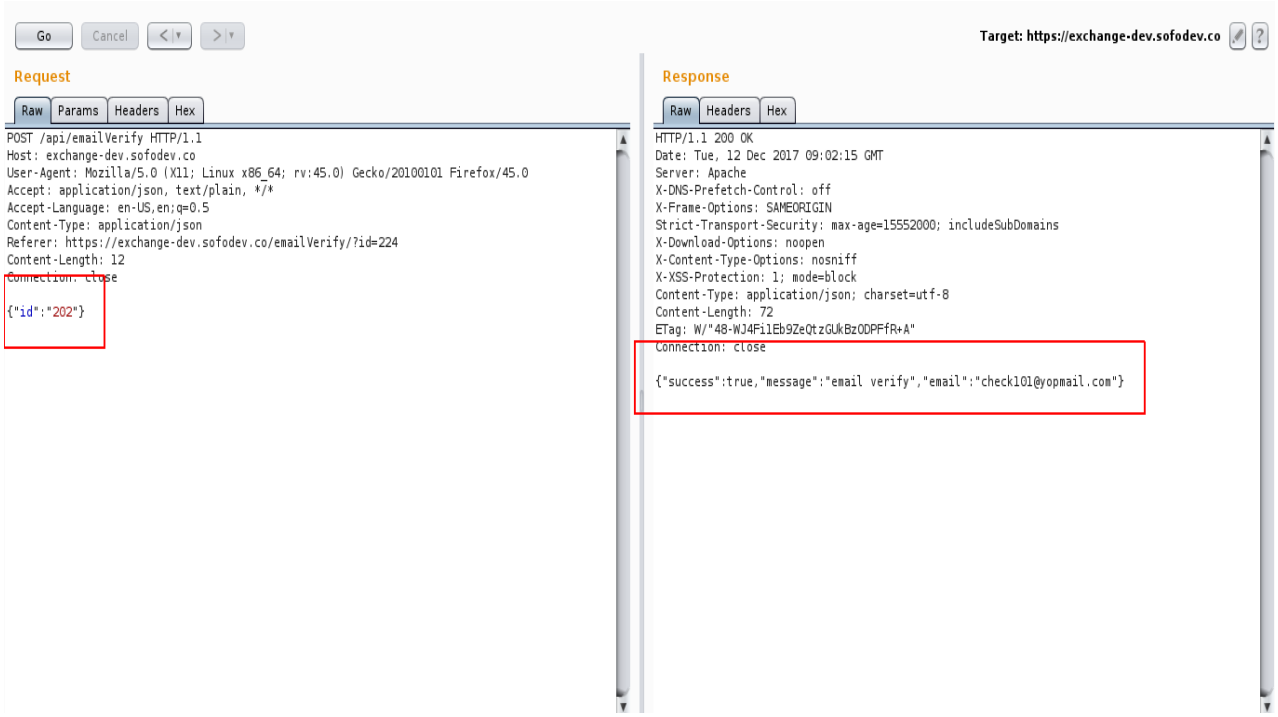
6.1.4 In Registration	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
URL	https://exchange-dev.sofodev.co/api/register

Screenshot 1: Response "Email already register" revealing we have a valid email:



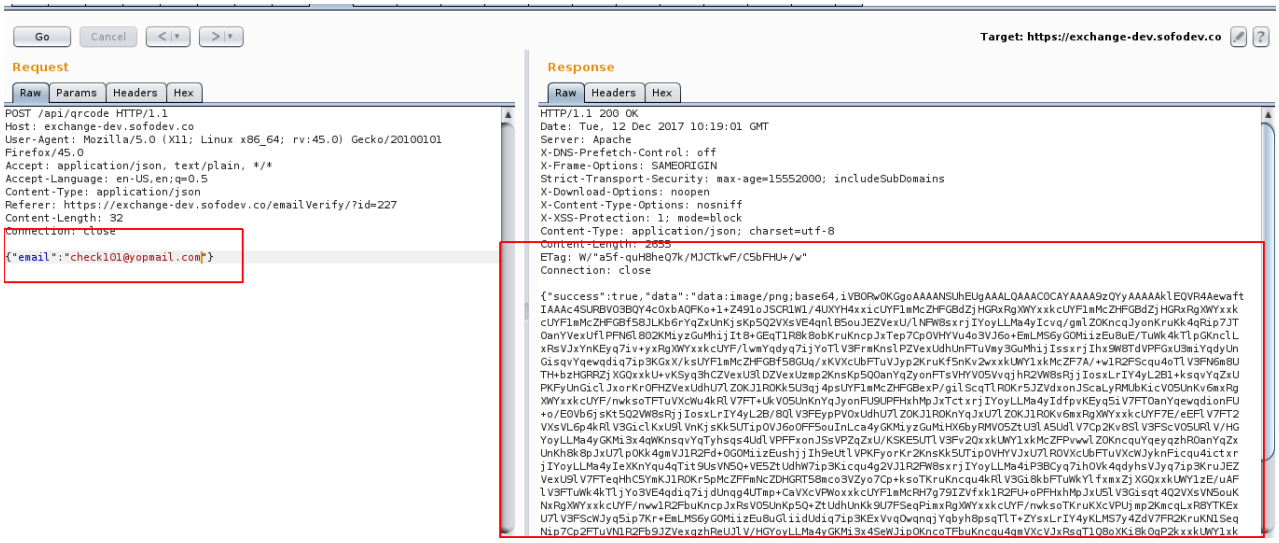
6.1.5 In Email Verify	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
URL	https://exchange-dev.sofodev.co/api/emailVerify

Screenshot 1: Response with email of a user:



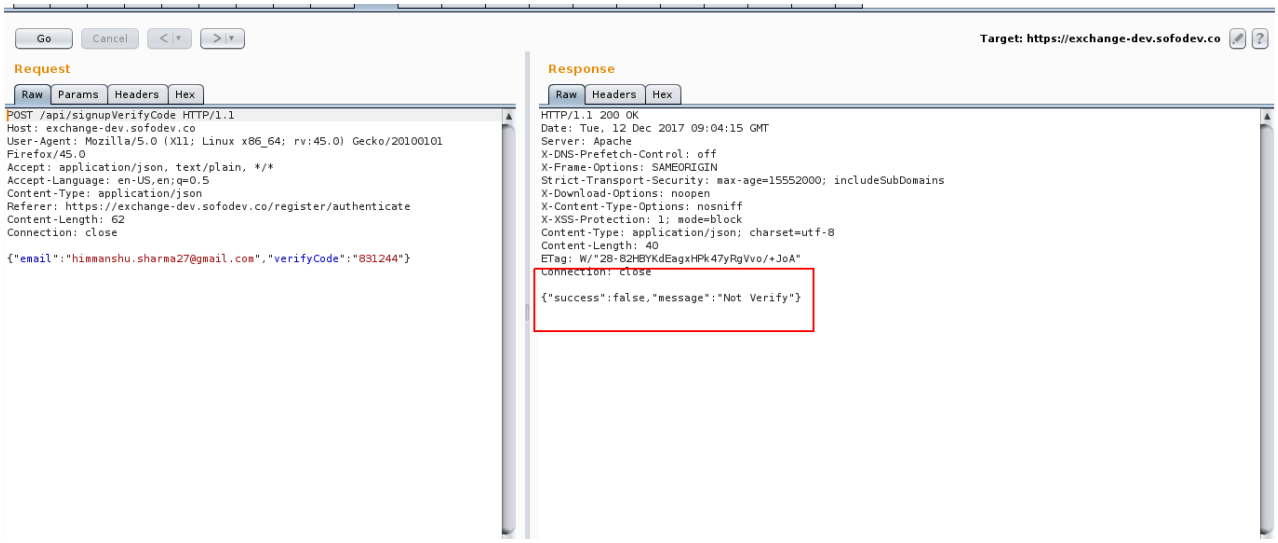
6.1.6 In 2FA Generation	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
URL	https://exchange-dev.sofodev.co/api/qrcode

Screenshot 1: Response with 2FA details revealing we have valid email:

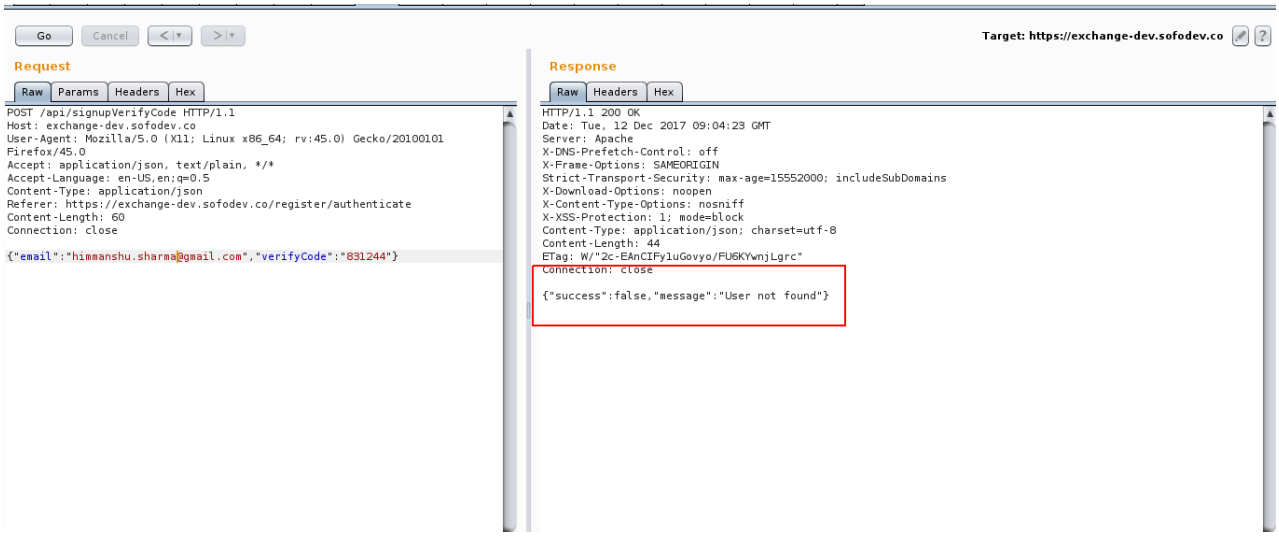


6.1.7 In Signup 2FA Code	
Vulnerability Name	User Enumeration
Severity	**MEDIUM**
URL	https://exchange-dev.sofodev.co/api/signupVerifyCode

Screenshot 1: Response revealing when we have valid email:

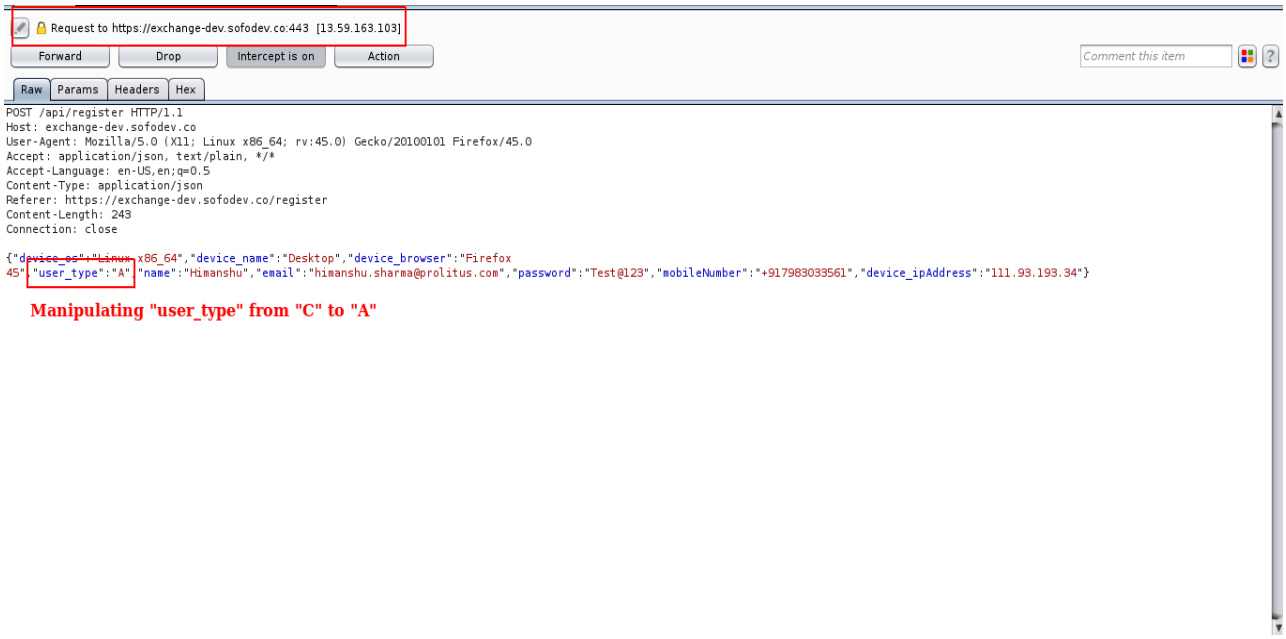


Screenshot 2: Response revealing when we are not having a valid email:

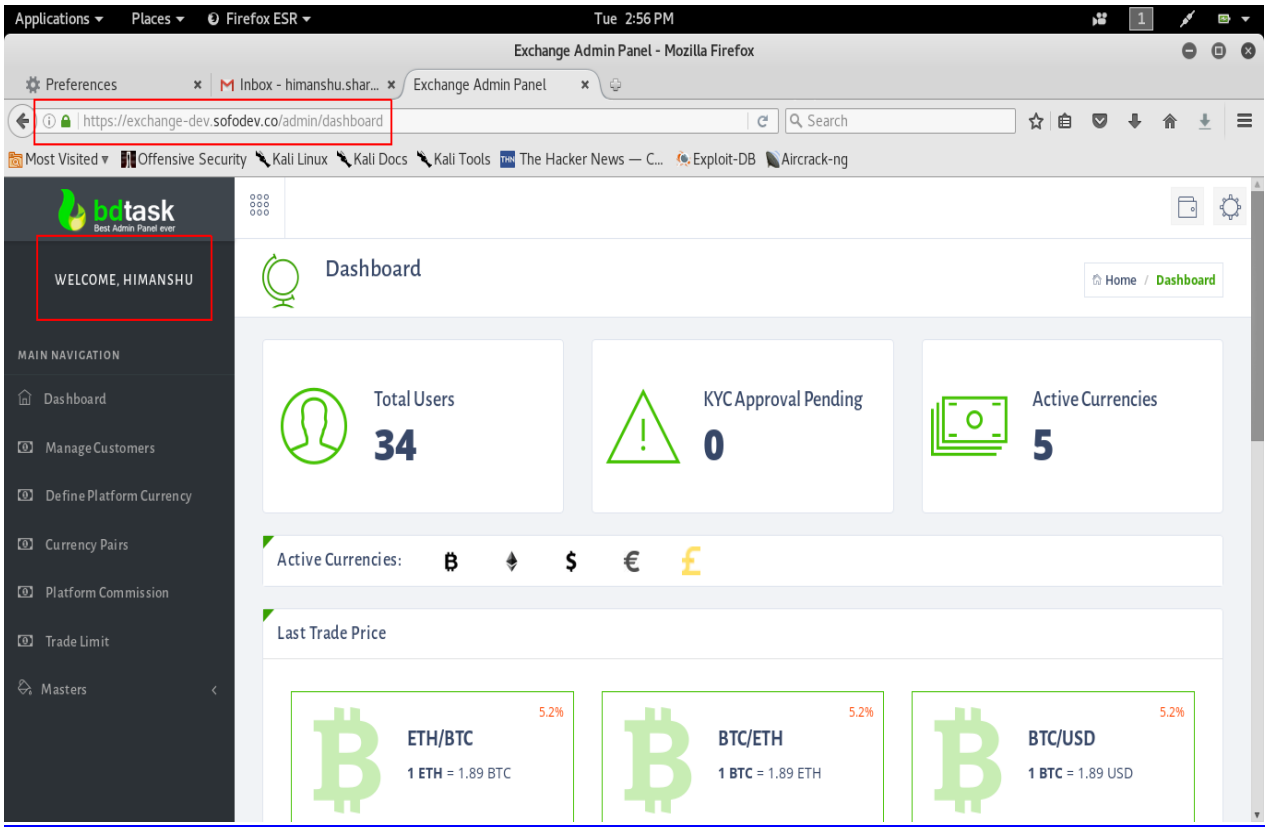


6.2 Application Logic Bypass (Register as Admin)	
Vulnerability Name	Application Logic Bypass (Register as Admin)
Severity	**CRITICAL**
Description	The Application fails to validation the parameters of registration at server side. A user sending his details while registration can register himself as an "Administrator" just by manipulating a parameter.
Recommendation	Implement proper checks at Server side according to Application's Logic.
URL	https://exchange-dev.sofodev.co/api/register
Parameter	user_type

Screenshot 1: Manipulating Parameter "user_type":



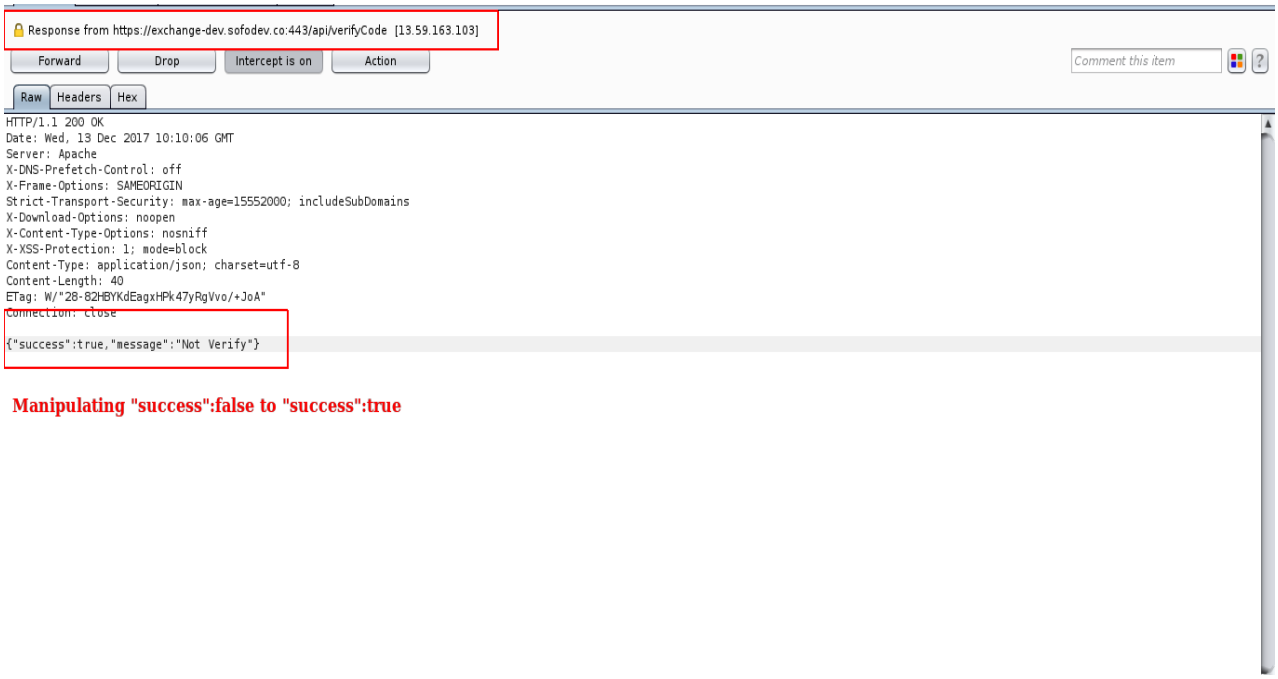
Screenshot 2: Successful in creating Admin "Himanshu":



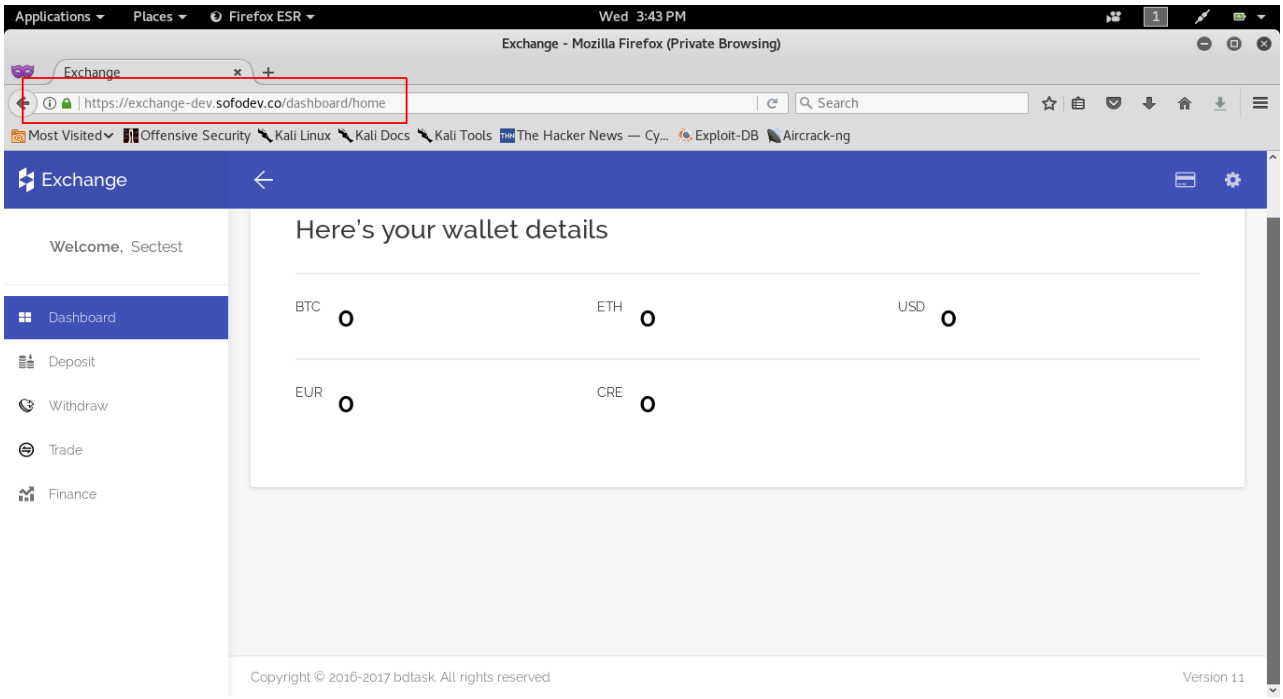
:

6.3 2FA Bypass at Login	
Vulnerability Name	2FA Bypass at Login
Severity	**HIGH**
Description	The Application is not Validating whether the user have Input correct SFA code or not while logging in. Allowing to bypass the 2FA just by editing response.Hence, allowing to Log in without need of 2FA code.
Recommendation	Implement proper checks at Server side according to Application's Logic.
URL	https://exchange-dev.sofodev.co/api/verifyCode
Parameter	N/A

Screenshot 1: Manipulating Response of 2FA code:

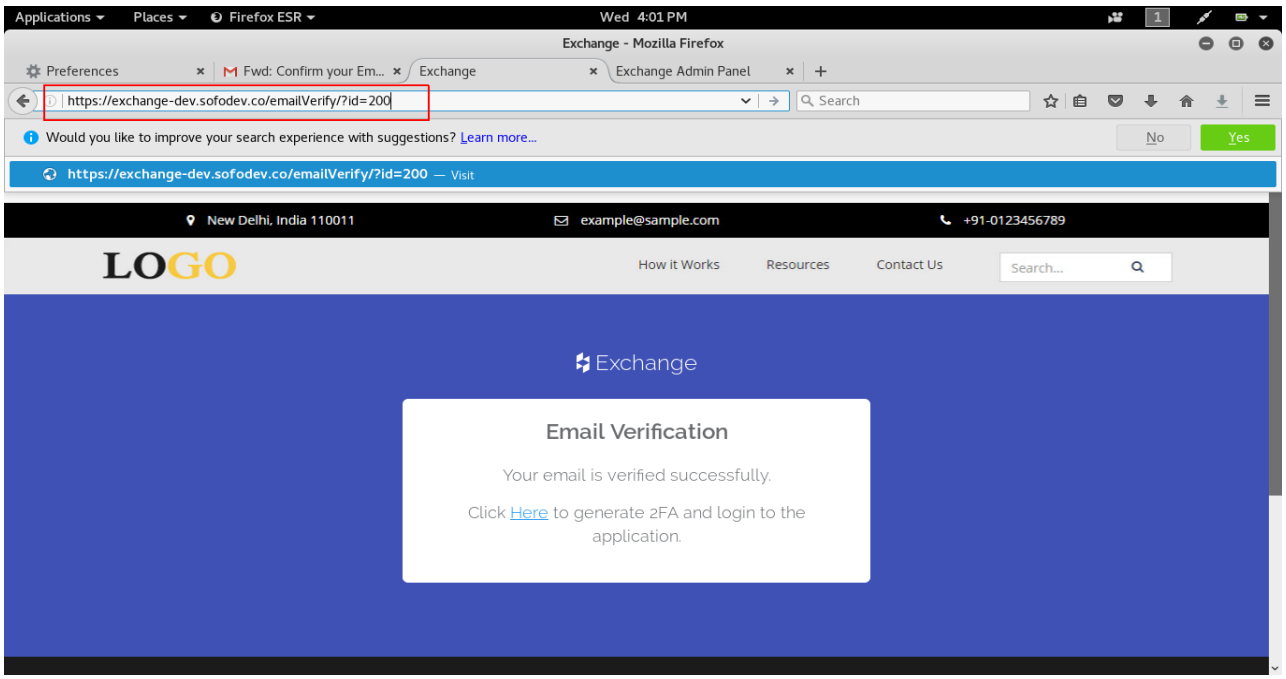


Screenshot 2: Successful Log in:

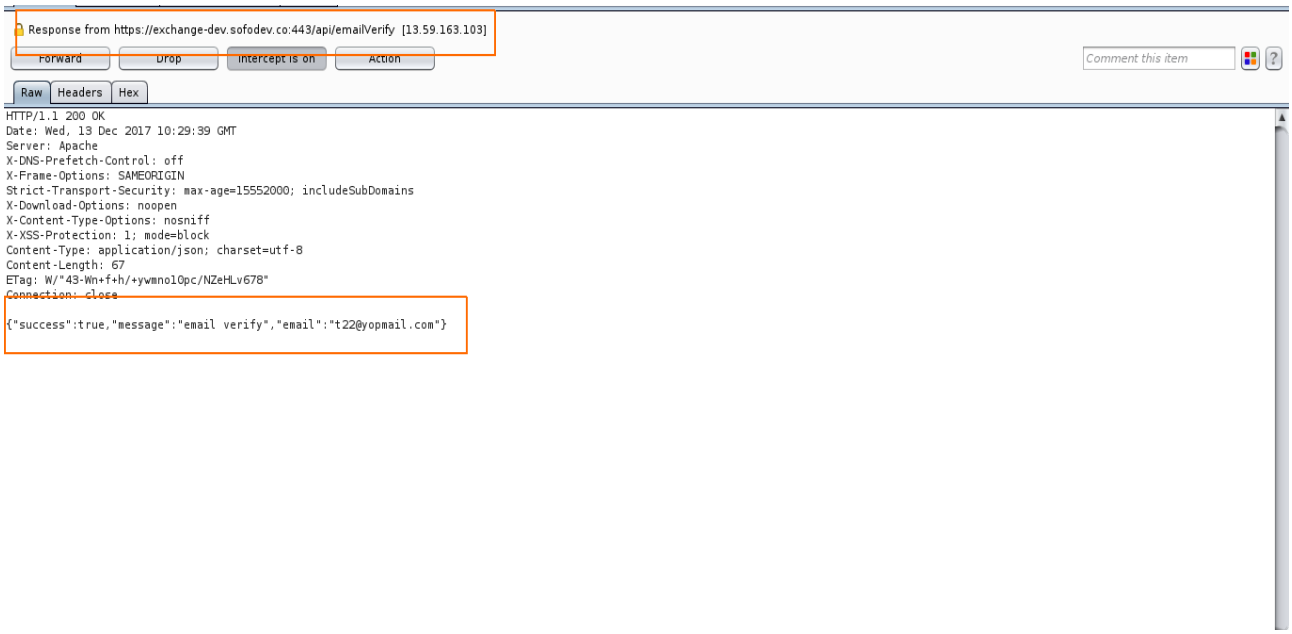


6.4 Insecure Direct Object Reference in Email Verify	
Vulnerability Name	Insecure Direct Object Reference in Email Verify
Severity	**CRITICAL**
Description	The Application is using very weak mechanism for verifying email of an account. A user with just a valid value of "id" parameter can easily access the functionality and can do further actions.
Recommendation	Use strong security measures for verifying emails like random tokens which cannot be manipulated.
URL	https://exchange-dev.sofodev.co/#emailVerify/?id=201
Parameter	id

Screenshot 1: Manipulating "id" to parameter:

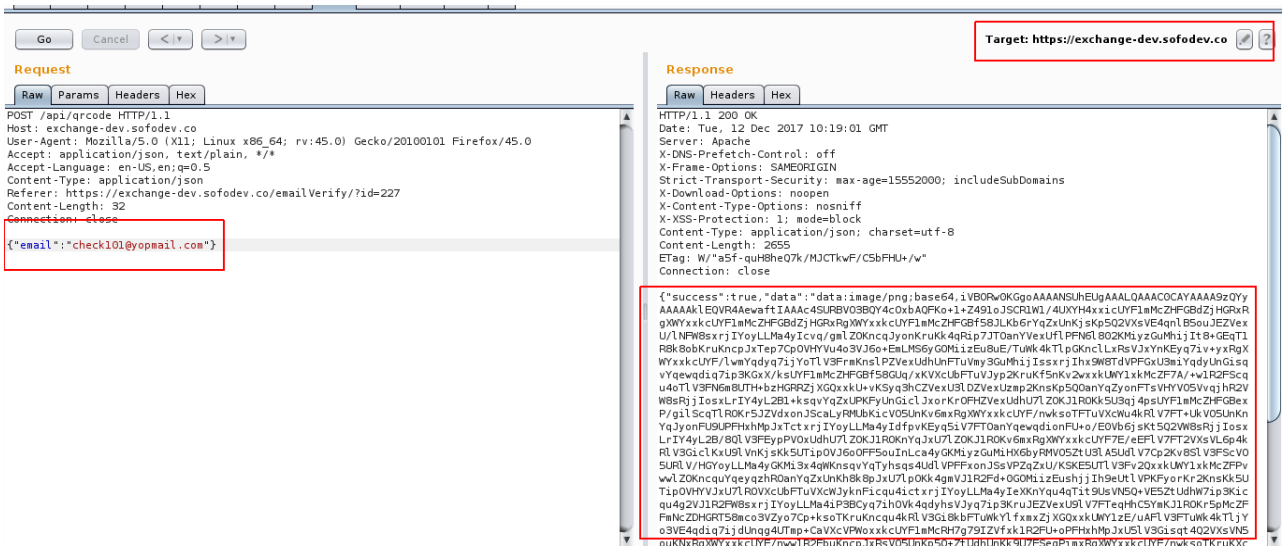


Screenshot 2: User details coming in response confirming of success:



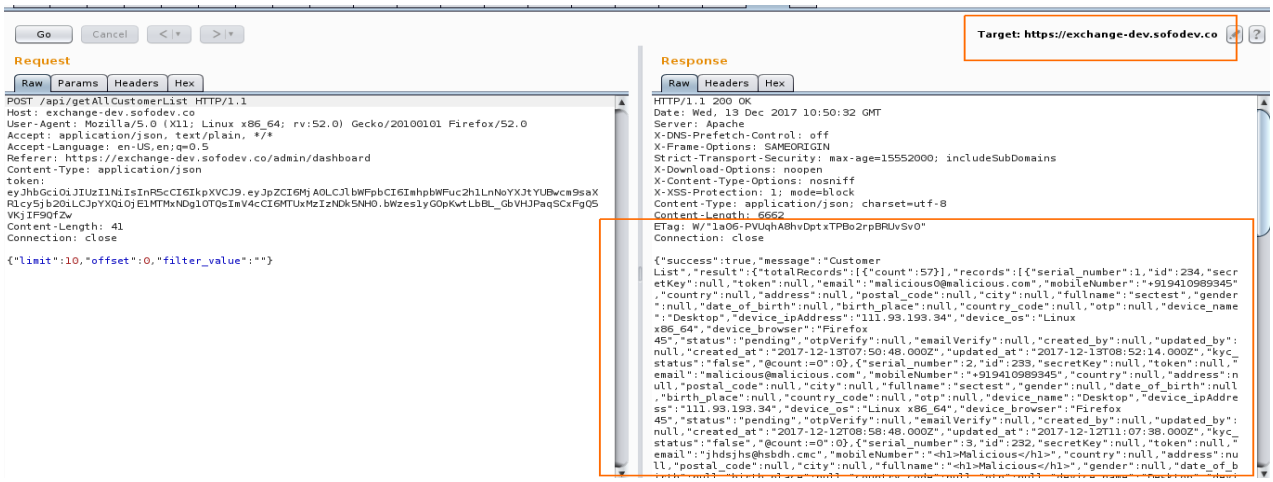
6.5 Insecure Direct Object Reference in 2FA Generation	
Vulnerability Name	Insecure Direct Object Reference in 2FA Generation
Severity	**CRITICAL**
Description	The Application is using very weak mechanism for responding to critical information. A user with just a valid value of "email" parameter can easily access the functionality and can do further actions.
Recommendation	Use strong security measures for verifying requests like random tokens which cannot be manipulated.
URL	https://exchange-dev.sofodev.co/api/qrcode
Parameter	email

Screenshot 1: Sending request with victim's email and got desired data:



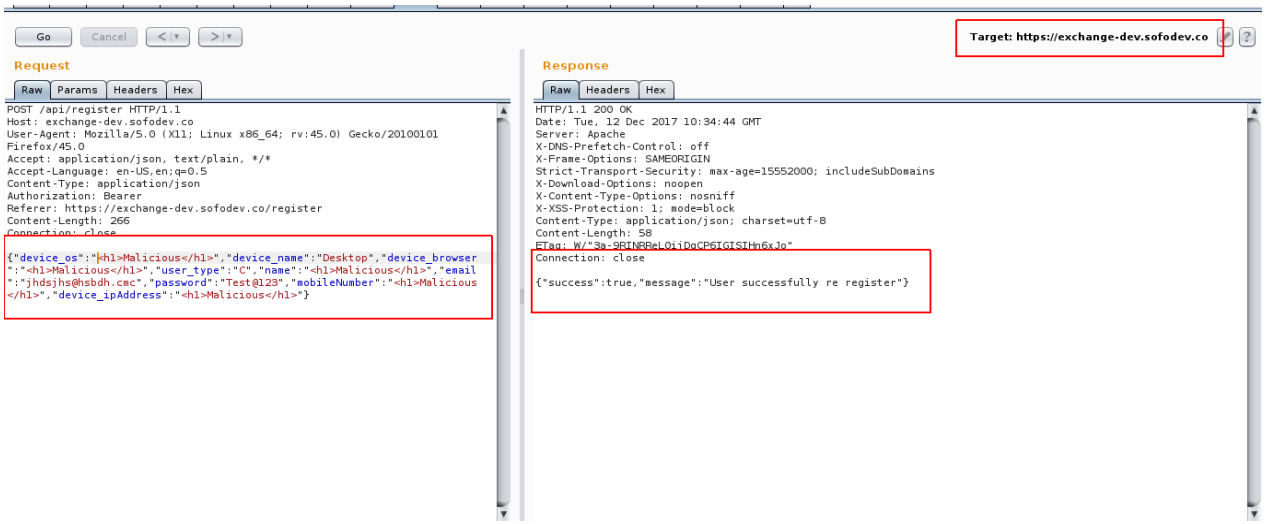
6.6 Improper Session Management	
Vulnerability Name	Improper Session Management
Severity	**MEDIUM**
Description	While testing we have observed that the Application is not implementing a valid session management scheme. After Logging out, a user still can access the information and functionality.
Recommendation	Discard the session token immediately after user logs out.
URL	https://exchange-dev.sofodev.co
Parameter	N/A

Screenshot 1: Admin has logged out still information can be fetched with previous token:

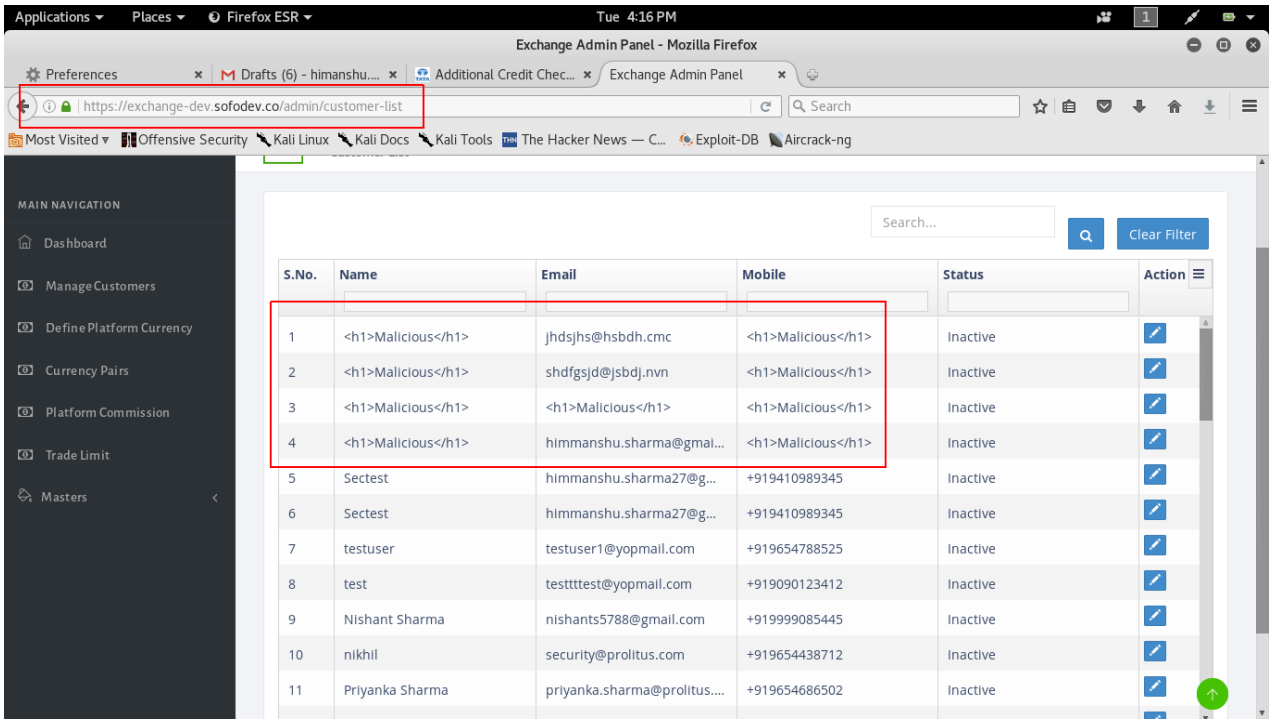


6.7 Improper Server Side Validation	
Vulnerability Name	Improper Server Side Validation
Severity	**MEDIUM**
Description	When software does not validate input properly, an attacker is able to craft the input in a form that is not expected by the rest of the application. This will lead to parts of the system receiving unintended input, which may result in altered control flow, arbitrary control of a resource, or arbitrary code execution.
Recommendation	Enforce Proper Server Side Sanitization and Validation of every user-controlled input.
URL	https://exchange-dev.sofodev.co
Parameter	N/A

Screenshot 1: Malicious Input Entry while Registration:



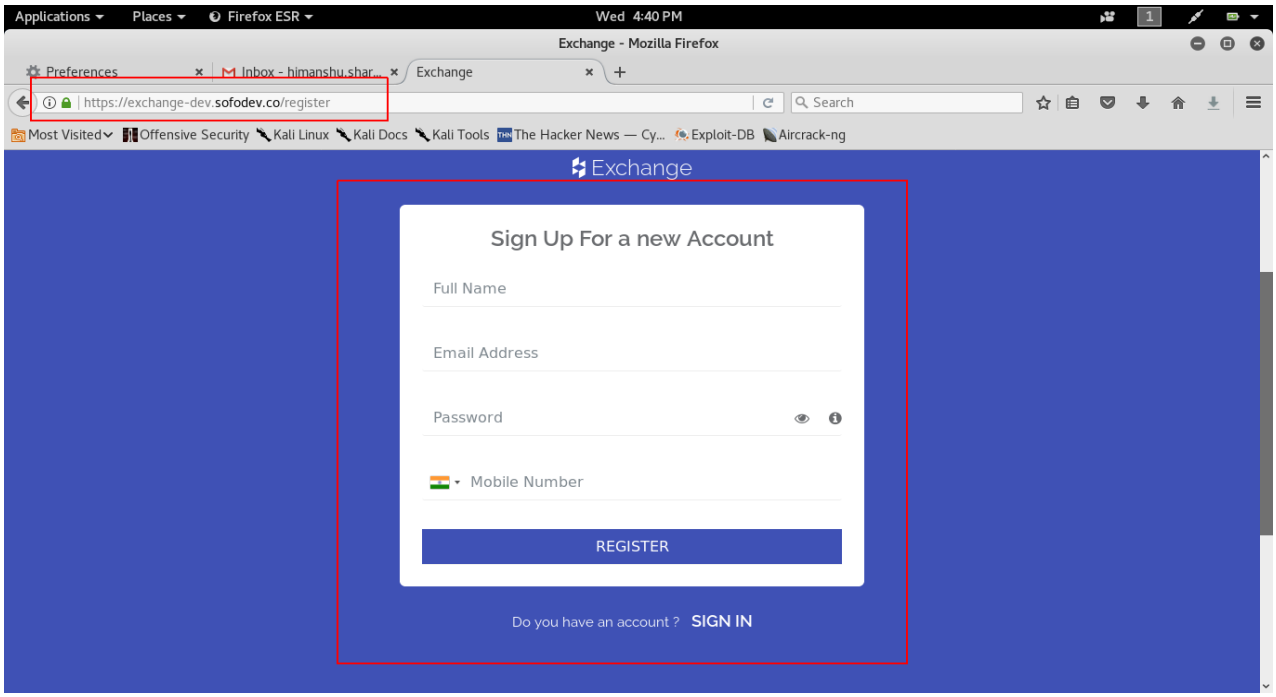
Screenshot 2: Invalid Input Accepted, see in users list:



The Vulnerability is almost throughout the Application as there is no safeguard!

6.8 No Captcha on Registration	
Vulnerability Name	No Captcha on Registration
Severity	**LOW**
Description	The Application has no Captcha Implementation, as a result a user with malicious intentions can fill out application resources by sending numerous registration request which can lead to DoS and exhausting the Application Server's resources.
Recommendation	Implement proper Captcha Implementation
URL	https://exchange-dev.sofodev.co/api/register
Parameter	N/A

Screenshot 1: No Captcha on Registration:



6.9 Multiple Time Registration of Customer	
Vulnerability Name	Multiple Time Registration of Customer
Severity	**LOW**
Description	A user can register himself again and again with using same credential which is being used by application for identification. This could allow an attacker to re register a user with his desired password, email e.t.c
Recommendation	Discard the re registration with same identification parameter such as email, phone number, etc
URL	https://exchange-dev.sofodev.co/api/register
Parameter	N/A

Screenshot 1: See response i,e only "register" in first time:

Go | Cancel | Target: https://exchange-dev.sofodev.co

Request

Raw | Params | Headers | Hex

```
POST /api/register HTTP/1.1
Host: exchange-dev.sofodev.co
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Referer: https://exchange-dev.sofodev.co/register
Content-Length: 257
Connection: close

{"device_os":"Linux
x86_64","token":"abcd1234","device_name":"Desktop","device_browser":"Firefox
45","user_type":"C","name":"sectest","email":"malicious0@malicious.com","password":"Test@12
3","mobileNumber":"+919410989345","device_ipAddress":"111.93.193.34"}
```

Response

Raw | Headers | Hex

```
HTTP/1.1 200 OK
Date: Tue, 12 Dec 2017 11:26:37 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 55
ETag: W/"37-rs1a2EZCB7orlG1mWSuuAL4z9X8"
Connection: close

{"success":true,"message":"User successfully register"}
```

First Registration

Screenshot 2: Now, second time "re register":

Go | Cancel | Target: https://exchange-dev.sofodev.co

Request

Raw | Params | Headers | Hex

```
POST /api/register HTTP/1.1
Host: exchange-dev.sofodev.co
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Referer: https://exchange-dev.sofodev.co/register
Content-Length: 257
Connection: close

{"device_os":"Linux
x86_64","token":"abcd1234","device_name":"Desktop","device_browser":"Firefox
45","user_type":"C","name":"sectest","email":"malicious0@malicious.com","password":"Test@12
3","mobileNumber":"+919410989345","device_ipAddress":"111.93.193.34"}
```

Response

Raw | Headers | Hex

```
HTTP/1.1 200 OK
Date: Tue, 12 Dec 2017 11:54:42 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 58
ETag: W/"3a-9RINRReLOijDgCP6IGISIHn6xJo"
Connection: close

{"success":true,"message":"User successfully re register"}
```

Second Registration, see "User successfully re register" in Response

6.10 Error Message Display	
Vulnerability Name	Error Message Display
Severity	**HIGH**
Description	Improper handling of errors can introduce a variety of security problems for a web site.The most common problem is when detailed internal error messages such as stack traces, database dumps, and error codes are displayed to the user (hacker). These messages reveal implementation details that should never be revealed. Such details can provide hackers important clues on potential flaws in the site.
Recommendation	➢ Set Error Messages Options to OFF. ➢ Try to Handle every Possible Exception. ➢ Set a Customized Error message.
URL	https://exchange-dev.sofodev.co
Parameter	N/A

Screenshot 1: Providing invalid malicious input as a result error generation:

Target: https://exchange-dev.sofodev.co

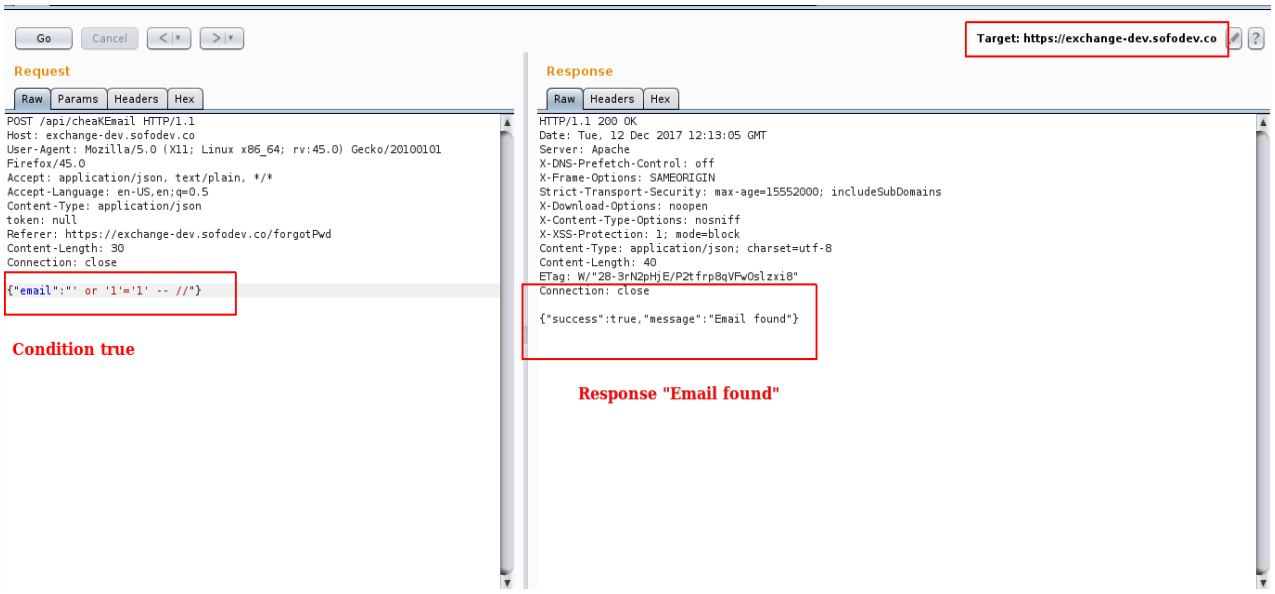
Response

Raw | Headers | Hex

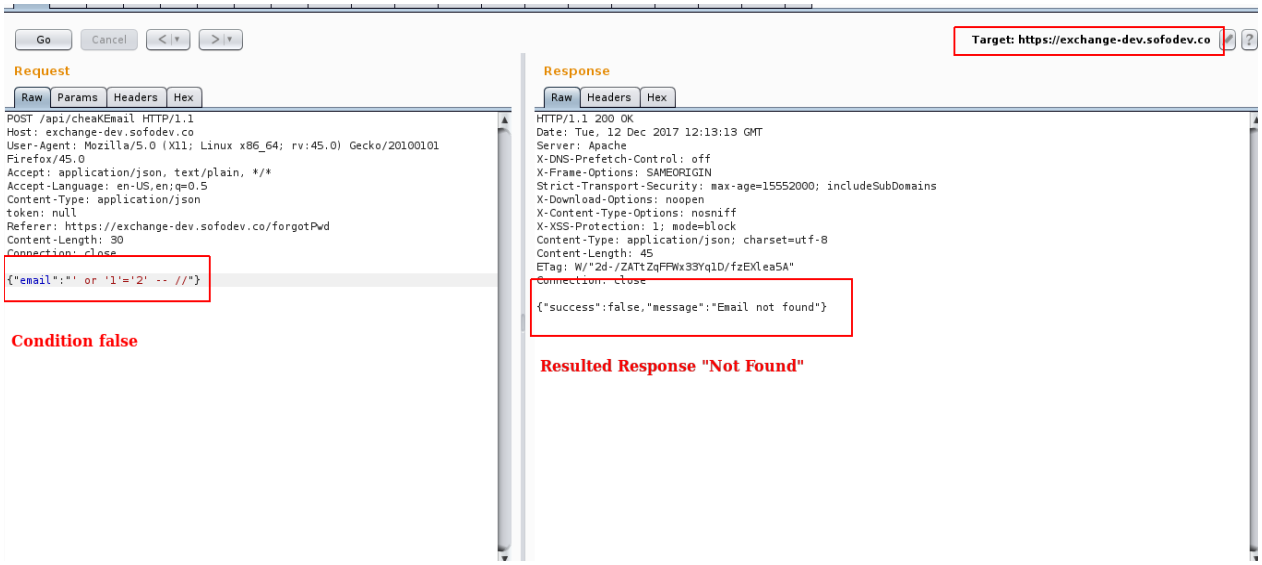
```
HTTP/1.1 200 OK
Date: Tue, 12 Dec 2017 12:28:04 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 332
ETag: W/"14c-6TYb8+NdrJtsY4Ag2SeoPAqk1v0"
Connection: close

{"success":false,"message":"error","error":{"code":"ER_WRONG_NUMBER_OF_COLUMNS_IN_SELECT",
"errno":1222,"sqlMessage":"The used SELECT statements have a different number of
columns","sqlState":"21000","index":0,"sql":"SELECT * FROM customer WHERE email
='himmanshu.sharma27@gmail.com' union select substring(version(),1,1)=4 -- //'"}}
```

6.11 SQL Injection	
Vulnerability Name	SQL Injection
Severity	**HIGH**
Description	SQL injection vulnerabilities arise when user-controllable data is incorporated into database SQL queries in an unsafe manner. An attacker can supply crafted input to break out of the data context in which their input appears and interfere with the structure of the surrounding query. A wide range of damaging attacks can often be delivered via SQL injection, including reading or modifying critical application data, interfering with application logic, escalating privileges within the database and taking control of the database server.
Recommendation	Some of Primary Defenses are: ● Use of Prepared Statements (with Parameterized Queries) ● Use of Stored Procedures ● White List Input Validation ● Escaping All User Supplied Input
URL	https://exchange-dev.sofodev.co
Parameter	N/A

Screenshot 1: Sending a true condition of SQL:



Screenshot 2: Now, sending false condition see response:



Screenshot 3: Sending two emails and with "or" condition i.e true:

Go | Cancel | < | >

Target: https://exchange-dev.sofodev.co

Request

Raw | Params | Headers | Hex

```
POST /api/adminLogin HTTP/1.1
Host: exchange-dev.sofodev.co
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101
Firefox/45.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Referer: https://exchange-dev.sofodev.co/admin
Content-Length: 103
Connection: close

{"email":"himmanshu.sharma27@gmail.com' or
'himanshu.sharma@prolitus.com' -- //","password":"Test@123"}
```

"email" having one admin and one cust. email with "or"

Response

Raw | Headers | Hex

```
HTTP/1.1 200 OK
Date: Wed, 13 Dec 2017 09:33:12 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 317
ETag: W/"13d-oIMLdg6w/wSH8sZ7Y7zfHCKiXM4"
Connection: close

{"success":true,"message":"login
success","token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MjA3LCJlbWFpbCI6ImhpbWlhbnNodS5zaGFybWEyNOBn
bWFpbC5jb20nIG9yICdoaW1hbnNodS5zaGFybWFAcHJvbGl0dXMuY29tJyAtLSAvLyIsImlhdCI6MTUxMzElNzU5MiwiZXhwIjoxNTEzMjQ
zOTkyfQ.j9lcboq4CkllFUXyeF8bfDwDBATkt92AgJDTS1kgqiw","name":"Sectest"}
```

Due to "or" Condition true, so Successful Login

Screenshot 4: Now, Sending two emails and with "and" condition i.e false:

Go | Cancel | < | >

Target: https://exchange-dev.sofodev.co

Request

Raw | Params | Headers | Hex

```
POST /api/adminLogin HTTP/1.1
Host: exchange-dev.sofodev.co
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101
Firefox/45.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Referer: https://exchange-dev.sofodev.co/admin
Content-Length: 104
Connection: close

{"email":"himmanshu.sharma27@gmail.com' and
'himanshu.sharma@prolitus.com' -- //","password":"Test@123"}
```

Now with "and" i.e, false condition

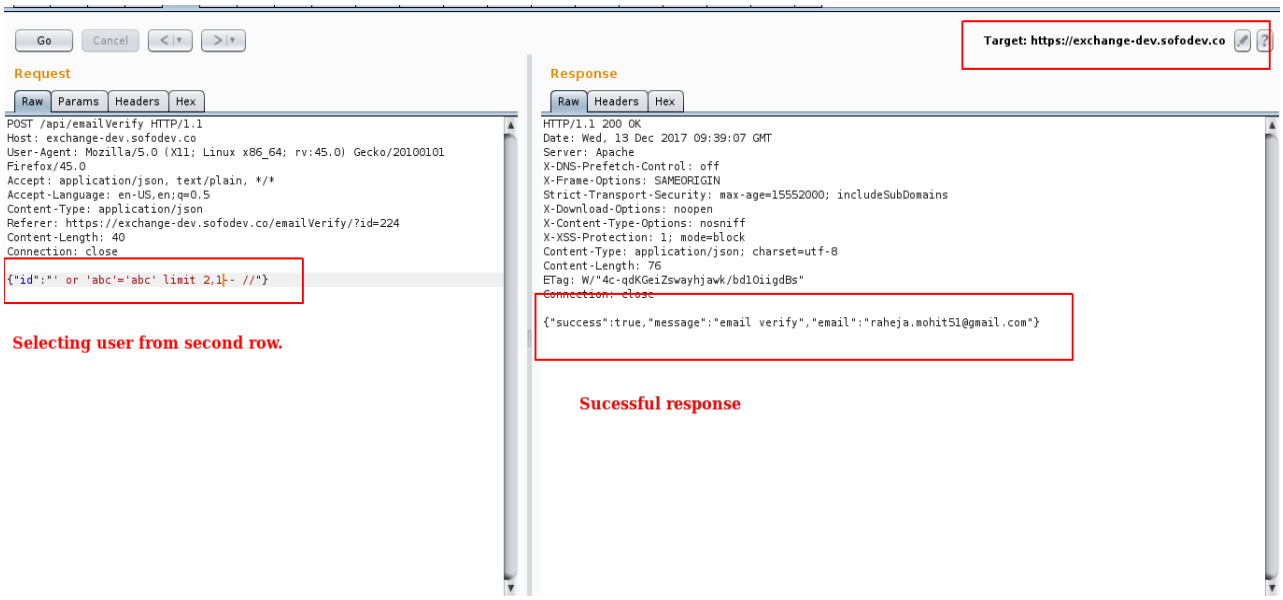
Response

Raw | Headers | Hex

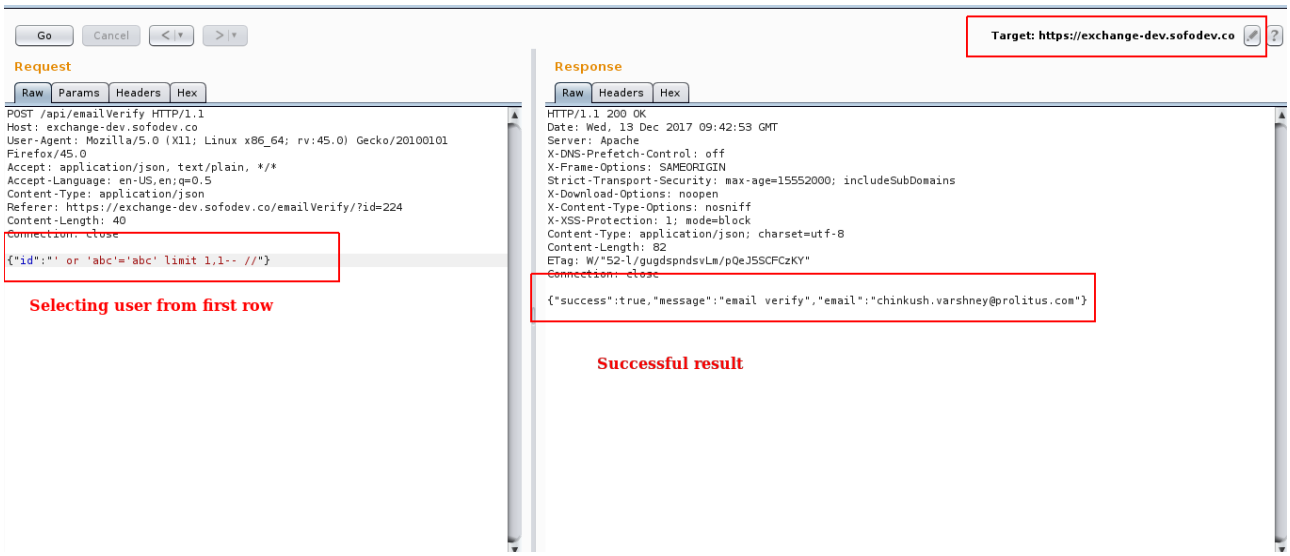
```
HTTP/1.1 200 OK
Date: Wed, 13 Dec 2017 09:33:30 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 47
ETag: W/"2f-vOu8/FGgxgMvzia8BNchdRAu9lU"
Connection: close

{"success":false,"message":"User not register"}
```

No Authentication

Screenshot 5: Sending query with limit to second row in selection:



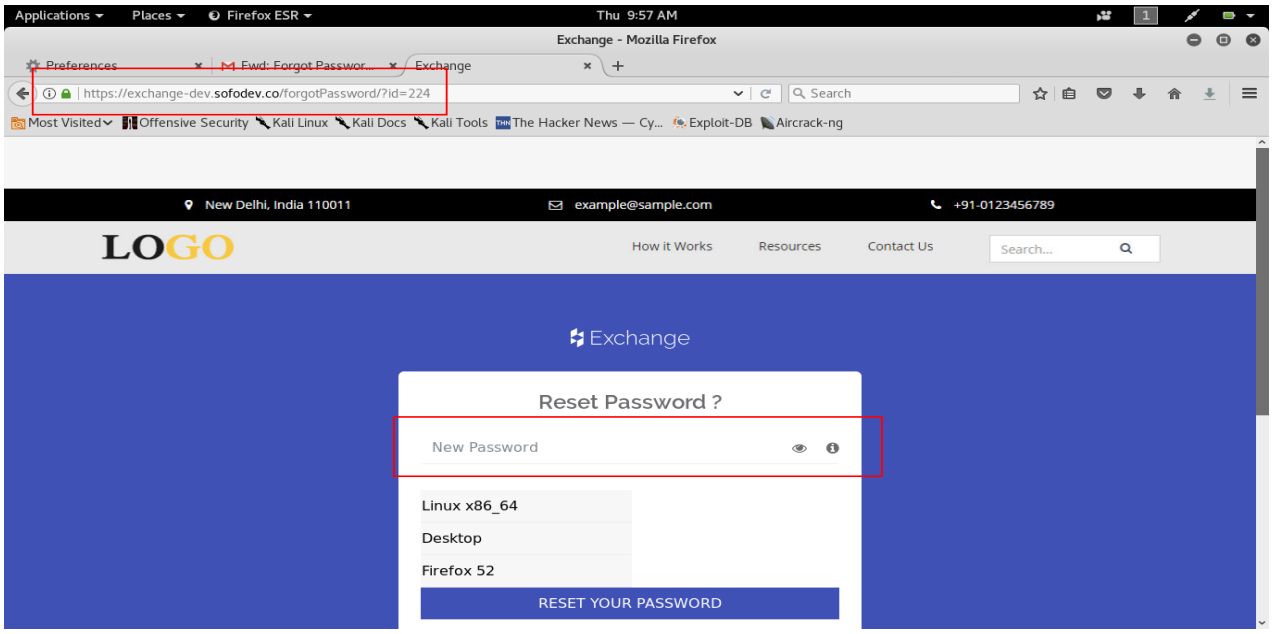
Screenshot 5: Sending query with limit to 1st row in selection:



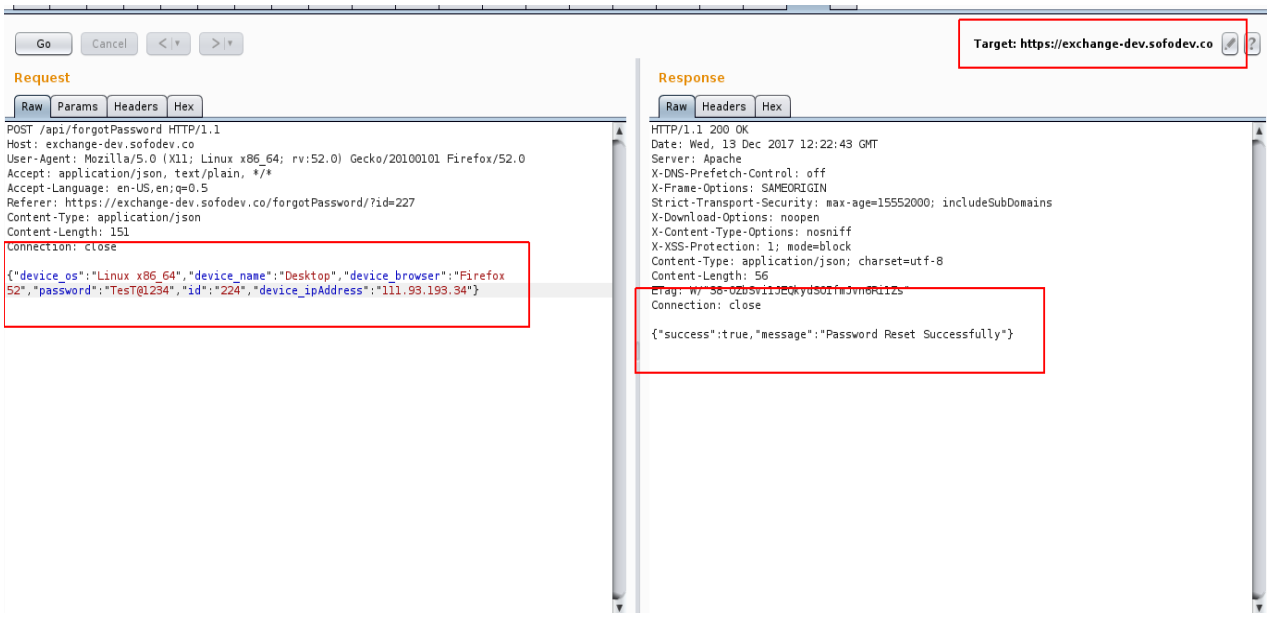
The SQL Injection Vulnerability is almost throughout the Application as there is no safeguard!

6.12 Insecure Direct Object Reference in Password Reset	
Vulnerability Name	Insecure Direct Object Reference in Password Reset
Severity	**CRITICAL**
Description	The Application is using very weak mechanism for resetting the password of an account. A user with just a valid value of "id" parameter can easily access the functionality and can change the password of any user.
Recommendation	Use strong security measures for verifying password reset requests like random tokens which cannot be manipulated.
URL	https://exchange-dev.sofodev.co/#/forgotPassword/?id=227
Parameter	id

Screenshot 1: Manipulating "id" to 224 i.e desired user id:



Screenshot 2: Reset Password request with our new password word, and success in it:



6.13 Improper Password Policy Implementation	
Vulnerability Name	Improper Password Policy Implementation
Severity	**HIGH**
Description	Applications fails to validate password at Server side. A user can set empty password for his account making the account vulnerable to be hijacked easily.
Recommendation	Implement password policy at both end i.e at Server side checks must also be applied properly.
URL	https://exchange-dev.sofodev.co
Parameter	N/A

Screenshot 1: Setting empty password:

Request to https://exchange-dev.sofodev.co:443 [13.59.163.103]

Forward | Drop | Intercept is on | Action | Comment this item

Raw | Params | Headers | Hex

```
POST /api/changePassword HTTP/1.1
Host: exchange-dev.sofodev.co
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Referer: https://exchange-dev.sofodev.co/dashboard/ChangePassword
Content-Type: application/json
token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MjA3LCJlbWFpbCI6ImhpbWlhbnNodS5zaGFybWEyNOBnbWFpbC5jb20iLCJkZXZpY2VfbmFtZSI6IkRlc2t0b3AiLCJpYXQiOjE1MTMyNDg4ODAsImV4cCI6MTUxMzMzNTI4MH0.3p6Ma8ZtT
191YMulmTSGjJpOJcH2E8CIaKG6gzc0jG4
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MjA3LCJlbWFpbCI6ImhpbWlhbnNodS5zaGFybWEyNOBnbWFpbC5jb20iLCJkZXZpY2VfbmFtZSI6IkRlc2t0b3AiLCJpYXQiOjE1MTMyNDg4ODAsImV4cCI6MTUxMzMzNTI4MH0.3p6Ma8ZtT
191YMulmTSGjJpOJcH2E8CIaKG6gzc0jG4
Content-Length: 167
Connection: close

{"device_os":"Linux x86_64","device_name":"Firefox 52","device_browser":"Desktop","oldPassword":"Test@123","newPassword":"","device_ipAddress":"111.93.193.34"}
```

Screenshot 2: Successful in setting empty password:

Response from https://exchange-dev.sofodev.co:443/api/changePassword [13.59.163.103]

Forward | Drop | Intercept is on | Action | Comment this item

Raw | Headers | Hex

```
HTTP/1.1 200 OK
Date: Thu, 14 Dec 2017 10:55:59 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 44
ETag: W/"2c-Qz+mUcUxlCw1MRK3QGlfXa4FgZI"
Connection: close

{"success":true,"message":"Password Change"}
```

Screenshot 3: Log in with empty password:

Request to https://exchange-dev.sofodev.co:443 [13.59.163.103]

Forward | Drop | Intercept is on | Action | Comment this item

Raw | Params | Headers | Hex

```
POST /api/login HTTP/1.1
Host: exchange-dev.sofodev.co
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Referer: https://exchange-dev.sofodev.co/login
Content-Type: application/json
Authorization: Bearer undefined
Content-Length: 174
Connection: close

{"device_os":"Linux x86_64","device_name":"Desktop","device_browser":"Firefox 52","email":"himmanshu.sharma27@gmail.com","password":"","device_ipAddress":"111.93.193.34"}
```

Screenshot 4: Successful Log in:

Response from https://exchange-dev.sofodev.co:443/api/login [13.59.163.103]

Forward | Drop | Intercept is on | Action | Comment this item

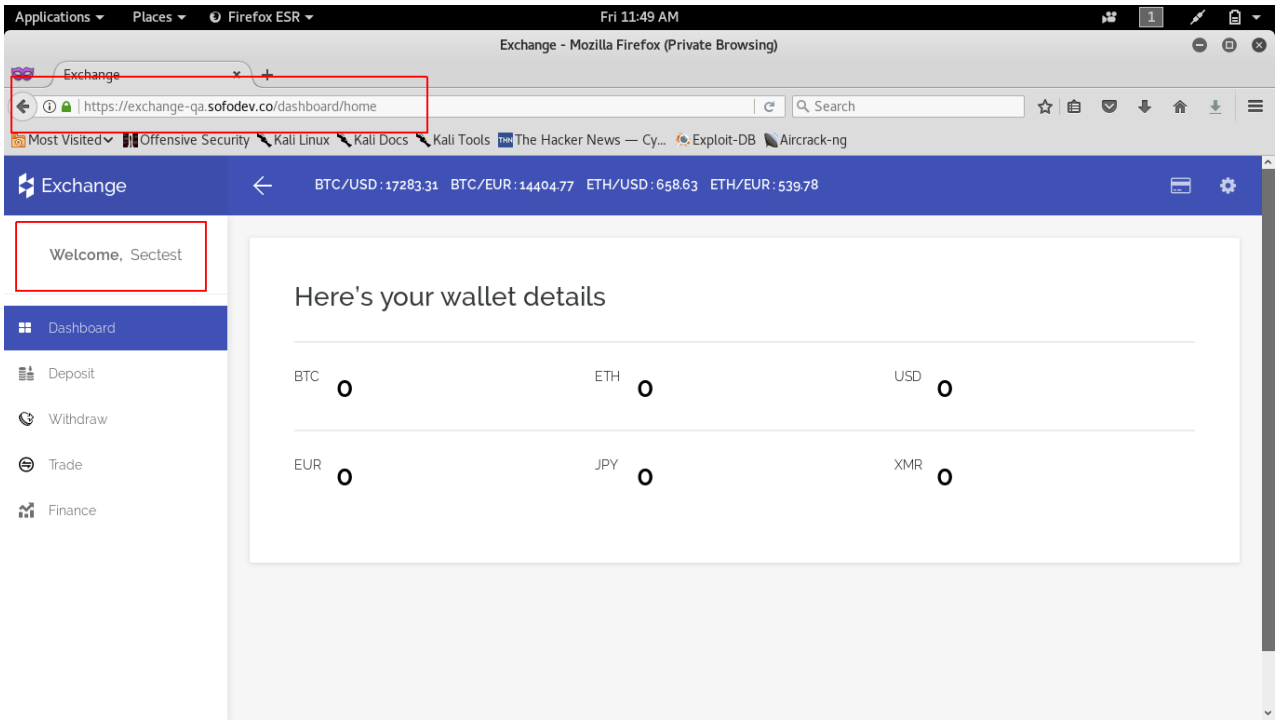
Raw | Headers | Hex

```
HTTP/1.1 200 OK
Date: Thu, 14 Dec 2017 10:50:52 GMT
Server: Apache
X-DNS-Prefetch-Control: off
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopen
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Type: application/json; charset=utf-8
Content-Length: 310
ETag: W/"136-7FVev8GMND4rPYx1/T8VD8KjYVQ"
Connection: close

{"success":true,"message":"login
success","token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MjA3LCJlbWFpbCI6ImhpbWlhbnNodS5zaGFybWEyN0BnbWFpbC5jb20iLCJkZXZpY2VfbmPtZSI6IkRlc2t0b3AiLCJpYXQiOjE1MTMyNDg2NTIsImV4cCI6MTUxMzM
zNTA1MnO.twOy9hN-d2ZBy4PHa7woTtijN6P6rABbxcXei6mzGOk","name":"Sectest","user_type":"C"}
```

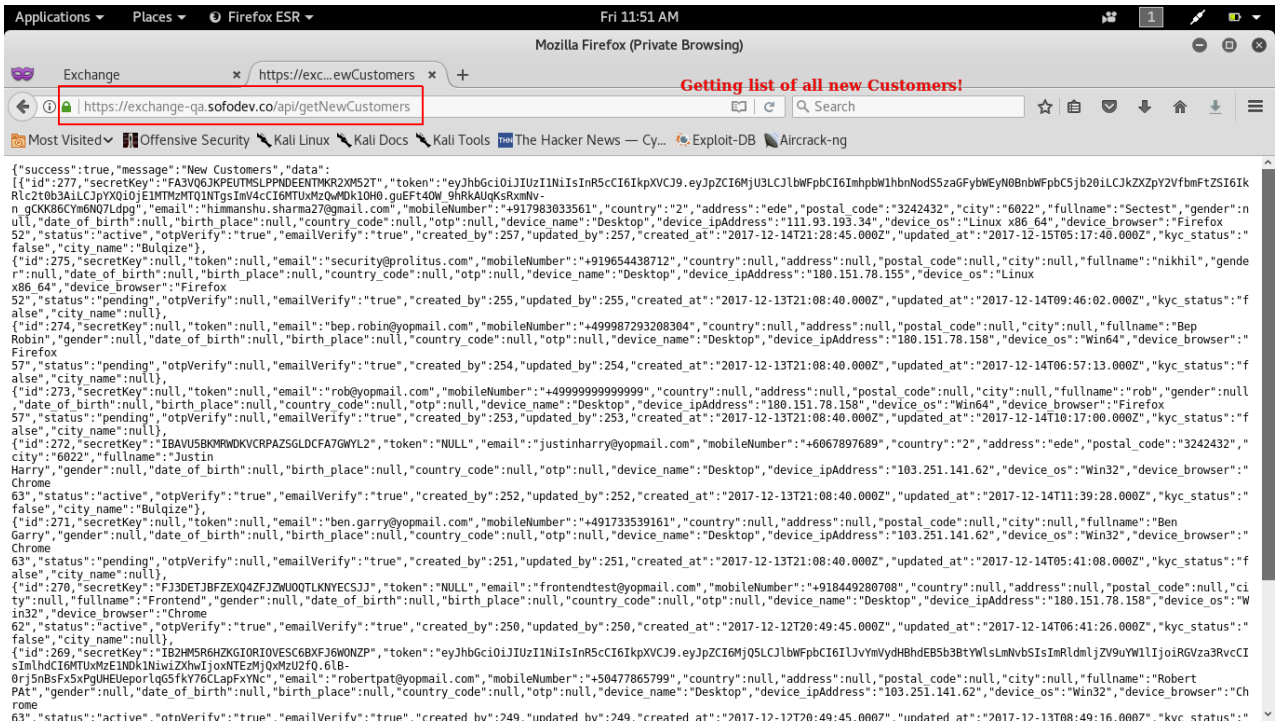
6.14 Privilege Escalation	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
Description	Application fails to apply business logic because it does not validate data on server side. Privilege escalation occurs when a user gets access to more resources or functionality than they are normally allowed, and such elevation or changes should have been prevented by the application.
Recommendation	It is recommended to implement proper business logic controls in the application.

6.14.1 In Get New Customer API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getNewCustomers

Screenshot 1: Logged in as Customer:

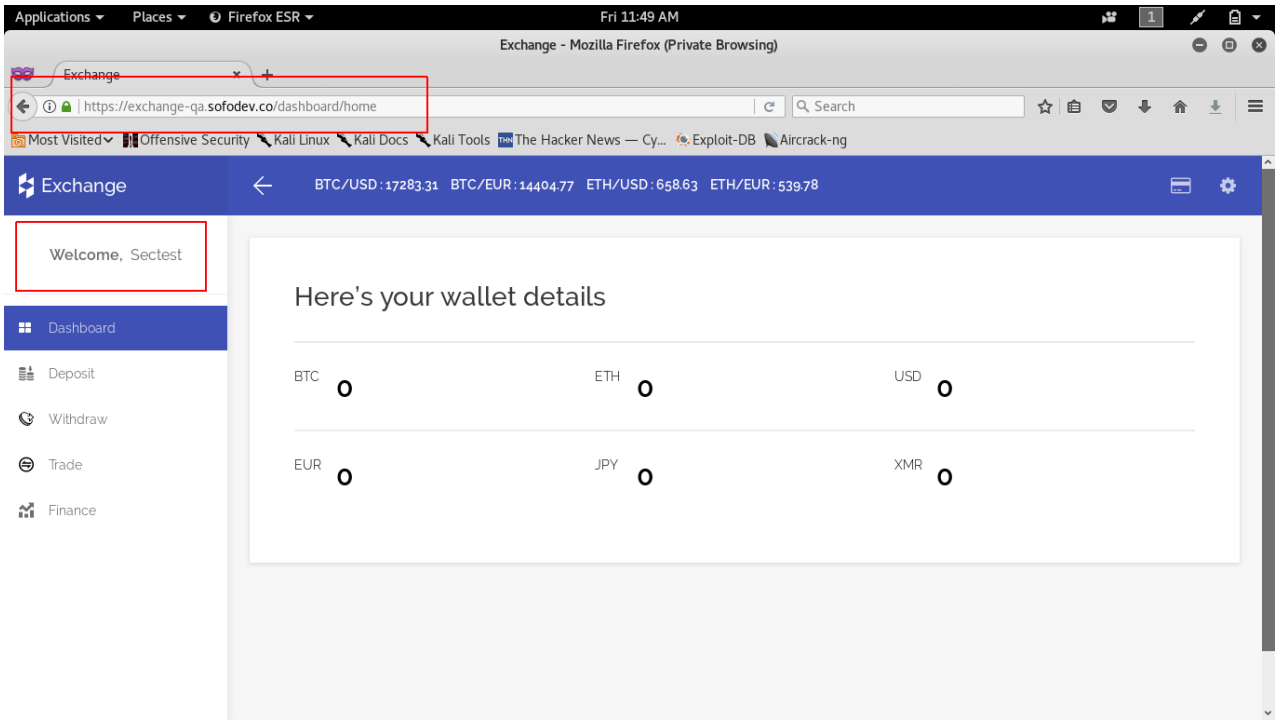


Screenshot 2: Still can access list:

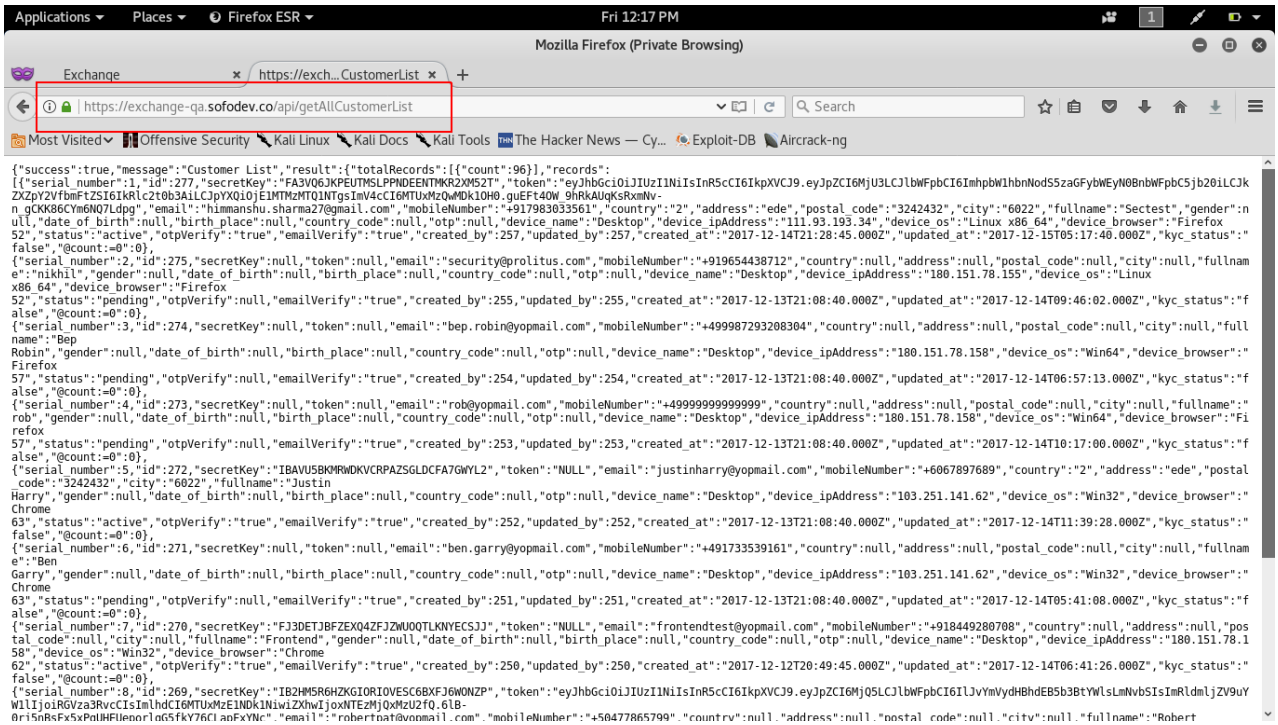


6.14.2 In Get All Customer List API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getAllCustomerList

Screenshot 1: Logged in as Customer:



Screenshot 2: Still can access "All Customer" list:



6.14.3 In Edit Customer Profile API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/editCustomerProfile/277

6.14.4 In Get Currency List API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getCurrencyList

6.14.5 In Edit Currency API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/editCurrency/16

6.14.6 In Update Currency API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/updateCurrency/16

6.14.7 In Add Currency API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/addCurrency

6.14.8 In Upload Currency Icon API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/upload/

6.14.9 In Get All Trade Currency Pairs API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getAllTradeCurrencyPairs

6.14.10 In Update Default Currency Pairs API	
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/updateDefaultPair/177

6.14.11	In Save Currency Pair API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/saveCurrencyPairs

6.14.12	In Delete Currency Pair API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/deleteCurrencyPair/187

6.14.13	In Get Commissions API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getCommissions

6.14.14	In Get Trade Limits API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getTradeLimits

6.14.15	In Get All Country List API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getAllCountryList

6.14.16	In Add Country API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/addCountry

6.14.17	In Edit Country API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/editCountry/257

6.14.18	In Update Country API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/updateCountry/257

6.14.19	In Delete Country API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/deleteCountry/257

6.14.20	In Get All State List API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getAllStateList

6.14.21	In Delete Country API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/deleteCountry/257

6.14.22	In Add State API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/addState

6.14.23	In Edit State API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/editState/4129

6.14.24	In Update State API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/updateState/4129

6.14.25	In Delete State API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/deleteState/4129

6.14.26	In Get All City List API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getAllCityList

6.14.27	In Get State By Id API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getStatesByCountryId/2

6.14.28	In Add City API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/addCity

6.14.29	In Edit City API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/editCity/48322

6.14.30	In Update City API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/updateCity/48322

6.14.31	In Delete City API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/deleteCity/48322

6.14.32	In Get Active Currency Icons API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getActiveCurrencyIcons

6.14.33	In Get Trade Currency Pairs API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getTradeCurrencyPairs

6.14.34	In Get States By Country Id API
Vulnerability Name	Privilege Escalation
Severity	**HIGH**
URL	https://exchange-qa.sofodev.co/api/getStatesByCountryId/2

6.15 Malicious File Upload	
Vulnerability Name	Malicious File Upload
Severity	**CRITICAL**
Description	Many application's business processes allow for the upload of data/information. The application may allow the upload of malicious files that include exploits or shellcode without any restriction leading to very serious flaw.
Recommendation	In Addition with Blacklisting/Whitelisting extensions, the remediation such as Content-Type Filtering should also be Implemented. Also, most important, if possible then file inspection for malicious code should also be implemented at Server-Side.

6.15.1 In Uploading Currency Icon	
Vulnerability Name	Malicious File Upload
Severity	**CRITICAL**
URL	https://exchange-qa.sofodev.co/api/upload/

Screenshot 1: Uploading Malicious file, see below:

Screenshot 2: Success in Upload of Malicious file: