A survey on CSRF (Cross-Site Request Forgery) attack in Web protection

Monica Gattupalli
moga20@student.bth.se
Masters in Computer science

Cross- Site Request Forgery (CSRF):

CSRF attack is seen in web applications when the authorized user logged into the application using search engines or web browser. The intruders attacked the web application without knowing to the authorized user just by sending any malicious message/mail/ request. If the user accepts the mail / message/ request at that point the CSRF attack is successfully activated. Now, the intruder can perform any malicious activates[1] [2].

In general, any web application request or link contains the information of the sender in this case it is difficult to identify the fake request or links. This vulnerability is used as an advantage by the intruders. Below figure shows gives an overview about CSRF attack.

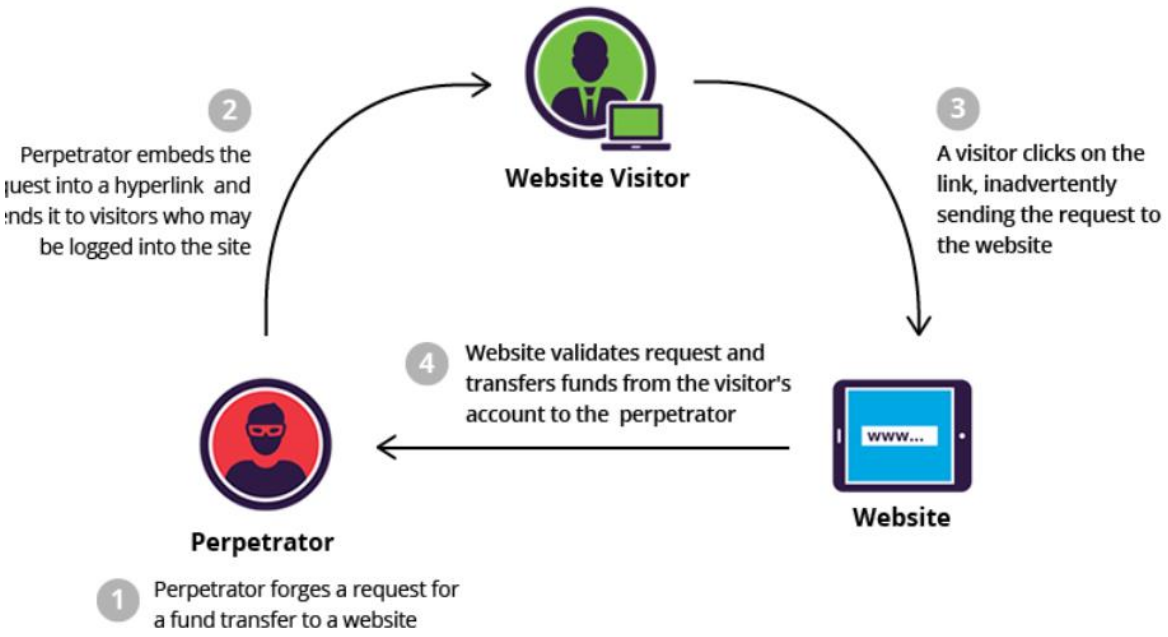

Figure:1 An overview of CSRF attack

CSRF attack can be launched in two ways

1. HTTP GET request
2. HTTP POST request

These HTTP is used to build a communication between client and server, GET and POST are the two methods in HTTP which enables a way to communicate[3].

HTTP GET request

"HTTP GET" is used to obtain the data from the source. In "HTTP GET" request way the intruder just modifies the link or message such that the intruder will be beneficiary of that. Below example gives the clear picture of "HTTP GET request"[1].

```
GET
http://bank.com/transfer.do?acct=PersonB&amount=$10
0 HTTP/1.1
```

Figure:2 original GET request by the user

```
GET
http://bank.com/transfer.do?acct=AttackerA&amount=$
100 HTTP/1.1
```

Figure:3 modified GET request by intruder/Attacker

HTTP POST request

HTTP POST request are used to send data to the web application. The send data may contain the information about the user credentials or any details of the user. In the HTTP POST request, the user need to logged-in into the web application before performing any operation because the user is sending the data, during this action server will make use of session cookies to authentication the user, during this session if the user logged into the malicious website, then the attacker will make use of those session cookies to logged into the web application. These malicious websites contain an HTML code which will give the access to the session cookies[1].

```
<body onload="document.forms[0].submit()">
  <form action="http://netbank.com/transfer.do" method="POST">
    <input type="hidden" name="acct" value="AttackerA"/>
    <input type="hidden" name="amount" value="$100"/>
    <input type="submit" value="View my pictures!"/>
  </form>
</body>
```

Figure: 4 HTML code for HTTP POST request attack

CSRF Mitigation methods:

CSRF attack can be avoided or mitigated in many ways, some of the best solutions are[2]

1. Logging out after the usage of the website.
2. Make sure to use the <a> tags in HTML code.
3. Better not to save password to the search engines.
4. Better not to accept all cookies.
5. Customizing the firewall rules such that firewall will resist against web application security attack
6. Including the captchas to alert the authorized user about the CSRF attack.

Some of the advanced techniques are double check cookie, synchronizer Token pattern.

Generally, Firewalls places a major role in protecting the web application from many of the security attacks. Some of the architectures allows the firewall to detect and prevent the attack at the lower levels itself. Making the firewall stronger by in cooperating the anti-hacking and injection techniques makes web application stronger towards the security attack/security breach.

Related work

The following research papers investigates CSRF attack and proposed different methodizes for detecting and preventing them.

In this research paper, authors used two approaches to prevent the CSRF attack first by analyzing the source code of the application and then using the existing techniques to attack the web application. If any CSRF vulnerability is found then a secured token pattern is attached to the code which generates a unique code when it triggers the change of web page in the application. This made an effective way in detecting the CSRF attack and performed well by producing 100% accuracy[1].

In this research paper, authors focused on the weakness of the application firewalls like signature-based and rule-based techniques. They designed an WAF which analysis the history of the web request made by the user and to check weather he/she made any similar request and the similarity is calculated based on the measure of business statics, if they similar application allows to do the action if they are not similar then the user need to give the extra authorization to prove he/she as a legitimate user. As a future work the authors are planning to implement the system [4].

In this research paper, authors focused on the client-side protection by implementing an algorithm which is designed under the bases of add-on feature in the Firefox browser which is also called as filtering algorithm, while designing the algorithm authors considered three different scenarios. The result of this implementation stood as a countermeasure for cross-origin security attacks and more [5].

Conclusion:

In the data era it is very common to attack the web application but making the security rules and the firewalls configuration stronger is our duty. Designing the new rules for the firewalls and allowing the trusted site to run in the browser and helps in reducing the security attacks of the web application.

Reference

[1] W. H. Rankothge and S. M. N. Randeniya, “Identification and Mitigation Tool For Cross-Site Request Forgery (CSRF),” in *2020 IEEE 8th R10 Humanitarian Technology Conference (R10-HTC)*, Dec. 2020, pp. 1–5. doi: 10.1109/R10-HTC49770.2020.9357029.

[2] “What is CSRF | Cross Site Request Forgery Example | Imperva,” *Learning Center*. https://www.imperva.com/learn/application-security/csrf-cross-site-request-forgery/ (accessed Jan. 22, 2022).

[3] “HTTP Methods GET vs POST.” https://www.w3schools.com/tags/ref_httpmethods.asp (accessed Jan. 22, 2022).

[4] M. Srokosz, D. Rusinek, and B. Ksiezopolski, “A New WAF-Based Architecture for Protecting Web Applications Against CSRF Attacks in Malicious Environment,” in *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Sep. 2018, pp. 391–395.

[5] https://link.springer.com/content/pdf/10.1007%2F978-3-642-23822-2_6.pdf (Accessed Jan. 22, 2022).