Volume: 12 Issue: 02 Pages: 4561-4566(2020) ISSN: 0975-0290

A Study on Cross-Site Request Forgery Attack and its Prevention Measures

Puneet Kour
M.Tech. Student
Department of Computer Science & IT, University of Jammu, J&K, India
Email: puneetkaur199tap@gmail.com

ABSTRACT

Today's security is the most important factor for online users to secure their confidential data, so identify vulnerabilities in a web application has been become a big challenge. OWASP (Open Web Application Security Project) states the ten topmost critical web application security vulnerabilities which affect the security mechanism of web applications. The main objective of the study is to determine the available solutions to prevent Cross-Site Request Forgery (CSRF) attacks. In order to test against the exploitation of the CSRF vulnerability were conducted after implementing the solutions into the web application to check the effectiveness of each of the solutions. The proposed research also combines the solution that unifies the passing of an unpredictable secret validation token through a hidden field and validating it on the server-side.

Keywords- **Web Vulnerabilities, CSRF Attack, Secret Validation Token.**

Date of Submission: September 12, 2020 Date of Acceptance: Oct 15, 2020

1. INTRODUCTION

The vulnerability is a (flaw) in the design or coding of an application that can be exploited by an attacker, to perform unauthorized actions within a computer system. One of the known web application vulnerabilities is Cross-Site Request Forgery (CSRF). According to the Open Web Application Security Project (OWASP), cross-site request forgery is listed as one of the top 10 web application vulnerabilities of 2013 leading to a security breach [1] and is often referred to as the "Sleeping Giant" among the critical vulnerabilities found in Web application [2]. The cross-site request forgery attacks was first introduced by Peter Watkins in a posting to the Bug Traq mailing list, and then it has been picked by web application developers [3]. CSRF is also known as XSRF, Sea Surf, Confused Deputy, One-click Attack, or Session Riding, in this type of CSRF attack hacker tricks a web browser into executing an unwanted (unauthorized forged request) action in an application to which a user is logged in [4]. An untrusted website can force the user browser to send the unauthorized valid request to a trusted website and this can be done without the knowledge of users. So it is a class of attack on web applications that exploit the trust of authenticated users. To make it a successful CSRF attack against the authenticated user, an attacker is able to instigate an arbitrary HTTP request using the user credentials to the vulnerable web application. A successful CSRF attack effectively re-direct the underlying authentication mechanism and it can compromise the end-user personal data and operation.

CSRF attack is when an attacker is able to make requests on the behalf of a user. The attacker takes advantage of the fact that the user is already authenticated to a web application or a particular website. CSRF attack only targets state-changing requests such as changing credentials like passwords, transferring funds, modifying settings, etc also attacker has no way to see the response to the forged request but with a little help of social engineering, an attacker changed the parameters of the script by which the users of a web application into executing actions of the attacker's choosing script. If the victim is a normal user, a successful CSRF attack can force the user to perform state-changing requests like transferring funds, changing their email address or password, and so forth. If the victim works on an administrative level account than CSRF can compromise the whole web application which may be harmful to the victim.

2. CLASSIFICATION OF CSRF

CSRF vulnerability classified in three stages where the attacker perform an unauthorized activities without any knowledge of the user. Existing CSRF vulnerability are classified as:

2.1 Stored CSRF

In stored CSRF, the attacker can use the application itself and provide the victim to exploit the link into the web application. Any web application that uses HTML is vulnerable To CSRF attack. The attacker stores the malicious code using different tags like IMG, IFRAME, LINK, and FORM etc. The victim who receives the malicious content or link is almost certainly currently authenticated to perform action. Examples of stored CSRF are shown below:

- <link> tag
   CLICK HERE TO GET REWARD
- <form> tag
  <form action = http://127.0.0.1/hello.php method = "POST">
  </form>
- <script> tag
  <script>

Volume: 12 Issue: 02 Pages: 4561-4566(2020) ISSN: 0975-0290

document.location = http://127.0.0.1/xyz.php/?password_new=1234xyz&password_conf=1234xyz&Change=Change#;
</script>

2.2 Reflected CSRF

A reflected CSRF vulnerability is one where the attacker performs the attack outside from the system application to bare the victim to exploit links. For exploitation the hacker using an email message, a blog, an instant message, or even an advertisement posted in a public with a URL that a victim type in. The reflected CSRF will be a failure, as users may not be currently logged into the target system when the exploit is tried, if the victim logged into the target system then the page can be submitted automatically. Example of such, are shown below:

- Auto Post Forms
  <body onload = "document.form[0].submit()">
  <form method = "POST" action= "http://127.0.0.1/hello.php">
  <input type = "hidden" name = "welcome" value = "987ads"/>
  </form>

3.2. Login CSRF

In this type of attack, personal data of victims is collected. In login CSRF attack, both the user and the attacker are authorized, users. But the hacker will send his identity to the victim through some link. Thus victim will now enter using the attacker's identity. Then all the information related to the user will get stored in the user browser, but actually in the attacker identity.

- <form action= https://www.abc.com/ method=POST target=invisibleframe>
  <input name=username value=attacker>
  <input name=password value=xyz>
  </form>

3. AIM AND OBJECTIVES

The main objective of this study is to find out the vulnerabilities leading to a CSRF attack in the web application. The dissertation also aims to study the mitigation techniques of the attack and propose a solution for the same.

4. LITERATURE WORK

CSRF attacks occur when a hacker is able to send a crafted (malicious link), request to an authenticated user where an attacker changes the necessary parameters in the form to complete a valid application request without the victim even realizing it. Many of the hackers created crafted (fake) web pages which contain the malicious script/data in order to trap the users by impersonating these pages as real web sites. Generally, the creating of fake websites is growing on the internet by which the attacker must focus on victims to steal their confidential information. Researchers are also carrying out their studies on this domain. Some of them are listed as:

Ramarao R. et.al. [5] presented a client-side proxy solution that works in the two-stage detection and prevention stage. In the prevention stage, the HTML elements are used to access the graphic images of the webpage. The client-side proxy is able to examine and alter client requests.

Adam Barth, Collin Jackson, and Mitchell Stanford [6] presented a study on CSRF vulnerabilities and provided three major different solutions of CSRF defense techniques in which the HTTP Referer header could work more effective defense solution in their experiment and for the long term solution, the author proposes the browser the Origin header, which provides the security benefits of the Referer header.

Wasim Akram Shaik and Rajesh Pasupuleti [7] proposed a technique to prevent CSRF attack using the TowFish security approach. In which the user can identify whether the website is vulnerable to CSRF. The Towfish security approach work in two stages, in the first stage hash value of URL is calculated and in the second stage for the respective site image-based authentication is used to validate the image of the website.

A. V. Barabanov and A. S. Markov [8] presented a paper and provided the results of consolidating information about the attack and protection measures. The results of the study demonstrated various distribution types, identified vulnerabilities as per the developer, security measures used in web-based applications, vulnerabilities as per the programming languages, also the number of security measures used in the study of web applications.

Emil Semastin, Sami Azam et.al. [9] proposed a solution that integrated the unpredictable token through a hidden layer and then validated the tokens on the server-side with the passing token through URL which can offer double-layer protection against Cross-Site Request Forgery attack.

Sheeghrata Agnihotri and Pawan Patidar [11] proposed a client-server mutual authentication technique in which two steps are considered first one is the authentication step and the second one is and these identification steps and these two steps are working separately. For the authentication, the token is provided to each user, and the token is provided in the form of an image that is encoded and decode using the base64encoding and decoded technique.

Jaya Gupt and Suneeta Gola [12] presented a paper on CSRF attack in which server-side protection again CSRF attack is implemented and the approach is named as CSRF gateway. In this approach, the token is generated when the HTTP request is generated but the user, it creates a session and embeds the token to the session using a custom tag library which provides the secured way to insert token and the application detects the CSRF attack has been occurred or not and redirect the user back to the login page.

5. EXPERIMENTAL SETUP

In this study, a banking website in PHP has been developed and hosted on the localhost (XAMPP sever). The experiments to exploit CSRF vulnerability on the website have been performed to steal user's credentials. To test the vulnerability in the web application is trace out using an online acunetix tool and there are many other

Volume: 12 Issue: 02 Pages: 4561-4566(2020) ISSN: 0975-0290

tools that are available online to check the vulnerabilities on the website before make the website dynamic (online). The study is focused on persevering and reflected attacks on the website that maintains the user's authentication state. Modern browsers like Google Chrome, IE, Firefox, and UC Browser have been used to carry out these experiments. The Fig.1 shows the architecture of CSRF for carrying out CSRF attacks on the target web application on the localhost.

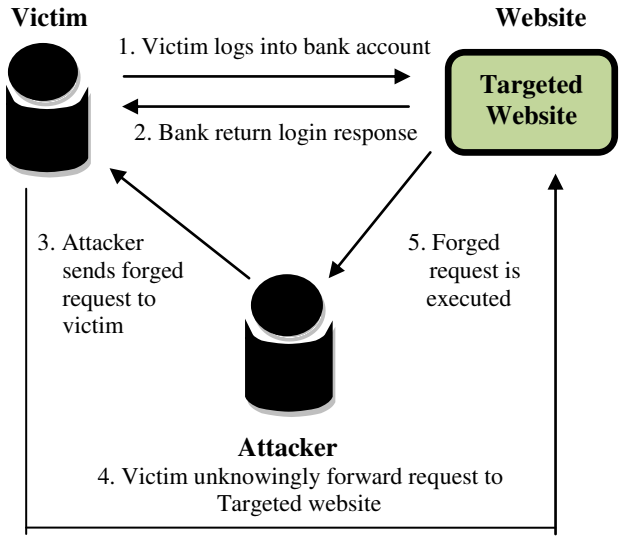

Fig.1: Architecture of CSRF attack

The above Fig. 1 shows the working of the CSRF attack, in which a user sends a login request to the bank website then the bank sends a login response to the user after that user becomes an authenticated to the bank website. When all this process are done now attacker parts come where attacker manage all the basic details of the victim with the help of social engineering. When the attacker gets all the basic details of the victim, he sends a forged request to the victim via. through email, from another website, or by instant message, etc where the malicious link is hidden in the backend (hidden information will be the victim's account number, set how much amount to be transferred) and the victim does not know about these changes. If the victim clicked on the malicious link the query is executed and the CSRF attack takes place without any knowledge of the victim because it is all hidden in the backend of the malicious link.

6. EXPERIMENTAL WORK

In this study, a banking website in PHP (http://localhost/banksite/insert.php) is developed and hosted on the localhost (XAMPP server). For attacking purposes (http://localhost/attackerblog/attck.php) is also developed and implemented on the virtual host (XAMPP server). The vulnerabilities in a web application are traced out using acunetix tool and to perform a CSRF attack by injecting a malicious link which is a completely different website and the experiment the possible CSRF attack that can lead to transfer funds from the user's account to the attacker's account. Firstly, the experiments are carried out without implementing any preventive measures on the website. The attacker attempted to inject malicious links to transfer all funds from the victim's account. The steps involved in transfer funds from a victim's account by sending a malicious link to the victim's browser and these steps are explained as follows:

STEP 1: The authenticated user first login into the website to check his account details, but for some reason, the user forgets to logout from the website.

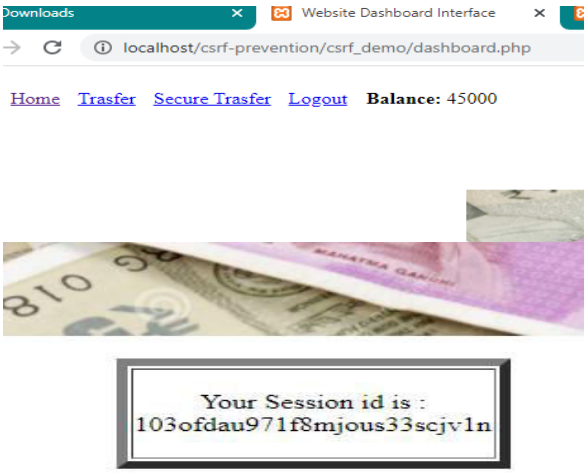

Fig.2: User Login with Session id with current balance

STEP 2: Then the attacker sends a forged request (i.e. malicious link) to the victim's browser to comment on this blog, if the user posts a comment on attacker blog then the malicious link is executed and the attacker steals the funds from the victim's account.

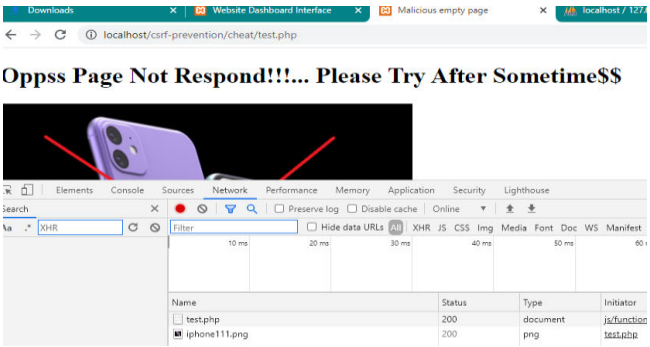

Fig.3: User Respond on Attaker's Malicious link

STEP 3: If the victim clicks on a malicious link or comment on the attacker's blog and at the same time the malicious link or script is executed which is harmful to the victim, it transfers all the funds from the victim's account to the attacker's account and the above all this process is done without the victim's knowledge.

Volume: 12 Issue: 02 Pages: 4561-4566(2020) ISSN: 0975-0290

Fig.4: Attacker transfer the money from victim's account

STEP 4: When the victim clicked on the malicious link the parameters changed in the form by the attacker is executed and then the amount set by the attacker in the form parameters is transferred to the attacker's account without any knowledge of the victim.

```
<form action= "" method= "POST">
<input type= "hidden" name= "account no." value= "708347235806">
<input type= "hidden" name= "amount" value= "35000">
<input type= "hidden" name= "transfer" value= "transfer">
</form>
```

Fig.5: Malicious script send to victim browser

Also to test or exploit the vulnerabilities of CSRF attack, online platforms DVWA (Damm Vulnerable Website Application) [13] is used. DVWA is a PHP/SQL web application that is damm vulnerable. Its main goals to provide aid for professionals and ethical hackers to test their skills and tools in a legal environment. It also helps web developers to better understand the processes of securing web applications in a safe environment

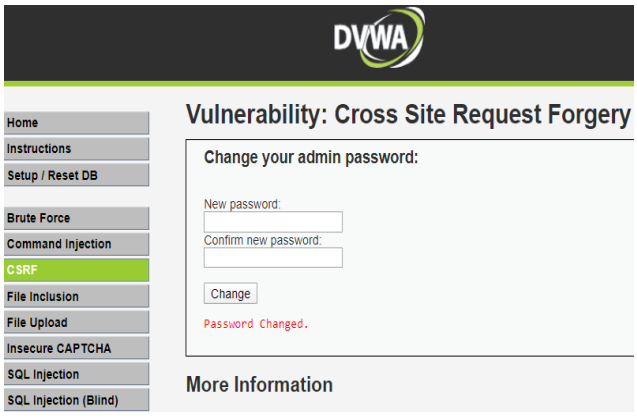

Fig.6: Exploitation of CSRF and Password change Query is generated

```
script>
document.location="http://localhost/dvwa/vulnerabilities/csrf/?password_new=abcd12 & password_conf = abcd12345 & Change=Change#";
</script>
```

Fig.7: Password Change Query Script

7. PROPOSED SOLUTION

Most of the CSRF attacks are working on state-changing requests where the attacker has no idea to see the response to the forged request but with the help of the social engineering site attacker changes the parameters of the form which is hidden in the backend of the form and sends the crafted script to the victim through different websites. The proposed technique will present an effective solution, which can provide secret validation token protection against CSRF attack. The proposed solution work as firstly the user can enter their login credentials, if the user has an account, the application simply accepts the request otherwise application rejects the request and sends a message to register or login first. If the user is authenticated to the application, he/she enters the dashboard of his/her account, where the user gets the currently going session id which is every time unique whenever user logged into the website. During the time of any transaction (user session id and tokes are matched) if they are matched then only the transfer query is executed otherwise application again rejects the request and sent a message to register first.

But at the same time, the attacker sends a forged request to the victim browser if the victim clicked on crafted link than the application rejects the request because the attacker is never aware of hidden tokens in the form to execute the query. The Fig. 8 shows the generated CSRF tokens and the attacker is unaware of these hash tokens. If the tokens and session-id are not matched with the current going session-id and CSRF tokens of victims and here the attacker's tricks are going to fail for stealing money from the victim's account. Then we can say the proposed solution work against CSRF attack and work effectively. The Fig. 9 shows the flowchart of the proposed solution and the working of the general progression of the algorithm.

Fig.8: CSRF Tokens are generated automatically

Volume: 12 Issue: 02 Pages: 4561-4566(2020) ISSN: 0975-0290

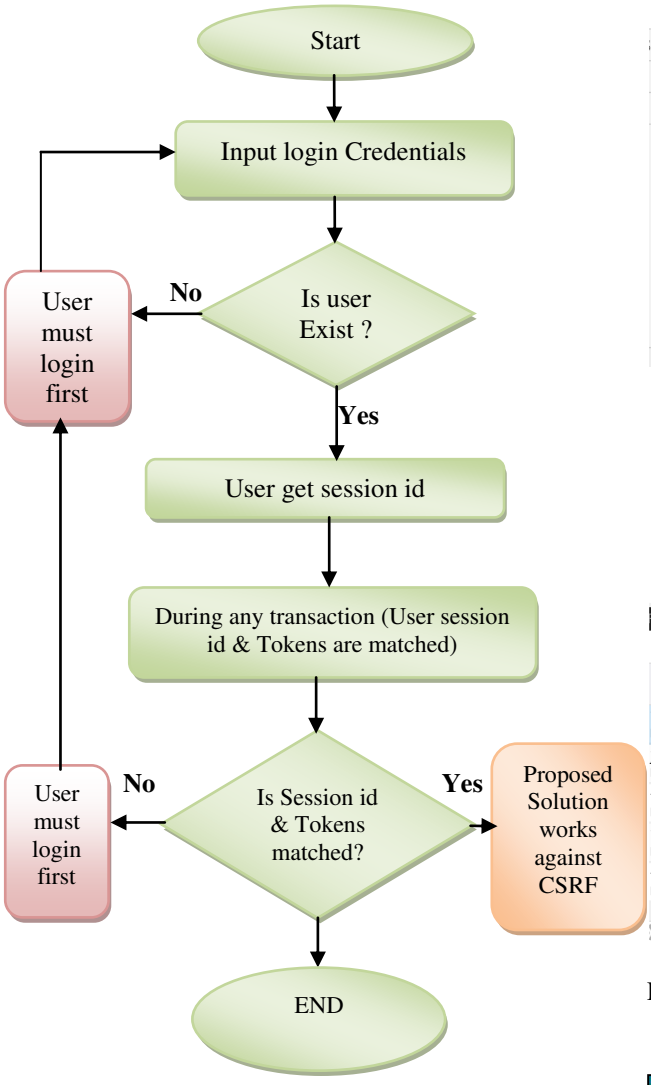

Fig.9: Flowchart of Proposed Solution

8. RESULTS AND DISCUSSION

By performing these experiments, it was found that secret validation tokens mitigate CSRF attack. The experiment involving the study of the behaviour of browsers for CSRF attacks in different browsers has been carried out successfully. Apart from these experiments, the proposed solution has been implemented on the web application to check the effectiveness of the solution.

The web pages with CSRF attacks have been checked by applying the proposed solution which is secret validation tokens with the user session-id. The results of which have been shown in Fig.10, Fig.11, Fig.12:

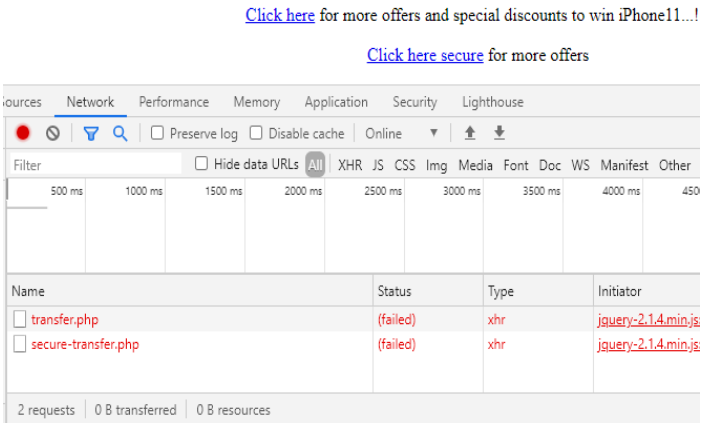

Fig.10: Malicious script not execute due to secrect tokens in Chrome

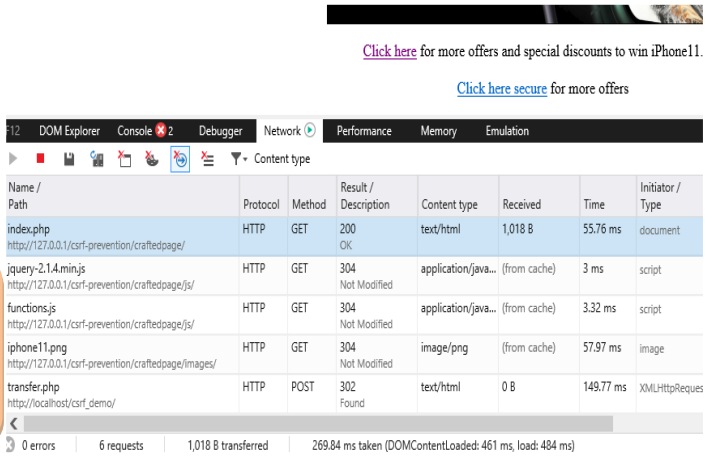

Fig.11: Malicious script not execute due proposed solution in Internet Explorer

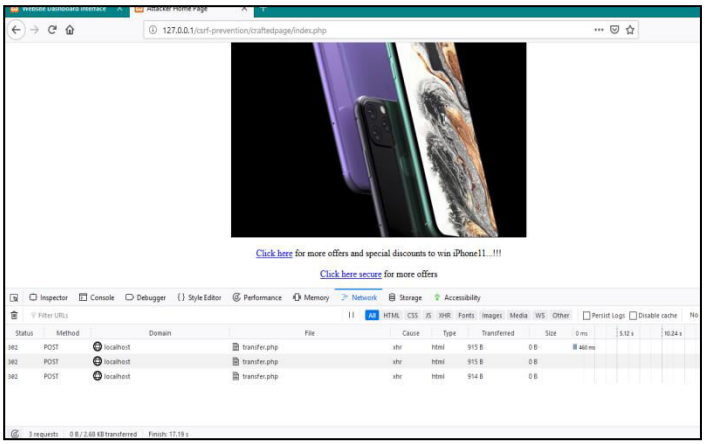

Fig.12: Malicious script not execute due to proposed solution on Firefox

The results have shown that the proposed solution for the CSRF attack successfully prevents the attack. The proposed solution has been successfully implemented in different browsers like Chrome, Internet Explorer, Firefox, UC browser.

Volume: 12 Issue: 02 Pages: 4561-4566(2020) ISSN: 0975-0290

9. CONCLUSION

Cross-Site Request Forgery is one such vulnerability and these attacks can be easily executed by the attackers by simply changing form parameters (mostly in hidden form parameters are stored) and take advantage of it. But passing secret validation tokens and session-id and validate it at the server-side is the most effective solution. By implementing this solution, double verification takes place in the form of currently running session-id and unpredictable CSRF tokens in the web application to ensure the prevention of the CSRF attack. The above mention work can be extended to provide better solutions for the CSRF attack by means of applying techniques to preventing the attack before the attacker's attack.

REFERENCES

[1] J. Grossman, "Whitehat website security statistics report Online". Available at: www.whitehatsec.com . 2007.

[2] Jovanovic, Kirda and Kruegel, "Preventing cross site request forgery attacks", in *Proceedings of IEEE Securecomm and Workshops*, 2006, 1-10.

[3] Rupali D. Kombade, Dr. B. B. Meshram, "CSRF Vulnerabilities and Defensive Techniques", *International Journal of Computer Network and Information Security*, 2012, 31-37.

[4] Kafer K, Cross Site Request Forgery, Online, Technical report, Hasso-Plattner-Institute, 2008.

[5] Ramarao R. Tool "Preventing Image Based CSRF attacks", Avalailabe at: http://isea.nitk.ac.in/rod/csrf/PreventImageCSRF/. Accessed on May, 2009.

[6] Adam Barth, Collin Jackson and Mitchell Stanford "Robust Defenses for Cross-Site Request Forgery", in *Proc. Association for Computing Machinery ACM*, 2008.

[7] Wasim Akram Shaik and Rajesh Pasupuleti, "Avoiding Cross Site Request Forgery (CSRF) Attack Using TwoFish Security Approach, *International Journal of Computer Trends and Technology*, *25(*2), 2015, 68-72.

[8] A. V. Barabanov, A. S. Markov and V.L. Tsirlov, "Information Security Controls against Cross-Site Request Forgery Attacks on Software Applications of Automated Systems", *International Conference Information Technologies in Business and Industry*, 2018.

[9] Emil Semastin, Sami Azam et al. "Prevention Measures for Cross Site Request Forgery Attacks on Web-based Applications", *International Journal of Engineering and Technology*, 2018, 130-134.

[10] Sentamilselvan K et.al. "Survey on Cross Site Request Forgery", *International Conference on Research and Development Prospects on Engineering and Technology*, *5*, 2013, 159-164.

[11] Sheeghrata Adnihotri and Pawan Patidar, Prevention against CSRF Attack using Client Server Mutual Authentication Technique, *International Journal of Engineering Science and Computing*, *9*(3), 2009, 20393-20398.

[12] Jaya Gupta and Suneeta Gola, Server Side Protection againstCross Site Request Forgery using CSRF Gateway, *Journal Information Technology & Software Engineering*, *6*(3), 2016, 3-8.

[13] Damm Vulnerable Web Application (DVWA), Available at: http://www.dvwa.cp.uk/.

[14] OWASP (Open web application security project) top ten project, http://www.owasp.org/index.php/ Category: OWASP-Top_Ten_Project 2013. Available at: http://www.owasp.org/index.php/CrossSiteRequestForgery .

[15] Xiaoli Lin, Pavol Zavarsky, Ron Ruhl and Dale Lindskog, "Threat Modeling for CSRF Attacks", *Proc. IEEE International Conference Computational Science and Engineering*, 2009, 486-491.

[16] P. Khurana and P. Bindal, "Vulmerabilities and degensive mechanism of CSRF", *International Journal of Computer Trends and Technology*, *13*(4), 2014, 2231-2803.

[17] Nenad Jovanovic, Engin Kirda and Christoper Kruegel, "Preventing cross site request forgery attacks", *Proc. IEEE International Conference on Security and Privacy in Communication Networks*, 2006.

[18] Hossain Shahriar and Mohammad Zulkernine, "Client-Side Detection of Cross-Site Request Forgery attacks", *IEEE International Symposium on Software Reliability Engineering*", 2010.

[19] R. P. Seenivasan and K. Suresh Joseph, "A Survey of Clickjacking Attack and Countermeasures in Web Environment", International Journal of Advance Networking and applications (IJANA), 2016, 206-213.

[20] Haneet Kour and Lalit Sen Sharma, "Tracing out Cross Site scripting Vulnerabilities in Modern Script", International Journal of Advance Networking and applications (IJANA), 2016, 2862-2867.

AUTHORS BIOGRAPHY

Puneet Kour is a Master of Technology (M.Tech) student in Department of Computer Science and IT, University of Jammu. She has obtained her Undergraduate degree in Information Technology (IT) from Mahant Bachittar Singh College of Engineering and Technology, Jammu. Her research interests are in the field of Networking and Network Security.