Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 6, Issue. 7, July 2017, pg.73 – 76

Automated Detection of Cross-site Request Forgery Vulnerability in Governmental Universities Websites

Mustafa ElGili Mustafa

Computer Department, Community College, Shaqra University, Shaqra, Saudi Arabia
Faculty of Computer Science and Information Technology–Neelian University, Khartoum, Sudan
mustgili@hotmail.com

Abstract--- In this paper, we explore the Cross-Site Request Forgery (CSRF) vulnerabilities. This paper made scan over (22 Websites) of Governmental Universities Websites to explain if they suffer from this f vulnerability, by using Acunetix Web Vulnerability Scanner, and as a results of these scan 81% of these websites may suffer from CSRF.
Keywords: Cross-site Request Forgery, Session-riding, XSRF, Website Vulnerability.

I. INTRODUCTION

Cross-site Request Forgery (CSRF) is one of the major attacks inflicted on the session management mechanisms. An attacker in a CSRF (also known as a one click attack, session-riding, or XSRF) attack forces a victim to execute unwanted actions on the web application for which the victim is currently authenticated [1]. Cross Site Request Forgery (CSRF) is among the most exploited web security vulnerabilities, along with Cross Site Scripting and SQL Injection. Yet CSRF has received comparatively less attention. In a CSRF attack, a malicious site instructs a victim's browser to send a request to an honest site, as if the request were part of the victim's interaction with the honest site, leveraging the victim's network connectivity and the browser's state, such as cookies, to disrupt the integrity of the victim's session with the honest site. It's a must for web site administrators to protect their web sites from CSRF attacks.[2]. CSRF attack is less discussed in security classes. Most of the web developers do not know about this attack and confused with XSS (Cross site scripting). Most of them who know about CSRF think that protection against XSS will also work in case of CSRF. But CSRF and XSS are not the same thing[3].

II. CROSS-SITE REQUEST FORGERY (CSRF)

Typically, today's websites implement cookies to identify authenticated users. After the Web server successfully authenticates the user, the browser will get an identity login cookie to remember the logged-in status. Later, when the user is visiting the Web pages of the target Website, the browser will automatically put the identity login cookie in the request. This will not be removed until the browser is closed or the user logged out. The adversary is able to abuse this duration to make the user's browser perform some actions that the user probably does not want [4]. Cross-Site Request Forgery (CSRF occurs when a malicious web site, email or web forum causes a victim's web browser to perform an undesired action on a trusted web site. [5].

The consequences of CSRF attacks may differ based on:

- The vulnerabilities exploited.
- The privileges of the victim/user being exploited by the attacker[5].

In CSRF, an attacker uses an external website to trigger an HTTP request from a client to a target website. Suppose that a client currently has an active session with a target website A and then visits a malicious website B. The attacker can put a malicious URL on website B that triggers the client's browser to send an HTTP request to website A using the currently active session. Then the credentials associated with website A will be attached to the triggered HTTP request, and website A will process the request using the client's privileges. There are variants of the CSRF attack, including Clickjacking and Login CSRF [6]. Cross Site Request Forgery is also a mayor security threat [6] consisting in sending a malicious request to vulnerable website, usually from an authenticated client of the server trusts; the malicious request could include performing data deletion, doing transactions or changing passwords. This attack is successful due the fact that developers tend to trust that a client will never send a request he/she is not entail to or one that the GUI is not designed to dispatch [7]. Cross Site Request Forgery (CSRF) is an attack that coerces an unaware user to send unauthorized requests to any web domain by interacting with a seemingly unrelated entity on a malicious website [8]. Cross Site Request Forgery (CSRF) allows an attacker to perform unauthorized activities without the knowledge of a user.The severity of the damage depends on how much sensitive operations can be performed through these requests [9]. An attacker gives a manipulated URL to a victim. When visited, the URL will actually do a request to web server without any acknowledgement from the victim. There are some conditions in order to successfully perform a CSRF attack:

- Attacker has to find a web application that does not check HTTP referrer header and anti CSRF token.
- Attacker has to find a request, which can be manipulated.
- Attacker has to determine the correct value for each field in a request.
- Attacker has to attract an authenticated victim to send the manipulated request [10].

To launch a reflected CSRF attack, what the attacker needs to do is just craft a simple link and put it on his malicious web page and lure the victim to click it. This can be done by sending out spam emails. When the victim clicks the malicious link, it will force the victim's browser to load logout.php. Normally, the victim's credentials (the session ID) will be sent with the request to logout.php that would logout the victim.[4]

The severity of the damage depends on how much sensitive operations can be performed through these requests [9].

III.BY- PASS CSRF PROTECTION

Even though some Web sites employ secret tokens to protect their users from CSRF attacks, the attacker has the ability to by-pass CSRF protection mechanisms, especially when the target website has insufficient session expiration vulnerability. Because the adversary can use the old cookies, what he could do is steal the user's session tokens and predict the CSRF random tokens via insufficiently random number weakness, or the attacker can leverage the errors within CSRF prevention routine [4].

IV.LIMITATION

There are many limitations of CSRF attack. The success of CSRF depends on many factors. You can see it as a blind attack where many factors are involve in it's success. These are some factors.

I. The attacker must find a website that does not check referrer header.

2. Attacker should have the idea of the structure of that website and it's forms.

3. The user must be logged in in the website while clicking on the link. These factors should be true while attacking.[3]

V. PROTECTION AGAINST CSRF:[3]

It's really hard to protect against CSRF attack. But you can use some of these or all of these to prevent an attacker to perform this type of attack.

1. **Use of random tokens:** This is the most powerful step against CSRF. Use random tokens each time with a form submission. It's very difficult for the attacker to guess the next random pattern to fill in malicious URL. But the random values should be unpredictable.
2. **Use of post in form rather than get:** Get and Post are the 2 methods of form submission. Post method is secure for form submission. In get method any one can see the variables and values in URL as a query strings.
3. **Limiting the lifetime of authentication cookies:** This is also a strong prevention against CSRF. Limit the lifetime to a short period of time. If user will go on other website then the cookies will expire after a short period of time. And user will have to login again for any action. If the attacker will try to send any HTTP request user will be able to know and he will not fill the password again.
4. **Damage limitation:** Damage limitation involves those steps which reduced the damage from CSRF. For example if an attacker did manage to perform CSRF on a website then any action done by him will require an authentication every time to limit the damage.
5. **Force user to use your form:** Always force user to use the form of website. Use of hidden fields are helpful for this purpose. But this way of protection is easy to bypass.

VI. VULNERABILITY ANALYSIS

Testing the CSRF vulnerability over the 22 relevant websites requires a lot of time and effort. we selected and used an automatic web vulnerability scanner: Acunetix Trial Edition. This tool collects information about different vulnerabilities just by providing it with the website URL.

VII. RESULTS

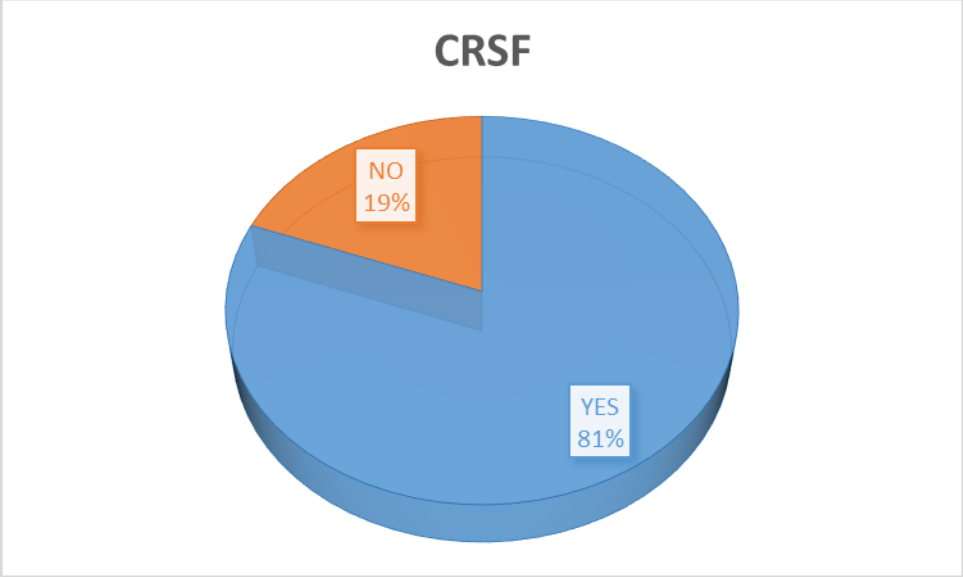

Figure 1 illustrate the percentage of CRSF vulnerability over tested websites

As shown in figure 1 81% of the governmental university website with no CRSF protection, and that means this sites threaded by CRSF attack.

VIII. CONCLUSION

As results explain the governmental university websites designers are ignored security side design and they did not protect their websites against CRSF vulnerability and Password auto complete vulnerability, which is mean the governmental university websites designers need to improve website protection techniques. Security should be considered during web application develop life cycle because attackers could abuse the web application because it is not secure. Developer needs to understand what they should consider or what they should do to avoid the potential risk mistakes. Users are not intuitive to understand the concern of information security; developers should put extra effort on securing their web application.

REFERENCES

[1] Automated Detection of Session Management Vulnerabilities in Web Applications, Yusuke Takamatsu and others, 2012 Tenth Annual International Conference on Privacy, Security and Trust.
[2] Research on Developing a Lab Environment for Cross Site Request Forgery, Huangcun Zeng, 2013 IEEE.
[3] Cross Site Request Forgery: A common web application weakness, Mohd. Shadab Siddiqui, 2011 IEEE.
[4] Threat Modeling for CSRF Attacks, Xiaoli Lin, 2009 IEEE.
[5] A Study of the Effectiveness of CSRF Guard, Boyan Chen, 2011 IEEE.
[6] A Privacy-Preserving Defense Mechanism Against Request Forgery Attacks, Ben S. Y. Fung and Patrick P. C. Lee, 2011 IEEE.
[7] An Analysis of XSS, CSRF and SQL Injection In Colombian Software And Web Site Development, Danny Alvarez E 2016 IEEE.
[8] Cross-Site Request Forgery: Attack and Defense, Tatiana Alexenko and other, 2010 IEEE.
[9] Client-Side Detection of Cross-Site Request Forgery Attacks, Hossain Shahriar and other, 2010 IEEE.
[10] A Vulnerability Scanning Tool for Session Management Vulnerabilities, Raymond Lukanta and others, 2014 IEEE.

