ORIENTAL JOURNAL OF
COMPUTER SCIENCE & TECHNOLOGY
An International Open Free Access, Peer Reviewed Research Journal
Published By: **Oriental Scientific Publishing Co., India.**
www.computerscijournal.org

ISSN: 0974-6471
June 2017,
Vol. 10, No. (2):
Pgs. 359-363

Secure ASP.NET Web Application by Discovering Broken Authentication and Session Management Vulnerabilities

RUPAL R. SHARMA[1], RAVI K. SHETH[2]

[1]M.Tech in Cyber Security, Student, Department of Information Technology,
Raksha Shakti University, Ahmedabad, Gujarat, India
[2]Department of Information Technology,
Raksha Shakti University, Ahmedabad, Gujarat, India
Corresponding author Email: Rupsharma23@gmail.com

http://dx.doi.org/10.13005/ojcst/10.02.15

(Received: March 25, 2017; Accepted: June 03, 2017)

ABSTRACT

Today, web application security is most significant battlefield between victim, attacker and resource of web service. The owner of web applications can't see security vulnerability in web application which develops in ASP.NET. This paper explain one algorithm which aim to identify broken authentication and session management vulnerability. The given method of this paper scan the web application files. The created scanner generator relies on studying the source character of the application limited ASP.NET files and the code be beholden files. A program develop for this motive is to bring about a report which describes vulnerabilities types by mentioning the indict name, disclose description and its location. The aim of the paper is to discover the broken authentication and session management vulnerabilities. The indicated algorithm will uphold organization and developer to repair the vulnerabilities and recover from one end to the other security.

Keywords: Session management, session hijack, Broken Authentication, Web security, ASP.NET

INTRODUCTION

World Wide Web has evolved from a position that delivers static pages to a proclamation that supports distributed applications, met with as internet applications and become a well-known of the most universal technologies for information and service delivery around Internet. The increasing currency of World Wide Web application can be showing several factors, including remote accessibility, cross-platform compatibility, hasty development, etc. Website security is a part of Information Security which deals with security of the website, World Wide Web services and web applications[1]. The open web application project defined website's top vulnerabilities which is listed below[2].

OWASP Top Ten vulnerabilities[2]

The OWASP define that web application related functions devoted to authentication and session management are not implement suitably which allow attacker to compromise key, password, session token or to exploit freak identities and implementation flaw[3].

Table 1: OWASP TOP 10 Vulnerabilities

1	**Injection**
2	Broken Authentication and Session Management
3	Cross-Site Scripting-XSS
4	Broken Access control
5	Security Misconfiguration.
6	Sensitive Data Exposure.
7	Insufficient Attack Protection
8	Cross-Site Request Forgery (CSRF)
9	Using Components with Known Vulnerabilities
10	Under-protected APIs

Web application security statistics report also shows average vulnerability age by risk which display below[4]. Following chart show that how many days need to fix or recover any web application which is affect by different attacks? By analysis we concluded that session management and broken authentication vulnerabilities are very harmful for web application. It is take more days to recover the web application.

To build custom session management and authentication scheme correctly is a complex process for the Developers. These custom scheme have flaws in well-known areas like as create account, logout, timeout, secret question, forget password etc. It is very difficult to find a flaw for unique implementation[2].

The flow of document is as follows. In these Section we provide introduction of vulnerabilities. Section II give detail of most harmful attacks which cause by broken authentication and session management vulnerabilities in asp.net web application. In Section III, Implementation of Suggested Algorithm, we present our propose algorithm and its steps. Section IV is the Result part, which show our research analysis of findings. Section V is the Conclusion, in these part of the paper, it show the conclusion of whole paper.

Vulnerability Types

Brute Force Attack

Brute force attack is one type of trial and error method used to receive information such as a user password, Key or personal identification number. In these attack, application programs used to decode the encrypted data[6].

Hijack Session

Session hijacking exploit legal session using same token to get information, unauthorized access or service of the system or website. It's nothing but hijacking a session. To hijack the session from victim, attacker needs its cookies. To implement it, create one form and submit to the attacker site.

```
<script> abc.submit () </script>
<form name = 'abc' method = 'post' action = 'addsite'
>
<input type = hidden value = '<script> + document.
cookie + </script>'>
</form>[7].
```

Replay Attack

A replay attack is one type of network attack in which data is fraudulently or maliciously transmit. The attacker intercept that data and he again transmits it. It's also part of a masquerade attack as stream cipher attack.

Session Fixation Attack

Session fixation attack try to exploit the vulnerability in program or in the system which allow user to set another user's session identifier. Session fixation attack mostly accepted from URL means POST data or query string which rely on session identifiers.

Session timeout

The Timeout property specifies the time-out life assigned to the Session object for the application, in minutes. If the user of application does not regenerate or request a page within the time-out period, the session ends[6].

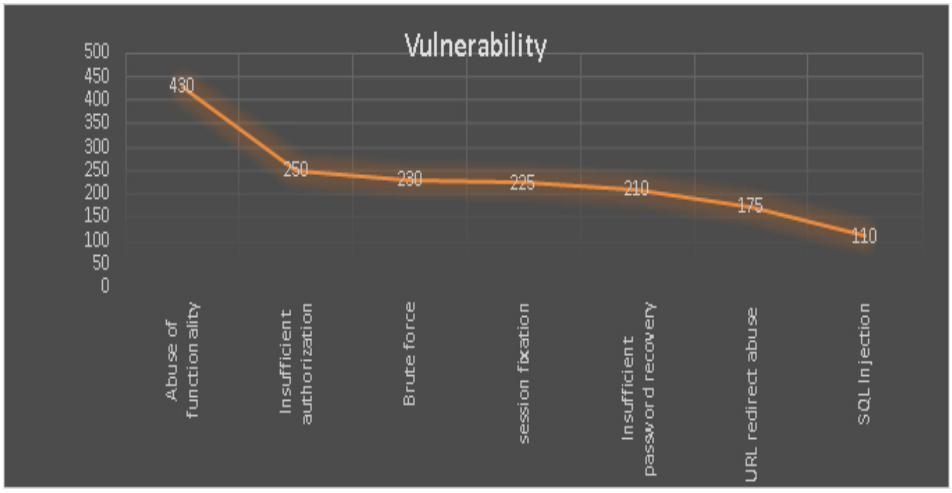

Fig. 1: Average time to fix vulnerability

Suggested Algorithm

Following algorithm describe the Vulnerabilities. We developed an algorithm for it and we will apply in python script[8]. This algorithm[9] consists of 11 steps where each steps of algorithm consider a specific kind of vulnerabilities. Check for Broken Authentication and session management.

These algorithms scan the following types of files which are 'aspx', 'aspx.cs and 'web.config[10]. Step 1: Detect the code if "Validate Request" attribute exits in the website's web configuration file and its value is "False", report, there's vulnerability.

We can secure it by showing of full errors information to the users that may be shown to the hackers

Step 2: Detect the code if any "debug" attributes exit in the website's web configuration file and its value is "True", report that there is vulnerability.

Step 3: Detect in the code if "Textbox's ID" value in designing file of .aspx doesn't validate using "Range Validator" or "Regular Expression Validator", report there's vulnerability.

Step 4: Detect the code, if "Session State" mode is off in website configuration file, report there's vulnerability.

Step 5: Detect the code "Timeout" of the session is define or not in session state of web.config. If timeout of the session does not exists in the configuration file, report there's vulnerability.

Step 6: Detect the authentication code is exists or not in the web.config. If exists then check that it's deny all anonymous user or not. If "deny users" are not exists in web.config then report there's vulnerability

Step 7: Detect the code in .aspx, and .aspx.cs files if "Request.QueryString and Request.cookies" commands haven't any kind of Anti XSS technique otherwise report there's vulnerabilities.

Step 8: check if after logout, cookie remove code exists or not in web.config. If it's not exists then report there is vulnerability.

Step 9: Check at the logout function, session destroy function exists or not, if it does not exist then report there is vulnerability.

Step 10: Detect the code "autocomplete" attribute in the form is on or off, if it is "on" then report there is vulnerability[5].

Step 11: Detect the code "cookie less" value exists in session state of web.config file. If it does not exists, report there's vulnerability.

RESULT AND DISCUSSION

Using several online web application, we started to examine the proposed algorithm. However, we didn't experience to gain any live web application. So, we have to examine it offline. For these motive we created web application and put it into IIS server. We have also downloaded some code from hereafter website:

https://www.codeproject.com

https://www.github.com

Sample of vulnerabilities scanning in asp. net web application is shown in the following screenshot:

We scan more than twenty website/web application in the various tools but we don't find root cause of the attacks and broken authentication and session management vulnerabilities. For that, we purpose this algorithm which shows the exact location in the source code which vulnerable and causes the attack.

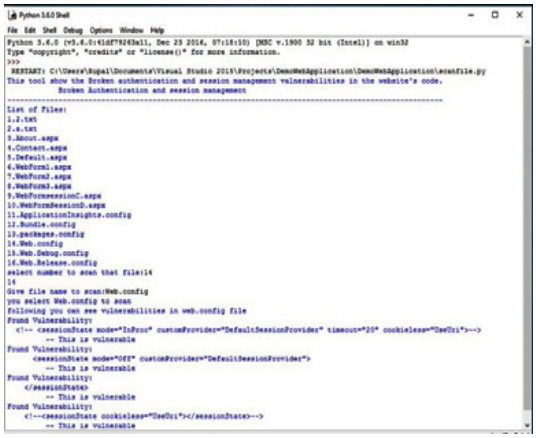

Fig. 2: Snapshot of scanning vulnerabilities

The Finding vulnerabilities in asp.net web application is a very difficult process, in which code is written by somebody else and doesn't not have any documentation which can explain the purpose or meaning of small number code. In the .net framework, programming code vary from the HTML code. There are two different files. Programming code is written in the compiled language like c#. All over the world, c# is the most useable languages with asp.net file. So, to construct our approaching algorithm, we use that language. Therefore, to scan processes, we have three types of files which are aspx.cs, aspx, and configuration. The developed python program scan these different files and forms and show the founded vulnerability.

Prevention Technique
Identity confirmation

Broken authentication and session management vulnerability highly avoided if we change the session ID when users try to log in. If every user require the authentication on every request whenever he logged into the website, the attacker also require the victim's id and login session. In this scenario, the victim want to do any important things however he visits link which has settled session id, he has to login in his account. At that time, his session id will change and attacker can't do anything using that anonymous ID[2].

HTTP cookie and session identifiers

By default, session identifier store in HTTP cookie, which is moderate level security and it's also disregard the GET and POST value.

Deploy SSL / TLS session identifier

Many web development languages do not provide the SSL or TLS session identifier which is secure and robust built in functionality. After enabling HTTPS security, system allow only application which obtain the SSL/TLS session identifier[12].

CONCLUSION

Reduce the broken authentication and session management vulnerability in any web application or website needs two things. First, developer have to aware of the install security at the beginning of developing any program or application[11]., and another things is that owner of web application must inspect his/her website or web application before publishing it. These paper represent the scanning tool which design to discover leaks in source code of web application which help to developer to reduce the vulnerabilities in web application. After the scanning, python script generate a report which describe all uncover vulnerability and leaks by showing name of infected files, location and description. The paper has limitations that it's only work for asp.net and their supported language. In future, we can make this type of algorithm for PHP or java base language.

REFERENCES

1. Xiaowei Li and Yuan Xue, "A survey on Web Application Security" 2012 Institute of Electrical and Electronics Engineers(IEEE)
2. OWASP Vulnerability Top ten, Retrieved on February,2017 from https://www.owasp.org/index.php/Category:Vulnerability
3. The Open Web Application Security Project Book, b OWASP Foundation, https://www.owasp.org/images/f/f8/OWASP-Top-10-2013
4. "VULNERABILITY LIKELIHOOD BY CLASS" , web security statistics report 2016[online] Retrieved on February,2017 from https://info.whitehatsec.com/rs/675-YBI-674/images/WH-2016-Stats-Report-FINAL.pdf
5. Tony Hunt, "OWASP Top ten for .net developers", by plural sight publication.
6. Rajyalakshmi A.G, "broken authentication and session management" Retrieved on March 2017,from http://www.triadsquare.com/broken-authentication-and-session-management
7. Huyam AL-Amro and Eyas El-Qawasmeh, "Security Vulnerabilities and Leaks in ASP.NET Websites*",* 2012 International Conference on E-Learning and E-Technologies in Education (ICEEE).
8. Paul Gries and Jennifer Campbell, Design Algorithm, Practical programming 2nd edition-A Introduction to computer science using python 3, 2013 The Pragmatic Programmers, LLC.
9. Paul Gries and Jennifer Campbell, Reading and writing files, Practical programming 2nd edition- A Introduction to computer science using python 3, 2013 The Pragmatic Programmers, LLC.
10. ASP.NET Web Forms page code model, https://msdn.microsoft.com/en-us/library/015103yb.aspx
11. B. Sullivan, "Top 10 security vulnerabilities in .NET configuration files", Retrieved on February, 2017 from [Online] http://www.devx.com/dotnet/Article/32493/1954.
12. SESSION Identifiers, Retrieved on March, 2017 from [Online] https://msdn,microsoft.com/en-us/libary/ms178582.aspx.