International Journal of Automation, Control and Intelligent Systems
Vol. 1, No. 1, 2015, pp. 16-20
http://www.aiscience.org/journal/ijacis

KEY-INTERFACE: Reducing Workload of Server Due to Password Comparison

Sheikh Muhammad Saqib*, Abdur Rashid Khan, Hamid Masood Khan

Institute of Computing and information Technology, Gomal University, Dera Ismail Khan, Pakistan

Abstract

Password provides security from malicious users. When a user opens web page based profile such as email, online resume etc, then password is matched with already stored information on server and is stored in session variables. Now user can visit different web pages related to that particular profile. Loading of each and every new web page requires authentication. Instead of entering user name and password again and again, previously entered password is stored in session variables for further comparisons. On loading of each new web page, session variables are compared with server side information. There are number of users and each user profile has number of related web pages, so server remains busy all the time in mating of username and password. To reduce this workload of server and to ensure security related to session comparison with server information, authors suggested a proposed methodology KEY-INTERFACE to handle such issues. It provides only single time comparison with server side information resulting workload reduction.

Keywords

Password, Security Issues, Session Variables, User Authentication

Received: March 30, 2015 / Accepted: April 19, 2015 / Published online: May 22, 2015

1. Introduction

Password is a set of secret characters or words utilized to gain access to a computer, web page, network resource, or data. Passwords help ensure that computers or data can only be accessed by those who have been granted the right to view or access them [5].

User authentication can be done through biometric i.e. through biological behavioural automatic recognition of individuals [6]. In [7] author has tried to identify the level of authentication through biometric technology. Different applications has been made on analogy of biometric, [8] has developed a monitoring system for attendance using biometric technology and [9] has made a security system on finger vein based biometric security system. These techniques is very useful for the first time user authentication but still authentication on loading of each profile based webpage will be done through session variable.

In web pages for user authentication, there are session variables which have lot of workload. Session is a variable that is tied to a message for its entire lifecycle [10]. In order to make a session variable persistent between two runs of an application, you must store the session variable in an HTTP cookie. The *cookie* is a text-only string containing the name-value pair and is saved into the memory of your browser [11].

When user login on authentication page, then password and user name are store in session variables. After loading of first page, each new web page can be loaded from first page then this temporarily stored user name and password are compared with already permanent stored user name and password at server side. There is no other way for session comparison.

A computer intruder is something that invades your computer without permission. Sometimes you don't even know that an interloper has arrived. In other cases, you may think you are

*Corresponding author
E-mail address: saqibsheikh4@hotmail.com (S. M. Saqib)

giving permission for one thing such as accepting a game download, but are actually opening your computer to attack. The Intruder may apply different operations on session variables when invoke to server [1] [2].

The Application and Session objects can be used to store values that are global either to a particular user (the Session) or to all users (the Application). Within the on start events, we can initialize these variables. We can also store new variables, or change existing values, in the code inside any other ASP page [3]. Researchers involved in issues related to security through password. Yun Huang [4] suggested a method for one time password, because again and again password traffic provides security threats.

Authentication of password stored in session on each relevant webpage may cause a bottleneck on server. And Server is already busy with different client as shown in Fig-1. Authors have suggested a technique KEY-INTERFACE, which can reduce bottleneck of server and also provides many other advantages to user.

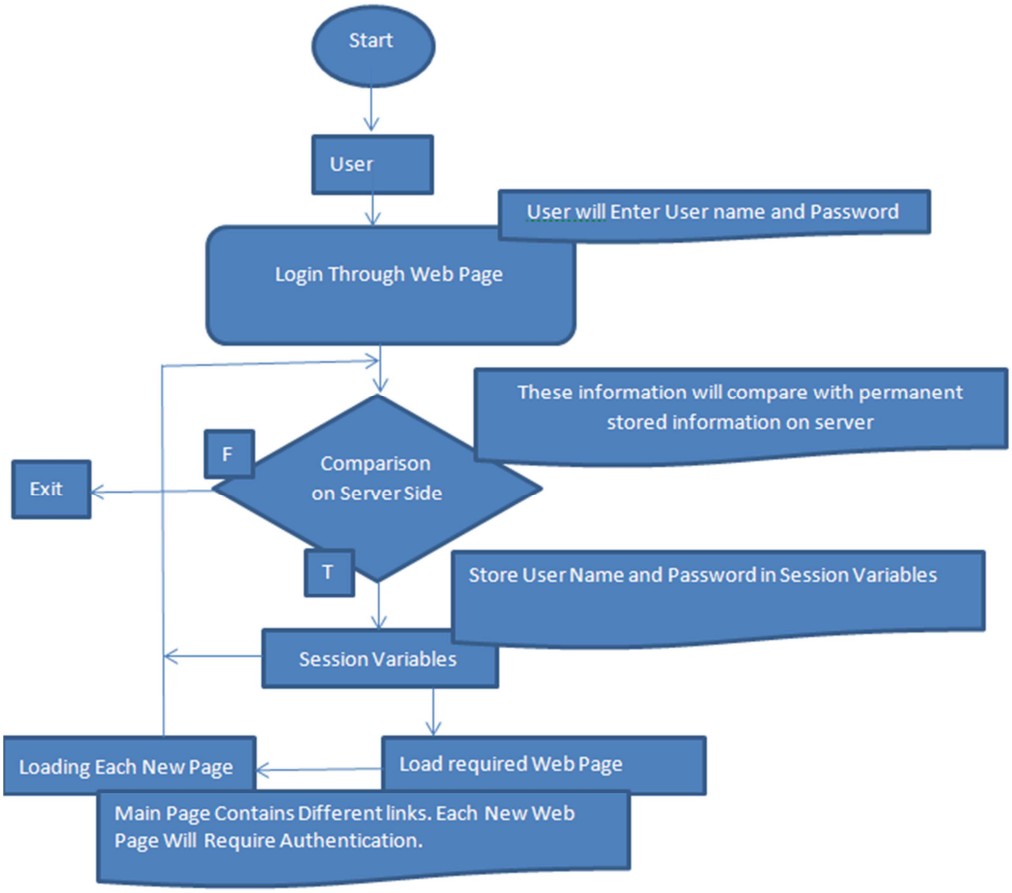

Fig. 1. Mechanism for User Authentication Considering Server as Reliable Source.

2. Proposed Solution for Reducing Server Workload

According to proposed solution authentication process can be completed in to two ways. As first time user will enter user name and password in input box through login page, first of all these user name and password should be matched with server side stored information. If authentication is done then these user name and password will be stored in session and required web page will be opened. Now there are different links for same user and each web page requires authentication. Instead of entering user name and password on each new page, session variables are used for such authentication. Sessions variables are compared with server side stored information and then display the next page. This is mandatory because if user authentication on each new page is not done, then anyone can copy the URL of new page and can paste in any web browser. In this way privacy cannot be maintained by user profile. That is why authentication on each new webpage is very necessary. Although user enters only one time user name and password through login page, while overhead of server is increased. This increased overhead can be resolved by using KEY-INTERFACE.

KEY-INTERFACE:

Session variables requires server side stored information for authentication. First time login form can use server side

information comparison while comparison on each new page can be done through key interface as shown in Fig-2.

Key interface is just like a component on main page contains only two input boxes, one for user name and another for password as shown in Fig. 2. User can enter and remove user name and password at any time. When user will remove name and password, then new webpage will not be loaded.

After login and opening of main page of user profile, each new page requires comparison of session variables with reliable source. This reliable source is server or user own. Server side burden can be replaced through user side. User can enter user name and password on key-interface on top of main page. Session will be compared through key-interface instead of server side. If user leaves the system for some minutes, he can erase the user name and password from key-interface. Now other user can not open any page. When other user will try to open the page then session variable will compare with key interface and due to empty key-interface, comparison becomes false and new page cannot be opened. When actual user come back and will enter user name and password on key-interface, then whole pages will work normally. Hence Existing flow chart of Session and server side comparison can be replaced as shown in Fig. 3.

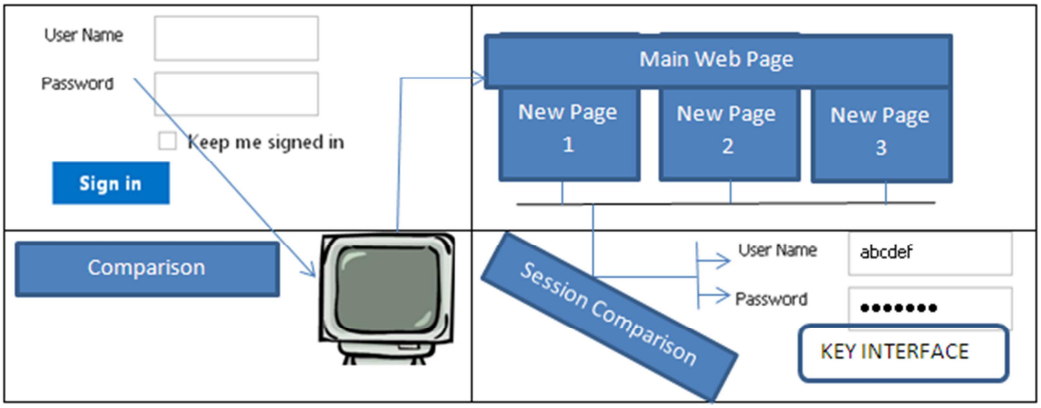

Fig. 2. Website with Embedded key-interface.

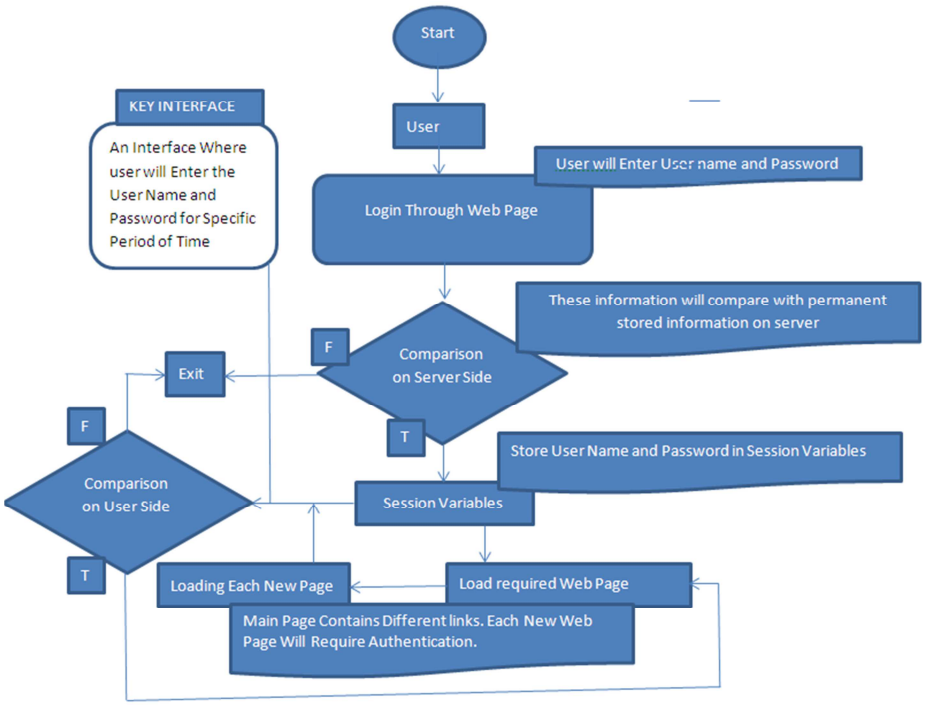

Fig. 3. Mechanism for User Authentication Considering User As Reliable Source

3. Results and Conclusion

Different authentication websites such as (Emails, Social network sites etc) requires session variable for authenticating each new page relevant to particular profile. This authentication is done between client and server through session variables at load of each new page. Suppose a user

profile has 50 of web pages and he may open all of these web pages then session variables will compare with server side stored information at least50 times. If there are lot of users then probability of such comparison are so high. Now we are finding the differences of session variable comparison through key-interface and without key-interface.

Suppose there are 10 users and each user profile consists of 50 web pages. When there is only one user then chance of server side comparison is 50, when there are 2 users then chance of server side comparison is 100 and when there are 10 users then chance of server side comparison is 500 as shown in Table1.

Table 1. Server Side Session Variables Comparison of 10 Users with 50 Web Pages for each User.

Total Users	Web Pages for Each users	Server Side Comparison
1	50	50
2	50	100
3	50	150
4	50	200
5	50	250
6	50	300
7	50	350
8	50	400
9	50	450
10	50	500

From Fig. 4, it is clear that when number of users is increased then workload of server is also increased.

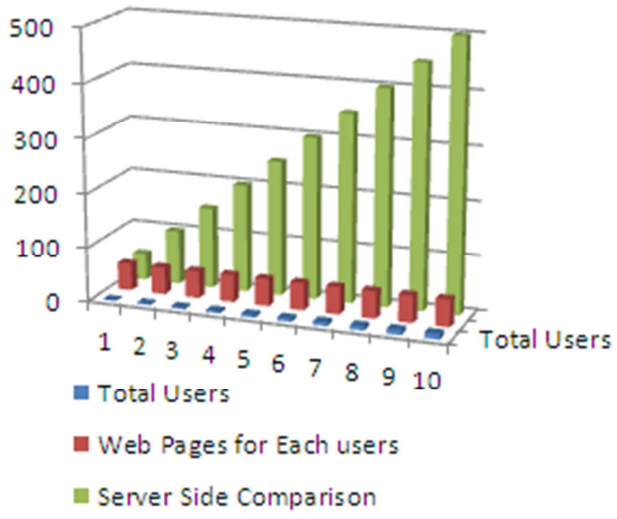

Fig. 4. Workload of Server without key-interface.

Key-interface provide a proposed solution, that server side comparison will be made only single time and remaining comparison will be done on client side. It means most of the work of server can be divided in to different clients as shown in Table2. As each client will relate to different machine then efficiency of each client could be maximum.

Table 2. Server Side and Client Side Session Variables Comparison of 10 Users with 50 Web Pages for each User.

Total users	Web Pages for Each user	Server Side Comparison	Client Side Comparison
1	50	1	49
2	50	1	49
3	50	1	49
4	50	1	49
5	50	1	49
6	50	1	49
7	50	1	49
8	50	1	49
9	50	1	49
10	50	1	49

Fig. 5 is showing that, there is little bit work of server side due to 1st time login and remaining work is distributed between different clients.

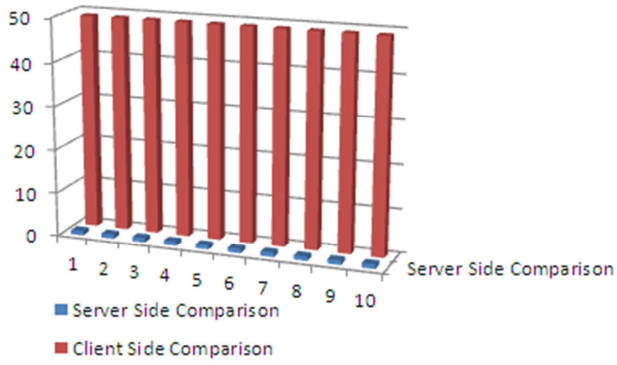

Fig. 5. Workload of Server using key-interface.

It is concluded from Table-1 and Table-2 that, when there are 10 users and with 50 web pages profile, number of comparison of user name and password through session variable without key-interface are 500 while through key-interface server side comparison are only 10.So, 490 comparisons will be divided into 10 clients. Through key-interface each user will perform 49 comparisons on his own client machine. Hence if there are 10 users, then comparison without key-interface and with key-interface ratios is 50to 1.

Also when user gets logged into his profile, and leaves the computer in hurry for a while, he should have to logout from his profile. Whereas upon incorporation of KEY-INTERFACE, user can only remove one or two characters of his password from KEY-INTERFACE, to lock his whole profile.

References

[1] PedroAdão, Protocolinsecuritywithafinitenumberofsessionsandacost-sensitiveguessingintruderisNP-complete, Theoretical Computer Science 538 (2014) 2–15, Elsevier

[2] Computer Intruders: Detection and Prevention, http://eduscapes.com/tap/topic121.htm

[3] Application and Session Variables, http://msdn.microsoft.com/en-us/library/ms952597.aspx

[4] Yun Huang1, Zheng Huang2, Haoran Zhao3, Xuejia Lai4, A new One-time Password Method, International Conference on Electronic Engineering and Computer Science, 2013

[5] Computer Hope, http://www.computerhope.com/jargon/p/password.htm visit on 9/6/2014

[6] Biometric authentication, Anil K. Jain (2008), Scholarpedia visit on date 13-4-2015http://www.scholarpedia.org/article/Biometric_authentication.

[7] Michael Zimmerman, BIOMETRICS AND USER AUTHENTICATION, SANS Institute 2002

[8] Seema Rao, Prof.K.J.Satoa, An Attendance Monitoring System Using Biometrics Authentication, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013

[9] Jose Anand, T. G. Arul Flora, Anu Susan Philip, FINGER-VEIN BASED BIOMETRIC SECURITY SYSTEM, International Journal of Research in Engineering and Technology eISSN: 2319-1163 | pISSN: 2321-7308, Volume: 02 Issue: 12 | Dec-2013

[10] Session Variable Transformer Reference, visit on date: 13-4-15, http://www.mulesoft.org/documentation/display/current/Session+Variable+Transformer+Reference

[11] Session Variables and Cookies, visit on date: 12-4-2015, http://www-01.ibm.com/support/knowledgecenter/SSBJG3_2.5.0/com.ibm.gen_appug.doc/c_gas_session_variables_cookies_001.htm?cp=SSBJG3_2.5.0%2F2-6-1-0-4.