Explore and Exploit Security Flaws in Web Applications for Implementing Efficient Security Provision Techniques

Kanwal Yousaf, Arta Iftikhar , and Ali Javed, Ali Tahir

***Abstract*—Security of any web-application is very important due to its excessive use in daily routine life (such as business, education, health etc). The advancement of technology raises a question mark to the security of any web-application. Serious attacks on web-application can cause an exposure of sensitive data or provide access to the system on which an application is stored and maintained. This paper aims to identify 3-tier defense mechanism of any web-application. Defense mechanism acts as a baseline for the classification of all possible flaws which make a web-application inaccessible. This paper also refers to the exploitation of flaws, found during classification phase, in order to make uncomplicated and resourceful techniques for the avoidance of security threats. These techniques develop reliability and trust-ability on web-applications.**

***Index Terms*—Web-application, security, defense mechanism, security-provisioning techniques**

I. Introduction

With the arrival of an internet, World Wide Web (www) was comprised of loads of websites. These websites were repositories of information. Information can be accessed by users through different web-browsers as shown in Fig. 1. In which user (such as personal home computer, organizations etc) accesses different websites with the help of web-browser and requests for information by using any web-application. These web-applications are interrelated with central server. This central server not only stores important information about the web-application (it stores the database of users or other record) but also provide communication with another user through web-applications.

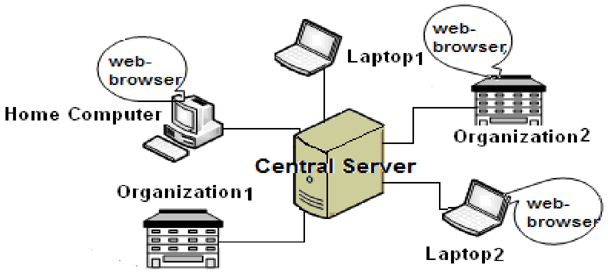


Fig. 1. Web-applications and central server architecture.

Initially, websites were not focused on authentication of users. Each user was treated in same manner. Attackers got an easy way to access information and modify it. But this accessibility was not that much beneficial. Because information stored in central server was already public and can be access by any user.

With the passage of time internet becomes more dynamic and unpredictable. Web-applications become an important part of diverse areas like:

1) Shopping (Amazon etc)
2) Social Networking (Facebook, twitter etc)
3) Electronic-Mail (yahoo mail, gmail, hotmail etc)
4) Web-Search (Google, ask etc)
5) Information Gathering (Wikipedia)
6) Blogs (word-press) etc.

These web-applications are providing a facility of financial transactions, login's, support of registration and accessibility to the web contents. The contents presented to the users are generated dynamically. So these contents should be well-protected and secure as nobody wants to use an application that have higher probability of an information leakage. Attackers or malicious users access that important information and affect user's possessions [1]. Serious attacks against web-applications lead to an exposure of important data and helps in gaining an un-restricted access to back-end system (local or main server).

As recent studies prove that attacks on web-application becomes more popular during last 5 years [2]. From this analysis, it has been identified that one of the potential reasons to security flaws relies on the low quality of web-application's security requirements and thus in its implementation, validation and different security development life cycle phases [3]. It's very important to understand all these attacks and its operational mechanism. The operation mechanism understandability will open a path of creating an effective defense strategy. Develop a defense mechanism that can help in distinct web-application's security. Securing web-applications can help in quality improvement of web-application [2].

II. Literature Review

Dalai and Jena [1] proposed that traditional SDLC (software development life cycle) is not an efficient technique that could be used for the security of web-applications. The security of internet applications needs some effective techniques and strong cryptographic patterns. They highlighted different security patterns for technical, configuration and security vulnerabilities of web-applications. But no specific techniques are being identified according to different scenarios of web.

Rezgui, Bouguettaya and Eltoweissy [4] suggested different facts, challenges and solutions of web-privacy. They clearly focused on web security from user's perspective. They categorized privacy defense into technology and

Manuscript received February 12, 2012; revised April 26, 2012

Authors are with the Department of Software Engineering University of Engineering and Technology Taxila, Pakistan (e-mail: ali.javed@uettaxila.edu.pk).

regulation-based solutions. Technology involves client-server communication while regulation-enabled solution was based on self and mandatory regulations. This solution was not particularly related to the challenges which are being faced by web-applications now-a-days. Secondly, these proposed solutions raise some issues in monitoring, information storage, information disclosure and its usage.

Palmer [5] explained about security of session cookies (tokens) generated by web-applications. He explained few attacking techniques on session tokens. He proposed an alternative, to the session cookie (token), that is "URL re-writing". Merge information on URL of web-application but it is insecure as it can cause breaches of web-application's security.

Kamal Kumar and Sandeep Jain [6] provided an authenticated technique against attack on web-applications i.e. SQL injection.[Structure Query language]. SQL injections affect an access level of web-applications. They proposed that SQL analyzer should be present between client and server to check all the requested data and web-application's response.

Fong, Gaucher, Okun, and E. Black [7] proposed combination of test-cases for web-applications scanner. A Web-application scanner is a tool used to check the security vulnerabilities of web-application. They explained about the defense level for different security attacks. They categorized the defense mechanism for SQL injections in to six levels. Starting from level 0 (No filtering of SQL query parameters) to level 5 (combination of previous levels and prepared statements).

III. Defense (Security) Mechanism of Web-Applications

In last few years, security of web-applications becomes very significant due to intense client-server communication [8]. So defense (security) mechanism within any web-applications is divided into three inter-related phases. This 3-tier defense mechanism can be represented as Fig. 2 [9]:

1) Authentication Phase
2) Session Management Phase
3) Data Access-Control Phase

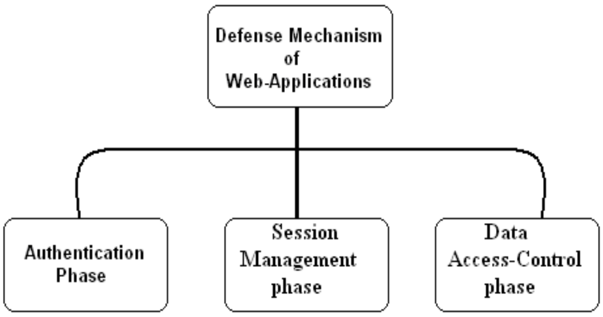

Defense Mechanism of Web-Applications
Authentication Phase
Session Management phase
Data Access-Control phase

Fig. 2. Web-application's 3-tier defense mechanism.

A. Authentication Phase

This phase is a frontline of defense against un-authorized access. It is basically a log-in page where verification of particular user's account has been made. User need to fill two fields, as shown in Fig. 3, i.e. user-name and password so that he/she can authenticate him/herself for particular web-application.

Fig. 3. Authentication phase (log-in page).

This application compares the details entered by user with information stored in local or main server. If information is correct then provide access to the application. If information is incorrect then don't allow the user to break into an application. In real-world, authentication phase is the weakest connection, which facilitates an attacker to gain an un-authorized access.

B. Session Management Phase

Session Management is a technique used by the developer of web-applications. Firstly, an authentication of user has been made by comparing data from the web-server. For making next HTTP request shouldn't allow the web-server to ask that person again for his/her log-in information. This phase identifies a user across variant requests and to handle the data that it gathers about the position of user's interaction with the web-application [5]. Gathered data is stored in the form of tokens. Web server is storage of session information. Each session has a distinct session identifier (session ID). The ID has been generated as a result of first authentication or request from client (user). Session data contains important information such as user name, password, account details etc.

C. Data Access-Control Phase

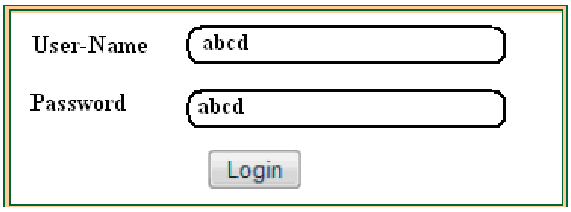
In this phase, two main things are being processed: 1) Access-control of variant users 2) Data storage accessibility. This phase is used to verify user's identity and check that particular user has rights to access information from web-application [9].

IV. Security Mechanism Flaws of Web-applications

Although web-applications have different security distribution levels still there are few missing entities in each tier of defense mechanism. The major reason is dynamicity of internet. That's why now-a-days web-applications are facing different security challenges and flaws. Some of the flaws in basic defense mechanism of web-applications are:

A. Authentication Phase

***Ghastly Passwords**:* Sometimes user selects such type of passwords that can easily be cracked by malicious users. Passwords such as same as username or common names or using minimum length passwords. This flaw can provide an un-authorized access to the user's account. In first scenario as shown in Fig. 4, user is having an account "abcd" and set his/her password same as user-name.

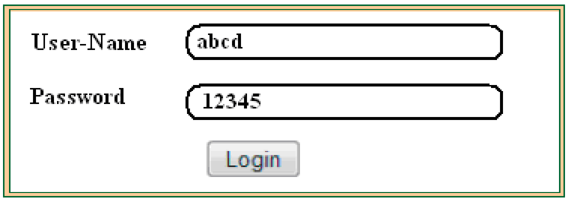


User-Name abcd

Password abcd

Fig. 4. Selection of password same as user name.

While in other scenario, as shown in Fig. 5, user is setting most common digits-based-password such as “12345”.



User-Name abcd

Password 12345

Fig. 5. Selection of simple password.

Brute-Force Logins: Web-applications are providing un-limited log-in attempts to the user. This technique can facilitate an attacker to guess passwords with the same user-name.

Insecure Storage of Record: User’s record is stored insecurely within the database of web-application (stored in central server) e.g. passwords are stored in textual format or weak encryption technique is applied in database etc

B. Session Management Phase

Defect in Token Generation: In few web-applications the tokens are generated in an unsafe pattern due to weak encryption techniques. It enables an attacker to interpret token that have been issued to other users [10]. During each session a unique session token or identifier is allocated to each user. HTTP based applications pass session tokens between client (user) and server. Some of the tokens contain important information like account username, user’s IP address, e-mail ID etc. In this case, an attacker can simply apply any technique to increment his assigned token in order to switch his control to other user’s session [11].

C. Data Access-Control Phase

SQL injection: Data Access-Control is a vital defense mechanism within the application because it is responsible for making key results. When it is defective, an attacker can easily accesses the entire application and take control of administrative functionality. This leads to the misusage of sensitive data. Fault injections provide an un-registered access to the database of web-applications [12].

Web-applications are commonly construct by using different SQL (structured query language) statements and incorporated user-entered information. SQL injections are basically wrong insertions or wrong entries into SQL queries. Queries like insert, select, delete, update etc. These SQL injections are the weakness of any web-application [13].

V. Security Provisioning Techniques in Web-Application

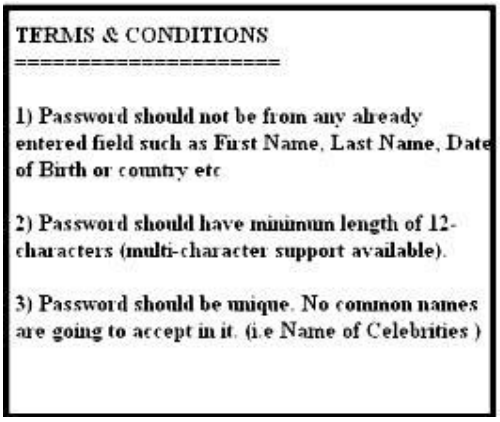
Implementing secure techniques in web-application is different from traditional application because web-applications depend upon 3-tier defense mechanism. Major security flaws in web-applications have been discussed in previous section IV. This section is proposing to overcome these security flaws.

A. Authentication Phase

1) Ghastly Passwords Prevention

Ghastly passwords can be avoided by two ways:

1) In web-application’s sign-up page, User’s Sign-up page is divided into three steps as shown in Fig. 7. First step involves personal information of user. Second step is verification of account name. If step 2 verifies then move to the final step 3 of selecting password. User just clicks on button “Terms and Conditions” mentioned in step 2 in order to view policies for selecting password. The example of policies is mentioned in Fig. 6.



TERMS & CONDITIONS
=====================

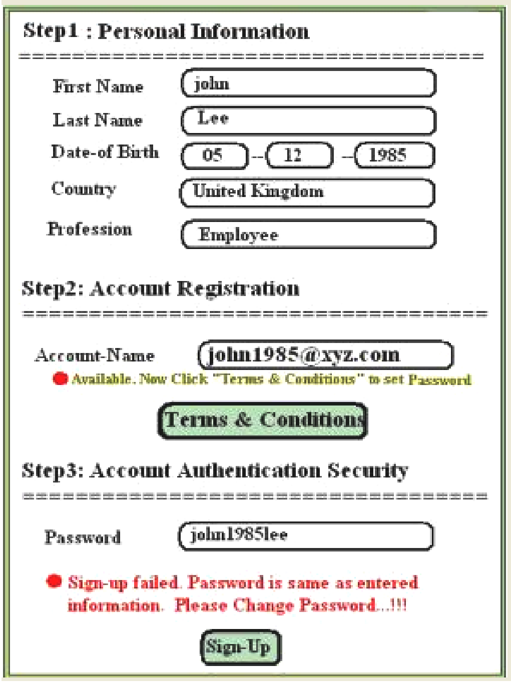
1) Password should not be from any already entered field such as First Name, Last Name, Date of Birth or country etc

2) Password should have minimum length of 12-characters (multi-character support available).

3) Password should be unique. No common names are going to accept in it. (i.e Name of Celebrities)

Fig. 6. Terms and conditions for web-application’s sign-up page.

Suppose if user is selecting password from the combination of already entered data (in case of Fig. 7, already entered record is first name, last name, Date of Birth, Country and Profession) then web-application compares the password with the record. As the first point of “terms and conditions” states that user cannot have password from already entered data, so web-application rejects the user input and display an error message.



Step1 : Personal Information
====================================

First Name john

Last Name Lee

Date-of Birth 05 -- 12 -- 1985

Country United Kingdom

Profession Employee

Step2: Account Registration
====================================

Account-Name john1985@xyz.com

● Available. Now Click "Terms & Conditions" to set Password

Step3: Account Authentication Security
====================================

Password john1985lee

● Sign-up failed. Password is same as entered information. Please Change Password...!!!

Fig. 7. Example of ghastly password entrance.

2) Web-applications are being developed in such fashion that suggests the minimum length of passwords would be strong length password's specification of currently running web-applications. For example now a day's if any web-application is offering 6 characters as its minimum length for selecting the password, it should be of 10 or 12 characters for the avoidance of security attack. Web-applications also provide multi-characters support. And web-applications also prevent the users from selecting common passwords i.e. Celebrities names etc [9].

2) Alternative to Brute-Force Logins

To avoid the attack of brute-force login, web-applications should provide 3 login attempts as shown in Fig. 8. The flow diagram shows that:

1) If user fails to enter correct password at first attempt then provide second attempt by verifying account's user through CAPTCHA technique. If user fails in second stage then account is locked temporarily. To unlock that account actual user should enter correct password or alternative account's password. The alternative account-name has been already mentioned by a user during Sign-Up phase of that web-application. If correct password is entered then access an account otherwise permanently lock the account. Permanently locked account can be unlocked by providing valid evidence etc.

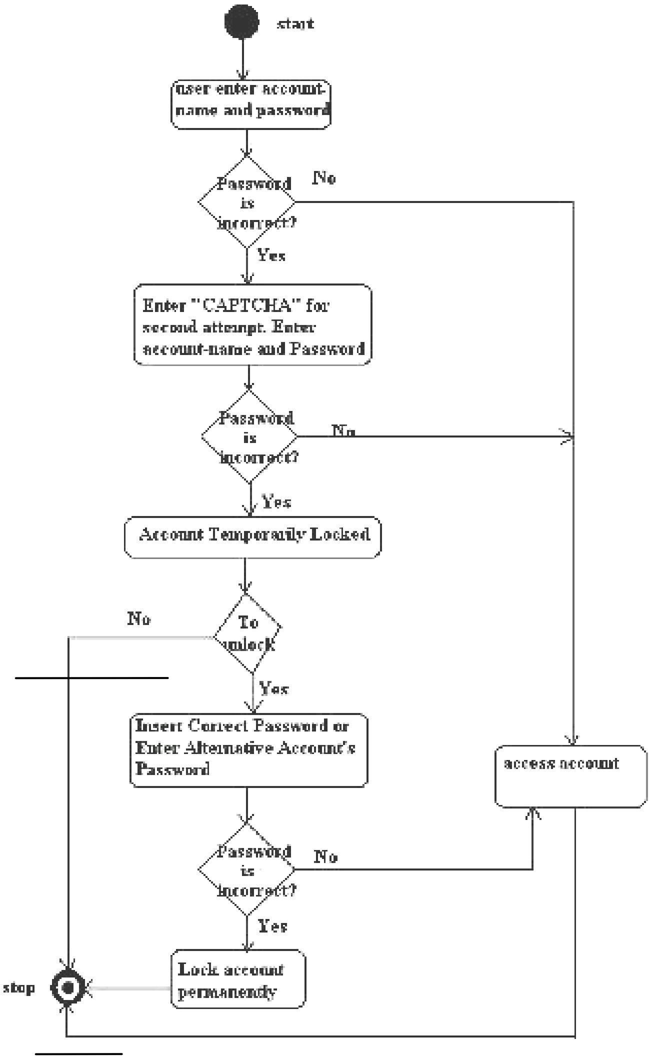

start
user enter account-name and password
Password is incorrect?
No
Yes
Enter "CAPTCHA" for second attempt. Enter account-name and Password
Password is incorrect?
No
Yes
Account Temporarily Locked
To unlock
No
Yes
Insert Correct Password or Enter Alternative Account's Password
access account
Password is incorrect?
No
Yes
stop
Lock account permanently

Fig. 8. Solution for brute-force login attempts.

3) Providence of Secure Storage for Record

To prevent record from intruder's attack apply effective encryption technique (triple data encryption standard, key management model etc) [14] where record of web-applications are being stored (local or main server). Periodic changes in encryption techniques will also overcome this security issue.

B. Session Management Phase

1) Overcome Defect in Token Generation

The token generation defect can be removed according to the proposed solution. This solution involves 2-phase approach.

- **First Phase**

In first phase, initially user checks tokens of any web-application at the start by simply login from his/her account. Then modify specific values of token (such as tokenID or session token identifier). Submit this token and check its validation. If token validates it means that the user enters into someone else account, than switch to the second phase. Otherwise keep on obtaining the tokens and change these token values. This phase is helpful in detecting the session management flaw of web-application where weak tokens can be detected by randomizing the token values.

- **Second Phase**

As mentioned in Fig. 9, this phase relates when user made a successful attempt to enter into someone's account through token ID. Take the token and divide into smaller sub-tokens. Information is filtered out from sub-tokens. Check if the token contains relevant or important data. If data is relevant then insert large set of token values between original token ID [5] after that apply random code generation technique into the token value. The resultant token is now encrypted by using strong cryptographic techniques [14].

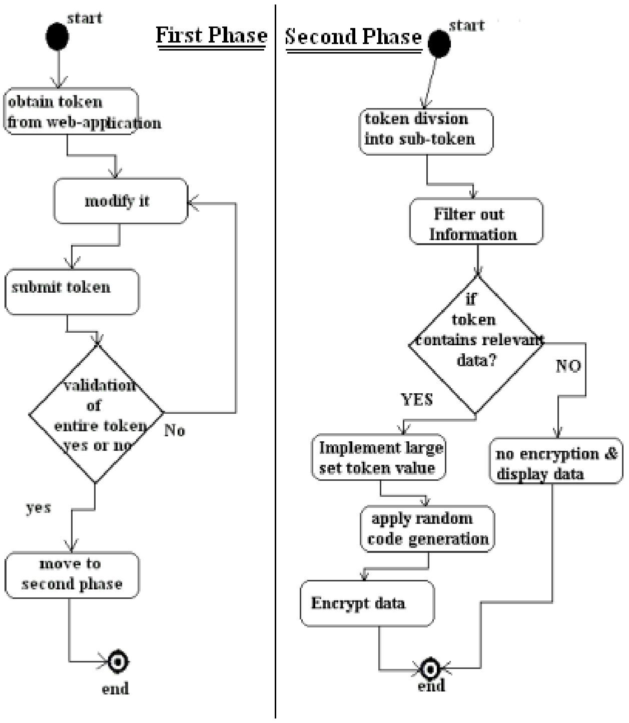

start
First Phase
Second Phase
start
obtain token from web-application
modify it
submit token
validation of entire token yes or no
No
yes
move to second phase
end
token divsion into sub-token
Filter out Information
if token contains relevant data?
YES
NO
Implement large set token value
no encryption & display data
apply random code generation
Encrypt data
end

Fig. 9. Two-phase token-defect removal method.

C. Data Access-Control Phase

1) SQL Injections Problem Removal through Ssql and Csql Analyzer

Defense mechanism of applications is inter-related with one another. Initially verification of authentication and session management phase occurs. So the security of data access-control is maintained. These can also reduce the probability of SQL attacks into any web-applications [15]. Otherwise use SQL analyzer at both ends i.e. client and server side. Before that only one SQL analyzer was interfaced between client and server [10]. This analyzer processes the request, if any malicious user neutralizes analyzer, results in accessibility to server-side database.

To avoid this attack we suggest the query analyzer at client and server side represented as CSQL (client) and SSQL (server) analyzer as mentioned in Fig. 10. Initially, User's request is forwarded to the CSQL analyzer that process the basic information entered by user (i.e. Username, Password).

If CSQL analyzer verifies inserted record forward it to the SSQL analyzer. SSQL analyzer will be more efficient than CSQL analyzer. SSQL analyzer will firstly verify the user's entered input fields. If verified then log-in user account. Basically this technique is providing enhanced security against intruders.

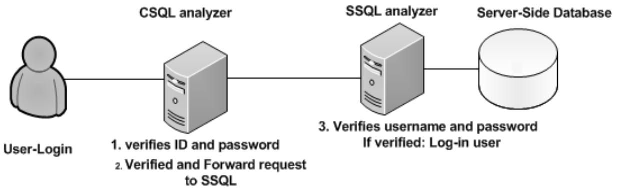

CSQL analyzer
SSQL analyzer
Server-Side Database
User-Login
1. verifies ID and password
2. Verified and Forward request to SSQL
3. Verifies username and password
If verified: Log-in user

Fig. 10. User log-in technique.

After the successful login, in next step SSQL analyzer will check requested information's level of accessibility to server-side database by CSQL as shown in Fig. 11. For example, if user is accessing more than 50% of record at a time, chances for an attack increase. SSQL analyzer will check the causes and effects of requested data to estimate the negative consequences on web-applications. If probability of negative consequences is exceeding a certain limit then block that particular user's account.

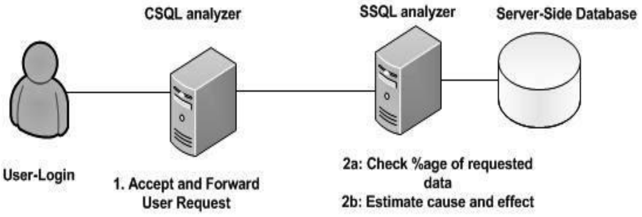

CSQL analyzer
SSQL analyzer
Server-Side Database
User-Login
1. Accept and Forward User Request
2a: Check %age of requested data
2b: Estimate cause and effect

Fig. 11. Processing of user request.

VI. Conclusion

In this paper, we highlighted major security defects in web-applications and proposed efficient strategies for providing security in application's 3-tier defense mechanism. These efficient security provisioning strategies protect application from the outside and inside attacks. By adopting our projected strategies in each defense mechanism phase, web-applications can be protected from the malicious users. The adoption of these strategies will enhance the security and will increase trust of users, concerned with security, on web-applications. The reliability, reliance and efficiency will play an important role in improvising the quality of web-applications. So quality of web-applications is clearly dependent on implementation of effective security mechanism in web-application.

Acknowledgment

We would like to thank our honorable supervisor Engr. Ali Javed for providing us with his useful suggestions. These suggestions helped us in the completion of this research paper.

References

[1] Asish Kumar Dalai and Sanjay Kumar Jena, "Evaluation of Web Application Security Risks and Secure Design Patterns," *ICCCS'11 February 12-14-2011, Copyright 2011 ACM 978-1-4503-0464-1/11/02*

[2] Rodrigo Elia Assad, Felipe Ferraz. Henrique Arcoverde, and Silvio Romero Lemos Meira, "Security Quality Assurance on Web Applications," *ICSEA 2011, The Sixth International Conference on Software Engineering Advances*

[3] Michael Howard and Steve Lipner, "The Security Development Life Cycle," Microsoft Press, May 2006, pp. 3-13.Available at: http://www.microsoft.com/MSPress/books/8753.aspx

[4] Abdelmounaam Rezgui, Athman Bouguettaya, and Mohamed Y. Eltoweissy, "Privacy on the Web: Facts, Challenges and Solutions," *IEEE Security and Privacy*, 2003

[5] Chris Palmer, "Secure Session Management with Cookies for Web Applications," *iSEC Partners, Inc*, September 10, 2008, Available at: https://www.isecpartners.com/

[6] Kamal Kumar and Sandeep Jain, "An Authentication Mechanism against SQL Injection on Web Platform," *International Journal of Engineering and Information Technology*, vol. 3, no. 1, Copyright© 2011, IJEIT 2011.

[7] Elizabeth Fong, Romain Gaucher, Vadim Okun, and Paul E. Black, "Building a Test Suite for Web Application Scanners."

[8] Ghulum Mustafa, Abid Ali Shah, Khadim Hussain Abid, and Amjad Ali, "A Strategy for testing web-based Software," *Information Technology Journal, 2007*

[9] Dafydd Stuttard and Marcus Pinto, "The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws," Second Edition, *Copyright © 2011 by Dafydd Stuttard and Marcus Pinto, Published by John Wiley and Sons, Inc., Indianapolis, Indiana*

[10] Mohamad Ibrahim Ladan, "Web Services: Security Challenges," *2011 International Conference of IEEE*

[11] Andrew Bortz and Adam Barth, Alexei Czeskis, "Origin Cookies: Session Integrity for Web Applications," *15th ACM Conference, 2011.*

[12] Yao-Wen Huang, Fang Yu, Christian Hang, Chung-Hung Tsai, D. T. Lee, and Sy-Yen Kuo, "Securing Web Application Code by Static Analysis and Runtime Protection," *WWW 2004*, New York, New York, USA. ACM 1-58113-844-X/04/0005, pp. 17–22, May, 2004.

[13] Yao-Wen Huang, Shih-Kun Huang, and Tsung-Po Lin,"Web Application Security Assessment by Fault Injection and Behavior Monitoring," *Copyright is held by the author/owner(s). WWW 2003,* Budapest, Hungary. ACM 1-58113-680-3/03/0005, pp. 20-24, May. 2003.

[14] Anne Voluntas Dei Massah Kayem "Adaptive Cryptographic Access Control for Dynamic Data Sharing Environments," *a thesis for the degree of Doctor of Philosophy Queen's University Kingston*, Ontario, Canada, 2008.

[15] Dimitris Mitropoulos, Vassilios Karakoidas, and Diomidis Spinellis, "Fortifying Applications Against XPath Injection Attacks," In A. Poulymenakou, N. Pouloudi, and K. Pramatari, editors, *MCIS 2009: 4th Mediterranean Conference on Information Systems,* pp. 1169–1179, September 2009.

Engr Kanwal Yousaf is MSc Scholar in Department of Software Engineering at University of Engineering and Technology, Taxila. She completed her Bachelors degree in Software Engineering from UET, Taxila in 2010. She is currently working on Web 2.0 based E-learning by using social media. Her area of interest is Software Quality Assurance, Internet application development and Wireless networks.

Engr. Arta Iftikhar is MSc Scholar in Department of Software Engineering at University of Engineering and Technology Taxila. She completed her Bachelor's degree in Software Engineering from University of Engineering and Technology Taxila in 2011.

Engr. Ali Javed has been an Assistant Professor since March 2012 in the Department of Software Engineering, University of Engineering and Technology Taxila, Pakistan. He is a PhD Scholar in the Department of Computer Engineering at university of Engineering and Technology Taxila, Pakistan. He accomplished his M.Sc in Computer engineering from university of Engineering and Technology Taxila, Pakistan in February, 2010. His areas of interest are Video Summarization, Digital Image Processing, Computer vision, Software Quality Assurance, Software testing and Software Requirements Analysis.

Engr. Ali Tahir is serving as a Lecturer at Jazan University, Saudia Arabia. He did his M.Sc Computer Engineering from UET TAXILA in 2010. He did his B.Sc Computer Engineering from UET TAXILA in 2005. His areas of interest are digital image processing, computer vision, network security and Databases.