Volume 5, Issue 3
June 2016

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

A Comprehensive Analysis on Web Security and Loopholes

Ali Raza Bhangwar[1,*], Abdul Wahid Memon[1], Abdul Sattar Saand[2], Adnan Ahmed[1]

[1]Department of Computer System Engineering, Quaid-e-Awam University of Engineering, Science and Technology, Nawabshah.
[2]Department of Electrical Engineering, Quaid-e-Awam University of Engineering, Science and Technology, Nawabshah.
Email: [1*]arbaloch@quest.edu.pk

ABSTRACT

Pakistan being one of the largest countries in the world for providing internet service, further 3G and 4G services have revolutionized the internet usage, the organizations from commercial to non-commercial and from government to social facilitates their customers/visitors by providing updated information all the time. The detail study of these websites has proved that most of the websites among these organizations possess web breaches. Little or no security measures have been taken while developing these websites. In this research we provides detail analysis by using penetration testing tools to identify web-breaches and suggest security measures for identified vulnerabilities while developing websites.

Keywords: *web vulnerabilities, SQL injection, cross site scripting*

I. INTRODUCTION

Many organizations in Pakistan are now using Internet to provide the required information to the public and to expand their publicity and growth. As Internet has become the backbone of the information exchange in today's world and offers new opportunities to the businesses and individuals, its utilization in Pakistan is also increasing on the daily basis.

The world of Internet is significantly improved with the arrival of web 2.0 and AJAX, where the user could interact in real time. At the same time, they are susceptible to the several security vulnerabilities and have opened the new windows for the hackers.

In order to minimize the web breaches, web-developers must be as skilled as the web-attackers. With the fast growth of Internet and the swift transition from the traditional to the web based approach, many web developers significantly ignore web application's security issues and mainly focus on developing attractive websites for the customers. In result, they develop web applications with variety of vulnerabilities, some of them are enlisted in [10] and [11]. In result of that, the malicious users always try to take benefit and exploit the susceptibilities.

In developed countries several organizations provide awareness by issuing a list of greatest vulnerabilities on yearly basis. But in Pakistan very less attention has been paid to the web security, its vulnerabilities and implications. This paper is therefore an attempt to provide awareness and susceptibilities regarding the web security in Pakistan and their tentative solutions. Auditing the websites for security breaches penetration testing is the well-known process. In the process after recognizing the susceptibilities, penetration tester provides necessary solutions to the identified vulnerabilities. This work basically uses the penetration technique to identify and then propose solution to the several web vulnerabilities.

The rest of the paper is organized as follows: Section 2 provides the detail of the penetration techniques and Section 3 presents the research methodology that has been used to identify the security vulnerabilities. Section 4 presents the different vulnerabilities that have been identified for the several websites in Pakistan. Section 5 shows the related work in the domain whereas the last Section 6 concludes the overall research work and presents some proposed recommendations.

II. SECURITY PREVENTION TECHNIQUES IN DETAIL

This section briefs the web vulnerabilities; there detail discussion is given in [1], [3] and [9]. Here we mainly focus on the prevention techniques for those security concerns. The major web vulnerabilities and their prevention techniques are given below:

SQL Injection

The SQL injection is most popular and conversed vulnerability. Various solutions are suggested to the same vulnerability [2], [3] and [4]. The code level defense is focused in this paper. This is most common and efficient technique since all other defensive techniques need to use this technique at the end. The root cause of the SQL Injection attack is the automatic SQL query establishing, the dynamic SQL queries forming can be avoided by using the parameterized statements [4].

Parameterized Statement

Parameterized Query is one of the solutions to avoid SQL injection attack, although injection attack can be avoided by some other techniques as well, using prepared statements is the preferred one [8]. Using parameterized queries the developer prepares all the SQL queries in advanced and pass all the data entered by user as parameters. In this way the attacker cannot change the query for manipulating his intents. Figure 1 [4] shows a vulnerable query if used instead of Parameterized Query can execute malicious code to harm the database.

```
Username = request("username")
Password = request("password")
Sql = "SELECT * FROM users WHERE Username='"+
username + "'AND password ='"+ Password + "'"
Result = Db.Execute(Sql)
If (Result) /*suceesful login */
```

Figure.1. Vulnerable SQL Query [4]

```
sqlConnection con = new SqlConnection
(ConnectionString);
string Sql = "SELECT * FROM users WHERE
username=@username" + "AND
password=@password";
cmd = new SqlCommand(sql, conn);
//Add parameters to SQL query
cmd.Parameters.add("@username",          // name
                  SqlDbType.NVarChar,    // data
type
                  16);                   //
length
Cmd.Parameters.Value("@username") = username; //
set parameters
Cmd.Parameters.Value("@password") = password; //
to supplied value
Reader = cmd.ExcuteReader( );
```

Figure.2. Parameterized SQL Query [4]

```
<asp:textbox id="username" runat="server"/>
<asp:RegularExpressionValidator id="usernameRegEx"
runat="server" ControlToValidate="user-name"
ErrorMessage="Username must contain 8-12 letters only."
ValidationExpression="^[a-zA-z]{8,12}$" />
```

Figure.3. Input validation on .NET [4]

The vulnerable query mentioned in Fig.1 can be used safely via using parameterized statement, as shown in Fig. 2 [4]. The parameterized statements provide a substitute for dynamic SQL building as well as checks for length and type of the input. If the entered inputs are correct in all aspect, like valid user name, password, length of the input and its type, then this data will be processed further otherwise it will be rejected at this stage.

Input Validation

In this process, the data received by the application will be checked against predefined set of inputs. If data is valid, process it further, application should consider entered data as invalid if it does not match with predefined set of input. Researchers have suggested different techniques for input validation [5], [6], and [7], but regular expressions are considered as the best way to sanitize inputs [7]. Figure 3 presents inputs validation in C# programming [4].

Stored Procedures

Like parameterized queries stored procedures are used to protect against SQL injection attack, the developer is required to develop SQL code with parameters in advance and store them in database, when required call them from application. As user has limited access to the database, cannot go beyond his limitation. Therefore if injection vulnerability is found and access permission is set properly, then the attacker cannot alter key data within the database. In some cases, it is observed that only Stored Procedures are not sufficient, stored procedure along with parameters should be practiced [16].

Output Encoding

In the process of output encoding, returning outputs by the application in reply of the request are handled safely. In such a situation where input data is to be processed for output, should be encoded, so that malevolent data may not be processed in its original form, otherwise there are fare chances that, output returned by application will leave space for SQL injection attack. The encoded data is necessary to be supplied to database. The encoding is SQL is accomplished as: sql = sql.Replace(" ' ", " ' ' ");.

Cross Site Scripting

It is most practical of all web attacks and allows introducing a malicious code into the website and make the visitor ultimate victim. The CERT highlighted the importance of XSS attack by publishing an adversary of security vulnerability and OWASP positioned this type attack at 2nd in 2010 list. Symantec also published this attack in 2008 [17] as most exploitable attack which affected 80% of the websites. There are two types of XSS vulnerability as given below.

Persistent Cross Site Scripting

Persistent XSS is sometimes termed as stored XSS, this type of vulnerability allows the date to be uploaded at the server, when other users visit the same website will be exploited. Figure 4 shows the Persistent Cross Site Scripting phenomenon [9] in which an attacker exploits stored XSS vulnerability and hijack user's session by submitting his malicious payload.

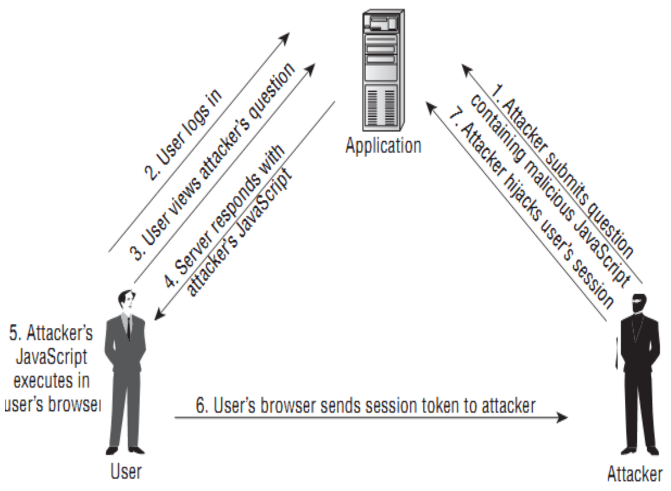

Application
1. Attacker submits question containing malicious JavaScript
2. User logs in
3. User views attacker's question
4. Server responds with attacker's JavaScript
5. Attacker's JavaScript executes in user's browser
6. User's browser sends session token to attacker
7. Attacker hijacks user's session
User
Attacker

Figure.4. Steps involved in stored XSS attack

Non-persistent Cross Site Scripting

An attacker with non-persisting XSS can execute malicious code and hijack user's session. Website vulnerable to reflected XSS does not properly sanitize user input and return malicious characters in resulting page. As in this type of attack, an attacker does not store any payload to the server, but malicious script is sent via victim's browser. The whole scenario is explained with the help of Fig. 5 taken from [9].

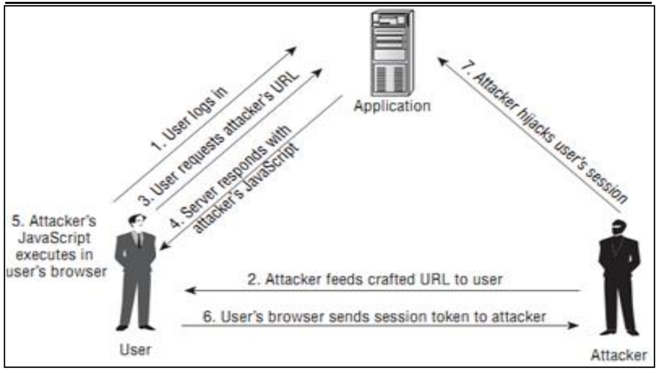

Application
1. User logs in
3. User requests attacker's URL
4. Server responds with attacker's JavaScript
7. Attacker hijacks user's session
5. Attacker's JavaScript executes in user's browser
2. Attacker feeds crafted URL to user
6. User's browser sends session token to attacker
User
Attacker

Figure.5. Steps involved in reflected XSS

Preventing Cross Site Scripting

The un-sanitization of user inputs is actually the XSS vulnerability, the reflected XSS as well stored XSS can be prevented easily, if all input data is sanitized properly. Several solution are proposed but defensive coding practice [9][17] is more applicable for this issue.

Input Validation

There are some guidelines for validating input, which should be strictly followed by the developer, to void scripting attack.

- The length of input data must be checked, it should not be exceeding the permitted length.
- Input data essentially be validated against allowed character set.
- Regular expressions should be used and each input data should be following the specific regular expression.

Output Validation

In output validation the problematic characters are represented in a specific way, so that these characters can be safely incorporated in a html document. As various characters have specific meaning. Via using html encoding these characters are replaced by their equivalent objects. The most common malicious characters and their equivalent html encoding are mentioned here.

“	$\longrightarrow$	&quot;
‘	$\Longrightarrow$	&apos;
&	$\longrightarrow$	&amp;
<		&alt;
>		&gt;

Cross Site Request Forgery

XSRF is a well-known form of XSS, where a request is made on behalf of a user by an attacker, xsrf exploits weakness of both, client’s browser as well server. Figure 6 explains the mechanism of XSRF attack where an attacker by some mean sends a link to a user having a valid account on a website which is vulnerable to xsrf attack. Attacker sends arbitrary requests via victims browser leaving him completely unaware about the requests being made on his behalf, and server also interpret all the requests as valid requests as these request are coming from authorized user. Many famous website had been victim of xsrf attack mentioned in [13].

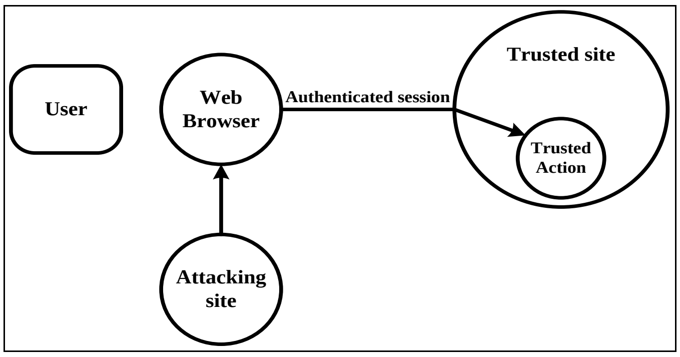


Figure.6. Cross site request forgery attack

Cross Site Request Forgery Prevention

In cross site request forgery (XSRF) attack, an attacker by some mean (like sending a malicious URL in email or sending an image embedded with malicious link) tricks and forces a victim to perform some action on his behalf on a website for which victim has a valid authentication. In XSRF attack mostly server does not differentiate between valid and fake request, as requests are made from victim’s browser and victim has valid authentication to perform any action. To overcome this type of attack researchers like [13], [14], [15], and [18] have suggested some prevention measures. The prevention measures suggested by [13] are mentioned here. The author has proposed two solutions to this attack:

Server Side Tool

In XSRF attack, an attacker takes the advantage of weakness of both server and client’s browser, author suggest that when any request is made by the user, server must validate the request by issuing a pseudorandom number, which should be stored in browser cookies and in form’s hidden field.

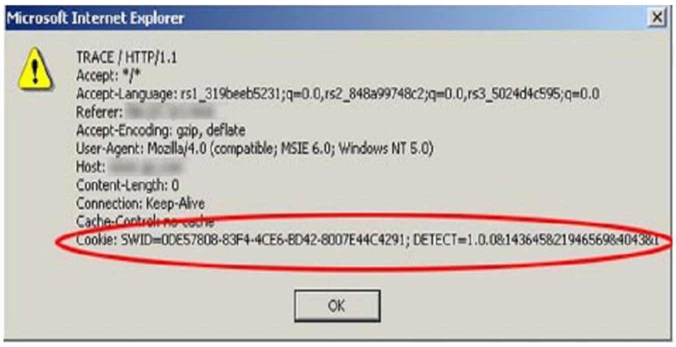
All those requests coming through post method from user’s browser essentially hold pseudorandom number in forms hidden field and in cookie. When attacker sends any request has to obtain pseudorandom number from cookies which is not changeable and from form’s hidden field, if pseudorandom number from both cookies and form’s hidden filed match then request will be consider by server as valid request.

Client Side Tool

At client side author have suggested a plugin. It can be installed as an extension on any browser, which will restrict sending fake request via victims’ browser.

Cross Site Tracing

Cross site tracing XST another form of XSS, in this attack attacker tries to steal user’s session by accessing victim’s cookies while httpOnly feature is in use. httpOnly feature was introduced by Microsoft in order to stop cookie steeling by document.cookie property. White Hate security experts in order evaluate httponly feature tried different ways to access cookies and finally it was found that cookies can be access via using trace method, no matter if httpOnly feature is enabled. When a request is made by using trace method, the server in response replies with every filed. Hence a malicious request can be sent using trace method to access cookies while HttpOnly feature is active. The author in article [16] discussed the attack scenario in detail in Fig.7 [16] the author got access to the cookies via using trace method while HttpOnly feature is active.



```
Microsoft Internet Explorer

TRACE / HTTP/1.1
Accept: */*
Accept-Language: rs1_319beeb5231;q=0.0,rs2_848a99748c2;q=0.0,rs3_5024d4c595;q=0.0
Referer: [redacted]
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Host: [redacted]
Content-Length: 0
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: SWID=0DE57808-83F4-4CE6-BD42-8007E44C4291; DETECT=1.0.0&1436458219465698404381

OK
```

Figure.7. Trace method for accessing cookie

XST Prevention

XST attack can be prevented by following some general guidelines described in WhiteHate security experts article[16].

- Trace request must not be active on webserver and end users should also be educated to deactivate trace method
- ActiveX Control always be denied for scripting

Always keep browsers updated for protecting against domain restriction bypass flaws.

III. RESEARCH METHODOLOGY

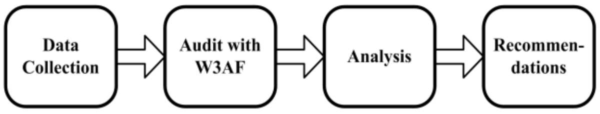


Figure.8. Research methodology

Data Collection

In order to audit, the most famous websites of Pakistan have been enlisted and divided into three different sectors and equal number of websites from each sector has been chosen. So that results may be compared to focus the most vulnerable segment.

Audit tool w3af

The penetration scanning tools are required to conduct penetration test, many freeware and purchased versions of auditing tools are available in the market; we choose w3af tool for scanning as it is free for education purpose.

Analysis

In this section we present list of audited websites from each sector and highlight the number of vulnerabilities.

Government Sector Websites

The most legendary government websites has been chosen, the list and audit report is presented in Table 1. It shows the number and type of the vulnerabilities. The results from this sector describe four different types of the vulnerabilities which are SQL Injection, Cross Site Scripting (XSS), Cross Site Request Forgery (XSRF) and Cross Site Tracing (XST). In this sector overall 61 percent of the websites are vulnerable and 39 percent of these websites are safe. The graphical results are shown in Fig.9.

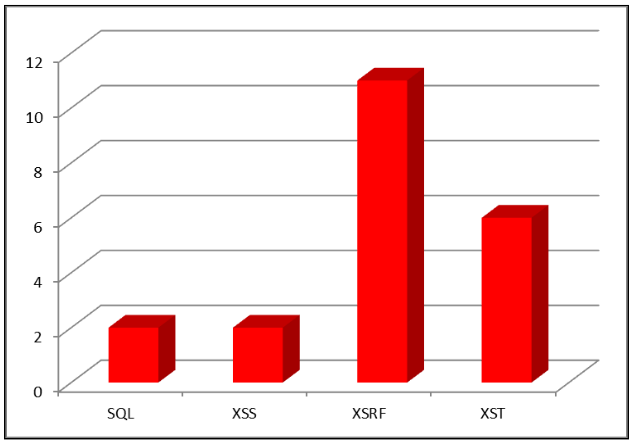


Figure .9. Vulnerability distribution in government sector websites

Education Sector Websites

The audit report of the education sector website suggests that greatest number of websites is vulnerable to cross site request forgery and cross site tracing attack. None of the website from this group was vulnerable to SQL injection, and one website identified which is susceptible to cross site scripting attack XSS. Table 2 presents the audit report of this sector and results are also plotted in the graphical form in Fig.10.

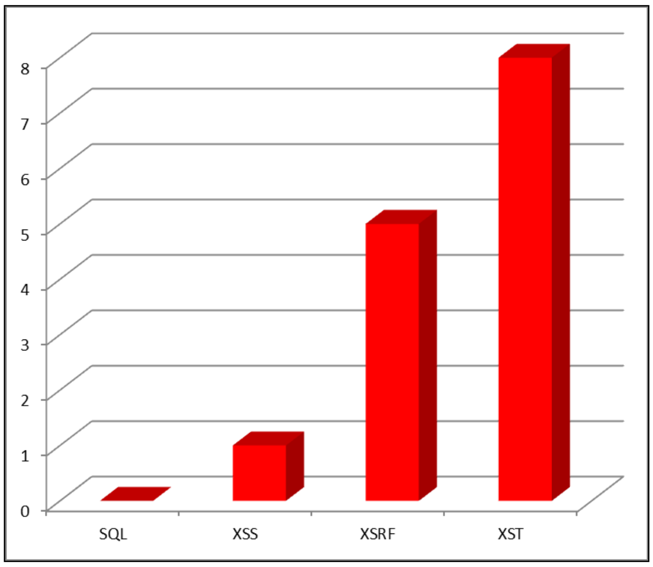


Figure.10. Vulnerabilities in Education Sector Websites

Commercial Sector Websites

The result from commercial sector websites is similar to education sector websites from XSRF point of view but it is also observed that few website are vulnerable to SQL Injection attack as well. Figure 11 and Table 3 summarizes our report from this sector.

Table 1: Govt. Sector Websites Audit Report.

S.#	Websites	SQL	XSS	XSRF	XST
1.	www.tourism.gov.pk	0	0	0	0
2.	www.fdsindh.gov.pk	0	1	1	0
3.	www.sindh.gov.pk	0	0	0	0
4.	www.finance.gov.pk	0	0	0	0
5.	www.e-government.gov.pk	0	0	0	1
6.	www.cbr.gov.pk	0	0	0	0
7.	www.na.gov.pk	0	0	1	0
8.	www.karachicity.gov.pk	0	0	0	0
9.	www.secp.gov.pk	0	0	1	0
10.	www.pakpost.gov.pk	0	0	0	0
11.	www.pta.gov.pk	0	0	1	0
12.	www.nadra.gov.pk	0	0	4	2
13.	www.moitt.gov.pk	0	0	0	1
14.	www.sngpl.gov.pk	0	0	0	1
15.	www.kse.com.pk	0	0	0	0
16.	www.fbr.gov.pk	0	0	2	0
17.	wwww.nbp.com.pk	0	0	0	0
18.	www.sbp.org.pk	0	0	1	0
19.	www.ptv.com.pk	2	0	0	0
20.	www.ppl.com.pk	0	1	0	1
21.	www.savings.gov.pk	0	0	0	0
	Total	**2**	**2**	**11**	**6**

Table 2: Education Sector Websites Audit Report.

S#	Website	SQL	XSS	XSRF	XST
1.	www.pec.org.pk	0	0	2	0
2.	www.hec.gov.pk	0	0	0	0
3.	www.neduet.edu.pk	0	0	0	1
4.	www.uok.edu.pk	0	0	0	1
5.	www.muet.edu.pk	0	0	0	1
6.	www.usindh.edu.pk	0	0	0	1
7.	www.au.edu.pk	0	0	0	0
8.	www.iqra.edu.pk	0	0	1	0
9.	www.nust.edu.pk	0	0	0	0
10.	www.lumhs.edu.pk	0	0	0	0
11.	www.quest.edu.pk	0	0	0	0
12.	www.giki.edu.pk	0	0	0	1
13.	www.iba-suk.edu.pk	0	0	0	0
14.	www.szabist.edu.pk	0	1	1	0
15.	www.qau.edu.pk	0	0	1	1
16.	www.nu.edu.pk	0	0	0	0
17.	www.ist.edu.pk	0	0	0	0
18.	www.iub.edu.pk	0	0	0	1
19.	www.vu.edu.pk	0	0	0	0
20.	www.hamdard.edu.pk	0	0	0	1
21.	www.nts.org.pk	0	0	0	0
	Total	**0**	**1**	**5**	**8**

Table 3: Commercial Sector Websites Audit Report.

S.#	Websites	SQL	XSS	XSRF	XST
1.	www.kese.com.pk	0	0	0	0
2.	www.ptcl.com.pk	0	0	1	1
3.	www.pakistanstores.com	1	0	0	0
4.	www.hbl.com	0	0	2	0
5.	www.ubl.com.pk	0	0	1	0
6.	www.mcb.com.pk	0	0	3	1
7.	www.homeshopping.pk	1	0	2	0
8.	www.vmart.pk	0	0	0	1
9.	www.mrlaptop.com.pk	2	0	1	0
10.	www.paperpk.com	1	1	0	0
11.	www.pakistanshopings.com	0	0	0	0
12.	www.galaxy.com.pk	0	0	0	0
13.	www.hamriweb.com	0	0	0	0
14.	www.hbm.com.pk	0	0	0	0
15.	www.honda.com.pk	0	0	0	0
16.	www.cokestudio.com.pk	0	0	0	0
17.	www.nation.com.pk	0	0	1	0
18.	www.zong.com.pk	0	0	0	1
19.	www.bankislami.com.pk	0	0	0	1
20.	www.telenor.com	0	0	1	0
21.	www.inboc.com.pk	0	0	0	1
	Total	**5**	**1**	**12**	**6**

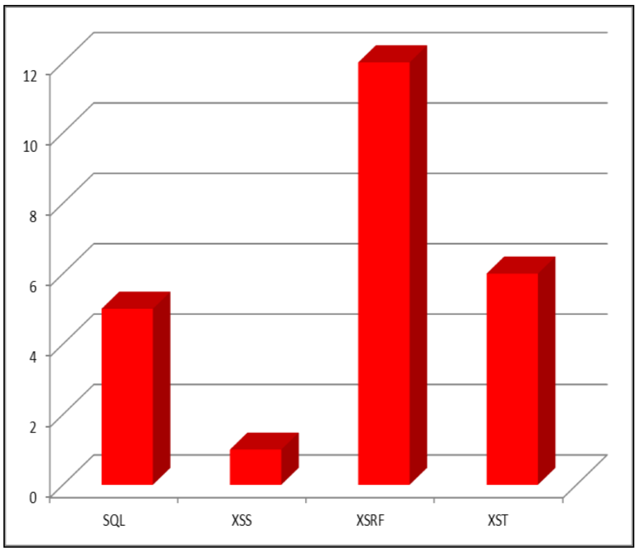


Figure.11. Vulnerabilities in commercial sector websites

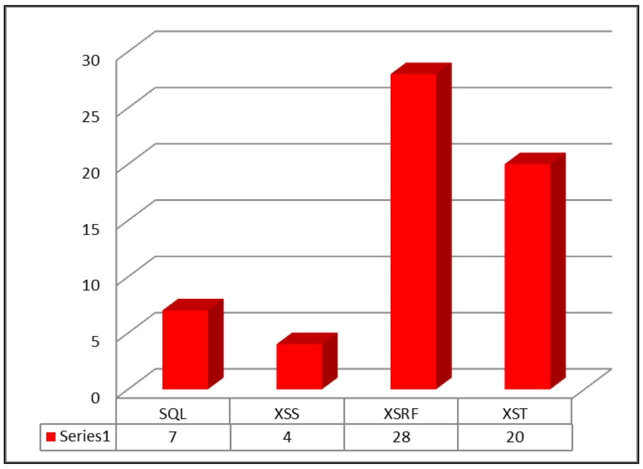


Figure.12. Vulnerability ratio from 61 websites

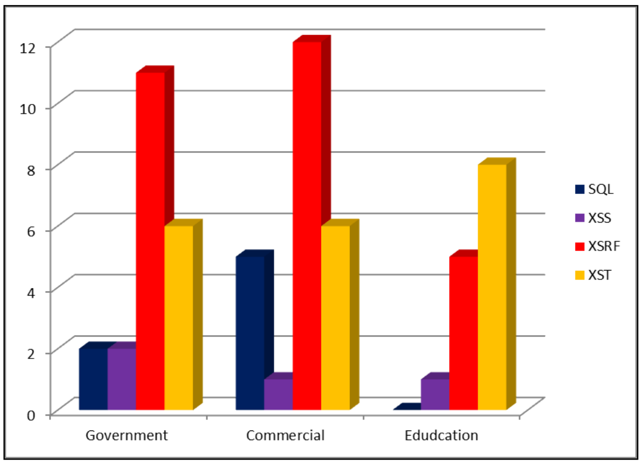


Figure.13. Vulnerability ratios in each sector from 21 websites

VI. COMPARISON AMONG THE SECTOR-WISE DISTRIBUTION

In this section the sector wise vulnerabilities have been compared in Table 4 and their graphical representation is shown in Fig.12 &13 respectively. Table 4 provides the listing of all the vulnerabilities which were identified in audited websites. Most of the websites are vulnerable to SQL Injection, XSS, XSRF and XST respectively. XSRF and XST are at the top of amongst all mentioned attacks.

Table 4: Sector wise vulnerabilities comparison

Vulnerability	Government	Education	Commercial	Total
SQL	2	0	5	7
XSS	2	1	1	4
XSRF	11	5	12	28
XST	6	8	6	20
Total	21	14	24	59

Most of the common vulnerabilities, the way attackers exploit them and several related examples are presented in our previous research work in [1]. Here we only highlight the prevention measures related to the SQL injection, XSS, XRF and XST.

Because of SQL Injection's popularity, it is the most discussed attack the authors in [2][3][4] have suggested different solutions to this vulnerability. For example; the parameterized technique has been used in [4] and the input validation method is used in [5], [6] and [7]. The authors in [7] use the regular expression technique which further improves the overall security measures. To further enhance the performance of the parameterized technique, the author has combined parameterized technique with stored procedure [8]. With the help of figures and examples, the author in [9] discussed in detail the prevention method for XSS and explains the way when an attacker exploits stored XSS vulnerability and hijack user's session by submitting his malicious payload. Whereas [9] explains the input and output validation methods related to the defensive coding for the XSS technique.

In YouTube, eBay, New York Times, ING Direct, Meta Filter the attackers have exploited the XSRF vulnerabilities [12]. The research work published in [13] and [14] also focuses on the XSRF prevention methods, while preventing XST experts suggest few guidelines mentioned in [15].

V. CONCLUSION & SUGGESTIONS

The causes of websites defected by the attackers in Pakistan are found out by the penetration test of some of the most famous websites from different sectors in Pakistan. A testing website is developed and audited, that does not observe any kind of security flaw and hence approve our technique suggestions. As most of the websites in Pakistan are developed by the newbies who often ignore all the security concern and just focus on the rapid development of these website. It is therefore very important that the web developers essentially confirm the safety of websites by testing with suitable scanners at the development time as well on quarterly basis or bi-annually so that both the common and new vulnerabilities can be identified and rectified in a timely manner.

REFERENCES

[1] Ali Raza, Asim Imdad Wagan, Zahid Hussain Abro, "Penetration Testing: A survey of web vulnerabilities", QUEST Journal Nawabshah, June 2011, Volume 10, ISSN 1665-8607.*

[2] Howard, M., LeBlanc, D, "Writing Secure Code", 2nd Edition, Microsoft Press 2003 publisher, ISBN: 9780735617223.

[3] Halfond, W.G., Viegas, J., Orso, A.: A Classification of SQL-Injection Attacks and Countermeasures. In: Proc. IEEE Int'l Sym. on Secure Software Engineering (March 2006).

[4] Clarke, J., July 2012, "SQL injection attacks and defense". 2nd Edition, Syngress Publisher, ISBN: 978-1597499637.

[5] Buehrer, G., Weide, B.W., Sivilotti, P.A.G., 2005. "Using parse tree validation to prevent SQL injection attacks", in: Proceedings of the 5th International Workshop on Software Engineering and Middleware. pp. 106–113, ISBN:1-59593-205-4.

[6] Kern, W., 2011. eGovWDF: "Validation-A new approach to input validation in Web based eGovernment applications", Technical Report.

[7] Jan, Goyvaerts,"Regular Expressions: The Complete Tutorial", First Printing: Feb: 2006, ISBN: 1-4116-7760-9.

[8] Preventing SQL Injection in ASP.NET, 2012. URL:http://www.mikesdotnetting.com/Article /113/Preventing-SQL-Injection-in-ASP.NET. [Last accessed: 20-Nov-2012].

[9] Dafydd, Stuttard, Marcus Pinto "The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws", ISBN-13: 978-0470170779, Publisher Wiley; 1st edition (October 22, 2007).

[10] Williams, J., Wichers, D., 2010. OWASP top 10–2010. "OWASP Foundation", http://www.owasp.org/ [Last accessed: 15-Nov-2012].

[11] CWE - 2011 CWE/SANS "Top 25 Most Dangerous Software Errors" [WWW Document], 2012. URL http://cwe.mitre.org/top25/ [Last accessed 16-Nov-2012].

[12] ZELLER, W. P., AND FELTEN, E. W. Cross-Site Request Forgeries: Exploitation and Prevention. Tech. rep., Princeton University, 2008.

[13] Siddiqui, M., Verma, D., others, 2011. "Cross site request forgery: A common web application weakness", in: Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference On. pp. 538–543.

[14] Luyi Xing, Yuqing Zhang, Shenlong Chen, "A client-based and server-enhanced defense mechanism for cross-site request forgery", RAID'10 Proceedings of the 13th international conference on Recent advances in intrusion detection, pp. 484–485, 2010, ISBN:3-642-15511-1978-3-642-15511-6.

[15] GROSSMAN,J.Cross-Site Tracing (XST) .http://www.cgisecurity.com/whitehatmirror/WhitePa per screen.pdf, 2003.

[16] How To: Protect From SQL Injection in ASP.NET [WWW Document], 2012. . URL http://msdn.microsoft.com/ens/library/ff648339.aspx, [Accessed February 27, 2012].

[17] Lwin Khin Shar and Hee Beng Kuan Tan *Nanyang Technological University, Singapore*, Published by the IEEE Computer Society, 0018-9162/March-12.

[18] Garskof, Robert. "Apparatus and Methods for Preventing Cross-Site Request Forgery." U.S. Patent Application 12/839,884, filed July 20, 2010.