ISSN 1605-8607

WEB SUSCEPTIBILITIES AND THEIR PREVENTION IN PAKISTAN

Ali Raza Bhangwar, Pardeep Kumar, Adnan Ahmed

ABSTRACT

The use of Internet in Pakistan has increased dramatically over the last years. Several organizations whether they are related to government, private, education or commercial sector use websites to provide required information to their visitors/customers. However, the detail study of such websites suggests that the web developers mostly ignore the security issues and just concentrate on how to make the website attractive. Several such websites are easily vulnerable to the major web security attacks such as SQL injection, cross site request forgery, scripting and tracing, denial of service, buffer overflow, etc. This research work attempts to identify the web vulnerabilities present in Pakistani websites by using the penetration techniques, proposes a suitable solution to these vulnerabilities and presents the statistical data of several websites from different sectors in a graphical form.

1. 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 arrival of web 2.0 and AJAX have further significantly improved the world of Internet 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. The researchers are constantly working on the security flaws and trying their best to minimize the consequences of the security flaws. Whereas the attackers are as skilled as the security experts so the attackers are also constantly working on it and finding the new ways to exploit these security flaws.

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 advantage and exploit these vulnerabilities.

Various organizations in the world, particularly in the developed countries, are now involved in providing awareness for these vulnerabilities by publishing a list of top vulnerabilities every year and by providing their solutions and by implementing them. 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.

Penetration testing is the well-known method of auditing the websites for security breaches. It is basically a combined approach of knowledge, skills and experience. It analyzes the website for security breaches by attacking the website with the same analogy as the attackers do. After identifying the vulnerabilities, 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.

This 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.

2. 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

2.1. SQL INJECTION

Because of SQL Injection's popularity, it is the most discussed vulnerability by the different authors and they have suggested different solutions to the same vulnerability [2], [3] and [4]. We in this paper focus on code level defense. The code level defense method is the most common and efficient technique since all other defensive techniques need to use this technique at the end. One of the fundamental assumptions of SQL Injection vulnerability is the dynamic building of SQL query. The alternative method of dynamic building SQL query is to use parameterized statements [4].

2.1.1. PARAMETERIZED STATEMENT

Parameterized statements are the place holders whose value is replaced at run time. Almost every programming language now supports parameterized statement. As an example we have taken a vulnerable query as shown in Figure 1 [4], which take user's inputs and generate dynamic SQL query, and then focus is made on how to secure it with the use of parameterized statements, for readers who want to read further on prevention of SQL Injection can read article [8].

```
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].

The above mentioned query takes user input and processes it for execution without sanitizing it.

The secure version of above mentioned example with the help of parameterized statements is as given in Figure 2 [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].

The parameterized statements not only provide an alternative for dynamic SQL building but also additionally checks for length and type of the input received. Username and password fields are now safe and will be checked that it contains a valid username and password and whether its type is correct along with the length. Request will be rejected if the input received will not match the required constraints.

2.1.2. INPUT VALIDATION

It's a method of analyzing the inputs the application gets against predefined set of inputs. The idea is to check all the received inputs which application might receive and process them if inputs are validated, if inputs differs from pre-defined set of inputs must be rejected. Researchers have suggested different techniques for input validation [5], [6], and [7], but the best method to sanitize inputs is to use regular expressions [7]. An example of validating inputs in C# programming is given in Figure 3 [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].

2.1.3. STORED PROCEDURES

Stored Procedures is a collection of statements in compiled forms within the database, it can significantly reduce the risk of SQL Injection vulnerability, as it gives restricted permission to access the data from the database. So if SQL Injection vulnerability is found, the attacker cannot change important information within the database if access permission is set properly by the developer. It may be noted that in some situations Stored Procedures alone may not protect from SQL Injection vulnerability, therefore it is best practice and suggested by Microsoft to use Parameters with Stored Procedures [16].

2.1.4. OUTPUT ENCODING

Output encoding is the process of safely handling the outputs which are about to be returned by the application in response of the request made, as it happens that sometimes the data supplied by the user will be processed by the application for output, therefore any malicious data which simply gets returned by the application for output processing can lead to potential SQL Injection. Therefore, it is necessary that whatsoever data is supplied to database must be encoded. In SQL encoding is done as: sql = sql.Replace(" ' ", " ' ' ");.

2.2. CROSS SITE SCRIPTING

Cross Site Scripting (XSS) is one of the most exploitable of all web attacks; XSS web vulnerability allows an attacker to inject a malicious code into the website which causes the visitors of the website to be the ultimate victim of the attacker. It was first observed by the CERT and stood 2nd position at OWASP for 2010 list. XSS vulnerability has victimized approximately 80.5% of all web attacks published by Symantec in 2008 [17]. There are two types of XSS vulnerability as given below.

2.2.1. PERSISTENT CROSS SITE SCRIPTING

Persistent XSS is sometimes called stored XSS, this vulnerability gives the attacker a chance to upload his data at the server and will be exploited when other users visit same website. Persistent Cross Site Scripting is explained with help of Figure 2.7 taken from [9] in which an attacker exploits stored XSS vulnerability and hijack user's session by submitting his malicious payload.

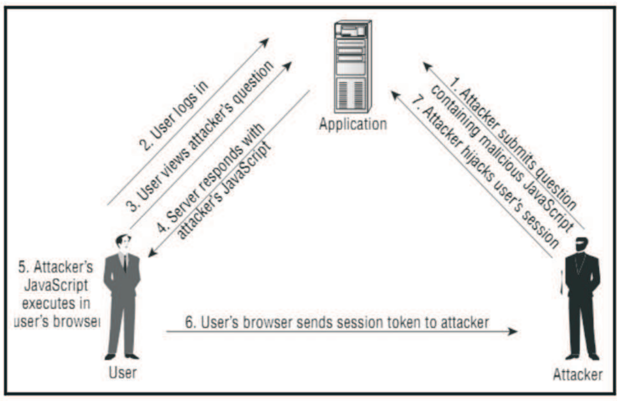

2. User logs in
3. User views attacker's question
4. Server responds with attacker's JavaScript
Application
1. Attacker submits question containing malicious JavaScript
7. Attacker hijacks user's session
5. Attacker's JavaScript executes in user's browse
6. User's browser sends session token to attacker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

Figure 4: Steps Involved in Stored XSS Attack.

2.2.2. NON-PERSISTENT CROSS SITE SCRIPTING

Non persistent cross site scripting is also called reflected XSS sometimes, another severe form of XSS vulnerability; this vulnerability takes the advantage of weak input validation of the web application. Unlike persistent XSS in which JavaScript is saved in the database, in this type, JavaScript is sent with the request to the vulnerable web application by the victim. The whole scenario is explained with the help of Figure 5 taken from [9].

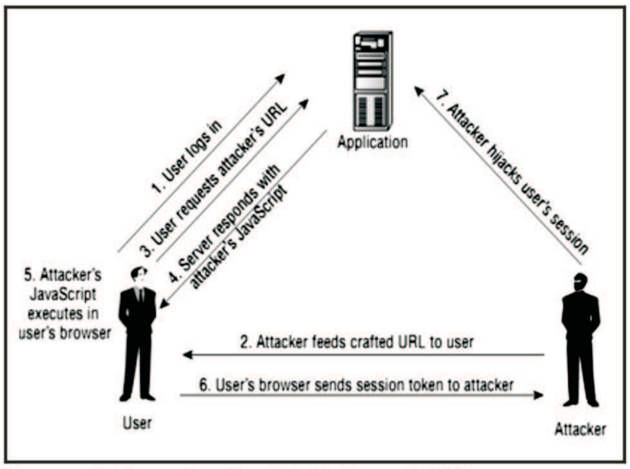

1. User logs in
3. User requests attacker's URL
4. Server responds with attacker's JavaScript
Application
7. Attacker hijacks user's session
5. Attacker's JavaScript executes in user's browser
2. Attacker feeds crafted URL to user
6. User's browser sends session token to attacker
User
Attacker

Figure 5: Steps Involved in Reflected XSS.

2.3. PREVENTING CROSS SITE SCRIPTING

The root cause for XSS vulnerability is un-sanitization of user inputs, In order to mitigate Reflected as well as Stored Cross Site Scripting it is necessary to find out all those insertion points in the website which takes user inputs and copy that input into response and sanitize all the inputs. Although there are other solutions suggested by the different authors but defensive coding practice [9], and [17] is more appropriate for defending against XSS.

2.3.1. INPUT VALIDATION

Input validation is the method for accepting all the inputs which the application developer thinks are good. Following are the few points to be noted when validating inputs.

- Data must be validated for permitted length that the data has not exceeded the length that the input field expects.
- Data must be checked for the permitted characters in the input field.
- Last but not the least is that the data must match with the particular regular expression
- The best defense against input validation is the use of Regular Expression

2.3.1. OUTPUT VALIDATION

Output validation is the process of HTML encoding the input returns in response by the application to request made by the end user. HTML encoding sanitizes the inputs for harmful characters by replacing the character with their equivalent objects. HTML encoding ensures that the inputs received will not harm the application logic and will be treated in right manner. The HTML encoding for the few characters which are well known malicious characters are as:

“	→	&quot;
‘	→	&apos;
&	→	&amp;
<	→	&alt;
>	→	&gt;

2.4. CROSS SITE REQUEST FORGERY

Cross Site Request Forgery (XSRF) sometimes called one-click attack is a web vulnerability which allows an attacker to send arbitrary request on behalf of user, it takes the advantage of both at server side as well as at client side's browser weak security mechanism. In XSRF attack attacker send JavaScript embedded in the link to the valid user of the website which is vulnerable to XSRF attack. The user is unaware of the attack and the vulnerable website is trusting on the request made by the user browser, as the requests are coming from trusted user's browser therefore server must respond to these requests, Server in this case is unable to identify whether the request is made by the user is a valid request?. Because request is coming from user's browser so there is no conformation about the request made and server blindly believes on the request which is coming from the user's browser. XSRF vulnerability had been exploited by the attackers on some major website such as YouTube, eBay, New York Times, ING Direct, Meta Filter [13]. The explanation of forged request is mentioned in the Figure 5 [13].

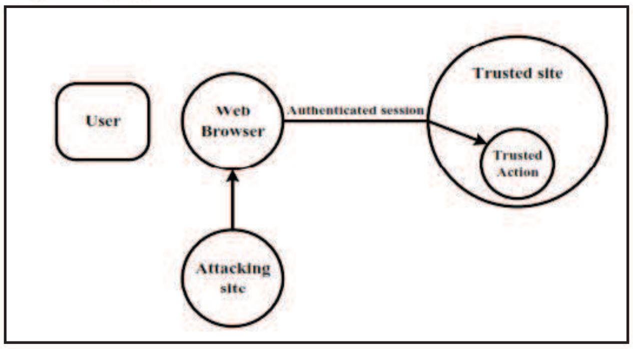


Figure 6: Cross Site Request Forgery attack.

2.5. CROSS SITE REQUEST FORGERY PREVENTION

XSRF Attack only occurs when the server is unable to distinguish whether the request is made by the trusted user which is coming from his browser with the session id issued by the server. In order to stop the exploiting this vulnerability there is a need of special measures which must be taken when making each request to the server. Researchers like [13], [14], [15], and [18] suggested some prevention measures. The prevention measures suggested by [13] are mentioned here. The author has proposed two solutions to this attack:

2.5.1. SERVER SIDE TOOL

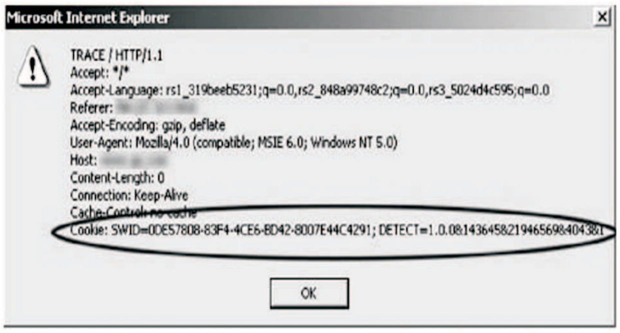
The idea behind this tool is that each request made by the user browser must be validated by the server for its authenticity. The author has proposed that a pseudorandom number must be generated by the server which will be stored in the user's cookies as well as in the form's hidden filed. Whenever any request is made by the browser via POST method, the form must contains the pseudorandom number within form's hidden field and also in the cookies which as per Same Origin Policy cannot be altered. In this method whenever an attacker wants to send any request via the browser of trusted user to the vulnerable website, attacker must have to obtain the hidden pseudorandom number that is to be sent to the server and when server receives any request from the trusted user it must validated both filed for pseudorandom number in the form's hidden field as well as in the cookies if both match then request is valid otherwise it is invalid.

2.5.2. CLIENT SIDE TOOL

To protect a user from being the victim of XSRF attack, author has developed a plugin which will protect user from many of the XSRF attacks. The author has implemented this tool as the Firefox extension.

2.6. CROSS SITE TRACING

As discussed earlier that the aim of the attacker in XSS is to hijack user session via getting access to the cookies by any means. In order to stop cookie theft via document.cookie property, Microsoft in 2002 introduced a feature namely HttpOnly. This feature successfully defended the theft of cookies via document.cookie property. The attacker who wants to capture user's cookies in order to hijack user session need to find another way of accessing cookies, as HttpOnly does not allow any JavaScript to get access to the cookies. Cross Site Tracing (XST) is a method which is capable of steeling cookies while HttpOnly feature is in use. White hat security experts after few weeks of introduction of HttpOnly feature, tried to analyze this feature and found a way of accessing a cookies without document.cookie property, the method used by the security expert was the TRACE method which is used for debugging and analysis purpose, by-default enabled on various web servers. Trace method responds to the request made by the user, it replies with all the fields requested by the browser. So if the attacker uses a JavaScript to issue a trace request from user's browser then the request will be responded by the server with the cookies while HttpOnly feature in use. The attack scenario is discussed in detail in this article [16]. The Figure 6 [16] shows the trace method issued for accessing cookies. It is to be noted that, it is not necessary that every TRACE request method will always bring you with the cookies it only works on those web server which supports this feature.



```
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
```

Figure 7: Trace Method for Accessing Cookie.

2.7. XST PREVENTION

There are some general recommendations for preventing the XST attack discussed by the WhiteHat security Experts in this article [16].

- Deactivate the TRACE request method on webservers.
- ActiveX Controls should not be used for scripting.
- Web browsers must be updated so that it can give sufficient protection against domain restriction bypass flaws.
- Users must be educated about disabling the TRACE on web Servers.

3. RESEARCH METHODOLOGY

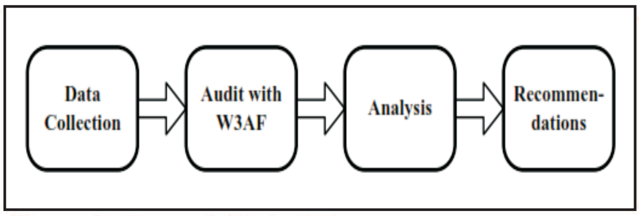


Figure 8: Research Methodology.

3.1. 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 with each other in order to highlight the most vulnerable sector among them

3.2. AUDIT WITH W3AF

In order to conduct penetration test, some penetration scanning tools are required. There are various scanning tools available in market some of them are open source and some of them are business version which can be purchased. w3af tool has been used for scanning. w3af has different types of plugins some of which are explained in later section.

3.2.1. DISCOVERY PLUGIN

Discovery plugin is responsible for finding new URL, forms and other "Injection points" this plugins works like web spider, at input field discovery plugin takes URL and in return discovery plugin gives Injection point.

3.2.2. AUDIT PLUGIN

The Injection points find out by the discovery plugins will be received by the Audit Plugin, and the audit plugin will perform testing so that it can identify the vulnerabilities by sending specially crafted data stored in its database.

3.2.3. GREP PLUGIN

The job of grep plugin is to explore the contents on web page in order to find vulnerabilities for all plugins which needs them.

3.1.4. EXPLOIT PLUGIN

This plugin is used to exploit the vulnerabilities identified in the audit phase.

3.1.5. OUTPUT PLUGIN

The data that is generated by the other plugins is saved in text format, w3af uses its output plugin to facilitate user with this data.

3.2.4. MANGLE PLUGIN

Request and response can be change with mangle plugin.

3.3. ANALYSIS

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

RESEARCH FINDINGS

3.3.1. GOVERNMENT SECTOR WEBSITES

In this section most famous websites from government sector has been chosen, the list and audit report of the entire websites from this sector is given in Table 1. The table describes the number and type of the vulnerabilities found in this sector. 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 Figure 8.

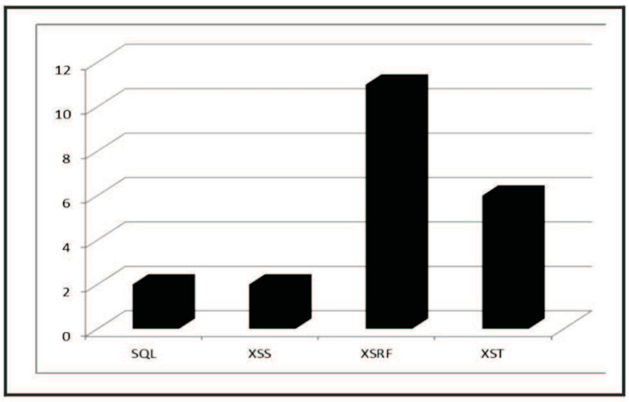


Figure 9: Vulnerability Distribution in Government Sector Websites.

3.3.2. EDUCATION SECTOR WEBSITES

In this section most famous websites from education sector has been chosen, the list and audit report of the entire websites from this sector is mentioned in Table 2. In this sector SQL Injection has not been identified, most of the websites are vulnerable to XSRF attack as well as XST and one website from this category is also vulnerable to XSS attack. The results are also plotted in the graphical form in Figures 9.

3.2.3. COMMERCIAL SECTOR WEBSITES

In this section the most famous websites from commercial sector has been chosen, the list of vulnerabilities of the entire websites from this sector is given below in Table 3. Commercial Sector websites are mostly vulnerable to XSRF attack as shown in figure 10.

Table 1: Audit List of Govt. Sector Websites.

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: Audit List of Education Sector Websites.

S#	Website	SQL	XSS	XS RF	XS T
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**

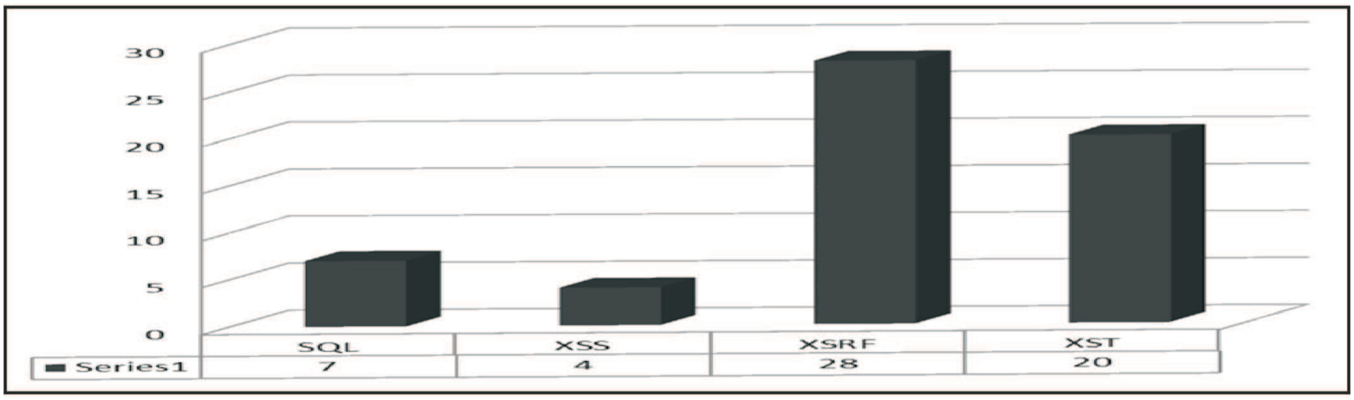


Figure 12: Number of time each vulnerability identified in all 63 websites.

Table 3: Audit list of Commercial Sector Websites.

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**

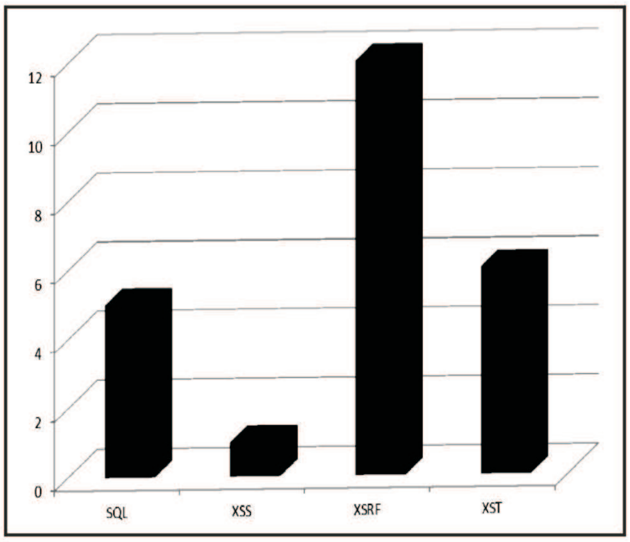


Figure 11: Vulnerabilities in Commercial Sector Websites.

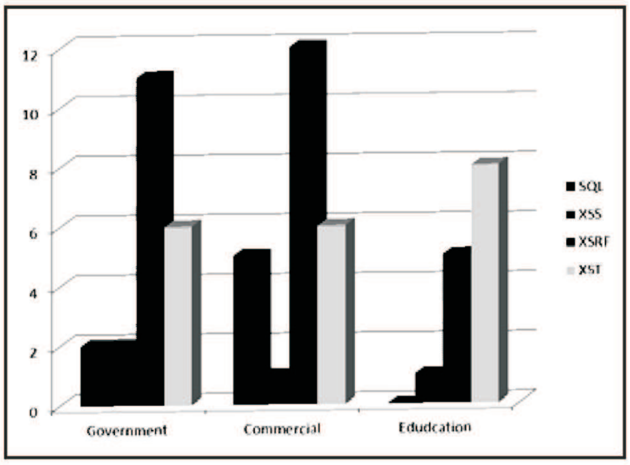


Figure 13: Vulnerability ratios in each sector from 21 websites.

Related Work:

3.3. COMPARISON AMONG THE SECTOR-WISE DISTRIBUTION

In this section the vulnerabilities found in each sector have been compared and the graphs of each sector according to their vulnerabilities are shown in Figures 11, & 12 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.

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.

Table 4: Comparison of vulnerabilities in each sector.

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	

Because of SQL Injection's popularity, it is the most discussed by the different authors in [2][3][4] and they have suggested different solutions to the same 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 that further improves the overall performance. To further enhance the performance of the parameter technique, it has been combined with stored procedure method in [8].

With the help of figures and examples, the authors in [9] detail the prevention method for XSS and explain 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.

XSRF vulnerability has been exploited by the attackers on some major website such as YouTube, eBay, New York Times, ING Direct, Meta Filter in [12]. The research work published in [13] and [14] also focuses on the XSRF prevention methods. There are some general recommendations for preventing the XST attack discussed by the WhiteHat security Experts in [15].

4. CONCLUSION & SUGGESTIONS

It has been seen that many websites in Pakistan are being rapidly defaced by the attackers. To find out its causes, the penetration test of some of the most famous websites from different sectors in Pakistan has been conducted in this work. It has been observed that majority of the websites are vulnerable to the security concerns. In order to secure these websites, some solutions have been recommended. Based on these suggestions, a testing website is developed and audited that does not observe any 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 must ensure the security of websites by testing websites with suitable scanners at the development time as well on quarterly 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", 13th international conference on Recent advances in intrusion detection, p. 484–485, 2010, ISBN:3-642-15511-6.

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

[16] How To: Protect From SQL Injection in ASP.NET [WWW Document], 2012. . URL http://msdn.microsoft.com/en-s/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.