Are our Educational Technology Systems Secured?

Mohamed Al-Ibrahim
College of Basic Education, PAAET, Kuwait

Abstract

The use of information and communication technology is an essential part of any contemporary higher-level educational system. Web applications that are accessed via web browsers over a network such as Internet or intranet are dominant in almost all education systems, such as universities, training, and research institutions. In particular, web based systems are used as informative or interactive web pages. Since these web pages contain critical information, securing educational systems is as important as securing any banking system. It has been noticed that some academic institutions have not fully secured their web pages against some class of vulnerabilities. In this empirical study, we elaborate these vulnerabilities and show their existence in the web sites of one of the academic institutions and describe the tools, techniques and results of our experiments.

1. Introduction

A web application is an application that is accessed with a web browser over a network such as the Internet or an intranet. Web applications are popular due to the ubiquity of the browser as a client. The ability to update and maintain web applications without distributing and installing software on potentially thousands of client computers is a key reason for their popularity. Web applications are used to implement E-commerce, online banking, webmail, business applications and many other functions [22]. One of the sectors that exploit the web technology in their services is the education sector such as research institutions, universities, training organizations …etc. Web application and web sites are heavily used in education for information dissemination, lectures, assignments, collaborations, discussions, conferences, grading, training, distance learning, research activities and many others.

Since the Internet is open systems, the security of the web applications is a main concern to many users of the web applications, especially when the web application is interactive and requires exchange of sensitive information such as money, passwords, or credit cards numbers. Therefore, there was great effort in both the research and industry community to provide secure communication services to web applications. A great deal of attention has been given to network-level security such as port scanning and great achievements have been accomplished at this level. However, it was found that about 75% of attacks were targeted to application-level, such as web servers [10].

Since web applications in education sector hold sensitive information such as passwords and grades that need to be secured from non-authorized users, the mission of securing web applications in the education sector is of high importance and unfortunately have not get great attention from the academicians.

The main goal behind this research paper is twofold. First, is to increase the awareness to the importance of securing education systems. Second, is to highlight to a new class of attacks targeting web applications in particular. The contributions include auditing web application security for the interactive web site of an academic institution, and the results reveal the fact that vulnerabilities of web applications in educational systems are indeed much serious. We suggested some defend techniques as counterattack. We also list a number of recommendations as security policy. The methodology and tools described later in this paper could be used as guideline. The main lesson to address is that educational systems have to revise their web-based applications against sort of vulnerabilities explained in this paper.

2. Research Background

Technically speaking, the model of the Internet was conceptually structured into a number of layers associated with specific protocols or services for each layer. The famous TCP/IP reference model consists of four layers. These layers are namely, Application-layer (web applications, emails, browsers, or servers), Transport-layer (TCP), Network-Layer (IP), and Physical-layer (cables, Wi-Fi, Bluetooth…etc).

The application-layer encompasses all the web applications of different services such as emails, browsers, chatting and so on. A web application is commonly structured as a three-tiered application as adopted in Figure 1. In its most common form, a web browser of a client is the first tier, a web server engine using some dynamic web content technology (such as ASP, ASP.NET, CGI, ColdFusion, JSP/Java, PHP, Python, or Ruby On Rails) is the middle tier, and a database server is the third tier. The web browser sends requests to the middle tier, which services them by making queries and updates against the database and generates a user interface. The web applications dynamically generate a series of web documents in a standard format supported by

common browser format such as HTML. Client-side scripting in a standard language such as JavaScript is commonly included to add dynamic elements to the user interface. Generally, each individual web page is delivered to the client as a static document, but the sequence of pages can provide an interactive experience as user input is returned through web form elements embedded in the page markup. During the session, the web browser interprets and displays the pages, and acts as the universal client for any web application.

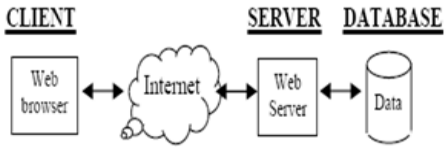

Figure 1.

Each service at the application-layer is defined with a port number at the Transport-layer, which lies behind the application-layer in the reference model. For example, port 80 is dedicated for HTTP protocol that work in web browsers, port 25 is for SMTP protocol used web mails, and so on. The Network-layer rely the traffic received from the source to the Transport- layer. It could be seen as border gate between different network boundaries. The physical-layer specifies the type of media through which data is transferred between the source and destination. Interested readers can refer to any textbook in computer networks for more details [17].

3. Literature Review

In the last few years, application-level vulnerabilities have been exploited with serious consequences: Hackers have tricked e-commerce sites into shipping goods for no charge, usernames and passwords have been harvested, and confidential information (such as addresses and credit-card numbers) has been leaked. Researchers start to investigate new tools and techniques which address the problem of application-level web security from multiple directions: pre, within, and post. Glisson, and Welland argue that security should be started first before the application development process upfront through an independent flexible methodology that contains customizable security components [6]. Scott and Sharp described a scalable structuring mechanism when developing an application facilitating the abstraction of security policies from large web-applications developed in heterogeneous multiplatform environments; and presented a set of tools which assist programmers in developing secure applications which are resilient to a wide range of common attacks [15]. Seo, Kim, Cho and Cha [9] developed web Intrusion Detection System (IDS) that uses anomaly-based intrusion detection and application-level IDS tailored to web services to detect any security anomalies in web application. On the other hand, Grier, Tang and King [7] noticed that web browsers itself are not secure enough, so they focused on building a new secure web browser that prevent various vulnerabilities that exist in current browsers. Other papers presented different ideas [2], [3].

3.1. Examples of Attacks

Attacks to organization resources can be launched to any layer of the reference model, but each attack has its own methods and tools and varies in the degree of difficulty. Hackers attacking web servers used to target their attacks starting from network-layer but this has become rare nowadays because most of the organizations that go online have the latest operating system and network hardware that are aware of the attacks at this level. Rather, hackers use different approach. One of the well known tactics is to launch attacks from the application-layer specifically through some essential service ports such as File Transfer Protocol (FTP). Attacks via these ports allows the hackers to access restricted areas and have access to administrator function, reveal or alter sensitive data, to get access or full control over other sites in the same web server or even to get control over the whole web server. The counteraction to this attack is to close the port. At Kuwait University, for example, most service ports are closed by default, and basic ports such as HTTP, FTP and Mail ports are only opened.

Attacks to organization resources can be launched to any layer of the reference model, but each attack has its own methods and tools and varies in the degree of difficulty. Hackers attacking web servers used to target their attacks starting from network-layer but this has become rare nowadays because most of the organizations that go online have the latest operating system and network hardware that are aware of the attacks at this level. Rather, hackers use different approach. One of the well known tactics is to launch attacks from the application-layer specifically through some essential service ports such as File Transfer Protocol (FTP). Attacks via these ports allows the hackers to access restricted areas and have access to administrator function, reveal or alter sensitive data, to get access or full control over other sites in the same web server or even to get control over the whole web server. The counteraction to this attack is to close the port. At Kuwait University, for example, most service ports are closed by default, and basic ports such as HTTP, FTP and Mail ports are only opened.

A new class of attacks to web application has emerged recently and is described in the following sections of this paper. An example of an attack of

this class is the SQL Injection. The primary form of SQL injection consists of direct insertion of code into user-input variables that are concatenated with SQL commands and executed on the database server. When the stored strings are subsequently concatenated into a dynamic SQL command, that are later passed to a database server and the malicious code is then get executed. The common solution to this problem took the form of suppressing the detailed error messages.

4. Web application security organizations

Due to the increase number of incidents of security attacks to web applications, many software vendors had fair efforts to clarify the web application security awareness and type of vulnerabilities on the web sites to customers. Nevertheless, special, non-profit, charitable organizations have established solely to promote to the concept of web application security. The most two important organizations in this area are the Open Web Application Security Project OWASP, (OWASP, 2005) and the Web Application Security Consortium, WASC [19]. OWASP is dedicated to finding and fighting the causes of insecure software. Everything in OWASP is free and open source. Participation in OWASP is free and open to all. OWASP provides many tools and much documentation to help in learning about the cause of web vulnerabilities. One of these tools is WebGoat, which is an online training environment for hands-on learning about application security. Another tool is WebScarab, which is a tool for performing all types of security testing on web applications and web services. On the documentation side, OWASP provides an awareness document that describes the top ten web application security vulnerabilities. Also, they provide OWASP Guide Project, a massive document covering all aspects of web application and web service security. Among other documentation and video presentations, a complete list of their projects can be found in their project home page [15].

5. A classification of attacks

Web site security attacks have been defined and classified by different organizations and research papers into a number of categories; such as authentication, authorization, client-side attacks, command execution, information disclosure, and logical attacks. Table 1 show a classification of major web security attacks according to WASC.

Describing the behavior of each threat in the table will exceed the limitation of this paper. Therefore, we will restrict the description to selected attacks that we used in our experiments [19].

Table 1. List of WASC classification of major web site attacks

Web Application Security Consortium (WASC) - Threat Classification	**Threat Category**
Abuse of Functionality	*Logical Attacks*
Brute Force	*Authentication*
Buffer Overflow	*Command Execution*
Content Spoofing	*Client-side Attacks*
Credential/Session Prediction	*Authorization*
Cross-site Scripting	*Client-side Attacks*
Denial of Service	*Logical Attacks*
Directory Indexing	Information Disclosure
Format String Attack	*Command Execution*
Information Leakage	*Information Disclosure*
Insufficient Anti-automation	*Logical Attacks*
Insufficient Authentication	*Authentication*
Insufficient Authorization	*Authorization*
Insufficient Process Validation	*Logical Attacks*
Insufficient Session Expiration	*Authorization*
LDAP Injection	*Command Execution*
OS Commanding Path Traversal	*Command Execution*
Predictable Resource Location	*Information Disclosure*
Session Fixation SQL Injection	*Authorization*
SSI Injection	*Command Execution*
Source-Code Disclosure	*Information Disclosure*
Weak Password Recovery Validation	*Authentication*
XPath Injection	*Command Execution*

5.1. Attacks description

(i) Brute Force

A Brute Force attack is an automated process of trial and error used to guess a person's username, password, credit-card number or cryptographic key. Many web sites allow easy passwords and many users choose easy to remember passwords. Using a dictionary word list, attackers can do millions of automated trials which may lead to successful login of actual user and password combinations.

(ii) SQL Injection

SQL injection is a technique that exploits a security vulnerability occurring in the database layer of an application. SQL injection is an attack in which malicious code is inserted into strings that are later passed to a database server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because the database server will execute all syntactically valid queries that it receives. Even parameterized data can be manipulated by a skilled and determined attacker. The primary form of SQL injection consists of direct insertion of code into user-input variables that are concatenated with SQL commands and executed. A less direct attack injects malicious code into strings that are destined for storage in a table or as metadata. When the stored strings are subsequently concatenated into a dynamic SQL command, the malicious code is executed.

(iii) Directory Indexing

Automatic directory listing/indexing is a web server function that lists all of the files within a requested directory if the normal base file is not present. Web servers allow a feature to index all files in a directory if there is no default page such as index.html or default.asp in this directory. Directory indexing may reveal sensitive data like users list files or files containing database connection information.

(iv) Information Leakage

Information Leakage is when a web site reveals sensitive data, such as developer comments or error messages, which may aid an attacker in exploiting the system. Sensitive information may be present within HTML comments, error messages, source code, or simply left in plain sight. There are many ways a web site can be coaxed into revealing this type of information. While leakage does not necessarily represent a breach in security, it does give an attacker useful guidance for future exploitation. Leakage of sensitive information may carry various levels of risk and should be limited whenever possible.

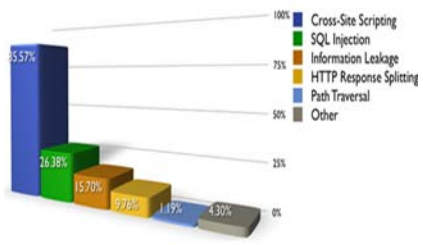

Figure 2. Percentage of top 5 websites vulnerability by cl

(v) Denial of Service

Denial of Service (DoS) is an attack technique with the intent of preventing a web site from serving normal user activity. DoS attacks, which are easily normally applied to the network layer, are also possible at the application layer. These malicious attacks can succeed by starving a system of critical resources, vulnerability exploit, or abuse of functionality.

Many times, DoS attacks will attempt to consume all of a web site's available system resources such as CPU, memory, or disk space. When any one of these critical resources reaches full utilization, the web site will normally be inaccessible.

(vi) Cross-site Scripting

Also known as XSS, is the most popular attack and is a technique that forces a web site to echo attacker-supplied executable code, which loads in a user's browser. Cross-site scripting occurs when a web application gathers malicious data from a user. The data is usually gathered in the form of a hyperlink which contains malicious content within it. The user will most likely click on this link from another website, instant message, or simply just reading a web board or email message. Usually the attacker will encode the malicious portion of the link to the site in HEX (or other encoding methods) so the request is less suspicious looking to the user when clicked on. After the data is collected by the web application, it creates an output page for the user containing the malicious data that was originally sent to it, but in a manner to make it appear as valid content from the website. Many popular guestbook and forum programs allow users to submit posts with HTML and Javascript embedded in them. If for example someone logs in as "john" and read a message by "joe" that contained malicious Javascript in it, then it may be possible for "joe" to hijack the session just by reading his bulletin board post. Further details on how attacks like this are accomplished via "cookie theft" are explained in detail below.

(vii) Source Code Disclosure

Full source code disclosure is any website owner’s worst nightmare and any hacker’s dream. They usually contain database connection information like IP address, port number and valid credentials. In certain cases, application test users’ login names and passwords may also be stored in these files. What makes this attack even more dangerous is that it will go completely unnoticed because it only exploits a functionality of the page! It will leave no unusual trail like an error log.

Brute Force	Site that does not restrict number of trials to login with same login or from same IP, is vulnerable to brute force attack either manually or automated; this test will only be applicable to site that contains login or authentication portal.
SQL Injection	The ability to inject arbitrary SQL statement that can be executed by the server. This kind of attack can reveal sensitive information from database, or even worse, it can allow attacker to add, alter or delete data from the database. This kind of attack is not applicable to site that does not have Database Management System (DBMS).
XSS	Cross Site Scripting attack tries to force the server to execute arbitrary code supplied by the attacker; this type of attack is one of the most common attacking techniques, and many sites are vulnerable to it.
Information Leakage	Sensitive information may be left in developer comments, or revealed with error messages; such information can allow the attacker to exploit the system
Directory Indexing	Web servers can show a list of all existing files in a directory if no base file such as index.html or Default.asp exists.
Source Code Disclosure	Web server can send the source code of web page rather than send the result of execution
Denial Of Service	Web server may be not able to response to request because of the allocation of resources

5.2. Vulnerabilities statistics

Figure 2 show some statistics about website vulnerabilities according to the study in conducted on non-educational web sites .

6. Methodology

In our study, we built our own list of vulnerabilities listed in Table 2 extracted from the list of threats in Table 1. The set of threats were chosen based on the availability of the resources to implement the attacks. Other threats require special resources such as penetration tools which were unavailable. We describe the conditions to the site/server that need to run the test of the corresponding attack over it.

6.1. Tools Used in Scan Process

The process of examining intensely to find security vulnerability is known as scanning. The software which is specialist in finding security holes or vulnerability is called Scanner. Scanners are used first to collect essential information about the web site such as web-server and OS type and their version and if any patches were installed; this information usually appears in system banner and is helpful to discover well-known vulnerabilities on the server [18].

Therefore, it is wise to hide such information from non-authorized. We used a web vulnerability scanner named Acunetix (Acunetix, 2006). This program is used to check a web site for a wide range of vulnerabilities, and it includes many innovative features such as:

- Automatic JavaScript analyzer
- Industry's most advanced and in-depth SQL injection and Cross-site scripting testing
- Visual macro recorder makes testing web forms and password protected areas easy
- Extensive reporting facilities including OWASP Top 10 vulnerabilities
- Multi-threaded and lightning fast scanner crawls hundreds of thousands of pages with ease
- Intelligent crawler detects web server and application language types
- Crawls and analyzes web sites including flash content

In addition to the automatic web vulnerability scanner, we used other tools for the auditing process. Some of them such as SQL injection cheat sheet [4] and XSS cheat sheet (Ha.ckers, n.d.) are nothing more than a guide. We also used Metasploit Framework to test some of the vulnerabilities and launch exploits against it [11]. Other tools for penetration testing were also explored but not really used because they are mainly used in network security penetration testing, such as telnet clients, port scanners and other hacking tools. Learning tools such as WebGoat from OWASP was also used.

6.2. Google as a hacker tool

Google is a powerful search engine that can be used to find sites with special vulnerabilities. We used Google to enumerate web sites that use PHP, ASP, ASPX, or JSP in Kuwait University web sites. (It was noticed that some of the web sites use more than one scripting language; such as the web site in the College of Engineering which uses JSP in some parts of the site and uses Perl, mod_ssl, and mod_perl in other parts.)

By submitting the following three queries to Google and seeing its response, we can enumerate the interactive web sites in Kuwait University: By submitting the following three queries to Google and seeing its response, we can enumerate the interactive web sites in Kuwait University: 1-inurl:kuniv

inurl:php, 2- inurl:kuniv inurl:asp, 3- inurl:kuniv inurl:jsp.

6.3. Sites to be tested

For each scanned web site, a table was constructed similar to Table 3, where essential information was collected.

Table 3. Template for web site information

OS Type	Windows
Server Banner	Microsoft-IIS/5.0
Port Scan Result	
Web Server type & version	IIS 5.0
Scripting Lang. Engine	ASP
Database System	Microsoft JET Database
IP Address	139.141.176.20

6.4. Manual vs. Automatic Scan

Automated scanners were never meant to replace the manual audit process; they are just an aid to finding cracks and possible problems in web applications. On the other hand, manual audit of a large web application is almost impossible, and one will likely miss many problems. Imagine an application with 100 (or even 1000) scripts, where each script accepts several parameters. Testing each parameter in each script for SQL injection, Cross Site Scripting, HTTP Response Splitting, etc. would definitely take months of work.

Automated scanners might have some shortcomings, but if used properly, they will save a lot of time and money. So, if one wants to perform a proper and thorough audit, it should use an automated scanner as well as manual audit, in order to get full coverage.

Table 4. List of tested web sites

Ref.	Site URL	Web server type	OS type	Scripting Lang. Engine
1	pubcouncil.kuniv.edu.kw	IIS/6.0	Windows	ASP
2	kjse.kuniv.edu.kw	IIS/6.0	Windows	ASP
3	cpe.kuniv.edu	IIS 5.0	Windows	ASP
4	cmtd.kuniv.edu.kw	IIS 6.0	Windows	ASP
5	onlinetrain.kuniv.edu	IIS/5.0	Windows	ASP
6	library.kuniv.edu.kw/	IIS/5.0	Windows	ASP, JSP
7	science.kuniv.edu.kw	Apache/2.0.39	Windows	PHP
8	law.kuniv.edu.kw/	Apache/2.0.40	Unix	PHP
9	dlis.kuniv.edu/	Apache 2.x	Unix	PHP, Perl, mod_ssl, mod_perl
10	faculty.eng.kuniv.edu.kw	Apache 1.x	Windows	JSP, mod_ssl mod_perl,openSSL
11	jael.kuniv.edu.kw/	Apache/2.0.40	Unix	PHP
12	gco.kuniv.edu.kw/	Apache/2.0.40	Unix	PHP

Table 4 above summarizes the features of the web sites that collected from scanning process. An interesting observation was that all the web sites that have been scanned were using either one of the two common web server types: Apache or IIS (in various versions.). Also, the operating system (OS) was either Windows or Unix. We will use (Ref.) as abbreviation for Reference Number to refer to the site URL. Figure 3 is a snapshot of one of the sites that was used in our test and belongs to the online-training portal web site at Kuwait University (Ref. 5) Each web site was tested against the list of threats, their vulnerability was checked, and its degree of harmfulness was determined. Then, we list some suggestions to solve the problem.

7. Results

After running the vulnerability tests on each web site listed in the table using the scanning tools, many serious threats were discovered. Figure 4 is an example of a snapshot obtained from running the scanning software on the web site of Ref. 5.
Table 5 summarizes the results obtained from testing the vulnerabilities of all the web sites and their frequency.

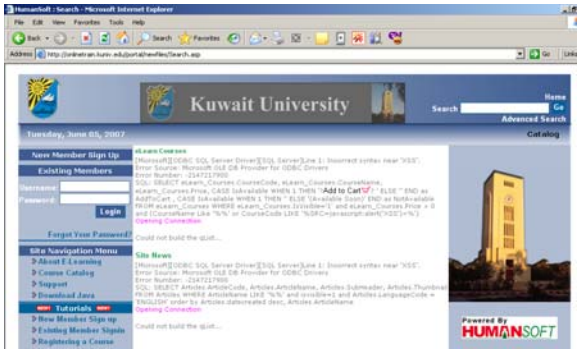

Figure 3. Snapshot of a tested web site

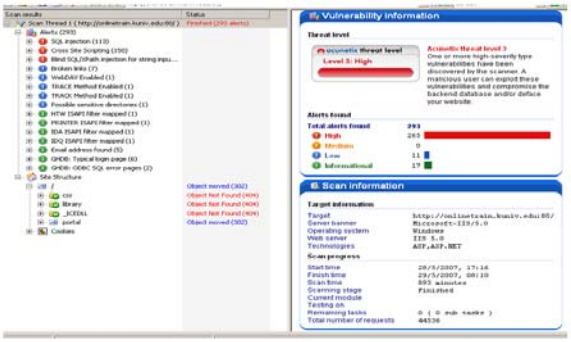

Figure 4. Snapshot of the result of tested web site by scanning tool

Table 5. Summary of vulnerabilities in the tested web sites

Attacks / Ref.	Brute Force	SQL Injection	XSS	Information Leakage	Directory Indexing	Source Code Disclosure	Denial Of Service
1	N/A	YES	No	YES	No	No	No
2	N/A	YES	No	YES	No	No	No
3	YES	YES	No	No	No	No	No
4	N/A	YES	No	YES	No	No	YES
5	YES	YES	YES	YES	No	No	No
6	N/A	YES	No	YES	No	No	YES
7	N/A	YES	YES	YES	YES	No	No
8	YES	No	YES	YES	No	No	No
9	YES	No	No	No	No	No	No
10	N/A	YES	No	No	YES	No	No
11	N/A	No	YES	YES	No	No	No
12	N/A	No	No	YES	No	No	No

8. Analysis

It is obvious from comparing the chart in Figure 5 obtained from Table 5, and the chart in Figure 2, obtained from WASC (on non-educational web sites) that the vulnerabilities: Information leakage, SQL injection and XSS appear in both studies as the top three threats (although we conducted our tests on short-list of vulnerabilities) . This proves our claim that education systems are insecure. It also shows the coincident of most common threats vulnerable to insecure web sites. It also provides a roadmap for the order in which one might follow to start securing the web sites.

The results of testing the web sites are summarized in Table 6 along with corresponding possible remedy action. Nevertheless, it is possible to measure the efficacy of the proposed defend strategy (Action) either by using Intrusion Detection System (IDS) which can detect configured anomalies and produce statistical reports. Also, many operating systems and applications have automatic updates.

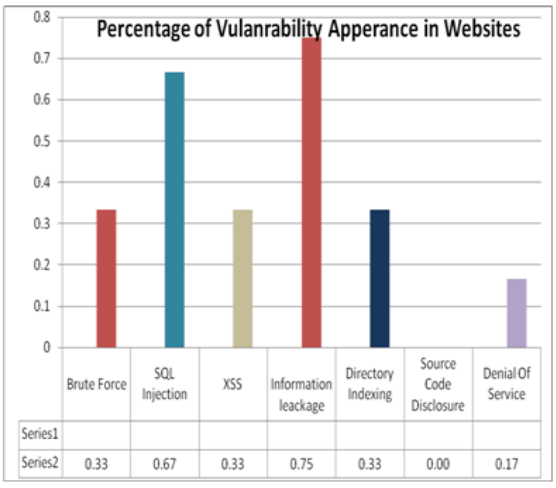

Figure 5. statistics of vulnerabilities

Table 6. Summary of vulnerability analysis on the scanned web sites

Ref.	Findings	Action
1	vulnerable to SQL injection attacks and reveals information such as field and table names	User Inputs should be sensitized before processing
2	Same as in Ref. 1	Same as in Ref. 1
3	The web site don't allow automatic scan & block IP from accessing the site again.	Same as in Ref. 1, and need to Update web server & O.S.
4	static web site in some parts; the interactive part will cause Microsoft FrontPage Server Extensions to reach 100% CPU utilization.	Front Page server extensions should be disabled / updated to new release
5	Submitting script in the search input box will generate an error which contains information about database structure	Update server & ASP engine, and revise enabled services in the web server, and Limit number of login attempts
6	message reveals some valuable information regarding table names and field names which may be used to modify table contents	Error messages should be reviewed
7	many but not harmful vulnerabilities	Disable Directory indexing , and Update server & PHP engine
8	the site uses Squirrel mail system which has many vulnerabilities. The guest book is vulnerable to XSS and& reveals information about the web server files.	Update web & mail servers User inputs should be sanitized before processing
9	static site, most of the pages are HTML, the site uses Claroline open source system which has many vulnerabilities.	
10	old version of apache server & venerable to serious problems. Able to retrieve information of critical files of the server.	Update web server and O.S., and Disable directory indexing
11		Same as Ref. 3.
12	Same as Ref 11. but not vulnerable to XSS	Same as Ref. 11.

9. Recommendations

From the above study, we can make the following recommendations:

1. The computer services and support in the academic institutions should dedicate a security department (unit) to follow up the security issues of their educational systems.
2. Security department has to focus on the security of the network and hardware level.
3. Web site security auditing processes should be applied to all the academic institution web sites even when there is no correlation among them. The method described in this paper and in the article can be used as a guide [7].
4. Academic institutions should set standards for their web site designs and applications since most of security problem emerges from poor development techniques.
5. Web developers in academic institutions should be trained for web security auditing.
6. Security features must be used in the web sites. This includes choosing proper browsers that have built-in security features such as handling cookies.
7. It is highly recommended to use secured browsing protocol (https) especially in web sites that deals with sensitive information such as grades, passwords. The "s" means that when accessing that particular web site, all web traffic between a web browser and the web site uses the Secure Sockets Layer (SSL) – in other words it is encrypted.
8. Web servers and operating systems must be periodically updated with the latest patches provided by the vendors since they provide protection against latest security holes or bugs in the system. The security department should build a mechanism to follow up and deploy these updates on their clients and servers.

9. Any web application implemented using readymade education packages such as MOODLE or MAMBO should be periodically checked for new releases and updates.
10. Many academic institution web sites are just a group of static web pages. Many of the open source Content Management Systems (CMS) can be used to build such sites.
11. Both automatic and manual scanning process should be done in parallel occasionally to ensure full and accepted auditing results.
12. Server type and O.S. and their version should be hidden in the system banner.
13. Passwords should be long and carefully chosen and contains combination of alphanumeric and special symbols and should be enforced to be changed frequently.
14. It is very important to protect the intranet of the enterprise with security devices such as Firewall and IDS, along with latest security software like anti-virus and web-sense.

10. Conclusion

The analyzed results in this study show that education technology systems using web services are insecure. Educational systems usually hold sensitive information and should be secured against application-level threats. Hacking of web sites and getting access to sensitive data is an easy task even if there is good network-level protection. Exploiting web site vulnerabilities can be an easy task if web developers do not shield web sites against certain threats. The Information leakage, SQL injection and XSS are the most common threats.

11. Acknowledgment

The authors wish to thanks Eng. Ehab Ali from Kuwait University for his great effort in conducting the experiments and tests.

12. References

[1] Acuntix. (2006). Auditing your web site security with Acunetix web vulnerability scanner. Retrieved March 15, 2009, from website: http://www.acunetix.com/.

[2] Cao, M., Xing, T., & Wang, C. (2009). Implementation of web security & identity scheme based on session & online table. Proceeding of the 4th ICCSE '09, 1278-1283.

[3] Dai, S. & Du,Y. (2009). Design and implementation of dynamic web security and defense mechanism Based on NDIS intermediate driver, Proceeding of APCIP ‘09,1, 506 –509.

[4] Ferruh Mavituna. (2007, March 15). SQL Injection Cheat Sheet, http://ferruh.mavituna.com/.

[5] Fonseca, J., & Vieira, M. (2008). Mapping software faults with web security vulnerabilities, Proceeding of IEEE International Conference on Dependable Systems and Networks, 257-266. doi: 10.1109/DSN.2008.4630094

[6] Glisson, W. & Welland, R. (2005). Web development evolution: the assimilation of Web engineering security, Proceeding of Third Latin American Web conference, 5 pp. doi: 10.1109/LAWEB.2005.48

[7] Grier, C., Tang, S. & King, S.T., (2008). Secure web browsing with the OP web browser, Proceeding of IEEE Symposium on Security and Privacy, 402-416. doi 1109/SP.2008.19

[8] Ha.ckers. (n.d). XSS (Cross Site Scripting) Cheat Sheet, Retrieved from http://ha.ckers.org/xss.html

[9] Jeongseok Seo, Han-Sung Kim, Sanghyun Cho, & Sung Deok Cha (2004). Web server attack categorization based on root causes and their locations, Proceedings of ITCC’04, 1, 90-96. doi: 10.1109/ITCC.2004.1286431

[10] Livshits, B., & Lam, M. (2005). Finding security vulnerabilities in Java applications with static analysis, Proceedings of the 14th conference on USENIX Security Symposium, 14, Retrieved 2009 from website http://www.portal.acm.org/.

[11] Metasploit, (2009). Metasploit Framework Development, Retrieved from website: www.metasploit.com

[12] OWASP (2005). Open Web Application Security Project. http://www.owasp.org/index.php/ OWASP_Top_Ten_Project.

[13] Runa Dwibedi (2005), XPath injection in XML databases, http://palisade.plynt.com/issues/2005Jul/xpath-injection/

[14] Scott, D. & Sharp, R. (2002). Developing secure web applications, Journal of Internet Computing, IEEE Publication, 6 (6), 38-45.

[15] Scott, D. & Sharp, R. (2003). Specifying and enforcing application-level web security policies, Journal of IEEE Transactions on Knowledge and Data Engineering,15(4),771–783. doi: 10.1109/TKDE.2003.1208998

[16] Security Focus Forum. (2009). Retrieved from website http://www.securityfocus.com/.

[17] Tanenbaum, A. (1996). Computer Networks. Third Edition, New York: Prentice Hall.

[18] Vieira, M., & Antunes, N., & Madeira, H. (2009). Using web security scanners to detect vulnerabilities in web services, Proceeding of the International Conference on Dependable Systems & Networks' 09, IEEE/IFIP, 566 - 571.

[19]WASC (2006,a), Web Application Security Consortium. Retrieved from website http://www.webappsec.org/projects/threat/

[20] WASC (2006,b). Classes of attacks, http://www.webappsec.org/projects/threat/classes_of_attacks.html

[21] Web Site Security Audit, (2009). http://www.beyondsecurity.com/pdf/wssa_wp.pdf

[22] Zhou, X., Zhang, Y., & Orlowska, E. (Eds.). (2003). Web technologies and applications, Proceedings of 5th Asia-Pacific Web Conference, Lecture Notes in Computer Science, Springer.

