Volume 2, Issue 7, July 2012 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Analysis of Web Application Vulnerability in Cloud Computing

Sachin D Choudhari *
Asst. Professor Department of Information technolgy
J L Chaturvedi college of engineering,Nagpur,India

Dr. S. K. Shrivastava
Director, SBITM College of Engineering
Betul, India

Abstract—***Web site because of growing demand is an area of concern in the IT security community because they are literally popping up all over. Public web service providers are available from Google.com, Amazon.com, Microsoft, Oracle/Sun, Canonical/Eucalyptus and many other vendors. But when we start our private web application, there are major security issues that should be noted. These issues are SQL injection, Cross Site Scripting (XSS), Username enumeration. The proposed work is concerned with discovery of these vulnerabilities in the web application, discovery of security solutions, and finding evidence that early-adopters or developers have grown more concerned with security.***

Keywords— ***Web application vulnerability, Security issues in web application, cloud computing***

I. Introduction

Now more than ever, Web applications are a critical part of business. Employees, customers and partners prefer to do business online, and expect to be able to access a variety of information and transactions through Web sites and Web services. As a result, Web applications [5] hold a treasure trove of data behind their front ends: credit card numbers, medical records, confidential company financial results, and this list goes on. Attackers are well-aware of the valuable information accessible through Web applications, and their attempts to get at it are often unwittingly assisted by several important factors. Conscientious organizations carefully product their perimeters with intrusion detection systems and firewalls, but these firewalls must keep ports 80 and 443 (SSL)[19] open to conduct online business. These ports represent open doors to attackers, who have figured out thousands of ways to penetrate Web applications. Further complicating the security picture, Web applications are often written in high-pressure environments on tight schedules by developers with little or no security training. Once development is complete, the applications are put through QA testing that typically focuses on performance and functionality, rather than security.

As a result, an application that emerges from QA with flying colours can still be riddled with exploitable flaws. It's no surprise, then, that Gartner reported that 75% of hacks happen at the application layer. Ting is a catch-all phrase that covers virtualized operating systems running on virtual hardware on untold numbers of physical servers. The "cloud" term has consumed High-Performance Computing (HPC), Grid computing and Utility Computing. It is imperative that companies identify and address vulnerabilities in their Web applications.

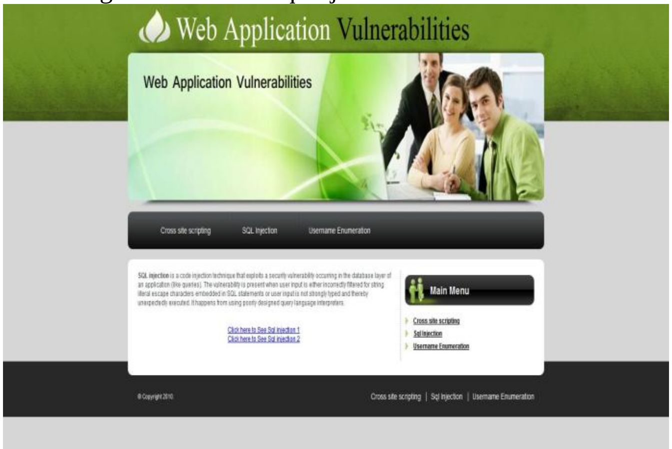
II. Web Application vulnerabilities

SQL injection [2]is a vulnerability that allows an attacker to alter backend SQL statements by manipulating the user input. An SQL injection occurs when web applications accept user input that is directly placed into a SQL statement and doesn't properly filter out dangerous characters. This is one of the most common application layer attacks currently being used on the Internet. Despite the fact that it is relatively easy to protect against, there is a large number of web applications vulnerable.

Impact:

An attacker may execute arbitrary SQL [19] statements on the vulnerable system. This may compromise the integrity of your database and/or expose sensitive information. Depending on the back-end database in use, SQL injection vulnerabilities lead to varying levels of data/system access for the attacker. It may be possible to not only manipulate existing queries, but to UNION in arbitrary data, use sub selects, or append additional queries. In some cases, it may be possible to read in or write out to files, or to execute shell commands on the underlying operating system. Certain SQL Servers such as Microsoft SQL Server contain stored and extended procedures (database server functions). If an attacker can obtain access to these procedures it may be possible to compromise the entire machine.

Following screen shows Sql injection attack:

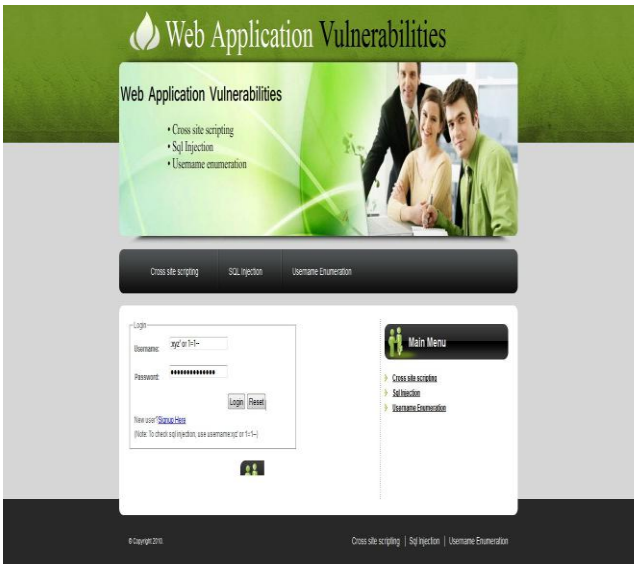


SQL injection is a code injection technique that exploits a security vulnerability occurring in the database layer of an application (like queries). The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. It happens from using poorly designed query language interpreters.

Click here to See Sql injection 1
Click here to See Sql injection 2

SQL injection[5] is a code injection technique that exploits a security vulnerability occurring in the database layer of an application (like queries). The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. It happens from using poorly designed query language interpreters.

In following screen user has written sql query that always returns true value. So If user is not registered to our system , then also he can login to system.



Login

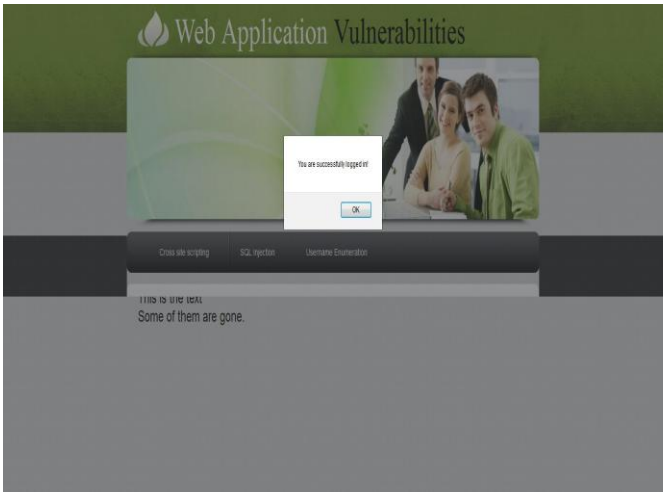
Username: xyz' or 1=1--

Password: ••••••••••••••

New user?Signup Here

(Note: To check sql injection, use username:xyz' or 1=1--)

Following screen shows Successful login screen if user enter sql query. This is demonstration of sql injection.[5]



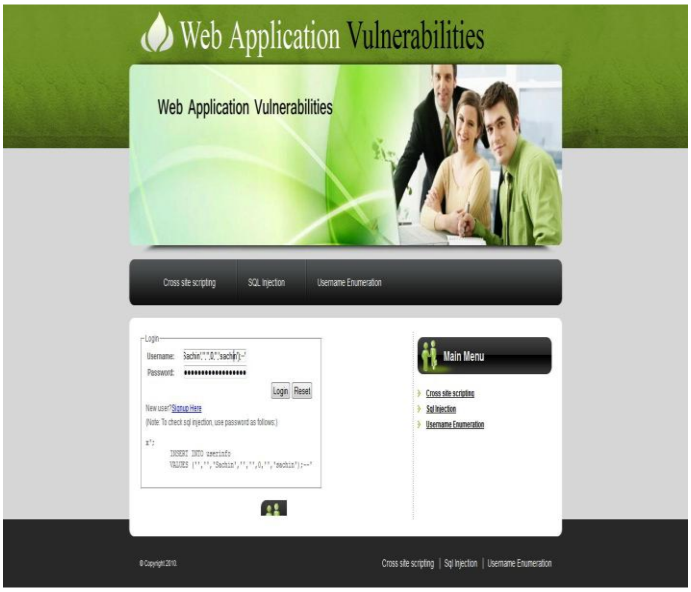
You are successfully logged in!

This is the text

Some of them are gone.

Following screen shows second example of sql injection where user enters insert sql query.

If user knows table names then he can write insert sql query as follows:



Login

Username: Sachin','','0','','sachin');--'

Password: ••••••••••••••••••

New user?Signup Here

(Note: To check sql injection, use password as follows:)

```
x';
     INSERT INTO userinfo
     VALUES ('','','Sachin','','',0,'','sachin');--'
```

Cross site scripting [6](also referred to as XSS) is a vulnerability that allows an attacker to send malicious code (usually in the form of JavaScript) to another user. Because a browser cannot know if the script should be trusted or not, it will execute the script in the user context allowing the attacker to access any cookies or session tokens retained by the browser.

This XSS variant usually appears when a PHP script is using one of following variables without filtering them:

PHP_SELF[19]
REQUEST_URI
SCRIPT_URL
SCRIPT_URI

Those variables are set either by Apache or the PHP engine. Apache is automatically ignoring anything in the URI after the .php extension for mapping script filename, but these variables are containing the full URI.

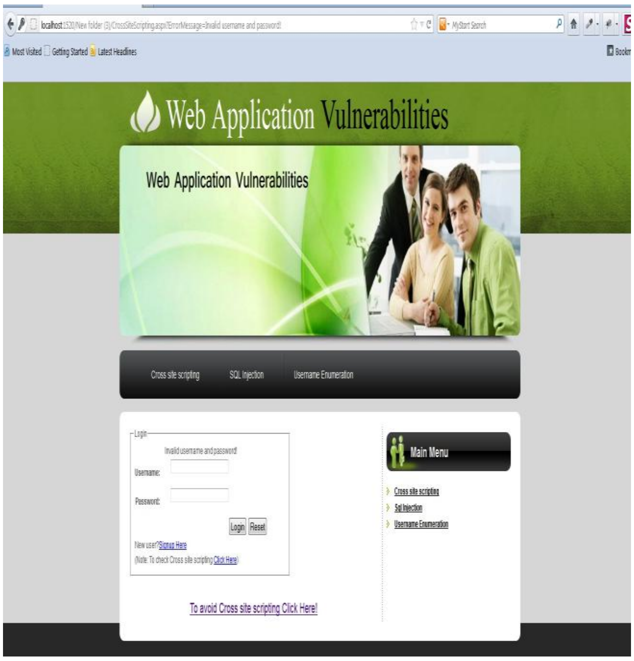
Impact

Malicious users may inject JavaScript,[19] VBScript, ActiveX, HTML or Flash into a vulnerable application to fool a user in order to gather data from them. An attacker can steal the session cookie and take over the account, impersonating the user. It is also possible to modify the content of the page presented to the user.

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications that enables attackers to inject client-side script into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same origin policy. Cross-site scripting carried out on websites accounted for roughly 80% of all security vulnerabilities documented by Symantec as of 2007.[1] Their effect may range from a petty nuisance to a significant security risk, depending on the sensitivity of the data handled by the vulnerable site and the nature of any security mitigation implemented by the site's owner.

Here the error message is dumped into URL. User can change this URL and can write any JavaScript code to execute.

(See URL: ErrorMessage=Invalid username and password!) Following screen shows Cross-site scripting:



localhost:1520/New folder (3)/CrossSiteScripting.aspx?ErrorMessage=Invalid username and password!

Invalid username and password!

Username:

Password:

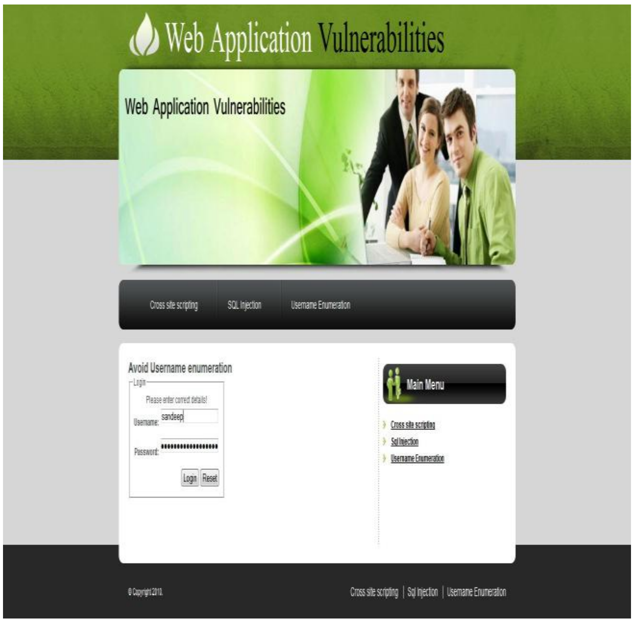
New user?Signup Here

(Note: To check Cross site scripting Click Here)

To avoid Cross site scripting Click Here!

.

Following screen avoids Cross site scripting[6] attack as it does not contain error message in url:



Avoid Username enumeration

Please enter correct details!

Username: sandeep

Password: ••••••••••••••••••

Username enumeration [7]is a vulnerability that allows an attacker to predict username and password of web application. Login facilities (login pages) are the most popular way to find username enumeration on web apps. However, for this very reason, its popularity, security-aware developers might have already considered the issues related to having a login error reveal existing usernames. In other words, you're less likely to enumerate usernames through a login page. Please note that I'm not saying it's rare to find username enumeration vulnerabilities on login pages, but rather that they are simply less likely to be found than other types.

The second problem with enumerating usernames through a login failure is that you are at risk of locking out accounts if a lockout account policy is enabled. Although only one authentication error should not lock out an account, you're playing with fire. Say that you're writing a script to enumerate usernames using a dictionary attack. While tweaking the script you may probe some usernames more than once, therefore taking the risk of locking out the target accounts.

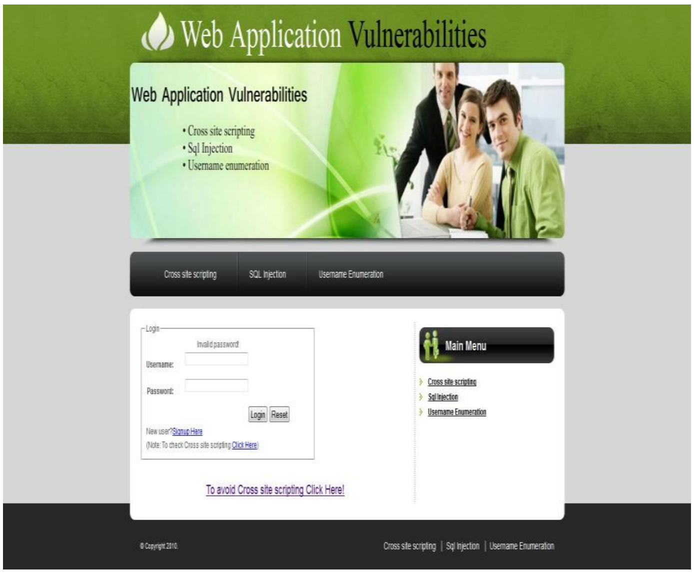
For the second method, analyzing changes in error messages password recovery facilities, again, as an attacker/pentester you're exploiting differences in the application response. Typically we find a Forgot password feature that allows you to receive an email with a new password (or a link that allows you to set a new password). All the user usually needs to do is enter his username or email address. Now, sometimes the email address is used as the username to log into the application. In fact, this is the case on most e-cart sites. Designing the application to use the user's email address as the username is common because it's less likely for someone to forget his email address than a login name.

Remember: there many web applications that allow users to set their username to something different to their email address. Thus, making automated username enumeration more feasible.

Following screen shows Username enumeration attack:

Username enumeration [7] vulnerabilities can be found when error messages of login authentication are carefully observed. In such applications, by trial and error technique, user can get clear idea of data present in database and he then successfully can crack the application.

In following screen if user enters correct username and wrong password, the message gives the clue to user, so that he can try for other password.



Invalid password!

Username:

Password:

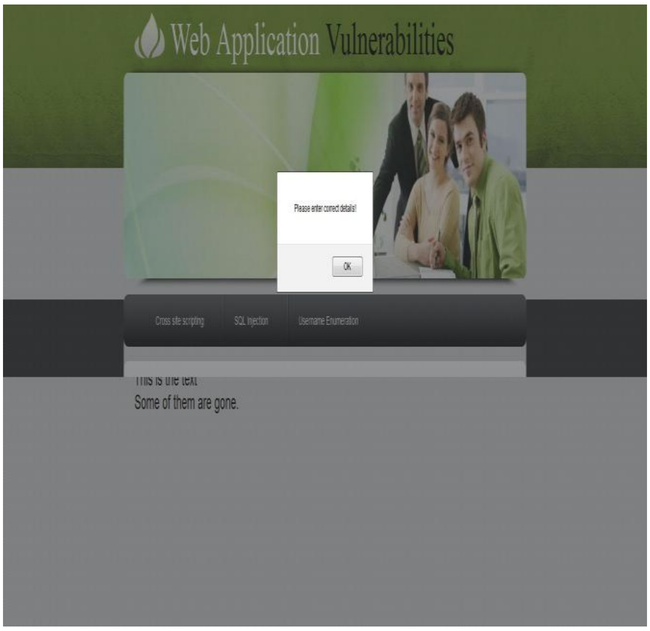
New user?Signup Here

(Note: To check Cross site scripting Click Here)

To avoid Cross site scripting Click Here!

Following screen avoids Username enumeration attack:

Here the message does not give any clue to the user thereby avoiding username enumeration attack [7].



Please enter correct details!

This is the text
Some of them are gone.

III. SOLUTION

SQL Injection [19] is the attack done by hacker on data layer of the system. Suppose we are having a query that that fetch records from database as follows:

"SELECT * FROM `users` WHERE `username` = "' + userName + "';"[19]

In query like this user can enter username as '1'='1' which is always true and hence user can easily login to system.

We will make use of stored procedures to avoid this type of attack. As stored procedures are present on database server then there is less chance vulnerability.

Cross Site Scripting [6] enables attackers to inject client-side script into web pages viewed by other users. Cross-site scripting (XSS) attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user.

e.g. The attacker can add his own script in the url to get sensitive information as follows.

http://www.phpnuke.org/user.php?op=userinfo&uname =<script>alert (document.cookie) ;< /script>

To avoid this type of attack we will do URL encoding in our project.

Username enumeration [6]is a type of attack where the backend validation script tells the attacker if the supplied username is correct or not. Exploiting this vulnerability helps the attacker to experiment with different usernames and determine valid ones with the help of these different error messages. Also user can try the username such as test, testadmin etc which are generally created during testing of a web site.

To avoid this type of vulnerability, we will not allow user to give usernames while registration, we will auto generate the use rid and password.

IV. CONCLUSION

Thus, using above solution we have avoided the web application vulnerabilities such as

- Sql Injection
- Cross site scripting
- Username enumeration

REFERENCES

[1] Basta, A., & Halton, W. (2007). *Computer Security and Penetration Testing* (1st ed.). Delmar Cengage Learning.

[2] Berre, A. J., Roman, D., Landre, E., Heuvel, W. V. D., Skår, L. A., Udnæs, M., Lennon, R., et al. (2009). Towards best practices in designing for the cloud. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications* (pp. 697-698). Orlando, Florida, USA.

[3] Towards best practices in designing for the cloud by Berre, A. J., Roman, D., Landre, E., Heuvel, W. V. D., Skår, L. A., Udnæs, M., Lennon, R., & Zeid, A. (2009).

[4] Christodorescu, M., Sailer, R., Schales, D. L., Sgandurra, D., & Zamboni, D. (2009). Cloud security is not (just) virtualization security: a short paper. In *Proceedings of the 2009 ACM workshop on Cloud computing security* (pp. 97-102). Chicago, Illinois, USA

[5] Cloud security is not (just) virtualization security: a short paper by Christodorescu, M., Sailer, R., Schales, D. L., Sgandurra, D., & Zamboni, D. (2009).

[6] Cloud computing is a trap, warns GNU founder | Technology | guardian.co.uk. (n.d.).

[7] Cloud Security Alliance (CSA) – security best practices for cloud computing. (2009). Retrieved April 16, 2010, from http://www.cloudsecurityalliance.org/

[8] *Cloud Security Alliance Guidance Version 2.1*. (2009). Cloud Security Alliance.

[9] CloudStandards. (2010 3). . Retrieved April 16, 2010, from http://cloud-standards.org/wiki/

[10] Hayes, B. (2008). Cloud computing. *Commun. ACM*, *51*(7), 9-11.

[11] Cloud computing by Hayes, B. (2008).

[12] Krügel, C., Toth, T., & Kirda, E. (2002). Service specific anomaly detection for network intrusion detection. In *Proceedings of the 2002 ACM symposium on Applied computing* (pp. 201-208). Madrid, Spain

[13] Milne, J. (2010, February 9). Private cloud projects dwarf public Initiatives

[14] Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., & Zagorodnov, D. (2009). The Eucalyptus Open-Source Cloud-Computing System. In *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid* (pp. 124-131). IEEE Computer Society.

[15] The Eucalyptus Open-Source Cloud-Computing System by Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., & Zagorodnov, D. (2009).

[16] OWASP. (2010 2). . Retrieved April 16, 2010, from http://www.owasp.org/index.php/

[17] Open Grid Forum. (2010). . Retrieved April 16, 2010, from http://www.ogf.org/

[18] Raj, H., Nathuji, R., Singh, A., & England, P. (2009). Resource management for isolation enhanced cloud services. In *Proceedings of the 2009 ACM workshop on Cloud computing security* (pp. 77-84). Chicago, Illinois, USA

[19] Saraswat, Vijay. (2010). *Report on the Programming Language X10*. x10-lang.org.

[20] Tsai, W., Jin, Z., & Bai, X. (2009). Internetware computing: issues and perspective. In *Proceedings of the* First *Asia-Pacific Symposium on Internetware* (pp. 1-10). Beijing, China