International Journal of Computer Science and Informatics

Volume 2 | Issue 1 Article 5

July 2012

Defending Against Modern Threats in Web Applications

Mohit Kumar
Department Computer Science, Juit, Solan, Himachal Pradesh, India, mohitsmailbox13@gmail.com

Abhishek Gupta
Department Computer Science, Juit, Solan, Himachal Pradesh, India, abhi.g2010@gmail.com

Azhar Shadab
Department Computer Science, Juit, Solan, Himachal Pradesh, India, azharshadab786@gmail.com

Lokesh Kumar
Department Computer Science, Juit, Solan, Himachal Pradesh, India, lokesh.chandravanshi@gmail.com

Vikas Kumar Tiwari
Department Computer Science, Juit, Solan, Himachal Pradesh, India, vikastiwari87@gmail.com

Defending Against Modern Threats in Web Applications

Mohit Kumar, Abhishek Gupta, Azhar Shadab, Lokesh Kumar & Vikas Kumar Tiwari

Department Computer Science, Juit, Solan, Himachal Pradesh, India
E-mail : mohitsmailbox13@gmail.com, abhi.g2010@gmail.com , azharshadab786@gmail.com, lokesh.chandravanshi@gmail.com , vikastiwari87@gmail.com

***Abstract* -** Web applications have become critical part of business. They hold a treasure trove of data behind their front ends. Now-a-days attackers are well aware of the valuable information accessible through web applications, so website security has become a major problem today. The number of vulnerabilities have multiplied in recent years. Vulnerabilities like cross site scripting(XSS),sql injection and cross site request forgery(CSRF) has emerged as a major threat to web applications. So, in order to protect web applications from these modern threats, at first vulnerability assessment should be carried out from time to time and also some preventive techniques should be followed to prevent these threats. The motivation of this paper is to promote the use of automated tools for vulnerability assessment and to follow preventive techniques in order to make web applications secure.

***Key words* -** *Vulnerability,vulnerability assessment,cross site scripting(XSS),sql injection and cross site request forgery(CSRF).*

I. INTRODUCTION

In computer security,vulnerability is a weakness which allows an attacker to reduce a system's information assurance.Now-a-days attackers are well aware of the valuable information accessible through web applications.Organizations use intrusion detection system and firewalls to save their product,but these firewalls must keep ports 80 and 443(SSL) open to conduct online business.These ports represents open doors to attackers,who have figured 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 very little or no security.Once the development phase gets over,the applications are put through quality assurance(QA) testing which typically focuses on performance and functionality,rather than security.As a result,an application that emerges from QA with flying colors can still be riddled with exploitable flaws.As web applications are becoming increasingly complex,tremendous amount of sensitive data including personal,medical and financial information are exchanged and stored.The need for securing web applications have become very important.securing a web application starts with discovering web applications vulnerabilities and then providing techniques for its remediation.Few years back there existed no proper tools search for vulnerability assessment,but now there are tons of tools available.Tools that are commonly used to find a vulnerable website are acunetix,nessus,retina,metasploit framework etc.The tool which we have used here is acunetix.It automatically checks the website for sql injection,cross site scripting(XSS) and cross site request forgery(CSRF).After the vulnerability assessment is over we have presented the preventive techniques that should be followed for counter-measuring these web threats.In this paper we have shown how vulnerabilities like sql injection,cross site scripting(XSS) and cross site request forgery(CSRF) can be prevented.First we will talk about vulnerability assessment and then we will move to prevention technique.

II. VULNERABILITY ASSESSMENT

A vulnerability in IT security is a group of conditions,which taken together,can leave a system open to unwanted access or unauthorized use by an intruder,denies availability of the system or otherwise represents a violation of the system security policy.It is imperative that companies identify and address vulnerabilities in their web applications.Companies that fails to protect themselves against the threat of web application exploits risk of loss of confidential data,business interruption and also damage to their reputation.So vulnerability assessment has become a very important part in keeping the web application secure.

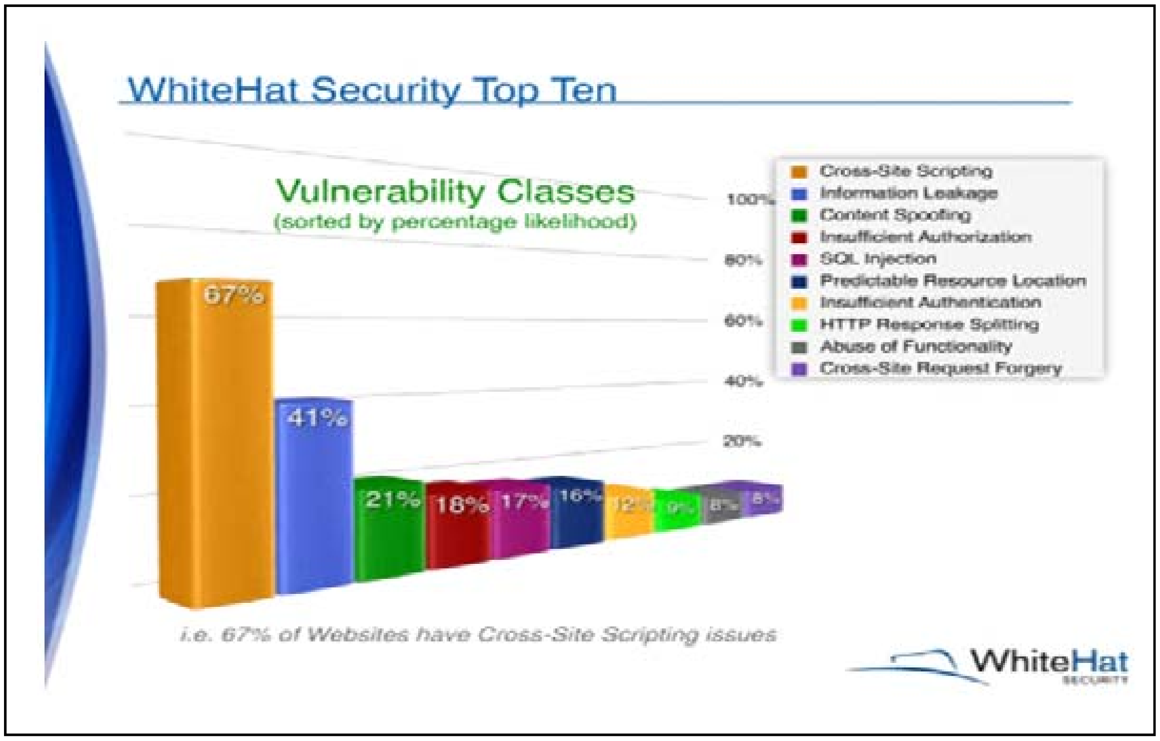
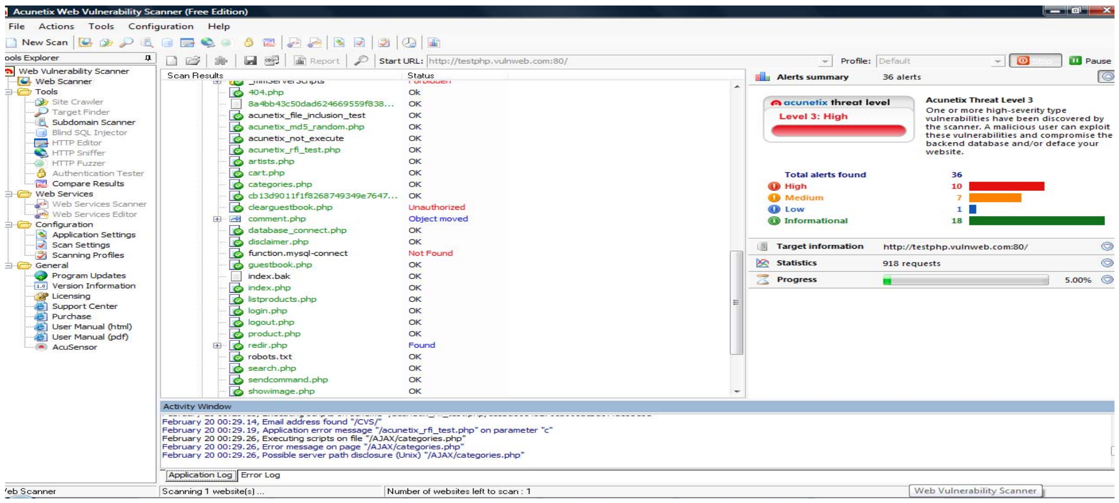


Fig. 1 : Shows the statistical view on current website vulnerabilities

It is important to know that vulnerability assessment tools should only be used on networks and computer systems with permission of owner of those networks and systems. The vulnerability assessment tool which we have used here is acunetix and we have performed the vulnerability scan on http://testphp.vulweb.com:80/. This test page was provided by acunetix company itself.



Start URL: http://testphp.vulnweb.com:80/ Profile: Default

Scan Results	Status
404.php	OK
8a4bb43c50dad624669559f838...	OK
acunetix_file_inclusion_test	OK
acunetix_md5_random.php	OK
acunetix_not_execute	OK
acunetix_rfi_test.php	OK
artists.php	OK
cart.php	OK
categories.php	OK
cb13d9011f1f8268749349e7647...	OK
clearguestbook.php	Unauthorized
comment.php	Object moved
database_connect.php	OK
disclaimer.php	OK
function.mysql-connect	Not Found
guestbook.php	OK
index.bak	OK
index.php	OK
listproducts.php	OK
login.php	OK
logout.php	OK
product.php	OK
redir.php	Found
robots.txt	OK
search.php	OK
sendcommand.php	OK
showimage.php	OK

Alerts summary 36 alerts

acunetix threat level
Level 3: High

Acunetix Threat Level 3
One or more high-severity type vulnerabilities have been discovered by the scanner. A malicious user can exploit these vulnerabilities and compromise the backend database and/or deface your website.

Total alerts found	36
High	10
Medium	7
Low	1
Informational	18
Target information	http://testphp.vulnweb.com:80/
Statistics	918 requests
Progress	5.00%

Activity Window

```
February 20 00:29.14, Email address found "/CVS/"
February 20 00:29.19, Application error message "/acunetix_rfi_test.php" on parameter "c"
February 20 00:29.26, Executing scripts on file "/AJAX/categories.php"
February 20 00:29.26, Error message on page "/AJAX/categories.php"
February 20 00:29.26, Possible server path disclosure (Unix) "/AJAX/categories.php"
```

'eb Scanner | Scanning 1 website(s) ... | Number of websites left to scan : 1

Fig. 2 : Shows vulnerability scan of a website using acunetix.

After the scan gets over acunetix will provide very useful information like the server and the operating system used by the target website.It also tells you about the number of attacks to which the website is vulnerable to and it also shows the total number of alerts found after the scan.Once the vulnerability scan is completed,Acunetix delvers an executable summary report as well as detailed summary report to the user.The detailed report includes an in-depth technical explanation of each vulnerability as well as instructions for remediation.So,vulnerability assessment has become a very important step in making the web applications more secure.

III. PREVENTION

Although, web application's development have evolved over the years but modern web threats are still seen as a major challenge in web applications. Today the web applications are protected by traditional network security techniques, like firewall and cryptography-based mechanism. The use of specific secure development techniques can help to mitigate the problem,however they are not always enough.So,in this section we present prevention techniques that should be followed to make web applications even more secure.We will discuss mainly about the preventive techniques for sql injection,cross site scripting(XSS) and cross site request forgery(CSRF).

3.1 SQL INJECTION

It is a code injection technique that exploits a security vulnerability in websites.SQL INJECTION attacks are not limited to ASP.NET applications.Classic ASP,JAVA,JSP and PHP applications are equally at risk.While it is easy to point to one or two key measures for the prevention of SQL INJECTION attacks,its best to take a layered approach to the problem,This way if one of your measures is circumvented because of some vulnerability,you are still protected.The recommended layers are summarized in Fig-3

Principle	Implementation
Never trust user input	Validate all textbox entries using validation controls, regular expressions, code, and so on
Never use dynamic SQL	Use parameterized SQL or stored procedures
Never connect to a database using an admin-level account	Use a limited access account to connect to the database
Don't store secrets in plain text	Encrypt or hash passwords and other sensitive data; you should also encrypt connection strings
Exceptions should divulge minimal information	Don't reveal too much information in error messages; use customErrors to display minimal information in the event of unhandled error; set debug to false

Fig. 3 : Preventing sql injection attack.

3.2 CROSS SITE SCRIPTING(XSS)

It is a type of computer security vulnerability typically found in web applications that enables attackers to inject client-side script into web pages.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.It can be prevented by following these steps.

1 Identify all system components other than the application itself. In a typical web application you will have at least the following:

1.1 Database

1.2 Browser output, which further consists of:

a. HTML

b. JavaScript

c. CSS

d. Response headers (e.g. redirection, cookies, etc)

2. Adopt one character encoding (use UTF-8) and make sure all components are configured to use it:

2.1 Databases typically need to be created with a character encoding in mind.

In the HTML pages you create, set the character encoding explicitly.

3. Then, for every component:

3.1 Identify safe characters.

3.2 Identify how to make unsafe characters safe by converting them into something else

3.3 Write a function that looks at characters one by one to determine if they are safe, and converts those that are not (whitelisting, not blacklisting!)

3.4 Every such function must be aware of the character encoding used in the application.

4. Then, for every piece of code that sends data from one component into another, make sure you use the correct function to encode data to make it safe

5. Check that every piece of data you receive is in the correct character encoding and that the format matches that of the type you are expecting (input validation). You must use whitelisting (as blacklisting does not work). This is especially important for user-supplied Internet addresses.

3.3 CROSS SITE REQUEST FORGERY(CSRF)

Cross site request forgery is a type of exploit where unauthorized commands are transmitted from a user that a website trusts.It exploits the trust that a site has in a user’s browser.With a little help of social engineering (like sending a link via email/chat) an attacker may force the users of a web application to execute actions of the attackers choosing. Preventing CSRF requires the inclusion of an unpredictable token as part of each transaction. Such tokens should at a minimum be unique per user session, but can also be unique per request.

1. The preferred option is to include the unique token in a hidden field. This causes the value to be sent in the body of the HTTP request, avoiding its inclusion in the URL,which is subject to exposure.
2. The unique token can also be included in the URL itself, or a URL parameter.However, such placement runs the risk that the URL will be exposed to an attacker, thus compromising the secret token.

IV. CONCLUSION

Web applications have become a popular and wide spread interaction medium in our daily lives.At the same time vulnerabilities that endangered the personal data of users are discovered regularly.There are various methods that can be used to discover web applications security vulnerabilities.However we encourage the use of automated tools for vulnerability assessment of web applications as it saves both time and money.We have presented few preventive techniques for the prevention of sql injection,cross site scripting(XSS) and cross site request forgery(CSRF) vulnerabilities in web applications.Our analysis is sound.And by following these methods we can defend web applications from these modern web threat.

REFERENCES

[1] Joaquin Garcia-Alfaro and Guillermo Navarro-Arribas “Prevention of Cross-Site Scripting Attacks on Current Web Applications” Springer-Verlag Berlin, Heidelberg , OTM'07 Proceedings of the 2007 OTM confederated international conference on the move to meaningful internet systems: COOPIS, DOA, ODBASE, GADA, and IS - Volume Part II.

[2] M. Bishop,“ Computer Security: Art and Science” Addison Wesley Professional, 2002.

[3] Nenad Jovanovic, Christopher Kruegel, and Engin Kirda “Pixy: A Static Analysis Tool for Detecting Web Application Vulnerabilities” IEEE Symposium on Security and Privacy (S&P’06) 1081-6011/06.

[4] Mattia Monga, Roberto Paleari and Emanuele Passerini “A hybrid analysis framework for detecting web application vulnerabilities” SESS’09, May 19, 2009, Vancouver, Canada 978-1-4244-3725-2/09/ 2009 IEEE.

[5] Fanglu Guo,Yang Yu and Tzi-cker Chiueh “Automated and Safe Vulnerability Assessment” Computer Security Applications Conference, 21st Annual 2005 , Page(s): 10 pp. – 159.

[6] www.acunetix.com/

[7] www.whitehatsec.com/

◈◈◈