Development of Vulnerable E-Commerce

Firouzeh Jalilian

December, 2005

1. Abstract

As Web applications are on the rise, their security is becoming an important issue due to continually evolving threats. Among Web applications, e-commerce is one of the most attractive targets for hackers given its potential financial rewards/impacts. The whole business is at a high risk by having an insecure e-commerce website. E-commerce websites always involve databases, and by browsing products and purchases, visitors have a channel to talk to the database and manipulate it.

1. Introduction

E-Commerce refers to the exchange of goods and services over the Internet. Nowadays every major store has a website available for customers to purchase products over the internet. The security of these e-commerce websites have to be insured otherwise their at risk. Security has three main concepts: confidentiality, integrity, and availability. Confidentiality ensures that information would not be accessed by unauthorized people. Integrity insures that information would not be altered by unauthorized; and availability ensures that one has access and is authorized to resources.

The next phase of this research is similar to the research previously done by Michel Cukier by analyzing the gathered quantitative attack data. During that research the attackers were monitored toward their architecture and it showed that port scans only proceed attacks 5% of the time. The goal of this research is to make an e-commerce website with architecture vulnerable enough to collect attack-related data.

In this paper we focus on vulnerabilities in e-commerce websites that also exist in the website we developed. The developed e-commerce website called "Doc's Sporting Goods" was vulnerable in many different ways. The website was implemented with PHP and MySQL; therefore, it is vulnerable to MySQL injection attack. There were no encryptions used in the transferring of the data. The website also had leakage of information in different ways. Another vulnerability of the website is that the client's input was not fully validated. Besides these vulnerabilities there were other minor ones as well.

2. MySQL Injection Attack

Web applications interact with databases to dynamically create customized data views for each user. SQL injection is a technique used to take advantage of non-validated input vulnerabilities to pass SQL commands through a Web application for execution by a backend database. Attackers can type some malicious SQL queries in the form inputs that would therefore be executed on the backend database server. Some examples of the attacks are as following.

For example, a product could be reviewed using the following URL:
`http://128.8.46.224/review.php?id=721069`
A malicious visitor can do the following:
`http://128.8.46.224/review.php?id=721069; DROP TABLE Products`
The semicolon is used to pass the database server multiple commands on one line and the `DROP TABLE Products` causes SQL server to delete the table `Products.`

The attacker can also retrieve data from other tables by using `UNION SELECT` statement. This statement allows the chaining of two separate SQL commands that have nothing in common. For example, the attacker can try the URL below:
`http://128.8.46.224/review.php?id=721069 UNION SELECT user-name, password FROM Users`
The result of this query is two columns, containing the results of the first and second queries respectively including all the user names along with their passwords.

Another possible attack is at the time of login:
If your query to check the username and password entered by the user was this:
`"SELECT * FROM users WHERE username = '".$_POST['username']."' AND password = '".$_POST['password']."'"`
Someone could login by using any username and for the password they would type `' OR ''=''` which would be placed into your MySQL query changing it to be:
`"SELECT * FROM users WHERE username = 'anyuser' AND password = '' OR ''=''"`
The user will probably get logged in and would get access to all the information about all the users in the database.

In most cases, users have `magic_quotes_gpc` turned on (the default for PHP) which will add backslashes to escape all ' (single-quote), " (double quote), (backslash) and

NULL characters. But, this is not foolproof because there are other characters that should be escaped to be safe. There is a function built into PHP that will escape all MySQL characters that could be used to inject additional SQL into your queries. The function is `mysql_real_escape_string()`.

3. No Encryption Used

Encryption is a generic term that refers to encoding of data so that those data can be securely transmitted via the Internet. Public key encryption is more widely used than private key encryption for the purpose of e-commerce. The big improvement of public key encryption is the usage of two keys: one key is public and the other is private. Public key encryption makes the transfer of data secure even if the physical network is insecure. If the data is captured, the data would be meaningless unless the cracker has the private key for decrypting the data. The "Doc's Sporting Goods" did not use any encryption algorithms for transferring and storing the data. Therefore, if the malicious user gets access to critical data while getting transferred or directly from the database the data can be easily read. The most common use of Public Key Encryption is the use of Digital Certificates issued by trusted third parties. The certificate authority or the trusted third party makes sure the public key being used belongs to the intended message recipient.

4. Information Leakage

A cookie is some information that your browser stores on your computer at the request of a web server, and passes back to the web server that created the cookie every time one connects to that web server. It is created when a web server asks your browser to store it. In the development of "Doc's Sporting Goods" there are many sessions and cookies created throughout the usage of the website. Specifically the website stores information in a cookie about the person who had last used that computer to purchase items. The cookie stores information about that user including the person's account number in order to get access to all the past purchases of that user. This has a high leakage of information if a malicious user gets access to this cookie. Since the account numbers are generated sequentially for all members of the website, the cracker can get a sense of all the account numbers available. Therefore, the cracker can change the account number in the cookie and view the past purchases of different users.

Another leakage of information available in the “Doc’s Sporting Goods” website is the MySQL errors displayed from time to time. Even though there were many error checking done in the MySQL queries, there were still some chances of the users being exposed to MySQL/PHP errors. This is a leakage of information since the error would give away information such as the line number in the PHP code or the exact MySQL query.

5. Weak Input Validation Done on Client Side

Input validation refers to how your application filters or rejects input before additional processing. In the “Doc’s Sporting Goods” website there were some basic input validations such as making sure a number is entered for number of items added to the shopping cart, the zip code, phone number, and credit card number. The website would also validate for the number of digits entered for zip code, phone number, and credit card number. Another validation was to make sure the credit card has not been expired. However, the website is still lacking major input validations such as the credit card being a valid credit card, or address and phone number being valid.

Another weakness of the validation being done by “Doc’s Sporting Goods” is that the validations are all done with JavaScript on the client’s side. Even though the client side input validations appear to restrict user input, they actually offer no security benefit at all. All client side checks can be easily bypassed by an attacker by downloading the webpage to the local computer, then removing the validation or by using a proxy.

6. Fail-safe Defaults Not Used

The principle of fail-safe defaults states that, unless a subject is given explicit access to an object, it should be denied access to that object. This principle requires that the default access to an object is none. And the system has to identify condition under which access would be permitted. This way if the user is unable to complete its action or task, or even if the program fails, the system is still safe. The alternative of this approach is the default of having access and finding the conditions under which the access should be refused. The second approach is what is used in the "Doc's Sporting Goods" website which is not as secure as the Fail-safe default.

7. Form Tampering vulnerabilities

There are form tampering vulnerabilities available in e-commerce shopping cart applications. It is possible for an attacker to take advantage of the form tampering vulnerabilities and order items at a reduced price on an e-commerce site. Many web-based shopping cart applications use hidden fields in HTML forms to hold parameters for items in an online store. In "Doc's Sporting Goods" these parameters include the item number, the quantity, and its price. An attacker could modify the HTML form on their local machine to change the price of the item and then load the page into a web browser. After submitting the form, the item is added to their shopping cart at the modified price. "Doc's Sporting Goods" website also uses hidden fields for discount rates; therefore, the attacker can modify the discount rate as well.

8. Conclusion

As e-commerce websites are increasing in number, their security is of utmost importance for customers. The customers are worried about giving away their information, the most important being their credit card information. Therefore, one of the major factors in the success of an e-commerce website is its security. There are many vulnerabilities available for Web applications, specifically e-commerce ones. There are many ways that these vulnerabilities can be prevented such as securing against MySQL injection attacks, use of encryption, not leaking critical information through cookies, good exception handling, server side input validation, usage of fail-safe defaults, and preventing against form tampering. The next step of this research is to deploy this website in a test-bed and to collect data about the attacks and analyze the data.

9. References

- http://www.netlobo.com
- http://cyber.law.harvard.edu
- http://www-128.ibm.com
- http://www.albion.com
- http://www.securityinnovation.com
- http://msdn.microsoft.com
- http://xforce.iss.net