Building security-enabled Web applications

PERES, Paulo Júnior de Jesus [1]

PINTO, Aurélio Guimarães [2]

FREITAS, Caio Guimarães [3]

LEITE, Francisco Canindé da Silva [4]

SILVA, Francisco Eronildo da [5]

OLIVEIRA, Geveson de Souza [6]

RIBEIRO, Dallas dos Santos [7]

ALMEIDA, Cristiany Caliri de [8]

MORAIS, Gilvanete Melo de [9]
Summary

With the growth of the Internet, and your increasing use as a means to provide services and communications, it is necessary that developers of web systems have as much attention as security practices in the development of systems. It is a fact that Build a secure Web application is a very difficult task in the present day, because for a system to be considered safe depends exclusively on the code developed by the programmer, but also application server, database server, the network of communication and many other things that are out of reach of the developer of the system. In this light, the goal of this article is to be a reference as to the main levels of attacks used by Hackers to code vulnerabilities of web systems and the possible ways of prevention. The concepts used in this article can be used in any programming language, because the theories are universal, however for the purpose of illustrating the codes are written in the PHP programming language as it currently is one of the programming languages more spread among the community. In this sense, this article comes to contribute with the community being a quick base of research on prevention of attacks on systems.

1. Introduction

In the current context, it is common for companies, public bodies and institutions look for agility in distribution of information to customers and employees agilizing the decision-making process in this medium. However in order for this strategy to work, it is imperative that there is safety in handling and display of information relating to the company's business.

In this sense, the Internet comes as the means to facilitate and provide such data distribution and activity information. This opportunity, virtual commerce or e-commerce. According To Teixeira:

*In Brazil, currently, 45.6% of your population has internet access (about 90 million people). If we make a comparison between the years of 2000 and 2012, we saw a significant increase, approximately 1,500%, the number of internet users in Brazil (TEIXEIRA, 2015, p. 19).*

Electronic commerce or *e-commerce*, according to Salvador (2013) can be defined as commercial operations made in the virtual environment, with the help of electronic means, i.e., is able to buy a particular item to kilometers away, without leaving home or Office, using smartphone, computer or other device. Where the convenience, ease of access and, mainly, by the prices consumers are attracted to *e-commerce*.

However, it is necessary that the developers pay attention to information security when developing applications for the Internet, because the information is available to all types of users. According to Aguilar, follows the concept of information security:
Information security refers to the existing protection of the information of a particular company or person, that is, apply the corporate information about people. Information means all and any content or data that has value to any organisation or person. It can be stored for use restricted or exposed to the public for consultation or acquisition. (AHMAD, 2008).

According to Ferreira (2017), in this universe of opportunities that the internet has become, malicious users have the opportunity to steal sensitive information like Bank passwords; confidential information, among other possibilities.

The purpose of this article show some techniques used to prevent yourself from information theft or invasion of systems and measures to ensure that developers of systems that operate via internet (WEB DEVELOPEMENT) develop secure systems, reducing vulnerability to attacks of various levels.

2. XSS (CROSS SITE SCRIPTING)

Second Flux (2017), the XSS occurs when a Web application take malicious data sent by the user. In general the intruder will take advantage of the lack of malice of the person who developed the Web application to inject malicious code that will fool the application or to collect restricted information system.

An example of this vulnerability is in the method of transmission to Get type variables (variables passed in the URL of the site) such as: http://www.igreja.com.br/mostar.php?site=sexo.php.

This can generate a real nuisance to the site because the malicious user could pass any website address above, what you could do for example, that this page would work as a pornographic site.

According to Flux (2017) to avoid that the application be vulnerable to this kind of attack, steps can be taken as:

- Using variables of type for sensitive data is contraindicated, because it compromises the security of the system;
- Use filters in the capture of user data for that block any kind of malicious code or URL is injected by external users. These filters can be implemented by the programmer or by capabilities of the application server.

3. SQL INJECTION

According to Mueller (2013) the attack via SQL Injection is the injection of SQL code in variables passed to Web forms and aims to run arbitrary SQL code in a Web application.
This is a very common and dangerous attack, an example is the following script called "remover.php", follows the transcription of the code:

```php
$sql = "DELETE FROM customer WHERE code =" . $_GET['codigo'] . ";
mysql_query ($sql) or die (mysql_error ());
```

Analyzing the above script, the following is an example of how the malicious user can perform a SQL INJECTION attack via URL: http://seusite.com.br/remover.php?codigo=666+OR+1. In this way, using this statement, the result would be deleting all the records from the customers Entity.

According to the *online* manual of PHP, the safest way is by controlling the type of variables that are being received by the forms fields, so, as a form of prevention, the following is an example of code:

```php
settype ($offset, 'integer');
$query = "SELECT id, name FROM products ORDER BY name LIMIT 20 OFFSET $offset;";
```

Please note the%d in the format string, using%s would be useless
```
$query = sprintf ("SELECT id, name FROM products ORDER BY name LIMIT 20 OFFSET% d;", $offset);
```

In the above code, the function "settype" has the function to test if the variable received really is of type integer, preventing malicious code snippets are incorporated in the variable.

**4. Regular Expression Denial of Service (secrets would be made known)**

The author Focke (2017) describes Regular Expressions (REGEX) as a technique to use functions of the programming language in use to treat information from the user. These functions try to encode the value of the variables so that they cannot damage or circumvent the system.

Follows in example in ASP programming language:
Follows another example using the PHP programming language:

```php
&lt;?php
If (!ereg "^\[0-9,a-z,A-Z\](+)\[.,_,-\]\[0-9,a-z,A-Z\](+)\[.,_,-\]\[0-9,a-z,A-Z\](+)\[.,_,-\]\[0-9,a-z,A-Z\](+)\[.,_,-\]\[0-9,a-z,A-Z\](+)\[0-9,a-z,A-Z\](+)\[.,_,-\]\[0-9,a-z,A-Z\](+)\[0-9,a-z,A-Z\]([a-z,A-Z]* ) (2.3[0-9,a-z,A-Z]\[.) ?"", $_ POS ("variable") )
"echo "invalid characters in the email field!";"
?
?&gt;
```

According to Mattos (2013), to prevent this type of attack, it is necessary for the developer to create capture filters to correct the errors of the algorithms run REGEX code standards by the programming language used.

5. CGI APPLICATIONS

Second Foster (2017) Common Gateway Interface (CGI), is a type of matalinguagem or platform-independent middleware provided by the Web servers that allow you to run programs/scripts from a URL. These scripts are usually written in Perl, Shell, Tcl, Java, Python or C (most written in interpreted languages) and located in a directory/cgi-bin.

CGI applications written without care may cause serious problems for vulnerabilities of Web servers. Second Foster:

*CGI programming differs from traditional programming, basically, due to your environment and your gateway function between the client and the server, which ends up influencing much in security aspect (it should be remembered that these applications operate in an insecure environment and that, Theoretically, can be executed by anyone). (Foster, 2017)*

A CGI script running under the same Web server UID is not necessarily a bad suit, but if any CGI application has a security hole that allows an attacker to run programs under UID of the webserver, it can cause a serious problem to your site. (Foster, 2017).
Second Person (2017), one way around this problem is via "WRAPPERS", that is, programs that involve other related programs to change the way they operate. Thus, in environments where users write independent CGI applications, is a good strategy to isolate them from one another, that is, implementing mechanisms in your server so that access a user's scripts will not interfere with other users' data. suexec (http://www2.idiscover.co.uk/apache/docs/suexec.html) solves this problem (there are other tools that also treat this problem) causing applications to run CGIs under the UID of the user himself, that is, the owner of the CGI application.

6. DATA TYPE

According to the online manual of PHP, that care must always be taken is that some programming languages use of global variables that meet any type of data. This is incorrect usage, because it causes a considerable level vulnerability into the system.

Person (2017) describes some examples of crossing the variables in the PHP programming language:

```php
$cod = $_GET['cod'] // tipo get
$cod = $_POST['cod'] // tipo post
$cod = $_SESSION['cod'] // tipo session
$cod = $_COOKIE['cod'] // tipo cookie
```

7. WRITE FILES VIA UPLOAD

Whenever you write to a file via UPLOAD the developer must be sure of the type of file being recording on the disc. For example, an image file can simply be treated using a validation of file types. That's because malicious users can send files that affect somehow the server or the application (Person, 2017).

According to the PHP online manual, follows a validation example using PHP programming language:

```php
"lt;? php
```
function type_up ()
{

    checks to see if the mime-type of the image file

    If (! eregi ("^ image /(pjpeg | jpeg png gif bmp | | |) $", $this-&gt; fil["type"]e)) {

        return 0;

    } else {

        return 1;

    }

}

8. INSURANCE FORMS WITH SYNCHRONIZER TOKEN

According to Bragil (2017), sincronizadora plug is a technique that uses a variable created in the beginning of the session, with an initial value set by the developer in order to ensure that the form was actually created by the server application.

According to Brazil (2017), to a site with authentication, you can start the chip value at the time of LOGIN, then make sure that the user is valid:

Example in PHP:

<?

    session_start ();

    ...

    Initializes the sincronizadora plug with a random value
$_SESSION["synctoken"] = uniqid (time ());

....

?>

Now the plug sincronizadora recorded in session, we can illustrate how to create a form protected by her:

<form name="frmteste" method="post" action="cadastro.php">
</form>

...

<!-- coloca todos os campos do formulário -->

<!-- a nossa ficha fica como um campo hidden -->

<input type="hidden" name="token" value="$_SESSION["synctoken"]">

...

<div class="nicdark_archive1 nicdark_padding010" style="box-sizing:border-box;"></div></div>

Viewing the source code, notice that the VALUE attribute of the TOKEN has a random value field, type "45jfg58955699j96uy8gtj85". Synchronization happens while processing the form. In our example, the ACTION of the form is the file "cadastro.php" that when you receive the form data, will test whether the value of the TOKEN field of the form is the same as the variable $ _SESSION sessio["synctoken"]n. If not, it's a sign that someone is trying to do what should not, in other words, it's probably someone who saved the page with the form on your micro and is trying to do some "art". In this case, you can take some steps, how to collect the citizen, display a sentence intimidating, among others. If the value is equal, green light, you can continue with the processing (Brasil, 2017).

The benefits are numerous, as this is a simple and efficient way to increase the security of a Web system.

Conclusion

The Web was designed without much concern, or almost no, safely. The main objective was to provide a more friendly information resources available at the time. With the rapid growth of the Web and the
diversification of your use, security became a point of crucial importance, especially for the Web as one of the main commercial appeals. In this case the development without security because a crucial impact on profits and on the work of the company.

We could cite other cases of invasion or insecurity in the systems development process, however we treat just the essential points of a secure development.

Finally, this work is fully extended, and other authors can use this article as a reference to expand the amount of attacks and defenses to be used by software development teams.

References


"Cross-SITE Scripting (XSS)". Available at: <http: br.ccm.net/contents/19-cross-site-scripting-xss=""">accessed: 20 Oct.</http: of 17


[1] Graduated in computer science, it acts as a public server on SUFRAMA, as Administrative Analyst-you. Database specialist for ULBRA.

[2] Graduated in computer science, it acts as a public server on SUFRAMA, as Administrative Analyst-you.

[3] Graduated in computer science, it acts as a public server on SUFRAMA, as Administrative Analyst-you.

[4] Graduated in computer science, it acts as a public server on SUFRAMA, as Administrative Analyst-you.

[5] Graduated in computer science, it acts as a public server on SUFRAMA, as Administrative Analyst-you.


[7] Graduated in computer science, acts as server SUFRAMA, as Administrative Analyst-you.

[8] Graduated in business administration, acts as public servant on SUFRAMA, as administrator.

[9] Graduated in economics, acts as public servant on SUFRAMA, as an economist.