An Approach to Security, Performance and Bandwidth Issues in ASP.NET Websites

Sahil Mehta
M.Tech Student (CSE)
Lovely Professional University
Phagwara, India

Parminder Singh
Lovely Professional University
Phagwara, India

ABSTRACT

The global presence of ASP.NET websites is increasing with each passing day. With increase in the number of ASP.NET websites, issues pertaining to website's security vulnerabilities, performance and bandwidth have come up to the surface. The security vulnerabilities in a website enable access to critical data and resources of the website. The most two common security holes in the ASP.NET Websites are SQL Injection and Cross Site Scripting. The performance and bandwidth issues in a website present usability obstacles for its intended users. ASP.NET websites mostly contains server side data bound controls and most of the developers use their default properties leading to performance issues and bandwidth issues. The global distribution of the ASP.NET websites further aggravates the problem of the impact of these issues. This paper aims to discuss issues related to the ASP.NET website's security, performance and bandwidth.

Keywords

ASP.NET, Websites, Vulnerability, Threats, Security, Script; JavaScript; HTML, SQL Injection, Cross Site Scripting XSS.

1. INTRODUCTION

Each day when browsing the internet, we visit a lot of websites some more complex, others just simple personal pages. A Website is a collection of webpages, Images, Script files, text. A website represents a summary of all the content you have put online. A webpage is typically a document written in plain text with proper formatted instructions of hypertext markup languages (HTML). Each web page represents various types of information present to the visitor in an aesthetic and readable manner. Most of the web pages are available on the World Wide Web, which makes them widely accessible to the Internet public. Others may be also available online but only restricted to a certain private network, such as a corporate intranet. Today's time most of the companies have their own websites which provide gates to the customer to read about the company information and its products. Websites are very important for business. Websites can showcase company's product/service in any shape or form. A high quality website provides an online for your business. A website is accessibly globally and any person in this world can reach to your website within fraction of sections. This does not matter from which location they are or how they will reach to you. Websites are gaining popularity day by day. A Website is place online that you own, where you control the information and the data.

There can be static or dynamic webpages. Static web pages consist of the same prebuilt content every time the webpage is loaded, whereas the content of dynamic webpages can be generated on-the-fly.

The standard HTML pages are static in nature. They consist of HTML code describing the structure and content of the webpage. Every time an HTML page is loaded, it looks like the same. The content of an HTML page can change only if its web developer updates and publishes the file.

ASP.NET webpages are dynamic in nature. These webpages consist of "server-side" code which allows the server to generate unique content every time the page is loaded. For instance, the server can show the current date and time on web page. The server, on the basis of web form filled by the user, can also output a unique response. The dynamic webpages can utilize server-side code in order to access database information, enabling the page's content to be developed from information in the database. Websites that generate webpages from database information are called database-driven websites.

ASP.NET is a framework which supports rapid development. Being a part of .NET platform, ASP.NET framework is used to build, deploy and run web applications. Asp.net development framework is used to make web applications and websites with the use of HTML, CSS, jQuery and JavaScript. ASP.NET consists of several built in data control and is best for the rapid development with powerful data access.

Websites developed by ASP.NET contains various vulnerabilities, performance and bandwidth issues. Vulnerability is a weakness in the websites which allows an attacker gain permission in the websites. Due to the presence of vulnerabilities attacker can also shut down the website and reduce the performance of the website. The most common vulnerabilities present in the ASP.NET websites are Injections namely SQL Injection which lets the attackers alters SQL queries send to a database and XSS cross site scripting that allows an attacker to execute his own JavaScript script into the websites which leads to steal account information and spread worms in the web page. These are the two of most dangerous threats according to the OSWAP[1]. Injections attacks advantages improperly coded by the developer to insert and execute/run attacker specified commands, enabling access to critical data and resources. ASP.NET websites consists of server side controls which enable to develop dynamic web pages. Data bound controls to maintain and display data on the web pages from the database. During the development time of the ASP.NET websites, most of the developers use server side control's default properties which enable rapid development of the website. With the presence of default properties the performance of websites gets decreased. In order to develop a high performance and scalable websites requires custom code which will be embedded in the server side controls. For example of custom code is custom paging which will increase the performance of the website rather than using the default property "Paging" of the GridView data bound server side control. Due to the presence of the default

properties the bandwidth of the website also get increased as lots of data transfer takes place. Bandwidth performance is one of the critical requirements of every website. In today's time major cost of the website is not hard disk space but its bandwidth. So transferring huge amount of data over the available bandwidth becomes very critical.

2. VULNERABILITY TYPES

There is much Vulnerability in ASP.NET website but this paper is focusing on most common vulnerabilities in terms of complexity, detection and recovers. Following are the main types

2.1 SQL Injection

SQL injection is a form of attack which applies on data driven applications. This attacks happens when the attacker inputs malicious text in the input field present in the web form and malicious input is not verified by the web application whether it is valid or not and attacker uses this malicious input to exploit the sensitive information present in the database. Attackers can enter SQL query as text in the input field, which will change the nature of the original query. Hence the malicious SQL query can enable attacker to access the sensitive information and delete the data present in the database.

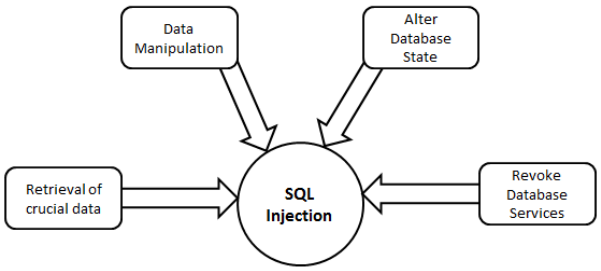

Figure 1: SQL Injection

For example:

An ASP.Net login page consists of two textboxes username and password, user will be authenticated if the record exists with username and password in the database otherwise login will not be successful. Below is the query to check the user is valid or not.

```
string strQry = "SELECT Count(*) FROM users WHERE username='" + txtUser.Text + "' AND password='" + txtPassword.Text + "'";

SqlCommand cmd = new SqlCommand(strQry, con);

int intRecs;

cmd.CommandType = CommandType.Text;

intRecs = (int)cmd.ExecuteScalar();

if (intRecs > 0)

{

//login succesfull

}
```

If the user inputs the username and password like ' OR 1=1 -- then above the query will become:

```
string strQry = "SELECT Count(*) FROM users WHERE username='" +  "OR 1=1 -- + "' AND password='" + "OR 1=1 -- + "'";
```

Which is always true because 1=1, Since 1=1 always evaluates to true, this query will always return more than 0 rows

Example 2:

```
string strSQL = "SELECT ProductId, ProductName, " + "Quantity, UnitPrice FROM Products WHERE ProductName LIKE '" + TextBox1.Text + "'";
```

Above query will gives the output by looking the ProductName in the database. If the user enters the following malicious input in the textbox:

```
' UNION SELECT 0, UserName, Password, 0 FROM Users –
```

Then the result is the number of entries present in the users table. Username and password will be shown.

2.2 Cross Site Scripting

Cross Site Scripting ('XSS') is very similar to the SQL injection. Cross Site Scripting also known as XSS. XSS vulnerabilities inject JavaScript that are embedded in the webpage which are fired on the client-side not in the server side. XSS allows an attacker to run his own client side scripts (especially JavaScript) into web pages viewed by other users. Cross Site Scripting occurs mainly in dynamic web pages that are mixing of browser data (HTML) with the code (<script> tag) which is embedded in the data. The script can be (JavaScript, VBScript, ActiveX, HTML, or Flash)[2][3]

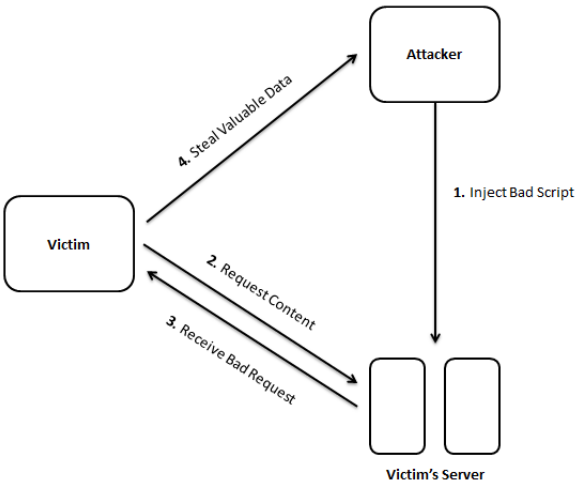

Figure 2: Cross Site Scripting

Types of Cross Site Scripting:

1) Non-Persistent

2) Persistent

Non-Persistent XSS Attack

Non persistent as the title suggests means that the injected script isn't permanent and just appears for the short time the user is viewing the page. It requires a user to visit the specially a link by the attacker. When the user visits the link,

the injected JavaScript code will executed in the user's browser.

Example:

An ASP.NET webpage contains a hyperlink which allows user to download the particular software that requested.

//download.aspx

string name = Request.QueryString["id"].ToString();

Label1.Text = name;

HLnk.NavigateUrl = "download.aspx?id=1&name=" +name;

Now the attacker will change an URL as follows and send it to the victim:

Download.aspx?name=guest<script>alert('attacked')</script>

When the user opens the above URL into the browser, he will see an alert box which says 'attacked'. Even though this example doesn't do any damage, but attacker can inject harmful script which can steal the sensitive information of the user which includes cookies, username, passwords etc.

Persistent XSS Attack

The stored (or persistent) Cross Site Scripting occurs when the data provided by the attacker is saved by the server, and then displayed permanently on "normal" pages returned to other users[9]. The damage caused by Persistent attack is more than the non-persistent attack.

Example:

- There are two types of users: "Admin" and "Normal" user.
- When "Admin" log-in, he can see the list of usernames. When "Normal" users log-in, they can only update their display name.

First the attacker log-in as a normal user, and he will enter the following in the textbox as his first name:

Hack

The above information entered by the attacker will be stored in the database (persistent).

Now, when the admin log in to the website, he will see list of registered users in which a link name "hack" along with other registered users. When admin click on the link named "hack", it will send the admin's cookie to attacker's website due to presence of session ID in the cookie, attacker can post a request to the website and can act like "admin" until session is not disposed.

3. PERFORMANCE AND BANDWITH ISSUES

ASP.NET contains built in data bound control and best for rapid development with powerful data access. When using ASP.NET data-bound control such as GridView is used to display data from a database, the easy method is to bind the data-bound control with a sqldatasource control and sqldatasource control automatically fetches the data from the database[5] after the Page.PreRender event[4] in the page life cycle , and displays it in the databound control.

<asp:SqlDataSource ID="datasrc1" runat="server" ConnectionString="<%$ ConnectionStrings:con %>" SelectCommand="storep1" SelectCommandType="storep1 "/>

<asp:GridView ID="GridView1" DataSourceID="datasrc1" …..>

<Columns>

<asp:BoundField DataField="name" HeaderText="name" SortExpression="name" …/>…

Following stored procedure is used to query data from the database

CREATE PROCEDURE [dbo].[storep1] AS

BEGIN

SELECT * FROM tbbook

END

GridView have a built-in mechanism for sorting and paging[5]. First, the sqldatasource control fetches all the data from the database server and then sorting will takes place and data will be displayed according to the GridView page size property (By default it is 10). For example, if table contains large number of records then all of the records are fetched and sorting takes place but only few records will be displayed.

This approach has two problems:

(i) Lots of data is transferred between the database server and the web server.

(ii) There is a large amount consumption of CPU and memory resources to sort large datasets on the web server.

4. SOLUTIONS TO THE ISSUES

4.1 SQL Injection

Use of parameterized query:

string strQry = "SELECT Count(*) FROM users WHERE username=@u AND password=@p";

SqlCommand cmd = new SqlCommand(strQry, con);

cmd.Parameters.Add("@u",SqlDbType.VarChar,50).Value=txtUser.Text;

cmd.Parameters.Add("@p",SqlDbType.VarChar,50).Value=txtPassword.Text;

4.2 Cross Site Scripting

By using below attributes in the page tag:

ValidateRequest="true" EnableEventValidation="true"

4.3 Performance And Bandwidth

Custom paging:

ALTER PROCEDURE pageing

(

@pagenumber int,

@pagesize int

)

```
AS
        DECLARE @srec int
        declare @sbid int
        declare @erec int
        declare @ebid int
        declare @rc int
        set @srec=@pagenumber*@pagesize-@pagesize+1
        declare c_book scroll cursor for
        select bookid from tbbook order by bookid
        open c_book
        fetch absolute @srec from c_book into @sbid
        select @rc=count(*) from tbbook where book>@sbid
        if @rc<@pagesize
        begin
        set @erec=@srec+@rc
        end
        else
        begin
        set @erec=@pagenumber*@pagesize
        end
        fetch absolute @erec from c_book into @ebid
        close c_book
        deallocate c_book
        select count(*) from tbbook
        select * from tbbook where bookid>=@sbid and bookid<=@ebid
```

5. LITERATURE REVIEW

Lwin Khin Shar and Hee Beng Kuan Tan had reported that the new major SQL injection threats in many web applications are due to the developer's lack of experience of work, development time or knowledge about security threats issues. Developers often misuse methods resulting in SQL injection vulnerabilities. Absence of data type check, misuse of delimiters in the query string, improper use of stored procedures and SQL parameters also leads to the create SQL injection threats in the ASP.Net web applications. They described defensive coding which includes parameterized and stored procedure queries, data type validation, remove of unwanted characters to the SQL injection threats in the ASP.Net web applications[6]

Lwin Khin Shar and Hee Beng Kuan Tan presented that XSS cross site scripting is one the most dangerous and serious attacks in web applications presented today. The absence of proper user data input validation and developer's less knowledge about the security issues result in XSS threats in the web application. XSS are present in various web applications written in different programming languages or platforms such as ASP.Net, Java and PHP. They implemented various techniques to detect the XSS flaws occurring at the client and server side[7]

6. CONCLUSION

Reducing In order to reduce the security vulnerabilities and performance issues in the ASP.NET websites/applications, conscious developers are required who are aware of these issues and of the various improved stability procedures/methods to remove these security vulnerabilities and performance issues. This paper discussed several issues related to the security, performance and bandwidth of the ASP.NET websites. Business corporations and organizations should make their ASP.NET websites secure against the prevalent SQL Injection and Cross Site Scripting attacks and implement various procedures/methods to make their websites perform well[8].

7. FUTURE WORK

This paper mainly focuses on the most common security vulnerabilities and performance issues present in the ASP.NET websites/applications. However, according to OSWAP there are several other security vulnerabilities[OWASP]and performance issues still pertinent in the ASP.NET websites which we will discuss in the second part of this paper.

8. REFERENCES

[1] OWASP. (2013, April) OWASP Top Ten Project. [Online]. https://www.owasp.org/index.php/Category?OWASP_Top_Ten_Project

[2] I. Poison. (2005, June) Cross site scripting: Common threats in web applications. [Online]. http://www.codeproject.com/Articles/10732/Cross-sitescripting- Common-threats-in-web-applica

[3] Dr. M. Ponnavaikko2 J. Shanmugam1, "Cross Site Scripting- Latest developments and solutions: A survey," *Pilani*, vol. 1, no. 2, September 2008.

[4] A. Freeman and M. Szpuszta M. MacDonald, "Pro ASP.NET 4 in C# 2010," ISBN-10: 9781430225294, APress, 2010.

[5] MSDN. (April, 2013) ASP.NET Data Access Overview. [Online]. http://msdn.microsoft.com/en-us/library/ms178359(v=vs.90).aspx

[6] Hee Beng Kuan Tan Lwin Khin Shar, "Defeating SQL Injection," *IEEE 2012*, 2012.

[7] Hee Beng Kuan Tan Lwin Khin Shar, "Defending against Cross-Site Scripting Attacks," *IEEE Computer Society*, pp. 55-62, 2012.

[8] B. Sullivan. Top 10 security vulnerabilities in.NET configuration files. [Online]. http://www.devx.com/dotnet/Article/32493/1954

[9] H. Singh and N. Kaur G. Singh, "Web application vulnerability assessment and preventing techniques," *International Journal of Enterprise Computing and Business. India*, vol. 2, no. 2230-8849, January 2012.