**SQL Injection:**

To test enter below string in search box. It will drop UserData table. Below is given query to re-create table

Live';drop table UserData;--

CREATE TABLE [dbo].[UserData](

[UserId] [int] NOT NULL,

[Name] [varchar](50) NULL,

[Address] [varchar](500) NULL,

[EmailId] [varchar](100) NULL

) ON [PRIMARY]

GO

SET ANSI\_PADDING OFF

GO

INSERT [dbo].[UserData] ([UserId], [Name], [Address], [EmailId]) VALUES (1, N'John', N'Banglore', N'john@gmail.com')

INSERT [dbo].[UserData] ([UserId], [Name], [Address], [EmailId]) VALUES (2, N'Martin', N'Pune', N'martin@gmail.com')

**HttpForbiddenHandler:**

Exclude unnecessary unused file extension. In our example we have excluded xml file. But allowed csv file. Below URL can be used to verify the same..

Allowed

<http://localhost:26641//Data/UserData.csv>

Not Allowed

<http://localhost:26641/Data/UserData.xml>

This configuration need to be specified for all file extension required to be blocked.

<system.web>

<httpHandlers>

<add verb="\*" path="\*.xml" type="System.Web.HttpForbiddenHandler" />

</httpHandlers>

</system.web>

<system.webServer>

<handlers>

<add verb="\*" path="\*.xml" type="System.Web.HttpForbiddenHandler" name="XML"/>

</handlers>

</system.webServer>

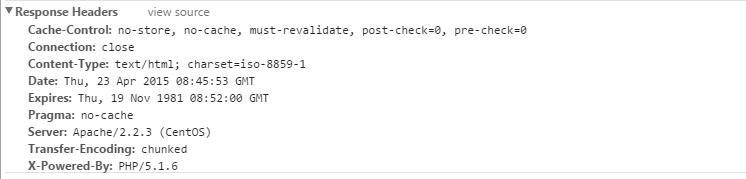
The httpHandlers element is used on sites running on IIS 5 – 6, or IIS 7.x in Classic mode (IIS 6 compatibility mode). The handlers element is used on sites running on IIS 7.x (Integrated mode).

**Cross Site Scripting:**

In write Review text area enter below text and save it. After that whenever page get loaded it will execute script in below text and open new window.

This product is good.Checkout some more<script>window.open('http://www.w3schools.com/', '\_blank');</script>

**Information Disclosure (HTTP Response Headers)**



Above data can be seen in network tab of browser which shows the server details which can help user to predict some of setting and get malicious information.

Below piece of code to be added to Global.aspx to remove some information from header.

protected void Application\_PreSendRequestHeaders(object sender, EventArgs e)

{

// Remove the "Server" HTTP Header from response

var app = sender as HttpApplication;

if (null != app && null != app.Context && null != app.Context.Response)

{

NameValueCollection headers = app.Context.Response.Headers;

if (null != headers)

{

headers.Remove("Server");

headers.Remove("X-AspNet-Version");

headers.Remove("X-AspNetMvc-Version");

headers.Remove("X-Powered-By");

}

}

}

Business Logic Bypassing (Client Side):

Relying only on client side validation can cause malicious users can easily bypass logic (Validation)

Steps To Reproduce:

1. Select an Item from Album.
2. Add item to Cart.
3. Click on **checkout.**
4. Enter Promo code as “FREE”.
5. Click on “**Submit**” button without filling any other details other than Promo code. You will see validation messages.
6. Disable JavaScript (How to disable and enable JavaScript [Click Here](http://enable-javascript.com/)).
7. Refresh the Checkout page.
8. Now click on “**Submit**” button without filling any other details other than Promo code. You can submit data without any validation messages.

Application errors with stack trace:

Steps To Reproduce:

1. Select an Item from Album.
2. When you go to Item Details page you will see the URL something Like <http://localhost:26641/Store/Details/6>
3. Change the Integer value appended at the end of the URL to some characters. (In above example from 6 to “some Characters”).
4. Press Enter key you will get the error message with Stack trace.

Always set a custom error page:

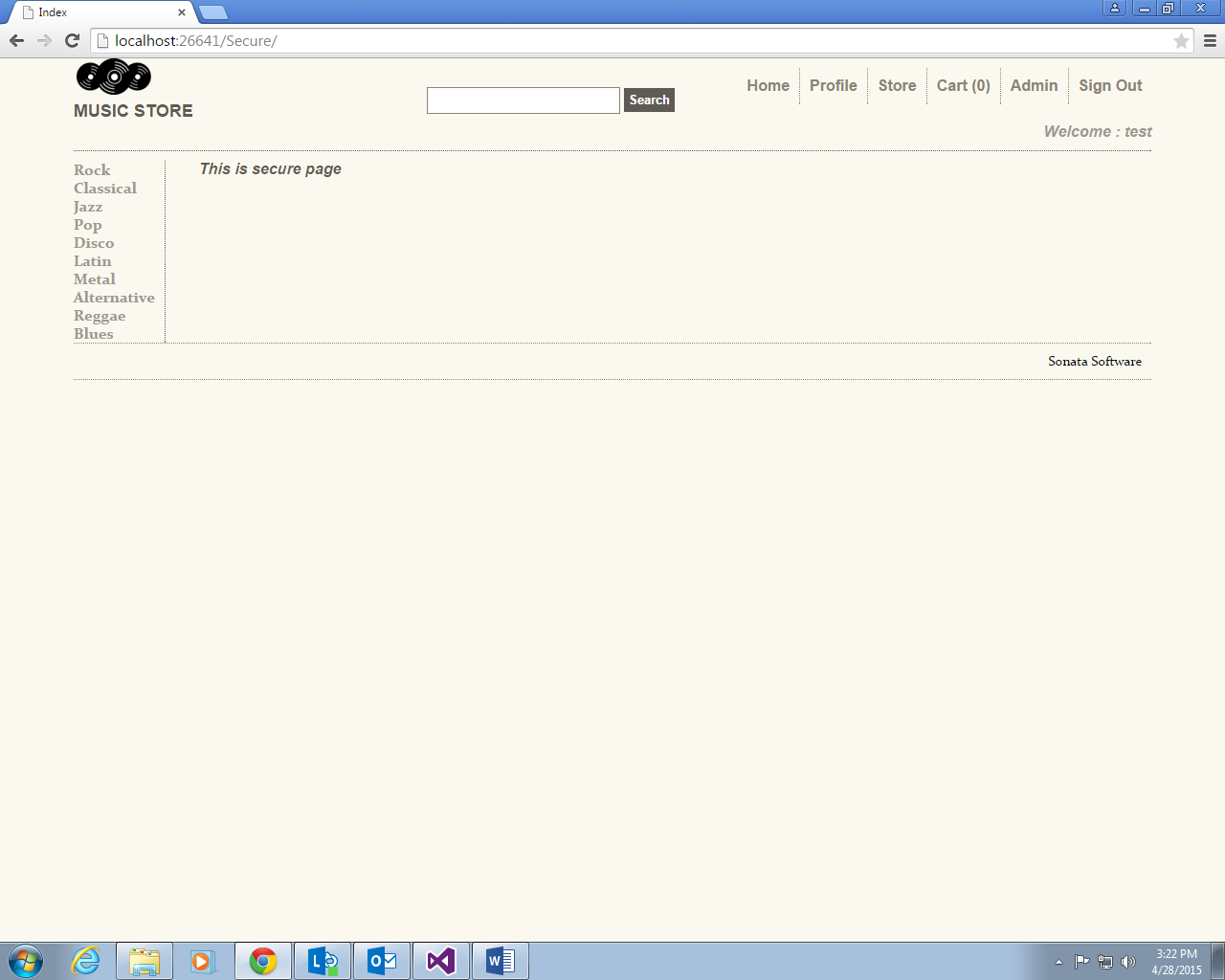
Steps To Reproduce:

1. Select an Item from Album.
2. When you go to Item Details page you will see the URL something Like <http://localhost:26641/Store/Details/6>
3. Change the Integer value appended at the end of the URL to 0 (Zero). (In above example from 6 to 0 (Zero))
4. Press Enter key you will get the source level error message.

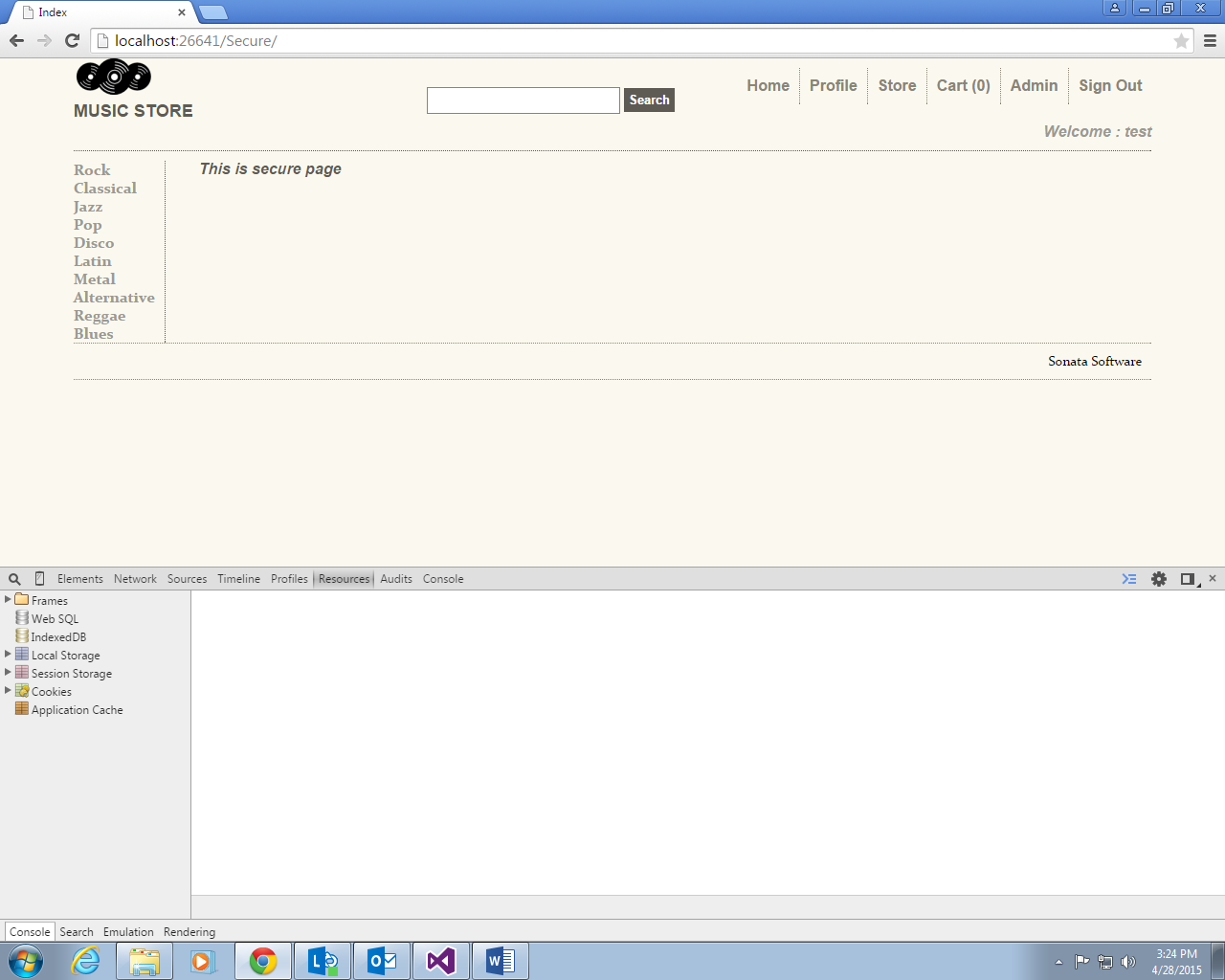
Session Fixation

Steps to Reproduce:

1. Click on **admin** Link.
2. You will get login screen, enter credentials and login.
3. Now click on **Profile** Link you will see the screen as below



1. Press F12 button on key board to open developer toll as below

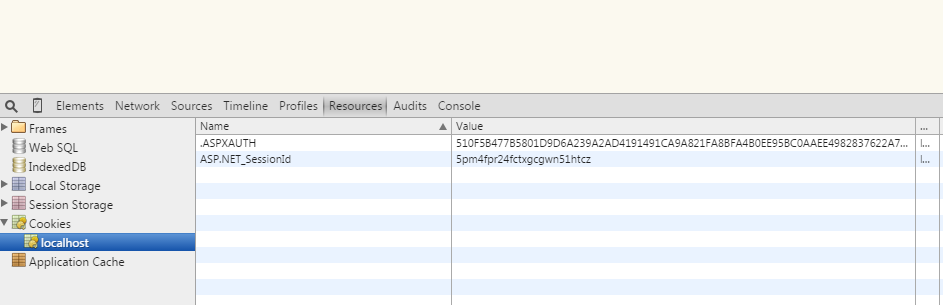


1. Expand Cookies tab as shown below

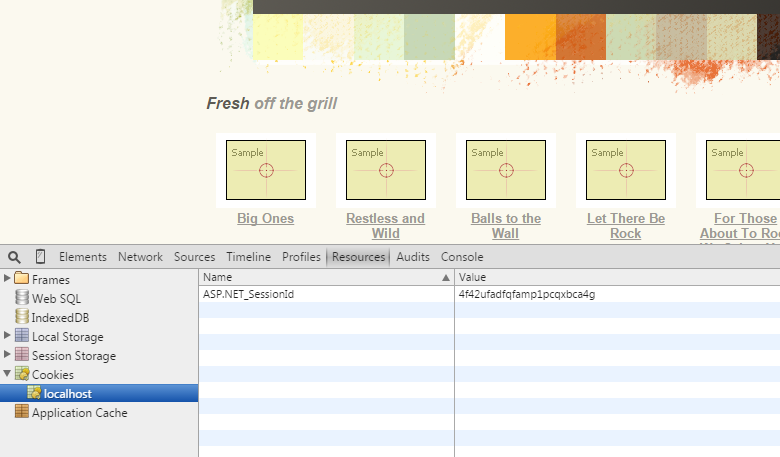
You will see two cookies

1) .ASPXAUTH (Authentication cookie)

2) ASP.NET\_SessionId(Default Session cookie)



1. Now Click on **Sign out** Menu.
2. Repeat the steps from 4 to 5.
3. You can still see the **ASP.NET\_SessionId** as shown below.



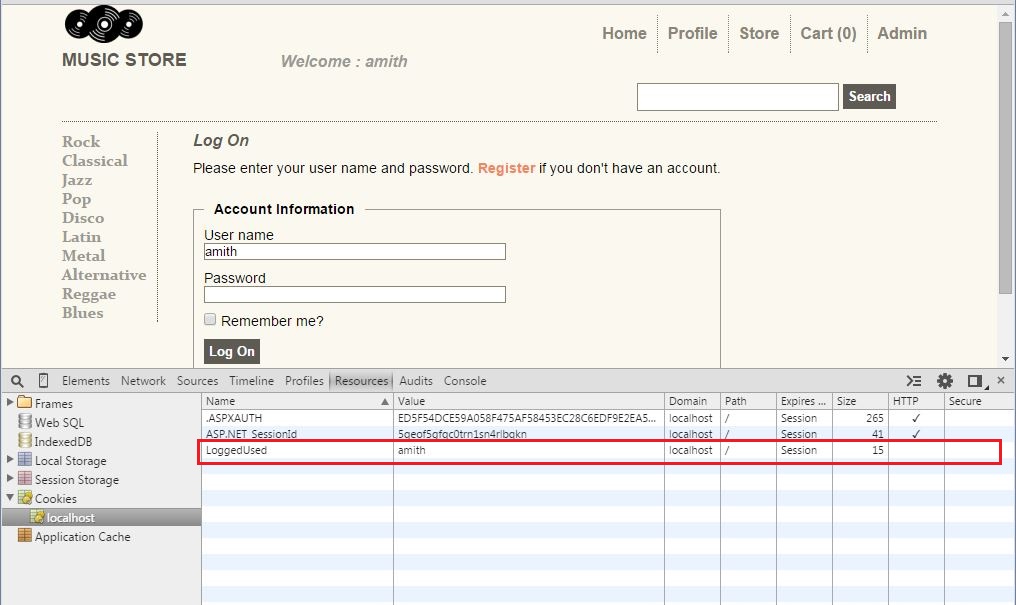
1. If you again click on **Profile** Link. Still can access this page because, in **profile** page only default session id is being checked.
2. To avoid this problem, need to generate an .ASPXAUTH (Authentication cookie (token)) using forms authentication.
3. For example if you go to Checkout page by selecting and adding an item to cart, you cannot access it because it is secured (Implemented with Forms authentication).

**Non-secured cookie**

A non-secured cookie is the one which does not include “HttpOnly” and “Secure” attributes.

Steps to reproduce -

1. Go to login page.
2. Enter the credentials and log in successfully.
3. You will see a new cookie named “LoggedUser” which does not set “HttpOnly” and/or “Secure” attributes to true as shown in the below screen shot.



1. To make the cookie secured then set the “HttpOnly” and “Secure” attributes as shown below.

HttpCookie loggedUser = new HttpCookie("LoggedUsed", model.UserName);

loggedUser.HttpOnly = true;

loggedUser.Secure = true;

Response.Cookies.Add(loggedUser);

**Authorization Bypassing:-**

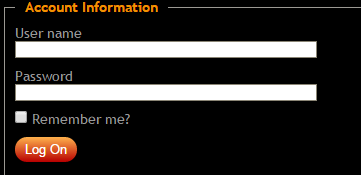
1:Run MVCMusic store application and click on Admin Page.

2:Without login as administrator , Try to manipulate Url

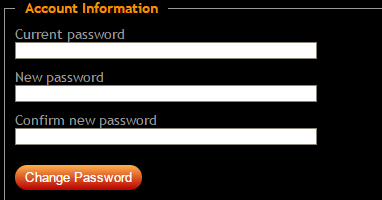
Change url from: http://localhost:26641/Account/LogOn?ReturnUrl=%2fStoreManager%2f to

http://localhost:26641/Account/ChangePassword .

3:Now you unable to redirect because only admin can redirect to change password Page.



4: After Making Authorization Bypass false if you try to access “<http://localhost:26641/Account/ChangePassword>” now you able to get change password screen



**Cross Site Request Forgery (CSRF):**

Cross Site Request forgery is a type of a hack where the hacker exploits the trust of a website on the user. In other words, the site trusts the user (because they have authenticated themselves) and accepts data that turns out to be malicious.

Attacker try to manipulate user details in CSRF.

1:By Adding a form request attacker change user details.

Fixing Problem:

1: First, we must add the unique token to the form to change the user's email when we display it.

2:Then add Html.AntiForgeryToken() it Generates a hidden form field (anti-forgery token) that is validated when the form is submitted.

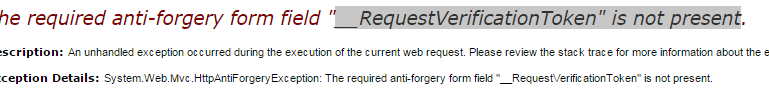
Generate scenario:

1:click on Admin tab.

2:click on AttackerchangeEmail link and change user email ,it will change in DB.

**Fixing the Problem**

1: Click on ChangeUserEmail link and try to change user mail. It will redirect to requestverificationtoken is not present error.



**Missing Anti-Scripting Mechanism:**

If you consider Sums up to 26 lower + 26 upper characters + 10 numbers + X (5) special chars ('!',, '$', '#', '@' and '-' ) which results in 67 possible characters.

If u use 6 digit password then 67 6 = 90,458,382,169 combinations which can be computed in about 43 s

If u use 7 digit password then 67 7 = 6,060,711,605,323 combinations which can be computed in about 48 minutes.

If u use 12 digit password then it will take 123,781 years to calculate all possible combinations.

1:Using BruteForceAttacks scripting Attacker able to generate password of a user.

**Generate Scenario:**

1:Click on Admin Tab.

2:Click on BruteForceAttacks link it able to hack and generate user password.

**Information Disclosure (HTTP Response Headers)**

1:Add MvcHandler.DisableMvcResponseHeader = true; in Application\_Start() global.asax

2:Add in global.asax

protected void Application\_PreSendRequestHeaders()

{

//Response.Headers.Remove("Server");

Response.Headers.Set("Server", "AntiHackerServer");

Response.Headers.Remove("X-AspNet-Version"); //alternative to above solution

Response.Headers.Remove("X-AspNetMvc-Version"); //alternative to above solution

}

3:Add in web.config

<!--<httpProtocol>

<customHeaders>

<remove name="X-Powered-By" />

</customHeaders>

</httpProtocol>-->

4:Add in web.config <httpRuntime enableVersionHeader="false" />

