Security Audit Points

1.web config file

<system.webServer>

<modules runAllManagedModulesForAllRequests="true" />

<httpProtocol>

<customHeaders>

<add name="X-Frame-Options" value="SAMEORIGIN" />

<remove name="X-AspNet-Version" />

<remove name="X-AspNetMvc-Version" />

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

<add name="Cache-Control" value="no-cache, no-store, must-revalidate" />

<!-- HTTP 1.1. -->

<add name="Pragma" value="no-cache" />

<!-- HTTP 1.0. -->

<add name="Expires" value="0" />

<add name="X-XSS-Protection" value="1; mode=block" />

<add name="X-Content-Type-Options" value="nosniff " />

<add name="Access-Control-Allow-Origin" value=" origin-list-or-null" />

<add name="Access-Control-Allow-Headers" value="Accept,Accept-Encoding,Accept-Language,Cache-Control,Content-Type,Host,Origin,Pragma,Referer,User-Agent" />

<add name="Access-Control-Allow-Methods" value="GET, PUT, POST, OPTIONS" />

<add name="Access-Control-Max-Age" value="1000" />

</customHeaders>

</httpProtocol>

<security>

<requestFiltering >

<verbs>

<add verb="TRACE" allowed="true" />

</verbs>

</requestFiltering>

</security>

</system.webServer>

protected void Application\_Start()

{

AreaRegistration.RegisterAllAreas();

GlobalConfiguration.Configure(WebApiConfig.Register);

FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);

RouteConfig.RegisterRoutes(RouteTable.Routes);

BundleConfig.RegisterBundles(BundleTable.Bundles);

MvcHandler.DisableMvcResponseHeader = true;

}

void Application\_BeginRequest(object sender, EventArgs e)

{

HttpContext.Current.Response.AddHeader("x-frame-options", "DENY");

this.Response.Headers["X-FRAME-OPTIONS"] = "DENY";

var application = sender as HttpApplication;

if (application != null && application.Context != null)

{

application.Context.Response.Headers.Remove("Server");

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

application.Context.Response.Headers.Remove("X-AspNet-Version");

application.Context.Response.Headers.Remove("X-AspNetMvc-Version");

application.Context.Response.Headers.Remove("X-Powered-By");

}

}

void Application\_Error(object sender, EventArgs e)

{

Exception exception = Server.GetLastError();

var httpException = exception as HttpException;

var val = httpException.GetHttpCode();

if (exception != null && (httpException.GetHttpCode().ToString()=="404" || httpException.GetHttpCode().ToString() == "403" || httpException.GetHttpCode().ToString() == "503"))

{

Response.Redirect("~/Error.html");

//log the error

}

}

3.Sql Injection Avoid from your site

Sol:

All Queries are change to Parameterize :

For EX:

OracleCommand cmd1 = new OracleCommand();

cmd1.CommandText= "select USER\_NAME from collector\_conf\_login WHERE USER\_NAME=:username";

cmd1.Parameters.Add(":username", OracleDbType.Varchar2, 50).Value = objmom.Username;

DataTable dt= RetrieveDataTable(cmd1);

public DataTable RetrieveDataTable(OracleCommand cmd)

{

try

{

con.Open();

if (con.State == ConnectionState.Open)

{

cmd.Connection = con;

dap = new OracleDataAdapter(cmd);

dt = new DataTable();

dap.Fill(dt);

if (dt.Rows.Count > 0)

{

return dt;

}

}

return null;

}

catch (Exception ex)

{

throw ex;

}

finally

{

if (con.State != ConnectionState.Closed)

{

con.Close();

}

if (cmd != null)

{

cmd.Dispose();

dap.Dispose();

}

}

}

4.Session Management and Session time out Maintenance

5.Every Login Capture Implemented and After Login Every Page Logout Button should be displayed.

6. Every Request Encrypted Data Pattern should be send