Part 1 - Software Security

Patrick Collins
1900609

Table of Contents

Abbreviations

SSE: Secure Software Engineering

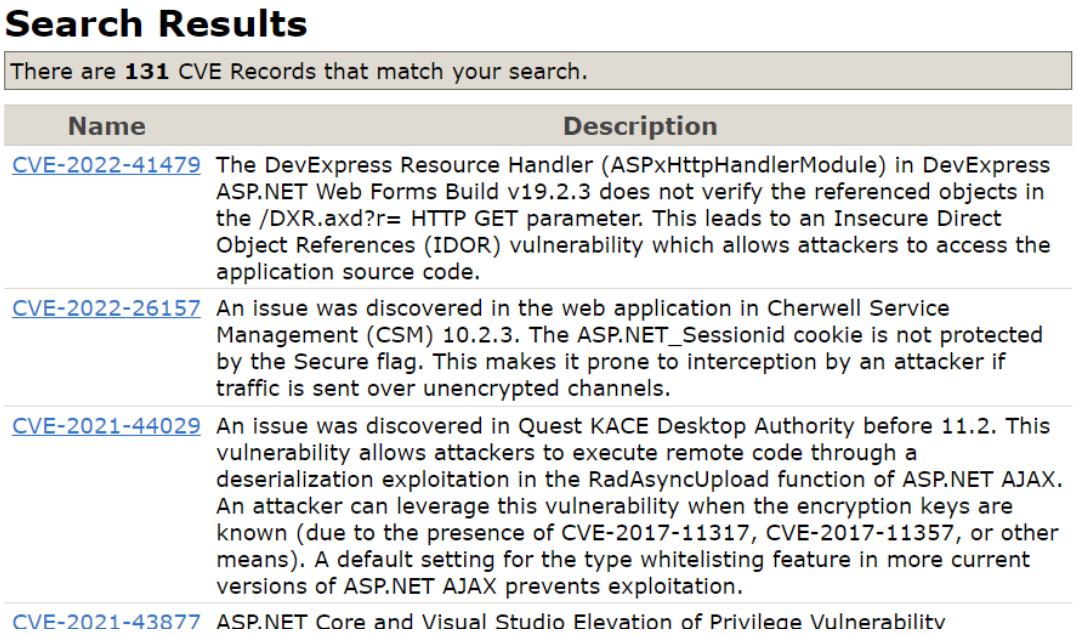
XSS: Cross-site Scripting

Context

ScottishGlen is a small company within the energy sector whose employees have been receiving messages from a hacktivist group, who threaten to target the company following a recent blog post by the CEO. It's unclear the nature of the attack they are planning so it is important for the development team to change their practice to incorporate a secure software engineering practice to prevent the attack on the company.

The web frontend, written with ASP.NET, is used by the HR staff to manage users and their details. This is deemed as a potential target by the IT manager and its security should be checked as the specific software system that ScottishGlen uses is possibly vulnerable to attacks from the hacktivist group.

A search on the CVE database for the term "ASP.NET" returns over 131 CVE's that this specific software system is vulnerable to as seen in Figure 1 (MITRE, 2023).



Search Results

There are **131** CVE Records that match your search.

Name	Description
CVE-2022-41479	The DevExpress Resource Handler (ASPxHttpHandlerModule) in DevExpress ASP.NET Web Forms Build v19.2.3 does not verify the referenced objects in the /DXR.axd?r= HTTP GET parameter. This leads to an Insecure Direct Object References (IDOR) vulnerability which allows attackers to access the application source code.
CVE-2022-26157	An issue was discovered in the web application in Cherwell Service Management (CSM) 10.2.3. The ASP.NET_Sessionid cookie is not protected by the Secure flag. This makes it prone to interception by an attacker if traffic is sent over unencrypted channels.
CVE-2021-44029	An issue was discovered in Quest KACE Desktop Authority before 11.2. This vulnerability allows attackers to execute remote code through a deserialization exploitation in the RadAsyncUpload function of ASP.NET AJAX. An attacker can leverage this vulnerability when the encryption keys are known (due to the presence of CVE-2017-11317, CVE-2017-11357, or other means). A default setting for the type whitelisting feature in more current versions of ASP.NET AJAX prevents exploitation.
CVE-2021-43877	ASP.NET Core and Visual Studio Elevation of Privilege Vulnerability

Figure 1: 131 CVE database results for search "ASP.NET".

One such set of attacks out of these results that the software system is vulnerable to is called user input validation attacks. As the HR staff use this website to log in and authenticate themselves the website will have a login method of some kind to authenticate the HR staff giving them the permission to access the system.

User Input Validation Attacks

User input validation attacks occur due to the input not being checked first before being used or executed. Not checking user input on the HR management system web frontend could lead to an attacker entering malicious input to gain access to the system. A successful attack could lead to obtaining authenticated credentials of the HR staff and the site be compromised.

There are many types of input validation attacks:

- Cross-Site Scripting (XSS)
- LDAP Injection
- SQL Injection

Here are some CVE examples of User input Validation attacks:

- CVE-2015-6099: XSS vulnerability in ASP.NET in Microsoft .NET Framework 4, 4.5, 4.5.1, 4.5.2, and 4.6 (MITRE, 2018).
- CVE-2019-18636: A cross-site scripting (XSS) vulnerability in JitBit AspNetForum <8.3.8 (MITRE, 2019)
- CVE-2020-24903: XSS vulnerability with Cute Editor in ASP.NET leading to the victim's cookie-based authentication credentials obtained (MITRE, 2021)
- CVE-2021-23335: LDAP injection with is-user-valid package to authenticate users (MITRE, 2021)

ASP.NET uses Request Validation but should not be used solely for XSS protection as advised by OWASP (Anderson, R, 2022, OWASP, 2023). The development team should perform further input validation checks alongside Request Validation in the frontend website they are developing by adopting a secure development practice ensuring the code is using input correctly.

Recommendation – Secure Code Review

The IT manager recommends the developers to adopt a secure code review practice checking Input Validation and Output Encoding (OWASP, 2021). This practice entails checking the source code of the web frontend for any possible vulnerabilities that could occur with malicious user input. Each new addition to the source code should be checked again if it has introduced vulnerabilities.

The developers should do this by using automated scanning tools that check for vulnerabilities in the web frontend source code. If there are any known problems with the code then it will found and have to be modified to prevent the security flaw. OWASP provide a list of source code analysis tools that developers may use for their chosen programming language (OWASP et al., 2023). As the threat identified is with ASP.NET there are multiple open source tools that aim to identify ASP.NET vulnerabilities in source code.

The tools listed are:

- Agnito
- Coverity Static Analysis
- CxSAST
- Fortify
- HCL AppScan Source/Cloud
- Security Code Scan
- Veracode

Furthermore, alongisde the automated scanning tools the developers should use OWASP's "DotNet Security Cheat Sheet" to identify how each class of security faults peviously identified should be avoided when programming the website (OWASP, 2021). The advised Secure Software Engineering (SSE) Practice can be seen in the flow diagram in Figure 2.

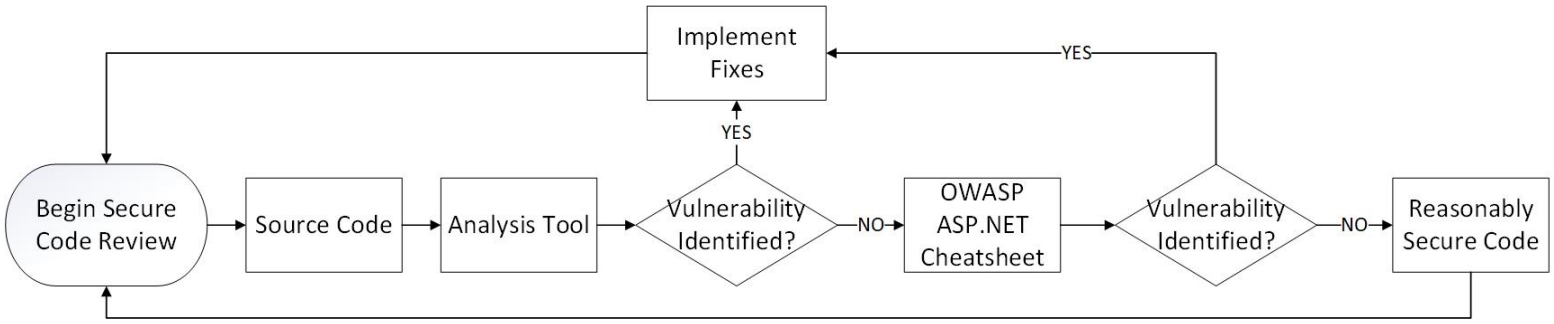


Figure 2: Flowchart diagram on the SSE practice advised here

When the code is identified as reasonably secure after the process finishes then the source code can be published and used in the HR system. This practice will efficiently and effectively stop vulnerabilities from appearing in any code the developers program. The class of security fault identified is one of the most common and known attack methods. Due to this any security issues in the source code that could enable these attacks on the HR system should be found.

Furthermore, the developers will be easily able to adopt this practice into their development lifecycle. There will be very little time wasted on training the team on carrying out this practice and using the tools. Moreover, secure code review is a trustworthy practice that is used by many large organisations, such as Facebook and Google, for years to prevent attacks against the products they are creating (Distefano et al., 2019, Sadowski et al., 2018).

Secure code review should be adopted by Scottish Glen over other secure software engineering practices as it will prevent the imminent attack on the company and secure systems as soon as possible. Choosing the practice of Continous Intergration Development to adopt using version control software will likely take too much time for the development team to learn and in the context of the security fault it will likely not find the website vulnerabilities effectively.

Normally a good suggestion for the development team would be to change the programming language used to create the website frontend to a more secure one. However, ASP.NET is among the secure website development programming languages and is safer to keep using for development of the HR system rather than switching to a different framework that could introduce more security issues for the company. Keeping with the programming language the developers are more accustomed to will also save time on teaching secure methods in that code as the team will just be adopting the secure advice for the langauge.

The IT manager recommends this approach over others as it is more suitable for the problem as OWASP's guidelines is a standard to use and follow in overall website security. As the problem identified is with a website it is justified to use a secure code review practice as advised by OWASP checking the code for any security issues before applying to the overall HR development system.

Implementation – SSE Practice

CVE-2019-18636

CVE description: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-18636

CVE-2019-18636 is a cross-site scripting (XSS) vulnerability in JitBit AspNetForum version 8.3.8 and lower that allows remote attackers to inject arbitrary web script or HTML via the gravatar URL parameter (MITRE, 2019).

Normal values like username are validated against XSS with a block list by replacing the text of input containing an arrow (such as left arrow) with the opposite arrow (right arrow). How the developers replaced malicious user input:

```
String username = tbUsername.Text.Replace("<","<").Replace(">",">");
```

However, the gravatar input is not checked for XSS using this method. This is probably a oversight from the developers to not implement this filter for the gravatar user input. In “editprofile.aspx” the Form field "DefaultAvatarInput" can be modified from Image file to any XSS injection crafted input. The example used in the CVE is:

```
" hidden/><script>alert('XSS')</script> <img hidden
```

The vulnerable code discussed in this vulnerability can be found in Appendix A (Jitbit, 2012).

How the development practice would prevent it.

The development practice advised would have prevented this XSS vulnerability in ASP.NET. This is due to the developers constantly checking their back-end code with static analysis tools and keeping to best practices whilst programming the web forum. From the CVE it is clear the developers were implementing simple filters adopting a block-list approach against the user input on the web forum, which goes against best practice from the OWASP DOTNET cheatsheet. *“It is a common mistake to use block list validation in order to try to detect possibly dangerous characters and patterns”* – (OWASP, 2021).

Had the developers implemented this practice into their development then the vulnerability would have been discovered, as following the ASP.NET advice from OWASP would have let the developers know that simply filtering user input is not enough to prevent an XSS attack.

OWASP XSS DOTNET ADVICE

As clearly mentioned in the DOTNET advice from OWASP cheatsheet, allow lists are always safe whereas block lists should be avoided (OWASP, 2021). Furthermore, the user input should have been encoded to prevent malicious user input. By encoding the user input a carefully crafted malicious input would become inexecutable (OWASP, 2021).

If the user input was properly encoded CVE-2019-18636 would have been prevented as the output would not be executed and instead be unreadable by the application.

Static Analysis Tools

The static analysis tool would have picked up on the fact that the user input was not being encoded when being printed to the web application and alerted the developers of this. Furthermore, it also may have identified the flawed approach of filtering user input which are notoriously easily bypassed by attackers sooner or later.

OWASP WEBGOAT.NET

OWASP have created a vulnerable ASP.NET web application called WebGoat.NET that contains common security flaws that programmers may use when programming their application. It also contains the secure practice alongside the vulnerable code to show developers the correct and secure way to program ASP.NET web applications as seen in figure 3 and Appendix B (Hoff, 2014). In this example application it contains multiple vulnerabilities of the class of security faults identified, XSS.

```
1 reference
void LoadCity (String city)
{
    DataSet ds = du.GetOffice(city);
    lblOutput.Text = "Here are the details for our " + city + " Office";
    dtlView.DataSource = ds.Tables[0];
    dtlView.DataBind();
}

0 references
void FixedLoadCity (String city)
{
    DataSet ds = du.GetOffice(city);
    lblOutput.Text = "Here are the details for our " + Server.HtmlEncode(city) + " Office";
    dtlView.DataSource = ds.Tables[0];
    dtlView.DataBind();
}
}
```

Figure 3: LoadCity causing XSS and secure alternative FixedLoadCity with html encoding.

To demonstrate and further prove that static analysis tools will pick up on these vulnerabilities one ASP.NET static analysis tool for Visual Studio called Security Code Scan was installed to test against the OWASP WeBGoat.NET application (JaroslavLobacevski, 2022).

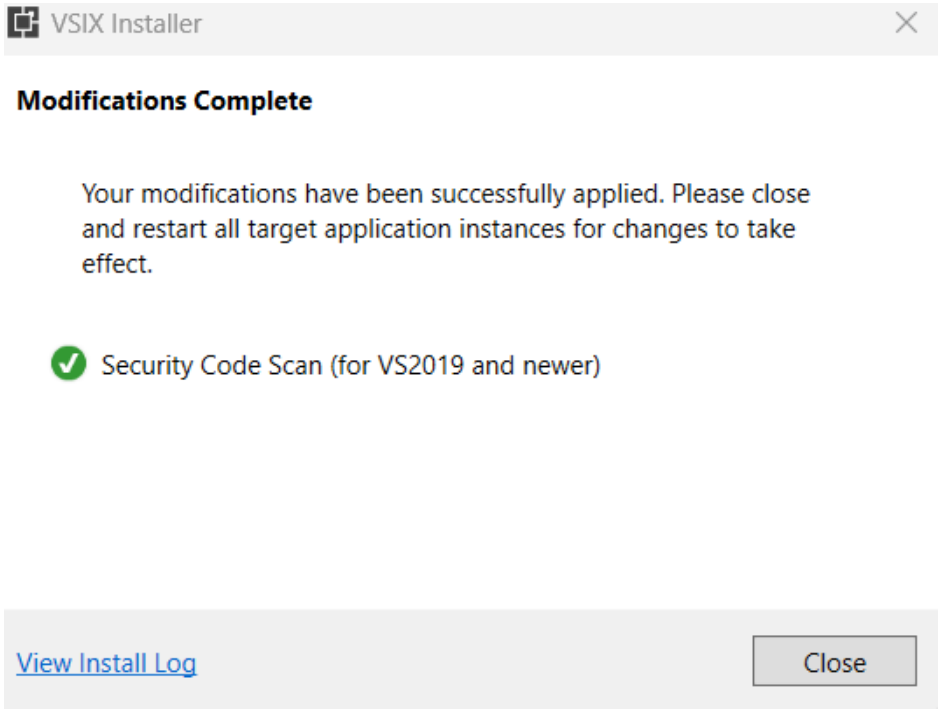


Figure 4: Installing Security Code Scan for Visual Studio 2019

After installing the WeBGoat.NET application it was opened in Visual Studio 2019. Warnings displayed at the bottom showing multiple vulnerabilities in ASP.NET such as insecure cookies, XXE and Path Traversal (Figure 5).

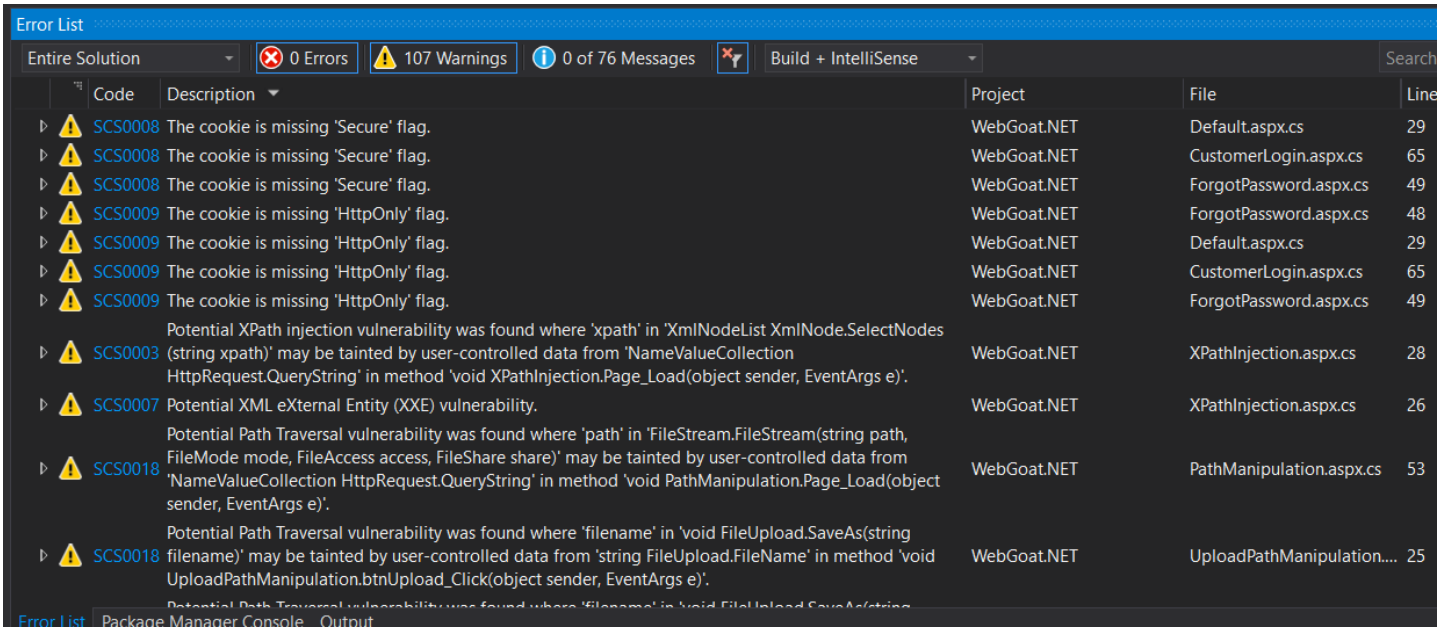


Figure 5: Warnings from static analysis tool showing multiple vulnerabilities in the code.

As the class of security identified was user input validation attacks the IT manager filtered the warnings for the vulnerability XSS (Figure 6). The static analysis tool successfully identified the XSS vulnerability in the function “LoadCity” created by OWASP for the WEBGOAT.NET vulnerable application. The full output for security flaws can be found in Appendix C.

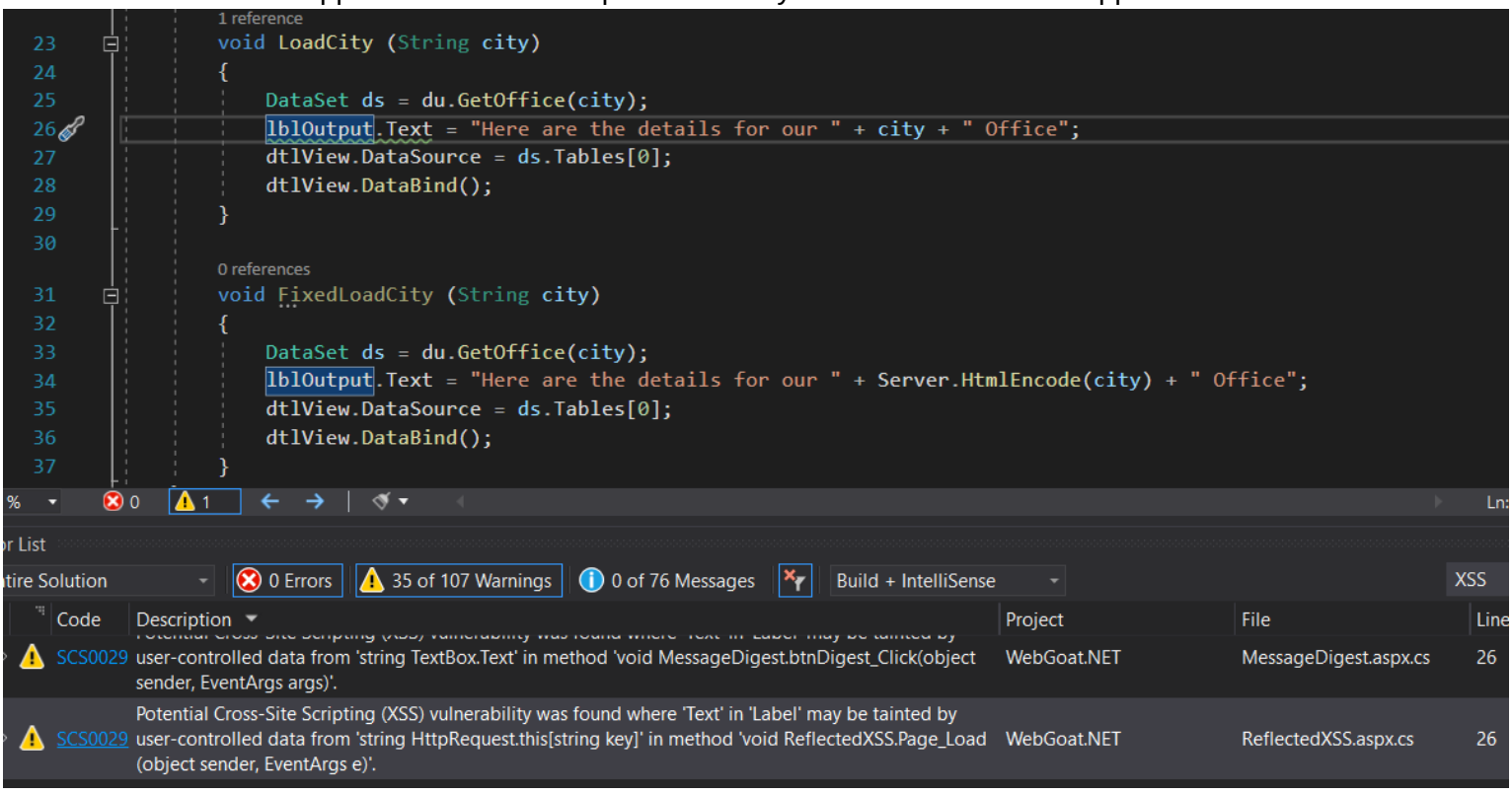


Figure 6: Filtered by XSS shows 35 instances of vulnerable code using Security Code Scan.

From the demonstration it is clear that the development team following the advised secure software engineering practice will identify and reduce vulnerabilities in the ASP.NET HR system. Following the best practices for ASP.NET will continue to keep the company as secure as possible with each new addition to the web frontend.

As proven, using the secure software engineering practice advised will prevent an attack on Scottish Glen and its systems and sensitive information will be protected from the threat of the hacktivist group.

Appendices

Appendix A – editprofile.aspx

```
<tr id="trGravatar" runat="server">
    <td class="gray" align="right"><a href="http://www.gravatar.com"
target="_blank">Gravatar...</a></td>
    <td class="gray">
        <div style="float:left;padding: 10px 10px 10px 10px;text-align:center;">
            <label for="rbGravatar"><img class="avatar" src="<%= GetGravatarUrl() %>"
width="<%= aspnetforum.Utils.Settings.AvatarResize %>" height="<%=
aspnetforum.Utils.Settings.AvatarResize %>" alt="avatar" /></label><br />
            <input type="radio" name="DefaultAvatarInput" id="rbGravatar" value="GRAVATAR" />
        </div>
    </td></tr>
<tr valign="top">
    <td class="gray" align="right">Predefined avatars:</td>
    <td style="width:60%">
        <asp:Repeater ID="rptDefaultAvatars" runat="server">
        <ItemTemplate>
            <div style="float:left;padding: 10px 10px 10px 10px;text-align:center">
            <label for="rbAvatar<%# Container.DataItem %>"><img class="avatar" src="images/<%#
Container.DataItem %>" width="<%# aspnetforum.Utils.Settings.AvatarResize %>" height="<%#
aspnetforum.Utils.Settings.AvatarResize %>" alt="avatar" /></label><br />
            <input type="radio" name="DefaultAvatarInput" id="rbAvatar<%# Container.DataItem
%>" value="<%# Container.DataItem %>" />
            </div>
        </ItemTemplate>
        </asp:Repeater>
        <asp:Label ID="lblAvatarsNote" CssClass="gray" runat="server">Administrators: you can
add more predefined avatars by adding images to the "images" folder, naming them
"AspNetForumAvatar1.gif", ""AspNetForumAvatar2.jpg" etc. The file extension shoud be one of
"jpg/gif/png".</asp:Label>
        <br /><br />
    </td>
</tr>
```

editprofile.aspx vulnerable code mentioned in CVE-2019-18636 (Jitbit, 2012)

Appendix B – ReflectedXSS.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using OWASP.WebGoat.NET.App_Code.DB;
using OWASP.WebGoat.NET.App_Code;

namespace OWASP.WebGoat.NET
{
   public partial class ReflectedXSS : System.Web.UI.Page
   {
      private IDbProvider du = Settings.CurrentDbProvider;

      protected void Page_Load(object sender, EventArgs e)
      {
         if (Request["city"] != null)
            LoadCity(Request["city"]);
      }

      void LoadCity (String city)
      {
         DataSet ds = du.GetOffice(city);
         lblOutput.Text = "Here are the details for our " + city + " Office";
         dtlView.DataSource = ds.Tables[0];
         dtlView.DataBind();
      }

      void FixedLoadCity (String city)
      {
         DataSet ds = du.GetOffice(city);
         lblOutput.Text = "Here are the details for our " + Server.HtmlEncode(city) + " Office";
         dtlView.DataSource = ds.Tables[0];
         dtlView.DataBind();
      }
   }
}
```

ReflectedXSS vulnerable code in OWASP WEBGOATNET (Hoff, 2014)

Appendix C – Static analysis tool “Security Code Scan” output

```
Severity Code Description Project File Line Suppression State
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'TableCell' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void EncryptVSEncode.btnGO_Click(object sender, EventArgs e)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\EncryptVSEncode.aspx.cs 60 Active
Warning SCS0006 Weak hashing function. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\EncryptVSEncode.aspx.cs 90 Active
Warning SCS0006 Weak hashing function. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\EncryptVSEncode.aspx.cs 93 Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Literal' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void ForgotPassword.ButtonCheckEmail_Click(object sender, EventArgs e)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\ForgotPassword.aspx.cs 37 Active
Warning SCS0008 The cookie is missing 'Secure' flag. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\ForgotPassword.aspx.cs 48 Active
Warning SCS0009 The cookie is missing 'HttpOnly' flag. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\ForgotPassword.aspx.cs 48 Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Literal' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void ForgotPassword.ButtonRecoverPassword_Click(object sender, EventArgs e)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\ForgotPassword.aspx.cs 66 Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Label' may be tainted by user-controlled data from 'NameValueCollection HttpRequest.Headers' in method 'void HeaderInjection.Page_Load(object sender, EventArgs e)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\HeaderInjection.aspx.cs 16 Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Label' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void MessageDigest.btnDigest_Click(object sender, EventArgs args)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\MessageDigest.aspx.cs 26 Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Label' may be tainted by user-controlled data from 'NameValueCollection
```

```
HttpRequest.QueryString' in method 'void PathManipulation.Page_Load(object sender,
EventArgs e)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay
University\Year 4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\Content\PathManipulation.aspx.cs  43  Active
Warning SCS0018 Potential Path Traversal vulnerability was found where 'path' in
'FileStream.FileStream(string path, FileMode mode, FileAccess access, FileShare share)' may
be tainted by user-controlled data from 'NameValueCollection HttpRequest.QueryString' in
method 'void PathManipulation.Page_Load(object sender, EventArgs e)'. WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\Content\PathManipulation.aspx.cs  53  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'value' in
'void HttpResponse.AddHeader(string name, string value)' may be tainted by user-controlled
data from 'NameValueCollection HttpRequest.QueryString' in method 'void
PathManipulation.Page_Load(object sender, EventArgs e)'.  WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\Content\PathManipulation.aspx.cs  79  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'buffer' in
'void HttpResponse.BinaryWrite(byte[] buffer)' may be tainted by user-controlled data from
'NameValueCollection HttpRequest.QueryString' in method 'void
PathManipulation.Page_Load(object sender, EventArgs e)'.  WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\Content\PathManipulation.aspx.cs  88  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'Label' may be tainted by user-controlled data from 'string HttpRequest.this[string key]' in method
'void ReflectedXSS.Page_Load(object sender, EventArgs e)'. WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\Content\ReflectedXSS.aspx.cs  26  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TextBox' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void
StoredXSS.btnSave_Click(object sender, EventArgs e)'.  WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\StoredXSS.aspx.cs  31  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TextBox' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void
StoredXSS.btnSave_Click(object sender, EventArgs e)'.  WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\StoredXSS.aspx.cs  31  Active
Warning SCS0018 Potential Path Traversal vulnerability was found where 'filename' in 'void
FileUpload.SaveAs(string filename)' may be tainted by user-controlled data from 'string
FileUpload.FileName' in method 'void UploadPathManipulation.btnUpload_Click(object sender,
EventArgs e)'.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay
University\Year 4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\Content\UploadPathManipulation.aspx.cs 25  Active
```

```
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Label' may be tainted by user-controlled data from 'string FileUpload.FileName' in method 'void UploadPathManipulation.btnUpload_Click(object sender, EventArgs e)'.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\UploadPathManipulation.aspx.cs 26  Active
Warning SCS0007 Potential XML eXternal Entity (XXE) vulnerability.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\XPathInjection.aspx.cs 26  Active
Warning SCS0003 Potential XPath injection vulnerability was found where 'xpath' in 'XmlNodeList XmlNode.SelectNodes(string xpath)' may be tainted by user-controlled data from 'NameValueCollection HttpRequest.QueryString' in method 'void XPathInjection.Page_Load(object sender, EventArgs e)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Content\XPathInjection.aspx.cs 28  Active
Warning SCS0008 The cookie is missing 'Secure' flag.    WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Default.aspx.cs    29  Active
Warning SCS0009 The cookie is missing 'HttpOnly' flag.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\Default.aspx.cs    29  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Label' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void ProxySetup.btnReverse_Click(object sender, EventArgs e)'.    WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\ProxySetup.aspx.cs 17  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 's' in 'void HttpResponse.Write(string s)' may be tainted by user-controlled data from 'string HttpRequest.this[string key]' in method 'void Autocomplete.ProcessRequest(HttpContext context)'.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\Autocomplete.ashx.cs  33  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in 'Literal' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void ChangePassword.ButtonChangePassword_Click(object sender, EventArgs e)'.    WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ChangePassword.aspx.cs    33  Active
Warning SCS0008 The cookie is missing 'Secure' flag.    WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\CustomerLogin.aspx.cs 65  Active
Warning SCS0009 The cookie is missing 'HttpOnly' flag.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems 1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\CustomerLogin.aspx.cs 65  Active
```

```
Warning SCS0027 Potential Open Redirect vulnerability was found where 'url' in 'void
HttpResponse.Redirect(string url)' may be tainted by user-controlled data from
'NameValueCollection HttpRequest.QueryString' in method 'void
CustomerLogin.ButtonLogOn_Click(object sender, EventArgs e)'. WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\CustomerLogin.aspx.cs 72  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'Literal' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void
ForgotPassword.ButtonCheckEmail_Click(object sender, EventArgs e)'.    WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ForgotPassword.aspx.cs    38  Active
Warning SCS0008 The cookie is missing 'Secure' flag.    WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ForgotPassword.aspx.cs    49  Active
Warning SCS0009 The cookie is missing 'HttpOnly' flag.  WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ForgotPassword.aspx.cs    49  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'Literal' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void
ForgotPassword.ButtonRecoverPassword_Click(object sender, EventArgs e)'.   WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ForgotPassword.aspx.cs    67  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'ImageUrl'
in 'Image' may be tainted by user-controlled data from 'HttpCookieCollection
HttpRequest.Cookies' in method 'void MainPage.Page_Load(object sender, EventArgs
e)'.    WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year
4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\MainPage.aspx.cs  32  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'ID' in
'Control' may be tainted by user-controlled data from 'HttpCookieCollection HttpRequest.Cookies'
in method 'void MainPage.Page_Load(object sender, EventArgs e)'.    WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\MainPage.aspx.cs  37  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TableCell' may be tainted by user-controlled data from 'HttpCookieCollection
HttpRequest.Cookies' in method 'void MainPage.Page_Load(object sender, EventArgs
e)'.    WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year
4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\MainPage.aspx.cs  41  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TableCell' may be tainted by user-controlled data from 'HttpCookieCollection
HttpRequest.Cookies' in method 'void MainPage.Page_Load(object sender, EventArgs
```

```
e)'.    WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year
4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\MainPage.aspx.cs  42  Active
Warning SCS0018 Potential Path Traversal vulnerability was found where 'filename' in 'void
FileUpload.SaveAs(string filename)' may be tainted by user-controlled data from 'string
FileUpload.FileName' in method 'void MainPage.btnUpload_Click(object sender, EventArgs
e)'.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year
4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\MainPage.aspx.cs  59  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where
'NavigateUrl' in 'HyperLink' may be tainted by user-controlled data from 'string
HttpRequest.RawUrl' in method 'void Orders.Page_Load(object sender, EventArgs
e)'.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year
4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\Orders.aspx.cs    77  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'Literal' may be tainted by user-controlled data from 'string HttpRequest.this[string key]' in method
'void Orders.Page_Load(object sender, EventArgs e)'. WebGoat.NET C:\Users\Paddy\OneDrive
- Abertay University\Abertay University\Year 4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\Orders.aspx.cs    83  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'Label' may be tainted by user-controlled data from 'string HttpRequest.this[string key]' in method
'void Orders.Page_Load(object sender, EventArgs e)'.    WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\Orders.aspx.cs    92  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where
'ContentType' in 'HttpResponse' may be tainted by user-controlled data from 'string
HttpRequest.this[string key]' in method 'void Orders.Page_Load(object sender, EventArgs e)'.
WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year
4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\Orders.aspx.cs    99  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'value' in
'void HttpResponse.AppendHeader(string name, string value)' may be tainted by user-controlled
data from 'string HttpRequest.this[string key]' in method 'void Orders.Page_Load(object sender,
EventArgs e)'.  WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay
University\Year 4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\Orders.aspx.cs    100 Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'filename'
in 'void HttpResponse.TransmitFile(string filename)' may be tainted by user-controlled data from
'string HttpRequest.this[string key]' in method 'void Orders.Page_Load(object sender, EventArgs
e)'. WebGoat.NET C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year
4\S2\Engineering Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\Orders.aspx.cs    101 Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TextBox' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void
ProductDetails.btnSave_Click(object sender, EventArgs e)'. WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
```

```
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ProductDetails.aspx.cs    42  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TextBox' may be tainted by user-controlled data from 'string HiddenField.Value' in method 'void
ProductDetails.btnSave_Click(object sender, EventArgs e)'.    WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ProductDetails.aspx.cs    42  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TextBox' may be tainted by user-controlled data from 'string TextBox.Text' in method 'void
ProductDetails.btnSave_Click(object sender, EventArgs e)'. WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
1\Assignment\VulnASP.NET\WebGoat\WebGoatCoins\ProductDetails.aspx.cs    42  Active
Warning SCS0029 Potential Cross-Site Scripting (XSS) vulnerability was found where 'Text' in
'TextBox' may be tainted by user-controlled data from 'HttpCookieCollection
HttpRequest.Cookies' in method 'void ProductDetails.LoadComments()'.   WebGoat.NET
C:\Users\Paddy\OneDrive - Abertay University\Abertay University\Year 4\S2\Engineering
Resilient Systems
```

Static analysis tool “Security Code Scan” output identifying multiple vulnerabilities (JaroslavLobacevski, 2022)

References

Anderson, R, 2022, Prevent cross-site scripting (XSS) in ASP.NET Core, Microsoft Learn. Available at: https://learn.microsoft.com/en-us/aspnet/core/security/cross-site-scripting?view=aspnetcore-7.0 [Accessed: March 11, 2023].

Anderson, R, 2022, Request validation - preventing script attacks, Microsoft Learn. Available at: https://learn.microsoft.com/en-us/aspnet/whitepapers/request-validation [Accessed: February 28, 2023].

Distefano, D. et al., 2019, Scaling static analyses at Facebook, ACM. Available at: https://cacm.acm.org/magazines/2019/8/238344-scaling-static-analyses-at-facebook/fulltext [Accessed: March 6, 2023].

Dotpaul, 2022, Ca3005: Review code for LDAP injection vulnerabilities (code analysis) - .NET, (code analysis) - .NET | Microsoft Learn. Available at: https://learn.microsoft.com/en-us/dotnet/fundamentals/code-analysis/quality-rules/ca3005 [Accessed: March 11, 2023].

Hoff, J, 2014, Jerryhoff/webgoat.net: Owasp WebGoat.NET, GitHub. Available at: https://github.com/jerryhoff/WebGoat.NET [Accessed: March 11, 2023].

JaroslavLobacevski, 2022, Security code scan (for VS2019 and newer) - visual studio marketplace, Marketplace. Available at: https://marketplace.visualstudio.com/items?itemName=JaroslavLobacevski.SecurityCodeScanVS2019 [Accessed: March 11, 2023].

Jitbit, 2012, Version history - .NET Forum Software (ASP.NET), jitbit. Available at: https://www.jitbit.com/asp-net-forum/versionhistory/ [Accessed: March 7, 2023]; Archived at Wayback Machine available at: https://web.archive.org/web/20120918183755/https://www.jitbit.com/asp-net-forum/versionhistory/ [Accessed capture: September 18, 2012]

MITRE, 2018, CVE Record | CVE , cve. Available at: https://www.cve.org/CVERecord?id=CVE-2015-6099 [Accessed: February 27, 2023].

MITRE, 2019, CVE-2019-18636, CVE. Available at: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-18636 [Accessed: March 11, 2023].

MITRE, 2021, CVE Record | CVE. Available at: https://www.cve.org/CVERecord?id=CVE-2020-24903 [Accessed: February 27, 2023].

MITRE, 2021, CVE Record | CVE. Available at: https://www.cve.org/CVERecord?id=CVE-2021-23335 [Accessed: February 27, 2023].

MITRE, 2023, Search results, CVE. Available at: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=ASP.NET [Accessed: February 27, 2023].

OWASP, 2021, DotNet security cheat sheet, DotNet Security - OWASP Cheat Sheet Series. Available at: https://cheatsheetseries.owasp.org/cheatsheets/DotNet_Security_Cheat_Sheet.html [Accessed: February 28, 2023].

OWASP, 2021, Input validation cheat sheet, Input Validation - OWASP Cheat Sheet Series. Available at: https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html [Accessed: March 6, 2023].

OWASP, 2021, LDAP injection prevention cheat sheet, LDAP Injection Prevention - OWASP Cheat Sheet Series. Available at: https://cheatsheetseries.owasp.org/cheatsheets/LDAP_Injection_Prevention_Cheat_Sheet.html#introduction [Accessed: March 11, 2023].

OWASP, 2023, ASP.NET request validation, ASP.NET Request Validation | OWASP Foundation. Available at: https://owasp.org/www-community/ASP-NET_Request_Validation [Accessed: February 27, 2023].

OWASP, et al, 2023, Source code analysis tools, Source Code Analysis Tools | OWASP Foundation. Available at: https://owasp.org/www-community/Source_Code_Analysis_Tools [Accessed: March 6, 2023].

Sadowski, C. et al., 2018, Lessons from building Static Analysis Tools at google, ACM. Available at: https://cacm.acm.org/magazines/2018/4/226371-lessons-from-building-static-analysis-tools-at-google/fulltext [Accessed: March 6, 2023].