Security Misconfiguration



Alexander Tushinsky

Cybersecurity & Software Development Consultant

@ltmodcs alextushinsky.com

Overview



- Common Security Misconfigurations
 - Use of HTTP instead of HTTPS
 - Debug / Trace information visible (Sensitive Data Exposure)
 - Cross-site Request Forgery exploits
 - XML External Entities (XXE)
- How do we address it?

Security Misconfiguration

- Happens when a component isn't configured for production
 - Default password
 - Sensitive file allowed to be viewed / downloaded
 - Incorrect setting
 - Debug/Trace mode enabled in a production environment
 - HTTPS (TLS) not required
 - Lack of AntiForgery token use



Demo



Security Misconfiguration

- Cross-site Request Forgery

Common Misconfigurations



OWASP Security Misconfigurations Controls

Proactive Controls

ASVS

C8: Protect Data Everywhere

V1.5: Input and Output Architecture

V8.1: General Data Protection

AntiForgery Token

- Prevents Cross-Site Request Forgery
- Validates that forms are being submitted from a known source
- Validation of this token can be implemented globally as a filter in .NET 6

Enable AntiForgery Token Validation

Configuration in Program.cs

```
builder.Services.AddControllersWithViews(options =>
{
    options.Filters.Add(new AutoValidateAntiforgeryTokenAttribute());
});

builder.Services.AddAntiforgery(options =>
{
    options.FormFieldName = "AntiForgeryToken";
    options.HeaderName = "X-CSRF-TOKEN";
    options.SuppressXFrameOptionsHeader = false;
});
```



```
public class AdminController : Controller
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Refund
               (decimal amount, string email)
[HttpPost]
[IgnoreAntiforgeryToken]
public IActionResult Refund
               (decimal amount, string email)
```

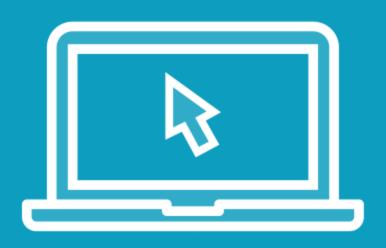
[AutoValidateAntiforgeryToken]

◆ Annotate at the class-level to protect the entire class

■ Annotation at the method level

■ You can also bypass the AntiForgery token check using the IgnoreAntiforgeryToken annotation

Demo



Remediation

- Add AntiForgery token checks

XML External Entities (XXE)

XML External Entity

 XML feature that allows web or local file content to be injected into an XML payload

```
<?xml version="1.0"?>
<catalog>
  <book id="bk101">
     <author>Gambardella, Matthew</author>
     <title>XML Developer's Guide</title>
     <genre>Computer
     <price>44.95</price>
     <publish_date>2000-10-01
     <description>An in-depth look at creating
                applications with XML.
     </description>
 </book>
  <book id="bk102">
     <author>0'Brien, Tim</author>
     <title>MSXML3: A Comprehensive Guide</title>
     <genre>Computer
     <price>36.95</price>
     <publish_date>2000-12-01
     <description>The Microsoft MSXML3 parser is
                 covered in detail, with attention to
                XML DOM interfaces, XSLT processing,
                SAX and more.
     </description>
  </book>
</catalog>
```

■ Typical XML file

```
<?xml version="1.0"?>
<!DOCTYPE external [</pre>
     <!ENTITY desc1 SYSTEM "file://c:/temp/desc1.txt">
     <!ENTITY desc2 SYSTEM "file://c:/temp/desc2.txt">
]>
<catalog>
 <book id="bk101">
     <author>Gambardella, Matthew</author>
     <title>XML Developer's Guide</title>
     <genre>Computer
     <price>44.95</price>
     <publish_date>2000-10-01
     <description>&desc1;</description>
 </book>
 <book id="bk102">
     <author>0'Brien, Tim</author>
     <title>MSXML3: A Comprehensive Guide</title>
     <genre>Computer</genre>
     <price>36.95</price>
     <publish_date>2000-12-01
     <description>&desc2;</description>
 </book>
</catalog>
```

◄ External entities being added to the file

■ &desc2; is replaced with the contents of c:\temp\desc1.txt

Code to Load XML



Vulnerability

- User allowed to upload an XML file
 - User can simply add the external entity reference to the file before upload
 - User can reference any file within the system
 - File will be loaded where XML variable is used

Remediation

- Principal of least privilege
- Use a custom XmlResolver and specify conditions around allowed resources

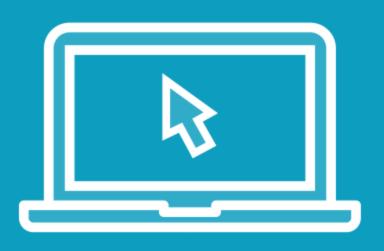
```
var xmlDocument = new XmlDocument
      XmlResolver = new CustomUrlResovler()
    xmlDocument.LoadXml(xml.ToString());
internal class CustomUrlResovler: XmlUrlResolver
  public override Uri ResolveUri(Uri baseUri,
                                                    string relativeUri)
    var uri = new Uri(relativeUri);
    return !uri.LocalPath.StartsWith("c:\\temp") ?
                 null : base.ResolveUri(baseUri, relativeUri);
```

■ Create your own UrlResolver class

■ CustomUrlResolver checks for specific path and compares it to the path coming from the XML

■ Returns null if it doesn't match your condition or loads the external document, if the path matches

Demo



Security Misconfiguration

- XML External Entities (XXE)
- Implement Custom Resolver

Summary



Security Misconfiguration

- Looked at common .NET security misconfigurations
- Reviewed potential issues
 - Cross-site Request Forgery
 - XML External Entities (XXE)
- Implemented fixes
 - AntiForgery token validation
 - Implemented custom UrlResolver



Up Next:

Vulnerable and Outdated Components

