DotNetNuke Code Access Security

Cathal Connolly

Version 1.0.0
Last Updated: June 20, 2006
Category: Security

DotNetNuke Code Access Security

Information in this document, including URL and other Internet Web site references, is subject to change without notice. The entire risk of the use or the results of the use of this document remains with the user.

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, places, or events is intended or should be inferred.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Perpetual Motion Interactive Systems, Inc. Perpetual Motion Interactive Systems may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Perpetual Motion, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

DotNetNuke® and the DotNetNuke logo are either registered trademarks or trademarks of Perpetual Motion Interactive Systems, Inc. in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Abstract

In order to clarify the intellectual property license granted with contributions of software from any person or entity (the "Contributor"), Perpetual Motion Interactive Systems Inc. must have a Contributor License Agreement on file that has been signed by the Contributor.

Contents

DotNetNuke Code Access Security

Introduction

In asp.net 1.1 Microsoft added the ability to configure asp.net web application security, by utilizing their Code Access Security (CAS) model. CAS allows you to grant permissions to assemblies, which give you granular control over what resources they can access such as file system locations, the registry and various namespaces.

To help organize these groups of policies together, Microsoft introduced the notion of trust levels. A trust level defines a security policy context, which all the asp.net files in a web application run under. Asp.net 1.1 shipped with 5 prepared trust levels, named full, high, medium, low and minimal, each of which progressively reduce levels of access to various capabilities. The default trust policy is Full, as this was required to maintain backwards compatibility with asp.net 1.0, which did not have a trust level implementation.

Microsoft recommend the medium trust policy for shared hosting servers and internet facing servers, as this provides a good level of security, whilst retaining sufficient functionality to deliver powerful web applications. With the release of DotNetNuke 3.0, a number of changes were made to DotNetNuke to allow it to run under a medium trust policy.

Hierarchical Configuration

ASP.NET uses a hierarchical model to apply configuration settings. The root configuration file is an XML file called machine.config, which applies asp.net configuration settings at the web server level.

It can be found at: systemroot\Microsoft.NET\Framework\versionNumber\CONFIG

where *versionNumber* is the asp.net version number (v1.1.4322 is the version for asp.net 1.1). The current default configuration is determined by the value indicated in the <trust level=" " originUrl=""/> node.

Individual web applications can also use web.config files to alter the permission set. When a web application is first accessed, asp.net calculates the effective policy for that resource by examining each configuration file found in the virtual directory path for the required resource, and caches the settings for reuse for future requests i.e. first the machine.config file, followed by any web.config file(s).

As mentioned, you can configure individual web applications and virtual directories with their own web.config files, which can add or modify permissions initially granted by the machine.config file. However, machine.config files can be marked with an optional attribute , allowOverride="false", which stops individual web.config files overriding it's server wide settings. This setting is commonly used by shared hosts to constrain individual sites ability to affect the server e.g. force them to run in the same trust level as all other sites.

Note: Trust levels can be configured on a per application basis, which means that you can run multiple asp.net applications inside of an IIS 6 app pool, and each of them can run on their own trust level.

What do I need to do to run in medium trust?

To run in a medium trust environment, you either need to set that as the designated policy in the <trust> node of the <SecurityPolicy> of the machine.config, or else add a <trust level="Medium"> instruction into your web.config file (Note: The default distribution of DotNetNuke has the trust level instruction commented out of the web.config file, so you need only uncomment the necessary lines.)

After changing trust level, it is usually necessary to recycle the web application if making the change at the application level, or else IIS, if applying it at the machine level.

Why would I choose to run in medium trust?

There are a number of reasons you may choose to run in medium trust

- Running with a minimized permission set, is a security best practice, as it allows you to minimize the potential attack surface a hacker can access, as well as limit the damage a successful exploit can yield.
- If developing modules, it allows you to ensure a larger potential audience, as many potential users/customers may need to deploy in partial trust environments.
- You may not have a choice. Microsoft are conducting a series of seminars on Shared Web Hosting where they are encouraging shared hosts to run in partial trust environments. As a shared host can apply the settings at the machine.config level, and mark that attribute as not allowing overrides, you will have to run as this trust level. Running as medium trust on your development servers may allow you to avoid difficulties when moving your DotNetNuke site to your hosting provider. https://www.ustechsregister.com/microsofthostingseminars/CitySelect.asp

Does medium trust limit any DotNetNuke functionality?

There were a number of areas where workarounds were developed to ensure that existing functionality worked, but perhaps in a different manner e.g. the add item to schedule dialog no longer uses Reflection to determine assemblies that contain scheduled tasks. Now, rather than picking from a dropdown list the user must now specify the full class name and assembly in a textbox.

Within the core-code itself, DotNetNuke does not make any calls to namespaces that do not run under medium trust. However, it is possible to use certain configurations that fail medium trust.

The two most common are:

- Medium trust does not allow access to Sql Server using the sa account and a blank password. Note: it is recommended that you do not use the sa account in web applications, and you should use a secure password for that account e.g. one that contains a mixture of alphanumeric characters and symbols.

- In Medium trust, web service permissions are limited. By default only calls to the current site are supported. See the next section for details on options for supporting additional site(s).

Enabling non-local Web service access

Adding a single additional domain

It's possible to add a single domain in your web.config file by utilising the originUrl= attribute (note: this supports wildcards so you can use url's such as "http://feeds.moreover.com/*" to access multiple feeds).

```
<trust level=" " originUrl="http://feeds.moreover.com/*"/>
```

Adding multiple additional domains

It's possible to add other allowed origins, but this requires access to the machine.config file, which is not always possible in shared hosting scenarios. If you have access to the file, then add the additional domains in the following format to the webpermission node.

```
<IPermission class="WebPermission" version="1">
 <ConnectAccess>
  <URI uri="$OriginHost$"/>
  <URI uri="http://www.somesite.com/.*"/>
  <URI uri="http://servername/.*"/>
  <URI uri="http://127.0.0.1/.*"/>
 </ConnectAccess>
</IPermission>
```

Allowing unlimited domains via policy changes

It is possible to alter the medium.config file to enabled all webservices. To do this open the medium.config file, and alter the webpermission block from:

```
 <IPermission
```

```
                    class="WebPermission"
                    version="1">
                  <ConnectAccess>
                    <URI uri="$OriginHost$"/>
                  </ConnectAccess>
                </IPermission>
```

to

```
<IPermission
                    class="WebPermission"
                    version="1"
                    Unrestricted="true"
                />
```

NOTE: if you wish to enable webservices in this way, it is recommended that you create a custom policy, rather than edit the existing policy file. To create a custom policy, copy the web_mediumtrust.config file, and rename the copied file e.g. web_dotnetnuke.config. Next, alter the file as per the settings above (or any other settings you wish to alter). Next, you will have to edit the machine.config file , and declare the new trust level e.g.

```
<trustLevel name="Dotnetnuke" policyFile="web_dotnetnuke.config"/>
```

Now this policy can be specified either via the machine.config or web.config.

Appendix A - Frequently asked questions

Q. What is a partial trust environment?

A. The default for asp.net 1.1 is Full trust, running under any other trust level is know as partial trust.

Q. Why have I not heard of trust levels and code access security before?

A. The capability for asp.net applications was only added to asp.net 1.1. Many hosts are either unaware of it, or choose to remain in Full trust, as it's the only trust level that allows access to OLEDB sources such as MS Access/MySQL. In asp.net 2.0, Microsoft are making this namespace, along with some others available to partial trust environments, which removes the need for many shared hosts to run in Full.

Q. Can I reduce the trust level below medium to be even more secure?

A. The next level down from Medium trust is Low. In Low trust the SQL client permissions are not present in the policy file, so DotNetNuke will not work. It is possible to alter Medium (or a custom policy based on Medium), to further constrain permissions, and still allow DotNetNuke to work, but care should be taken. Common scenarios would include stopping all webservice access, as well as stopping access to mail. Note: DotNetNuke is not tested with trust levels below medium.

Q. Are the core team implementing any changes to help with code access security exceptions in 3rd party modules?

A. Yes, we are testing code to allow for graceful handling of failures due to CAS violations.

Q. What type of 3rd party modules are likely to fail in Medium trust?

A. Amongst others, modules that try to use unmanaged code via pinvoke, access events logs or the registry or use OLE DB sources will fail. See Appendix B for a full list of trust level capabilities

Q. I've read that Assemblies in the Global Assembly Cache (GAC) can be used in partial trust. Is this true, and if so can I just add my 3rd party module to the GAC and then use it?

A. This is partially true. Whilst an assembly in the GAC is usually available for use in partial-trust environments, the assembly can carry a list of required permissions within it's manifest, that explicitly requires a trust level e.g. the OLEDB namespace has a Full trust link demand.

Q. Is there anywhere I can learn more about medium trust?

A. MSDN TV: Working with Medium Trust in ASP.NET

Appendix B -Default ASP.NET Policy Permissions & Trust Levels

Permission	State	High	Medium	Low	Minimal
DnsPermission	Unrestricted	Yes	Yes		
EnvironmentPermission	Unrestricted	Yes	TEMP; TMP; USERNAME; OS; COMPUTERNAME		
	Read				
	Write				
EventLogPermission					
FileIOPermission	Unrestricted	Yes			

Permission	State	High	Medium	Low	Minimal
	Read		$AppDir$	$AppDir$	
	Write		$AppDir$		
	Append		$AppDir$		
	PathDiscovery		$AppDir$	$AppDir$	
IsolatedStorageFilePermission	Unrestricted				
	AssemblyIsolationByUser		Yes	Yes	
	Unrestricted UserQuota		Yes	1mb (can vary)	
OleDbClientPermission	Unrestricted				
PrintingPermission	Unrestricted				
	DefaultPrinting	Yes	Yes		
ReflectionPermission	Unrestricted				
	ReflectionEmit	Yes			

Permission	State	High	Medium	Low	Minimal
RegistryPermission	Unrestricted	Yes			
SecurityPermission	Unrestricted				
	Assertion	Yes	Yes		
	Execution	Yes	Yes	Yes	Yes
	ControlThread	Yes	Yes		
	ControlPrinicipal	Yes	Yes		
	RemotingConfiguration	Yes	Yes		
SocketPermission	Unrestricted	Yes			
SqlClientPermission	Unrestricted	Yes	Yes		
WebPermission	Unrestricted	Yes	$OriginHost$		

Notes:

$AppDir$ refers to the initial folder the application points to, and any subfolders

$OriginHost$ refers to the currently executing website.

Additional Information

The DotNetNuke Portal Application Framework is constantly being revised and improved. To ensure that you have the most recent version of the software and this document, please visit the DotNetNuke website at:

http://www.dotnetnuke.com

The following additional websites provide helpful information about technologies and concepts related to DotNetNuke:

DotNetNuke Community Forums
http://www.dotnetnuke.com/tabid/795/Default.aspx

Microsoft® ASP.Net
http://www.asp.net

Open Source
http://www.opensource.org/

W3C Cascading Style Sheets, level 1
http://www.w3.org/TR/CSS1

Errors and Omissions

If you discover any errors or omissions in this document, please email marketing@dotnetnuke.com. Please provide the title of the document, the page number of the error and the corrected content along with any additional information that will help us in correcting the error.

Appendix A: Document History

Version	Last Update	Author(s)	Changes
1.0.0	Aug 17, 2005	Shaun Walker	• Applied new template