



# **DotNetNuke® Performance Configuration Best Practices Guide/Recommendations**

**Document Version: 1.02**

**Prepared By: Mitchel Sellers  
Reviewed By: Mitchel Sellers  
Date Created: 07/25/2009  
Date Last Modified: 07/02/2010**

## Table of Contents

<b>Revision History</b>	<b>4</b>
<b>Disclaimer</b>	<b>4</b>
<b>Copyright Notices</b>	<b>4</b>
<b>1. DotNetNuke® Performance Overview</b>	<b>5</b>
1.1 IIS/ASP.NET (“Hosting Environment”)	5
1.2 DotNetNuke Host Settings/Features	5
1.3 DotNetNuke Scheduler	5
1.4 Other Considerations/Next Steps	5
<b>2. Hosting Environment Configuration/Selection</b>	<b>5</b>
2.1 Shared Hosting	6
2.1.1 Selecting a Good Hosting Provider	6
2.1.2 ASP.NET Memory Limitations	6
2.1.3 ASP.NET Keep Alive	7
2.2 Virtual Private Server Hosting (VPS)	7
2.2.1 Amount of RAM and Type of RAM	7
2.2.2 SQL Server Configuration	8
2.2.3 ASP.NET Keep Alive or IIS Configuration	8
2.3 Cloud Computing / Cloud Hosting	8
2.3.1 Get Customer Testimonials	8
2.4 Dedicated Hosting	8
2.4.1 Amount of RAM	8
2.4.2 SQL Server Configuration	8
2.4.3 IIS Configuration	9
2.5 Recommended IIS Application Pool Configurations	9
<b>3. DotNetNuke Host Settings Configuration</b>	<b>9</b>
3.1 Authentication Providers	9
3.1.1 Changing Enabled Providers (DNN 5.x)	9
3.1.2 Changing Enabled Providers (DNN 4.7.0 - 4.9.5)	10
3.2 Performance Settings	11
3.2.1 Page State Persistence	11
3.2.2 Module Caching Method (4.x) / Module Cache Provider (5.x)	12
3.2.3 Performance Setting (4.x) / Cache Setting (5.x)	12
3.2.4 Compression and Whitespace Filter	12
3.2.5 Output Cache Provider (5.x - PE only by default)	12
3.3 Other Settings	13
3.3.1 Disable Users Online (4.x) / Enable Users Online (5.x)	13
3.3.2 Site Log History	13
3.3.3 Scheduler Mode	13
3.3.4 Enable Event Log Buffer?	14
3.3.5 Auto-Sync File System	14
<b>4. DotNetNuke Scheduler</b>	<b>14</b>
<b>5. DotNetNuke Output Cache Provider</b>	<b>14</b>

5.1	Obtaining an Output Cache Provider	14
<b>6.</b>	<b>Other Configurations and Next Steps</b>	<b>14</b>
6.1	Regular Clearing of Event Log	14
6.2	Regular Monitoring of Event Log Errors	15
6.3	Skin Modifications	15
6.3.1	Switch from SolPart/DNN Menu to Snapsis or Telerik	15
6.3.2	Create an XHTML Compliant CSS Based Skin	15
6.4	Third-Party Components	15
6.5	Seek Outside Assistance	15
<b>7.</b>	<b>Contact IowaComputerGurus</b>	<b>16</b>

## Revision History

Name	Date	Change and Reason For Changes	Version
Mitchel Sellers	7/25/2009	First document created	1.00
Mitchel Sellers	3/28/2010	Updated to include changes in DotNetNuke 5.x and DotNetNuke Professional Edition	1.01
Mitchel Sellers	7/02/2010	Updated with most recent DNN information and updated shared hosting information	1.02

## Disclaimer

Information contained in this document has been compiled and is general recommendation information provided by IowaComputerGurus Inc. This information is provided “as-is” and we do not offer any guarantee or warranty on the information provided within. The reader is responsible for performing due-diligence verification that the recommendations in this document fit their needs as well as the specific DotNetNuke environment.

In addition, it is strongly recommended that a backup of all site files and database be completed and VERIFIED before testing any of the configuration recommendations outlined in this document.

## Copyright Notices

The information contained within this document is protected under international copyright laws with a content owner of IowaComputerGurus Inc. This document may be re-distributed to anyone, however, it must remain intact and with this disclaimer visible.

The terms “DotNetNuke” and “DNN” are both registered trademarks of DotNetNuke corporation, <http://www.dotnetnukecorp.com>

## 1. DotNetNuke® Performance Overview

The performance of a DotNetNuke based website is typically one of the biggest concerns to any site administrator. Sadly there have been many sites and some history that have made statements such as “DotNetNuke is slow” and “getting performance out of DNN is hard” be the most common attitude when it comes to performance tuning and DotNetNuke.

However, that is not the case you simply need to have a basic understanding of a few key items and how those can impact your performance. The remaining points in this section will cover the highlights of impact areas, with the remainder of the document outlining how you can work within those elements to properly configure sites for performance.

### 1.1 IIS/ASP.NET (“Hosting Environment”)

IIS and the ASP.NET runtime are the two most commonly overlooked areas that can impact the performance of a DotNetNuke site. IIS and ASP.NET control the “box” that DotNetNuke runs inside, being an ASP.NET application it is controlled by ASP.NET and IIS, these two systems depending on configuration as well as hosting hardware can have varying impacts on the performance of a DotNetNuke® site. See section two of this document for a complete overview and recommendation.

### 1.2 DotNetNuke Host Settings/Features

Once you have validated the configuration of the physical/virtual hardware that is used to host the DotNetNuke site is time to start looking into the actual configuration of DotNetNuke Features and Functionality to ensure that it is optimized for your environment as well as for the best performance. Sadly, at this time the default configuration is typically not configured properly for most environments simply because it is really hard to get a baseline configuration that is optimized for each item.

Section three of this guide walks through the relevant sections within Host Settings and recommendations regarding configuration.

### 1.3 DotNetNuke Scheduler

The final configuration item that is typically used to help ensure proper performance is to review/modify the scheduled tasks that are installed/used by DotNetNuke. These scheduled tasks can have varying impacts on performance based on site structures.

Section four of this guide walks through the configuration/management of these sections.

### 1.4 Other Considerations/Next Steps

Finally after reviewing the DNN configuration there is a number of other items that can have an impact on the hosting environment. Section five of this document is dedicated to this.

## 2. Hosting Environment Configuration/Selection

It is best to start your performance configuration by looking at your hosting environment since this is the foundation of any DotNetNuke® installation, even minor changes here can have dramatic effects on the overall site performance.

Typically you will not see any limitations imposed on an application based on the IIS version, but most specifically on the type of hosting. Therefore the recommendations for configuration here are based on the type of hosting environment used. The following sections discuss the four primary types of hosting and things that are important to consider or modify when working in these environments.

*NOTE: The below recommendations are ONLY around performance considerations with DotNetNuke and do not necessary represent all considerations needed when selecting a hosting provider.*

## **2.1 Shared Hosting**

The shared hosting environment is one of the hardest to work with when it comes to application configuration since in 99.9% of all cases you are NOT granted any configuration abilities over the environment. However, we do have the following recommendations when working with a shared hosting environment.

### *2.1.1 Selecting a Good Hosting Provider*

When it comes to shared hosting environments not all hosting providers are the same. When looking for a DNN host to ensure that you have the best experience both in functionality and performance there are a few key features that you want to be sure that you have.

- ASP.NET Process With Root Permissions – DotNetNuke is a powerful framework and installing modules, skins, and other extensions often requires that DotNetNuke be able to make changes to the file system. Some hosting providers do not allow the application to have full permissions at the root, without this you will start to experience issues.
- Full-Trust Support – Running DotNetNuke in full trust is going to get the best flexibility possible.
- Installation to site root – one of the most common limitations are sites that do not allow installations to the root, installing to the root is important for both SEO and maintenance/performance reasons.
- Known Support – it is important to go with a hosting provider that is known for having good support/systems for running DotNetNuke.
- ASP.NET Memory Limitations - Many shared hosting providers are starting to impose hard set limitations on ASP.NET application memory usage, depending on the action taken when the limit is hit and the limit that is set your site can be adversely affected in these situations. Obviously you will want to get a hosting provider that has a very high limit, or that takes a less-lethal action when the limit has been exceeded. See section 2.1.2 for more detailed information.

Due to the above listed items, IowaComputerGurus has come across two hosting providers that we recommend for shared hosting. 3Essentials and PowerDNN, both provide great customer service and offerings, specific plans and recommendations might change at any time. (Referral links available at [www.iowacomputergurus.com](http://www.iowacomputergurus.com), referrals appreciated).

### *2.1.2 ASP.NET Memory Limitations*

As noted in the previous section hosting companies are starting to get more aggressive with the monitoring and limitation of ASP.NET memory and depending on the action taken your site can be impacted in a negative manner. The following sections are to help educate you on the specifics of what is a good limit, the possible actions taken, what the impacts might be to your site, and how you can monitor the memory usage.

#### *2.1.2.1 What is a Good Limit?*

Out of the box with a default configuration your average DotNetNuke installation will have a running footprint of anywhere between 50 and 100 megabytes of memory. In the case of IowaComputerGurus Inc. websites our largest site with the most traffic typically uses between 120 and 130 megabytes of memory. Overall a typical DotNetNuke site that is setup to run in a shared hosting environment should use a similar amount of ram. Keep in mind that the number and quality of installed third-party modules as well as the configuration of the site will impact the exact memory footprint of an application.

Therefore it is our recommendation to ensure that a hosting provider will allow for usage of 180-200 megabytes of memory for any application, this will help to ensure that your site has a little bit of room to grow before it has to get away from a shared hosting environment.

#### *2.1.2.2 What Actions Can They Take?*

The most common question that we receive regarding memory limitations is; What do they do to us if we exceed the limit? Well this is a question that is going to vary a lot from vendor to vendor, but there are two typical types of action. Application recycle or application lockout.

In the case of an application recycle the hosting provider will simply stop your application and allow it to restart. This frees the memory that was used by your site and allows it to start fresh. In this scenario your application will continue to serve content to all visitors, however, the users that are on the site when it is recycled will notice a delay when the next page loads due to the startup time for ASP.NET and pre-compilation. (Detailed discussion of this behavior in section 2.1.3). With this type of limitation if a site gets out of control is is a big deal, but the content is still there.

In the case of an application lockout the application will be configured to show a "Service Unavailable" message for a pre-defined amount of time if the memory limit is exceed. During this time your users will see an error message and NO content will be available. This is NOT a situation or action that you want a hosting provider to take. It is the most safe from their perspective in that it helps them manage resources but as a website owner it causes you lost traffic.

### 2.1.2.3 How to Monitor

The final point when looking at ASP.NET memory limitations is that many hosting providers do NOT provide you default access to monitor the memory usage of your application. The limitation discussed when talking about the ASP.NET worker process is NOT that of your disk space but the actual running system memory used by the ASP.NET process. Most of the standard control panels, including Plesk, do NOT provide a method to analyze the usage patterns of application memory use. You will want to check with your hosting provider to see if any reporting/monitoring tool is available.

### 2.1.3 ASP.NET Keep Alive

When working in a shared hosting environment you will not have control over the configuration of IIS, this can be a limiting factor from a performance perspective as IIS by default has a "Idle Timeout" that applies to ASP.NET applications. This timeout is by default set to 20 minutes, this means after 20 minutes of inactivity your site will unload from memory. The next page request will re-load the application; however, the user requesting the page can see a sometimes lengthy delay in application startup.

To get around this one of two items is recommended; talking to the hosting provider to have them extend the default timeout value or to employ a web based keepalive service. A keepalive service is something that simply pings your site every X minutes to simulate traffic keeping the application from unloading. IowaComputerGurus offers a service, MyWebKeepAlive (<http://www.mywebkeepalive.com>) with affordable yearly pricing. A number of additional services also exist from providers such as Pingdom or Host-Tracker. The key is that any service used MUST visit the site in intervals less than 20 minutes!

## 2.2 Virtual Private Server Hosting (VPS)

A VPS hosting environment is a configuration where a user is given their own Windows Server installation that is on a virtualized hardware environment. Typically with higher quality VPS hosting plans you will see that 4-8 VPS accounts will exist on one single physical server.

In general, IowaComputerGurus has not encountered any issues with VPS hosting as long as you properly validate the plans being purchased and the load being placed on the server. The following are recommendations when working with VPS accounts.

### 2.2.1 Amount of RAM and Type of RAM

The biggest consideration when working with a VPS is the amount of ram, both guaranteed and burst as well as how that RAM is actually given. For example two systems that provide 1GB of RAM but one provides that via physical RAM and the other via Virtual RAM you will notice massive performance improvements using a system that provides physical.

The amount of RAM is also important, IowaComputerGurus does NOT recommend using any configuration with less than 1GB of RAM for any DotNetNuke installation. If you are unsure of the specific RAM allotments and configurations contact the hosting provider to validate their configurations. IowaComputerGurus has had good luck

with HostMySite.com and their VPS+ accounts for this type of hosting.

### **2.2.2 SQL Server Configuration**

Many times when individuals opt for a VPS hosting plan it will start out with SQL Server being installed on the same system as the IIS server. Now, this isn't necessarily an issue, however it is very important that you properly configure SQL Server. By default SQL Server will NOT limit the total memory that it can use, and therefore over time it will keep taking more and more memory, eventually fighting IIS and other applications for available memory. Simply configuring a reasonable memory limitation on SQL Server will prevent a potential disastrous out of memory situation.

### **2.2.3 ASP.NET Keep Alive or IIS Configuration**

As with shared hosting the default behavior of ASP.NET with a 20 minute idle timeout is still something to be concerned about. However, once you have reached the VPS level of hosting you are typically granted remote desktop access to the server which will allow you to simply modify the timeout value. In certain circumstances such as providers that use specific control panels such as PLESK this might not be an option. If you are not able to configure this value, it will be necessary to pursue a keepalive service out outlined in 2.1.2.

See section 2.5 for full Application Pool configuration recommendations

## **2.3 Cloud Computing / Cloud Hosting**

As we continue moving up the chain in hosting environments we get to the cloud computing environment. At this time the characteristics of this environment are very hard to keep consistent as each cloud provider offers different styles of service. For example Mosso provides an environment similar to a Shared Hosting environment simply with more resources available. Places such as GoGrid on the other hand are more of dynamically scalable server instances with full remote desktop access.

Therefore overall the general recommendation for this area is to see which of the traditional hosting systems are similar to that of the cloud provider and use that as a base for evaluating the service offering. Otherwise there is only one other key performance indicator.

### **2.3.1 Get Customer Testimonials**

Given the dynamic nature of this hosting sector the best way to truly evaluate the validity of the environment is to contact and work with individuals that are either using the service or have moved away from it. These customer testimonials can tell a lot about the potential performance implications regarding any specific hosting provider.

## **2.4 Dedicated Hosting**

A dedicated hosting environment although the most costly of the major types provides a DotNetNuke site administrator the best control and ability to tune the system for optimal system performance. However, the hard part here is in selecting the proper hardware and configuration. Therefore many of the considerations for performance here are similar to that of the VPS environment.

### **2.4.1 Amount of RAM**

Working in a dedicated hosting environment it is possible that you will have IIS and SQL Server on the same machine, in these cases a minimum of 4Gb of RAM would be recommended. If working with a dual server configuration 2Gb of RAM will be sufficient for your starting environment. However, when building a dedicated server, more RAM is better!!

### **2.4.2 SQL Server Configuration**

Just like with VPS hosting, if you have both SQL Server and IIS configured on the same machine, it is very important to set memory limitations for SQL Server to ensure that it does not overrun IIS in a fight for memory. Typical baseline recommendations would to not allow SQL Server to use more than 50% of the total available physical RAM in these types of situations.

### 2.4.3 IIS Configuration

As mentioned in earlier sections the default 20 minute idle timeout is a configuration setting that is recommended to be changed to prevent un-necessary application recycles.

See section 2.5 for full Application Pool configuration recommendations

## 2.5 Recommended IIS Application Pool Configurations

As previous sections have mentioned, at the server level one of the most important items to configure is the idle timeout value for the application pools to prevent IIS from unloading the application. In our experience we have found that two specific configuration changes from the default result in the best overall server configuration.

The first is to modify the idle timeout from 20 minutes to 120-240 minutes. This will still allow an application to shut down if it is incredibly low traffic. Secondly we also recommend to do a regular recycle of the application pool at an off peak hour. For our sites we recycle the application pool at 2AM. Doing this helps to ensure that memory use and overall server health is maintained.

In addition to the idle timeout it is important to remember that a DNN application should have its own application pool, and in the case of multi-portal installations DO NOT setup the same home directory in two different IIS Websites or application pools. Doing this will result in all scheduled jobs running twice and sub optimal server performance.

## 3. DotNetNuke Host Settings Configuration

DotNetNuke is a very flexible platform and as such has a number of configuration options available. These configuration options grant a great flexibility but are often items that are considered “too complex” or are simply overlooked by site administrators. The following sub sections discuss items existing under the “Host” -> “Host Settings” menu option and how they can impact/modify performance of a DotNetNuke installation.

This document only covers the items relevant to performance. For a full overview of available host settings please see this article - <http://www.mitchellsellers.com/blogs/articletype/articleview/articleid/189/dotnetnuke-host-settings-explained.aspx>

### 3.1 Authentication Providers

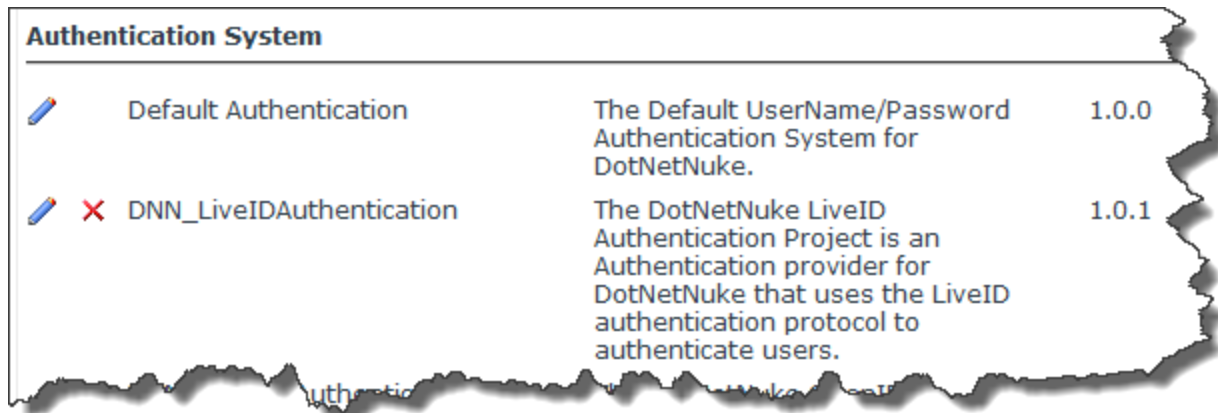
Starting with DotNetNuke 4.7.0 an authentication provider module was introduced that allow website administrators to add functionality to authenticate users against different sources of record. Most DotNetNuke sites that we work with have only a single authentication provider in use, and that is the standard DotNetNuke configuration. However, behind the scenes you can have a performance hit when other providers enabled, even if they are not being used. This performance issue is typically limited to the login page itself where DotNetNuke polls all enabled providers to see if they must be rendered, but the performance change can be major.

It is IowaComputerGurus' recommendation that you disable all providers that are not being used. Disabling the other providers on our website reduced login page load time from 3.5 seconds to 0.75 seconds with no other system configuration changes.

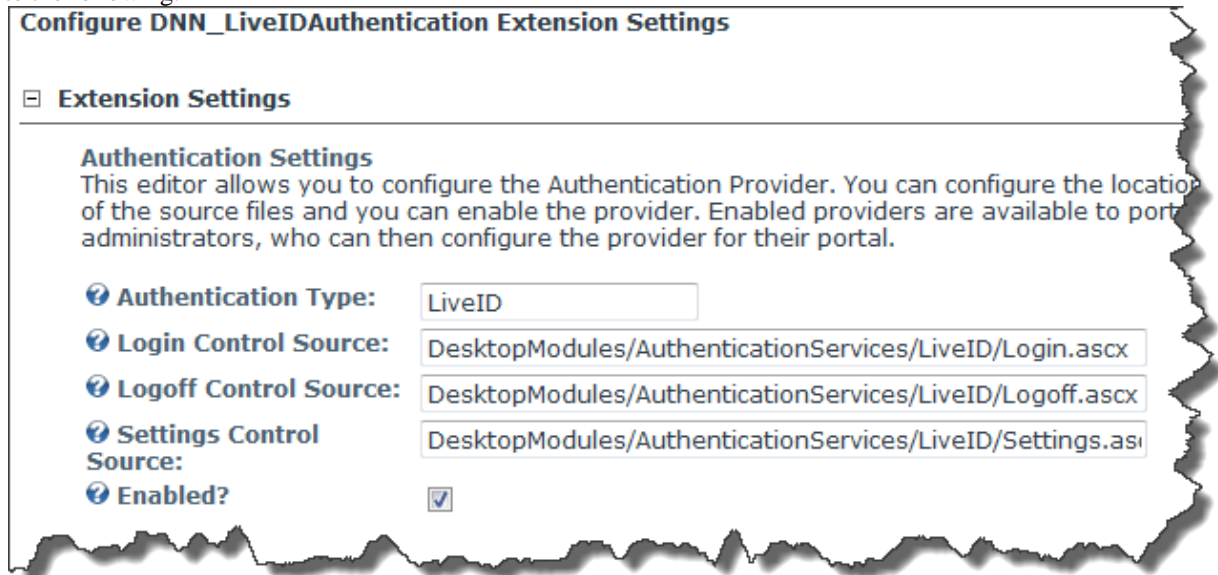
It is important to note that by default DotNetNuke installs three providers, so this configuration change is needed on ALL DotNetNuke portals.

#### 3.1.1 Changing Enabled Providers (DNN 5.x)

In DotNetNuke 5.x the administration process to enable/disable authentication providers became more complicated requiring an edit to each providers definition. To enable/disable a provide start by going to "Host" -> "Extensions", on this page you will see an "Authentication System" section similar to the below image.



You might have more or less providers listed here, the standard DotNetNuke provider is labeled "Default Authentication". To disable a provider you must click "Edit" on each entry and you will be taken to a page similar to the following.



From here you can un-check the "Enabled" box. NOTE: This disables the provider at the host level and makes it not available to ANY child portal.

### 3.1.2 Changing Enabled Providers (DNN 4.7.0 - 4.9.5)

If you are working in a DotNetNuke 4.x environment this process is much similar, simply navigate to "Host" -> "Host Settings" and under "Advanced Settings" -> "Authentication Settings" you will see the following display. Simply un-check any providers that should be disabled. NOTE: this setting is saved as you make the changes, update is not necessary!

Type	Enabled	Settings Control Src	Login
DNN	<input checked="" type="checkbox"/>	DesktopModules/AuthenticationServices/DesktopModules/DNN/Settings.ascx	DesktopModules/AuthenticationServices/DesktopModules/DNN/Login.ascx
✗ LiveID	<input type="checkbox"/>	DesktopModules/AuthenticationServices/DesktopModules/LiveID/Settings.ascx	DesktopModules/AuthenticationServices/DesktopModules/LiveID/Login.ascx
✗ OpenID	<input type="checkbox"/>	DesktopModules/AuthenticationServices/DesktopModules/OpenID/Settings.ascx	DesktopModules/AuthenticationServices/DesktopModules/OpenID/Login.ascx

### 3.2 Performance Settings

Default DotNetNuke performance settings are ok for testing environments but most other environment require items to change in at least one of these areas. The following sub sections will outline recommendations for configuration of these items. NOTE that the performance settings have changed a bit between DotNetNuke 4.x and 5.x, below are samples of configurations for both a 4.x and 5.x site. In the following section any item that is specific to 5.x will note in the header, one additional option is shown by default for DNN PE only and will also be discussed.

#### 4.x

**Performance Settings**

**Page State Persistence:**  Page  Memory

**Module Caching Method:**  Memory  Disk

**Performance Setting:** Moderate Caching [Clear Cache](#)

**Authenticated Cacheability:** ServerAndNoCache

**Compression Setting:** No Compression

**Use Whitespace Filter:**

#### 5.x

**Performance Settings**

**Page State Persistence:** **\*\* Warning: Memory page state persistence is not recommended.**  Page  Memory

**Module Cache Provider:** File [Clear Cache](#)

**Cache Setting:** Moderate

**Authenticated Cacheability:** ServerAndNoCache

**Compression Setting:** No Compression

**Use Whitespace Filter:**

#### 3.2.1 Page State Persistence

Although changing this option to “Memory” could reduce the overall size of the request sent to the user, our experience has proven that this option most commonly causes other issues and we recommend that you NEVER set it to anything other than “Page”.

### 3.2.2 *Module Caching Method (4.x) / Module Cache Provider (5.x)*

The module caching method configures how DotNetNuke stores its cache of module objects. The proper configuration of this varies depending on the specific environments that you are hosting in. With Shared Hosting and Dedicated hosting typically you will see better results using Memory caching. This is because you have memory available, and you want to reduce the amount of disk IO. When looking at some cloud computing environments disk based caching makes sense as website content is stored on a SAN or similar device with very good write speeds, and memory availability is either limited or sporadic.

Overall, starting with a baseline you can determine which option suits your needs the most after seeing the impacts.

### 3.2.3 *Performance Setting (4.x) / Cache Setting (5.x)*

The performance setting is used to control how much other underlying data is cached by DotNetNuke. IowaComputerGurus recommendation is to always leave this set to *Heavy*. Moderate caching simply does not offer enough caching.

### 3.2.4 *Compression and Whitespace Filter*

The final configuration item in this section for performance is to turn on gZip compression. Enabling compression on the DotNetNuke side of things is an easy way to reduce the size of the site payload and get a nice boost in performance. It is possible to leave DotNetNuke compression disabled and use IIS, however, we have experienced issues with some third party modules and IIS level compression. Therefore it is recommended to use the DNN functionality unless there is a specific need to compress at the IIS level.

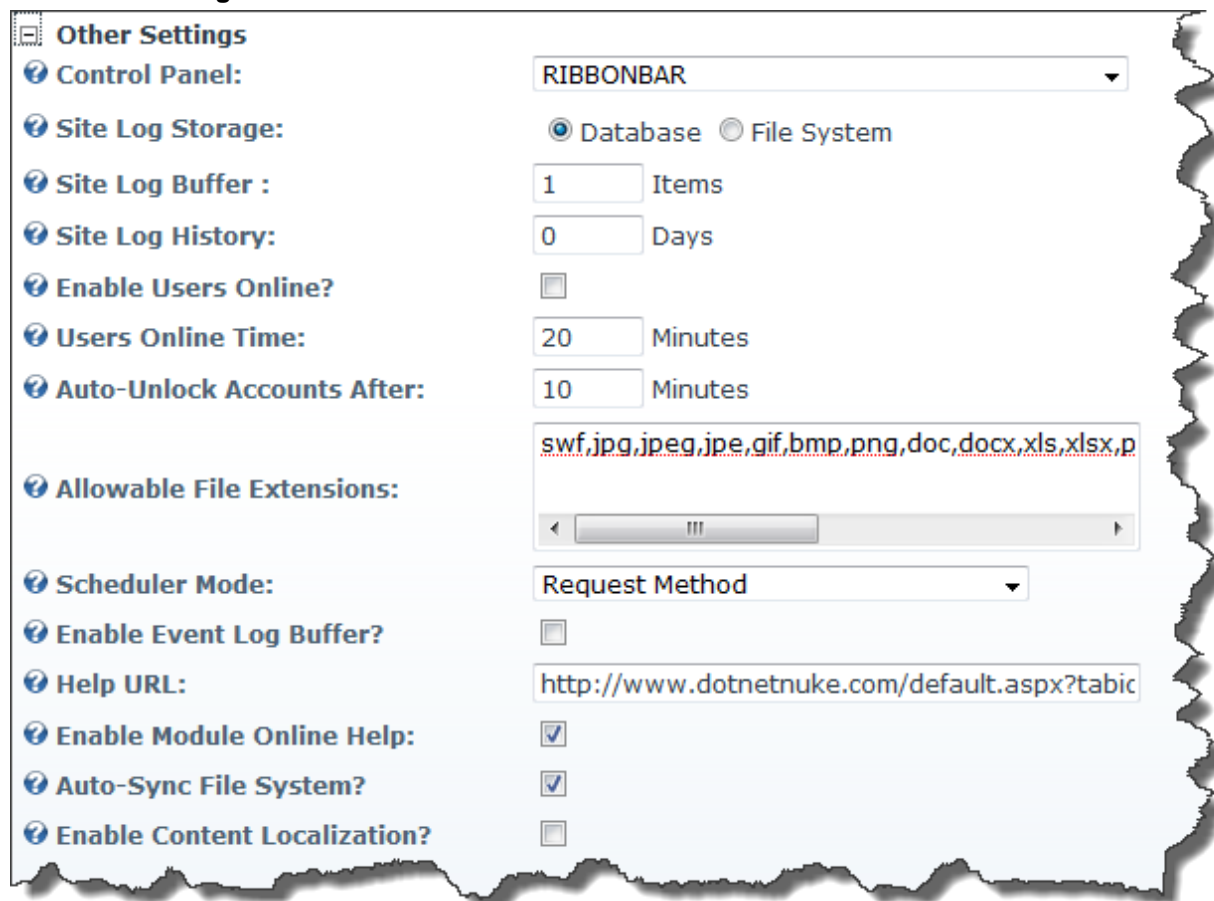
### 3.2.5 *Output Cache Provider (5.x - PE only by default)*

DotNetNuke Professional Edition added another type of caching provider into the available extension points. Specifically the Output Cache Provider was created, this provider is used to cache the entire content of a generated page allowing for all database lookups and page generation activities to be skipped for un-changed content.

The functionality provided by this cache provider is very similar in nature to a portion of the functionality offered by the commercial PageBlaster module available from Snapsis. The extension point for this provider type is also available for DotNetNuke Community Edition, however at this time there are no implementations of the provider available.

For users with this option enabled IowaComputerGurus recommends using the same setting for this as what was used for "Module Caching". As the same rules and impacts apply to this type of caching. Section 5 of this document talks a bit more about the Output Cache Provider.

### 3.3 Other Settings



**Other Settings**

**Control Panel:** RIBBONBAR

**Site Log Storage:**  Database  File System

**Site Log Buffer :** 1 Items

**Site Log History:** 0 Days

**Enable Users Online?**

**Users Online Time:** 20 Minutes

**Auto-Unlock Accounts After:** 10 Minutes

**Allowable File Extensions:** swf,jpg,jpeg,jpe,gif,bmp,png,doc,docx,xls,xlsx,p

**Scheduler Mode:** Request Method

**Enable Event Log Buffer?**

**Help URL:** http://www.dotnetnuke.com/default.aspx?tabid

**Enable Module Online Help:**

**Auto-Sync File System?**

**Enable Content Localization?**

This is another section of the configuration that is often overlooked, not all settings here apply to performance optimization, however, the specific items that do apply are discussed in detail below.

#### 3.3.1 Disable Users Online (4.x) / Enable Users Online (5.x)

Users online is a facility that allows DotNetNuke to show you which users are online at the current time. This process adds a fair amount of overhead and it is recommended that you do NOT have users online enabled. (In 5.x the box should be unchecked, in 4.x the box should be checked).

#### 3.3.2 Site Log History

Site Log History controls the data retention policy of the Site Log functionality available in the DotNetNuke core. For performance reasons IowaComputerGurus Inc recommends that the Site Log History setting be set to 0 days which disables the site log functionality.

This recommendation is due to the plethora of other processes available to obtain site statistics and the dramatic performance reduction that is realized when the log data fills up.

#### 3.3.3 Scheduler Mode

This setting controls the manner in which DotNetNuke scheduled tasks are triggered. The default configuration of “Request Method” requires that on page requests a check is performance and any needed tasks are triggered. It has been our experience that this mode introduces additional request overhead and can delay end-user experiences. Therefore our recommended configuration is to use the “Timer Method”. This launches the scheduler on a separate thread rather than using the request to poll to tasks. With the other recommended configurations we have also

experienced more consistent execution schedules on tasks.

#### **3.3.4 Enable Event Log Buffer?**

Various items in DotNetNuke write entries to the event log, user login, page changes, etc. Each time without this option enabled the insert is done right as the action occurs. Enabling the buffer allows DotNetNuke to queue these items and lets the user move on to other items faster. Therefore it is our recommendation that this setting be enabled on all installations.

#### **3.3.5 Auto-Sync File System**

This is the final configuration option in this section and relates to an automatic sync functionality that updates the file system information every time a user uses the UrlControl. It is our recommendation to disable this functionality as it is only needed if files are added to the portal via FTP on a regular basis.

### **4. DotNetNuke Scheduler**

Another default DotNetNuke configuration that is worth reviewing to ensure that a site is optimized for the best performance is the collection of Scheduled tasks. Scheduled tasks provide vital functions for DotNetNuke, however, if a specific task runs too frequently it can cause an adverse reaction to the site performance, in some cases it can even render a site useless.

Typically the most common scheduled item to modify is the "Search Engine Indexer" task. This scheduled task is used to index site content so that users can search the site. The default site configuration has this task running every 30 minutes, in most cases this is much too frequent. DotNetNuke when indexing content searches through ALL content, not just updated content. Therefore on large sites the index process itself can take more than 30 minutes to process, having an adverse impact on the system.

Regardless of error conditions that might arise from long running tasks, this task also uses a high amount of SQL Server resources which then can slow the site down. Therefore our recommendation is to modify the schedule from once every 30 minutes to once every 12-24 hours. This way the content still gets updated on a regular basis for search, but we limit the potential performance impact of the operation.

### **5. DotNetNuke Output Cache Provider**

Starting with the DotNetNuke 5.2.x release a new extension point was introduced in DotNetNuke Community Edition (CE) and a new provider was bundled with the DotNetNuke Professional Edition (PE). This new extension point is to allow the support of an Output Cache Provider. What is this? Well in laymen's terms it is a method that allows an entire page to be cached, in the output form. What this translates to is no database communication is necessary to serve the page if it is successfully cached, from a performance perspective, this is not just a "massive improvement" it is a night and day difference when enabled. The following sections will talk about options available

#### **5.1 Obtaining an Output Cache Provider**

Out of the box DotNetNuke CE does not contain any Output Cache Providers. Certain third-party developers

### **6. Other Configurations and Next Steps**

Sections two through four of this document discuss basic configuration items that can be modified to ensure that DotNetNuke performs to the best level possible. This section covers the remaining items, other tips and tricks to ensure that the site has been fully optimized AND stays running at top performance.

#### **6.1 Regular Clearing of Event Log**

One of the built in functions within DotNetNuke is the "Event Log" (Admin -> Event Log), this is a log that records information such as user login success/failure, general module exceptions, and critical site level changes. The event log is a great resource, however, it is also very important that this table be managed. DotNetNuke by default does not clear this table, therefore, over time the log can grow to be very large.

As the size of the EventLog increases, performance will decrease due to the additional costs of inserting records. In a typical situation IowaComputerGurus recommends truncating this table on a daily basis. We make this recommendation because this allows for a 24 hour rolling history for any diagnostics purposes and avoids rapid transaction log growth when deleting records.

This can be accomplished a number of ways; SQL Server Scheduled Job, manual process, or using a module such as the free "Scheduled SQL Jobs" module which is available from our website.

## **6.2 Regular Monitoring of Event Log Errors**

Along with the regular cleaning of the event log it is important to review the error messages that are reported. If recurring errors are found it is important to resolve these items. The process of throwing exceptions and logging add un-necessary overhead to the system. Many recurring errors are common items that are easy to find, if you find a recurring error you can easily go to the DotNetNuke Forums or our forums to ask for advice on how you might resolve the recurring errors.

Examples of common error messages would include regular messages each time the application starts that refer to a module that was not successfully installed, or a module that has invalid code at an extension point that is not readily visible to the site administrator. An example of this might be a module that integrates with the DotNetNuke search system and throws an error every time DNN tries to index the module's content. These errors are ONLY reported to the event log and would otherwise go un-noticed.

## **6.3 Skin Modifications**

Another key piece of a site that can impact the performance of a site is the way in which the skin is built. Since the skin defines the user interface if the code here is in-efficient or overly heavy it can add additional time to download and render, thus slowing the site. There are two key recommendations that we have when it comes to skin design and performance.

### *6.3.1 Switch from SolPart/DNN Menu to Snapsis or Telerik*

One of the biggest performance improvements that we have been able to find is to switch away from the SolPart menu to a menu provider that renders the menu using a bulleted list markup rather than a Java-Script or other type interface. Switching to the Snapsis menu for us on our own site, realized a 60% improvement in page load times with no additional changes.

Sadly though this is not a change that is just a quick swap, it does take a bit of experience to modify the skin to get things working properly with the different provider.

### *6.3.2 Create an XHTML Compliant CSS Based Skin*

The other item here that helps with site performance is to ensure that your skin is XHTML compliant and uses a CSS based layout. The XHTML compliance ensures that you have valid HTML which allows the browser to more quickly render the content once downloaded from the web server. A CSS based skin is typically lighter in terms of HTML markup which reduces the overall size of rendered content, reducing the time necessary to download the content.

## **6.4 Third-Party Components**

In addition to the above if you find that your site performance just isn't up to the level that you need there are third party solutions out there that implement even more aggressive caching strategies. One of the most popular items here is Snapsis Page Blaster which can cache content to static HTML files, with proper configuration you can realize even more performance benefits, however, it does take some planning/testing/configuring to realize benefit.

## **6.5 Seek Outside Assistance**

Finally if none of the items in this document have resolved your performance issues there are many vendors in the DotNetNuke ecosystem that offer performance optimization services. If you are interested in IowaComputerGurus performance optimization services you may e-mail Mitchel directly at msellers@iowacomputergurus.com.

## **7. Contact IowaComputerGurus**

We welcome any feedback that you might have to this document as we have been constantly updating the document based on new findings and information as it comes available. If you have specific questions of the content of this document or would like to share any feedback we can be reached through any of the following communication channels.

IowaComputerGurus Inc.  
12508 Horton Avenue  
Urbandale, Iowa 50323

**Phone:** (515) 270-7063

**Fax:** (866) 591-3679

**Twitter:** @IowaCompGurus

**Email:** [webmaster@iowacomputergurus.com](mailto:webmaster@iowacomputergurus.com)

**Website:** <http://www.iowacomputergurus.com>