

Sitecore CMS 7.0 or later

Scaling Guide

An administrator's guide to configuring multiple related Sitecore instances

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

Table of Contents

Chapter 1 Introduction ... 3
Chapter 2 Standard Multi-Instance Configuration ... 4
2.1 Configuring a Standard Multi-Instance Environment ... 5
2.2 Configuring the Database Servers ... 6
2.3 Configuring the Content Management Environment ... 7
2.4 Configuring the Content Delivery Environment ... 8
Chapter 3 Advanced Configuration Topics ... 9
3.1 Configuring a Remote Publishing Target ... 10
3.2 Isolating the CM and CD Environments ... 11
3.2.1 Overview ... 11
3.2.2 Addressing Security Synchronization ... 11
3.3 Isolating Instances within the CM or CD Environments ... 13
3.4 Configuring Automatic File Synchronization ... 15
3.4.1 Windows Distributed File System (DFS) ... 15
3.4.2 Web Deployment ... 16
3.5 Session State Management ... 17
3.6 Synchronizing Link Database, IDTable, and Search Indexes ... 18
3.6.1 Link Database ... 18
3.6.2 The IDTable ... 18
3.6.3 Lucene Search Indexes ... 19
3.7 Scaling Sitecore DMS ... 20
3.7.1 DMS Configuration per Instance ... 20
Configuring the Default Analytics Definition Database ... 20
Configuring Scheduling ... 20
3.8 SQL Server Database Mirroring and Failover ... 22
3.8.1 Conceptual Overview ... 22
3.8.2 Configuring Database Mirroring in SQL Server ... 23
3.8.3 Configuring Database Mirroring in Sitecore ... 23
Enabling the SQL Server Retryer in the Web Config ... 23
Enabling Mirroring in the Connection String Config File ... 24
Chapter 4 Configuration Tasks ... 25
4.1 How to: Enable Scalability Settings ... 26
4.2 How to: Disable or Enable File Based Media ... 27
4.3 How to: Configure the Name of the Publishing Instance ... 28
4.4 How to: Configure HTML cache clearing ... 29
4.5 How to: Configure a Machine Key ... 31
4.6 How to: Remove References to the Master Database ... 32
Using an Include File to Remove References to Master Database ... 32
Manually Removing References to the Master Database ... 32
4.7 How to: Restrict Access to Client Interfaces ... 34
4.8 How to: Configure a Remote Publishing Target ... 35
Creating a Publishing Target Database ... 35
Registering a Publishing Target Database ... 35
4.9 How to: Configure SQL Server Replication ... 36
4.10 How to: Address Security Caching Clearing ... 37
4.11 How to: Configure DFS ... 38
4.12 How to: Configure Web Deployment ... 47
How to: Configure Multiple Web Deployment Targets ... 48

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

Chapter 1

Introduction

This document describes how to scale Sitecore CMS solutions by configuring multiple instances in one or more environments, such as, Content Management (CM) and Content Delivery (CD) environments. There is also a description of some alternative configurations that could be implemented to meet certain infrastructure and security requirements.

Readers should be familiar with the Sitecore Content Editor and should understand the Sitecore application architecture, including installation, the Sitecore databases, and the Sitecore security system. Readers should also be familiar with Microsoft Windows, Internet Information Services (IIS), and either Microsoft SQL Server or Oracle. Some of the instructions in this document require the user to have administrative rights in Sitecore and Windows.

Before you read this *Scaling Guide*, read the *Introducing Sitecore CMS 6.3* manual in its entirety. Before installing Sitecore CMS in a multi-instance configuration, we recommend that you read the *Chapter 2* of this *Scaling Guide* in its entirety.

The information in this manual applies to Sitecore CMS 6.5 and 6.6 or later. The information in this manual does not apply to Sitecore Azure implementations. For more information about Sitecore Azure, see the manual *Getting Started with Sitecore Azure*.

This manual contains the following chapters:

- **Chapter 1 — Introduction**
  This introduction.
- **Chapter 2 — Standard Multi-Instance Configuration**
  This chapter describes how to configure Sitecore in a multi-instance environment.
- **Chapter 3 — Advanced Configuration Topics**
  This chapter describes how to perform a number of more advanced tasks including how to configure a remote publishing target and isolate the different environments.
- **Chapter 4 — Configuration Tasks**
  This chapter describes some other important tasks that administrators need to perform.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

Chapter 2

Standard Multi-Instance Configuration

This chapter describes how to configure a standard multi-instance solution. This solution contains multiple instances of both Content Management (CM) and Content Delivery (CD) servers, all of which are hosted at the same physical location.

It is important to apply configuration steps from this chapter to all instances in every Sitecore solution that consists of more than one instance. These are required and sufficient steps to configure Sitecore instances for the production-ready usage and enable such features as automatic cache synching and distributed publishing.

This chapter contains the following sections:

- Configuring a Standard Multi-Instance Environment
- Configuring the Database Servers
- Configuring the Content Management Environment
- Configuring the Content Delivery Environment

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

2.1 Configuring a Standard Multi-Instance Environment

The Sitecore configuration described in this chapter is appropriate for multi-instance solutions that include multiple instances of both Content Management (CM) and Content Delivery (CD), all of which are hosted at the same physical location. This is the simplest configuration and may be appropriate for the majority of Sitecore installations.

The standard single-instance Sitecore configuration includes three databases:

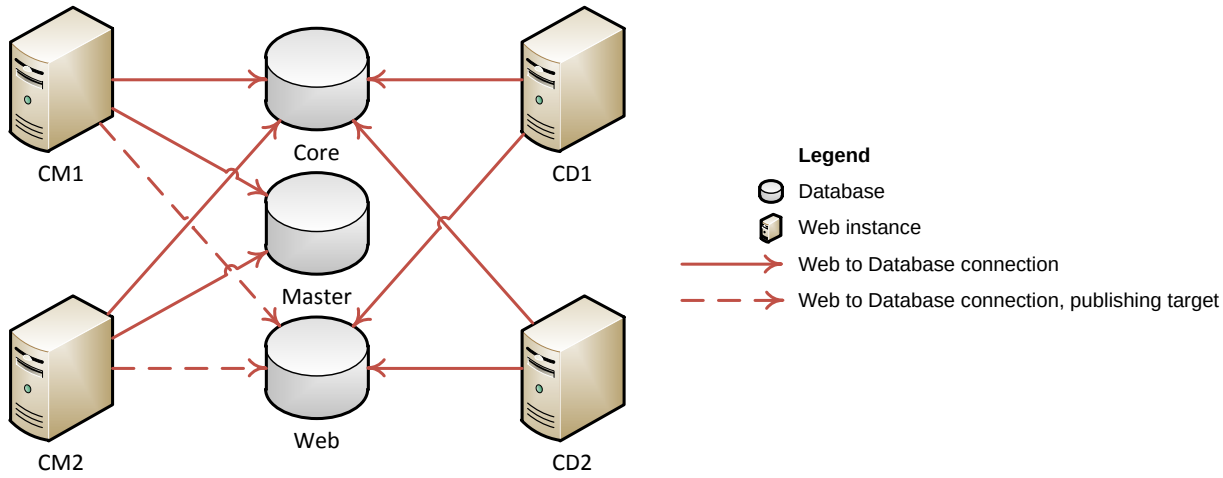
- The Master database contains every version of every item in every language, including unpublished work in progress and is the source of the published website.
- The Web database is the default publishing target and only contains the published version(s) of each item including the different language versions that have been published.
- The Core database contains the configuration information about the Sitecore user interfaces.

The standard multi-instance Sitecore setup uses the same three databases, shares the Core and Web databases between all the CM and CD instances, and shares the Master database between all the CM instances.

In the standard multi-instance Sitecore setup, you must make one of the CM instances the publishing CM instance.

In the standard multi-instance Sitecore setup, you must store all media item data in the Sitecore databases and you must therefore disable file based media.

The following diagram illustrates the connections between the Web instances and the Sitecore databases in the standard multi-instance Sitecore setup:



Note

If you are using Sitecore DMS, a multi-instance setup also includes an Analytics database. The Analytics database should be shared between every instance in both the CM and the CD environments. For more information about scaling Sitecore DMS, see the *Scaling Sitecore DMS* section.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

2.2 Configuring the Database Servers

Sitecore recommends that you install the Core, Master, and Web databases on a single database server or multiple database servers that are located on the same network as all of the Web servers.

If you are using Sitecore DMS, we recommend that you install the Analytics database on a dedicated database server. For more information about scaling Sitecore DMS, see *Scaling Sitecore DMS* section.

You can use the Sitecore installation program to install the Sitecore databases or you can install them manually from the Sitecore package. These are both available on the SDN.

For more information on installing the Sitecore SQL and Oracle databases, see the *Sitecore CMS Installation Guide*.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

2.3 Configuring the Content Management Environment

To configure the CM environment, you must set up one or more CM instances on one or more Web servers.

Sitecore recommends that you use a dedicated Web server for every Sitecore CMS instance in the environment.

To configure an instance of Sitecore CMS in a CM environment:

1. Install a Sitecore CMS web application on the server.

   For more information about installing Sitecore CMS, see the *Sitecore CMS Installation Guide.*

2. Configure the web application to connect to the Core, Master, and Web databases from the database server.

3. Enable the scalability settings.

   For more information about enabling scalability settings, see the section

   How to: Enable Scalability Settings.

4. Disable all file based media.

   For more information about disabling file based media, see the section *How to: Disable or Enable File Based Media*.

   For more information about an alternative way of working with file based media if you do not want to disable them, see the section *Configuring Automatic File Synchronization*.

5. Configure the name of the publishing instance.

   For more information about configuring the name of the publishing instance, see the section *How to: Configure the Name of the Publishing Instance*.

6. Configure HTML cache clearing.

   For more information about configuring HTML cache clearing, see the section *How to: Configure HTML cache clearing*.

7. Configure the machine key section.

   For more information about configuring the machine key section, see the section *How to: Configure a Machine Key*.

8. Synchronize the operating system time between Content Management and Content Delivery environments.

   The operating system time must be the same for all CM, CD and database instances on all Web Servers and Database Servers.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

2.4 Configuring the Content Delivery Environment

To configure the CD environment, you must set up one or more CD instances on one or more Web servers.

Sitecore recommends that you use a dedicated Web server for every Sitecore CMS instance in the environment.

To configure an instance of Sitecore CMS in a CD environment:

1. Install a Sitecore CMS web application on the server.

   For more information about installing Sitecore CMS, see the *Sitecore CMS Installation Guide*.

2. Configure the web application to connect to the Core and Web databases from the database server, but not to the Master database.

3. Remove every reference to the Master database.

   For more information about removing references to the Master database, see the section *How to: Remove References to the Master Database*.

4. Enable the scalability settings.

   For more information about enabling scalability settings, see the section

   How to: Enable Scalability Settings.

5. Disable all file based media.

   For more information about disabling file based media, see the section *How to: Disable or Enable File Based Media*.

   For more information about an alternative way of working with file based media if you do not want to disable them, see the section *Configuring Automatic File Synchronization*.

6. Configure HTML cache clearing.

   For more information about configuring HTML cache clearing, see the section *How to: Configure HTML cache clearing*.

7. Configure the machine key section.

   For more information about configuring the machine key section, see the section *How to: Configure a Machine Key*.

8. Restrict access to the Sitecore CMS client interfaces.

   For more information about restricting access to the Sitecore CMS client interfaces, see the section *How to: Restrict Access to Client Interfaces*.

9. Synchronize the operating system time between Content Management and Content Delivery environments.

   The operating system time must be the same for all CM, CD and database instances on all Web Servers and Database Servers.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

Chapter 3

Advanced Configuration Topics

This chapter describes some possible deviations from the standard multi-instance Sitecore CMS setup that you might need to implement to meet various infrastructure and security requirements. It contains configuration instructions for solutions that have additional requirements.

This chapter contains the following sections:

- Configuring a Remote Publishing Target
- Isolating the CM and CD Environments
- Isolating Instances within the CM or CD Environments
- Configuring Automatic File Synchronization
- Session State Management
- Synchronizing
- Scaling Sitecore DMS
- SQL Server Database Mirroring and Failover

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.1 Configuring a Remote Publishing Target

The standard Sitecore CMS multi-instance configuration includes three databases — Core, Master, and Web. The Web database is the default publishing target used for rendering the website on both the CM and CD environments.

Another typical option is to configure a remote publish target to be used on the CD instances for rendering the website. This allows you to use the default Web target on the CM environment to publish and preview the content of the website on the CM environment without affecting the CD environment.

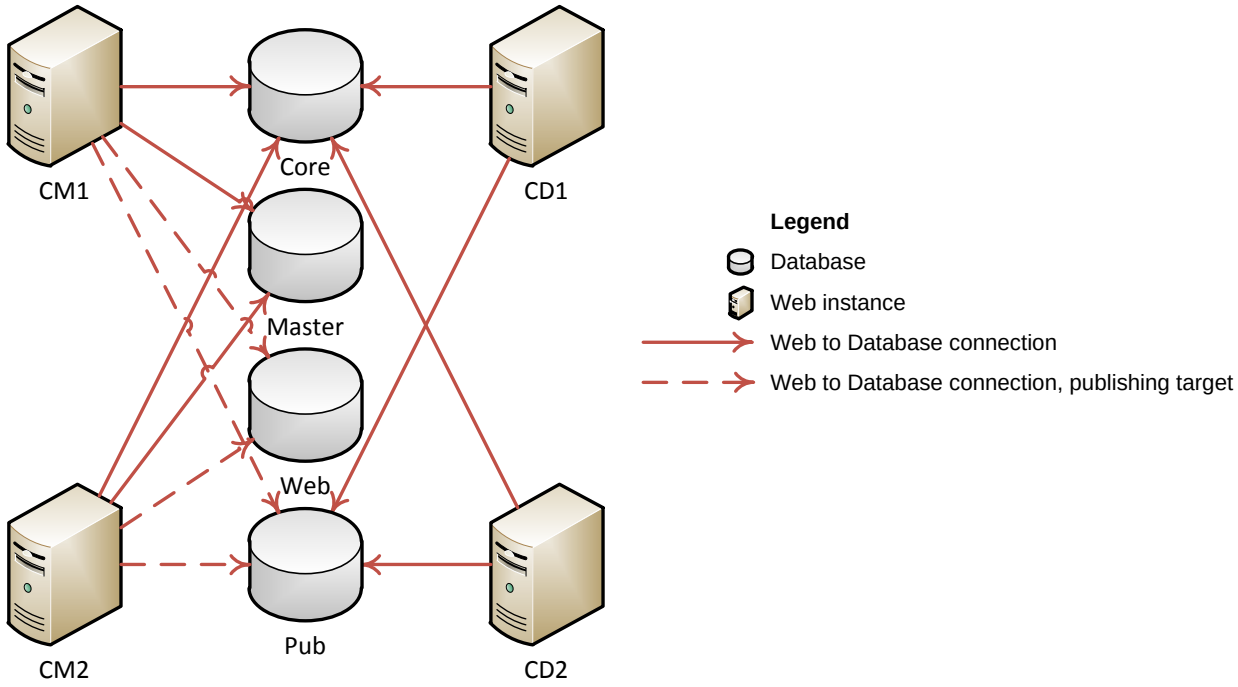
A Sitecore CMS multiple-instance configuration like this includes four databases — Core, Master, Web, and Pub (CD Web).

The CM instances connect to all four databases and have two publishing targets that link to Web and Pub.

The CD instances connect to the Core and Pub databases.

For more information about configuring a remote publishing target, see the section *How to: Configure a Remote Publishing Target*.

The following diagram illustrates the connections between the Web instances and the Sitecore databases for the standard multi-instance Sitecore setup with a remote publishing target:



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.2 Isolating the CM and CD Environments

The CM and CD environments typically need to be isolated to meet security or infrastructure requirements. You can also use the information in this section when you configure geographically distributed CM and CD environments.

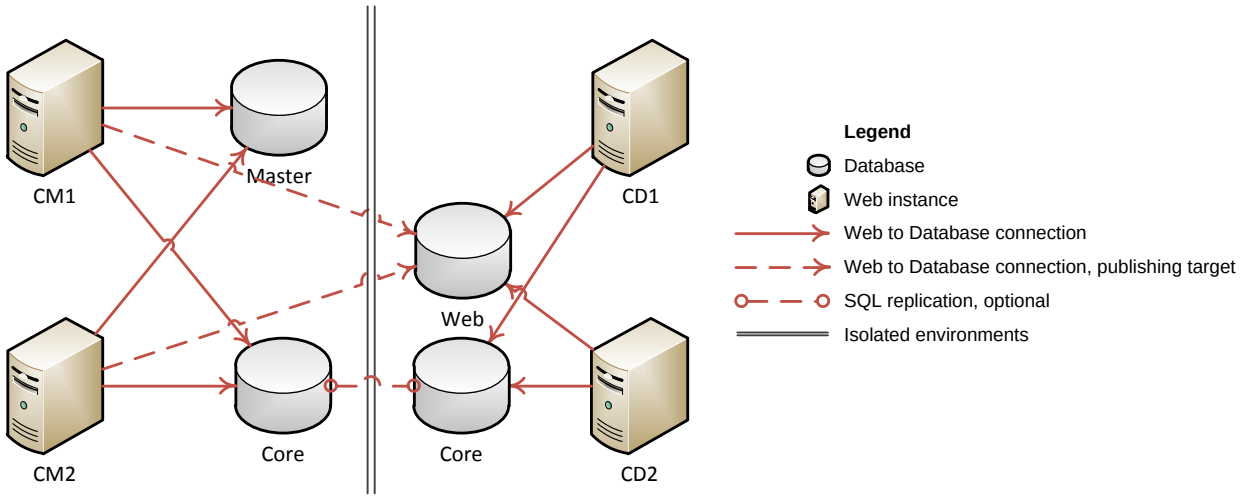
3.2.1 Overview

A typical isolated CM and CD setup consists of:

- Two separate database servers, one for the CM environment and one for the CD environment.
- A Core and a Master database on the CM database server.
- A Core and a Web database on the CD database server.
- One or more CM web instances that are connected to the CM databases and the Web database on the CD database server.
- One or more CD web instances that are connected to the CD databases.

Each CM web instance and each CD web instance should be configured according to the instructions in *Chapter 2*.

The following diagram illustrates the connections between the web instances and the Sitecore databases in a Sitecore setup with isolated CM and CD environments:



Note
You can to use the instructions in the section *How to: Configure a Remote Publishing Target* to configure this setup so that the Web database is placed in the CM environment and the Pub database is placed in the CD environment.

3.2.2 Addressing Security Synchronization

In an isolated CM and CD environment, security data and caches are automatically synchronized within every instance of the CM environment, as well as within every instance of the CD environment. This happens because every instance within the environment shares the same databases, and therefore

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

shares the same security data and system event queue. However, security data and caches are not automatically synchronized between the isolated CM and CD environments.

This only includes security data about the membership, role, and profile information stored in .NET security providers that are configured for the Sitecore application. This does not include the security permissions that users are assigned to content items which are stored in the Sitecore content databases. This Sitecore security information is automatically synchronized when the content database is shared between environments and during the publishing process.

For more information about the system event queue, see *Introducing Sitecore CMS 6.3.*

Note
For more information about how the .NET security provider is implemented and configured in Sitecore, see the manual *Low level Sitecore Security and Custom Providers*.

There are different ways to address security synchronization in an isolated CM and CD setup depending on the implementation requirements.

- If your solution does not require synchronization of security data and caches, you do not need to perform any of the modifications described in this section. For example, if the CD instances do not use any extranet security, synchronization may be not required.
- If your solution requires live synchronization of the security data and caches between the CM and CD environments from the default SQL membership, role, and profile providers, plus security caches from additional security providers (Active Directory, CRM, or custom providers), we recommend that you configure database replication to synchronize the CM Core database with the CD Core database.

  This will ensure that the security account data that is stored in these two databases, as well as related security events that are required for synchronizing security caches are replicated between the environments.

  For more information about configuring database replication, see the section *How to: Configure SQL Server Replication*.
- If your solution does not require live synchronization of security data between the CM and CD environments from the default SQL membership, role, and profile providers, but requires synchronization of security caches from additional security providers (Active Directory, CRM, custom providers), it is enough to only address security cache clearing rather than replicating the Core database.

  For more information about addressing security cache clearing, see the section *How to: Address Security Caching Clearing*.

Note
You can also use the latter approach to addressing security caching if the SQL security providers' data is shared between the environments, while the Core database that stores the system event queue is not. This requires extracting the security related tables from the Core database to a dedicated database, reconfiguring all the SQL security providers, and has a negative effect on performance in geographically distributed setups. This approach is currently not documented.

For more information about the implementation and configuration of Sitecore CMS security and other caches, see the manual *Cache Configuration Reference*.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.3 Isolating Instances within the CM or CD Environments

In addition to isolating the CM and CD environments, infrastructure requirements may force different instances within the same environment to be isolated. This is typically relevant when you are configuring geographically separated Sitecore instances within an environment.

To geographically separate instances within the CM or CD environment to two or more locations, perform the configuration steps described in the *Isolating the CM and CD Environments* section, with the following differences:

- Every Sitecore database within the environment should have a copy located in each geographical location.
- Every web instance within a geographical location in the environment should connect to its local copies of the Sitecore databases (and in the CM environment, it should connect to the nearest Web database).
- Every Sitecore database in the CM environment should have database replication configured for all of its copies within the environment.
- Every Sitecore database in the CD environment should optionally have database replication configured for all of its copies within the environment.

Note
When each CD Web database is configured as a separate publishing target, it is not necessary to configure Web database replication in the CD environment.

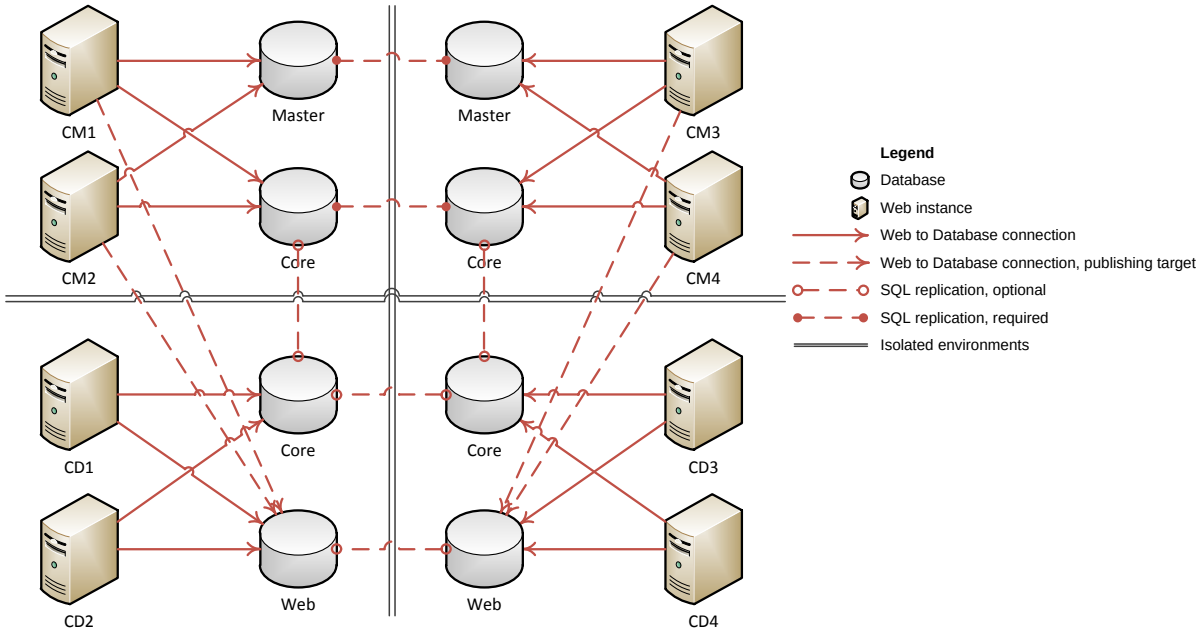
For more information about configuring remote publishing targets, see the section *How to: Configure a Remote Publishing Target*.

For more information about when it is not necessary to configure Core database replication in the CD environment, see the section *Addressing Security Synchronization*.

For more information about configuring database replication, see the section *How to: Configure SQL Server Replication*.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

The following diagram illustrates the connections between the Web instances and the Sitecore databases for a setup that contains isolated instances within the CM and CD environments:



Note
You can apply the instructions in the *How to: Configure a Remote Publishing Target* section to this configuration so that the setup contains the Web database on the CM environments and the Pub database on the CD environments.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.4 Configuring Automatic File Synchronization

In multi-instance Sitecore solutions, if you use the file system to store media data for Sitecore media items or if you change the Sitecore domain configuration, file and folder synchronization is required.

Sitecore supports two ways to configure automatic file system synchronization between multiple Sitecore instances:

- Windows Distributed File System (DFS)
- Web Deployment

Sitecore supports using these techniques for synchronizing media files and folders that correspond to Sitecore media items (`/App_Data/MediaFiles` folder), as well as security domain configuration files (`/App_Config/Security` folder).

Warning
Sitecore does not support using these techniques to synchronize files from folders other than those specified here. This is because the nature of ASP.NET applications, as well as the implementation details of DFS and Web Deployment that can cause conflicts when you configure both utilities to synchronize the same folders on the file system.
Synchronizing folders and files other than those specified here is not being a part of a regular content authoring process. Such folders and files are recommended to be deployed in the same manner as new builds of the Sitecore site – for example manually, using Sitecore Packager, or other deployment tools.

In a multi-instance Sitecore configuration, DFS and Web Deployment serve different purposes and have different limitations:

- DFS only allows you to synchronize files between web servers that are located in the same Windows Server domain, and in a bidirectional way.
- Web Deploy allows you to synchronize changes between servers from different domains, but only supports unidirectional synchronization.

We recommended that you use:

- DFS to synchronize file systems within the CM environment.
- Web Deployment to replicate media file changes to the deployment target in the CD environment.
- DFS within the CD environment to replicate those deployed changes to other CD instances.

3.4.1 Windows Distributed File System (DFS)

If you use file media, but the different instances do not share a common file system for media files, you can configure DFS to synchronize the subdirectory specified by the `Media.FileFolder` setting in the `/App_Config/Include/ScalabilitySettings.config` or in the `web.config` file between the different instances in the environment.

DFS synchronizes file systems between groups of servers. You can use DFS to synchronize file systems across the Sitecore instances in CM or CD environments. DFS works with subdirectories on local disk drives, without network drives or other types of shared resources.

If you add, remove, or update security domains, you can configure DFS to synchronize the directory containing the security domain configuration file.

When you configure DFS, you must configure a DFS replication group for each environment. There is typically one DFS replication group for the CM environment and another for the CD environment. When

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

you add additional CD or CM instances to your environments, you must also add these instances to the appropriate DFS replication group.

For more information about configuring DFS, see the section *How to: Configure DFS*.

3.4.2 Web Deployment

You can configure Sitecore to use the Microsoft Web Deployment tool to transfer file system changes from a source instance to one or more target instances. Specifically, you can configure Web Deployment of file media from the CM environment to the CD environment.

Sitecore uses the Web Deployment tool to deploy all additions, changes, and deletions from the specified source subdirectories and files to the target file systems. Unlike DFS, which replicates files to and from all servers in a replication group, Web Deployment is unidirectional. Web Deployment therefore only deploys changes from the CM publishing instance to the CD target instance and never from the target to the source.

Warning
Do not configure Web Deployment for files or subdirectories other than the subdirectory specified in the `Media.FileFolder` setting in the `/App_Config/Include/ScalabilitySettings.config` file or in the `web.config` file.

To minimize the use of network bandwidth, Web Deployment compares the file systems, and only deploys the differences.

For more information about configuring Web Deployment, see the section *How to: Configure Web Deployment*.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.5 Session State Management

The Sitecore CMS user interfaces require in-process ASP.NET session management. For CM instances, the value of the `mode` attribute of the `/configuration/system.web/sessionState` element in the `web.config` file must be `InProc`.

In-process session management requires you to configure the CM load balancer for server affinity — also known as sticky sessions. You can use other values for the `mode` attribute in the CD environment.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.6 Synchronizing Link Database, IDTable, and Search Indexes

In multi-instance Sitecore setups, there are some more Sitecore CMS features that you must configure if you have implemented the following features on your installation.

3.6.1 Link Database

You can use the Sitecore link database API to access all the items that are referenced by or refer to a specific Sitecore item.

All the link database information is stored in the Core database by default, and aggregates data for the items from every Sitecore content database.

You configure the location of the link database in the `web.config` in the `Link Database` section file in the `connectionStringName` parameter:

```
            <!-- LINK DATABASE -->
            <LinkDatabase type="Sitecore.Data.$(database).$(database)LinkDatabase,
Sitecore.Kernel">
              <param connectionStringName="core" />
            </LinkDatabase>
```

In a single-server Sitecore setup, the link database is automatically updated.

In a multi-server Sitecore setup, the link database is only automatically updated for databases that are referenced from the CM environment.

To ensure that Sitecore automatically updates the link database in the CD environment:

- The CD and CM instances must use the same name to refer to the publishing target database across the environments (typically Web).
- One of the following conditions should be met:
  - The Core database should be shared or replicated between the CM and CD instances.
  - The Link database data should be configured to be stored in a database which is shared between CM and CD publishing target database (typically Web).

3.6.2 The IDTable

You can store persistent mappings of arbitrary keys to Sitecore IDs in the Sitecore IDTable.

By default, this is only used by the Sitecore CMS WebDAV functionality to keep locks on media items in the CM environment and does not need to be synchronized with the CD environment.

You configure the location of the IDTable in the `web.config` file in the `IDTable` section in the `connectionStringName` parameter:

```
            <IDTable type="Sitecore.Data.$(database).$(database)IDTable, Sitecore.Kernel"
singleInstance="true">
              <param connectionStringName="master" />
              <param desc="cacheSize">500KB</param>
            </IDTable>
```

To synchronize the contents of the IDTable between the CM and CD environments, change the `connectionStringName` parameter in the IDTable configuration section to point to a Core database that is shared or replicated between the environments. Alternatively, it can point to a shared Web database.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.6.3 Lucene Search Indexes

Sitecore automatically maintains the Lucene search indexes for items in the Master and Core databases in the CM environment, and for items in the Core database in the CD environment by default. Sitecore keeps a local copy of the Lucene index in the file system of each instance and does not support sharing the indexes between instances.

When you configure the Lucene indexes for the Web database, you should enable History Engine for this database on all the Sitecore instances in the CM and CD environments to ensure that the indexes are automatically updated and synchronized.

To enable History Engine for a Sitecore database:

In the `web.config` file, add the following section to the `/configuration/sitecore/databases/database` element, where `id` equals the name of the database:

```
<Engines.HistoryEngine.Storage>
  <obj type="Sitecore.Data.$(database).$(database)HistoryStorage, Sitecore.Kernel">
    <param connectionStringName="$(id)" />
    <EntryLifeTime>30.00:00:00</EntryLifeTime>
  </obj>
</Engines.HistoryEngine.Storage>
<Engines.HistoryEngine.SaveDotNetCallStack>false</Engines.HistoryEngine.SaveDotNetCallStack>
```

When a Sitecore item is changed, the Lucene indexes are updated immediately on the Sitecore instance where the change was made.

On remote servers in a multi-server environment, the Lucene indexes are not updated immediately after an item is changed. The Lucene indexes are automatically updated after the interval that is defined in the `web.config` file, in the `Indexing.UpdateInterval` setting and with the minimum wait time between the two consequent updates defined in the `Indexing.UpdateJobThrottle` setting.

For more information about integrating and configuring Lucene search integration, see the *Sitecore Search and Indexing* article on the SDN.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.7 Scaling Sitecore DMS

Information and recommendations about the server requirements, installation, and configuration of Sitecore DMS and CMS are available in the:

- *Installation Guide for DMS on SQL Server.*
- *Engagement Analytics Configuration Reference.*
- *DMS Performance Tuning Guide.*
- *CMS Performance Tuning Guide.*

When you configure Sitecore DMS in a multi-instance Sitecore setup, we recommend that:

- You install the *Analytics* database on a dedicated database server for performance reasons.
- If you use separate databases for collecting and reporting DMS statistics, you should install these on dedicated servers.
- All Sitecore CMS instances share the same Analytics database.

3.7.1 DMS Configuration per Instance

Configuring the Default Analytics Definition Database

The analytics definition database is the database that contains all of the DMS settings that you have configured for your website.

The default database that is used to store analytics definition data is the *master* database. However, in a production environment, this is generally not the *master* database. You should therefore ensure that your configuration files point to the correct database – typically the *web* database.

To configure the default analytics definition database:

On each CD instance, in the `/configuration/sitecore/settings` node of the `/App_Config/Include/Sitecore.Analytics.config` file, add the following setting:

```
<setting name="Analytics.DefaultDefinitionDatabase" value="web" />
```

Configuring Scheduling

The DMS can distribute reports by email. When you scale the environment, you can configure which instances should send them. Typically, you will have one Sitecore CM instance that send reports.

To prevent a particular Sitecore instance from sending reports by email, comment out the following string in `Analytics.config` file:

```
        <scheduling>
              <agent type="Sitecore.Analytics.Tasks.EmailReportsTask, Sitecore.Analytics"
method="Run" interval="1:00:00">
                <DatabaseName>master</DatabaseName>
              </agent>
              <agent type="Sitecore.Analytics.Tasks.SubscriptionTask, Sitecore.Analytics"
method="Run" interval="00:15:00">
                <Minutes>15</Minutes>
              </agent>
        </scheduling>
```

The `Sitecore.Analytics.Tasks.EmailReportsTask` agent is responsible for sending reports. For more information about this, see the *Sitecore Analytics Configuration Reference*. If you use this agent not in the CM instance, specify the `DatabaseName`.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

The `Sitecore.Analytics.Tasks.SubscriptionTask` agent is responsible for sending reports for the subscribed users. For more information about this, see the *Marketing Operations Cookbook*.

Note
The `Sitecore.Analytics.Tasks.UpdateReportsSummaryTask` agent must also be executed only from one server of the entire environment to prevent multiple rebuilding of the report data.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.8 SQL Server Database Mirroring and Failover

Implement SQL Server database mirroring and failover to increase the availability of your Sitecore databases. Also, in the event of a disaster, you can use automatic failover to quickly bring a standby copy of your database back online.

3.8.1 Conceptual Overview

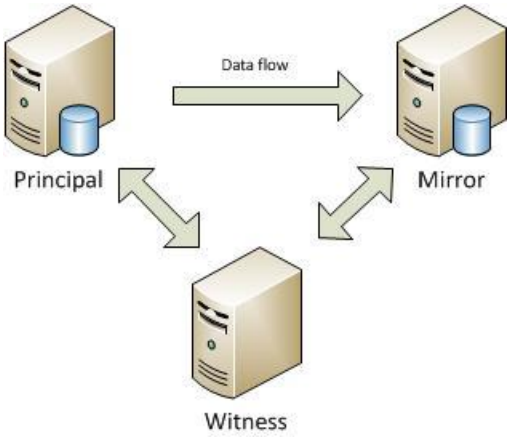
SQL Server database mirroring requirements:

- *SQL Server 2008 or later*.

To set up database mirroring with automatic failover you need at least three database server instances:

- *Principal* – primary database server (must be set to the full recovery model).
- *Mirror* – full backup copy of the principal database.
- *Witness* – monitors the other two database server instances (no databases are installed on this server instance).

Use SQL Server Standard or Enterprise edition for the Principal, Mirror and Witness instances. The Principal and Mirror instances should have the same version of SQL Server (including hotfixes) installed.



Database mirroring transfers transaction log records directly from one server to another (between the *Principal* and *Mirror* instance) and can quickly fail over to the standby server.

When you make any change in Sitecore, the changes are first recorded in the transaction log in the *Principal* database. At the same time, the transaction log of the *Mirror* is updated. This means that the changes are written to the *Principal* and *Mirror* database at the same time, so the *Mirror* is always kept in sync with the *Principal* database.

The *Witness* server instance constantly monitors the other database instances to detect problems. If it detects that the principle server has become unavailable then it enables failover and switches the principal role to the mirroring server.

SQL Server database mirroring can failover quickly with minimal data loss, does not require any special hardware, is quick to set up, and easy to manage.

For more detailed conceptual information and steps on how to configure SQL Server for database mirroring, see the MSDN article *Database Mirroring (SQL Server)*.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

3.8.2 Configuring Database Mirroring in SQL Server

This section contains some considerations and an overview of the steps you need to take when you implement SQL Server database mirroring. For more detailed information and a complete list of the steps to follow, refer to Microsoft Books Online or the MSDN website.

Some important things to note about database mirroring:

- The *Principal* database must be in the FULL recovery model. Sitecore databases use the Simple recovery model by default. You should change this to the FULL recovery model for mirroring.
- The *Mirror* database must be initialized from a restore of the *Principal* database with NORECOVERY, followed by restores in sequence of *Principal* transaction log backups.
- The *Principal* and the *Mirror* databases must be running the same version of SQL Server.
- Check that the *Mirror* server has sufficient disk space for the *Mirror* database.
- Because the *Mirror* database is in a recovering state, you cannot access it directly. You can create database snapshots on the *Mirror* to indirectly read the *Mirror* database at a point in time.

Summary of steps:

1. Make a backup of the database and restore it to the intended *Mirror* server without recovery.
2. Create a trust relationship between all the database server instances that you intend to include.
3. Configure Endpoints.

   Endpoints are the point of entry into a SQL Server database. You use them much like you would use a firewall. Endpoints ensure that traffic is authorized to connect to a database instance over the internet. In SQL Server, you can use a wizard to configure endpoints in each of your database instances.
4. Specify a *Partner* and *Witness* server instance.

   Once you have specified partners, mirroring begins immediately.

For a full explanation of how to perform each of these steps, see *Setting Up Database Mirroring (SQL Server)* on MSDN.

3.8.3 Configuring Database Mirroring in Sitecore

When you set up SQL Server database mirroring with Sitecore you perform most of the configuration steps in your SQL Server instances. However, to ensure that Sitecore and SQL Server communicate correctly with each other and the *Mirror* database, there are several steps you need to follow in Sitecore.

Enabling the SQL Server Retryer in the Web Config

The RETRYER section of the *web.config* file handles requests to the *Mirror* database. When you install Sitecore, this is included in the *web.config* file by default.

```
<!-- RETRYER
           When enabled, the Retryer retries database requests a specified number of times (disab
led by default).
           For example, this is useful if you configure an instance to support hot failover for d
atabase calls.
      -->
    <retryer disabled="true" type="Sitecore.Data.DataProviders.Retryer, Sitecore.Kernel">
      <param desc="Number of tries">6</param>
      <param desc="Interval between tries">00:00:00.500</param>
      <param desc="Log each exception (should be used for debug only)">true</param>
    </retryer>
```

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

You must enable the RETRYER section to start database mirroring (by default this is disabled).

To enable database mirroring, change the disabled setting to *false*.

```
<retryer disabled="false" type="Sitecore.Data.DataProviders.Retryer, Sitecore.Kernel">
```

Save your changes to the *web.config* file.

Note
You may need to edit some of the other parameters in the RETRYER section to enable better performance. However, be cautious if you decide to change any of these settings. For best performance, we recommend that you leave all default parameters unchanged.

Enabling Mirroring in the Connection String Config File

To enable database mirroring and failover in your Sitecore solution, you need to update the connection string config file by adding a reference to the database that you want to use as a *Mirror*. To do this, use the *Failover Partner* parameter:

```
Failover Partner=<Name of failover Mirror instance>
```

For example, the *Master* database connection string:

```
<add name="master" connectionString="user id=sa;password=******;Data Source=10.38.10.224;Failover Partner=10.38.10.213;Database=Sitecore_Master"/>
```

Add this parameter to the connection string for each of the databases that you want to mirror. In a standard Sitecore solution there are three databases; *Master*, *Core* and *Web*. Add a failover parameter for each of these databases.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

Chapter 4

Configuration Tasks

This chapter describes some of the tasks that you must perform when you configure a multi-instance installation.

This chapter contains the following sections:

- How to: Enable Scalability Settings
- How to: Disable or Enable File Based Media
- How to: Configure the Name of the Publishing Instance
- How to: Configure HTML cache clearing
- How to: Configure a Machine Key
- How to: Remove References to the Master Database
- How to: Restrict Access to Client Interfaces
- How to: Configure a Remote Publishing Target
- How to: Configure SQL Server Replication
- How to: Address Security Caching Clearing
- How to: Configure DFS
- How to: Configure Web Deployment

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.1 How to: Enable Scalability Settings

To enable the scalability settings for a Sitecore CMS application:

- Rename the `/App_Config/Include/ScalabilitySettings.config.example` file to `/App_Config/Include/ScalabilitySettings.config` (remove the `.example` extension).

Note
The values in the `/App_Config/Include/ScalabilitySettings.config` file or any other `/App_Config/Include/*.config` file override the values in the `web.config`.

You can also enable the scalability settings by editing the `web.config` file and applying the list of configuration changes that the `ScalabilitySettings.config` file contains. However, we do not recommend this approach as it is error prone.

The `Web.config` file sets value of the `EnableEventQueues` setting to true, which is a required step to configure Sitecore to record and process remote events, perform automatic cache synchronizing between multiple instances and run distributed publishing.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.2 How to: Disable or Enable File Based Media

To disable or enable file based media for a Sitecore CMS application:

- In the `/App_Config/Include/ScalabilitySettings.confg` file, set the value of the `Media.DisableFileMedia` setting to the corresponding value.

Note
Changing the value of this setting controls whether or not users can choose between file system storage and database blob storage when they upload new media, or whether or not they can upload media to database blobs.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.3 How to: Configure the Name of the Publishing Instance

In a single instance CM environment, you don't need to make any changes.

If the CM environment contains two or more instances:

- Choose a CM instance that will act as a publishing instance.

  The publishing CM instance can still be an authoring instance, or it can also be used as a dedicated instance that is solely responsible for publishing. The latter option can be used to separate the publishing from the authoring CM instances for performance reasons, deployment strategy, or for other considerations. Publish operations can be initiated from any CM instance in the environment. They are queued, and then the publishing instance picks them up as a sequence.

- In the `/App_Config/Include/ScalabilitySettings.confg` file, set the value of the `Publishing.PublishingInstance` setting to the name of the publishing instance. You must configure the `Publishing.PublishingInstance` to the same value on all the CM instances in the environment.

The name of the Sitecore CMS application instance is defined in the `web.config` file by the value of the `InstanceName` setting. It can also be defined in a `.config` file in the `/App_Config/Include` subdirectory, such as `/App_Config/Include/ScalabilitySettings.config`.

The `InstanceName` setting is empty by default, in which case its effective value is the concatenation of the NetBIOS machine name and the name of the IIS Web site, separated by a hyphen character ("-"). For example, the value of the `InstanceName` setting for a Website named `ProjectName` on a machine named `CM-SERVER-A` is `CM-SERVER-A-ProjectName`.

If Sitecore cannot determine the name of the IIS Web site, then the default value of the `InstanceName` setting is the machine name.

If Sitecore cannot determine the machine name, Sitecore raises an error.

You can update the `InstanceName` setting to control the instance names rather than using names that are generated automatically. This is useful if you wish to shorten the instance names, or if Sitecore cannot determine the NetBIOS machine name or the Web site name.

You can check the currently configured instance name of the Sitecore application by looking for an entry in the Sitecore log file that looks like this:

```
5224 12:00:00 INFO Instance Name: CM-SERVER-A-ProjectName
```

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.4 How to: Configure HTML cache clearing

By default, Sitecore is configured to automatically clear the HTML cache for the managed site with the name "website" right after the publishing operation ends.

For more information about Sitecore HTML Caches, see the section *Output Caching* in the *Presentation Component Reference Guide*.

For more information about multiple managed websites, see the article *Configuring Multiple Sites* on the SDN.

In addition to the default configuration, you must manually configure HTML cache clearing if all of the following three conditions are met:

- You have configured the output of one or more renderings or sublayouts to be cached.
- Your solution contains one or more custom site definitions — that is, other sites than the managed site with the name "website" — where these renderings/sublayouts are used.
- Your custom site definitions in web.config have HTML caching enabled and HTML cache size set to a positive value.

For more information about enabling and configuring the size of the HTML cache for managed Sitecore sites, see the manual *Cache Configuration Reference*.

In other words: You can skip the rest of this section if you do not cache the output of any renderings/sublayouts or if you do cache the output of renderings/sublayouts but your solution does not define any custom sites.

There are two way to configure automatic HTML cache clearing for your custom sites: You can configure the system to clear HTML caches periodically using a scheduled agent or right after the publishing operation ends.

To configure HTML cache clearing after publishing operation:

- In `web.config` file, to the `/events/event[@name="publish:end"]/handler/sites` node, add another `site` node with the name of the site to clear HTML caches for.

- In `web.config` file, to the `/events/event[@name="publish:end:remote"]/handler/sites` node, add another `site` node with the name of the site to clear HTML caches for.

For example, if the custom site name is `mysite`, related configuration section may look like:

```
      <event name="publish:end">
        <handler type="Sitecore.Publishing.HtmlCacheClearer, Sitecore.Kernel"
method="ClearCache">
          <sites hint="list">
            <site>website</site>
            <site>mysite</site>
          </sites>
        </handler>
      </event>
      <event name="publish:end:remote">
        <handler type="Sitecore.Publishing.HtmlCacheClearer, Sitecore.Kernel"
method="ClearCache">
          <sites hint="list">
            <site>website</site>
            <site>mysite</site>
          </sites>
        </handler>
      </event>
```

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

To configure HTML cache clearing using a scheduled agent:

- In `web.config` file, for the `/scheduling/agent[@name="Sitecore.Tasks.HtmlCacheClearAgent"]` node, set the value of `interval` attribute to the required value.

For example, if HTML caches are required to be cleared every 5 minutes, related configuration section may look like:

```
<agent type="Sitecore.Tasks.HtmlCacheClearAgent" method="Run" interval="00:05:00" />
```

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.5 How to: Configure a Machine Key

In a single instance environment, you shouldn't change the configuration.

If the environment contains two or more instances:

- In the `web.config` file, set the `validationKey` and the `decryptionKey` attributes of the `/configuration/system.web/machineKey` element to a non-auto generated value. Ensure that the values are identical for all the instances within the environment and that the `IsolateApps` modifier is not present in either value.

For more information about the `<machineKey>` section of the `web.config` file, see http://msdn.microsoft.com/en-us/library/w8h3skw9(VS.90).aspx.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.6 How to: Remove References to the Master Database

There are two ways to remove references to the Master database:

- Use the Include file.
- Remove them manually.

Using an Include File to Remove References to Master Database

You can use an include file to remove any references to the Master database on a CD Sitecore instance when you are running Sitecore CMS 6.4 and later:

- Copy the `SwitchMasterToWeb.config` file that you can download from http://sdn.sitecore.net/reference/sitecore%207/scaling%20guide.aspx to the `/App_Config/Include` folder.

Note
The filename is important. Because of the way Sitecore CMS processes include files, the `SwitchMasterToWeb.config` file name is listed alphabetically after the `Sitecore.WebDAV.config` file.

In the `/App_Config/ConnectionStrings.config` or in the `/App_Config/ConnectionStringsOracle.config` file, comment out the `master` connection string.

Manually Removing References to the Master Database

As an alternative to the instructions in the previous section or if you are running on Sitecore CMS 6.3, you can manually remove references to the Master database.

To manually remove references to the Master database:

In the `web.config` file:

- Change the `content` attribute of the `/sites/site[@name="shell"]` node to `web`.
- Change the `content` attribute of the `/sites/site[@name="modules_shell"]` node to `web`.
- If it is present, comment out the `/sites/site[@name='testing']` node.
- Change the `connectionStringName` attribute of the `/IDTable/param[@connectionStringName="master"]` node to `web`.
- Comment out the `/databases/database[@id='master']` node.
- Comment out the `/search/configuration/indexes/index/locations/master` node.
- Comment out the `Sitecore.Tasks.DatabaseAgent` that has its `database` parameters set to `master`.

In the `/App_Config/Include/Sitecore.WebDAV.config` file:

- Comment out the `/scheduling/agent[@type="Sitecore.Tasks.CleanupFDAObsoleteMediaData"] /databases/database/[@name='master']` node.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

In the `/App_Config/ConnectionStrings.config` or in the `/App_Config/ConnectionStringsOracle.config` file:

- Comment out the `master` connection string.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.7 How to: Restrict Access to Client Interfaces

To prevent unauthorized access to the Sitecore Client interfaces, you must restrict access to them on every instance of the CD environment.

To restrict access to client interfaces, we recommend that you implement IP-based security restrictions or disable Anonymous IIS access to the `/sitecore/admin`, `/sitecore/login`, `/sitecore/shell` folders and the `/sitecore/default.aspx` page.

The `/sitecore/service` folder should be excluded from the IIS restrictions because it contains a number of service ASPX pages that are used by the Sitecore application when reporting various conditions that may occur in the application, such as, 404 Page Not Found or 403 Forbidden, to the web client.

You can move files from the `/sitecore/service` folder outside the `/sitecore` folder, and then you must also update the following settings in the `web.config` file:

```
ErrorPage
NoAccessUrl
NoLicenseUrl
LayoutNotFoundUrl
ItemNotFoundUrl
LinkItemNotFoundUrl
```

For more information about configuring IP-based security restrictions in IIS 7 and later, see http://www.iis.net/ConfigReference/system.webServer/security/ipSecurity.

For more information about configuring IP-based security restrictions in IIS 6 and earlier, see http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/128d26dd-decb-42f9-8efb-30724d1a2f29.mspx?mfr=true.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.8 How to: Configure a Remote Publishing Target

To configure a remote publishing target, you must:

- Create a publishing target database.
- Register the publishing target database.

Creating a Publishing Target Database

To create a publishing target database, on the database server:

- Create the database from an archive of the default `web` publishing target in the Sitecore `.zip` distributive.

or

- Duplicate an existing publishing target database.

Registering a Publishing Target Database

To register the publishing target database, on each of the CM instances:

1. In the `web.config` file, duplicate the `/configuration/sitecore/databases/database` element with an `id` of `web`, and then change the `id` attribute in the new `<database>` element to the name of the publishing target database connection string name, such as `pub`.
2. In the `/App_Config/ConnectionStrings.config` or in the `/App_Config/ConnectionStringsOracle.config` file, duplicate the `/connectionStrings/add` element named `web`, change the `name` attribute from `web` to the connection string name, such as `pub`, and then change the `connectionString` property to access the publishing target database.
3. On one of the CM instances, in the **Content Editor,** select the `/Sitecore/System/Publishing targets` item.
4. In the **Content Editor**, insert a publishing target definition item that is named after the database and is based on the `System/Publishing target` data template.
5. In the **Content Editor**, in the new publishing target definition item, in the `Data` section, in the `Target database` field, enter the name of the publishing target database, such as `pub`.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.9 How to: Configure SQL Server Replication

For more information about how to configure SQL server replication for Sitecore databases, see the SQL Server Replication Guide.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.10 How to: Address Security Caching Clearing

There are two ways to address security caching:

- Security caches expiration
- Forwarding of security events

After you implement the instructions in the *How to: Enable Scalability Settings* section, security caches expiration is automatically enabled. The only disadvantage of this approach comparing to the forwarding of security events is that the security caches can remain outdated for longer periods of time.

Forwarding of security events allows security caches to be updated automatically right after the relevant security events occur. However, this approach requires one environment (CM or CD) to write to another environment's (CD or CM) remote database whenever the relevant events occur. This is only recommended for setups where instant updates of security caches are critical.

To configure security cache expiration:

- On each instance of the environment that does not perform security data updates, but needs to read them (typically CD), set the value of the `Caching.SecurityCacheExpiration` setting in the `/App_Config/Include/ScalabilitySettings.config` or `web.config` file to an appropriate value, such as `00:20:00`.

To configure the forwarding of security events:

1. On each instance of the environment that performs security data updates (typically CM), in the `/App_Config/Include` subdirectory, rename the file named `ForwardingSecurityEvents.config.example` to `ForwardingSecurityEvents.config` and remove the `.example` extension.
2. In the `/App_Config/Include/ForwardingSecurityEvents.config` file, within the `/sitecore/pipelines/initialize/processor/domains` element, add an element named after the domain that requires synchronization. Use the name of the domain as the text of the new element.

   For example, to forward security events for a domain named `domainname`, add the `<domainname>` element as shown here:

```
...
<domains hint="list">
  <extranet>extranet</extranet>
  <domainname>domainname</domainname>
</domains>
...
```

3. In the `/App_Config/Include/ForwardingSecurityEvents.config` file, within the `/sitecore/pipelines/initialize/processor/targets` element, add an element named after the remote database from the CD environment. Use the name of the database as the text of the new element.

   In a typical setup, to forward security events to the database named `pub`, change the `<web>` element to `<pub>` as shown here:

```
...
<targets hint="list">
  <pub>pub</pub>
</targets>
```

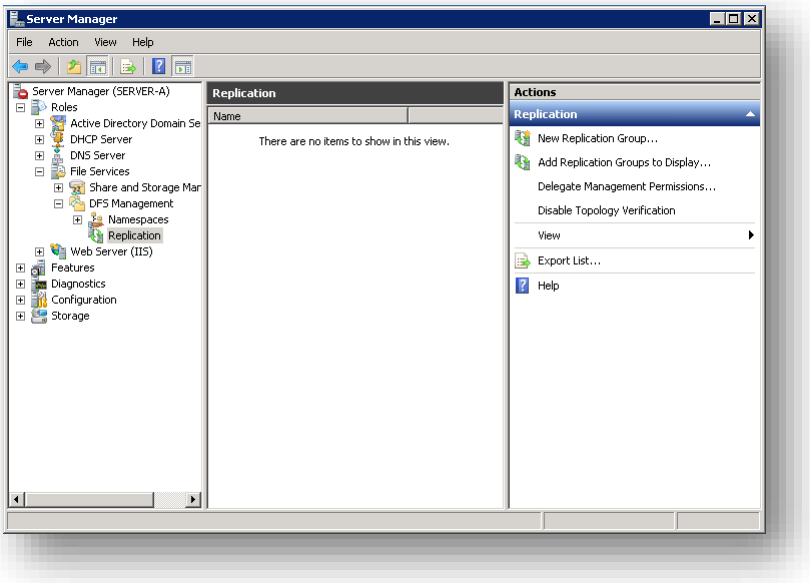
4. In the `/App_Config/Include/ForwardingSecurityEvents.config` file, remove the `/sitecore/databases` element.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.11 How to: Configure DFS

To configure DFS:

1. Designate one of the servers in the group as the primary server. Replication begins initially from the primary server, but subsequent file system events on any server in the group replicate to the others.
2. In Microsoft Windows, on the server hosting the primary instance, start the **Server Manager**.
3. In the **Server Manager**, expand **Roles**, **File Services**, **DFS Management**, and then select **Replication**.



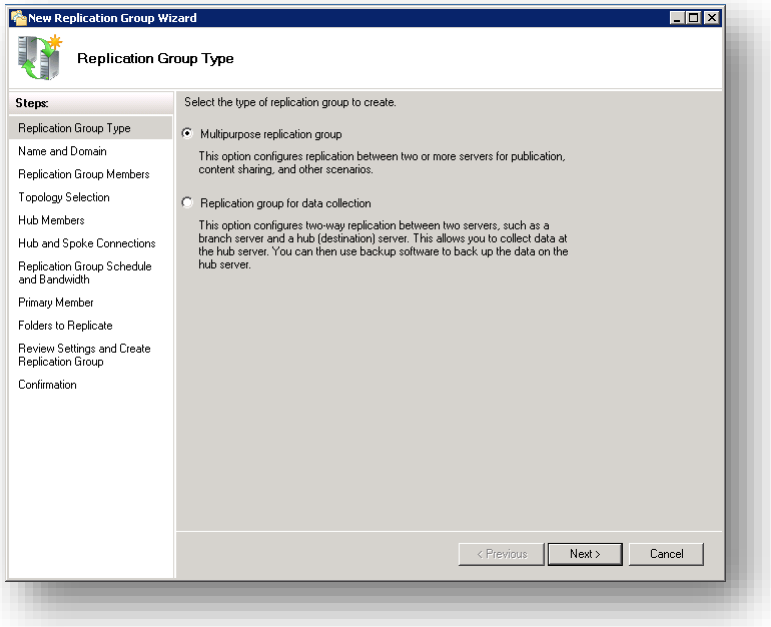
4. In the **Actions** panel, click **New Replication Group...** and the **New Replication Group** wizard appears.

You use the New Replication Group wizard to configure the replication group.

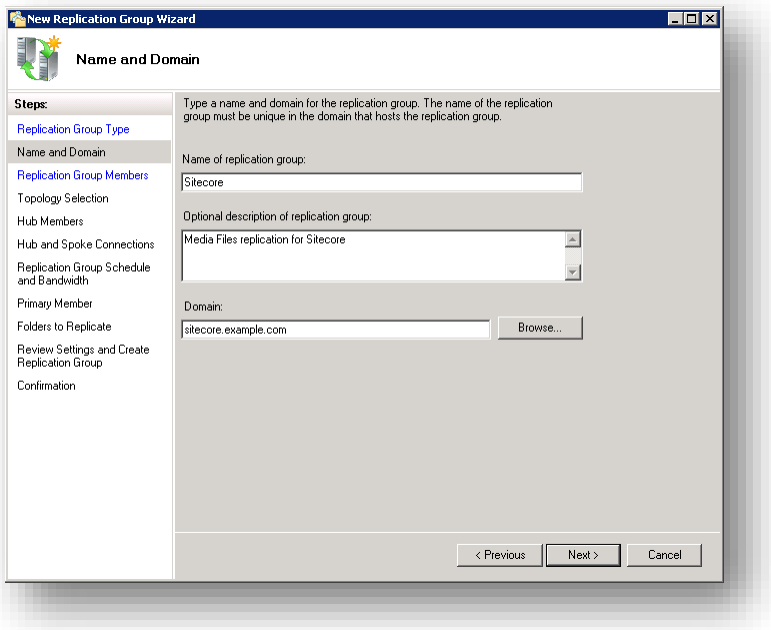
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

To configure the replication group:

1. In the **New Replication Group** wizard, in the **Replication Group Type** window, select the **Multipurpose replication group** option.



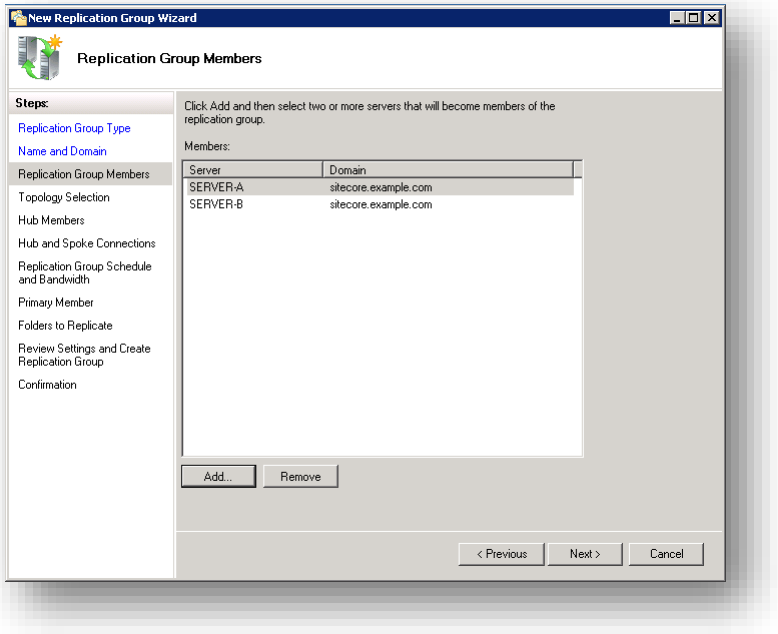
2. Click **Next** and the **Name and Domain** window appears.



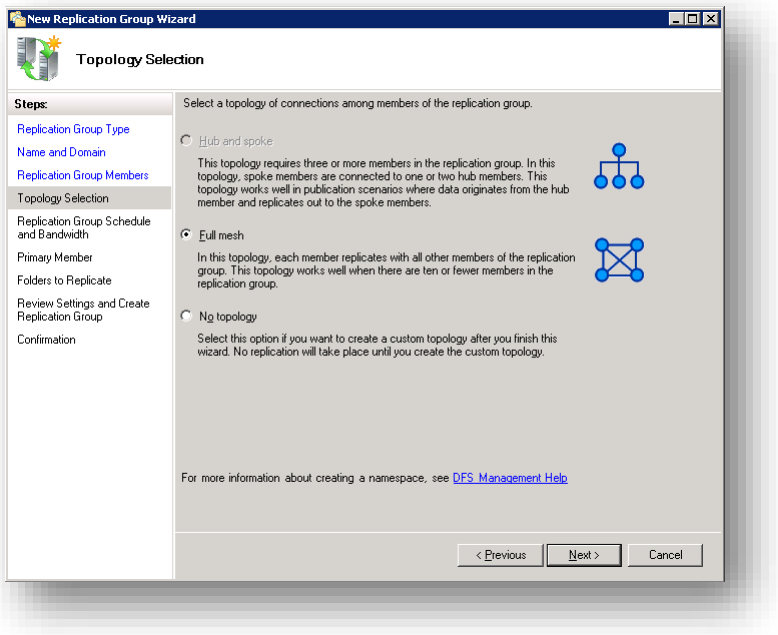
3. In the **Name and Domain** window, in the **Name of replication group** field, enter a name
4. In the **Optional description of replication group** field, enter a description of the environment and its purpose, such as *CM Production File Media*.
5. In the **Domain** field, click **Browse** and select the appropriate domain.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

6. Click **Next** and the **Replication Group Members** window appears.



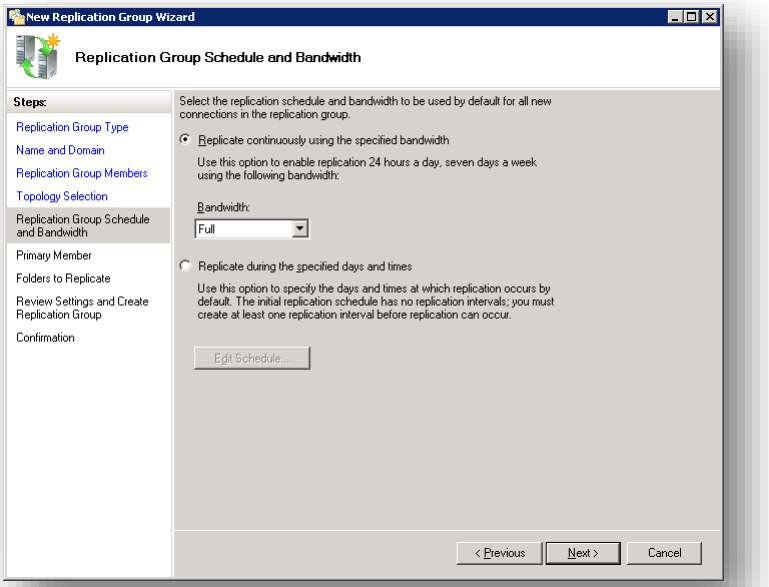
7. In the **Replication Group Members** window, click **Add** and add the servers that will participate in replication.
8. Click **Next** and the **Topology Selection** window appears.



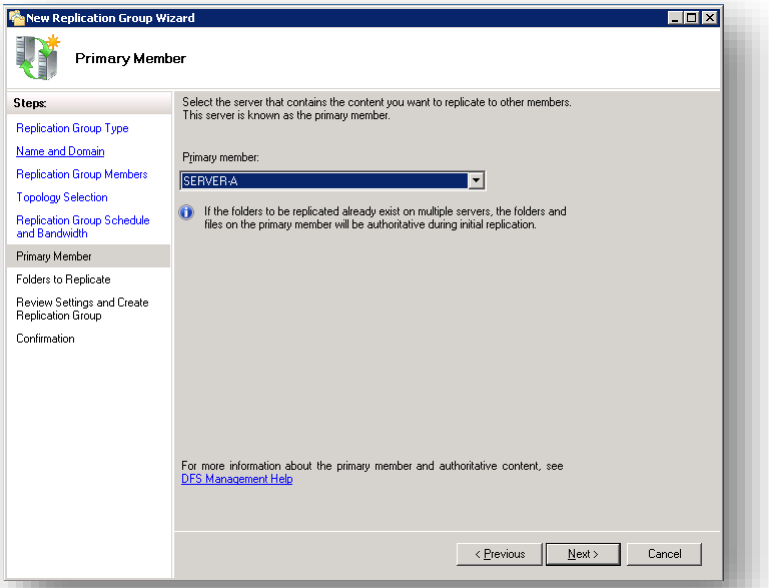
9. In the **Topology Selection** window, select the **Full mesh** option.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

10. Click **Next** and the **Replication Group Schedule** window appears.



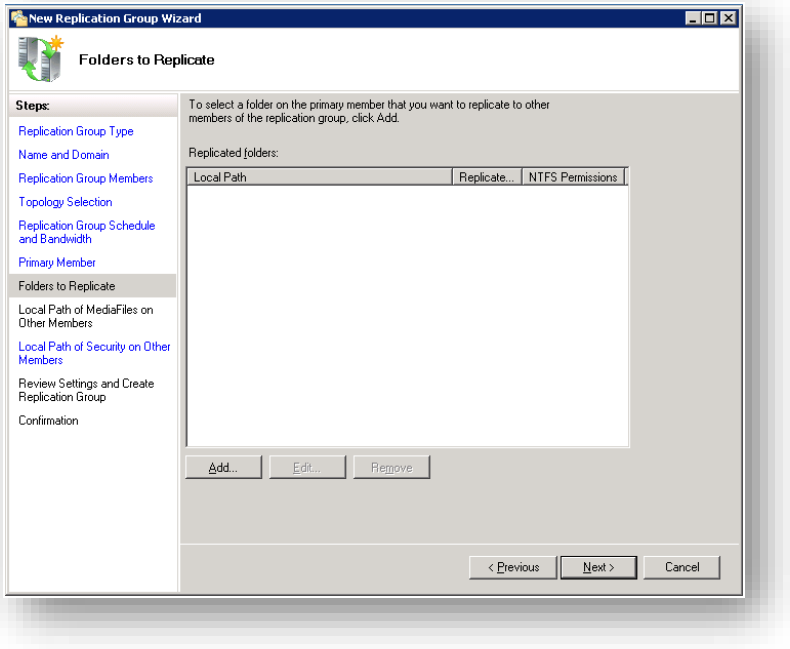
11. In the **Replication Group Schedule and Bandwidth** window, select the **Replicate continuously using the specified bandwidth** option, and in the **Bandwidth** field, select *Full*.
12. Click **Next** and the **Primary Member** window appears.



13. In the **Primary Member** window, in the **Primary member** field, select the primary member server.

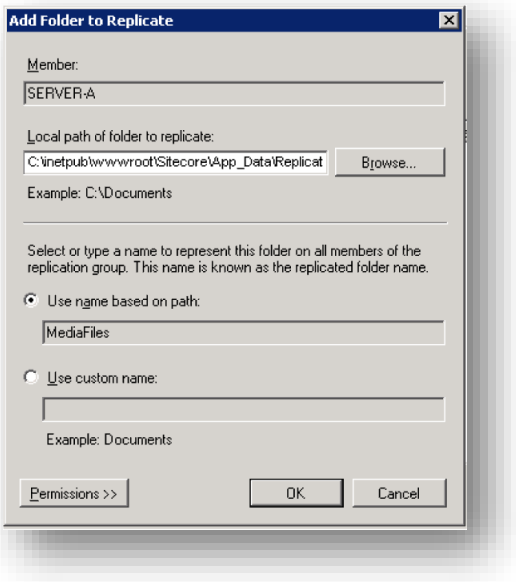
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

14. Click **Next** and the **Folders to Replicate** window appears.



You use this window to select the subdirectories that you want to replicate.

15. In the **Folders to Replicate** window, click **Add** and the **Add Folder to Replicate** dialog box appears.



16. In the **Add Folder to Replicate** dialog box, in the **Local path of folder to replicate** field, click **Browse**, and select the subdirectory that you want to replicate, and then click **OK**.

    To configure the replication of file media, select the parent directory specified by the `Media.FileFolder` setting in the `/App_Config/Include/ScalabilitySettings.config` or `web.config` file.

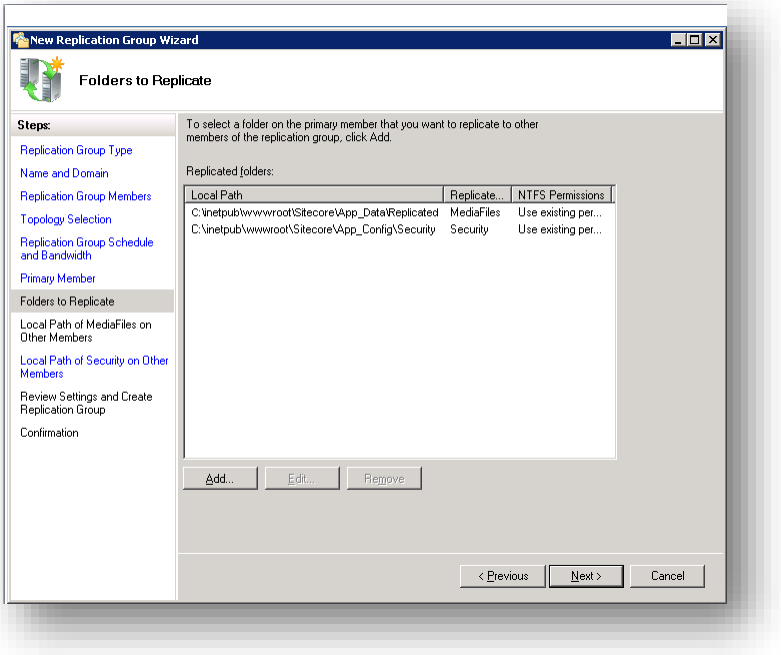
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

For example, if the `Media.FileFolder` setting is set to `/App_Data/Replicated/MediaFiles`, you must select the `/App_Data/Replicated` folder — `C:\inetpub\sitecore\projectName\website\App_Data\Replicated`.

To synchronize the security domain configuration file, select the subdirectory containing the security domain configuration file as specified by the `arg0` attribute in the `/sitecore/configuration/configStores/add` element named `domains` in the `web.config` file, such as `/App_Config/Security` (`C:\inetpub\sitecore\wwwroot\Sitecore\App_Config\Security`).

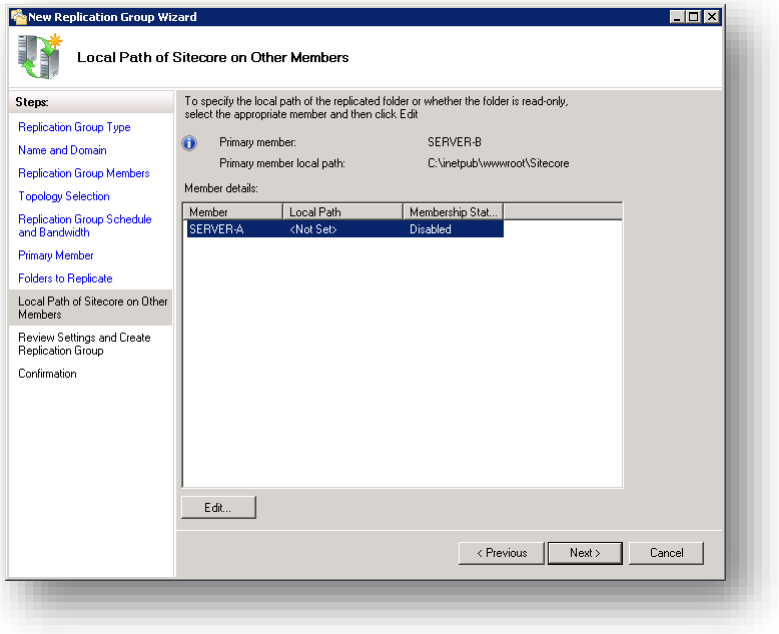
Warning
Do not configure DFS for subdirectories other than those specified in this section.

The **Folders to Replicate** window should look something like this:



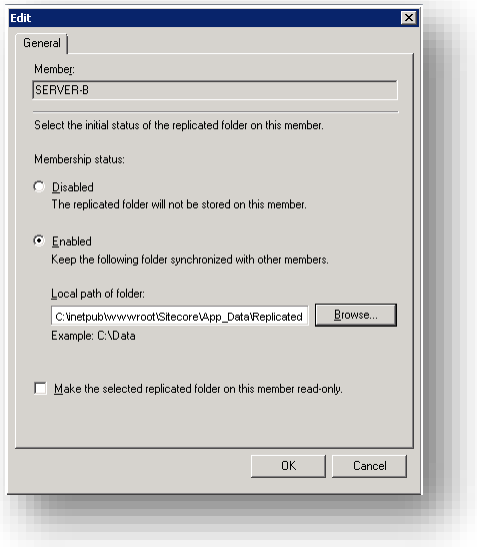
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

17. Click **Next** and the **Local Path** window appears.



Configure the target path for each replicated subdirectory on each server in the replication group.

18. In the **Local Path** window, select each server and then click **Edit**. The **Edit** dialog box appears.



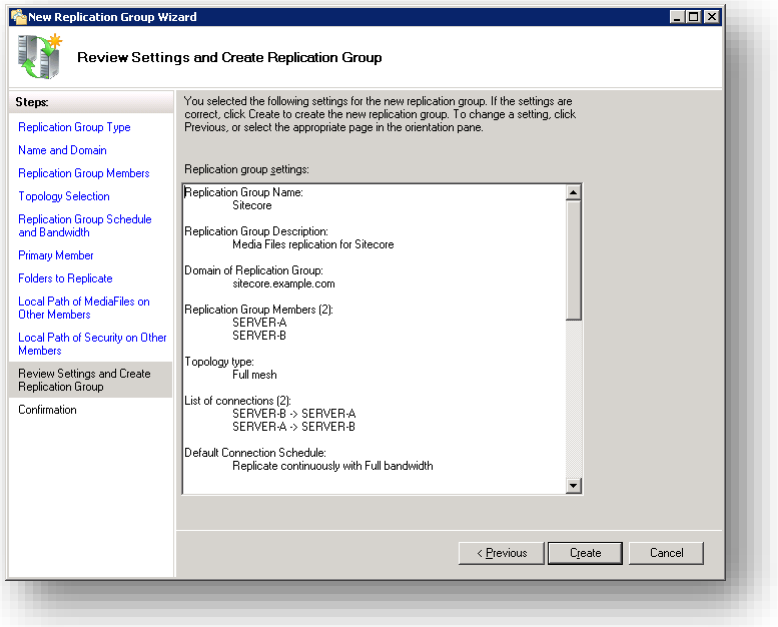
19. In the **Edit** dialog, select the **Enabled** option, and then in the **Local path of folder** field click **Browse** and select the appropriate local path, and then click **OK**.

    To configure replication of file media, select the parent directory of the directory specified by the `Media.FileFolder` setting in the `/App_Config/Include/ScalabilitySettings.config` or `web.config` file of the target instance, such as `/App_Data/Replicated/MediaFiles` (`C:\inetpub\sitecore\wwwroot\Sitecore\App_Data\Replicated\MediaFiles`).

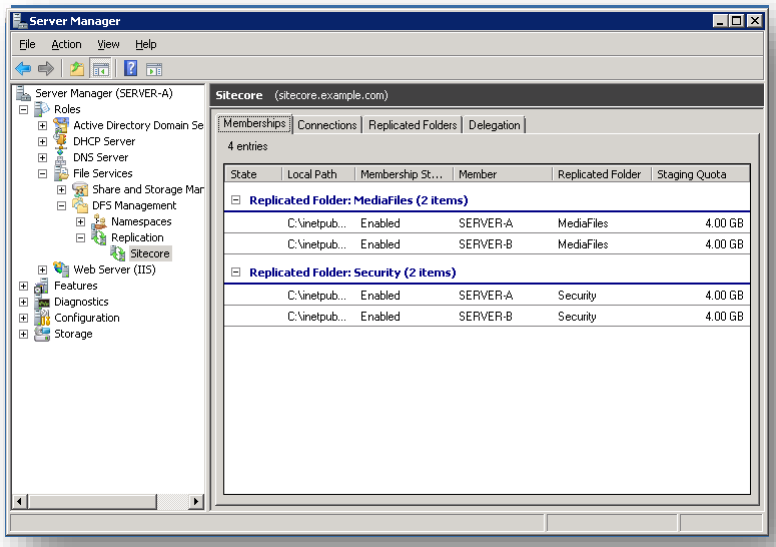
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

To synchronize the security domain configuration file, select the subdirectory containing the security domain configuration file as specified by the `arg0` attribute in the `/sitecore/configuration/configStores/add` element named `domains` in the `web.config` file of the CD Web Deployment target instance, such as `/App_Config/Security` (`C:\inetpub\sitecore\projectName\website\App_Config\Security`).

20. In the **Local Path** window, click **Next** and the **Review Settings and Create Replication Group** window appears.

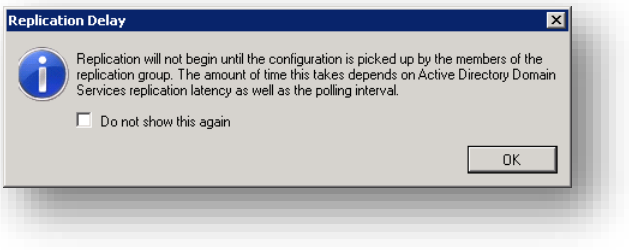


21. Ensure that all of the settings you specified are correct and the click **Create**.
22. In the **Confirmation** window, click **Close**.



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

Depending on latency in applying the Windows domain group policy, replication may not begin immediately, and you may see a message like this:



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

4.12 How to: Configure Web Deployment

To configure Web Deployment of file media from CM to CD, on the publishing instance or all CM instances:

1. In the `/App_Config/Include` subdirectory, rename the `WebDeploy.config.example` file to `WebDeploy.config` (remove the `.example` extension).
2. In the `/App_Config/Include/WebDeploy.config` file, configure parameters for the `/configuration/sitecore/events/event/handler` element as follows:

Element	Purpose
`<TargetDatabase>`	The event handler invokes Web Deployment when you publish to the identified publishing target database. `TargetDatabase` matches the `name` attribute of a `/configuration/add` element in the `/App_Config/ConnectionStrings.config` file.
`<TargetServer>`	The IP address or DNS name of the target server that will receive the files, or the URL of the Web Deployment Agent service on the target server. For more information, see the Web Deployment Tool Installation article.
`<UserName>`	The Windows domain and username of an account with permissions to modify the required file systems on the remote server, such as `DOMAIN\username` (you can omit the domain for local Windows accounts). **Note** Because of a bug in Web Deploy 2.0, the user must be either the built-in Administrator or a member of the Domain Administrators security group. The Sitecore reference number for the issue is 351405. For more information about this requirement, see http://learn.iis.net/page.aspx/1023/web-deploy-error-codes/.
`<Password>`	The password of the user specified by the `<UserName>` parameter described previously.
`<LocalRoot>`	The full path to the subdirectory containing files to deploy. If omitted, the effective value of the `<LocalRoot>` element is the document root of the source Web site.
`<RemoteRoot>`	The full path to the subdirectory to receive deployed files on the remote server. If omitted, the effective value of the `<RemoteRoot>` element is the document root of the target Web site.
`<items/*>`	One or more local subdirectories or files to deploy, each configured with a separate element within the `<items>` element, relative to the subdirectory specified by the `<LocalRoot>` element. Subdirectories and files deploy to the corresponding locations on the target server relative to the value specified by the `<RemoteRoot>` element. The names of elements within the `<items>` element are arbitrary.

Sitecore uses a `publish:begin` event handler to initiate Web Deployment. Because CD instances do not publish, this event handler is irrelevant in CD instances. In load-balanced CM environments, only the CM publishing instance invokes Web Deployment; the event handler that initiates Web Deployment is irrelevant to other CM instances. This event handler has no effect in CD environments and on all but the publishing instance in CM environments.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.

By default, the `publish:begin` event handler that invokes Web Deployment is asynchronous, using one or more background threads. This can cause content to appear in the publishing target database before deployment of file media referenced by that content completes. To cause the handler to operate synchronously, causing deployment of file media to complete before content publishes, create an element named `<Synchronous>` with a value of `true` within the `/configuration/sitecore/events/event/handler` element in the `/App_Config/Include/WebDeploy.config` file. For example:

```
        <configuration>
          <sitecore>
            <events>
              <event name="publish:begin">
                <handler type="Sitecore.Publishing.WebDeploy.PublishHandler,
                  Sitecore.Publishing.WebDeploy" method="OnPublish">
                  <synchronous>true</synchronous>
                  <tasks hint="list:AddTask">
        ...
```

How to: Configure Multiple Web Deployment Targets

You can use Web Deployment to deploy files from a single CM instance to multiple target instances in the CD environment.

To configure multiple deployment targets:

1. In the `/App_Config/Include/WebDeploy.config` file, duplicate an existing `/configuration/sitecore/events/event/handler/tasks/*` element.
2. In the `/App_Config/Include/WebDeploy.config`, in the new element, update parameters as described in the table provided in the previous section.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2014 Sitecore. All rights reserved.