CHAPTER 6

Migrating IIS Web Sites to IIS 6.0

You can move Web sites and applications to Internet Information Services (IIS) 6.0 in two ways. *Migrating* is installing the Microsoft® Windows® Server 2003 operating system and IIS 6.0 on a new server and then moving, or reinstalling, existing Web sites and applications to that server. *Upgrading* is installing Windows Server 2003 and IIS 6.0 on an existing server running the Microsoft Windows NT® Server 4.0 operating system and IIS 4.0 or the Microsoft Windows® 2000 Server operating system and IIS 5.0. Both processes involve minimal outage of service to users who access the Web sites and applications, while retaining the majority of the original configuration settings and fully preserving the content of the Web sites and applications.

In This Chapter

Overview of Migrating IIS Web Sites to IIS 6.0 182
Preparing for Migration 188
Deploying the Target Server 205
Migrating Web Sites with the IIS 6.0 Migration Tool 208
Migrating Web Sites Manually 217
Configuring IIS 6.0 Properties 226
Performing Application-Specific Migration Tasks 238
Completing the Migration 246
Additional Resources 250

Related Information

For information about securing IIS Web sites, see "Securing Web Sites and Applications" in this book.

For information about upgrading from IIS 4.0 or IIS 5.0 to IIS 6.0, see "Upgrading an IIS Server to IIS 6.0" in this book.

For information about migrating Apache Web sites, see "Migrating Apache Web Sites to IIS 6.0" in this book.

Overview of Migrating IIS Web Sites to IIS 6.0

You begin the migration process by determining the compatibility of your existing Web sites and applications with IIS 6.0 and the Microsoft® Windows® Server 2003, Standard Edition; Windows® Server 2003, Enterprise Edition; Windows® Server 2003, Datacenter Edition; or Windows® Server 2003, Web Edition operating system. Next, you install Windows Server 2003 and IIS 6.0 on the *target server*, which is the server that will host your Web sites after migration. Then, you migrate the Web site content and configuration settings from the *source server*, which is a server running the Microsoft Windows NT® Server 4.0 operating system and IIS 4.0 or the Windows 2000 Server operating system and IIS 5.0, to the target server.

After the migration of your Web site content, you customize the configuration of IIS 6.0, based on your Web sites and applications. Finally, after you have completed the customization of IIS 6.0, you back up the target server, enable client access to the Web sites and applications on the target server, and decommission the source server.

The process in this chapter focuses on transferring the Web site content and configuration settings, and not on the details of how to make application code changes in dynamic content. If your Web sites contain only static content, you can most likely complete the migration process in a few simple steps. However, if your IIS Web sites contain dynamic content, such as Active Server Pages (ASP), ASP.NET, or Common Gateway Interface (CGI) scripts, you might need to modify the code in the dynamic content separately. In addition, any provisioning scripts or setup programs for your existing Web sites and applications might need to be modified after the migration process is complete. Ensure that you test any modifications after the migration process is complete. For more information about potential modifications, see “Preparing for Migration” later in this chapter.

Also, if the existing Web sites and applications depend on software other than the Windows operating system and IIS, the complexity of the migration process increases. For example, the process for migrating a Web server that hosts Web sites and applications that were designed to run on Windows 2000 Server and IIS 5.0 is relatively simple. On the other hand, the process for migrating a Web server that hosts Web sites, applications, and other software — such as Microsoft Commerce Server, Microsoft BizTalk® Server, monitoring software, custom applications, or other non-Microsoft software — is more difficult because all of the software must be compatible with Windows Server 2003 and IIS 6.0.

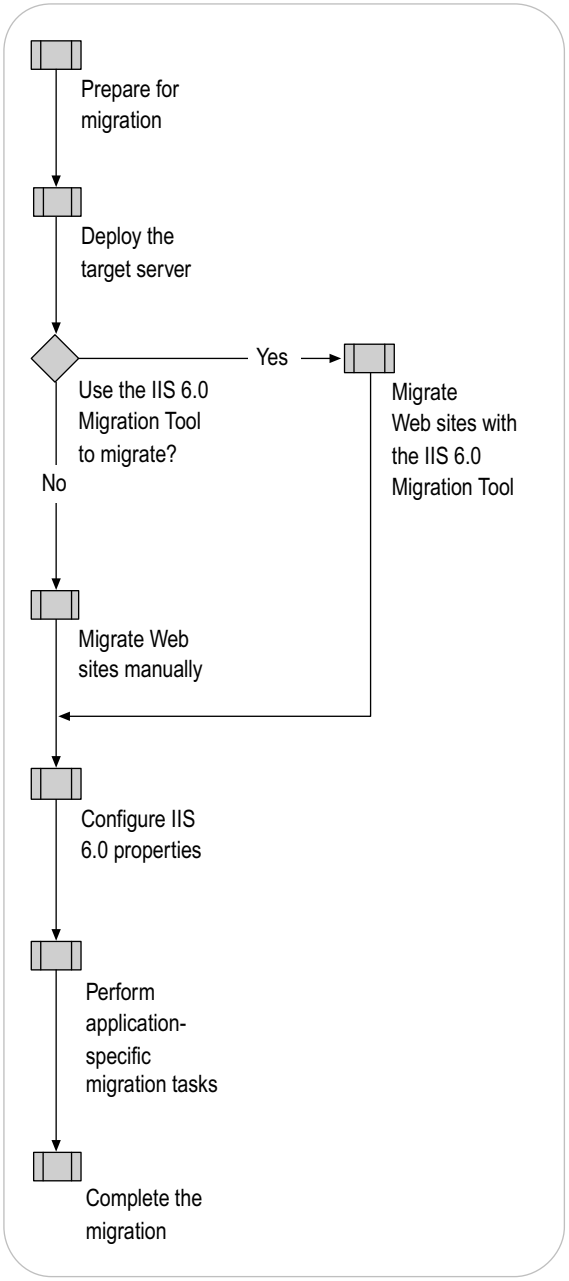
In some cases, you can simplify the Web site migration process by using the IIS 6.0 Migration Tool. For more information about using the IIS 6.0 Migration Tool, see “Preparing for Migration” later in this chapter.

Process for Migrating IIS Web Sites to IIS 6.0

The process for migrating Web sites hosted on IIS consists of preparing for and performing the migration. Before performing the migration, you need to evaluate the compatibility of the software installed on your existing Web server (including software that generates dynamic content, database connections, and any non-Microsoft software) with Windows Server 2003 and IIS 6.0. You can then perform the migration manually or with the IIS 6.0 Migration tool. After the migration is complete, you must further customize the configuration of IIS 6.0.

Figure 6.1 illustrates the process for migrating existing IIS Web sites to IIS 6.0.

Figure 6.1 Migrating IIS Web Sites to IIS 6.0



Tip

To migrate a Web farm, use the process described in this chapter to migrate Web sites from each Web server in the source Web farm. Then, use provisioning or Web site staging software to propagate the migrated content and site configuration to other Web servers in the target Web farm.

Depending on your familiarity with Windows server operating systems, IIS, and the migration process, you might require less information to complete the IIS 6.0 migration process. To facilitate the fastest possible migration, the following quick-start guide is provided. You can use this guide to help identify the steps of the IIS 6.0 migration process that you need additional information to complete, and then you can skip the information with which you are already familiar. In addition, all of the procedures that are required to complete the IIS migration process are documented in "IIS Deployment Procedures" in this book.

Prepare for Migration

1. Identify which Web site and application components to migrate.
2. Determine compatibility with Windows Server 2003.
3. Determine application compatibility with worker process isolation mode by evaluating the following:
   - The benefits of worker process isolation mode
   - The application changes that are required for worker process isolation mode
   - The management and provisioning script changes that are required for worker process isolation mode
   - The results of lab tests that were completed to verify application compatibility with worker process isolation mode
4. Determine application compatibility with the Microsoft .NET Framework on Windows Server 2003.
5. Select one of two methods for migration:
   - Using the IIS 6.0 Migration Tool
   - Completing the migration process manually
6. If you are using the IIS 6.0 Migration Tool, identify the following:
   - Tasks that are automated by the migration tool
   - Subsequent tasks that must be performed manually

Deploy the Target Sever

1. Install Windows Server 2003.
2. Install and configure IIS 6.0.

Migrate Web Sites with the IIS 6.0 Migration Tool

1. Install the IIS 6.0 Migration Tool.
2. Verify that clients are not accessing Web sites.
3. Run the migration tool.
4. Verify that the migration tool ran successfully.
5. Migrate additional Web site content that is in the following two locations:
   - Outside the home directory of the Web site
   - Inside virtual directories
6. Modify IIS metabase properties that reference the systemroot folder.

Migrate Web Sites Manually

1. Verify that clients are not accessing the Web sites.
2. Create Web sites and virtual directories.
3. Migrate Web site content to the target server.
4. Configure Web site application isolation settings by completing the following tasks, if appropriate:
   - Document the current application isolation settings on the source server.
   - Configure application isolation settings in IIS 5.0 isolation mode.
   - Configure application isolation settings in worker process isolation mode.
5. Modify IIS 6.0 metabase properties that reference the systemroot folder.

Configure IIS 6.0 Properties

1. Configure IIS 6.0 properties that reference local user accounts.
2. Configure Web service extensions.
3. Configure Multipurpose Internet Mail Extensions (MIME) types.
4. Migrate server certificates for Secure Sockets Layer (SSL).
5. Migrate Microsoft FrontPage® users and roles.

6. Configure IIS 6.0 to host ASP.NET applications by completing the following tasks:
   - Configure IIS to use the correct version of the .NET Framework.
   - Configure the .NET Framework.
   - Review how ASP.NET applications run in each application isolation mode.
   - Migrate Machine.config attributes to their equivalent IIS 6.0 metabase property settings.
7. Determine whether to run the IIS Lockdown Tool and UrlScan.

Perform Application-Specific Migration Tasks

1. Modify application code for compatibility with Windows Server 2003 and IIS 6.0 by doing the following:
   - Modify references to Windows platform components and application programming interfaces (APIs) that are no longer supported in Windows Server 2003.
   - Modify references to IIS 6.0 metabase properties that have changed or are no longer supported in IIS 6.0.
   - Modify applications to be compatible with worker process isolation mode.
2. Install additional software required by applications.
3. Migrate Microsoft Transaction Server (MTS) packages, Component Object Model (COM) objects, and COM+ applications that are required by applications.
4. Modify data source names (DSNs) and Open Database Connectivity (ODBC) connections required by applications.
5. Create IP addresses that are used by applications.
6. Create users and groups that are used by applications.
7. Create registry entries required by applications on the target server.

Complete the Migration

1. Verify that the Web sites and applications migrated successfully.
2. Back up the target server.
3. Enable client access.

Preparing for Migration

Before migrating your existing IIS Web sites, ensure that the Web sites, applications, and their components are compatible with Windows Server 2003 and IIS 6.0. If you have setup programs or provisioning scripts that you are currently using on the source server and that you intend to continue using after migration, then ensure that the setup programs and provisioning scripts are compatible with Windows Server 2003 and IIS 6.0.

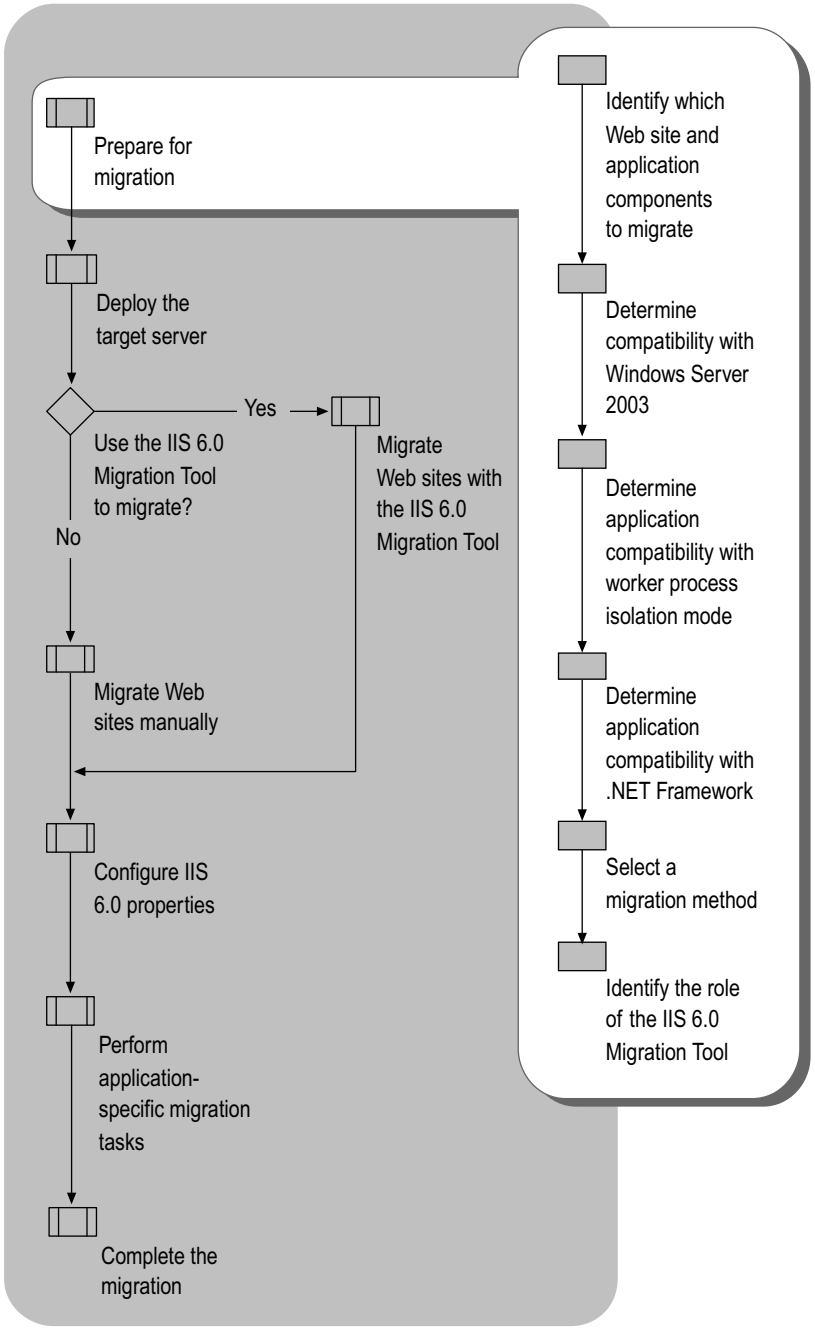
In addition, determine whether the existing Web sites and applications are compatible with worker process isolation mode in IIS 6.0. In cases where the existing Web sites and applications are not compatible with worker process isolation mode, you can still migrate to IIS 6.0 and Windows Server 2003 by configuring IIS to run in IIS 5.0 isolation mode. This allows the Web sites and applications to utilize other Windows Server 2003 and IIS 6.0 improvements

Also, if you are migrating ASP.NET applications, you must ensure that the ASP.NET applications are compatible with the latest version of the .NET Framework. If your ASP.NET applications require version 1.0 of the .NET Framework, you must configure IIS to run in IIS 5.0 isolation mode.

Lastly, you must determine whether you can perform the migration with the IIS 6.0 Migration Tool or if you need to perform the migration manually. You need to be aware of the additional migration steps that you must complete manually, regardless of the migration method that you choose. This allows you to have the appropriate tools and resources available when you are ready to perform the migration.

Figure 6.2 illustrates the process for preparing to migrate Web sites to IIS 6.0.

Figure 6.2 Preparing for Migration of Web Sites to IIS 6.0



Identifying Which Web Site and Application Components to Migrate

Before you begin the migration, identify the *components* that comprise the Web sites and applications. In addition to identifying the components, you must determine if there are any special circumstances associated with migrating these components. This Web site and application component migration is in addition to the Web site content and configuration that needs to also be migrated.

If you have setup, installation, or provisioning scripts for these Web sites and applications, you can use them to help you identify the components. If no setup, installation, or provisioning scripts exist, you must identify the Web site and application components manually.

Table 6.1 illustrates common application components that require a specific action when migrating to IIS 6.0.

Table 6.1 Migration Issues Associated with IIS Web Site Components

Web Site Component	Description	Migration Issues
MTS packages, COM objects, and COM+ applications	**MTS, COM, and COM+ provide object-oriented access to business logic and other data in n-tier applications.**	**MTS packages and COM+ applications need to be re-created in COM+ on the target server. COM objects that are registered on the source server need to be reregistered on the target server.** **In some cases, you might need to modify the MTS packages, COM objects, and COM+ applications, to make them compatible with Windows Server 2003 and IIS 6.0.**
Registry entries	**These entries are custom registry entries that are required by the application.**	**Applications might save custom configuration information in the Windows registry. The registry entries must be identified and then re-created on the target server.**
DSN data connection strings	**DSN data connection strings are used to provide connectivity to databases for IIS Web sites.**	**The DSNs might need to be modified on the target server for each of the Web sites and applications that access databases.**
ODBC data connections	**ODBC data connections are used to provide connectivity to databases for IIS Web sites.**	**The ODBC settings might need to be configured on the target server for each of the Web sites and applications that use ODBC to access databases.**

(continued)

Table 6.1 Migration Issues Associated with IIS Web Site Components *(continued)*

Web Site Component	Description	Migration Issues
ISAPI applications	ISAPI applications are DLLs that are called by Web sites and applications. There are ISAPI applications (ISAPI extensions) that are shipped with IIS 6.0, such as the ASP.NET ISAPI extension. In addition, your Web sites and applications might rely on ISAPI applications written by your organization.	The ISAPI application must be installed on the target server and then enabled from the Web services extensions node in IIS Manager. In some cases, you might need to modify the ISAPI application to make it compatible with IIS 6.0 and Windows Server 2003.
SSL Server Certificates	SSL server certificates are installed on the server and they enable encrypted communications with clients	Certificates must be copied to, or re-created on, the target computer
Custom user and group accounts	User and group accounts that are created specifically for the Web sites and applications.	If the user and group accounts are local to the source server, the user and group accounts need to be created on the target server.
Additional software	Additional software refers to installed commercial software and applications required by the applications on your server.	This additional software must be installed on the target server. This software might require modification to be compatible with Windows Server 2003 and IIS 6.0.

Determining Compatibility with Windows Server 2003

At a minimum, your existing system hardware and software must be compatible with the Windows Server 2003 family before migrating the Web sites and applications to IIS 6.0. You must identify any software or hardware devices that are incompatible with
Windows Server 2003.

Compatibility of Existing Hardware

When you select the computer that will be the target server, ensure that the computer is compatible with Windows Server 2003. The most common hardware incompatibility is a device driver that is no longer supported or is not yet supported in Windows Server 2003. When a device is no longer supported, remove the existing device and then install an equivalent device that is supported by Windows Server 2003. When a device is not supported, look for updated drivers on the device manufacturer's Web site or see the Windows Update link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources. It is also important that you have the latest BIOS version that is available from your computer manufacturer.

For example, the target server might have a network adapter that is not included with Windows Server 2003. You can review the manufacturer's Web site to obtain a driver that is compatible with Windows Server 2003.

For more information about the hardware devices supported by the Windows Server 2003 operating systems, see the *Hardware Compatibility List* on the product CD-ROM or see the Hardware Driver Quality link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources.

Compatibility of Existing Software

Before migrating, you need to consider the compatibility of your existing applications, or other software that is installed on your server, with Windows Server 2003. This includes software and tools from manufacturers other than Microsoft, as well as Microsoft server products that do not ship with the Windows operating system. Make sure that you have the latest versions of all pre-existing software or service packs that are compatible with Windows Server 2003.

The most common software incompatibilities are caused when your application depends on software from another manufacturer that does not support Windows Server 2003. Or, you might have applications that were designed to run on Windows NT Server 4.0 or Windows 2000 Server operating systems, which reference APIs that have been changed or removed in
Windows Server 2003.

To help you determine the compatibility of your existing software with Windows Server 2003, use the Windows Application Compatibility Toolkit before migrating your Web sites and applications to the target server. To download the latest version of the Windows Application Compatibility Toolkit, see the Windows Application Compatibility link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources. For the latest information about compatibility with Windows Server 2003, see the Windows Server 2003 link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources.

Determining Application Compatibility with Worker Process Isolation Mode

IIS 6.0 can run in one of two distinct modes of operation, which are called application isolation modes. *Application isolation* is the separation of applications by process boundaries that prevent the applications from affecting one another, and it is configured differently for each of the two IIS application isolation modes: IIS 5.0 isolation mode and worker process isolation mode.

Worker process isolation mode uses the redesigned architecture for IIS 6.0. This application isolation mode runs all application code in an isolated environment. However, unlike earlier versions of IIS, IIS 6.0 provides isolation without a performance penalty because fewer processor instructions are run when switching from one application pool to another. Worker process isolation mode is compatible with most existing Web sites and applications. Whenever possible, run IIS 6.0 in worker process isolation mode benefit from the enhanced performance and security in IIS 6.0.

IIS 5.0 isolation mode provides compatibility for applications that depend upon the process behavior and memory model of IIS 5.0. Run IIS in this mode only when a Web site or application cannot run in worker process isolation mode, and run it only until the compatibility issues are resolved.

Important

IIS 6.0 cannot run both application isolation modes simultaneously on the same server. Therefore, on a single server running IIS 6.0, you cannot run some Web applications in worker process isolation mode and others in IIS 5.0 isolation mode. If you have applications that require separate modes, you must run them on separate servers.

During the migration process, you install Windows Server 2003 and IIS 6.0 on the target server. IIS 6.0 is configured to run in worker process isolation mode by default. Before you begin migrating your production Web sites and applications, evaluate whether your Web sites and applications are compatible with worker process isolation mode. Most of the compatibility issues with IIS 6.0 occur when configuring IIS 6.0 to run in worker process isolation mode.

One of the most common reasons for incompatibility with worker process isolation mode is that applications do not recognize custom Internet Server API (ISAPI) extensions or dynamic-link libraries (DLLs) that depend on the memory and request processing models used by earlier versions of IIS. Determine application compatibility in your lab before migrating your existing IIS Web sites and application, and if you determine that your applications are not compatible with worker process isolation mode, you can run the applications in IIS 5.0 isolation mode.

Note

Identifying a complete list of potential incompatibilities that applications can experience with worker process isolation mode is beyond the scope of this book. Even after following the guidelines in this chapter, you need to verify in a lab whether your Web sites and applications are compatible with worker process isolation mode.

Determine the compatibility of an application with worker process isolation mode by completing the following steps:

1. Evaluate the benefits of moving to worker process isolation mode.
2. Evaluate the application changes that are required so that the applications can run in worker process isolation mode.
3. Evaluate the management and provisioning script changes that are required to set up programs and provisioning scripts in worker process isolation mode.
4. Verify the compatibility of the application with worker process isolation mode in a lab.

Evaluating the Benefits of Worker Process Isolation Mode

Worker process isolation mode provides higher levels of security and availability for Web sites and applications than IIS 5.0 isolation mode. Therefore, it is recommended that you configure IIS 6.0 to run in worker process isolation mode.

Worker process isolation mode provides the following improvements to IIS 6.0.

Security Enhancements

IIS 6.0 includes a variety of security features and technologies that help ensure the integrity of your Web site content, and of the data that is transmitted through your sites. The following security enhancement is only available when IIS 6.0 is running in worker process isolation mode.

Default process identity for Web sites and applications is set to NetworkService

In IIS 5.0 isolation mode, the default process identity is LocalSystem, which enables access to, and the ability to alter, nearly all of the resources on the Web server. The potential of attacks is reduced in worker process isolation mode because Web sites and applications run under the NetworkService identity. The NetworkService identity is granted less privileges, which helps prevent an attack from compromising the Web server, which is possible with the LocalSystem identity.

Performance and Scaling Enhancements

Future growth in the utilization of your Web sites and applications requires increased performance and scalability of Web servers. By increasing the speed at which Hypertext Transfer Protocol (HTTP) requests can be processed and by allowing more applications and sites to run on one Web server, the number of Web servers that you need to host a site is reduced. The following are a few of the performance improvements included in worker process isolation mode.

Support for processor affinity for worker processes in an application pool

You can configure all of the worker processes in an application pool to have affinity with specific processors in a multiprocessor or server. Processor affinity allows the worker processes to take advantage of more frequent processor caching (Level 1 or Level 2).

Elimination of inactive worker processes and reclamation of unused resources

You can configure application pools to have worker processes request a shutdown if they are idle for a certain amount of time. This can free unused resources for other active worker processes. New worker processes are then started only when they are needed.

Distributing client connections across multiple worker processes

You can configure an application pool to have more than one worker process servicing client connections, also known as a *Web garden*. Because there are multiple worker processes, the incoming client connections are distributed across the worker processes and throughput is not constrained by a single worker process.

Ability to Isolate Web sites and applications from each other

You can isolate Web sites and applications without incurring a performance penalty. This is because the Web site and applications, and their associated ISAPI filters, run in the same process.

Availability Enhancements

Because worker process boundaries isolate the applications in an application pool from the applications in other application pools, if an application fails, it does not affect the availability of other applications running on the server. Deploying applications in application pools is a primary advantage of running IIS 6.0 in worker process isolation mode.

Reduced number of Web server restarts required when administering Web sites and applications

Many of the common operation tasks do not force the restart of the server or the Web service restart. These tasks, such as upgrading site content or components, debugging Web applications, or dealing with faulty Web applications, can be performed without affecting service to other sites or applications on the server.

A fault-tolerant request processing model for Web sites and applications

In IIS 5.0 isolation mode, each Web site or application has only one worker process. However, in worker process isolation mode, you can create a *Web garden* by configuring a number of worker processes to share the processing. The benefit of a Web garden is that if one worker process stops responding, other worker processes are available to accept and process requests.

Isolation of failed worker processes from healthy worker processes

In worker process isolation mode, IIS can determine that a worker process is failing and start a new worker process to replace the failing worker process. Because a new worker process is created before the old worker process terminates, users requesting the Web site or application experience no interruption of service. After IIS creates the new worker process, the failed worker process can be separated, or *orphaned*, from the application pool. The advantage of orphaning a worker process rather than terminating it is that debugging can be performed on the orphaned worker process.

Health monitoring of Web sites and applications

In worker process isolation mode, you can configure an application pool to monitor not only the health of the entire application pool, but also individual worker processes servicing the application pool. Monitoring the health of a worker process allows IIS to detect that a worker process is unable to serve requests and to take corrective action, such as recycling the failed worker process.

In addition, worker process isolation supports other responses when a failed worker process or application pool is detected. For example, IIS can attach a debugger to an orphaned worker process or notify an administrator that an application pool has failed due to rapid-fail protection.

Prevention of Web sites or applications that fail quickly from consuming system resources

In some cases, availability can be affected by Web sites and applications that fail very quickly, are automatically restarted, and then fail quickly again. The endless cycle of failure and restarting can consume system resources, causing other Web sites and applications to experience denial of services because of system resource shortages.

Worker process isolation mode includes *rapid-fail protection* that stops an application pool when too many of the worker processes assigned to an application pool are found to be unhealthy within a specified period of time.

Automatic restart of poorly performing Web sites and applications

Some Web sites and applications have memory leaks, are poorly coded, or have other unidentified problems. In IIS 5.0 isolation mode, these applications can force you to restart the entire Web server. The recycling feature in worker process isolation mode can periodically restart the worker processes in an application pool in order to manage faulty applications. Worker processes can be scheduled to restart based on several options, such as elapsed time or the number of requests served.

Evaluating Application Changes Required for Worker Process Isolation Mode

In most cases, the existing Web sites and applications can be hosted in worker process isolation mode without modification. However, the following are known application issues that can create incompatibilities with worker process isolation mode and require you to run IIS 6.0 in IIS 5.0 isolation mode.

- **Requires Inetinfo.exe.** When the application must run in the same process with Inetinfo.exe, IIS must be configured to run in IIS 5.0 isolation mode because Inetinfo.exe does not process Web requests in worker process isolation mode. In worker process isolation mode, W3wp.exe processes Web requests.
- **Requires Dllhost.exe.** When the application depends on Dllhost.exe to process Web requests for the application, IIS must be configured to run in IIS 5.0 isolation mode because Dllhost.exe is not available in worker process isolation mode.
- **Requires read raw data filters.** If the application uses an ISAPI raw data filter, IIS must be configured to run in IIS 5.0 isolation mode.
- **Requires version 1.0 of the .NET Framework.** When the application requires version 1.0 of the .NET Framework, IIS must be configured to run in IIS 5.0 isolation mode because version 1.0 of the .NET Framework is only supported in IIS 5.0 isolation mode.

- **Requires ISAPI filters or extensions that are incompatible with worker process isolation mode.** When the application requires ISAPI filters or extensions that are incompatible with worker process isolation mode, IIS must be configured to run in IIS 5.0 isolation mode. ISAPI filters or extensions might be incompatible if the filter or extension has one of the following characteristics:
  - Runs in multiple instances and expects to be recycled by using the recycling provided in IIS 5.0.
  - Expects to have exclusive lock on a resource, such as a log file.

If any of the Web sites and applications running on the existing Web server has one or more of these characteristics, then do one of the following:

- Configure IIS 6.0 to run in IIS 5.0 isolation mode to ensure Web site and application compatibility. If the Web sites and applications need to be hosted in IIS 5.0 isolation mode, then the migration process is complete.
- Modify the existing applications to remove the dependencies.

Evaluating Management and Provisioning Script Changes Required for Worker Process Isolation Mode

When management or provisioning scripts exist for your Web sites and applications, you might need to modify them so that they properly set up the Web sites and applications for the application isolation mode that is running on the Web server — IIS 5.0 isolation mode or worker process isolation mode. If you do not modify your management and provisioning scripts as required, you will be unable to use them to install and configure your Web sites and applications on IIS 6.0.

For example, you might have a provisioning script that uses Microsoft Active Directory® Service Interfaces (ADSI) to create a Web site and configure the site for High isolation in IIS 5.0 isolation mode, which does not exist in worker process isolation mode. After migration and when the Web server is running in worker process isolation mode, you need to modify the script to create application pools instead.

Common modifications that might be necessary to your setup programs and provisioning scripts include:

- When the script installs an ISAPI extension or filter, you might need to modify the script to add an entry for the ISAPI extension or filter to the Web service extensions restriction list and set the status of the entry to **Allowed**.

For more information about modifying your setup programs or provisioning scripts to install and enable an ISAPI extension or filter, see “Configuring Web Service Extensions” later in this chapter.

- When the script installs an application that contains dynamic content, you might need to modify the script to set the status of the appropriate Web service extensions to **Allowed**, so that IIS allows the dynamic content to run.

For more information about modifying your setup programs or provisioning scripts to enable dynamic content, see "Configuring Web Service Extensions" later in this chapter.

- When the script installs a Web site or application that runs in High isolation in IIS 5.0, you might need to modify the script to create an application pool and configure the application pool with settings that are comparable to the original IIS 5.0 isolation settings.

For more information about modifying your setup programs or provisioning scripts to create and configure application pools, see "Configuring Application Isolation Settings in Worker Process Isolation Mode" later in this chapter.

- When the script references metabase properties, you might need to modify the script if it references metabase properties that have changed or are no longer supported in IIS 6.0.

For more information about modifying your setup programs or provisioning scripts to reference the proper metabase properties in IIS 6.0, see "Modifying References to IIS 6.0 Metabase Properties" later in this chapter.

Verifying Application Compatibility with Worker Process Isolation Mode in a Lab

After modifying your Web sites, applications, setup programs, and provisioning scripts to be compatible with worker process isolation mode, you need to test your modifications in a lab. Be sure to test for compatibility before performing the migration process on a production Web server.

Verify the compatibility of your Web sites, applications, setup programs, and provisioning scripts with worker process isolation mode in a lab by completing the following steps:

1. Make an image backup of the source server.
2. Restore the backup to a Web server in your lab, referred to hereafter as a *test source server*. Ensure that the test source server is not connected to your production network, to prevent any problems encountered during the migration from affecting your production network.
3. Perform a migration from the test source server to the test target server.
4. Configure the test target server to run in worker processor isolation mode or IIS 5.0 isolation mode.
5. Make the necessary modifications to the Web sites, applications, setup programs, and provisioning scripts so that they are compatible with worker process isolation mode.
6. Verify that the Web sites, applications, setup programs, and provisioning scripts run correctly on the test target server.

For more information about setting up a test lab, see “Designing a Test Environment” in *Planning, Testing, and Piloting Deployment Projects* of the *Microsoft® Windows® Server 2003 Deployment Kit*.

Determining Application Compatibility with the .NET Framework

For a successful migration to Windows Server 2003 and IIS 6.0, you need to determine whether your ASP.NET applications are dependent on specific versions of the .NET framework. Windows Server 2003 ships with version 1.1 of the .NET Framework, while most of the .NET applications that were developed before the release of Windows Server 2003 were designed to run on version 1.0 of the .NET framework.

Note

Version 1.0 of the .NET Framework is only supported in IIS 5.0 isolation mode. Therefore, you can only run version 1.0 and version 1.1 of the .NET Framework on the same server when IIS 6.0 is configured to run in IIS 5.0 isolation mode. Version 1.1 of the .NET Framework is supported in IIS 5.0 isolation mode or worker process isolation mode.

Typically, ASP.NET applications running on version 1.0 of the .NET Framework are compatible with version 1.1 of the .NET Framework. However, there might be some incompatibilities, the majority of which are security related and can be corrected by configuring the .NET Framework to be less restrictive.

For example, in the .NET Framework version 1.0, the .NET Framework only examines the **SQLPermission.AllowBlankPassword** attribute if the user actually includes the password keyword in their connection string. If an administrator or user sets the **SQLPermision.AllowBlankPassword** attribute to **False**, it is possible to specify a connection string like “server=(localhost);uid=sam” and succeed. In the .NET Framework version 1.1, this connection string fails.

You can use both version 1.0 and version 1.1 of the .NET Framework on the same server running IIS 6.0, which is also known as *side-by-side configuration*. Side-by-side configuration allows you to run a mixture of applications that require version 1.0 or version 1.1 of the .NET Framework. During your lab testing, determine the version of the .NET Framework that is required by your applications.

For a current list of possible compatibility issues when upgrading from version 1.0 to version 1.1 of the .NET Framework, see the Compatibility Considerations and Version Changes link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources. For more information about using multiple versions of the .NET Framework in side-by-side configuration, see "Configuring IIS to Use the Correct Version of the .NET Framework" later in this chapter.

Important

ASP.NET is not available on the following operating systems: Microsoft® Windows® XP 64-Bit Edition; the 64-bit version of Windows® Server 2003, Enterprise Edition; and the 64-bit version of Windows® Server 2003, Datacenter Edition. For more information, see "Features unavailable on 64-bit versions of the Windows Server 2003 family" in Help and Support Center for Windows Server 2003.

Selecting a Migration Method

Before migrating your Web sites and applications to IIS 6.0, you need to determine whether to perform the migration manually or with the IIS 6.0 Migration Tool. It is recommended that you use the migration tool to begin the migration process, except when one of the following is true:

- **You have set up programs, installation scripts, or provisioning scripts for the Web sites and applications that you are migrating.** When the Web sites and applications that you are migrating have setup programs, installation scripts, or provisioning scripts, use those programs or scripts to install the Web sites and applications on the target server. Ensure that the setup programs, installation scripts, and provisioning scripts have been properly modified to install the Web sites and applications on IIS 6.0.

The target server is configured to run in IIS 5.0 isolation mode. When the Web sites and applications that you are migrating require the target server to run in IIS 5.0 isolation mode, you must perform the migration manually. To determine whether your Web sites and applications are compatible with worker process isolation mode, see "Determining Application Compatibility with Worker Process Isolation Mode" earlier in this chapter.

- **The source server has a significant number of FrontPage extended Web sites.** When the FrontPage extended Web sites make extensive use of the administrative and publishing security-related settings found in FrontPage 2000 Server Extensions, perform the migration manually. To ensure that these security-related settings are migrated properly to the target server, perform the migration manually and use FrontPage publishing to transfer the Web site to the target server.
- **You want to migrate individual virtual directories.** When you want to migrate individual virtual directories, perform the migration manually. The IIS 6.0 Migration Tool only moves Web site content and configuration settings at the Web site level, which means that all of the virtual directories beneath the Web site are migrated to the target server.

If you determine that you need to perform the migration manually, then you can proceed to the next step in the migration process — Deploying the Target Server. Otherwise, you need to know which steps in the migration process are automated by the IIS 6.0 Migration Tool, and which steps you must complete manually after running the migration Tool. Knowing the role of the IIS 6.0 Migration Tool in the migration process enables you to have the appropriate tools and resources available when you are ready to begin the migration.

Identifying the Role of the IIS 6.0 Migration Tool

The IIS 6.0 Migration Tool is a command-line utility that automates some of the steps in the process for migrating Web sites and applications hosted on IIS 4.0 or IIS 5.0 to IIS 6.0. The migration tool does not provide an end-to-end migration solution, but it automates some of the time-consuming, repetitive migration tasks. Although the migration tool is conceptually similar to the IIS 5.0 Migration Tool, the IIS 6.0 Migration Tool is a completely new tool designed specifically for Windows Server 2003 and is not compatible with the previous tool.

Migration Tasks That Are Automated by the IIS 6.0 Migration Tool

The IIS 6.0 Migration Tool automates the following steps in the IIS migration process:

- **Transferring the Web site content.** All of the files and folders located in the home directory of the Web site and virtual directories (which can be located outside of the home directory of the Web site) are copied from the source server to the target server. The NTFS file system permissions assigned to the files and directories that make up the Web site content on the source server are granted to the corresponding files and directories on the target server. Any content referenced by the Web site or application that is not located in subdirectories of the home directory of the Web site or in virtual directories is not migrated.
- **Transferring the Web site configuration in the IIS metabase.** The configuration for each Web site and application, which is stored in the IIS metabase properties on the source server, is translated and then the corresponding IIS 6.0 metabase properties are appropriately configured on the target server.
- **Translating the application isolation configuration.** The target server is running in worker process isolation mode. The application isolation configuration for each Web site and application on the source server is translated into application pool configuration settings on the target server.
- **Backing up the IIS metabase configuration to the target server.** The IIS metabase configuration is backed up on the target server before migration. You can use this backup to restore the target server to a known state in the event that the migration process is not successful.

Migration Tasks That Must Be Completed Manually

The following steps in the Web site migration process must be completed after running the IIS 6.0 Migration Tool.

Migrating Additional Web Site and Application Content

The IIS 6.0 Migration Tool migrates all of the content that is located in the home directory of the Web site and in any subdirectories contained in that home directory. You can migrate any Web site and application content that is not located in these directories by completing the following steps:

- **Migrate content located outside the home directory and subdirectories of the Web site.** When the Web sites and applications reference content that is located in folders outside of the home directory of the Web site or the virtual directories beneath the home directory, you must migrate this content manually.
- **Migrate content located in a virtual directory.** When the virtual directory content on the source server is stored on a disk volume, such as F:, that does not exist on the target server, you must migrate this content manually.

Configuring Additional Web Site and Application Properties

After running the IIS 6.0 Migration Tool, the Web sites are configured comparably to how they were configured on the source server. However, depending on the configuration of the Web sites and applications on the source server, you might need to configure additional Web site and application properties, by completing the following steps:

- **Change the IIS metabase settings to reflect where Windows is installed.** If the Windows Server 2003 systemroot path does not match the Windows systemroot path on the source server, you must modify the metabase settings on the migrated Web sites to reference the correct folder on the target server. For example, if Windows was installed on C:\WINNT on the source server, the IIS metabase entries for **ScriptMaps**, and **HTTPErrors** properties might still reference these paths, and therefore need to be updated on the target server.
- **Configure IIS properties that reference local user accounts.** There are a number of Web site configuration properties on the source server that you can configure to utilize user accounts that are local to the source server. Local user and group accounts are not migrated from the source server to the target server by the IIS 6.0 Migration Tool. As a result, the migrated Web sites reference user accounts that do not exist on the target server. In these cases, you must configure the Web sites to utilize user accounts that are domain-based or local to the target server, and then re-create the file system permissions on migrated content by completing the following steps:
  - **Configure local user NTFS permissions on content.** If NTFS permissions are granted to local user accounts on the source server, you must create new user accounts, or designate existing user accounts, for use on the target server and then grant the corresponding NTFS permissions to the user accounts on the target server.

    - **Configure anonymous account properties for Web sites and virtual directories.** If a Web site or virtual directory is configured to use a user account on the source server for anonymous access (other than the default IUSR_computername account), you must create a new user account, or designate an existing user account, for use on the target server.
    - **Configure IIS 4.0 and IIS 5.0 application isolation identities.** If the Web sites or applications are isolated and are configured to use a local user account on the source server as the application isolation identity, you must create a new user account, or designate an existing user account, for use on the target server. Then you must configure the corresponding application pool, on the target server, to use the newly created account as the identity of the application pool.
- **Add Web service extensions for dynamic content used by the Web sites.** Any dynamic content types, including ISAPI extensions, ISAPI filters, or CGI applications, which are not automatically migrated by the IIS 6.0 Migration Tool need to be added to the Web service extensions list.
- **Add MIME types for static content used by the Web sites.** MIME types define the types of static files that are served by the Web server. You must identify the MIME types defined on the source Web server and then create the same associations of MIME types to file name extensions on the target Web server.
- **Configure SSL certificates.** You must export server certificates for Secure Sockets Layer (SSL)-enabled Web sites from the source server, and then install the certificates on the target server after the migration process is complete.
- **Configure FrontPage Server Extensions users and roles.** If FrontPage Server Extensions is configured to use a local user account on the source server as the FrontPage administrator, you must create a new user account, or designate an existing user account, for use on the target server. In addition, you must assign the user the same FrontPage role as the corresponding user had on the source server.

Configure IIS for ASP.NET applications. If you migrate ASP.NET applications from the source server, you might need to migrate the attribute settings in the Machine.config file that are used to set process-model behavior. When the target server is configured for worker process isolation mode, the attribute settings in the Machine.config file need to be converted to corresponding application pool settings. This is because ASP.NET uses the IIS process model when IIS is running in worker process isolation mode. For more information, see "Migrating Machine.config Attributes to IIS 6.0 Metabase Property Settings" later in this chapter.

Performing Application-Specific Migration Tasks

In addition to the IIS 6.0 configuration changes that are required, you might need to complete any migration tasks that are specific to the applications running on the source server. The application-specific steps that you might need to perform include the following:

- **Modifying application code for compatibility with Windows Server 2003 and IIS 6.0.** When the applications on the source server are not compatible with Windows Server 2003 and IIS 6.0, you need to modify the applications. Most of these modifications are required when applications use application programming interfaces (APIs) no longer supported by Windows Server 2003 and IIS 6.0.
- **Installing additional software required by the applications.** When applications on the source server require additional software that was developed by your organization, by Microsoft, or by other organizations, you need to install that software on the target server. This software can include filters and ISAPI extensions. You must obtain a version of the software that is compatible with Windows Server 2003 and IIS 6.0.
- **Migrating MTS packages, COM objects, and COM+ applications.** When your applications include MTS packages, COM objects, or COM+ applications, you must migrate them to the target server. For MTS packages, you need to rewrite the MTS as a COM+ application.
- **Creating IP addresses that are used to uniquely identify the applications.** Web sites and applications are uniquely identified by a unique IP address, a unique combination of an IP address and a TCP port, or host headers. When Web sites and applications on the source server are uniquely identified by IP addresses, you must create corresponding IP addresses on the target server and then configure the migrated Web sites and applications to use the new IP addresses.
- **Creating users and groups that are used by the applications.** When users that access the applications on the source server have accounts that are local to the source server, you need to create new accounts on the source server and then assign the appropriate NTFS permissions on the target server to the new accounts. When the user accounts are in Active Directory, you only need to assign the appropriate NTFS permissions on the target server to the accounts in Active Directory.
- **Creating registry entries for the applications.** When your applications store configuration information in the registry, you might need to create the same registry entries for the applications on the target server.

Note

The IIS 6.0 Migration Tool does not migrate the IIS logs from the source server to the target server. The migration of the IIS logs is not necessary for proper operation of the Web sites and applications on the target server. However, you might want to archive the IIS logs before decommissioning the source server for historical reference to events that occurred on the source server before migration.

For information about using the IIS 6.0 Migration Tool to perform your migration, see "Migrating Web Sites with the IIS 6.0 Migration Tool" later in this chapter.

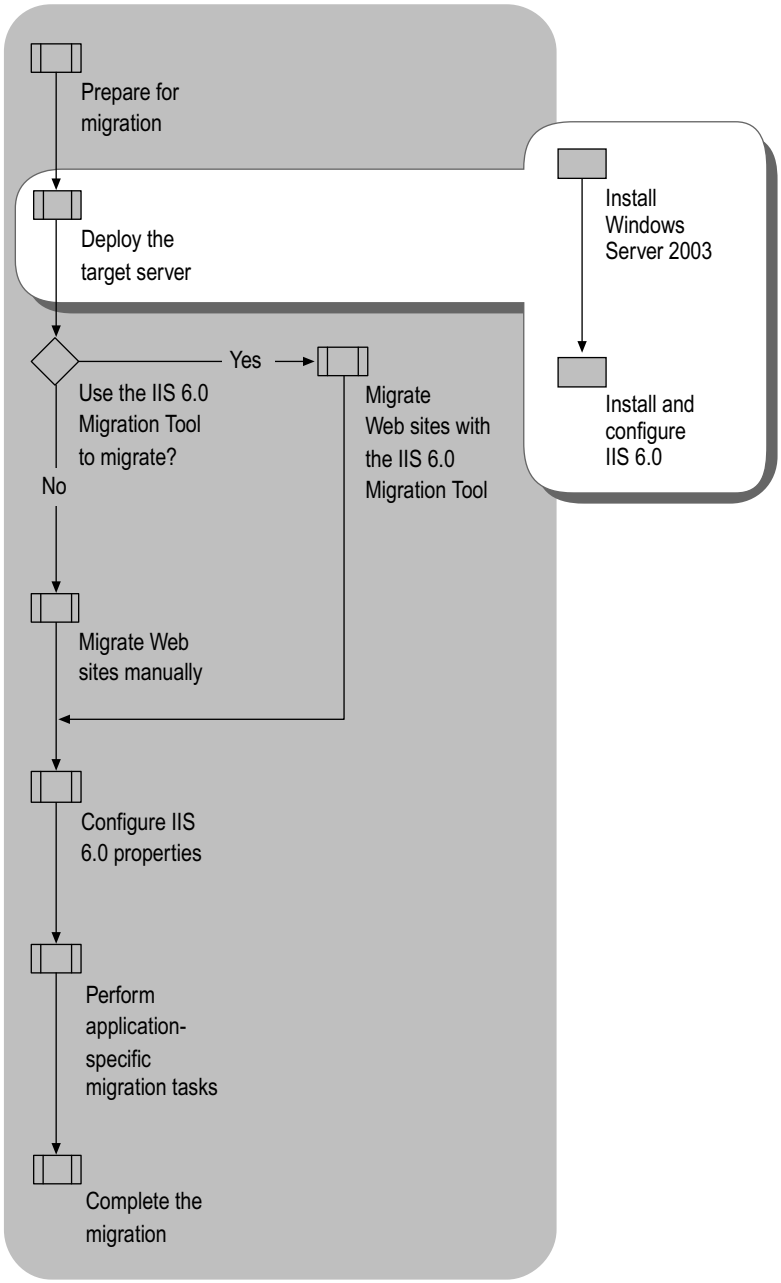
Deploying the Target Server

You must install Windows Server 2003 on the target Web server before you can migrate the Web sites and applications. In addition to installing Windows Server 2003, you must install and configure IIS 6.0 on the target server. To follow the process described in this chapter, install Windows Server 2003 and IIS 6.0 with the default options.

You must also configure the target server to run in worker process isolation mode or in IIS 5.0 isolation mode. Run IIS in IIS 5.0 isolation mode only when a Web site or application cannot run in worker process isolation mode. If there are no incompatibilities, configure IIS to run in worker process isolation mode to utilize the benefits of IIS 6.0 features.

Figure 6.3 illustrates the process for deploying the target server.

Figure 6.4 Deploying the Target Server



If you have already installed Windows Server 2003 on the target server, and you have already installed and configured IIS 6.0 on the target server, you can proceed to "Migrating Web Sites with the IIS 6.0 Migration Tool" or Migrating Web Sites Manually later in this chapter.

Installing Windows Server 2003

The primary concern when installing Windows Server 2003 is to ensure that the security of the target server is maintained. When you install Windows Server 2003 as a dedicated Web server, the default components and services are configured to provide the lowest possible attack surface. You can further reduce the attack surface of the target server by enabling only the essential Windows Server 2003 components and services.

The migration process presented in this chapter assumes that you install Windows Server 2003 with the default options. If you use other methods for installing and configuring Windows Server 2003, such as unattended setup, your configuration settings might be different.

Note

When you complete the installation of Windows Server 2003, Manage Your Server automatically starts. The migration process assumes that you quit Manage Your Server and then further configure the Web server in **Add or Remove Programs** in Control Panel.

For more information about enabling the essential Windows Server 2003 components and services required by your server, see "Enabling Only Essential Windows Server 2003 Components and Services" in "Securing Web Sites and Applications" in this book.

Installing and Configuring IIS 6.0

Because IIS 6.0 is not installed during the default installation of Windows Server 2003, the next step in deploying the target server is to install and configure IIS 6.0. The migration process presented here assumes that you install IIS 6.0 with the default options in **Add or Remove Programs** in Control Panel. If you use other methods for installing and configuring Windows Server 2003, such as Manage Your Server, the default configuration settings might be different.

Install and configure IIS 6.0 by completing the following steps:

1. Install IIS 6.0 with only the essential components and services.

   As with installing Windows Server 2003, the primary concern when installing and configuring IIS 6.0 is to ensure that the security of the target server is maintained. Enabling unnecessary components and services increases the attack surface of the target server. You can help ensure the target server is secure by enabling only the essential components and services in IIS 6.0.

   For more information about how to install IIS 6.0, see "Install IIS 6.0" in "IIS Deployment Procedures" in this book. For more information about the IIS 6.0 protocols and services, see "Enabling Only Essential IIS 6.0 Components and Services" in "Securing Web Sites and Applications" in this book.

2. If the source server has Web sites with FrontPage Server Extensions, install FrontPage 2002 Server Extensions from Microsoft on the target server.

   For more information about how to enable FrontPage Server Extensions, see "Configure Web Service Extensions" in "IIS Deployment Procedures" in this book.

3. Configure IIS 6.0 to run in IIS 5.0 isolation mode.

   If you determined that one or more of your Web sites or applications are incompatible with worker process isolation mode (the target server is currently configured to run in worker process isolation mode), configure IIS 6.0 to run in IIS 5.0 isolation mode in IIS Manager, or by setting the IIS metabase property **IIs5IsolationModeEnabled** to a value of **True**.

Note

If you configure IIS 6.0 to run in IIS 5.0 isolation mode and then decide to change the configuration back to worker process isolation mode, the original worker process isolation mode settings are retained. Similarly, if you configure IIS 6.0 to run in IIS 5.0 isolation mode, change to worker process isolation mode, and then change back to IIS 5.0 isolation mode, the IIS 5.0 isolation mode settings are retained.

For more information about how to configure IIS 6.0 to run in worker process isolation mode or in IIS 5.0 isolation mode, see "Configure Application Isolation Modes" in "IIS Deployment Procedures" in this book. For more information about determining compatibility with worker process isolation mode, see "Evaluating Application Changes Required for Worker Process Isolation Mode" earlier in this chapter.

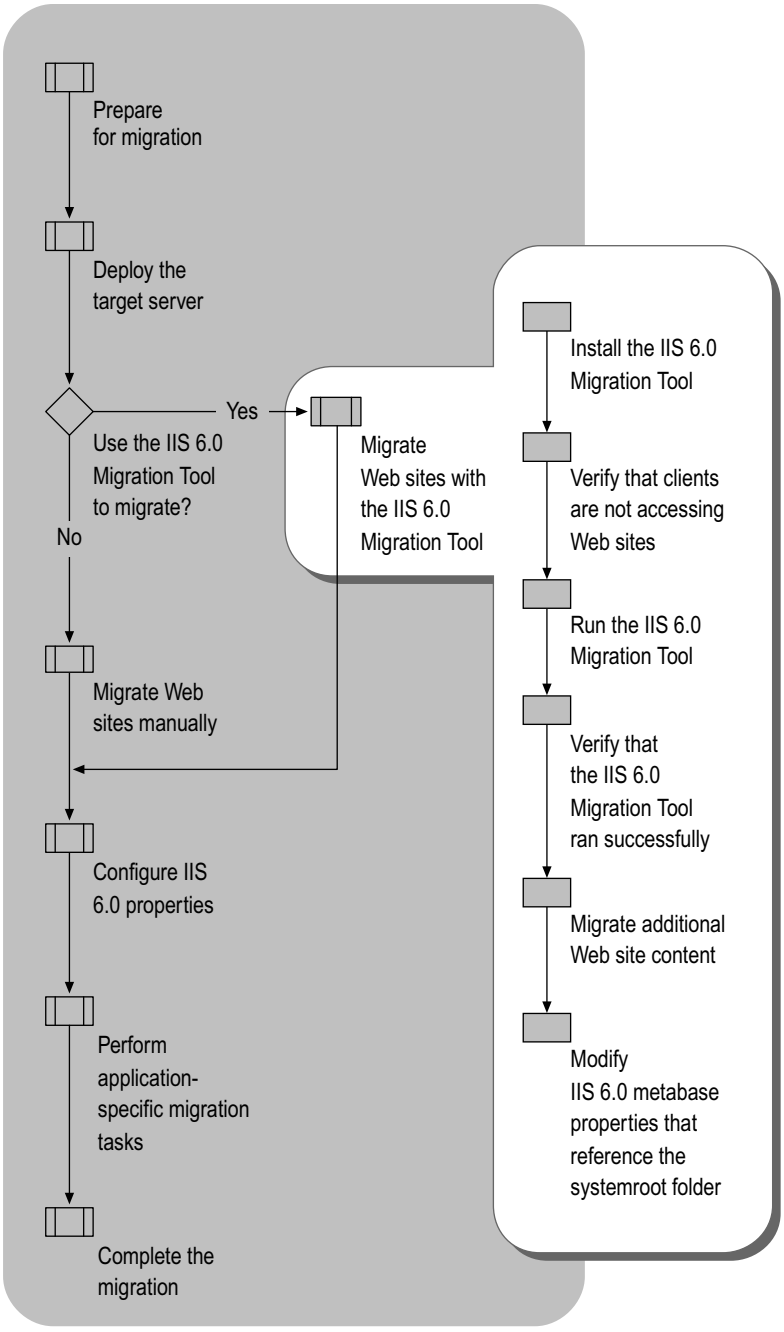
Migrating Web Sites with the IIS 6.0 Migration Tool

Earlier in the process, you determined whether to migrate your Web sites by using the IIS 6.0 Migration Tool or to migrate them manually. If you decided to use the IIS 6.0 Migration Tool to complete the migration process, perform the steps that are described in this section.

If you decided to complete the migration process manually, you can skip this step and proceed directly to "Migrating Web Sites Manually" later in this chapter. For more information about how to determine whether to perform the migration with the IIS 6.0 Migration Tool, see "Selecting a Migration Method" earlier in this chapter.

Figure 6.5 illustrates the process of performing the migration to IIS 6.0 with the IIS 6.0 Migration Tool.

Figure 6.5 Migrating Web Sites to IIS 6.0 with the IIS 6.0 Migration Tool



Installing the IIS 6.0 Migration Tool

The IIS 6.0 Migration Tool and its accompanying user documentation are included in the default installation of the *Windows Server 2003 Deployment Kit* companion CD and the *Internet Information Services (IIS) 6.0 Resource Kit* companion CD. There is no setup program for the migration tool; you need only copy the IIS 6.0 Migration Tool to the target server. You do not need to install any software on the source server. To avoid installing the entire contents of the companion CD onto your target server, install the companion CD on a workstation first, and then copy the migration tool to the target server.

Install the IIS 6.0 Migration Tool by completing the following steps:

1. Install the *Windows Server 2003 Deployment Kit* companion CD or the *Internet Information Services (IIS) 6.0 Resource Kit* companion CD on a computer other than the source server or target server.
2. Create a folder on the target server to contain the migration tool.
3. Copy the IIS 6.0 Migration Tool (Iismt.exe) from Program Files\IIS Resources\IIS 6.0 Migration Tool, on the computer listed in Step 1, to the folder on the target server that you created in Step 2.
4. Verify that the migration tool operates correctly by typing **iismt** at the command prompt.

   This starts the migration tool and displays Help for the tool only. The migration process does not actually begin.

Verifying That Clients Are Not Accessing Web Sites

The IIS 6.0 Migration Tool requires only read access to the Web site content and configuration settings on the source server. Therefore, the source server can remain online in your production environment. However, you might need to remove the source server from your production network and move it to a private network segment that has direct network connectivity to the target server under the following circumstances:

- The number of files and amount of configuration information being copied across the network generates a high volume of traffic and slows the production network.
- Firewalls that exist between the source and target servers prevent the migration tool from performing the migration. This often occurs because the migration tool uses DCOM ports that are blocked by the firewalls for communicating with the source and target servers.

- Security-related configuration settings on the source server need to be modified to allow the migration tool to work.

  Examples of these security-related configuration settings that can prevent the migration tool from working include the following:

  - Remote access to disk volumes through administrative shares is prohibited.

    The IIS 6.0 Migration Tool requires access to the disk volume that contains the Web site content to perform the migration. For example, if the Web site content is stored in D:\Inetpub\Wwwroot, the migration tool must access the administrative share (D$) of the disk volume. The administrative shares are often removed to help prevent unauthorized access to the Web server. In order to use the migration tool, you must re-create the appropriate administrative shares.

  - Remote access to the source server must be allowed for members of the local Administrators group on the source server.

    For security reasons, many organizations restrict the members of the local Administrators group so that they can only log on locally, not over the network. However, the migration tool must be able to remotely access the source server over the network, as a member of the local Administrators group.

Verify that clients are no longer accessing Web sites by completing the following steps:

1. Prevent new clients from accessing the sites by pausing the sites.

   For more information about how to pause Web sites, see “Pause Web or FTP Sites” in “IIS Deployment Procedures” in this book.

2. Monitor the active Web connections to determine when clients are no longer accessing the source server.

   For more information about how to monitor the active Web connections, see “Monitor Active Web and FTP Connections” in “IIS Deployment Procedures” in this book.

3. When the number of active Web counters is zero, stop the WWW service or move the source server to another network segment.

   If you elect to stop the WWW service, ensure that the IIS Admin service is running because the migration tool requires the IIS Admin service. For more information about how to stop the WWW service, see “Stop the WWW Service” in “IIS Deployment Procedures” in this book.

Running the IIS 6.0 Migration Tool

The IIS 6.0 Migration tool is a command-line utility that is designed to run on the target server. To run the migration tool, type **iismt** at the command prompt, and provide the parameters, listed in Table 6.2, that are appropriate to the Web sites and applications that you are migrating. For more information about each of the parameters and on how to perform the migration with the IIS 6.0 Migration Tool, see the "IIS 6.0 Migration Tool User Guide" on the *Windows Server 2003 Deployment Kit* companion CD or the *Internet Information Services (IIS) 6.0 Resource Kit* companion CD.

The IIS 6.0 Migration Tool uses the following syntax:

iismt.exe *Server Website* [**/user** *Username*] [**/password** *Password*] [**/path** *path*] [**/serverbindings** *ServerBindings String*] [**/siteid** *SiteID* | **Replace**] [**/configonly**] [**/fpse**] [**/verbose**] [**/overwrite**] [**/noninteractive**]

Table 6.2 lists all the command-line parameters that the migration tool accepts, although not all of the parameters listed are required.

Table 6.2 Command-Line Parameters Accepted by the IIS 6.0 Migration Tool

Parameter	Required or Optional	Description
SourceServer	**Required**	**Identifies the source server by providing the following:** • DNS or NetBIOS name for the source server. • IP address of the source server.
WebSite	**Required**	**Identifies the site to be migrated by providing the following:** • Web site description, such as "Default Web Site." • Metabase key path, such as W3SVC/1.
/user ***UserName***	**Optional**	**Specifies the user name of an account that is a member of the Administrators group on the source server. This parameter is not necessary if you log on with an account that is a member of the Administrators group on both the source server and the target server.**
/password ***Password***	**Optional**	**Specifies the password that is associated with the user name.**
/path ***Path***	**Optional**	**Specifies a different directory location for the home directory of the Web site on the target server. This parameter is ignored if /configonly is included.**

(continued)

Table 6.2 Command-Line Parameters Accepted by the IIS 6.0 Migration Tool *(continued)*

Parameter	Required or Optional	Description
/serverbindings ***ServerBindingsString***	**Optional**	**Allows a change to the IP address, host header, or port configuration of the Web site during the migration.**
/siteid ***SiteID*** | **Replace**	**Optional**	**Specifies the site ID on the target server, which can be specified as one of the following:** • *Site*ID- Overwrites the site ID on the target server. • Replace - Overwrites the site ID on the target server with the site ID from the source server.
/configonly	**Optional**	**Migrates only the Web site configuration and not the Web site content.**
/fpse	**Optional**	**Re-applies FrontPage Server Extensions to the migrated site on the target server. This parameter is ignored if** /configonly **is included.**
/verbose	**Optional**	**Displays metabase path copy and file copy operations to the screen during the migration process.**
/overwrite	**Optional**	**Does not display messages that prompt the user to confirm the overwrite of an existing destination folder or file when content is being copied from the source server to the target server.**
/noninteractive	**Optional**	**Does not display messages that prompt the user for input. The migration tool will exit on the first error condition. This is a useful switch for invoking the migration tool from a batch file or script program to perform an unattended migration.**

Verifying That the IIS 6.0 Migration Tool Ran Successfully

Before continuing with the Web site migration, verify that the IIS 6.0 Migration Tool migrated the Web site content and configuration information successfully. When you run the migration tool, the output displayed by the migration tool indicates the success or failure of the migration. If the output indicates that errors occurred, you can use the IIS 6.0 Migration Tool log file to resolve any errors.

Verify that the IIS 6.0 Migration Tool ran successfully by completing the following steps:

1. Open *systemroot*\System32\LogFiles\IISMT\iismt_*date_time*.log in a text editor and determine if any errors occurred (where *date* is the date when the tool ran and *time* is the time the tool started).
2. Review the log file and resolve any problems that occurred during migration before proceeding to the next step in the process.

Migrating Additional Web Site Content

Some Web sites and applications have content that is not located in the home directory of the Web site or in subdirectories that are inside the home directory. The IIS 6.0 Migration Tool only migrates Web site content from the following locations:

- Within the home directory and subdirectories of the Web site.
- In virtual directories whose disk volume letter exists on both the source server and the target server.

If the code in your applications directly references content that is located outside the home directory and subdirectories of the Web site, or if a virtual directory is stored on a disk volume letter that does not exist on the target server, you must migrate this Web site content manually.

Migrating Content Located Outside the Home Directory of the Web Site

A Web site or application can have content that is referenced by the Web site, but is located outside the home directory and subdirectories of the Web site. This content must be migrated manually because the IIS 6.0 Migration Tool migrates content only in the home directory and subdirectories of the Web site.

Migrate content that is located outside the home directory and subdirectories of the Web site by completing the following steps:

1. Create a folder on the target server to contain the content.

   Ensure that the folder is in the same relative location to the home directory of the Web site. For example, if a Web site on the source server is in D:\Wwwroot*WebSite* and the content to be migrated is in D:\Program Files*SiteContent*, create the same folder (D:\Program Files*SiteContent*) on the target server.

2. Copy the content from the source server to the target server.

   Use the process of your choice for copying the content as described in “Migrating Web Site Content” later in this chapter.

3. If necessary, modify any references to the content that is located in directories that are external to the home directory of the Web site on the target server.

   The target server might have a different disk configuration than the source server, or you might be migrating the content to a different disk volume letter on the target server than the source server. You need to modify any references in the Web site to content that is located outside the home directory and subdirectory of the Web site.

Migrating Content Located in Virtual Directories

Virtual directory content on the source server might be stored on a disk volume, such as F:, that does not exist on the target server. This content must be migrated manually because the IIS 6.0 Migration Tool migrates virtual directories only when the disk volume letter on which the virtual directory is located exists on both the source server and the target server.

Migrate content that is located in virtual directories, which are stored on a disk volume that does not exist on the target server, by completing the following steps:

1. Create a folder on the target server to contain the virtual directory and the content.

   Ensure that the folder is in the same relative location on the target servers as on the source server.

2. Copy the content from the source server to the target server.

   Use the process of your choice for copying the content as described in “Migrating Web Site Content” later in this chapter.

3. Create the virtual directory under the appropriate Web site on the target server that references the folder created in Step 1.

For more information about how to create virtual directories on the target server, see “Create a Virtual Directory” in “IIS Deployment Procedures” in this book.

Modifying IIS 6.0 Metabase Properties That Reference the Systemroot Folder

The IIS 6.0 Migration Tool migrates metabase properties that reference the systemroot folder on the source server, but does not update the references to the systemroot folder on the target server. If the location of systemroot folder on the target server does not match the location of the systemroot folder on the source server, you must modify the metabase settings on the migrated Web sites to reference the correct systemroot folder on the target server. Because the default systemroot folder name changed from WINNT to Windows in Windows 2000 Server and later versions, you might need to manually modify the metabase properties that reference the systemroot folder on the target server.

These metabase properties that reference the location of the systemroot folder can include the following:

- **HttpErrors.** The **HttpErrors** metabase property specifies the custom error string sent to clients in response to HTTP 1.1 errors. Each string in the list specifies the HTTP error code and subcode, indicates whether the handler will be a URL or a file, and specifies which URL or file the client will be sent. Each string can be in either a URL or a file format. If you migrate the Default Web Site on Windows NT 4.0 or Windows 2000, you must reset the HttpErrors metabase property to contain the systemroot folder for Windows Server 2003, even if the systemroot folder exists on the same drive on both the source and target computer.
- **ScriptMaps.** The **ScriptMaps** metabase property specifies the file name extensions of applications that are used for script processor mappings. This property contains references to paths with default ISAPIs, such as C:\Windows\system32\inetsrv\asp.dll.

Compensate for the differences in the location of the systemroot folder on the target server by completing the following steps:

1. Enable the IIS 6.0 metabase edit-while-running feature.
2. Open the MetaBase.xml file in Microsoft Notepad.
3. Search for any references to the systemroot folder on the source server and replace these references with the systemroot folder path on the target server.

   For example, if the source server had been installed in C:\WINNT and the target server is installed into C:\Windows, you should replace any occurrences of “C:\WINNT” with “C:\Windows”.
4. Save the MetaBase.xml file.
5. Disable the metabase edit-while-running feature.

   For more information about the edit-while-running feature, see “Metabase Edit-While-Running Feature” in IIS 6.0 Help, which is accessible from IIS Manager.

Migrating Web Sites Manually

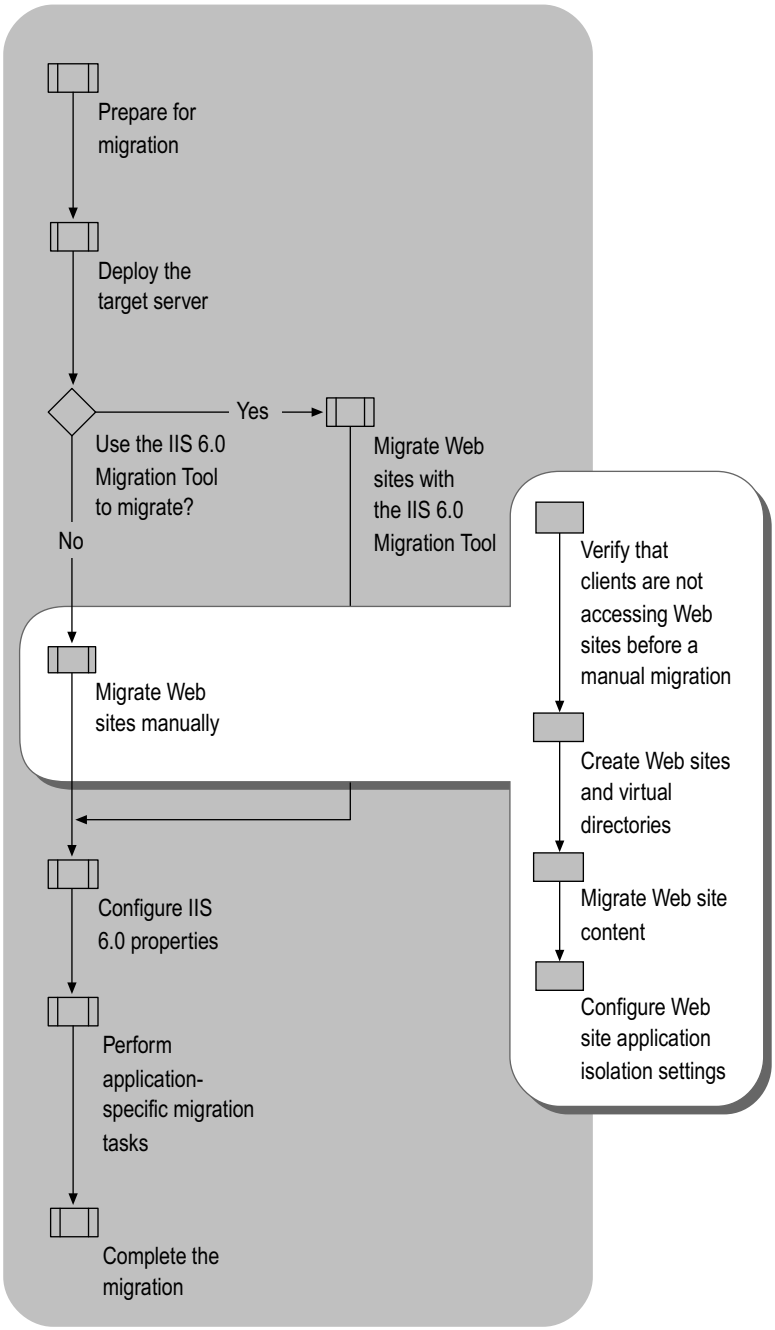
Earlier in the process, you decided whether to migrate your Web sites with the IIS 6.0 Migration Tool or to migrate them manually. If you are unable to use the migration tool, you must migrate manually, or use existing provisioning or setup scripts. For more information about how to determine whether you can perform the migration with the IIS 6.0 Migration Tool, see “Selecting a Migration Method” earlier in this chapter.

When there are provisioning or setup scripts for your Web sites and applications, use these scripts to install the Web sites and applications on the target server. These scripts might require modification to be compatible with worker process isolation mode, which is discussed in “Evaluating Application Changes Required for Worker Process Isolation Mode” earlier in this chapter.

Because the scripts install and configure the Web sites and applications, no migration is required. In this case, run the scripts to install and configure the Web sites and applications and then proceed to “Configuring IIS 6.0 Properties” later in this chapter to continue with the process.

Figure 6.6 illustrates the process migrating Web sites manually to IIS 6.0.

Figure 6.6 Performing a Manual Migration to IIS 6.0



Verifying That Clients Are Not Accessing Web Sites Before a Manual Migration

Before migrating your existing Web sites and applications, ensure that no active client sessions are running. For more information about verifying that clients are not accessing Web sites, see “Verifying That Clients Are Not Accessing Web Sites” earlier in this chapter.

Creating Web Sites and Virtual Directories

For each Web site and virtual directory on the source server, you must create a corresponding Web site and virtual directory on the target server. Later in the migration process, you will copy the content into these Web sites and virtual directories.

Create the Web sites and virtual directories by completing the following steps:

1. Create the Web sites and home directories on the target server.
2. Create the virtual directories.

Creating Web Sites and Home Directories on the Target Server

Each Web site must have one home directory. The home directory is the central location for your published pages. It contains a home page or index file that welcomes visitors and contains links to other pages in your site. The home directory is mapped to the Web site's domain name or to the name of the Web server.

Create a Web site and home directory on the target server by completing the following steps:

1. Create the folder that will be the home directory for the Web site on the target server.

   The folder that is the home directory of the Web site contains all of the content and subdirectories for the Web site. The folder can be created on the Web server or on a UNC-shared folder on a separate server. At a minimum, create the folder on the following:

   - An NTFS partition, which helps ensure proper security.
   - A disk volume other than the system volume, which reduces the potential of an attack on a Web site bringing down the entire Web server, and improves performance.

   For more information about securing Web sites and applications see “Securing Web Sites and Applications” in this book. For more information about creating directories for your Web sites see “Create a Web Site” in “IIS Deployment Procedures” in this book.

2. Determine whether to generate the Web site identification number incrementally, or from the Web site name.

   Although site identification numbers were generated incrementally in IIS 5.1 and earlier, when you create a new site on IIS 6.0, a Web site identification number is randomly generated by using the name of the Web site. If you have administration scripts, setup programs, or provisioning scripts that depend upon the IIS 5.1 method of generating site identification numbers, you can force IIS 6.0 to use incremental site identification numbers by creating the **IncrementalSiteIDCreation** registry entry in the subkey HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\InetMgr\Parameters with the data type REG_DWORD and the value set to 0x1.

   For more information about configuring the Web site identification number see "Configure the Web Site Identification Number" in "IIS Deployment Procedures" in this book.

3. Create the Web site on the target server.

   Configure the Web site to have the same configuration as the corresponding Web site on the source server. For more information about how to create a Web site, see "Create a Web Site" in "IIS Deployment Procedures" in this book.

4. If the Web site on the source server is FrontPage extended, then configure the Web site on the target server to be FrontPage extended.

   For more information about how to configure a Web site to be FrontPage extended see "Configure a Web Site to be FrontPage Extended" in "IIS Deployment Procedures" in this book.

Creating Virtual Directories

For each virtual directory within each Web site on the source server, you must create a corresponding virtual directory on the target server. A *virtual directory* is a folder name, used in an address, which corresponds to a physical directory on the Web server or a Universal Naming Convention (UNC) location. This is also sometimes referred to as *URL mapping*. Virtual directories are used to publish Web content from any folder not located in the home directory of the Web site. When clients access content in a virtual directory, the content appears to be in a subdirectory of the home directory even though it is not.

For security reasons, you might want to move the Web site content to a different disk volume during the migration process. You can move the content to another disk volume on the target server or to a shared folder on a separate server. You can use virtual directories to specify the UNC name for the location where the content is placed, and provide a user name and password for access rights.

For each virtual directory in each Web site on the source server, create a corresponding virtual directory on the target server by completing the following steps:

1. Create the folder on the target server to contain the virtual directory content.

   Create the folder in the same location on the target server unless you are placing the content on a different disk volume than the source server or you are using a UNC share to store the content. Ensure that you create the folder in a secure manner that does not compromise the security of the target server.

   For more information about securing virtual directories, see "Preventing Unauthorized Access to Web Sites and Applications" in "Securing Web Sites and Applications" in this book.

2. Create the virtual directory under the appropriate Web site on the target server.

   For more information about how to create virtual directories, see "Create a Virtual Directory" in "IIS Deployment Procedures" in this book.

Migrating Web Site Content

For each Web site and virtual directory on the source server, you must migrate the content to the corresponding Web site and virtual directory on the target server. You can migrate the content from the source server to the target server by using one of the following methods:

- Run the **Xcopy** command to migrate Web site content to the target server on an intranet or internal network.
- Use Windows Explorer to migrate Web site content to the target server on an intranet or internal network.
- Use the **Copy Project** command in Microsoft Visual Studio® .NET to migrate Web site content to the target server on an intranet or internal network, if the application has been developed by using Visual Studio .NET.

Note

Front Page Server Extensions must be installed on the Web server to use the **Copy Project** command.

- Use the **Publish Web** command in FrontPage to migrate Web site content to the target server on an intranet or over the Internet, if the Web site has been developed using FrontPage.

For more information about how to publish Web site content on the target server by using FrontPage, see "Publish Web Site Content with FrontPage" in "IIS Deployment Procedures" in this book.

Configuring Web Site Application Isolation Settings

Based on the application isolation mode settings of the target server, you need to configure the application isolation settings for each migrated Web site. Configure the Web site application isolation settings so that the Web sites provide the highest possible security and availability.

Configure the Web site applications isolation settings by completing the following steps for each Web site on the source server:

1. Document the current application isolation settings for each Web site on the source server.
2. When the target server is configured for IIS 5.0 isolation mode, configure the target server to use the same the isolation settings as the source server.
3. When the target server is configured for worker process isolation mode, convert the isolation settings on the source server to application pool settings on target server.

Documenting the Current Application Isolation Settings on the Source Server

Before you configure the application isolation settings, document the existing application isolation settings of the Web sites and applications that are hosted on the source server. Later in the migration process, you will use these settings for configuring the application isolation mode for your Web sites and applications.

For each Web site and application currently running on the server, document the following:

Application isolation settings

Earlier versions of IIS can host Web sites and applications in pooled or isolated process configurations. For information about how to view the current application isolation mode, see “View Application Isolation Configuration” in “IIS Deployment Procedures” in this book.

If you are running IIS 4.0 on Windows NT Server 4.0, your applications are isolated in one of the following ways:

- In-process (running in-process with Inetinfo.exe)
- Isolated (running under MTS)

If you are running IIS 5.0 on Windows 2000, your applications are isolated in one of the following ways:

- In-process (running in-process with Inetinfo.exe)
- Pooled (running in the pooled COM+ application)
- Isolated (running in an isolated COM+ application)

Process identity that is used by the Web site or application

Each Web site or application configured in High isolation, or pooled isolation, uses a configurable *identity*. An identity is a user account that provides a security context for worker process servicing the Web site or application. The identity can be used to secure content, by using NTFS permissions or data, such as data stored in Microsoft SQL Server™. For more information about how to view the identity for each Web site or application, see "View Web Site and Application Process Identities" in "IIS Deployment Procedures" in this book.

Note

All Web sites and applications that are configured to run in the Inetinfo.exe process run under the security context of LocalSystem.

Configuring Application Isolation Settings in IIS 5.0 Isolation Mode

When the target Web server is configured to run in IIS 5.0 isolation mode, configure the application isolation settings on the target server identically to the settings on the source server. Web sites and applications on a Web server running in IIS 5.0 isolation mode can be configured with the following application isolation settings:

- Low (in-process).
- Medium (pooled).
- Low (isolated).

If the identity on the source server is an account local to the source server, you need to create a service account.

Configure the application isolation settings when IIS 6.0 is configured to run in IIS 5.0 isolation mode by completing the following steps:

1. Review the application isolation settings on the source server, documented earlier in the migration process.

   For more information about how the application isolation settings were documented, see "Documenting the Current Application Isolation Settings on the Source Server" earlier in this chapter.

2. Create any required local service accounts used for application isolation identities on the target server.

   When the application pool identity is a service account that is local to the source server, you need to create a new service account, or designate an existing service account, on the target server. Create the service account in Active Directory to:

   - Provide centralized administration of the account.
   - Provide stronger security because the account is stored in Active Directory rather than locally on the Web server.
   - Allow more than one Web server (for instance, in a Web farm) to use the same service account for the same instance of the application pool on other Web servers.

   For more information about how to create a service account to be used as an identity, see "Create a Service Account" in "IIS Deployment Procedures" in this book.

3. Configure the application isolation settings for the Web sites on the target server to be identical to the settings in Step 1.

   For more information about how to configure the application isolation settings for a Web site, see "Configure Application Isolation Settings for IIS 5.0 Isolation Mode" in "IIS Deployment Procedures" in this book.

4. Configure the application process identities for the Web sites on the target server to be identical to the settings in Step 1.

   For more information about how to configure the application process identities for the Web site, see "Configure Application Identity for IIS 5.0 Isolation Mode" in "IIS Deployment Procedures" in this book.

Configuring Application Isolation Settings in Worker Process Isolation Mode

When the target server is configured to use worker process isolation mode, you need to configure the application isolation settings to closely approximate their configuration in IIS 5.0 isolation mode by assigning them to *application pools*. An application pool is a grouping of one or more Web sites or applications served by one or more worker processes. You might need to apply additional configurations so that the applications retain their original isolation settings.

After converting to worker process isolation mode, all applications run in the preexisting application pool named "DefaultAppPool." If all of the applications run in the same process in the previous version of IIS, then they all are assigned to the default application pool.

However, if any one of the applications in the same application pool fails, the other applications can be adversely affected. For this reason it is recommended that you isolate your applications into separate application pools whenever possible.

Configure Web sites and applications to run in their own application pool by completing the following steps:

For each Web site or application configured in High isolation in IIS 5.0

1. Create a new application pool to be used by the Web site or application.

   For information about how to create application pools, see "Isolate Applications in Worker Process Isolation Mode" in "IIS Deployment Procedures" in this book.

2. If the Web site or application previously ran under an identity that is still required by the Web site or application, configure the application pool to use that same identity.

   For information about how to configure the identity for an application pool, see "Configure Application Pool Identity" in "IIS Deployment Procedures" in this book.

3. Assign the Web site or application to the new application pool.

   For information about how to assign the Web site to the new application pool, see "Isolate Applications in Worker Process Isolation Mode" in "IIS Deployment Procedures" in this book.

For each Web site or application configured in Low or Medium isolation in IIS 5.0

In earlier versions of IIS, applications ran in-process as DLLs in Inetinfo.exe (Low isolation) and the default process identity (account that the application runs under) was LocalSystem. With worker process isolation mode in IIS 6.0, applications never run in Inetinfo.exe. However, any applications that are not explicitly assigned to an application pool are assigned to the default application pool, which runs under the NetworkService process identity by default. Because LocalSystem has the same permissions and user rights as a member of the Administrators group, run Web sites and applications under the security context of the NetworkService account.

For each Web site or application that ran in Low or Medium isolation in IIS 5.0, do one of the following:

- When the Web site or application is able to function under the identity of the NetworkService account in the default application pool, continue to host the Web sites or applications in the default application pool, named "DefaultAppPool."
- When the Web site or application is unable to function under the identity of the NetworkService account in the default application pool, perform the following steps:

  1. Create a new application pool.
  2. Create a service account to be used as the identity for the application pool.

  For more information about how to create a service account to be used as an identity for an application pool, see "Create a Service Account" in "IIS Deployment Procedures" in this book.

  3. Configure the application pool identity to use the service account.

  For more information about how to configure the identity for an application pool, see "Configure Application Pool Identity" in "IIS Deployment Procedures" in this book.

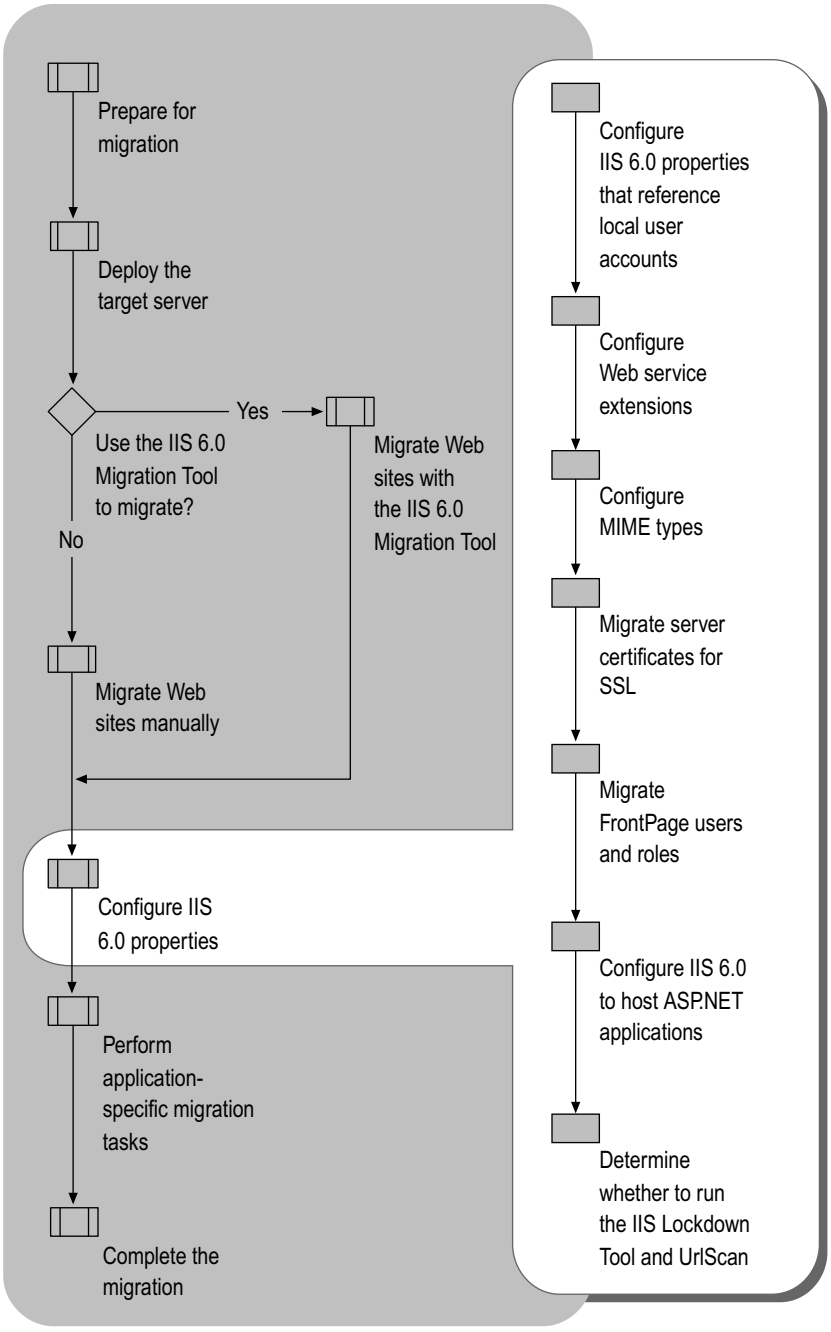
  4. Place the Web site or application in the new application pool.

Configuring IIS 6.0 Properties

Up to this point in the migration process, you have migrated the Web site content and configuration settings from the source server — either manually or with the IIS 6.0 Migration Tool. However, you might need to further configure the IIS 6.0 properties on the target server so that the Web sites run as they did before they were migrated. In addition, you should configure your target server even further to utilize the enhanced security and availability capabilities of IIS 6.0.

Figure 6.7 illustrates the process for configuring the IIS 6.0 properties on the target server.

Figure 6.7 Configuring IIS 6.0 Properties



Configuring IIS 6.0 Properties That Reference Local User Accounts

The configuration of IIS and the Web sites on the source server can reference user accounts that are stored in the local account database on the source server. These accounts that are stored locally on the Web server are known as *local user accounts*. Local user accounts are valid only on the Web server where they are stored, not on any other Web servers.

As a result, when IIS, or any of the Web sites on the source server, reference local user accounts you must configure IIS 6.0 and the Web sites on the target server to reference:

- Domain-based user accounts that you create.
- Local user accounts that you create on the target server.

For each configuration that references a local account on the source server, you need to do the following on the target server:

1. Create, or designate, a domain-based or local user account that you can use to configure IIS.

   For more information about creating a service account that is domain-based or local to the target server, see “Create a Service Account” in “IIS Deployment Procedures” in this book.

2. Modify the IIS 6.0 properties, Web site properties, or content configuration settings, based on the type of property that you are configuring.

   The types of IIS 6.0 properties, Web site properties, or content configuration settings that can reference or use local user accounts include:

   - **NTFS permissions assigned to Web content.** Grant the same NTFS permissions to the account created in step 1on the target sever as were granted to the local user account on the source server. For more information about granting NTFS permission to content, see “Configure NTFS Permissions” in “IIS Deployment Procedures” in this book.
   - **Anonymous accounts for Web sites.** Configure the anonymous account identity for a Web site to use the account created in Step 1 on the target server. For more information about configuring the anonymous account identity, see “Configure Anonymous User Identity” in “IIS Deployment Procedures” in this book.

- **Application isolation settings.** Configure the application isolation identity based on the application isolation mode configured for the server.

When the target server is configured to run in worker processor isolation mode, configure the identity properties of the application pools. For more information about configuring the application isolation settings when the target server is configured for worker process isolation mode, see "Configure Application Pool Identity" in "IIS Deployment Procedures" in this book.

When the target server is configured to run in IIS 5.0 isolation mode, configure the identity properties of the COM+ applications that correspond to the Web site. For more information about configuring the application isolation settings when the target server is configured for IIS 5.0 isolation mode, see "Configure Application Isolation Settings in IIS 5.0 Isolation Mode" in "IIS Deployment Procedures" in this book.

Configuring Web Service Extensions

Many Web sites and applications that are hosted on IIS 6.0 have extended functionality beyond static Web pages, including the generation of dynamic content. Providing dynamic content and other enhanced capabilities requires executable code, such as ASP, ASP.NET, and ISAPI extensions. The handlers that extend IIS functionality beyond serving static pages are known as *Web service extensions*.

If you installed IIS 6.0 as described earlier in this chapter, all Web service extensions are disabled by default. If you used another method to install IIS 6.0, such as using Manage Your Server, the configuration of IIS might be different.

Enabling all of the Web service extensions ensures the highest possible compatibility with your Web sites. However, enabling all of the Web service extensions creates a security risk because it increases the attack surface of IIS by enabling functionality that might be unnecessary for your server.

Web service extensions allow you to enable and disable the serving of dynamic content. *MIME types* allow you to enable and disable the serving of static content. For more information about enabling and disabling the serving of static content, see "Configuring MIME Types" later in this chapter.

Tip

If the appropriate Web service extension is not enabled, the Web server returns a 404 error to the client when attempting to serve the dynamic content. When the 404 error is returned as a result of a Web service extension not being enabled, a 404.2 error entry is placed in the IIS log. For more information about troubleshooting IIS 6.0, see "Troubleshooting" in IIS 6.0 Help, which is accessible from IIS Manager.

Configure the Web service extensions by completing the following steps:

1. Enable the essential predefined Web service extensions based on the information in Table 6.3.

Table 6.3 Predefined Web Service Extensions

Web Service Extension	Description
Active Server Pages	Enable this extension when one or more of the Web sites or applications contains ASP content.
ASP.NET version 1.1.4322	Enable this extension when one or more of the Web sites or applications contains ASP.NET content.
FrontPage Server Extensions 2002	Enable this extension when one or more of the Web sites are FrontPage extended.
Internet Data Connector	Enable this extension when one or more of the Web sites or applications uses the Internet Data Connector (IDC) to display database information (content includes .idc and .idx files).
Server-Side Includes	Enable this extension when one or more of the Web sites uses server-side include (SSI) directives to instruct the Web server to insert various types of content into a Web page.
WebDAV	Enable this extension when you want to support Web Distributed Authoring and Versioning (WebDAV) on the Web server, but it is not recommended for dedicated Web servers.

2. For each Web service extension that is used by your applications and is not a one of the default Web service extensions, add a new entry to the Web service extensions list and configure the status of the new entry to **Allowed**.

   For example, one of your applications might use an ISAPI extension to provide access to a proprietary database. Set the ISAPI extension used by the application to **Allowed** to explicitly grant it permission to run. For information about how to add a Web service extension to the list, see “Configure Web Service Extensions” in “IIS Deployment Procedures” in this book.

3. Use a Web browser on a client computer to verify that the Web sites and applications run on the server.

Configuring MIME Types

IIS 6.0 serves only the static files with extensions that are registered in the Multipurpose Internet Mail Extensions (MIME) types list. IIS 6.0 is preconfigured to recognize a default set of global MIME types, which are recognized by all configured Web sites. You can define MIME types at the Web site and directory levels, independently of one another or the types defined globally. IIS also allows you to change, remove, or configure additional MIME types. For any static content file extensions used by the Web sites hosted by IIS that are not defined in the MIME types list, you must create a corresponding MIME type entry.

Configure the MIME types after migration by completing the following steps:

1. For each static file type used by your Web site, ensure that an entry exists in the MIME types list.

   When your application uses the standard MIME types that are included in IIS 6.0, no new MIME types entry is required. For information about how to add a MIME type to the MIME types list, see "Configure MIME Types" in "IIS Deployment Procedures" in this book.

2. Use a Web browser on a client computer to verify that the Web sites and applications run on the server.

Migrating Server Certificates for SSL

If you use Secure Sockets Layer to encrypt confidential information exchanged between the Web server and the client, you must migrate the server certificate from the source server to the target server, install the certificate on the target server, and then configure the Web site to use the certificate.

Note

Server certificates are installed on the Web server and typically require no additional configuration on client servers. Server certificates allow clients to verify the identity of the server. Alternatively, some Web sites and applications might require client certificates. Client certificates are installed on the client servers and allow the server to authenticate the clients. For more information about configuring client certificates, see "About Certificates" in IIS 6.0 Help, which is accessible from IIS Manager.

Migrate the server certificate for SSL by completing the following steps for each Web site and application that uses SSL:

1. Export the server certificate for the Web site from the source server.

   For more information about exporting a server certificate, see "Export a Server Certificate" in "Deployment Procedures" in this book.

2. Install the server certificate to be used by the Web site on the target server.

   For more information about installing the server certificate on the Web server by using the Certificate MMC snap-in, see "Install a Server Certificate" in "Deployment Procedures" in this book.

3. Assign the server certificate to the Web site.

   For more information about assigning the server certificate to the Web site, see "Assign a Server Certificate to a Web Site" in "IIS Deployment Procedures" in this book.

Migrating FrontPage Users and Roles

When the source server has Web sites that are FrontPage extended and FrontPage roles have been assigned to the Web site users, you need to migrate the FrontPage roles to the target server. The FrontPage roles control the types of access that users have on FrontPage extended Web sites. FrontPage 2002 Server Extensions are administered through the Microsoft SharePoint™ Team Services HTML administration tool, which is installed with FrontPage 2002 Server Extensions.

The predefined FrontPage roles include the following:

- **Administrator.** Users assigned this role can view, add, and change all server content; and manage server settings and accounts.
- **Advanced author.** Users assigned this role can view, add, and change pages, documents, themes, and borders; and recalculate hyperlinks.
- **Author.** Users assigned this role can view, add, and change pages and documents.
- **Contributor.** Users assigned this role can view pages and documents, and view and contribute to discussions.
- **Browser.** Users assigned this role can view pages and documents.

In addition to the predefined FrontPage roles, custom FrontPage roles might be defined on the source server.

Migrate FrontPage users and roles by completing the following steps:

1. Identify the FrontPage roles on the source server and compare them to the FrontPage roles on the target server.
2. Create any FrontPage roles on the target server that exist on the source server but do not exist on the target server.
3. For each FrontPage user on the source server that is local to the source server, create a corresponding user on the target server, and then assign that user the same FrontPage roles that are assigned to the corresponding user on the source server.
4. For each FrontPage user on the source server that is in Active Directory, assign the user the same FrontPage roles on the target server.

For more information about configuring the FrontPage 2002 Server Extensions users and roles, see “Configure FrontPage Server Roles” in “IIS Deployment Procedures” in this book. For more information about administering FrontPage 2002 Server Extensions, see the SharePoint Team Services Administrator’s Guide link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources.

Configuring IIS 6.0 to Host ASP.NET Applications

If you migrated any ASP.NET applications to the target server, you need to configure IIS 6.0 to use the correct version of the .NET Framework, and you must configure the .NET Framework to support your applications.

Configure IIS 6.0 to host ASP.NET applications by completing the following steps:

1. Configure IIS 6.0 to use the correct version of the NET Framework.
2. Configure the .NET Framework.
3. Review how ASP.NET applications run in each application isolation mode.
4. Migrate Machine.config attributes to IIS 6.0 metabase property settings.

Configuring IIS to Use the Correct Version of the .NET Framework

If you have migrated ASP.NET applications that were developed with version 1.0 of the .NET Framework, you might have to install version 1.0 of the .NET Framework on the target server to ensure that your applications continue to function properly. After migration, version 1.1 of the .NET Framework is installed on the target server and the applications are configured to use version 1.1 of the .NET Framework. After installing version 1.0 of the .NET Framework, both version 1.0 and 1.1 of the .NET Framework are installed on the target server. This is referred to as *side-by-side support*.

Running versions 1.0 and 1.1 of the .NET Framework side-by-side is only supported when IIS is configured to run in IIS 5.0 isolation mode. If you have already configured IIS to run in worker process isolation mode, then you can only use version 1.1 of the .NET Framework on the target server. In most cases, ASP.NET applications function correctly with version 1.1 of the .NET Framework. For more information about possible application incompatibilities when migrating from version 1.0 to version 1.1 of the .NET Framework, see “Determining Application Compatibility with the .NET Framework” earlier in this chapter. When your ASP.NET application is incompatible with version 1.1 of the .NET Framework, configure the application to use version 1.0 of the .NET Framework and configure IIS to run in IIS 5.0 isolation mode.

You can configure each ASP.NET application to use a specific version of the .NET Framework by registering a *script map* in IIS for the application. A script map associates a file name extension and HTTP verb with the appropriate ISAPI for script handling. For example, when IIS receives a request for a .aspx file, the script map for the corresponding application directs IIS to forward the requested file to the appropriate version of the ASP.NET ISAPI for processing.

The script map for each ASP.NET application can be applied directly to an application, or inherited from a parent application. However, ASP.NET supports only one version of the .NET Framework for each application pool. For more information about how to configure the script map for an ASP.NET application, see “Configure an ASP.NET Application for ASP.NET” in “IIS Deployment Procedures” in this book.

Configuring the .NET Framework

The configuration method for the .NET Framework is determined by the application isolation mode that you use to configure IIS 6.0. Table 6.4 lists the methods for configuring the .NET Framework that are associated with each IIS 6.0 application isolation mode.

Table 6.4 Methods for Configuring the .NET Framework

Application Isolation Mode	Configuration Method for the .NET Framework
IIS 5.0 isolation mode	**Configured by making changes to the Machine.config file in the *systemroot*\Microsoft.NET\Framework*VersionNumber*\Config folder.**
Worker process isolation mode	**Configured by making changes to the IIS 6.0 metabase.**

When IIS 6.0 is configured to run in IIS 5.0 isolation mode, the .NET Framework uses the **<processModel>** section of the Machine.config file (in the *systemroot*\Microsoft.NET\Framework*versionNumber*\Config folder) for its configuration and no additional steps are required.

However, if you configured IIS 6.0 to run in worker process isolation mode, the .NET Framework ignores the **<processModel>** section of the Machine.config file, and instead gets its process configuration from the IIS 6.0 metabase. Because the migration process does not migrate the existing settings in the Machine.config file, you must manually convert any settings required by the ASP.NET applications.

For information about how to convert the Machine.config attribute settings to IIS 6.0 metabase property settings, see "Migrating Machine.config Attributes to IIS 6.0 Metabase Property Settings" later in this chapter. For more information about configuring IIS 6.0 for ASP.NET applications, see "Deploying ASP.NET Applications in IIS 6.0" in this book.

Reviewing How ASP.NET Applications Run in Each Application Isolation Mode

When running IIS 6.0 in worker process isolation mode, ASP.NET applications use the W3wp.exe worker process and application pool properties, which are stored in the IIS 6.0 metabase. When you configure IIS 6.0 to run in IIS 5.0 isolation mode, ASP.NET applications use the ASP.NET request processing model, Aspnet_wp.exe, and configuration settings. These configuration settings are stored in the Machine.config file.

Behavior of ASP.NET Applications That Are Running in IIS 5.0 Isolation Mode

By default, ASP.NET applications are configured to run in worker process isolation mode. If your application can only run in the ASP.NET process model, you must configure the server to run in IIS 5.0 isolation mode to be able to run the application on IIS 6.0. When IIS 6.0 is configured to run in IIS 5.0 isolation mode, ASP.NET applications behave as follows:

- The applications run within Aspnet_wp.exe.

  Aspnet_wp.exe is a request processing model that is similar to worker process isolation mode, and it contains worker process management capabilities similar to the WWW service in IIS 6.0. Aspnet_wp.exe includes most of the IIS application management features, such as recycling, health detection, and *processor affinity*. Processor affinity is the ability to force worker processes to run on specific microprocessors.

- The configuration settings are stored in the Machine.config file.

  When IIS 6.0 is running in IIS 5.0 isolation mode, the configuration settings for ASP.NET applications are managed by modifying the Machine.config file, not the IIS metabase file. Because there is no administrative console for the Machine.config settings, any configuration settings for ASP.NET must be made directly to the Machine.config file.

Important

> In IIS 5.0 isolation mode, the .NET Framework ignores any configuration changes made in the IIS metabase. Administrative consoles, such as IIS Manager, make changes to the IIS 6.0 metabase, but those changes are not read by the .NET Framework.

When IIS 6.0 is configured to run in IIS 5.0 isolation mode, your ASP.NET applications should behave as they did in IIS 5.0. However, incompatibilities can result when running version 1.1 of the .NET Framework. For more information about configuring IIS to support ASP.NET applications that use version 1.0 of the .NET Framework, see “Running Different Versions of ASP.NET Side-by-Side” in “Deploying ASP.NET Applications in IIS 6.0” in this book.

Behavior of ASP.NET Applications That Are Running in Worker Process Isolation Mode

When IIS 6.0 is configured to run in worker process isolation mode, ASP.NET applications behave as follows:

- The process model within the ASP.NET ISAPI extension is disabled, and ASP.NET applications run using worker process isolation mode in IIS 6.0.

  In this configuration, the ASP.NET application runs in worker process isolation mode like any other application, such as an ASP application. In addition, IIS 6.0 provides all of the management features such as recycling, health detection, and processor affinity.

- The ASP.NET ISAPI extension is configured by a combination of configuration settings that are stored in the IIS metabase (MetaBase.xml) and configuration settings in the Machine.config file.

  When IIS 6.0 is running in worker process isolation mode, the majority of the application configuration settings are stored in the IIS metabase. You can adjust these settings directly in MetaBase.xml or from administrative consoles, such as IIS Manager, or scripts.

However, if there are existing settings in the **<processModel>** section of the Machine.config file, those configuration settings must be converted to the appropriate application pool settings when the Web server is configured to run in worker process isolation mode Additionally, there are other configuration settings that are still modified in the Machine.config file, regardless of the application isolation mode. For more information about converting Machine.config attributes to worker process isolation mode settings, see "Migrating Machine.config Attributes to IIS 6.0 Metabase Property Settings" later in this chapter.

When IIS 6.0 is configured in worker process isolation mode, your ASP.NET applications should behave as they did on IIS 5.0. Before deploying your ASP.NET applications on your production Web servers, test compatibility with IIS 6.0 running in worker process isolation mode and version 1.1 of the .NET Framework. For more information about determining application compatibility with IIS 6.0 running in worker process isolation mode, see "Determining Application Compatibility with Worker Process Isolation Mode" earlier in this chapter. For more information about determining compatibility with version 1.1 of the .NET Framework see "Determining Application Compatibility with the .NET Framework" earlier in this chapter.

If you determine that your ASP.NET applications are incompatible with worker process isolation mode, reconfigure IIS to run in IIS 5.0 isolation mode. If your ASP.NET applications are incompatible with version 1.1 of the .NET Framework, configure IIS to use version 1.0 of the .NET Framework with your ASP.NET application.

For more information about configuring IIS 6.0 to run in IIS 5.0 isolation mode, see "Configure Application Isolation Modes" in "IIS Deployment Procedures" in this book. For more information about configuring IIS to use version 1.0 of the .NET Framework with your ASP.NET application, see "Running Different Versions of ASP.NET Side-by-Side" in "Deploying ASP.NET Applications in IIS 6.0" in this book.

Migrating Machine.config Attributes to IIS 6.0 Metabase Property Settings

When IIS 6.0 is configured to run in IIS 5.0 isolation mode, the .NET Framework uses the **<processModel>** section of the Machine.config file (in the *systemroot*\Microsoft.NET\Framework*versionNumber*\Config folder) for its runtime configuration. When the target server is running in IIS 5.0 isolation mode, you do not need to convert the attribute configuration settings on the source server to their equivalent IIS 6.0 metabase property settings on the target server so you can proceed to the next step in the migration process. To proceed to the next step in the migration process, see "Determining Whether to Run the IIS Lockdown Tool and UrlScan" later in this chapter.

However, if you configured IIS 6.0 to run in worker process isolation mode, the .NET Framework ignores the **<processModel>** section of the Machine.config file, and instead gets its process configuration from the IIS 6.0 metabase. Because the migration process does not migrate the existing settings in the Machine.config file, you must manually migrate any settings that are required by your ASP.NET applications.

For information about how to migrate the Machine.config settings to IIS 6.0 metabase settings, see "Migrating Machine.config Attributes to IIS 6.0 Metabase Property Settings" in "Upgrading an IIS Server to IIS 6.0" in this book.

Determining Whether to Run the IIS Lockdown Tool and UrlScan

The IIS Lockdown Tool and UrlScan are IIS security related programs designed for IIS 5.1 and earlier. Each tool provides different types of protection for earlier versions of IIS. The IIS migration process does not install the IIS Lockdown Tool and UrlScan on the target server.

IIS Lockdown Tool

The IIS Lockdown Tool is provided to assist administrators in configuring optimal security settings for existing IIS servers. You cannot install the IIS Lockdown Tool after migration because all of the default configuration settings in IIS 6.0 meet or exceed the security configuration settings made by the IIS Lockdown Tool.

UrlScan

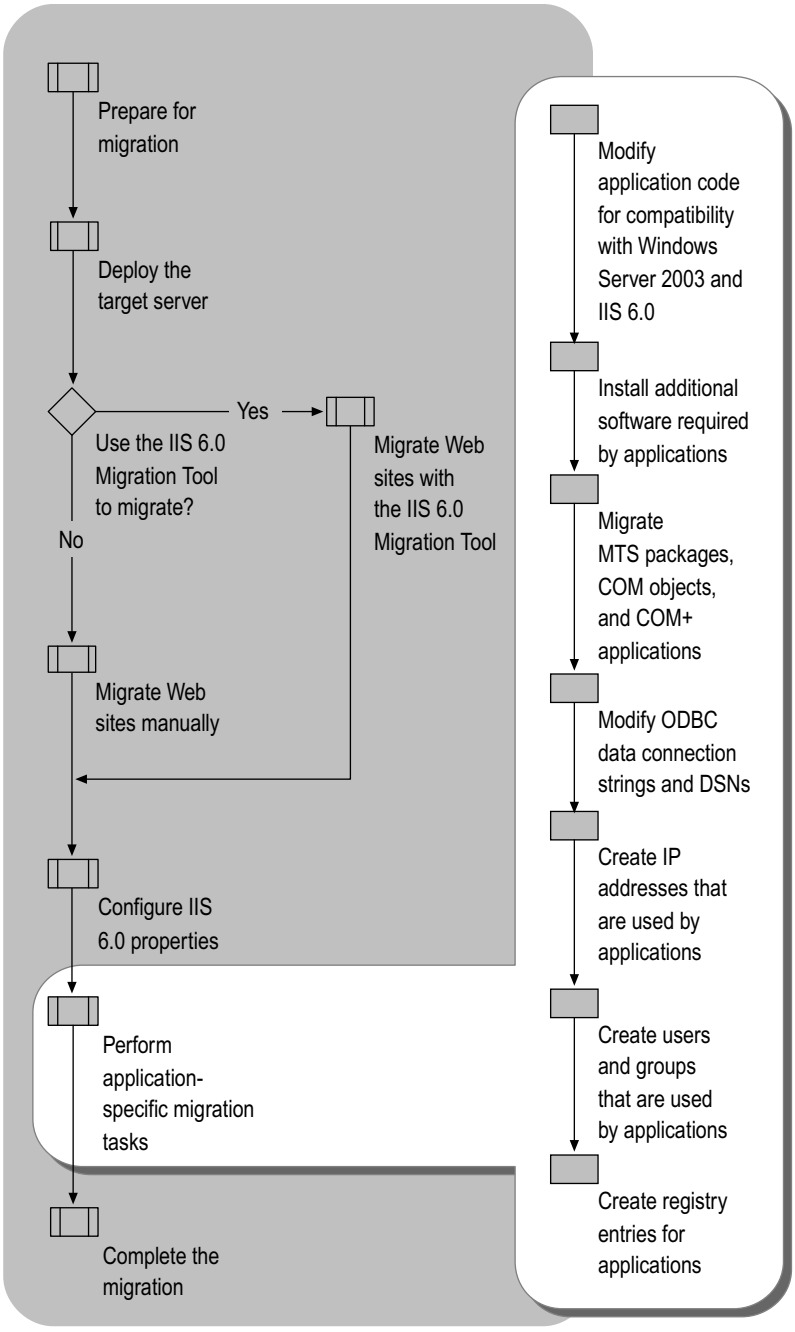
UrlScan is a tool that is provided to reduce the attack surface of Web servers running earlier versions of IIS. By default, IIS 6.0 has features that significantly improve security by reducing the attack surface of the Web server. UrlScan provides flexible configuration for advanced administrators, while maintaining the improved security in IIS 6.0. When you need this flexibility in configuring your Web server, you can run UrlScan on IIS 6.0.

For more information about determining whether to run UrlScan after migrating your server to IIS 6.0, see the Using UrlScan link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources.

Performing Application-Specific Migration Tasks

When no setup or provisioning scripts exist for your applications, you might need to perform additional application-specific migration tasks, whether you have migrated content and metabase information manually or with the IIS 6.0 Migration Tool. Some of these tasks involve the modification of application code and might require the assistance of the application developers. Depending on your application, you might need to perform any combination of the steps in Figure 6.8.

Figure 6.8 Performing Application-Specific Migration Tasks



Modifying Application Code for Compatibility with Windows Server 2003 and IIS 6.0

If any application code needs to be changed or recompiled to run on Windows Server 2003, you must make those changes manually. The most common application code changes include the following:

- Code that references Windows platform components or APIs no longer supported in Windows Server 2003.
- Code that references IIS metabase properties that have changed or are no longer supported in IIS 6.0.
- Code that is incompatible with worker process isolation mode.

Modifying References to Windows Platform Components and APIs No Longer Supported in Windows Server 2003

Applications developed to run on earlier versions of Windows server operating systems and IIS can call APIs that are not supported in Windows Server 2003 and IIS 6.0. Usually these APIs are replaced with newer APIs that provide additional functionality.

If your Web sites and applications use an API that is not supported in Windows Server 2003 and IIS 6.0, you must modify your code to use an API that provides the same functionality and is supported in Windows Server 2003 and IIS 6.0.

For example, the APIs supported by the Collaboration Data Objects for Windows NT Server 4.0 (CDONTS) DLL are not supported in Windows Server 2003. The functionality provided by CDONTS is supported by Collaboration Data Objects for Windows 2000 (CDOSYS), which, in turn, is supported by Windows Server 2003. Therefore, if your applications use CDONTS, you can modify your code to use CDOSYS instead. For more information about modifying your application to use CDOSYS, see the Collaboration Data Objects Roadmap link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources.

Modifying References to IIS 6.0 Metabase Properties

Some metabase properties that were used to configure features in earlier versions of IIS are no longer supported in IIS 6.0. Because some features are eliminated, or implemented differently in IIS 6.0, the corresponding unused metabase properties are not referenced by any code in IIS 6.0. In cases where the feature is implemented differently in IIS 6.0, new metabase properties have been created to replace the obsolete, or unused, properties.

In addition, there is one IIS 5.0 metabase property — **CPUResetInterval** — whose behavior has changed because of architectural changes made to IIS 6.0.

To determine whether any of your Web sites, applications, or setup programs reference these changed or unsupported IIS metabase properties, see “Changes to Metabase Properties in IIS 6.0” in this book. You can then follow the recommendations associated with each changed metabase property listed in this appendix to accommodate functionality changes in IIS 6.0.

Tip

The metabase properties that are no longer supported in IIS 6.0 are not available in IIS 6.0, even when IIS 6.0 is configured to run in IIS 5.0 isolation mode.

Modifying Applications To Be Compatible with Worker Process Isolation Mode

Most applications that were developed to run on earlier versions of IIS run in worker process isolation mode without modification. However, you might need to modify your applications to make them compatible with worker process isolation mode. For more information about the types of modifications that you might need to make to your applications, see “Application Changes Required for Worker Process Isolation Mode” earlier in this chapter.

Your applications can require other modifications that are not described in this chapter because of the myriad of approaches to developing applications. For the most current information about modifying your applications to be compatible with worker process isolation mode, see the MSDN Online link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources, and search for relevant articles.

Installing Additional Software Required by Applications

Some of your applications might require or be dependent upon additional software that is not in the same folder with the Web site content. This software can be developed by your organization, by Microsoft, or by other organizations. You must install this software on the target server by running the setup or installation program that accompanies the software.

Examples of this type of software include software that creates reports, integrates databases with dynamic content generation, or provides connectivity with applications running on other servers with non-Microsoft operating systems.

Migrating MTS Packages, COM Objects, and COM+ Applications

In addition to installing any software that you installed earlier in the migration process, there can be MTS packages, COM objects, and COM+ applications that need to be migrated. Table 6.5 lists the tasks that you need to complete to migrate MTS packages, COM objects, and COM+ application from earlier version of Windows.

Table 6.5 Migrating MTS Packages, COM Objects, and COM+ Applications

Source Server	Target Server	Migration Tasks
MTS packages on Windows NT 4.0 Server	**COM + applications**	**Convert the MTS package to a COM+ application.**
COM objects	**COM objects**	**Copy COM object files (.dll or .exe) from the source server to the target sever and register the COM object by running the** regsrv32 **command.**
COM+ applications on Windows 2000 Server	**COM + applications**	**Create the COM+ application on the target server and use the same configuration on the target server that existed on the source server.**

For more information about migrating MTS packages, COM objects, and COM+ applications to Windows Server 2003, see the following articles: “COM: Delivering on the Promises of Component Technology”, which you can find on the Microsoft Web site at http://www.microsoft.com; and “Microsoft .NET/COM Migration and Interoperability”, which you can find by seeing the MSDN Online link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources.

An example of registering a COM object that is running on the source server is CDONTS. If you decide to not modify your code to us CDOSYS instead of CDONTS, you must copy the Cdonts.dll file from the source server to the target server and then register Cdonts.dll with Windows Server 2003. For more information about how to migrate CDONTS, see “Migrate CDONTS” in “IIS Deployment Procedures” in this book.

Modifying ODBC Data Connection Strings and DSNs

If your application establishes database connectivity through an ODBC data connection string or through an ODBC data source name (DSN), you must do any combination of the following:

- Manually modify ODBC DSNs when the ODBC data connection string references a database that is stored on the source server and the database has not been migrated to the target server.
- Manually create a system ODBC DSN on the target server for each system ODBC DSN on the source server.

Modifying ODBC DSNs

ODBC data connection strings might need to be modified if they reference the database or are migrated to a different location. For more information about modifying ODBC data connection strings, see the article "Connection String Format and Attributes." To find this article, see the MSDN Online link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources, and then search for "ODBC connection string".

Creating System ODBC DSNs

In addition, if you have system ODBC DSNs that are defined on the source server, you need to create corresponding ODBC DSNs on the target server. Table 6.6 lists methods for administering ODBC DSNs on different Windows server operating systems. You can administer ODBC DSNs on the source server and the target server by using the methods listed in this table.

Table 6.6 Administering ODBC DSNs on Source Servers and Target Servers

Operating System	Administered Through	Additional Information
Windows Server 2003	Data Sources (ODBC) in Administrative Tools	For more information about using Data Sources (ODBC) in Windows Server 2003, see Help and Support Center for Windows Server 2003.
Windows 2000 Server	Data Sources (ODBC) in Administrative Tools	For more information about using Data Sources (ODBC) in Windows 2000 Server, see Windows 2000 Server Help.
Windows NT 4.0 Server	ODBC Data Sources in Control Panel	For more information about using ODBC Data Sources in Windows NT 4.0 Server, see Help in the ODBC Data Source Administrator in the ODBC Data Sources, in Control Panel.

Creating IP Addresses That Are Used by Applications

You can uniquely identify a Web site or application by associating the Web site or application with a unique IP address, a unique combination of an IP address and a TCP port number, or host headers. For Web sites and applications on the source server that are uniquely identified by IP addresses, you must create corresponding IP addresses on the target server and then configure the Web sites and applications to use the newly created IP addresses.

To create IP addresses that are used by applications, complete the following steps:

1. Determine which Web sites and applications on the source server are uniquely identified by IP addresses.

   For more information about how to determine which Web sites and applications on the source server are uniquely identified by IP addresses see, “Determine Web Sites Uniquely Identified by IP Addresses” in “IIS Deployment Procedures” in this book.

2. For each Web site and application identified in Step 1, assign new a new IP address to the TCP/IP properties of the network adapter in the target server that the clients use to access the Web sites and applications.

   For more information about how to assign a new IP addresses to the network adapter in the target server that the clients use to access the Web sites and applications see, “Assign Additional IP Addresses to a Network Adapter” in “IIS Deployment Procedures” in this book.

3. Configure the migrated Web sites and applications on the target server to use the IP addresses assigned in Step 2.

   For more information about how to configure the IP addresses assigned to Web sites and applications see, “Configure IP Address Assigned to Web Sites” in “IIS Deployment Procedures” in this book.

Creating Users and Groups That Are Used by Applications

The Web sites and applications on the source server might be accessed by accounts that are local to the source server. The local user and group accounts need to be created on the target server so that the accounts can access the Web sites and applications on the target server.

In cases where the users and groups that are used by the applications exist in Active Directory, no steps are required and you can continue to the next migration step. To continue to the next step in the migration process, see “Creating Registry Entries for Applications” later in this chapter.

Create the users and groups that are used by applications and are local to the source server by completing the following steps:

1. Identify the users and groups that are local to the source server.
2. For each group on the source server, create a corresponding group on the target server.
3. For each user on the source server, create a corresponding user on the target server.
4. For each user created in Step 3, assign the user to the same groups as the corresponding user on the source server.

   For more information about viewing the users and groups on the source server, see "Creating user and group accounts" in Windows 2000 Server Help and Windows NT Server 4.0 Help. For more information about creating users and groups in Windows Server 2003, see "Create a Service Account" in "IIS Deployment Procedures" in this book.
5. For each user created in Step 3, assign the same user rights assigned to the corresponding user on the source server.

   For more information about assigning user rights to a user, see "Grant User Rights to a Service Account" in "IIS Deployment Procedures" in this book.
6. Assign the same NTFS permissions for the content on the target server as the NTFS permissions for the content on the source server.

   For more information about configuring NTFS permissions on the target server, see "Configure NTFS Permissions" in "IIS Deployment Procedures" in this book.

Creating Registry Entries for Applications

Some applications save configuration information in the Windows registry. If the setup program or provisioning script creates the registry entries, run the setup program or provisioning script on the target server. Otherwise, you must manually identify the registry entries and then re-create them on the target server.

Caution

Do not edit the registry unless you have no alternative. The registry editor bypasses standard safeguards, allowing settings that can damage your system, or even require you to reinstall Windows. If you must edit the registry, back it up first and see the Registry Reference on the *Microsoft Windows Server 2003 Deployment Kit* companion CD or on the Web at http://www.microsoft.com/reskit.

Create application registry entries on the target server manually by completing the following steps:

1. Identify the registry entries required by the applications that are currently running on the source server.

   Earlier in the migration process, you identified the registry entries required by the applications on the source server. For more information about identifying registry entries, see "Identifying Which Web Site and Application Components to Migrate", earlier in the chapter.

2. Back up the registry entries on the source server that you identified in the previous step by using the registry editor Regedit.exe.

   For more information about backing up copies of registry entries, see "Back Up and Restore Registry Entries" in "IIS Deployment Procedures" in this book.

3. Copy the .reg backup file, created in Step 2, to the target server.

4. Modify the .reg backup file created in the previous step to accommodate for changes in disk volume letters, such a F:, or paths to folders by completing the following steps:

   a. In Notepad, open the .reg backup file created in Step 2.

   b. Search for specific references to disk volume letters or paths that you want to change, and update them to reflect the disk volume letters or paths on the target server.

   c. Save the .reg file.

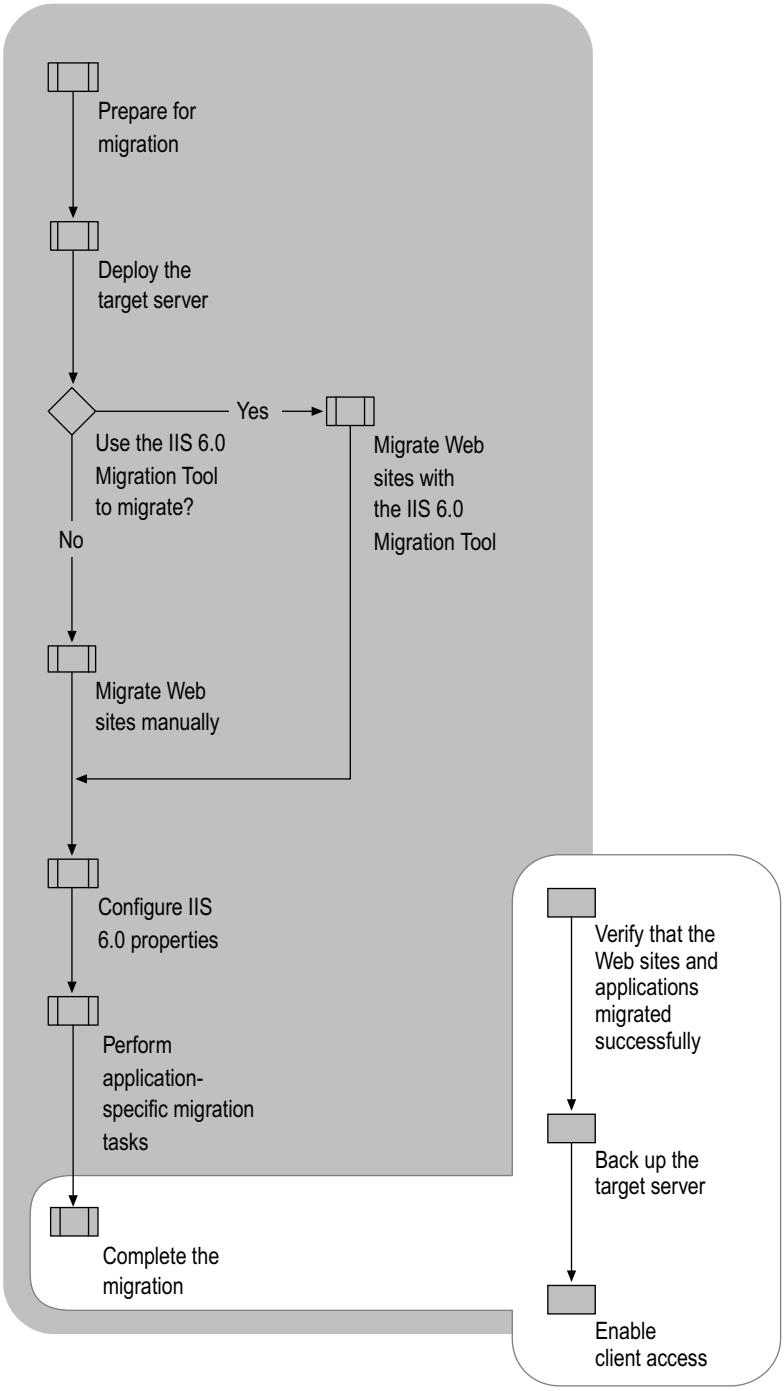
5. Restore the registry backup on the target server.

For more information about restoring registry entries, see "Back Up and Restore Registry Entries" in "IIS Deployment Procedures" in this book.

Completing the Migration

At this point in the process, you have migrated your Web sites and applications to the target server and configured the IIS 6.0 properties to settings on the source server. Now you need to verify that the migration completed successfully, capture the current configuration of the target server, and enable client access to the target server. After you complete these last steps, your migration is complete. Figure 6.9 illustrates the process for completing the migration to IIS 6.0.

Figure 6.9 Completing the Migration to IIS 6.0



Verifying That the Web Sites and Applications Migrated Successfully

Before deploying the target server to a production environment, verify that the Web site content and configuration information migrated successfully by completing the following steps:

1. Review the system log in Windows Server 2003 on the target server to determine whether any of the Web sites did not start.

   IIS 6.0 creates entries in the system log when a Web site fails to start for any reason. Search the System log on the target server for to determine if any errors occurred. For more information about how to troubleshoot Web sites that fail to start, see "Troubleshooting" in IIS 6.0 Help, which is accessible from IIS Manager.

2. Verify that the Web site content migrated to the target server.

   Compare the files and folders for each Web site on the target server with the original files and folders on the source server to determine whether the Web site content has been migrated correctly. Ensure that the number and size of the files and folders on the target server approximates the number and size of the same files on the source server.

3. Verify that the Web site configuration migrated to the target server.

   View the Web site configuration information for a random sampling of the Web sites that were migrated. Compare the Web site configuration on the target server with the corresponding Web site configuration on the source server. Ensure that the configuration for the Web sites has been migrated correctly.

4. Perform functional testing of the migrated Web sites and applications to ensure that the Web sites and applications behave as expected.

   You can find possible causes of application failure by reviewing the Windows Server 2003 and IIS 6.0 Migration Tool logs and application configuration, but the only way to accurately assess whether your Web sites and applications migrated successfully is to perform functional testing. Functional testing is designed to ensure that Web sites and applications are functioning correctly in the most common usage scenarios (for example URLs and inputs).

Procedures for performing functional testing of Web sites and applications are beyond the scope of this chapter. For more complete information about general testing, see the MSDN Online link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources, and search for "testing".

Backing Up the Target Server

Before you enable client access to the target server, perform a complete image backup of the target server. Performing this image backup provides you with a point-in-time snapshot of the Web server. If you need to restore the target server in the event of a failure, you can use this backup to restore the Web server to a known configuration.

Important

Do not proceed further unless you have a successful backup of the entire target server. Otherwise, you can lose Web sites, applications, or data that you migrated to the target server.

For more information about how to back up the Web server, see “Back Up and Restore the Web Server to a File or Tape” in “IIS Deployment Procedures” in this book.

Enabling Client Access

After you have migrated your Web sites from the source server to the target server, you are ready to enable client access to the Web sites on the target server while maintaining the DNS entries to the source servers. After a period of time that meets your business needs, you can remove the DNS entries that point to the Web sites on the source server.

Enable client access to the Web sites that are on the target server by completing the following steps:

1. Create the appropriate DNS entries for the Web sites and applications running on the target server.

   For more information about how to create DNS entries for your Web sites and applications see “Managing resource records” in Help and Support Center for Windows Server 2003.

2. Monitor client traffic to determine whether clients are successfully accessing the target server.

   For more information about how to monitor client traffic to Web sites on the target server, see “Monitor Active Web and FTP Connections” in “IIS Deployment Procedures” in this book.

3. Establish a monitoring period, such as a few hours or a day, to confirm that clients accessing Web sites on the target server are experiencing response times and application responses that meet or exceed your requirements.

4. Remove the DNS entries that are pointing to the Web sites and applications on the source server.

   For more information about how to remove DNS entries for your Web sites and applications, see "Managing resource records" in Help and Support Center for Windows Server 2003.

5. Prevent new clients from accessing the Web sites on the source server by pausing the Web sites on the source server.

   For more information about how to pause Web sites on the source server, see "Pause Web or FTP Sites" in "IIS Deployment Procedures" in this book.

6. Monitor client traffic to the Web sites on the source server to determine when clients are no longer accessing the source server.

   For more information about how to monitor client traffic to Web sites on the source server, see "Monitor Active Web and FTP Connections" in "IIS Deployment Procedures" in this book.

7. When clients are no longer accessing the Web sites on the source server, decommission the hardware for the source server.

Additional Resources

These resources contain additional information and tools related to this chapter.

Related Information

"Deploying ASP.NET Applications in IIS 6.0" in this book for information about configuring IIS 6.0 for ASP.NET applications.

"IIS Deployment Procedures in this book for information about specific procedures for migrating IIS Web sites to IIS 6.0.

"Migrating Apache Web Sites to IIS 6.0" in this book for information about migrating Apache Web sites.

"Migrating Machine.config Attributes to IIS 6.0 Metabase Property Settings" in "Upgrading an IIS Server to IIS 6.0" in this book for information about how to migrate the Machine.config settings to IIS 6.0 metabase settings.

"Securing Web Sites and Applications" in this book for information about improving the security of your IIS Web sites after migration.

"Designing a Test Environment" in *Planning, Testing, and Piloting Deployment Projects* of the *Microsoft® Windows® Server 2003 Deployment Kit* for information about setting up a test lab.

"COM: Delivering on the Promises of Component Technology", which you can find on the Microsoft Web site at http://www.microsoft.com, for information about migrating MTS packages, COM objects, and COM+ applications to Windows Server 2003.

The Collaboration Data Objects Roadmap link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources for information about modifying your applications to use CDOSYS.

The Compatibility Considerations and Version Changes link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources for a current list of possible compatibility issues when upgrading from version 1.0 to version 1.1 of the .NET Framework.

The *Hardware Compatibility List* on the product CD-ROM or the Hardware Driver Quality link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources for information about the hardware devices supported by Windows Server 2003.

"Microsoft .NET/COM Migration and Interoperability", which you can find by seeing the MSDN Online link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources, for information about migrating MTS packages, COM objects, and COM+ applications to Windows Server 2003.

The SharePoint Team Services Administrator's Guide link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources for information about administering FrontPage 2002 Server Extensions.

The Using UrlScan link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources for information about determining whether to run UrlScan after migrating your server to IIS 6.0.

The Windows Server 2003 link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources for the latest information about compatibility with Windows Server 2003.

Related IIS 6.0 Help Topics

- "About Certificates" in IIS 6.0 Help, which is accessible from IIS Manager, for information about configuring client certificates.
- "Metabase Edit-While-Running Feature" in IIS 6.0 Help, which is accessible from IIS Manager, for information about the edit-while-running feature.
- "Troubleshooting" in IIS 6.0 Help, which is accessible from IIS Manager, for information about troubleshooting IIS 6.0.

Related Windows Server 2003 Help Topics

For best results in identifying Help topics by title, in Help and Support Center, under the **Search** box, click **Set search options**. Under **Help Topics**, select the **Search in title only** check box.

- "Features unavailable on 64-bit versions of the Windows Server 2003 family" in Help and Support Center for Windows Server 2003 for information about features, such as ASP.NET, that are not supported on 64-bit versions of Windows Server 2003.
- "Managing resource records" in Help and Support Center for Windows Server 2003 for information about how to create DNS entries for your Web sites and applications.

Related Tools

- The "IIS 6.0 Migration Tool User Guide" on the *Windows Server 2003 Deployment Kit* companion CD or the *Internet Information Services (IIS) 6.0 Resource Kit* companion CD for information about using the IIS 6.0 Migration Tool.

The Windows Application Compatibility link on the Web Resources page at http://www.microsoft.com/windows/reskits/webresources to download the latest version of the Windows Application Compatibility Toolkit.