Novell NetWare® 6.5

www.novell.com

APACHE WEB SERVER ADMINISTRATION GUIDE

August 25, 2003

Novell®

Legal Notices

Novell, Inc. makes no representations or warranties with respect to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

Further, Novell, Inc. makes no representations or warranties with respect to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to make changes to any and all parts of Novell software, at any time, without any obligation to notify any person or entity of such changes.

You may not export or re-export this product in violation of any applicable laws or regulations including, without limitation, U.S. export regulations or the laws of the country in which you reside.

U.S. Patent No. 5,157,663; 5,349,642; 5,455,932; 5,553,139; 5,553,143; 5,572,528; 5,594,863; 5,608,903; 5,633,931; 5,652,854; 5,671,414; 5,677,851; 5,692,129; 5,701,459; 5,717,912; 5,758,069; 5,758,344; 5,781,724; 5,781,733; 5,784,560; 5,787,439; 5,818,936; 5,828,882; 5,832,274; 5,832,275; 5,832,483; 5,832,487; 5,859,978; 5,870,561; 5,870,739; 5,873,079; 5,878,415; 5,884,304; 5,893,118; 5,903,650; 5,903,720; 5,905,860; 5,910,803; 5,913,025; 5,913,209; 5,915,253; 5,925,108; 5,933,503; 5,933,826; 5,946,002; 5,946,467; 5,956,718; 5,956,745; 5,964,872; 5,974,474; 5,983,223; 5,983,234; 5,987,471; 5,991,810; 6,002,398; 6,014,667; 6,016,499; 6,023,586; 6,029,247; 6,052,724; 6,061,726; 6,061,740; 6,061,743; 6,065,017; 6,081,774; 6,081,814; 6,094,672; 6,098,090; 6,105,062; 6,105,069; 6,105,132; 6,115,039; 6,119,122; 6,144,959; 6,151,688; 6,157,925; 6,167,393; 6,173,289; 6,216,123; 6,219,652; 6,233,859; 6,247,149; 6,269,391; 6,286,010; 6,308,181; 6,314,520; 6,324,670; 6,338,112; 6,345,266; 6,353,898; 6,424,976; 6,466,944; 6,477,583; 6,477,648; 6,484,186; 6,496,865; 6,510,450; 6,516,325; 6,519,610; 6,532,451; 6,532,491; 6,539,381; RE37,178. Patents Pending.

Novell, Inc.
1800 South Novell Place
Provo, UT 84606
U.S.A.

www.novell.com

Apache Web Server Administration Guide for NetWare 6.5

July 1, 2003

Online Documentation: To access the online documentation for this and other Novell products, and to get updates, see www.novell.com/documentation.

Novell Trademarks

Third-Party Trademarks

About This Guide

This guide describes how to install, configure, and manage the Apache Web Server on NetWare® using Apache Manager.

HINT: If you are already familiar with the Apache Web server, you can manage it in the same way you do on other platforms: by manually modifying Apache directives the httpd.conf file. Refer to the official Apache HTTP Server Version 2.0 Documentation (http://httpd.apache.org/docs-2.0) on the Apache Web site.

The procedures in this manual document how to configure Apache using Apache Manager. Because Apache Manager uses all of the same Apache directives you use when manually editing the httpd.conf configuration file, hypertext links to the official online Apache documentation set are included throughout this manual. These links are intended to lead you to additional information about Apache directives, including how and why each directive is used. This information can help you understand the effects of the changes you make using Apache Manager.

This guide is intended for Web or network administrators who install, configure, and manage the Apache Web server on NetWare. Developers might also find the information to be helpful. It is divided into the following chapters:

- Chapter 1, "Overview: Apache Web Server," on page 7
- Chapter 2, "Installation and Configuration," on page 15
- Chapter 3, "Managing Apache Web Server Preferences," on page 23
- Chapter 4, "Managing Server Content," on page 37
- Chapter 5, "Managing Apache Modules," on page 49
- Chapter 6, "Managing Multiple Instances of Apache," on page 59

Additional Documentation

Refer to the following online resources for official Apache documentation and related information:

- Apache 2.0 Documentation (http://httpd.apache.org/docs-2.0)
- Apache Quick Reference Card (http://www.refcards.com)

Also, a copy of the official Apache documentation set is installed to your server in the *volume*:\Apache2\manual directory. You can access it using a Web browser after you have installed NetWare 6.5. Use your server's URL with /manual at the end of it. For example,

```
http://myserver.mycompany.com/manual
```

Documentation Conventions

In this documentation, a greater-than symbol (>) is used to separate actions within a step and items within a cross-reference path.

Also, a trademark symbol (®, ™, etc.) denotes a Novell trademark. An asterisk (*) denotes a third-party trademark.

1 Overview: Apache Web Server

The Apache Web server is the Web server of choice for more than 67% of all Web servers being used on the World Wide Web today. Its popularity comes from the fact that it is the most reliable and secure Web server available. It is open source software, created by the Apache Foundation, a conglomerate of technical professionals from all over the world.

Apache runs on all major platforms and is capable of hosting even the most complex Web sites and can scale to handle thousands of simultaneous connections.

This overview includes the following topics:

- "Benefits of Apache on NetWare" on page 7
- "How Apache Is Used on NetWare" on page 8
- "What's Different with Apache on NetWare" on page 9
- "Apache Manager: A Web-based Administration Tool" on page 11
- "Hosting a Web Site" on page 11
- "Using Servlets with Apache on NetWare" on page 12
- "Hosting Web Services and Applications" on page 12
- "What's Next" on page 13

Benefits of Apache on NetWare

Apache provides many business benefits to your existing network that together can increase productivity, improve communication between departments and employees and, when used in conjunction with the Novell® exteNd™ Application Server, turn your legacy applications and processes into integrated solutions that speed up your business.

HINT: If you install Apache as part of the Novell AMP (Apache, MySQL*, PHP, Perl) pre-configured server installation option, you can choose from thousands of ready-to-run applications available from the World Wide Web from such Web sites as hotscripts.com. For more information about AMP, see NetWare AMP (Apache, MySQL, PHP, Perl) Server*NetWare 6.5 Web and Application Services Overview*.

Here are some of the key uses and benefits of using Apache on NetWare:

- Provides a highly reliable and fast Web server for hosting simple or complex Web sites, which can be used as
  - A method for securely sharing department- or company-wide information for use by employees and business partners, regardless of where they are located
  - A corporate Web server for hosting your company Web site on the World Wide Web
  - A method for sharing project information and improving team collaboration
  - A method for sharing company policies and procedures

- Offers tight integration with Novell® eDirectory™ and Secure Sockets Layer (SSL) through the use of a customized NetWare specific Apache module, providing a highly secure method for sharing sensitive company information over the Internet
- Has an easy-to-use graphical user interface that lets you
  - Manage the Apache Web server
  - Manage all Apache Web servers in your network from one interface
  - Execute common Apache directives without having to manually change the httpd.conf file, which can introduce errors
- Provides a Web container for the J2EE* environment included with NetWare 6.5, letting you create and host money- and time-saving Web services, such as
  - Integration of existing incompatible legacy software applications
  - Interaction of business systems between two or more companies to improve efficiencies of information exchange
- Is preconfigured to work with Jakarta-Tomcat, the servlet container created by the Apache Foundation, which can be used to host servlets for automating business processes
- Is compatible with the new Novell exteNd Application Server for deploying Web applications and Web services
- Is ideal for Web application development and testing

How Apache Is Used on NetWare

The Apache Web server is used on NetWare® 6.5 in two ways: as an administration server for Novell services and as a dedicated Web server, if you choose to install a Web server.

To accomplish this, two instances of Apache are configured on your server. Of course, if during the NetWare 6.5 installation you didn't choose to have a Web server, only the administration instance is installed.

Using Apache As a NetWare Administration Server

Apache is used as a NetWare 6.5 administration server for several products such as Novell iFolder® and iManager. Some products, such as NetWare Remote Manager (NRM) don't depend on Apache because they have their own HTTP stacks.

So when you use iManager, accessible from any Web browser (including the new Web browser now available from the NetWare GUI), it is this instance of the Apache Web server that is serving up the data between the Web browser and NetWare 6.5.

For this reason, Apache is installed by default, even if you do not choose it as your Web server. However, the administration instance of Apache is created in its own directory (sys:\adminsrv).

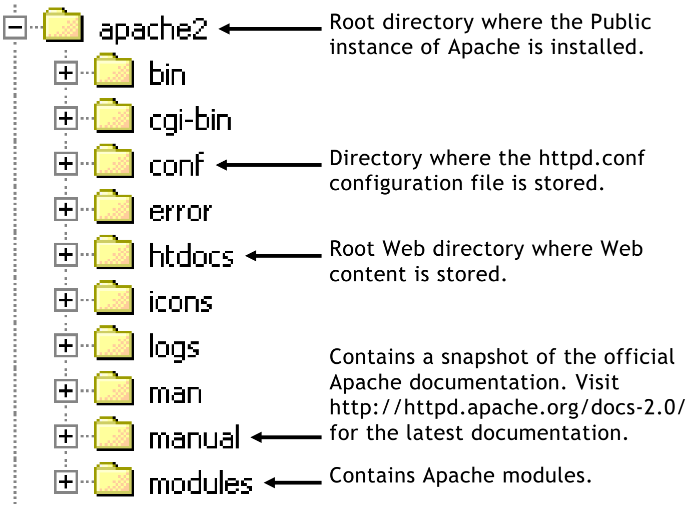
If you did not install Apache as your Web server, the Apache2 directory does not exist.

Using Apache As a Dedicated Web Server

When you choose Apache as your Web server, a second instance is installed in the operating system address space where you can utilize it as a dedicated Web server.

Whether you need it for hosting a simple department intranet site or for use in hosting more complex Web services or business-to-business solutions, Apache provides very fast and reliable HTTP services.

The public instance of Apache is installed in the Apache2 directory located at the root of your server’s sys volume. It contains several sub-directories described in the following figure.



Root directory where the Public instance of Apache is installed.

Directory where the httpd.conf configuration file is stored.

Root Web directory where Web content is stored.

Contains a snapshot of the official Apache documentation. Visit http://httpd.apache.org/docs-2.0/ for the latest documentation.

Contains Apache modules.

Unbeatable Security with Apache, eDirectory, and Built-in SSL

Running Apache on NetWare provides one of the industry’s most secure Web servers. This is because of NetWare’s tight integration with eDirectory and the built-in services of SSL that run at the core of the NetWare operating system.

Together, eDirectory and SSL keep your business information safe from intruders yet accessible from anywhere by people who have the proper access rights.

What’s Different with Apache on NetWare

If you are already familiar with the Apache Web server running on other platforms, you will find almost no differences on the NetWare platform. All of the same modules available on other platforms are available on NetWare 6.5, with a few additional modules such as mod_edir. Mod_edir enables Web pages to be served up from a user's home directory and provides remote file system access and authentication services. Auth_LDAP enables LDAP authentication to LDAP directories including Novell eDirectory.

Therefore, Apache functions in the same way on NetWare as it does on other platforms. However, there are a few key differences.

Apache on NetWare Is Multi-Threaded

Because Apache on NetWare is multi-threaded, it does not use a separate process for each request, as Apache does in some UNIX* implementations. Instead, multiple threads run simultaneously: a parent thread, and multiple worker threads which handle the requests.

Because of this, the directives used for managing processes are utilized differently, as described in the following table.

Directive	Usage on NetWare
MaxRequestsPerChild	As on UNIX, this directive controls how many requests a worker thread serves before exiting. The recommended default (0) causes the thread to continue servicing requests indefinitely. **IMPORTANT:** Unless there is a specific reason for setting this to something other than the default value, we recommend that it be kept at 0.
MaxSpareThreads	Instructs the server to begin terminating worker threads if the number of idle threads ever exceeds this value. We recommend that you use the default setting of 75.
MaxThreads	Limits the total number of worker threads to a maximum value. We recommend that you use the default setting of 250.
MinSpareThreads	Instructs the server to spawn additional worker threads if the number of idle threads ever falls below this value. We recommend that you use the default setting of 10.
StartThreads	Specifies how many threads the server should start with. We recommend that you use the default setting of 50.
ThreadStackSize	Specifies the stack size of each worker thread. We recommend that you use the default setting of 65536.

Syntax of Pathnames

Directives that accept filenames as arguments must use fully qualified NetWare pathnames, including the volume name. For example, sys:/apache2/htdocs. If the volume name is not specified, Apache defaults to the sys: volume.

Also, because Apache uses UNIX style pathname conventions internally, you must use forward slashes (/) in place of backslashes (\) in directive arguments.

Loading Modules at Runtime

Apache on NetWare has the ability to load modules at runtime, without having to recompile the server.

A number of external modules can be loaded from the \Apache2\modules directory. To activate these, or other modules, the LoadModule directive must be used. For example, to active the status module, use the following (in addition to the status-activating directives in access.conf):

```
LoadModule status_module modules/status.nlm
```

See Apache Module mod_so (http://httpd.apache.org/docs-2.0/mod/mod_so.html#creating) for more information about creating loadable modules.

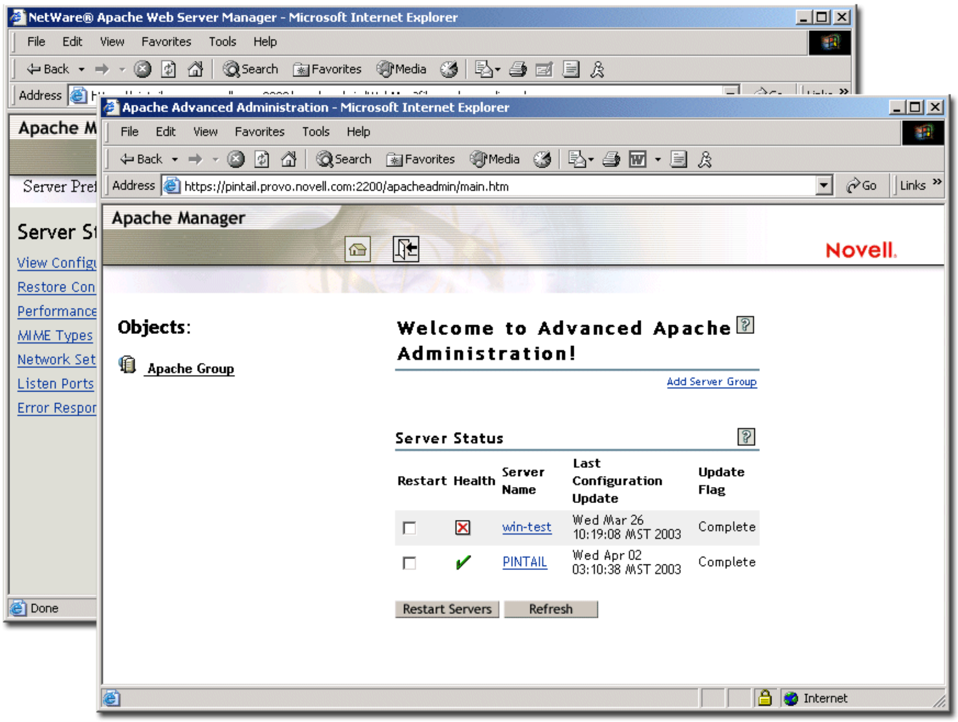
When configuring Apache manually, refer to the Apache 2.0 documentation (http://httpd.apache.org/docs-2.0).

Apache Manager: A Web-based Administration Tool

Whereas other platforms require you to manually edit configuration files to configure Apache, NetWare 6.5 includes a simple, Web-based graphical user interface that updates the configuration files for you: Apache Manager.

If you have multiple instances of Apache running on various platforms in your network—sometimes called a server farm—you can control them all from the Multiple Server Administration pages of Apache Manager, giving you single-point access to, and control over, all of your Web servers.

Figure 1 Apache Manager’s Single Server and Multiple Server Administration pages.



NetWare® Apache Web Server Manager - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Apache M
Server Pre
Server St
View Config
Restore Con
Performance
MIME Types
Network Set
Listen Ports
Error Respor
Done
Apache Advanced Administration - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address https://pintail.provo.novell.com:2200/apacheadmin/main.htm
Go Links
Apache Manager
Novell.
Objects:
Apache Group
Add Server Group
Server Status
Restart Servers
Refresh
Internet

Welcome to Advanced Apache Administration!

Restart	Health	Server Name	Last Configuration Update	Update Flag
□	☒	win-test	Wed Mar 26 10:19:08 MST 2003	Complete
□	✓	PINTAIL	Wed Apr 02 03:10:38 MST 2003	Complete

If you are already familiar with Apache and are comfortable configuring it manually, you can do so on NetWare as well. However, a single typographical error in the context of the configuration file can render its content inaccessible or even shut down the Apache Web server.

Using Apache Manager decreases the potential for human error, saving you and your customers time and unnecessary frustration.

Also, Apache Manager lets you control Apache from anywhere that you have Internet access, even from remote locations, provided you have access rights to connect through your company firewall.

For more information about Apache Manager, see “About Configuring and Managing Apache On NetWare” on page 17.

Hosting a Web Site

Web sites are not all created equal. Some are simple collections of HTML pages that contain static information, such as company background information. Even though some scripting, such as

JavaScript*, might be used for creating navigation effects like rollover buttons, a simple Web site largely consists of static files. When the files are updated, it is usually by hand. Little or no processing of data is done at the server.

Of course, the main purpose of having a Web server is to host a Web site, or to use Apache as the HTTP server in a partnership with an application server, such as the Novell exteNd Application Server.

Conversely, a dynamic Web site is one in which information is created dynamically as it is requested either from a user or another computer.

Building dynamic Web sites involves the use of servlets or Web applications, and might also involve databases (such as MySQL) and scripting languages (such as PHP or Perl). If you are integrating legacy applications or creating business-to-business solutions, you might also need to use SOAP, UDDI, and WSDL. NetWare 6.5 includes all of these open source solutions.

Web sites where products or services are bought or sold, such as is done on Amazon.com, are examples of dynamic Web sites. Other dynamic Web sites are not seen by users, but are used instead as part of a supply chain process between businesses.

Regardless of the complexity of your Web site, Apache is designed to be fast and reliable.

Using Servlets with Apache on NetWare

Servlets are like small Web applications and are often used to accomplish less robust processing. They can be used to save time and money by processing information very quickly, in ways that users cannot.

For example, Novell's Web Search Server is used to index file and Web content, allowing users to search for and find specific information from within large collections of information stored on one or more Web or file servers.

NetWare Web Search Server uses five primary servlets. The Highlighter servlet marks up the content of search results, highlighting all instances of the keyword that a user is searching for. (For more information about NetWare Web Search Server, see the *NetWare 6.5 Web Search Server Administration Guide*.)

Other servlets might include such things as online calculators, shopping carts, or calendars.

Because NetWare 6.5 is J2EE compliant, servlets created to run on other J2EE compliant platforms also run on NetWare 6.5 without the need for customization or rewriting any code. Simply copy the servlets to NetWare 6.5 and they run.

To run servlets, you must use Tomcat, also included with NetWare 6.5, and a key component of J2EE.

Also created by the Apache Foundation, Tomcat is a servlet container that processes servlet requests. Apache on NetWare is preconfigured to run with Tomcat.

Hosting Web Services and Applications

NetWare 6.5 offers a reliable, high-performance J2EE environment for the development and deployment of Java based Web applications and services. In addition to the open source products included with NetWare 6.5 (Apache Web server, Tomcat, and MySQL), NetWare 6.5 also includes the new Novell exteNd Application Server.

Using the exteNd Application Server, you can

- Integrate legacy applications, breaking down information silos that bog down the exchange of information between the organizations within your company
- Interact with the business systems of other companies, such as partners and clients, by building in Web services functionality (SOAP, UDDI, and WSDL)

For more information about the exteNd Application Server and building Web applications and services, visit www.silverstream.com (http://www.silverstream.com/Website/app/en_US/AppServer).

What's Next

- If you have not yet installed Apache as your Web server, see Chapter 2, "Installation and Configuration," on page 15.
- If you have already installed Apache and want to begin managing it, see Chapter 3, "Managing Apache Web Server Preferences," on page 23.
- For an overview of J2EE and Web services, see Overview: Novell Web and Application Services.

2 Installation and Configuration

Apache Web Server 2.0 is installed by default during the NetWare® 6.5 installation process. This instance of Apache is used by NetWare 6.5 features and products, acting as an administration server. (For more information, see "How Apache Is Used on NetWare" on page 8.)

However, you can install a public instance of Apache for your own use in hosting Web sites. You can also install Apache with the Tomcat Servlet Container 4.1 and host servlets and JavaServer Pages (JSPs*). If you want to host a database application, you can also select the AMP (Apache, MySQL, PHP, Perl) Server from the list of pre-configured servers during the NetWare 6.5 installation.

The Apache Manager, which also runs on the Apache administration server, offers a graphical interface for making common configuration changes to Apache, including the ability to edit multiple instances of Apache wherever they are running in your network, regardless of the platform on which they are running.

This chapter includes the following topics:

- "Upgrading to Apache from the NetWare Enterprise Web Server" on page 15
- "About Installing Apache on NetWare" on page 16
- "About Configuring and Managing Apache On NetWare" on page 17
- "Introducing Apache Manager" on page 18
- "Switching Administration Modes" on page 20
- "About Using Apache with Novell Cluster Services for High Availability" on page 21
- "What's Next" on page 21

Upgrading to Apache from the NetWare Enterprise Web Server

If you are upgrading to NetWare 6.5 and your existing server uses the NetWare Enterprise Web Server, you must upgrade to Apache. A special migration tool has been created to handle the upgrade for you.

What the Migration Tool Does

During the installation process, the migration tool leaves your Web content and related files intact, in the same directory structure that is already in place, which in most cases is novonyx/suitespot/docs. This path could be different if you had configured an alternate root directory. Apache is then configured to recognize this path so that your content is still accessible.

A copy of the previous Apache configuration is then saved to httpd.conf.001. A new Apache configuration file is created called httpd.conf. It contains directives based on settings in your Enterprise configuration files.

The migration tool does not automatically migrate all settings. Some final adjustments require manual configuration changes by editing the httpd.conf file or by using Apache Manager (See "About Configuring and Managing Apache On NetWare" on page 17).

Settings that are not migrated from Enterprise to Apache include the following:

- NSAPI plug-ins
- Tomcat
- LCGIs
- MIME types
- Authentication settings
- Authorization settings

When Upgrading a Server Running Apache Web Server 1.03

When upgrading a NetWare server that is running Apache 1.03, NetWare 6.5 disables version 1.03. As long as Apache 2.0 is selected during the install, the install transitions Novell software that is dependent on Apache to use Apache 2.0.

If you have other programs configured to use Apache 1.03, we recommend that you reconfigure them to use Apache 2.

About Installing Apache on NetWare

You can install the public instance of Apache during or after the NetWare 6.5 installation. The administration instance of Apache is installed by default.

During the NetWare 6.5 installation process, you can customize your server by selecting individual components to be installed, or you can select a specific server type from a list of fifteen pre-configured servers, which installs the necessary software needed to support a specific server solution.

For example, if you wanted to dedicate a NetWare server to hosting Web sites, you could select the Apache/Tomcat Server from the Pre-Configured Servers list. Or, if you simply wanted a Web server available on your NetWare server, you could select Apache2 Web Server and Tomcat 4 Servlet Container from the Components dialog box when you choose the Custom NetWare Server option.

The following table can help you decide which installation option you should choose based on how you want to use the Apache Web server on NetWare 6.5.

If You Want To	Do This
Dedicate a NetWare 6.5 server to hosting Web site content and deploying and hosting servlets and JSPs	Select Apache/Tomcat Server from the Pre-configured Servers list during the NetWare 6.5 installation. This option installs Apache Web Server 2.0 and the Tomcat Servlet Container 4.1.

If You Want To	Do This
Dedicate a NetWare 6.5 server to hosting Web applications using Apache, MySQL, PHP, and Perl	Select NetWare AMP (Apache, MySQL, PHP, Perl) Server from the Pre-configured Servers list during the NetWare 6.5 installation. This option installs Apache Web Server 2.0, MySQL 4.0, and the PHP and Perl scripting engines.
Add Apache as a component to your customized NetWare 6.5 installation, which you can use to host Web site content and deploy and host servlets and JSPs	Select Apache2 Web Server and Tomcat 4 Servlet Container from the Components list when selecting the Customized NetWare Server option. This option installs Apache Web Server 2.0 and the Tomcat Servlet Container 4.1.
Install Apache after installing NetWare 6.5	Run the NetWare post-install program from the NetWare GUI and select Apache2 Web Server and Tomcat 4 Servlet Container. **NOTE:** You can also install Apache remotely using the NetWare Deployment Manager. This option installs Apache Web Server 2.0 and the Tomcat Servlet Container 4.1.

HINT: For information about installing Apache from binary files, or installing it manually from your own build files, see Installing Apache on NetWare (http://httpd.apache.org/docs-2.0/platform/netware.html#inst) on the Apache Web site.

Installing Apache

Once you have decided which installation option to use, see the *NetWare 6.5 Overview and Installation Guide* for detailed installation instructions.

Installing the Administration Instance of Apache

As mentioned above, the administration instance of the Apache Web server is installed by default during the NetWare 6.5 installation. However, if for any reason you need to install it after you have installed NetWare, you can do so by running the NetWare post-installation program from the NetWare GUI or the NetWare Deployment Manager and then selecting Apache2 Admin Server.

For detailed information about installing products after installing NetWare 6.5, see Installing Products and Updates in the *NetWare 6.5 Overview and Installation Guide*.

About Configuring and Managing Apache On NetWare

The Apache Web server is configured primarily through the use of Apache directives, which are commands with values assigned to them in the Apache2/conf/httpd.conf file. Apache reads this file at startup (and periodically thereafter) and runs according to the specified values.

The Apache configuration file is a simple text file containing all of the directives necessary to configure the Web server and any additional modules that might be loaded. These directives and modules are well documented on the Apache Web site, making it relatively easy to configure and manage your server.

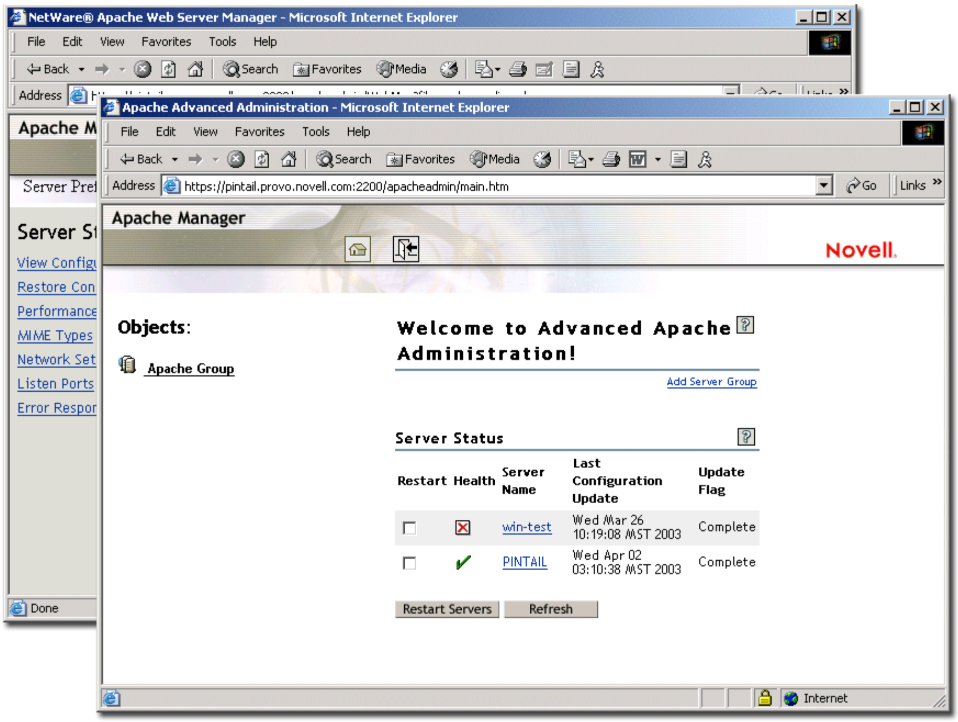
However, when manually editing the httpd.conf file, it is easy to introduce errors by incorrectly typing the name of a directive or omitting other necessary components of the syntax. A single typo or incorrect syntax can cause problems for your server and interrupt the services you provide. In addition, if you are managing several installations of Apache, keeping all of their httpd.conf files synchronized can waste time and cause additional problems.

Apache Manger, a new Web-based administration tool included with NetWare 6.5, offers a simple GUI alternative to the httpd.conf file, making it easier to manage Apache. But Apache Manager is more than a simple GUI for the httpd.conf file. Using the Multiple Server Administration mode, you can manage multiple installations of Apache running on multiple servers in your network. This is sometimes referred to as a server farm. And because Apache Manager is a Java application, it is platform independent.

Introducing Apache Manager

Apache Manager offers many advantages over manually configuring Apache:

- Changes to directives are done electronically, reducing the risk of errors.
- You don’t have to know all of the Apache directives or modules to configure Apache.
- Lets you manage multiple installations of the Apache Web server from a single interface.
- Eliminates the need to edit and maintain a large number of configuration files where many of the same directives are being used on each Apache Web server.
- A thorough help system that includes hypertext links to this manual (*Apache Web Server Administration Guide for NetWare 6.5*), and to specific topics within the official Apache documentation set (on apache.org).


NetWare® Apache Web Server Manager - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Apache M
Server Pref
Server St
View Configu
Restore Con
Performance
MIME Types
Network Set
Listen Ports
Error Respor
Done
Apache Advanced Administration - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Go
Links
Apache Manager
Novell.
Objects:
Apache Group
Add Server Group
Restart Servers
Refresh
Internet

https://pintail.provo.novell.com:2200/apacheadmin/main.htm

Welcome to Advanced Apache Administration!

Server Status

Restart	Health	Server Name	Last Configuration Update	Update Flag
☐	☒	win-test	Wed Mar 26 10:19:08 MST 2003	Complete
☐	✓	PINTAIL	Wed Apr 02 03:10:38 MST 2003	Complete

Apache Manager includes two interfaces: Single Server Administration and Multiple Server Administration. The first interface is used to manage a single instance of the Apache Web server at a time. The second interface is best used if you are running multiple instances of Apache and want to consolidate the configuration changes you make.

NOTE: You can now install Apache Manager on Windows or Linux. For more information, see Appendix A, "Installing the Apache Manager Daemon on Windows or Linux," on page 73.

Using Apache Manager's Single Server Administration

If you are configuring a single Apache Web server, use the Single Server Administration interface of Apache Manager. It is designed to let you manage one Apache Web server at a time.

The Single Server Administration lets you:

- Manage server preferences, such as adjusting the thread stack size, modifying network settings, and configuring MIME types
- View access and error logs and adjust log settings
- Manage content settings, such as setting up additional document or user home directories, configuring URL forwarding or CGI extensions, and setting up virtual hosts
- Enable or disable Apache modules (mod_PHP, mod_Perl, mod_NSN, and mod_Cache)
- Change Apache configuration administration modes, which specifies where the Apache configuration is be stored (See "Switching Administration Modes" on page 20.)

To start Apache Manager's Single Server Administration:

1. Using a Web browser, open the secure version of the NetWare 6.5 Welcome Web site using your server's URL. For example,

   ```
   https://myserver.mycompany.com:2200
   ```

   or

   ```
   https://123.456.789.100:2200
   ```

   HINT: For more information about the Welcome Web site, see NetWare 6.5 Welcome Web Site in the *NetWare 6.5 Administration Overview*.

2. When prompted, enter your administrator username and password and click Login.
3. In the left frame of the NetWare 6.5 Welcome Web site home page, click ⊞ next to Open Source.
4. Click Apache 2.0.
5. Under Apache 2.0 Links in the upper right corner of the right frame, click Administer Single Apache Server.

Using Apache Manager's Multiple Server Administration

If you are configuring multiple installations of Apache, you can use the Multiple Server Administration mode of Apache Manager. Its primary purpose is to share configurations between several Apache Web servers and involves the use of Novell® eDirectory™.

IMPORTANT: This mode of Apache Manager requires that you have a good understanding of Apache directives. You will be required to type directives and to know the correct syntax. If you are familiar with Apache directives, then you should be able to use the Multiple Server Administration mode.

When using the Multiple Apache Administration interface, a special daemon is used to record Apache configuration changes using eDirectory. Understanding how this works and how eDirectory makes it possible to share configurations between different Apache servers can help you understand how to take advantage of the Multiple Apache Administration interface.

For detailed information about using this version of Apache Manager, see Chapter 6, “Managing Multiple Instances of Apache,” on page 59.

To start Apache Manager’s Multiple Server Administration:

1 Using a Web browser, open the secure version of the NetWare 6.5 Welcome Web site using your server’s URL. For example,

```
https://myserver.mycompany.com:2200
```

or

```
https://123.456.789.100:2200
```

HINT: For more information about the Welcome Web site, see NetWare 6.5 Welcome Web Site in the *NetWare 6.5 Administration Overview*.

2 When prompted, enter your administrator username and password and click Login.

3 In the left frame of the NetWare 6.5 Welcome Web site home page, click ⊞ next to Open Source.

4 Click Apache 2.0.

5 Under Apache 2.0 Links, click Administer Multiple Apache Servers.

Switching Administration Modes

The Single Server Administration interface lets you choose where Apache’s configuration is stored. Configurations can be stored in one of two places: in a file (httpd.conf) on the same server that is running Apache Manager, or in eDirectory. These are referred to as *file mode* and *directory mode*.

When using file mode, the configuration changes you make using Apache Manager are made directly to the httpd.conf configuration file stored by default in the *volume*:\\Apache2\conf directory.

Use file mode if you are running a single instance of the Apache Web server. There is no added value to using directory mode if you are using only one or two instances of Apache running on the same server. However, if you are running multiple installations of Apache across multiple servers, you should consider using directory mode.

When using directory mode, a special Java* daemon, referred to as the configuration daemon, imports the contents of the Apache configuration file (httpd.conf) from each Web server into eDirectory where it is stored and managed. The configuration daemon constantly checks for changes to httpd.conf and updates the configurations stored in eDirectory. It then restarts the Web server so that the changes can take effect.

To switch Apache Manager to eDirectory administration mode:

1 From NetWare system console prompt, type **`ap2webman`** to run the configuration daemon.

2 From a Web browser running on a client in your network, open the Single Server Administration interface of Apache Manager.

3 Click Administration Mode.

4 Select eDirectory and click Save.

5. Click Save and Apply.

When the configuration daemon is started, it automatically imports your Web server's configuration into eDirectory. The Apache configuration for each instance of the Apache Web server that is installed on the server where you run the configuration daemon is imported into eDirectory. When you have finished running the daemon and changed the administration mode to eDirectory, a list of all of your Web servers is returned in your Web browser. You can then click on any one of them to administer it.

For more information about the configuration daemon and how the directory is used, see Chapter 6, "Managing Multiple Instances of Apache," on page 59.

HINT: If you are running Apache on other servers and want them imported into eDirectory as well, you must run the configuration daemon on each of them. If they are running on other platforms, such as UNIX*, you can download a version of the daemon for use on your platform from the Novell Developer Kit (NDK).

To switch back to file mode from directory mode:

1. Open the Single Server Administration interface of Apache Manager.
2. Click Administration Mode.
3. Select File and click Save.
4. Click Save and Apply.

About Using Apache with Novell Cluster Services for High Availability

Apache can be used with Novell Cluster Services™ (NCS) to provide high availability support to the customers you service. This means that if one server goes down, another server takes over and customers never experience an interruption to the services you provide.

Using Apache with NCS requires that you first successfully install and configure NCS and then set up Apache to work in the NCS environment. For information about how to set up Apache with NCS, see Configuring Apache with Novell Cluster Services in the *Novell Cluster Services 1.7 Resource Configuration Guide*.

What's Next

Once you have installed Apache, you might want to look at the following chapters:

- Chapter 3, "Managing Apache Web Server Preferences," on page 23
- Chapter 4, "Managing Server Content," on page 37
- Chapter 5, "Managing Apache Modules," on page 49
- Chapter 6, "Managing Multiple Instances of Apache," on page 59
- Performance Tuning Tips (http://httpd.apache.org/docs-2.0/misc/perf-tuning.html)
- Developer Documentation for Apache 2.0 (http://httpd.apache.org/docs-2.0/developer)

3 Managing Apache Web Server Preferences

From the Preferences page of Apache Manager's Single Server Administration, you can perform many tasks, including starting and stopping the Web server, adjusting thread stack sizes for performance tuning purposes, and managing listen ports.

This chapter contains the following topics:

Managing Apache Server Status

Once installed, Apache runs constantly, listening for and accepting requests. You can start and stop Apache using Apache Manager, Novell® iManager, or the NetWare® system console.

When you stop the Apache server, all threads that are currently running are allowed to finish. The apache.nlm does not actually shut down, so restarting the server is much faster than in prior versions.

After you shut down the server, it might take a few seconds for the server to complete its shutdown process and for the status to change to Down.

You can use the Server Status page of Apache Manager to verify whether Apache is running. You can also start, stop, or restart Apache.

Starting or Stopping Apache

1. From the Apache Manager home page, click Apache Web Server *servername*.
2. Click Start Server or Stop Server.

If the Apache Web server is already running, the Start Server button reads *Restart Server*. Click Restart Server to have it shut down and then start up again.

IMPORTANT: If you run Apache Manager in Directory mode, you cannot verify whether the server is running. To verify whether an Apache server is running, you must do so from the system console or by pointing a Web browser at the server. If you are able to view your Web content, the server is running.

Starting Apache from the System Console

To start Apache, enter **`Ap2webup`** at the system console. This loads Apache into the operating system (OS) address space. If you want to load Apache into protected address space, you can do so at the console prompt using the LOAD command. For example:

```
load address space = apache2 apache2
```

Using this command loads Apache into an address space called *apache2*.

After starting Apache, it listens on port 80 for requests from client Web browsers, unless you changed the Listen directive in the configuration files.

Once Apache is started, open a Web browser either from the NetWare GUI or from a client computer in your network. Enter the URL to your Apache Web server, which can be either an IP address or a DNS name. For example:

```
http://myserver.mycompany.com
```

or

```
http://012.345.678.910
```

HINT: If there is no response, look in the *volume*:/apache2/logs/error_log file for details.

If the Apache server is running correctly, a default Web page appears (index.html.*language_code*). The actual file is found in the *volume*:/apache2/htdocs directory, the directory defined by default as the root Web directory. Replace this file with your own home page.

Once Apache is running correctly, you can make changes to its default configuration by editing the files in the conf directory.

Stopping Apache from the System Console

To unload Apache running in the OS address space, enter **`ap2webdn`** at the system console.

If apache is running in a protected address space, specify the address space in the unload statement. For example, at the console prompt, enter

```
unload address space = apache2 apache2
```

or,

```
apache2 shutdown -p address space
```

Running Additional Instances of Apache Simultaneously

You can run multiple instances of Apache concurrently on NetWare by loading each additional instance into its own protected address space.

To do so, each additional instance must have its own address space name. For example:

```
load address space = apache3 apache2
load address space = apache4 apache2
```

Using the examples above would create two additional instances of Apache with the unique address space names of apache3 and apache4.

In addition, each instance must be using its own httpd.conf file wherein unique ports, error and access log filenames can be specified. Attempting to use the same configuration file causes various errors, including port conflicts.

Using an Alternate Configuration File

If you want to run Apache using an alternate httpd.conf file, use the following command:

```
load address space = Instance1 apache2 -f path_to_httpd.conf
```

This is an effective method for avoiding port conflicts because using alternate httpd.conf files allows you to specify alternate port numbers.

Starting or Stopping the Apache Admin Server

To start the Apache Admin server, enter **admsrvup** at the system console. To stop the Apache Admin server, enter **admsrvdn**.

Verifying Server Status from the NetWare Console

The following command line directives can be used at the system console to modify or display information about Apache.

IMPORTANT: At the NetWare console prompt, each directive must be preceded by **apache2**, as in *servername*:**apache2** ***directive***. Also, Apache must be running. (See "Starting or Stopping Apache" on page 23.)

Directive	Effect
DIRECTIVES	Displays a list of all available directives.
MODULES	Displays a list of loaded modules, both built-in and external.
RESTART	Instructs Apache to terminate all running worker threads as they become idle, reread the configuration file, then restart each worker thread based on the new configuration.
SETTINGS	Enables or disables the thread status display on the console. When enabled, the number of threads currently running is displayed along with the status of each thread.
SHUTDOWN	Terminates the running instance of the Apache Web server.
VERSION	Displays version information about the currently running instance of Apache.

If you are already familiar with Apache on other platforms, see Using Apache with Novell NetWare (http://httpd.apache.org/docs-2.0/platform/netware.html) on the Apache.org Web site for information about what is different about Apache on NetWare.

Viewing Configuration Settings

The View Configuration page lists all Apache directives and lets you configure them by clicking on a directive.

The settings are stored in the Apache2/conf/httpd.conf file. For more information about Apache configuration files, see Configuration Files (http://httpd.apache.org/docs-2.0/configuring.html) in the Apache documentation.

The server's content settings depend on its configuration. Common server content settings include the server's document directory, its index filenames, name and location of its access log, and default MIME type.

Performance Tuning

Apache 2.0 includes performance enhancements that increase throughput and scalability. Most of these are enabled by default. In addition, you can change the configuration of Apache to best serve the needs for which you are using it.

For example, you can increase the maximum number of threads allowed to run simultaneously if your Web server is getting a larger number of client visits. You can also disable the Keep Alive feature to restrict persistent connections, which some Web clients request when they connect to your server.

Adjusting Thread Settings

Because Apache is very self-regulating, most sites do not need to adjust the default values of any of the thread directives. However, if you need to make changes to any of the thread settings, continue reading.

For more information about thread directives, see ThreadStackSize (http://httpd.apache.org/docs-2.0/mod/mpm_netware.html#threadstacksize) on the Apache Web site.

NOTE: Because Apache for NetWare is multi threaded, it does not use a separate process for each request, as Apache does in some UNIX implementations. Apache for NetWare uses a parent thread and multiple child threads, which handle all requests.

Modifying the Thread Stack Size

A *thread stack* is a piece of scratch memory that a thread uses to store information temporarily. If there is not enough stack space and the thread requires more in order to continue, the server will abend. Intensive applications usually require more stack space. Modules such as mod_perl or mod_php might require a thread to yield more stack space. However, 65,536 bytes is typically large enough.

Keep in mind that increasing the stack size results in consuming more system resources because each thread requires a certain amount of space. Therefore, increasing the stack size should be done only after considering what is required based on the applications and modules that are being used.

The ThreadStackSize (http://httpd.apache.org/docs-2.0/mod/mpm_netware.html#threadstacksize) directive tells the server what stack size to use for each running thread. If a stack overflow occurs, you need to increase this number.

To modify the thread stack size:

1 From the Performance Tuning page, enter a numerical value in the Thread Stack Size field.

The default is 65536.

2 Click Save.

Modifying the Number of Start Threads

The StartThreads (http://httpd.apache.org/docs-2.0/mod/mpm_common.html#startthreads) directive specifies the number of child server processes that are to be created when the Web server is started. Because the number of processes is dynamically controlled according to system load, there is usually little reason to adjust this parameter.

To modify the number of start threads:

1 From the Performance Tuning page, enter a numerical value in the Start Threads field.

The default is 50.

2 Click Save.

Modifying Minimum Spare Threads

The MinSpareThreads (http://httpd.apache.org/docs-2.0/mod/mpm_common.html#minsparethreads) directive defines the minimum number of idle threads set aside to process surges in client requests to the Web server.

Different multiprocessing modules (MPMs) deal with this directive differently. On NetWare, the mpm_netware module is used to control all of the threading directives and functionality.

To modify the number of start threads:

1 From the Performance Tuning page, enter a numerical value in the Minimum Spare Threads field.

The default is 10.

2 Click Save.

Modifying Maximum Spare Threads

The MaxSpareThreads (http://httpd.apache.org/docs-2.0/mod/mpm_common.html#maxsparethreads) directive lets you define the maximum number of idle threads allowed. Again, different MPMs deal with this directive differently. On NetWare, the mpm_netware module is used. Therefore, this directive tracks the minimum spare threads value on a server-wide basis.

To modify the maximum number of spare threads:

1 From the Performance Tuning page, enter a numerical value in the Maximum Spare Threads field.

The default is 75.

2 Click Save.

Modifying Maximum Total Threads

The MaxThreads (http://httpd.apache.org/docs-2.0/mod/mpm_netware.html#maxthreads) directive specifies the maximum number of worker threads allowed.

To modify the maximum total threads:

1. From the Performance Tuning page, enter a numerical value in the Maximum Total Threads field.

   The default is 250.

2. Click Save.

Adjusting Keep Alive Settings

Keep Alive provides live HTTP sessions that allow multiple requests to be sent over the same TCP connection. In some cases this has been shown to result in an almost 50% increase in latency times for HTML documents with many images.

To modify keep alive settings:

1. From the Performance Tuning page, click Yes or No to enable or disable Keep Alive.
2. Make your changes and click Save.

Limiting the Number of Keep Alive Requests

When Keep Alive is enabled, it limits the number of requests allowed per connection. Entering zero (0) in the Maximum Keep Alive Requests field allows an unlimited amount of connections. (See MaxKeepAliveRequests (http://httpd.apache.org/docs-2.0/mod/core.html#maxkeepaliverequests) on the Apache Web site.

For maximum server performance, we recommend that this setting be kept to a higher value. The default setting is 100.

Specifying a Time-out Limit for Keep Alive Requests

The KeepAliveTimeout (http://httpd.apache.org/docs-2.0/mod/core.html#keepalive) directive lets you specify (in seconds) how long Apache waits for a subsequent request before closing a TCP connection. Once a request has been received, the time-out value specified by this directive applies.

Setting Keep Alive Timeout to a high value can cause performance problems for heavily loaded servers. The higher the timeout, the more server processes are kept busy waiting on connections with idle clients.

Using DNS

When enabled, the HostnameLookups (http://httpd.apache.org/docs-2.0/mod/core.html#hostnamelookups) directive records the names of clients or their IP addresses: www.apache.org (when on, or enabled) or 204.62.129.132 (when off, or disabled).

The default is set to Off. This is because when enabled, every client request would result in at least one lookup request to the nameserver, causing unnecessary congestion on DNS servers and the Internet.

For additional information about DNS issues on Apache, see Issues Regarding DNS and Apache (http://httpd.apache.org/docs-2.0/dns-caveats.html) on the Apache Web site.

Additional Performance Tuning Information

You can also adjust the settings of the Mod_Cache module. For more information about Mod_Cache, see Chapter 5, “Managing Apache Modules,” on page 49.

For additional information about performance tuning, see Apache Performance Notes (http://httpd.apache.org/docs-2.0/misc/perf-tuning.html) on the Apache Web site.

Managing MIME Types

Multipurpose Internet Mail Extension (MIME) is a specification used to identify a file type by its extension so that when Apache receives a request for a file, it knows how to handle the file. A list of MIME types that Apache already knows about is included in the conf/mime.types file.

The Global MIME Types page saves you the trouble of manually entering a new MIME type or modifying an existing one. MIME types created on the Global MIME Types page are not added to the conf/mime.types file, but rather are listed in the httpd.conf file under the AddType (http://httpd.apache.org/docs-2.0/mod/mod_mime.html#addtype) directive.

MIME types added to the httpd.conf file override MIME types of the same name that already exist in the mime.types file.

Files can have more than one extension and their order does not typically matter. For example, if the extension .rus maps to Russian and HTML maps to HTML, then the files text.rus.html and text.html.rus are treated alike.

However, unrecognized extensions, such as .xyz, wipes out all extensions to their left. Therefore, text.rus.xyz.html is treated as HTML but not as Russian.

HINT: If you will be downloading NetWare Loadable Module^{TM} (NLM^{TM}) applications to your Web server, you might want to add NLM as a MIME type. If you do, use application/octet-stream as the content type and .nlm as the suffix.

To create a new MIME type:

1. From the MIME Types page, type a name in the Content Type field that describes the new MIME type.
2. Type the character extension in the Suffix field.

   Enter a period, followed by letters or numbers.
3. Click New Type.

To edit an existing MIME type:

1. From the MIME Types page, locate the MIME type to be edited or removed.
2. Click Edit and make the required changes to the Content Type and Suffix fields.
3. Click Edit Type.
4. Click Save and Apply to save the changes to the httpd.conf file and restart Apache to apply the changes.

   or

   Click Save to save the changes to the httpd.conf file without applying the changes by restarting Apache. You will have to restart Apache eventually.

   or

Click Undo to cancel the changes.

Default MIME Types

When a document is sent to a client, the server includes a section that identifies the document's type, so the client can present the document in the correct way. However, sometimes the server can't determine the proper type for the document because the document's extension is not defined for the server. In those cases, a default value is sent.

The default is usually Text/Plain, but you should set it to the type of file most commonly stored on your server. Some common MIME types include the following:

text/plain	text/html
text/richtext	image/tiff
image/jpeg	image/gif
application/x-tar	application/postscript
application/x-gzip	audio/basic

Additional Information About MIME Types

For more information about MIME types, see Content Negotiation (http://httpd.apache.org/docs/content-negotiation.html) on the Apache Web site.

Specifying an Administrator E-Mail Address for Inclusion in Error Messages

If users receive an error message, such as a 404 Not Found error, you can include the e-mail address of the Apache administrator as a means of providing customers with a method of notifying you about problems on your Web site or with your Web applications.

For example, if you specified john@digitalairlines.com as the value of the ServerAdmin directive and a user received a 404 Not Found error, a text message would include john@digitalairlines.com as the administrator to contact for further assistance.

The ServerAdmin (http://httpd.apache.org/docs-2.0/mod/core.html#serveradmin) directive sets the e-mail address that the server includes in any error messages it returns to the client.

To specify an administrator e-mail address:

1. From the Network Settings page of Server Preferences, type a valid e-mail address users should contact about error messages.
2. Click Save.

For information about customizing the error messages themselves, see "Managing Error Responses" on page 32.

Setting Up Server-Side Includes

Server-side includes (SSIs) provide a means of adding dynamic content to existing HTML documents without the use of a CGI program or other dynamic technology.

SSIs are directives placed in HTML pages and evaluated on the server while the pages are being served. Wherever you add SSI directives within an HTML page, that is where the results of the SSI code show up. For example, you could embed the current date or time into a Web page by adding the following code to an existing HTML file:

```
<!--#echo var="DATE_LOCAL" -->
```

SSI code appears like an HTML comment. However, if SSI is configured properly, Apache processes it as SSI code and in this sample, the current date appears on your Web page.

Enabling and Configuring SSI

Before SSI codes are recognized by Apache, you must first enable it. You must also specify the file extension you will use for files containing SSI directives. This helps Apache identify which files contain SSI.

To enable and configure SSI:

1. From the Network Settings page of Server Preferences, click On next to Server-Side Includes.
2. Specify the file extension to be used by files containing SSI directives.

   Typically, this is shtml, but you can specify any file extension you want, including simply html.
3. Click Save and Save and Apply.

For a more in depth discussion of SSI, see Introduction to Server-Side Includes (http://httpd.apache.org/docs-2.0/howto/ssi.html) on the Apache Web site.

Managing Listen Ports

You can direct Apache to listen to only specific IP addresses or ports; by default it responds to requests on all IP addresses. This directive is required. If it is not in the httpd.conf file, the server fails to start. This is a change from previous versions of Apache.

You can specify multiple ports. If you do so, Apache responds to requests from any of the listed addresses and ports.

To specify a new port number:

1. From the Listen Ports page, type the IP address, followed by a colon (:), followed by a port number.

   For example:

   ```
   123.456.789.100:2003
   ```

   IMPORTANT: Be sure to verify that the port number you use is not already in use by another service. One way to verify what ports are in use is through NetWare Remote Manager. You can access it through the administrator's version of the NetWare 6.5 Welcome Web site. Once there, click Remote Manager under Server Management. Once Remote Manager starts, click IP Address Management under Manage Server.

2. Under Encryption, click On if you want to use Secure Sockets Layer (SSL) with the newly specified port number.
3. If necessary, select an alternate server certificate from the Server Certificates drop-down list.
4. Click Save.

To edit or remove a port:

1. From the Listen Ports page, click Edit in the row of the Current Listen Ports table of the port you want to edit.
2. Modify the port information above the table and click Save and Apply to save the changes to the httpd.conf file and restart the server.

   or

   Click Save to save the changes to httpd.conf without restarting the server.

   or

   Click Undo to cancel the changes.
3. Click Remove to delete a port from the Current Listen Ports table.
4. Click OK to delete the port, or click Cancel.

For more information, see the Listen (http://httpd.apache.org/docs-2.0/mod/mpm_common.html#listen) directive on the Apache Web site.

Managing Error Responses

In the event of a problem or error, Apache can be configured to do one of four things:

- Output a simple hard-coded error message
- Output a customized message
- Redirect to a local URL path to handle the error
- Redirect to an external URL to handle the error

The first option is the default, while the remaining options are configured using the ErrorDocument (http://httpd.apache.org/docs-2.0/mod/core.html#errordocument) directive, which is followed by the HTTP response code and a URL or a message. Apache will sometimes offer additional information regarding the problem or error.

Working with Server Logs

To effectively manage a Web server, it is necessary to get feedback about the activity and performance of the server as well as any problems that might be occurring. Apache provides very comprehensive and flexible logging capabilities.

NOTE: Access logging is enabled on Apache by default. If your server experiences a high volume of traffic, you should consider either disabling it or configuring it so that the log files can’t grow larger than your server’s hard drive capacity. See “Setting Log Preferences” on page 34.

For more information about access and error logging on the Apache Web server, see Log Files (http://httpd.apache.org/docs-2.0/logs.html) on the Apache Web site.

Viewing the Access Log

The access log records information about clients who access your Web server, such as their IP addresses and the date and time when they accessed the Web server.

This information can be very useful. Here are a few examples:

- *Tracking advertising success*: Identifies the success of banner ads by viewing how often a banner ad has been clicked.
- *Tracking visibility to search engines*: Identifies which search engines are indexing your site.
- *Tracking efficiency of a purchase system*: Identifies how long customers are spending in your electronic purchasing process.

The type of information displayed depends on the settings of the Log Preferences page. A typical log shows an IP address, date, time, and the requested URL. For example:

```
137.65.67.133 - [27/Oct/2002:22:40:05 -0700] 200 - "GET HTTP/1.1" "http://
www.digitalairlines.com/"
```

Viewing the Error Log

The View Error Log page displays the contents of Apache's error log, which is the most important of the log files. The error log file is where the Apache httpd sends diagnostic information and where any errors related to processing requests are recorded.

Because the error log data can be viewed through the NetWare Apache Manager, you can view it from where ever you have Web access.

The error log is the first place to look when a problem occurs with starting the server or with the operation of the server, because it often contains details of what went wrong and how to fix it.

For more information about the error log, see Log Files (http://httpd.apache.org/docs-2.0/logs.html) on Apache.org.

Filtering Access and Error Log Data

You can filter log data displayed on the View Access Log page by specifying the maximum number of entries to be returned at one time. You can also filter the access log so that only entries containing specific information is returned, such as a specific IP address or date.

To filter the number of access log entries displayed:

1. In the Number of Entries field, enter the number of log entries you want displayed at one time.

   This can be any number between 1 and 500.

2. Click OK.

To filter log entries containing specific alphanumeric information:

1. In the Only Show Entries With field, enter an alphanumeric string.

   For example, 22/Aug/2003.

2. Click OK.

Setting Log Preferences

The Log Preferences page lets you enable or disable access logging, log rotations, the location of access log files, and the type of data to capture in the access log.

To enable (or disable) access logging:

1. From the Log Preferences page under Server Logs, click Yes.
2. In the Log File field, specify the path to the access log file.

   The default path is Apache2/logs/access_log.
3. Click Save.

Enabling Log Rotation

Even on a moderately busy server, the quantity of information stored in log files is very large. The access log file typically grows 1 MB or more per 10,000 requests. To manage growing log files, Apache can be directed to create a new log file once the initial log file reaches a certain size, or once a specified number of minutes has passed.

However, as an administrator, you must periodically delete or move the log files to prevent them from taking over your server's disk space. You can do this manually or by creating a batch file to delete them for you. Either way, you must manage the log files yourself.

Deleting or moving the log files cannot be done while the server is running, because Apache continues writing to the old log file as long as the file remains open. Instead, the server must be restarted after the log files are moved or deleted so that it can open new log files.

To enable the rotation of log files based on log file size:

1. From the Log Preferences page, click the By Size radio button to have the logs switched when a specific size (in megabytes) is reached in the first log file.
2. Click the MB drop-down list and select the number of megabytes at which the logs should be rotated.
3. Click Save.

To enable the rotation of log files based on a specified time period:

1. From the Log Preferences page, click the By Time radio button to have the logs switched when a specified period of time (in minutes) has elapsed.
2. In the Minutes field, type the number of minutes between each log file rotation.
3. Click Save.

To disable log file rotation:

1. From the Log Preferences page, click the Do Not Rotate radio button to disable log rotation.

   When selected, a single log file is used. If your Web site supports a high volume of traffic, we recommend that you use log file rotation.
2. Click Save.

Specifying a Log File Format

Common Log Format (CLF) is required by many off-the-shelf log analyzers such as wusage or ANALOG. If you will be using one of these tools to analyze your log files, select CLF.

The CLF format is:

```
host ident authuser date request status bytes
```

Alternately, you can select from the list of data types that you want Apache to log by checking one or more of the items in the Only Log list.

To specify the common log file format:

1. From the Log Preferences page, click Use Common Log File Format.
2. Click Save.

To customize the log file format:

1. From the Log Preferences page, click Only Log.
2. Select each of the items you want Apache to log.
3. Click Save.

For more information about logging, see the LogFormat (http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#logformat) directive on the Apache Web site.

What's Next

Once you have configured the Apache server preferences and log files, you can focus on managing the content that your Web server will serve up to Web clients.

Continue on to Chapter 4, "Managing Server Content," on page 37.

4 Managing Server Content

You can use the Novell® Apache Manager to help manage Web server content. You can create HTML pages and other files such as graphics or text and then store those files on your server. When users connect to your server, they can view your files provided they have access to them.

This chapter contains the following sections:

Modifying the Primary Document Directory

You probably don't want to make all the files on your file system available to remote clients. An easy way to restrict access is to keep all of your server's documents in a central location, known as the document root or primary document directory.

Another benefit of the document directory is that you can move your documents to a new directory (perhaps on a different disk, if for example, you are moving your Web site to a new server) without changing any of your URLs, because the paths specified in the URLs are relative to the primary document directory.

For example, if your document directory is sys:/Apache2/htdocs, a request such as http://www.novell.com/products/info.html tells the server to look for the file info.html in sys:/Apache2/htdocs.

If you change the primary document directory (by moving all the files and subdirectories), you only have to change the primary document directory that the server uses, instead of mapping all URLs to the new directory or telling clients to look in the new directory.

By default, the primary document directory is set to the *volume*:/Apache2/htdocs/ directory using the DocumentRoot (http://httpd.apache.org/docs-2.0/mod/core.html#documentroot) directive. The primary document directory is the directory from which Apache serves files.

It is unlikely that you will need to change the default primary document directory. However, if you do, keep in mind that the deeper into a file structure Apache has to go, the longer it takes Apache to examine the directories. To optimize performance, keep the primary document directory as close to the root of your server's volume as possible.

To modify the primary document directory:

1 From the Primary Document Directory page under Content Management, type the path, including the directory name that Apache should serve files from.

  Type a full path, such as SYS:/Apache2/htdocs.

  IMPORTANT: Do not include a trailing slash, as in SYS:/Apache2/htdocs/.

2 Click Save and Apply to save the changes to the httpd.conf file and restart the server.

For more information related to mapping directories to URLs, see Mapping URLs to Filesystem Locations (http://httpd.apache.org/docs-2.0/urlmapping.html) on the Apache Web site.

Setting Up Additional Document Directories

Most of the time you keep all of your documents in the primary document directory. But sometimes you might want to serve documents from a directory outside of your document root. You can do this by setting up additional document directories. By serving from a directory outside of your document root, you can let someone manage a group of documents without giving them access to your primary document root.

For example, if you have a directory named *marketing* at the root of your server volume, or even on another server in your network that is accessible using TCP/IP, you could add that directory as an additional document directory. You could then access it from a Web browser using the URL you specify in the URL Prefix field of the Additional Document Directories page. The actual path might be SYS:/marketing, but the URL would be http://www.digitalairlines.com/marketing.

You can also manage several options for each additional directory, such as enabling CGI scripting or server-side includes (SSIs). And if the content of an additional directory is not for general public use, you can easily apply access control restrictions using the Directory Access Control page.

Adding an Additional Document Directory

Once you have created directories on your server, you must identify them as additional document directories so that Apache knows where they are. You can then add new directories using the Additional Document Directories page of Apache Manager.

To add an additional document directory:

1 From the Additional Document Directories page under Content Management, type a name for the directory in the URL Prefix field.

2 Type the path to the directory on your server.

  You can use either a relative path, or a fully qualified path. For example:

  /marketing

  or

  SYS:/marketing

3 Click Save and then Save and Apply.

4 (Optional) If you made a mistake, click Undo on the Save and Apply page.

Once added, open a Web browser and enter the URL prefix you specified. If you have enabled indexing, a list of files currently held in the directory are displayed. (For information about how to enable indexing, see "Directory Indexing" on page 40.)

To delete an additional document directory:

1. From the Additional Document Directories page under Content Management, click Remove in the row of the document directory that you want deleted.
2. Click OK, Save and Apply, and then OK again.

Configuring Options for an Additional Document Directory

You can set specific options for each of your additional directory options from the Directory Configuration Options page. From this page, you can

- Enable CGI execution
- Enable SSIs
- Configure the use of directory indexing
- Enable multiple views
- Enable symbolic links

To configure options for an additional document directory:

1. From the Additional Document Directories page under Content Management, click Options in the row of the additional directory that you want to configure.
2. Make the needed changes, click Save, and then click Save and Apply.

CGI Execution

When enabled, CGI scripts contained in the additional directory can be executed. It is impossible to execute CGI from within an additional directory if this feature is not enabled.

For more information about using CGI, see Dynamic Content with CGI (http://httpd.apache.org/docs-2.0/howto/cgi.html) on the apache.org Web site.

Symbolic Links

If you create hard links to a file, such as marketing.html and then someone else deletes the file and replaces it with another one of a different name, your hard link no longer works.

To prevent this from happening, you can enable symbolic links, sometimes called soft links, which has the ability to keep the link accurate, even in the above scenario.

NOTE: Symbolic linking is not currently available on the NetWare® platform. However, because Apache Manager can be used to configure Apache running on other platforms in your network, it is included on the Directory Configuration Options page of Apache Manager.

For more information, see the Options (http://httpd.apache.org/docs-2.0/mod/core.html#options) directive on the Apache Web site.

Server-Side Includes

SSIs provide a method for adding dynamic content to existing HTML documents.

SSIs are directives placed in HTML pages that are evaluated on the server while the pages are being served. They let you add dynamically generated content to an existing HTML page, without having to serve the entire page using a CGI program.

The decision of when to use SSI and when to have your page entirely generated by a program is typically a matter of how much of the page is static, and how much needs to be recalculated every time the page is served.

SSI is a great way to add small pieces of information, such as the current time. But if a majority of your page is being generated at the time that it is served, you need to look for some other solution.

For more information about working with SSI, see Introduction to Server Side Includes (http://httpd.apache.org/docs-2.0/howto/ssi.html) on the Apache Web site.

Directory Indexing

If a URL to a directory is requested but there is no index.html file in that directory, the server returns a formatted listing of the directory.

Directory indexing also includes the ability to define the level of detail returned to the user or to disable indexing altogether, which would return a 404 Not Found error to the user.

To enable directory indexing:

1. From the Additional Document Directories page under Content Management, click Options in the row of the directory that you want to configure.
2. (**Optional**) Click Fancy to have Apache return an index that can be sorted and that includes additional details about the contents of the folder.
3. (**Optional**) Click Simple to have Apache return a list of files with no additional details and no sorting functionality.
4. (**Optional**) Select None to disable indexing.

   When disabled, and if there is no index file present, users receive the 404 Forbidden error message.
5. Click Save, and Save and Apply.

For more information, see the Options (http://httpd.apache.org/docs-2.0/mod/core.html#options) directive on the Apache Web site.

Multiple Views

Apache has the ability to return content in a way that best matches the client Web browser that requested it.

For example, you might have some content on your Web site that is available in different languages or different media types, or a combination of both. One way of selecting the best choice for the requesting client browser would be to return an index page and let the user make a selection.

However, it is possible for the server to choose automatically. This works because most browsers request information according to preferences selected by their users. Therefore, a browser could specify French as its preferred language, and English as its second choice. Multiple Views can then return the French document if there is one, and if not, return the English version in its place.

For more information, see Content Negotiation (http://httpd.apache.org/docs-2.0/content-negotiation.html) on the Apache Web site.

Controlling Access to Document Directories

If you have information on your web site that is sensitive or intended for only a small group of people, you can use authentication to control who has access to specific directories.

HINT: Before you can configure access for a particular directory, you must first create the directory. For more information, see "Adding an Additional Document Directory" on page 38.

Authentication is any process by which you verify that someone is who they claim they are. Authorization is also any process by which someone is allowed to be where they want to go, or to have information that they want to have.

Using Apache Manager, you can configure the Apache authorization module to control who has access to specific directories on your Apache Web server. Documents placed in a controlled directory can only be accessed by users who have been given rights to that directory.

To configure access control to a specific directory:

1. From the Additional Document Directories page under Content Management, click Access Control in the row of the directory that you want to configure.
2. From the Access Control Type drop-down list, select the type of user authentication you want used for the document directory you are configuring.
    - *Public Access*: Select this option of you want to allow general access to the directory by any user who can visit your Web site.
    - *Auth LDAP Mode*: (Recommended) Select this option of you want to use your LDAP server to authenticate specified users to the document directory. Users or groups should be specified under the Access Control fields. (For more information, see the mod_auth_ldap (http://httpd.apache.org/docs-2.0/mod/mod_auth_ldap.html) documentation on the apache.org Web site.)
    - *Auth Module*: Select this option if you want to use password files you create using Apache's htpasswd utility. For information, see Authentication (http://httpd.apache.org/docs-2.0/howto/auth.html) in the Apache documentation. (For more information, see the mod_auth (http://httpd.apache.org/docs-2.0/mod/mod_auth.html) on the apache.org Web site.)
    - *Auth DBM Module*: Similar to Auth Module but involves the use of a simple database rather than flat files. If, for some reason, you don't want to use LDAP and you have a large number users that you want to grant access rights to, use this option. (For more information, see the mod_auth_dbm (http://httpd.apache.org/docs-2.0/mod/mod_auth_dbm.html) on the apache.org Web site.)
3. Specify the level and method of access control.
    - *Any Valid User*: Click this option to allow any valid user to access the document directory you are configuring. A valid user is anyone who can log in to the server.
    - *User/Group List*: Click this option if you want to specify individual usernames or group names to whom access should be given. When typing multiple usernames or group names, separate each entry with a blank spaces.
    - *Use eDirectory Rights*: Verifies directory and file access rights in addition to verifying user credentials. User accounts must include specific rights to the directory for a user to have access to it. When running Apache on NetWare, no additional configuration is required on Apache.

4 **(Optional)** If you selected Auth Module or Auth DBM Module as your access control type in Step 2, type the absolute path to the password file in the User File field and the group password in the Group File field (if you created one).

For more information about using password files, see Authentication (http://httpd.apache.org/docs-2.0/howto/auth.html) on the apache.org Web site.

5 Type the context in the directory where the search for user rights should begin.

For example, o=employees.

HINT: For more information about this step and the following three steps, see the AuthLDAPUrl (http://httpd.apache.org/docs-2.0/mod/mod_auth_ldap.html#authldapurl) directive on the apache.org Web site.

6 Select which attribute should be searched for by clicking either UID or CN.

UID is the recommended context on which a search should be performed.

7 Select the scope of the search by selecting either Subtree or Container Only.

If you know your users are stored in a specific container, select Container Only, especially if your tree is large. This searches the container you specified in the Base DN for Search field. Otherwise, select Subtree.

8 Select Yes to enable Secure LDAP as a method of protecting usernames and passwords from being intercepted.

If you do not want to enable secure LDAP, click No.

For more information about securing LDAP, see the LDAPTrustedCA (http://httpd.apache.org/docs-2.0/mod/mod_ldap.html#ldaptrustedca) directive on the apache.org Web site.

9 Type the full path to the server certificate.

For example,

```
sys:\system\RootCert.der
```

10 From the Certificate Type drop-down box, select the type of certificate that is on your server.

On NetWare, the default certificate type is Der File.

11 Click Save and Save and Apply.

For more information about authentication to directories on your Apache Web server, see Authentication (http://httpd.apache.org/docs-2.0/howto/auth.html) in the Apache documentation.

Configuring User Home Directories

User home directories let you to set up home directories for each user in your directory. A great advantage to setting up home directories is that users can then access their own files using a Web browser. In addition, they can share information with the Web community by moving content into their own public_html directory. The public_html directory serves as the user's own primary document directory.

Complete the following tasks for each user who requires a home directory:

1. Create a home directory for each User object and specify the path to it.
2. Create a public_html directory in each user's home directory.
3. Select a method for how Apache should access the directory.

4. Activate User Document Directories from Apache Manager.

See the following sections for details on completing the above tasks.

About User Home Directories

A home directory is simply a directory that has been created and named after the user for whom it was created. Typically, home directories are created on a volume of the server dedicated for this purpose. You can create each user home directory using iManager or ConsoleOne®. This can be done either when you create each user object, or it can be enabled later on.

Once each home directory is created, you must specify the path to it within each User object in the directory. You might have already done this when you first created the directory.

For information about creating user home directories, see the *ConsoleOne 1.3.x User Guide*.

Creating Public_HTML Directories

The public_html directory is the user's personal primary document directory. Whatever is placed in the public_html directory is typically visible to all other users.

Create the public_html directory as a subdirectory within each of the users' home directories. To help your users, you could create a default index.html file and place it in their public directories. That way, they will get something when they point their Web browsers at the new directory for the first time, which could prevent support calls.

Selecting a Method for Accessing eDirectory

In order for Apache to authenticate to eDirectory, choose one of the following two methods: assign rights to the public user object in eDirectory, or create a new generic user object.

Regardless of which method you choose, you must then assign the following attributes to the container where the user object is stored:

- Home Directory Rights
- Host Resource Name
- Host Server

While you can assign these rights at either the context or individual user levels, we recommend that you assign them at the context level.

Creating a New Generic User Object

Using iManager or ConsoleOne, create a user object, such as hdiruser, in a container in your Novell® eDirectory™ tree. It doesn't matter where you create the object, so long as you assign the attributes described above to the container where you create the user object.

Using the Public User Object

If you want to use the public user object, make sure you assign the rights described above to the container where your public user object is stored.

Once you have chosen which user object to use and assigned proper attributes to the container where the user object resides, then you are ready to enable user home directories on Apache.

Enabling User Home Directories on Apache

Before user home directories can work you must first enable it. Once enabled, users can view the content of their user home directory by typing the domain name, followed by a slash (/), followed by *~usersname*.

To enable user home directories:

1. From the User Home Directories page of Content Management, click On.
2. In the User URL Prefix, specify the character to be used to indicate to Apache the text that follows is referring to a user home directory.

   The default character is ~ because it is the most expected character in use today for home directories. But you can specify any character or number.
3. In the Subdirectory field, type the name of the directory you created for each user as their primary document directory.

   The default name is public_html, although it is whatever name you used when you created the public directory within the user home directory.
4. In the User Search Contexts field, specify the search context where your user objects are stored.

   Because this is done using LDAP, you must specify the user contexts using LDAP syntax, which requires commas rather than periods for separating multiple contexts, and no leading periods. For example,

   ```
   ou=provo,ou=novell
   ```

   The search begins in the specified context and searches all sub-contexts until the user is found.
5. Under Configuration Option, select which method (Assign Public Rights or Username and Password) to use for logging Apache in to eDirectory:

   For more information, see "Selecting a Method for Accessing eDirectory" on page 43.

   WARNING: Using this option places the username and password in Apache's httpd.conf configuration file. If a user can access this file, they could identify the username and password and thereby have access to eDirectory. However, if you have assigned the proper read-only attributes to the generic user, the user would only be allowed to browse user directories.
6. (**Optional**) If you selected Username and Password as your configuration option, enter the username and password of the user object you created in eDirectory.
7. Click Save.
8. Click Save and Apply.

Changing the Default Index Filename

If a document name is not specified in a URL, Apache looks for a specific filename such as index.html and returns it to the Web browser. Which filename the Web Server looks for can be configured from the Document Preferences page under Content Management. If the specified filename cannot be found, the Web browser displays a listing of files and folders located at the URL.

By default, Apache defines index.html as the default home page filename, but you can set this to whatever filename you choose.

If more than one name is specified, the server searches in the order in which the names appear in this field until one is found. For example, if your index filenames are index.html and home.html, the server first searches for index.html and, if it doesn't find it, the server then searches for home.html.

If Apache can't find a filename that matches the default index filename, and if the requested directory has directory indexing enabled (see "Directory Indexing" on page 40), Apache generates its own index file that lists the contents of the directory.

For example, a request for http://myserver/docs/ would return http://myserver/docs/index.html if it exists, or would list the directory if it did not.

Keep in mind that the default index file does not need to be relative to the directory. For example, any of the following would work:

- index.html
- index.txt
- /cgi-bin/index.pl

Including three of these in order would cause the /cgi-bin/index.pl CGI script to be executed if neither index.html or index.txt existed in a directory.

To change the current default index filename:

1. On the Document Preferences page under Content Management, type a filename in the Index File Name field.
2. Click Save, and then Save and Apply.

For more information, see the DirectoryIndex (http://httpd.apache.org/docs-2.0/mod/mod_dir.html#directoryindex) directive on the Apache Web site.

Redirecting Visitors to an Alternate URL

URL forwarding is a method for the Web server to tell a user that a URL has changed—for example, if you have moved files to another directory or server. You can also use redirection to send a person who requests a document on one server to a document on another server.

To map a URL to another server, you must first specify the prefix of the URL you want the server to redirect. Then, you need to choose which URL to redirect to. You can redirect to a URL prefix if the directory on the new server is the same as in the mapped URL; you can also redirect to a fixed URL (hostname, directory, and filename).

To define a URL to be forwarded:

1. From the URL Forwarding page of Content Management, type the portion of the old URL to be forwarded.
2. In the Forward Requests To field, type the URL where requests should be forwarded to.
3. Click Save, and then Save and Apply.

If you forward to a URL prefix, the forwarding keeps the full pathname and substitutes one prefix for another. For example, if you forward http://www.novell.com/info/docs to a prefix cambridge.com, the URL http://www.novell.com/info/docs redirects to http://cambridge.com/info/docs.

However, if the directory structure on the new server is not the same as in the mapped URL, you could forward the URL to a fixed URL. For example, you could forward http://www.novell.com/info/docs to http://cambridge.com/new-files/info/docs.

Sometimes you might want to redirect requests for all the documents in one subdirectory to a specific URL. For example, if you had to remove a directory because it was causing too much traffic or because the documents were no longer to be served for any reason, you could direct a request for any one of the documents to a page explaining why the documents were no longer available. For example, a prefix on /info/docs could be redirected to http://www.novell.com/explain.html.

For more information, see the Redirect (http://httpd.apache.org/docs-2.0/mod/mod_alias.html#redirect) and Alias (http://httpd.apache.org/docs-2.0/mod/mod_alias.html#alias) directives on the Apache Web site.

Also, for more information about general issues surrounding URL redirection, see the URL Rewriting Guide (http://httpd.apache.org/docs-2.0/misc/rewriteguide.html) on the Apache Web site.

Configuring CGI Extensions

Common Gateway Interface (CGI) provides a method for Web servers to interact with external content-generating programs, which are often referred to as CGI programs or CGI scripts. It is the one of the simplest methods for adding dynamic content to your Web site.

Using Apache Manager, you can control how Apache finds the interpreter used to run CGI scripts. For example, if you specify sys:\foo.nlm as the path and .foo as the extension, all CGI script files with a .foo extension are passed to the foo interpreter (foo.nlm).

To define CGI extensions:

1. From the CGI Extensions page of Content Management, type the complete path to an interpreter used to run CGI scripts in the CGI Path field.

   HINT: On NetWare, you need to create an NLM^{TM} that can serve as your CGI interpreter. Writing an NLM requires programming knowledge. If you are a developer, visit the Novell Developer Web site at http://developer.novell.com (http://developer.novell.com).

2. In the Extension column, type the extension Apache should recognize as a CGI file.

   The default extension is .cgi.

3. Click Add.
4. Click Save and Apply.

To edit a defined CGI path and extension:

1. From the CGI Extensions page of Content Management, click Edit in the row of the CGI extension you want to modify.

   HINT: If there are no CGI extensions listed below the Add CGI Extension section, none have yet been defined. Define one following the instructions above.

2. After making your changes, click Save.
3. Click Save and Apply.

To remove a CGI extension:

1. From the CGI Extensions page of Content Management, click Remove in the row of the CGI extension that you want deleted.
2. Click Remove and Save and Apply.

For a more thorough discussion of CGI on Apache, see Apache Tutorial: Dynamic Content with CGI (http://httpd.apache.org/docs-2.0/howto/cgi.html) on the Apache.org Web site.

Creating Virtual Hosts

The term *virtual host*—sometimes called virtual servers—refers to the practice of running more than one Web site on a single computer (such as, www.company1.com and www.company2.com). Virtual hosts can be IP-based, meaning that you have a different IP address for every Web site, or name-based, meaning that you have multiple DNS names assigned to a single IP address. Visitors to the Web sites are unaware that both sites are running on the same physical server.

For more information about IP-based virtual hosting, see IP-based Virtual Host Support (http://httpd.apache.org/docs-2.0/vhosts/ip-based.html). For more information about when and how to use name-based virtual hosting, see Name-based Virtual Host Support (http://httpd.apache.org/docs-2.0/vhosts/name-based.html).

To create a virtual host:

1. On the Virtual Hosts page of Content Management, type the IP address of your server, followed by a colon and the port number you want to use.

   For example:

   ```
   123.456.789.100:80
   ```

   If you do not include a port number, Apache assumes port 80.

   IMPORTANT: If you are setting up a name-based virtual host and assigning an alternate port number, you must first configure Apache to listen to the port number you assign. See "Managing Listen Ports" on page 31.

2. (**Optional**) To instruct Apache to also listen on a secure port, press the Space bar and then add the same IP address followed by the secure port number. For example:

   ```
   123.456.789.100:443
   ```

3. In the Server Name field, type a hostname for your server, such as www.mycompany.com.
4. Select the Host Type to be used.

   If you are going to use one virtual host per IP/port combination, then you should select IP-based virtual hosting. Otherwise, select name-based virtual hosting.

5. Click Save > Save and Apply.

Creating Your Own Web Site

You can use any HTML editor to create a Web site, although most functional corporate Web sites are created by professional designers. But depending on your needs and resources, your implementation tool can range from any of the readily available Web site creation programs (some of which are free) to a team of programmers. Another avenue is to out-source the creation of your Web site.

Creating personal and departmental Web sites can be simple, requiring only minutes to assemble. You can use any HTML editor to create the pages of your Web site.

When you create your home page, save the file as index.htm or .html and that file automatically appears when your Web site is accessed. You can then create links to other pages and graphics with any filenames.

HINT: You can configure the Apache to recognize a specific filename and extension so that when a user enters your Web server's URL, it automatically displays your home page. See "Modifying the Primary Document Directory" on page 37.

Accessing Your Web Site

If you have already successfully installed NetWare 6.5 and the Apache Web server, you can access it right now. A sample Web page has been included. You can remove these pages and replace them with your own content.

To view the sample Web site, open a client Web browser on a workstation in your network and enter your NetWare server's IP address or DNS name. For example:

http://*server_IP_address*

or

http://*domain_name*

Adding Content to Your Web Site

Apache has a document root or primary document directory. By default, the path to the primary document directory is *volume*:/Apache2/htdocs. This is where the temporary index page is stored and where you will place your home page.

All content placed in this folder is visible to your Web site audience. If necessary, you can easily specify another directory as the primary document root directory. (See "Modifying the Primary Document Directory" on page 37.)

Once Apache is running, you can start posting content for the world (or your department or company) to see by placing files in Apache's primary document directory. You can also create additional document directories, which is a good idea if departments want to publish their own content to the company Web site but you don't want to give users control of the primary document directory. (See "Adding an Additional Document Directory" on page 38.)

What's Next

Once you've created content and configured the server to run optimally, you might want to learn more about Apache modules and how to enable the Mod_PHP, Mod_Perl, Mod_NSN, and Mod_cach modules for use in hosting dynamic content. (See Chapter 5, "Managing Apache Modules," on page 49.)

5 Managing Apache Modules

One of the strengths of Apache is its ability to extend the power of the Web server through the use of modules. In fact, most of the functionality that exists in the Apache Web server is provided by modules.

Using Apache Manager, you can enable or disable four modules that enable scripting technology on Apache. This chapter provides instructions for how to enable these modules—an easy procedure—and then discusses some of the Apache modules that are unique to NetWare®.

This chapter includes the following topics:

About Apache Modules

There are two types of Apache modules: external and internal (or built-in). An external module contains a set of functions that are wrapped up into a separate executable file. Having a module as a separate file allows the administrator to add, replace, or remove the module as needed. If a newer version of a module becomes available, the administrator simply has to copy the new executable file into the *volume*:\Apache2\modules directory and restart the server. On NetWare, an executable file has an NLM™ extension. For example, mod_cache.nlm.

The second type of module is an internal (or built-in) module. An internal module also contains a set of functions, except that those functions are compiled right in to the Apache executable when it is compiled from the Apache source code. But internal and external modules function the same way.

HINT: An external module can be compiled directly into the Apache executable by simply including the source as part of the core Apache code.

Requests received by the Apache Web server must pass through a series of stages in order for them to be completely handled. The architecture of Apache allows a module to insert itself into any one or more of these stages.

Three of these stages deal with Web server security: access control, authentication, and authorization. There are currently various Apache modules available that supply handlers for one or more of these stages in order to give the Apache Web server a certain level of security.

For more information about security on Apache, see Authentication, Authorization and Access Control (http://httpd.apache.org/docs-2.0/howto/auth.html) on the Apache.org Web site.

Enabling Scripting Modules

To use Perl, Novell Sripting, or PHP, you must first enable each of their modules from the Modules page of Apache Manager.

To enable the PHP, Perl, or NSN modules:

1. From the Modules page, click the module name of the scripting language you want enabled.
2. Click Yes.
3. Click Save.

For more information about any of these scripting languages, visit the Novell Developer Kit (http://developer.novell.com/ndk) Web site.

Enabling the Caching Module

You can also enable Mod_cache, which implements an RFC 2616-compliant HTTP content cache that can be used to cache either local content or content available through a proxy. Once enabled, additional options appear that allow you to configure mod_cache.

To enable the caching module:

1. From the Modules page, click Mod_Cache.
2. Click On.
3. Click Save, and then Save and Apply.
4. In the Default Cache Expiration field, specify a default time, in seconds, to cache a document if neither an expiry date nor last-modified date are provided with the document.

   NOTE: The value specified in the Maximum Cache Expiration field does not override this setting.
5. In the Maximum Cache Expiration field, specify the maximum number of seconds that HTTP documents, capable of being cached, are retained without checking the origin server.

   This means that cached documents are never older than the number of seconds you specify here. This maximum value is enforced even if an expiry date was supplied with the document.
6. (**Optional**) Next to Ignore Cache Control, select Yes if you want Apache to cache documents that contain a no-cache or no-store header value.

   Some Web documents might contain a no-cache or no-store header value, which means that the Web server won't store them in the server's cache. Ignore Cache Control overrides these header values by telling the server to cache documents even if they contain these header values. Documents requiring authorization are never cached.
7. In the Cache Size field, specify the maximum amount of memory to be used by cache, in KBytes (1024-byte units).

   If a new object needs to be inserted in the cache and the size of the object is greater than the remaining memory, objects are removed until the new object can be cached. The object to be removed is selected using the algorithm specified by the MCacheRemovalAlgorithm directive.

See the MCacheRemovalAlgorithm Directive (http://httpd.apache.org/docs-2.0/mod/mod_mem_cache.html#mcacheremovalalgorithm) documentation on Apache.org for more information.

8. In the Maximum Object Count field, specify the maximum number of objects that can be cached.

   This value is used to create the open hash table. If a new object needs to be inserted in the cache and the maximum number of objects has been reached, an object will be removed to allow the new object to be cached. The object to be removed is selected using the algorithm specified by the MCacheRemovalAlgorithm directive.

   See the MCacheRemovalAlgorithm Directive (http://httpd.apache.org/docs-2.0/mod/mod_mem_cache.html#mcacheremovalalgorithm) documentation on Apache.org for more information.

9. In the Minimum Object Size field, specify the minimum size (in bytes) a document must be in order for it to be cached.

10. In the Maximum Object Size field, specify the maximum size (in bytes) a document can be in order for it to be cached.

For more information about the mod_cache module, see Apache Module mod_cache (http://httpd.apache.org/docs-2.0/mod/mod_cache.html) on the Apache.org Web site.

Using Mod_edir

Mod_edir adds authorization services to the mod_auth_ldap Apache authentication module. It requires that mod_auth_ldap be loaded before the edir_module because it relies on mod_auth_ldap for the authentication services. In addition, mod_edir also provides support for access to Novell® eDirectory™ based user home directories and remote file systems.

This module can only be used on NetWare and relies on eDirectory and the NetWare file system for file rights enforcement.

Mod_edir includes the following directives:

- eDirServer
- eDirUserAccount
- eDirPassword
- eDirCacheTimeout
- hDirUserTag
- hDirUserSubDirectory
- hDirSearchContexts
- HomeDirEnabled
- RemoteDirEnabled
- Require edir-user

Anonymous Versus Authenticated Modes

Mod_eDir has the ability to provide authorization, home directory and remote file access functionality. In order to provide this functionality, mod_eDir must be able to make a connection to eDirectory as well as to remote servers. There are two modes in which mod_eDir can make these connections. These two modes are "Anonymous" and "Authenticated" modes. The basic difference between the two modes is whether mod_eDir accesses the information in eDirectory or remote file systems through public rights or uses a special user ID and password to login.

Anonymous Mode

When mod_eDir is configured in anonymous mode, it does not need to use a user ID or password to login before extracting information from eDirectory or a remote file system. In order for anonymous mode to work correctly, the administrator must allow public access to certain attributes within eDirectory. The most important attribute that is required by mod_eDir is the "Home Directory" attribute of each user object. This attribute stores the server, volume and path to each user's home directory.

Two requirements must be satisfied before anonymous mode works correctly. The first requirement has to do with allowing access to the "Home Directory" attribute of each user object within eDirectory. The second requirement deals with allowing access to a remote server's file system. When a request is made to retrieve a web page from a user home directory, the URL should contain the home directory tag followed by a user ID (i.e. http://myserver.com/~bnicholes/index.html). Mod_eDir will then make an anonymous request through LDAP to retrieve the value of the "Home Directory" attribute of the specified user. If the home directory attribute has not been assigned public access rights, the anonymous request will fail to extract the required information. What this means is that the [PUBLIC] object within eDirectory must be allowed to read this attribute. In order to allow access to a remote server's file system, the Apache server must be able to login as server to the remote file server. Being able to login as server requires that the NetWare server that is running the Apache web server, must have a local eDirectory replica and the server object within eDirectory must have must have file scan and read right on the remote server's file system.

Advantages:

- Does not require that the administrator stores a user ID and password on the file system in the clear.
- Configuring the remote directory and home directory support in the Apache configuration file is much easier and requires fewer directives.
- User home directory availability can be controlled by allowing or disallowing public access to the attribute for any given user object.

Disadvantages:

- Requires that the administrator gives public access rights to either the entire eDirectory tree or to the "Home Directory" attribute of each individual user that is allowed home directory functionality.
- Requires administrator intervention before a new user is able to access his home directory through the web.
- A local replica of the eDirectory tree must exist on the NetWare server that is running the Apache Web server.

- The server object of the NetWare server that is running the Apache Web server must be given rights to all remote file systems it intends to access.

Authenticated Mode

Configuring mod_eDir in authenticated mode allows it to free access all of the required information both in eDirectory as well as remote file systems without having to assign public access rights. But authenticated mode requires that a user ID and password be stored in an Apache configuration file. It also requires that a user object for the Apache Web server be created within eDirectory and assigned all of the necessary rights to allow it to access the "Home Directory" attribute of all user objects and File Scan and Read rights to all remote file systems that it intends to access.

We suggest that the user ID and password not be stored in the Apache httpd.conf configuration file or any other primary configuration file, but rather store them in a separate file that can be secured through additional file system rights. In other words, create an addition.conf file that holds only the directives for specifying the user ID and password to the Apache user object. Then either place the additional.conf file in a secure location on the file system or assign sufficient rights to the file so that only an administrator can view it. Then from within the httpd.conf file, simply include the additional.conf file wherever necessary. Also, for additional security, you might want to assign only administrator rights to the httpd.conf file.

Advantages:

- Does not require administrator intervention before a user is able to access the home directory through the web.
- Allows the Apache module to bind directly to LDAP rather than having to depend on public rights granted through eDirectory.
- Allows the Apache server to acquire the "Home Directory" attribute information from any LDAP server rather than requiring a local replica of eDirectory.
- All access to home directories and remote file systems can be controlled through a single Apache user object within eDirectory.

Disadvantages:

- Requires that a password be stored on the file system of the NetWare server.
- Requires the administrator to create an "Apache" user object and grant it the appropriate read and file scan rights for both the user objects and the remote server file systems before home directory and remote directory functionality is available.

Combining Mod_edir with Mod_auth_LDAP: An Example

The example below shows how mod_edir can be combined with mod_auth_ldap to provide both authentication and authorization services:

```
LoadModule ldap_module modules/utilldap.nlm
<IfModule util_ldap.c>
   LoadModule auth_ldap_module modules/authldap.nlm
   LoadModule edir_module modules/mod_edir.nlm  Alias /secure sys:/webpages/
secure
  <Directory sys:/webpages/secure>
    Order deny,allow
    Allow from all
    AuthType Basic
    AuthName LDAP_Protected_Site
    AuthLDAPURL ldap://my.ldap.server/o=my_context
    require edir-user
  </Directory>
</IfModule>
```

The following is an example that shows an anonymous mode configuration of mod_edir for home directory and remote directory support:

```
LoadModule edir_module modules/mod_edir.nlm
<IfModule mod_edir.c>
   hDirSearchContexts o=users   Alias /rdocs "remotesrv/data:/webpages/
remote"
   <Directory "data:/webpages/remote">
      Options Indexes MultiViews
      Order allow,deny
      Allow from all
   </Directory>
</IfModule>
```

The next example shows an authenticated mode configuration of mod_edir (in httpd.conf):

```
LoadModule edir_module modules/mod_edir.nlm
<IfModule mod_edir.c>
   include edirauth.conf   hDirSearchContexts o=users   Alias /rdocs
"remotesrv/data:/webpages/remote"
   <Directory "data:/webpages/remote">
      Options Indexes MultiViews
      Order allow,deny
      Allow from all
   </Directory>
</IfModule>
```

The following is in the edirauth.conf file:

```
<IfModule mod_edir.c>
   eDirServer MY_SERVER
   eDirUserAccount cn=apache_server.o=admin_objects   eDirPassword secret
</IfModule>
```

eDirServer

Description: Specifies the eDirectory server that will be access through LDAP

Syntax: eDirServer <Server-Name>

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use the eDirServer directive to specific the server that will be used to login in and extract eDirectory information. This directive is only required if running in authenticated mode. (See "Anonymous Versus Authenticated Modes" on page 52).

eDirUserAccount

Description: Specifies a user ID for logging into eDirectory

Syntax: eDirUserAccount <User-ID>

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use the eDirUserAccount to specify the user ID of the eDirectory user object that has been granted rights to access eDirectory information such as the "Home-Directory" attribute of each user object and any remote file system that will be accessed from the Apache server box. Please see the explanation of the authenticated vs. anonymous modes.

eDirPassword

Description: Specifies the password the eDirectory user account password

Syntax: eDirPassword <Password>

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use the eDirPassword to specify the password that corresponds to the user ID defined by eDirUserAccount. (See “Anonymous Versus Authenticated Modes” on page 52).

eDirCacheTimeout

Description: Specifies the number of seconds before a cache entry times out

Syntax: eDirCacheTimeout <nnn-seconds> (Default = 300 seconds)

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use the eDirCacheTimeout directive to specify the number of seconds each cache entry will remain in the cache before timing out. The default value if no time out value has been specified is 300 seconds. A cache time out value of 0 will disable the cache.

hDirUserTag

Description: Specifies the URL tag used to indicate a user home directory

Syntax: hDirUserTag <Tag> (Default = '~')

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use the hDirUserTag directive to change the default tag used on in the URL to indicate that the following name specifies a user. Mod_eDir will use the user name to look up that user's home directory in eDirectory and then attempt to server the requested web page from that location.

hDirUserSubDirectory

Description: Specifies the subdirectory name with in a user home directory

Syntax: hDirUserSubDirectory <Sub-Dir-Name> Default ("public_html")

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use the hDirUserSubDirectory directive to change the name of the default subdirectory where mod_eDir will attempt to access the requested web page. After mod_eDir has extracted the user home directory from eDirectory, it then appends the name of the sub-directory specified by hDirUserSubDirectory and then attempts to access the requested web page from that location. The default location for any user would be "server/volume:/home-dir-path/public_html".

hDirSearchContexts

Description: Specifies a list of search contexts

Syntax: hDirSearchContexts <Context, Context, ...>

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use hDirSearchContexts to specify a list of contexts that will be search to resolve a user ID to a user home directory. By default each context and all sub-contexts will be searched until a matching user ID is found. Mod_eDir will stop searching as soon as it finds a matching user ID. Therefore all user ID's must be unique within the search contexts specified.

HomeDirEnabled

Description: Enables or disables user home directory support

Syntax: HomeDirEnabled <On|Off> (Default = On)

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use HomeDirEnabled to enable or disable the user home directory support in mod_eDir. The default is to enable home directory support.

RemoteDirEnabled

Description: Enables or disables remote directory support

Syntax: RemoteDirEnabled <On|Off> (Default = On)

Context: server config, virtual host

Status: Extended

Module: mod_edir

Use RemoteDirEnabled to enable or disable the remote file system access support in mod_eDir. The default is to enable remote file system support.

Require edir-user

Description: Specifies that only an eDirectory user has access a resource

Syntax: Require edir-user

Context: directory, .htaccess

Override: AuthConfig

Status: Extended

Module: mod_edir

Require edir-user must be accompanied by AuthName and AuthType directives, and AuthLDAPURL in order to work correctly (see the example above).

Access controls which are applied in this way are effective for all methods. This is what is normally desired. If you wish to apply access controls only to specific methods, while leaving other methods unprotected, then place the Require statement into a <Limit> section.

See the mod_auth_ldap documentation (http://httpd.apache.org/docs-2.0/mod/mod_auth_ldap.html) on the Apache.org Web site.

What's Next

For more information about Apache modules, see the Module Index (http://httpd.apache.org/docs-2.0/mod) on the Apache.org Web site.

6 Managing Multiple Instances of Apache

If you have several Apache Web servers running anywhere in your network, on any platform, they can all be configured at one time from the Multiple Server Administration interface of Apache Manager.

Multiple Server Administration uses a special configuration daemon to communicate with Novell® eDirectory™ where bits and pieces of the httpd.conf Apache configuration file are stored as inheritable objects. These objects can then be shared between Web servers, letting you synchronize and share common configuration settings across all of your Web servers. This is ideal for managing a Web farm, a term used to refer to a collection of Web servers.

IMPORTANT: This mode of Apache Manager requires that you have a good understanding of Apache directives and how to use them. You will be required to type directives and to know the correct syntax. Also, a basic understanding of the object hierarchy used by eDirectory is helpful, though not required.

This chapter contains the following topics:

About Multiple Server Administration

The Multiple Server Administration interface of Apache Manager stores Apache configurations in eDirectory so that a change to one Web server's configuration can be inherited by all other servers defined in a Server Group. This makes server management and configuration faster, easier, and more accurate.

To do this, Apache Manager uses two key components: eDirectory and a configuration daemon.

A basic understanding of these two components can help you to better understand and use the Multiple Server Administration interface.

eDirectory

eDirectory is used by Apache Manager in two ways: first, as a database where Apache configuration directives are stored, and second, as an environment that allows Apache configuration objects to be shared and inherited.

Apache Manager divides the Apache configuration file (httpd.conf) into a hierarchy of configuration objects and then stores them in eDirectory. By storing directives in a hierarchy of objects, they can be applied to a single server, a group of servers, or to an entire Web farm.

Some blocks are unique from others, such as virtual host or Apache module blocks, because they utilize directives unique to them. Therefore, they appear as unique and separate objects.

The httpd.conf file is divided into a set of five object classes: server group, server, virtual host, module, and block. These object classes are defined in the following table.

Object Class	Definition
Server Group	Represents a set of Apache configuration directives common to all server objects contained in a group. It can contain any number of server groups, servers, modules, and blocks.
Server	Represents a single Apache server. It contains standard attributes such as Server Name. It is used to define any single server and serves as an anchor point for each server's configuration. It can contain any number of virtual hosts, modules, and blocks.
Virtual Host	Represents a virtual host within an instance of an Apache server. It contains the necessary attributes to create a VirtualHost block in the Apache configuration file. It must be contained within a server object and can itself contain any number of block objects.
Module	Represents an Apache module. It defines the LoadModule directive and the IfModule tag within a configuration file. A module can be defined at any level of the hierarchy so that it can be inherited by one or more server configurations. This allows the module to be loaded and configured in exactly the same way by multiple Web servers without having to redefine the module for each server. It can contain any number of block objects.
Block	Literally, blocks are specific directives used to enclose a set of configurations. A block object represents a Directory, Location, or File block, or any of their derivatives. A block object defines the Directory|Location|File tags within a configuration file. It can be defined at any level of the hierarchy so that one or more server configurations can inherit it. This allows the block definition to be applied in exactly the same way by multiple Web servers without having to redefine the block for each instance. A block cannot contain any other objects.

By defining a server object within a directory and combining the object with one or more virtual host, module, and block objects, an entire configuration for an Apache Web server can be stored, manipulated, and shared.

Each object class contains a set of attributes that store the data that is required to produce a portion of the configuration in a complete httpd.conf file. Additionally, each object class can store any number of directives that you might want defined at that level of the object hierarchy.

Configuration Daemon

The configuration daemon is a small Java application that runs along side the Apache server software. It serves as a conduit between each Web server's actual configuration file and the directory service that holds Apache's configuration objects.

The configuration daemon extracts Apache directives from directory server objects, assembles them, and then creates a new httpd.conf configuration file. Once the configuration file is created, the daemon restarts Apache so that new changes can be read by the Apache Web server.

HINT: When Apache Manager is started, so is the configuration daemon. However, if you need to start the daemon manually, type **ap2webman** at the NetWare® system console. Also, using this command automatically imports your current Apache Web server configurations, including any virtual hosts. For more information, see "Adding or Removing Servers to or from a Server Group" on page 65.

The first time the daemon is run for a specific instance of the Apache Web server, it creates a server object in the object hierarchy and ensures that the configuration stored in the directory matches the current configuration file. The daemon then continues to monitor the directory for any changes made to a particular server. If you manually change the configuration file stored on the server's hard drive, the daemon detects the change and imports it into the server configuration that was previously stored in the directory.

1 Configuration daemon is started

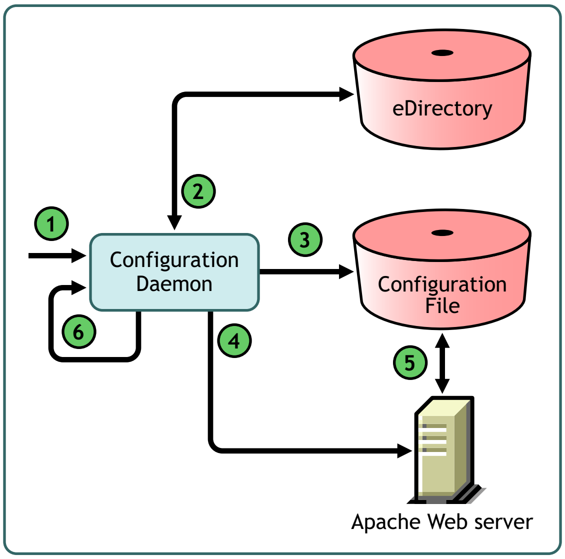
2 New configuration data is extracted from the directory

3 Configuration data is written to a configuration file on the local file system

4 Apache Web server is started and restarted

5 Apache reads the configuration file

6 Configuration daemon waits for changes in the directory and restarts from step 2 when necessary



This process ensures that the configuration file used to configure a specific instance of Apache remains synchronized with the shared configuration objects in the directory. If a change is detected, it updates the configuration file and notifies Apache that it is time to reload the configuration file. If a configuration has not yet been created in the directory, the daemon imports the current configuration file.

About the startup.properties File

The startup.properties file stored in *volume*:\Apache2\conf\daemon is used to configure the daemon. By default, it contains the following configurations:

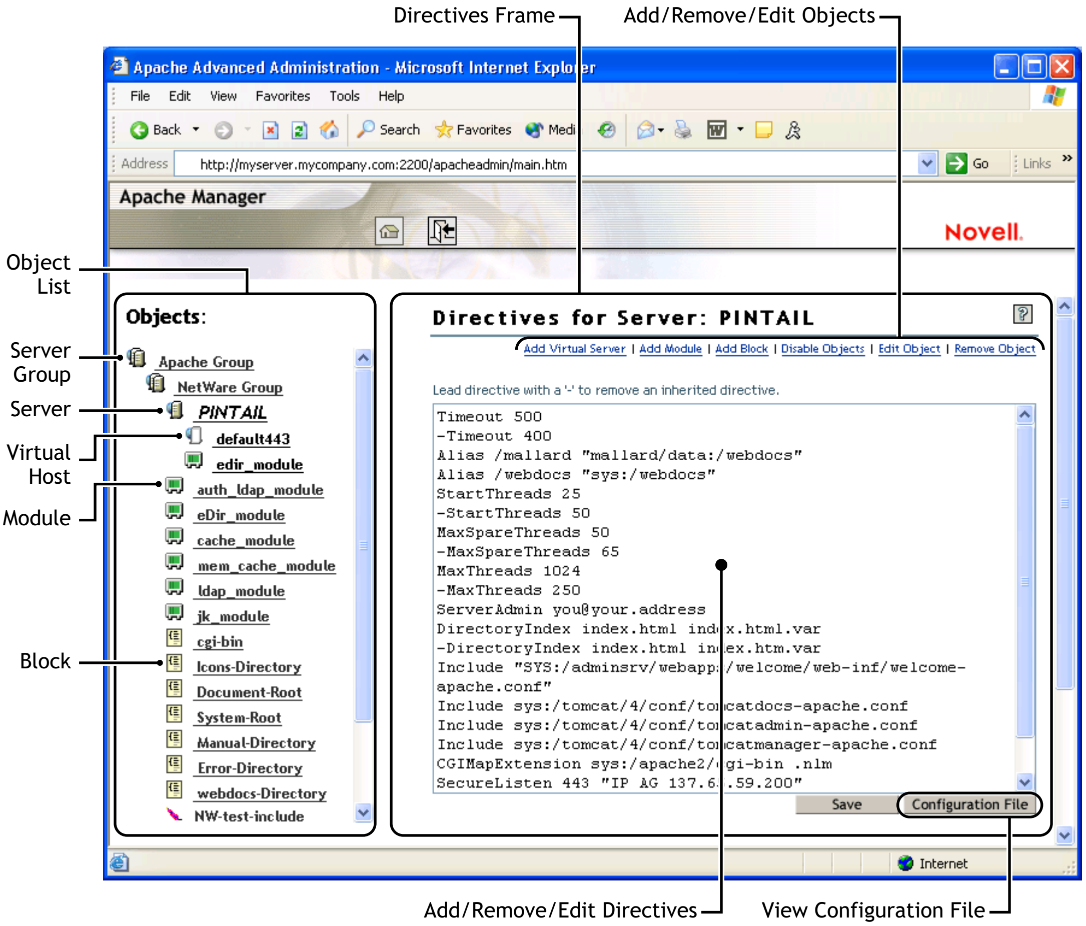
```
# NWConfVersion = 2

InitialContextFactory = com.sun.jndi.ldap.LdapCtxFactory

ProviderURL = tycoon.provo.novell.com
```

```
#Port = 389

Port = 636

UseSSL = yes

SecurityAuthentication = simple

UserID = cn=admin,o=cents

#Password = <Your-Password>

ServerDN = cn=TYCOON,cn=NetWare Group,cn=Apache Group,o=cents

ServerName = tycoon.provo.novell.com

ConfigFile = sys:/apache2/conf/httpd.conf

BackupDir = sys:/apache2/conf/backup/

StartApacheCmd = sys:/system/ap2webup.ncf

RestartApacheCmd = sys:/system/ap2webrs.ncf

StopApacheCmd = sys:/system/ap2webdn.ncf

RestartDelay = 10000

StartupErrorLog=sys:/apache2/logs/startup.err

StorageMode = FILE
```

Depending on changes you might make to your server, you might have to modify some of the paths. Most critically, you might have to make changes to UserID and ServerDN, making sure that if you move your administrator user object, you must specify the new location in this file.

Using the Multiple Server Administration Interface

The Multiple Server Administration interface displays the current configuration for each Apache server as it is stored in the directory and lets the you manipulate each configuration object.



Directives for Server: PINTAIL

Lead directive with a '-' to remove an inherited directive.

```
Timeout 500
-Timeout 400
Alias /mallard "mallard/data:/webdocs"
Alias /webdocs "sys:/webdocs"
StartThreads 25
-StartThreads 50
MaxSpareThreads 50
-MaxSpareThreads 65
MaxThreads 1024
-MaxThreads 250
ServerAdmin you@your.address
DirectoryIndex index.html index.html.var
-DirectoryIndex index.html index.htm.var
Include "SYS:/adminsrv/webapps/welcome/web-inf/welcome-
apache.conf"
Include sys:/tomcat/4/conf/tomcatdocs-apache.conf
Include sys:/tomcat/4/conf/tomcatadmin-apache.conf
Include sys:/tomcat/4/conf/tomcatmanager-apache.conf
CGIMapExtension sys:/apache2/cgi-bin .nlm
SecureListen 443 "IP AG 137.65.59.200"
```

The Objects Frame

The Objects frame on the left provides a view of the Apache configuration objects that are currently stored in the directory and lets the administrator navigate the hierarchy.

By default, the object hierarchy begins with a server group object called Apache Group. You can then define additional groups. Server groups can be organized in whatever way you choose. You might organize them according to platform (such as NetWare, Linux*, or UNIX) or by deployment (such as a Novell iFolder® group and an iPrint group).

The Directives Frame

The Directives frame—the frame to the right of the Objects frame—displays the configuration contents held by the currently selected object. This frame lets you manage the directives that are contained in the currently select object. You can add child objects or alter the attribute values of the object itself.

When working with server objects, the Directives frame also lets you preview the server's configuration file as it will be created by the configuration daemon. In the Configuration File view, which you can see by clicking Configuration File, all directives are linked back to the object where they were first defined, making it easier to find an object that holds a specific directive without having to know where the directive came from.

Starting Multiple Server Administration

Once you understand how the Apache Manager's Multiple Server Administration interface works with eDirectory and the configuration daemon, you can begin creating and defining your server groups.

To start Apache Manager's Multiple Server Administration:

1. Using a Web browser, open the secure version of the NetWare 6.5 Welcome Web site using your server's URL. For example,

   ```
   https://myserver.mycompany.com:2200
   ```

   or

   ```
   https://123.456.789.100:2200
   ```

   HINT: For more information about the Welcome Web site, see NetWare 6.5 Welcome Web Site in the *NetWare 6.5 Administration Overview*.

2. When prompted, enter your administrator username and password and click Login.
3. In the left frame of the NetWare 6.5 Welcome Web site home page, click ⊞ next to Open Source.
4. Click Apache 2.0.
5. Under Apache 2.0 Links, click Administer Multiple Apache Servers.

Creating Server Groups

Server groups are created to hold Apache servers that use common directives. This allows you to modify directives from one location that are automatically applied to all servers in the group.

For example, directives for Apache running on NetWare will have different values than Apache running on Linux. If you were running Apache on both NetWare and Linux, you might create a NetWare server group and a Linux server group.

To create a server group:

1. **(Optional)** From the Multiple Server Administration home page, or the first page you see after logging in, click Apache Group if you want to create the new server group under the Apache Group.

   If you want to create the new server group on the same level as the Apache Group, skip this step and proceed to Step 2.

2. Click Add Server Group.
3. In the Add Server Group dialog box, type a name for your server group in the Common Name field.

   This can be any name you choose. Keep in mind that it is the name that will appear in the Objects list.

4. Click OK.

Repeat this procedure for each server group you want to create. Once you have created your server groups, you can create additional server groups within them by following the same procedure, or you can begin adding servers to each server group.

Adding or Removing Servers to or from a Server Group

Once a server group is created, you then add Apache Web server objects to it. A server object contains all configurations specific to a single Apache server. However, server objects inherit configurations from objects above them in the Objects list unless you specifically disinherit them. (See “Viewing and Editing An Object’s Configuration” on page 69.)

You can manually add a server to a server group, or you can import it along with all of its configuration settings using the **ap2webman** command at the server console. Importing a server is much faster and more accurate.

To automatically import a server and its settings:

1. From the NetWare console of the server that is running the Apache Web server that you want to import, go to the system console prompt.
2. Enter **ap2webman** to run the configuration daemon.
3. When prompted, enter your administrator password and press Enter.
4. When prompted, press Y to import the server and its configurations.

Repeat this procedure on each server that you want imported.

HINT: If you want to import Apache Web server configurations from Linux or Windows*, you can download the configuration daemon from the Novell Developer Kit (NDK) and run it on those servers as well. You can also download LDIF files for use in adding Windows and Linux groups to eDirectory.

To manually add a server to a server group:

1. From the Objects list, click a server group name to select it, and then click Add Server.
2. In the Add Server dialog box, type a name for your server in the Common Name field.
3. In the Server Name field, type the full DNS name of your server. For example:

   ```
   www.myserver.com
   ```

   HINT: If you don’t know the DNS name of your server, contact a network administrator who manages your DNS server. If you simply want to test your server and have not yet registered with a DNS server, you can edit the hosts files on both NetWare and your client computer, which lets you see your server from any client where you have correctly modified the hosts file.

   On NetWare, the hosts file is in the *volume*:\\ETC directory. On Windows, search for the hosts file, which on Windows XP is typically in the *drive*:\\winnt\system32\drivers\etc folder. For both hosts files, add the IP address, port number, and a temporary DNS name at the end of each file. For example:

   ```
   137.65.67.132   tycoon2.provo.novell.com
   ```
4. Click OK.

When you have added the server, you must add all of the necessary Apache directives. Begin by clicking on the server from the Objects list and then typing (or copying and pasting) the directives directly into the Directives frame.

To remove a server from a server group:

1. From the Objects list, click a server group that contains the server you want to remove.
2. Click on the server you want removed and click Remove Object.

   WARNING: Once you remove a server, the only way to get it back is to re-create it either manually or by importing it again using the **ap2webman** command.

3 Click OK in the Remove Server dialog box.

Adding an Apache Module to a Server or Group Object

Module objects create an IfModule *mod_name.c* block in the configuration file. Adding a module to a server group makes it available to all other servers in and below the current object's group. For more information about Apache modules, including those that are unique to NetWare, see Chapter 5, “Managing Apache Modules,” on page 49.

When you add a module, you must know the name of its executable file and its module identifier. Before you add a module, refer to its documentation on the Apache Web site. See http://httpd.apache.org/docs-2.0/mod/ (http://httpd.apache.org/docs-2.0/mod).

To add an Apache module to a server or group object:

1 From the Objects list, click the server group that you want to add the module to.

2 In the Add Module dialog box, type a name for the Apache module in the Common Name field.

3 In the Source File field, type the name of the file that contains the code of the module you are adding. For example:

```
mod_auth_ldap.c
```

4 In the Object File field, type the relative path to the module’s executable file, which by default is in the *volume*:\\Apache2\modules directory. For example, type:

```
modules/authldap.nlm
```

5 In the Module Identifier field, type the string that identifies the module you are adding.

Each Apache module has a module identifier assigned to it. To find out which module identifier to use, look up the module name in the Apache documentation. Visit http://httpd.apache.org/docs-2.0/mod/ (http://httpd.apache.org/docs-2.0/mod).

6 (**Optional**) To disable the module, select the Disable Module check box.

NOTE: When you check the Disable Module check box, the LoadModule statement is not added to the httpd.conf configuration file.

7 Click OK.

Adding, Editing, or Removing Apache Blocks

Apache blocks are directives used to enclose a set of configurations. For example, the Directory directive is used to enclose a group of directives that apply only to a directory (and its subdirectories) specified within the enclosure. For example:

```
<Directory sys:\Apache2\htdocs>

     Options Indexes FollowSymLinks

</Directory>
```

In this example, the directives included between the Directory tags (compare to standard HTML tag syntax) are all applied as part of the Apache block.

A block can be defined at any level of the objects hierarchy so that one or more server configurations can inherit it. This allows the block definition to be applied in exactly the same way by multiple Web servers without having to redefine the block for each server. A block cannot contain any other objects.

Before adding a block, you might want to look at the documentation for its directives in the Directive Quick Reference (http://httpd.apache.org/docs-2.0/mod/quickreference.html) on the Apache.org Web site.

To add a new block:

1. From the Objects list, click a server group, server name, or virtual host object where you want to add the block object.
2. In the Add Block dialog box, type a name for the block object in the Common Name field.
3. In the Scope field, type the file path used in the block's open statement.

   Scope refers to the file path specified in the open statement of a block. For example, if the block type is Directory, the block’s open statement might be:

   ```
   <Directory sys:\Apache2\htdocs>
   ```

   In this case, you would enter sys:\Apache2\htdocs in the Scope field.
4. Click the Block Type drop-down list and select the type of block being defined.

   Because each block type is an actual directive, you can look up the directive in the Apache documentation for more information about each block type. See the Directive Quick Reference (http://httpd.apache.org/docs-2.0/mod/quickreference.html) on the Apache.org Web site.
5. Click OK.

To edit a block:

1. From the Objects list, click a server group, server name, module, or virtual host object that contains the block you want to edit.
2. Click the block that you want edited and then click Edit Object.
3. In the Edit Block dialog box, make the necessary changes.
4. Click OK.

To remove a block:

1. From the Objects list, click a server group, server name, module, or virtual host object that contains the block you want to remove.
2. Click the block that you want removed and then click Remove Object.

   WARNING: Once you remove a block, there is no way to retrieve it. You must create it again.
3. Click OK.

Adding, Editing, or Removing a Virtual Host

Before you can add a virtual host, you must first create it. For information about creating virtual hosts on Apache, see “Creating Virtual Hosts” on page 47.

When you add a virtual host object to the Objects list, a VirtualHost *ip:port* block is created in the configuration file that references the virtual server you are adding.

To add a virtual host:

1. From the Objects list, click a server group and then the name of the server where you want to add a virtual host.
2. In the Add Virtual Server dialog box, type a name for the virtual host in the Common Name field.
3. In the Server Name field, type the full DNS name of the virtual server. For example:

   ```
   myserver.mycompany.com
   ```
4. In the IP Address:Port Combinations field, enter the IP address and port number assigned to your virtual host. For example:

   ```
   123.456.789.100:80
   ```
5. Click OK.

To edit a virtual host:

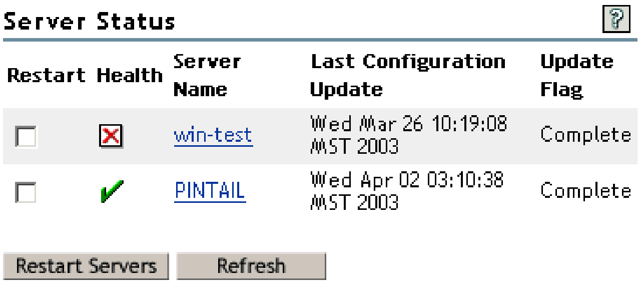
1. From the Objects list, click the virtual host that you want to edit.
2. Click Edit Object.
3. In the Edit Virtual Server dialog box, modify the servername or IP and port information.
4. Click OK.

To remove a virtual host:

1. From the Objects list, click the virtual host that you want to remove.
2. Click Remove Object.
3. In the Remove Virtual Server dialog box, click OK.

Checking the Status of Each Web Server

You can verify the health of each individual Web server and you can restart each one individually, from the Server Status table.



Server Status

Restart	Health	Server Name	Last Configuration Update	Update Flag
□	☒	win-test	Wed Mar 26 10:19:08 MST 2003	Complete
□	✓	PINTAIL	Wed Apr 02 03:10:38 MST 2003	Complete

In the Server Status table, a ✓ indicates that Apache is currently responding to the daemon and the current configuration is valid. A ☒ indicates that Apache is down. Apache might be down at the request of an administrator.

The Last Configuration Update column indicates the date of the last configuration update by the Apache daemon. If the configuration file has been modified by hand, this field reads MOD_BY_HAND.

To restart one or more Apache Web servers from the Server Status list:

1. In the Directives frame of Multiple Server Administration, scroll down if necessary to the Server Status table.
2. Click the check boxes in the Restart column in the row of each server that you want restarted.
3. Click Restart Servers.

Viewing and Editing An Object's Configuration

The Directives frame displays the configuration of the current object, or the object you most recently selected from the Objects list. The directives shown are those that are unique to the currently selected object. Additional directives are inherited from objects higher up in the Objects list and are typically not visible (unless you click Configuration File, which lets you view all directives being applied to the current object).

For example, if you created a server called Tycoon in a server group called NetWare Servers and selected it from the Objects list, the Directives frame would display the following list of directives, or the current configuration for Tycoon:

```
StartThreads 25

-StartThreads 50

MaxSpareThreads 50

-MaxSpareThreads 65

MaxThreads 1024

-MaxThreads 250

Listen 82

ServerAdmin you@your.address

DirectoryIndex index.html index.html.var index.php index.pl

-DirectoryIndex index.html index.html.var

Include "SYS:/adminsrv/webapps/welcome/web-inf/welcome-apache.conf"

Include sys:/tomcat/4/conf/tomcatdocs-apache.conf

Include sys:/tomcat/4/conf/tomcatadmin-apache.conf

Include sys:/tomcat/4/conf/tomcatmanager-apache.conf

Include sys:/apache2/conf/mod_nsn.conf

Include sys:/apache2/conf/mod_perl.conf

Include sys:/apache2/conf/mod_php.conf

NameVirtualHost 137.65.67.132:80
```

Notice that the StartThreads directive appears twice. The first StartThreads directive sets Tycoon's total number of starting threads to 25. The second StartThreads directive is inherited from the configuration settings of the NetWare Group. Because Tycoon's own configuration already contains a StartThreads directive, Apache Manager disinherited the second StartThreads directive by adding a minus sign (-) at the start of the directive.

If you clicked the server group object NetWare Group to which Tycoon belongs, the directives frame would display the following list of directives, or the current configuration for the NetWare Group:

```
ServerRoot "SYS:/APACHE2"

ThreadStackSize    65536

StartThreads          50

MinSpareThreads       10

MaxSpareThreads       65

MaxThreads           250

MaxRequestsPerChild    0

SecureListen 443 "SSL CertificateDNS"

DocumentRoot "SYS:/APACHE2/htdocs"

ErrorLog "|SYS:/APACHE2/bin/rotlogs.nlm sys:/apache2/logs/error_log 5M"

CustomLog "|sys:/apache2/bin/rotlogs.nlm sys:/apache2/logs/access_log 5M"
common

Alias /icons/ "SYS:/APACHE2/icons/"

Alias /manual "SYS:/APACHE2/manual"

Alias /error/ "SYS:/APACHE2/error/"

ScriptAlias /cgi-bin/ "SYS:/APACHE2/cgi-bin/"
```

These directives are inherited by Tycoon and by all other objects below NetWare Group.

Disinheriting Directives

Directives that have been inherited from objects higher up in the Objects list can be disinherited. This allows common configurations to be shared among all Apache Web servers but also lets you customize each Web server’s configuration.

To disinherit directives from an object:

1. Select the server group, server name, virtual host, module, or block that you want to modify.
2. In the Directives frame, enter a minus sign (-) before each directive that you want disinherited.
3. Click Save.

Disabling Inherited Objects

You can disable inherited objects that the current Apache server (the one you last clicked in the Objects list) might inherit for its configuration. Doing so removes them from the configuration of the current server.

Inherited objects are inherited from objects one or more levels above the object that you want to modify.

To disable a server’s inherited objects:

1. Select the server that is inheriting objects that you want disabled.

2. Click Disable Objects.
3. On the Disable Objects For Server: servername page, select each object from the list of objects by clicking their check boxes.

   Objects that are not selected remain inherited.
4. Click Save.

What's Next

Refer to the following resources for more information about managing the Apache Web server:

- To learn more about how to improve the overall performance of Apache, see Performance Tuning Tips (http://httpd.apache.org/docs-2.0/misc/perf-tuning.html).
- If you are a developer and need more information about the inner workings of Apache, see Developer Documentation for Apache 2.0 (http://httpd.apache.org/docs-2.0/developer).
- For the complete Apache 2.0 documentation, see Apache HTTP Server Version 2.0 Documentation (http://httpd.apache.org/docs-2.0).

A Installing the Apache Manager Daemon on Windows or Linux

Novell® Apache Manager is a browser-based tool used for configuring the Apache Web server. It provides a graphical user interface to most of the Apache directives, making it easier to quickly (and more accurately) configure and manage Apache's behavior and performance.

Apache Manager also lets you manage multiple installations of Apache in your network, regardless of what other platforms they are running on.

This section provides instructions for installing the Apache Manager Daemon on Windows* or Linux and includes the following topics:

- "Downloading the Daemon Install File" on page 73
- "Before Installation" on page 73
- "Installing the Daemon on Windows" on page 73
- "Installing the Daemon on Linux" on page 75
- "Troubleshooting the LDAP Connection" on page 77

Downloading the Daemon Install File

The Apache Manager Daemon can be downloaded from the Novell Forge Web site (http://forge.novell.com/modules/news). Novell Forge is a collaboration tool for research and development of open-source products and solutions.

Visit http://forge.novell.com/modules/xfmod/project/?apache_manager (http://forge.novell.com/modules/xfmod/project/?apache_manager) to download the install.exe (for Windows) or install.bin (for Linux) file to the computer where you will install the daemon.

Before Installation

Before installing the Apache Manager daemon on either Windows or Linux, verify the following on the computer where you will be installing the daemon:

- Java 2 version 1.3.1 or later is installed, and the java.exe is in the Windows path on a Windows-based computer, or in the path environment variable if installing to Linux.
- The LDAP server is running and is accepting connections.
- Apache Web Server 2.0 or later is installed and running.

Installing the Daemon on Windows

1 Run the install.exe file.

2. Click Next on the Introduction page.

3. In the Choose Install Folder dialog box, type a path to where the daemon files should be installed, then click Next.

   A new directory called apacheadmin is created at the location you specify.

4. In the User Object FDN field, type the username the daemon should use to authenticate to the directory.

   The user you specify must be a member of the apchadmn-Administrators group on the LDAP server. The username is stored in the startup.properties file.

5. Type the password for the user you specified and click Next.

   Although the password is used to authenticate the user, it is not stored in a file. However, each time the daemon is started, you are prompted to enter this password.

6. In the LDAP Server Name field, type the DNS name or IP address of the LDAP server where the daemon should store the configuration file.

7. In the LDAP Server Port field, specify the LDAP server's port number.

8. If the port number you specified is secure (uses SSL), select Yes, then click Next.

9. In the Path to Apache.exe field of the Current Apache Path Information dialog box, type the path to the Apache executable file.

   The daemon must be able to start, stop, and restart the Apache server. Typically, Apache.exe is found in the Apache2\bin directory where Apache was installed.

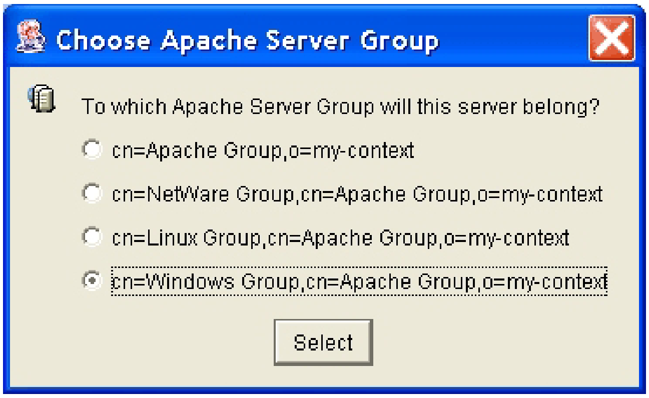
10. In the Path to Apache Configuration File field, type the path to Apache's primary configuration file and click Next.

    The Apache configuration file is typically found in the directory where Apache was installed.

11. In the Choose Apache Server Group dialog box, select an Apache server group where the Apache server should be placed and click Next.

    IMPORTANT: If you receive a message in place of this dialog box indicating that no group exists in the directory for the platform you are running on, click Yes to have a new group created.

    If neither of these dialog boxes appear, the install was unable to connect to the LDAP directory. You can exit the install and troubleshoot or you can manually enter this information in the next step and troubleshoot the LDAP connection when you run the daemon. For more information, see "Troubleshooting the LDAP Connection" on page 77.



Choose Apache Server Group

To which Apache Server Group will this server belong?

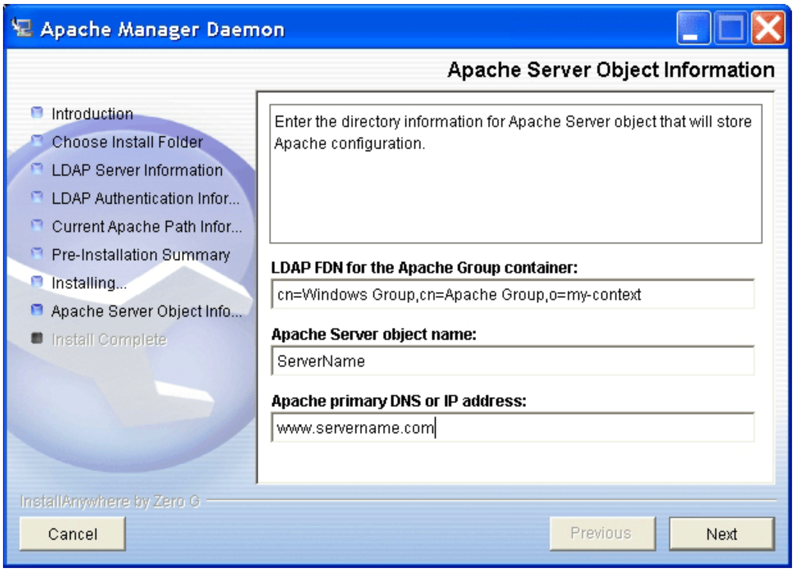
○ cn=Apache Group,o=my-context

○ cn=NetWare Group,cn=Apache Group,o=my-context

○ cn=Linux Group,cn=Apache Group,o=my-context

● cn=Windows Group,cn=Apache Group,o=my-context

12 If you were able to specify a server group in the previous step, the first field in the Apache Server Object Information dialog box should already contain the information it needs. Otherwise, you must specify the group server information manually.



Enter the directory information for Apache Server object that will store Apache configuration.

LDAP FDN for the Apache Group container: cn=Windows Group,cn=Apache Group,o=my-context

Apache Server object name: ServerName

Apache primary DNS or IP address: www.servername.com

13 In the Apache Server Object Name field, type the name of your Apache Web server.

14 In the Apache Primary DNS or IP Address field, type your Web server’s DNS name or IP address. For example, www.mycompany.com.

15 In the Pre-Installation Summary dialog box, review the information to ensure that it is correct, and then click Install.

16 When installation is complete, click Done.

To run the daemon, run Ap2webman.bat located in the apacheadmin directory.

For more information about the Apache Manager daemon and how it is used, see Chapter 6, “Managing Multiple Instances of Apache,” on page 59.

Installing the Daemon on Linux

1 From the Linux prompt, run the install file. For example, enter

```
sh install.bin
```

2 Click Next on the Introduction page.

3 In the Choose Install Folder dialog box, type the location where the daemon files should be installed. For example, /usr/local/apache2.

The install program creates a directory called apacheadmin at the location you specify.

4 Type the username the daemon should use to authenticate to the directory.

The user you specify must be a member of the apchadmn-Administrators group on the LDAP server. The username is stored in the startup.properties file.

5 Type the password for the user you specified and click Next.

Although the password is used to authenticate the user, it is not stored in a file. However, each time the daemon is started, you are prompted to enter the password.

6 In the LDAP Server Port field, specify the LDAP server's port number.

7 If the port number you specified is secure (uses SSL), select Yes, and then click Next.

8 In the Path to Apachectl Script field, type the path to the Apache executable file.

The daemon must be able to start, stop, and restart the Apache server. Typically, apache is found in the Apache2/bin directory where Apache was installed.

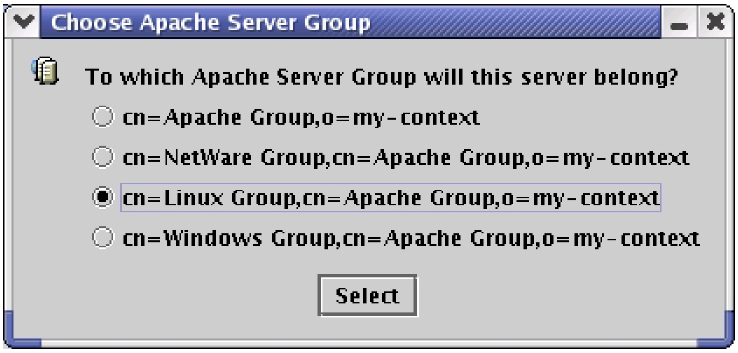
9 In the Path to Apache Configuration File field, type the path to Apache's primary configuration file and click Next.

The Apache configuration file is typically found in the directory where Apache was installed.

10 In the Choose Apache Server Group dialog box, select an Apache server group where the Apache server should be placed and click Next.

IMPORTANT: If you receive a message in place of this dialog box indicating that no group exists in the directory for the platform you are running on, click Yes to have a new group created.

If neither of these dialog boxes appear, the install was unable to connect to the LDAP directory. You can exit the install and troubleshoot or you can manually enter this information in the next step and troubleshoot the LDAP connection when you run the daemon. For more information, see "Troubleshooting the LDAP Connection" on page 77.



To which Apache Server Group will this server belong?

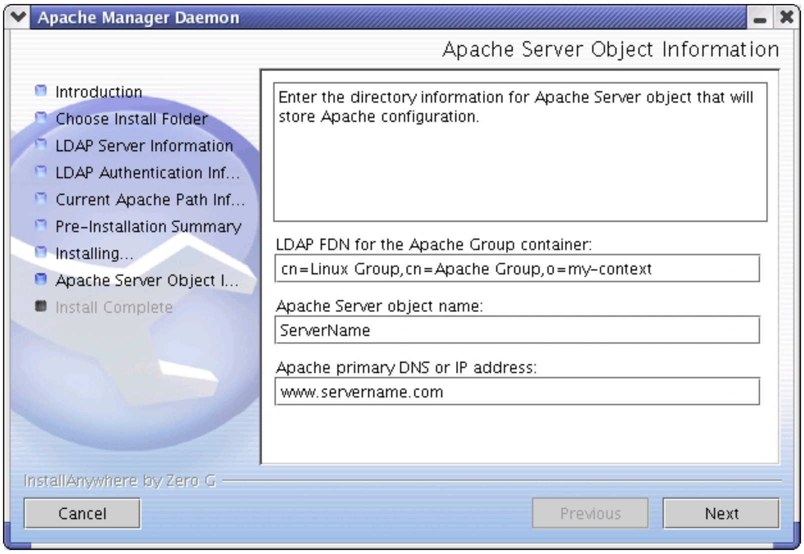
○ cn=Apache Group,o=my-context

○ cn=NetWare Group,cn=Apache Group,o=my-context

● cn=Linux Group,cn=Apache Group,o=my-context

○ cn=Windows Group,cn=Apache Group,o=my-context

11 If you were able to specify a server group in the previous step, the first field in the Apache Server Object Information dialog box should already contain the information it needs. Otherwise, you must specify the group server information manually.



Apache Server Object Information

Enter the directory information for Apache Server object that will store Apache configuration.

LDAP FDN for the Apache Group container:
cn=Linux Group,cn=Apache Group,o=my-context

Apache Server object name:
ServerName

Apache primary DNS or IP address:
www.servername.com

12. In the Apache Server Object Name field, type the name of your Apache Web server.

13. In the Apache Primary DNS or IP Address field, type your Web server's DNS name or IP address. For example, www.mycompany.com.

14. In the Pre-Installation Summary dialog box, review the information to ensure that it is correct and then click Install.

15. When installation is complete, click Done.

To run the daemon, run Ap2webman.bat located in the apacheadmin directory.

For more information about the Apache Manager daemon and how it is used, see Chapter 6, "Managing Multiple Instances of Apache," on page 59.

Troubleshooting the LDAP Connection

If the Apache Manager Daemon Install or the Apache Manager daemon is having trouble connecting to the LDAP server, check each of the following possible causes:

- Verify that the JVM is installed correctly. You must use Java 2, version 1.3.1 or later, and the path to the java.exe needs to be correctly identified.
- Verify that the LDAP server is running.
- If the daemon is producing a certificate exception, you might need to run the Key Import Wizard. To do so, open the apacheadmin\lib\KeyImport.bat batch file in a text editor and verify that the last line in the file includes the correct name of your LDAP server and the secure port used by it. Then run the batch file.

  The Key Import Wizard imports the LDAP server's certificate into Java's default key store. The Apache Manager daemon can then make a secure connection to the LDAP directory.
- Open the apacheadmin\conf\startup.properties file and verify that the LDAP server and LDAP port information is correct. Also, verify that the administrator username and all other settings are correct.