Secure Web Server for Tru64 UNIX Administration Guide

Printed in the US
Published: December 2006

Table of Contents

List of Figures

List of Tables

About This Document

This document describes the administration tasks for configuring and managing the Secure Web Server for Tru64 UNIX™ (powered by Apache).

1 Intended Audience

This document is intended for the system administrator who manages Internet services on a Tru64 UNIX AlphaServer system.

2 Document Organization

This document is organized as follows:

Chapter 1	Provides an overview of the Secure Web Server.
Chapter 2	Describes the general administration tasks for managing a Secure Web Server.
Chapter 3	Explains how to implement dynamic modules.
Chapter 4	Explains how to implement the Tomcat Java Servlet and JSP Engine.
Chapter 5	Describes how to enable the Secure Socket Layer (SSL) to provide secure Internet connections to your server.
Appendix A	Lists the Secure Web Server 1.3 components and modules.
Appendix B	Lists the Secure Web Server 2.0 components and modules.

This manual also contains a glossary and an index.

3 Typographic Conventions

This document uses the following typographical conventions:

`%`, `$`, or `#`
A percent sign represents the C shell system prompt. A dollar sign represents the system prompt for the Bourne, Korn, and POSIX shells. A number sign represents the superuser prompt.

audit(5)
A manpage. The manpage name is *audit*, and it is located in Section 5.

Ctrl+x
A key sequence. A sequence such as **Ctrl+x** indicates that you must hold down the key labeled **Ctrl** while you press another key or mouse button.

Key
The name of a keyboard key. **Return** and **Enter** both refer to the same key.

`User input`
Commands and other text that you type.

`Variable`
The name of a placeholder in a command, function, or other syntax display that you replace with an actual value.

4 Related Information

Consult the following documentation for information on installing, configuring, and administering Internet solutions on Tru64 UNIX operating systems.

Internet Express Documentation

For information on installing, configuring, and administering Open Source and other Web server-related software, see the Documentation Bookshelf provided with HP Internet Express. (HP includes the Internet Express CD-ROMs with Tru64 UNIX AlphaServer systems.)

- Reading documentation using the Administration utility:

  After installation of the Secure Web Server subset (IAEAPCH), the Internet Express Documentation subset (IAEDOC), and the Internet Express Administration utility (IAEADM subset), you can access the Administration utility for Internet Express and read the documentation following the link from the Web page at:

`http://hostname.domain:8081`.

- Reading documentation using the public Web server:

  You can also read the documentation without the Administration utility using the public Web server (if you chose to configure one) to access the documentation index page at `http://hostname.domain/documents/bookshelf.html`. If this URL does not work, verify that the Web server configuration file, `/usr/internet/httpd/admin/conf/httpd.conf`, contains the following line:

  ```
  Alias /documents/ "/usr/internet/docs/IASS/"
  ```

  You can read the installed documentation directly from the file system using a Web browser running on the same system by using the file URL: `file:/usr/internet/docs/IASS/bookshelf.html`.

- Reading documentation from the Internet Express Installation and Documentation CD–ROM:

  You can access the Documentation Bookshelf on the Internet Express CD–ROM. The documentation is available in the following formats:

  — HTML

  — PDF

Tru64 UNIX Operating System Documentation

The Documentation Overview describes the documentation that comes with the HP Tru64 UNIX operating system. The Tru64 UNIX documentation main page can be accessed from the following URL:

`http://h30097.www3.hp.com/docs/`

Tru64 UNIX Best Practices Documentation

Tru64 UNIX Best Practices describe specific tasks for Internet as well as other topics. You can find these documents at the following URL:

`http://h30097.www3.hp.com/docs/best_practices/`

5 HP Encourages Your Comments

HP encourages your comments concerning this document. We are committed to providing documentation that meets your needs. Send any errors found, suggestions for improvement, or compliments to:

feedback@fc.hp.com

Please include the following information along with your comments:

- The full title of the document
- The section numbers and page numbers of the information on which you are commenting
- The versions of Tru64 UNIX and Internet Express that you are using
- If known, the type of processor that is running Tru64 UNIX

1 Overview

The Secure Web Server (powered by Apache) is an implementation of the Apache Software Foundation's (ASF) Apache HTTP server for Tru64 UNIX. It contains a packaged, integrated and tested version of many of the popular components of the Apache Web server (mod_ssl, PHP, fastcgi, and others) and the modules that are used with it. In addition, the Apache Software Foundation's Tomcat Java Servlet and JavaServer Pages (JSP) are provided to handle Java based dynamic content.

Note:

For Internet Express Version 6.0 and later, two versions of the Secure Web Server are offered: Secure Web Server 1.3 (based on Apache Version 1.3) and the Secure Web Server 2.0 (based on Apache Version 2.0). At installation, you can choose either version for the public Web server. (The Administration Web Server continues to use the Secure Web Server 1.3.) See Section 1.1.1: Choosing a Web Server for more information.

The Secure Web Server integrates other features beyond the core modules supplied by ASF, including:

- Support for Dynamic Shared Objects (DSO).
- Support for SSL connections (HTTPS) using a DSO module.
- Support for APXS, which allows third party modules to be built against and used with an installed Secure Web Server.
- Support for SUexec, once it is enabled in the server (an additional configuration is required to enable it).
- Support for the Atalla hardware accelerator cards.
- Support for the Tomcat Java Servlet and Java Server Pages (JSP) container.

In addition, all modules provided with the Apache code base are built-in or provided as a dynamic shared object (DSO).

The Secure Web Server provides a Web-based administration interface that allows an administrator to perform common management tasks on the Web server. You access these administration pages from the Web Server Administration utility (see Section 1.1: Accessing the Secure Web Servers).

The Secure Web Server is available on the Associated Products CD-ROM, included with the Tru64 UNIX operating system distribution, and is also available with Internet Express for Tru64 UNIX. HP includes the Internet Express CD-ROM with Tru64 UNIX AlphaServer systems. If you need the Internet Express CD-ROM, you can contact your HP representative. The part number for the Internet Express product is QB-3NCAA-SA. The Secure Web Server is also available for download from the HP Web site:

http://www.tru64unix.compaq.com/internet/

1.1 Accessing the Secure Web Servers

The Secure Web Server provides the following servers for managing Internet services:

- Public — The public Web server is an instance of the Secure Web Server, listening on Port 80, that can be configured as a general purpose Web server and used by anyone. The installation procedure lets you select this port for the Apache Version 2.0 or Apache Version 1.3. If you choose to install both Apache versions for public Web servers, you can use Port 80 for one server instance only and must select a different port for the other.
- Administration — The Administration Web Server is an instance of the Secure Web Server that listens on Port 8081. It provides an administration interface for the Secure Web Servers accessible from a Web browser.

The URL takes the form:

```
http://host.domain.name:port
```

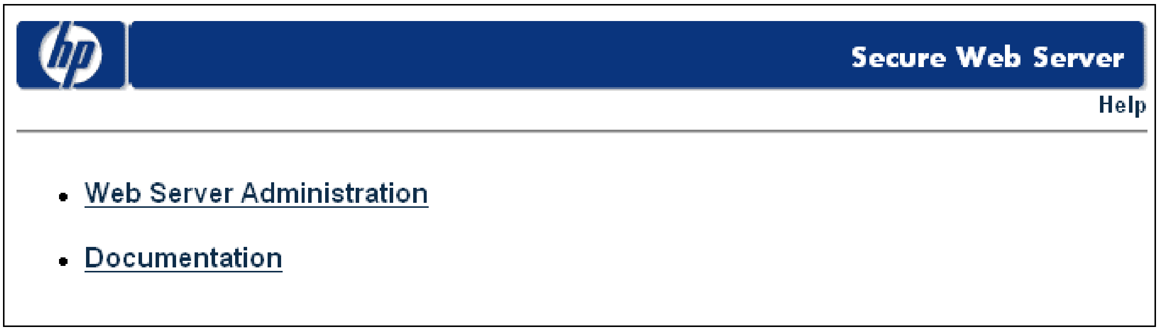
where *host.domain.name* represents the fully qualified host name of the local system (the system on which Internet Express is installed) and *port* represents the port Web server is listening on (either Port 80 or 8081).

The Administration Web Server is initially accessible from the local system only. To allow access from a remote system when running the Secure Web Server, see Section 2.9: Allowing Remote Access to the Administration Web Server.

To access the Secure Web Servers, follow these steps:

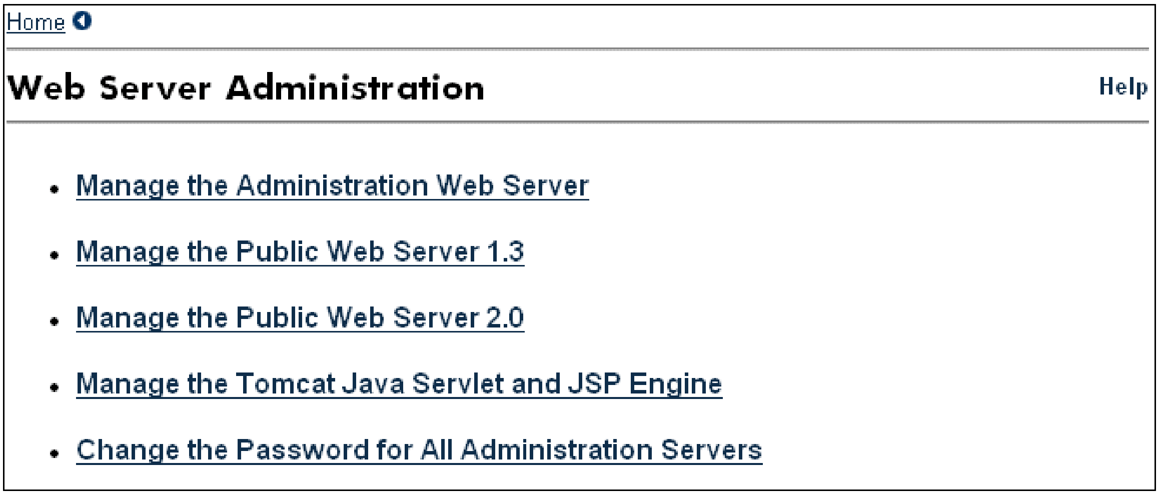
1. From an HTML-based Web browser (such as Netscape Navigator Version 4.5 or later, Microsoft Internet Explorer Version 4.0 or later, or Mozilla), enter the URL, including the proper port. When you access the Administration server, you are prompted for a user name and password.
2. Enter a user name and password. The default user name for Web server administration is `admin`. During installation, the system administrator sets a password to be used for the Web server administration accounts. To change the password for the Administration Web Server, see Section 1.1.3: Managing the Administrator Password. Figure 1-1 shows the Secure Web Server main menu when you first log in.

Figure 1-1 Secure Web Server Main Menu



When you choose Web Server Administration from the main menu, the Web Server Administration menu is displayed, listing the available servers and providing a link for changing server passwords. Figure 1-2 shows the Web Server Administration menu. In addition to the links for the Administration Web Server and Public Web Server, when you install Apache Version 2.0, an additional link, Manage the Public Web Server 2.0, is displayed.

Figure 1-2 Secure Web Server Administration Menu


Home
Web Server Administration
Help
Manage the Administration Web Server
Manage the Public Web Server 1.3
Manage the Public Web Server 2.0
Manage the Tomcat Java Servlet and JSP Engine
Change the Password for All Administration Servers

When you access the Web server, you are given access to privileged files and can perform system management tasks until exiting the browser. Do not leave an administration session unattended. Limit access to the `admin` account to those individuals authorized to perform Internet system management tasks.

1.1.1 Choosing a Web Server

For Internet Express Version 6.0 and later, you have the choice of installing any of three Web servers: Apache Version 2.0, Apache Version 1.3, and a standalone Tomcat server.

Each Web server has strengths and weaknesses that you should evaluate prior to choosing which server to install. It is possible to install all of the server options and configure them to respond to different ports. If the port you choose is other than the default HTTP port (port 80), then you must include the port number in the URL of the request.

The following sections evaluate each of the Web server options.

1.1.1.1 Secure Web Server 1.3

The Secure Web Server 1.3 is the traditional Web server powered by the Apache Version 1.3 code base. This server has a long history of reliability and the largest selection of Apache modules (see Appendix A: Secure Web Server 1.3 Components and Modules).

Use this Web server under the following conditions:

- You are running applications where the existing Web content depends on Apache modules that have not been ported to the Apache Version 2.0 API.
- You are running an older version of Apache Version 1.3 and you need to minimize transitional risks (i.e., upgrading to a newer version of Apache).

The Secure Web Server 1.3 offers the following strengths:

- The Secure Web Server 1.3 offers a proven code base, and provides robust and fault tolerant service.
- Most Apache modules were written for the Apache Version 1.3 code base, therefore most existing modules will work properly. As the Secure Web Server 1.3 is not multithreaded, it does not have problems with modules that use libraries that are not thread safe.
- The PHP module provided in Apache Version 1.3 contains the most features.
- The Secure Web Server 1.3 is easily configured to support SSL connections.
- The Secure Web Server 1.3 is supported as a TruCluster multi-instance application.

The Secure Web Server 1.3 has the following weaknesses:

- It does not support IPv6.
- The Apache Version 1.3 code base is no longer in active development by the Apache Software Foundation.
- JSP and Java Servlet requests must be forwarded to Tomcat.

1.1.1.2 Secure Web Server 2.0

The Secure Web Server 2.0 is powered by the new Apache Version 2.0 code base. Apache Version 2.0 is a major revision of the Apache code base that provides for running applications in a multithreaded environment.

Use the Secure Web Server 2.0 for new installations where there are no dependencies on Apache modules that are not yet available for the Apache Version 2.0 code base. (See Appendix B: Secure Web Server 2.0 Components and Modules for a list of the Apache Version 2.0 modules.)

Migrating existing Web servers based on Apache Version 1.3 to Apache Version 2.0 is normally a straight forward process for most installations. The main configuration file, `httpd.conf`, requires very little modification to work with the new Apache architecture. The major issue with migration is the availability of Apache modules where a subset of modules do not yet support the Apache Version 2.0 API.

The Secure Web Server 2.0 offers the following strengths:

- The new code design uses threads and is intended to provide for more efficient delivery of content.
- The Secure Web Server 2.0 supports IPv6.
- The Apache Software Foundation is actively developing this code base.
- The Secure Web Server 2.0 is easily configured to support SSL connections.
- The Secure Web Server 2.0 is supported as a TruCluster multi-instance application.

The Secure Web Server 2.0 has the following weaknesses:

- Existing Apache modules based on the Apache Version 1.3 API require porting to the new module API. Currently, only a small number of modules have been ported. In addition, because Apache Version 2.0 uses multithreaded programming, all Apache modules (and the libraries they depend on) must also be thread safe.
- The PHP module does not include some features that are present in servers based on Apache Version 1.3 due to thread-safety issues of the libraries used by the optional features.

- JSP and Java Servlet requests must be forwarded to Tomcat.
- The Secure Web Server 2.0 currently does not support FrontPage, and Auth_ldap extensions.

1.1.1.3 Tomcat Java Servlet and JavaServer Pages Container

The Tomcat Java Servlet and JavaServer Pages container (i.e., Tomcat server) was originally used as a service for handling requests forwarded by another Web server. The current version of Tomcat is a full-featured Web server. Choose the Tomcat server when your applications primarily supply Java-based dynamic content in the form of Java Servlet and JavaServer Pages (JSP) with the supporting static Web content.

The Tomcat server has the following strengths:

- It provides the fastest handling of Java Servlet and JSP requests.
- When used with the Fast Java Virtual Machine and adequate resources, the Tomcat server provides comparable performance to the Apache Web servers.
- Using the Tomcat server standalone provides significant improvements to response time, compared to requests forwarded through an Apache Web server to Tomcat.

The Tomcat server has several weaknesses:

- Although traditional CGI and Server Side Includes (SSI) are supported (requires Java 1.3 or later), they are disabled by default. Traditional CGI and Server Side Includes (SSI) are not considered strengths of Tomcat and are recommended primarily for transitional use during the development of a Web application. CGI and SSI can bypass any Java-based security settings that are configured into the Java Security Manager.
- The Tomcat server does not support common Apache extensions such as PHP.
- Configuring Tomcat to support SSL connections is a complicated process, requiring installation and configuration of Java components that are not included with this kit. Information on configuring Tomcat to support SSL connections can be found at the Tomcat Web site:

  http://jakarta.apache.org/tomcat
- The Tomcat server runs as a single instance in a TruCluster.
- By default, the Tomcat server does not support running as a non-privileged user when listening on privileged ports (less than 1024). This kit provides a program that allows Tomcat to avoid this restriction, starting Tomcat as the root user, and then switching to an unprivileged user after opening ports.

1.1.2 Managing the Apache Web Servers

All installed Web servers are configured to restart when the system reboots. It is also possible to start and restart the servers from the command line.

You can configure the Web servers to make them disabled. A disabled Web server will not be started on system reboot and will not start when the startup script is invoked. This feature allows a system administrator to better manage the migration of the Secure Web Server from Version 1.3 to Version 2.0 by configuring them to use the same ports while allowing only one version to run.

Table 1-1 lists the commands for managing the Secure Web Server 1.3.

Table 1-1 Command-Line Commands for Managing the Secure Web Server 1.3

Action	Command
Start the server	`/sbin/init.d/httpd_public start`
Stop the server	`/sbin/init.d/httpd_public stop`
Restart the server	`/sbin/init.d/httpd_public restart`
Start the server on all Cluster Members	`/sbin/init.d/httpd_public cluster-start`
Stop the server on all Cluster Members	`/sbin/init.d/httpd_public cluster-stop`
Restart the server on all Cluster Members	`/sbin/init.d/httpd_public cluster-restart`

Table 1-1 Command-Line Commands for Managing the Secure Web Server 1.3 *(continued)*

Action	Command
Enable the server	`/sbin/rcmgr -c set APACHE1 ENABLE`
Disable the server	`sbin/rcmgr -c set APACHE1 DISABLE`
Check server status	`/sbin/rcmgr -c get APACHE1`

Table 1-2 lists the commands for managing the Secure Web Server 2.0.

Table 1-2 Command-Line Commands for Managing the Secure Web Server 2.0

Action	Command
Start the server	`/sbin/init.d/hpapache2 start`
Stop the server	`/sbin/init.d/hpapache2 stop`
Restart the server	`/sbin/init.d/hpapache2 restart`
Start the server on all Cluster Members	`/sbin/init.d/hpapache2 cluster-start`
Stop the server on all Cluster Members	`/sbin/init.d/hpapache2 cluster-stop`
Restart the server on all Cluster Members	`/sbin/init.d/hpapache2 cluster-restart`
Enable the server	`/sbin/rcmgr -c set APACHE2 ENABLE`
Disable the server	`/sbin/rcmgr -c set APACHE2 DISABLE`
Check server status	`/sbin/rcmgr -c get APACHE2`

Table 1-3 lists the commands for managing the Tomcat Web server.

Table 1-3 Command-Line Commands for Managing the Tomcat Web Server

Action	Command
Start the server in a TruCluster environment	`caa_start tomcat`
Start the server	`/sbin/init.d/tomcat start`
Stop the server in a TruCluster environment	`caa_stop tomcat`
Stop the server	`/sbin/init.d/tomcat stop`
Enable the server	`/sbin/rcmgr -c set TOMCAT ENABLE`
Disable the server	`/sbin/rcmgr -c set TOMCAT DISABLE`
Check server status	`/sbin/rcmgr -c get TOMCAT`

1.1.2.1 Migration Utilities for the Secure Web Server

This section describes the following utilities for migrating the iPlanet Web Server to the HP Secure Web Server for Tru64 UNIX:

- `certmig` — Migrates iPlanet 4.x certificates to the Secure Web Server. This utility, an extension of the PK12UTIL utility provided by the Mozilla community, uses Network Security Services (NSS) libraries to convert iPlanet certificates, key translations, and certificate chains to those of the Apache Web Server.
- `test_certmig.sh` — Provides a wrapper around `certmig` for importing and extracting the list certificates in an iPlanet 4.1.x Certificate database.
- `mkcert.sh` — Generates private keys, certificate signing requests, and certificates.
- `cache_util.pl` — Helps optimize file caching by reviewing the most commonly accessed files in `logs/access_log` and by creating a caching file list. This file caching utility is bundled with HP Apache 2.0.

1.1.2.2 Tuning Tru64 UNIX for the Secure Web Server

Proper system tuning can have a significant impact on the performance of the Secure Web Servers. The Secure Web Server software includes the `kerneltuner` shell script that sets the primary tuning recommendations for Internet server performance. These recommendations are described in the *Tuning Tru64 UNIX for Internet Services* manual, available from the following URL:

http://h30097.www3.hp.com/docs/internet/TITLE.HTM

Additional system performance tuning settings are also described in this document.

Note:

Because Internet server configurations differ, a recommended setting may not provide optimal performance for all configurations. Carefully plan your Tru64 UNIX system performance tuning settings.

The `kerneltuner` utility is installed in the `bin` subdirectory under the Web server `root` directory for each Secure Web Server Public Web server. Run this utility as root. For example:

```
# su root
# cd /usr/opt/hpapache2/bin
# ./kerneltuner
```

The above example runs the `kerneltuner` utility in interactive mode. In this mode, the root user can choose which system tuning recommendations will be configured. Optionally, a `kerneltuner` script can configure all the system tuning recommendations without interaction from the user by running it with the `kerneltuner` utility with the `-s` command line option:

```
# /usr/opt/hpapache2/bin/kerneltuner -s
```

You do not have to run the *kerneltuner* utility for each Secure Web Server you installed. Instead, run the utility once on each system or TruCluster member where the Secure Web Server software is installed.

For the primary tuning settings to take effect, reboot your system. You can reboot the system after running the `kerneltuner` utility, or you can wait for a more convenient time.

1.1.3 Managing the Administrator Password

During installation of the Secure Web Server, a single Web administration user is created for accessing all Administration Web Server instances. The username is `admin`. The administrator password is set to the password that you entered during installation.

If you know the administrator password, you can change it using the Web Server Administration utility (Section 2.8: Changing the Password for the Administration Web Server).

If you received your Secure Web Server software preinstalled from HP or if you have forgotten your administrator password, the `/usr/internet/httpd/bin/dbmmange` command lets you create, view, add, and update the contents of the Administrator user database.

To run the `dbmanage` command and change the administrator password, follow these steps:

1. Login as the root user, then run the following command:

   ```
   # /usr/internet/httpd/bin/dbmmanage /usr/internet/httpd/userdb/admin adduser admin
   ```

2. The `dbmanage` command initializes the database (if needed) and prompts you for a password for Web user administrator, `admin`. If the user administrator already exists, the following message is displayed:

   ```
   Sorry user 'admin' already exists !
   ```

   If you receive this message, rerun the `dbmanage` command as follows (note that `adduser` is now `update`):

   ```
   # /usr/internet/httpd/bin/dbmmanage /usr/internet/httpd/userdb/admin update admin
   ```

The command prompts you for a password and updates the database.

3. After updating the database, if the Administration Web Server is already running, you should not have to restart it. If you need to restart the Administration Web Server, run the following command:

```
# sbin/init.d/httpd_admin restart
```

1.2 Web Server Administration Functions

All Secure Web Server administration functions are performed using Port 8081. All activity is recorded in the associated log files (Section 2.5: Viewing Web Server Reports and Log Files).

Management tasks available from the Secure Web Server administration menus include:

- Changing configuration parameters, including tuning parameters, access control entries, listening ports and addresses, virtual hosts, URL defaults, HTML directory aliases, CGI directory aliases, and logging and reporting parameters.
- Managing user accounts, displaying status, and viewing information for the public Web server
- Changing passwords for the Administration Web Server
- Allowing remote access to the Administration Web Server
- Viewing server activity reports, access log files, and error log files, and refreshing these files
- Starting and stopping the Secure Web Servers

These tasks are described in detail in Chapter 2.

1.3 Dynamic Modules

Dynamic modules, also called Dynamic Shared Objects (DSO) or shared libraries, are loaded into the server process space only when necessary to assure that overall memory usage is reduced. You can use DSO modules to customize the Secure Web Server. Chapter 3 describes how to activate the Apache DSO modules. Appendix A lists the standard Apache Version 1.3 modules and Appendix B lists the standard Apache Version 2.0 modules provided with this release.

1.4 Tomcat Java Servlet and JavaServer Pages Container

Tomcat, provided with the Secure Web Server, is a Java Servlet and JavaServer Pages (JSP) container developed by the Apache Software Foundation's Jakarta project. The Tomcat engine is most commonly used with commercial grade Web servers such as Apache and can also be used as a standalone Web server.

Tomcat is provided as an optional Secure Web Server subset that, when installed, allows the public instance of the Secure Web Server to be configured to seamlessly pass requests for Java Servlet and JSP pages to the Tomcat container.

Chapter 4 describes how to start Tomcat, use the Tomcat examples, and locate Tomcat directories and files.

1.5 SSL and the Secure Web Server

The Secure Web Servers have built-in support for the Secure Socket Layer (SSL) on port 443. The HTTPS protocol is the most widely-used method for performing secure transactions on the Web. The protocol is supported by most Web servers and clients.

SSL provides privacy, guaranteed through encryption. Although information can be intercepted by a third party, the perpetrator cannot read the information without a private encryption key. If the information received will not decrypt properly, the recipient can determine whether the information has been tampered with during transmission.

SSL also provides authentication through digital certificates that are generated for SSL, although the source of digital certificates might not always be credible for online payment transactions. Secure Web Server SSL encryption uses a secret key nested within public key encryption and authenticated through digital certificates.

Chapter 5 describes the administration functions that you perform to enable SSL on your server:

- Generating a private key.
- Generating a request for a digital certificate.
- Generating and installing a test certificate, installing an actual certificate, and then viewing its details.
- Enabling (or disabling) SSL capabilities.
- Testing your SSL connection.
- Limiting access to HTTPS connections.
- Special considerations for enabling SSL on public Web servers.

2 Managing the Secure Web Server

This chapter describes the administration tasks available from the Web Server Administration utility that allow you to manage the Secure Web servers.

Access the Web Server Administration utility by logging in to the Administration Web Server that listens on port 8081. The Secure Web Servers currently installed and managed by the administration utility are listed on the top-level Web Server Administration form. When you install additional versions of the Secure Web Server, they are added to the Web Server Administration form. Choosing a Web server on the Web Server Administration form displays the administration tasks for that server. The administration tasks for each Web server differ based upon the Web server version and the Web server type: administration or public.

Administration tasks that apply only to the management of the Administration Web Server:

- Change the password for the Administration Web Server (Section 2.8: Changing the Password for the Administration Web Server)
- Allow remote access to the Administration Web Server (Section 2.9: Allowing Remote Access to the Administration Web Server)

Administration tasks that apply only to management of the Public Web server:

- Display public Web server status (Section 2.3: Displaying Public Web Server Status)
- Display public Web server information (Section 2.4: Displaying Public Web Server Information)
- Dynamically add or remove server modules (Section 2.10: Dynamically Adding and Removing Server Modules)

Administration tasks that apply to both Administration and public Web servers:

- Change configuration parameters for the Secure Web Servers (Section 2.1: Changing Configuration Parameters)
- Change public Web server user accounts (Section 2.2: Changing Public Web Server User Accounts)
- View server activity reports or the access and error log files for the Web servers (Section 2.5: Viewing Web Server Reports and Log Files)
- Refresh the access log and error log files for the Web servers (Section 2.6: Refreshing the Administration Web Server Log Files)
- Start and stop the Web servers (Section 2.7: Starting and Stopping the Secure Web Server)
- Enable and manage the Secure Socket Layer (SSL) with the Secure Web Server (Chapter 5).

2.1 Changing Configuration Parameters

A configuration parameter is specified by a directive and is stored in one of the configuration files listed in Table 2-1.

Table 2-1 Configuration Files for Secure Web Servers

Server	Configuration File
Administration Web Server	`/usr/internet/httpd/admin/conf/httpd.conf`
Public Web Server 2.0	`/usr/opt/hpapache2/conf/httpd.conf`
Public Web Server 1.3	`/usr/internet/httpd/conf/httpd.conf`

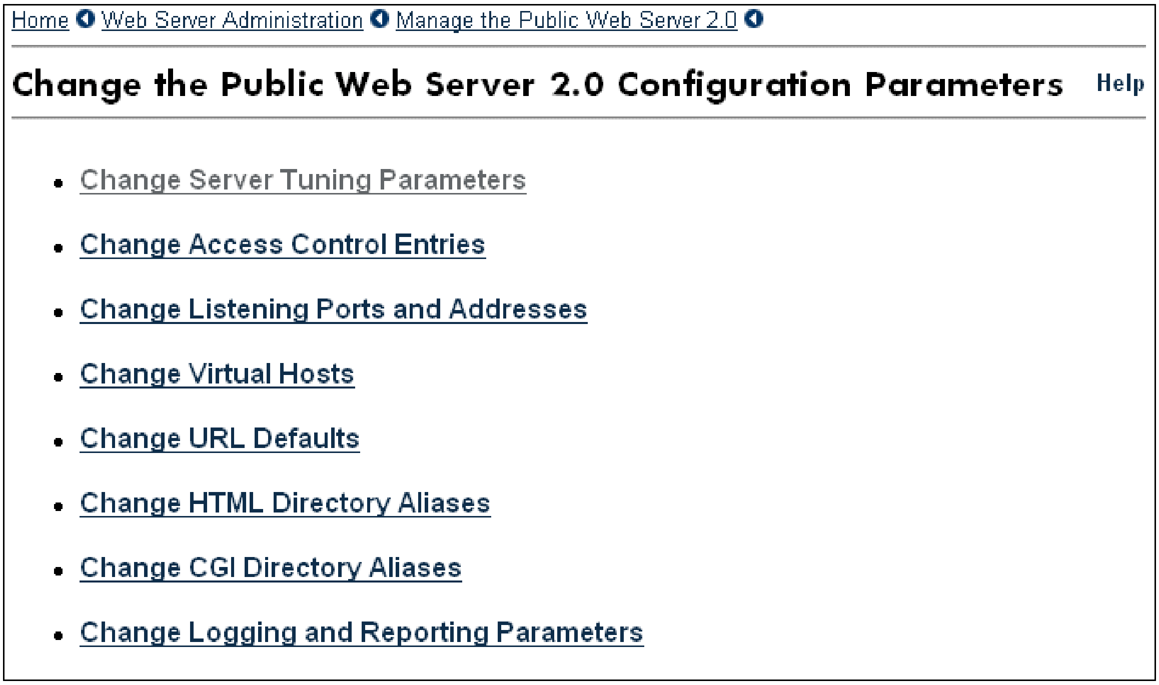
You can specify the following types of configuration parameters:

- Server tuning parameters (Section 2.1.1: Changing Server Tuning Parameters)
- Access control entries (Section 2.1.2: Changing Access Control Entry Parameters)
- Listening ports and addresses (Section 2.1.3: Changing Listening Port and IP Address Parameters)
- Virtual hosts (Section 2.1.4: Changing Virtual Host Parameters for the Public Web Servers)
- URL defaults (Section 2.1.5: Changing URL Default Parameters for the Public Web Servers)
- HTML directory aliases (Section 2.1.6: Changing HTML Directory Alias Parameters for the Public Web Server)

- CGI directory aliases (Section 2.1.7: Changing CGI Directory Alias Parameters for the Public Web Server)
- Logging and reporting parameters (Section 2.1.8: Changing Logging and Reporting Parameters)

Figure 2-1 shows the menu for changing the configuration parameters for the Public Web Server 2.0.

Figure 2-1 Change Configuration Parameters Menu for Public Web Server 2.0



Home ◀ Web Server Administration ◀ Manage the Public Web Server 2.0 ◀

Change the Public Web Server 2.0 Configuration Parameters Help

- Change Server Tuning Parameters
- Change Access Control Entries
- Change Listening Ports and Addresses
- Change Virtual Hosts
- Change URL Defaults
- Change HTML Directory Aliases
- Change CGI Directory Aliases
- Change Logging and Reporting Parameters

The Secure Web Server configuration files are read in the following order:

- httpd.conf
- ssl.conf (Secure Web Server 2.0 only)
- srm.conf
- access.conf

Note:

By default, the access.conf and srm.conf configuration files do not contain any directives. While they remain supported in the Secure Web Server 1.3, all directives are defined in httpd.conf. Directives either exist in the httpd.conf file itself or are included in it by use of the Include directive.

If you specify the same directive in more than one configuration file, the first directive found takes precedence.

In the tables in the following sections, a directive enclosed in angle brackets can be defined using multiple lines and must be delimited by a <*directive*>...</*directive*> pair, where *directive* is the directive name. The following example shows the proper syntax for a multiple-line directive:

```
<Limit GET POST>
order deny,allow
deny from all
allow from host1.domain.name domain2.name
</Limit>
```

Through the Change Configuration Parameters menu for each server, the Web Server Administration utility allows you to set many of the frequently used configuration parameters described in this section. If you want to take advantage of more specialized functionality, you must manually edit the Secure Web Server configuration files listed in Table 2-1. Avoid modifying the configuration parameters that are handled by the Administration utility when manually editing these files.

For a complete listing of configuration file directives, see the Apache Web site for the appropriate directives for the Web server version:

For the Secure Web Server 1.3:

http://www.apache.org/docs/mod/directives.html

For Secure Web Server 2.0:

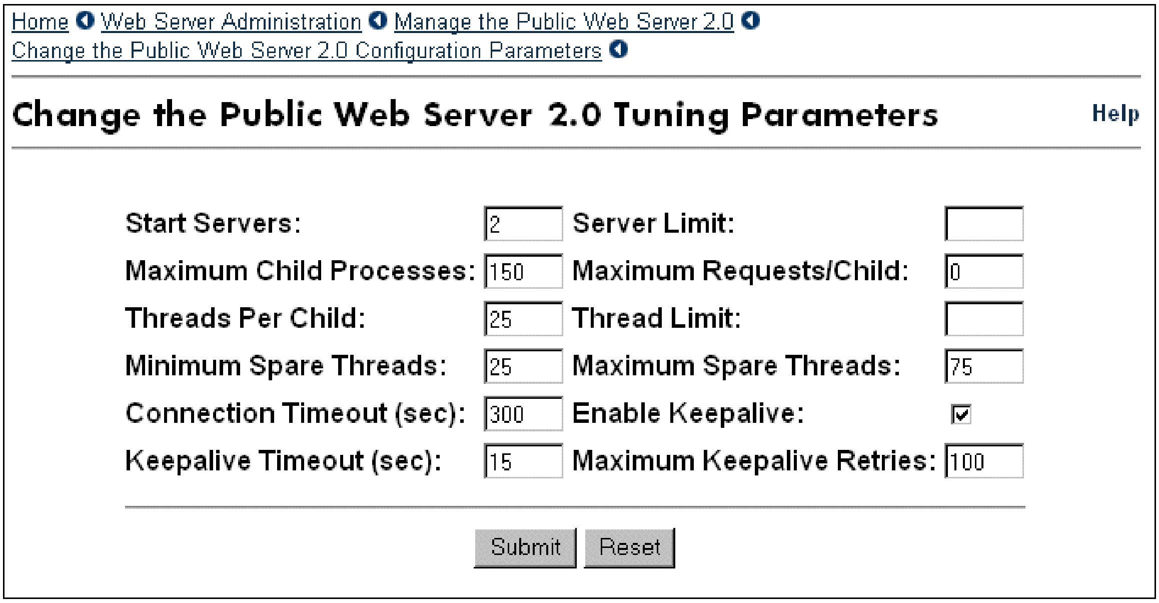
http://httpd.apache.org/docs-2.0/mod/directives.html

2.1.1 Changing Server Tuning Parameters

To change the server tuning parameters:

1. From the Web Server Administration menu, choose the link to the Web server that you want to change. You can change the configuration parameters on either the Public or Administration Web servers. For example, choose Manage the Public Web Server 2.0 to change the tuning parameters of the Public Web server 2.0.
2. From the Manage the Public Web Server 2.0 menu, choose Change Configuration Parameters.
3. From the Change Public Web Server 2.0 Configuration Parameters menu, choose Change Server Tuning Parameters. Figure 2-2 shows the Change Tuning Parameters form for the Public Web server 2.0. A Change Tuning Parameters form is also available for the Public Web Server 1.3, but it has different tuning parameters than the Public Web Server 2.0. A Change Tuning Parameters form is also available for the Administration Web Server. (See Figure 2-13.)

Figure 2-2 Change Tuning Parameters Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 2.0 ◀
Change the Public Web Server 2.0 Configuration Parameters ◀

Change the Public Web Server 2.0 Tuning Parameters Help

Start Servers:	2	Server Limit:	
Maximum Child Processes:	150	Maximum Requests/Child:	0
Threads Per Child:	25	Thread Limit:	
Minimum Spare Threads:	25	Maximum Spare Threads:	75
Connection Timeout (sec):	300	Enable Keepalive:	☑
Keepalive Timeout (sec):	15	Maximum Keepalive Retries:	100

4. On the Change Server Tuning Parameters form (Figure 2-2), change one or more of the parameters.

   Table 2-2 shows which Web Server directive is associated with each parameter field on the Change Tuning Parameters form and the type of value expected. Parameters that apply only to Public Web Server 2.0 are specified for 2.0 only.

Table 2-2 Server Tuning Parameters and Associated Directives

Parameter	Directive	Description
Minimum Spare Servers	MinSpareServers *number*	Minimum number of unused server child processes to maintain
Maximum Spare Servers	MaxSpareServers *number*	Maximum number of unused server child processes left running before additional child processes are killed

Table 2-2 Server Tuning Parameters and Associated Directives *(continued)*

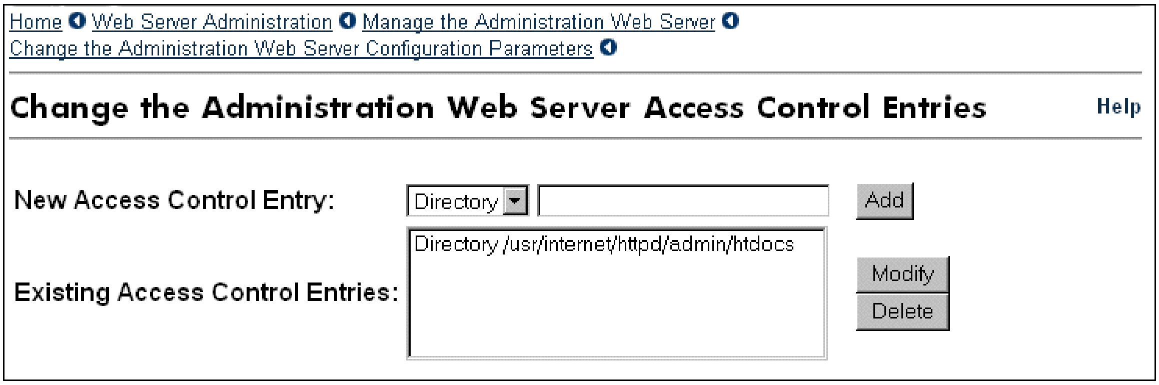
Parameter	Directive	Description
Start Servers	StartServers *number*	Initial number of server child processes
Server Limit (for 2.0 only)	ServerLimit *number*	Upper limit on configurable number of processes
Thread Limit (for 2.0 only)	ThreadLimit *number*	Upper limit on the configurable number of threads per child process
Maximum Connections	MaxClients *number*	Maximum number of server processes for client connections
Maximum Requests /Connection	MaxRequestsPerChild *number*	Number of requests handled before child process is terminated
Threads Per Child (for 2.0 only)	ThreadsPerChild *number*	The number of threads created by each server child process
Maximum Spare Threads (for 2.0 only)	MaxSpareThreads *number*	Maximum number of idle threads
Minimum Spare Threads (for 2.0 only)	MinSpareThreads *number*	Minimum number of idle threads available to handle request spikes
Connection Timeout (secs)	Timeout *number*	Time (seconds) to wait for response before terminating a connection
Enable Keepalive	KeepAlive on | off	Whether or not to hold open a connection after the initial connection is lost
Keepalive Timeout (secs)	KeepAliveTimeout *number*	Time (seconds) to wait for subsequent connection on a KeepAlive connection
Maximum Keepalive Retries	MaxKeepAliveRequests *number*	Number of times to reuse a connection

2.1.2 Changing Access Control Entry Parameters

You can change access control entries for any of the installed Web Servers. The steps in this section describe how to change the access control entry for the Public Web server 2.0. The steps are similar for the other Web servers.

1. From the Change Public Web Server 2.0 Configuration Parameters form, choose Change Access Control Entries. Figure 2-3 shows the Change Access Control Entries form for the Administration Web Server. This form is also available for the Public Web servers.

Figure 2-3 Change Access Control Entries Form



Home ◀ Web Server Administration ◀ Manage the Administration Web Server ◀
Change the Administration Web Server Configuration Parameters ◀

Change the Administration Web Server Access Control Entries

New Access Control Entry: Directory

Existing Access Control Entries: Directory /usr/internet/httpd/admin/htdocs

2. By default, each Web server has one main access control entry controlling access to the document root directory of the server. In general, this entry should be the only entry you might want to change, though many access control entries are listed. The access control entries for each Web server's document root are as follows:

- `/usr/internet/httpd/admin/htdocs` (Administration Web Server)
- `/usr/internet/httpd/htdocs` (Public Web server 1.3).
- `/usr/opt/hpapache2/htdocs` (Public Web server 2.0)

You can also change access control entries for the following locations (for the public Web servers only):

- `/server-status`
- `/server-info`

You can add an access control entry for a directory or location you have created for the Web server.

Table 2-3 shows which Secure Web Server directive is associated with each parameter field on the Change Access Control Entries form and the type of value expected.

Table 2-3 Access Control Parameters and Associated Directives

Parameter	Directive	Description
Type and Specification	<Directory *path* >| <Location *name*> | <Files *filename*>	*path*, *name*, and *filename* can contain wildcards.
Limit Access Methods	<Limit *method*>	Specify one of the following Limit Access Methods: • `GET`—Standard HTML access; parameters can be passed as part of the URL. • `POST`—Form access; parameters are passed separately. • `GET POST` • `All Methods` When you choose `All Methods` (the default), the `Limit` directive is not specified in the `access.conf` file for this Type and Specification (directory, location, or file).
Precedence	order[1] deny,allow | order allow, deny	Specifies the order in which to process the `deny from` and `allow from` directives.
Hosts Allowed Access	allow from[1]all | allow from *host_list*	List of fully or partially qualified host or domain names, separated by spaces. You cannot use wildcards and you must use complete DNS fields (for example, `domain.com` does not match `mydomain.com`).
Hosts Denied Access	deny from[1]all | deny from *host_list*	List of fully or partially qualified host or domain names, separated by spaces. You cannot use wildcards and you must use complete DNS fields (for example, `domain.com` does not match `mydomain.com`).
User Authentication and Selected Users	require *user_list*	To authenticate only specific users, set User Authentication to For Selected Users, and select one or more users from the Selected Users list. (These users are defined in the file specified by the `AuthDBMUserFile` directive. To add a user to this list, use the Change Web Server User Accounts form.) To authenticate all users, set User Authentication to For All Valid Users. If no Web server user accounts exist, Authentication is disabled.
Authentication Prompt Name	AuthName *string*	Portion of the string displayed in the Username/Password dialog box that prompts for user name (“Enter username for *name* at *host:port*:”).
CGI Execution	Options ExecCGI[2]	When the Enable CGI Script Execution check box is selected, allows CGI scripts to be executed from within the specified directory.

[1] The Administration utility expects this directive to be defined within the context of the `Limit` directive.

[2] The Administration utility expects this directive to be defined within the context of the `Directory` directive.

In the following example, the `Limit` directive allows access to the specified domains and hosts only:

```
<Limit GET POST>
order deny, allow
deny from all
allow from host1.domain1.name domain2.name
</Limit>
```

In the following example, access is allowed to everyone except the specified hosts and domains:

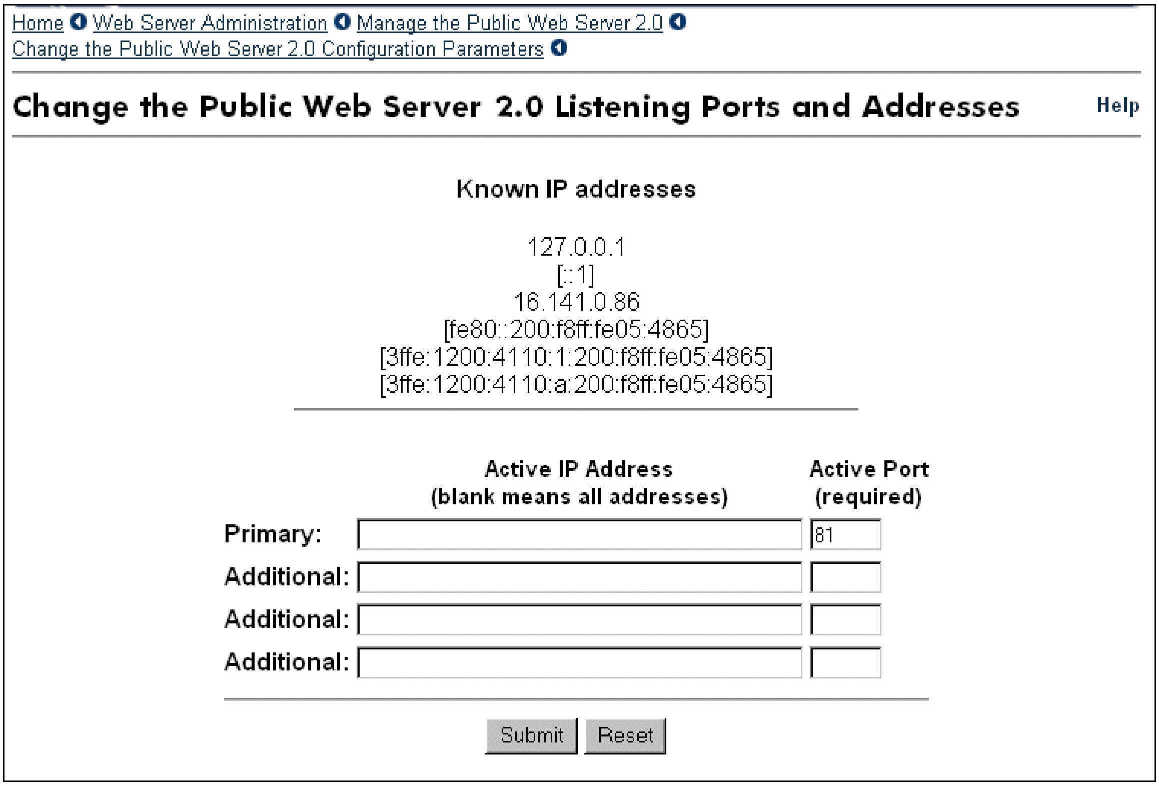
```
<Limit GET POST>
order allow, deny
allow from all
deny from host1.domain1.name domain2.name
</Limit>
```

2.1.3 Changing Listening Port and IP Address Parameters

Normally, a Web server listens for HTTP requests on all known IP addresses on a system. The default (or primary) port, port 80, is used for each address. The Change Listening Ports and Addresses form allows you to limit the IP addresses and ports a Web server listens to by allowing you to enter specific addresses and ports for the server. If your system has been configured to support the IPv6 protocol, IPv6 style addresses can be entered in this form as well. However, IPv6 addresses can be used only with the Secure Web Server 2.0 public Web server. IPv6 style addresses should not be used in the Change Listening Ports and Addresses form for the Secure Web Server 1.3.

From the Change Public Web Server 2.0 Configuration Parameters form (Figure 2-1), choose Change Listening Ports and Addresses. Figure 2-4 shows the Change Listening Ports and Addresses form for Public Web Server 2.0. A similar form is also available from the Change Administration Web Server Configuration Parameters form. (See Figure 2-13.)

Figure 2-4 Change Listening Ports and IP Addresses Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 2.0 ◀
Change the Public Web Server 2.0 Configuration Parameters ◀

Change the Public Web Server 2.0 Listening Ports and Addresses

Known IP addresses

127.0.0.1
[::1]
16.141.0.86
[fe80::200:f8ff:fe05:4865]
[3ffe:1200:4110:1:200:f8ff:fe05:4865]
[3ffe:1200:4110:a:200:f8ff:fe05:4865]

	Active IP Address (blank means all addresses)	Active Port (required)
Primary:		81
Additional:		
Additional:		
Additional:		

Table 2-4 shows which Secure Web Server directive is associated with each parameter field on the Change Listening Ports and Addresses form and the type of value expected.

Table 2-4 Listening Port/IP Address Parameters and Associated Directives

Parameter	Directive	Description
Active IP Address and Active Port (Primary and Additional)	Listen *[IP address:]port*	Specifies one or more ports or IP addresses to listen on
Active IP Address and Active Port (Primary only)	Port *port*	Defines the `SERVER_PORT` environment variable used by CGI scripts.

For example, if your system has eight IP addresses configured, but you want the public Web server to listen on only two of those ports, you can explicitly define these two addresses as the Active IP Addresses for the server. Optionally, you can specify a different port for each address. (Port 80 is normally used.)

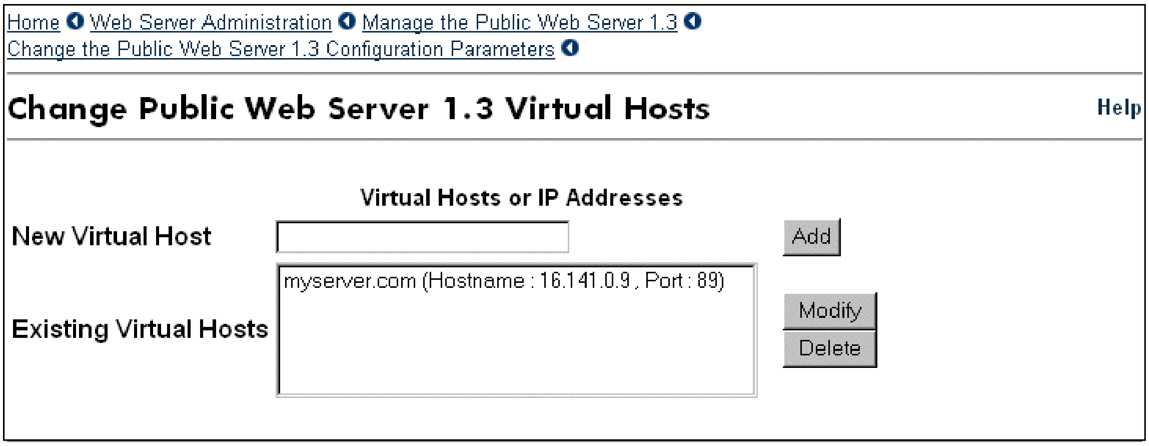
If you want to listen to all known IP addresses on more than one port (for example, Ports 80 and 81), specify Active Port 80 and Active Port 81 and leave the Active IP Address field blank for both ports.

2.1.4 Changing Virtual Host Parameters for the Public Web Servers

You can specify virtual host parameters for the public Web servers only.

1. From the Change Configuration Parameters form for either the Public Web Server 1.3 or Public Web Server 2.0 (for example, see Figure 2-1), choose Change Virtual Hosts. Figure 2-5 shows the Change Public Web Server 1.3 Virtual Hosts form.

Figure 2-5 Change Public Web Server 1.3 Virtual Hosts Form



Home Web Server Administration Manage the Public Web Server 1.3 Change the Public Web Server 1.3 Configuration Parameters

Change Public Web Server 1.3 Virtual Hosts

Virtual Hosts or IP Addresses

New Virtual Host

Existing Virtual Hosts

myserver.com (Hostname : 16.141.0.9 , Port : 89)

The first time you access the Change Virtual Hosts form (Figure 2-5), the only choice is to add a new virtual host. Thereafter, each virtual host you add is displayed on this form in the Existing Virtual Hosts list box.

Note:

Before you create each virtual host, you must create a Listen directive in the Public Web Server for the virtual host using the Change Listening Ports and Addresses form. If no Listen directive is specified for the virtual host, the Public Web Server will not respond to client requests that match the host names, IP addresses, and ports specified in its virtual host directives. See section Section 2.1.3: Changing Listening Port and IP Address Parameters for information on creating Listen directives for the Public Web Server.

To add a new virtual host:

a. Enter the host names and/or IP addresses (with optional port values) into the New Virtual Host field on the form as they would appear in a virtual host Web server directive.

b. Click on Add. The Add Public Web Server Virtual Host form is displayed. (This form is similar to the Modify Public Web Server Virtual Host form in Figure 2-6).

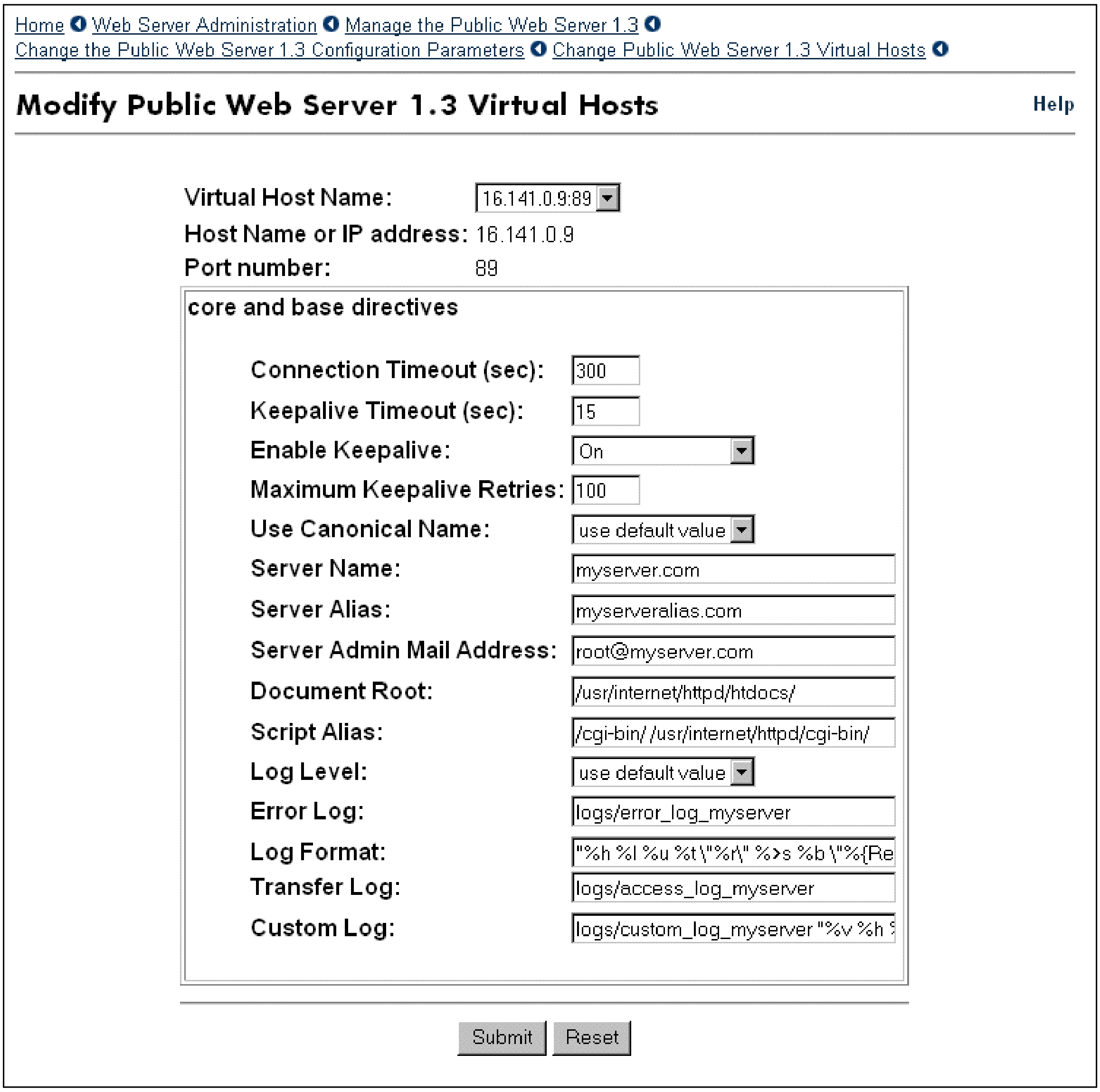
c. Specify the type of virtual host (Name-based or IP-based) and any additional directives for the new virtual host:

- When the value for the Virtual Host Name field matches the hostnames and IP address values you entered, a Name-based virtual host is created.
- When you set the value of the Virtual Host Name field to NONE, an IP-based virtual host is created.

d. Click on Submit to add the new virtual host to the Public Web Server.

2. To change the configuration for an existing virtual host, select the virtual host from the list box and click on Modify. The Modify Public Web Server Virtual Hosts form is displayed (Figure 2-6).

Figure 2-6 Modify Public Web Server 1.3 Virtual Hosts Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 1.3 ◀ Change the Public Web Server 1.3 Configuration Parameters ◀ Change Public Web Server 1.3 Virtual Hosts ◀

Modify Public Web Server 1.3 Virtual Hosts

Virtual Host Name: 16.141.0.9:89
Host Name or IP address: 16.141.0.9
Port number: 89

core and base directives

Connection Timeout (sec): 300
Keepalive Timeout (sec): 15
Enable Keepalive: On
Maximum Keepalive Retries: 100
Use Canonical Name: use default value
Server Name: myserver.com
Server Alias: myserveralias.com
Server Admin Mail Address: root@myserver.com
Document Root: /usr/internet/httpd/htdocs/
Script Alias: /cgi-bin/ /usr/internet/httpd/cgi-bin/
Log Level: use default value
Error Log: logs/error_log_myserver
Log Format: "%h %l %u %t \"%r\" %>s %b \"%{Re
Transfer Log: logs/access_log_myserver
Custom Log: logs/custom_log_myserver "%v %h

Table 2-5 shows which Secure Web Server directive is associated with each field on the Modify Public Web Server Virtual Hosts form (Figure 2-6) and the type of value expected.

When a field on this form is left blank or when the “use default value” option is selected for the field, the directive associated with the field is not included in (or removed from) the virtual host. In this case, the virtual host inherits the value of the associated directive from the global-specified value of the directive for the Public Web Server:

- If no global value is specified for the associated directive, the directive's default value is used as the value of the directive in the virtual host.
- If the default value for the directive is “unspecified” (for example, as with Script Alias), the directive does not apply to the virtual host when the field in the form is left blank.

Table 2-5 Virtual Hosts Parameters and Associated Directives

Parameter	Directive	Description
Virtual Host Name	NameVirtualHost *hostname[:port]* | *IP address[:port]*	Name of the Name-based virtual host. When this directive is not set to NONE, the value should always match the Host Name or IP Address and Port Number of the virtual host. Setting the value to NONE creates an IP-based virtual host by not setting the NameVirtualHost directive.
Host Name or IP Address and Port Number	VirtualHost *hostname[:port]* | *IP address[:port]*	Host name or IP address of the virtual host; port number is optional.
Connection Timeout	Timeout *seconds*	The amount of time the server supporting the virtual host will wait for the following events: the total amount of time it takes to receive a GET request, the amount of time between receipt of TCP packets on a POST or PUT request, the amount of time between ACKs on transmissions of TCP packets in responses, the default value of this directive is 300 second.
Keepalive Timeout	KeepAliveTimout *seconds*	The number of seconds the server supporting the virtual host will wait for a subsequent request before closing the connection. Once a request has been received, the timeout value specified by the Timeout directive applies. Setting KeepAliveTimeout to a high value may cause performance problems in heavily loaded servers. The higher the timeout, the more server processes will be kept occupied waiting on connections with idle clients. The default value of this directive is 15 seconds.
Enable Keepalive	KeepAlive [On | Off |use default value]	Provides for long-lived HTTP sessions that allow multiple requests to be sent over the same TCP connection. These connection types are the default for HTTP 1.1 clients. To enable KeepAlive connections, set KeepAlive to On. The default value of this directive is On.
Maximum KeepAlive Retries	MaxKeepAliveRequests *number*	Limits the number of requests allowed on a persistent connection (KeepAlive On) for the virtual host. The default value of this directive is 100.
Use Canonical Name	UseCanonicalName [On | Off | DNS | use default value]	Configures the way the server serving the virtual host determines its own name and port. With UseCanonicalName set to On, the server will use the hostname and port specified in the Server Name directive. With UseCanonicalName set to Off, the server will form self-referential URLs using the hostname and port supplied by the clients Host: header. Set UseCanonicalName to DNS for use with mass IP-based virtual hosting when you need to support older client systems that do not provide a Host: header. The default value for this directive is On.
Server Name	ServerName *hostname*	Host name; used in URL parsing.
Server Alias	ServerAlias *hostname* [*hostname*]...	Sets the alternate names for the virtual host.
Document Root	DocumentRoot *path*	Full path of the directory containing the default Web homepage for the specified Host Name or IP Address. The default value for this directive is /usr/local/apache/htdocs.

Table 2-5 Virtual Hosts Parameters and Associated Directives (*continued*)

Parameter	Directive	Description
Virtual Document Root	VirtualDocumentRoot `interpolated-directory`	Dynamically configures the location of the document root for a given virtual host based on the value of server name. If *interpolated-directory* is set to `none`, then VirtualDocumentRoot is turned off. This directive cannot be used in the same context as the VirtualDocumentRootIP directive. This directive applies only to the Secure Web Server 2.0.
Virtual Document Root IP	VirtualDocumentRootIP `interpolated-directory`	Dynamically configures the location of the document root for a given virtual host based on the value of the server IP address. If *interpolated-directory* is set to `none`, then VirtualDocumentRootIP is turned off. This directive cannot be used in the same context as the VirtualDocumentRoot. This directive applies only to the Secure Web Server 2.0.
Script Alias	ScriptAlias `URL-pathfile-path` | `directory-path`	Allows CGI scripts to be stored in the local file system other than under the Document Root. URLs with a (%-decoded) path beginning with *URL-path* will be mapped to local files beginning with the second argument, which is a full pathname on the local file system.
Virtual Script Alias	VirtualScriptAlias `interpolated-directory` | `none`	Dynamically configures the location of the CGI directory for a given virtual host based on the value of the server name. If *interpolated-directory* is set to `none` then VirtualScriptAlias is turned off. This directive cannot be used in the same context as the VirtualScriptAliasIP directive. This directive applies only to the Secure Web Server 2.0.
Virtual Script Alias IP	VirtualScriptAliasIP `interpolated-directory` | `none`	Dynamically configures the location of the CGI directory for a given virtual host based on the value of the server IP address. If *interpolated-directory* is set to `none` then VirtualScriptAliasIP is turned off. This directive cannot be used in the same context as the VirtualScriptAlias directive. This directive applies only to the Secure Web Server 2.0.
Server Admin Mail Address	ServerAdmin `e-mail address`	E-mail address of the Web system administrator.
Log Level	LogLevel `emerg | alert | crit | error | warn | notice | info | debug | use default value`	Adjusts the verbosity of the messages recorded in the ErrorLog file for the virtual host. When set to use default value, the LogLevel directive is not set for the virtual host, and the value is inherited from the global server value. The default value of this directive is `warn`.
Error Log	ErrorLog `path`	Full or relative path to error log file. Relative paths are specified from the Web server's root directory. The default value of this directive is `logs/error_log`.

Table 2-5 Virtual Hosts Parameters and Associated Directives (*continued*)

Parameter	Directive	Description
Log Format	LogFormat *format* | nickname [*nickname*]	The LogFormat directive can take one of two forms. In the first form, where only one argument is specified, this directive sets the log format that will be used by logs specified in subsequent TransferLog directives. The second form of the LogFormat directive associates an explicit format with a nickname. This nickname can then be used in subsequent CustomLog directives. When a nickname is specified, this directive does not affect subsequent TransferLog directives. The default value of this directive is "%h %l %u %t \"%r\" + .
Transfer Log	TransferLog *path*	Full or relative path of the transfer (access) log file. Relative paths are specified from the Web server root directory.
Custom Log	CustomLog *path*	Full or relative path and format of a log file for the virtual host. Relative paths are specified from the Web server root directory.

For a comprehensive document on virtual host support, see the following Web sites:

`http://www.apache.org/docs/vhosts/index.html (for Secure Web Server 1.3)`

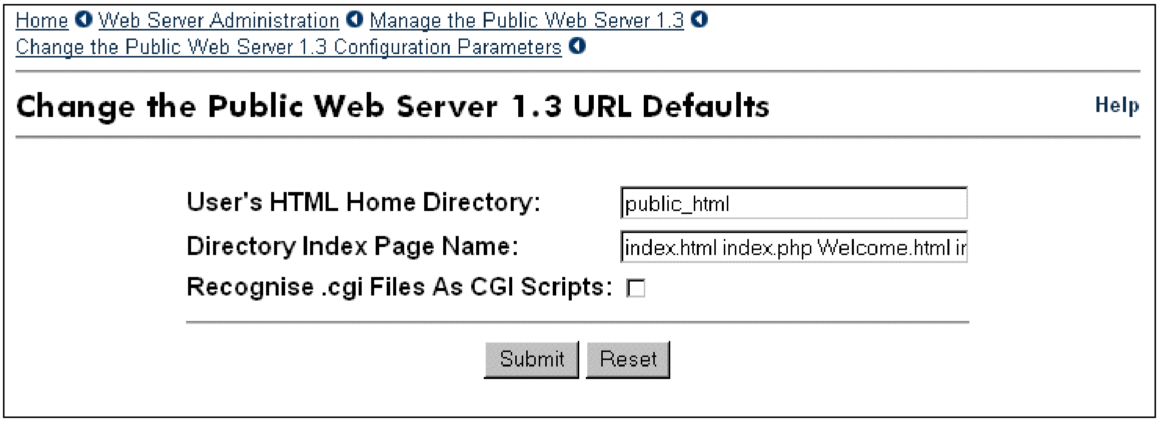
`http://httpd.apache.org/docs-2.0/vhosts (for Secure Web Server 2.0)`

2.1.5 Changing URL Default Parameters for the Public Web Servers

This section describes the steps to change the URL Default Parameters for Public Web Server 1.3. The steps for Public Web Server 2.0 are similar. You can specify the URL default parameters for the public Web server only.

1. From the Change Public Web Server 1.3 Configuration Parameters form (Figure 2-1), choose Change URL Defaults. Figure 2-7 shows the Change Public Web Server 1.3 URL Defaults form.

Figure 2-7 Change Public Web Server URL Defaults Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 1.3 ◀ Change the Public Web Server 1.3 Configuration Parameters ◀

Change the Public Web Server 1.3 URL Defaults

User's HTML Home Directory: public_html

Directory Index Page Name: index.html index.php Welcome.html ir

Recognise .cgi Files As CGI Scripts: □

2. From the Change Public Web Server URL Defaults form (Figure 2-7), specify the default HTML directory and default homepage (index page) for users on your system. By convention, the default HTML directory is `public_html` and the default homepage is `index.html` on UNIX systems.

When the Recognize .cgi Files As CGI Scripts parameter is enabled, files with the `.cgi` extension in the user's default HTML directory (or in a directory where CGI script execution is enabled) are executed as CGI scripts.

Table 2-6 shows which Secure Web Server directive is associated with each field on the Change URL Defaults form and the type of value expected.

Table 2-6 URL Default Parameters and Associated Directives

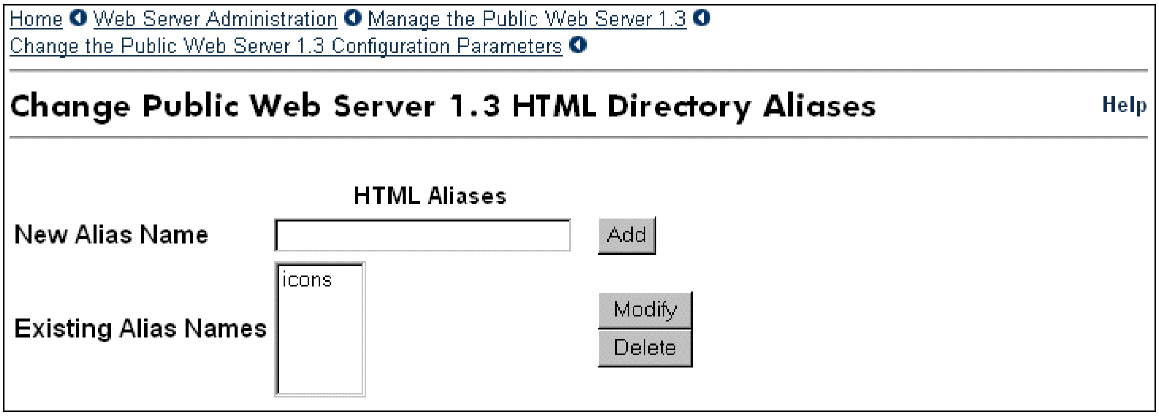
Parameter	Directive	Description
User's HTML Home Directory	UserDir *path*	Path relative to a user's home directory for the user's HTML home directory. The default is `public_html`.
Directory Index Page Name	DirectoryIndex *filename list*	One or more file names, separated by spaces, that define the default page displayed when an HTTP request specifies a directory path only (without a file name).
Recognize .cgi Files As CGI Scripts	AddHandler `cgi-script.cgi`	When this field is enabled, the comment character in this line is removed from the `httpd.conf` file. When this field is disabled, the line is commented out.

2.1.6 Changing HTML Directory Alias Parameters for the Public Web Server

You can specify the HTML Directory Alias parameters for the public Web servers only. This section describes the steps to change the HTML Directory Aliases for the Public Web server 1.3. The steps are similar for the Public Web server 2.0.

From the Change Public Web Server 1.3 Configuration Parameters form (Figure 2-1), choose Change HTML Directory Aliases. Figure 2-8 shows the Change Public Web Server 1.3 HTML Directory Aliases form.

Figure 2-8 Change Public Web Server HTML Directory Aliases Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 1.3 ◀ Change the Public Web Server 1.3 Configuration Parameters ◀

Change Public Web Server 1.3 HTML Directory Aliases

URL paths are rooted only by aliases, not by actual directories. The system-defined aliases are as follows:

- `icons` — Defines the directory to search for browser-specific icons.

  When an HTTP request specifies a directory other than the user's HTML home directory (Table 2-6), the icons used in the resulting display to identify subdirectories and files are obtained from the directory associated with the `icons` alias.

- `copyrights` — Defines the directory in which the copyright information is installed.
- `documents` — Defines the directory in which the book files are installed.

Normally, these aliases should not be changed or deleted. However, you can specify a new HTML alias for any directory by providing an alias name and the full path name of the directory you want to associate with the alias.

To add a new HTML alias:

1. On the Change HTML Directory Aliases form, enter the new alias name in the New Alias Name field and click on Add.
2. On the Add HTML Directory Aliases form, specify the full pathname for the directory associated with the new alias in the Actual Directory field.
3. Click on Submit.

   The Web Server Administration utility displays a confirmation message indicating that the configuration file has been successfully updated.

4. Click on Submit to have the public Web server on the indicated port reread its configuration file. Wait a few seconds before using the navigation bar.

When you determine that an alias is no longer useful, you can remove it by selecting the alias name from the Existing Alias Names list box and clicking on Delete.

Table 2-7 shows which Secure Web Server directive is associated with each field on the Change Public Web Server HTML Directory Aliases form and the type of value expected.

Table 2-7 HTML Directory Alias Parameters and Associated Directives

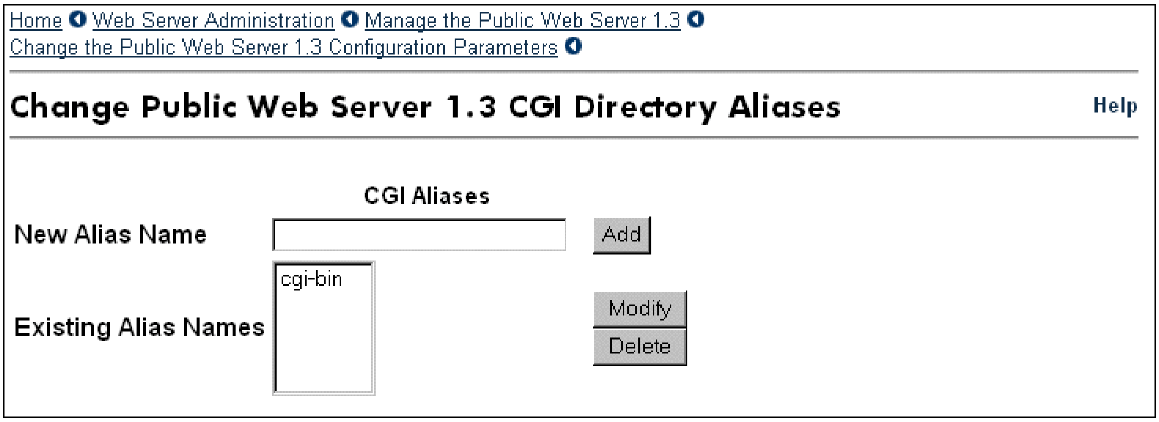
Parameter	Directive	Description
Alias Specification and Actual Directory	Alias *aliaspath*	Alias Specification (New Alias Name) specifies the *alias* part of the directive and Actual Directory specifies the *path*.

2.1.7 Changing CGI Directory Alias Parameters for the Public Web Server

You can specify the CGI Directory Alias configuration parameters for the public Web server only.

1. From the Change Public Web Server Configuration Parameters form (Figure 2-1), choose Change CGI Directory Aliases. Figure 2-9 shows the Change Public Web Server 1.3 CGI Directory Aliases form.

Figure 2-9 Change Public Web Server CGI Directory Aliases Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 1.3 ◀
Change the Public Web Server 1.3 Configuration Parameters ◀

Change Public Web Server 1.3 CGI Directory Aliases

2. Specify an alias name and the full path name of the directory you want to associate with the alias.

Table 2-8 shows which Secure Web Server directive is associated with each field on the Change Public Web Server CGI Directory Aliases form (Figure 2-9) and the type of value expected.

Table 2-8 CGI Directory Alias Parameter and Associated Directive

Parameter	Directive	Description
Alias Specification and Actual Directory	ScriptAlias *aliaspath*	Alias Specification (New Alias Name) specifies the *alias* part of the directive and Actual Directory specifies the *path*.

2.1.8 Changing Logging and Reporting Parameters

Use the Change Logging and Reporting Parameters form to specify the following:

- The host name associated with an IP address in the log file. (Server performance can decrease when you enable host name lookup.)
- E-mail address for mail intended for the server administrator (if not specified anywhere else in the configuration files).
- The URL of the HTML page to display when the browser receives any of the following error codes:

Note:

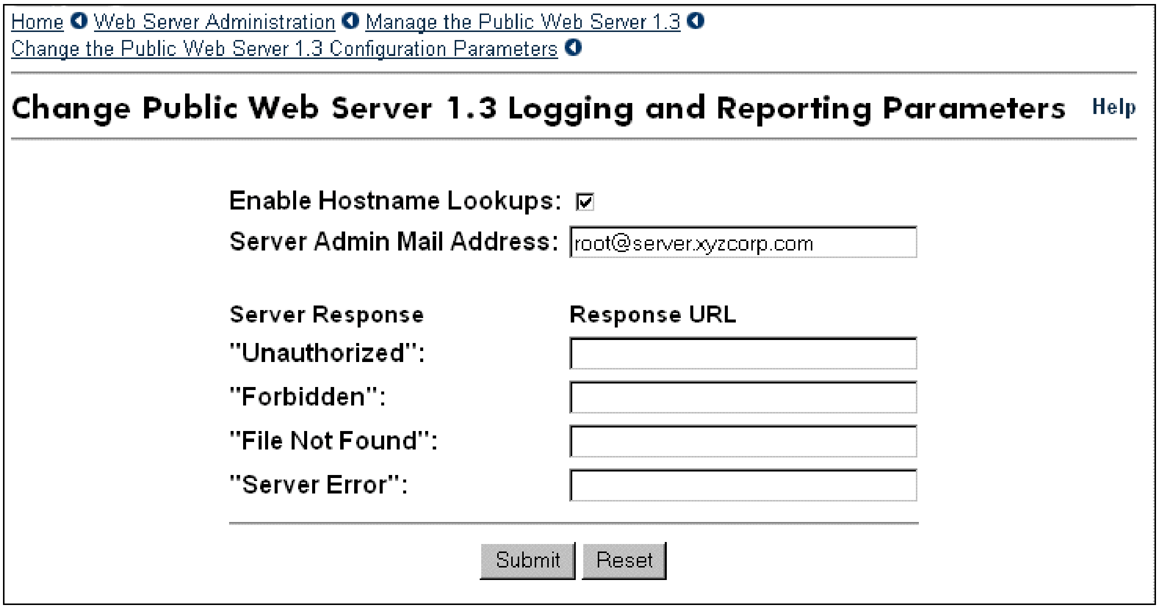
Although both servers are capable of generating the following errors, the Change Logging and Reporting Parameters form for different versions of the Secure Web Server display different lists of errors. Errors that appear only in the Secure Web Server 2.0 Change Logging and Reporting Parameters form are specified for Version 2.0 only.

- — Bad Gateway — The server, when acting as a gateway or proxy, received an invalid response from a server (Version 2.0 only).
- — Bad Request — Usually caused by a malformed URL (Version 2.0 only).
- — Unauthorized — Usually caused by an incorrect user name or password.
- — Forbidden — Access to the directory, location, or file is explicitly prohibited or the file is protected.
- — File Not Found — File or path name alias does not exist.
- — Gone — The requested resource is no longer available at the server (Version 2.0 only).
- — Method Not Allowed — File or path name alias does not exist (Version 2.0 only).
- — Not Implemented — The server does not support the functionality required to fulfill the request (Version 2.0 only).
- — Precondition Failed — The precondition given in one or more of the request-header fields evaluated to false (Version 2.0 only).
- — Request Timeout — The client did not produce a request within the time that the server was prepared to wait (Version 2.0 only).
- — Request Entity Too Large — The request entity is larger than the server is willing or able to process (Version 2.0 only).
- — Service Unavailable — The server is temporarily overloaded or maintenance is required (Version 2.0 only).
- — Server Error — Usually caused by a malformed HTTP header generated by a CGI script.
- — Variant Also Varies —The HTTP variant also varies; the status is not yet defined. (Version 2.0 only).

From the Change Public Web Server 2.0 Configuration Parameters form (Figure 2-1), choose Change Logging and Reporting Parameters.

Figure 2-10 shows the Change Logging and Reporting Parameters form for the Public Web Server 2.0. The form for the Public Web Server 1.3 is has fewer server responses. (See Table 2-9.) This form is also available for the Administration Web Server.

Figure 2-10 Change Public Web Server Logging and Reporting Parameters Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 1.3 ◀
Change the Public Web Server 1.3 Configuration Parameters ◀

Change Public Web Server 1.3 Logging and Reporting Parameters

Enable Hostname Lookups: ☑

Server Admin Mail Address: root@server.xyzcorp.com

Server Response	Response URL
"Unauthorized":	
"Forbidden":	
"File Not Found":	
"Server Error":	

Table 2-9 shows which Secure Web Server directive is associated with each field on the Change Logging and Reporting form (Figure 2-10) and the type of value expected.

Table 2-9 Logging and Reporting Parameters and Associated Directives

Parameter	Directive	Description
Enable Hostname Lookups	HostnameLookups on | off	When set to on, the server performs DNS lookups on IP addresses to include host names in logging records.
Server Admin Mail Address	ServerAdmin *`e-mail address`*	E-mail address displayed with some error pages.
"Unauthorized" Error Response URL	ErrorDocument 401 *`URL`* | *`string`*	Specifies a page or text string to display upon receiving a "bad password" error. If specified, the URL for 401 errors must be local. (The `http://host.domain.name` prefix is not permitted.)
"Forbidden" Error Response URL	ErrorDocument 403 *`URL`* | *`string`*	Specifies a page or string to display upon receiving a "no authorization" or "file access" error.
"File Not Found" Error Response URL	ErrorDocument 404 *`URL`* | *`string`*	Specifies a page or text string to display upon receiving a "file not found" error.
"Server Error" Error Response URL	ErrorDocument 500 *`URL`* | *`string`*	Specifies a page or text string to display upon receiving an internal error or CGI format error (most likely related to a problem with HTTP header information).

2.2 Changing Public Web Server User Accounts

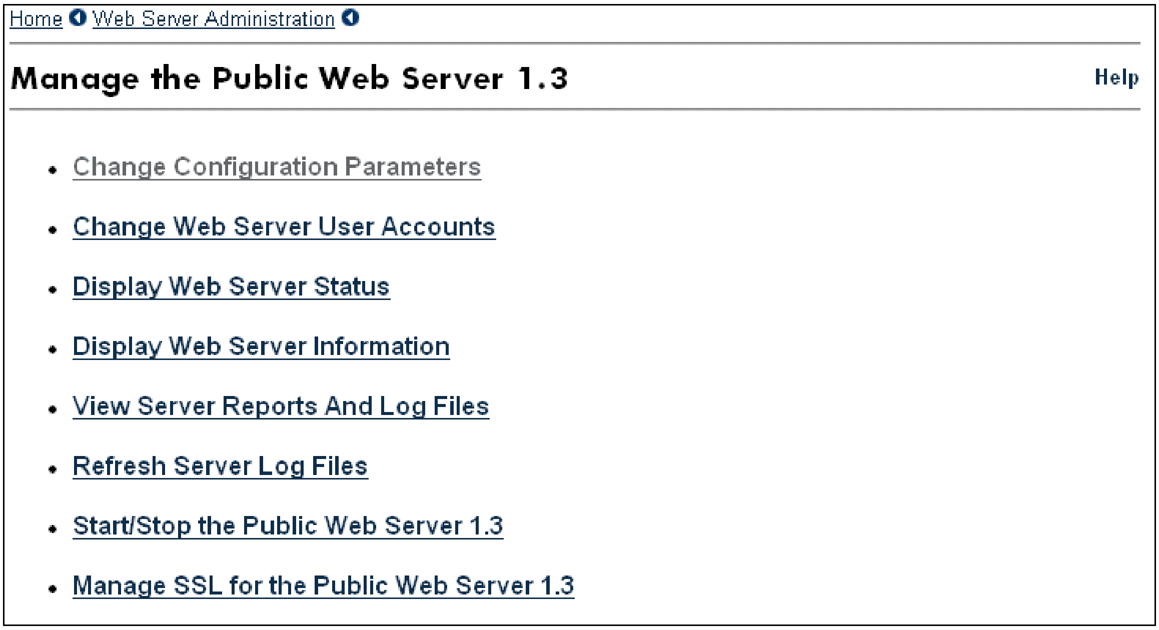
You can establish Secure Web Server user accounts to control access to the public Web servers. You can enable a different level of access to each combination of user name and password that you specify. The password you specify for a Web server user account is not a UNIX system password; that is, you will not find these passwords in the `/etc/passwd` file.

The first time you access the Change Web Server User Accounts menu, the only option is to add a new Web server user account. Thereafter, each user account you create is displayed on this menu in the Existing Web Server Users list box, allowing you to change the password for the account or delete the account.

To add a Web server user account to control access to the public Web server:

1. On the Web Server Administration menu, choose the public Web server you want to manage. Figure 2-11 shows the Manage the Public Web Server 1.3 menu and available options.

Figure 2-11 Manage the Public Web Server Form



Home ◀ Web Server Administration ◀

Manage the Public Web Server 1.3

Help

- Change Configuration Parameters
- Change Web Server User Accounts
- Display Web Server Status
- Display Web Server Information
- View Server Reports And Log Files
- Refresh Server Log Files
- Start/Stop the Public Web Server 1.3
- Manage SSL for the Public Web Server 1.3

2. From the Manage the Public Web Server 1.3 menu, choose Change Web Server User Accounts.
3. On the Change Public Web Server User Accounts form, enter the account name in the New Web Server User field.
4. Click on Add. The Add Public Web Server User Account form is displayed.
5. Enter a password in the New Password field.
6. Verify the password for the user by typing the same password in the Verify Password field.
7. Click on Submit.

   The Web Server Administration utility displays a confirmation message indicating that the new user account has been successfully created. You can use the navigation bar at the top of the page to return to the Change Public Web Server User Accounts form.

To change a user's password, select the user name from the Existing Web Server Users list box and click on Modify. Specify a new password, verify the password, and click on Submit.

To delete a user account, select the user name from the Existing Web Server Users list box and click on Delete.

2.3 Displaying Public Web Server Status

To display the status of Public Web Server 1.3, from the Manage the Public Web Server 1.3 menu, choose Display Web Server Status. Similarly, to display the status of Public Web Server 2.0, from the Manage the Public Web Server 2.0 menu, choose Display Web Server Status.

The Web Server Status page allows you to see how well your server is performing. The current server statistics are displayed in an easy-to-read form.

The Display Server Status and Display Server Information links under the Manage the Public Web Server menu return a “Forbidden server” error if you try to access them from a remote Web browser after opening up access controls to remote systems on the Administration server. To avoid this problem, open access controls on the Location `/server-info` and Location `/server-status` entries for the public Web server in the Change Access Control Entries form under Change Configuration Parameters.

For more information on the data displayed on the Web Server Status page, go to one of the following Apache Web site URLs:

`http://www.apache.org/docs/mod/mod_status.html (for Secure Web Server 1.3)`

`http://httpd.apache.org/docs-2.0/mod/mod_status.html (for Secure Web Server 2.0)`

2.4 Displaying Public Web Server Information

To display information for the public Web server, on the Manage the Pubic Web Server menu (1.3 or 2.0), choose Display Web Server Information.

The Web Server Information page displays a comprehensive overview of the server configuration, including all installed modules and directives in the configuration files.

The Display Server Status and Display Server Information links under the Manage the Public Web Server menu return a "Forbidden server" error if you try to access them from a remote Web browser after opening up access controls to remote systems on the Administration server. To avoid this problem, open access controls on the Location `/server-info` and Location `/server-status` entries for the public Web server in the Change Access Control Entries form under Change Configuration Parameters.

For more information on the data displayed on the Web Server Information page, go to one of the following Apache Web site URLs:

`http://www.apache.org/docs/mod/mod_status.html (for Secure Web Server 1.3)`

`http://httpd.apache.org/docs-2.0/mod/mod_status.html (for Secure Web Server 2.0)`

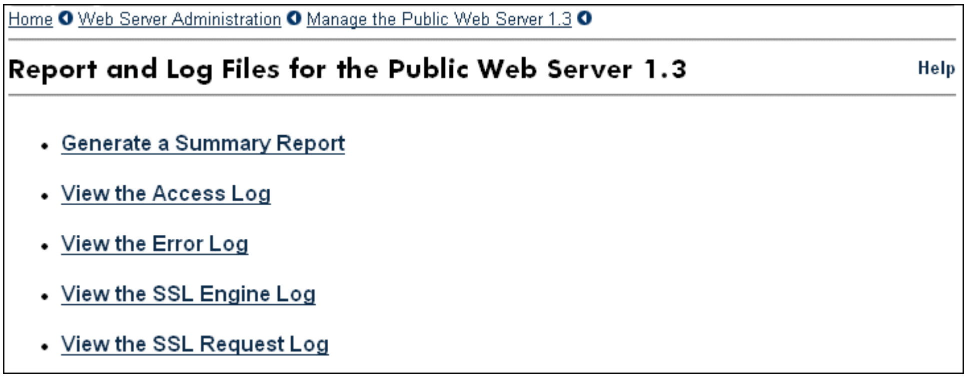
2.5 Viewing Web Server Reports and Log Files

During its normal operation, the Web server puts information in two log files. The access log keeps track of requests for use of this server and the information requested. The error log maintains a record of errors that occurred since the log file was last refreshed. You should periodically save and recreate these log files so they do not get too large. See Section 2.6: Refreshing the Administration Web Server Log Files.

To view the access log file or error log file for a Web server:

1. From the Web Server Administration menu, choose the Manage form for the version of the server whose log you want to view (for example, the Public Web Server 1.3). In this case, the Manage the Public Web Server 1.3 menu is displayed. See Figure 2-11.
2. From the Manage the Public Web Server 1.3 menu, choose View Server Reports and Log Files. The Report and Log Files for the Public Web Server 1.3 menu is displayed (Figure 2-12).

Figure 2-12 Reports and Log Files for the Public Web Server


Home
Web Server Administration
Manage the Public Web Server 1.3
Report and Log Files for the Public Web Server 1.3
Help
Generate a Summary Report
View the Access Log
View the Error Log
View the SSL Engine Log
View the SSL Request Log

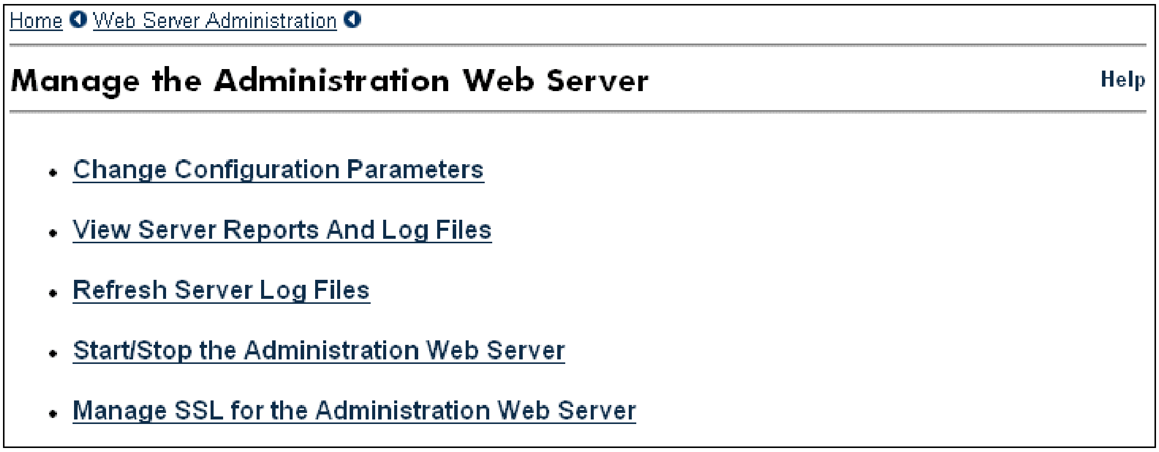
3. Choose the item corresponding to the log file you want to view.

   The entries in the chosen log file are shown 100 lines at a time with the most recent entries first. You can use the navigation bar at the top of each page to return to the Report and Log Files menu.

To generate the activity reports for any one of the Web Server instances:

1. From the Web Server Administration menu, choose the server for which you want to generate activity statistics; for example, the Administration Web Server. The Manage the Administration Web Server menu is displayed (Figure 2-13).

Figure 2-13 Manage the Administration Web Server Menu



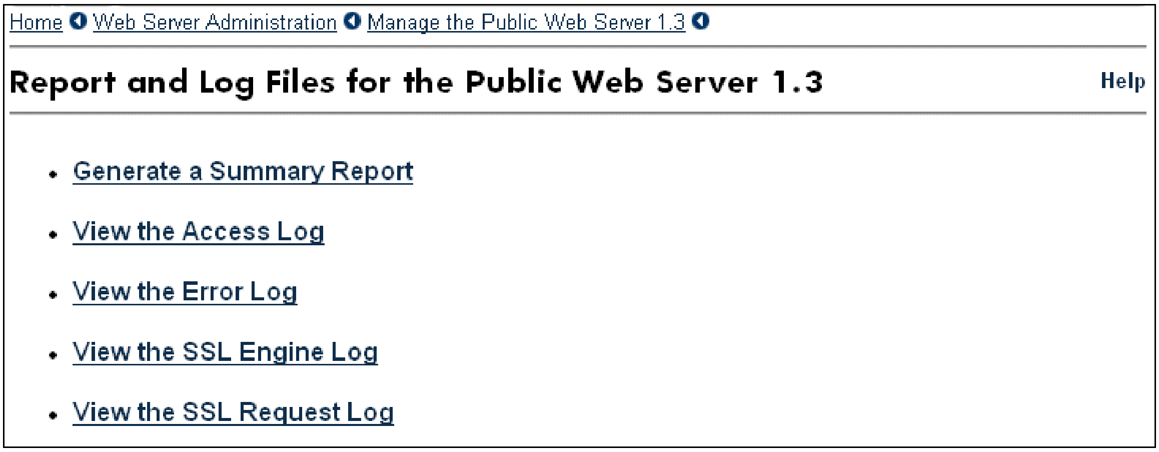
Home ◀ Web Server Administration ◀

Manage the Administration Web Server

- Change Configuration Parameters
- View Server Reports And Log Files
- Refresh Server Log Files
- Start/Stop the Administration Web Server
- Manage SSL for the Administration Web Server

2. From the Manage the Administration Web Server menu, choose View Server Reports and Log Files. The Report and Log Files for the Administration Web Server menu is displayed (Figure 2-14). This menu contains more reports than listed on the Reports and Log Files for the Public Web Server menu.

Figure 2-14 Reports and Log Files for the Administration Web Server Menu



Home ◀ Web Server Administration ◀ Manage the Public Web Server 1.3 ◀

Report and Log Files for the Public Web Server 1.3

- Generate a Summary Report
- View the Access Log
- View the Error Log
- View the SSL Engine Log
- View the SSL Request Log

3. From the Reports and Log Files for the Administration Web Server menu, click on Generate a Summary Report.

   For your convenience, a link to the Analog HTML documentation is also provided at the bottom of the page; look for `This analysis was produced by analogx.xx`, where *x.xx* indicates the version number.

   The activity reports are generated using `analog`, an Open Source utility that analyzes log files. The `analog` configuration file is located in `/usr/internet/httpd/admin/analog/analog.cfg`.

   Table 2-10 describes the various activity reports that you can generate for the public and administration instances of the Secure Web Server:

Table 2-10 Activity Reports for the Secure Web Servers

Report	Description
Summary Report	For the time period shown at the top of the page, the following statistics are shown: the total requests that were completed, failed, and redirected; the number of distinct hosts served; the number of corrupt log file entries; and the total bytes transferred.
Monthly Report	Shows how many requests were processed by month.
Daily Summary Report	Shows how many requests were processed each day since the last time the server was started.
Hourly Summary Report	Shows how many requests were processed each hour.

Table 2-10 Activity Reports for the Secure Web Servers *(continued)*

Report	Description
Domain Report	Shows all domains with any traffic, sorted by amount of traffic.
Directory Report	Shows all directories to depth 1 with at least 0.01% of the traffic, sorted by amount of traffic.

For more information, visit the `analog` Web site:

http://www.analog.cx

2.6 Refreshing the Administration Web Server Log Files

To refresh the access log, the error log, or both, follow these steps:

1. From the Secure Web Server Administration menu, choose the server for which you want to refresh the log files; for example, the Administration Web Server. The Manage the Administration Web Server menu is displayed. (See Figure 2-13.)
2. From the Manage the Administration Web Server menu, choose Refresh Server Log Files.
3. On the Refresh Server Log Files form, select the check box corresponding to the log file you want to refresh. You can select one file or more files.
4. Click on Submit.

   For each log file you select, the Web Server Administration utility makes a backup copy of the log file and creates an empty file to replace it. The Web Server Administration utility also restarts the `httpd` server daemon.

2.7 Starting and Stopping the Secure Web Server

To stop or restart the Secure Web Server instances:

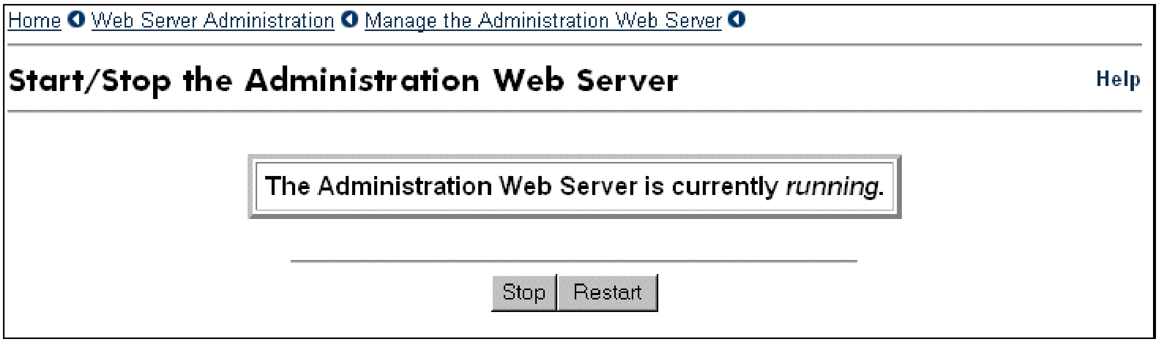
1. From the Web Server Administration menu, choose the server you want to start or stop; for example, the Administration Web Server. The Manage the Administration Web Server menu is displayed (Figure 2-13).
2. From the Manage the Administration Web Server menu, choose Start/Stop the Administration Web Server.

   If the server is running, the Web Server Administration utility shows you the current status of the server and offers the following operations:

   - Stop—Shuts down the server daemon listening on the port shown in the title of the form. Use this operation to prevent the server from responding to requests.
   - Restart—Restarts the server daemon listening on the port shown in the title of the form. Use this operation to enable any change to the server configuration files.

   Figure 2-15 shows the Start/Stop the Administration Web Server form when the server is running.

Figure 2-15 Start/Stop the Administration Web Server Form



Start/Stop the Administration Web Server

The Administration Web Server is currently *running.*

Home
Web Server Administration
Manage the Administration Web Server
Help
Stop
Restart

If the server is not running, the utility offers the following control operations:

- Start — Starts the server daemon listening on the port shown in the title of the form.
- Restart — Stops and restarts the server daemon listening on the port shown in the title of the form. Use this operation to enable any change to the server configuration files.

3. Click on the button corresponding to the operation you want to perform. The Web Server Administration utility confirms the request and performs the operation.

The Start/Stop form for Public Web Server 1.3 or Public Web Server 2.0 offers the following additional control operation:

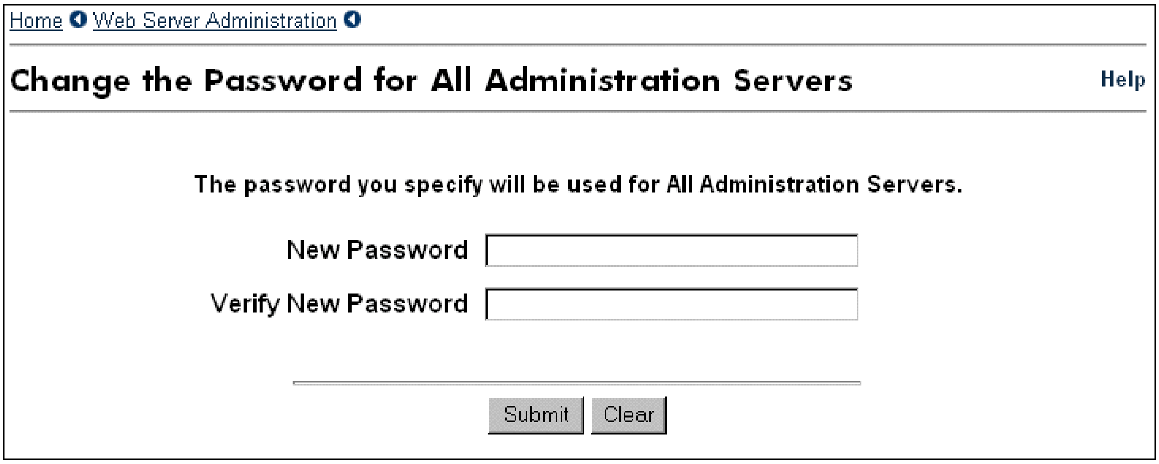
Save Options — Saves the Web server daemon command line options. The options are displayed in the text field labeled "Start with options" in the Start/Stop form. Changing the options displayed in the "Start with options" text field changes the options that will be saved. These options take affect the next time the Web server is started. Clicking on the Start or Restart button in the form also saves the displayed options.

2.8 Changing the Password for the Administration Web Server

To change the password used for the Administration Web Server:

1. From the Web Server Administration menu, choose Change the Password for All Administration Servers. The Change the Password for All Administration Servers form is displayed (Figure 2-16).

Figure 2-16 Change the Password for All Administration Servers Form



Home ◂ Web Server Administration ◂

Change the Password for All Administration Servers

The password you specify will be used for All Administration Servers.

New Password

Verify New Password

2. Enter the new password in the New Password field and again in the Verify New Password field.
3. Click on Submit. The new password takes effect immediately.

If you decide not to change the password, cancel the operation by clicking on one of the following:

- The Clear button at the bottom of the form
- One of the links on the navigation bar at the top of the form to go to another Administration menu

2.9 Allowing Remote Access to the Administration Web Server

The installation procedure installs the Administration Web Server on port 8081 and initially allows access to the server from the local system only.

Note:

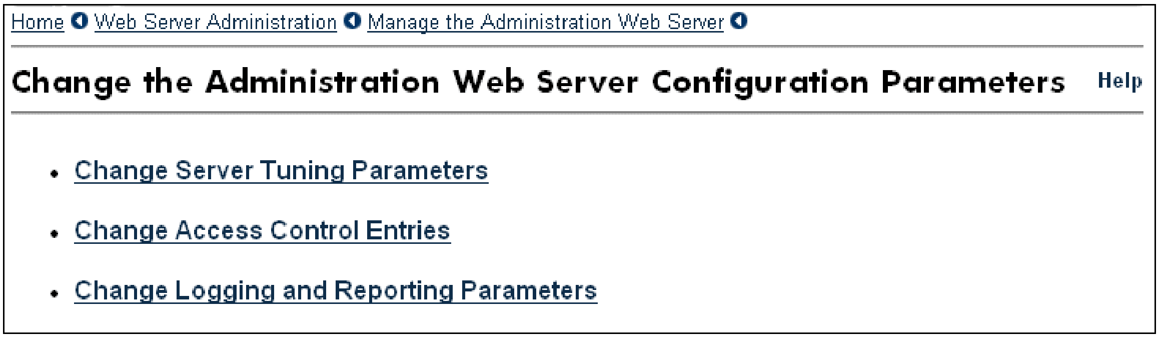
Using a Web server on a remote system to manage user accounts and other system services poses a security risk. That is, unless the Secure Socket Layer (SSL) is enabled on your Web server, all data, including passwords, is transmitted between the Web server and the browser in clear text. This unencrypted data is subject to interception by network snooping.

Carefully evaluate the security risks to your system before you enable remote access to the Administration utility or other server administration. See Chapter 5 for information on setting up your Web server with SSL.

To allow access to the Administration Web Server from remote systems:

1. From the Web Server Administration menu, choose Manage the Administration Web Server. The Manage the Administration Web Server menu is displayed (Figure 2-13).
2. From the Manage the Administration Web Server menu, choose Change Configuration Parameters. The Change Administration Web Server Configuration Parameters menu is displayed (Figure 2-17).

Figure 2-17 Change Administration Web Server Configuration Parameters Menu



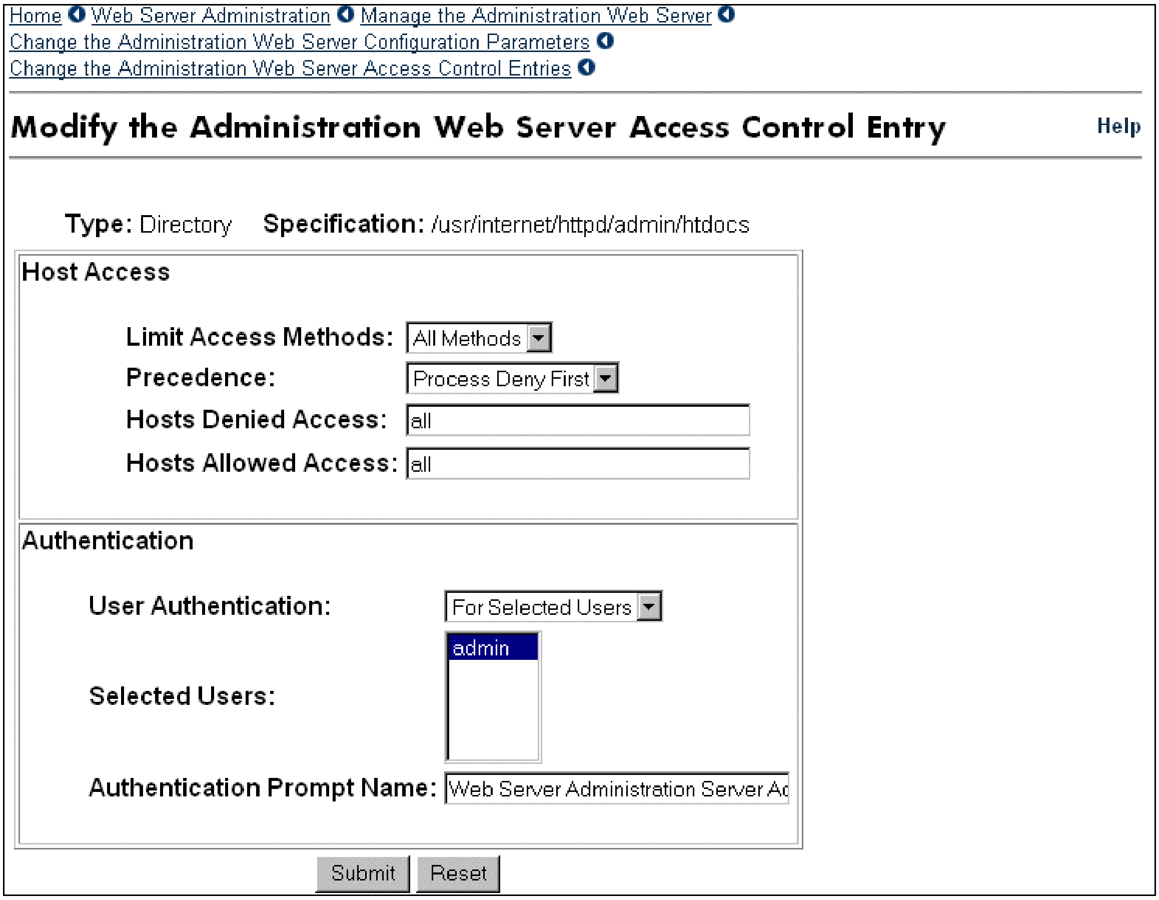
Home › Web Server Administration › Manage the Administration Web Server ›

Change the Administration Web Server Configuration Parameters

- Change Server Tuning Parameters
- Change Access Control Entries
- Change Logging and Reporting Parameters

3. From the Change Administration Web Server Configuration Parameters menu, choose Change Access Control Entries (Figure 2-3).
4. On the Change Access Control Entries menu, select `Directory /usr/internet/httpd/admin/htdocs` from the Existing Access Control Entries list box, then click on Modify. The Modify Administration Web Server Access Control Entry form is displayed (Figure 2-18).

Figure 2-18 Modify Administration Web Server Access Control Entry Form



Home › Web Server Administration › Manage the Administration Web Server ›
Change the Administration Web Server Configuration Parameters ›
Change the Administration Web Server Access Control Entries ›

Modify the Administration Web Server Access Control Entry

Type: Directory Specification: /usr/internet/httpd/admin/htdocs

Host Access

Limit Access Methods: All Methods
Precedence: Process Deny First
Hosts Denied Access: all
Hosts Allowed Access: all

Authentication

User Authentication: For Selected Users
Selected Users: admin
Authentication Prompt Name: Web Server Administration Server Ac

5. In the Hosts Allowed Access field, enter one of the following:
   - *host.domain.name* for a specific host
   - *.domain.name* for a specific domain
   - `all` for any remote host

   For more information on the Allow command, see the Apache documentation at the following Web site:

   http://www.apache.org/
6. Click on Submit.

   The Web Server Administration utility displays a confirmation message.
7. On the confirmation page, click on Submit to reload the Administration Web Server configuration file.

2.10 Dynamically Adding and Removing Server Modules

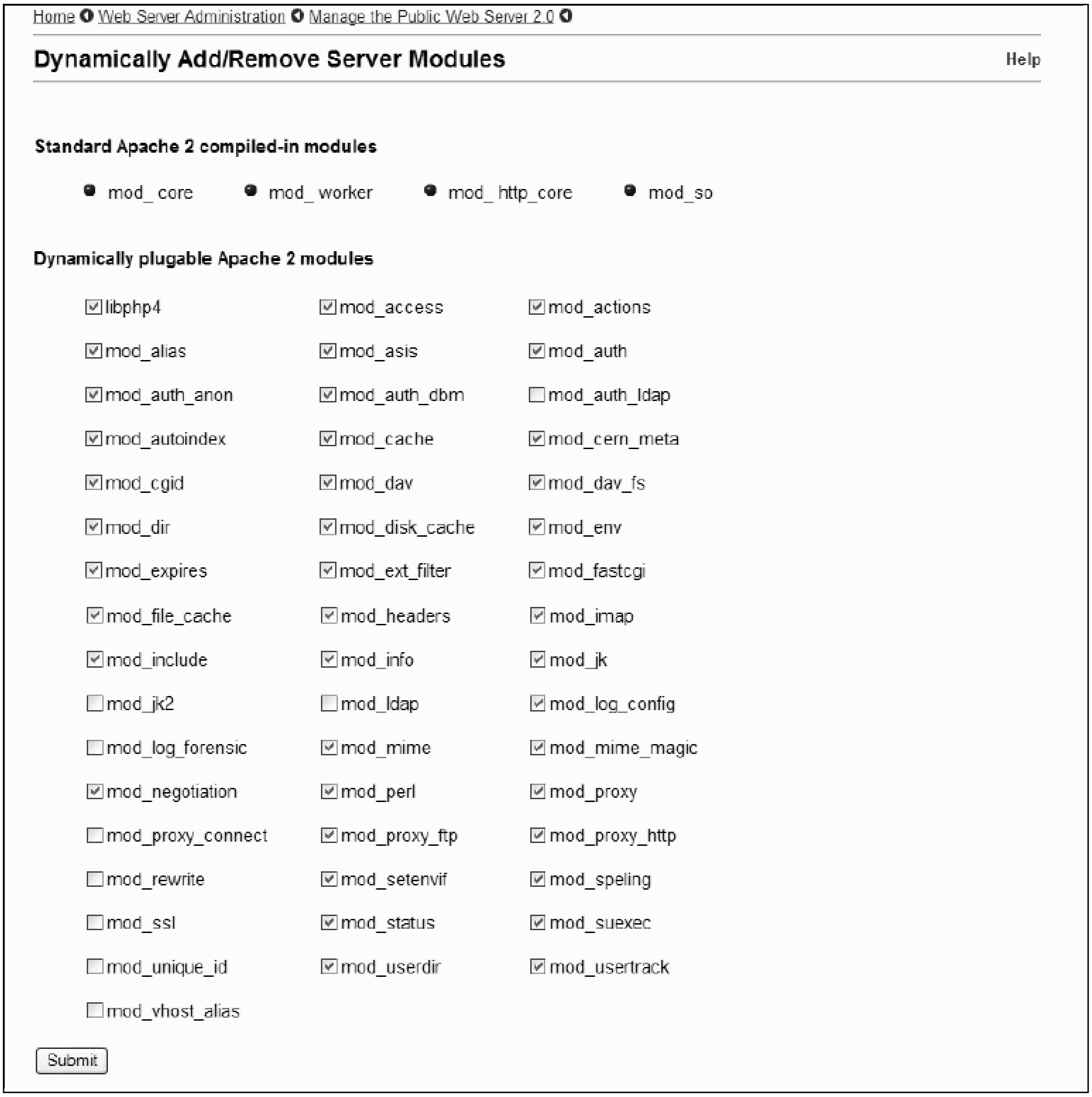
This section describes how to dynamically add or remove server modules. Appendix A lists the standard Apache Version 1.3 modules and Appendix B lists the standard Apache Version 2.0 modules provided with this release.

To dynamically add or remove server modules, follow these steps:

1. From the Secure Web Server Administration menu, choose the server for which you want to refresh the log files; for example, the Administration Web Server. The Manage the Administration Web Server menu is displayed. (See Figure 2-13.)
2. From the Manage the Administration Web Server menu, choose Manage Public Web Server 2.0.
3. On the Manage Public Web Server page, choose Dynamically Add/Remove Server Modules.

   A form (Figure 2-19) is displayed that shows the available server modules.

Figure 2-19 Dynamically Add/Remove Server Module Form



Home ❯ Web Server Administration ❯ Manage the Public Web Server 2.0 ❯

Dynamically Add/Remove Server Modules

Standard Apache 2 compiled-in modules

● mod_ core ● mod_ worker ● mod_ http_core ● mod_so

Dynamically plugable Apache 2 modules

☑ libphp4	☑ mod_access	☑ mod_actions
☑ mod_alias	☑ mod_asis	☑ mod_auth
☑ mod_auth_anon	☑ mod_auth_dbm	☐ mod_auth_ldap
☑ mod_autoindex	☑ mod_cache	☑ mod_cern_meta
☑ mod_cgid	☑ mod_dav	☑ mod_dav_fs
☑ mod_dir	☑ mod_disk_cache	☑ mod_env
☑ mod_expires	☑ mod_ext_filter	☑ mod_fastcgi
☑ mod_file_cache	☑ mod_headers	☑ mod_imap
☑ mod_include	☑ mod_info	☑ mod_jk
☐ mod_jk2	☐ mod_ldap	☑ mod_log_config
☐ mod_log_forensic	☑ mod_mime	☑ mod_mime_magic
☑ mod_negotiation	☑ mod_perl	☑ mod_proxy
☐ mod_proxy_connect	☑ mod_proxy_ftp	☑ mod_proxy_http
☐ mod_rewrite	☑ mod_setenvif	☑ mod_speling
☐ mod_ssl	☑ mod_status	☑ mod_suexec
☐ mod_unique_id	☑ mod_userdir	☑ mod_usertrack
☐ mod_vhost_alias		

4. To add a server module, check in the corresponding check box. To remove a sever module, uncheck the corresponding check box.
5. Click on Submit.

Not all modules can be added or removed. If the add/remove operation reports an error, check the module error log.

3 Using Dynamic Modules

Dynamic modules provide a means of allowing developers and third parties to extend the capabilities of the Secure Web Server. When properly configured in the server configuration file, these modules will be loaded into the Web server at startup time. As the Secure Web Server is powered by Apache, modules conforming to the Apache DSO API can be use with the Secure Web Server.

The Secure Web Server can be customized using dynamic modules provided from Apache and other sources. For a complete list of the Secure Web Server 1.3 modules (DSO and integrated), see Appendix A. See Appendix B for a complete list of the Secure Web Server 2.0 modules.

This chapter provides the following information:

- Secure Web Server support for dynamic modules (Section 3.1: Secure Web Server Support for Apache Dynamic Modules).
- Activating the Apache dynamic modules (Section 3.2: Activating the Apache DSO Modules).

3.1 Secure Web Server Support for Apache Dynamic Modules

Apache dynamic modules can be obtained from many sources. These include:

- Modules that are part of the Apache distribution.
- Modules from the Apache Module Register on the Web: http://modules.apache.org. The Module registry contains a list of modules that are available for use with Apache based servers. These modules derive from a variety of sources, and can often be easily built and used with the Secure Web Server.
- Custom-written modules from various sources. Occasionally, a Web site has special needs that cannot be easily addressed using the existing Secure Web Server functionality of and are not readily available from existing modules. In these cases, a Web site can create custom modules based on the Apache DSO API that extends the functionality of the Secure Web Server.

The Secure Web Server integrates many Apache modules and provides many other modules as dynamic shared objects (DSO). The DSO modules are not integrated because they are not usually part of the default configuration of an Apache server. Section 3.1.1: Standard Modules Provided as DSO Modules lists the standard Apache modules provided as DSO modules. Section 3.1.2: Nonstandard Modules Provided as DSO Modules lists nonstandard DSO modules provided in the Secure Web Server kit. (Nonstandard DSO modules are modules not from the Apache Foundation.)

Administrators can build Apache modules for use with the Secure Web Server. The standard Apache `apxs` utility (`/usr/internet/httpd/bin/apxs` or `/usr/opt/hpapache2/bin/apxs`) is provided with the Secure Web Server to assist you during compilation and installation of the Apache modules.

Instructions for compiling modules using the `apxs` utility are usually included with the source code for that particular module. Instructions can also be found in the standard Apache documentation on the Web: http://www.apache.org.

Standard Apache documentation is also included with the Secure Web Server in the `IAEAPDOCxxx`subset, installed in `/usr/internet/httpd/apdocs`. Standard Apache 2.0 documentation is installed in `/usr/hpapache2/manual`.

3.1.1 Standard Modules Provided as DSO Modules

The Secure Web Server provides several standard Apache modules as DSO modules. These are modules that are included in source form as part of the Apache Version 1.3 source distributions and Apache Version 2.0 source distribution. Although there are many modules that are included in the distributions, many of them are not included in the default server configuration. The optional modules are provided in the form of DSO modules so they can be activated if needed. Table A-2 lists and describes the modules for the Secure Web Server 1.3. Table B-2 lists and describes the modules for the Secure Web Server 2.0.

3.1.2 Nonstandard Modules Provided as DSO Modules

In addition to the standard Apache modules, the Secure Web Server provides DSO modules from sources outside the Apache Foundation.

Table A-3 and Table A-4 provide a list of nonstandard modules for the Secure Web Server 1.3. Table B-3 provides a list of nonstandard modules for the Secure Web Server 2.0.

3.2 Activating the Apache DSO Modules

To activate an Apache DSO module, you must modify the `httpd.conf` server configuration file, as follows:

1. Add a `LoadModule` directive.
2. Add an `AddModule` directive, if the `ClearModule` directive is used.
3. Add additional module configuration-specific directives.
4. Restart the server for the configuration changes to take effect.

Usually, an Apache DSO module will not perform any useful function until the module-specific configuration directives activate the module's functionality. The module-specific documentation explains the module configuration directives. For Apache DSO modules provided with the Secure Web Server, refer to either the Apache documentation provided with the Secure Web Server kit, or the documentation available on the Apache Web site:

http://httpd.apache.org

3.2.1 Using the LoadModule Directive

The `LoadModule` directive enables dynamic module loading by the Web server. This directive must occur before any other configuration directive for the module being loaded.

The `LoadModule` directive has the following form:

```
LoadModule module_identifier filename
```

The `module_identifier` variable is the internal module definition variable name, that is documented in the module documentation.

The `filename` parameter is the loadable module on disk that the server will load. The file name can be an absolute path or a path relative to the server root as defined by the `httpd.conf` file.

If the `ClearModule` directive is present, it signals the server that the list of loaded modules needs to be reordered. The `ClearModule` directive begins the process by clearing the internal list of modules, thus providing the configuration file with a complete list of all modules (integrated and DSO). Each module is readded to the module list with an AddModule directive (Section 3.2.2: Using the AddModule Directive).

3.2.2 Using the AddModule Directive

The `AddModule` directive is used by the Web server to build a list of modules in order of precedence. The directive is only needed if the ClearModule directive is used to zero out the module list. If the ClearModule directive is not used, then the module precedence is in the order that they were loaded.

The `AddModule` directive has the form:

```
AddModule source_file
```

The `source_file` is the file name of the compilation unit that contained the module declaration. To determine the proper file name to use, see the module-specific documentation.

3.2.3 Verifying the Configuration File

After adding the configuration directives to the `httpd.conf` file, the file syntax can be reviewed prior to restarting the server. The following command directs the server to verify the specified configuration file:

For Secure Web Server 1.3:

```
# /usr/internet/httpd/bin/httpd -t -f /usr/internet/httpd/conf/httpd.conf
```

For Secure Web Server 2.0:

```
# /usr/opt/hpapache2/bin/httpd -t -f /usr/opt/hpapache2/httpd.conf
```

3.2.4 Activating an Apache DSO Module — Example

The example in this section shows how to activate the mod_usertrack Apache DSO module, which uses Cookies to track users. To activate the module, use the httpd.conf configuration file for the default public server /usr/internet/httpd/conf/httpd.conf and follow these steps:

1. Define the LoadModule directive as follows:

   ```
   LoadModule usertrack_module   libexec/mod_usertrack.so
   ```

2. If the ClearModule directive is defined, add the module to the module list with an AddModule directive:

   ```
   AddModule mod_usertrack.c
   ```

3. Define the module-specific directives, for example:

   ```
   CookieName OurTestCookie
   CookieTracking on
   ```

4. Track the generated cookies by using a directive for another Apache module (that is, one that is integrated in the Secure Web Server). This directive will cause the cookies to be logged into a file called clickstream, in the logs directory relative to the server root. For example:

   ```
   CustomLog logs/clickstream "%{cookie}n %r %t"
   ```

4 Implementing the Tomcat Java Servlet and JavaServer Pages Container

This chapter contains the following information:

- Overview of Tomcat (Section 4.1: Tomcat Overview)
- Using Tomcat as a standalone public Web server (Section 4.2: Using Tomcat as a Standalone Public Web Server)
- Locating Tomcat directories (Section 4.3: Locating Tomcat Directories)
- Selecting a Java environment (Section 4.4: Selecting a Java Environment)
- Starting Tomcat (Section 4.5: Starting Tomcat)
- Accessing the Tomcat administration and manager Web-based applications (Section 4.6: Accessing the Tomcat Administration and Manager Applications)
- Accessing the Tomcat examples (Section 4.7: Accessing the Tomcat Examples)
- Locating additional information (Section 4.8: Locating Additional Information)

4.1 Tomcat Overview

Tomcat is a Java Servlet and JavaServer Pages (JSP) container provided through the Apache Software Foundation Jakarta project. The Tomcat container is most commonly used with commercial grade Web servers such as Apache. It can also be used as a standalone Web server.

Tomcat is provided with the Secure Web Server as an optional subset (`IAETOMCAT`*`xxx`*, where *`xxx`* is the version number of the Secure Web Server release). When this subset is installed:

- The public instance of the Secure Web Server can be configured to seamlessly pass requests for Java Servlet and JavaServer Pages to the Tomcat engine. With the Secure Web Server handling the direct interaction with the browser, it can provide additional functionality that Tomcat cannot provide by itself, such as IPv6 support and Apache modules used for access control.
- Tomcat can be configured to act as a standalone Web server. For sites with primarily Java-based dynamic content, this option generally improves performance and response time.
- Tomcat may be configured to act simultaneously a standalone Web server and as a request handler from the Secure Web Server.

The Tomcat subset creates a Cluster Application Availability (CAA) resource when it is used in a TruCluster environment. Tomcat is configured as a single instance resource so that it can take advantage of sessions and other capabilities of a Java Servlet environment. The public instances of the (multi-instance) Secure Web Server are configured to communicate with the Tomcat instance and will handle the failover of the Tomcat resource.

For more information on developing and installing servlets and JSPs, see Section 4.8: Locating Additional Information. After installing or updating an application, the Tomcat container and the public Web server may need to be restarted. For information on starting Tomcat, see Section 4.5: Starting Tomcat.

4.2 Using Tomcat as a Standalone Public Web Server

Tomcat may be installed as a standalone public Web server. When you install Tomcat in this manner, the server is configured to respond to HTTP requests on a specified port. Tomcat was previously used exclusively as a backend to an Apache-based Web server. It has now evolved into a full-featured Web server.

A Web server is normally run as a non-privileged user to reduce potential security risks. The Secure Web Server has provisions for switching to the user `httpd` after it has opened its ports. To perform the same actions with Tomcat, you must run a program called `jsvc` whenever the Web server port configured is less than 1024. The `jsvc` program is part of the Tomcat source distribution. This installation will automatically use the `jsvc` program, if needed, when Tomcat is started with the `/sbin/init.d/tomcat` script or by using the Manage the Tomcat Java Servlet and JSP Engine link from the Web Server Administration menu (see Section 4.5.1: Starting and Stopping Tomcat from the Administration Utility).

4.3 Locating Tomcat Directories

After installation, Tomcat resides in the `/usr/internet/httpd/tomcat` directory. Applications (Java Servlets and JSPs) are installed under the `/usr/internet/httpd/tomcat/webapps` directory.

Tomcat-specific configuration information is installed in the `/usr/internet/httpd/tomcat/conf` directory.

4.4 Selecting a Java Environment

In some cases, multiple Java Development Environment kits (JDK) might be installed on a system. You can customize the Java environment used to power the Tomcat container by specifying which Java Development Environment kit (JDK) to use and by adding optional parameters to the `java` command. Customizing the Java environment is also useful when special tuning of the Java Runtime is desired.

Tomcat first uses values specified in the `/usr/internet/httpd/tomcat/bin/setenv.sh` initialization file. If no values are specified in this file, Tomcat uses the Java system default . When you first install the Tomcat subset, it creates the `/usr/internet/httpd/tomcat/bin/setenv.sh` initialization file (if it does not already exist) so that Tomcat will use the newest version of the JDK that is present at installation.

If you want to use a different version of the JDK, change the `JAVA_HOME` and `JAVA_CMD` variables defined in the initialization file. The `JAVA_HOME` and `JAVA_CMD` variables should always reference the same JDK; if they do not, Tomcat will fail to operate properly. The `JAVA_CMD` variable should be quoted to allow for the addition of arguments.

The following example shows a `/usr/internet/httpd/tomcat/bin/setenv.sh` initialization file with additional `java` command arguments:

```
JAVA_HOME=/usr/opt/java/131
JAVA_CMD="$JAVA_HOME/bin/java -classic"
export JAVA_HOME JAVACMD
```

4.5 Starting Tomcat

Section 4.5.1: Starting and Stopping Tomcat from the Administration Utility describes how to start, stop, and restart Tomcat as a standalone server from the Web Server Administration utility.

Starting or restarting Tomcat is different if you are running your Secure Web Server in a TruCluster environment. Section 4.5.2: Restarting Tomcat in a Non-TruCluster Environment describes how to restart Tomcat using the `tomcat` startup script when it is not in a TruCluster environment. Section 4.5.3: Restarting Tomcat in a TruCluster Environment describes how to restart Tomcat using the `caa_start` and `cluster_restart` scripts when running Tomcat in a TruCluster environment.

Section 4.5.4: Tomcat Log Files gives the location of the Tomcat log files.

The Secure Web Server startup script is designed to detect the presence of Tomcat and start the Web server with additional command-line directives that enable the communication with Tomcat and cause the dynamic configuration files generated by Tomcat to be used.

The Tomcat engine creates a number of configuration files each time it is started. These configuration files contain directives that determine the information the Secure Web Server will serve and the requests that should be directed to the Tomcat engine.

Note:

To take full advantage of the changes in Web contexts, the Secure Web Server configuration must be changed to map the new contexts and then the web server must be restarted whenever you change the Tomcat configuration.

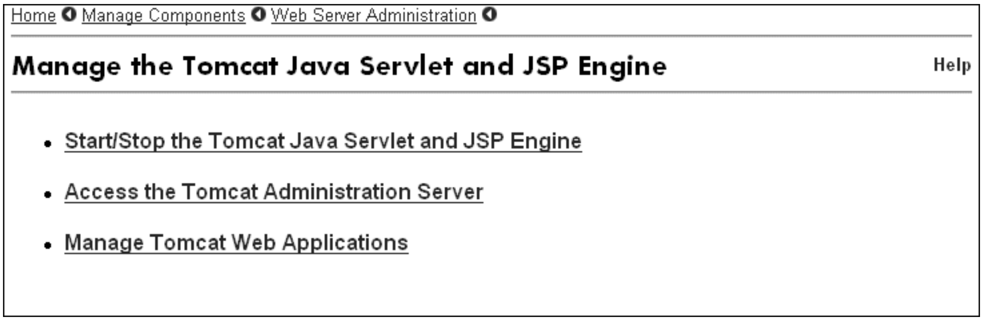
4.5.1 Starting and Stopping Tomcat from the Administration Utility

You can start, stop, and restart the standalone Tomcat server from the Web Server Administration utility.

To start the Tomcat server:

1. On the Web Server Administration menu, choose Manage the Tomcat Java Servlet and JSP Engine. The Manage the Tomcat Java Servlet and JSP Engine form is displayed.
2. On the Manage the Tomcat Java Servlet and JSP Engine form, click on Start/Stop the Manage the Tomcat Java Servlet and JSP Engine (Figure 4-1). Tomcat starts the server and displays a message that the server is running.

Figure 4-1 Manage the Tomcat Java Servlet and JSP Engine Form



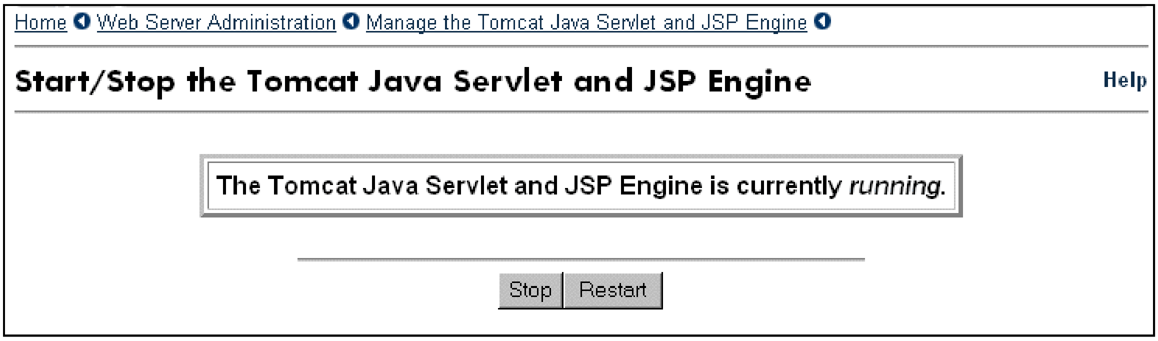
Manage the Tomcat Java Servlet and JSP Engine

- Start/Stop the Tomcat Java Servlet and JSP Engine
- Access the Tomcat Administration Server
- Manage Tomcat Web Applications

To stop and restart the Tomcat server:

1. On the Manage the Tomcat Java Servlet and JSP Engine form, click on Start/Stop the Manage the Tomcat Java Servlet and JSP Engine (Figure 4-1). The Start/Stop the Manage the Tomcat Java Servlet and JSP Engine form is displayed (Figure 4-2).

Figure 4-2 Start/Stop the Tomcat Java Servlet and JSP Engine Form



Home
Web Server Administration
Manage the Tomcat Java Servlet and JSP Engine
Help
Stop
Restart

Start/Stop the Tomcat Java Servlet and JSP Engine

The Tomcat Java Servlet and JSP Engine is currently *running.*

2. Click on stop to stop the server. Tomcat displays a message that the server is stopped.
3. To restart the server, click on restart. Tomcat displays a message that the server is again running.

4.5.2 Restarting Tomcat in a Non-TruCluster Environment

To start Tomcat and the Secure Web Server that is not part of a TruCluster environment:

1. If Tomcat is running, use the `stop` command to stop the server:

   ```
   # /sbin/init.d/tomcat stop
   ```

   And this command to restart the server:

   ```
   # /sbin/init.d/tomcat restart
   ```

2. Start the Tomcat server using the `start` command:

   ```
   # /sbin/init.d/tomcat start
   ```

3. Restart the public instance of the Secure Web Server using the `restart` command:

   ```
   # /sbin/init.d/httpd_public restart
   ```

4.5.3 Restarting Tomcat in a TruCluster Environment

To restart Tomcat and the Secure Web Server in a TruCluster environment:

1. If Tomcat is running, use the `caa_stop` command to stop the server:

   ```
   # caa_stop tomcat
   ```

   And this command to restart the server:

   ```
   # /sbin/init.d/tomcat restart
   ```

2. Start the Tomcat Server using the `caa_start` command:

   ```
   # caa_start tomcat
   ```

3. Restart (on all cluster members) the public instance of the Secure Web Server using the `cluster-restart` command:

   ```
   # /sbin/init.d/httpd_public cluster-restart
   ```

4.5.4 Tomcat Log Files

Tomcat records log files in the `/usr/internet/httpd/logs` directory. Table 4-1 lists the log files Tomcat creates and provides a description of each file.

Table 4-1 Tomcat Log Files

Log File	Description
`tomcat_startup.log`	Tomcat startup log
`catalina.out`	Tomcat general log
`access_log`	Access requests log, shared with the Secure Web Server
`error_log`	Failed access requests log, shared with the Secure Web Server
`tomcat.pid`	The process ID of the Tomcat process.

4.6 Accessing the Tomcat Administration and Manager Applications

Tomcat provides Web-based applications for administering the Tomcat deployment and for managing the lifecycle of Web applications running within the Tomcat container. To access the Tomcat Web-based administration features:

1. On the Web Server Administration menu, choose Manage the Tomcat Java Servlet and JSP Engine. The Manage the Tomcat Java Servlet and JSP Engine form is displayed. This menu allows you to start or stop the Tomcat Java Servelet and JSP Engine (see Section 4.5.1) and perform the following Web-based administration functions:
   - Access the Tomcat administration server – An Open Source Web-based server for administering the Tomcat Server, server resources (for example, data sources, mail sessions), and server user definitions (i.e., users, groups, and roles).
   - Manage Tomcat Web applications – An Open Source management tool for managing Tomcat-based Web applications.
2. To access the Tomcat administration server, click on Access the Tomcat Administration Server. For Internet Express users, the administrator username and password is identical to your Internet Express username (`admin`) and password.

   If the Internet Express or Secure Web Server Administration Server is not installed, you must manually create the Tomcat Administrator username and password. Refer to the Configuring Manager Application Access section on the following URL for instructions on how to create a Tomcat Administrator username and password:

   `http://`*`Tomcat standalone Web server name`*`:8082/tomcat/tomcat-docs/manager-howto.html`

3. To manage Tomcat Web Applications, click on Manage Tomcat Web Applications.

By default, access to these management applications is limited to browsers running on the local host and requires that users successfully authenticate before being granted access. The local host restriction is established by access control valves in the `/usr/internet/httpd/tomcat/conf/server.xml` file.

To modify this restriction:

- Edit the `server.xml` file to change the list of allowed hosts, or
- Delete the Valve element entirely to remove host-based restrictions.
- Restart Tomcat for the changes to take effect. See the specific instructions in Section 4.5.1: Starting and Stopping Tomcat from the Administration Utility for restarting Tomcat in a non-cluster or cluster environment.

Note:

The default restriction requires that a browser on the local host must access the management applications using URLs that begin with `http://localhost/`. If you attempt to access the applications with URLs that begin with `http://`*`actual_hostname_of_local_host`*`/`, it will be rejected.

User authentication is provided by a custom realm that allows a user who successfully authenticates as the Secure Web Server administration user to be mapped to the Tomcat user roles `admin` and `manager`. These roles are required to access the administration and Web application management utilities. If this initial authentication attempt fails, the realm then attempts to authenticate via Tomcat's default user authentication database, defined by the `/usr/internet/httpd/tomcat/conf/tomcat-users.xml`file. To change the behavior of this custom realm, modify the `/usr/internet/httpd/tomcat/conf/server.xml` file as necessary and then restart Tomcat.

4.7 Accessing the Tomcat Examples

The Tomcat subset includes Web applications for the Tomcat documentation and example Java Servlets and JavaServer Pages. The Web applications are installed in the `/usr/internet/httpd/tomcat/samples` directory . The Web applications are deployed by default and are accessed by specifying the name of your host domain, as follows:

`http://`*`yourhost.domain`*`/tomcat/`

To prevent the Web applications from being deployed, perform the following task:

Edit the `/usr/internet/httpd/tomcat/conf/server.xml` file and comment out or remove the following entries:

```
<Context path="/tomcat"
 docBase="/usr/internet/httpd/tomcat/samples/ROOT.war"
 reloadable="false" debug="0" />

<Context path="/tomcat/examples"
 docBase="/usr/internet/httpd/tomcat/samples/examples.war"
 reloadable="false" debug="0" />

<Context path="/tomcat/docs"
docBase="/usr/internet/httpd/tomcat/samples/tomcat-docs.war"
reloadable="false" debug="0" />

<Context path="/tomcat/webdav"
docBase="/usr/internet/httpd/tomcat/samples/webdav.war"
reloadable="false" debug="0" />
```

4.8 Locating Additional Information

Additional information about Tomcat can be found in the following locations:

- The Jakarta Project Web site: http://jakarta.apache.org
- Tomcat documentation Web site: http://jakarta.apache.org/tomcat/tomcat-5.0-doc/index.html

5 Enabling the Secure Socket Layer Protocol

To enhance the security of communications between your Web browser and administrative instances of the Secure Web Server, the Secure Web Servers have built-in support for the Secure Socket Layer (SSL) protocol. This chapter describes how SSL provides secure Internet connections and how to use Internet Express to enable SSL on your server.

5.1 SSL Concepts

The SSL protocol is a widely used method for performing secure transactions on the Web. This protocol is supported by most Web servers and clients including Netscape Navigator and Microsoft Internet Explorer.

SSL provides privacy, guaranteed through encryption. Although information can be intercepted by a third party, the perpetrator cannot read the information without the private encryption key (session key). If the information is received and will not decrypt properly, the recipient can determine that the information has been tampered with during transmission. Authentication is provided through digital certificates generated for SSL, though the source of digital certificates might not always be credible for online payment transactions.

SSL encryption uses a secret key nested within public key encryption, authenticated through certificates. Secret key encryption provides faster access than public-key encryption alone. Initially, the client and server exchange public keys, and then the client generates a session key for a specific transaction. The client encrypts the session key with the server's public key and sends the information to the server. Then, and for the remainder of the transaction, the client and the server use the session key for private key encryption.

Completing a transaction with an SSL-enabled server follows this general procedure:

1. A client sends a request for a document to be transmitted using the https: protocol by prefixing the URL with `https://`.
2. The server sends the client its certificate.
3. The client verifies that the certificate was issued by a trusted Certificate Authority (CA). If the client does not verify the CA, it gives the user the option to continue or to terminate the transaction.
4. The client compares information in the certificate with information received concerning the site; specifically, the domain name and the public key. If the information matches, the client accepts the site as authentic.
5. The client tells the server what ciphers (encryption algorithms) it uses to communicate.
6. The client generates a session key using the agreed-upon cipher.
7. The client encrypts the session key with the server's public key and sends the information to the server.
8. The server receives the encrypted session key and decrypts the information with the session key.
9. The client and the server then use the session key for the remainder of the transaction.

For additional information about SSL, see the `mod-ssl` Web site:

http://www.modssl.org/docs

5.2 Enabling SSL Support from the Web Server Administration Utility

Using the Web Server Administration utility, you can manage support for SSL connections. Follow these steps:

1. On the Secure Web Server Administration menu, select the Web server for which you want to enable (or disable) SSL support, for example, the public Web server. The Manage the Public Web Server menu is displayed (see Figure 2-11).
2. Choose Manage SSL for the Public Web Server. The SSL menu options are displayed (Figure 5-1: Manage SSL for the Public Web Server Menu), initially showing the following options:
   - Generate a private key (Section 5.3: Generating a Private Key)
   - Generate a certificate request (Section 5.4: Generating a Certificate Request)

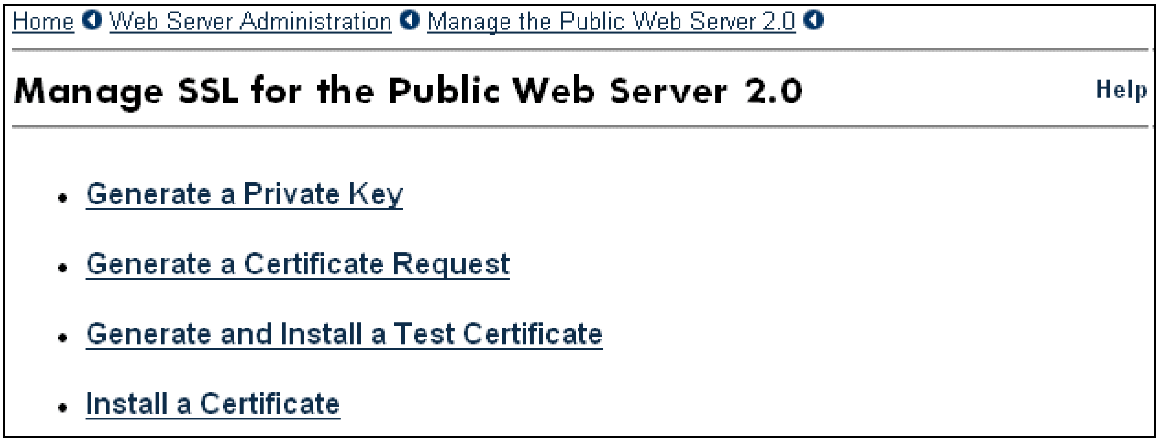
- Generate and install a test certificate (Section 5.5: Generating and Installing a Test Certificate)
- Install a certificate (Section 5.6: Installing a Digital Certificate)

3. Proceed to generate a private key (Section 5.3: Generating a Private Key) and request a digital certificate (Section 5.4: Generating a Certificate Request).

Note:

The steps for managing SSL that are described in this chapter use public Web server examples. Steps for managing SSL for the Administration Web Server are identical.

Figure 5-1 Manage SSL for the Public Web Server Menu



Home ◀ Web Server Administration ◀ Manage the Public Web Server 2.0 ◀

Manage SSL for the Public Web Server 2.0

- Generate a Private Key
- Generate a Certificate Request
- Generate and Install a Test Certificate
- Install a Certificate

When you enable SSL for the first time, you must generate a private key and then generate a certificate request. A Certificate Authority (CA), such as VeriSign (http://www.verisign.com), processes the request and provides you with an official digital certificate. While waiting for your official digital certificate, you can generate and install a test certificate. These steps are described in "Generating a Private Key" through Section 5.5: Generating and Installing a Test Certificate.

For information on setting up an Apache Web server with SSL without using the Secure Web Server Administration utility, visit the Apache Web site at the following URL:

`http://www.apache.org/`

5.3 Generating a Private Key

SSL uses on asymmetric key encryption to encode and decode data that is transmitted to and from the Web server. SSL key encryption requires two keys: a private key and a public key. The private key resides on the Web server system and must be kept secure. Before you can perform other steps to set up an SSL connection, you must generate a private key.

To generate a private key, perform these steps:

1. From the server administration menu, choose Manage SSL for the desired server. For example, from the Manage the Public Web Server menu, choose Manage SSL for the Public Web Server. The Manage SSL for the Public Web Server menu is displayed.
2. Choose Generate a Private Key. The Generate a Private Key menu is displayed, informing you whether a private key already exists.
3. Click Submit to generate a private key. When you generate a private key, the key is saved to following file for each server:

   `/usr/internet/httpd/`*`server`*`/conf/ssl.key/server.key`

   Where *`server`* is the name of the server you are modifying for SSL, as follows:

   - Public Web servers:

     `/usr/internet/httpd/conf/ssl.key/server.key`

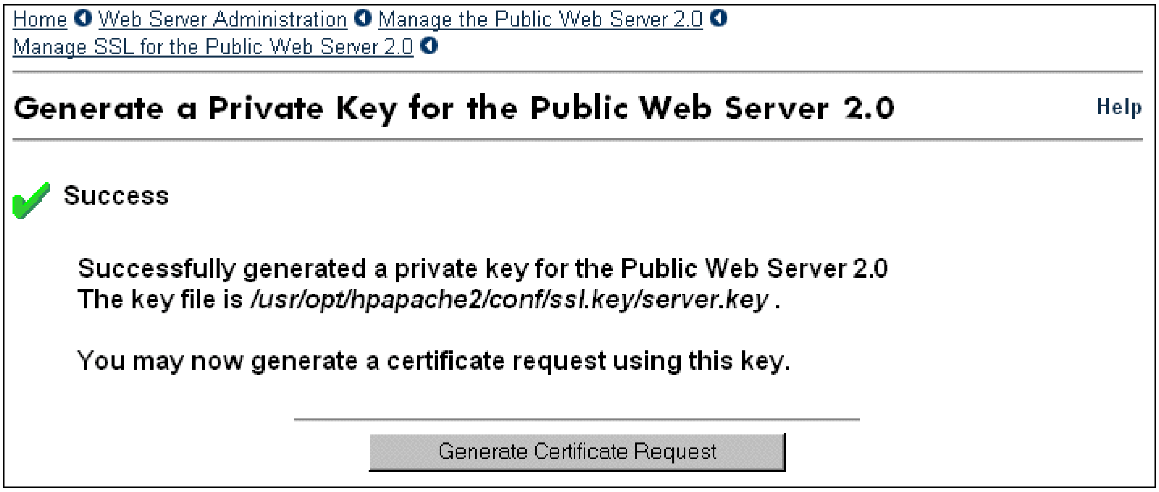
   - Administration Web Server:

     `/usr/internet/httpd/admin/conf/ssl.key/server.key`

If a private key already exists, the existing key is saved in a separate `server.n.key` file where *n* is an integer incrementing from 1.

Figure 5-2 shows that a private key has been generated for the public Web server and the location of the `server.key` file. Note that you can generate a certificate request directly from this page, from which you can display the Generate a Certificate Request form. See Section 5.4 for complete steps for generating a certificate request.

Figure 5-2 Generate a Private Key — Results



Home
Web Server Administration
Manage the Public Web Server 2.0
Manage SSL for the Public Web Server 2.0
Help

Generate a Private Key for the Public Web Server 2.0

✓ Success

Successfully generated a private key for the Public Web Server 2.0
The key file is */usr/opt/hpapache2/conf/ssl.key/server.key* .

You may now generate a certificate request using this key.

Generate Certificate Request

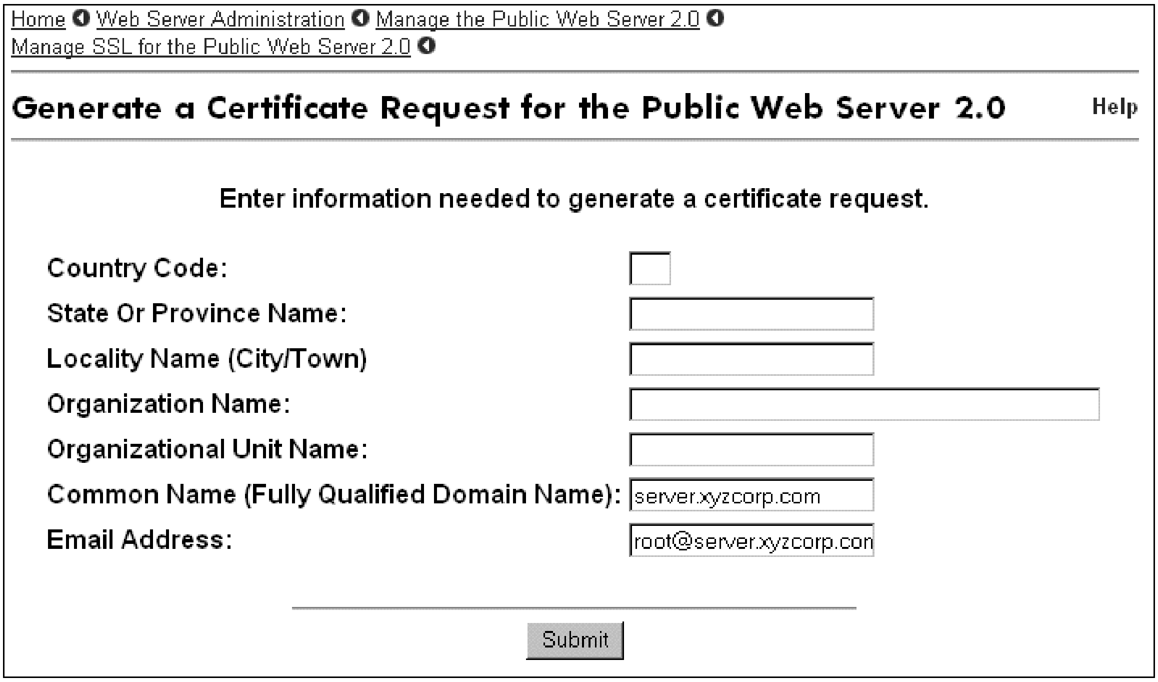
5.4 Generating a Certificate Request

After you have generated a private key (Section 5.3), you can generate a certificate request that provides information about your company and private key to a Certificate Authority (CA). From this request, an X.509 certificate signing request (CSR) is created.

To generate a certificate request, perform these steps:

1. From the server administration menu, choose Manage SSL for the desired server. For example, from the Manage the Public Web Server menu, choose Manage SSL for the Public Web Server. The Manage SSL for the Public Web Server menu is displayed.
2. Choose Generate a Certificate Request. The Generate a Certificate Request form is displayed (Figure 5-3).

Figure 5-3 Generate a Certificate Request Form



Home ◀ Web Server Administration ◀ Manage the Public Web Server 2.0 ◀
Manage SSL for the Public Web Server 2.0 ◀

Generate a Certificate Request for the Public Web Server 2.0 Help

Enter information needed to generate a certificate request.

Country Code:	
State Or Province Name:	
Locality Name (City/Town)	
Organization Name:	
Organizational Unit Name:	
Common Name (Fully Qualified Domain Name):	server.xyzcorp.com
Email Address:	root@server.xyzcorp.con

3. Enter your company data in the text fields.

   A two-character country code is required by your CA. For the country code, enter an official ISO standard two-character country code as defined in *ISO Standard 3166-1*:

   `http://www.iso.org/iso/en/ISOOnline.frontpage`

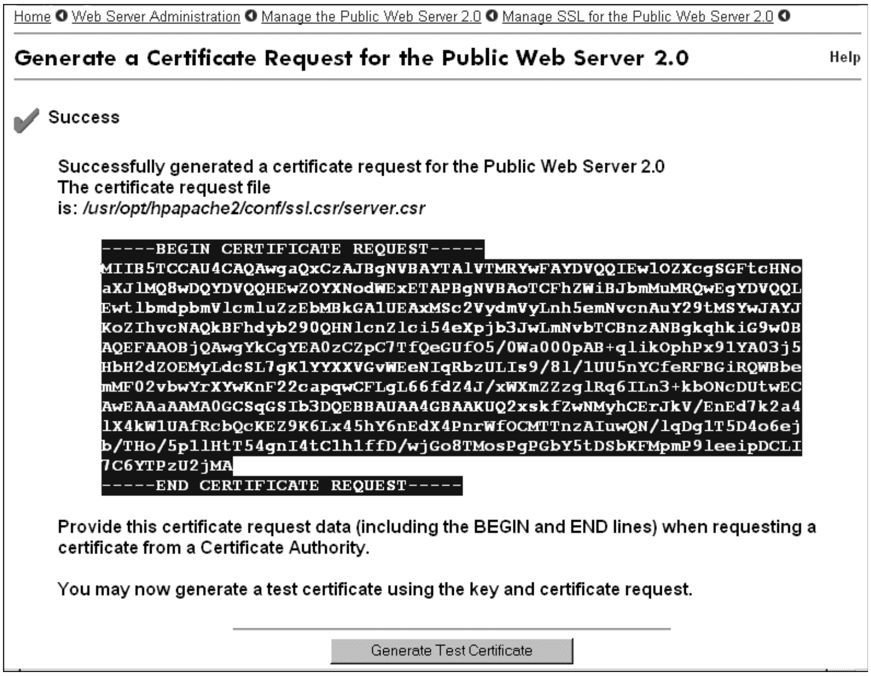
   For the Common Name field, use the fully qualified domain name of your server (for example, `www.server.wyxcorp.com`).

4. Click on Submit. From the information you provided, an X.509 certificate signing request (CSR) is created. The certificate request displays in your browser window and is saved in the `/usr/internet/httpd/`*`server`*`/conf/ssl.csr/server.csr` file, where *`server`* is the name of the server you are modifying. For the Web Server Public Instance, the certificate request file is saved in `/usr/internet/httpd/conf/ssl.csr/server.csr`.

5. To complete the certificate request process, copy information from the browser window or copy the contents from the CSR file and send the information (along with required paperwork and payment) to a Certificate Authority (CA) such as VeriSign (http://www.versign.com). The highlighted text in Figure 5-4 shows the information that you send to your CA.

   Note that you can generate a test certificate directly from this page. See Section 5.5 for complete steps for generating a test certificate.

Figure 5-4 Certificate Request Success Notice



Home ◐ Web Server Administration ◐ Manage the Public Web Server 2.0 ◐ Manage SSL for the Public Web Server 2.0 ◐

Generate a Certificate Request for the Public Web Server 2.0

✓ Success

Successfully generated a certificate request for the Public Web Server 2.0
The certificate request file
is: */usr/opt/hpapache2/conf/ssl.csr/server.csr*

```
-----BEGIN CERTIFICATE REQUEST-----
MIIB5TCCAU4CAQAwgaQxCzAJBgNVBAYTAlVTMRYwFAYDVQQIEwlOZXcgSGFtcHNo
aXJlMQ8wDQYDVQQHEwZOYXNodWExETAPBgNVBAoTCFhZWiBJbmMuMRQwEgYDVQQL
EwtlbmdpbmVlcmluZzEbMBkGA1UEAxMSc2VydmVyLnh5emNvcnAuY29tMSYwJAYJ
KoZIhvcNAQkBFhdyb290QHNlcnZlci54eXpjb3JwLmNvbTCBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEA0zCZpC7TfQeGUfO5/0Wa000pAB+qlikOphPx91YA03j5
HbH2dZOEMyLdcSL7gKlYYXXVGvWEeNIqRbzULIs9/8l/lUU5nYCfeRFBGiRQWBbe
mMF02vbwYrXYwKnF22capqwCFLgL66fdZ4J/xWXmZZzglRq6ILn3+kbONcDUtwEC
AwEAAaAAMA0GCSqGSIb3DQEBBAUAA4GBAAKUQ2xskfZwNMyhCErJkV/EnEd7k2a4
lX4kW1UAfRcbQcKEZ9K6Lx45hY6nEdX4PnrWfOCMTTnzAIuwQN/lqDglT5D4o6ej
b/THo/5pllHtT54gnI4tClhlffD/wjGo8TMosPgPGbY5tDSbKFMpmP9leeipDCLI
7C6YTPzU2jMA
-----END CERTIFICATE REQUEST-----
```

Provide this certificate request data (including the BEGIN and END lines) when requesting a certificate from a Certificate Authority.

You may now generate a test certificate using the key and certificate request.

5.5 Generating and Installing a Test Certificate

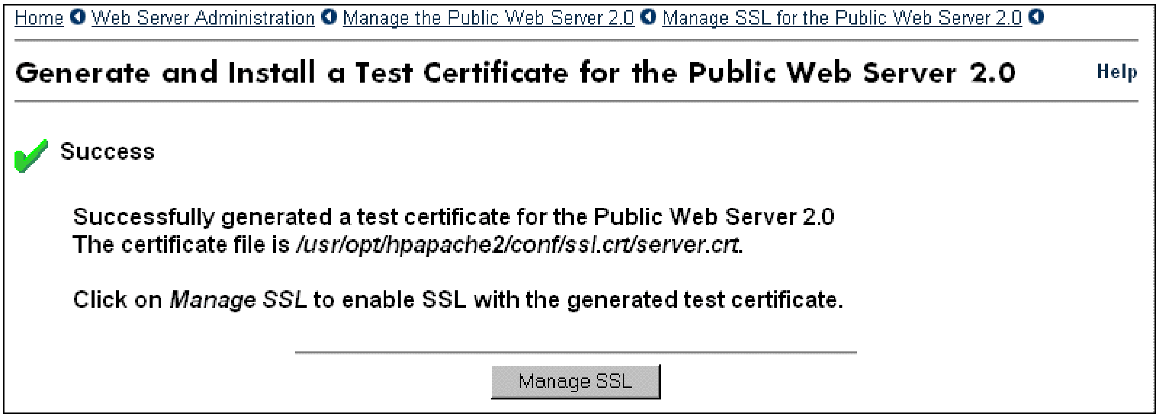
Before you receive your official certificate, you can generate a self-signed certificate and test establishing secure connections from your server.

To generate and install a test certificate, perform these steps:

1. From the server administration menu, choose Manage SSL for the desired server. For example, from the Manage the Public Web Server menu, choose Manage SSL for the Public Web Server. The Manage SSL for the Public Web Server menu is displayed.
2. Choose Generate and Install a Test Certificate. The Generate and Install a Test Certificate form is displayed.
3. Click on Submit. The test certificate is saved in the `/usr/internet/httpd/`*`server`*`/conf/ssl.crt/server.crt` file, where *`server`* is the name of the server you are modifying. When you generate and install an official certificate, it is saved in the same file.

   For example, for the public Web server, the test certificate or official certificate is stored in `/usr/internet/httpd/conf/ssl.crt/server.crt`. If you had a previous certificate, the new certificate overwrites the `server.crt` file. However, existing certificates are saved under a new name. Figure 5-5 shows the message displayed when successfully installing the test certificate.

Figure 5-5 Test Certificate Installation Success Notice



Home ◐ Web Server Administration ◐ Manage the Public Web Server 2.0 ◐ Manage SSL for the Public Web Server 2.0 ◐

Generate and Install a Test Certificate for the Public Web Server 2.0

✓ Success

Successfully generated a test certificate for the Public Web Server 2.0
The certificate file is */usr/opt/hpapache2/conf/ssl.crt/server.crt*.

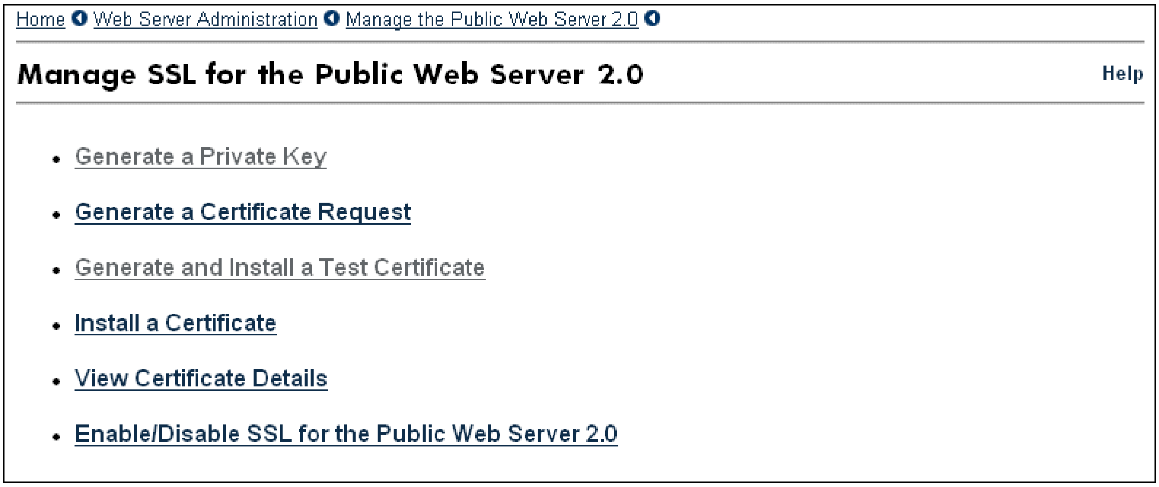
Click on *Manage SSL* to enable SSL with the generated test certificate.

With a test certificate in place, you can now try enabling SSL on the server.

4. On the Generate and Install Test Certificate form (Figure 5-5), click on the Manage SSL button to enable SSL using the installed test certificate. The Manage SSL for the Public Web Server form will be displayed as shown in Figure 5-9. See Section 5.8 for instructions on enabling and disabling SSL for a Web server using this form.

When you generate a test certificate, it is automatically installed. The Manage SSL main menu changes to show two additional options, as shown in Figure 5-6.

Figure 5-6 Manage SSL Main Menu with Additional Options



Home ◂ Web Server Administration ◂ Manage the Public Web Server 2.0 ◂

Manage SSL for the Public Web Server 2.0

- Generate a Private Key
- Generate a Certificate Request
- Generate and Install a Test Certificate
- Install a Certificate
- View Certificate Details
- Enable/Disable SSL for the Public Web Server 2.0

With a test certificate in place, you can now try connecting to the SSL-enabled system. Note that when you make an SSL connection to a server using a self-signed test certificate, you are warned that the certificate is signed by an untrusted source. The system gives you the option to accept the certificate and connect to the Web server.

To view the contents of the certificate, see Section 5.7: Viewing Certificate Details.

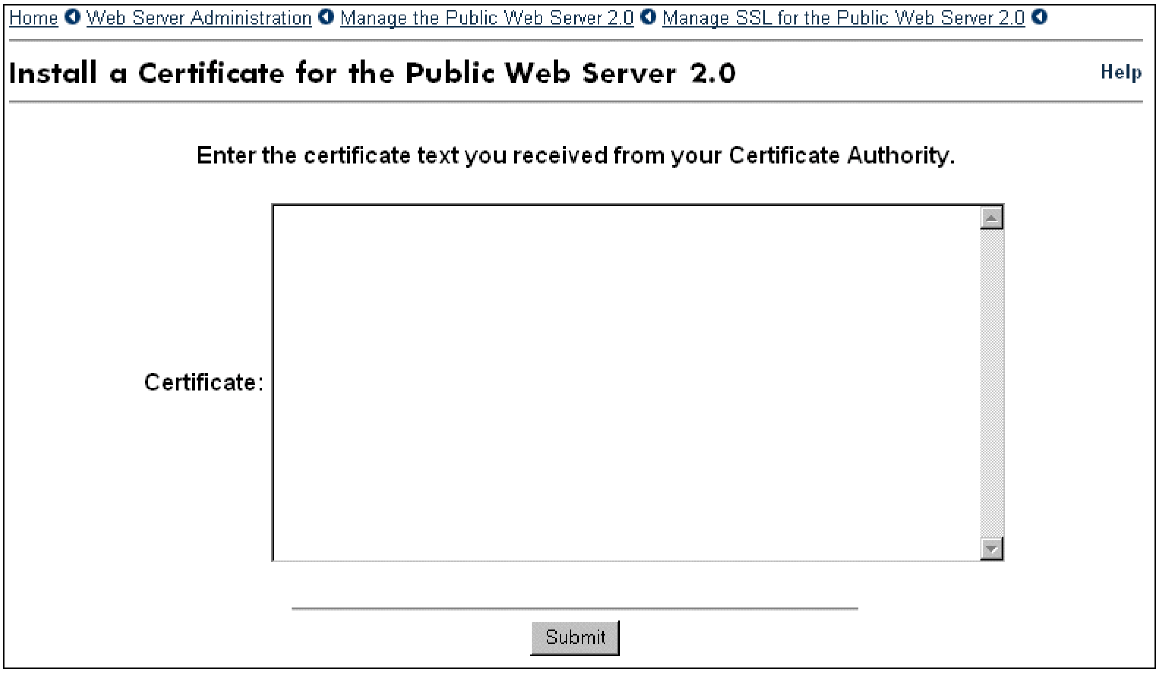
5.6 Installing a Digital Certificate

When you receive a digital certificate from your Certificate Authority, you must then install it to the proper location.

To install a certificate, perform these steps:

1. Determine that the certificate you received is compatible with the private key you created in Section 5.3: Generating a Private Key. The key and certificate must be compatible for the certificate to install properly.
2. From the server administration menu, choose Manage SSL for the desired server. For example, from the Manage the Public Web Server menu, choose Manage SSL for the Public Web Server. The Manage SSL for the Public Web Server menu is displayed.
3. Choose Install a Certificate. The Install a Certificate form is displayed.
4. Cut and paste the contents from the official certificate into the Install a Certificate text field, shown in Figure 5-7.

Figure 5-7 Install a Certificate Text Field



Home ◀ Web Server Administration ◀ Manage the Public Web Server 2.0 ◀ Manage SSL for the Public Web Server 2.0 ◀

Install a Certificate for the Public Web Server 2.0

Enter the certificate text you received from your Certificate Authority.

Certificate:

5. Click on Submit. The certificate file is copied to the `/usr/internet/httpd/`*`server`*`/conf/ssl.crt/server.crt` file, where *`server`* is the name of the system you are modifying.

   If you generated and installed a test certificate (Section 5.5: Generating and Installing a Test Certificate), the test certificate file (`server.crt`) is overwritten with the official certificate file and is saved under a new name (for example, `server.2.crt`).

After successfully installing the certificate, the Manage SSL main menu provides two additional options that let you:

- View certificate details (Section 5.7: Viewing Certificate Details)
- Enable or disable SSL capabilities (Section 5.8: Enabling and Disabling SSL for a Web Server)

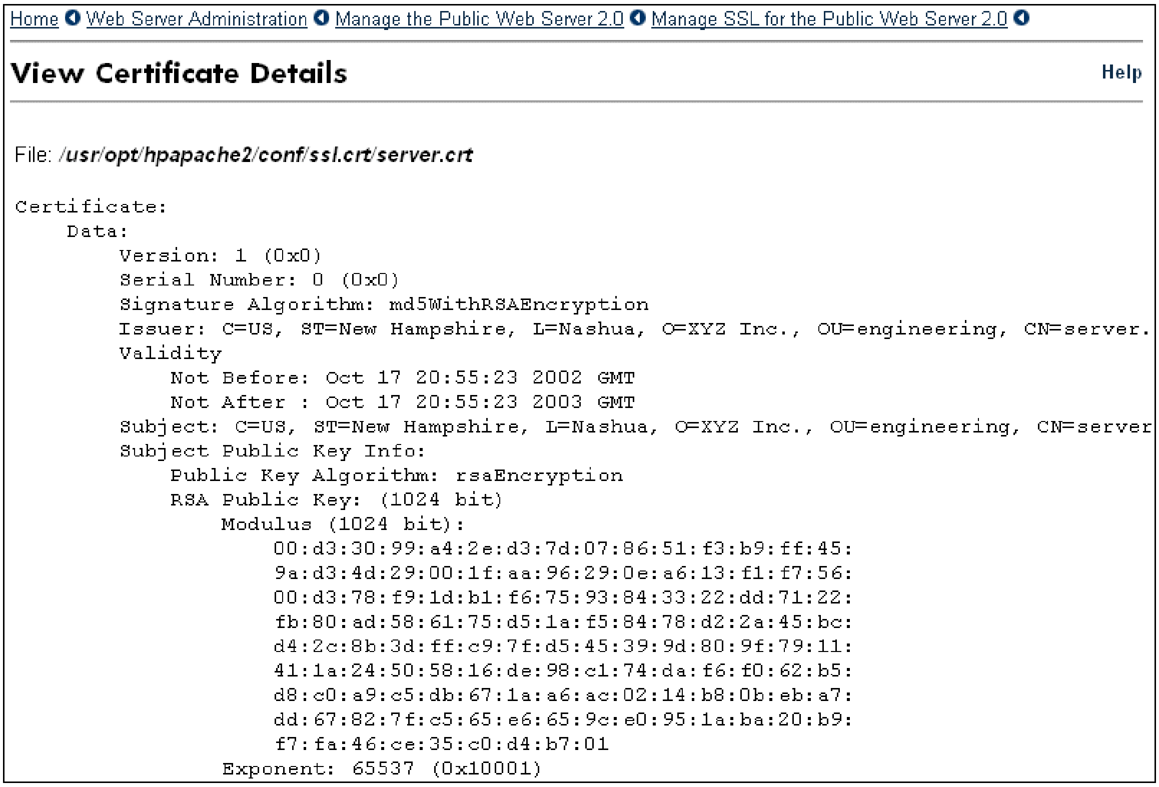
These options also appear on the Manage SSL main menu when you generate and install a test certificate (Figure 5-6).

5.7 Viewing Certificate Details

The View Certificate Details option enables you to display certificate information in a readable format. This certificate file includes information you provided when you requested the certificate (Section 5.4: Generating a Certificate Request), information from the CA, and information about your public key.

1. From the server administration menu, choose Manage SSL for the desired server. For example, from the Manage the Public Web Server menu, choose Manage SSL for the Public Web Server. The Manage SSL for the Public Web Server menu is displayed.
2. Choose View Certificate Details. The certificate stored in `/usr/internet/httpd/`*`server`*`/conf/ssl.crt/server.crt` is displayed in readable format. Figure 5-8 shows the information in an example certificate.

Figure 5-8 Certificate File in Readable Format


Home
Web Server Administration
Manage the Public Web Server 2.0
Manage SSL for the Public Web Server 2.0
View Certificate Details
Help

File: */usr/opt/hpapache2/conf/ssl.crt/server.crt*

```
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number: 0 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=US, ST=New Hampshire, L=Nashua, O=XYZ Inc., OU=engineering, CN=server.
        Validity
            Not Before: Oct 17 20:55:23 2002 GMT
            Not After : Oct 17 20:55:23 2003 GMT
        Subject: C=US, ST=New Hampshire, L=Nashua, O=XYZ Inc., OU=engineering, CN=server
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:d3:30:99:a4:2e:d3:7d:07:86:51:f3:b9:ff:45:
                    9a:d3:4d:29:00:1f:aa:96:29:0e:a6:13:f1:f7:56:
                    00:d3:78:f9:1d:b1:f6:75:93:84:33:22:dd:71:22:
                    fb:80:ad:58:61:75:d5:1a:f5:84:78:d2:2a:45:bc:
                    d4:2c:8b:3d:ff:c9:7f:d5:45:39:9d:80:9f:79:11:
                    41:1a:24:50:58:16:de:98:c1:74:da:f6:f0:62:b5:
                    d8:c0:a9:c5:db:67:1a:a6:ac:02:14:b8:0b:eb:a7:
                    dd:67:82:7f:c5:65:e6:65:9c:e0:95:1a:ba:20:b9:
                    f7:fa:46:ce:35:c0:d4:b7:01
                Exponent: 65537 (0x10001)
```

5.8 Enabling and Disabling SSL for a Web Server

After obtaining a private key and an official certificate, you can enable (or disable) SSL capabilities for your Web server. When you enable SSL, the Web server's runtime configuration file (`/usr/internet/httpd/server/conf/.httpdrc`) is revised to instruct the Web server to use SSL when it restarts. Enabling SSL affects the public and administration servers as follows:

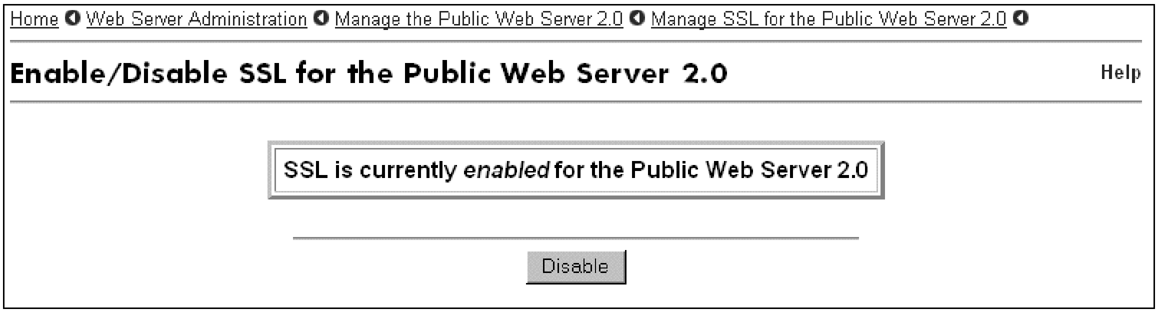
- For the public Web server instance, you can connect to the secure Port 443 using the `https:` prefix. Note that you also can continue connecting to Port 80 using a non-secure connection using the `http:` prefix.
- For the Administration Web Server instance, the port defined for each server is changed. You can only connect as a secure port using the `https:` prefix and the same port number. Once you connect to one of the administration servers, the Administration utility manages the connection to the other servers, using the proper type of connection.

To enable (or disable) SSL capabilities for a Web server:

1. From the server administration menu, choose Manage SSL for the desired server. For example, from the Manage the Public Web Server menu, choose Manage SSL for the Public Web Server. The Manage SSL for the Public Web Server menu is displayed.
2. Choose the Enable button or the Disable button at the bottom of the form to enable or disable SSL for the server, respectively.

   Figure 5-9 shows the confirmation page after SSL has been enabled on the server.

Figure 5-9 Enable SSL Confirmation Page



Home ◀ Web Server Administration ◀ Manage the Public Web Server 2.0 ◀ Manage SSL for the Public Web Server 2.0 ◀

Enable/Disable SSL for the Public Web Server 2.0

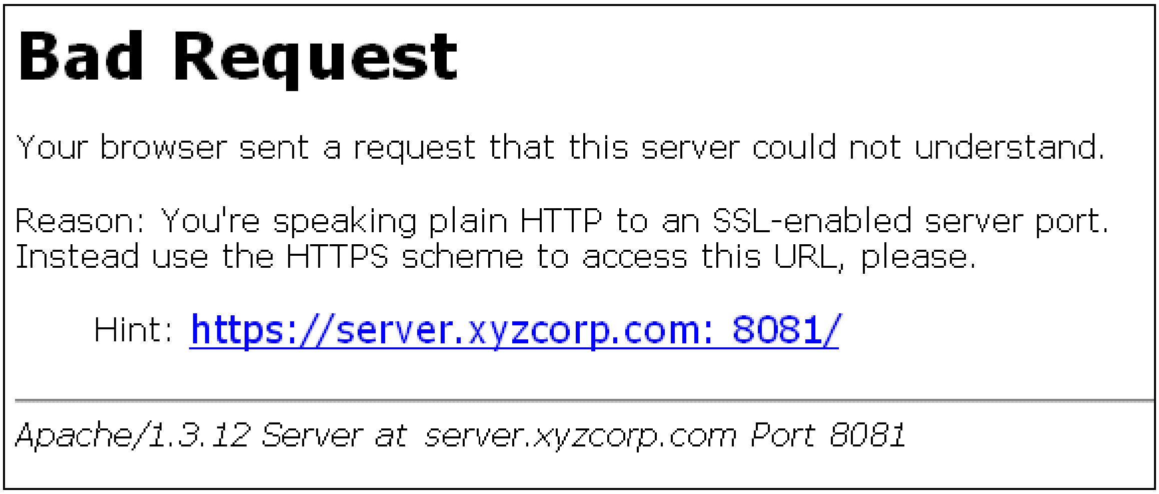
SSL is currently *enabled* for the Public Web Server 2.0

5.9 Testing Your SSL Connection

Test your secure connection after enabling SSL for public and administration servers, as follows:

- For a public Web server, the standard `https:` port is 443. When you specify a URL as `https://www.xxx.com/`, for example, it automatically uses Port 443. Also test access to the nonsecure public server Port 80 by using `httpd:` after enabling SSL. (See Section 5.10 for additional considerations when enabling SSL for public Web servers.)
- For an administration server, the same port is used for all connections, regardless of whether SSL is enabled. With SSL enabled, you can only access this server using `https://www.xxx.com:8081/`. Using `http:` will produce an error message asking you to specify `https:` (Figure 5-10). The administration server menus determine which protocol to use, `http:` or `https:`, and will advise you when you first connect.

Figure 5-10 SSL Connections Error Message



Bad Request

Your browser sent a request that this server could not understand.

Reason: You're speaking plain HTTP to an SSL-enabled server port.
Instead use the HTTPS scheme to access this URL, please.

Hint: https://server.xyzcorp.com: 8081/

Apache/1.3.12 Server at server.xyzcorp.com Port 8081

Note:

After enabling SSL and changing a connection from nonsecure to secure, you might not be able to use the Back button of your browser to navigate to pages viewed prior to enabling SSL. Similarly, disabling SSL and changing a connection from secure to nonsecure might affect use of the Back button. This happens because the saved prefix might no longer be valid.

5.10 Specifying Public Web Server Access to HTTP and HTTPS Connections

After enabling SSL for the Web Server Public Instance, the data hierarchy you created (by default, `/usr/internet/httpd/htdocs`) will be accessible either using the standard `http:` protocol or the SSL-enabled `https:` protocol.

To limit access just to `https:` connections, perform these steps:

1. From the Secure Web Server Administration menu, choose Manage the Public Web Server.
2. Choose Change Configuration Parameters.
3. Choose Change Listening Ports and Addresses and remove Port 80 from the list of active ports and make Port 443 the primary port.
4. Click Submit to update the configuration file and restart the public Web server.

If you want the public Web server to respond to both `http:` requests on Port 80 and `https:` requests on Port 443 while maintaining separate data hierarchies, you must manually change the public Web server configuration file `/usr/internet/httpd/conf/httpd.conf`. Any `https:` directories must be defined within the SSL VirtualHost directive. (In the configuration file, search for the line `<VirtualHost _default_:443>`.)

Directory, Location, or File directives placed within the SSL VirtualHost directive, as well as Alias and ScriptAlias directives placed within the SSL VirtualHost directive, can only be accessed when SSL is enabled and when `https:` connections are used. By changing the value of the DocumentRoot directive within the SSL VirtualHost directive, you can specify a default location specific to `https:` connections.

5.11 Migrating Your Netscape Digital Certificate to the Secure Web Server

This section describes how to migrate a Netscape Web Server digital certificate to the Secure Web Server, which will then allow you to migrate Netscape (iPlanet) Web Server SSL users to an SSL-enabled Secure Web Server.

5.11.1 Prerequisites for Migration

Before you can migrate your Netscape digital certificate to the Secure Web Server, you must first access the Netscape Web Server's private key. You use this key as the Secure Web Server's private key when installing the digital certificate. You must also save a copy of the Netscape Web Server's digital certificate in order to install it in the Secure Web Server.

The Secure Web Server must have the same Common Name and IP address as the Netscape Web Server. This data was used when creating the Certificate Signing Request that you sent to your Certificate Authority when requesting the digital certificate. The Common Name is usually the same as the fully qualified host name of the server.

5.11.2 Migrating the Netscape Digital Certificate

Follow these steps to migrate your Netscape Web Server private key and digital certificate to the Secure Web Server:

1. Login as root on the system where you installed both Web servers and start the Netscape Communicator 4.X Web browser:

   ```
   # su root
   # /usr/bin/X11/netscape &
   ```

2. Create a backup copy of the Web browser certificate file and the private key database file in the root user's `$HOME/.netscape` directory:

   ```
   # cp -pf  /.netscape/key3.db /.netscape/key3.db.orig
   # cp -pf  /.netscape/cert7.db /.netscape/cert7.db.orig
   ```

3. Copy the Netscape Web Server digital certificate file and private key database file from the Web server root to the `/.netscape` directory, overwriting the Web browser certificate file and key database file:

   ```
   # cp -pf Netscape server root/alias/server key database name-key3.db/\
   .netscape/key3.db
   # cp -pf Netscape server root/alias/server certificate database name-cert7.db /\
     .netscape/cert7.db
   ```

4. Export the Web Server private key database to a PKCS#12 (PFX) format certificate file using Netscape Communicator, as follows:
   a. Under the Communicator pull down menu, select the Security Info option in the Tools menu. (Alternately, click on the padlock icon in the bottom left hand corner of the Web browser.) The Security Info dialog box is displayed.
   b. Select the Yours option under Certificates in the Security Info dialog box. The Server-Cert certificate appears in the displayed list.
   c. Select the Server-Cert certificate and click on the Export button.

   Note:

   You must use the same password you used for the Netscape Web Server key database when prompted to enter the passwords for accessing and exporting the certificate.

   d. Export the certificate to the PKCS#12 format certificate file by entering the name of the file (for example, `cert.p12`) in the Save As pop-up menu, then click on OK.

5. Extract the private key from the PKCS#12 format certificate file (cert.p12) using the OpenSSL `pkcs12` command. Save the private key to a PEM-format private key file, using the same password you entered for the import password and PEM pass phase:

```
# /usr/internet/httpd/bin/openssl pkcs12 -nocerts -in/\
.netscape/cert.p12 -out /.netscape/key.pem
Enter Import Password: password
    MAC verified OK
    Enter PEM pass phrase:
    Verifying password - Enter PEM pass phrase: password
```

6. Remove the PEM pass phase from the private key file using the OpenSSL `rsa` command:

```
# /usr/internet/httpd/bin/openssl rsa -in /.netscape/\
key.pem -out /.netscape/keyout.pem
    read RSA key
Enter PEM pass phrase: password
writing RSA key
```

7. Create the Secure Web Server Private Key directory and copy the private key file to the `server.key` file into the directory:

```
# mkdir -p /usr/internet/httpd/server name/conf/ssl.key
# chown root:system /usr/internet/httpd/server name/conf/ssl.key
# chmod 640 /.netscape/keyout.pem
#chown root:nobody /.netscape/keyout.pem
#cp -pf /.netscape/keyout.pem /usr/internet/httpd/\
server name/conf/ssl.key/server.key
```

Note:

The *`server name`* directory should be omitted when creating the private key file for the Public Web Server instance.

8. Copy back the original Web browser certificate file and key database file overwriting the Web Server certificate file and key database file, then remove the files you created:

```
# cp -pf  /.netscape/key3.db.orig /.netscape/key3.db
# cp -pf  /.netscape/cert7.db.orig /.netscape/cert7.db
# rm -f /.netscape/cert.p12 /.netscape/key.pem /.netscape/keyout.pem
```

9. Using the copy of the Netscape Web Server certificate you recieved back from your Certificate Authority, install the digital certificate into the Secure Web Server using the Install a Certificate form provided in the Secure Web Server Administration server (see Section 5.6).

A Secure Web Server 1.3 Components and Modules

Table A-1 lists the major components of the Secure Web Server 1.3 kit, along with a URL for more information about each component.

Table A-2 lists the Apache source distribution standard modules that are provided with the Secure Web Server 1.3. Modules are either provided built in to the httpd image or as Dynamic Shared Objects.

Table A-3 lists modules that are part of the mod_ssl distribution that is integrated in the Secure Web Server 1.3 kit.

Table A-4 lists other modules provided with the Secure Web Server.

Table A-1 Secure Web Server 1.3 Components

Component	Description	URL
Apache HTTPD	Apache HTTP Server.	http://www.apache.org
mod_ssl	An Open Source toolkit that implements the Secure Sockets Layer (SSL Version 2/Version 3) and Transport Layer Security (TLS Version 1) protocols, as well as a complete, general purpose cryptography library.	http://www.openssl.org
PHP	A server-side, cross-platform HTML embedded scripting language.	http://www.php.net
fastcgi	A language-independent, scalable, open extension to CGI that provides high performance without the limitations of server-specific APIs.	http://www.fastcgi.com
auth_ldap	An LDAP authentication module for Apache.	http://www.rudedog.org/auth_ldap/
Analog	A popular log file analyzer.	http://www.analog.cx

Table A-2 Standard Secure Web Server 1.3 Modules

Module	DSO or Integrated	Description
mod_access	Integrated	Provides access control based on client host name or IP address.
mod_actions	Integrated	Executes CGI scripts based on media type or request method.
mod_alias	Integrated	Maps different parts of the host file system in the document tree, and URL redirection.
mod_asis	Integrated	Sends files that contain their own HTTP headers.
mod_auth	Integrated	Provides user authentication using text files.
mod_auth_anon	DSO	Allows “anonymous” user access to authenticated areas.
mod_auth_db	DSO	Provides user authentication using Berkeley DB files.
mod_auth_dbm	Integrated	Provides user authentication using DBM files.
mod_autoindex	Integrated	Generates automatic directory listings.
mod_cern_meta	DSO	Support for HTTP header metafiles.
mod_cgi	Integrated	Invokes CGI scripts.
mod_digest	DSO	Provides MD5 authentication.
mod_dir	Integrated	Provides basic directory handling.
mod_expires	DSO	Applies Expires: headers to resources.
mod_headers	DSO	Adds arbitrary HTTP headers to resources.
mod_imap	Integrated	Provides the image map file handler.
mod_include	Integrated	Provides server-parsed documents.

Table A-2 Standard Secure Web Server 1.3 Modules (*continued*)

Module	DSO or Integrated	Description
`mod_info*`	Integrated	Provides server configuration information.
`mod_log_agent`	DSO	Provides a log of user agents.
`mod_log_config`	Integrated	Provides user-configurable logging replacement for `mod_log_common`.
`mod_log_referer`	DSO	Provides a log document references.
`mod_mime`	Integrated	Determines document types using file extensions.
`mod_mime_magic`	DSO	Determines document types using “magic numbers”.
`mod_mmap_static`	DSO	Experimental file caching; maps files into memory to improve performance.
`mod_negotiation`	Integrated	Negotiates content.
`mod_proxy`	DSO	Provides caching proxy abilities.
`mod_rewrite`	DSO	Provides powerful URI-to-file name mapping using regular expressions.
`mod_setenvif`	Integrated	Sets environment variables based on client information.
`mod_so *`	Integrated	Provides support for loading modules at run time.
`mod_speling`	DSO	Automatically corrects minor typos in URLs.
`mod_status`	Integrated	Displays server status.
`mod_unique_id`	DSO	Generates unique request identifier for every request.
`mod_userdir`	Integrated	Provides user home directories.
`mod_usertrack`	DSO	Provides user tracking using Cookies (replacement for `mod_cookies.c`).
`mod_vhost_alias`	DSO	Provides support for dynamically configured mass virtual hosting.

Table A-3 Secure Web Server 1.3 – Related Modules

Module	DSO or Integrated	Description
`mod_ssl`	DSO	Provides SSL connections.
`mod_define`	DSO	Provides support for variables in configuration directives.

Table A-4 Additional Modules Provided with the Secure Web Server

Module	DSO or Integrated	Description
`mod_auth_ldap`	DSO	Provides support for authentication with an LDAP database.
`mod_fastcgi`	DSO	Supports connections to FastCGI processes.
`mod_frontpage`	DSO	Provides support for Microsoft's FrontPage extensions.
`mod_php4`	DSO	Provides the PHP processing engine.
`mod_jk`	DSO	Provides an interface to Java Servlet engines.
`mod_jk2`	DSO	Provides an interface to Java Servlet engines.

B Secure Web Server 2.0 Components and Modules

Table B-1 lists the major components of the Secure Web Server 2.0 kit, along with a URL for more information about each component.

Table B-2 lists the Apache 2.0 source distribution standard modules that are provided with the Secure Web Server 2.0. Modules are either provided built in to the `httpd` image or as Dynamic Shared Objects.

Table B-3 lists other modules provided with the Secure Web Server 2.0.

Table B-1 Secure Web Server 2.0 Components

Component	Description	URL
Apache HTTPD	Apache HTTP Server.	http://www.apache.org
`fastcgi`	A language-independent, scalable, open extension to CGI that provides high performance without the limitations of server-specific APIs.	http://www.fastcgi.com
PHP	A server-side, cross-platform HTML embedded scripting language.	http://www.php.net

Table B-2 Standard Secure Web Server 2.0 DSO Modules

Module	Description
`mod_access`	Provides access control based on client hostname, IP address, or other characteristics of the client request.
`mod_actions`	Provides for executing CGI scripts based on media type or request method.
`mod_alias`	Provides for mapping different parts of the host file system in the document tree and for URL redirection.
`mod_asis`	Sends files that contain their own HTTP headers .
`mod_auth`	Provides user authentication using text files.
`mod_auth_anon`	Allows “anonymous” user access to authenticated areas.
`mod_auth_dbm`	Provides for user authentication using DBM files.
`mod_autoindex`	Automatically generates directory indexes similar to the UNIX `ls` command or the Win32 `dir` shell command .
`mod_cache`	Provides content cache, keyed to URIs.
`mod_cern_meta`	Provides CERN `httpd` metafile semantics.
`mod_cgi`	Executes CGI scripts.
`mod_cgid`	Executes CGI scripts using an external CGI daemon.
`mod_charset_lite`	Specifies character set translation or recoding.
`mod_digest`	MD5 authentication
`mod_dav`	Provides Distributed Authoring and Versioning (WebDAV) functionality.
`mod_deflate`	Compresses content before it is delivered to the client.
`mod_dir`	Provides for “trailing slash” redirects and for serving directory index files.
`mod_echo`	Provides a simple echo server to illustrate protocol modules.
`mod_env`	Modifies the environment which is passed to CGI scripts and SSI pages.
`mod_expires`	Generates Expires HTTP headers according to user-specified criteria.
`mod_ext_filter`	Passes the response body through an external program before delivery to the client.
`mod_file_cache`	Caches a static list of files in memory.
`mod_headers`	Customizes HTTP request and response headers.
`mod_imap`	Provides server-side imagemap processing.

Table B-2 Standard Secure Web Server 2.0 DSO Modules (*continued*)

Module	Description
`mod_include`	Provides for server-parsed HTML documents (Server Side Includes)
`mod_info`	Provides a comprehensive overview of the server configuration.
`mod_log_config`	Logs requests made to the server.
`mod_logio`	Logs input and output bytes per request.
`mod_mime`	Associates the requested filename's extensions with the file's behavior (handlers and filters) and content (mime-type, language, character set and encoding).
`mod_mime_magic`	Determines the MIME type of a file by looking at a few bytes of its contents.
`mod_negotiation`	Provides for content negotiation.
`mod_proxy`	Runs the HTTP/1.1 proxy/gateway server.
`mod_rewrite`	Provides a rule-based rewriting engine to automatically rewrite requested URLs.
`mod_setenvif`	Allows the setting of environment variables based on characteristics of the request.
`mod_so`	Loads executable code and modules into the server at start-up or restart time.
`mod_speling`	Attempts to correct mistaken URLs that users might have entered by ignoring capitalization and by allowing up to one misspelling.
`mod_ssl`	Provides strong cryptography using the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols.
`mod_status`	Provides information on server activity and performance
`mod_suexec`	Allows CGI scripts to run as a specified user and group.
`mod_unique_id`	Provides an environment variable with a unique identifier for each request.
`mod_userdir`	Defines user-specific directories.
`mod_usertrack`	Provides clickstream logging of user activity on a site.
`mod_vhost_alias`	Provides support for dynamically configured mass virtual hosting.

Table B-3 Secure Web Server 2.0 – Additional Modules

Module	DSO or Integrated	Description
`mod_fastcgi`	DSO	Supports connections to FastCGI processes.
`mod_php4`	DSO	Provides the PHP processing engine.
`mod_jk`	DSO	Provides an interface to Java Servlet engines.
`mod_jk2`	DSO	Provides an interface to Java Servlet engines.

Glossary

Apache Web Server A freely available UNIX-based Web server. It is currently the most commonly used server on Internet connected sites. HP's implementation of the Apache Web Server is called the Secure Web Server for Tru64 UNIX. For Internet Express Version 6.0 and later, two versions of the Apache Web Server are offered: 1.3 and 2.0.

certificate authority A third party organization that confirms the relationship between a party to the https transaction and that party's public key. Certification authorities may be widely known and trusted institutions for Internet-based transactions. Where https is used on a company's internal network, an internal department within the company may fulfill this role.

digital certificate A token which underpins the principle of trust in SSL-encrypted transactions. The information within a certificate includes the issuer (the Certificate Authority that issued the certificate), the organization that owns the certificate, the public key, the validity period (usually one year) of the certificate, and the host name for which the certificate was issued. It is digitally signed by the Certificate Authority so that none of the details can be changed without invalidating the signature. See also certificate authority, digital signature.

digital signature A use of public key cryptography to authenticate a message. Digital signatures use a private key to indicate that the signature was made by the owner of that key. See also public key cryptography, private key.

distinguished name Also called DN. A sequence of relative distinguished names (RDNs). See also relative distinguished name.

DN See distinguished name.

DNS Domain Name System. A general-purpose, distributed, replicated data query service chiefly used on the Internet to translate host names into Internet addresses. See also fully qualified domain name.

Domain Name System See DNS.

dynamic module A module that provides the means for building program code in a format that can be loaded into the address space of an executable program at run time. Dynamic modules are loaded into the server process space only when necessary and assure that overall memory usage is reduced.

firewall Hardware and software that lies between two networks, such as an internal network and an Internet service provider. The firewall protects your network by blocking unwanted users from gaining access and by disallowing messages to specific recipients outside the network.

FQDN See fully qualified domain name.

fully qualified domain name Also called FQDN. The full name of a system, consisting of its local host name and its domain name. A fully qualified domain name is usually precise enough to determine an Internet address for any host on the Internet.

HTTP Hyper Text Transfer Protocol. The protocol used between a Web browser and a server to request a document and transfer its contents. The specification is maintained and developed by the World Wide Web Consortium. See also HTTPS.

HTTPS Ordinary HTTP exchanged over an SSL-encrypted session.

port A logical channel in a communications system.

private key The part of the key in a public key system that is kept secret and is used only by its owner. This is the key used for decrypting messages, and for making digital signatures. Compare with public key.

public key The part of the key in a public key system which is distributed widely and is not kept secure. This is the key used for encryption (as opposed to decryption) or for verifying signatures. Compare with private key.

public key cryptography Public key cryptography uses a key for encryption and a different key for decryption. Although the keys are related, it is not possible to calculate the decryption key from only the encryption key in any reasonable amount of computation time. In most practical systems, the public key system is used for encoding a session key which is used with a symmetric system to encode the actual data. RSA is an example of a public key algorithm.

RDN See relative distinguished name.

relative distinguished name Also called RDN. One or more attribute/value pairs stored on an LDAP server that uniquely identify an entry from its sibling in an object tree.

secret key Part of a symmetric cipher in which the same key is used for encryption and decryption. SSL encryption uses a secret-key nested within a public key and authenticated through certificates. Secret-key encryption provides faster access than public-key encryption alone. See also public key cryptology.

Secure Socket Layer See SSL.

session key A key used for one message or set of messages. In a typical system, a random session key is generated for use with a symmetric algorithm to encode the bulk of the data. Only the session key is communicated using public key encryption. See also public key cryptology.

SSL Secure Socket Layer. A protocol developed by Netscape for encrypted transmission over TCP/IP networks. SSL sets up a secure end-to-end link over which http or any other application protocol can operate. The most common application of SSL ishttps for SSL-encrypted HTTP.

TCP/IP Transmission Control Protocol/Internet Protocol. Ethernet protocols incorporated into 4.2 BSD UNIX. While TCP and IP specify two protocols, the combined term is used to refer to the entire Department of Defense protocol suite, including Telnet and FTP.

Telnet The Internet standard protocol for remote logins. UNIX BSD includes the telnet program, which uses the protocol, and acts as a terminal emulator for remote login sessions.

VeriSign A dominant certificate authority on the Internet, though many of its certificates are signed as RSA Data Security. Early versions of Microsoft and Netscape browsers had RSA Data Security configured as the only trusted certificate authority. This mandated that users who want to use certificates on the internet must obtain them from VeriSign and use server software accredited by VeriSign. Current versions of the Microsoft and Netscape browsers allow users to add new certificate authorities. As older versions of the browsers are replaced, new certificate authorities (such as Thawte) have emerged.

virtual host An alias name assigned to an FTP server.

Web server A server process, running at a Web site, that sends out Web pages in response to HTTP requests from remote browsers.

Index