Sun WebServer Installation Guide

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto , CA 94303–4900
U.S.A.

Part No: 805–5730
July 1, 1998

Contents

CHAPTER 1

Sun WebServer 2.0 Installation

The installation process is broken up into the following sections:

- Installation Requirements - Lists the minimum system requirements for installing the Sun WebServer.
- Planning - Lists issues to think of before installing the Sun WebServer.
- Installing Sun WebServer - Gives instructions on installing the Sun WebServer.
- Packages - Describes the packages that need to be installed.

Installation Requirements

Before installing Sun WebServer on your machine, check the following minimum system requirements:

Platforms:

- SPARC
- Intel

Operating System:

- Solaris 2.6

Disk Space:

- 10 MB for executables and configuration files

- 3 MB or more additional space for web site documents, scripts, and log files depending on number and size of web sites hosted.

Memory:

- SPARC: 64 MB minimum, 128 MB recommended
- Intel: 64 MB minimum, 128 MB recommended

Patches (SPARC):

- 105181–06 Kernel
- 106257–01 PAM authentication library (/usr/lib/libpam.so.1) patch.
- 105390–02 SGML man pages update
- 106123–02 SGML tools for man commands
- 105284–08 Motif patch for AWT 1.1

Patches (Intel):

- 105182–06 Kernel
- 106258–01 PAM authentication library (/usr/lib/libpam.so.1) patch.
- 106061–02 SGML man pages update
- 106124–02 SGML tools for man commands
- 105285–08 Motif patch for AWT 1.1

Swap Space:

- SPARC 64 MB minimum, 96 MB recommended
- Intel 64 MB, 96 MB recommended
- Swap space should be one and a half to two times the required memory multiplied by the number of processors.

Web Browser Administration:

- Sun Microsystems HotJava™ 1.1.4 and greater
- Java™ Developers Kit (JDK) 1.1.5 or greater.

Planning Your Installation

Before you install Sun WebServer, you'll want to think about the following:

- *Java™ Servlet Support* - Using servlets rather than the more usual CGI scripts provides the cross-platform advantages of Java on the server side while improving processing speed. Java Servlet Support requires the `SUNWhtsvl` package and Solaris™ JDK™ 1.1.5 or greater. You can download the lastest version of Solaris JDK at URL `http://www.sun.com/solaris/java/`.
- *SSL for Secure HTTP Communications* - Sun WebServer includes secure-socket layer (SSL 3.0) functionality and support for Verisign certificates for safe, encrypted communications.
- *Microsoft FrontPage Support* - Sun WebServer supports popular FrontPage 98 server extensions to allow for Microsoft FrontPage authoring, administration, and dynamic content. Support for FrontPage 98requires additional software available on Microsoft's web site at URL `http://www.microsoft.com/frontpage/wpp/`.
- *User name* - User name under which the server process runs. Running the server as `root` could present a possible security risk as CGI scripts could be run as root. Choose the CGI user, CGI suffix enable, CGI maps, publishing, and site restrictions carefully. Running the server as `nobody` is safe, but limits the system files and resources the server can access.

Installing Sun WebServer

▼ To install Sun WebServer using the `pkgadd` command:

1. **Become superuser.**

```
% su
```

2. **Change (`cd`) to the Sun WebServer directory on the CD or to the directory where you unpacked the archives. For example:**

Note - The packages are located in the directory where you unpacked the archives; however, the actual directory names may vary depending on how you purchased Sun WebServer.

```
# cd /tmp
```

3. **If you want SSL support, complete this step. If you do not want SSL support, skip this step and move on to Step 4.**

   You should install the packages in the following order as some packages have dependencies on other packages.

   The installation process is interactive and will prompt you for answers. Answer "yes" to all questions in the `pkgadd` process.

▼ SSL Installation

a. **Remove the `SUNWfns` package installed with Solaris 2.6.**

   Disregard the warnings regarding the disposal of `SUNWfns`.

   Note - `SUNWfns` should be upgraded to the version we supply on Solaris 2.6 systems. On Solaris 7, you do not need to remove or upgrade the exisitng `SUNWfns`.

```
# pkgrm SUNWfns
```

b. **Change directories to the directory that contains the SSL packages, and install the updated version `SUNWfns` along with the `SUNWski` and `SUNWssl` packages.**

   Note - If you have a global export version, install `SUNWssl` instead of `SUNWssld`. Also, there is no `SUNWskild` package to install. The `SUNWskild` package should be installed after the `SUNWski` package to ensure the integrity of all symbolic links.

```
#
pkgadd -d . SUNWfns SUNWski SUNWskild SUNWskica\ SUNWskicw SUNWskimc SUNWskimu SUNWssld
```

4. **Change directories to the directory containg the Sun WebServer core packages, and install the packages.**

   The installation process is interactive and will prompt you for answers. Answer "yes" to all questions in the `pkgadd` process.

   Note - `SUNWhtsvl` is the Java Servlet Support package and is only required if you want support for servlets. It requires Solaris™ JDK™ 1.1.5 or greater. You can download the lastest version of Solaris JDK at URL `http://www.sun.com/solaris/java/`. `SUNWhtadm` is the Sun WebServer Administration Console applet and Administration Server package. It is recommended for easy management of Sun WebServer.

   ```
   # pkgadd -d . SUNWhttp SUNWhtsvl SUNWixklg SUNWixavm SUNWhtadm SUNWhttpc
   ```

5. **Change directories to the directory containing the Sun WebServer online help and man pages, and install the documentation packages:**

   ```
   # pkgadd -d components/SUNWhttp/common/pkgs SUNWhtdoc SUNWhtman
   ```

Packages

Sun WebServer includes the following packages:

`SUNWhtadm`	Administration Package
`SUNWhtsvl`	Servlet Support Package
`SUNWhttp`	Daemon and Supporting Binaries Package
`SUNWhttpc`	Configuration Package
`SUNWhtdoc`	Online Help Package
`SUNWhtman`	Sun WebServer Man Pages Package
`SUNWfns`	Federated Naming System

`SUNWski`	SKI 1.0 Software (User Package)
`SUNWskica`	SKI 1.0 Software (CA Package)
`SUNWskicw`	SKI 1.0 Software (Licensing Package for CA)
`SUNWskimc`	SKI 1.0 Software (CA Manual Page Package)
`SUNWskimu`	SKI 1.0 Software (CA User Manual Page Package)
`SUNWssl`	SSL 1.0 Software (Global Version Library)
`SUNWskild`	SKI 1.0 Software Package (US and Canada Library)
`SUNWssld`	SSL 1.0 Software (US and Canada Library)
`SUNWixklg`	Solaris for ISPs KLG classes.
`SUNWixavm`	Solaris for ISPs JMAPI AVM classes.

CHAPTER **2**

Configuring the Administration Server

The Sun WebServer Administration Console is a browser-based, secure, graphical administration tool for administering Sun WebServer instances and sites. You must be running an administration server to run the console. Although some of the following tasks can also be performed using the command line, using the Administration Console is strongly recommended. .

This chapter explains what need to be configured before starting the Administration Server:

- Access Control: Create an administrator and grant permission to access the Sun WebServer Administration Console. Additional administrators can be created through the Administration Console once its running.
- Port: Specify the port on which the administration server runs. The default port is 2380.
- Starting the Administration Server: Start the administration server as root with `htserver start -a`.

Administration Server Access Control

You must designate a realm that defines the server administrators; by default, the realm name is `serverAdmin`, and the source of information for user name and password is `HTPASSWD`. The source of user information determines whether user list is created with `HTPASSWD` utility or from the user information on the system.

You must be `root` to change realm information for the Administration Server.

You need at least one administrator of the realm so that at least one user has the ability to change the realm itself.

▼ Using the Default Realm

1. **Create a user** `admin` **as the administrator in the** `serverAdmin` **realm:**

```
#
htrealm add -r serverAdmin -u admin Setting password for the user admin. Password: Confirm Password:
```

2. **Assign the user** `admin` **administrative privileges for the realm so that it can create other users later:**

```
# htrealm add -r serverAdmin -u admin -A
```

▼ Creating a New Server Administration Realm

To use a different realm or to change the source of user information in `serverAdmin`, complete the following steps:

1. **Delete the current** `serverAdmin` **definition:**

```
# htrealm delete -r serverAdmin
```

2. **Create a new realm.**

   The realm source can be `HTPASSWD`, or `UNIXSYS`.

```
# htrealm add -r serverAdmin -s UNIXSYS
```

3. **Name an existing realm user as the administrator of the realm. This user will have permission to delete or change the realm.**

```
# htrealm add -r serverAdmin -u userName -A
```

4. **If you used a realm name other than** `serverAdmin`**, you'll need to replace the access control in** `/etc/http/access.conf` **with your realm name.**
5. **For detailed help on** `access.conf` **and** `htrealm`**, refer to the** `access.conf(4)` **and** `htrealm(1m)` **man pages.**

Administration Server Port

The administration server uses port 2380 by default, but you may specify any port for the Sun WebServer administration server to use. To change the default port, edit the administration server's configuration file (located in `/usr/http/admin_server/conf/admin.httpd.conf`) and change the port number for the `port {}` block. If the Administration Server is already running, you must restart it to use the new port.

The default `port {}` block looks like:

```
port 2380 {
...
}
```

Start the Administration Server

To start the Administration Server, run `htserver start` with the –`a` option:

```
# htserver start -a admin : Started successfully.
```

To verify that your server is running, use the `htserver list` command. If the server instance named `admin` has a process ID listed, then the server is running:

```
#
htserver list  Instance : admin         Enabled : Yes         pid : 29159         Config file : /usr/ht
```

If you want the Administration Server automatically restarted if it stops or at reboot, you must enable the server. If the administration server is enabled, it will be restarted whenever `htserver start` is run with no arguments.

To enable the server:

```
# htserver enable -a  admin: Enabled.
```

For more information on `htserver`, refer to the `htserver(1m)` man page.

Using the Sun WebServer Administration Console

Using the Administration Console The Sun WebServer Administration Console allows you to configure settings and maintain the server.

▼ To use the Sun WebServer Administration Console:

1. **Open HotJava™.**

2. **Go to URL** `http://`*hostname*`:`*port*`/admin/admin.html`**.**

   For site administration, select Web Sites from the open folder list on any server configuration screen or go directly to URL `http://`*hostname*`:`*port*`:admin/admin.html?`*server*`:`*site*.

   This URL opens with the Server Properties screen.

CHAPTER **3**

SSL Configuration

SSL Requirements Overview

This section explains each of the major components required at your site to use Secure Sockets Layer (SSL) with web sites. This is intended to give you an overview of what you must have in place and how the components interact. A list of the procedures to follow to configure SSL is in the “SSL Configuration Procedures” on page 14.

Before a web site can use SSL, it must have public and private keys for encryption and an X.509 certificate which it can present to clients. The certificate contains the web site’s identity (distinguished name), the identity of the issuer, the web site’s public key, and the digital signature of the issuer. Public web sites typically get certificates signed by a third-party certificate authority (CA) such as VeriSign; if a client also has the public key of the third-party CA, it can trust that the site’s identity has been verified and is authentic.

Note - "Credentials" in this document refers to a key package-public and private encryption keys-and an associated certificate.

Sun WebServer includes software for running a CA. The CA can create SSL credentials for web sites. Other tools allow you to install the web site’s credentials for use by a Sun WebServer instance, to get credentials signed by a third-party, and to install third-party certificates.

The following must be completed at your site to run SSL:

- Create root certificate authority (Root CA).
- Install federated naming system (FNS).
- Install Sun WebServer with SSL.

- Use unique IP address for web sites that use SSL.
- (Recommended) Request signed certificates from an independent CA.

Root Certificate Authority (Root CA)

You need to create a Root CA at your site to create credentials for web sites. A Root CA user will create credentials for itself, and then use the credentials to create key packages and sign certificates for web servers in your network. You may store the credentials in the Federated Naming Service (FNS) for easy accessibility from other machines, or you can store them only in files on the Root CA machine to limit access. By default, they are stored in `/var/fn`.

The Root CA host (where credentials are created) does not need to be the same machine as Sun WebServer, and for security reasons you may want to run the Root CA on a different machine or a machine with no network access at all.

Note - The preceding steps need to be completed only for self-signed certificates. It is not needed if only third-party certificates are used.

Root CA User

You can use any user name except for `root` (UID 0) on the Root CA host to be the Root CA user. The Root CA user is the only user that can create credentials for web sites. The Root CA user will have its own, password-protected credentials which are used to sign all of the certificates it creates.

The Root CA credentials are bound to a distinguished name (DN) entry. All credentials are bound to a DN. The Root CA distinguished name uses the following attributes:

Attribute Type	Abbreviation	Example
Common name	cn	cn=rootca
Email address	em	em=rootca@A.net
Serial number	serial	serial=no12345
Organizational unit name	ou	ou=web
Organization Name	o	o=A.net
Locality name	l	l=internet

Attribute Type	Abbreviation	Example
State or province name	st	st=California
Country name	c	c=US

The order of the attributes matters in the DN. The DN must begin with the most specific attribute and continue to the least specific. The attributes are listed in the table from most specific (common name) to least specific (country).

All credentials are stored in a directory owned by the Root CA user, which should not be publicly readable. The Root CA user's credentials (as well as each web site's credentials) will be available through the Federated Naming Service (FNS).

Root CA Host

All computers that use SSL or key packages will need to have the security tools packages installed. There must be at least one machine, the Root CA host, where

- The user name of the Root CA exists.
- The credentials of the Root CA are stored.
- FNS is properly installed.

The Root CA will create and store credentials for web sites on this host.

Running Sun WebServer on the Root CA host is not necessary. A Sun WebServer machine can get access to the credentials for web sites it hosts by copying the files from the Root CA hosts.

Federated Naming Service (FNS)

FNS is the interface the SSL security tools use to access credentials. The Sun WebServer security packages create a name service context in which to store and retrieve credentials from files.

Remove any existing FNS packages and install the `SUNWfns` package bundled with Sun WebServer.

This step is done before you install Sun WebServer. If you did not remove `SUNWfns` and install the bundled package, do so now on hosts where you will use SSL, including the Root CA host.

Note - The `SUNWfns` package that is part of Solaris 2.6 will not work with SSL. Removing the package will not cause you to lose any data, and your existing FNS contexts will be preserved once the new `SUNWfns` package is installed. Both packages are version 11.6.0, but the full `VERSION` from `pkginfo -l SUNWfns` is `11.6.0,REV=1998.02.08.15.10` for the supported package.

SSL-Enabled Sun WebServer

To support SSL for web sites, you need an instance of Sun WebServer that has all of the SSL packages and libraries available and has SSL enabled on a port for each IP address where SSL will be used.

Once credentials are created by the Root CA, the credentials must be installed on the Sun WebServer machine where the site is hosted.

Unique IP for Web Site

The Root CA user creates credentials for a web site, using the web site's host name and IP address. The credentials must be bound to a unique host name and IP address, so there must be a unique IP address with an SSL port for each SSL-enabled site.

Certificate Signing

When a client connects to an SSL enabled port on a web site, it requests the site's credentials. To verify the credentials, they must be signed by a CA for which the client has a public key and which the client trusts.

Since most clients will not have your local Root CA's public key, you will want to get site credentials signed by a well known CA, such as VeriSign.

SSL Configuration Procedures

This section lists the top level procedures to follow to configure SSL. Each procedure has a link to more detailed steps.

If you are unfamiliar with the environment required for running SSL and a Certificate Authority, please refer to "SSL Requirements Overview " on page 11 before performing the following procedures:

1. “Root CA Configuration” on page 13
2. “Creating Credentials” on page 17
3. “Enabling SSL on a Web Site ” on page 20
4. (optional) “Requesting Signed Certificates ” on page 23

Root CA Configuration

The Root Certificate Authority (CA) is required to create key packages and certificates for web sites on your network. See “SSL Requirements Overview ” on page 11 if you are unfamiliar with the role of the Root CA user and Root CA machine.

You need to configure the Root CA machine, and then create the Root CA that can create credentials for web sites.

▼ To Configure the Root CA Machine

1. **Make sure that the correct packages for generating credentials are installed on the machine:**

SUNWhttp	Contains tools and scripts for running the Root CA.
SUNWfns	Up-to-date Federated Naming Service files.
SUNWski	SKI library.
SUNWskica	Encryption software for generating key packages and certificates.
SUNWskicw	Licensing software for the Root CA.
`SUNWskimc`	SKI 1.0 Software (CA Manual Page Package)
`SUNWskimu`	SKI 1.0 Software (CA User Manual Page Package)
`SUNWssl`	SSL 1.0 Software (Global Version Library)
`SUNWskild`	SKI 1.0 Software Package (US and Canada Library)
`SUNWssld`	SSL 1.0 Software (US and Canada Library)

These packages are installed during Sun WebServer installation if you choose to install SSL.

2. **Sun WebServer installation will start the processes required for generating security keys and certificates. Make sure that the following processes are running:**
   - `/usr/lib/security/skiserv`
   - `/usr/lib/security/cryptorand`

3. **Select or create a user to be the Root CA user.**

Note - This document will refer to this user as `rootca`, but you may choose any UNIX user name from `/etc/passwd`.

---

4. **Create a directory owned by rootca where you can store credentials.**

   This directory should not be readable by others. For example:

```
# mkdir /var/SSL_CERTS # chmod 700 /var/SSL_CERTS #
chown rootca /var/SSL_CERTS
```

▼ To Create the Root CA

1. **Determine the distinguished name (DN) entry for the root CA.**

   For details see "Root CA User " on page 12. An example DN is `cn=rootca, o=A.net, st=California, c=US`.

2. **Log in to the Root CA machine as the Root CA user.**

3. **Run** `create_rootca`

   If `create_rootca` is not available in `/usr/bin`, you have not installed the `SUNWski` package on this machine.

4. **Enter the DN for the Root CA.**

```
Enter Distinguished Name (e.g. "o=SUN, c=US") or q[uit]: cn=rootca, o=A.net, st=California, c=US
```

5. **Enter the directory name where credentials will be stored.**

```
Enter directory pathname under which the key package and certificate will be stored, or q[uit].  Directo
```

The script will generate public and private encryption keys for the Root CA. All key packages are protected by a password to prevent unauthorized use.

6. **Enter a password for the Root CA key package.**

```
keypkg: Enter your NEW key package password:  keypkg: Reenter your NEW key package password:
```

7. **You have the option of making the key packages available in the naming service. To store the key package in the naming service, you will need the machine's root password.**

   Key packages are always stored in files. Making the key package available in the naming service allows other security tools to locate the keys without the full path name.

   The Root CA is now configured. The next step is for the Root CA to generate a key package and certificate for a web site.

   Continue with the next configuration procedure, "Creating Credentials" on page 15.

8. **If you don't choose this option, save the credentials in FNS, and then store the credentials manually:**

```
# skistore -d dirname
```

   where *dirname* is as specified in step 5.

Creating Credentials

The Root CA creates and stores credentials for web sites on the Root CA machine. The certificate can then optionally be signed by another CA, such as VeriSign. When the credentials are ready, they are installed on the Sun WebServer machine for use by the web site.

Note - "Credentials" in this document refers to a key package public and private encryption keys and an associated certificate.

▼ To Create Credentials for a Web Site

1. **Determine the distinguished name entry for the web site, using the Fully Qualified Domain Name (FQDN) as the common name (cn).**

   For details on the distinguished name in certificates, see “Root CA User ” on page 12.

2. **Login to the Root CA machine as the Root CA user.**

3. **Create a directory where you can store the credentials you are about to create.**

```
rootca % mkdir /var/SSL_CERTS/121.122.123.12/
```

4. **Run** `/usr/http/bin/setup_creds` **with the appropriate options.**

   Valid options are:

`–d` ***output_directory***	Specifies the directory where credentials should be stored; for example, `/var/SSL_CERTS/121.122.123.12/`.
`–f` ***trusted_file***	(Not required; used to add certificates for other trusted CAs at setup time.) Specifies the full pathname to the file containing the Root CA certificate, for example, `/export/skirca2/certs/skirca2.CERT`.
`–i` ***IP_Address***	Specifies the IP Address of the web site for which credentials are being created.
`–r` ***rootca***	(Optional) Specifies the name of the Root CA user (the user name you have used to run the script). If `–r` is omitted, `setup_creds` will ask for the user name of the Root CA user on this system.

```
#
/usr/http/bin/setup_creds -r rootca \ -d /var/SSL_CERTS/121.122.123.12/ -i 121.122.123.12
```

5. **Enter the host name only as the name of the web site. You will be asked to enter the domain name next.**

   For example, if the web site is `www.V.com`, enter **`www`**.

```
Enter host name on which you run httpd server: (Hit return to use localhost) www
```

6. **Enter the domain name for of the web site.**

```
Enter domain name for your server (for example, eng.sun.com) V.com
```

7. **Enter the DN attributes for the web site, without the common name (cn).**

```
Enter Distinguished Name Suffix for your server (eg: o=SUN, c=US) : o="Company V", st=California, c=US
```

8. **Enter a new password for this web site's credentials.**

   Each key package has a password, which should be different from the password for the Root CA's credentials.

```
Please provide the password to encrypt your server's private key. You will need it when you install the
```

9. **The key package and certificate for the site will be generated and stored in the output directory you named.**

   The location of the certificate is `output_directory/certs/IP_Address.cert`. In this example, it would be `/var/SSL_CERTS/121.122.123.12/certs/121.122.123.12.cert`.

   You will need the certificate if you reinstall this certificate over another one.

10. **Repeat Steps 1 through 9 to generate credentials for additional web sites.**

You now have a "self-signed" certificate. You can use this for SSL encryption if the connecting browser has your Root CA in its list of trusted CAs. This is useful within your organization where you can update browsers that need to use SSL (for example, if you protect the Sun WebServer Administration Console with SSL). Most clients on the Internet, however, will not know about your Root CA so you will want certificates signed by a third party for public SSL sites. Refer to “Requesting Signed Certificates ” on page 23 after you have installed the credentials on the Sun WebServer machine.

Continue with the procedure in “Enabling SSL on a Web Site ” on page 18.

Enabling SSL on a Web Site

Enabling SSL on a web site requires three procedures: ensuring that the SSL packages are installed, installing the credentials on the machine where Sun WebServer is running, and configuring SSL on a port that the web site can use.

Creating Site Credentials

▼ To Install Site Credentials on a Sun WebServer Machine

1. **The directories where the site’s credentials are stored need to be copied to the Sun WebServer machine.**

   If the Root CA machine and the Sun WebServer machine are the same, skip this step.

   You can move the directory to a floppy disk or other portable medium, or you can share the directory with the Sun WebServer machine over NFS.

   In either case, copy the directory you specified for the output of `setup_creds` and all of its subdirectories. The directory should contain:

   - `certs/`*IP_Address*`.CERT`
   - `keypkgs/`*IP_Address*`.KEYPKG`

   where *IP_Address* is the address used by the web site.

2. **As root on the Sun WebServer machine, run** `/usr/http/bin/install_certs`**.**

   You will need to specify the path to the credentials, the IP address of the web site, and the user ID (`uid`) of the Sun WebServer process. For example

```
# /usr/http/bin/install_certs -p /floppy/cert_floppy -i \ 121.122.123.12 0
```

3. **Enter the key package password for this web site.**

   This is the password specified in Step Step 8 on page 19 in "Creating Credentials" on page 17.

```
/usr/bin/skilogin: Enter host key package password
```

   The credentials are now stored on the Sun WebServer machine. Follow the next procedure to configure the web site to use SSL.

Configuring a Web Site for SSL

You must create port on the web site's IP address that uses SSL. The default port used for SSL connections is 443.

These instructions assume you are using the Sun WebServer Administration Console. You can also configure the port by editing the configuration file for the web site's server instance (for example, `/etc/http/sws_server.httpd.conf`). Please refer to the man page for `httpd.conf(4)` if you choose to edit the configuration file.

▼ To Configure A Web Site for SSL:

1. **Connect to the Sun WebServer Administration Console and log in.**

   For information on connecting, see Chapter 2.

2. **Find the server instance that hosts the web site in the Server List. Click on the + to expand the folder if the configuration pages are not listed.**

3. **If you do not know the IP address of the web site, choose the Web Sites page.**

   The IP address(es) used by the web site are shown in the list.

   Note - The IP Address must not be used by multiple web sites. The SSL certificate is bound to a unique IP address and host name.

4. **Click the IP/Ports page to add a port to the web site's IP address.**

   The Network Connections list will display on the right, showing all of the IP addresses and ports used by this server instance.

5. **Click on Add to create a Network Connection using the web site's IP address and port 443.**

The Network Connection Dialog opens.

6. **Fill in the IP Address and Port fields with the web site's IP address and the port on which you want SSL active (usually 443). Set the Timeout and whether you want to Allow HTTP 1.0 Keepalive.**

   If you are unsure about Timeout and Allow HTTP 1.0 Keepalive, click on Help in the dialog. For adequate performance, set the Timeout to 300 seconds and allow HTTP 1.0 Keepalive.

7. **Click the Enable SSL box.**

8. **If you want to accept connections only from clients that have valid personal certificates, click the Require Client Certificate box.**

   For more information on this field click on Help in the dialog.

9. **Set the cipher suites you want to enable.**

   The server will negotiate with the client to use a common cipher suite. If the client and server have more than one suite in common, the strongest suite will be used.

   If you have the US/Canada encryption software, you may choose 128-bit, 40-bit, or both. Select both, unless you explicitly want to require a certain set from clients.

   If you have global encryption software, you can only use the 40-bit cipher suite. Click the 40-bit box.

10. **Click on OK to save your changes, then choose Serve->Save IP/Ports.**

11. **If you are configuring SSL on the default site for the server instance, skip the remaining steps.**

    The default site on a server listens to all connection endpoints defined for that server, so there is no need to add the new SSL connection to the web site.

12. **From the Server List, choose the Web Site page and select the web site in the list. Choose Server->Edit Web Site.**

    In the Edit Web Site dialog, find the SSL enabled network connection in the Connections list and choose it. The connections are listed as `IP_Address:Port` combinations.

13. **Click on > to move the connection in the Site Connections list.**

Note - This option is disabled for default sites because default sites automatically listen in on all connection endpoints for the server. If you are configuring the default site for the server instance, skip steps 12 through 15.

---

14. **Click on** `Save` **to confirm the web site changes.**

15. **Select** `Server->Save` **to save your changes.**

    Continue with the next configuration procedure, “Requesting Signed Certificates ” on page 21.

Requesting Signed Certificates

Having a web site’s certificate signed by an independent CA is the equivalent of having an independent auditor vouch for the site’s identity. Clients may not believe that a secure site is what it claims to be unless its credentials are "digitally signed" by a CA that the client trusts.

Sun WebServer currently only supports VeriSign as a third party CA. You can use the tools that come with Sun WebServer to send a certificate and a certificate signing request (CSR) to VeriSign via their public web site.

▼ To Request a Signed Web Site Certificate

1. **Your local Root CA must generate credentials and store them on the Sun WebServer machine.**

   Refer to “Creating Credentials” on page 17 and “Enabling SSL on a Web Site ” on page 20.

2. **Log in to the Sun WebServer machine as super-user (**`root`**).**

3. **Run the** `send_request` **utility to generate a certificate that can be sent to a CA.**

   On the command line, you must specify the IP address of the site whose certificate you want signed. The portable certificate will be stored in a file in `/tmp`, unless you use `–o` to specify a different directory (the directory must already exist).

```
# mkdir /var/SSL_CERTS/requests
# /usr/http/bin/send_request -o /var/SSL_CERTS/requests \ 121.122.123.12
```

4. **Enter the key package password for the web site.**

   This is not the Root CA's key package password. This is the password you created when you ran `setup_creds`.

5. **The certificate signing request will be stored in the directory you named or `/tmp`, in a file named `cert.request`.**

   The contents of this file can be sent to VeriSign through their web site.

6. **You will need to follow the CA's procedures for requesting a signed certificate. At some point, you will need to supply the generated certificate file to the CA.**

   To request a VeriSign certificate, visit `http://www.verisign.com/idcenter/new/`. You will need to request a server certificate for server software from Sun Microsystems.

7. **When the CA sends the signed certificate, save it in a file.**

   For example, save the reply in `/tmp/121.122.123.12.cert`.

Caution - Do not save the certificate from the CA in the directory the Root CA uses to store credentials.

---

8. **As root, run `/usr/http/bin/install_external` to make the signed certificate available for SSL.**

Enabling Client Authentication

You can use SSL as a method for authenticating client connections if clients have digital IDs. Currently Sun WebServer supports personal digital IDs from VeriSign. VeriSign offers three levels of personal digital ID, based on the strength of the key and the insurance protection:

- Class1
- Class2
- Class3

You can configure an SSL web site to require client authentication and define which level(s) of personal IDs to accept.

▼ To require client authentication for SSL

1. **Log in to the Sun WebServer Administration Console and go to the IP/Ports list for the server where the SSL-enabled web site is hosted.**

2. **Select the SSL-enabled connection used by this web site, and click Edit.**

   The Edit Network Connections dialog opens.

3. **In the dialog box, click on the "Require Client Certificate" box, then click OK.**

4. **Choose Selected->Save IP/Ports to save the configuration.**

5. **Return to the command line and become superuser.**

6.

   The syntax for the command is:

   `setup_client_auth –e | –d –i` *IP_Address Signer*

   The `–e` flag enables, and the `–d` flag disables, access to clients with certificates signed by the *Signer*. The *IP_Address* is the IP address of the SSL enabled web site.

   Signer can be one of the VeriSign classes: Class1, Class2, or Class3.

   Run `setup_client_auth` multiple times to enable or disable multiple signers. The enabled CAs are added to the web site's trusted key list.

CHAPTER 4

Migration from Sun WebServer 1.0 to Sun WebServer 2.0

New Terminology/Structure

Sun WebServer 2.0 differs from Sun WebServer 1.0 in significant ways, so the Sun WebServer 1.0 configuration files must be converted into files compatible with Sun WebServer 2.0. This document is a guide to this conversion. Refer to the Sun WebServer 2.0 documentation for details on the new features of Sun WebServer 2.0.

While Sun WebServer 1.0 and Sun WebServer 2.0 can both be installed on the same machine, they cannot coexist in the same location. You must remove Sun WebServer 1.0 completely before installing Sun WebServer 2.0. Sun WebServer 1.0 can only coexist with Sun WebServer 2.0 if Sun WebServer 1.0 is installed to a non-default directory (a directory other than "/"). Refer to the Sun WebServer 1.0 Installation Guide for instructions on installing Sun WebServer 1.0 to a non-default directory. The examples in this document refer to Sun WebServer 1.0 configuration files using their default installation location (`/etc/http/` for both `httpd.conf` and `access.acl`). After removing the Sun WebServer 1.0 packages (which must be done before installation of Sun WebServer 2.0), the configuration files are moved to a backup directory `/etc/http.bak[.n]/` (for example, `/etc/http.bak/`, `/etc/http.bak.1/`, `/etc/http.bak.2/`, ...). The configuration files for the most recent uninstall will be located in the directory with the highest value for ‘n’.

Instances and Web Sites

Even though Sun WebServer 1.0 was able to support multiple instances of the HTTP server running concurrently, starting, stopping, and restarting the server worked best with a single instance of the HTTP daemon. Sun WebServer 2.0 improves support for multiple instances by providing each httpd daemon on the system with a unique name. One is able to start, stop, and restart individual instances by using their names. See the man page `htserver(1m)` for more information.

Sun WebServer 2.0 has also expanded support of virtual hosts. Each virtual host is now associated with a web site and given its own subtree in the file system, which contains the configuration files for that host as well as the host's document root. While Sun WebServer 1.0 maintained all configuration information for a virtual host in the global configuration file `httpd.conf`, Sun WebServer 2.0 stores most of this information in a site configuration file located relative to the root of the web site. Placing most of the site-specific configuration information at the web site rather than in a single file makes Sun WebServer 2.0 more scalable than Sun WebServer 1.0 and simplifies the administration of individual web sites.

Configuration File Locations

Sun WebServer 1.0 had two primary configuration files in the default location `/etc/http/`:

- `httpd.conf`: Basic configuration information for the server and all its virtual hosts
- `access.acl`: URL access control settings

Sun WebServer 2.0 partitions the directives in these files into server-level configuration files for server instances and site-level configuration files for individual web sites.

Server-Level Configuration Files

Server-level configuration files are installed in `/etc/http/` by default. <*instance_name*> below is the unique name of the httpd instance using the file. An instance name is associated with a server instance when it is created using the Sun WebServer Administration Console or the `htserver add` command.

- `<instance_name>.httpd.conf`: Basic server configuration information
- `access.conf`: URL access controls
- `realm.conf`: Realm information
- `mime.types`: MIME types and encodings

Site-Level Configuration Files

Site-level configuration files are installed to the web site subtree by default. <*site_name*> below refers to the name of the web site using this file.

- `<site_name>.site.conf`: Basic site configuration information
- `access.conf`: URL access controls
- `realm.conf`: Realm information
- `mime.types`: MIME types and encodings
- `map.conf`: URL mappings
- `content.conf`: Content information

All of the file names listed above are suggested names. The only fixed configuration file name is `/etc/http/httpd-instances.conf`. Each server instance name and basic configuration file is listed in `httpd-instances.conf`, and each server configuration file in turn refers to the other configuration files by name.

Command-Line Utilities

Sun WebServer 2.0 has added numerous command line utilities for modification of its configuration files. These utilities are used at various locations in this document to explain migration to Sun WebServer 2.0. Please refer to the Sun WebServer 2.0 man pages for details on all Sun WebServer utilities available from the command line. For the commands referenced in this document, refer to the `htserver(1m)`, `hthost(1m)`, and `htrealm(1m)` man pages. For information on the referenced configuration files see `httpd.conf(4)`, `httpd.site.conf(4)`, `access.conf(4)`, and `realms.conf(4)`. Sun WebServer 2.0 also has an Administration Console, located in the package `SUNWhtadm`, that can be used to administer all aspects of the HTTP server. To access the Administration Console, you must also install `SUNWixklg` and `SUNWixavm`.

Creating a Sun WebServer 2.0 Server

Before converting a Sun WebServer 1.0 configuration into the Sun WebServer 2.0 format, first add a basic Sun WebServer 2.0 instance to the system. This server will contain the general file structure used in Sun WebServer 2.0 and will provide default configuration files that can be modified with the values from the Sun WebServer 1.0 server.

▼ To create a new Sun WebServer 2.0 instance named "server1":

1. **Type the following at the command line (as** `"root"`**):**

```
# htserver add "server1"
```

This command creates a new Sun WebServer 2.0 server-level configuration file and a default web site. The locations where new files are installed are listed below:

- `/etc/http/server1.httpd.conf`: Server configuration file
- `/var/http/server1/`: Root of the new server
- `/ver/http/server1/websites/default_site/`: Site path of default web site
- `/var/http/server1/websites/default_site/conf/`: Location of site configuration file

Migrating the Sun WebServer 1.0 `httpd.conf` File

The Sun WebServer 1.0 `/etc/http/httpd.conf` file contains the basic directives for configuring the server as a whole, the individual virtual hosts, and the ports. The three primary block types are the `server{}` block, the `url{}` block, and the `port{}` block. Each of these types will be discussed in this section.

`server{}` Block

Converting the `server{}` block from Sun WebServer 1.0 to Sun WebServer 2.0 format is relatively straightforward. Most of the directives in 1.0 are the same in 2.0, with the following exceptions listed below:

`acl_delegate_depth`

The `acl_delegate_depth` directive, and the concept of delegation of access controls, is no longer supported in Sun WebServer. See "Migrating the Sun WebServer 1.0 `access.acl` File" in this document.

`acl_enable`

The `acl_enable` directive has been changed to `access_enable`.

▼ To convert the `acl_enable` directive

1. **Search for the following Sun WebServer 1.0 directive in** `/etc/http/httpd.conf`:

```
acl_enable    yes
```

2. **Replace it in** `/etc/http/server1.httpd.conf` **with the following directive**:

```
access_enable   yes
```

acl_file

The `acl_file` directive is no longer supported in the `server{}` block, because the server-level access control file is now configured to be: `/etc/http/access.conf`. This file name cannot be changed.

map

The `map` directive is no longer supported in the `server{}` block. All URL mappings must be made in a web site's `map.conf` file.

mime_add

The `mime_add` directive is no longer supported in Sun WebServer 2.0. All mime types must be specified in the `mime.types` file at either the server or site level.

▼ To convert the `mime_add` directive

1. **Given the following Sun WebServer 1.0 directive**:

```
server {
 mime_add   "image/.jpeg"   "JPG"
}
```

2. **Make sure a** `mime.types` **file is specified in** `/etc/http/server1.httpd.conf`:

Note - If this file is shared among all the servers, changes will effect all httpd daemons

---

```
server {
 mime_file    "/etc/http/mime.types
}
```

3. **Add the new MIME mappings to the** `/etc/http/mime.types` **file:**

```
image/jpeg    JPG
```

▼ `server{}` Block Conversion

1. **Given the following Sun WebServer 1.0** `server{}` **block in** `/etc/http/httpd.conf`:

```
server {
 server_root       "/var/http/demo/"
 server_user       "root"
 mime_add          "image/jpeg" "JPG"
 mime_default_type  text/html
 acl_enable        "yes"
 acl_file          "/etc/http/access.acl"
 acl_delegate_depth   3

 map   /cgi-bin/    /var/http/cgi-bin/   cgi
}
```

2. **Modify the** `server{}` **block in** `/etc/http/server1.httpd.conf`:

```
server {
 server_root       "/var/http/demo/"
 server_user       "http"
 mime_file         "/etc/http/mime.types"
 mime_default_type  text/html
 access_enable     "yes"
}
```

3. **Add the MIME mapping to the `/etc/http/mime.types` file specified in (2):**

```
image/jpeg    JPG
```

`url{}` Block

In Sun WebServer 1.0, all the configuration parameters for a virtual host were found in a single location in the `/etc/http/httpd.conf` file. In Sun WebServer 2.0, global information for the virtual host is located in the server-level configuration file, and more specific information is located in the site-level configuration file.

▼ To convert a Sun WebServer 1.0 `url{}` block to 2.0

1. **Add a new web site to the 2.0 server using the hthost command line utility.**

2. **Modify the new configuration files created with the Sun WebServer 1.0 values, or with Sun WebServer 2.0 equivalents.**

▼ To create a 2.0 web site

1. **Add a web site (widgets), to the current server (server1):**

```
# hthost add -i server1 -h widgets \  -s /var/http/server1/websites/widgets
```

This will add the following entry to the server-level configuration file, `/etc/http/server1.httpd.conf`:

```
url //widgets {
 site_enable     "yes"
 site_path       "/var/http/server1/websites/widgets"
 site_config     "conf/widgets.site.conf"
}
```

It will also create a new web site at: `/var/http/server1/websites/widgets/`. The following configuration files will be placed in the directory `/var/http/server1/websites/widgets/conf/`:

- widgets.site.conf
- access.conf
- realms.conf
- map.conf
- content.conf
- servlets.properties

Modifying the Server-Level Configuration File

Most of the configuration parameters from the Sun WebServer 1.0 `url{}` blocks will be placed in the site-level configuration file, `widgets.site.conf`. One exception is setting the ports on which a virtual host listens. For each virtual host, one must specify on which IP addresses and ports it accepts connections. (This is different than in 1.0, where this information was stored in the `port{}` blocks using the `hosts_supported` directive.) IP address:port pairs are termed "connection end points" in Sun WebServer 2.0. All web sites should have their connection end points specified (with the exception of the default web site, which listens on all interfaces and does not allow connection end points to be defined). For more information on connection end points, see `httpd.conf (4)`.

▼ Setting the connection end points of a web site

1. **To have the virtual host, widgets, accept connections on IP address 129.128.127.126 on port 80, and all IP addresses on port 1880, write the following in** `/etc/http/server1.httpd.conf`:

```
url //widgets {
 site_enable      "yes"
 site_path        "/var/http/server1/websites/widgets"
 site_config      "conf/widgets.site.conf"
 conn_end_points  129.128.127.126:80:1880
}
```

Note - Not specifying IP address to the left of the colon (:) defaults to "all IP addresses" in a HTTP 1.1 virtual host. Also note that all connection end points specified must be mapped by `port{}` blocks.

Modifying the Site-Level Configuration File

Most of the Sun WebServer 1.0 `url{}` block configuration directives can be copied directly into the Sun WebServer 2.0 site-level configuration file, `/var/http/server1/websites/widgets/conf/widgets.site.conf`, with following exceptions:

map

All Sun WebServer 2.0 URL mappings are now stored in a separate file: `map.conf`.

▼ To convert the map directive

1. **Given the following Sun WebServer 1.0 directive in** `/etc/http/httpd.conf`:

```
url //widgets {
 map   /cgi-bin/   /var/http/shared/cgi-bin/   cgi
}
```

2. **Add the following line to the** `/var/http/server1/websites/widgets/conf/map.conf` **file at the widgets site:**

```
map   /cgi-bin/   /var/http/shared/cgi-bin/   cgi
```

`mime_add`

▼ To convert the `mime_add` directive

1. **Given the following Sun WebServer 1.0 directive in** `/etc/http/httpd.conf`:

```
url //widgets {
 mime_add     "image/jpeg"     "JPG"
}
```

2. **Uncomment the** `mime_file` **directive in** `/var/http/server1/websites/widgets/conf/widgets.site.conf`:

```
url //widgets {
 mime_file    "conf/mime.types"
}
```

3. **Add the new mime mappings to the** `/var/http/server1/websites/widgets/conf/mime.types` **file:**

```
image/jpeg     JPG
```

`port{}` Block

There are two primary changes to the `port{}` blocks in Sun WebServer 2.0: the `hosts_supported` directive is no longer valid as the ports a host listens on are now listed using the `conn_end_points` directive in the `url{}` block, and all ports must have an `ip_address` specified. The IP address 0.0.0.0 is special and means "all valid IP addresses on this machine".

▼ To convert a `port{}` block for a specific IP address

1. **Given the following Sun WebServer 1.0** `port{}` **block:**

```
port 80 {
 ip_address         129.128.127.126
 keepalive_enable   "yes"
 request_timeout    180
 hosts_supported     widgets
}
```

2. **Modify the widgets** `url{}` **block in** `/etc/http/server1.httpd.conf` **to have** `conn_end_points` **specified correctly:**

```
url //widgets {
 site_enable      "yes"
 site_path        "/var/http/server1/websites/widgets"
 site_config      "conf/widgets.site.conf"
 conn_end_points   129.128.127.126:80
}
```

3. **Add the** `port{}` **block to** `/etc/http/server1.httpd.conf` **(without the** `hosts_supported` **directive):**

```
port 80 {
 ip_address         129.128.127.126
 keepalive_enable   "yes"
 request_timeout    180
}
```

▼ To convert a `port{}` block for all IP addresses

1. **Given the following Sun WebServer 1.0** `port{}` **block example:**

```
port 1880 {
 keepalive_enable    "yes"
 request_timeout     180
 hosts_supported     widgets
}
```

2. **Modify the widgets** `url{}` **block in** `/etc/http/server1.httpd.conf` **to have** `conn_end_points` **specified correctly:**

```
url //widgets {
 site_enable       "yes"
 site_path         "/var/http/server1/websites/widgets"
 site_config       "conf/widgets.site.conf"
 conn_end_points   :1880
}
```

3. **Add the** `ip_address` **directive with the value 0.0.0.0 to the** `port{}` **block in** `/etc/http/server1.httpd.conf`**:**

```
port 1880 {
 ip_address         0.0.0.0
 keepalive_enable   "yes"
 request_timeout    180
}
```

Migrating the Sun WebServer 1.0 `access.acl` File

Access control has changed substantially between Sun WebServer 1.0 and 2.0. Sun WebServer 2.0 access control documentation should be reviewed before attempting to migrate the old Sun WebServer access control structure to the new. The most significant changes are listed below:

- The concept of realm has been greatly expanded, and a new configuration file, `realm.conf`, has been added.
- Delegation of access control has been removed.
- The access controls specified for the "`/sws-administration`" URL are used throughout Sun WebServer to determine the server administrator when the URL is specified in the server-level `access.conf` or the site administrator when specified in the site-level `access.conf`.
- There is a single server-level `access.conf` file for all the instances on the system (located at `/etc/http/access.conf`). This file contains a single "`/sws-administration`" URL to define who is allowed to administer all instances.
- Each web site also has its own site-level `access.conf` file to specify site administrators.

- Realms can now use more sources for user data. A realm can authenticate a user against a standard HTPASSWD database or NIS+ database.

To migrate the Sun WebServer 1.0 access.acl file to 2.0, one needs to create or select authentication realms, modify the access control syntax for the mapped URLs to reflect the new 2.0 syntax, and collect the access controls in the access.acl file and delegated files to the access.conf files in each of the individual web sites.

Realms

In Sun WebServer 1.0, the realm directive had little significance other than as an identifier for the browser (printed in the authentication dialog box). In Sun WebServer 2.0, the realm specifies a pre-existing set of users and groups used for authenticating access to a URL. Realm information is stored in the realms.conf file. HTPASSWD realms have a directory associated with them as well (defined in realm.conf) which contains the "users" and "groups" files to be used in the authentication. The password_file and group_file directives in Sun WebServer 1.0 ACLs are therefore obsolete. All 1.0 realms will be HTPASSWD realms in 2.0 since these user-created databases were the only source available in 1.0.

▼ Migrating to Sun WebServer 2.0 realms

1. **Given the following Sun WebServer 1.0 access control in** /etc/http/access.acl **for host widgets:**

```
url /reports {
 realm               Managers
 authentication_type  basic

 password_file       /usr/auth/Managers/Maners_users
 group_file          /usr/auth/Managers/Managers_groups

 + group              report_managers
 - user               Joe
}
```

2. **Create a realm to hold the password and group file. This can be done as follows:**

```
# htrealm add -i server1 -h widgets -r Managers -s HTPASSWD
```

This command will add the following entry to the `realms.conf` for the site widgets:

```
realm Managers {
 realm_source HTPASSWD
}
```

It also creates the following directory at the site widgets:

```
/var/http/server1/websites/widgets/conf/realms/Managers/
```

3. **Copy the users and groups file into the** `Managers` **directory:**

```
# cp /usr/auth/Managers/Managers_users \  /var/http/server1/websites/widgets/conf/realms/Managers/users
```

4. **Remove the file directives from the Sun WebServer 1.0 ACL and place the new ACL in the Sun WebServer 2.0 ACL file:** `/var/http/server1/websites/widgets/conf/access.conf`:

```
url /reports {
 realm                Managers
 authentication_type   basic

 + group               report_managers
 - user               Joe
}
```

Delegation

The concept of delegation has been removed from Sun WebServer 2.0. All the access controls that were previously located within delegated files must be relocated into the single `access.conf` for a particular site.

▼ Converting a delegated ACL

1. **Given the following Sun WebServer 1.0 ACL and delegated file:**

- ACL in `/etc/http/access.acl`:

```
url /statistics {
 delegate   /var/http/acls/.admin_acl
}
```

- `/var/http/acls/.admin_acl` file:

```
realm          admin
password_file  /usr/auth/admin_user
group_file     /usr/auth/admin_group

+ group        stat_admins
```

- These must be collapsed into a single ACL:

Note - The realm `admin` must have been created first; see the previous example.

```
url /statistics {
 realm    admin
 + group  stat_admins
}
```

▼ Converting a delegated ACL (advanced)

1. **Given the following Sun WebServer 1.0 ACL and delegated file:**
   - ACL in `access.acl`:

```
url /statistics {
 delegate   /var/http/acls/.admin_acl
}
```

- `/var/http/acls/.admin_acl` file (the ownership of this file is `joe:adm`)

```
realm           admins
password_file   /usr/auth/admin_user
group_file      /usr/auth/admin_group

+ group         stat_admins
```

2. These must be collapsed into a single ACL:

```
url /statistics {
 realm      admins

 administrators {
  user      joe
  group     adm
 }

 + group    stat_admins
}
```

Sun WebServer Conversion

The example below shows a full conversion of Sun WebServer 1.0 `httpd.conf` and `access.acl` files to Sun WebServer 2.0. New 2.0 directives are ignored in the example below, unless they are explicitly required for the conversion.

▼ Sun WebServer 1.0 `httpd.conf` to 2.0

1. Given the following Sun WebServer 1.0 `/etc/http/httpd.conf` **file:**

```
server {
 server_root        "/var/http/demo"
 server_user        "root"
 mime_file          "/etc/http/mime.types"
 mime_default_type   text/html

 acl_enable         "yes"
 acl_file           "/etc/http/access.acl"
 acl_delegate_depth  3
 cache_enable       "yes"
 cache_small_file_cache_size 8
 cache_large_file_cache_size 256
 cache_max_file_size         1
 cache_verification_time     10

 map    /cgi-bin/     /var/http/demo/cgi-bin/   cgi
 map    /sws-icons/   /var/http/demo/sws-icons/

 mime_add  "appication/java"    class
 mime_add  "audio/basic"        au
 mime_add  "audio/basic"        snd
}

url {
 doc_root          "/var/http/demo/public"
 user_doc_enable   "no"
 user_doc_root     "public_html"
 cgi_enable        "no"
 cgi_dns_enable  "no"
 cgi_suffix_enable     "no"
 cgi_user          "nobody"
 log_type          "elf"
 log_prefix        "/var/http/logs/http"
 log_max_files      7
 log_cycle_time     1440
 log_max_file_size  1048576
 ssi_enable         "no"
 ssi_exec           "no"
 ssi_xbithack       "off"

 mime_add        "application/x-csh"  csh
 mime_add        "application/xsh"    sh
}

url //widgets {
 doc_root         "/var/http/widgets/public"
 user_doc_enable  "yes"
 user_doc_root    "public_html"
 cgi_enable       "yes"
 cgi_dns_enable   "yes"
 cgi_suffix_enable  "yes"
```

(continued)

```
 cgi_user         "nobody"

 log_type        "clf"
 log_prefix      "/var/http/logs/widgets"
 log_max_files    7
 log_cycle_time   1440
 log_max_file_size  1048576
 ssi_enable       "yes"
 ssi_exec         "yes"
 ssi_xbithack     "full"

 map   /cgi-bin/    /var/http/widgets/cgi-bin/  cgi
 map   /sws-icons/  /var/http/widgets/sws-icons/
}

port 80 {
 keepalive_enable   "yes"
 request_timeout    180
}

port 1880 {
 ip_address          129.128.127.126
 keepalive_enable   "yes"
 request_timeout    180
 hosts_supported    widgets
}
```

2. **Create a new server for conversion (we will modify the configuration files that are created for the new server with the 1.0 values):**

```
#htserver add "server1"
```

3. **Add a web site, widgets:**

```
# hthost add -i server1 -h widgets \  -s /var/http/server1/websites/widgets
```

4. **Modify the server-level file** `/etc/http/server1.httpd.conf`**:**

```
server {
 server_root      "/var/http/server1/"
 server_user      "root"
 mime_file        "/etc/http/mime.types"
 mime_default_type   text/html
 access_enable     "yes"
 cache_enable      "yes"
 cache_small_file_cache_size  8
 cache_large_file_cache_size  256
 cache_max_file_size          1
 cache_verification_time      10
}

url {
 site_path     /var/http/server1/websites/default_site
 site_config   "conf/default_site.site.conf"
 site_enable   "yes"
}

url //widgets {
 site_enable    "yes"
 site_path      "/var/http/server1/websites/widgets"
 site_config    "conf/widgets.site.conf"
 conn_end_points 129.128.127.126:1880 :80
}

port 80 {
 ip_address       0.0.0.0
 keepalive_enable "yes"
 request_timeout  180
}

port 1880 {
 ip_address      129.128.127.126
 keepalive_enable "yes"
 request_timeout   180
}
```

5. **Modify** `/etc/http/mime.types`:

```
application/java    class
audio/basic         au snd
```

6. **Modify** `/var/http/server1/websites/default_site/conf/default_site.site.conf`:

```
url {
 doc_root      /var/http/demo/public

 map_file      conf/map.conf
 realm_file    conf/realms.conf
 access_file   conf/access.conf
 content_file  conf/content.conf
 mime_file     conf/mime.types

 user_doc_enable  "no"
 user_doc_root    "public_html"
 cgi_enable       "no"
 cgi_dns_enable   "no"
 cgi_suffix_enable "no"
 cgi_user         "nobody"
 log_type         "elf"
 log_prefix       "/var/http/server1/logs/default"
 log_max_files     7
 log_cycle_time    1440
 log_max_file_size  1048576
 ssi_enable       "no"
 ssi_exec         "no"
 ssi_xbithack     "off"
}
```

7. **Create** `/var/http/server1/websites/default_site/conf/mime.types`:

```
application/x-csh   csh
application/x-sh    sh
```

8. **Modify** `/var/http/server1/websites/default_site/conf/map.conf`:

```
map   /cgi-bin/     /var/http/demo/cgi-bin/  cgi
map   /sws-icons/   /var/http/demo/sws-icons/
```

9. **Modify** `/var/http/server1/websites/widgets/conf/widgets.site.conf`:

```
url {
 doc_root      /var/http/widgets/public

 map_file      conf/map.conf
 realm_file    conf/realms.conf
 access_file   conf/access.conf
 content_file  conf/content.conf
 mime_file     conf/mime.types

 user_doc_enable   "yes"
 cgi_enable        "yes"
 cgi_dns_enable    "yes"
 cgi_suffix_enable  "yes"
 cgi_user          "nobody"
 log_type          "clf"
 log_prefix        "/var/http/server1/logs/widgets"
 log_max_files     7
 log_cycle_time    1440
 log_max_file_size  1048576
 ssi_enable        "yes"
 ssi_exec          "yes"
 ssi_xbithack       "full"
}
```

10. Modify `/var/http/server1/websites/widgets/conf/map.conf`**:**

```
map  /cgi-bin/     /var/http/widgets/cgi-bin/  cgi
map  /sws-icons/   /var/http/widgets/sws-icons/
```

▼ Sun WebServer 1.0 `access.acl` to 2.0

1. **Given the following Sun WebServer 1.0** `/etc/http/access.acl` **file and delegated file** `/var/http/widgets/widgets.acl`**:**
   - `/etc/http/access.acl`:

```
url "/sws-administration" {
 authentication_type   md5
 realm                serverAdmin
 password_file         /etc/http/swsadmin.pw
 + user                *
}

url "/statistics" {
 authentication_type   basic
 realm                statsRealm
 password_file        /var/http/demo/stats/usrs
 group_file           /var/http/demo/stats/grps
 + user               *
}

url "//widgets" {
 delegate          /var/http/widgets/widgets.acl
}
```

- `/var/http/widgets/widgets.acl`:

```
url "/" {
 authentication_type   basic
 realm                 widgetsRealm
 password_file         /var/http/widgets/users
 group_file            /var/http/widgets/groups

 + user       *
 - user       Joe
 - group      thoseDenied
}
```

2. **Create a global** `serverAdmin` **realm (in** `/etc/http/realms/`**), and replace its users file with** `/etc/http/swsadmin.pw`

Note - If the realm already exists, then run just the copy command.

```
# htrealm add -r serverAdmin -s HTPASSWD # cp /etc/http/swsadmin.pw /etc/http/realms/serverAdmin/users
```

3. **Create a global** `statsRealm` **and replace its users and groups files with those specified in the** `"/statistics"` **URL above. Add this new realm to the** `realms.conf` **files of both the default site and the widgets site:**

Note - <*hostname*> below refers to the *hostname* of the workstation, which is used to specify the default site.

```
# htrealm add -r statsRealm -s HTPASSWD # cp /var/http/demo/stats/usrs /etc/http/realms/statsRealm/users
```

4. **Create a local *widgetsRealm* at the widgets site and replace its users and groups files with those specified above:**

```
# htrealm add -i server1 -h widgets -r widgetsRealm -s HTPASSWD # cp /var/http/widgets/users \  var/http
```

5. **Modify** `/etc/http/access.conf`:

```
url /sws-administration {
 authentication_type    md5
 realm                 serverAdmin

 + user                *
}
```

6. **Modify** `/var/http/server1/websites/default_site/conf/access.conf`:

```
# Specify /sws-administration ACL here for site administration,
#  create a siteAdmin realm and add administrators to that realm
# url "/sws-administration" {
#  authentication_type   md5
#  realm                siteAdmin
#  + user               *
# }

url "/statistics" {
 authentication_type    basic
 realm                 statsRealm

 + user                *
}
```

7. **Modify** `/var/http/server1/websites/widgets/conf/access.conf`:

```
# Specify /sws-administration ACL here for site administration
url "/statistics" {
 authentication_type    basic
 realm                 statsRealm

 + user               *
}

url "/" {
 authentication_type    basic
 realm                  widgetsRealm

 + user                *
 - user                Joe
 - group               thoseDenied
}
```