Content Server Enterprise Edition

Version: 5.5

Installing CS-Satellite

Document Revision Date: Oct. 31, 2003

FatWire SOFTWARE

CS-Satellite Installation Guide
Document Revision Date: Oct. 31, 2003
Product Version: 5.5

FatWire Technical Support
Web: http://www.fatwire.com/Support

FatWire Headquarters
FatWire, inc.
330 Old Country Road
Suite 207
Mineola, NY 11501
Web: www.fatwire.com

Table of

Contents

Chapter 1

CS-Satellite Configurations

CS-Satellite is a product that works with your CSEE content management system to provide three things:

- An additional layer of caching, supplementing the layer of caching that is provided by the Content Server cache.
- The ability to quickly and economically scale your CSEE system by adding remote installations of CS-Satellite.
- The ability to improve your website’s performance and reduce the load on Content Server by moving content closer to the web site visitors who will view it.

This chapter introduces you to the configurations that you implement in order to recieve these benefits.

You can configure CS-Satellite in three ways:

- Co-Resident, which provides a second layer of caching.
- Co-Resident and Remote Combined, which improves performance and scalability.
- Remote Only (Delivery System), which you implement when you have enough remote CS-Satellites to make the co-resident CS-Satellite unnecessary.

The following sections describe these configurations and what they are used for in greater detail.

Co-Resident

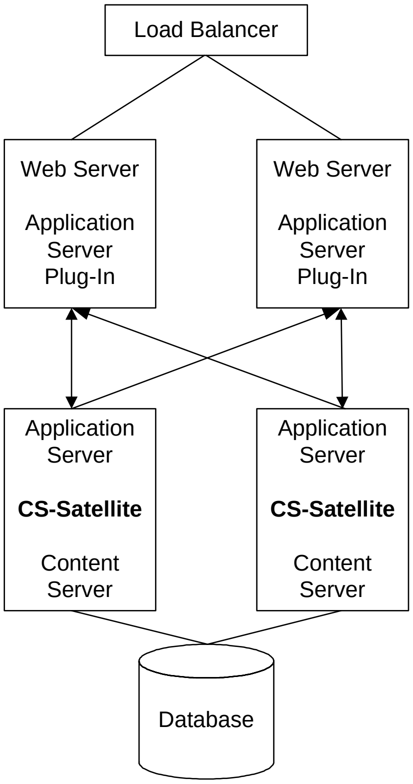
CSEE ships with a copy of CS-Satellite that is installed on the same machine as your Content Server software. This is your **co-resident** CS-Satellite. The co-resident CS-Satellite provides a layer of caching in addition to that provided by Content Server's cache.

You will have a co-resident CS-Satellite on all of your systems: development, management, delivery, and testing.

CS-Satellite and the Content Server cache work in tandem to provide **double-buffered caching**, where copies of cached pages are stored in both the CS-Satellite and the Content Server caches. For more information about double-buffered caching, see the caching chapter of the *CSEE Developer's Guide*.

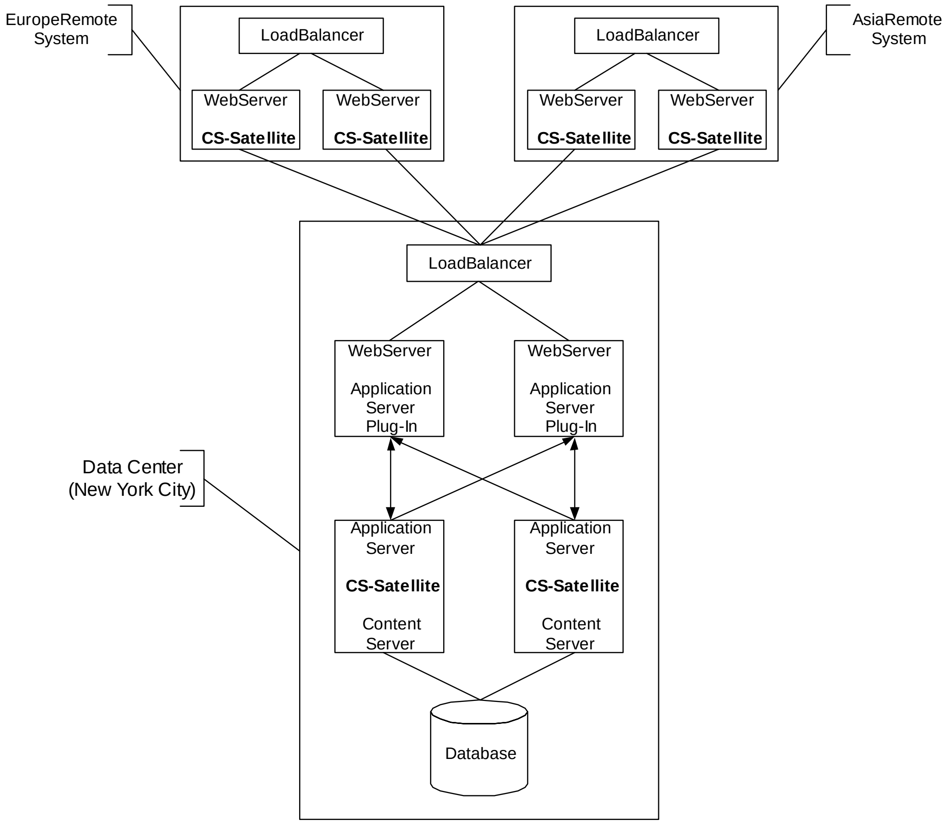
The co-resident CS-Satellite installs automatically when you install CSEE. Co-resident CS-Satellite will run automatically once CSEE is installed on your system. However, FatWire recommends that you tune your co-resident CS-Satellite host to optimize your CSEE system's performance, as described in Chapter 4, "Tuning CS-Satellite."

The following diagram illustrates a co-resident installation of CS-Satellite:



Co-Resident and Remote Combined

In addition to the co-resident CS-Satellite, you can also install remote instances of CS-Satellite as part of your delivery system. The remote CS-Satellite systems are on hardware that is close to the web site's audience, as shown in the following diagram:



Remote installations of CS-Satellite provide several benefits in addition to allowing double-buffered caching:

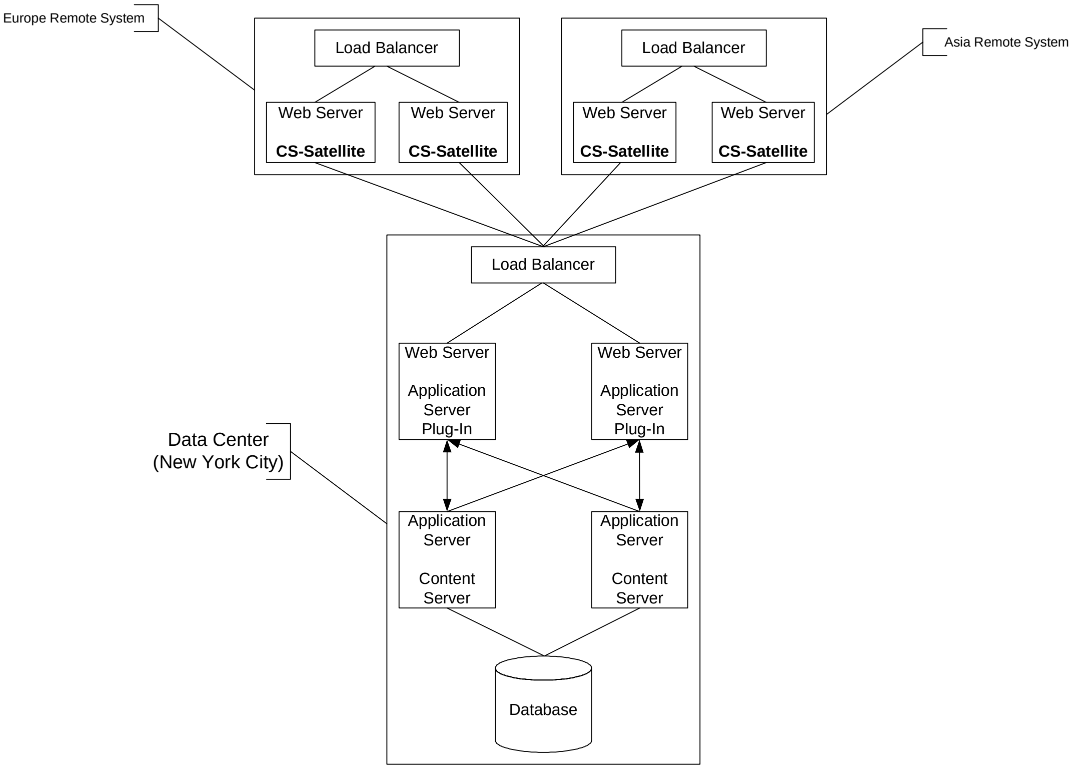
- They improve the performance of the web site by moving the content closer to its audience. In the preceeding diagram, for instance, the main data center is located in New York City, while the
- They remove load from Content Server. Because remote CS-Satellites do not require the same sort of hardware that a full installation of CSEE does, adding them to your CSEE system is a simple and economical way to make CSEE scalable.

For information on how to install and configure remote instances of CS-Satellite, see Chapter 2, "Installing Remote CS-Satellites," and Chapter 4, "Tuning CS-Satellite" in this book.

Remote Only (Delivery System)

At some point, you may add enough remote CS-Satellites to your system that the co-resident installation of CS-Satellite on your delivery system becomes unnecessary. Performance testing allows you to determine when you have reached this point; you will see less and less CPU utilization on your co-resident box, and more CPU utilization on the remote CS-Satellite boxes.

When this occurs, you can disable your co-resident CS-Satellite, creating the following configuration:



For instructions on how to disable your co-resident CS-Satellite on the delivery system, and how to tune for this configuration, see Chapter 4, “Tuning CS-Satellite.”

Chapter 2

Installing Remote CS-Satellites

Co-resident CS-Satellites are installed automatically when CSEE is installed on your hardware. If your system is using co-resident CS-Satellite only, you can skip this chapter and turn to Chapter 4, “Tuning CS-Satellite” for information on how to tune your system.

If you plan to add additional remote CS-Satellites to your system, you must install CS-Satellite on remote boxes. This chapter describes how to do that.

Unlike for a Content Server installation, when you install remote instances of CS-Satellite, you do not need to install an application server, a database, or the JDK. If you plan to use Resin as your web server, everything you need to run CS-Satellite is included in a single archive file. Otherwise, you can install certain web servers to serve your web pages, as described in Chapter 3, “Installing a Web Server for CS-Satellite.”

Note that installing and configuring remote instances of CS-Satellite is an iterative process. You must initially install, configure, and test one remote CS-Satellite, then install, configure, and test your other remote CS-Satellite installations.

After you have completed the installation and initial configuration of your CS-Satellite software, tune each Satellite host to achieve optimum performance. For more information about tuning CS-Satellite, see Chapter 4, “Tuning CS-Satellite.”

To install and configure remote instances of CS-Satellite, you must complete the following steps:

- Step 1: Install Required Hardware and Software
- Step 2: Expand the Installation File
- Step 3: Configure the satellite.ini Properties
- Step 4: Configure the futuretense.ini Properties
- Step 6: (Optional) Install the Web Server
- Step 7: Start CS-Satellite
- Step 8: Test the Configuration

Step 1: Install Required Hardware and Software

Before you install CS-Satellite, be sure that you have the required hardware and software.

FatWire frequently revises the specific software and hardware configurations that are supported by Content Server and CS-Satellite. For the latest information, go to:

`http://cswww.fatwire.com/products/ContentServer`

Networking Requirements

The connection between the CS-Satellite hosts and the Content Server host is the primary gating factor for uncached data. The following table describes the minimum networking requirements:

Connection Between CS-Satellite Hosts and Content Server Hosts	100 Mbps
Connection Between Load Balancer and CS-Satellite Hosts	100 Mbps

You must have a load balancer. FatWire does not require a particular brand of load balancer, but we do recommend that you use a load balancer that supports session affinity.

Configuration Requirements

Your CS-Satellite hosts must meet or exceed the following requirements:

Table 1: Solaris Requirements

Operating System	Solaris 2.6
CPU	Sun Ultra 10 (running at 440 MHz)
Physical Memory	512 Megabytes (1 Gigabyte recommended)
Disk Space	200 Megabytes

Table 2: Windows 2000 Requirements

Operating System	Windows 2000
CPU	Pentium III (running at 500 MHz)
Physical Memory	512 Megabytes (1 Gigabyte recommended)
Disk Space	200 Megabytes

FatWire recommends using a homogeneous set of CS-Satellite hosts. A heterogeneous set of CS-Satellite hosts complicates performance tuning. If you decide to use a heterogeneous set of CS-Satellite hosts, FatWire strongly recommends configuring your load balancer to distribute the load based on the relative strength of each machine.

CS-Satellite Contents

CS-Satellite comes with Resin, a servlet engine, and a Java Runtime Environment. CS-Satellite can serve web pages using Resin as both its web server and its servlet engine. If you want to use a full-featured web server to serve your CS-Satellite pages, however, you can install one of the web servers described in Chapter 3, “Installing a Web Server for CS-Satellite.”

Step 2: Expand the Installation File

The installation file is named `satelliteserver.tar` for Solaris installations and `satserv.exe` for Windows 2000 installations. You extract this file to a host machine.

Windows 2000

The installation file for Windows 2000, `satserv.exe`, is a self-extracting `.zip` file. When you extract it, make sure that you extract the files into a complete folder hierarchy, not into one large folder.

Solaris

You can untar into any target directory. However, for performance reasons, it is better to untar it into a directory on a local partition rather than an NFS-mounted directory on another host. Assuming the `satelliteserver.tar` file is in the current directory, the following command performs a verbose untar:

```
$ tar -xf satelliteserver.tar
```

Extracting `satelliteserver.tar` creates a subdirectory named `SatelliteServer`. Do *not* change the name of this directory or the names of any of its subdirectories.

Step 3: Configure the satellite.ini Properties

Next, you configure the CS-Satellite properties that are located in each remote CS-Satellite’s property file, `satellite.ini`.

To configure the `satellite.ini` properties, complete the following steps:

1. Invoke the Property Editor for `satellite.ini`, by running one of the following scripts:
   - For a Solaris installation: `settings.sh` from the ssinstall folder
   - For a Windows installation: `settings.bat` from the ssinstall folder

   The Property Editor appears.
2. Edit the **host** property. The host property sets the host name on which Content Server is running. This is a required property. There is no default value.

The host name that you specify is different depending upon the configuration that you choose. The following table contains host settings for each remote CS-Satellite configuration:

Configuration	Property Value for Host
Remote Only	URL of the data center's load balancer.
Combination	• For the co-resident CS-Satellite installations, leave the value set to `localhost`. • For the remote CS-Satellite installations, set `host` to the URL of the data center's load balancer

3. Edit the **port** property. This property specifies the port number on which Content Server is running. The default value is `80`.
4. Edit the **username** and **password** properties.

   The **username** and **password** properties specify a login name and password combination for this instance of CS-Satellite. By default, the login name and password are both set to `ftuser`.

   For security reasons, set the **username** and **password** properties to some combination that is difficult to guess.

   Note the username and password that you set for each Satellite Server host; you will need this information to configure the `cs.satelluteusers` and `cs.satellitepassword` properties in the `futuretense.ini` file.
5. Edit the **service** property.

   This property sets the pathname portion of the Content Server URL. On iPlanet, set the `service` property to the following value:

   ```
   /NASApp/cs/ContentServer
   ```

   On other application servers, including Sun ONE, set the property to the following value:

   ```
   /servlet/ContentServer
   ```
6. Edit the **bservice** property.

   This property sets the pathname portion of the BlobServer URL for retrieving blobs. On iPlanet, set the `bservice` property to the following value:

   ```
   /NASApp/cs/BlobServer
   ```

   On other application servers, including Sun ONE, set the property to the following value:

   ```
   /servlet/BlobServer
   ```

Step 4: Configure the futuretense.ini Properties

You must now configure the CS-Satellite properties that are located in Content Server's property file, `futuretense.ini`. This file is located in the Content Server installation directory on the machine that hosts Content Server.

To configure the `futuretense.ini` properties, complete the following steps:

1. Invoke the Property Editor and open `futuretense.ini`. For instructions on how to do this, see the properties section of the *CSEE Administrator's Guide*.

Note

On Solaris, you must be running an X Windows server. If you are not, the Property Editor does not appear on the screen.

2. Edit the `cs.satellitehosts` property. `cs.satellitehosts` is a comma-seperated list of the hostnames of all of your CS-Satellite installations, both co-resident and remote.

   The value for each host must include the path to the Content Server servlets.

   Use the following format:

   ```
   http://hostname:port/servlet/
   ```

   You can use https or special ports, if necessary. If required by your configuration, be sure to specify a fully qualified domain name.

   The Satellite host that resides on the Content Server box is listed by default.

3. Edit the `cs.satelliteusers` property.

   `cs.satelliteusers` is a comma-separated list of user names for each of the CS-Satellite hosts that you listed above.

   Note that the usernames that you specify here must be in the same order as the hosts that you specified in the `cs.satellitehosts` property; for example:

   ```
   cs.satellitehosts: Host1,Host2,Host3
   cs.satelliteusers:Host1Username,Host2Username,Host3Username
   ```

   The user name for the Satellite servlet on the Content Server box is listed by default.

4. Edit the `cs.satellitepassword` property.

   `cs.satellitepassword` is a comma-separated list of passwords for the CS-Satellite hosts specified in the `cs.satellitehosts` property.

   The passwords must be listed in the same order as the hosts specified in the `cs.satellitehosts` property, and also in the same order as the usernames specified in the `cs.satelliteusers` property. For example:

   ```
   cs.satellitehosts:Host1,Host2,Host3
   cs.satelliteusers:Host1Username,Host2Username,Host3Username
   cs.satellitepassword: Host1Password,Host2Password,Host3Password
   ```

   Note that the value of this property is encrypted as a single string. Therefore, when you edit the value of this property, you must enter all the passwords for all the Satellite servlet hosts, including the comma delimiter.

The password for the Satellite host on this server is listed by default.

5. Edit the `satellite.blob.cachecontrol.default` property.

   `satellite.blob.cachecontrol.default` specifies a default value for the cachecontrol parameter for the `satellite.blob` and `RENDER.SATELLITEBLOB` tags and their JSP equivalents.

   Set this property to a value that is appropriate for the majority of your blobs, and then use the cachecontrol parameter with the `satellite.blob` and `RENDER.SATELLITEBLOB` tags to override this value for individual blobs.

   Use the following format to set a value:

   ```
   hours:minutes:seconds daysOfWeek/daysOfMonth/months
   ```

   For more information about this format, see the *CSEE Administrator's Guide*.

6. Edit the `satellite.page.cachecontrol.default` property.

   `satellite.page.cachecontrol.default` specifies a default value for the `cachecontrol` parameter for the `satellite.page`, and `RENDER.SATELLITEPAGE` tags and their JSP equivalents.

   Set this property to a value that is appropriate for the majority of your pages and pagelets, and then use the `cachecontrol` parameter with the `satellite.page` and `RENDER.SATELLITEPAGE` tags to override this value for individual pages and pagelets.

   Use the following timepattern to set a value:

   ```
   hours:minutes:seconds daysOfWeek/daysOfMonth/months
   ```

   For more information about this timepattern, see the *CSEE Administrator's Guide*.

7. Edit the `sharesession` property. `sharesession` specifies whether the ContentServer servlet and the Satellite servlet share a user session.

   Set this value to `false` for remote hosts.

Step 5: Configure Resin

CS-Satellite comes with a servlet engine called Resin. The Satellite servlet, which is the heart of CS-Satellite, runs on Resin. Resin listens for requests on a configurable port, and passes those requests on to the Satellite servlet based on the URL being requested.

Configure the HTTP Port

By default, Resin listens for requests on port 80. If you are using Resin as your web server and if port 80 is already being used, or if you are installing multiple CS-Satellites on one machine, you must change the port that Resin listens on.

To modify the port Resin listens on, change the line in the `resin.conf` file (located at *`install_directory/`*`SatelliteServer/resin1.2.3/conf/resin.conf)` from:

```
<http port = ‘80’/>
```

to

```
<http port = ‘portNumber’ />
```

where *portNumber* is the number of the port that you want Resin to listen on.

After modifying the port number, delete the following line:

```
<srun host = ‘127.0.0.1’ port = ‘6802’/>
```

Configure the Host Port

If you are using something other than Resin as your web server, Resin acts as a plug-in that communicates with the web server. By default, the port that Resin uses to communicate with the web server is 6802. If port 6802 is already in use, you must change the port that Resin communicates on.

To change the port that Resin communicates on, change the following line in the `resin.conf` file from:

```
<srun host = ‘127.0.0.1’ port = ‘6802’/>
```

to

```
<srun host = ‘127.0.0.1’ port = ‘portNumber’ />
```

where *portNumber* is the number of the port you want Resin to listen on.

After modifying the port number, delete the following line:

```
<http port = ‘80’/>
```

You do not have to modify your web server’s configuration file because both Resin and the web server get the port number from the `resin.conf` file.

Step 6: (Optional) Install the Web Server

If you are using something other than Resin as your web server, you need to install your web server now. For instructions on installing and configuring your web server with CS-Satellite, see Chapter 3, “Installing a Web Server for CS-Satellite.”

After you are done installing the web server, return to step 7.

Step 7: Start CS-Satellite

Now you must start CS-Satellite.

Starting CS-Satellite on Windows 2000

To start CS-Satellite on a Windows 2000 system, run the `server.bat` file.

CS-Satellite logs information to standard output. By default, standard output appears in the shell window. You can redirect standard output by invoking the batch file as follows:

```
C:\SatelliteServer\> server.bat > satellite_server.log
```

Starting CS-Satellite on Solaris

There are two ways to start CS-Satellite if you are using a Solaris machine:

If Resin is Your Web Server

If you are using Resin as your web server, start CS-Satellite by starting up any UNIX shell and invoking the shell script stored at the following pathname:

```
$ cd $SatelliteServerRoot/SatelliteServer
$ server.sh
```

CS-Satellite logs information to standard output. By default, standard output goes directly into the shell window. You can redirect standard output by invoking the batch file as follows:

```
$ server.sh > satellite_server.log&
```

If Resin is Not Your Web Server

If you are using something other than Resin as your web server, start CS-Satellite by starting up any UNIX shell and invoking the following shell script:

```
$ plugin.sh
```

Step 8: Test the Configuration

Before you install CS-Satellite on other machines, test the first CS-Satellite machine to make sure that it is communicating properly with Content Server.

To test your configuration:

1. Configure your load balancer to send all Content Server requests to the first CS-Satellite machine. The Resin web server running on the CS-Satellite is, by default, listening on port 80.

   If you are using Resin as your web server, make sure that no other process is using the same port as Resin. You can determine the port on which Resin is running by examining the configuration file at the following pathname:

   `$SatelliteServerRoot/``SatelliteServer/resin1.2.3/conf/resin.conf`

2. Using a browser, go to a CS-Satellite URL. For example:
   `http://`*`yourhost`*`:`*`yourport`*`/servlet/Satellite?pagename=`*`MyPage`*

3. If you configured everything properly, then the following should occur:

- Your browser displays the selected page.
- The *`$SatelliteServerRoot/`*`SatelliteServer/cache` directory on your CS-Satellite host machine should contain at least one file that holds the HTML for the page that you selected.

If your browser did not display the selected page, review the following:

- *Did you set up the load balancer properly?* Remember, for this test, every request for Content Server has to go to the CS-Satellite machine. (The other machines haven’t been set up yet, so they will not know how to handle these requests.)

- *Did you set the CS-Satellite properties properly?* In particular, make sure that you set the host and port to the proper values.
- *Did you request an invalid Content Server page?*

Step 9: Install CS-Satellite on Additional Remote Boxes

After you have installed and tested CS-Satellite on your first remote machine, you must install and configure CS-Satellite on your other remote machines.

If your CS-Satellite boxes are not homogeneous, you will have to install and configure the software for each box individually, by following steps 1 through 8 of the installation procedure.

If your machines are homogeneous, however, you can expedite the installation process by copying the `satellite.ini` and `resin.conf` files that you modified for your initial CS-Satellite host to your other CS-Satellite hosts. To accomplish this, complete the following steps:

1. Make a copy of the first remote machine's `satellite.ini` and `resin.conf` files.
2. Install all of your other remote instances of CS-Satellite, as detailed in "Step 2: Expand the Installation File".
3. If necessary, install the web server for each remote installation, as described in Chapter 3, "Installing a Web Server for CS-Satellite." Replace the `satellite.ini` and `resin.conf` files in each remote installation with your copies of `satellite.ini` and `resin.conf`.
4. (Optional) Open the Property Editor for each `satellite.ini` file and set the username and password properties. Note each username and password pair, as well as the name of the remote CS-Satellite host.
5. Add the new hostnames, usernames, and passwords to the `futuretense.ini` file.
6. Test your remote CS-Satellites.

Chapter 3

Installing a Web Server for CS-Satellite

This chapter describes how to install and configure a web server for use with CS-Satellite software.

This chapter contains the following sections:

- Installing the iPlanet Web Server on Solaris
- Configuring the iPlanet Web Server on Solaris
- Installing the iPlanet Web Server on Windows 2000
- Configuring the iPlanet Web Server on Windows 2000
- Installing the Apache Web Server on Solaris
- Configuring the Apache Web Server on Solaris
- Configuring Microsoft Internet Information Server (IIS) on Windows 2000

Installing the iPlanet Web Server on Solaris

For more information on installing iPlanet, see the iPlanet Web Server documentation.

To install the iPlanet Web Server for use with CS-Satellite:

1. Log in as **`root`**.
2. Insert the iPlanet Web Server installation CD in your CD drive.
3. Change to the CD-ROM directory and copy `entprise.tar` to your home directory or a temporary directory.
4. Navigate to the directory where you copied `entprise.tar` and untar the file.
5. Type **`./setup`** and press **Enter**. The iPlanet Web Server installation begins.
6. A welcome screen appears. Press **Enter** to continue with the installation.
7. Type **`Yes`** to accept the license agreement, then press **Enter**.

8. Choose **Typical** as the Installation Type**,** then press **Enter**.
9. Enter **`/export/home/netscape/suitespot`** as the server root directory**,** then press **Enter**.
10. Choose **iPlanet Web Server, all components**, and press **Enter**.
11. Press **Enter** to choose the default subcomponents.
12. Enter your machine's name, or press **Enter** to accept the default.
13. Enter the UNIX username and group to use when running iPlanet. The default user and group is **`nobody`**. If you are going to use SNMP, however, the user name must be **`root`**. Once you have chosen the user and group names**,** press **Enter.**
14. Enter the UNIX user name to use when running the Administration Server, or press **Enter** to accept the default.
15. Enter a Server Administration ID and password for your system.

> **Note**
>
> The default Server Administration ID and password is `admin`.

16. Enter **`11111`** as the Administration Port; then press **Enter**.
17. Type the port number of the iPlanet Web server; then press **Enter**.
18. Enter **`No`** at the LDAP-based server prompt; then press **Enter**.
19. Press **Enter** to accept the default settings for Java support.
20. The installation program extracts and installs the iPlanet Web server. After installation is complete, press **Enter**.

Configuring the iPlanet Web Server on Solaris

To run CS-Satellite with the iPlanet Web Server, you must configure the server to work with the Resin plug-in. Complete the following steps to configure your web server:

Step 1: Install and Configure the CS-Satellite Software

You must install and configure the CS-Satellite software before configuring the iPlanet Web Server. See Chapter 2, "Installing Remote CS-Satellites," for instructions on installing and configuring CS-Satellite.

Step 2: Modify the iPlanet Configuration File

Find the iPlanet configuration file, `obj.conf`, and add the following lines:

```
Init fn="load-modules" shlib="/path_to_Satellite_Server_install/
SatelliteServer/resin-1.2.3/bin/caucho_nsapi.so/"
funcs="caucho_service,caucho_filter,caucho_status"
```

```
<Object name="default">
NameTrans fn="caucho_filter" conf=”/
path_to_Satellite_Server_install/SatelliteServer/resin-1.2.3/conf/
resin.conf” name=”resin”
NameTrans fn="assign-name" from="/caucho-status” name=”caucho-
status”
</Object>
<Object name=”resin”>
Service fn=”caucho_service”
</Object>
```

Step 3: Test the Configuration

To verify the installation and configuration:

1. Install the Portal example.
2. Start the iPlanet Web Server.
3. Start Resin by running `plugin.sh` (on a Solaris system).
4. Use a browser to view the following URL:

   `http://myhost:myport/servlet/Satellite?pagename=OpenMarket/Samples/Portal/main`

If the front page of the Portal site appears on the screen, the installation and configuration is successful.

Installing the iPlanet Web Server on Windows 2000

To install the iPlanet Web Server on Windows 2000:

1. Insert the iPlanet Web Server installation CD in your CD drive.
2. Change to the CD ROM directory and copy `entprise.exe` to a temporary directory.
3. Click `enterprise.exe`. The iPlanet installation begins.
4. A welcome screen appears. Press **Enter** to continue with the installation.
5. Click **Yes** to accept the license agreement, then press **Enter**.
6. Choose **Typical** as the Installation Type**,** then press **Enter**.
7. Enter **`/export/home/netscape/suitespot`** as the server root directory, then press **Enter**.
8. Choose **iPlanet Web Server, Enterprise Edition**, then press **Enter**.
9. Enter a Server Administration ID and password for your system.

   Note

   The default Server Administration ID and password is `admin`.

10. Enter **`11111`** as the Administration Port, then press **Enter**.

11. Type the port number of the iPlanet Web Server, then press **Enter**.
12. Press **Enter** to accept the default settings for Java support.
13. Click **Install** to install the web server.

Configuring the iPlanet Web Server on Windows 2000

To run CS-Satellite with the iPlanet Web Server, you must configure the server to work with the Resin plug-in. Complete the following steps to configure your web server.

Step 1: Install and Configure the CS-Satellite Software

Because the iPlanet Web server works with the Resin servlet engine, which is included with the CS-Satellite software, you must install and configure the CS-Satellite software before configuring the iPlanet Web Server. See Chapter 2, “Installing Remote CS-Satellites,” for instructions on installing and configuring CS-Satellite.

Step 2: Configure Resin for the Web Server

To configure Resin to work with the iPlanet Web Server:

1. Run `setup.exe`, found in `Satellite_Server_installation_directory/SatelliteServer/resin1.2.3/bin`.
2. Select the Netscape check box, and enter the path to the web server in the adjoining field.
3. Click **OK**.

Step 3: Modify the iPlanet Configuration File

Find the iPlanet configuration file, `obj.conf`, and add the following lines:

```
Init fn="load-modules" shlib=”/path_to_Satellite_Server_install/
SatelliteServer/resin-1.2.3/bin/caucho_nsapi.so/"
funcs="caucho_service,caucho_filter,caucho_status"

<Object name="default">
NameTrans fn="caucho_filter" conf=”/
path_to_Satellite_Server_install/SatelliteServer/resin-1.2.3/conf/
resin.conf” name=”resin”
NameTrans fn="assign-name" from="/caucho-status” name=”caucho-
status”
</Object>
<Object name=”resin”>
Service fn=”caucho_service”
</Object>
```

Step 4: Test the Configuration

To verify that the installation and configuration was successful:

1. Install the Portal example.
2. Start the iPlanet Web Server.
3. Start Resin by running `plugin.sh` (on a Solaris system) or `plugin.bat` (on a Windows 2000 system).
4. Use a browser to view the following URL:

   ```
   http://myhost:myport/servlet/Satellite?pagename=OpenMarket/
   Samples/Portal/main
   ```

   If the front page of the Portal site appears on the screen, the installation and configuration is successful.

Installing the Apache Web Server on Solaris

The modules that make up the Apache Web Server are located at the Apache Software Foundation web site:

```
www.apache.org
```

Download, compile, and install the Apache Web Server as described in the Apache documentation.

Configuring the Apache Web Server on Solaris

To run the Apache Web Server with CS-Satellite, you must configure the web server to run with the Resin plug-in. Complete the following steps to configure the Apache Web Server.

Step 1: Install and Configure the CS-Satellite Software

Because the Apache Web Server works with the Resin servlet engine, which is included with the CS-Satellite software, you must install and configure the CS-Satellite software before configuring the Apache Web Server. See Chapter 2, “Installing Remote CS-Satellites,” for instructions.

Step 2: Modify the Apache Configuration File

Find the Apache configuration file, `httpd.conf`, and add the following lines:

```
LoadModule caucho_module "path_to_Satellite_installation/
SatelliteServer/resin-1.2.3/bin/mod_caucho.so"

AddModule mod_caucho.c
<IfModule mod_caucho.c>
  CauchoConfigFile "path_to_Satellite_installation/SatelliteServer
  /resin-1.2.3/conf/resin.conf"
</IfModule>
```

Step 3: Test the Configuration

To verify the installation and configuration:

1. Install the Portal example.
2. Start the Apache Web Server.
3. Start Resin by running `plugin.sh.`
4. Use a browser to view the following URL:

   ```
   http://myhost:myport/servlet/Satellite?pagename=OpenMarket/
   Samples/Portal/main
   ```

   If the front page of the Portal site appears on the screen, the installation and configuration is successful.

Configuring Microsoft Internet Information Server (IIS) on Windows 2000

Microsoft Internet Information Server (IIS) is installed with Windows 2000. Complete the steps in the following sections to configure IIS to work with CS-Satellite.

Step 1: Install and Configure the CS-Satellite Software

Because IIS works with the Resin servlet engine, which is included with the CS-Satellite software, you must install and configure the CS-Satellite software before configuring IIS. See Chapter 2, “Installing Remote CS-Satellites,” for instructions on installing and configuring CS-Satellite.

Step 2: Configure Resin to Run With IIS

To configure Resin to work with IIS:

1. Run `setup.exe`, located in `Satellite_Server_installation_directory/SatelliteServer/resin1.2.3/bin`.
2. Select the `IIS/IWS` check box, and enter the path to the web server in the adjoining field.
3. Click **OK**.

Step 3: Test the Configuration

To verify the installation and configuration:

1. Install the Portal example.
2. Start IIS.
3. Start Resin by running `plugin.bat.`
4. Use a browser to view the following URL:

   ```
   http://myhost:myport/servlet/Satellite?pagename=OpenMarket/
   Samples/Portal/main
   ```

If the front page of the Portal site appears on the screen, the installation and configuration is successful.

Chapter 4

Tuning CS-Satellite

After you have installed your CS-Satellite hosts (or, in the case of the co-resident CS-Satellite host, the host has been installed along with CSEE), you need to tune them in order to achieve the best performance on your CSEE system.

This chapter explains how to tune your CS-Satellite hosts. It contains the following sections:

Tuning the Co-Resident CS-Satellite Host

CS-Satellite stores pages both in memory and on disk. In the case of the co-resident CS-Satellite host, this means that the CS-Satellite shares memory with your Content Server installation.

To achieve optimum performance on a system with co-resident CS-Satellite, you should adjust the `file_size` property, located in the `satellite.ini` file on the Content Server host.

`The file_size` property separates disk-cached pagelets and blobs from memory-cached pagelets and blobs. To set the `file_size` property, specify a size in kilobytes. CS-Satellite caches any pagelet or blob larger than this size to disk, and caches any pagelet or blob smaller than this size to memory.

Setting `file_size` to `0` instructs CS-Satellite to cache all pagelets and blobs to disk. Setting `file_size` to a large number (for example, `1,000,000`) instructs CS-Satellite to cache all pagelets and blobs to memory. The appropriate setting for your system will be somewhere in between these two extremes.

To determine the proper setting for your system, experiment with values for this property, watching the memory usage on both Content Server and CS-Satellite with each alteration.

Your goal is to adjust the property so that CS-Satellite stores as many items as possible in memory, while still allowing Content Server enough memory to run quickly.

Tuning Remote CS-Satellite Hosts

Because they do not share hardware or memory with your installation of Content Server, you tune your remote CS-Satellite hosts differently than you would the co-resident host.

The following sections provide some tuning guidelines.

Tuning Homogeneous CS-Satellite Hosts

If every CS-Satellite host has the same CPU, the same amount of physical memory, and the same amount of disk space, then each CS-Satellite should have the same set of properties.

In order to determine the appropriate settings for your system, run performance tests while you experiment with various property values, noting which changes improve performance.

The following property values have an especially large impact on performance and should be tuned carefully:

- `readtimeout`
- `blocktimeout`
- `cache_folder`
- `file_size`
- `expiration`
- `cache_check_interval`
- `cache_max`
- `cache_debug`

For more information about these properties, see "Satellite.ini Properties" on page 29.

For a complete listing of all of the CS-Satellite properties, see the *CSEE Administrator's Guide*.

After you have found the best settings for your system, you can copy the modified `satellite.ini` file to your other homogeneous remote CS-Satellite hosts, as described in "Step 9: Install CS-Satellite on Additional Remote Boxes" on page 17.

Tuning Heterogeneous CS-Satellite Hosts

If your remote CS-Satellite hosts have different strengths, consider adjusting the various caching parameters and your harware configuration.

For example, if one host has significantly more physical memory than the others, then you might consider increasing the value of the `file_size` property to increase the number of pagelets that get cached in memory.

Evaluate each of the properties listed in the Tuning Homogeneous CS-Satellite Hosts section of this chapter, as their optimum values will differ with the differing hardware of each host.

You can also improve performance by tuning your hardware to take advantage of machines with more memory and processing power. To do this, configure your load balancer to send more requests to "stronger" hosts, and fewer requests to the hosts with less power and less memory.

Satellite.ini Properties

The properties described in this section are those that have the greatest impact on performance, and are the ones that you are most likely to tune. For a complete list of CS-Satellite properties, see the *CSEE Administrator's Guide.*

readtimeout

Use the `readtimeout` property to specify a timeout period. The timeout period is the number of milliseconds that CS-Satellite waits for Content Server to fulfill a request. For example, `readtimeout` is set to 3000 (3-second wait time). CS-Satellite requests a pagelet, but because of some network problem, Content Server fails to respond within three seconds. In this case, CS-Satellite writes a message to the CS-Satellite startup window.

Setting `readtimeout` to `0` (the default) means that the Java Runtime Environment establishes the timeout period.

blocktimeout

Use the `blocktimeout` property to specify the number of seconds a request waits when another thread is in the process of requesting the same data from the host. Waiting for one thread to return helps reduce the load on the host server when the cache is empty; however, the benefit of a reduced load comes at the expense of individual user response time.

The default value is `45`. A value of `-1` means to wait until the previous thread returns. A value of `0` means to never wait.

This value must be tuned based on the host performance, average request size, and network latency. It is safe to use a large number or `-1`.

cache_folder

Use this property to specify the directory into which CS-Satellite caches pagelets to disk. The default directory is:

```
$SatelliteServerRoot/SatelliteServer/satellite/cache
```

You can specify only one directory. The directory that you specify is not required to be on the same drive as `/SatelliteServer`. FatWire recommends that it is the same drive to improve performance.

file_size

Use this property to separate disk-cached pagelets and blobs from memory-cached pagelets and blobs. You specify a size (in kilobytes). CS-Satellite caches to disk any pagelet or blob larger than this size and caches to memory any pagelet or blob smaller than this size. For example, you set `file_size` to 4. CS-Satellite caches to memory any pagelets smaller than 4 kilobytes and caches to disk any pagelets 4 kilobytes or larger.

To optimize CS-Satellite performance, FatWire recommends that you experiment with this property.

Setting `file_size` to `0` instructs CS-Satellite to cache all pagelets and blobs to disk. Setting `file_size` to a large number (for example, `1,000,000`) instructs CS-Satellite to cache all pagelets and blobs to memory. If you have a large amount of memory or a relatively small web site, FatWire recommends caching everything to memory.

The `file_size` property can significantly influence performance. To optimize performance, maximize the amount of memory caching. Be careful not to exceed the host's memory capacity.

expiration

The `expiration` property sets the default expiration time from for pages, pagelets, and blobs when a cache expiration value is not specifically set for that item with the `RENDER.SATELLITEPAGE` or `RENDER.SATELLITEBLOB` tag that generated the item.

If you are using the Satellite servlet to cache at the pagelet level, you can override this property on a pagelet-by-pagelet basis by using Render tags. The containing page expires at the default setting. Note, however, that you **cannot** use Satellite tags to modify the default expiration time for the containing page.

Setting `expiration` as follows tells CS-Satellite that pagelets and blobs should never expire for time reasons:

```
never
```

Such objects are not guaranteed to stay in the cache indefinitely. For example, if the cache is full, CS-Satellite still removes pages and pagelets on an based on a LRU (least recently used) algorithm.

Setting `expiration` as follows tells CS-Satellite not to cache pages, pagelets, or blobs at all:

```
immediate
```

If you are not caching your containing pages, set the `expiration` property to `immediate`, and override this value for individual pagelets, cookies, and blobs by setting the `cachecontrol` parameter for every pagelet, cookie, and blob in your site, or by setting the default expiration properties in `futuretense.ini`.

To set a specific set of expiration dates and times, assign a string that uses the following format for the `expiration` property:

```
hh:mm:ss W/DD/MM
```

where:

Parameter	Legal Values	Description
hh	0-23	The hour. For example, `0` means midnight, `12` means noon, `15` means three in the afternoon, and so on.
mm	0-59	The number of minutes past the hour.
ss	0-59	The number of seconds past the minute.
W	0-6	The day of the week. For example, `0` means Sunday, `1` means Monday, and so on.
DD	1-31	The day of the month.

Parameter	Legal Values	Description
MM	1-12	The month of the year. For example, 1 means January, 2 means February, and so on.

For example, the following expiration value means 3:30 in the afternoon every Monday and on the 15th of April:

```
15:30:00 1/15/4
```

If you specify a value for both *W* and *DD*, both values apply. Thus, pages expire on Monday (the *W* field) and on the 15th (the *DD* field). To indicate a day-of-week expiration only, place an asterisk in the *DD* field. For example, to indicate expiration at 3:30 in the afternoon every Monday in April, set the expiration value to:

```
15:30:00 1/*/4
```

To indicate a day-of-month expiration only, place an asterisk in the *W* field. For example, to indicate expiration at 3:30 in the afternoon on April 15, set the expiration value to:

```
15:30:00 */15/4
```

Setting the *hh*, *mm*, *ss*, or *MM* fields to an asterisk means all legal values. For example, to indicate expiration at 3:30 in the afternoon on Mondays and the 15th of **every** month, set the expiration value to:

```
15:30:00 1/15/*
```

You can also place multiple values for any of the six fields by separating the values with commas. To represent a range of values, use a minus sign. For example, the following expiration value represents 6:00 (morning), 1:00 (afternoon), and 5:00 (afternoon), Monday through Friday in June.

```
6,13,17:00:00 1-5/*/6
```

To indicate that pages must expire every 15 minutes, set the expiration value to the following:

```
*:15,30,45:0 */*/*
```

The default value is:

```
5:0:0 */*/*
```

This means that everything in the CS-Satellite cache expires every day at 5:00 a.m.

cache_check_interval

When a disk-cached page expires, CS-Satellite does not immediately delete the page from the disk. Instead, CS-Satellite removes this page from its list of active pages. CS-Satellite does, however, contain a cache-pruning thread that runs periodically and deletes expired objects from the cache. Use the `cache_check_interval` property to define the period (in minutes) at which the cache-pruning program should run. The default value is `3600`, meaning that the cache-pruning program runs every 60 hours.

Do not set the `cache_check_interval` value too low; the cache-pruning program consumes a significant amount of resources. However, do not set `cache_check_interval` so high that your disk drive or memory fills up with expired pages.

Note

CS-Satellite never serves expired pages. If a page is expired but is still in the cache, CS-Satellite does not serve it.

cache_max

Use this property to specify the maximum number of objects (pagelets and blobs) that can be cached (memory cache and disk cache combined) at a time. The default value is `500`, meaning that CS-Satellite caches up to 500 objects at a time.

CS-Satellite uses an LRU (Least Recently Used) algorithm to determine which pages must be removed from cache when the cache maximum is exceeded. For example, set the `cache_max` to `1000`. When CS-Satellite receives a request to cache the 1001st object, CS-Satellite removes the object that has not been used in the longest time.

Although you should set `cache_max` to a high level, note that each entry in CS-Satellite's cache table consumes some memory. Also, note that setting `cache_max` to a very high value causes the cache-pruning program to take a longer time to run.

cache_debug

If `cache_debug` is set to `true`, CS-Satellite prints cache status messages to the Java console. These messages can be useful in debugging. If `cache_debug` is set to `false` (the default), CS-Satellite does not log cache status messages.

Index