SIOS Protection Suite for Linux v8.3.1

Apache Web Server Recovery Kit Administration Guide

September 2014

To maintain the quality of our publications, we welcome your comments on the accuracy, clarity, organization, and value of this document.

Address correspondence to:
ip@us.sios.com

Table of Contents

Apache Recovery Kit Administration Guide

Introduction

The LifeKeeper® for Linux Apache Web Server Recovery Kit provides fault resilience for Apache Web Server software in a LifeKeeper environment.

Document Contents

This guide explains the following topics:

- **LifeKeeper Documentation.** A list of all the LifeKeeper for Linux documentation and where the information is available.
- **Requirements.** Before you can install and set up the recovery software, your server must meet certain hardware and software requirements. You should refer to the *SPS Installation Guide* for specific instructions on how to install or remove the LifeKeeper Apache Recovery Kit.
- **Configuring Your Recovery Kit.** To ensure that your LifeKeeper configuration provides the protection and flexibility you require, you need to be aware of the configuration rules. To appropriately plan your configuration, you must understand your network configuration, interface selection, user system setup, hierarchy options and the Apache configuration tasks. In addition to planning your configuration, this section also includes configuration examples and the specific tasks required to configure your Recovery Kit.
- **Troubleshooting.** This section provides a list of informational and error messages with recommended solutions.

LifeKeeper Documentation

The following is a list of LifeKeeper related information available from SIOS Technology Corp.:

- *SPS for Linux Release Notes*
- *SPS for Linux Technical Documentation* (available from the Help menu within the LifeKeeper GUI)
- *SPS for Linux Installation Guide*

This documentation, along with documentation associated with other LifeKeeper Recovery Kits, is provided online at:

http://docs.us.sios.com

Reference Documents

The following is a list of reference documents associated with the Apache Web Server application and the LifeKeeper Apache Recovery Kit:

- Apache Online documentation
- Apache: The Definitive Guide, 2nd Edition, Ben Laurie and Peter Laurie, O'Reilly & Associates, Inc. 1999

Requirements

Before attempting to install or remove the Apache Recovery Kit, you must understand the hardware and software requirements for the package and the installation and removal procedures.

Kit Hardware and Software Requirements

Before installing and configuring the LifeKeeper Apache Recovery Kit, be sure that your configuration meets the following requirements:

- **Servers.** The Recovery Kit requires two or more *supported* computers configured in accordance with LifeKeeper requirements described in the *SPS for Linux Technical Documentation* and the *SPS Release Notes*, which are located on our SIOS Technical Documentation site at docs.us.sios.com.
- **LifeKeeper software.** You must install the same version of LifeKeeper software and any patches on each server. Please refer to the *SPS Release Notes* and *SPS for Linux Technical Documentation* for specific LifeKeeper requirements.
- **LifeKeeper IP Recovery Kit (Optional).** You must have the same version of this Recovery Kit on *each* server.
- **LifeKeeper Recovery Kit for EC2™ (Optional).** If you create an Apache resource hierarchy in an Amazon EC2™ instance, you must have the same version of this Recovery Kit on each server.
- **IP network interface.** Each server requires at least one Ethernet TCP/IP-supported network interface. In order for IP switchover to work properly, user systems connected to the local network should conform to standard TCP/IP specifications.
  Note: Even though each server requires only a single network interface, you should use multiple interfaces for a number of reasons: heterogeneous media requirements, throughput requirements, elimination of single points of failure, network segmentation, and so forth.
- **TCP/IP software.** Each server also requires the TCP/IP software.
- **Apache software.** Each server must have the Apache Web Server software installed and configured prior to configuring LifeKeeper and the LifeKeeper Apache Web Server Recovery Kit, including any DSO (Dynamic Shared Object) modules that will be used. The same versions of all web server software packages should be installed on each server. Consult the *SPS Release Notes* or your sales representative for the latest release compatibility and ordering information.

Refer to the *SPS Installation Guide* for specific instructions on how to install or remove the LifeKeeper Apache Recovery Kit.

Configuring Apache Web Server with LifeKeeper

This section contains definitions and examples of typical LifeKeeper Apache Web Server configurations and information you should consider before you start to configure Apache Web Server.

Please refer to the *SPS for Linux Technical Documentation* for instructions on configuring LifeKeeper Core resource hierarchies.

Configuration Definitions and Examples

Apache Web Server supports multiple instances of the **httpd** daemon running at the same time. Each LifeKeeper Apache Web Server hierarchy corresponds to a separate Apache instance with its own "server root" directory. Each instance may support one or more web sites, depending on whether or not it has been configured to use "virtual hosts."

Primarily, the server root directory defines an Apache Web Server instance, since this directory will contain the *conf/httpd.conf* configuration file that specifies how the web instance is configured. The Apache configuration directives within this file will determine where the log files, web documents, other configuration files, etc. are located for the instance, as well as which IP and/or domain name addresses will be used.

It is useful to characterize Apache Web Server configurations with LifeKeeper based on whether or not a LifeKeeper file system (which uses shared storage) will be used. A single shared file system may be used for the server root directory (along with the configuration file *conf/httpd.conf*) and/or the document root directories (and optionally the **httpd** executable itself). Whether you choose to use a local or a shared configuration for a particular Apache instance will depend on two main factors: the difficulty of maintaining separate, identical copies of the configuration files and/or web site documents, and the availability and accessibility of storage which can be shared (or mirrored) between two or more servers. Note, however, that you may choose to configure both local and shared Apache instances on the same servers.

The following sections provide examples of *Local* and *Shared* Apache Web Server configurations in a LifeKeeper environment and summarize the main characteristics of each.

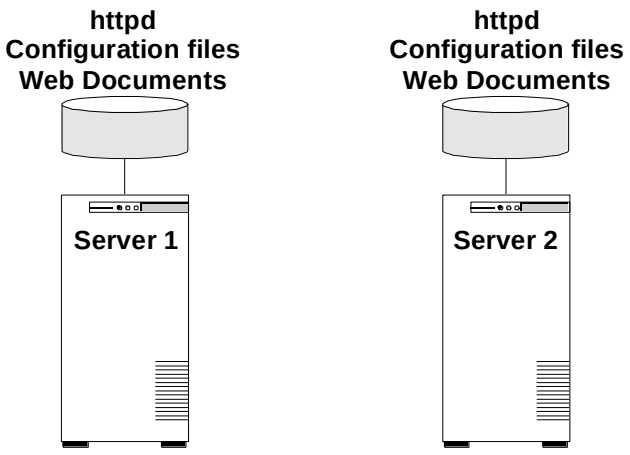
Local Configuration

In a typical local configuration, nothing is shared between the servers. Identical copies of the Apache Web Server configuration file, web documents, DSO modules (and their configuration files, if any), and the **httpd** executable reside in exactly the same locations on each server. It is the responsibility of the Apache administrator to maintain identical copies of the Apache components on the different servers.

If each web site is assigned an IP address – or a domain address that maps to a particular IP address - through the configuration file, a LifeKeeper IP address is created for each and added to the Apache resource hierarchy. When the Apache hierarchy is switched over from one server to another, this particular **httpd** instance is stopped and the IP addresses are deactivated on the first server, then the IP addresses are reactivated and the instance started on the other server. Clients will then be automatically connected via TCP/IP to the identical web site on the other server.

It is also possible not to assign an IP address for each web site. In this case, nothing is added to the Apache resource hierarchy and it does not perform any activating/deactivating of the IP address when the Apache hierarchy is switched over from one server to another.

Figure 1. Local Configuration



Configuration Notes:

- Figure 1 is an example of a local configuration where nothing is on a shared file system.
- Each server has the same version of the Apache Web Server executable at the same location (typically */usr/bin/httpd*).
- Each server has the same server root directory where identical copies of the configuration file for each instance are placed.
- Each server has the same document root directory(s) where identical copies of the web document for each instance are placed.
- If DSO modules are being used, each server has identical copies at the same location.

Creating an Apache Web Server resource hierarchy on Server 1:

Server:	Server1
Web Server Binary Location:	/usr/sbin/httpd
Web Server Root Directory:	/home/www/examples/instance1/
Root Tag	apache-www.examples.instance1

Extending an Apache Web Server resource hierarchy to Server 2:

Template Server:	Server1
Tag to Extend	apache-www.examples.instance1
Target Server	Server2
Target Priority:	10

Note that when an Apache resource hierarchy is extended to one or more additional servers, the same Web Server Binary Location and Web Server Root Directory must be used on all servers, regardless of whether this is a local or a shared configuration. See the discussion above and the section on *Specific Configuration Considerations for Apache Web Server* for additional information. Also during hierarchy extension, LifeKeeper extends all the dependent resources which are part of the Apache resource hierarchy.

Shared Configuration

In a typical shared configuration, the server root directory and the document root directories are all on the same shared file system. The same configuration file and web documents are shared between the servers, so there is no need to maintain identical copies on each server. If DSO modules are being used, they also can be located on the same shared file system, along with any configuration files or resources they may need.

Note that you may choose to place only the web documents on a shared file system. This will still appear much like a typical local configuration, since the server root directories will be local, but the hierarchy will also include a shared file system.

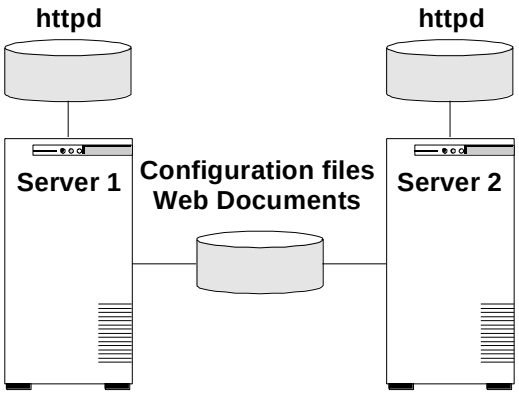
If you wish to use a particular version or a separate copy of the Apache executable for this Apache resource hierarchy, you may place this executable on the shared file system as well and it will be available only to this instance. To do this, simply enter the full path of the **httpd** executable on the shared file system when prompted for the Web Server Binary Location.

Note that only one shared file system may be used, since this assures that all required components which are on shared storage will be available at the same time. If you choose to use a Web Server Binary Location on a shared file system, you must also choose a Web Server Root Directory on the same shared file system, and all *DocumentRoot* directories configured for this server root must be on the same shared file system. Likewise, when you choose a Web Server Root Directory on a shared file system, all *DocumentRoot* directories must be on the same shared file system. If neither the binary nor the server root is placed on a shared file system, but any of the *DocumentRoot* directories are shared, all *DocumentRoot* directories must be shared on the same file system.

These rules can be summarized as follows:

- If the Apache executable is shared, then the server root directory must be shared.
- If the server root directory is shared, then all *DocumentRoot* directories must be shared
- If any *DocumentRoot* directory is shared, they must all be shared.
- Only one shared file system is allowed for each Apache resource hierarchy.

Figure 2. Shared Configuration



Configuration Notes:

- Figure 2 is an example of a shared configuration with shared configuration files and web documents.
- You may choose to place only the web documents on a shared file system. This will still appear much like a typical local configuration, except that the hierarchy will also include a shared file system.
- If DSO modules are used, they may reside on the shared file system, along with any configuration files or resources they need.

Creating an Apache Web Server resource hierarchy on Server 1:

Server:	Server1
Web Server Binary Location:	*/usr/sbin/httpd*
	…..OR….
	/shared/example/instance2/bin/httpd
Web Server Root Directory:	*/shared/example/instance2*
Root Tag	apache-shared.example.instance2

Extending an Apache Web Server resource hierarchy to Server 2:

Template Server:	Server1
Tag to Extend	apache-shared.example.instance2
Target Server	Server2
Target Priority:	10

Active/Standby and Active/Active Configurations

Apache Web Server is called an Active/Active application with LifeKeeper. This means that more than one instance of Apache can be running on a server at any time. For example, if two servers are running an instance of Apache and one server fails, the Apache instance on this server can fail over to the other server and it can continue to run its own instance as well. Some applications simply don't support this, so you would have to keep a server available for each instance of the application. These are called Active/Standby applications. Some applications can be configured either way.

There may be circumstances when you might want to operate Apache in an "active/standby" mode, particularly if only one of your servers is used primarily for running Apache. In this particular case, you should disable the automatic startup of the standard Apache default installation so that nothing is running on the backup server(s).

By manually bringing the Apache instances In Service on one or more particular servers, you can distribute the workload as you like. And by adjusting the server priorities for each of your instances, you can configure the Apache instances to fail over to a particular server only as a last resort, or to fail over to different servers to distribute the workload when a failure occurs.

If you disable automatic startup of Apache on all servers in the cluster, it is possible to use the default server root directory "*/etc/http*" for a single LifeKeeper Apache resource hierarchy by simply configuring this instance to use either LifeKeeper IP addresses or the WILDCARD IP Address (*) - and possibly using a shared file system for the document root directories.

Note, however, that this would be an Active/Standby configuration (as described above), so you could no longer start up the default instance in the usual way. Of course, the default server root directory cannot be used for more than one hierarchy, since the server root must be unique.

Configuration Considerations for Apache Web Server

Before you create Apache resource hierarchies, you will need to make sure you have completed the following configuration tasks for the Apache Web Server application:

1. The standard default Apache instance will conflict with the LifeKeeper-protected instance unless modifications are made. When the Apache package is installed, the default Apache instance is automatically configured to be started during system setup via the runlevel (rc) scripts in the */etc/rc.d* directory. The *httpd.conf* file associated with the default Apache instance has no BindAddress or Listen directives, equivalent to listening for **all** IP addresses BindAddress *.

   If you wish to use the standard default Apache instance in addition to Apache instances protected by LifeKeeper, you must modify the default configuration file (*/etc/httpd/conf/httpd.conf*)* to listen on one or more specific, non-LifeKeeper IP addresses using BindAddress or Listen directives. The **httpd** daemon must be stopped and restarted to pick up the change. The daemon may be stopped and restarted via the following commands (on SuSE, the name of the script is */etc/init.d/apache*):

   /etc/rc.d/init.d/httpd stop

   /etc/rc.d/init.d/httpd start

If you do not modify this configuration, you must disable the automatic startup of the Apache default instance otherwise it will interfere with the operation of your LifeKeeper Apache instances. Refer to your Apache Web Server documentation for specific instructions on disabling the automatic startup of the default instance.

* **For Apache on SuSE:** The default installation of Apache on SuSE does not place the *httpd.conf* configuration file in a subdirectory of *ServerRoot* called *conf*. If you are using the default installation of Apache on SuSE, you must relocate the configuration file to the directory */etc/httpd/conf*.

2. You must create a separate, distinct root directory for each LifeKeeper Apache Web Server hierarchy. This "server root" directory corresponds to the Apache "ServerRoot" configuration and command line parameters. Each LifeKeeper Apache resource hierarchy will correspond to a unique Apache instance with its associated server root directory. Note that the server root directory must be identical on all servers that are configured for a particular Apache hierarchy. You must place all configuration file information for the web site in the standard location relative to the server root (*conf/httpd.conf*) so that it can be found and accessed by the LifeKeeper software.
3. You must also follow Apache's recommendation that all configuration information be placed in a single *httpd.conf* file. The standard *srm.conf* and *access.conf* files shipped with Apache Web Server contain directives and must be disabled so they will not interfere with your configuration (by default, Apache will look for these files in */etc/httpd/conf* regardless of the server root directory being used). To disable the use of these two configuration files, include the following directives in the main body of *httpd.conf*:

   ```
   AccessConfig    /dev/null
   ResourceConfig  /dev/null
   ```

4. If the web sites (virtual hosts) will listen on specific LifeKeeper IP addresses using BindAddress or Listen directives then these LifeKeeper-protected IP addresses must already be created and available to be brought in-service where the Apache hierarchy is to be created. They will automatically be added to the Apache resource hierarchy. If the web sites (virtual hosts) will not listen on LifeKeeper IP addresses, you should edit the BindAddress or Listen directives to use * or 0.0.0.0. on BindAddress or Listen directives. This will configure Apache to listen on all IP addresses. In this case, the IP resources are not added to the Apache hierarchy.

   If you will be using a LifeKeeper shared file system, you must make all necessary preparations for the file system creation prior to creating the Apache hierarchy. In particular, the file system must be mounted on the server where the Apache hierarchy is to be created. If the LifeKeeper file system hierarchy has not already been created, it will automatically be created along with the Apache hierarchy, then joined to the Apache resource hierarchy.

   Consult the Apache Web Server documentation for detailed information on configuring virtual hosts. For example, the configuration file for an instance that combines IP-based and name-based virtual hosts would include directives like the following:

   ```
   User webuser
   Group webgroup
   ServerName localhost

   AccessConfig /dev/null
   ResourceConfig /dev/null
   ```

```
Listen 172.17.100.55:8000
NameVirtualHost 172.17.100.55:8000
Listen 172.17.100.56:80

<Virtualhost site.name_one:8000>
ServerName   site.name_one
DocumentRoot /shared/site/name_one
</VirtualHost>

<VirtualHost site.name_two:8000>
ServerName   site.name_two
DocumentRoot /shared/site/name_two
</VirtualHost>

<VirtualHost 172.17.100.56:80>
ServerName   site.ip
DocumentRoot /shared/site/ip
</VirtualHost>
```

5. If SSL support is enabled for your Apache instance, SSL Listen directive must be a similar configuration to the BindAddress/Listen directive on httpd.conf. SSL Listen directive is often found in an ssl.conf file. Otherwise, the creation of your Apache hierarchy will fail with an error. Note that SSL support is enabled by default in the Apache configuration files of some Linux distributions.

   For example, when you want to change listen on the IP address 172.17.100.55, change the following entry in the default SSL configuration file at */etc/httpd/conf.d/ssl.conf* from

   ```
   Listen 0.0.0.0:443
   ```

   to

   ```
   Listen 172.17.100.55:443
   ```

6. For a **Local** configuration, you must install and configure Apache in the same location on both the primary and all backup servers and set up identical (or equivalent) configuration files in the same server root directory on all servers. Also, all document root directories must exist on all servers and should contain identical files. (See the section on *Local Configuration* in *Configuration Definitions and Examples*.)
7. For a **Shared** configuration, you will typically configure the server root directory on a LifeKeeper shared file system. Note that only one shared file system may be used, since this assures that all required components which are on shared storage will be available at the same time. Therefore, all document root directories must be subdirectories of the same shared file system, but they need not be subdirectories of the server root directory itself. You may place an Apache executable on the same shared file system as well, but this executable will only be available for use by this particular Apache resource hierarchy.

   Note: You don't necessarily need to place the server root directory on a shared file system in order to make use of shared storage. You may choose a local server root directory for configuration files, etc., and place only the document root directories on a shared file system. However, you must configure identical server root directories and identical (or equivalent) configuration files on all servers (as for a Local configuration as described above), and all

document root directories must be on the same shared file system. (See the section on *Shared Configuration* in *Configuration Definitions and Examples*.)

8. Some web site implementations make use of DSO (Dynamic Shared Object) modules to extend Apache support for certain features. For example, there are modules available that implement functionality for PHP and Perl. These modules can be loaded and accessed at runtime by the Apache core. If you are using modules, they must be identically configured on every server in the cluster. Consult the documentation for the module package, and the vendor-supplied documentation for configuring Apache to use modules on your Linux platform. Depending on the module and the resources it uses, some objects may be required to reside on shared storage to facilitate proper failover. In some cases, a module may even need to be protected separately using the *Generic Application Recovery Kit*, or a custom recovery kit.
9. If you are using the SSL (Secure Sockets Layer) module with Apache, it is important that the server not be password protected. When the web server is password protected, the administrator must interactively type in the password at a prompt each time the daemon starts. Since this manual step is not consistent with a High Availability environment where recovery time is critical, LifeKeeper does not support password protected instances. Use the following command to remove the password:

   openssl rsa -in server.key -out unprotected_server.key

   Enter the server key password when prompted. To preserve the security of your site, make sure that the file is readable only by root!

   chmod 400 unprotected_server.key

   During the hierarchy creation of an Apache instance, the Recovery Kit checks that the resource is not password protected. If it is password protected, hierarchy creation will fail with an error message. However, when the instance is extended to another server, the Recovery Kit does not check for password protection on the backup server. You need to make sure that the hierarchy you are extending is not password protected.

   The server key file(s) (specified by the **SSLCertificateKeyFile** directive(s) in the Apache configuration file) must have the same name and be at the same location on all servers in the cluster.

   Note: The PID file name of the httpd process that LifeKeeper uses has the following format:

   "/var/ran/httpd.<TAG name>.pid"

   This PID file name is different from the default PID file name used by the OS. If you need to reference this PID file (ex. log rotate), please note the LifeKeeper PID file name and format.

LifeKeeper Configuration Tasks

You can perform the following configuration tasks from the LifeKeeper GUI. The following four tasks are described in this guide, as they are unique to an Apache resource instance, and different for each Recovery Kit.

- **Create a Resource Hierarchy.** Creates an application resource hierarchy in your LifeKeeper cluster.
- **Extend a Resource Hierarchy.** Extends a resource hierarchy from the primary server to a backup server.
- **Unextend a Resource Hierarchy.** Unextends (removes) a resource hierarchy from a single server in the LifeKeeper cluster.
- **Delete a Resource Hierarchy.** Deletes a resource hierarchy from all servers in your LifeKeeper cluster.

The following tasks are described in the GUI Administration section within the *SPS Technical Documentation*, because they are common tasks with steps that are identical across all Recovery Kits.

- **Create a Resource Dependency.** Creates a parent/child dependency between an existing resource hierarchy and another resource instance and propagates the dependency changes to all applicable servers in the cluster.
- **Delete a Resource Dependency.** Deletes a resource dependency and propagates the dependency changes to all applicable servers in the cluster.
- **In Service.** Brings a resource hierarchy into service on a specific server.
- **Out of Service.** Takes a resource hierarchy out of service on a specific server.
- **View/Edit Properties.** View or edit the properties of a resource hierarchy on a specific server.

Note: Throughout the rest of this section, we explain how to configure your Recovery Kit by selecting certain tasks from the **Edit** menu of the LifeKeeper GUI. You can also select each configuration task from the toolbar. You may also right click on a global resource in the Resource Hierarchy Tree (left-hand pane) of the status display window to display the same drop down menu choices as the **Edit** menu. This, of course, is only an option when a hierarchy already exists.

You can also right click on a resource instance in the Resource Hierarchy Table (right-hand pane) of the status display window to perform all the configuration tasks, except *Creating a Resource Hierarchy*, depending on the state of the server and the particular resource.

Creating an Apache Web Server Resource Hierarchy

IMPORTANT:

Before you create your Web Server resource hierarchy:

The web sites (virtual hosts) must be configured to listen on LifeKeeper IP addresses, or you must edit the BindAddress or Listen directives to use * or 0.0.0.0. on BindAddress or Listen directives.

In a shared environment where the web documents and/or configuration files are on a shared disk, you must make sure that the shared file system is mounted. It is also important to remember that you require a working communication path (i.e. heartbeat) before you can extend your resource to a backup server.

To create a resource instance from the primary server, you should complete the following steps:

1. From the LifeKeeper GUI menu, select **Edit**, then **Server**. From the menu, select **Create Resource Hierarchy**.

   The Apache Web Server should not be running when you create the resource. However, if you set up the listen variable in the system configuration file, the default daemon can be allowed to run.

   The *Create Resource Wizard* dialog box will appear with a drop down list box displaying all recognized recovery kits installed within the cluster.

2. Select Apache Web Server and click **Next**.
3. You will be prompted to enter the following information. When the **Back** button is active in any of the dialog boxes, you can go back to the previous dialog box. This is helpful should you encounter an error requiring you to correct previously entered information. You may click **Cancel** at any time to cancel the entire creation process.

Field	Tips
Switchback Type	Choose either *intelligent* or *automatic* This dictates how the Apache instance will be switched back to this server when the server comes back up after a failover. The switchback type can be changed later from the General tab of the Resource Properties dialog box.
Server	Select the **Server** where you want to place the Apache Web Server (typically this is referred to as the primary or template server). All the servers in your cluster are included in the drop down list box.
Web Server Binary Location	Select or enter the full path name (including the file name) of the **httpd** Apache Web Server daemon. The default is */usr/sbin/httpd.*
Web Server Root Directory	You must provide the full path of the Web Server Root directory; a relative path or symbolic link may not be used. The Apache Web Server configuration file is located in *conf/httpd.conf* relative to the Server Root. **Note:** At this point, LifeKeeper will check that there is a protected IP or the wildcard (*) or 0.0.0.0. IP address is configured. It will also validate that you have provided valid data to create your Apache Web Server resource hierarchy. If LifeKeeper detects a problem with either of these validations, an ERROR box will appear on the screen. If the Web Server Root Directory path is valid, but there are errors with the Apache configuration itself, you may pause to correct these errors and continue with the hierarchy creation. You may even pause to create any LifeKeeper IP resources that are required.
Root Tag	Select or enter the tag name given to the Web Server hierarchy. You can select the default, which is apache<root directory>, or enter your own tag name.

4. Click **Create.** The *Create Resource Wizard* will then create your Apache resource hierarchy. LifeKeeper will validate the data entered. If LifeKeeper detects a problem, an error message will appear in the information box.
5. An information box will appear indicating that you have successfully created a Apache resource hierarchy, and you must Extend that hierarchy to another server in your cluster in order to achieve failover protection. Click **Next**.

   Note: You may encounter error messages indicating that the new Apache instance has failed to start correctly. Note that the new Apache hierarchy must be started (In Service) before it can be extended to another system. You may pause at this point and correct the problem

based on the error message displayed, then bring the new hierarchy In Service before proceeding with extending the hierarchy.

6. Click **Continue.** LifeKeeper will then launch the *Pre-Extend Wizard*. Refer to Step 2 under Extending an Apache Resource Hierarchy (below) for details on how to extend your resource hierarchy to another server.

   If you click **Cancel**, a dialog box will appear warning you that you will need to come back and extend your Apache resource hierarchy to another server at some other time to put it under LifeKeeper protection.

Extending Your Hierarchy

This operation can be started from the **Edit** menu, or initiated automatically upon completing the **Create Resource Hierarch**y option, in which case you should refer to Step 2 below.

1. On the **Edit** menu, select **Resource**, then Extend Resource Hierarchy. The Pre-Extend Wizard appears. If you are unfamiliar with the Extend operation, click **Next**. If you are familiar with the LifeKeeper **Extend Resource Hierarchy** defaults and want to bypass the prompts for input/confirmation, click **Accept Defaults**.
2. The *Pre-Extend Wizard* will prompt you to enter the following information.
   Note: The first two fields appear only if you initiated the Extend from the **Edit** menu.

Field	Tips
Template Server	Enter the server where your Apache resource is currently in service. It is important to remember that the **Template Server** you select now and the **Tag to Extend** that you select in the next dialog box represent an *in service* resource hierarchy. An error message will appear if you select a resource tag that is not in service on the template server you selected. The drop down box in this dialog provides the names of all the servers in your cluster.
Tag to Extend	Select the name of the Web Server instance you wish to extend from the template server to the target server. The wizard will list in the drop down list box all the resources that you have created on the template server, which you selected in the previous dialog box.
Target Server	Select the **Target Server** where you are extending your Web Server resource hierarchy. The drop down box provides the names of the servers in your cluster that are not already in the selected hierarchy.

Switchback Type	Select either *intelligent* or *automatic.* This dictates how the Web Server instance will be switched back to this server when it comes back into service after a failover to the backup server. Intelligent switchback requires administrative intervention to switch the instance back to the primary/original server. Automatic switchback means the switchback will occur as soon as the primary server comes back on line and reestablishes LifeKeeper communication paths.
Template Priority	Select or enter a priority for the template hierarchy. Any unused priority value from 1 to 999 is valid, where a lower number means a higher priority (1=highest). The extend process will reject any priority for this hierarchy that is already in use by another system. The default value is recommended. **Note:** This selection will appear only for the initial extend of the hierarchy.
Target Priority	Select or enter the **Target Priority** of your extended Web Server resource. The priority is a number between 1 and 999 indicating a server's priority in the cascading failover sequence for the resource. The hierarchy priorities are sorted numerically, where a lower number means a higher priority (the number 1 indicates the highest priority). Note that LifeKeeper automatically assigns the number "1" to the server that the hierarchy is created on. The priorities need not be consecutive, but no two servers can have the same priority for a given resource.
	After receiving the message that the pre-extend checks were successful, click **Next**. Depending upon the hierarchy being extended, LifeKeeper will display a series of information boxes showing the Resource Tags to be extended, which cannot be edited. Click **Extend.**
Network Interface	Select or enter the **Network Interface.** This is the name of the network interface (i.e. Ethernet card) the IP resource will use on the target server.
Backup Interface	Select a **Backup Interface** if you want to engage the IP Local Recovery feature on the server to which you are extending the IP resource. The default value is *none*; however, if you have another network interface card configured on this server, it should be listed in the drop down list.
IP Resource Tag	Select or enter the **IP Resource Tag.** This is the resource tag name to be used by the IP resource being

	extended to the target server.
Root Tag	Select or enter the **Root Tag**. This is the tag name given to the Web Server hierarchy. By default, the Root Tag name should be the same on both the template and target server.
Mount Point	*This selection appears only when the Web Server Root Directory is on a shared file system.* Select or enter the **Mount Point** of the shared file system where the Web Server Root Directory is located. The Template Server and Target Server should have the same mount point for the shared Web Server Root Directory. The default mount point provided in the dialog box should be selected in most cases.
Root Tag	*This selection appears only when the Web Server Root Directory is on a shared file system.* Select or enter the **Root Tag**. This is the tag name of the shared file system.

3. An information box will appear verifying that the extension is being performed. Click **Next Server** if you want to extend the same Apache resource instance to another server in your cluster. This will repeat the Extend Resource Hierarchy operation.

   If you click **Finish**, LifeKeeper will verify that the extension of the Web Server resource was completed successfully.

4. Click **Done** in the last dialog box to exit from the Extend Resource Hierarchy menu selection.

Note: Be sure to test the functionality of the new instance on *both* servers.

Unextending Your Hierarchy

1. On the **Edit** menu, select **Resource**, then **Unextend Resource Hierarchy**
2. Select the **Target Server** where you want to unextend the Web Server resource. It cannot be the server where the Web Server is currently in service. (This dialog box will not appear if you selected the Unextend task by right clicking on a resource instance in the right pane.)

   Click **Next**.

3. Select the Web Server hierarchy to unextend and click **Next**. (This dialog will not appear if you selected the Unextend task by right clicking on a resource instance in either pane).
4. An information box appears confirming the target server and the Web Server resource hierarchy you have chosen to unextend. Click **Unextend**.
5. Another information box appears confirming that the Web Server resource was unextended successfully.
6. Click **Done** to exit.

Deleting a Resource Hierarchy

It is important to remember that if you delete a hierarchy before you take it out-of-service, the resource hierarchy will be removed from LifeKeeper protection, but the Apache instance will continue to run on the currently active server unless it is manually stopped or the system is rebooted. Attempting to recreate the same Apache hierarchy with a different IP address(s) or to create a new Apache hierarchy using the previously used IP address(s) (but using a different Server Root), will result in conflicts with the Apache instance that was left running with that same address.

To delete a resource hierarchy from all the servers in your LifeKeeper environment, complete the following steps:

1. From the LifeKeeper GUI menu, select **Edit**, then **Resource**. From the drop down menu, select **Delete Resource Hierarchy**.
2. Select the name of the **Target Server** where you will be deleting your Web Server resource hierarchy. Click **Next** to proceed to the next dialog box.

   Note: If you selected the Delete Resource task by right clicking from the right pane on an individual resource instance, or from the left pane on a global resource where the resource is on only one server, the Target Server dialog will not appear.

3. Select the **Hierarchy to Delete**. (This dialog will not appear if you selected the Delete Resource task by right clicking on a resource instance in the left or right pane.) Remember that the list box displays every hierarchy on the target server , both in service and out of service. If you want to stop the Apache instance and remove the resource hierarchy from LifeKeeper protection, you must make sure that the hierarchy you choose is out-of-service before deleting it.

   Click **Next**.
4. An information box appears confirming your selection of the target server and the hierarchy you have selected to delete. Click **Next**..
5. Another information box appears confirming that the Web Server resource was deleted successfully.
6. Click **Done** to exit.

Testing Your Resource Hierarchy

You can test your Apache resource hierarchy by initiating a manual switchover. This will simulate a failover of a resource instance from the primary server to the backup server.

Performing a Manual Switchover from the GUI

You can initiate a manual switchover from the LifeKeeper GUI by selecting **Edit**, then **Resource**, then finally **In Service** from the drop down menu. For example, an *in service* request executed on a backup server causes the application hierarchy to be placed in service on the backup server and taken out of service on the primary server. At this point, the original backup server is now the primary server and original primary server has now become the backup server.

If you execute the **Out of Service** request, the application is taken out of service without bringing it in service on the other server.

Recovery Operations

When the primary server fails, the Apache Recovery Kit software performs the following tasks:

- If there is an IP resource under the Apache resource hierarchy, it brings Apache into service on the backup server by bringing *in service* the IP address(s) on one/more of that server's physical network interfaces
- Mounts the file system—if one is being used—on the shared disk on that server
- Starts the daemon processes related to Apache

After recovery, Apache Web Server users may reconnect by clicking on the Reload/Refresh button of their browsers.

Troubleshooting

This section provides a list of messages that you may encounter during the process of creating and extending a LifeKeeper Apache Web Server resource hierarchy, removing and restoring a resource, and, where appropriate, provides additional explanation of the cause of the errors and necessary action to resolve the error condition. Messages from other SPS components are also possible. In these cases, please refer to the **Message Catalog** (located on our Technical Documentation site under "**Search for an Error Code**") which provides a listing of all error codes, including operational, administrative and GUI, that may be encountered while using SIOS Protection Suite for Linux and, where appropriate, provides additional explanation of the cause of the error code and necessary action to resolve the issue. This full listing may be searched for any error code received, or you may go directly to one of the individual Message Catalogs for the appropriate SPS component.

Messages in this section fall under these topics:

- **Hierarchy Creation**
- **Extend Hierarchy**
- **Hierarchy Remove, Restore and Recovery**

Hierarchy Creation Errors

The error messages that might be displayed during the Apache hierarchy creation are listed below, along with a suggested explanation for each. Error messages displayed by the LifeKeeper core and by other recovery kits are not listed in this guide. Note that you may stop to correct any of the problem(s) described here, and then continue with hierarchy creation from the point where you left off – including creating any new LifeKeeper resources you might need for your Apache configuration.

During Validation of Web Server Binary Location

"Error: valid_httpd_path: Must specify absolute path to httpd executable."

Enter the full, absolute path name to a valid Apache **httpd** executable.

"Error: valid_httpd_path: File does not exist at path specified."

A valid Apache **httpd** executable does not exist at the location specified.

"Error: valid_httpd_path: Httpd failed to display Server version."

The **httpd** executable at the location specified does not display the standard Apache "Server version."

"Error: valid_httpd_path: Incorrect version $MAJOR.$MINOR.$POINT of Apache at $HTTPD_PATH.

The Apache **httpd** executable at the location specified displays the incorrect "Server version."

During Validation of Web Server Root Directory

"Error: valid_http_root: Cannot find Apache configuration file at $CONF_FILE."

Must have valid Apache configuration file at *conf/httpd.conf* relative to the Server Root directory specified. Note that the default installation of Apache on SuSE does not place the *httpd.conf* configuration file in a subdirectory of ServerRoot called *conf*. If you are using the default installation of Apache on SuSE, you must relocate the configuration file to the directory */etc/httpd/conf*.

"Error: valid_http_root: Must specify absolute path to Apache server root directory."

Enter full, absolute path name to a Server Root directory.

"Error: valid_http_root: Apache instance at $HTTP_ROOT is already under LifeKeeper protection."

Each instance must have its own, unique Server Root directory, with configuration file located at *conf/httpd.conf*. The Server Root directory specified is already being used by another Apache instance.

"Syntax error on line <line number> of <configuration file path>, etc…"

Syntax error(s) were found in the Apache configuration file. These error messages were displayed by the **httpd -T** command when used to check the syntax of $CONF_FILE. See the error messages displayed for details.

"Error: valid_http_root: Since $HTTPD_PATH is shareable on $HTTPD_PATH_SHARED, $HTTP_ROOT must be also."

If the **httpd** executable is on shared/shareable storage, the Server Root and all DocumentRoot directories must be also.

"Error: valid_http_root: Since $HTTP_ROOT is shareable on $HTTP_ROOT_SHARED, all document root directories must shareable on this same filesystem."

If the Server Root is on shared/shareable storage, all DocumentRoot directories must be also.

"Error: http_docs_shared: Since one/more Apache document root directories are shareable on $docs_shared, $curr_root must be also.

If any DocumentRoot directories are on a shared/shareable file system, all DocumentRoot directories must be located on the same file system.

"Error: valid_http_root: Must include BindAddress or Listen directives for each Apache instance. Check the Apache configuration file at $CONF_FILE."

In order to run multiple instances of Apache, each configuration file must contain BindAddress or Listen directives. Please refer to the Configuration Considerations for Apache Web Server section earlier in this guide for further detail.

"Error: valid_http_root: A Listen directive is being used which specifies an IP address but no port. Check the Apache configuration file at $CONF_FILE."

The correct syntax for the Listen directive is Listen [*IP address*:] port number. This is not caught as a syntax error by Apache, but is interpreted incorrectly (as though the first number in the IP address was a port number specification).

"Error: valid_http_root: IP address $ip is not LifeKeeper protected."

The Apache configuration file refers to an IP address or domain name not configured under LifeKeeper protection. If you set the default IP address* on the configuration file, you must create these LifeKeeper IP address resources in advance.

During Apache Resource Hierarchy Creation

"Error: Could not find IP resource for $IP_ADD on machine $MACH."

If you set the default IP address* on the configuration file, you must create this resource before the Apache resource creation will succeed.

"Error: Create Apache file system hierarchy failure for filesystem $FSNAME used by server root $HTTP_ROOT."

"Error: Failure bringing Apache Resource $TAG into service on machine $MACH."

Check the Apache error logs for messages (default location is */var/log/httpd/error_log*, but other logs may be listed).

The most likely cause of this problem is an error in the Apache configuration file. You may be able to bring this resource into service manually after correcting the problem.

"LifeKeeper: RESTORE: *ERROR* Apache: The instance is Password Protected."

The LifeKeeper Apache Web Server Recovery Kit cannot support password protected Private Key files for SSL-enabled web servers, since this would require manual interaction each time Apache starts up, and would prevent automatic restart and failover. The section *Specific Configuration Considerations for Apache Web Server* in this document explains how to remove password protection from the Private Key file (specified by the **SSLCertificateKeyFile** directive). This message applies only in an environment where the SSL module is used with Apache.

Extend Hierarchy Errors

The error messages that might be displayed during Apache hierarchy extension are listed below, along with a suggested explanation for each. Note that these error messages appear when the GUI indicates it is "Executing the pre-extend script…." to validate the hierarchy prior to extending it to the new system.

Each will be preceded by an error message like:

"Error - canextend(template_server, tag, app_type/resource_type, target_server) -".

Each will be followed by an error message like:

"Error - extmgr(template_server, tag, target_tag, target_server) -".

During Validation of Web Server Binary Location

See errors listed for validation of Web Server Binary Location under Hierarchy Creation Errors.

During Validation of the Apache Configuration File on the Target System

"Cannot find Apache configuration file at $CONF_FILE on $TARGET_SYS."

Must have a valid Apache configuration file at *conf/httpd.conf* relative to Server Root directory specified.

"DocumentRoot directory "$doc" in $CONF_FILE on $TARGET_SYS was not found in the configuration file on $TEMPLATE_SYS."

or

"DocumentRoot directory "$doc" in $CONF_FILE on $TEMPLATE_SYS was not found in the configuration file on $TARGET_SYS."

While comparing the configuration files on target and template servers, one or more DocumentRoot directories were found which do not match between the two. Check the details of the error messages displayed to determine the differences between the two. Note that if a DocumentRoot directory path is typed incorrectly, you will generally see both of these error messages, since each configuration file will appear to have an entry not in the other file.

"IP:port combination "$ipp" in $CONF_FILE on $TARGET_SYS was not found in the configuration file on $TEMPLATE_SYS."

or

"IP:port combination "$ipp" in $CONF_FILE on $TEMPLATE_SYS was not found in the configuration file on $TARGET_SYS."

While comparing the configuration files on target and template servers, one or more IP/port combinations were configured for use on one server but not on the other. Note that the IP/port combinations used may be specified in terms of IP addresses, ports, and domain names using a variety of Apache configuration directives. It is the actual IP/port combinations used which are compared, not the directives used to specify them. Check the details of the error messages displayed to determine the differences between the two.

"SSLCertificateKeyFile "$file" in $CONF_FILE on $SYS1 was not found in the configuration file on $SYS2."

The filename specified for the SSLCertificateKeyFile in the Apache configuration file on the target system does not match the one specified on the template system. These configurations must be identical. This message applies only in an environment where the SSL module is used with Apache.

"Apache SSLCertificateKeyFile exists on $SYS1 but not on $SYS2."

The SSLCertificateKeyFile specified in the Apache configuration files exists on one system, but not on the other. The file must be present on both nodes. This message applies only in an environment where the SSL module is used with Apache.

"WARNING: PHP configuration file $PHP_CONFIG appears to be different on $SYS1 and $SYS2."

The configuration file for the PHP module on the target system is not identical to the one on the template system. Inspect the configuration on both servers to ensure that they are the same. This message applies only in an environment where the PHP module is used with Apache.

During Apache Resource Hierarchy Creation on Target Server

See errors listed for Apache resource hierarchy creation under Hierarchy Creation Errors.

Hierarchy Restore, Remove, and Recover Messages and Errors

The following information and error messages are printed to the LifeKeeper error log.

They may be viewed by typing “lk_log log”.

Bringing an Apache Resource In Service (Restore)

"LifeKeeper: RESTORE: APACHE: RESTORING $TAG TO SERVICE START AT: <date>"

Informational message. Records when the restore begins. Logged at the start of every restore.

"LifeKeeper: RESTORE APACHE RESOURCE $TAG END err=$err AT: <date>"

Informational message. Records when the restore completes. Logged at the end of every restore. If any errors occur during the restore, additional messages will be logged between these two messages and the value displayed for err=$err will be non-zero.

"Apache: No instance information found for Tag=$TAG."

Error: Indicates no instance is defined with the tag value passed to the “restore” script. Unlikely to occur with the GUI, since only tags known to LifeKeeper are available as choices for the In Service and Out of Service actions.

"LifeKeeper: RESTORE: Apache: Tag=$TAG already running."

Informational message. Indicates that the instance appeared to already be up and running.

"LifeKeeper: RESTORE: Apache: Existing processes terminated for ID=$ID."

Informational message. Existing **httpd** processes were found running for this instance ID, but the PidFile is either missing or invalid. Therefore, the running processes were terminated.

"LifeKeeper: RESTORE: Apache: Invalid PidFile=$PIDFILE has been deleted."

Informational message. An existing PidFile was found for this instance, but its contents were invalid. Therefore, the PidFile was deleted.

"LifeKeeper: RESTORE: Apache: Tag=$TAG is being restarted."

Informational message. Indicates that the instance appeared to partially running, but needed to be restarted. If a PidFile still exists (which contains the process ID of the parent **httpd** process), the instance is restarted with a HUP signal. If the PidFile is missing, the instance is completely stopped and restarted.

"LifeKeeper: RESTORE: *ERROR* Apache: Error in web server configuration file $CONF_FILE for instance $ID."

"LifeKeeper: RESTORE: *ERROR* Apache: Execute the following command to check the syntax of this file:"

"LifeKeeper: RESTORE: *ERROR* Apache: $HTTPD_PATH -t -d $SERVER_ROOT -f $CONF_FILE."

Prior to instance startup, the syntax of the configuration file is checked using the **httpd -t** option. The **-d** option checks the ServerRoot directory. Additional options related to modules may also be displayed if you have configured Apache to use modules. Any syntax errors caught during hierarchy creation are displayed in the LifeKeeper GUI, but syntax errors introduced later will not be displayed in the GUI or the LifeKeeper logs. You must manually run the following command to determine what is wrong with your configuration (add additional options for modules, if applicable):

$HTTPD_PATH -t -d $SERVER_ROOT -f $CONF_FILE

"LifeKeeper: RESTORE: *ERROR* Apache: Error starting web server instance $INSTANCE."

"LifeKeeper: RESTORE: *ERROR* Apache: Restore of tag $TAG failed."

"LifeKeeper: RESTORE: *ERROR* Apache: Examine the Apache error log at $ERROR_LOG"

"LifeKeeper: RESTORE: *ERROR* Apache: to determine the cause of the problem."

An error occurred executing the **httpd** daemon with the parameters specified. Check the **httpd** executable being used, configuration file, and general configuration for possible problems.

"LifeKeeper: RESTORE: *ERROR* Apache: Web server instance $ID did not start correctly."

"LifeKeeper: RESTORE: *ERROR* Apache: Restore of tag $TAG failed."

"LifeKeeper: RESTORE: *ERROR* Apache: Examine the Apache error log at $ERROR_LOG"

"LifeKeeper: RESTORE: *ERROR* Apache: to determine the cause of the problem."

Note that in many cases the **httpd** daemon will appear to start even if its web sites don't respond as expected. The restore script checks all IP/port combinations used to make sure all sites configured are fully functional. If they are not, this message is printed and the restore fails.

Note that although the site is left in the Out of Service state, one/more **httpd** processes may still be left running. (This is intentional, since one/more web sites may be operational and we don't want to kill them off). You should resolve the problem as soon as possible and bring the instance In Service. If you don't, LifeKeeper will eventually attempt to recover the instance and restore it to service automatically. If it can't, it will fail over the hierarchy to another server.

Taking an Apache Resource Out of Service (Remove)

"LifeKeeper: REMOVE: APACHE: REMOVE $TAG FROM SERVICE START AT: <date>"

Informational message. Records when the remove begins. Logged at the start of every remove.

"LifeKeeper: REMOVE APACHE RESOURCE $TAG END err=$err AT: <date>"

Informational message. Records when the remove completes. Logged at the end of every remove.

If any errors occur during the remove, additional messages will be logged between these two messages and the value displayed for err=$err will be non-zero.

"LifeKeeper: REMOVE: *WARNING* APACHE: Error attempting to kill parent process for INSTANCE=$INSTANCE."

There was an error attempting to kill the parent **httpd** process (whose process ID is stored in the Pidfile).

"LifeKeeper: REMOVE: *ERROR* APACHE: Error attempting to kill all processes for INSTANCE=$INSTANCE."

Although the parent **httpd** process appeared to be killed successfully, one/more processes for this instance are still running. Normally the remove will be able to terminate any/all processes for this instance. When it cannot, this message is printed and the remove fails.

Bringing an Apache Resource Back In Service (Recover)

The LifeKeeper core periodically checks the health of every Apache instance In Service on the local server by running an Apache "quickCheck" script, which checks the web sites using the same scripts used to check the state of the instance during restore and remove. If the instance is not fully functional, a "recover" script is invoked to attempt to restart the instance. This simply logs the first message shown below, invokes "restore," prints the final error or success message shown below—depending on error or success of the "restore" script—and returns the same result as "restore." If restore/recover fails, this instance is failed over to another server.

"LifeKeeper: RECOVER: APACHE: Invoking restore for Apache instance "$ID" at: <date>"

"LifeKeeper: RECOVER: APACHE: Restore for Apache instance $ID returned error $RET at: <date>"

"LifeKeeper: RECOVER: APACHE: Restore for Apache instance $ID successful at: <date>"