



Chris Rayns  
 Rochelle Fisher  
 Avinoam Hirschberg  
 Robert Hugh  
 Ury Segal

# Linux systems management using Aduva Director

## Preface

There are many Linux zSeries systems management tools available. This Redpaper discusses the Aduva Director tool, the paper reviews the installation steps required to install the product, and then focuses on the usage of the product.

## The team that wrote this redpaper

**Chris Rayns** is the Security Project Leader at the International Technical Support Organization, Poughkeepsie Center.

**Rochelle Fisher** is the Technical Writer Team Leader of Aduva, Inc. Her work for Aduva received an Award of Merit from the Society of Technical Communicators (2001 Online Help Competition).

**Avinoam Hirschberg** is the Director of Products at Aduva, Inc. and has more than 20 years of experience in system management on VM/VSE and mixed environments (Intel, UNIX(AIX) and AS/400), as well as extensive experience in designing and deploying management tools on mainframe, AS/400 and Intel platforms.

**Robert Hugh** currently works for Aduva, Inc.

**Ury Segal** is CTO and co-founder of Aduva, Inc. He has worked on the Mosix Kernel, a multicomputer operating system, has several years of experience working on GUI work including Motif and Java, and has done kernel programming in Solaris, HP/UX, AIX, and SYSV.

# Objectives

This Redpaper has the following objectives:

- Provide an overview of the Aduva Director
- Provide an overview of the Private Component Editor (PCE)
- Describe Aduva Director installation
- Describe how to start the Aduva Director Console
- Examine system definition

# Introduction

This Redpaper provides an overview of Aduva Director and how it can be used to simplify management tasks. We look at how Aduva Director can be used to manage Linux images running on z/Series under z/VM.

Aduva Director lets you centrally configure, manage, and monitor multiple Linux images within an S/390 or zSeries mainframe system.

**Important:** Aduva Director is branded and sold by BMC Software, Inc. as Deployment Manager for Linux. Aduva and BMC Software have worked together on specific extensions for the zSeries environment, to add value to our customers.

The following topics are covered:

- ▶ Introduction to Aduva Director
- ▶ Private Component Editor (PCE)
- ▶ Aduva Director installation overview
- ▶ Starting the Aduva Director Console
- ▶ Defining the system
- ▶ Profile jobs
- ▶ Inventory jobs
- ▶ Job tasks
- ▶ Confirming actions
- ▶ Advanced features

## Introduction to Aduva Director

In this section we describe what Aduva Director can do for you, how it works, and how it meets the unique challenges of the Linux environment.

### Aduva Director overview

Aduva Director automates tasks required to manage, support, and monitor Linux images. Its graphical interface makes it easy to perform even complicated tasks with minimal Linux expertise.

Aduva Director provides the following features:

Software:

- ▶ Detects and lists installed software packages of supported types
- ▶ Finds and fixes missing dependencies
- ▶ Provides automatic upgrades, installs, uninstalls, and downgrades
- ▶ Corrects packages for optimized security

Multiple remote Host maintenance:

- ▶ Maintains image configuration definitions (profiles) and checks that all Hosts belonging to that group conform to this configuration
- ▶ Maintains resolve policies, to resolve conflicts automatically and to set the level of automation of the Aduva Director
- ▶ Provides robust task scheduling for convenient and periodic times

- ▶ Delegates responsibility to different types of managers, preventing job conflicts between users
- ▶ Provides easy access and comparison to multiple Host inventories

## Technical concepts

The entities involved in a complete Aduva Director system are: its main rpm packages, the Aduva servers, Linux components supported by and recognized by the Director, and the known interdependencies of these components.

### Main packages

The Director packages are the Engine, the Database, Consoles, and Agents. Table 1 lists and describes each package.

Table 1 Aduva Director main packages

Package	Description
Engine	This handles communication between the Agents and the Console, and manages Agent uptime.
Database	This stores Director activity data, entity properties, configuration profiles; it is installed on the Engine machine.
Agent	This is a daemon that runs on each image that you want to control with the Director.
Console	This is the GUI from which the Director users, called <i>managers</i> , initiate tasks on the to be performed on the Agents

### Servers

The Director is able to intelligently perform complicated jobs by connecting the Agents and the Consoles to the two Servers. One Server contains Linux *components*, logical units that are (or can be) part of an image, and the other Server holds dependency rules. Together they form the Knowledge Base. Table 2 lists and compares the functions of these Servers.

Table 2 Aduva Servers, the Aduva Knowledge Base

Server name	Function and comparison
rserver (Rules Server)	This contains knowledge about components: what they are, their dependencies, and rules on how they are installed. It operates as a Mirror server. It becomes a clone of the parent server when it gets updates, either because of an update request sent over a direct connection, or through data supplied by Aduva. If the rserver receives a request for data that cannot be found locally, the request fails.
bserver (Blobs Server)	This contains those parts of the known components which can be downloaded. The bserver operates as a proxy server. It contacts a parent server only if the proxy receives a request for data (either a downloadable component part, or a signature of one) that cannot be found locally. The bserver <i>must</i> have a direct connection to a parent.

Agents pull from the Knowledge Base to get the components and rules they need to complete jobs.

Consoles pull from the Knowledge Base to get a complete list of available downloadables in order to give you a comprehensive list of components available for download. The Console displays the inventory (which is a list of recognized installed components) of a Host, and enables you to quickly compare a Host's inventory with what is available from the Servers.

### Connecting the Aduva Servers

Aduva supplies public servers, accessible over the Internet, or you can install the Aduva Servers on your local network. Table 3 describes various connection scenarios.

Table 3 Aduva Servers connection scenarios

Direct Internet connection	Aduva Director uses SSL and certificates to ensure security (through encryption) and data authentication. Once the connection is established, all communication is done with SSL, using a 128-bit encryption 3DES algorithm. Initial communication uses RSA/DSA, with 1024-bit keys, to enable the Public/Private key encryption scheme. The basis for Aduva Director-to-Aduva Servers security is that no communication is initiated by the servers; everything starts with Aduva Director. In short, you are in control of any data that you receive from the servers. You will never receive unsolicited or unknown data from the Aduva Servers.
Proxy to Internet	If you prefer to add another level of security, you do not have to connect directly from Aduva Director to the Internet Aduva Servers. You can go through a local Web proxy server, or a local SOCKS server. If your organization is using this scheme, you must configure the Preferences for the proxy server.
Local connection	If you prefer, you can set up local versions of the Aduva Servers. This option is mandatory if you are planning on using the PCE. It might be optimal for your organization if speed is a priority. The main drawback to the local option is planning and implementation of an update scheme—making sure you have the latest components in your local version of the Knowledge Base.

### Components

Linux components are culled from numerous commercial and public domain sources, tested for successful installation on a large variety and number of configurations and images, and stored on the Aduva Servers for safe download.

The components are structured into a tree:

- ▶ Nodes - recursive categories of components
- ▶ Elements - the final category that holds one single type of component
- ▶ Leaf - a logical structure that points to one blob
- ▶ Blob - the actual downloadable component; an rpm, a driver, a module.

### Linux dependencies

The Servers contain rules for operational installation of any component: information on the set of dependencies, reverse dependencies, and conflicts between components.

*Dependencies* and *dependency conflicts* are an important aspect of Linux operations. Most Linux components depend upon the prior installation of existing libraries or other packages to operate in known system configurations. These other components are dependencies.

To install a component manually, without using Aduva Director, you must discover and install the complete dependent component list before you can install or run the original component. Installing a component with Aduva Director means that dependency issues are taken care of automatically.

When different components need incompatible versions of dependent components, a dependency conflict arises. For example, one application depends on the existence of a specific version of a certain library. Another application demands a different, and incompatible, version of the same library. Installing both applications causes a dependency conflict. Aduva Director finds potential conflicts, warns you of them, and suggests alternatives before installing a component.

## Summary of the Aduva Director configuration

The Aduva Director entities work together in the following manner:

1. You create a job on the Console.
2. When you run the job, the Engine picks it up and distributes it to the selected Agents.
3. The Agents pull the rules and components from the Servers to complete the job. Each Agent runs its own job in the way that is most efficient for its own configuration. You can create a single job with the Console, send it to hundreds of Agents, and for each Agent, the job may be completed in a different way.
4. The job results are sent to the Engine, which updates the Console and the Database.

## System entities

After you have the Consoles, the Agents, and the Engine set up and communicating with each other and with the Servers, you can begin to customize the Aduva Director system.

Images with the Director Agent rpm installed are Hosts. You can combine Hosts into Groups, to simultaneously send jobs to all Hosts in a Group.

Users of the Console are Managers. There are different levels of permissions for managers, as listed and described in Table 4.

*Table 4 Aduva Director Manager types*

Manager type	Description of permissions
Aduva Director	The highest level user, this is the only user that can create and edit groups and managers, and control all Hosts, modifying any of their Aduva-supported components.
System Administrator	This type can create deploy-profile jobs and control the permitted components of remote images in the permitted groups.
Auditor	This type is read-only; it can monitor Hosts' inventories and their compliance to a profile by running verify-profile jobs on remote images in the permitted groups.

System Administrators and Auditors are restricted to Hosts and components that an Aduva Director manager assigns to them as Permitted.

## Job entities

Managers control Agents by using the Console to create jobs and to send them, over the Engine, to the Agents. Jobs are a list of tasks. Some jobs are simple: Install this package. Some are more complex: Use this image configuration and change the Agents until they comply with it.

Almost every job that you send will have more tasks involved than you asked for, because the Director takes care of installing, upgrading, or uninstalling packages that conflict with, or are needed by, the packages you specifically asked for.

### ***Job Resolve***

The Resolve is a set of algorithms that describe all possible solutions for the configuration and maintenance of supported components. The Resolve for a job is run by the Agents, so its solution is based on the needs of—and the most cost-effective solution—for each image.

### ***Inventory jobs***

The Agent compiles a list of all recognized components on its image; this is an Inventory. The Console gets a full list of available components from the Servers. In the Console, in the Inventory window, you can see the inventory of a Host compared with the full inventory: what is installed, what can be upgraded, and so on. You can create a simple job by selecting components in the Inventory window for installation, upgrade, or uninstallation.

Inventory jobs help you to manage Hosts easily. Dependency issues are taken care of automatically, and you can send one job simultaneously to a group of Hosts.

### ***Profile jobs***

Inventory jobs are easy and simple. Aduva Director provides a more complex feature for image management.

A *profile* is a definition of the component configuration of an image; what its inventory should, or should not, include. For example, how would you define a Web server? What components should exist on all Web servers? What components should not be on these images?

After you create a profile, you can use it to verify that Hosts comply with it, and you can deploy the profile; that is, actually change the inventory of Hosts until they do comply with the profile.

For example, you create a profile called WebServers that defines what your Web servers should have installed and what they should not have installed. You have a group of servers already set up. You run the WebServers profile as a verify job.

The results show you what has to be done on each of the servers to make them comply with your definition of a Web server. Then you can run the profile as a deploy job, actually installing the necessary components, uninstalling those you don't want, and letting Aduva Director automatically install, upgrade, or uninstall dependent components.

## **Private Component Editor (PCE)**

The Private Component Editor (PCE) expands the power of Aduva Director to include your own files and scripts. Using the PCE Web application, you can upload proprietary components to local Aduva Servers and thus allow Aduva Director users to easily distribute these private components to multiple Hosts. In a later section we describe the details of the Private Component Editor and explain how to use it.

## **Aduva Director installation overview**

The installation process is comprised of installing RPMs and modifying a few configuration parameters. We take you step by step through the process to install and configure the entire Aduva Director system. While we cover many possibilities, the actual implementations that you need may be slightly different from those given here.

Be sure to read the installation instructions in the Aduva Director documentation before performing a complete installation.

In this section, an instruction to input a command in a command line terminal begins with the hash sign (#).

## Environment

In our environment, we installed the Aduva Director Agent on nine Linux images (the first character is a lower case L):

- ▶ lnx2
- ▶ lnx4
- ▶ lnx5
- ▶ lnx8
- ▶ lnx9
- ▶ lnx10
- ▶ lnx12
- ▶ lnx13
- ▶ lnx17

We also installed an Engine, the Aduva Servers, and PCE on lnx5. We used a dedicated Linux image for these installations, which is optional but can simplify some setup issues.

We installed the Console on a RedHat 6.2 laptop, connected to the 9.12.6.x network.

## Preparation

Check the following requirements before you begin the installation.

- ▶ Make sure that you have root permissions. All Aduva Director installation procedures are made as root.
- ▶ Make sure that the systems that will be Consoles have X windows.
- ▶ Make sure that the Engine has TCP/IP connectivity to Agents and the Console.
- ▶ Make sure that the Agents and the Console have TCP/IP connectivity to the Engine and Aduva Servers.
- ▶ Make sure that the Aduva Servers have TCP/IP connectivity to parent Aduva Servers (either a higher proxy, or the public Aduva Servers over the Internet), and to the Agents and Console.
- ▶ Make sure that the machine with the PCE Web application has Apache, a browser, and TCP/IP connectivity to the Aduva Servers.

**Tip:** In most common scenarios, all systems would be able to see each other, but some firewall setups may change this. Advanced firewalls, remote Agent installations, and components in a DMZ can be accommodated.

Verify TCP/IP for these connections:

**Aduva Servers <- Console -> Engine <-> Agent -> Aduva Servers**

## Installing MySQL

The Engine requires an SQL database; MySQL, by default. For simplicity, we installed it on the same Linux image, but the Engine can be configured to use a remote MySQL server. For our environment, we installed the MySQL server, client, and library of version 3.22.32 for the S/390.

Follow this procedure to install the packages. Make sure you install them in the order listed.



### To install MySQL:

1. Install the MySQL server: `# rpm -ivh mysql-3.22.32.s390.rpm`
2. Install the MySQL library: `# rpm -ivh mysql11ib-3.22.32.s390.rpm`
3. Install the MySQL client: `# rpm -ivh mysqlnt-3.22.32.s390.rpm`

MySQL starts automatically. To verify that it is running: `#ps -ef | grep -i mysql`

## Installing the Aduva Director packages

The order in which the Aduva Director rpms are installed is important, because the Aduva Database is a dependent component of the Aduva Engine; the Engine won't install if the Database is not already installed. If you follow the steps given here, you should have no problems.

There are different versions of the rpms, depending on which Linux distribution is installed. You'll be installing the S/390 version for the Database, Engine, and Agents. Make sure that you correctly match the Console rpm with the distribution installed on the Console PC. The supported Linux distributions are: Red Hat 6.x, Red Hat 7.x, and SuSE.

The Database and Engine are usually installed on the same image. The Agent must be installed on every image that will be managed by Aduva Director.

Remember to log in with root permissions before installing.

### To install the Aduva Director packages:

1. Install the Database.  
`# rpm -ivh director-db-1.3-1.s390.rpm`  
It loads a fresh database into MySQL.
2. Install the Engine.  
`# rpm -ivh director-engine-1.3-1.s390.rpm`  
The Engine starts automatically. Some errors may be reported.
3. Install the Agent rpm on every Linux image that you want to control with the Aduva Director.  
`# rpm -ivh director-agent-1.3-1.s390.rpm`
4. Install the Console on a standalone PC with network connections to the mainframe:
  - a. Verify the Linux distribution and version of the machine that will be Console.  
`# cat /etc/issue`  
The output contains the name of the installed Linux distribution. It should be similar to:  
Red Hat Linux release 6.2  
**Note:** It is possible (and not difficult) to change the contents of /etc/issue. If you have any doubt about the authenticity of the information displayed, consult with your system administrator.
  - b. Install the Console package that matches the PC's Linux distribution.  
`# rpm -ivh director-console-1.3-1.i386.rpm`
5. Install the Aduva Director PCE on an image on the mainframe:
  - a. Make sure that Apache is installed on this image.
  - b. Install the PCE package.  
`# rpm -ivh aduva-pce-1.0-1rc5_3.s390.rpm`

- c. Add the contents of pce\_httpd.conf to Apache's httpd.conf.  
`# cat /usr/local/aduva/pce/bin/pce_httpd.conf>>/etc/httpd/httpd.conf`
- d. Restart Apache.  
`# /etc/rc.d/apache restart`
6. Install the local Aduva Servers.  
`# rpm -ivh director-servers-1.3-1.s390.rpm`  
**Note:** You may not need to do this procedure. Before you begin, make sure that your company wants to install the Aduva Servers, and that you are the person to do it.
7. If desired, you can verify that the Aduva Director packages were successfully installed.  
`# rpm -qa | grep -i director`  
 The output should contain the names of the Aduva Director rpms you just installed.

## Configuring parameters

First, configure the Aduva Servers to be proxy servers for Aduva Director communication. Then, configure the Engine, the Agents, and the Console to recognize each other and the servers. To perform these procedures, you need to know the hostnames or IP addresses of all of the Aduva Director entities: the Aduva Servers, the Engine, the Agents, and the Console.

### To configure the local Aduva Servers:

1. Go to the Servers configuration file directory.  
`# cd /usr/aduva/bin`
2. Find the configuration file.  
`# ls`  
 Locate the file named aduva.rc. This file contains the configurable options and their default values. We will create a file called .aduva.rc (notice the dot at the beginning of the filename), which will override the default values with values that fit your local environment.
3. Open two terminal windows or text editing windows.
4. In one window, open the aduva.rc file.  
 For example, if you like the vi editor: `# vi aduva.rc`
5. In the second window, create and open a file named .aduva.rc.  
`# vi .aduva.rc`

From the original file (aduva.rc), copy the following parameters and paste them into the empty override file (.aduva.rc), giving the values shown here or those that describe the local environment:

```
( server,client,console ) ( connection.__server.db_server_name, "rs390.aduva.com" );
( all ) ( connection.__server.blobs_server_name, "bs390.aduva.com");
( all ) ( connection.server.__names.proxy_server_name, "-" );
( all ) ( connection.server.port.__group1.proxy_server_port, 8080 );
( server ) ( connection.__server.blob_servicing_port, 82 );
( server ) ( connection.__server.rule_servicing_port, 81 );
( all ) ( connection.__security.secure_connection,true );
```

The values in the override .aduva.rc file are based on this environment:

- a. rs390.aduva.com is the parent Rules server, from which the local rserver will be updated.
- b. bs390.aduva.com is the parent Blobs server, from which the local bserver will be updated.

- c. If there is a Web proxy server, in the proxy\_server\_name parameter, type the hostname or IP address of the proxy.
- d. If there is a Web proxy server, in the group1.proxy\_server\_port parameter, type the open port of the Web proxy server.
- e. In the blob\_servicing\_port parameter, type the port number on which the local bserver is to run.
- f. In the rule\_server\_port parameter, type the port number on which the local rserver is to run.

**Restriction:** Ports 80 and 443 are reserved for the HTTP traffic of the PCE and cannot be used for any other entity of the Aduva Director system.

- g. Make sure that the value of secure\_connection is set to true.

**Note:** If this value is set to true, then Aduva Director will always check that the components are coming from the public Aduva Servers when the local Aduva Servers need to update their Knowledge Base.

- 6. Close the default aduva.rc file (it's important that this file *never* be changed).

#### To configure the Engine:

- 1. On the Engine's image, open the default aduva.rc.

```
# cd /usr/local/aduva/director_engine/bin
# vi aduva.rc
```

- 2. Create the override .aduva.rc.

```
# vi .aduva.rc
```

From aduva.rc, copy that parameters listed below, then paste them into .aduva.rc and give them the values shown here:

```
( all ) ( invisible.server.blobs__general.blob_server_name,"lnx5" );
( all ) ( invisible.server.blobs__general.blob_server_port,"81" );
( all ) ( server.rules.__general.db_server_name, "lnx5" );
( all ) ( server.rules.__general.server_port, "82" );
( all ) ( engine.__general.distrizor_port, "8088" );
( all ) ( engine.__general.distrizor_host, "lnx5" );
( engine ) ( engine.engine_security.__general.rsa_passphrase, "-");
( all ) ( server.__general.secure_connection,false );
```

The values are based on this environment:

- a. Note that lnx5 is the hostname of the image where the rserver, the bserver, and the Engine are installed.
- b. The ports 81 and 82 are open for Aduva Director transmissions.

**Reminder:** Ports 80 and 443 are reserved for the HTTP traffic of the PCE and cannot be used for any other entity of the Aduva Director system.

- c. secure\_connection must be false to connect successfully with local Aduva Servers.

- d. If you leave the dash (-) in the `rsa_passphrase` parameter, you'll have to enter the PEM passphrase that you created when you generated the keys (see "Generating encryption keys" on page 12) after starting the Engine.

As an alternative, you can replace the dash and enter the passphrase. In this case, the passphrase is entered automatically and the Engine starts without asking for it.

#### To configure the Agents and the Consoles:

Follow the same procedure as for the Engine on the images that contain the Agent daemon or the Console application. The `aduva.rc` file is found in the following directories:

Agent	<code>/usr/local/aduva/director_agent/bin</code>
Console	<code>/usr/local/aduva/director_console/bin</code>

**Note:** When you are modifying the Agents and the Consoles, the `distrizor_host` value is the hostname or IP address of the Engine image.

## Generating encryption keys

Aduva Director uses public/private keys encryption in its communications between the Consoles, the Agents and the Engine. You generate the encryption keys with a `keygen` application that comes with Aduva Director. When you installed Aduva Director, the `keygen` ran automatically and created a default public/private key pair. Of course, using the default key isn't recommended, so the next procedure explains how to generate your own keys.

#### To generate a public/private key pair:

1. On the Engine, go to the directory where the `keygen` application is installed:  
`# cd /usr/local/aduva/director_engine/bin`
2. List the contents of the directory: `# ls -a`  
You should see the **aduva\_key\_gen** application.
3. Run the key generation: `# ./aduva_key_gen`  
A new pair of public and private keys is generated. Two files are created: `aduva.public`, and `aduva.private`.
4. You are asked to enter a PEM passphrase, and then to verify it.
5. Copy `aduva.public` to `/usr/local/aduva/director_console/bin` on the Console PC, and to `/usr/local/aduva/director_agent/bin` on *every* Agent image.  
The `aduva.private` file remains only on the Engine.

## Activating the Aduva Director System

By now, everything is installed and communications between the Engine, the Agents, and the Consoles are set and secured. Now it is time to power up.

#### To activate the Servers:

1. On the Aduva Servers image, start the servers:  
`# cd /usr/aduva/bin`  
`# ./rserver`  
`# ./bserver`

2. Verify that the Servers are running:
 

```
# ps -ef | grep -i bserver
# ps -ef | grep -i rserver
```
3. If there is a problem, verify that the Console can connect to the Aduva Servers. Use the hostnames and port numbers of the local Aduva Servers:
 

```
# telnet 1nx5 81
# telnet 1nx5 82
```

If the servers are running and telnet is successful, the local Aduva Servers are ready for operations.

#### **To activate the Engine:**

The Engine has to be reactivated after changing its configuration, so even if the Engine is currently running, you must still follow this procedure.

1. On the Engine image, go to the Engine's initiation directory: `# cd /etc/rc.d/init.d`
2. If the Engine was already on, restart it: `# ./director_engine restart`
3. If the Engine was not on, start it now: `# ./director_engine start`

#### **To activate the Agents:**

Run this procedure after reconfiguration of the rc parameters on every image with the Agent daemon installed.

1. On the Agent images, go to the initiation directory: `# cd /etc/rc.d/init.d`
2. If the Agent was already on, restart it: `# ./director_agent restart`
3. If the Agent was not on, start it now: `# ./director_agent start`

#### **To activate the Console:**

1. On the system where the Console is installed, start the Console.
 

```
# director_console
```
2. Wait while the Console starts up. When the Login window appears, you are ready to use Aduva Director.

#### **To activate PCE:**

1. Open a browser on a connected PC.
2. Enter the URL of the PCE (where 1nx5 is the name of the image where you installed the PCE package):
 

```
http://1nx5/aduva-pce
```

## **Using Aduva Director**

In the following sections we describe how to use the Aduva Director, including starting the Main Console, registering and predefining Hosts, creating Groups, and other tasks.

## **Starting the Aduva Director Console**

After you login, the Main Console appears, as shown in Figure 1.

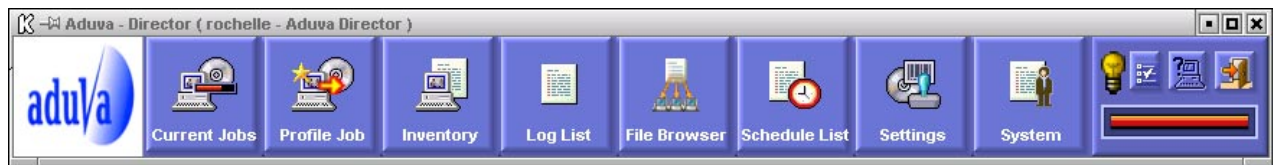














Figure 1 Main Console

Its buttons access all Aduva Director features; they are explained in Table 5 in the order they appear on the Console, from right to left.

Table 5 Main Console buttons

	Click	Action
	Log Out	Close the Console and log out the current manager from the Aduva Director network
	Answer Questions	Confirm or deny actions to complete a resolve
	Preferences	View and edit operations, behavior, directories
	Connection Status	Status of the Console's connection to Engine (on or gray)
	System	Manage Hosts, Groups, and Managers
	Password	Change your login password
	Settings	Create, edit, delete profiles and resolve policies
	Schedule List	View a list of jobs scheduled for the future
	File Browser	View and edit configuration files of remote Hosts
	Log List	View a log of Aduva Director actions
	Inventory	View list of components available for download and lists of components installed on Hosts; compare Hosts; create jobs based on components
	Profile Job	Create jobs based on profiles
	Current Jobs	Monitor ongoing and queued processes

## Defining the system

The first task of an Aduva Director-type manager is to define the Director network. It includes various procedures:

- ▶ Define the structure and hierarchy of the network by creating Groups of multiple Hosts to be managed as single units.
- ▶ Register Agents as Aduva Director Hosts and assign them to Groups.
- ▶ Delegate responsibilities. Create personalized manager names and define their permission levels.

These tasks are carried out in the System window, which is only available to Aduva Director-type managers. You can perform them in any order that you wish: create the managers first, or register Agents first, or create empty Groups first. There are advantages to each order, as explained in following sections.

## Creating managers

This task explains how to create new managers, users of the Console. Use this procedure to create one Aduva Director-type manager, who will fulfill the other tasks. Then, come back to this task to create System Administrator-type and Auditor-type managers after you have created Groups. These lower-level managers have permissions over specific Groups, so you can't create the managers until you have created Groups.

You must log into the Console with the username and password of an Aduva Director-type manager to perform this task. If you have not yet created such a manager, log into the Console with the default `aduva_root` name and 123 password.

Do not use the default manager for more than the first login, to create another manager. The `aduva_root` manager has special permissions to change the preferences of the Engine, which could cause faulty operations or Denial of Services if used incorrectly.

### To create a Manager:

1. In the Main Console, click the System  button.

The System window opens; refer to Figure 2.



Figure 2 System window

- Open the Managers tab and click **New Manager**.  
The New Manager window opens; see Figure 3.



Figure 3 New Manager window

- Type a name and password for the manager, and verify the password.
- Choose the type of manager that you want to create:

Aduva Director	Permissions over all groups and all components, abilities to define the system, and to create and deploy jobs
System Administrator	Permissions over specific groups and specific components, abilities to create and deploy jobs
Auditor	Permissions to monitor specific groups.

- If you create an Aduva Director-type manager, click **OK**.  
If you create a System Administrator-type or an Auditor-type manager, specify the manager's permissions in the next steps.
- For both types of managers, define the Permitted Groups.  
Select a Group in the Groups list and then click **Add**. The Group is copied to the Selected Groups list. These managers may now view only that information that is relevant to these Groups.
- If you create an Auditor-type manager, click **OK**.  
If you create a System Administrator-type manager, continue with the next step.
- Define the manager's Permitted Components.  
Open the Permitted Components tab. The Components list displays the thousands of components available in the Knowledge Base.



9. For each component that you select, click either **Allow** or **Deny**.

Any component that you do not Deny is Allowed. Use the Deny option to specify that certain components are not to be changed by this manager.

If you want this manager to be able to use the File Browser, make sure that *files* is an Allowed component.

10. Make sure that the Permitted Components list is what you want, and then click **OK**.

The New Manager window closes, and you are returned to the System window, where the Managers list shows the new manager's name.

Log in as the new Aduva Director-type manager, and then register Hosts and create Groups. Then, come back to this task and create more managers.

## Registering Hosts

In this task, you will confirm the properties of images that have the Agent installed on them, and thus register them as Director Hosts.

You need to be an Aduva Director-type manager to perform this task. If you have not yet created such a manager, do so before beginning this task. Then log into the Console with the Aduva Director's name and password.

### To register Hosts:

1. In the Main Console, click the System button.

The System window opens. In the Hosts list, there is a group called Unregistered Hosts.

2. Open the Unregistered Hosts Group and select a Host.

The Properties button is enabled.

3. Click **Properties**.

The Host Properties window opens; refer to Figure 4.

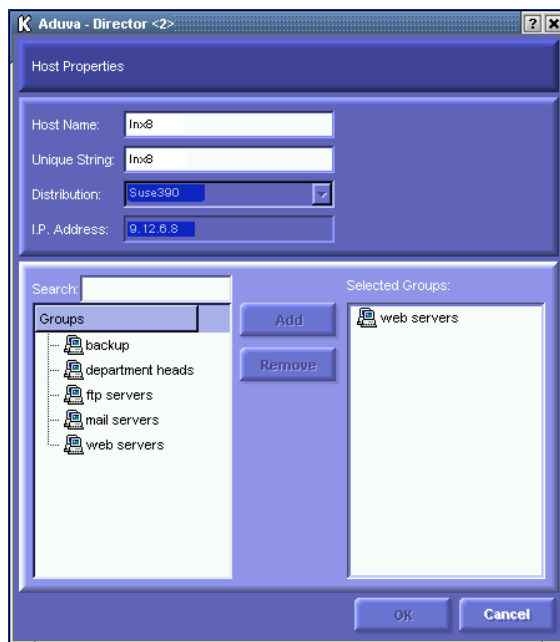


Figure 4 Host Properties window

The properties of this image are displayed, mostly as read-only. You can change the Host Name if you want; it affects only the listing of the image in Aduva Director, not the image itself.

4. In the Groups list, select a Group, click **Add**, and then click **OK**.

The Host Properties window closes and you are returned to the System window. If you expand the Group, you will see the Host assigned to it.

Assigning a Host to a custom Group is an optional feature. You can register a Host simply by opening the Host Properties window once and then clicking **OK**. The Host is automatically added to the All Hosts Group. However, there are advantages to assigning a Host to a custom group:

- ▶ You can send jobs simultaneously to all Hosts in a Group.
- ▶ You can assign this Host to fall under the responsibilities of a System Administrator-type or Auditor-type manager. These managers cannot control or monitor Hosts that are not assigned to a custom Group.

The Unregistered Hosts group may not appear in the System window for various reasons:

- ▶ If the Agents are already registered, they appear in the All Hosts Group.
- ▶ If the Agent daemon has not yet been installed on images, the All Hosts Group will be empty. You may predefine Hosts if you wish; see the next task.

## Predefining Hosts

By default, images that contain the Agent daemon automatically begin to communicate with the Engine. This task explains how to override the default automation and manually predefine Hosts.

You need to be an Aduva Director-type manager to perform this task. If you have not yet created such a manager, do so before beginning this task. To enable this task, you must start the Console in Manual Host mode—so if it is already open in default mode, close the Console before beginning.

**Note:** You can change the mode of the Console (in the Console's Preferences) to allow manual Host creation by default.

### To predefine a Host:

1. Start the Console with the Manual Host option:  
`director_console -manual_host_create true`
2. Log in as an Aduva Director-type manager.
3. In the Main Console, click the **System** button.  
The System window opens. The New Host button is available.
4. Click **New Host**.  
The New Host window opens; see Figure 5 on page 19.

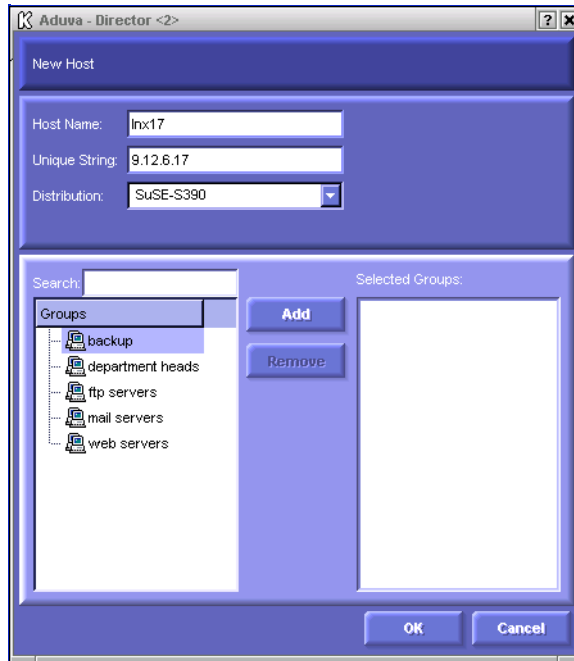


Figure 5 New Host window


5. Type the following details:

Host Name	This string is for the Director; it does not affect the image.
Unique String	This is the real name of the computer; it affects the image's connections to the servers.
Distribution	Select the Linux distribution that is installed (or will be installed) on the image.

6. Select a group to which this Host will be assigned and then click **Add**.

This Host is assigned to the group that appears in the Selected Groups list. You can assign it to as many groups as you want.

7. Click **OK**.

The New Host window closes and you are returned to the System window. This Host appears in the groups to which you assigned it. If it does not yet have the Agent installed on it, it will be marked as disconnected .

## Creating groups

This task explains how to create custom groups. Creating groups is not mandatory. There are two default groups: *All Hosts* and *Unregistered Hosts*. However, creating custom groups is essential if you want to create lower-level managers with specific permissions, or to simultaneously control and monitor multiple Hosts.

You need to be an Aduva Director-type manager to perform this task. If you have not yet created such a manager, do so before beginning this task. Then log into the Console with the Aduva Director's name and password.

**To create custom Groups:**

1. In the Main Console, click **System**.

The System window opens.

2. Do *one* of the following:

- Click **New Group**. The New Group window opens with empty properties.
- Or select Hosts from the Hosts list and then click **New Group**. The New Group window opens with the selected Hosts already included in it.
- Or select an existing group from the Hosts List and then click **New Group**. The new group will be a descendent of the existing group.

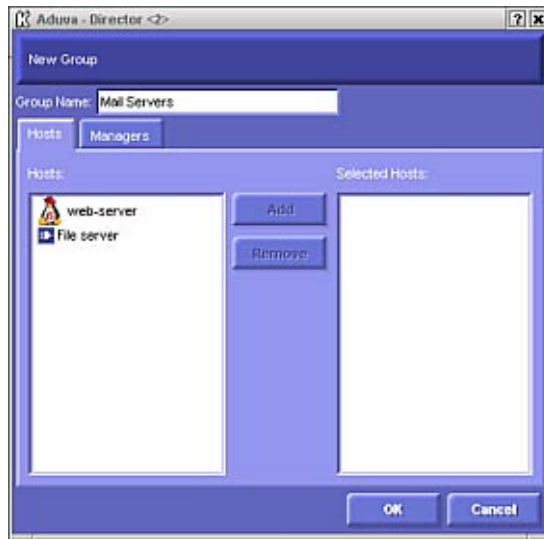


Figure 6 New Group window

3. Type a name for the group.
4. Select the Hosts that you want to put into this group and then click **Add**.

The Selected Hosts list displays the Hosts that are to be assigned to this group.

**Note:** If there are no Hosts registered yet, you can leave the group empty and assign Hosts later.

5. Open the Managers tab; see Figure 7 on page 21.



Figure 7 New Group window - Managers tab

If System Administrator-type and Auditor-type managers have been created, they are listed here. If you have not yet created lower-level managers, this list is empty.

Aduva Director-type managers have permissions over all groups, and so do not appear in the Managers tab; they will automatically have permissions over this new group.

The next step is optional; you do not have to assign lower-level managers to this group. In addition, you can assign groups to managers when you create the managers.

6. If you want this group to be a Permitted Group of a lower-level manager, select the manager's name and then click **Add**.
7. Add as many managers as you want and then click **OK**.

The New Group window closes and you are returned to the System window. The new group appears in the Hosts list and in the Permitted Groups list of the lower-level managers to which it was assigned.

## Editing system entities

This task explains how to edit and delete Hosts, groups, and managers.

You need to be an Aduva Director-type manager to perform this task. If you have not yet created such a manager, do so before beginning this task. Then log into the Console with the Aduva Director's name and password.

### To edit Hosts, groups, and managers:

1. In the Main Console, click **System**.  
The System window opens.
2. Select the Host, group, or manager that you want to edit and then click **Properties**.  
The appropriate Properties window opens; see Figure 8 on page 22.

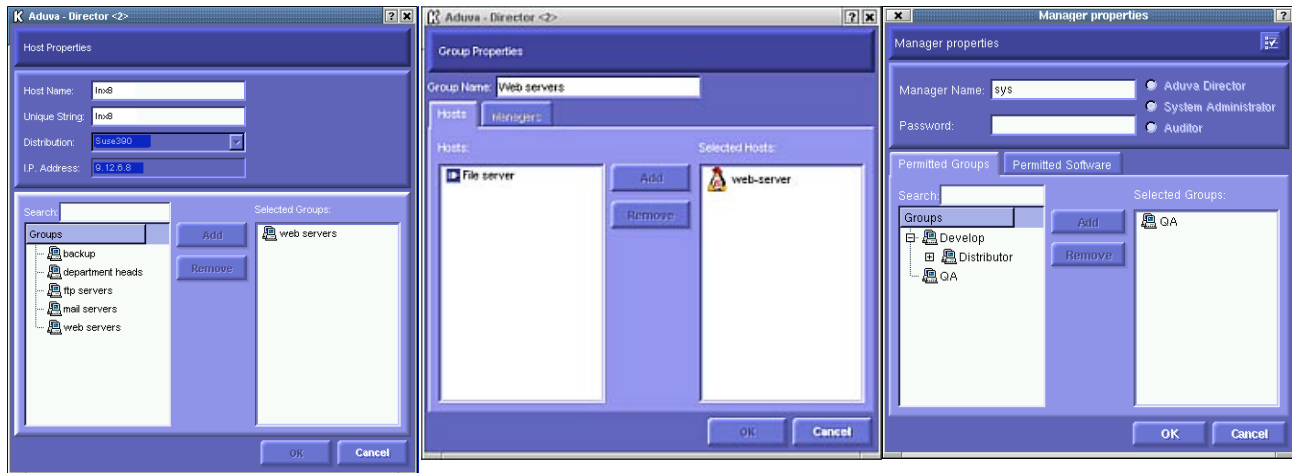


Figure 8 Properties Windows for Hosts, groups, and managers

3. Change any of the following properties:

Host Properties	<p>In the Host Properties window, you can change the Aduva Director name of the Host and the groups to which it belongs.</p> <ul style="list-style-type: none"> <li>If you started the Console with the Manual Host option, you can also change the Unique String.</li> </ul>
Group Properties	<p>In the Group Properties window, you can change the list of Hosts contained in the group and the name of the group.</p> <ul style="list-style-type: none"> <li>You <i>cannot</i> edit the All Hosts group or the Unregistered Hosts group.</li> </ul>
Manager	<p>In the Manager Properties window, you can change the password.</p> <ul style="list-style-type: none"> <li>You cannot change the manager type.</li> <li>For System Administrator-type and Auditor-type managers, you can also edit their Permitted Groups list, as long as you leave each manager at least one selected group.</li> <li>You cannot change the Permitted Components of System Administrator-type managers.</li> </ul>

4. If you want to remove a Host from a group, do *one* of the following:

- In the System window, select the Host from the group and click **Delete**. In the confirmation dialog box that opens, click **Delete** again.
- Or, in the Host Properties window, select the group from the Selected Groups list and then click Remove.
- Or, in the Group Properties window, select the Host from the Selected Hosts list and then click **Remove**.

5. If you want to delete a group, select the group in the System window and then click **Delete**. In the confirmation dialog box that opens, click **Delete** again.

6. If you want to delete a manager, select the manager in the System window and then click **Delete**.

In the confirmation dialog box that opens, click **Delete** again.

**Note:** You cannot delete a logged-in manager.

## Changing your own password

In this task, System Administrator-type and Auditor-type managers change their passwords. Initially, they are given passwords by the Aduva Director-type manager who created their usernames and properties. Before performing any action, lower-level managers should make sure that they have a private password.

**Important:** This task is relevant only for System Administrator-type and Auditor-type managers. Aduva Director-type managers can change their passwords at any time in the Manager Properties window.

By default, when you log in for the first time, the Change Password dialog box opens and you must change your password and verify the new one before you can perform any Aduva Director action.

This task explains how you can change your password at any time, after logging in as a System Administrator-type or Auditor-type manager.

To change your password:

1. In the Main Console, click the **Password**  button.

You may notice that this is the same icon as the System button for Aduva Director-type managers. The function of this button changes according to the level of the logged-in manager.

The Change Password window opens; see Figure 9.



Figure 9 Change Password window

2. Type and verify a new password.
3. Click **OK**.

The new password takes effect immediately.

## Profile jobs

Now that you have the system entities set up, you can manage and monitor images with jobs.

The following topics explain how to create, deploy, and verify profiles, and how to create and use resolve policies to automate profiles.

## What is a profile

The *profile* is a powerful management tool that enables you to manipulate multiple images in a consistent way while automating many tedious tasks. It is a definition of an image; which components you want this type of image to include, and which to exclude. After you create a profile, you can deploy it or verify it on one (or multiple) Hosts.

When you verify a profile, Aduva Director checks whether the selected Hosts comply with the profile.

When you deploy a profile, Aduva Director makes whatever changes are necessary (installations, uninstallations, upgrades, downgrades) to make the Hosts comply with the profile.

## Default profiles

Aduva Director comes with some profiles already created and ready for you to use; see Table 6. The purpose of these profiles is to run routine maintenance checks.

Table 6 Default profiles

Profile name	Function
System Check	Checks for missing dependent packages
Security Check	Checks for packages to be patched against local exploits
High Security Check	Checks for packages to be patched against remote exploits
Update Aduva	Checks for an updated version of the Aduva Agent
Bug Fixes	Checks for packages to be patched to fix known bugs

If you run these profiles as verify jobs, the results will tell you what tasks must be done to fix all dependency issues, to optimize security, or to update the Agent. If you run them as deploy jobs, Aduva Director will install or uninstall whatever components needed to fix dependencies or optimize security.

## Creating profiles

In this task, you will create a new profile.

You should have a manager name and password. You should already have the Console running and be logged in, before you start this task.

### To create a profile:

1. In the Main Console, click the **Settings**  button.

The Settings window opens; see Figure 10 on page 25.





Figure 10 Settings window - Profile Tab

2. Open the Profile tab and then click **New**.

A blank Profile Editor window opens; see Figure 11.

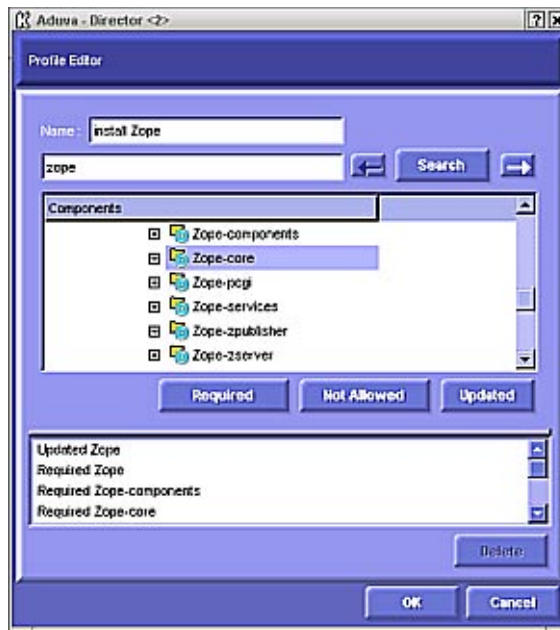


Figure 11 Profile Editor window

If you select an existing profile in the Settings window and then click **Copy** (instead of New), the Profile Editor window opens with the properties of the selected profile. You can now create a new one, based on the selected one.

**Note:** If you are a System Administrator-type manager, the Components list shows only your Permitted Components.

3. Type a name for the profile. The name must be unique in the list of profiles.

If you are a System Administrator-type manager, the list of profiles that you can see contains only those profiles that are concerned with your Permitted Components. If you pick a name that is already taken, even by a profile you cannot see, you will receive an error. In this case, change the name of the new profile.

4. Select a component (this can be an entire category, or a single package, or anything in between) and assign it a rule, as follows:

Required	The image <i>must</i> have this component. If it does not, the image conflicts with the profile. During a deployment of this profile, the component, or at least one of its descendents, will be installed on the image.
Not Allowed	The image must <i>not</i> have this component. If it does, the image conflicts with the profile. During a deployment of this profile, the component, and all of its descendents, will be uninstalled from the image.
Updated	<p>The version of this component is checked against the versions available on the Knowledge Base. If there is a newer version available, the component will be updated.</p> <ul style="list-style-type: none"><li>• If the component, or at least one of its descendents, is not installed on the image, Updated does nothing. Therefore, it's good practise to mark all components that are Updated also as Required.</li></ul>

5. Click **OK**.

The Profile Editor window closes and the profile is created. It appears in the Settings window.

When you create a profile, other managers see it as read-only. That means they can deploy it, or use it for verification. They can copy it to create their own profile based on yours. However, they cannot delete or change it.

For optimal speed and job conditions, numerous activities should be sent as multiple jobs, rather than one large one. Any job that deploys a profile may include more tasks than the ones you specified (the added tasks take care of dependency conflicts). Therefore, you might want to split up one long profile into many smaller ones.

## What is a resolve policy

When you send a job to Hosts, every Agent finds the most cost-effective (in terms of number of tasks and resource consumption) solution for its own Host to complete the job. This is the *resolve*.

Aduva Director provides the *resolve policy* feature. Following are the advantages for creating and using a resolve policy are:

- You can predefine some actions as unacceptable, which will reject any resolve that contains these tasks. Thus, you can determine the direction of the resolve, without giving up on the automated features. Here's a typical scenario.

### Example:

You are told to install the latest version of a security package on a hundred Linux images.

Some of these images are still operating on an old kernel version, and you don't want them to recompile just for this one package. Therefore, you create a resolve policy that does not allow kernel compilations to be part of the resolve.

When you run the job on the group of images (all one hundred at one time!), those images with old kernels either find a way to install the new package in the present environment, or they do not do the job.

The job succeeds on the newer kernels. You don't have to go through the backend configurations to determine which Hosts should receive the job and which should not; you can send it out to everyone and the Hosts that you didn't want to do the job, won't do it.

- You can predefine some actions as automatic, so they will be carried out without asking for user-intervention. By default, you are asked for confirmation before any task is carried out. Again, we offer a scenario.

**Example:**

You run a verify job and get a list of tasks that need to be done on a group of Hosts to complete a job. You go through the tasks and decide that you want to do them all. Let's say that all of these tasks are relevant to software components only.

You create a resolve policy that automatically accepts any task done to a software component. Now you can deploy the job, with this resolve policy, and run it on the Hosts overnight or while you're doing something else. The job runs automatically on all Hosts.

## Creating a resolve policy

In this task, you will create a resolve policy.

You should have a manager name and password of either an Aduva Director-type or a System Administrator-type manager. Resolve policies are relevant only for deploy jobs, not verify jobs, and Auditor-type managers cannot create deploy jobs. In addition, you should already have the Console running and be logged in, before you start this task.

**To create a resolve policy:**

1. In the Main Console, click **Settings**.

The Settings window opens, displaying the Resolve Policy tab; see Figure 12.



Figure 12 Settings window - Resolve Policy tab

2. In the Resolve Policy tab, click **New**.

A blank Resolve Policy Editor window opens; see Figure 13 on page 28.

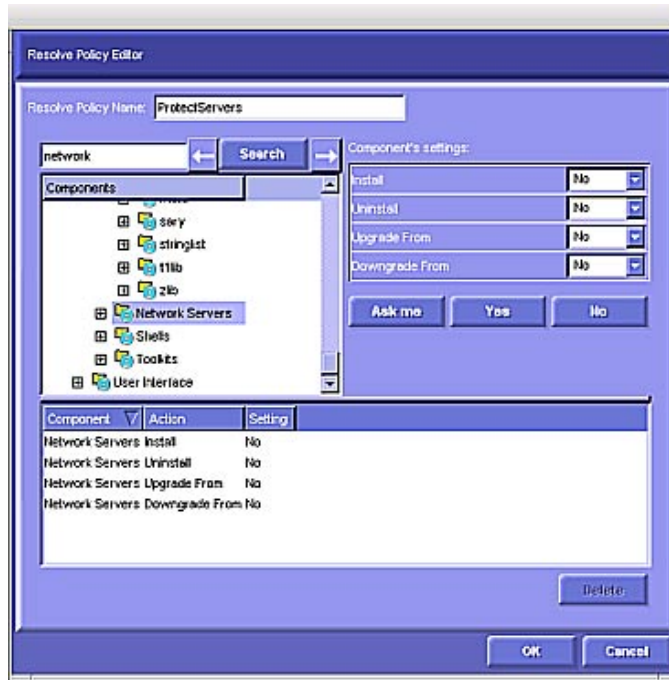


Figure 13 Resolve Policy Editor window

If you select an existing resolve policy in the Settings window and then click **Copy** (instead of New), the Resolve Policy Editor window opens with the properties of the selected resolve policy. You can now create a new one, based on the selected one.

If you are a System Administrator-type manager, the Components list in the Editor shows only your Permitted Components.

3. Type a name for the policy. The name must be unique in the list of resolve policies.

If you are a System Administrator-type manager, the list of resolve policies that you can see contains only those that are concerned with your Permitted Components. If you pick a name that is already taken, even by a resolve policy that you cannot see, you will receive an error. In this case, change the name of the new resolve policy.

4. Select a component (this can be an entire category, or a single package, or anything in between) and for each possible action, assign a policy as follows:

Ask Me	Pause the resolve and ask for user confirmation before performing the selected action on this component or its descendants
Yes	Carry out the necessary action on this component, or its descendants, without asking for confirmation
No	Reject a resolve that includes the selected action on this component, or its descendants; fail job for selected Host if it can't be done without this action

5. Add as many component-action policies as you want.

6. Click **OK**.

The Resolve Policy Editor window closes and the resolve policy is created. It appears in the Settings window.

When you create a resolve policy, other managers see it as read-only. They can deploy it, or use it for verification. They can copy it to create their own resolve policy based on yours. They cannot delete or change it.

## Using profiles and resolve policies

After creating profiles and resolve policies, you use them in jobs. Profiles can be deployed or verified. Resolve policies are relevant only for deploy jobs.

The following tasks explain how to select options and resolve policies for profile jobs and how to schedule profile jobs.

### Creating profile jobs

This task explains how to use a profile. You can deploy it on Hosts (actually changing component inventories to comply with the profile), or verify it on Hosts (find out what tasks would need to be done to make the Hosts comply).

You should have a manager name and password. You should already have the Console running and be logged in, before you start this task.

If you want to run a deploy job, you must be logged in as an Aduva Director-type or System Administrator-type manager. Auditor-type managers may run only verify jobs.

#### To create a profile job:

1. In the Main Console, click **Profile Job**.



The Job Editor window opens; see Figure 14.

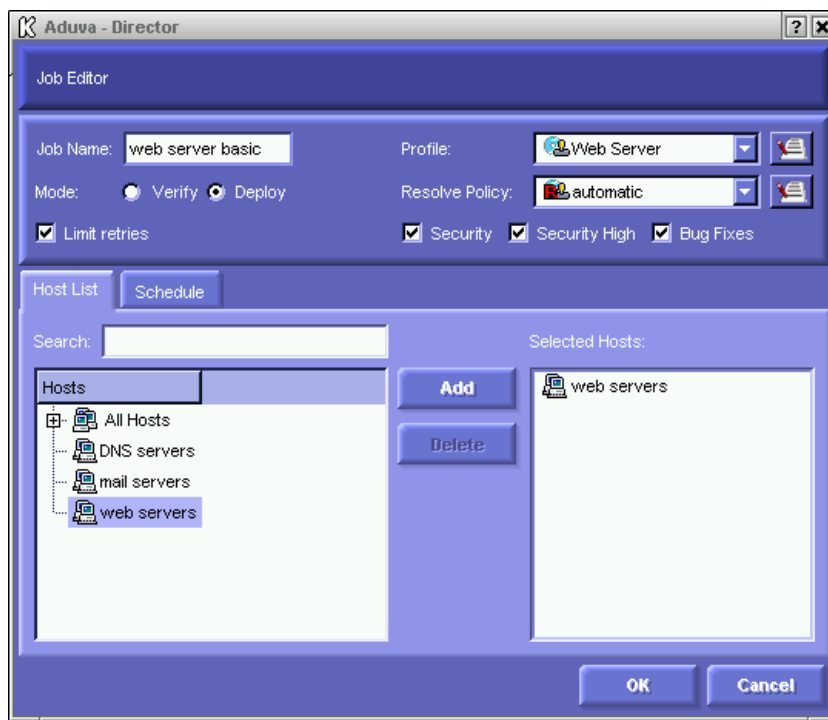




Figure 14 Job Editor window

2. Type a name for the job.
3. Select the type of job you want to create, as follows:

Verify	Check the inventory of the Hosts for compliance with the selected profile, without affecting them
Deploy	Actually change the inventory of the Hosts to comply with the selected profile

**Note:** Auditor-type managers do not have the option of selecting deploy mode.

4. Select a profile. Use *one* of the following methods, depending on what you wish to achieve:
  - To run a profile that you or another manager created, select it from the list.
  - To create a new profile, click **Editor**.  A blank Profile Editor opens.
  - To edit an existing profile, select one from the list, and then click **Editor**. A Profile Editor window opens with this profile already entered in it.
  - To run a check for dependency issues, select the default System Check profile.
  - To run a security check job, select either the Security Check profile (to check for software that can be upgraded against local exploits) or Security High Check (to check for software that can be upgraded against remote exploits).
  - To run a check for packages containing known bugs, select the Bug Fixes profile.
  - To check for, and upgrade to, a new released version of the Aduva Director Agent, select the Update Aduva profile. This default profile will upgrade the Agent daemon on a selected Host, if that Host does not have the latest version.
5. If you want, select a resolve policy.

You can choose one from the list, or open the Resolve Policy Editor by clicking **Editor**. 
6. If you want, check **Limited Retries**.

This option means that disconnected Agents do not waste resources; if an Agent cannot be reached after a specific number of tries, it is marked as disconnected by Aduva Director.

If you do not select this option, the job will run on the disconnected Agent as soon as it rises again.
7. If you want, check **Security**, **Security High**, or **Bug Fixes**.

These options ensure that the job will not cause known security issues or install components that are known to contain bugs.
8. Select a Host or group and then click **Add**.

It is added to the Selected Hosts list and will receive the job.
9. If you want, schedule the job.
10. Click **OK**.

The Job Editor window closes. If the job is scheduled for Now (default), it is sent to the Current Jobs window. If this job is scheduled for a future or periodic deployment, it is sent to the Schedule list.

## Scheduling profile jobs

In this task, you will learn to set a future or a periodic schedule for a job in the Job Editor window.

You should already have the Console running and be logged in. You should have followed the previous task and created a job.

### To schedule a job:

1. In the Job Editor window, open **Schedule**; see Figure 15 on page 31.

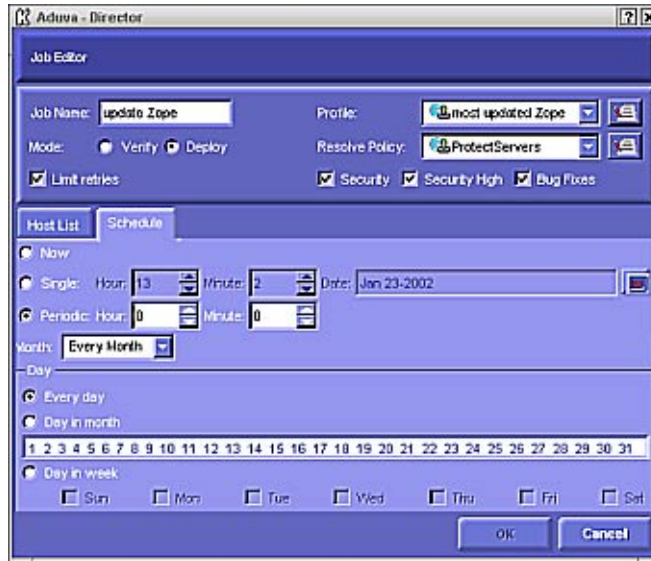


Figure 15 Job Editor window - Schedule tab

2. Select a schedule type:
    - To run the job immediately, leave **Now** selected by default.
    - To run the job at a future time, select **Single** and set a time and date.
    - To create a routine for multiple runs, select **Periodic**.
  3. If you selected Periodic, set a time and then select the other details of the routine:
    - Select either Every Month or a specific month.
    - Select either every day, or once a month, or days in every week.
- After a job is scheduled, it appears in the Schedule List.

## Inventory jobs

Using Aduva Director, it is easy to see which components are installed on Hosts, and which components are available for installation from the Knowledge Base. The installed components are the *inventory* of a Host.

Besides viewing a Host's inventory, you can create a job based on changes you want to make to it. You can select available components to install, and you can uninstall or upgrade components. You can store inventory lists and compare them.

## Viewing an inventory

This task explains how to open the Inventory window and understand the information displayed there.

### To view an inventory:

1. In the Main Console, click **Inventory**.



The Inventory window opens; see Figure 16 on page 32.

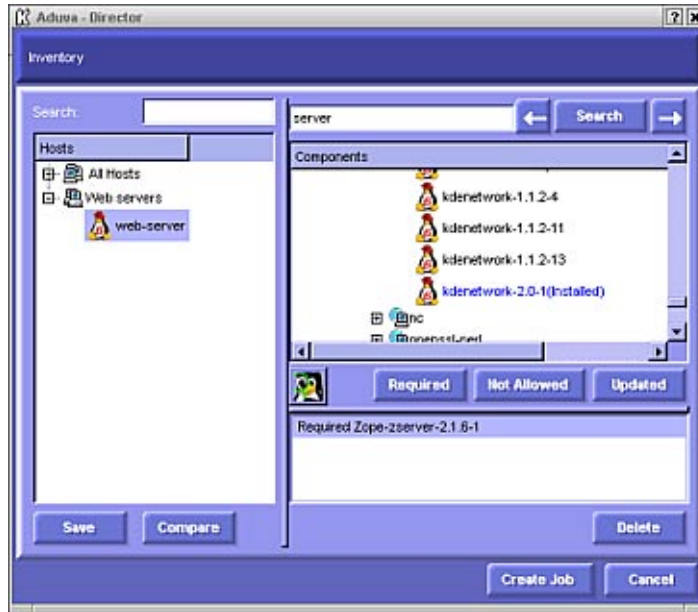


Figure 16 Inventory window

2. In the Hosts list, select groups and Hosts.

If you are a System Administrator-type or Auditor-type manager, you see only the Hosts of your Permitted Groups. In the Components list, you see only the Components over which you have permissions.

Components that are installed on the selected Hosts are marked:

black <b>Installed</b>	At least one child of this component is installed on at least one of the selected Hosts; at least one child is not installed on any of them
blue <b>Installed</b>	The component is installed on at least one of the selected Hosts
no <b>Installed</b> mark	The component is not installed on any of the selected Hosts
a ratio	<ul style="list-style-type: none"> <li>The first number: how many of the selected Hosts have this component installed</li> <li>The second number: how many Hosts were selected</li> </ul>

3. If you want to see the exact list of Hosts that have a component installed, select a component and then click **Component Info**.



The Component Info window opens, displaying the exact Host list. For example, the System Environment component is marked with (Installed 93/105). This means that 93 of the 105 Hosts selected have System Environment components installed on them.

Deeper inside the System Environment component you find the component called SDL. This component is marked with (Installed 5/105) and *Installed* is in blue. This tells you that five of the 105 Hosts selected have one of the supported versions of SDL.



## Creating inventory jobs

This task explains how to create and run jobs based on specific components, rather than on profiles.

You should go through the previous task to understand the information displayed in the Inventory window. This task assumes that you are logged in as an Aduva Director-type or System Administrator-type manager.

### To change the inventory of a Host or group:

1. Select the Hosts.
2. Select a component.
3. Click an action for the selected component: **Install**, **Uninstall** or **Updated**.

In the bottom pane, the command appears: [action][component name].

For optimal speed and job conditions, numerous activities should be sent as multiple jobs. To remove an action from the job, select the relevant line in the pane, and click **Delete**.

4. Enter as many component actions as you want.
5. Click **Create Job**.

The Job Info window opens, displaying a summary of the selected actions and providing the Run Now button to start the job; see Figure 17.

The Job Editor button is also available. This opens the Job Editor with rules already entered as a new profile, called *Job's Profile*.



Figure 17 Job Info window

6. Click **Run Now**.

The Job Info window closes and you are returned to the Inventory window.

7. Click **Cancel**.

The Inventory window closes. The job is sent to the Current Jobs list.

## Comparing inventories

This task explains how to store inventories and how to use them for various jobs (such as to reset computers to a previous state, to make inventories of different images similar, to troubleshoot operational problems, and so on). It has two tasks: store inventories, and then compare them.

**Note:** Comparisons of Hosts with different Linux distributions is not supported; such comparisons would create irrelevant sets of results.

### To store an inventory:

1. In the Inventory window, select a single Host.  
The Save button is enabled.
2. Click **Save**.  
A dialog box opens with the date entered in the text-entry box.
3. Enter a name for this inventory and then click **OK**.  
Keep the date as part of the name. Use the rest of your allotted 120 characters to create a meaningful name for this inventory.

### To compare inventories:

1. In the Inventory window, click **Compare**.  
The Inventory Comparison window opens; see Figure 18.

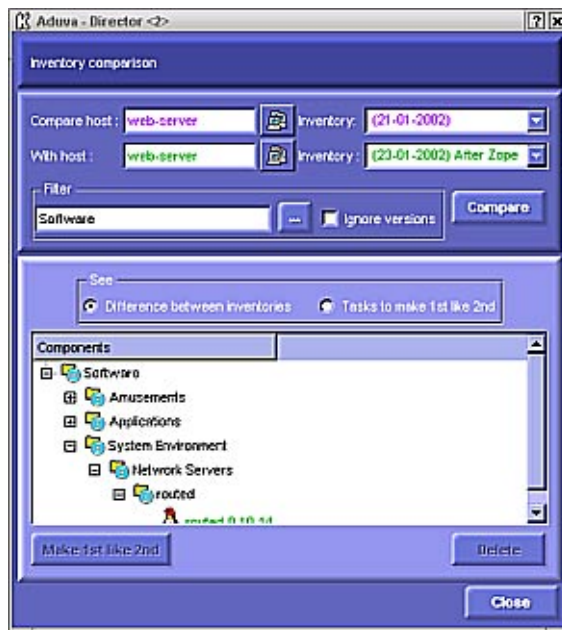



Figure 18 Inventory Comparison window

2. In the Compare Host box, select the Host that you are working on.  
Either enter the Aduva Director Host Name of the Host, or click  and in the Host Selection dialog box that opens, browse to the Host you want.
3. In the Inventory box to the right of the Compare Host box, select either Current Inventory or an inventory that you stored previously.

4. In the With Host box, select the Host that you are using as a base, as the optimal inventory.
5. To the right, select either Current Inventory or an inventory that you stored previously:
  - If you want to restore an image to its previous component inventory, select the current inventory of the Host for the first, and the ideal previous inventory of the same Host for the second.
  - If you want to see how a Host was changed during a period of time, select stored inventories of the same Host, from different times.
  - If you want to compare two Hosts, and change the first to be like the second, select the working Host as the first and its current inventory. Then select the ideal Host and its ideal inventory, either current or stored, for the second.
  - If you want to compare two stored inventories, select the two Hosts and the stored inventories of the Hosts.

**Note:** The second Host may be the same Host as the first, but the stored inventories must be different. (If you select the same Host for both first and second, and the same inventory for each Host, of course there will be no difference!)

6. Click the browse button to the right of the Filter box.

The Component Selection window opens; see Figure 19. Select the type of inventory that you are interested in: hardware, kernel, software, private, or any subcategory of components.



Figure 19 Component Selection window

The more specific (the deeper into the tree of components) you can make the filter, the more meaningful the comparison will be.

7. If you want, you can select the Ignore Versions checkbox.

This will delete from the comparison results any components that are the same except for version number.

8. Click **Compare**.

The results of the comparison are displayed. You can view them either in a tree view (to see the differences between the two inventories simply as differences), or you can view a list of tasks that would be taken if the first inventory were to be changed to be like the second.

If the first inventory is current, and the comparison results in some difference, the “Make 1st like 2nd button” is enabled.

9. If you want to change the inventory of the first Host, click the **Make 1st like 2nd** button.

If the job (to make the first inventory like the second) is not too large, the Job Info window opens, and you can run the job.

If the job is too large, this message appears:

The job exceeds the recommended size of 20 tasks.  
Do you want to continue?

For every task that is sent as part of a job, more tasks are added, to automatically handle dependency issues. If a job contains too many tasks, there is an increased possibility that some part of the job will fail before it can all be done successfully.

You can click **Continue** and try to run the job as is; or you can click **Cancel**, then delete some of the tasks from the Compare window, and click **Make 1st like 2nd** again.

## Job tasks

Whether you create a job from a Host inventory or through a profile, there are some activities that are common to all jobs.

## Monitoring logs

Aduva Director performs many tasks automatically. It does what needs to be done not only to fulfill the specific tasks that you ask for, but also to handle any resulting dependency or security issues. There are windows that provide information about the automatic actions that Aduva Director does for you.

In the Current Jobs window, you can monitor job processes as they occur. In the Log List window, you can view a filtered history of tasks performed.

### Current Jobs window

The Current Jobs window displays details of jobs that are currently running, are in queue to begin, or have been completed but not yet removed from this list. This window shows only those jobs that you created; other managers' jobs are not shown.

To open the Current Jobs window, in the Main Console, click **Current Jobs**.

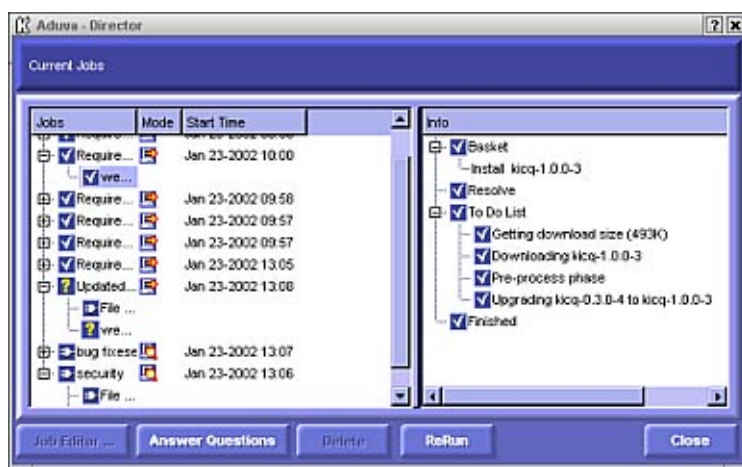




Figure 20 Current Jobs window

Table 7 *Current Jobs window column headers*

Column header	Information displayed
Jobs	The name of the job and its current status icon
Mode	Job type; either Deploy  or Verify 
Start Time	The day/month/year and hour:minute that this job started
Info	Details for the selected job or Host

Jobs cannot have duplicate names; however, you might see the same name appear in this list more than once. This happens if you schedule a job for a periodic run. Every time the Aduva Director Engine starts the job on its routine, it is entered in this list again.

The entries in the Jobs column are expandible. Expand each job name to see the list of Hosts that received this job. Each Host has a status icon, which changes dynamically.

### **Current Jobs window icons and buttons**

The entries in the Jobs column are expandible. Expand each job name to see the list of Hosts that received this job. Each Host has a status icon, which changes dynamically.

Table 8 *Current Jobs window icons*






Icon	Indicates
	Processing the job on at least one of the selected Hosts
	Successful completion on all the selected Hosts
	Failed for at least one of the Hosts
	Paused for a Host until you answer questions
	A Host is disconnected

Table 9 *Current Jobs window buttons*

Click	To
Job Editor	Create a new job based on a current job <ul style="list-style-type: none"> <li>• Opens the Job Editor window.</li> <li>• Enabled when a job name (not a Host) is selected in the left panel.</li> </ul>
Answer Questions	Answer processing questions: should a particular task included in the resolve be carried out? <ul style="list-style-type: none"> <li>• Opens the Answer Questions window.</li> <li>• Enabled if the resolve of a selected job or Host has been paused until you answer these questions.</li> </ul>
Delete	Remove the selected job from the current jobs list <ul style="list-style-type: none"> <li>• If the job is still in process, deleting it from the list stops it.</li> <li>• If the job has already finished or failed, you can still see it in the log list.</li> <li>• Enabled if a job or a Host is selected.</li> </ul>
ReRun	Run the selected job again <ul style="list-style-type: none"> <li>• Enabled if one job or Host is selected.</li> </ul>



### **Job information**

If you select a job in the left-most column, the following information is given in the right-most column: Job Name and Status (same as on the left side), but if the job has a schedule, then (S) is displayed after the name.

- ▶ If the profile is a default check, its name is displayed: System, Security, Security High or Aduva Update.
- ▶ If this job was scheduled for a Single future time, schedule information is displayed.
- ▶ Profile data - the action-component tasks that make up the job.
- ▶ Resolve Policy data - both the set of predefined answers sent with the job (if a resolve policy was selected before the job began) and the answers given interactively during the process of the job.

### **Host information**

If you select a specific Host in the left-most column, the following information is given in the right-most column:

- ▶ The first phase may show any of the following:
  - Waiting - The job is in queue.
  - Basket - Aduva Director has scanned the Host and found the components that need to be installed, uninstalled or updated to make the Host complete the job.
  - If a default profile was selected, a list of tasks are listed under the profile name: System, Security, Security High, or Aduva update.
  - Resolve - Aduva Director has picked an algorithm that, while complying with any No settings of a selected Resolve Policy, will make the Host conform to the profile.
  - Once a Resolve is found for a Host, a To Do list is built:
    - i. The download size of the entire basket is calculated.
    - i. The components are downloaded from the Aduva servers.
    - i. The Host is checked to make sure that it can accept the job actions.
    - i. The actions (install, uninstall, upgrade) are carried out.
  - The final status is either Finished or Failed. The word Finished appears as soon as the To Do list is made up, but the job is not complete until either the Finished  or the Failed  icon appears.

### **Log List**

The Log List shows a detailed history of install and uninstall actions performed on registered Hosts. You will see that actions beyond those that you requested were performed, because Aduva Director provides a seamless install and uninstall of dependent components. The log is built upon a set of filters that you define in the Filters window.

### **Filters window**

In the Filters window, set the criteria for viewing logged events in the Log List.

To open the Filters window, in the Main Console, click **Log List**.





Table 12 Filters window buttons

Click	To
All Actions	Show all actions for the selected component
Install	Show only installation actions for the selected component
Uninstall	Show only uninstallation actions for the selected component.
Add	Add the selected Hosts to the Selected Hosts list
left Delete	Remove the selected action-component filter
right Delete	Remove the selected Host from the filter
OK	Open the Log List according to the filters

To apply the filters, you must:

- ▶ Set a date; either Any Date or a valid Date Range.
- ▶ Select at least one component and specified action.
- ▶ Select at least one Host.

After you set and apply the filters, click **OK**. The Log List appears; see Figure 22.

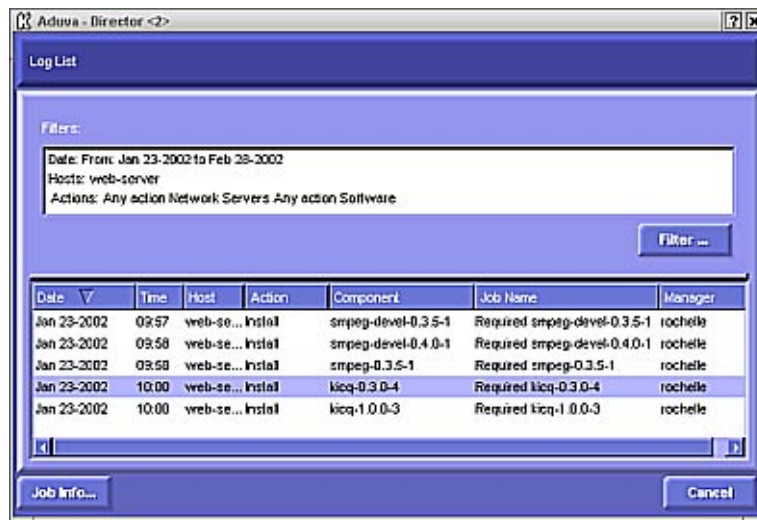


Figure 22 Log List window

The top part of this window, the Filters List, details the filters that you selected for the Log List: date range, Hosts, and actions logged for selected components.

### Log List window buttons and column headers

Table 13 Log List buttons


Click	To
Filter	Return to the Filters window to view or change the filters that you selected
Job Info	<p>View more information on a selected job</p> <ul style="list-style-type: none"> <li>• Opens the Job Info window, which also has access to job editing and rerunning functions.</li> <li>• This button is available when you select an entry in the list.</li> <li>• This button is not available to Auditor-type managers, and it is not enabled for you if the selected job was created by a different manager.</li> </ul>



Table 14 Log List column headers

Column header	Information displayed
Date	The date that the action started
Time	The time that the action started
Host	The Host that received the action
Action	Install or Uninstall
Component	The complete name of the package that was installed or uninstalled
Job Name	The name of the job for which the action was carried out
Manager	The user that created the job

## Confirming actions


In the Current Jobs window, you may see a job marked with the Questions  icon. This indicates that the job is paused until you give confirmation for specific tasks.

If you create a resolve policy, you can predefine which actions are acceptable and which are not. Any action for which you do not create a resolve policy will be paused until you confirm it.

## Answering questions

In this task, you will answer the questions that Agents send the Console about actions needed to complete a job.

### To answer questions:

1. To find if a job has been paused for your confirmation, look at the Main Console.  
If the Answer Questions  button is enabled, at least one job has been paused for confirmation.
2. Do one of the following:
  - In the Main Console, click **Answer Questions**.
  - In the Current Jobs window, select a job that has questions and then click **Answer Questions**.The Answer Questions window opens; see Figure 23 on page 42.

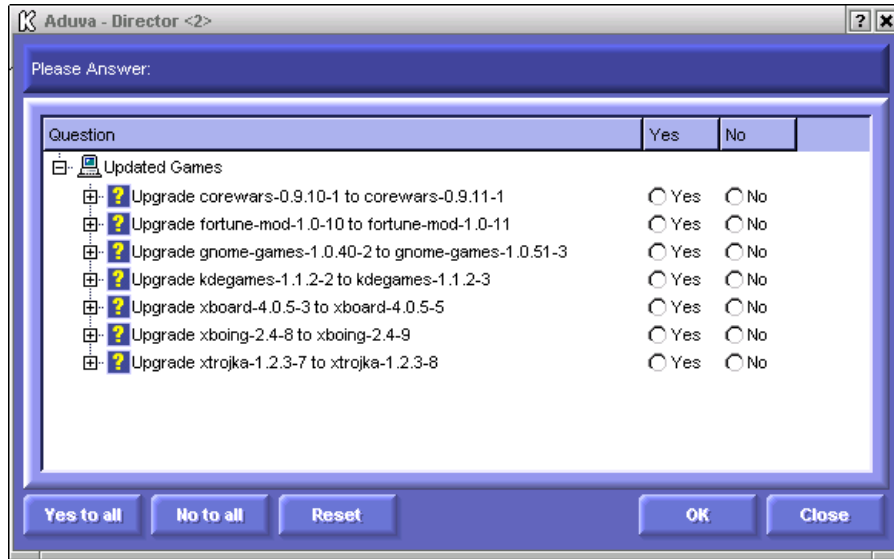


Figure 23 Answer Questions window

3. Answer the questions.

Yes	Confirm the action
No	Do not allow this action on this Host

If you answer No to a question, the current resolve is rejected. The Agent runs the search again for a new resolve, on that does not include the unpermitted actions. If such a resolve is not found, the Agent fails the job.

## Advanced features

The goal of Aduva Director is to bring Linux management tasks, with simultaneous control over multiple images, to the reach of more users. Almost all of the functions could be described as “advanced”, but Aduva Director makes them easy to use. The following topics describe more challenging situations and how to deal with them.

## Configuration files

The File Browser window provides easy access to files and logs found on registered Hosts, in the `/usr/local/Aduva Director_product_directory`, the `/etc`, and the `/var/log` directories.

This window is available if you log in as a System Administrator-type manager without permissions over the `files` component.

From the File Browser window, you can:


- ▶ View files
- ▶ Download a copy to the local Console machine
- ▶ Edit files and save the changes on the local Console machine
- ▶ Edit files and save the changes on the Host, overwriting the original files

## Downloading files

In this task, you will select files from a remote Host and download them in an archive.

Make sure that you have permissions to access the File Browser window. If you do not see the File Browser button in the Main Console, you cannot access this window.

### To download files:

1. In the Main Console, click **File Browser**. 

The File Browser window opens; see Figure 24.

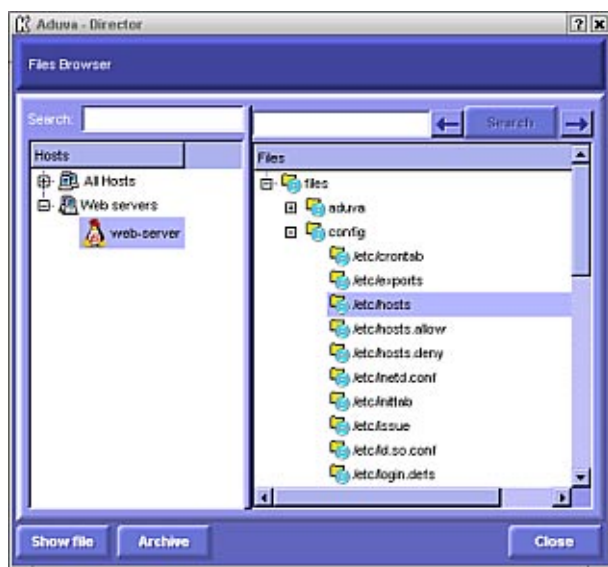


Figure 24 Files Browser window

2. Select a Host.

You can view files from one Host at a time.

3. Select the files that you want to download.

The Archive button is enabled when you select files of one Host.

4. Click **Archive**.

The Download Archive window opens; see Figure 25.



Figure 25 Download Archive window

5. Enter a local destination path and filename and a description of the file.

6. Click **Download**.

While the files are being copied from the remote image to the local machine, the word Downloading flashes in the window. Then a message appears:

Archive **your\_destination\_file.inf** for host **selected\_host** complete.

The files you selected from the remote Host were copied and archived, and the archive is now stored on the local machine.

Go to the directory you chose as the destination. Two new files are there: \*.inf, which is a text file containing information about the download; and \*.tgz, which is a compressed file containing the downloads themselves.

## Changing files

In this task, you will select files from a remote Host, view their contents, edit them, and save them.

Make sure that you have permissions to access the File Browser window. If you do not see the File Browser button in the Main Console, you cannot access this window.

### To view, edit, and save files:

1. Follow steps 1, 2, and 3 from the previous procedure.

The Show File button is enabled.

2. Click **Show File**.

The File Editor window opens; see Figure 26. Wait until the contents of the file appear in the editor.



Figure 26 File Editor window

3. Edit the file as you wish.
4. Click one of the buttons to save your changes:

Save As	Save the file on the local machine
Commit	Upload the changed file back to the Host, overwrite the original file

## Private Component Editor details

The Private Component Editor expands the power of the Aduva Director to include your own files and scripts. Using the PCE Web application, you can upload proprietary components to local Aduva Servers and thus allow Aduva Director users to easily distribute these private components to multiple Hosts.

### PCE component tree

When the Aduva Servers are expanded to include PCE components, a *PCE* category is added to the component tree in the Aduva Director Console.

The basic component tree of the Aduva Director is a multi-tier hierarchy (Figure 27 on page 46 shows a sample component tree):

- ▶ A node, or a category, is analogous to a Linux directory. Nodes can be as deeply recursive as you wish.
- ▶ An element is the final directory. Elements hold either exactly one leaf for each Linux distribution (Macros, Actions, Probes), or multiple leaves for different versions and for different distributions of each version (Files, Aduva packages).
- ▶ A leaf points to one blob, except for the Unrecognized leaf of a File, which is empty.
- ▶ A blob is the downloadable component; the file, script, binary, or package.

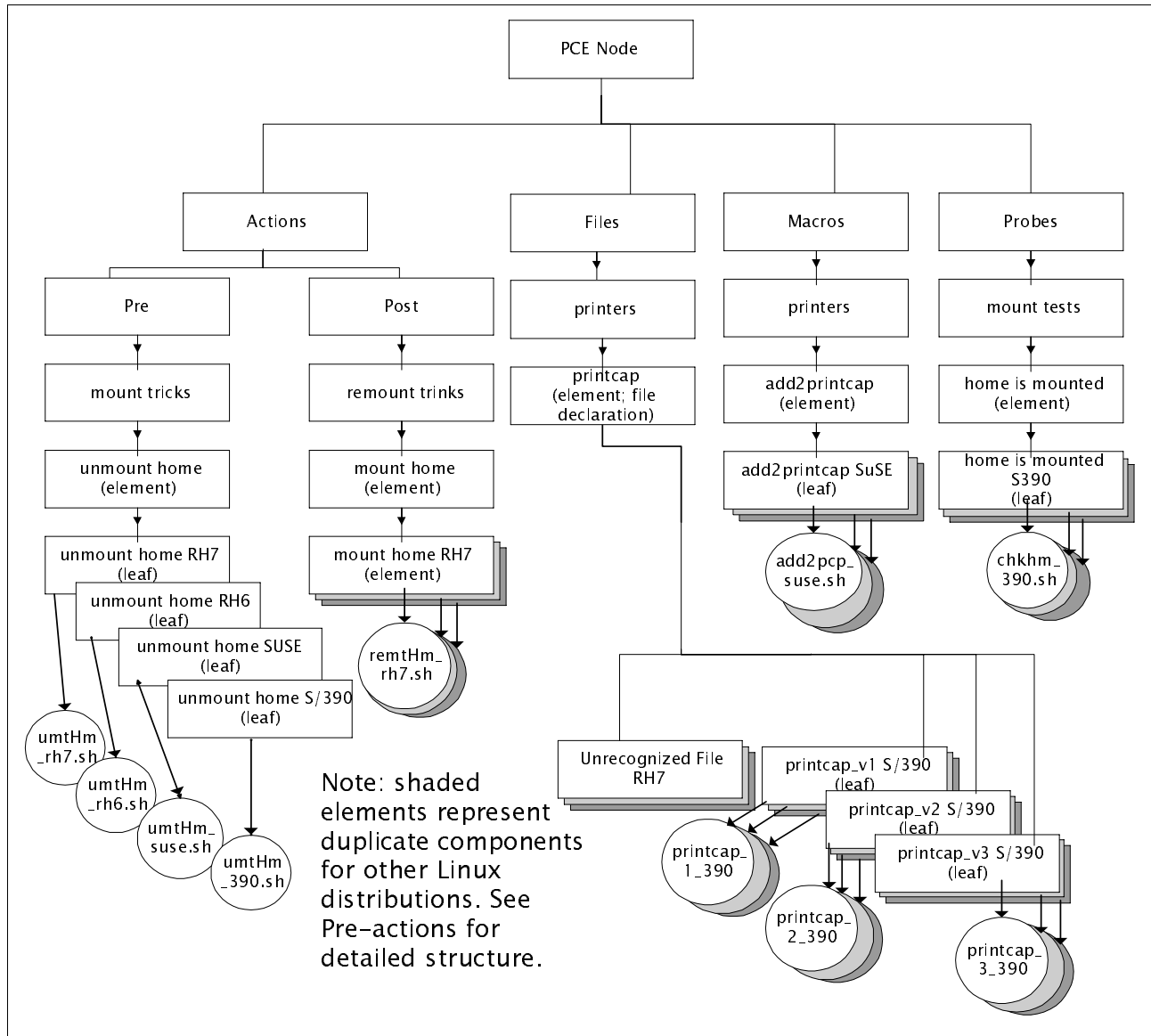


Figure 27 Sample PCE component tree

Under *PCE*, there are *Actions*, *Files*, *Macros*, and *Probes* (displayed in alphabetical order).

Under *Actions* there are two categories: *Post* and *Pre*. Under each, there may be custom nodes and elements containing the Action scripts (or binary executable). Custom nodes may go as deep as you like.

*Probes* and *Macros* may also contain custom nodes, elements, and blobs (the scripts or binary executables); or just the elements and their scripts directly under the top nodes. Each Action, Probe, and Macro element is created automatically when you add a script to the PCE repository and may hold only one script for each supported distribution.

*Files* is organized slightly differently. File elements, also called File Declarations, have more functionality in Aduva Director than do the elements of Actions, Probes, and Macros.

## PCE and the Aduva Servers

PCE components are uploaded to the Aduva servers. The PCE connects to the rserver to get the Aduva updates, both new components certified by Aduva, and all PCE components that have been uploaded.

This connection to the correct rserver is set in the `/usr/aduva/pce/aduva.rc` file, the `db_server_name` parameter, whose value is the hostname or IP address of the Aduva rserver.

When you commit a PCE component, it is added to the component tree, to a directory defined in the Aduva config file: `/usr/local/aduva/pce/aduva.rc`; the default path is: `/usr/aduva/pce/blobs`. The parameter is: `server.__general.pce_blob_conents_path_by_id`.

When you commit a component, an entry for its placement in the tree is added to the tree file on the bserver, but the component itself is not added to the rserver until you specify **Send to servers**. Once the component is in the bserver, it is not dependent upon the existence of the component in its original filepath.

## PCE Server features

The following table summarizes the PCE Server features and describes how to initiate them in the PCE Web application.

Get PCE Updates	Update the component tree from the servers; get a clean database from both the Aduva components and the PCE components database. Any PCE components that you committed but did not yet upload to the rserver are lost.
Get Aduva Updates	Update the component tree with new Aduva components.
Servers List	Add or delete connections to rservers.
Send to Servers	Upload committed component trees to rservers.

## PCE components

The PCE category of the component tree contains scripts and files, characterized by their functionality both in the Aduva Director system and as standalone resources.

The options for such characterization are:

- ▶ Actions
- ▶ Probes
- ▶ Files
- ▶ Macros

### Actions

An Action is a shell script or binary that does something to a Host when you send it as an Aduva Director job.

## **Use**

Actions are used to:

- ▶ Do tasks necessary for the success of a job and maintenance of an image
- ▶ Save time and resources that would otherwise be wasted on a job that could not succeed, because a required prerequisite cannot be done
- ▶ Pinpoint troubleshooting issues

## **Design**

Actions are one of two types: pre- or post-.

Add Actions that you will run before a job begins to the Pre-Actions node. The job is run only if the pre-Action returns a success value of zero (0).

Actions that you will run after a job is completed go in the Post-Actions node. If a post-Action fails, the completed job is marked as failed.

Make sure that your Actions exit with zero on success and non-zero on failure, to ensure that Aduva Director acts appropriately.

## **Example**

A pre-Action is Stop YpBind. A post-Action is Start YpBind. So, the pre-Action is done (stops YpBind), then the Aduva Director job is executed (some installation that conflicts with a running YpBind), and then the post-Action is done (restarts YpBind).

## **Probes**

A Probe is a shell script or binary executable that seeks the Boolean value of a specific parameter. This parameter could describe any part of an image's current status or functionality: disk space, kernel version, installation of a package, running of a service, and so on. Probes do not change Hosts.

## **Use**

Probes are run automatically on all Aduva Director Agents (all images connected to the Aduva Director system) whenever they rise, start a job, or finish a job.

The return value of a probe enables the Aduva Director to update the Inventory window with current information on the status of the Hosts.

Probes are also job prerequisites. Only if the Probe returns a true value, signalling that the Host has the required status to successfully complete the job, is the job begun.

Aduva Director users can set Probes as either "Required" or "Not Allowed." If a Probe is Required, it is true if conditions of its script or binary are true. If a Probe is Not Allowed, it is true if the conditions are false.

For example, say you wanted to install a package that could not be installed while YpBind is running. You create a Probe script called *YpBind\_is\_On*. To make the job prerequisite true when YpBind is off (and enable the job to begin), you must say that *Ypbind\_is\_On* is Not Allowed.

## **Design**

While creating probes, keep the following points in mind.

- ▶ Make the name of the Probe a Boolean statement, to make its value and use in Aduva Director as clear as possible. For example, after running the following Probes, the Current Jobs window may display the status shown:



Probe, as a Job Prerequisite		Explanation
✓	Required: Enough_Disk_Space	There is more space on the image than the minimum threshold set in the Probe.
✓	Not Allowed: YpBind_is_On	YpBind is off, and so the job may begin.
✗	Required: home_is_mounted	/home is not mounted. This probe found that the prerequisite was not fulfilled. The job will not begin.
✓	Required: private_rpm_installed	This probe searched for and found a proprietary package of your organization, whose installation you set as a prerequisite for a job.

- ▶ All Probes are run on all Agents every time they rise, begin a job, or end a job. To ensure that Aduva Director's operations are efficient, make sure that your Probes are as short and fast as possible.
- ▶ You can use the STDOUT shell command to output information in addition to the return value. This extra data is printed to Aduva Director's Component Info window (see the Aduva Director User Guide).

Note that the Component Info window can display only 124 characters. Aduva Director marks a Probe as installed only if its true, so the Component Info window is available only if the Probe is true.

For example, in the Probe script that finds if there is more disk space than a minimum, you could add a command that prints the actual used disk space to STDOUT. When this Probe is true for a specific Host, you can open the Component Info window of this Host, and see its disk space displayed.

## Files

A File is a log file, a configuration file, or any text file. Using PCE Files with Aduva Director, you can simultaneously and consistently configure the backend of multiple remote Hosts.

### Design

There are two levels of Files, and this causes exceptions to the structural rules of the other components in the PCE component tree. The following table explains the difference between the functionality of elements and leaves as they are used for Files, and as they are used for the other PCE components.

	File	Action, Macro, Probe
<b>Element</b>	You create the File Declaration. An empty leaf, called <i>Unrecognized</i> , is created automatically. Other than this leaf, this element can be standalone; it does not have to contain a File Leaf. The name can be anything you want. The definition is the full file path.	Created automatically when you add a leaf; there can be no element without a leaf. The name of the element is the same as the name of the leaf.

<b>Leaf</b>	Add a leaf for every specific version of this File that you want to be recognized. For each file version you can add more leaves for multiple distributions.	Only one leaf for each supported distribution.
-------------	---	--

### **Use**

When you open the File Browser window in the Aduva Director Console, a new category is added to this specific component tree: the *PCE Files*. Therefore, you are given all of the features of the File Browser to operate on the PCE Files: remote viewing, editing, and overwriting of backend files.

In both the File Browser window and the Inventory window, you can use File Declarations, even without adding actual files, to see if an Aduva Director-recognized file is installed.

If there is a file in the path on the remote Host that matches the path that is the File Declaration's definition, the Unrecognized File leaf is marked as installed on the Aduva Director GUI.

If you add real Files to the File Declaration, Aduva Director compares them to the file in the File Declaration path and tells you which version of the file is installed on each Host.

### **Example**

You change the printcap configuration file. You create a File Declaration called `printcap_versions`, with the path of `/etc/printcap`. You upload the original file as `printcap_v1` and the new file as `printcap_v2`. The File Declaration also contains the Unrecognized empty File leaf. Aduva Director will show you, for each Host Inventory, which of the three versions is installed.

## **Macros**

The Macro explanation is left last, because Macros are used only with PCE Files. A *Macro* is a short shell script that prints out a single line to STDOUT.

### **Use**

The output line of a Macro is used as a value to replace a variable in a PCE File. Using Macros, you can automatically and simultaneously change the parameters of a File to reflect the local values of each image that receives the File.

The variable that is replaced by the Macro value is a sign: `<^AM^>macro.sh<^AM^>`, where `macro.sh` is the name of the Macro. (AM stands for Aduva Macro.) You type this variable, this Macro signal, in a File. The Aduva Director user sends the file to Agents to download. Each Agent sees the AM sign and downloads the named Macro script. When the Macro is run on the Agent, its value, the line it outputs, is entered in place of the AM sign.

### **Design**

Macros are downloaded and run whenever Aduva Director has to read a PCE File with a Macro sign. So, when creating your Macros, remember to make them as fast as possible: the Macro should read from the local Host whenever possible.

### **Example**

You create a shell script called `hostname.sh` that contains the command **hostname**, to return the local hostname. In a PCE File that you want to install on a thousand Hosts, you replace the original hostname with `<^AM^>hostname.sh<^AM^>`.

When an Aduva Director Agent downloads the File, it downloads and runs the Macro called `hostname.sh`. It replaces the sign with the return value line; in this case, with the local `hostname`.

The Aduva Director user can see, from the Inventory window, that all of the Hosts have the file installed. Using the File Browser, the Aduva Director user can see that the only difference between the files on all of the Agents is the `hostname`.

## Starting the PCE

The PCE is a Web application, so to start, open a Web browser and enter the location of the PCE in place of a URL. The Login window appears. The default PCE user is `aduva_root`, with the password 123. Use this ID to create a private user ID.

### To change a PCE user ID:

1. In the Login window, click **User Administration**.  
The PCE User Info window appears.
2. Enter the user ID and password.
3. Enter a new password or ID and then click **New Password** or **New PCE ID**. You are returned to the Login window.

### To create or delete a user ID:

1. In the Admin PCE ID and Password boxes, enter `aduva_root` and the password.  
This is the only user ID that can create or delete a new user.
2. Enter the new PCE ID and Password and then click **Add PCE ID**.  
Or, enter the user ID that you want to delete and then click **Delete PCE ID**.  
You are returned to the Login window.

To log in, enter a valid ID and password. The Home page appears.

## Understanding the PCE frames

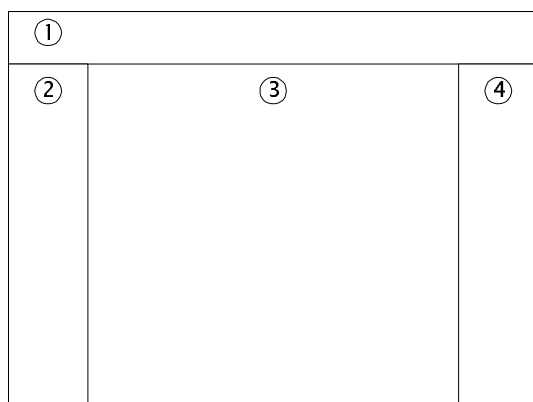


Figure 28 PCE Home Page frames

The Home is divided into the following frames:

1. Path - The top frame dynamically displays where you are in the PCE component tree. To navigate along the path, click a category name.
2. Components - The left frame shows the components contained in the current Path.

3. PCE Features - The right frame is a menu of options: Logout, Help, and Server Configuration options (see “PCE Server features” on page 47).
4. Workspace - The middle frame is the workspace. It displays text-entry forms, the Commit button when appropriate, relevant options, and messages, according to the current Path, selected component, and performed action.

## Replacing Aduva packages

The Components frame at the root of the tree shows more than the PCE category. The other categories are those supplied by default by Aduva. They contain the components that Aduva tests, certifies, and makes available over the public servers.

When you select an Aduva element that holds packages, the workspace displays a top frame. This frame shows that you can replace one of the displayed packages with a package on the PCE machine.

Your private component is stored in a new directory in the Aduva Servers, so it is not overwritten when the Aduva Servers are updated from a parent server or the public Aduva Servers.

### To replace an Aduva package:

1. In the Components frame, click the select button of the package that you want to replace.
2. In the workspace top frame, click **Replace Software Blob**.

The workspace displays the form needed to replace the public package with a private one.

3. Browse to the package and then click **Commit**.

Your package is copied to the server directory.

While Aduva Director is linked to the PCE, the private version of the package will be used instead of the one supplied by Aduva.

## Working with components - in general

A *component* is any item in the component tree: a category (also called a node), an element, a leaf, or a blob. You can add as many recursive categories as you want for any of the PCE component types, and you can edit any component after adding it to the PCE.

### To add a category:

1. In the top frame of the workspace, click **Category**.

The workspace displays the form needed to create a new category.

2. Enter a name and a description for the category.
3. Click **Commit**.

The new custom category is added to the PCE component tree. The workspace notifies you of the success, or a reason for failure.

The top frame of the workspace adds an Edit option.

### To edit a component:

1. In the Components frame, check the component you want to edit.
2. In the top frame of the workspace, click **Edit**.

The workspace displays the Data Editor.

3. Change the available aspects of the component (usually names and descriptions).
4. Click **Commit**.

The component tree is updated.

## Working with Files

In the Components frame, click **Files**. The top frame of the workspace shows the PCE Files options:

- ▶ Add a category - a custom Files category can hold more custom categories (nodes) or File Declarations (elements). Make the tree as deep as you wish.
- ▶ Add a File Declaration - a File Declaration is a PCE element that has a full file path and can hold multiple versions of the same file. Files cannot be held in a node, only in a File Declaration.

### To add a File Declaration:

1. In the Components frame, click the category you want to hold the new File Declaration.
2. In the top frame of the workspace, click **File Declaration**.

The workspace displays the form needed to add a File Declaration. Before you enter any information, make sure that the Parent name is that of the category you want to hold this File Declaration.

3. Enter a name, the full filepath, and a description.
4. Click **Commit**.

A default Unrecognized File is added to the File Declaration. The component tree is updated. The top frame of the workspace adds an Edit option.

5. If you have files to add to the File Declaration, continue with the next procedure.

### To add a File:

1. In the Components frame, click **File Declaration**.
1. The top frame of the workspace displays the option to add a File.
2. In the top frame of the workspace, click **File**.
3. The workspace displays the form needed to add a File.
4. Enter a name.
5. Select the file, from the local file system, to upload to the bserver.
6. Enter a description.
7. Select the appropriate Linux distribution (Red Hat 6 or 7, SuSE or SuSE S/390).
8. Click **Commit**.

The component tree is updated. The PCE indicates that a component is a File by adding an icon, which also indicates the File Distribution, to the component's entry in the Components frame.

The top frame of the workspace adds Replace and Edit options. You can edit the name, or replace the File with another one.

## Working with Actions

In the Components frame, under PCE, click **Actions**. The top frame of the workspace shows the PCE options:

- ▶ Add a category - a custom category can hold more custom categories (nodes) or elements. Make the tree as deep as you wish.
- ▶ Add a leaf - add an Action script or binary. An element of the same name is created to hold the leaf.

### To add an Action leaf:

1. In the Components frame, under PCE, click **Actions**, and then either **Pre** or **Post**.  
Pre-Actions are done before a job begins; post-Actions are done after a job is completed.  
The top frame of the workspace displays the option to add an Action:  
Add to Pre (or Post): Category Action\_leaf
2. Click **Action\_leaf**.  
The workspace displays the form needed to add a leaf.
3. Enter a name, which will be the name of both the element and the leaf, although the Action itself may have a different name.
4. Browse to the Action file from the local file system to enter its full filepath and filename.
5. Enter a description.
6. Select the appropriate Linux distribution.
7. Click **Commit**.

The component tree is updated. The top frame of the workspace adds an Edit option.

## Working with Macros

In the Components frame, under PCE, click **Macros**. The top frame of the workspace shows the PCE options:

- ▶ Add a category - a custom category can hold more custom categories (nodes) or elements. Make the tree as deep as you wish.
- ▶ Add a leaf - add a Macro script. An element of the same name is created to hold the leaf.

### To add a Macro leaf:

1. From the Path **Root -> PCE -> Macros**, click **Macro\_leaf**:  
**Add to Macros: Category Macro\_leaf**  
The workspace displays the form needed to add a leaf.
2. Enter a name, which will be the name of both the element and the leaf, although the Macro script itself may have a different name.
3. Browse to the Macro file from the local file system to enter its full filepath and filename.
4. Enter a description.
5. Select the appropriate Linux distribution.
6. Click **Commit**.

The component tree is updated. The top frame of the workspace adds an Edit option.

## Working with Probes

In the Components frame, under PCE, click **Probes**. The top frame of the workspace shows the PCE options:

- ▶ Add a category - a custom category can hold more custom categories (nodes) or elements. Make the tree as deep as you wish.
- ▶ Add a leaf - add a Probe script. An element of the same name is created.

### To add a Probe leaf:

1. From the Path **Root -> PCE -> Probes**, click **Probe\_leaf**:

The workspace displays the form needed to add a leaf.

2. Enter a name, which will be the name of both the element and the leaf, although the Probe script itself may have a different name.
3. Browse to the Probe file from the local file system to enter its full filepath and filename.
4. Enter a description.
5. Select the appropriate Linux distribution.
6. Click **Commit**.

The component tree is updated. The top frame of the workspace adds an Edit option.





# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

## **COPYRIGHT LICENSE:**

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

This document created or updated on October 4, 2002.



Send us your comments in one of the following ways:

- ▶ Use the online **Contact us** review redbook form found at:  
[ibm.com/redbooks](http://ibm.com/redbooks)
- ▶ Send your comments in an Internet note to:  
[redbook@us.ibm.com](mailto:redbook@us.ibm.com)
- ▶ Mail your comments to:  
IBM Corporation, International Technical Support Organization  
Dept. HYJ Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400 U.S.A.

## Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®


IBM®

IBM eServer™

S/390®

z/VM™

zSeries™

Redbooks(logo)™ 

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.