Novell ZENworks® Orchestrator

1.2

www.novell.com

ADMINISTRATION GUIDE

January 31, 2008

Novell®

Legal Notices

Novell, Inc. makes no representations or warranties with respect to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

Further, Novell, Inc. makes no representations or warranties with respect to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to make changes to any and all parts of Novell software, at any time, without any obligation to notify any person or entity of such changes.

Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classification to export, re-export or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in the U.S. export laws. You agree to not use deliverables for prohibited nuclear, missile, or chemical biological weaponry end uses. See the Novell International Trade Services Web page (http://www.novell.com/info/exports/) for more information on exporting Novell software. Novell assumes no responsibility for your failure to obtain any necessary export approvals.

Novell, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed on the Novell Legal Patents Web page (http://www.novell.com/company/legal/patents/) and one or more additional patents or pending patent applications in the U.S. and in other countries.

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

Online Documentation: To access the latest online documentation for this and other Novell products, see the Novell Documentation Web page (http://www.novell.com/documentation).

Novell Trademarks

For Novell trademarks, see the Novell Trademark and Service Mark list (http://www.novell.com/company/legal/trademarks/tmlist.html).

Third-Party Materials

All third-party trademarks are the property of their respective owners.

Contents

About This Guide

This *Administration Guide* introduces Novell® ZENworks® Orchestrator 1.2, including its basic administration environment, Which is accessed through the ZENworks Orchestrator Console and various command line tools. The guide provides an introductory overview of the Orchestrator and explains how it administers and manages work on the resources of the data center. The guide is organized as follows:

- Chapter 1, “Understanding the Orchestrator Job Scheduler,” on page 9
- Appendix A, “The Zosadmin Command Line Tool,” on page 27

Audience

This book is intended for data center managers and IT or Operations administrators. It assumes that users of the product have the following background:

- General understanding of network operating environments and systems architecture.
- Knowledge of basic UNIX* shell commands and text editors.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation, or go to www.novell.com/documentation/feedback.html (http://www.novell.com/documentation/feedback.html) and enter your comments there.

Documentation Updates

For the most recent version of this *Administration Guide*, visit the ZENworks Orchestrator 1.0 Web site (http://www.novell.com/documentation/zos1).

Documentation Conventions

In Novell documentation, a greater-than symbol (>) is used to separate actions within a step and items in a cross-reference path.

A trademark symbol (®, ™, etc.) denotes a Novell trademark. An asterisk (*) denotes a third-party trademark.

When a single pathname can be written with a backslash for some platforms or a forward slash for other platforms, the pathname is presented with a backslash. Users of platforms that require a forward slash, such as Linux* or UNIX, should use forward slashes as required by your software.

Understanding the Orchestrator Job Scheduler

1

The Job Scheduler in Novell® ZENworks® Orchestrator Server can be used to automatically start deployed jobs on your grid by using either time or event triggers.

You can think of the functionality provided by the time triggers as being similar to a distributed cron system (in fact, time triggers can be described in cron syntax). This triggering, coupled with the job control functions in ZENworks Orchestrator, allows for the sophisticated automation of routine data center tasks.

For example, suppose you want to periodically harvest a large log file in a coordinated way from a farm of several hundred machines. First, you could create an Orchestrator job that uses the datagrid for file movement. The job control options specify that the job should run on not more than three machines at once and sweep across the entire grid. You would then create a schedule to run this job at the desired interval.

As another example, you could use the Job Scheduler to trigger a discovery job every time a new resource is added to the grid. In this case, the job developer writes the discovery job to discover and set facts about the resource. Next, you would create a schedule to run this job on the Run On Resource Start event. In fact, this type of triggered job is currently used in the standard set of deployed discovery jobs to detect specific resource CPU and OS information.

Another example would be to run a job on server startup that sends a notification e-mail to an administrator.

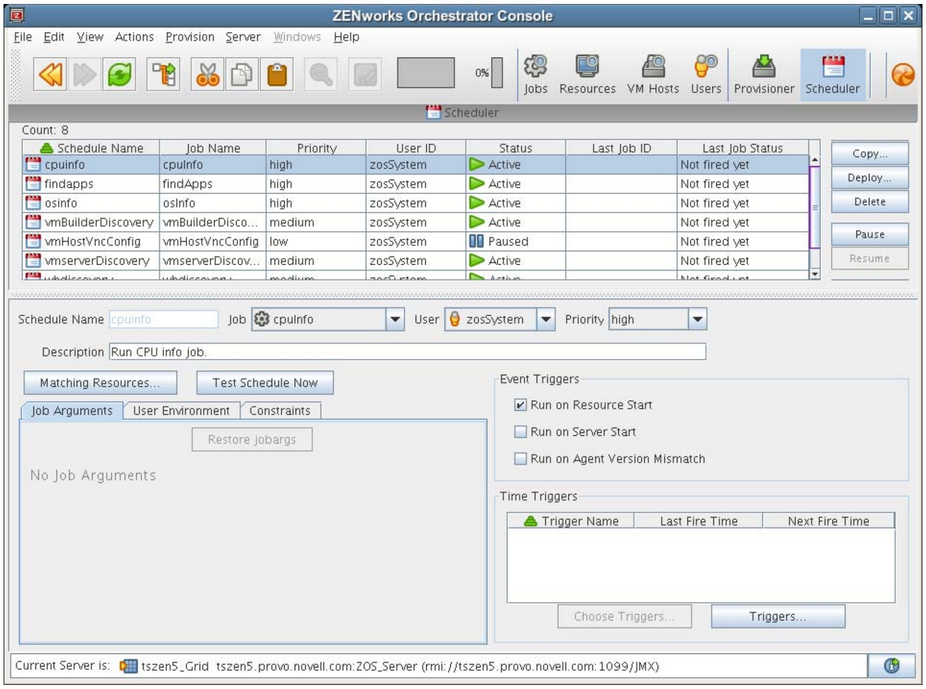
This section of the *ZENworks Orchestrator Administration Guide* includes the following information:

- Section 1.1, "The Job Scheduler View," on page 9
- Section 1.2, "Walkthrough: Scheduling a System Job," on page 18

1.1 The Job Scheduler View

Click *Scheduler* on the main toolbar of the ZENworks Orchestrator Console to open the Job Scheduler view.

Figure 1-1 *Job Scheduler View of the ZENworks Orchestrator Console*



Count: 8

Schedule Name	Job Name	Priority	User ID	Status	Last Job ID	Last Job Status
cpuinfo	cpuInfo	high	zosSystem	Active		Not fired yet
findapps	findApps	high	zosSystem	Active		Not fired yet
osinfo	osInfo	high	zosSystem	Active		Not fired yet
vmBuilderDiscovery	vmBuilderDisco...	medium	zosSystem	Active		Not fired yet
vmHostVncConfig	vmHostVncConfig	low	zosSystem	Paused		Not fired yet
vmserverDiscovery	vmserverDiscov...	medium	zosSystem	Active		Not fired yet

Schedule Name cpuinfo Job cpuInfo User zosSystem Priority high

Description Run CPU info job.

Event Triggers

☑ Run on Resource Start

□ Run on Server Start

□ Run on Agent Version Mismatch

Time Triggers

Trigger Name	Last Fire Time	Next Fire Time

Current Server is: tszen5_Grid tszen5.provo.novell.com:ZOS_Server (rmi://tszen5.provo.novell.com:1099/JMX)

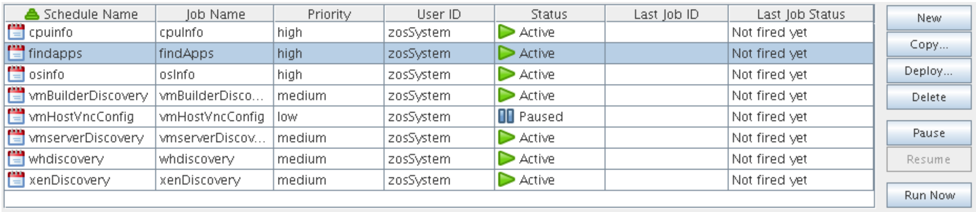
This section includes information to help you understand the functions of the Job Scheduler and how to use it to launch Orchestrator jobs.

- Section 1.1.1, “Navigating The Job Schedules Table,” on page 10
- Section 1.1.2, “Creating or Modifying a Job Schedule,” on page 12

1.1.1 Navigating The Job Schedules Table

ZENworks Orchestrator includes several predefined and predeployed discovery jobs that have predefined launch schedules. Among these jobs are the `cpuinfo`, `findapps`, `osinfo`, and other jobs, depending on the options (that is, the “server profile”) you chose and the configuration you used during the installation. After installation, these jobs are listed by name in a table in the Job Scheduler view.

Figure 1-2 *The Job Schedules Table in the Job Scheduler View*



Schedule Name	Job Name	Priority	User ID	Status	Last Job ID	Last Job Status
cpuinfo	cpuInfo	high	zosSystem	Active		Not fired yet
findapps	findApps	high	zosSystem	Active		Not fired yet
osinfo	osInfo	high	zosSystem	Active		Not fired yet
vmBuilderDiscovery	vmBuilderDisco...	medium	zosSystem	Active		Not fired yet
vmHostVncConfig	vmHostVncConfig	low	zosSystem	Paused		Not fired yet
vmserverDiscovery	vmserverDiscov...	medium	zosSystem	Active		Not fired yet
whdiscovery	whdiscovery	medium	zosSystem	Active		Not fired yet
xenDiscovery	xenDiscovery	medium	zosSystem	Active		Not fired yet

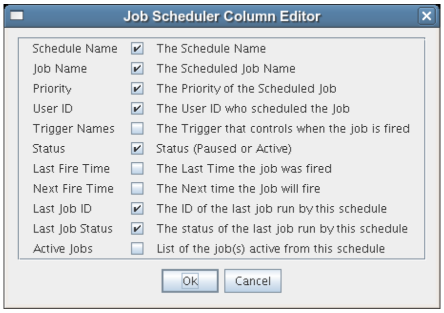
By default, ZENworks Orchestrator uses schedule names that are similar to the job name so that schedules are easy to match (although this is not required). The schedules list shows all of the existing job schedules that accompany predefined jobs, along with the schedules that you create in the Job Scheduler.

NOTE: The Job Scheduler view is not a real-time monitor view of jobs, so if a job attribute (for example, Last Job Status or Last Fire Time) has changed, it might not be displayed until you click *Refresh*.

The Job Schedules Table has functionality that lets you decide how you want to display information about the job schedules:

- You can drag any column in the table to move it left or right in the table according to your preference.
- You can mouse over any column heading in the table to view tool tip text about the purpose of the data in that column.
- You can right-click any column heading in the table to open the Job Scheduler Column Editor dialog box.

Figure 1-3 *Job Scheduler Column Editor Dialog Box*



Schedule Name	☑	The Schedule Name
Job Name	☑	The Scheduled Job Name
Priority	☑	The Priority of the Scheduled Job
User ID	☑	The User ID who scheduled the Job
Trigger Names	□	The Trigger that controls when the job is fired
Status	☑	Status (Paused or Active)
Last Fire Time	□	The Last Time the job was fired
Next Fire Time	□	The Next time the Job will fire
Last Job ID	☑	The ID of the last job run by this schedule
Last Job Status	☑	The status of the last job run by this schedule
Active Jobs	□	List of the job(s) active from this schedule

You can select any column heading in this dialog box to display it in the Job Schedules Table. The columns display the attributes of a previously configured job schedule. As the figure shows, this dialog box also includes text that clarifies the purpose of the data in each column.

In the Job Scheduler view, there are seven function buttons next to the Job Schedules Table that let you take action on any schedule you select inside the table. (Only one schedule at a time can be selected.)

- **New:** Opens a dialog box where you can create a new schedule. When you create a new schedule, the Job Scheduler adds a new line to the Job Schedules Table. When the new line is added, you can use the Job Schedule Editor to edit the attributes of the schedule. A new schedule must be given a unique schedule name.

  The Job Scheduler forces a new schedule to be created in the *Paused* state to prevent it from running while it is being defined. You click *Resume* when a job is ready to be used.
- **Copy:** Copies a schedule you have selected in the Job Schedules Table. Clicking this button opens a dialog box where you rename the copy. If you want to create a schedule similar but not identical to an existing schedule, use this button to save time in adding attributes to a job schedule configuration. A copy of a schedule must be given a unique schedule name.
- **Deploy:** Opens a dialog box where you can select a schedule (that is, a deployable `.sched` file) to deploy.
- **Delete:** Deletes the selected schedule from the Job Schedules Table. You cannot recover a deleted job schedule.

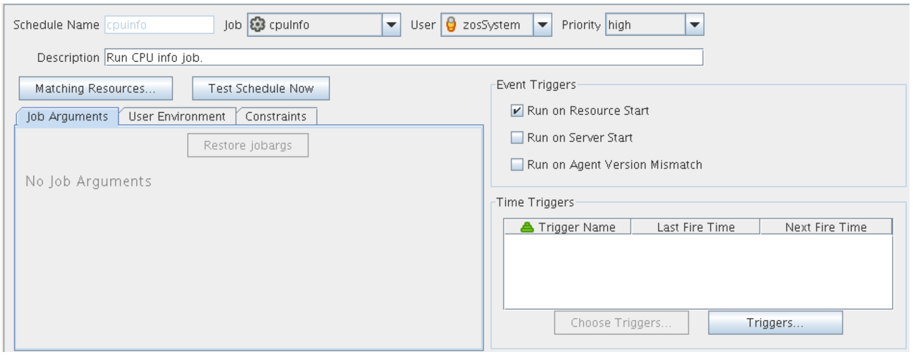
NOTE: Deleting a schedule that was deployed as part of a `.job` or `.sar` displays a confirmation dialog box. Deleting the schedule undeploys all contents of the `.job` or `.sar` that contains the schedule.

- **Pause:** Pauses (disables) the selected schedule in the Job Schedules Table. The jobs associated with the schedule are not re-run, but any currently running instances of this job continue to run.
- **Resume:** Resumes (enables) a paused job schedule.
- **Run Now:** Forces the specified schedule to run immediately. This updates statistics such as *Last Fire Time*.

1.1.2 Creating or Modifying a Job Schedule

The Job Schedule Editor is located immediately below the Job Schedules Table in the Job Scheduler view.

Figure 1-4 *The Job Schedule Editor in the Job Scheduler View*



Schedule Name cpuinfo
Job cpuInfo
User zosSystem
Priority high
Description Run CPU info job.

Event Triggers
☑ Run on Resource Start
☐ Run on Server Start
☐ Run on Agent Version Mismatch

No Job Arguments

There are several times when you can use this part of the Job Scheduler tool:

- When you create a new schedule by clicking *New*
- When you modify the attributes of an existing schedule (available when you select a schedule in the table)
- When you create a copy of an existing schedule by clicking *Copy*

The Job Schedule Editor lets you create or modify a job schedule by specifying its attributes.

You can use the following controls and data when you create or modify a job schedule:

- "Schedule Name" on page 13
- "Job" on page 13
- "User" on page 13
- "Priority" on page 13
- "Description" on page 13
- "Matching Resources" on page 13
- "Test Schedule Now" on page 14
- "Job Arguments" on page 14
- "User Environment" on page 14

- “Constraints” on page 15
- “Event Triggers” on page 15
- “Time Triggers” on page 16
- “Choose Triggers” on page 17

Schedule Name

When you create a new schedule, the unique name you specify is displayed in this field. If you select a schedule from the Job Schedules Table, the name of the schedule is displayed in this field. The field is not editable, because schedules cannot be renamed after they are created. (You can use a copy if this is required.)

Job

When you create a new schedule, you need to associate a deployed job with it. You can select the job you want to run from this drop-down list.

If you want to use a previously created schedule to run a different job, you can change the job here.

User

When you create a new schedule, you need to associate a user with it. The user represents the user for whom the job will run. The choice of user might affect the permissions, privileges and constraints of the job. You can select the user from this drop-down list.

If you want a different user to run a job on a previously created schedule, you can change the user here.

If you decide to change the user who runs the job, check the *Priority* field to make sure that the priority you want is selected.

Priority

When you create a new schedule and associate a job and a user with it, a list of possible run priorities becomes available in this drop-down list. The list of priorities varies, depending on the user that is specified in the previous field. In this field, you select the priority of the job that is to be run so that if other jobs are to start concurrently or are competing for resources, the Orchestrator can determine which job takes priority.

Description

For predeployed jobs, this field contains a default description of what the job’s schedule does. The field is editable, so you can enter a description of your own for job schedules that you create.

Matching Resources

This button displays a list of resources where the job runs now or where it could run. This list is useful for checking the context of constraints that might have been affected by a choice of user or by manually specifying additional constraints under the *Policy* tab. The list is also useful to verify that a discovery job (that is, one that is triggered by the *Run on Resource Start* option) runs on the preferred set of machines.

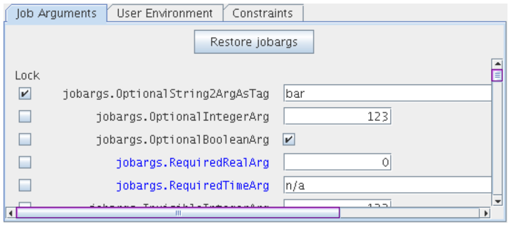
Test Schedule Now

Click this button to test the new or modified schedule you are working with. The test runs the new or modified schedule without permanently saving the current configuration of the schedule or recording statistics. This control differs from the *Run Now* control in the Job Schedules Table, which runs a saved (persisted) schedule, disregarding any unsaved modifications that have been made to it in the Job Schedule Editor.

Job Arguments

This tab displays an area (in the lower left corner of the Job Schedule Editor) where possible job arguments are listed. If you select an existing schedule in the Job Schedules Table, any optional job arguments (jobargs) for the associated job are displayed in this area.

Figure 1-5 *The Job Arguments Area of the Job Scheduler View*



Lock

☑ jobargs.OptionalString2ArgAsTag bar

□ jobargs.OptionalIntegerArg 123

□ jobargs.OptionalBooleanArg ☑

□ jobargs.RequiredRealArg 0

□ jobargs.RequiredTimeArg n/a

The jobargs are defined by the deployed job. Some jobs might already have a default value displayed, but others must have values specified in order for the job to be able to run.

IMPORTANT: Job arguments displayed in blue are required. You must supply data in the accompanying fields.

A job argument defines the values that can be passed in when a job is invoked. These values let you statically define and control job behavior. To learn more about each job argument, mouse over each jobarg line to display tool tip text.

The Job Scheduler uses the values you enter into the fields of this area to build a jobargs namespace in the policy for this job.

Each job argument has an accompanying *Lock* check box. When *Lock* is not selected, the accompanying job argument uses the default value specified in the job's policy. When *Lock* is selected, the value specified in the field is locked down and overrides the default value in the policy. A locked value continues to be used even if the policy value is modified.

You can click *Restore Jobargs* to restore job arguments to the values specified in the job policy. This function removes any changes you might have specified in the Job Scheduler and deselects all *Lock* check boxes.

For more information, see "Job Arguments and Parameter Lists" in the *Novell ZENworks Orchestrator 1.2 Developer Guide and Reference*.

User Environment

This tab displays an area (in the lower left corner of the Job Schedule Editor) that includes the *Pass User Environment* check box. Select this check box if you want to pass the assigned user's environment variables to the job when it runs. When environment variables are recorded on the user account, selecting the *Pass User Environment* check box makes those environment variables available to the job and joblet.

A user's environment is recorded under the `user.env` fact on his or her account. This fact can be set when a user logs in to the Orchestrator and is persisted until changed. A user's environment variables can be uploaded with the zos command line tool at login time in one of two variations:

- `zos login --user=`*`foo`*` --env`

  This command uploads the entire environment to the Job Scheduler. The upload can also be seen on the User object in the Orchestrator Console.

- `zos login --user=`*`foo`*` --env=PATH`

  When the user logs in, he or she can specify one or more environment variables to use at login. The example above would result in just the PATH environment variable being uploaded.

  Multiple environment variables can be specified by delimiting with a comma, as in the following example:

  ```
  --env=PATH,LD_PATH,ID
  ```

NOTE: The user's environment variables can also be passed to the server when the user implements the zos command line tool when running a job (as opposed to logging in). The command passes the environment variable only for that particular job run.

```
zos run jobname --env=environment_variable
```

Constraints

This tab displays a constraint editor that you can use to create additional constraints for the job being scheduled. Typically, additional "resource constraints" (such as "start") are useful to delay the start of a job when it is triggered.

Event Triggers

An event trigger is the signal to the Job Scheduler to initiate, or "fire" a job when a given event occurs. An event trigger can be used in conjunction with a time trigger to allow flexibility in scheduling the job application for maximum effectiveness or convenience.

The Job Scheduler has three possible event triggers: *Run on Resource Start*, *Run on Server Start*, and *Run on Agent Version Mismatch*. The first two triggers are self-explanatory. The third trigger, *Run on Agent Version Mismatch*, triggers the job when an Orchestrator Agent of an incompatible version attempts to connect to this Orchestrator Server. It is used to initiate a configuration management tool for an agent software update. Jobs triggered by *Agent Version Mismatch* must specify a job argument (jobarg) named "`resource.id`" as in the following example:

```
<jobargs>
    <fact name="resource.id"
        type="String"
        description="The id of the mismatched resource"
        value=""
        visible="true" />
    <fact name="resource.ip"
        type="String"
        description="IP address of the mismatched resource"
        value=""
```

```
            visible="true" />
    </jobargs>
```

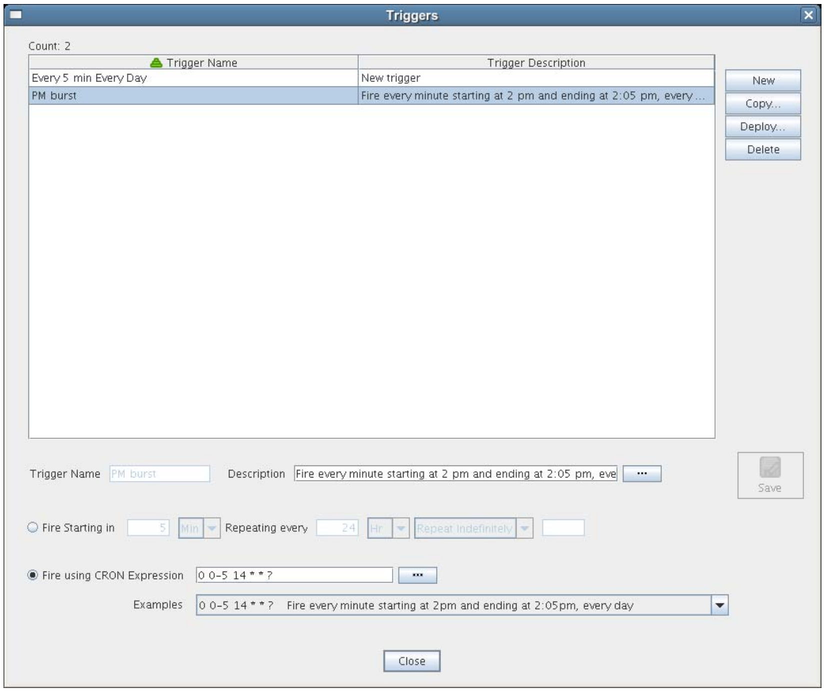
You can select any combination of these event triggers.

Time Triggers

A time trigger is the signal to the Job Scheduler to initiate, or “fire” a job at a given point (or points) in time. A time trigger can be used in conjunction with an event trigger to allow flexibility in scheduling the job application for maximum effectiveness or convenience.

When you click *Triggers* in the Job Scheduler view, the following dialog box opens:

Figure 1-6 *The Time Trigger Dialog Box*



Triggers

Count: 2

Trigger Name	Trigger Description
Every 5 min Every Day	New trigger
PM burst	Fire every minute starting at 2 pm and ending at 2:05 pm, every ...

Trigger Name: PM burst

Description: Fire every minute starting at 2 pm and ending at 2:05 pm, eve

○ Fire Starting in 5 Min Repeating every 24 Hr Repeat Indefinitely

● Fire using CRON Expression 0 0-5 14 * * ?

Examples: 0 0-5 14 * * ?   Fire every minute starting at 2pm and ending at 2:05pm, every day

In this dialog box, you can define the time triggers you want to associate with the launching of your jobs. You can create as many time triggers as you want to meet any scheduling situation you might have. Multiple time triggers can be associated with a given schedule and multiple schedules can use a given time trigger.

The time triggers you create are retained by the Job Scheduler for you to choose from when you create a schedule for a job. These are displayed in the list with the name and the description of the trigger.

The following controls and information are available in the dialog box:

- **New:** Opens a secondary dialog box where you can create a new time trigger name. When you create the trigger name, the attribute fields in the Triggers dialog box are cleared and you can specify new attributes for the trigger. A new trigger must be given a unique trigger name.
- **Copy:** Lets you modify an existing time trigger by giving it a new name and attributes. This can be helpful if there are only slight differences in the new attributes. A copy of a trigger must be given a unique trigger name.

- **Delete:** Deletes a selected time trigger.

  IMPORTANT: Deleted triggers are not recoverable. If the trigger is used by existing schedules, it is removed from all of those schedules when it is deleted.

- **Trigger Name:** Specifies the unique name of the trigger you are creating or modifying. This name is displayed in the Job Scheduler if you choose to associate this trigger with the schedule. After you create the trigger name, it cannot be modified.
- **Description:** Specifies a description for the time trigger you are creating or modifying. The description is optional and can be as detailed as you want.

  Click the … button to open a dialog box where you can edit more lengthy time trigger descriptions.
- **Save:** Saves the defined time trigger and its attributes.
- **Fire Starting In:** Displays multiple fields specifying the time increment and frequency to be used by the trigger to fire the job. If you select this type of time trigger, the *Fire using CRON Expression* button becomes inactive.

  NOTE: You can use this control to create either a “one-shot” time trigger or a “reoccurring” time trigger.

  A one-shot time trigger fires just once after a specified period of time. To specify a one-shot trigger, click *Fire Starting in*, specify the amount of time before firing, and then enter 0 as the time to *Repeat Indefinitely*.

  A reoccurring time trigger fires after a specified period and then either fires repeatedly for an indefinite number of times or it fires for a specified number of times. To specify a reoccurring, indefinite trigger, click *Fire Starting in*, specify the amount of time before firing, then select *Repeat Indefinitely*. To specify a reoccurring but finite trigger, click *Fire Starting in*, specify the amount of time before firing, select *Repeat Range*, then enter the number of times you want the trigger to fire.

- **Fire using CRON Expression:** Specifies the cron expression that enables the job to fire automatically at a specified time or date. You need to be familiar with cron to use this field.

  A list box of selected cron expressions and their associated descriptions is located just below this button. You can use a listed expression as is, or use it as a template to modify the expression to meet your needs.

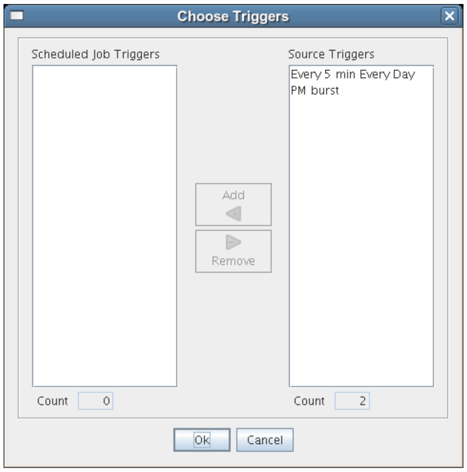
  If you select this type of time trigger, the *Fire Starting In* button becomes inactive.

  For an example of how a cron expression can be implemented in a trigger, see “Creating and Assigning a Time Trigger for the New Schedule” on page 24.

Choose Triggers

Opens a dialog box where you can choose the predefined time triggers you want to associate with this job schedule.

Figure 1-7 Choose Triggers Dialog Box



In this dialog box, you can click *Add* to move a selected trigger to the active, scheduled time triggers that are to be associated with this job schedule. You can also click *Remove* to unassociate a trigger.

When a trigger is moved to the scheduled list, it becomes associated to the job schedule and it is displayed in the Job Scheduler view.

1.2 Walkthrough: Scheduling a System Job

This section demonstrates how you can use the Orchestrator Console to deploy and schedule an existing system job named `auditcleaner.job`. This example job is included in the `examples` directory of your ZENworks Orchestrator installation.

This section includes the following information:

- Section 1.2.1, "Deploying a Sample System Job," on page 18
- Section 1.2.2, "Creating a New Schedule for the Job," on page 20
- Section 1.2.3, "Defining the New Schedule," on page 20
- Section 1.2.4, "Activating the New Schedule," on page 25
- Section 1.2.5, "Running the New Schedule Immediately," on page 26

1.2.1 Deploying a Sample System Job

Before a job can run, the ZENworks Orchestrator administrator must deploy it, which involves moving it from a developed package state to a state where it is ready and available for users. Only the administrator has the necessary rights to deploy a job.

There are four methods you can use to deploy a job:

- Deploy it from the ZENworks Orchestrator Console by right-clicking the *Jobs* container in the Explorer panel.
- Deploy it from the ZENworks Orchestrator Console by selecting the *Actions* menu in the console.
- Deploy it from the `zosadmin` command line (`zosadmin deploy` *`path_to_job`*).

- Copy the deployable component to the “hot” deployment directory under the Orchestrator server instance directory. Typically, this directory is located at `/var/opt/novell/zenworks/zos/server/deploy`. Using this method, deployment proceeds within a few seconds. ZENworks Orchestrator monitors this directory.

A runnable job can also be scheduled, which means that the schedule for running the job and the trigger or triggers that initiate or “fire” the schedule (or both) are configured and packaged with the job.

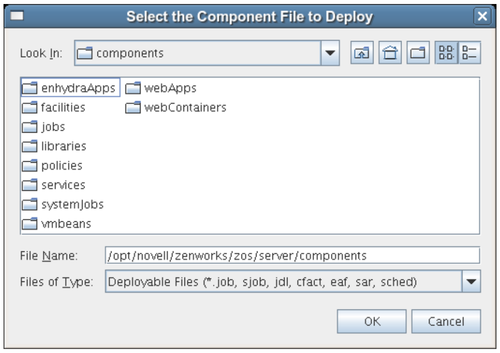
For this walkthrough, you deploy one of several system jobs (`auditCleaner.job`) developed for ZENworks Orchestrator customers to demonstrate how system jobs are deployed and run. This job package, which is actually a `.jar` archive, includes only a `.policy` component and a `.jdl` component. It does not have a `.sched` component. You can use the Job Scheduler in the Orchestrator Console to add the `.sched` component separately.

NOTE: An Orchestrator Job Developer can create and package jobs that include a `.jdl` file, a `.policy` file, and a `.sched` file (schedule). The presence of the `.sched` file in the job package is also typical of the predeployed discovery jobs installed with Orchestrator, which run without intervention when the criteria for firing the schedule are satisfied. Such jobs are visible in the Job Scheduler because they already include the `.sched` components.

For more information about developing jobs and schedules in a job archive, see “Job Scheduling” in the *Novell ZENworks Orchestrator 1.2 Developer Guide and Reference*.

Although this walkthrough demonstrates only the first method listed above for deploying, the other methods are relatively simple, so no further examples are provided to illustrate them.

1 In the Explorer panel of the ZENworks Orchestrator Console, right-click the *Jobs* container, then click *Deploy Job* to open the Select the Component File to Deploy dialog box.


Select the Component File to Deploy

Look In: components

File Name: /opt/novell/zenworks/zos/server/components

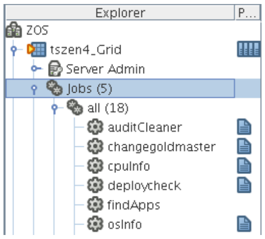
Files of Type: Deployable Files (*.job, sjob, jdl, cfact, eaf, sar, sched)

2 Open the *Look In* drop-down list, then navigate to the location of the job you want to deploy.

Although a job developer can store Orchestrator jobs at any location on the network, the sample jobs shipped with ZENworks Orchestrator are limited to the directories where the product is installed. For this walkthrough, navigate to the `/opt/novell/zenworks/zos/server/components/systemJobs` directory.

3 Select *auditCleaner.job*, then click *OK* to deploy the job to the *Jobs* container.

The job appears in the *all* container and in the *examples* container in the tree.

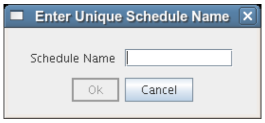


1.2.2 Creating a New Schedule for the Job

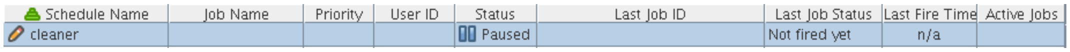
When a job has been deployed, you can create a schedule to specify when you want it to run. In this walkthrough, you create a schedule for the `auditCleaner` job by using the Scheduler tool in the Orchestrator Console.

1 From the toolbar in the Orchestrator Console, click the Job Scheduler icon 📅 to open the Job Scheduler view.

2 In the Job Scheduler view, click *New* to open the Enter Unique Schedule Name dialog box.



3 Specify a name for the schedule you want to create for this job. For this walkthrough, specify the name `cleaner` in the *Schedule Name* field, then click *OK* to return to the Job Scheduler view.



Schedule Name	Job Name	Priority	User ID	Status	Last Job ID	Last Job Status	Last Fire Time	Active Jobs
cleaner				Paused		Not fired yet	n/a	

The new schedule is highlighted in the Job Schedules Table and is flagged with a pencil icon, signifying that the schedule has not been committed yet. Continue with Section 1.2.3, "Defining the New Schedule," on page 20 to define this new schedule by adding the specific information you want.

1.2.3 Defining the New Schedule

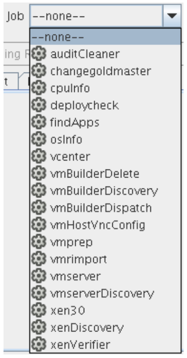
Defining a new job schedule consists of selecting its general properties, its specific properties, and the triggers you want to be associated with it. This section includes the following information:

- "Choosing General Properties for a New Schedule" on page 20
- "(Optional) Adding Specific Parameters to the New Schedule" on page 21
- "Creating and Assigning a Time Trigger for the New Schedule" on page 24

Choosing General Properties for a New Schedule

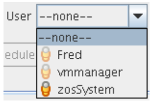
After you have created a new job schedule, its name cannot be changed, but you can add properties to it that help to identify and classify it in a general way. Use the following steps to add these properties:

1 In the Job Schedule Editor panel of the Scheduler view, select the *Job* drop-down list.



2 From the list of available jobs, select *auditCleaner* as the job to which this schedule applies.

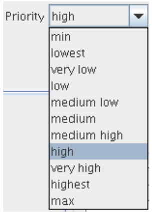
3 In the Job Schedule Editor, select the *User* drop-down list.



4 From the list of available users, select *zosSystem* as the user who runs this job.

The zosSystem user is the built-in user that is always present. It is commonly used for routine jobs like this example.

5 In the Job Schedule Editor, select the *Priority* drop-down list.



6 From the list of available priorities, select *medium* as the priority for this job schedule.

7 Click the *Save* icon on the toolbar of the Orchestrator Console to save the general properties you have selected for the new schedule.

The schedule is now committed, and the attribute columns in the Job Schedules Table are populated with the name of the job that the schedule will run, the user it will run as, the priority at which it will run, and its current status. Because the schedule has not been activated yet, it remains in a *Paused* state.

When you have chosen the general properties of the new schedule, you can either continue by "(Optional) Adding Specific Parameters to the New Schedule" on page 21 or by proceeding directly to "Creating and Assigning a Time Trigger for the New Schedule" on page 24.

(Optional) Adding Specific Parameters to the New Schedule

You can now add specific parameters to the schedule to edit its job arguments, to choose whether you want to pass the user environment variables to the schedule, or to specify policy constraints to further focus the purpose of this schedule when it fires.

For the purpose of this walkthrough, none of these specific parameters is modified, although a general overview of how to do so is explained.

The following specific parameters can be managed in the Job Scheduler Editor:

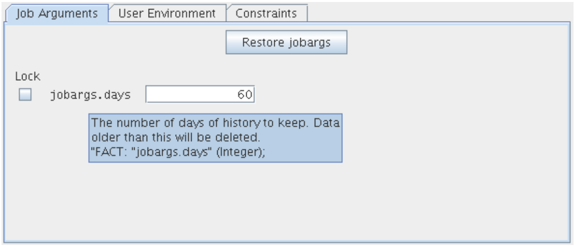
- "Job Arguments" on page 22
- "User Environment" on page 22
- "Constraints" on page 23

Job Arguments

As explained in Section 1.1.2, "Creating or Modifying a Job Schedule," on page 12, a job argument defines the values that can be passed in to the process when a job is invoked. These values let you statically define and control job behavior. The job arguments that appear in the *Job Arguments* tab of the Schedule Editor depend on the job. The job might have no arguments.

By default, the `auditCleaner` job lists only one job argument, *jobargs.days*.

Figure 1-8 *The Job Arguments Tab of the Job Schedule Editor*



□ jobargs.days 60

The number of days of history to keep. Data older than this will be deleted.
"FACT: "jobargs.days" (Integer);

According to the tool tip text, this argument is the number of days of job history to keep, so this job cleans up the history of the job in the Orchestrator audit database after the job reaches the age of 60 days. Data older than 60 days is to be deleted. If you want to, you can change this parameter, or any other parameter in a job argument.

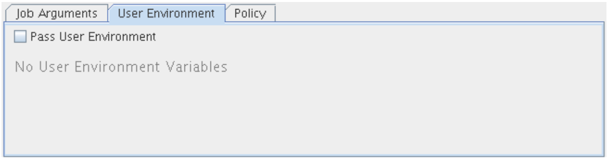
If the default value for a job argument parameter is missing, the job might fail, so you should inspect these parameters carefully.

User Environment

As explained in Section 1.1.2, "Creating or Modifying a Job Schedule," on page 12, a user's environment variables are available in the Job Scheduler only if that user utilizes the zos command line tool and elects to pass those environment variables to the server at login time or when he or she runs a job (running the job creates the environment variables as facts in the job). The `zos run` command passes the environment for that particular job run only.

In this walkthrough, the `zosSystem` user shows no user environment variables.

Figure 1-9 *The User Environment Tab of the Job Scheduler Editor, No User Environment Variables Available*



□ Pass User Environment

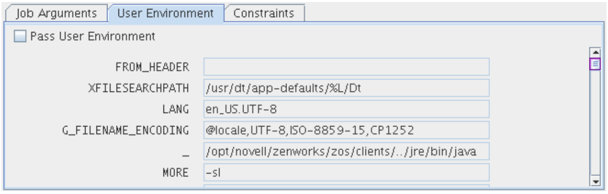
No User Environment Variables

Because there are no environment variables listed, there are none to pass to the schedule, so it is not necessary to select the *Pass User Environment* check box. By default, this check box is not selected, even if environment variables are present for a user specified to run the job.

Sometimes a job is written to work from a user's environment variables. In this case, if a user has not logged in or has not run the job from the zos command line using the necessary environment option, the schedule must pass those variables to the job when it is invoked.

If you associated a user who had user environment variables with this schedule, you would see a list of those environment variables as they would be passed to the job.

Figure 1-10 *The User Environment Tab of the Job Schedule Editor, User Environment Variables Available*



□ Pass User Environment

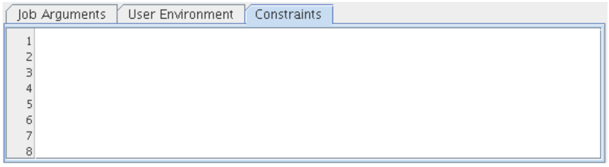
FROM_HEADER	
XFILESEARCHPATH	/usr/dt/app-defaults/%L/Dt
LANG	en_US.UTF-8
G_FILENAME_ENCODING	@locale,UTF-8,ISO-8859-15,CP1252
_	/opt/novell/zenworks/zos/clients/../jre/bin/java
MORE	-sl

Selecting the *Pass User Environment* check box in this scenario would create these variables as facts used for this job invocation.

Constraints

As explained in Section 1.1.2, "Creating or Modifying a Job Schedule," on page 12, the *Constraints* tab displays a constraint editor that you can use to create additional constraints for the job being scheduled.

Figure 1-11 *The Constraints Tab of the Job Schedule Editor*



Any other constraints associated with the context of this job invocation (including but not limited to this job), with the user you've selected, with that user's group, with the jobs group, with the resources that the job uses, or with the resource groups that the job uses, run in spite of the policy that you define here. These additional constraints usually restrict or refine what the job does when this schedule fires.

These constraints are passed to the job only when this schedule is invoked. For example, you could add a start constraint to delay the start of a job, a resource constraint to run on only one of three named machines, or a continue constraint to automatically time out the job if it takes too long to run. Anything you can do with a regular job policy constraint, you can add as a special constraint here for this particular schedule invocation.

Click *Save* icon to update the Orchestrator Server with the new schedule.

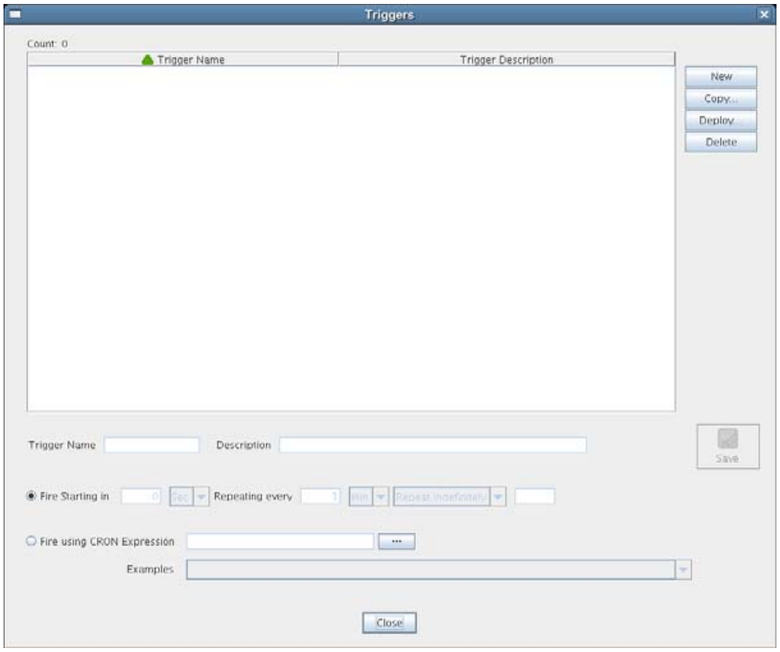
For more information about policies, see "Developing Policies" in the *Novell ZENworks Orchestrator 1.2 Developer Guide and Reference*.

Creating and Assigning a Time Trigger for the New Schedule

A job already defined in a schedule can be triggered with two main themes: the occurrence of an event or the arrival of a point in time. In this walkthrough, you define a time trigger for the `cleaner` schedule.

In this example, there are no defined time triggers in the Job Scheduler, so use the following steps to define a time trigger.

1 In the Job Schedule view, click *Triggers* to display the Triggers dialog box.



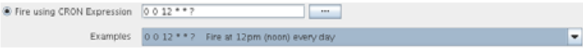
Time triggers are shareable across schedules. After a time trigger is defined, it is added to a list of triggers in this dialog box. You can select a predefined schedule from this list when you create a new schedule, or you can create a new time trigger, as the next steps demonstrate.

2 In the Triggers dialog box, click *New* to clear and activate the fields in the dialog box for the creation of a unique time trigger.

3 In the Enter Unique Trigger Name dialog box, specify `24 hour` as the unique name of this time trigger, then click *OK*.

4 In the *Description* field, specify `Runs every 24 hours at noon` as the description for this time trigger.

5 Click *Fire Using CRON Expression* to activate the fields for defining a cron expression.



● Fire using CRON Expression 0 0 12 * * ?

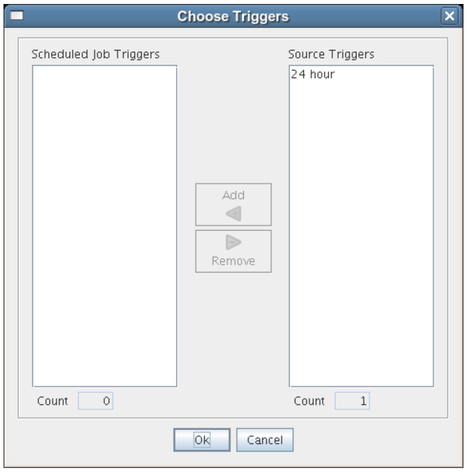
Examples 0 0 12 * * ? Fire at 12pm (noon) every day

6 Click the drop-down list of sample cron expressions, then select the default cron expression, `0 0 12 * * ?`, which is listed first.

The sample expressions in the drop-down list show cron strings with accompanying descriptions to remind you how a cron string is constructed. The examples are selectable and editable and can be used in the schedule, just as you have done in this step.

7. Click the *Save* icon to save the trigger you just created, then click *Close* to return to the Job Scheduler view.

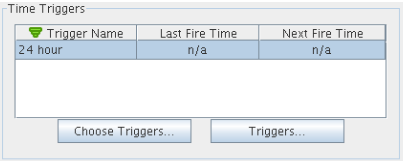
8. From the Job Scheduler view, make sure that the *cleaner* schedule is selected, then click *Choose Triggers* to display the Choose Triggers dialog box.



9. In the Choose Triggers dialog box, select *24 hour* (the name of the trigger you created), click *Add* to move the trigger definition to the *Scheduled Job Triggers* list, then click *OK* to return to the Job Scheduler view.

NOTE: You can select and combine as many time triggers as you want to apply to a given schedule. You can also combine time triggers with event triggers on a given schedule.

In the *Time Triggers* list of the Job Scheduler view, the *24 hour* trigger is now associated with the new schedule.



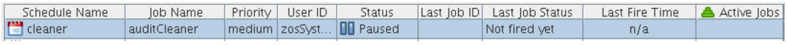
Trigger Name	Last Fire Time	Next Fire Time
24 hour	n/a	n/a

10. Click the *Save* icon to update the Orchestrator Server with the new schedule/trigger association.

1.2.4 Activating the New Schedule

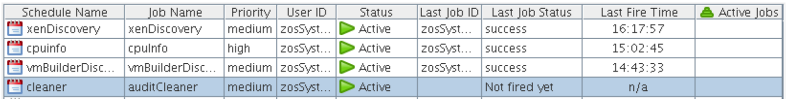
When the new schedule has been created and its triggers defined, you need to take it from the paused state to an active state where it is ready to run.

1. In the Job Scheduler view, select the newly created job. The job shows that it is in a *Paused* state.



Schedule Name	Job Name	Priority	User ID	Status	Last Job ID	Last Job Status	Last Fire Time	Active Jobs
cleaner	auditCleaner	medium	zosSyst...	Paused		Not fired yet	n/a	

2. Click *Resume* to activate the schedule.



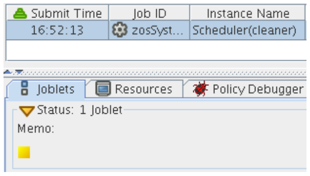
Schedule Name	Job Name	Priority	User ID	Status	Last Job ID	Last Job Status	Last Fire Time	Active Jobs
xenDiscovery	xenDiscovery	medium	zosSyst...	▶ Active	zosSyst...	success	16:17:57	
cpuinfo	cpuInfo	high	zosSyst...	▶ Active	zosSyst...	success	15:02:45	
vmBuilderDisc...	vmBuilderDisc...	medium	zosSyst...	▶ Active	zosSyst...	success	14:43:33	
cleaner	auditCleaner	medium	zosSyst...	▶ Active		Not fired yet	n/a	

The schedule is now active, but has not run yet.

1.2.5 Running the New Schedule Immediately

You can trigger the schedule immediately, rather than waiting for the triggers to fire.

1 In the Job Schedules Table of the Job Scheduler view, select *cleaner* (the name of the schedule you want to run), click *Run Now*, then click the job monitor icon on the toolbar to open the Job Monitor view.



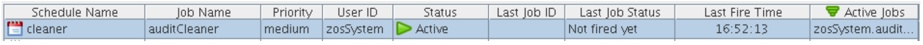
Submit Time	Job ID	Instance Name
16:52:13	zosSyst...	Scheduler(cleaner)

Status: 1 Joblet

Memo:

The joblet icon shows that the job is running.

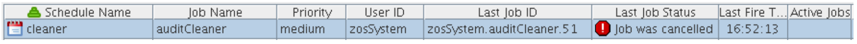
2 Click the Job Scheduler icon on the toolbar to open the Job Scheduler view.



Schedule Name	Job Name	Priority	User ID	Status	Last Job ID	Last Job Status	Last Fire Time	Active Jobs
cleaner	auditCleaner	medium	zosSystem	▶ Active		Not fired yet	16:52:13	zosSystem.audit...

The `cleaner` schedule is listed as an active job. This indicates that the schedule has started the job as anticipated.

If you click the refresh icon, you can see that the job now has a Job ID.



Schedule Name	Job Name	Priority	User ID	Last Job ID	Last Job Status	Last Fire T...	Active Jobs
cleaner	auditCleaner	medium	zosSystem	zosSystem.auditCleaner.51	! Job was cancelled	16:52:13	

If the job invocation fails, as in this example, a red exclamation icon is also displayed.

The Zosadmin Command Line Tool

A

The zosadmin command line tool is used by Novell® ZENworks® Orchestrator Server administrators to log in to the server, add or remove server components such as jobs and policies, to report on the status of nodes, users, and the audit database, and to perform other administrative functions.

This section includes information about the following:

- Section A.1, "List of Zosadmin Commands," on page 27
- Section A.2, "Getting Started with the Zosadmin Command," on page 28
- Section A.3, "Details, Usage, and Syntax Examples of Zosadmin Commands," on page 30

A.1 List of Zosadmin Commands

The following table includes a list of zosadmin commands in order of common usage, and a description for each command.

Table A-1 *Available Zosadmin Commands and Their Descriptions*

Command	Description
auditclean	Clean the audit database by removing old data
auditcount	Count the number of jobs in the audit database
auditreport	Generate an audit report
cancelalljobs	Cancel all running jobs
create	Create a new server instance
deploy	Deploy a new component onto a server
dump	Dump contents of the namespace (advanced diagnostics)
get	Retrieve an attribute
help	Displays help for any of the commands in this list
init	Restore a server configuration to initial state
invoke	Invoke an mbean method (advanced diagnostics)
list	List the running servers
login	Log in to the server
logout	Log out of the server
nodes	Retrieve the list of active or inactive nodes
password	Change admin password

Command	Description
redeploy	Redeploy a component on a server
rotatelogs	Back up and rotate the log files
sessions	Display session information
set	Set an attribute
start	Start a local server
status	Collect the status of a server
stop	Shut down the server
undeploy	Undeploy a component from the server
upgrade	Upgrade old server snapshot to current version
users	Retrieve the list of active or inactive users
verify	Verify a component for syntax

A.2 Getting Started with the Zosadmin Command

All zosadmin commands begin with zosadmin on the command line. The general format for a zosadmin command is `zosadmin` followed by the command name, followed by command line parameters, if needed:

```
zosadmin [standard_options] command [command_options_and_arguments]
```

Before you use the zosadmin CLI, make sure that your path is correctly pointing to the Orchestrator `/bin` directory.

This section includes the following information:

- Section A.2.1, "Logging In," on page 28
- Section A.2.2, "Checking Login Status," on page 29
- Section A.2.3, "Logging Out," on page 29

A.2.1 Logging In

Login is required to operate on a running server. The commands `start`, `list`, `init`, and `create` do not require a login. Use the following syntax to log in:

```
>zosadmin login -user=username Orhestrator_Server_name
Please enter current password for 'username': *******
Logged into Orchestrator_grid_name> on Orchestrator_Server_name
```

The login should now be complete.

Login information is stored in the `/home` directory, so further `zosadmin` commands use the saved login information. To operate on a different Orchestrator Server, run `zosadmin login` to log in to the new Orchestrator Server.

A.2.2 Checking Login Status

Enter the following command and parameter to retrieve the status of the current login:

```
>zosadmin login -c
Currently logged into testgrid on server 'tszen5'
```

A.2.3 Logging Out

Enter the following command to log out of the Orchestrator Server:

```
>zosadmin logout
Logged out from testgrid
```

A.3 Details, Usage, and Syntax Examples of Zosadmin Commands

This section includes a detailed list of the `zosadmin` commands you can use. It also includes examples for using these commands and shows the syntax of typical commands.

NOTE: Items shown in brackets [ ] are optional. Items shown in *italics* are contextual examples.

The section is organized according to the command names, which include the following:

- "auditclean" on page 31
- "auditcount" on page 32
- "auditreport" on page 33
- "cancelalljobs" on page 34
- "create" on page 35
- "deploy" on page 36
- "dump" on page 37
- "get" on page 38
- "init" on page 39
- "invoke" on page 40
- "list" on page 41
- "login" on page 42
- "logout" on page 44
- "nodes" on page 45
- "password" on page 46
- "redeploy" on page 47
- "rotatelogs" on page 48
- "sessions" on page 49
- "set" on page 50
- "start" on page 51
- "status" on page 52
- "stop" on page 53
- "undeploy" on page 54
- "upgrade" on page 55
- "users" on page 56
- "verify" on page 57

auditclean

This command cleans the audit database by removing old data.

Syntax

```
zosadmin auditclean     --dayskept= [--matrix=]
```

Options

-d, --dayskept=

Specify the number of days of history kept in the database (0 removes all).

-M=, --matrix=

Specify a different grid (the default grid is the server's grid, * means all).

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

auditcount

This command retrieves the count of the number of jobs in the audit database.

Syntax

```
zosadmin auditcount     [--root] [matrix=]
```

Options

-R, --Root

Display the count for root jobs only (default counts all jobs).

-M=, --matrix=

Specify a different grid (default grid is the server’s, * means all).

-v, --verbose

Turns on verbose output during this operation.

-V, --debug

Turns on debug output during this operation.

-h, --help

Displays a help message for this operation.

auditreport

This command displays an audit report generated from the audit database.

Syntax

```
zosadmin auditreport    [--username=] [--from=] [--to=] [--limit=]
                        [--matrix=] [--childjobs]
```

Options

-m, --username=

Specify the username of the user who ran the job (default is all users).

-F=, --from=

Select jobs submitted on or after this date. For example, `Fri, 12 Oct 2007` or `Fri, 12 Oct 2007 13:30:00`. The default day is the current day.

-T=, --to=

Select jobs submitted on or before this date. For example, `Fri, 12 Oct 2007` or `Fri, 12 Oct 2007 13:30:00`. The default day is the current time.

-L=, --limit=

Limit the number of jobs reported (the default is 500).

-M=, --matrix=

Specify a different grid (the default grid is the server's grid, * means all).

-I=, --childjobs

Include child jobs (the default is root jobs only).

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

cancelalljobs

This command cancels all running jobs.

Syntax

```
zosadmin cancelalljobs
```

Options

-v, --verbose

Turns on verbose output during this operation.

-V, --debug

Turns on debug output during this operation.

-h, --help

Displays a help message for this operation.

create

This command creates a new server instance.

Syntax

```
zosadmin create    [--newdir=] [--profile=] [--id=] [--grid=] [--user=]
                   [--passwd=] [--rmiport=] [--rmihost=] [--httpport=]
                   [--adminport=]
```

Options

-N, --newdir=

The location where the new Orchestrator Server should be installed.

-e, --profile=

Create a new instance based on named profile (the default is `server`).

-i, --id=

Select by kernel ID.

-g, --grid=

Select by grid name.

-u, --user=

Username used in accessing secure remote sites.

-p, --password=

Password used in accessing secure remote sites.

-P, rmiport=

Select by RMI port.

-H, --rmihost=

Select or specify the RMI host (can be different from *server host*).

-W=, --httpport=

Specify the http port used for the User Portal (the default is 80).

-M=, --adminport=

Specify the http port used for Administrative Information Portal (default 8001).

-V, --debug

Turns on debug output during this operation.

-v, --verbose

Turns on verbose output during this operation.

-h, --help

Displays a help message for this operation.

deploy

This command deploys a new component onto a server.

Syntax

```
zosadmin deploy   [--port=] file|dir
```

Arguments

file|dir

Specify the deployable file or a directory containing deployable files.

Options

-P=, --port=

The deployment port of the target server (the default is 8001, set with the `-M` option in the `create` command).

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

dump

This command dumps the contents of namespace.

Syntax

```
zosadmin dump     [--dir=] --jndi=
```

Options

-d=, --dir=

Specify the working directory for a Orchestrator Server installation.

-j=, --jndi=

Specify the JNDI path to be displayed.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

get

This command retrieves an attribute.

Syntax

```
zosadmin get      --mbean= --attr=
```

Options

-m=, --mbean=

Specifies the name of the mbean to view.

-a=, --attr=

Specifies the attribute name of the mbean to view.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

init

This command restores a server configuration to its initial state.

Syntax

```
zosadmin init
```

Options

-d, --dir=

The working directory for a Orchestrator Server installation.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

invoke

This command invokes an mbean method.

Syntax

```
zosadmin invoke   --mbean= --method=
```

Options

-m=, --mbean=

Specify the name of the mbean to view.

-i=, --method=

Specify the method name of the mbean to invoke.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

list

This command lists all of the running servers.

Syntax

```
zosadmin list     [--grid=] [--id=] [--server=] [--rmiport]
                  [--rmipath=] [--rmihost=] [--rmiurl=] [--timeout=]
```

Options

-g, --grid=

Select by grid name.

-i, --id=

Select by kernel ID.

-s=, --server=

Select by host name.

-P=, --rmiport=

Select by RMI port.

-J=, --rmipath=

Select or specify the RMI bind path.

-H=, --rmihost=

Select or specify the RMI host (can be different from *server host*).

-r=, --rmiurl=

Select by full RMI URL.

-t=, --timeout=

Sets the query timeout (in seconds).

-V, --debug

Turn on debug output during this operation.

-v, --verbose

Turn on verbose output during this operation.

-h, --help

Display a help message for this operation.

login

This command logs into the Orchestrator Server.

Syntax

```
zosadmin login    [--grid=] [--id=] [--rmiport=] [--rmipath=]
                  [--rmihost=] [--rmiurl=] [--timeout=] [--user=]
                  [--passwd=] [--check=] [serverhost]
```

Arguments

server host

Enter the server host name that you are logging into. The command polls for a server if none is supplied.

Options

-g, --grid

Select by grid name.

-i=, --id=

Select by kernel ID.

-P=, --rmiport=

Select by RMI port.

-J=, --rmipath=

Select or specify the RMI bind path.

-H=, --rmihost=

Select/specify RMI host (can be different from *server host*).

-r=, --rmiurl=

Select by full RMI URL.

-t, --timeout

Sets the query timeout (in seconds).

-u, --user

Username used in accessing secure remote sites.

-p, --passwd

Password used in accessing secure remote sites.

-c, --check

Turn on debug output.

-V, --debug

Turns on verbose output during the login operation.

-v, --verbose

Turns on verbose output during the login operation.

-h, --help

Display a help message for this operation.

logout

This command logs out of the Orchestrator Server.

Syntax

```
zosadmin logout
```

Options

-v=, --verbose=

Specifies the terminal width for formatting. The user must enter a value with this option.

-V, --debug

Turns on debug output during the logout operation.

-h, --help

Displays a help message for this operation.

nodes

This command retrieves a list of active or inactive nodes.

Syntax

```
zosadmin nodes    [--offline=]
```

Options

-o=, --offline=

Retrieve a list of offline users and nodes.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

password

This command lets the administrator or developer change his or her password on the Orchestrator Server.

Syntax

```
zosadmin password [--passwd=] [--newpasswd=]
```

Options

-p=, --passwd=

Password used in accessing secure remote sites.

-l=, --newpasswd=

New password to use in accessing secure remote sites.

-v, --verbose

Turns on verbose output during this operation.

-d, --debug

Turns on debug output during this operation.

-h, --help

Displays a help message for this operation.

redeploy

This command redeploys a component onto a server.

Syntax

```
zosadmin redeploy [--port] file|dir
```

Arguments

file|dir

Specify the deployable file or a directory containing deployable files.

Options

-p=, --port=

Specifies the deployment port of the target server (default is 8001, set with the `-M` option in the `create` command).

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

rotatelogs

This command backs up and rotates the log files.

Syntax

```
zosadmin rotatelogs     [--recursive]
```

Options

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

sessions

This command displays session information.

Syntax

```
zosadmin sessions [id=] [--node=] [--user=] [--all] [--allNodes] [--
allUsers] [--subSessions] [--inactive] [--full] [--noDns]
```

Options

-i, --id <value>

Display this session for this session ID.

-n, --node <value>

Display sessions for the specified node.

-u, --user <value>

Display sessions for the specified user.

-a, --all

Display sessions for all clients.

-N, --allNodes

Display sessions for all nodes.

-U, --allUsers

Display sessions for all users.

-s, --subSessions

Include subsession entries.

-I, --inactive

Include inactive entries.

-f, --full

Include full session information.

-d, --noDns

Do not look up host names in DNS.

-V, --debug

Turn on debug output during this operation.

-v, --verbose

Turn on verbose output during this operation.

-h, --help

Display a help message for this operation.

set

This command sets an attribute.

Syntax

```
zosadmin set      --mbean= --attr= --value=
```

Options

-m, --mbean=

Specify the name of the mbean to view.

-a, --attri=

Specify the attribute name of the mbean to view.

-o=, --value=

Specify the attribute value of the mbean to set.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

start

This command starts a local server.

Syntax

```
zosadmin start    [--dir=] [--jvmargs=] [--javaargs=] [--timeout=]
```

Options

-d=, --dir=

Specify the working directory for a Orchestrator Server installation.

-b=, --jvmargs=

Pass the value as an argument to an invoked JVM process.

-B=, --javaargs=

Pass the value as an argument to an invoked Java program.

-t=, --timeout=

Specify the query timeout (in seconds).

-G=, --upgrade

Switch upgrade behavior for snapshot handling.

-S=, --snapshot=

Upgrade a server using the snapshot directory of an existing server.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

status

This command displays the status of a server.

Syntax

```
zosadmin status
```

Options

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

stop

This command shuts down the server.

Syntax

```
zosadmin stop     [--force]
```

Options

-f=, --force=

Do not prompt for server shutdown confirmation.

-n=, --nosnap

Do not write out a snapshot of server state for later upgrade.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

undeploy

This command undeploys a component from the server.

Syntax

```
zosadmin undeploy [--port=] component|dir
```

Arguments

component|dir

Specify the deployed file or a directory containing the deployed files.

Options

-P=, --port=

Specify the deployment port of the target server (the default is 8001, set with the `-M` option in the `create` command).

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

upgrade

This command upgrades an older server snapshot to a current version.

Syntax

```
zosadmin upgrade snapshotdir
```

Arguments

snapshotdir

Specify the directory containing a server configuration snapshot.

Options

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

users

This command retrieves a list of active or inactive users.

Syntax

```
zosadmin users    [--offline=]
```

Options

-o=, --offline=

Retrieve a list of offline users.

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.

verify

This command verifies a component for syntax.

Syntax

```
zosadmin verify   file
```

Arguments

<file>

Specify the deployable file to verify.

Options

-v, --verbose

Turn on verbose output during this operation.

-V, --debug

Turn on debug output during this operation.

-h, --help

Display a help message for this operation.