



Zenoss Core Administration Guide

Release 6.0.0

Zenoss, Inc.

www.zenoss.com

Zenoss Core Administration Guide

Copyright © 2017 Zenoss, Inc. All rights reserved.

Zenoss, Own IT, and the Zenoss logo are trademarks or registered trademarks of Zenoss, Inc., in the United States and other countries. All other trademarks, logos, and service marks are the property of Zenoss or other third parties. Use of these marks is prohibited without the express written consent of Zenoss, Inc., or the third-party owner.

Amazon Web Services, AWS, and EC2 are trademarks of Amazon.com, Inc. or its affiliates in the United States and/or other countries.

Flash is a registered trademark of Adobe Systems Incorporated.

Oracle, the Oracle logo, Java, and MySQL are registered trademarks of the Oracle Corporation and/or its affiliates.

Linux is a registered trademark of Linus Torvalds.

RabbitMQ is a trademark of Pivotal Software, Inc.

SNMP Informant is a trademark of Garth K. Williams (Informant Systems, Inc.).

Sybase is a registered trademark of Sybase, Inc.

Tomcat is a trademark of the Apache Software Foundation.

VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions.

Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

All other companies and products mentioned are trademarks and property of their respective owners.

Part Number: 1611.17.311

Zenoss, Inc.
11305 Four Points Drive
Bldg 1 - Suite 300
Austin, Texas 78726

Contents

| | |
|--|-----------|
| About this guide..... | 7 |
| | |
| Chapter 1: Using Zenoss Core..... | 8 |
| Initial login..... | 8 |
| Interface and navigation..... | 10 |
| Customizing the dashboard..... | 13 |
| Search..... | 17 |
| Navigating the event console..... | 18 |
| Running a command from the browser interface..... | 21 |
| Working with triggers and notifications..... | 22 |
| Advanced user interface configuration..... | 34 |
| | |
| Chapter 2: Adding, discovering and modeling devices..... | 36 |
| Add a single device..... | 36 |
| Add multiple devices..... | 37 |
| Discovering devices..... | 37 |
| Modeling devices..... | 40 |
| About modeler plugins..... | 42 |
| Debugging the modeling process..... | 43 |
| | |
| Chapter 3: Working with devices..... | 44 |
| Viewing the device list..... | 44 |
| Working with devices..... | 44 |
| Managing devices and device attributes..... | 54 |
| | |
| Chapter 4: Configuration properties..... | 60 |
| Configuration property types..... | 60 |
| Configuration properties inheritance and override..... | 60 |
| Viewing and overriding device properties..... | 62 |
| Viewing and overriding event properties..... | 75 |
| Viewing and overriding network properties..... | 77 |
| | |
| Chapter 5: Monitoring templates..... | 79 |
| Creating templates..... | 79 |
| Renaming templates..... | 79 |
| Template binding..... | 79 |
| Example: Defining templates in the device hierarchy..... | 81 |
| Example: Applying templates to multiple areas in the device hierarchy..... | 81 |
| | |
| Chapter 6: Basic monitoring..... | 82 |
| Availability monitoring..... | 82 |
| Monitoring using ZenCommand..... | 91 |
| SNMP monitoring..... | 93 |
| Monitoring devices remotely through SSH..... | 94 |

| | |
|--|------------|
| Network map..... | 95 |
| Chapter 7: Performance monitoring..... | 98 |
| Monitoring templates..... | 99 |
| Template binding..... | 99 |
| Data sources..... | 100 |
| Data points..... | 101 |
| Data point aliases..... | 102 |
| Thresholds..... | 105 |
| Performance Graphs..... | 109 |
| Performance data retention..... | 111 |
| Chapter 8: Event management..... | 112 |
| Basic event fields..... | 112 |
| Other fields..... | 114 |
| Details..... | 115 |
| De-duplication..... | 115 |
| Auto-clear correlation..... | 116 |
| Event consoles..... | 117 |
| Event sources..... | 122 |
| Creating events manually..... | 123 |
| Event classes..... | 124 |
| Mapping and transformation..... | 125 |
| Event life cycle..... | 128 |
| Capturing email messages as events..... | 130 |
| SNMP traps and event transforms..... | 131 |
| Chapter 9: Production states and maintenance windows..... | 136 |
| Production states..... | 136 |
| Maintenance windows..... | 137 |
| Chapter 10: Organizers and path navigation..... | 140 |
| Classes..... | 140 |
| Groups..... | 142 |
| Systems..... | 143 |
| Locations..... | 143 |
| Inheritance..... | 146 |
| Chapter 11: User commands..... | 148 |
| Defining global user commands..... | 148 |
| Running global user commands..... | 149 |
| Defining user commands for a single device..... | 149 |
| Running user commands for a single device..... | 150 |
| Defining user commands for all devices in an organizer..... | 150 |
| Running user commands for devices in an organizer..... | 151 |
| User command example: Echo command..... | 151 |
| Chapter 12: Managing users..... | 152 |
| Creating user accounts..... | 152 |

| | |
|---|------------|
| Editing user accounts..... | 152 |
| User groups..... | 155 |
| Roles..... | 156 |
| Device access control lists..... | 156 |
| Chapter 13: Reporting..... | 159 |
| Organizing reports..... | 160 |
| Device reports..... | 160 |
| Event reports..... | 162 |
| Performance reports..... | 163 |
| Graph reports..... | 170 |
| Multi-Graph reports..... | 172 |
| Creating a custom device report..... | 176 |
| Scheduling reports..... | 177 |
| Chapter 14: ZenPacks..... | 178 |
| Displaying the list of installed ZenPacks..... | 178 |
| ZenPack information resources..... | 179 |
| Preparing to install or upgrade a ZenPack..... | 179 |
| Installing or upgrading a ZenPack..... | 180 |
| Removing a ZenPack..... | 181 |
| Creating a ZenPack..... | 182 |
| Chapter 15: General administration and settings..... | 183 |
| Events settings..... | 183 |
| Rebuilding the events index..... | 184 |
| Working with the job manager..... | 185 |
| Appendix A: Using the Appliance Administration menu..... | 187 |
| Configure Network and DNS..... | 187 |
| Configure IPv6 Network CIDR..... | 194 |
| Configure Timezone..... | 194 |
| Change Root Password..... | 195 |
| Change ccuser Password..... | 195 |
| Update System..... | 196 |
| Change SSL settings..... | 196 |
| Root Shell..... | 197 |
| Reboot / Poweroff System..... | 197 |
| Appendix B: SNMP device preparation..... | 199 |
| Net-SNMP..... | 199 |
| SNMP v3 support..... | 199 |
| Community information..... | 200 |
| System contact information..... | 201 |
| Extra information..... | 201 |
| Appendix C: Syslog device preparation..... | 202 |
| Forwarding syslog messages from UNIX/Linux devices..... | 202 |
| Forwarding syslog messages from a Cisco IOS router..... | 202 |

| | |
|---|------------|
| Forwarding syslog messages from a Cisco CatOS switch..... | 203 |
| Forwarding syslog messages using syslog-ng..... | 203 |
| Appendix D: TALES expressions..... | 205 |
| Examples..... | 205 |
| TALES device attributes..... | 206 |
| TALES event attributes..... | 207 |
| Appendix E: Monitoring large file systems..... | 209 |
| Procedure..... | 209 |
| Glossary..... | 210 |

About this guide

Zenoss Core Administration Guide provides an overview of Zenoss Core architecture and features, as well as procedures and examples to help use and configure the system.

Related Zenoss Core publications

| Title | Description |
|---|---|
| <i>Zenoss Core Administration Guide</i> | Provides an overview of Zenoss Core architecture and features, as well as procedures and examples to help use the system. |
| <i>Zenoss Core Configuration Guide</i> | Provides required and optional configuration procedures for Zenoss Core, to prepare your deployment for monitoring in your environment. |
| <i>Zenoss Core Installation Guide</i> | Provides detailed information and procedures for creating deployments of Control Center and Zenoss Core. |
| <i>Zenoss Core Planning Guide</i> | Provides both general and specific information for preparing to deploy Zenoss Core. |
| <i>Zenoss Core Release Notes</i> | Describes known issues, fixed issues, and late-breaking information not already provided in the published documentation set. |
| <i>Zenoss Core Upgrade Guide</i> | Provides detailed information and procedures for upgrading deployments of Zenoss Core. |

Additional information and comments

Zenoss welcomes your comments and suggestions regarding our documentation. To share your comments, please send an email to docs@zenoss.com. In the email, include the document title and part number. The part number appears at the end of the list of trademarks, at the front of this guide.

Using Zenoss Core

The browser interface of Zenoss Core provides a variety of means for navigating and managing your environment.

Initial login

The first time you log in to Zenoss Core, you will immediately be taken to a startup wizard where you will perform the following tasks:

- Set your admin password
- Set your personal login
- Discover devices (optional)
- Add Infrastructure (optional)

- 1 Launch your Zenoss Core application the first time by clicking on the Virtual Host Name in Control Center. You will be presented with the following page showing you the initial steps to follow:

Figure 1: Installation Wizard

Zenoss Installation Wizard

This wizard will guide you through the initial setup of Zenoss. Click Get Started to begin.



- 2 Click **Get Started** to begin the wizard. The Setup Users page appears.

Figure 2: Setup Users**Step 1: Setup Users**

Set admin password

The admin account has extended privileges, similar to Linux's root or Windows' Administrator. Its use should be limited to administrative tasks.

Password Must:

- Contain 8 or more characters
- Contain at least one number
- Contain at least one upper and lower case character

Admin password:

Confirm password:

Create your account

Enter information for your personal user account. You'll use this to perform most tasks.

Username:

Password:

Retype password:

Your email:

[« Previous](#) [Next »](#)

- 3 Set your admin password and create your personal user account. Click **Next**. The Network Discovery page appears.

Figure 3: Network Discovery**Step 2: Network Discovery**

Devices found via Discovery will be placed in the /Discovered Device Class

| Networks/Range | SNMP | SSH Authentication | Windows Authentication |
|--|---|--|--|
| Enter one or more networks (such as 10.0.0.0/24) or IP ranges (such as 10.0.0.1-50): <input type="text"/> | Community Strings: public private | Username: <input type="text"/> Password: <input type="text"/> | Administrator Username: <input type="text"/> Password: <input type="text"/> |
| Discover | | | |

Discoveries

| Status | Networks | Credentials | Duration | Job Log | Remove |
|--------|----------|-------------|----------|---------|--------|
| | | | | | |

Add network discoveries using the above form

[« Previous](#) [Next »](#)

- 4 You can discover devices by setting networks or IP ranges on this page. If you are not ready to discover devices, you can skip this page and add devices later. For more information about the fields on this page see [Adding, discovering and modeling devices](#) on page 36. When you are ready to continue, click **Next**. The Add Infrastructure page appears.

Figure 4: Add Infrastructure

Step 3: Add Infrastructure

| Category | Type | Connection Information | | | | | | |
|---------------------------------------|-------|--|--|--|--|--|--|--|
| <input checked="" type="radio"/> HTTP | /HTTP | Enter multiple similar devices, separated by a comma, using either hostname or IP Address: | | | | | | |
| <input type="radio"/> KVM | | | | | | | | |
| <input type="radio"/> Network | | | | | | | | |
| <input type="radio"/> Ping | | | | | | | | |
| <input type="radio"/> Power | | | | | | | | |
| <input type="radio"/> Printer | | | | | | | | |
| <input type="radio"/> Server | | | | | | | | |

| Status | Host | Credentials | Type | Duration | Job Log | Remove | Retry |
|---|------|-------------|------|----------|---------|--------|-------|
| Add infrastructure using the above form | | | | | | | |

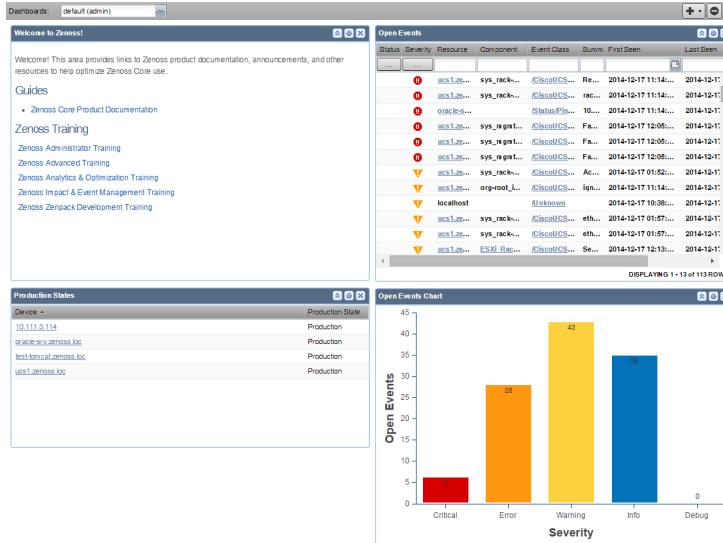
« Previous
✓ Finish

- 5 You can add devices from different categories on this page. For each device, select the category, type and enter the connection information. If you are not ready to add devices, you can skip this page. When you are ready to continue, click **Done**. You will be taken to the Dashboard view of Zenoss Core.

Note When you launch Zenoss Core in the future, you will go directly to the login screen.

Interface and navigation

After you install the system and navigate to the interface from your Web browser, the Dashboard appears. The Dashboard provides at-a-glance information about the status of your IT infrastructure. It is the primary window into devices and events that the system enables you to monitor.

Figure 5: Dashboard

The Dashboard can be customized to display a variety of information including:

- System information resources and other Web pages, such as internal portal pages
- Important error-level device events
- Geographical high-level view
- "Troubled" devices
- Devices in a certain production state You can also create additional dashboards that can be cloned from existing dashboards and can also be restricted to certain users or user groups.

Key Dashboard and interface areas include:

- Navigation menu
- User information area
- Portlets
- System Network Map

Navigation

The Navigation menu lets you access major system features. In addition to the Dashboard, the menu is divided among several functional areas:

- **EVENTS-** Guides you to the event management area, where you can monitor event status, triggers, and event transforms. You also can track changes made to events.
- **INFRASTRUCTURE-** Offers access to network infrastructure, including, devices, networks, processes, and services.
- **REPORTS-** Allows you access to pre-defined and configurable reports.
- **ADVANCED-** Provides access to monitoring templates, collectors, MIBs, system settings, and tuning.

User information area

Figure 6: User information area

The user information area offers information and selections:

- **Search-** Search area to find information within the application. Click the down arrow in the search box to manage your saved searches.
- **Login ID-** The ID of the user currently logged in appears at the far left of this area. Click the ID to edit user settings, such as authentication information, roles, and groups. (You also can access user settings from the **ADVANCED > Settings > Users** page.)
- **Sign Out-** Click to log out of the system.
- **Help icon** - Click to access product documentation.

Portlets

The main content of the Dashboard contains portlets that provide information about the system and infrastructure.

Available portlets are as follows, listed alphabetically:

Daemon Processes Down

Contains system self-monitoring information.

Device Chart

Allows the display of a graph of multiple data points for a selected device class.

Device Issues

Displays a list of devices associated with color-coded events of critical, error, or warning severity levels. To view details, click a device name. To go to the event console for the device, click an event.

Figure 7: Device Issues Portlet

| Icon | Device | Events |
|------|-------------------------------|--------|
| | 10.87.207.88 | 2 |
| | 10.87.207.89 | 36 |
| | 10.87.207.90 | 2 |
| | 10.87.207.124 | 1 |
| | 10.87.208.163 | 9 |

Event View

Displays the list of events similar to the view on the Event console.

Google Maps

Shows configured device locations and configured network connections.

Figure 8: Google Maps Portlet**HTML Portlet**

Displays HTML content. You must use HTML markup in this portlet. If you want to populate a portlet with content from a specific URL, use the **Site Window** portlet instead.

Multi-Graph Report

Displays an existing Multi-Graph Report (created by using the **Reports** page). You can choose a specific graph group from the multi-graph report and select the time range for the portlet.

Network Map

Displays a network map for a defined network that is being monitored. You can define the refresh interval and level of depth of the map.

Open Events Chart

Displays a chart of the number of open events grouped by severity. You can define the event class to be used and the number of past days for which to show events.

Organizers

Choose the root organizer (devices, locations, systems, groups) and then child organizer options are enabled.

Past Events Line Chart

Displays a line chart of past events grouped by severity. You can define the event class to be used and the number of past days for which to show events.

Production States

Shows devices assigned to a particular production state. If needed, you can define multiple production states to display.

Site Window

Initially provides links to resources such as product guides, forums, and training events.

You can customize the portlet to display content from any URL. However, Zenoss recommends that you keep a portlet with the default URL so that you can stay up-to-date with Zenoss training and product updates. You can have multiple **Site Window** portlets defined on your Dashboard.

Watch List

Allows the display of high-level status device classes, groups, systems, event classes, and locations that you select.

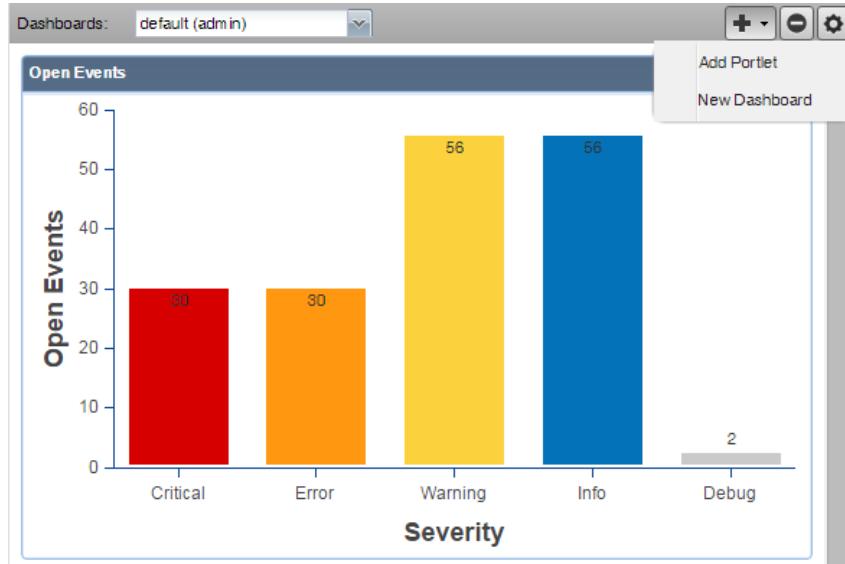
Customizing the dashboard

You can customize the dashboard by:

- Creating multiple dashboards
- Selecting the portlets you want to view

- Arranging portlets
- Defining who can view the dashboard
- Changing the dashboard column layout

The following image shows the Add menu.



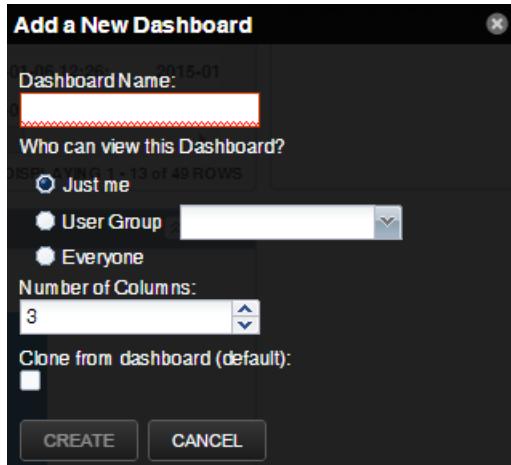
Adding a dashboard

A default admin dashboard is created when you launch the system. Administrators can customize this dashboard. However, the default dashboard cannot be deleted.

Users that are not administrators initially see a read-only version of the default admin dashboard. Non-administrators can create dashboards that display distinctive information or are targeted to a specific type of user or to only themselves. To customize a dashboard, select who can view it, and select and customize portlets to display the most important information. The number of customized dashboards is not limited.

To create a dashboard:

- 1 From the **Add** icon on the Dashboard controls, select **New Dashboard**.
- 2 Complete fields in the **Add a New Dashboard** dialog box as follows:



Dashboard Name

When this dashboard name is displayed in the Dashboards drop-down list, the user name who created it is appended as part of the name. This convention gives everyone who can see the dashboard an indication of who created it.

Who can view this Dashboard?

Choose who can view this dashboard. To specify a **User Group**, the group must exist in the system and the user name used to create this dashboard must be a member of the group. You can only assign dashboards to groups to which you belong.

Number of Columns

Choose the number of columns to display in the dashboard. The default is 3.

Clone from dashboard

If you want the new dashboard to be a clone of the dashboard you were previously viewing, select the check box. Otherwise, you begin with a completely blank dashboard.

- 3 Click **Create**.

Adding portlets

You can customize your dashboard by adding portlets that display information you are interested in. Your dashboard can display more than one of the same portlet type. For example, you could have several Device Chart portlets with each one showing a different device class.

To add a portlet to the Dashboard:

- 1 Click the **Add** icon in the top right of the Dashboard main area and select **Add portlet**.
- 2 Select a portlet from the drop-down list. The portlet appears at the top right of the Dashboard main area

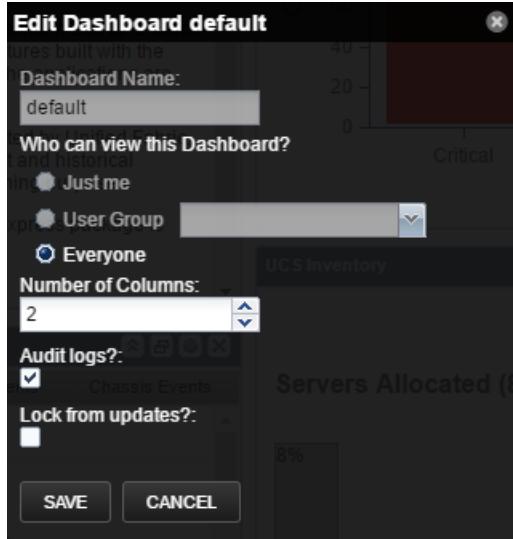
Arranging portlets

To arrange portlets, click the portlet header and drag the portlet to any location on the dashboard. The other portlets are automatically rearranged.

Editing the dashboard settings

Customize your dashboard to display a different number of columns or limit access to the dashboard.

- 1 Click the **Action** icon in the upper-right side of the dashboard. The **Edit Dashboard** window appears.



- 2 Change the value of the users who can view this dashboard.

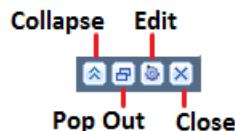
User groups: To enable a user group to view the dashboard, the group must exist in the system and you must be logged in with credentials of a member of the user group.

- 3 Change the number of columns to use in this dashboard if needed.
- 4 To ensure that changes are logged properly, verify the **Audit logs** setting. Clear the check box if you do not want logging on any changes.
- 5 Select **Lock from updates** to prevent editing of the dashboard.
- 6 Click **Save**.

Working with portlets

There are several options to control the portlet display.

Figure 9: Dashboard Portlet Controls



- Click the **Collapse** icon to collapse the portlet so that only the title appears on the dashboard.
- Click the **Pop Out** icon to show the portlet in a full screen view. Click **Close** to return to the dashboard view.
- Click the **Edit** icon to edit the portlet settings. You can edit the title of the portlet, its height and how often it refreshes. Some portlets may have additional configuration options. A preview of the portlet is provided on the right side of the dialog box. Click **Save** to update the portlet configuration.
- Click the **Close** icon to remove the portlet from the dashboard.

In tabular portlets, you can control the display by sorting columns as well as adding and hiding columns.

- To sort based on a column, hover over the column header and click the arrow to display the sort and display options.
- To add or hide columns, hover over the **Columns** entry and check or clear the boxes of the columns to add or hide.

| Status | Severity | Resource | Component | Event Class | Summary | First Seen | Last Seen | Count |
|------------------------------|------------------|--------------|-----------|-------------|---------|------------|-----------|-------|
| ... | ... | | | | | | | |
| ! | ga-centos-7-s... | | | | | | | |
| ! | ga-centos-5-s... | | | | | | | |
| ! | Cisco 10.171... | | | | | | | |
| ! | Cisco 10.171... | | | | | | | |
| !! | nexus-5k.zeno... | VSAN0080 ... | /Status | | | | | |
| !! | nexus-5k.zeno... | VSAN0001 ... | /Status | | | | | |
| !! | N6001-2 | Fex-108 A... | /Status | | | | | |
| !! | N6001-2 | Fex-108 P... | /Status | | | | | |
| !! | N6001-2 | Fex-109 A... | /Status | | | | | |
| !! | N6001-2 | Fex-115 A... | /Status | | | | | |
| !! | N6001-2 | Fex-131 P... | /Status | | | | | |
| !! | N6001-2 | Fex-114 P... | /Status | | | | | |
| !! | N6001-2 | Fex-110 A... | /Status | | | | | |
| DISPLAYING 1 - 13 of 29 ROWS | | | | | | | | |

Search

The Zenoss Core search facility supports locating devices and other system objects, as well as events and services.

In the Zenoss Core interface, the search feature is part of the user information area. Enter part or all of a name in the search box at the top right of the interface. The system displays matches, categorized by type.

Figure 10: Search Results

To view all search results, click the indicator at the top of the list.

Figure 11: All Search Results

| Category | Count | Description |
|------------------|-------|--|
| Device | 25 | bbbscout.zenoss.loc (10.87.209.23) |
| Event | 91 | cob2800.zenoss.loc (10.175.200.2) |
| CiscoStackModule | 1 | colo560g.zenoss.loc (10.175.211.2) |
| CPU | 28 | gatecolo.zenoss.loc (10.175.211.1) |
| FileSystem | 60 | mail.zenoss.com (74.83.38.28) |
| IpInterface | 188 | test-ax53-1.zenoss.loc (10.204.210.19) |
| IpRouteEntry | 89 | test-ax81.zenoss.loc (10.204.210.16) |
| IpService | 250 | test-cent5-84-1.zenoss.loc (10.175.211.15) |
| OSProcess | 1 | test-exch2003-1d.zenoss.loc (10.175.211.164) |
| VPNTunnel | 7 | test-exch2007-1d.zenoss.loc (10.175.211.165) |
| WinService | 250 | test-hpux.zenoss.loc (10.204.210.23) |
| | | test-jboss.zenoss.loc (10.175.211.235) |
| | | test-rhel54.zenoss.loc (10.175.211.137) |
| | | test-solaris10.zenoss.loc (10.175.211.23) |
| | | test-sq2005-1d.zenoss.loc (10.175.211.133) |
| | | test-sq2008-1d.zenoss.loc (10.175.211.63) |

From here, you can display search results by category. Click in the left panel to filter search results by a selection.

You can save the search to access later.

- 1 Click **Save As** (at the bottom left of the Search Results page). The Save Search As dialog box appears.
- 2 Enter a name for the search, and then click **Submit**.

You can access saved searches from:

- Action menu located at the bottom of the Search Results page.
- Search box located at the top of the interface. Click the arrow, and then select **Manage Saved Searches**.

Navigating the event console

The event console is the system's central nervous system, enabling you to view and manage events. It displays the repository of all events that are detected by the system.

To access the event console, click **Events** in the Navigation menu. The Event Console appears.

Figure 12: Event Console

| Status | Severity | Resource | Component | Event Class | Summary | First Seen | Last Seen | Count |
|--------|----------|----------|------------------------------|-------------|--|------------------------|------------------------|-------|
| ... | ... | ucs1 | Fan 3/4/2 | | Fan 2 in Fan Module 1-4 under chassis 3 speed: lower-non-recoverable | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 3 |
| ... | ... | ucs1 | extpol_reg_clients_client... | | UCS Domain ucs1-faba is registered with UCS Central without a valid license. | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 4 |
| ... | ... | ucs1 | Fan Module 3/4 | | Fan 2 in Fan Module 1-4 under chassis 3 speed: lower-non-recoverable | 2015-07-29 08:45:41 am | 2015-07-29 08:45:41 am | 1 |
| ... | ... | ucs1 | fabric_san_A_phys-fcoes... | | FCoE uplink port 1/8 is down | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 4 |
| ... | ... | ucs1 | Fan 3/4/2 | | Fan 2 in Fan Module 1-4 under chassis 3 operability: inoperable | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 3 |

Sorting and filtering events

You can sort and filter events that appear in the event console to customize your view.

You can sort events by any column that appears in the event console. To sort events, click a column header. Clicking the header toggles between ascending and descending sort order.

Filter options appear below each column header. A match value can be any full string or a subset of a string, optionally with the wildcard (*) contained in the values in that column. You can also use " | | " (OR), or " ! ! " (NOT) expressions to further target your filters. For example, typing ! windows in the Event Class filter will return all the non-Windows device events.

Figure 13: Event Console filter options

| Status | Severity | Resource | Component | Event Class | Summary | First Seen | Last Seen | Count |
|--------|----------|----------|------------------------------|-------------|--|------------------------|------------------------|-------|
| ... | ... | ucs1 | Fan 3/4/2 | | Fan 2 in Fan Module 1-4 under chassis 3 speed: lower-non-recoverable | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 3 |
| ... | ... | ucs1 | extpol_reg_clients_client... | | UCS Domain ucs1-faba is registered with UCS Central without a valid license. | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 4 |
| ... | ... | ucs1 | Fan Module 3/4 | | Fan 2 in Fan Module 1-4 under chassis 3 speed: lower-non-recoverable | 2015-07-29 08:45:41 am | 2015-07-29 08:45:41 am | 1 |
| ... | ... | ucs1 | fabric_san_A_phys-fcoes... | | FCoE uplink port 1/8 is down | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 4 |
| ... | ... | ucs1 | Fan 3/4/2 | | Fan 2 in Fan Module 1-4 under chassis 3 operability: inoperable | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 3 |
| ... | ... | ucs1 | Fabric FC SAN Port Chan... | | san port-channel 5 on fabric interconnect A oper state: failed, reason: No operatio... | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 4 |

You can filter the events that appear in the list in several ways, depending on the field type:

- **Resource** - Enter a match value to limit the list.

- **Component** - Enter a match value to limit the list.
- **Event Class** - Enter a match value to limit the list.
- **Summary** - Enter a match value to limit the list.
- **First Seen** - Enter a value or use a date selection tool to limit the list.
- **Last Seen** - Enter a value or use a date selection tool to limit the list.
- **Count** - Enter a value to filter the list, as follows:
 - N - Displays events with a count equal to N .
 - $:N$ - Displays events with a count less than or equal to N .
 - $M:N$ - Displays events with a count between M and N (inclusive).
 - $M:$ - Displays events with a count greater than or equal to M .

To clear filters, select **Configure > Clear filters**.

You also can re-arrange the display order of columns in the event console. Click-and-drag column headers to change their display.

Creating an actionable view

For users that are not Administrators, an option filters the list of events to show only those that are not read-only for the user's permission level, and enable the action buttons above the event table header.

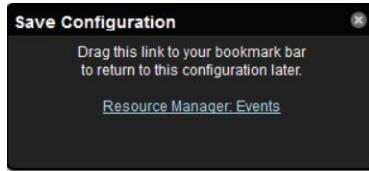
To turn on the actionable view, click **Configure** and select the **Only show actionable events** check box. The view is changed to show only events that can have an action performed on them based on the user's permission level. For more information, see [Managing events](#) on page 20.

Saving a custom view

Save a custom event console view by bookmarking it for quick access.

- 1 Select **Configure > Save this configuration**.
- 2 In the dialog box, select the link, and then drag it to the bookmarks area of the browser window.

Figure 14: Saving a custom view (bookmark)



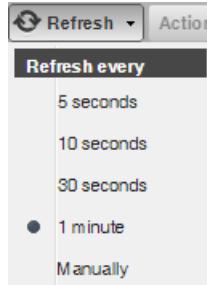
The browser adds a link to the bookmarks list.

- 3 Change the title of the bookmark to distinguish this event console view.

Refreshing the view

You can refresh the list of events manually or specify that they refresh automatically. To manually refresh the view, click **Refresh**. You can manually refresh at any time, even if you have an automatic refresh increment specified.

To configure automatic refresh, select one of the time increments from the Refresh list. By default, automatic refresh is enabled and set to refresh each minute.

Figure 15: Automatic refresh selections

Viewing event details

You can view details for any event in the system. To view details, double-click an event row.

Note Do not double-click on or near the device (resource) name, component, or event class in the row. Doing this displays details about that entity, rather than information about the event.

The Event Detail area appears.

Figure 16: Event Detail

A screenshot of the 'Event Detail' window. At the top, it shows a yellow exclamation mark icon and the text 'Server 4/6 (service profile:) has'. Below this is a list of event details:

- Resource: [10.87.207.89](#)
- Component: [chassis-4/blade-6](#)
- Event Class: [/CiscoUCS/Fault](#)
- Status: New
- Message: Server 4/6 (service profile:) has ins number of DIMMs, CPUs and/or ad

At the bottom of the window, there are several links: 'Event Management...', 'Device State...', 'Event Data...', and 'Event Details...'. A red box highlights the 'Event Details...' link, and a red arrow points to the 'Open in new window' button located at the top right of the window.

To see more information about the event, click **Event Details**.

You can use the Log field (located at the bottom of the area) to add specific information about the event. Enter details, and then click **Add**.

Selecting events

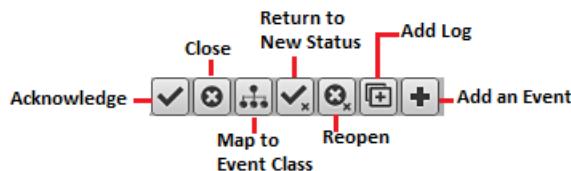
To select one or more events in the list:

- To select a single event, click a row.
- To select multiple events, Ctrl-click each row or Shift-click rows to select a range of events.
- To select all events, click **Select > All**.

Managing events

Use the event console to manage events or add an event. Click an event row and use the following tools to perform actions.

Figure 17: Event Management Options



- Acknowledge the event.
- Close the event.
- Reclassify the event by associating it with a specific event class.
- Return the event to New status (revoke its Acknowledged status).
- Reopen the event.
- Add a note to the log.
- Add an event. This feature is useful for testing a specific condition by simulating an event.

You also can

Running a command from the browser interface

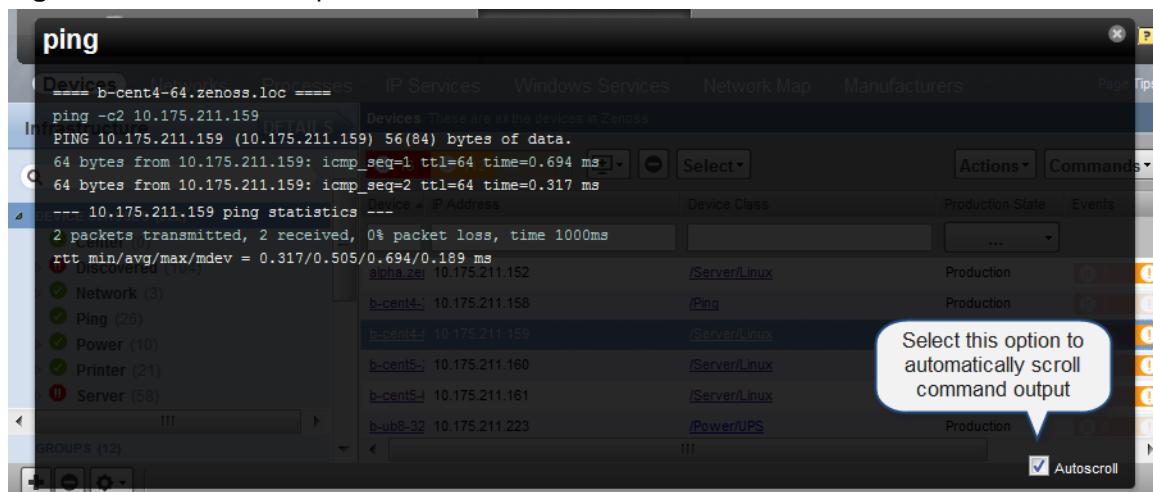
Zenoss Core allows commands to be run through the browser interface. You can run commands on a single device or on a group of devices.

The system includes several built-in commands, such as `ping` and `traceroute`.

To run commands from the browser interface:

- 1 Navigate to the **INFRASTRUCTURE** tab.
- 2 In the **Devices** list, select one or more devices.
To select a device, click anywhere in the row—except on the link.
- 3 Click **Commands** and select a command from the list.

Figure 18: Command Output



You can resize the command output window. You also can stop automatic scrolling by de-selecting the Autoscroll option at the bottom right corner of the output window.

Working with triggers and notifications

You can create *notifications* to send email or pages, create SNMP traps, or execute arbitrary commands in response to an event. Use notifications to notify other management systems and execute arbitrary commands to drive other types of integration. How and when a notification is sent is determined by a *trigger* that specifies a rule (one or more conditions).

To set up a notification:

- Create a trigger by defining rules.
- Create a notification by selecting one or more triggers that cause it to run.
- Choose appropriate options and subscribers depending on the notification type.

Working with triggers

Setting up a trigger involves:

- Creating the trigger and the rules that define it
- Setting trigger permissions

Creating a trigger

This procedure describes how to create a trigger.

1 Select EVENTS > Triggers.

The Triggers page displays all defined triggers.

2 Click the Add icon.

3 In the Add Trigger dialog, enter a name for the trigger, and then click Submit.

Only uppercase letters, lowercase letters, digits, and underscores are allowed in trigger names.

The trigger is enabled and added to the list.

To complete the trigger, proceed to the next task.

Editing a trigger

This procedure describes how to edit a trigger.

1 Select EVENTS > Triggers.

The Triggers page displays all defined triggers.

2 Open the Edit Trigger dialog box of the trigger to edit.

In the Triggers table, double-click a trigger, or select a trigger and then click the Action icon.

Figure 19: Edit Trigger dialog box

The screenshot shows the 'Edit Trigger - SampleTrigger' dialog box. The 'Trigger' tab is selected. The 'Name:' field contains 'SampleTrigger'. The 'Enabled:' checkbox is checked. The 'Rule:' dropdown is set to 'all' of the following rules. Below the dropdown, there are two rows of conditions: 'Device Production Stat equals Production' and 'Severity is greater than or equal Error'. There are also buttons for adding and removing rules.

- 3 Define rules for the trigger, and then click SUBMIT.**

Trigger rules combine Boolean logic with event values to decide whether to send notifications. A trigger can use any or all of its rules to make a decision, and each rule can have subordinate rules or branches. If a trigger rule contains text, note that text string evaluation is case-sensitive, so take this into consideration when adding a text-based trigger rule.

Note Device production states can change during maintenance windows. If you want the same trigger to apply during maintenance windows, be sure to edit your trigger to account for all production states that apply to your trigger.

Setting global trigger permissions

You can set global permissions for viewing, editing, and managing triggers. Global permissions are given to any user with "manage" permission, which includes:

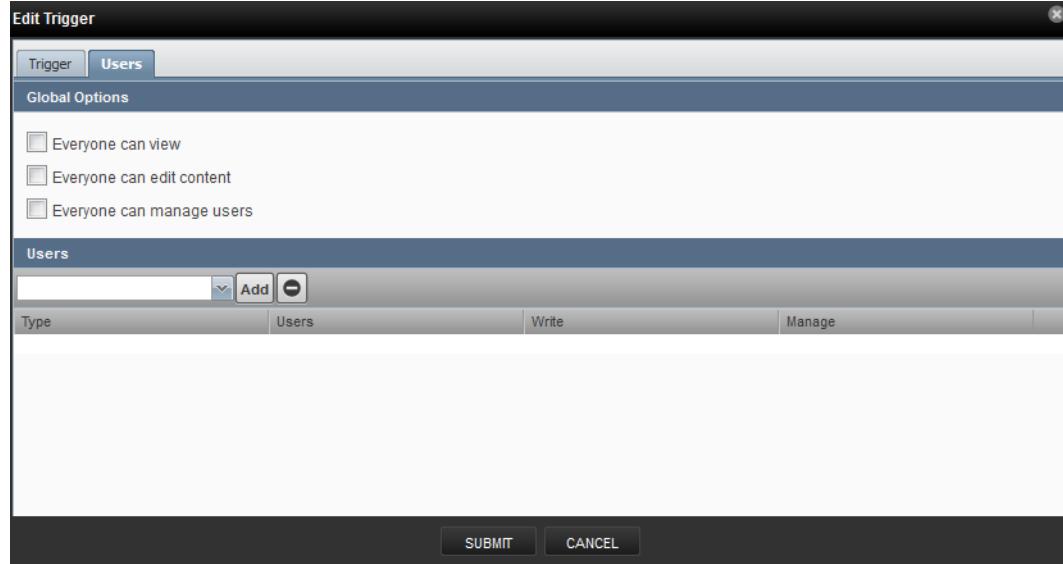
- Admin, Manager, and ZenManager roles
- Trigger owner

Edit global permissions from the Users tab on the Edit Trigger dialog.

Global options are:

- **Everyone can view** - Provides global view permission.
- **Everyone can edit content** - Provides global update permission.
- **Everyone can manage users** - Provides global manage permission.

Figure 20: Edit Trigger - Users tab



Setting individual trigger permissions

You can grant permissions to individual users.

- **Write** - Select this option to grant the user permission to update the trigger.
- **Manage** - Select this option to grant the user permission to manage the trigger.

To set an individual's trigger permissions:

- 1 In the **Edit Trigger** dialog box **Users** section, choose a user.

- 2 Click **Add**.
- 3 Select check boxes for the permissions to assign.
- 4 Repeat these steps to add more user trigger permissions.
- 5 Click **Submit**.

To remove an individual's trigger permissions:

- 1 Select the row of the user's permissions.
- 2 Click the **Remove** icon.
- 3 Repeat these steps to remove other user trigger permissions.
- 4 Click **Submit**.

Working with notifications

Setting up a notification includes the following tasks:

- Create the notification.
- Define notification content (for email- or page-type notifications).
- Define the SNMP trap host (for SNMP trap-type notifications).
- Define commands to run (for command-type notifications).
- Set notification permissions.
- Set up notification schedules.

Creating a notification

This procedure describes how to edit a notification.

- 1 Select **EVENTS > Triggers**.
- 2 In the left panel, select **Notifications**.

Figure 21: Notifications

| Notifications | | | | | Notification Schedules | | |
|---------------|------------|------------|---------|-------------|------------------------|------|-----------|
| Enabled | ID | Trigger | Action | Subscribers | Enabled | ID | Start |
| Yes | rancid-run | rancid-run | command | 0 | No | test | 1/14/2015 |

The **Notifications** area includes a table of notifications and a table of notification schedules.

- 3 Click the **Add** icon.
- 4 In the **Add Notification** dialog, provide a name and specify an action.
 - a Enter the **Id** of the notification.
Spaces are not allowed in notification Ids.
 - b Associate an action with the notification.
For more information, see [Notification actions](#) on page 24.
 - c Click **SUBMIT**.

To complete the notification, proceed to the next task.

Notification actions

| Action | Description |
|---------|---|
| Command | Invoke a shell command when events occur. Common uses of this action include: |

| Action | Description |
|------------|--|
| | <ul style="list-style-type: none"> ■ <i>Auto-remediation of events.</i> You can use ssh or wincommand to restart services on Linux and Windows devices. ■ <i>Integration with external systems.</i> For example, opening tickets in an incident management system. ■ <i>Extending alerting mechanisms.</i> Zenoss Core supports email and pagers as alerting mechanisms "out of the box" through normal alerting rules. |
| Email | Sends an HTML or text email message to authorized subscribers when an event matches a trigger rule. |
| Syslog | Sends a message to the syslog. |
| SNMP Trap | Sends an SNMP trap when an event matches a trigger rule. |
| WinCommand | Sends or clears a Windows CMD command. |

Editing a notification

This procedure describes how to edit a notification.

- 1 Select **EVENTS > Triggers**.
- 2 In the left panel, select **Notifications**.

Figure 22: Notifications

| Enabled | ID | Trigger | Action | Subscribers |
|---------|------------|------------|---------|-------------|
| Yes | rancid-run | rancid-run | command | 0 |

| Enabled | ID | Start |
|---------|------|-----------|
| No | test | 1/14/2015 |

The **Notifications** area includes a table of notifications and a table of notification schedules.

- 3 Open the **Edit Notification** dialog of the notification to edit.

In the **Notifications** table, double-click a notification, or select a notification and then click the **Action** icon.

Figure 23: Edit Notification

- 4 Define the settings for the notification.

For more information, see [Notification settings](#) on page 26.

Notification settings

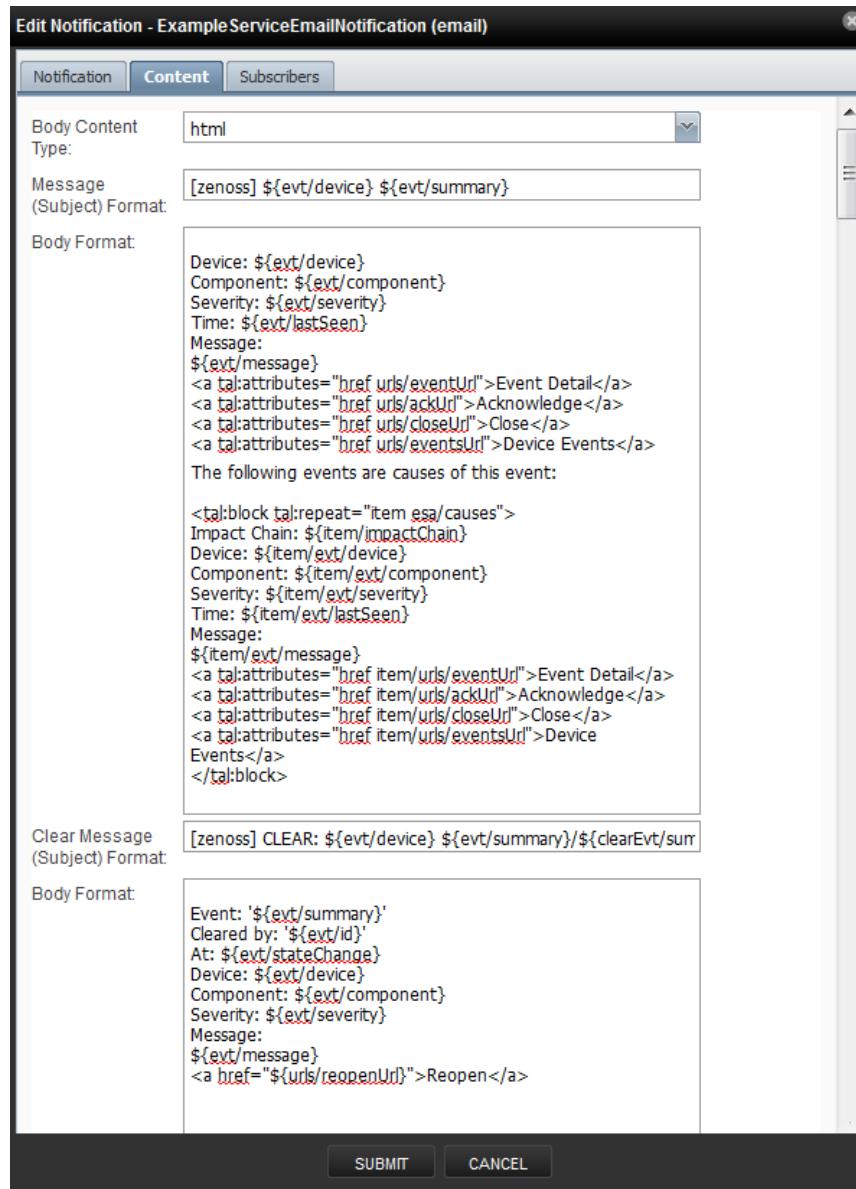
| Setting | Description |
|---------------------------------|--|
| Enabled | Check the checkbox to enable the notification. |
| Send Clear | Send a notification when the problem is resolved by a clearing event. |
| Send only on Initial Occurrence | Send a notification only when the first triggering event occurs. |
| Delay (seconds) | <p>The minimum number of seconds to wait before performing a notification. A delay prevents notifications of transient problems and multiple notifications of the same problem.</p> <p>For example, if five events that match the trigger occur in 45 seconds, a delay of 60 seconds ensures that only one notification is sent. Also, if a triggering event repeats 15 seconds after the initial event, followed by a clearing event at 45 seconds, a 60-second delay ensures that no notifications are sent.</p> |
| Repeat (seconds) | The interval between repetitions of the notification, in seconds. The notifications repeat until the triggering event is resolved. |

Defining notification content

To define notification content, click the **Content** tab.

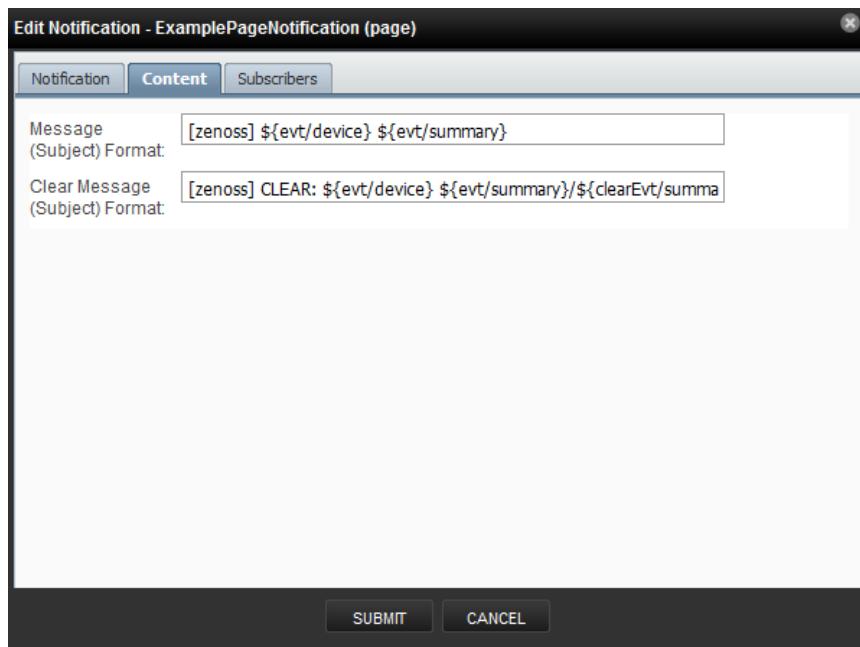
For email-type notifications, you can use the default configuration for the following fields, or customize them:

- **Body Content Type** - Select HTML or text.
- **Message (Subject) Format** - Sent as the subject of the notification.
- **Body Format** - Sent in the notification.
- **Clear Message (Subject) Format** - Sent when a notification clears.
- **Body Format** - Sent when a notification clears.
- **From Address for Emails** - Sent as the sender's email address.
- **Various SMTP settings** - Used to define SMTP host, port, user name, and password. To set these system-wide, use the **ADVANCED > Settings** page.

Figure 24: Define notification content (email)

For page-type notifications, you can use the default configuration for the following fields, or customize them:

- **Message (Subject) Format** - Sent as the subject of the notification.
- **Clear Message (Subject) Format** - Sent when a notification clears.

Figure 25: Edit Notification content (page)

Notification content variables

Within the body of email, page, and command notifications, you can specify information about the current event in the following form:

```
'${objectname/objectattribute}'
```

Note Do not escape event command messages and event summaries. For example, write this command as: \${evt/summary} (rather than echo '\$evt/summary').

Object names can be `evt`, `evtSummary`, or `urls`; or for clearing event context, `clearEvt` and `clearEventSummary`. For each object name, the following lists show valid attributes (for example, '\${evt/DevicePriority}'):

Table 1: evt/ and clearEvt/

| Value | Description |
|----------------|--|
| DevicePriority | value of the priority of the device |
| agent | Typically the name of the daemon that generated the event. For example, an SNMP threshold event has <code>zenperfsnmp</code> as its agent. |
| clearid | id of the event this clear event will clear |
| component | component this event is related to |
| count | how many times this event occurred |
| created | when the event was created |
| dedupid | dynamically generated fingerprint that allows the system to perform deduplication on repeating events that share similar characteristics |

| Value | Description |
|----------------|--|
| device | device this event is related to |
| eventClass | class of this event |
| eventClassKey | Free-form text field that is used as the first step in mapping an unknown event into an event class. |
| eventGroup | Free-form text field that can be used to group similar types of events. This is primarily an extension point for customization. Currently not used in a standard system. |
| eventKey | Free-form text field that allows another specificity key to be used to drive the de-duplication and auto-clearing correlation process. |
| eventState | state of the event |
| evid | unique id for the event |
| facility | the syslog facility |
| firstTime | First time that the event occurred. |
| ipAddress | IP address |
| lastTime | Most recent time that the event occurred. |
| manager | value of manager |
| message | a message communicated by the event |
| ntevid | windows event id |
| ownerid | ownerid |
| priority | syslog priority |
| prodState | production state of the device |
| severity | the severity of the event expressed as a number (0, 1, 2, 3, 4, or 5) |
| severityString | the severity of the event expressed as a string (Clear, Debug, Info, Warning, Error, or Critical) |
| stateChange | last time that the event status changed |
| status | the status of the event |
| summary | a short message summarizing the event |

Note Some of the values in the following table are direct duplicates of fields on evt . For example, `uuid -> evt.evid`.

Table 2: eventSummary/ and clearEventSummary/

| Value | Description |
|-----------------|----------------|
| uuid | evt.evid |
| occurrence | evt.count |
| status | evt.eventState |
| first_seen_time | evt.firstTime |

| Value | Description |
|--------------------------|---|
| status_change_time | evt.stateChange |
| last_seen_time | evt.lastTime |
| count | evt.count |
| current_user_uuid | UUID of the user who acknowledged this event |
| current_user_name | name of the user who acknowledged this event |
| cleared_by_event_uuid | UUID of the event that cleared this event (for events with status == CLEARED) |
| notes | event notes |
| audit_log | event audit log |
| update_time | last time a modification was made to the event |
| created_time | evt.lastTime |
| fingerprint | evt.dedupid |
| event_class | evt.eventClass |
| event_class_key | evt.eventClassKey |
| event_class_mapping_uuid | If this event was matched by one of the configured event class mappings, it contains the UUID of that mapping rule. |
| actor | event actor |
| summary | evt.summary |
| message | evt.message |
| severity | evt.severity |
| event_key | evt.eventKey |
| event_group | evt.eventGroup |
| agent | evt.agent |
| syslog_priority | evt.priority |
| syslog_facility | evt.facility |
| nt_event_code | evt.ntevid |
| monitor | evt.monitor |
| tags | event tags |

Table 3: urls/

| Value | Description |
|-----------|---------------------------------|
| ackUrl | URL for acknowledging the event |
| closeUrl | URL for closing the event |
| reopenUrl | URL for reopening the event |
| eventUrl | URL for viewing the event |

| Value | Description |
|-----------|---|
| eventsUrl | URL for viewing events for the relevant device, or all events |

ZenPacks can define additional notification actions and can extend the context that is available to notifications to add objects or attributes.

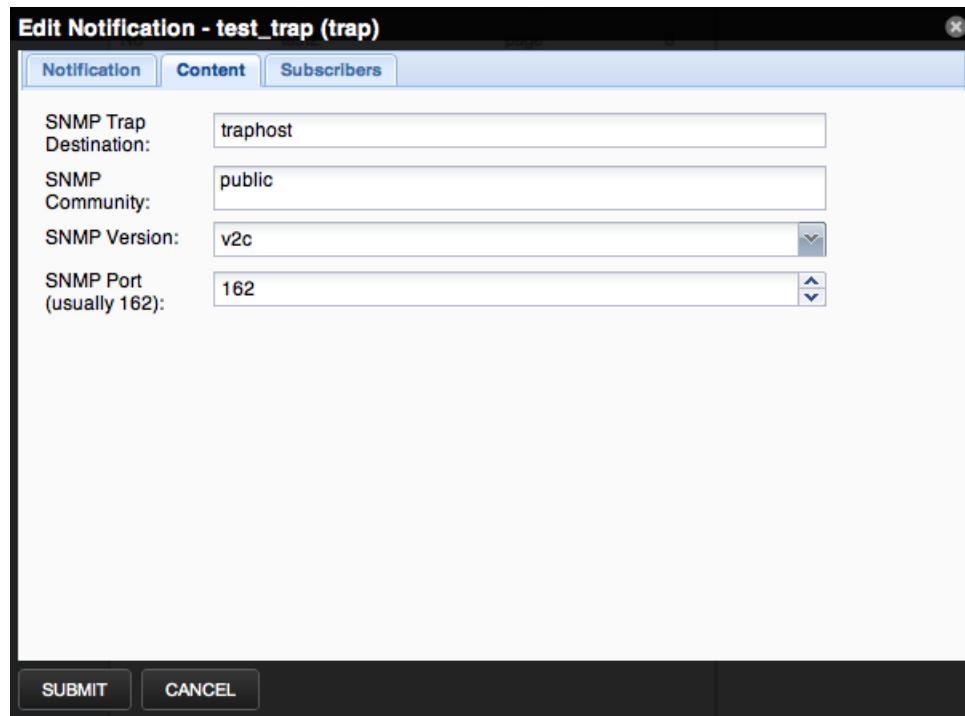
Defining the SNMP trap host

For SNMP trap-type notifications, on the **Edit Notification Content** tab, provide the following information:

- **SNMP Trap Destination** - Specify the host name or IP address to which the trap should be sent.
- **SNMP Community**- Specify the SNMP community. By default, this is public.
- **SNMP Version**- Select v2c (default) or v1.
- **SNMP Port**- Specify the SNMP port. Typically, this is 162.

SNMP traps sent as a result of this notification are defined in the ZENOSS-MIB file. You can find this MIB file on any Zenoss Core server at \$ZENHOME/share/mibs/site/ZENOSS-MIB.txt.

Figure 26: Edit Notification Content (SNMP trap)



Defining commands to run

For command-type notifications, specify the command to run when configured triggers are matched. On the **Edit Notification Content** tab, provide the following information:

For SNMP trap-type notifications,

- **Command Timeout** - By default, 60 seconds.
- **Command** - Command to run when a trigger is matched.
- **Clear Command** - Optional command to run when the triggering event clears.
- **Environment variables** - Optional field to define any environmental variables.

Figure 27: Edit Notification Content (command)

The screenshot shows a web-based configuration interface for editing a notification. At the top, it says "Edit Notification - command_notification (command)". Below that, there are three tabs: "Notification", "Content" (which is selected and highlighted in blue), and "Subscribers". The main area contains several input fields:

- "Command Timeout (seconds)": A dropdown menu set to "60".
- "Command": An empty text input field.
- "Clear Command": An empty text input field.
- "Environment variables": An empty text input field.

At the bottom of the form are two buttons: "SUBMIT" and "CANCEL".

Global notification permissions

By establishing permissions, you can control which users have the ability to view, manage, and update notifications. Permissions are granted based on the user's assigned role. The following table lists account roles and their associated notification permissions:

| Role | Permissions |
|---|---|
| Admin, Manager, ZenManager | Users assigned the Admin, Manager, or ZenManager roles can view, update, and manage any notification. |
| Notification owner | When a user creates a notification, that user is designated the owner of that notification. During the life of the notification, the owner can view, update, and manage it. |
| All other users (including those assigned ZenUser role) | Must be specifically granted permissions through the interface to view, edit, or manage notifications. |

You can set global permissions for viewing, updating and managing a notification. Global permissions are given to any user with "manage" permission, which includes:

- Admin, Manager, and ZenManager roles
- Notification owner

Edit global permissions from the Subscribers tab on the Edit Notification Subscription panel.

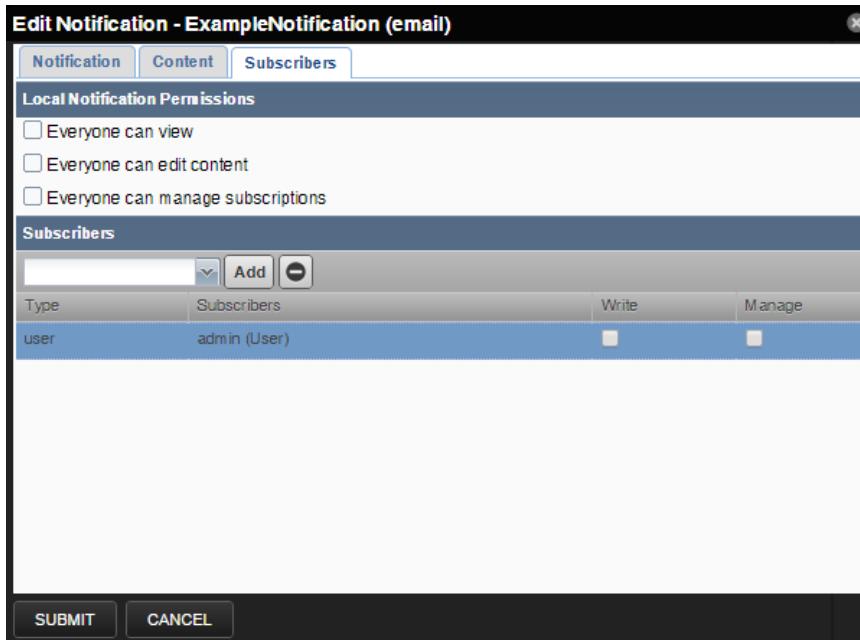
Global options are:

- **Everyone can view** - Provides global view permission.
- **Everyone can edit content** - Provides global update permission.
- **Everyone can manage subscriptions** - Provides global manage permission.

Permission checks occur when the data is sent to the browser and when any action occurs. To determine where a user can make modifications to a particular tab, permission checks are performed on global roles,

then managerial roles, and then individual roles. Any role that provides the required permission will allow that permission's associated behavior.

Figure 28: Edit Notification



Setting individual notification permissions

You can grant permissions to individual users or groups.

- **Write** - Select this option to grant the user or group permission to update the notification.
- **Manage** - Select this option to grant the user or group permission to manage the notification.

You can manually enter in the name of a user or group, or select one from the list of options.

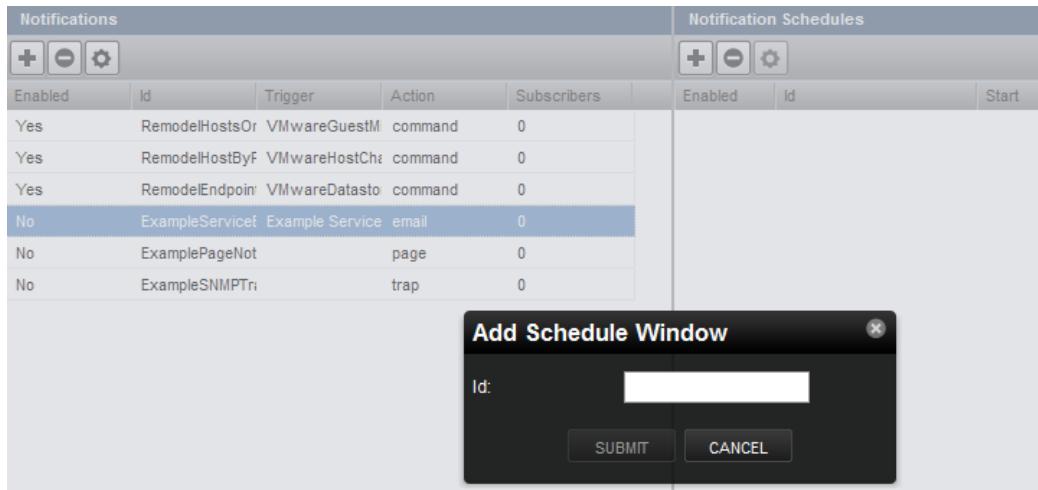
Adding notification schedules

You can establish one or more notification schedules for each defined notification. These notification schedules allow you to receive bulk email notifications at a specific time.

Note If a notification has a schedule, then notifications are sent only if a schedule is active when a notification arrives.

To add a schedule:

- 1 In the **Notifications** area on the left, select a notification.
- 2 In the **Notification Schedules** area on the right, click **Add**.

Figure 29: Add Schedule Window dialog box

- 3 In the **Add Schedule Window** dialog box, enter an identifier for the schedule, and then click **Submit**.
- 4 In the **Notification Schedules** area, double-click the new schedule.
- 5 In the **Edit Notification Schedule** dialog box, complete the following fields, and then click **Submit**.

| Field | Description |
|---------------------------|--|
| Enabled | Check the box to enable the schedule. |
| Start Date | The calendar date of the start of the schedule. |
| Repeat | The schedule frequency, one of the following values: <ul style="list-style-type: none"> ■ Never ■ Daily ■ Every Weekday ■ Weekly ■ Monthly ■ First Sunday of the Month |
| Duration (minutes) | The period of time during which the notification window is active. |

Advanced user interface configuration

To access advanced user interface configuration options, select **ADVANCED > Settings**, and then select **User Interface**. Configure options as follows to control how data loads, how much data is loaded, and filter and search options.

- **Enable Hyperlinks in Event Fields**- Click to enable this option to show hyperlinks in event fields. By default, this feature is disabled.
- **Enable Infinite Grids for Events**- Disable this option to turn off infinite grids for events. By default, this feature is enabled.
- **Enable Infinite Grids for Components**- Disable this option to turn off infinite grids for components. By default, this feature is enabled.
- **Enable Live Filters**- Disable this option to turn off the live filters feature. If you disable this feature, you will need to press enter on all search and filter inputs.
- **Enable Incremental Tree Loading on the Infrastructure Page**- Enable this option to load the infrastructure tree one node at a time. If disabled (the default), then the infrastructure tree is loaded all at

once. You might enable this option if you have a complex hierarchy of organizers and device classes and want to improve your UI load response time.

- **Show Tree Event Severity Icons-** Disabling this option may speed up page loading.
- **Load Infrastructure Page's Device Tree from Catalog-** Enable this option to load the device tree directly from the catalog.
- **Enable Tree Filters-** If disabled, then tree filters (the text input area that allows you to filter the information displayed) are hidden on all pages. This option is enabled by default.
- **Show Page Statistics-** Enable this option to show debugging information. By default, this option is not enabled.
- **Default Time Range-** Specify the default time range for all graphs. The default amount is **Last Hour**. Other possible values include Last 24 Hours, Last Week, Last 30 Days, or Last Year.
- **Number of Graph Columns-** The number of graph columns to display on graphs in the Device Overview page. Default value is **Auto**, which means that the number of columns will increase as the browser's width increases. Other possible values include setting the number of columns to 1, 2, or 3.

Note Component graphs are always one column regardless of this setting.

- **Device Grid Buffer Size-** Specify the number of device data rows to fetch from the server for each buffer request. The default buffer size is 100 rows.
- **Component Grid Buffer Size-** Specify the number of component data rows to fetch from the server for each buffer request. The default buffer size is 25 rows.
- **Event Console Buffer Size-** Specify the number of event rows to fetch from the server for each buffer request. The default buffer size is 200 rows.
- **Device Move Job Threshold-** Specify the limit at which devices are moved immediately. If the number of devices to be moved exceeds this threshold, then the move occurs in a job; otherwise, they are moved immediately. The default value is 5 devices.
- **Job Notification Refresh Interval-** Specify the refresh interval, in seconds, for the job notification dialog . The default time is 10 seconds.
- **Job Grid Buffer Size-** Specify the number of job data rows to fetch from the server for each buffer request. The default buffer size is 100 rows.

When complete, click **Save**.

Adding, discovering and modeling devices

Modeling is the process by which the system:

- Populates the device database
- Collects information about the devices in the system (such as operating system type or file system capacity)

The system models devices when they are added to the database, either manually or through the discovery process.

Add a single device

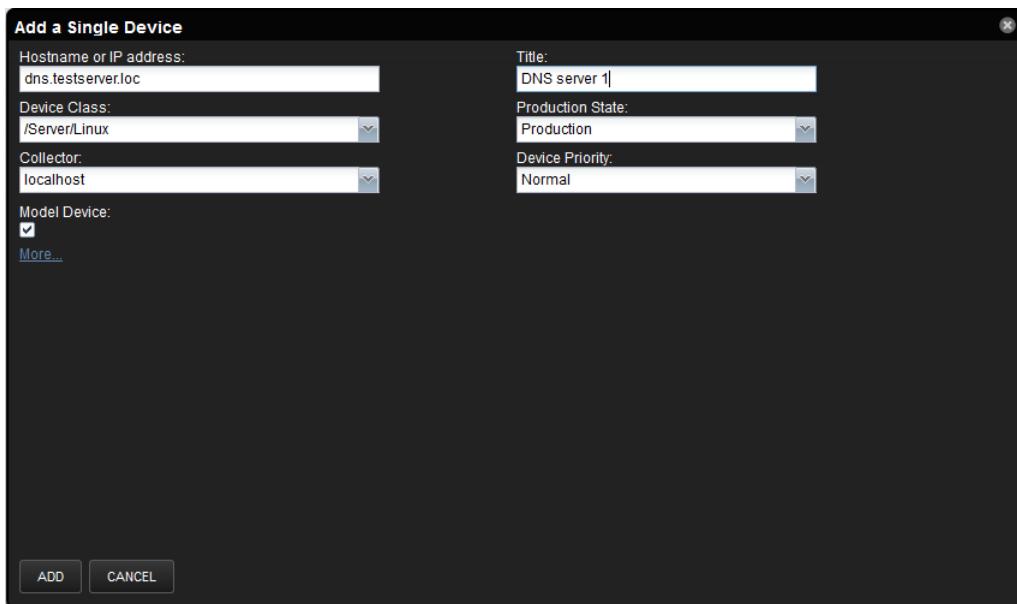
When you manually add a device, information that you provide might conflict with information that the system discovers about the device. Therefore, you can add a device by providing only the following required information.

- **Hostname or IP address** - Enter the network (DNS) name or IP address of the device.
- **Device Class** - Select a device class to which this device will belong. For example, if the new device is a Linux server, then select /Server/Linux.
- **Collector** - By default, this is localhost. Select a collector for the device.

An exception is if you are adding a Cisco router in a device class other than /Network. In this case, before adding the device, set the value of the zIfDescription configuration property to True. Changing the value gives you additional information about Cisco routers. By default, this option is set to True for the /Network class.

without additional information or selections.

- 1 From the navigation menu, select **INFRASTRUCTURE**. The **Devices** page appears.
- 2 Click the Add Devices icon  and choose **Add a Single Device**. The **Add a Single Device** dialog box appears.

Figure 30: Add a Single Device

- 3 Enter information or make selections to add the device.
By default, **Model** is selected. If you do not want the device to be modeled when it is added, de-select this option.
- 4 Optional: To display additional fields that are specific to the chosen device class, click **More**. For example, on the expanded page, you can
 - Enter device-specific details.
 - Edit SNMP settings.
 - Set hardware and operating system information.
 - Add device comments.
- 5 Click **ADD**.
- 6 Optional: To view the Add Device job in progress, click **View Job Log** in the notification that appears when you add the device.
When the job completes, the device is added in the selected device class.

Add multiple devices

Follow these steps to manually add multiple devices:

- 1 From the Navigation menu, select **Infrastructure**. The Devices page appears.
- 2 From the Add Devices icon , select **Add Multiple Devices**. The Add Infrastructure page appears.
- 3 Select the category, type, and connection information for each device you want to add.
- 4 Click **Add** to add the device to the system. You will see the status at the bottom of the page. Continue to add more devices until you are done.
- 5 When you have completed adding your devices, click **Done**.

Discovering devices

You can provide network or IP address range information so that the system can discover your devices.

Follow these steps to discover devices:

- 1 From the Navigation menu, select **Infrastructure**. The Devices page appears.
- 2 Click the Add Devices icon  and select **Discover Networks** from the drop-down list. The **Network Discovery** page appears.

Network Discovery

Devices found via Discovery will be placed in the /Discovered Device Class

| | | | |
|---|---|---|---|
| Networks/Range Enter one or more networks (such as 10.0.0.0/24) or IP ranges (such as 10.0.0.1-50): <input type="text"/> | SNMP Community Strings: <input type="text"/> <input type="text"/> | SSH Authentication Username: <input type="text"/> Password: <input type="text"/> | Windows Authentication Administrator Username: <input type="text"/> Password: <input type="text"/> |
| <input type="button" value="Discover"/> | | | |

- 3 For each network or IP range in which you want the system to discover devices, enter an address or range. For example, you might enter a network address in CIDR notation 192.0.2.0/24 or a range of IP addresses 192.0.2.1-50

Note Trying to add a /16 or /8 network can take a very long time, and may have unintended consequences.

- 4 For each network or IP range, specify the Windows, SSH, or SNMP credentials you want Zenoss Core to use on the devices it discovers. You can enter only one of each. Zenoss Core will attempt to use the same credentials on each device it discovers within the networks or IP ranges specified, but will not try to automatically classify the devices.
- 5 Click **Discover**. The discovery process iterates through every IP address in the networks and IP ranges you specify, adding each device that responds to a ping request. Further, the process adds information to any device that responds to an SNMP, WinRM, or SSH request.

Note Zenoss Core uses Advanced Encryption Standard (AES) with a 256-bit key size to encrypt all passwords, and stores them in the Zope object database.

The system places discovered routers in the device path `/Network/Router`. Devices are placed in the `/Discovered` device class.

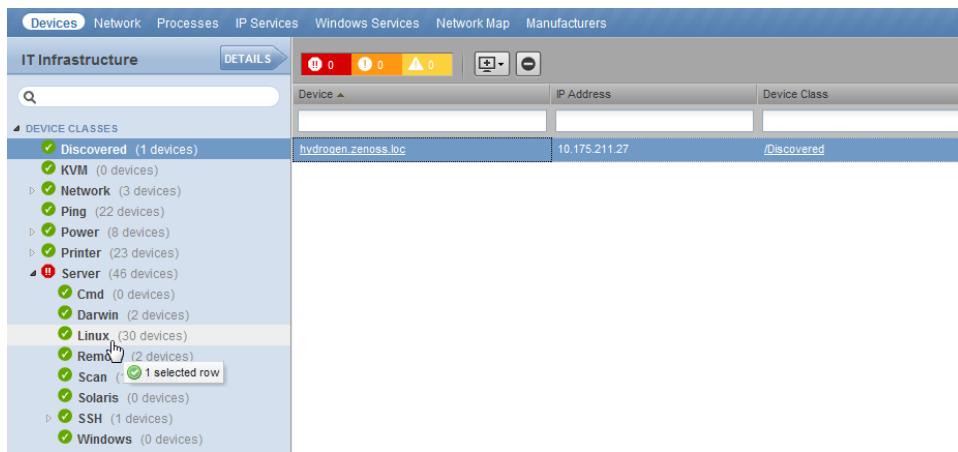
Classifying discovered devices

Once discovery is complete, you must move discovered devices (placed, by default, in the `/Discovered` class) to an appropriate device class in the hierarchy. Moving devices to their correct hierarchy location makes it possible for monitoring to begin.

Servers are organized by operating system. If the system discovers Windows devices, for example, you might choose to relocate them to `/Server/Windows`. Similarly, you might choose to classify discovered Linux devices in `/Server/Linux` (if you want to monitor and model using SNMP), or `/Server/SSH/Linux` (if you want to monitor and model using SSH).

To classify discovered devices:

- 1 Select one or more discovered devices (highlight one or more rows) in the device list.
- 2 Drag the selected devices to the new device class in the tree view.

Figure 31: Classifying discovered devices

The Move Devices dialog box appears.

- 3 Click **OK**. The list of devices refreshes, and the devices now appear in the newly selected class.

Updating device authentication details

For each device added to the database and set to its proper device class, Zenoss Core may require additional or different authentication information before it can gather device information and monitor the device.

For example, for a device in the /Server/Windows class, you must supply your Windows user name and password before the system can monitor the device. To do this:

- 1 Click a device name in the devices list. The Device summary page appears.
- 2 Select **Configuration Properties** from the left panel.
- 3 Double-click the zWinRMUser configuration property to display the Edit Config Property dialog.
- 4 Enter your Windows user name in the Value field, and then click **Submit**.
- 5 Double-click the zWinRMPassword configuration property to display the Edit Config Property dialog.
- 6 Enter your Windows password in the Value field, and then click **Submit**.

Similarly, for a device in the /Server/SSH/Linux class, you must supply your SSH user name and password. Set these values in the device's zCommandUsername and zCommandPassword configuration properties.

Note After making changes, you should remodel the device to ensure the authentication changes are valid.

Note Zenoss Core uses Advanced Encryption Standard (AES) with a 256-bit key size to encrypt all passwords, and stores them in the Zope object database.

Adding or editing information on a device record

To add or edit information:

- 1 Click a device name in the devices list. The Device overview page appears.
- 2 You can select values to change, or click the "edit" link adjacent to a label to edit that value. Enter or change information in one or more areas, and then click **Save** to save your changes.

Modeling devices

To model devices, the system can use:

- SSH
- WinRM
- SNMP (legacy option)

Note SSH and WinRM are the recommended options.

The modeling method you select depends on your environment, and on the types of devices you want to model and monitor.

By default the system remodels each known device every 720 minutes (12 hours).

Note You can change the frequency with which devices are remodeled. Edit the value of the Modeler Cycle Interval in the collector's configuration.

For larger deployments, modeling frequency may impact performance. In such environments, you should set the `startat` configuration setting inside the `zenmodeler.conf` file to change the scheduling of the daemon. The `startat` value only dictates the initial start time of `zenmodeler`. Each subsequent run interval is determined by the `zenmodeler` cycle time (number of minutes between runs) configured on the daemon settings page inside the parent's collector folder which can be accessed in Control Center. See the following KB article for more information: [How To Edit The Zenmodeler File To Configure Model Scheduling In Zenoss 5.x](#)

Configuring Windows devices to provide data through SNMP

To monitor Microsoft Windows Server 2008 R2 systems, Zenoss Core uses SNMP v1/v2 or WinRM. (There is no SNMP v3 support.) For Windows 2012, there is no SNMP support.

By default, Windows may not have SNMP installed. To install SNMP on your particular version of Windows, please refer to the Microsoft documentation.

After setting up and configuring the SNMP service, you must set the `zSnmpCommunity` string in Zenoss Core to match, to obtain SNMP data.

If you want processor and memory monitoring, install [SNMP-Informant](#) on the device.

To collect Windows event logs or log files from a Windows box using `syslog`, install the [SyslogAgent](#) Windows add-on.

Configuring Linux devices to provide data through SNMP

To configure a Linux machine for monitoring, it must have SNMP installed. A good Linux SNMP application is `net-snmp`. Download, install, and configure `net-snmp` to then use SNMP to monitor Linux devices.

Modeling devices using SSH/COMMAND

You can gather additional information by running commands on the remote device and interpreting the results. This provides a more scalable and flexible way to gather information that may not be available through any other means.

Each built-in modeling command plugin is differentiated by the platform on which it runs. To determine the platform for the device you want to model, run the `uname` command in a shell on the device.

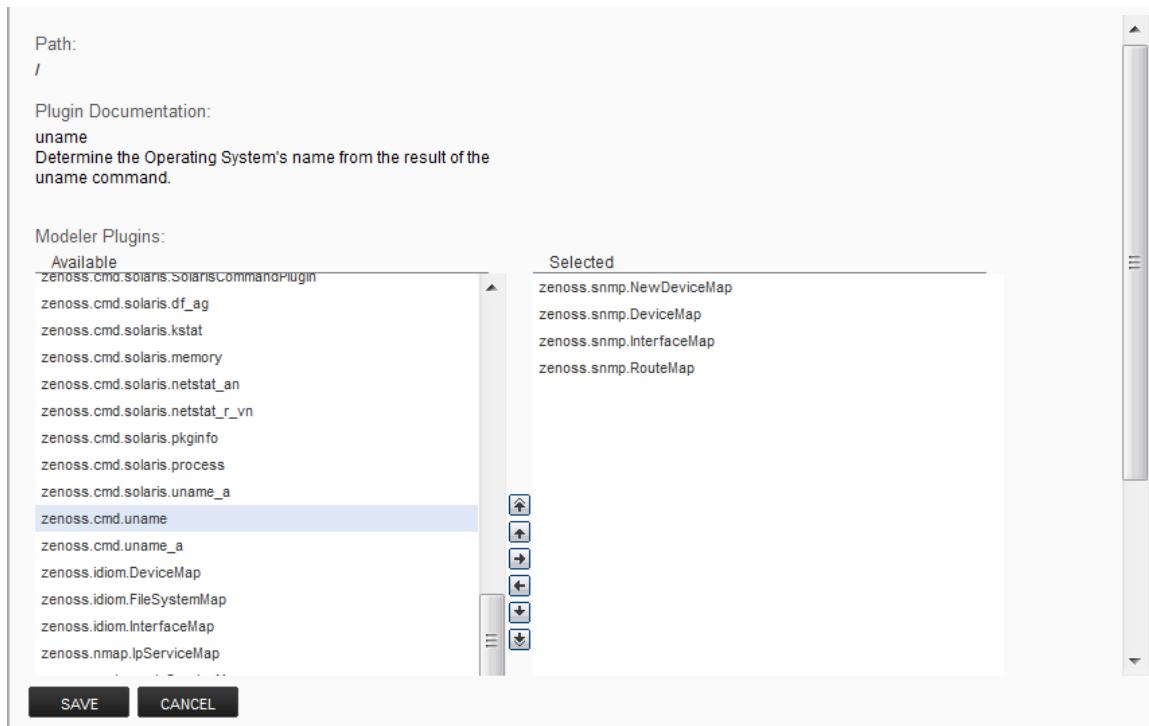
To model a device using command plugins, first add the device by using the protocol "none," and then choose the plugins you want to apply:

- 1 From the Navigation menu, select **Infrastructure**.
- 2 Click the Add Devices  icon and select **Add a Single Device** from the drop-down list. The Add a Single Device window appears.
- 3 Enter values for Name or IP and Device Class.
- 4 Clear the Model Device option.
- 5 Click **Add**.
- 6 After adding the device, select the device name in the devices list. The Device Overview page appears.
- 7 In the left panel, select **Configuration Properties**.
- 8 If necessary, set the values of the zCommandUsername and zCommandPassword configuration properties to the user name and password of the device (or set up authentication by using RSA/DSA keys.)

Note `~/ .ssh/id_rsa`

- 9 In the left panel, select **Modeler Plugins**. The list of plugins appears. The left column displays available plugins; the right column shows those currently selected.
- 10 Select `zenoss.cmd.uname` from the Available list, and then use the right arrow control to move it to the Selected list on the right. Use the controls to place it at the top of the list.

Figure 32: Add plugin



- 11 Use the left arrow control to move the other Selected plugins from the Selected list to the Available list.
- 12 Click **Save**.
- 13 Model the device by clicking the **Model Device** button.

Using device class to monitor devices using SSH

The /Server/Cmd device class is an example configuration for modeling and monitoring devices using SSH. The zCollectorPlugins have been modified (see the section titled "Modeling Using SSH/Command"), and the

device, file system, and Ethernet interface templates used to gather data over SSH have been created. You can use this device class as a reference for your own configuration; or, if you have a device that needs to be modeled or monitored via SSH/Command, you can place it in this device class to use the pre-configured templates and configuration properties. You also must set the `zCommandUsername` and `zCommandPassword` configuration properties to the appropriate SSH login information for each device.

Modeling devices using port scan

You can model IP services by doing a port scan, using the [Nmap Security Scanner](#). You must provide the full path to your system's nmap command.

To determine where nmap is installed, at the command line, enter:

```
which nmap
```

If your system returns a result similar to:

```
/usr/bin/which: no nmap in (/opt/zenoss/bin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/opt/zenoss/bin)
```

then nmap is not installed. Install it, and then try again.

After locating the nmap command (including the directory beginning with `/`), enter the following as the `zenoss` user on the Zenoss Core server:

```
cd $ZENHOME/libexec ln -s  
Full_Path_to_nmap
```

Note To execute a command using `$ZENHOME` (`/opt/zenoss` for the `zenoss` user), you must be attached to the container holding the Zenoss Core application. See the Control Center documentation for `serviced` commands.

To model a device using a port scan:

- 1 Select the device in the device list.
- 2 In the left panel, select **Modeler Plugins**.
- 3 Select the `zenoss.nmap.ipServiceMap` plugin in the list of Available plugins, and then use the right arrow control to move it to the list of Selected plugins.
- 4 Click **Save**.
- 5 Remodel the device by clicking the **Model Device** button.

Using the /Server/Scan device class to monitor with port scan

The /Server/Scan device class is an example configuration for modeling devices by using a port scan. You can use this device class as a reference for your own configuration; or, if you have a device that will use only a port scan, you can place it under this device class and remodel the device.

About modeler plugins

Zenoss Core uses plug-in maps to map real world information into the standard model. Input to the plug-ins can come from SNMP, SSH or Telnet. Selection of plug-ins to run against a device is done by matching the plug-in name against the `zCollectorPlugins` configuration property.

- **DeviceMap**— Collects basic information about a device, such as its OS type and hardware model.
- **InterfaceMap**— Collects the list of network interfaces on a device.
- **RouteMap**— Collects the network routing table from the device.
- **IpServicesMap**— Collects the IP services running on the device.
- **FileSystemMap**— Collects the list of file systems on a device.

Viewing and editing modeler plugins for a device

Plugins are controlled by regular expressions that match their names. To view a list of plugins for any device:

- 1 Click the device name in the devices list.
- 2 In the Device summary page, select **Modeler Plugins**.

The Modeler Plugins page appears.

Adding plugins

To add a plugin to a device:

- 1 Use the right arrow control to move one or more plugins from the Available list (on the left) to the Selected list (on the right).
- 2 Click **Save**.

Reordering plugins

Plugins run in the order in which they are listed. To re-order plugins, use the up and down arrow controls, and then click **Save**.

Deleting plugins from a device

To delete a plugin from a device, use the left arrow control to move the plugin from the Selected list to the Available list.

Debugging the modeling process

You can run the modeler from the command line against a single device. This feature is useful when debugging issues with a plugin.

By passing the `--collect` command to the modeler, you can control which modeler plugins are used. For example, the following command runs only the interface plugin against the `build.zenoss.loc` device:

- 1 Log in to the Control Center host as a user with `serviced` CLI privileges.
- 2 Attach to the `zenmodeler` service.

```
serviced service attach zenmodeler
```

- 3 Change to the `zenoss` user.

```
su - zenoss
```

- 4 Run the `zenmodeler` command.

```
$ zenmodeler run -v10 --collect=IpInterface -d build.zenoss.loc
```

If the command returns any stack traces, check the community forums for assistance.

Working with devices

This chapter provides information and procedures for managing devices in the system.

Viewing the device list

The device list shows all devices in the system. From this view, you can search for devices and perform a range of management tasks on all devices.

To access the device list, from the navigation menu, select **INFRASTRUCTURE**.

Devices hierarchy

Devices are organized in the tree view by:

- Devices
- Groups
- Systems
- Locations

Click the indicator next to each category name to expand it and see included devices.

Managing multiple devices from the device list

You can perform some management tasks for more than one device at a time. You can:

- Move devices to a different class
- Assign devices to groups, systems, and locations
- Remove devices
- Perform actions such as assigning priority, production state, or collector
- Lock devices

Working with devices

To view details for a single device, click its name in the device list. The device overview page appears.

Figure 33: Device overview

The screenshot shows the Zenoss Core interface for managing network devices. At the top, there's a navigation bar with links like DASHBOARD, EVENTS, INFRASTRUCTURE, REPORTS, ADVANCED, and a search bar. Below the navigation is a header for the device 'qa-centos-5.zenoss.lab' (IP 10.88.120.103). The main content area is divided into several sections:

- Overview:** A sidebar with links for Events, Graphs, Component Graphs, Modeler Plugins, Software, Custom Properties, Configuration Properties, Device Administration, Monitoring Templates, and a specific entry for 'Device (/Server/Linux)'.
- Device ID:** 10.88.120.103
- Connection Information:** Set to 'public'
- Uptime:** Unknown
- First Seen:** 2016-03-10 11:17:23 am
- Last Change:** 2016-03-10 11:17:36 am
- Model Time:** Not Modeled
- Locking:** Unlocked
- Memory/Swap:** Unknown/Unknown
- Collector:** localhost
- Production State:** Production
- Priority:** Normal
- Tag:** (empty)
- Serial Number:** (empty)
- Rack Slot:** (empty)
- Save** and **Cancel** buttons
- Systems:** None
- Groups:** None
- Location:** None
- Links:** (empty)
- Comments:** (empty)
- SNMP SysName:** (empty)
- SNMP Location:** (empty)
- SNMP Contact:** (empty)
- SNMP Description:** (empty)
- SNMP Community:** public
- SNMP Version:** v2c

Event status is shown in the "event rainbow" at the top of the page. Other key information that appears at the top of the device overview page includes:

- Device name
- IP address used to communicate with the device
- Device status (shows the current results of a ping test)
- Production state (Pre-Production, Production, Test, Maintenance, or Decommissioned)
- Priority

When you open the page, device overview information displays. This view provides classification and status information. From here, you can edit device information (indicated by text fields or edit links). Editable fields include:

- Device Name
- Production State
- Priority
- Tag
- Serial Number
- Rack Slot
- Collector
- Hardware and software manufacturer and model
- Systems
- Groups
- Location

The Links area displays links between the device and other external systems. Click **Show Links** to view the links.

The left panel of the device overview page allows you to access other device management views, such as:

- Events
- Components
- Graphs (Performance)
- Component Graphs
- Modeler Plugins
- Software
- Custom Properties
- Configuration Properties
- Device Administration
- Monitoring Templates

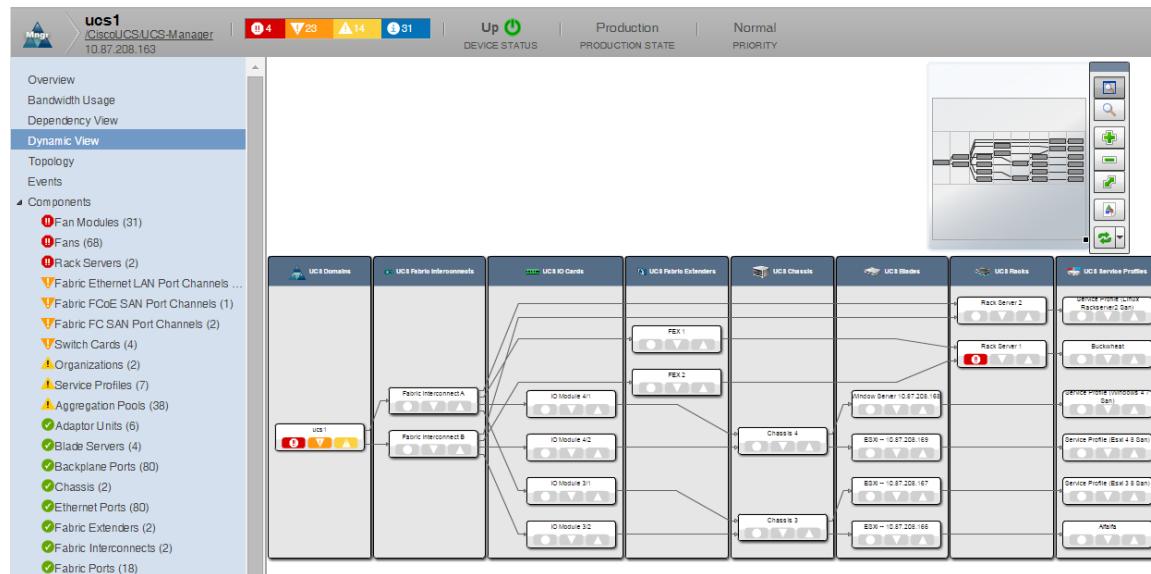
Information that appears here varies depending on device type.

Dynamic view

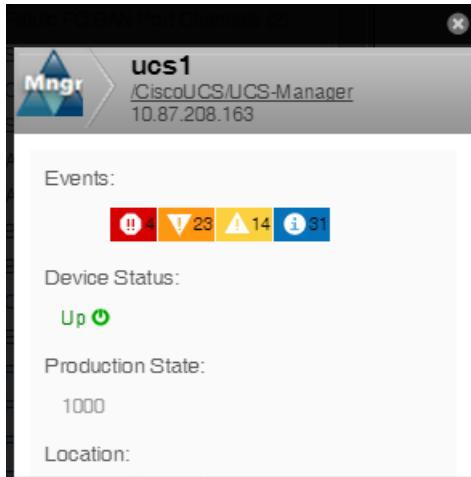
Zenoss Core provides a dynamic visualization of system objects and their relationships to other objects.

You can access a dynamic view from a device overview, a group, a system, or a location. Depending on the object type, different relationships are illustrated. Each dynamic view shows related objects in a graph. Each object in that graph displays its associated event information.

Figure 34: Dynamic view

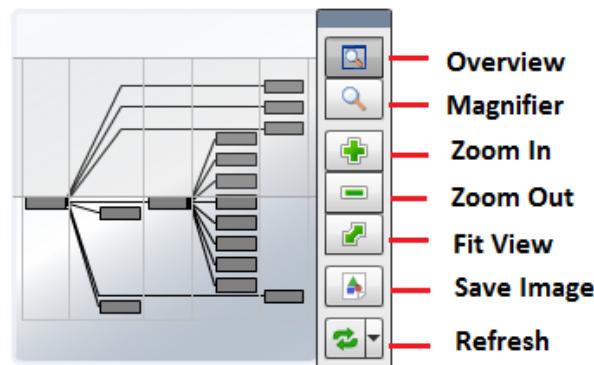


When you click an object in the graph, the "inspector" panel appears. This panel provides detailed information about the object and links directly to it. Information that appears in the inspector depends on the object type selected.

Figure 35: Dynamic view: Inspector panel

View controls appear to the right of the graph. These allow you to adjust your view:

- **Overview** - Toggles display on and off of the graph overview illustration.
- **Magnifier** - Toggles on and off the magnifier, which allows you to magnify selected portions of the graph.
- **Zoom In** - Zooms in on the graph.
- **Zoom Out** - Zooms out on the graph.
- **Fit View** - Fits the graph to the browser page.
- **Save Image** - Saves the dynamic view as a .png image.
- **Refresh** - Refreshes the graph.

Figure 36: Dynamic view controls

Events

Detailed information about events, scoped to the device, appears in the Events view. From here, you can:

- Sort event and event archive information by a range of categories
- Classify and acknowledge events
- Filter events by severity, state, or by one of several categories

Components

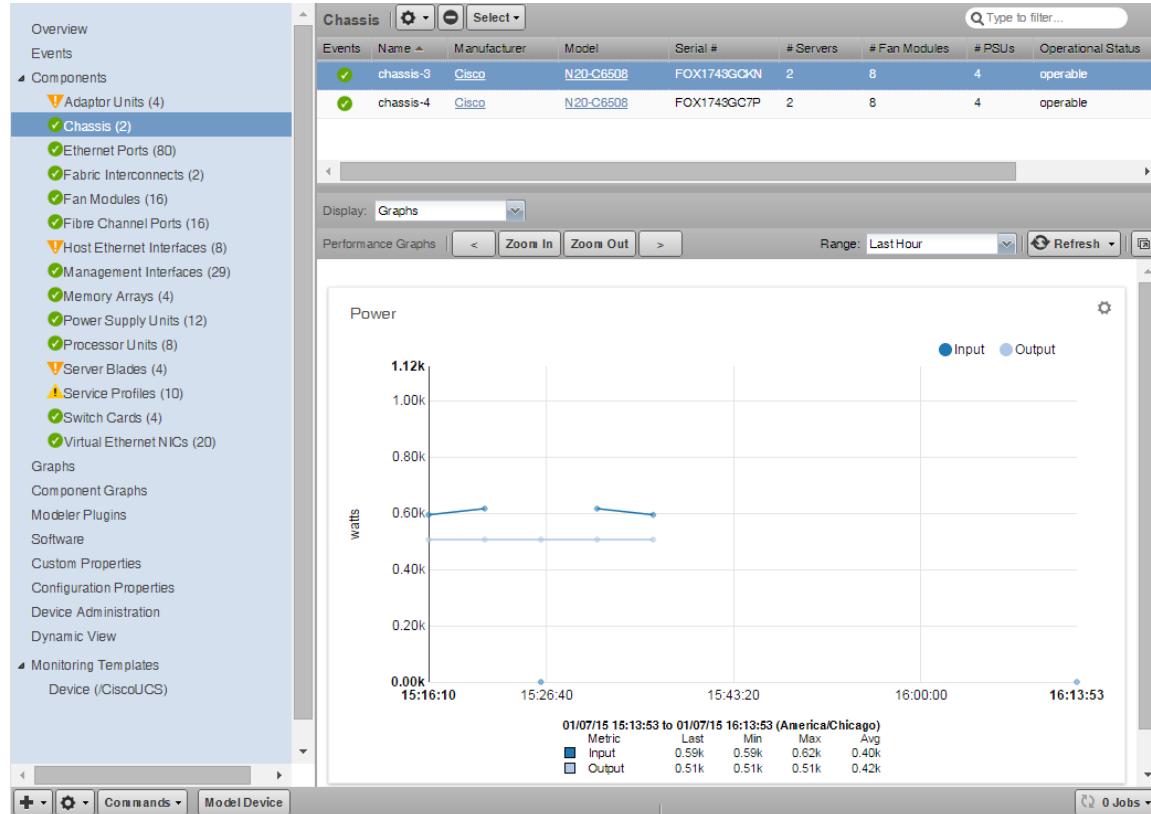
The Components view provides information about the different types of device components, including:

- IPService

- WinService
- IpRouteEntry
- IpInterface
- CPU
- FileSystem

To access components information, select Components in the left panel, and then select a component type. The components available will vary based on the type of device.

Figure 37: Device (Components)



The status of each device component type, as shown by the color of its indicator, is determined by the collective status of the monitored components of the same type. For example, if the IpService status is green, then all monitored IpServices on the device are functioning normally. If there is an event related to a monitored IpService, then the highest severity event associated with that component is displayed.

Note If there is an event unrelated to a known component, then the system places it in the component type Other.

From this view, you can:

- Lock components
- Turn on or off component monitoring
- Delete components

Disabling component monitoring

There may be occasions when you want to stop monitoring certain components of your monitored resources.

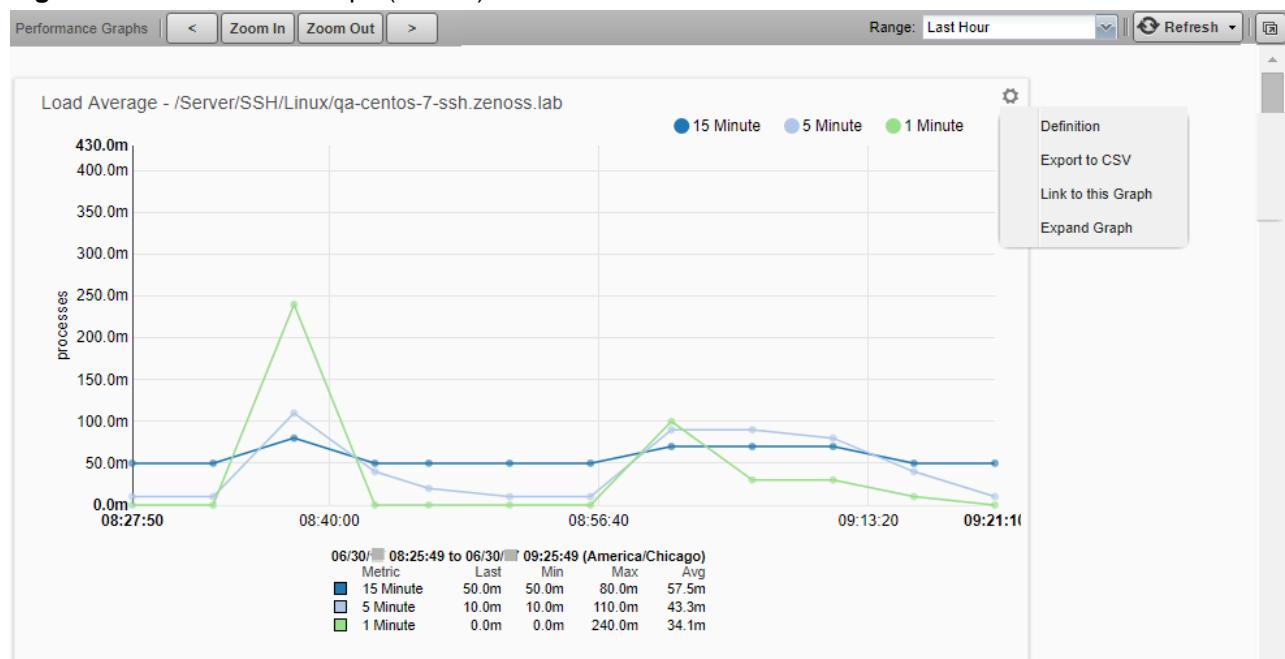
To disable monitoring on one or more components:

- 1 On the Device overview page, select the component group.
 - 2 In the Component list, select the components for which you want to disable monitoring.
 - 3 Click **Action > Monitoring**.
 - 4 Click **NO** to disable monitoring.
- You may want to clean up the Event log of any events that were created by these components prior to the disabling of monitoring, see [Closing events](#) on page 122.

Graphs (Performance)

The Graphs view shows performance graphs that are defined for the device or component. To access graphs, in the left panel, select **Graphs**.

Figure 38: Performance Graph (Device)



You can control the following performance graph options:

- **Time range controls** - To narrow or expand the size of the time range, click **Zoom In/Zoom Out**. To scroll through time on the graph, click the forward and back arrowheads. Clicking these controls automatically puts you into a custom time range. You can also Zoom In and center around a graph point by clicking the point inside the graph. This will center the graph around the selected point. If there are other graphs displayed in a view, they will also be affected by any change to the range.
- **Range** - Select the span of time that the graph displays, as follows:
 - Last Hour
 - Last 24 Hours
 - Last Week
 - Last 30 days
 - Last Year
 - Custom - Select the Start and End time to display. To set the end time to the current time, check **Now**. After changing a custom range setting, click **Refresh** to update the graph.
- **Refresh** - To modify the refresh value (by default, 30 minutes), click the drop-down list. If you set the refresh rate to manual, click **Refresh** each time you want an updated graph.

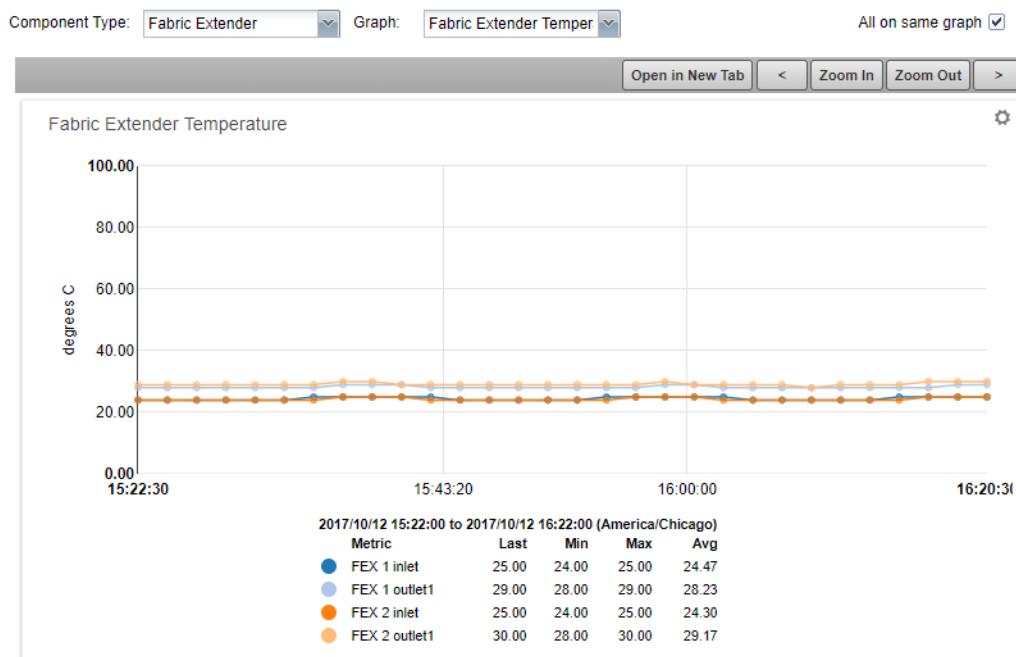
- **Pop-out** - To render the current graphs in full-screen mode, click the icon in the upper right corner of the page.
- **Action (gear)** - To open a submenu of the following actions, click the icon in the upper right corner of the graph.
 - **Definition** - View the JSON definition.
 - **Export to CSV** - Export the datapoints as a .csv file for use in a spreadsheet. Only data contained in the defined range will be included.
 - **Link to this Graph** - Generate a link to this graph to save in browser bookmarks or use the URL to directly point to the graph in another Web page or dashboard. For example, to show the graph in the Dashboard, create a **Site Window** portlet and insert the URL to the graph.
 - **Expand graph** - Render the current graph in full-screen mode.
- **Table Legend** - To highlight a particular data set, hover the pointer over a legend description. To toggle the displayed legend description, click it. A solid dot indicates that data will be displayed. A hollow dot indicates data will be hidden.

For more information about performance monitoring and performance graphs, see [Performance monitoring](#) on page 98.

Component graphs

The component graphs view shows component graphs defined for the device. To access these graphs, select **Component Graphs** in the left panel. The following figure shows a fabric extender temperature graph that has all metrics displayed on the same graph.

Figure 39: Component graph (Device)



Note To narrow or expand the size of the time range, click **Zoom In/Zoom Out**. To scroll through time on the graph, click the forward and back arrowheads. Clicking these controls automatically puts you into a custom time range. You can also Zoom In and center around a graph point by clicking the point inside the graph. This will center the graph around the selected point. If there are other graphs displayed in a view, they will also be affected by any change to the range.

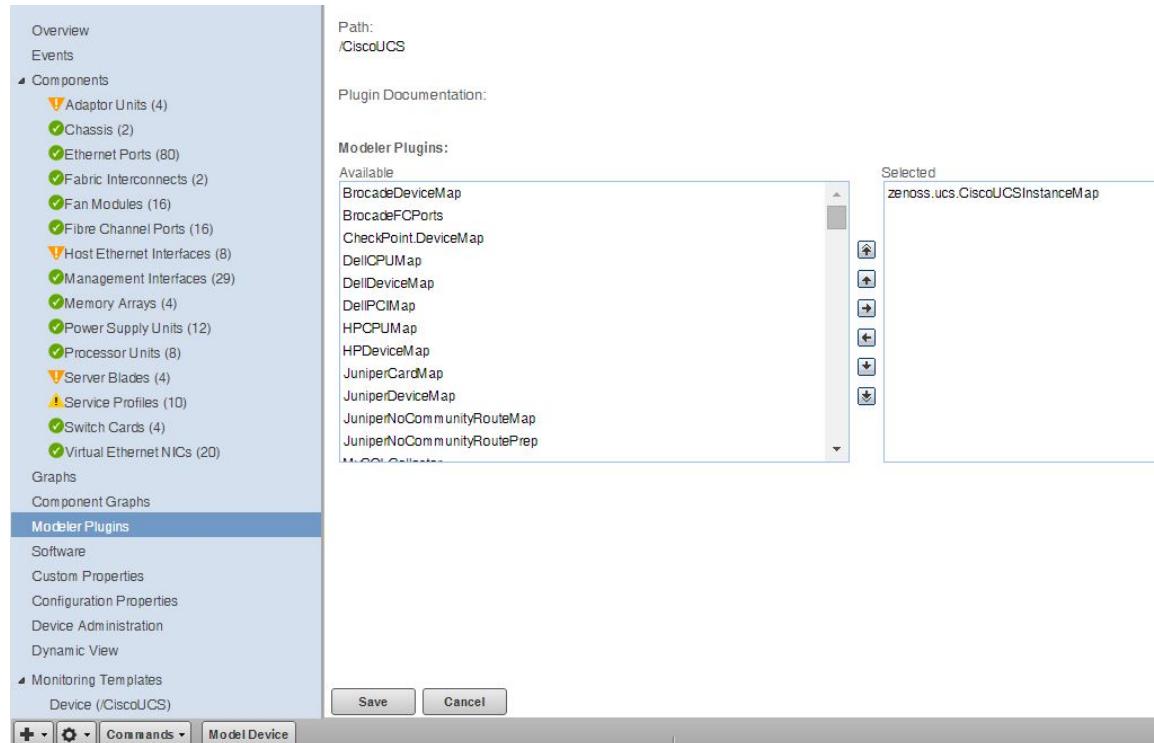
You can control these component graph options:

- **Component Type** - Drop-down of available components based on the type of device being monitored.
- **Graph** - Drop-down of available graphs based on the selected component type.
- **All on Same Graph?** - Select this check box to display all the metrics on one graph. Clear this check box to have a separate graph for each metric.

Modeler plugins

Use the Modeler Plugins view to manage plugins that are run against a device. To access plugins, select Modeler Plugins in the left panel.

Figure 40: Device (modeler plugins)



Software

The Software view lists software installed on the device. The details provided in this area depend on the method used to model the device.

Listed software links into the system's inventory of software in your IT infrastructure. You can view this inventory from the Manufacturers link on the sub-navigation menu.

To access software information, select Software in the tree view.

Custom properties

Use the Custom Properties view to edit the values of custom properties associated with a device.

To display the Custom Properties view, click on a device name in the device list, and then select **Custom Properties** in the left panel. You can perform the following actions on custom properties:

- Add
- Edit

- Refresh
- Delete

Figure 41: Device (custom properties)

| Property Name | Description | Value | Path | Type | Is Local |
|---------------|-------------|--------------------------------|------|------|----------|
| cDateTest | | 1900/01/01 00:00:00 US/Central | / | date | |

Note The Custom Properties view allows you to edit the value of a custom property on an individual device, but not to define new custom properties for device classes. For more information about custom properties, refer to knowledge base article "[How To Add Custom Properties](#)" on the [Zenoss Support](#) site.

Configuration properties

From the **Configuration Properties** view, you can set certain configuration properties for a device, and delete local properties from a device.

To access configuration properties, in the left pane, choose **Configuration Properties**.

Figure 42: Device (configuration properties)

| Delete Local Copy | | | | |
|--|-----------------------------|-------------------|-------|------|
| Is Local | Category | Name | Value | Path |
| Cisco | zCiscoACEUseSSL | true | | / |
| Cisco | zCiscoRemoteEventClassKeys | | | / |
| Cisco UCS | zCiscoUCSCIMCEventsInterval | 60 | | / |
| Cisco UCS | zCiscoUCSCIMCPerInterval | 300 | | / |
| Cisco UCS | zCiscoUCSManagerPort | 443 | | / |
| Cisco UCS | zCiscoUCSManagerUseSSL | true | | / |
| Cisco UCS | zCiscoUCSManagerUser | admin | | / |
| Modeler Controls | zCollectorClientTimeout | 180 | | / |
| Modeler Controls | zCollectorDecoding | utf-8 | | / |
| Misc | zCollectorLogChanges | false | | / |
| zencommand | zCommandCommandTimeout | 15 | | / |
| zencommand | zCommandExistanceTest | test-f %s | | / |
| zencommand | zCommandLoginTimeout | 10 | | / |
| zencommand | zCommandLoginTries | 1 | | / |
| zencommand | zCommandPassword | | | / |
| zencommand | zCommandPath | \$ZENHOME/libexec | | / |
| zencommand | zCommandPort | 22 | | / |
| zencommand | zCommandProtocol | ssh | | / |
| zencommand | zCommandSearchPath | | | / |
| zencommand | zCommandUsername | | | / |
| Control Center | zControlCenterHost | \$(here/managelp) | | / |
| Control Center | zControlCenterModeCycle | 3600 | | / |
| Control Center | zControlCenterPassword | | | / |
| Control Center | zControlCenterPortFirst | 22 | | / |

For information about working with configuration properties, see [Configuration properties](#) on page 60.

Device administration

Use the Device Administration view to:

- Add, delete, and run custom user commands

- Manage maintenance windows
- Determine who holds administration capabilities for the device, and their roles

To access administration options, select **Device Administration** in the left panel.

Figure 43: Device administration

The screenshot shows the Zenoss interface with the following details:

- Maintenance Windows:** A table showing three entries:

| Enabled | Name | Start | Duration | Repeat | State |
|---------|----------------------|---------------------|--------------|-----------------------|-------------|
| No | 1st of Month | 2014/12/01 02:00:00 | 01:00:00 hrs | Monthly: day of month | Maintenance |
| Yes | Every Thursday Night | 2014/12/11 22:00:00 | 30:00 mins | Weekly | Maintenance |
| Yes | One Time Testing | 2014/12/20 08:00:00 | 04:00:00 hrs | Never | Test |
- User Commands:** A table showing five entries:

| Name | Command |
|-------------|---|
| DNS forward | host \${device/id} |
| DNS reverse | host \${device/manageIp} |
| ping | \${device/pingCommand} -c2 \${device/mana... |
| snmpwalk | snmpwalk -S\${device/z\$SnmpVer} -c\${device/z... |
| traceroute | \${device/tracerouteCommand} -q 1 -w 2 \${d... |
- Administrators:** A table showing one entry:

| Name | Role | Email | Pager |
|------------|---------|---------|-------|
| cgilchrist | Manager | cgil... | |

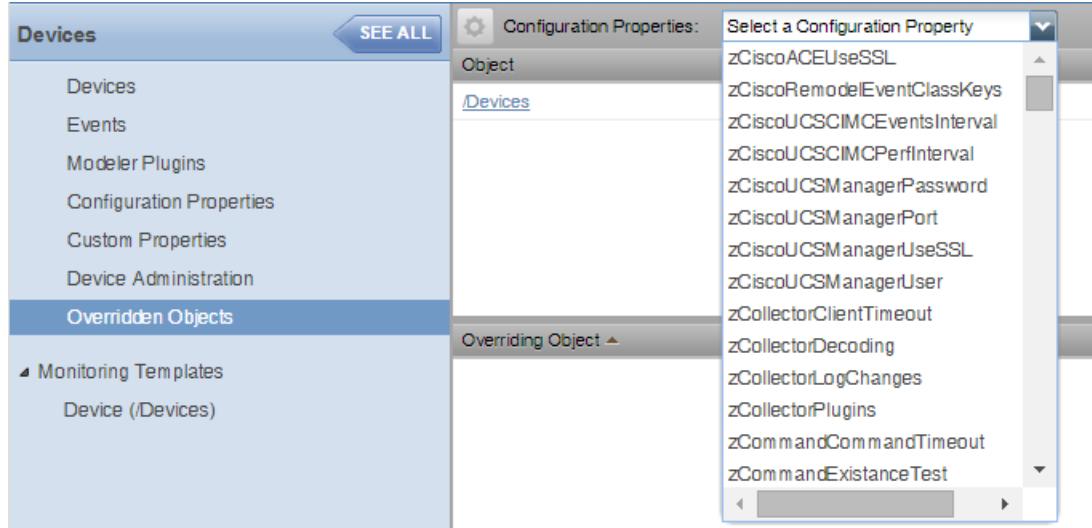
For more information about device administration tasks, see the following topics:

- [Maintenance windows](#) on page 137
- [Defining user commands for a single device](#) on page 149
- [Defining user commands for all devices in an organizer](#) on page 150
- [Adding administrators](#) on page 154

Overridden objects

Use the Overridden Objects view to see the objects that have overrides on their configuration properties. This view is available when looking at details of all devices.

To display the Overridden Objects view, navigate to the **Infrastructure > Devices** pages and click **Details**. Then, click **Overridden Objects** from the left-column menu.



Select a configuration property from the drop-down list to view the overridden objects for that property. Double-click the row of the overriding object to open an edit dialog box.

Note Do not click the link of the overriding object. You will be taken to that object's page in the infrastructure view. Instead, double-click the clear area of the row of the overriding object to view the Edit Configuration Property dialog.

Monitoring templates

Monitoring templates determine how the system collects performance data for devices and device components.

To access monitoring templates, expand Monitoring Templates in the left panel, and then select Device. The page shows all of the monitoring templates that are bound by name to this device.

Figure 44: Device (monitoring templates)

| Data Sources | | | | Thresholds | |
|------------------------------|------------------|---------------------------|---------|------------|--------------|
| Data Points by Data Source ▾ | | Source | Enabled | Type | Name |
| ▷ | IaLoadInt1 | 1.3.6.1.4.1.2021.10.1.5.1 | true | SNMP | high load |
| ▷ | IaLoadInt15 | 1.3.6.1.4.1.2021.10.1.5.3 | true | SNMP | low CPU idle |
| ▷ | IaLoadInt5 | 1.3.6.1.4.1.2021.10.1.5.2 | true | SNMP | |
| ▷ | memAvailReal | 1.3.6.1.4.1.2021.4.6.0 | true | SNMP | |
| ▷ | memAvailSwap | 1.3.6.1.4.1.2021.4.4.0 | true | SNMP | |
| ▷ | memBuffer | 1.3.6.1.4.1.2021.4.14.0 | true | SNMP | |
| ▷ | memCached | 1.3.6.1.4.1.2021.4.15.0 | true | SNMP | |
| ▷ | ssCpuIdle | 1.3.6.1.4.1.2021.11.11.0 | true | SNMP | |
| ▷ | ssCpuRawWait | 1.3.6.1.4.1.2021.11.54.0 | true | SNMP | |
| ▷ | ssCpuSystem | 1.3.6.1.4.1.2021.11.10.0 | true | SNMP | |
| ▷ | ssCpuUser | 1.3.6.1.4.1.2021.11.9.0 | true | SNMP | |
| ▷ | sslIORawReceived | 1.3.6.1.4.1.2021.11.58.0 | true | SNMP | |
| ▷ | sslIORawSent | 1.3.6.1.4.1.2021.11.57.0 | true | SNMP | |
| ▷ | sysUpTime | 1.3.6.1.2.1.25.1.1.0 | true | SNMP | |

| Graph Definitions | |
|-------------------|--------------------|
| + | - |
| Name | Load Average |
| | CPU Utilization |
| | Memory Utilization |
| | IO |

For detailed information about monitoring templates, see [Performance monitoring](#) on page 98.

Managing devices and device attributes

Read the information and procedures in this section to learn about specific device management tasks, including:

- Clearing heartbeat events
- Pushing configuration changes to the system
- Locking device configuration
- Renaming devices
- Remodeling devices
- Setting the device manage IP address

Clearing heartbeat events

If you have devices configured to send a recurring event that is mapped to a heartbeat class, you can clear stale heartbeat events.

To clear the heartbeat events associated with a device:

- 1 Navigate to **ADVANCED > Settings**.
- 2 In the left panel, select **EVENTS**.
- 3 At the bottom of the Event Configuration page, click the **Clear** button in the Clear Event Heartbeats section. The system displays a brief message banner.

Locking device configuration

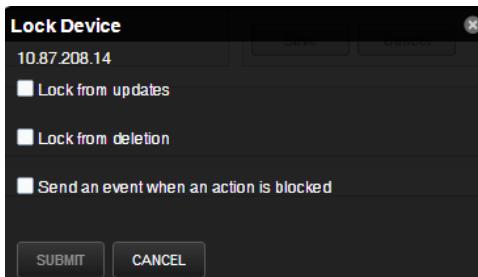
You can lock a device's configuration to prevent changes from being overwritten when remodeling the device. Two levels of locking are available. You can lock the configuration from deletion and updates, or solely from deletion.

Note Device locking prevents changes and deletion due to remodeling. It does not prevent manual changes and deletion.

To edit lock selections for a device configuration:

- 1 Navigate to the device in the device list.
- 2 At the bottom of the device overview page, select **Locking** from the Action menu. The Lock Device dialog box appears.

Figure 45: Lock Device Dialog



- 3 Select the type of lock you want to implement or remove.
- 4 To send events when actions are blocked by a lock action, select the "Send an event..." option. The lock or unlock action is implemented on the device, and the system displays a confirmation message of the action.

Renaming a device

Because the system uses the manage IP to monitor a device, the device name may be different than its fully qualified domain name (FQDN). The device name must always be unique in the system.

To rename a device:

- 1 Navigate to the device in the device list. Click the device name.
- 2 On the device overview page, edit the Device Name field with the new device name.
- 3 Click **Save**. The system renames the device and displays a confirmation message of the action.

Re-identifying a device

Changing the device ID in the system is different from changing the device name. If you change the ID, you must associate existing performance data with the new device ID; otherwise, you lose the data for this device. However, the re-association process takes time, and during processing, metrics are not collected or graphed for this device.

You have the option of deleting existing performance data and starting fresh with the new device ID. Collection of new performance data begins immediately.

- 1 Navigate to the device in the device list and click the device name.
- 2 At the bottom of the device overview page, click the **Action** menu and choose **Reidentify Device**.
- 3 In the **Reidentify Device** dialog box, enter a new ID for the device.
- 4 Choose whether to re-associate existing performance data for the device or delete it.
- 5 Click **SUBMIT**.
On successful completion of the job, collection resumes automatically and no further action is required.
- 6 Only if the job fails, manually resume data collection for the device as follows:
 - a Log in to the Zenoss Core user interface as a user with ZenManager privileges.
- 7 At the bottom of the device overview page, click the **Action** menu and choose **Resume Collection**.

Remodeling a device

Remodeling forces the system to re-collect all configuration information associated with a device. Normally, the system models devices every 720 minutes; however, if you want to remodel a device immediately, follow these steps:

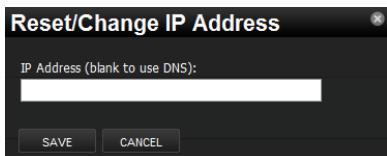
- 1 Navigate to the device in the device list and click on the Device name.
- 2 At the bottom of the Device Overview page, click the **Model Device** button. The system remodels the device. A dialog box appears that shows progress of the action.

Resetting the device manage IP address

You might reset the manage IP address if the IP address of a device changed and you want to maintain the historical data at the original IP address. To reset the manage IP address of a device:

- 1 Navigate to the device in the device list.
- 2 At the bottom of the device overview page, select Reset/Change IP Address from the Action menu. The Reset IP dialog box appears.

Figure 46: Reset IP dialog box



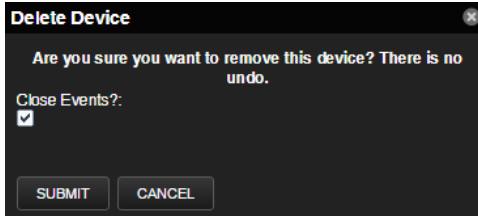
- 3 Enter the new IP address for the device, or leave the field blank to allow the IP address to be set by DNS.
- 4 Click **Save**. The IP address for the device is reset.

Deleting a device

To delete a device from the system:

- 1 Navigate to the **INFRASTRUCTURE** page.
- 2 Select the device you want to remove from the system by clicking on its row. You can select multiple devices by Ctrl-clicking or Shift-clicking the devices. Be sure to click on the row in an area that is not defined by a link. The Delete Device dialog appears.

Figure 47: Delete Device



- 3 Optional: Change the selection to close current events for the device. By default, event data is removed.
- 4 Click **Submit**. The system removes the devices and associated data (if selected), and displays a confirmation message of the action.

Exporting device list to load into another system

Use the `zenbatchdump` command to write the names of your devices, their device classes, locations, groups, and systems to a text file. You can then use the `zenbatchload` command to import the devices into another system instance.

The following command options might be helpful. For information about all options, run the following command:

```
zenbatchdump -- help
```

- `--root=ROOT` - The default path is `/Devices`. Use this option to set the root device path to dump. For example,

```
/Devices/Servers  
/Devices/Network/Cisco/Nexus
```

- `--prune` - Specifies whether device classes should only be dumped if they are part of the root path.
- `--noorganizers` - Specifies whether organizers, such as device classes and groups, should be dumped.

- 1 Log in to the Control Center host as a user with `serviced` CLI privileges.
- 2 Start a shell in a Zope service container.

```
serviced service shell zope
```

This command mounts `/mnt/pwd` so you can use files that exist on the host.

- 3 Change to the `zenoss` user.

```
su - zenoss
```

- 4 Export the device list to the text file.

```
zenbatchdump -o mydevicelist.txt
```

This command writes the names of your devices, their device classes, locations, groups, and systems to a file named `mydevicelist.txt`.

- 5 If you plan to use the text file in a batch load on another system, copy the file to that host.
- 6 Exit from the `zenoss` user shell in the container.

```
exit
```

- 7 Exist from the Zope container shell.

```
exit
```

Batch loading or modifying devices

Use the `zenbatchload` command to load devices into the system or modify device properties. The utility creates new device classes automatically, and by default, models each device.

Before you can load or modify devices, you need a text file to use as input. You can use a file created by the batch dump utility (`zenbatchdump`), or create a text file that specifies the following information for each device:

- Zenoss Core device class where the device should be loaded, for example, Linux SSH or SNMP, Windows WMI or WinRM, Network, Storage.
- Device name (resolvable hostname for the device or IP address and descriptive name).
- Optional: Provide additional data for device classes or individual devices; for example, login credentials, specific IP address if DNS resolution is not available, and the collector on which to load the device.

Configuration file syntax is as follows.

```
/Devices/Destination/Device/Class
Hostname | IP-Address [option='value' [, option='value'] ...]
Hostname | IP-Address [option='value' [, option='value'] ...]
```

The following command options might be helpful. For information about all options or a sample configuration file, run the following commands:

```
zenbatchload -- help
```

```
zenbatchload -- sample_configs
```

- `--nocommit` - Perform a test run of the batch load process; do not commit changes to the ZODB. If necessary, edit your configuration file.
- `--nomodel` - Do not model the remote devices. You must be able to commit changes to the ZODB.

The following example shows SNMP, SSH, and Windows monitored devices:

```
/Devices/Server zSnmpCommunity='underwriting'
/Devices/Server/Linux
linuxsnmp.hypothetical.loc

/Devices/Server/Microsoft/Windows zWinKDC='winkdc.hypothetical.loc',
zWinRMPassword='Zenny456', zWinRMUser='zenmonitor@hypothetical.loc'
windows.hypothetical.loc
```

```
/Devices/Server/SSH/Linux zCommandPassword='Zenny123',  
zCommandUsername='zenmonitor'  
linuxssh.hypothetical.loc  
  
/Devices/Server/Linux/WordPress zDeviceTemplates=['Device', 'Apache']  
wordpress.hypothetical.loc
```

For additional ways to add and discover devices, see [Adding, discovering and modeling devices](#) on page 36.

- 1 Log in to the Control Center host as a user with serviced CLI privileges.
- 2 Start a shell in a Zope service container.

```
serviced service shell zope
```

This command mounts /mnt/pwd so you can use files that exist on the host.

- 3 Change to the zenoss user.

```
su - zenoss
```

- 4 Load devices and components that are listed in a text file into Zenoss Core.

```
zenbatchload mydevicelist.txt
```

- 5 Exit from the zenoss user shell in the container.

```
exit
```

- 6 Exist from the Zope container shell.

```
exit
```

Configuration properties

Configuration properties are individual values that you can set up on major system entities.

- **Devices** *Device configuration properties* control the way that devices are monitored.
- **Events** *Event configuration properties* control the rules that process events as they are received by the system.
- **Networks** *Network configuration properties* control options that are used when you perform network discovery.

To customize the system when you add ZenPacks, you can add configuration properties and values to your ZenPacks.

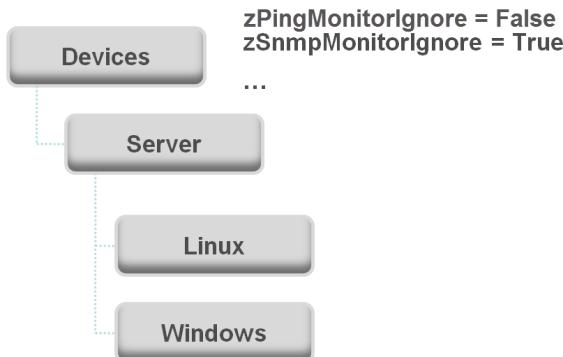
Configuration property types

Configuration properties are one of the following types:

- **String** - Text value that can be ASCII or Latin-1 encoded
- **Integer** - Whole number
- **Float** - Number that can have a decimal value
- **Boolean** - True or False
- **Lines** - List of values separated by a return. The system stores these as an array.
- **Password** - Character-masked password value

Configuration properties inheritance and override

The following diagram illustrates a portion of the standard device class hierarchy. A *device class* is a type of organizer that manages how the system models and monitors devices. At the root of the device hierarchy is the **Devices** object, under which all device class configuration properties are defined. Property values at the root level provide the default values for the entire hierarchy.

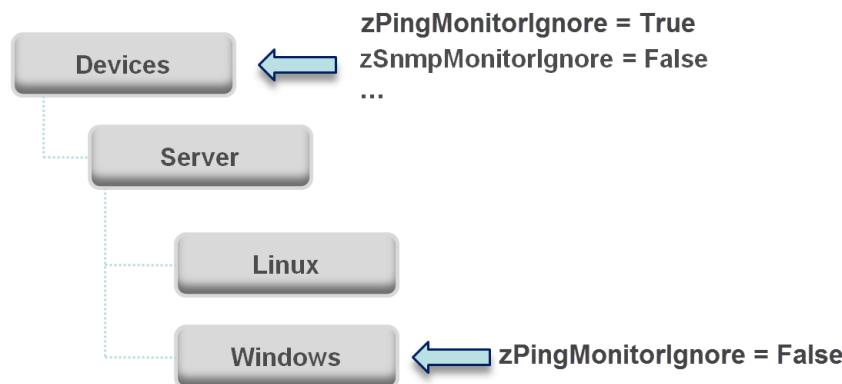
Figure 48: Device class hierarchy

The illustration shows the following defined configuration properties:

- **zPingMonitorIgnore**- Turns off all daemons that use ping. By default, its value at the root of the hierarchy is False.
- **zSNMPMonitorIgnore** - Turns off all daemons that use SNMP. By default, its value at the root of the hierarchy is True.

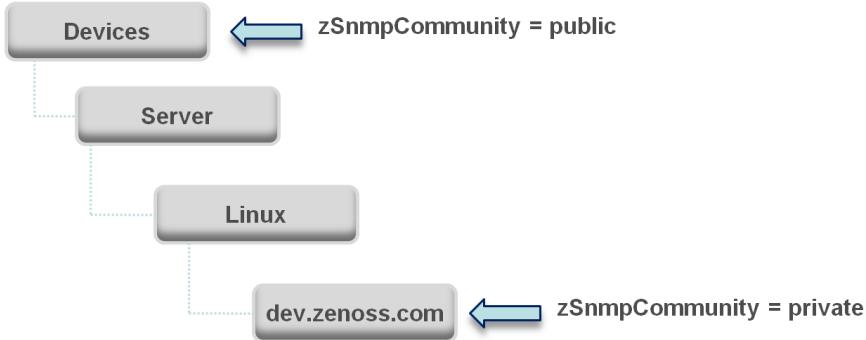
Through *inheritance*, properties that are defined at the root of the hierarchy apply to all objects beneath that node. So, at the /Devices/Server/Linux level of the device class hierarchy, the value of these two properties is the same as at /Devices, even though the property is not set explicitly at /Devices/Server/Linux. Inheritance simplifies system configuration because default values that are set at the root level apply to all devices regardless of their device class.

To further customize the system, you can change a specific configuration property at a lower level of the hierarchy without changing the definitions of other configuration properties. As shown in the following illustration, the value of `zPingMonitorIgnore` is changed so that ping monitoring is performed at the /Devices/Server/Windows level.

Figure 49: Device class hierarchy - locally defined value (override)

This locally defined value for `zPingMonitorIgnore` *overrides* the value that is set at the root of the hierarchy. No other properties at this level are affected by this local change; they continue to inherit the value that is set at the root.

Configuration properties allow you to configure the system at a very granular level, down to a particular device. For example, in the following illustration, for the device named `dev.zenoss.com`, the value of `SNMPCommunity` is set to `private`. This value overrides the root value of `public`.

Figure 50: Device class hierarchy - value set on device

If you change the SNMPCommunity value of **dev.zenoss.com** to `public`, it matches the value that is set at the root, but is still explicitly defined. Only if you remove the locally defined property does it again inherit the value of the property that is set at the root.

Viewing and overriding device properties

This section further illustrates the characteristics of configuration properties from the browser interface perspective.

To view configuration properties:

- 1 From the navigation bar, choose **INFRASTRUCTURE**.
- 2 In the tree view, click **DETAILS > Configuration Properties**. The **Configuration Properties** page for the selected device class appears.

The following figure shows device configuration properties that are defined at the root level. The `zCollectorClientTimeout` configuration property has a default value of 180.

Figure 51: Defined device configuration properties - root level

The screenshot shows a configuration management interface with a navigation bar at the top and a main content area below. The navigation bar includes tabs for Devices, Networks, Processes, IP Services, Windows Services, Network Map, Manufacturers, Page, and Tips. The 'Devices' tab is selected. Below the navigation bar is a search bar with placeholder text 'SEE ALL' and a 'Delete Local Copy' button.

The main content area displays a table of configuration properties. The table has columns: Is Local, Category, Name, Value, Label, Description, and Path. The table lists numerous properties, many of which are categorized under 'Cisco' or 'Misc'. Some properties include 'zAggregatorCollectionInterval', 'zCiscoACEUseSSL', 'zCiscoNXAPIPort', and 'zCiscoRemodelEventClassKeys'. The table is scrollable, with a message at the bottom indicating 'DISPLAYING 1 - 35 of 195 ROWS'.

In the following figure, the `zCollectorClientTimeout` configuration property value is set to 170 at the `/Server/Linux` device class. The device class value overrides the default value at this node of the hierarchy.

Figure 52: zCollectorClientTimeout configuration property - local value set

| Configuration Properties | | | | Delete Local Copy |
|--------------------------|------------------|--------------------------|------------|---------------------------------------|
| Is Local | Category | Name ▾ | Value | Path |
| | CiscoUCS | zCiscoUCSManagerPassword | / | |
| | CiscoUCS | zCiscoUCSManagerPort | 80 | / |
| | CiscoUCS | zCiscoUCSManagerUseSSL | false | / |
| | CiscoUCS | zCiscoUCSManagerUser | / | |
| Yes | Modeler Controls | zCollectorClientTimeout | 1 | /Server/Linux/devices/t-cent5a-64.zen |
| | Modeler Controls | zCollectorDecoding | / | |
| | Misc | zCollectorLogChanges | true | / |
| | zencommand | zCommandCommandTimeout | 15 | / |
| | zencommand | zCommandCycleTime | 60 | / |
| | zencommand | zCommandExistanceTest | test -f %s | / |
| | zencommand | zCommandLoginTimeout | 10 | / |
| | zencommand | zCommandLoginTries | 1 | / |

DISPLAYING 1 - 13 OF 127 ROWS

- 3 To remove the override and once again inherit the value from the root of the hierarchy:
 - a Select the property in the list.
 - b Click **Delete Local Copy** and, when prompted, click **OK**.

Device configuration properties

The following table lists device configuration properties. For information about accessing device configuration properties, see [Viewing and overriding device properties](#) on page 62.

Table 4: Device configuration properties

| Property Name | Property Type | Description |
|-------------------------------|---------------|--|
| zAggregatorCollectionInterval | int | Aggregator collection interval in seconds. Default is 300. |
| zCiscoACEUseSSL | boolean | Whether to use SSL when connecting to ACE XML API. |
| zCiscoNXAPIInterval | int | Value in seconds of the rediscovery interval using the Cisco NX-API protocol. Default value is 300. |
| zCiscoNXAPIPort | int | Port for connecting to NX-API. |
| zCiscoNXAPIUseSSL | boolean | Whether to use SSL when connecting to NX-API. |
| zCiscoRemodelEventClassKeys | lines | Allows you to modify the list of SNMP traps that will cause the product schedule an immediate remodeling of the device from which the trap was sent. |
| zCiscoUCSCIMCEventsInterval | int | Event collection interval in seconds. Default is 60. |
| zCiscoUCSCIMCPerfInterval | int | Metric collection interval in seconds. Default is 300. |
| zCiscoUCSCIMCSSLProtocol | string | The SSL/TLS protocol used to connect to a CIMC device. |

| Property Name | Property Type | Description |
|------------------------------|---------------|---|
| zCiscoUCSManagerPassword | password | Password for UCS Manager user name. |
| zCiscoUCSManagerPerfInterval | int | Seconds between UCS Manager statistics collections. Default value is 300. |
| zCiscoUCSManagerPort | int | Port used to connect to the UCS Manager or CIMC XML APIs. Default is 443 and typically should not be changed. |
| zCiscoUCSManagerUseSSL | boolean | Whether to use SSL when connecting to the UCS Manager or CIMC XML APIs. Default is true and typically should not be changed. |
| zCiscoUCSManagerUser | string | UCS Manager user name. |
| zCollectorClientTimeout | int | Allows you to set the timeout time of the collector client in seconds |
| zCollectorDecoding | string | Converts incoming characters to Unicode. |
| zCollectorLogChanges | boolean | Indicates whether to log changes. |
| zCommandCommandTimeout | float | Specifies the time to wait for a command to complete. |
| zCommandExistanceTest | string | DEPRECATED - No longer used. |
| zCommandLoginTimeout | float | Specifies the time to wait for a login prompt. |
| zCommandLoginTries | int | Sets the number of times to attempt login. |
| zCommandPassword | password | Specifies the password to use when performing command logins and SSH. |
| zCommandPath | string | Sets the default path where ZenCommand plug-ins are installed on the local Zenoss Core box (or on a remote box where SSH is used to run the command). |
| zCommandPort | int | Specifies the port to connect to when performing command collection. |
| zCommandProtocol | string | Establishes the protocol to use when performing command collection. Possible values are SSH and telnet. |
| zCommandSearchPath | lines | Sets the path to search for any commands. |
| zCommandUsername | string | Specifies the user name to use when performing command collection and SSH. |
| zControlCenterHost | string | Control Center host name. Defaults to device name. |
| zControlCenterModelCycle | int | Control Center modeling interval. Defaults to 3600s. |
| zControlCenterPassword | password | Password for the Control Center user name. |
| zControlCenterPerfCycle | int | Control Center performance collection interval. Defaults to 300s. |
| zControlCenterPort | int | Port for Control Center. Defaults to HTTPS TCP/443 |
| zControlCenterUser | string | Control Center user name. |

| Property Name | Property Type | Description |
|---------------------------------|---------------|---|
| zDBInstances | *** | ***instancecredentials. This setting is only relevant when the zenoss.winrm.WinMSSQL modeler plugin is enabled. Multiple instances can be specified to monitor multiple SQL Server instances per server. The default instance is MSSQLSERVER. Fill in the user and password to use SQL authentication. Leave the user and password blank to use Windows authentication. |
| zDatasourceDebugLogging | boolean | True or False to debug calculated/aggregated data sources on a single device. |
| zDeviceTemplates | lines | Sets the templates associated with this device. Linked by name. |
| zEnablePassword | boolean | True or False to specify use of password for Cisco routers. |
| zFileSystemMapIgnoreNames | string | Sets a regular expression of file system names to ignore. |
| zFileSystemMapIgnoreTypes | lines | Do not use. |
| zFileSystemSizeOffset | int | SNMP typically reports the total space available to privileged users. Zenoss Core (like the df command) reports capacity based on the space available to non-privileged users. The value of zFileSystemSizeOffset should be the fraction of the total space that is available to non-privileged users. The default reserved value is 5% of total space, so zFileSystemSizeOffset is preset to .95. If the reserved portion is different than 5%, then adjust the value of zFileSystemSizeOffset accordingly. The fraction should be set according to the value (<i>Used + Avail</i>) / <i>Size</i> when the df -PkH command is run at the command line. |
| zHardDiskMapMatch | string | Regular expression that uses the disk ID in the diskstats output to filter disk activity statistics for inclusion in performance monitoring. |
| zIcon | lines | Specifies the icon to represent the device wherever device icon is shown, such as on the network map and device status page. |
| zIdiomPassword | password | IDIOM API password for Cisco IDS/IPS devices. |
| zIdiomUsername | string | IDIOM API username for Cisco IDS/IPS devices. |
| zIfDescription | boolean | Shows the interface description field in the interface list. |
| zIgnoreUnmounted | boolean | True or false to specify whether to ignore unmounted drives. |
| zInterfaceMapIgnoreDescriptions | string | Filters out interfaces based on description. |

| Property Name | Property Type | Description |
|---------------------------------|---------------|---|
| zInterfaceMapIgnoreNames | string | Filters out interfaces that should not be discovered. If you want to use an expression to define this property, note that only Python regular expressions are valid. |
| zInterfaceMapIgnoreTypes | string | Filters out interface maps that should not be discovered. |
| zIpServiceMapMaxPort | int | Specifies the highest port to scan. The default is 1024. |
| zJBossJmxManagementAuthenticate | boolean | DEPRECATED - No longer used. |
| zJBossJmxManagementPassword | password | JMX password |
| zJBossJmxManagementPort | int | The port number used to gather JMX information |
| zJBossJmxManagementUsername | string | JMX username for authentication. |
| zJmxAuthenticate | boolean | True or False to enable/disable authentication. |
| zJmxManagementPort | int | Port that enables JMX management |
| zJmxPassword | password | JMX username password |
| zJmxUsername | string | JMX username |
| zKeyPath | string | Sets the path to the SSH key for device access. |
| zLDAPBaseDN | string | DEPRECATED - No longer used. The Base Distinguished Name for your LDAP server. Typically this is the organization's domain name (for example, dc=foobar,dc=com). |
| zLDAPBindDN | string | DEPRECATED - No longer used. The Distinguished Name to use for binding to the LDAP server, if authentication is required. |
| zLDAPBindPassword | string | DEPRECATED - No longer used. The password to use for binding to the LDAP server, if authentication is required. |
| zLDOMsAutodiscover | boolean | Specify true or false for autodiscovery of LDOMs. |
| zLTMVirtualServerIgnoreNames | string | Regular expression that can be used to prevent matching LTM Virtual Servers from being modeled. |
| zLinks | string | Specifies a place to enter any links associated with the device. |
| zLocalInterfaceNames | string | Regular expression that uses interface name to determine whether the IP addresses on an interface should be incorporated into the network map. For instance, a loopback interface "lo" might be excluded. |
| zLocalIpAddresses | string | Specifies IP addresses that should be excluded from the network map (for example, 127.x addresses). If you have addresses that you reuse for connections between clustered machines they might be added here as well. |

| Property Name | Property Type | Description |
|-----------------------------|---------------|---|
| zMaxOIDPerRequest | int | Sets the maximum number of OIDs to be sent by the SNMP collection daemons when querying information. Some devices have small buffers for handling this information so the number should be lowered. |
| zMySQLConnectionString | string | Enables setting the path, type and credentials for the MySQL connections. For example: Path: /server/Linux/devices/localhost.localdomain, Type: miltinecredentials, MYSQL connection credentials: User Password Port |
| zMySqlPassword | password | MySQL user password |
| zMySqlPort | string | MySQL connection port |
| zMySqlTimeout | int | MySQL timeout. Default value is 30s. |
| zMySqlUsername | string | MySQL user name. |
| zNetAppNumRecordsPerRequest | int | Number of records per NetApp Monitor request. Default is 100. |
| zNetAppSSL | boolean | Boolean true or false to enable SSL use with the NetApp Monitor. |
| zNmapPortscanOptions | string | Options used on nmap when scanning ports. Used in IpServiceMap. |
| zNodesAutodiscover | boolean | Specify true or false for autodiscovery of nodes. |
| zPingMonitorIgnore | boolean | Whether to ping the device. |
| zProdStateThreshold | int | Production state threshold at which Zenoss Core will begin to monitor a device. |
| zPropertyMonitorInterval | int | Polling interval of the configured property data sources. System-wide setting. Default is 300s. |
| zPythonClass | string | DO NOT USE |
| zRancidGroup | string | DEPRECATED - No longer used. RANCID group attribute. Controls what router.db file the device is written to. Can be set at the device class or device level. Default is router on the /Network/Router/Cisco class. |
| zRancidRoot | string | DEPRECATED - No longer used. File system directory where RANCID is installed. It may be NFS mounted from the RANCID server. Default is /opt/rancid. |
| zRancidType | string | DEPRECATED - No longer used. RANCID type attribute. Controls what device type is written to the router.db file. Can be set at the |

| Property Name | Property Type | Description |
|------------------------------|---------------|---|
| | | device class or device level. Default is <code>cisco</code> on the / Network/Router/Cisco class. |
| zRancidUrl | string | DEPRECATED - No longer used. Base URL. |
| zRMMonCCHost | string | Host name or IP address of the Control Center instance on which the Zenoss Core to monitor is running. |
| zRMMonCCUser | string | User name of the for this Zenoss Core. |
| zRMMonCCPassword | password | Password for the Control Center user name. |
| zRMMonCCPort | int | Port for Control Center. Defaults to HTTPS TCP/443. |
| zRMMonRabbitPassword: | password | Password for the RabbitMQ user name. Defaults to the system default. |
| zRMMonRabbitUser: | string | User name for RabbitMQ. Defaults to the system default. |
| zRMMonTenantHost | string | Zenoss Core host name. |
| zRMMonTenantPassword | password | Password for the Zenoss Core user name. |
| zRMMonTenantPerfCycle | int | Performance collection interval. Defaults to 30s. |
| zRMMonTenantPort | int | Port for Zenoss Core. Defaults to HTTPS TCP/443. |
| zRMMonTenantUseSsl | boolean | True or false value to use SSL. Defaults to <code>true</code> . |
| zRMMonTenantUser | string | Zenoss Core user name. |
| zRouteMapCollectOnlyIndirect | boolean | Only collect routes that are indirectly connected to the device. |
| zRouteMapCollectOnlyLocal | boolean | Only collect local routes. (These usually are manually configured rather than learned through a routing protocol.) |
| zRouteMapMaxRoutes | int | Sets maximum number of routes to collect. Default value is 500. |
| zSnmpAuthPassword | password | The shared private key used for authentication. Must be at least 8 characters long. |
| zSnmpAuthType | string | Use "MD5" or "SHA" signatures to authenticate SNMP requests |
| zSnmpCollectionInterval | int | Defines, in seconds, how often the system collects performance information for each device. |
| zSnmpCommunities | lines | Array of SNMP community strings that ZenModeler uses when collecting SNMP information. When you set this property, communities are tried in order; the first in the list that is successful is used as <code>zSnmpCommunity</code> . If none is successful, then the current value of <code>zSnmpCommunity</code> is used. The default value for the entire system is "public." |

| Property Name | Property Type | Description |
|----------------------------|---------------|--|
| zSnmpCommunity | string | Community string to be used when collecting SNMP information. If it is different than what is found by ZenModeler, it will be set on the modeled device. |
| zSnmpContext | string | Configures zSNMP context to map logical network entity such as a topology or protocol instance. |
| zSnmpDiscoveryPorts | int | List of UDP ports to try when performing SNMP discovery. Defaults to 161 if not set. |
| zSnmpEngineId | string | SNMPv3 engine ID for the device. Will be discovered when SNMPv3 is used. |
| zSnmpMonitorIgnore | boolean | Whether or not to ignore monitoring SNMP on a device. |
| zSnmpPort | int | Port that the SNMP agent listens on. |
| zSnmpPrivPassword | password | The shared private key used for encrypting SNMP requests. Must be at least 8 characters long. |
| zSnmpPrivType | string | "DES" or "AES" cryptographic algorithms. |
| zSnmpSecurityName | string | The Security Name (user) to use when making SNMPv3 requests. |
| zSnmpTimeout | float | Timeout time in seconds for an SNMP request |
| zSnmpTries | int | Amount of tries to collect SNMP data |
| zSnmpVer | string | SNMP version used. Valid values are v2c, v1 |
| zSshConcurrentSessions | int | Maximum number of sessions supported by the remote device's MAX_SESSIONS parameter. Common values for AIX are 2 or 10. |
| zStatusConnectTimeout | float | The amount of time that the zenstatus daemon should wait before marking an IP service down. |
| zSugarCRMBase | - | DEPRECATED - No longer used. |
| zSugarCRMPassword | password | DEPRECATED - No longer used. Password for the zSugarCRMUsername user. |
| zSugarCRMTestAccount | - | DEPRECATED - No longer used. |
| zSugarCRMUsername | string | DEPRECATED - No longer used. User name that is allowed to log in to the Sugar CRM server. |
| zSysedgeDiskMapIgnoreNames | string | Regular expression used by zenoss.snmp.SysedgeDiskMap modeler plugin. Disks with matching names will not be modeled. |
| zTelnetEnable | boolean | When logging into a Cisco device issue the enable command to enable access during command collection. |
| zTelnetEnableRegex | string | Regular expression to match the enable prompt. |

| Property Name | Property Type | Description |
|----------------------------------|---------------|---|
| zTelnetLoginRegex | string | Regular expression to match the login prompt. |
| zTelnetPasswordRegex | string | Regular expression to match the password prompt. |
| zTelnetPromptTimeout | float | Time to wait for the telnet prompt to return. |
| zTelnetSuccessRegexList | lines | List of regular expressions to match the command prompt. |
| zTelnetTermLength | boolean | On a Cisco device, set term length to Zero. |
| zTomcatJ2EEApplicationName | string | Used to construct MBean names for a specific application deployed on Tomcat, typically used for JSP and Servlet statistics. |
| zTomcatJ2EEServerName | string | Used to construct MBean names for a specific application deployed on Tomcat, typically used for JSP and Servlet statistics. |
| zTomcatJmxManagementAuthenticate | - | DEPRECATED - No longer used. |
| zTomcatJmxManagementPassword | password | JMX password. |
| zTomcatJmxManagementPort | int | The port number used to gather JMX information. |
| zTomcatJmxManagementUsername | string | JMX username for authentication. |
| zTomcatListenHost | string | The hostname on which Tomcat is listening for web requests. This is used to construct MBean names |
| zTomcatListenPort | string | The Tomcat connector, which is a port and protocol (http, jk...) that Tomcat is listening on. This is used to construct MBean names that monitor bytes, error and requests on that connector. |
| zTomcatServletName | string | Specific Servlet name to monitor. |
| zTomcatServletUri | string | URI of Servlet to monitor. |
| zTomcatWebAppUri | string | URI path for a Tomcat web application. Used to construct MBean names. |
| zUsesManageIp | boolean | True or False to specify use of manage IP. Used to avoid setting manage IP when a device is added. Default is True for most device classes. |
| zUsesStandardDeviceCreationJob | boolean | True or False to specify whether to use the standard device creation job. |
| zVCloudPassword | password | DEPRECATED - No longer used. Password for the zVCloudUsername user. |
| zVCloudPort | int | DEPRECATED - No longer used. Value of the cell port number. |
| zVCloudUsername | string | DEPRECATED - No longer used. |

| Property Name | Property Type | Description |
|---|---------------|--|
| | | User name for the cell in the form <code>username@organization</code> . For example, if you want to define the cell administrator, enter <code>administrator@system</code> . |
| <code>zVSphereEndpointHost</code> | string | vSphere host name |
| <code>zVSphereEndpointPassword</code> | password | vSphere username password |
| <code>zVSphereEndpointPort</code> | int | Port that is used to connect to vSphere Endpoint. |
| <code>zVSphereEndpointUseSsl</code> | boolean | vSphere boolean true or false for SSL use |
| <code>zVSphereEndpointUser</code> | string | vSphere username |
| <code>zVSphereHostCollectionClusterWhitelist</code> | lines | The whitelist filters the hosts that are monitored, based on cluster names. |
| <code>zVSphereHostPingBlacklist</code> | lines | List of regular expressions to control which management IP address to ping (matches against <code>hostname:nicname:ip</code>). |
| | | Note <code>zVSphereHostPingWhitelist</code> takes precedence over <code>zVSphereHostPingBlacklist</code> . |
| <code>zVSphereHostPingWhitelist</code> | lines | List of regular expressions to control which management IP address to ping (matches against <code>hostname:nicname:ip</code>). |
| | | Note <code>zVSphereHostPingWhitelist</code> takes precedence over <code>zVSphereHostPingBlacklist</code> . |
| <code>zVSphereHostSystemPassword</code> | password | Password that is used to access ESX hosts via ssh and API. |
| <code>zVSphereHostSystemUser</code> | string | User name that is used to access ESX hosts via ssh and API. |
| <code>zVSphereLUNContextMetric</code> | boolean | Controls whether to use LUN-specific metric names when storing performance data. |
| | | Note: The default value is False. Changing the value to True causes historical metrics to become inaccessible. For more information, contact Zenoss Support. |
| <code>zVSphereModelCache</code> | lines | vSphere model cache |
| <code>zVSphereModelIgnore</code> | lines | vSphere model ignore |
| <code>zVSphereModelMpIndexObjs</code> | int | Advanced tuning parameter. For more information, contact Zenoss Support. |
| <code>zVSphereModelMpLevel</code> | int | Advanced tuning parameter. For more information, contact Zenoss Support. |
| <code>zVSphereModelMpObjs</code> | int | Advanced tuning parameter. For more information, contact Zenoss Support. |

| Property Name | Property Type | Description |
|-------------------------------------|---------------|---|
| zVSpherePerfDelayCollectionMinutes | int | Value of how long to lag performance data collection. Default value is 0. |
| zVSpherePerfMaxAgeMinutes | int | Default value is 28 minutes. |
| zVSpherePerfParallelQueries | int | Default value is 6. |
| zVSpherePerfQueryChunkSize | int | Value of how many performance requests to make at a time. Default value is 250. |
| zVSpherePerfQueryRaw20 | boolean | Default value is <code>true</code> . |
| zVSpherePerfQueryTimeout | int | Default value is 200. |
| zVSpherePerfQueryVcChunkSize | int | Default value is 64. |
| zVSpherePerfQueryVcRaw20 | boolean | Default value is <code>false</code> . |
| zVSpherePerfRecoveryMinutes | int | Default value is 240 minutes. |
| zVSpherePerfTimeoutRecoveryMinutes | int | When a timeout error occurs in querying a specific metric, it is "blacklisted" for this number of minutes before it is retried. This action avoids repeated errors due to attempts to query a metric that is not working properly. The default is one hour. If gaps appear in the graphs, you can safely lower the value. |
| zVSphereVMContextMetric | boolean | Controls whether to use VM-specific metric names when storing performance data. Note: The default value is False. Changing the value to True causes historical metrics to become inaccessible. For more information, contact Zenoss Support. |
| zVSpherePerfWindowSize | int | DEPRECATED - No longer used. |
| zWBEMPassword | password | WBEM password |
| zWBEMPort | int | Value of the WBEM port number. Default value is 5989. |
| zWBEMUseSSL | boolean | True or false value to use SSL. Default value is <code>true</code> . |
| zWBEMUsername | string | WBEM username |
| zWebLogicJmxManagementAuthenticate- | | DEPRECATED - No longer used. |
| zWebLogicJmxManagementPassword | password | JMX password |
| zWebLogicJmxManagementPort | int | The port number used to gather JMX information |
| zWebLogicJmxManagementUsername | string | JMX username for authentication |
| zWebTxAgent | string | Default value is <code>ZenWebTx/1.0</code> . |
| zWebTxPassword | password | WebTx password |
| zWebTxRealm | string | WebTx realm |
| zWebTxUser | string | WebTx user |

| Property Name | Property Type | Description |
|------------------------|---------------|---|
| zWebsphereAuthRealm | string | Used for HTTP basic authentication. This field is not required, and is empty by default. |
| zWebsphereNode | string | Used by the provided template to build the queries for the data to collect. You must supply a value for this field. For example: <i>serverA</i> |
| zWebspherePassword | password | Used for HTTP basic authentication. This field is not required, and is empty by default. |
| zWebsphereServer | string | Used by the provided template to build the xpath queries for the data to collect. You must supply a value for this field. For example: <i>serverAB</i> , There is no default value. |
| zWebsphereURLPath | string | Path to the PMI servlet on a WebSphere instance. The default value is the default path on a WebSphere installation: <i>wasPerfTool/servlet/perfServlet</i> |
| zWebsphereUser | string | Used for HTTP basic authentication. This field is not required, and is empty by default. |
| zWinKDC | string | IP address or Fully Qualified Domain Name of a valid Windows domain controller. Must be set if domain authentication is used. |
| zWinKeyTabFilePath | string | This property is currently used and reserved for future use when keytab files are supported. |
| zWinPerfmonInterval | int | Interval, in seconds, at which Windows Perfmon datapoints will be collected. Default value is 300. It is possible to override the collection interval for individual counters. |
| zWinRMClusterNodeClass | string | Path under which to create cluster nodes. |
| zWinRMEvelopeSize | int | Used when WinRM configuration setting "MaxEnvelopeSizekb" exceeds default of 512k. |
| zWinRMKrb5DisableRDNS | boolean | Set to <code>true</code> to disable reverse DNS lookups by Kerberos. Only set at /Server/Microsoft level. |
| zWinRMKrb5includedir | string | Directory path for Kerberos configuration files. |
| zWinRMLocale | string | Communication locale to use for monitoring. Reserved for future use. |
| zWinRMPassword | password | Password for the user defined by <code>zWinRMUser</code> |
| zWinRMPort | int | The port on which the Windows server is listening for WinRM or WS-Management connections. Default value is 5985. It is uncommon for this to be configured as anything else. |
| zWinRMServerName | string | This property should only be used in conjunction with domain authentication when the DNS PTR record for a monitored server's managed IP address does not |

| Property Name | Property Type | Description |
|------------------|---------------|---|
| | | resolve to the name by which the server is known in Active Directory. For example, if myserver1 is known as myserver1.ad.example.com by Active Directory and is being managed by IP address 192.51.100.21, but 192.51.100.21 resolves to www.example.com, you will have to set zWinRMServerName to myserver1.ad.example.com for domain authentication to work. |
| | | If many Windows servers in your environment do not have DNS PTR records that match Active Directory, it is recommended that you set the name of the Zenoss device to be the fully-qualified Active Directory name and set zWinRMServerName to \${here/titleOrId} at the /Server/Microsoft/Windows device class. This avoids the necessity of setting zWinRMServerName on every device. |
| | | It is recommended to leave zWinRMServerName blank if local authentication is used, or DNS PTR records match Active Directory. This allows Zenoss to not rely on DNS resolution while monitoring, and avoids the overhead of configuring zWinRMServerName. |
| zWinRMUser | string | The syntax used for zWinRMUser controls whether Zenoss will attempt Windows local authentication or domain (kerberos) authentication. If the value of zWinRMUser is <i>username</i> , local Windows authentication will be used. If zWinRMUser is <i>username@example.com</i> , domain authentication will be used. The zWinKDC and potentially the zWinRMServerName properties become important. |
| zWinRSCodePage | int | Code page used by monitoring user account. |
| zWinScheme | string | Must be set to http or https. Default value is http. |
| zWinTrustedKDC | string | Windows Trusted KDC. |
| zWinUseWsmanSPN | boolean | Set to true if HTTP/HTTPS service principles are exclusively for use by a particular service account. |
| zWinTrustedRealm | string | Windows Trusted Realm. |

Viewing and overriding event properties

- 1 From the navigation menu, choose **EVENTS > Event Classes**.
- 2 From the drop-down list, choose **Configuration Properties**.

| Label | Description | Path |
|--------------------------|--------------|--------------------------|
| zEventAction | | / |
| zEventClearClasses | | / |
| zEventMaxTransformFails | 10 | zEventMaxTransformFails |
| zEventSeverity | Default (-1) | zEventSeverity |
| zFlappingIntervalSeconds | 3600 | zFlappingIntervalSeconds |
| zFlappingSeverity | Error (4) | zFlappingSeverity |
| zFlappingThreshold | 4 | zFlappingThreshold |

- 3 To view and override event configuration properties for a specific event class:
 - a Navigate to that class, and then select **Configuration Properties**.
 - b Double-click the configuration property that you want to change.
 - c In the **Edit Config Property** dialog box, make changes and then click **SUBMIT**.
The presence of **Yes** in the **Is Local** column indicates an overriding value.
- 4 To remove an override and once again inherit the value from the root of the hierarchy:
 - a Select the property in the list.
 - b Click **Delete Local Copy** and, when prompted, click **OK**.

Event configuration properties

The following table lists event configuration properties.

Table 5: Event configuration properties

| Property Name | Property Type | Description |
|---------------|---------------|--|
| zEventAction | string | Specifies the database table in which an event will be stored. Possible values are: status, history and drop. Default is status, meaning the event will be an “active” event. History sends the event directly |

| Property Name | Property Type | Description |
|--------------------------|---------------|---|
| | | to the history table. Drop tells the system to discard the event. |
| zEventClearClasses | lines | Lists classes that a clear event should clear (in addition to its own class). |
| zEventMaxTransformFails | int | After the specified number of failures, disable bad transforms from executing. Default is 10. |
| zEventSeverity | int | Overrides the severity value of events from this class. Possible values are 5-Critical, 4-Error, 3-Warning, 2-Info, 1-Debug, 0-Clear, and -1-Default. |
| zFlappingIntervalSeconds | int | Defines the time interval to check for event flapping (changing severity level repeatedly). Default value is 3600 seconds. |
| zFlappingSeverity | int | Drop-down list to set the severity to check for event flapping. If the severity level on an event changes from this value a certain number of times (zFlappingThreshold) within a certain time range (zFlappingIntervalSeconds) then an event flapping event is generated. Possible values include: 5-Critical, 4-Error, 3-Warning, 2-Info, 1-Debug, and 0-Clear. |
| zFlappingThreshold | int | Number of times an event severity must flap within an interval. One of the parameters to define in order to generate event flapping events. |

Viewing and overriding network properties

The **Networks** page lists networks by IP address. You can view and change network configuration property values and inheritance selections.

- From the navigation menu, choose **INFRASTRUCTURE > Networks**.

Figure 53: Networks

| Is Local | Category | Name | Value | Path |
|----------|----------|---------------------|----------------------------------|------|
| Yes | Misc | zAutoDiscover | true | / |
| Yes | Misc | zDefaultNetworkTree | 24 32 | / |
| Yes | Misc | zDrawMapLinks | true | / |
| Yes | Misc | zIcon | /zport/dmd/img/icons/network.png | / |
| Yes | Misc | zPingFailThresh | 168 | / |
| Yes | Misc | zPreferSnmpNaming | false | / |
| Yes | Misc | zSnpStrictDiscovery | false | / |

- From the **Display** drop-down menu, select **Configuration Properties**.
- To remove an override and once again inherit the value from the root of the hierarchy:
 - Select the property in the list.
 - Click **Delete Local Copy** and, when prompted, click **OK**.

Network configuration properties

The following table lists network configuration properties.

Table 6: Network configuration properties

| Property Name | Property Type | Description |
|----------------------|---------------|---|
| zAutoDiscover | boolean | Specifies whether zendisc should perform auto-discovery on this network. (When performing network discovery, this property specifies whether the system should discover devices and subnetworks on the network.) |
| zDefaultNetworkTree | lines | A network subnet is automatically created for each modeled device, based on that device's subnet mask setting. To create higher-level subnets automatically from the discovery and modeling processes, add the specific subnet mask breakpoints. For example: 8, 16. If you then model a device with, for example, an IP address of 192.0.2.0, and a subnet mask of 255.255.255.0 (corresponding to a /24 subnet), device discovery will create a 192.0.0.0/8 network containing 192.0.2.0/16, containing 192.0.2.0/24, containing your device. |
| zDrawMapLinks | boolean | Calculating network links "on the fly" is resource-intensive. If you have a large number of devices that have been assigned locations, then drawing those map links may take a long time. You can use this property to prevent the system from drawing links for specific networks (for example, a local network comprising many devices that you know does not span multiple locations). |
| zIcon | string | Use to specify device icons that appear on the device status page, Dashboard, and network map. |
| zPingFailThresh | int | Specifies the number of pings sent without being returned before zendisc removes the device. |
| zPreferSnmpNaming | boolean | Specifies that when network discovery occurs, it uses the device name comes from SNMP rather than reverse DNS. |
| zSnmpStrictDiscovery | boolean | Specifies that if SNMP does not exist on the device during network discovery, ignore the device. |

5

Monitoring templates

The system stores performance configuration data in *templates*. Templates contain other objects that define where and how to obtain performance data, thresholds for that data, and data graphs.

You can define a template anywhere in the device class hierarchy, or on an individual device.

Templates are divided among three types:

- Device
- Component
- Interface

Creating templates

You can create a template by overriding an existing template. To override a template:

- 1 Navigate to the template you want to copy.
- 2 From the Action menu, select **Copy/Override Template**. The Copy/Override dialog box appears.
- 3 Select the bound template to override, and then click **Submit**. The copied template appears in the list of templates as locally defined.

Renaming templates

To rename an existing template:

- 1 Select **Advanced > Monitoring Templates**.
- 2 Expand the organizer containing the template to be renamed, and then the class containing the template.
- 3 From the Action menu, select **View and Edit Details**. The Edit Template Details dialog box appears.
- 4 Enter a new name in the Name field.
- 5 Click **Submit**.

Template binding

The determination of which templates apply to what objects is called *binding*. Templates are bound in different ways, depending on the objects to which they are bound.

Device templates

Device templates are applied to devices, one to each device. The system employs a single rule to bind device templates to devices: the value of the zDeviceTemplates property. For most device classes, this is "Device."

Common device templates are:

- Device
- MySQL
- Apache
- Active Directory
- MSExchangeIS
- MSSQLServer
- IIS

For the Server/Linux/MySQL device class, the zDeviceTemplates property might contain, for example, "Device" and "MySQL." The system would collect CPU and memory information by using the Device template, and MySQL-specific metrics by using the MySQL template.

Binding templates

To bind a device template to a device class or device:

- 1 From the devices list, select a device class or device.
- 2 On the Overview page, select **Bind Templates** from the Action menu. The Bind Templates dialog box appears.

Figure 54: Bind Templates



- 3 Move templates between the Available and Selected lists using the arrows.
- 4 Click Save.

Resetting bindings

Resetting template bindings removes all locally bound templates and uses the default template values. To reset bindings for a selected device or device class:

- 1 Select **Reset Bindings** from the Action menu.

The Reset Template Bindings dialog box appears.

- 2 Click **Reset Bindings** to confirm the action.

Component templates

Component templates are named exactly according to the name of the underlying class that represents a component. For example, the FileSystem template is applied to file systems. Component templates can be applied multiple times to each device, depending on how many of the device's components match the template. Configuration properties do not control the application of component templates.

Note Do not manually bind component templates.

Common component templates are:

- FileSystem, HardDisk, IPService, OSProcess, WinService
- Fan, PowerSupply, TemperatureSensor
- LTMVirtualServer, VPNTunnel

Interface templates

Interface templates are applied to network interfaces by using a special type of binding. Instead of using the name of the underlying class, the system looks for a template with the same name as the interface type. You can find this type in the details information for any network interface.

If Zenoss Core cannot locate a template that matches the interface type, then it uses the ethernetCsmacd template.

Example: Defining templates in the device hierarchy

You add a new device at /Devices/Server/Linux named Example1Server. You have not edited the value of its zDeviceTemplates property, so it inherits the value of "Device" from the root device class (/Devices). Zenoss Core looks to see if there is a template named Device defined on Example1Server itself. There is not, so it checks /Devices/Server/Linux. There is a template named Device defined for that device class, so that template is used for Example1Server. (There also is a template named Device defined at the root level (/Devices), but the system does not use this one because the template at /Devices/Server/Linux overrides it.)

Example: Applying templates to multiple areas in the device hierarchy

You want to perform specific monitoring of servers running a certain Web application, but those servers are spread across several different device classes. You create a template at /Devices called WebApplication with the appropriate data sources, thresholds and graphs. You then append the name "WebApplication" to the zDeviceTemplates configuration property for the devices classes, the individual devices running this Web application, or both.

Basic monitoring

Availability monitoring

The availability monitoring system provides active testing of the IT infrastructure, including:

- Devices
- Network
- Processes
- Services

Availability monitoring is facilitated by:

- **Zenping**- The system's Layer-3 aware, topology-monitoring daemon. Zenping performs high-performance, asynchronous testing of ICMP status. The most important element of this daemon is that Zenoss Core has built a complete model of your routing system. If there are gaps in the routing model, the power of Zenping's topology monitoring will not be available. If there are gaps, this issue can be seen in the `zenping.log` file. Zenping uses Nmap to build a ping tree and perform Layer 3 suppression.
- **Zenstatus**- Performs active TCP connection testing of remote daemons.

Controlling ping cycle time

Follow these steps to modify the ping cycle time:

- 1 Open Control Center and click your instance of Zenoss Core to open the application overview page.
- 2 In the Services section, click **zenping** to open the zenping Overview page.
- 3 In the Configuration Files section, click **Edit** next to the `/opt/zenoss/etc/zenping.conf` file. The Edit Configuration window appears.
- 4 Uncomment the `zenhubpinginterval` line and edit the default value of 30 to your desired ping cycle time.
- 5 Click **Save**.

Using the predefined /Ping device class

The /Ping device class is a configuration for devices that you want to monitor only for availability. The system does not gather performance data for devices placed in this class. You can use the /Ping device class as a reference for your own configuration; or, if you have a device that you want to monitor solely for availability, you can place it under this class.

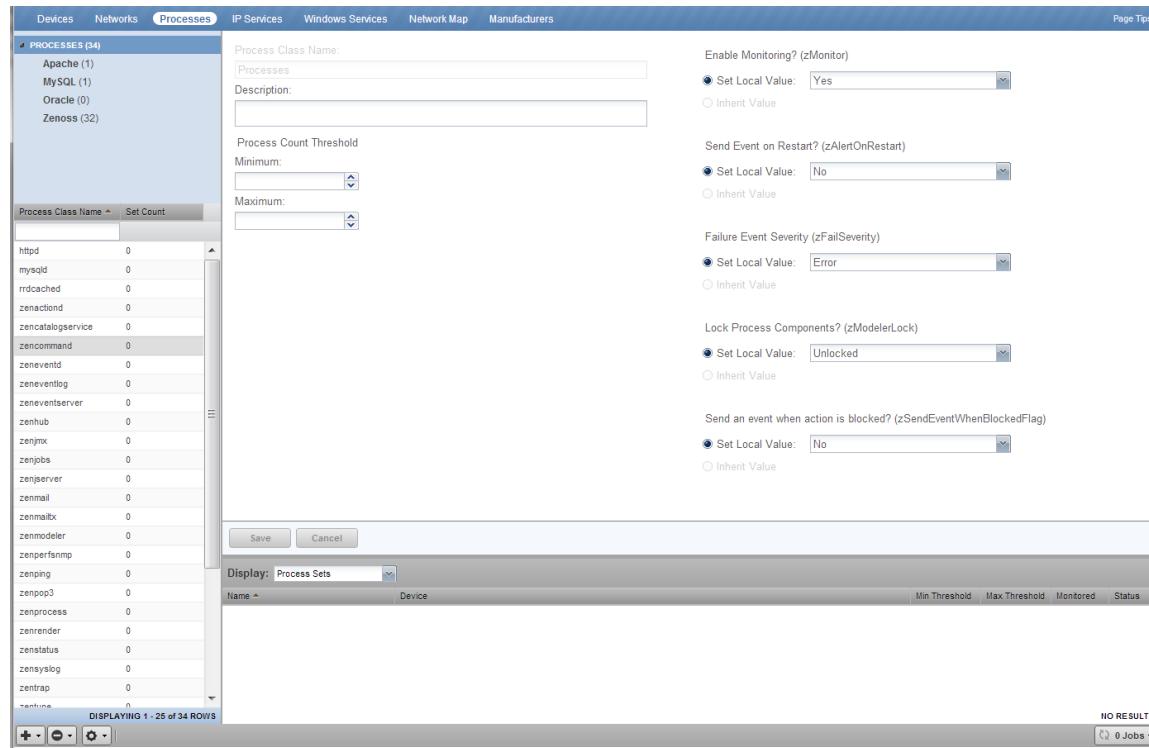
Monitoring processes

Zenoss Core provides process availability monitoring for hosts that support SNMP or SSH access. Process monitoring features include:

- Process classes, defined by Python regular expressions. Classes may generate one or more process sets, each containing one or more process instances.
- Process sets may include process instances running on multiple hosts. This captures related or redundant processes, enabling a more wholistic view of data center services.
- Process set names, to replace the often-cryptic names of process instances with descriptive labels.
- Process set locking, to maintain continuity of data collection if the members of a given process set are not running during modeling.
- A testing dialog, to discover and refine the sets a class generates.

Use the Processes page (**Infrastructure > Processes**) to create and manage process classes and process sets.

Figure 55: Processes page



The tree view shows process class organizers (at the top) and the list of process classes in each organizer (the rest of the view). You may filter the list with the active search field, at the top of the list.

Example: Creating a process class

This section provides an example of using process availability monitoring to create a new process class, for database processes. The database runs on a Linux host, and the following output is a partial list of the results of the `ps axho args` command on the database host. Process monitoring uses the output of that command (or its equivalent) as input for regular expression matching.

```
ora_pmon_orcl ora_psp0_orcl ora_vktm_orcl ora_gen0_orcl ora_diag_orcl
ora_dbrm_orcl ora_dia0_orcl ora_mman_orcl ora_dbw0_orcl ora_lgwr_orcl
ora_ckpt_orcl ora_smon_orcl ora_reco_orcl ora_mmon_orcl ora_mmn1_orcl
ora_d000_orcl ora_s000_orcl ora_s001_orcl ora_s002_orcl ora_s003_orcl
```

```

ora_s004_orcl ora_s005_orcl ora_s006_orcl ora_s007_orcl ora_s008_orcl
ora_s009_orcl ora_p000_orcl ora_p001_orcl ora_p002_orcl ora_p003_orcl
ora_p004_orcl ora_qmnc_orcl ora_n000_orcl ora_l000_orcl ora_l001_orcl
ora_l002_orcl ora_l003_orcl

```

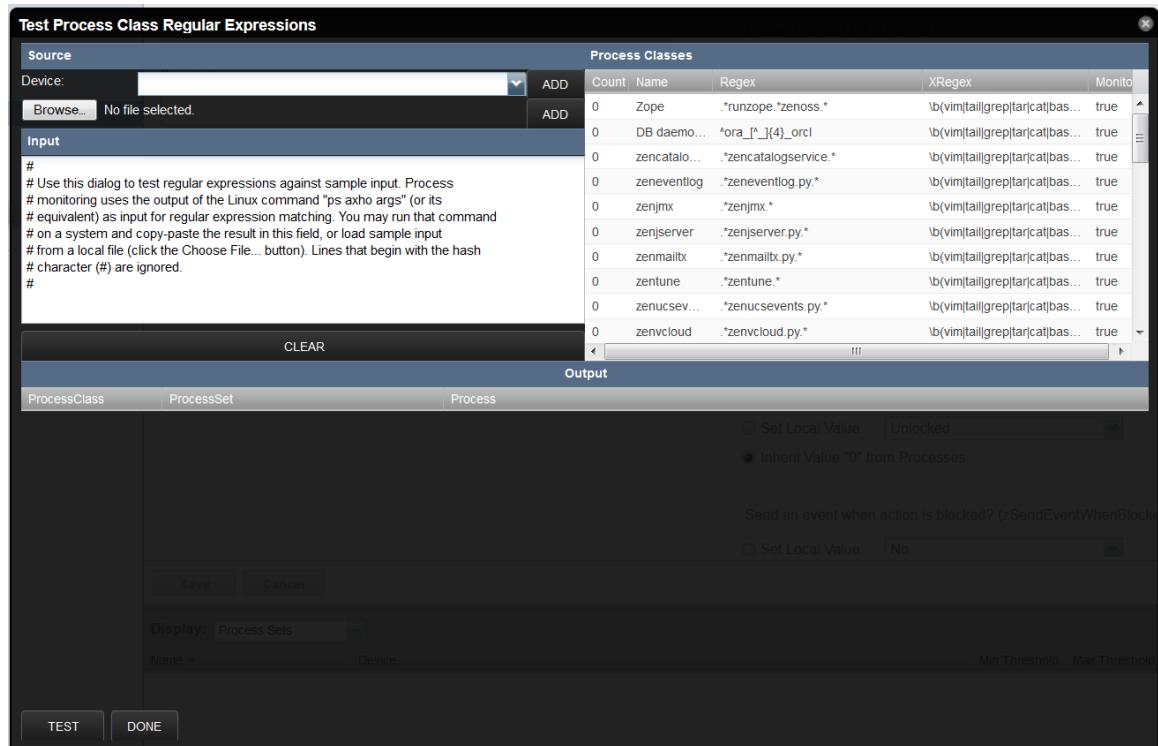
The following subsections provide procedures for creating a process class that captures a selection of the preceding process instances in process sets.

Test existing process classes

The existing process classes may already capture the process sets you want. Follow these steps to test the existing process classes.

- 1 Log in to the Zenoss Core user interface.
- 2 Navigate to **Infrastructure > Processes**.
- 3 In the lower-left corner of the tree view, click the Action menu, and select **Test All Process Classes Regular Expressions**.

Figure 56: Test all process classes dialog



The top-right portion of the dialog box displays all of the process classes, in the order in which they are evaluated.

- 4 Select the process names in the previous section, and copy them into a paste buffer.
- 5 In the **Test Process Class Regular Expressions** dialog, select all of the existing text in the **Input** area, and then paste the process names from the buffer. Alternatively, you may paste the process names into an empty file, save the file on the system from which your browser is launched, and then use the **Choose File** button to insert the process names.
- 6 At the bottom-left corner of the dialog, click **Test**.
- 7 If any process sets are created, they are displayed in the list area, above the **Test** button.

Create an organizer

Complete the previous section, and then follow these steps to create a process class organizer.

- 1 Log in to the Zenoss Core user interface.
- 2 Navigate to **Infrastructure > Processes**.
- 3 In the lower-left corner of the tree view, click the **Add** menu, and select **Add Process Class Organizer**.
- 4 In the **Add Process Class Organizer** dialog, enter Database, and then click **Submit**.

Create a process class

Complete the previous section, and then follow these steps to create a process class.

- 1 At the top of the tree view, double-click **Processes**, the root organizer to open it.
- 2 Select **Database**.
- 3 In the lower-left corner of the tree view, click the **Add** menu, and select **Add Process Class**.
- 4 In the **Add Process Class** dialog, enter DB daemons, and then click **Submit**.
- 5 At the top of the tree view, double-click **Processes** to open it, and then select **Database**.

Define the regular expression series of a process class

Complete the previous section, and then follow these steps to create the series of regular expressions that define a process class, and generate process sets.

- 1 In the list area of the tree view, select **DB daemons**.

Figure 57: Process class definition page

Process Class Name: DB daemons

Description: Database daemons

Enable Monitoring? (zMonitor)

Set Local Value: Yes

Inherit Value "Yes" from Processes

Send Event on Restart? (zAlertOnRestart)

Set Local Value: Yes

Inherit Value "No" from Processes

Failure Event Severity (zFailSeverity)

Set Local Value: Error

Inherit Value "Error" from Processes

Lock Process Components? (zModelerLock)

Set Local Value: Unlocked

Inherit Value "0" from Processes

Send an event when action is blocked? (zSendEventWhenBlockedFlag)

Set Local Value: No

Inherit Value "No" from Processes

Matching Rules (performance metrics will start over if changed)

Include processes like: DB daemons

Exclude processes like: \b(vim|tail|grep|tar|cat|bas)

Replace command line text:

With:

Process Count Threshold

Minimum:

Maximum:

Process Class

Database

- 2 In the **Description** field, enter Database daemons.
- 3 In the **Include processes like** field, replace DB daemons with a *Python regular expression* that selects the database processes. For example, `ora_[^_]{4}_orcl`. The example regular expression selects all of the processes in the sample process instance list.

- 4 In the **Exclude processes like** field, enter a regular expression to remove processes from the results of the preceding regular expression. The default entry excludes common user commands. The default entry does not exclude any of the processes in the sample process instance list.
- 5 The next two fields, **Replace command line text** and **With**, work together to simplify the names of process sets.
 - In the **Replace command line text** field, enter a regular expression containing one or more pattern groups.
 - In the **With** field, enter replacement text, along with the sequence number of one or more of the pattern groups defined in the previous field.

For example, to create four pattern groups for database processes, enter the following regular expression in the **Replace command line text** field:

```
^(ora_) ([a-z]) (.{4}) (orcl)
```

To use two of the pattern groups in the replacement text, enter the following text in the **With** field:

```
DB [\4] daemons starting with [\2]
```

Each unique replacement generated by the combination of the text plus the inserted pattern sequences becomes a process set. In the case of this example, pattern group 4 does not vary, but pattern group 2 does. So the number of process sets generated by this class will equal the number of unique alphabetic characters found in the first position after the first underscore.

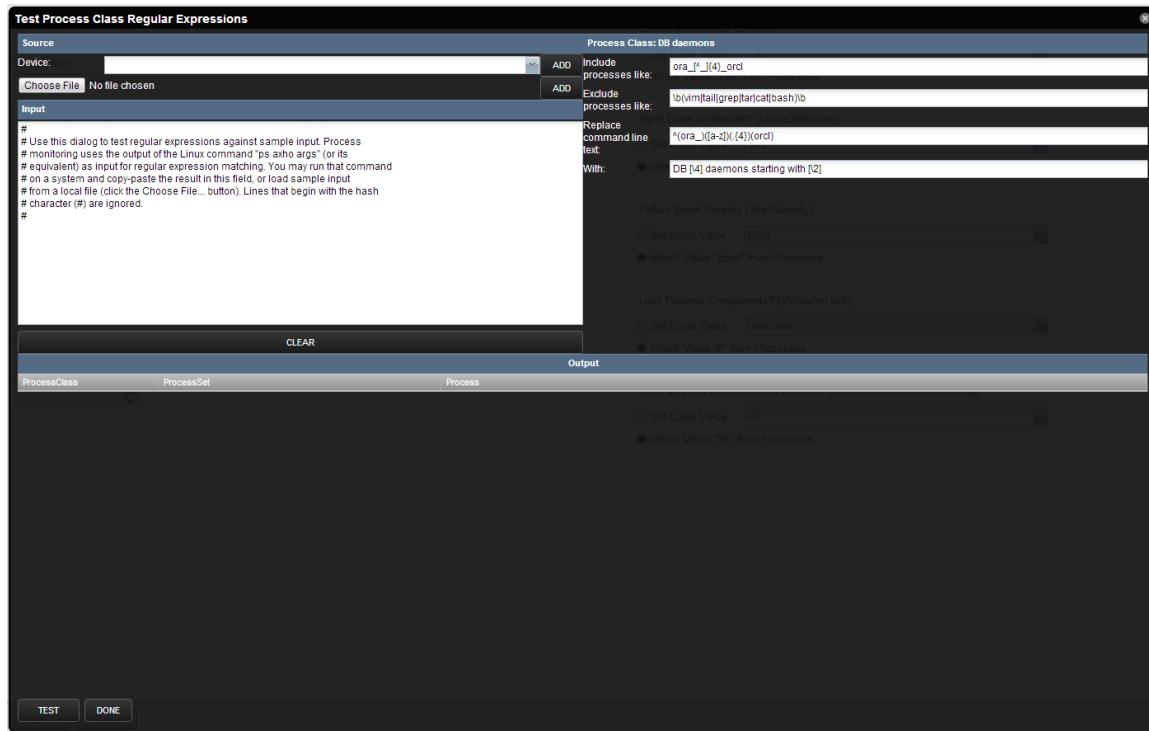
- 6 Click **Save**.

Test a process class

Complete the previous section, and then follow these steps to test a single process class.

- 1 In the lower-left corner of the tree view, click the Action menu, and select **Test Process Class Regular Expressions**.

Figure 58: Test process class dialog



The top-right portion of the dialog box displays the regular expression series that defines this process class.

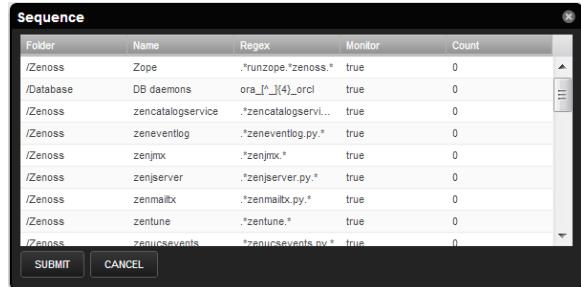
- 2 Select the process names in the previous section, and copy them into a paste buffer.
- 3 In the **Input** area, select all of the existing text, and then paste the process names from the buffer.
- 4 At the bottom-left corner of the dialog, click **Test**.
- 5 The **Output** area displays each individual match, along with the count of unique process sets. You may refine the regular expressions and retest as often as you like.
- 6 Click **Done**. Changes made to regular expressions in this dialog are copied to the process class definition page. However, the changes are not saved until you click the **Save** button on that page.

Test and review the process class sequence

The order in which process classes are evaluated is significant. During modeling, each time a process matches a class, the matching process is put into a process set, and then removed from the list of processes that are passed on to the next class in the sequence. New process classes are inserted into the process class sequence automatically, and may not be in the appropriate position in the sequence.

Complete the previous section, and then follow these steps to test and review the process class sequence.

- 1 In the lower-left corner of the tree view, click the Action menu, and select **Test All Process Classes Regular Expressions**.
- 2 Select the process names in the previous section, and copy them into a paste buffer.
- 3 In the **Input** area, select all of the existing text, and then paste the process names from the buffer.
- 4 At the bottom-left corner of the dialog, click **Test**.
- 5 The number of processes matched and process sets created in this test should be identical to the results of testing the process class alone. If they are not, follow these steps to adjust the process class sequence.
 - a In the **Test Process Class Regular Expressions** dialog, click **Done**.
 - b From the Action menu, select **Change Sequence**.

Figure 59: Sequence dialog

- c Scroll through the list of process classes, and then select the class to move.
- d Drag the class to an earlier (higher) location in the sequence.
- e Click **Submit**.
- 6 Re-open the **Test Process Class Regular Expressions** dialog, and re-test the sequence.

Test the process class on a host

Process sets are created during modeling. To test a process class, choose a device host that is configured for SNMP or SSH access, and model it manually.

Note For more information about device support for process monitoring, refer to the release notes.

Complete the previous section, and then follow these steps to test the process class on a host.

- 1 Navigate to **Infrastructure > Devices**.
- 2 Select a host that is configured for SNMP or SSH access, and is running process that match the new class. For example, the list of processes used in this section is collected from a VirtualBox virtual appliance downloaded from the [Oracle Technology Network](#).
- 3 Open the host's **Overview** page. From the Action menu, select **Model Device**. When modeling completes, the **OS Processes** section of the tree view is updated to include the new process sets.
- 4 Navigate to **Infrastructure > Processes**.
- 5 In the tree view, select the **DB daemons** class.

Figure 60: Process class page with process sets

The screenshot shows the 'Process Class' configuration page for the 'DB daemons' class. It includes sections for 'Process Class Name' (DB daemons), 'Description' (Database daemons), 'Matching Rules' (including 'Include processes like' and 'Exclude processes like'), 'Process Count Threshold' (Minimum and Maximum values), and various monitoring options like 'Enable Monitoring?' (Set Local Value: Yes, Inherit Value "Yes" from Processes), 'Send Event on Restart?' (Set Local Value: Yes, Inherit Value "No" from Processes), 'Failure Event Severity' (Set Local Value: Error, Inherit Value "Error" from Processes), 'Lock Process Components?' (Set Local Value: Unlocked, Inherit Value "0" from Processes), and 'Send an event when action is blocked?' (Set Local Value: No, Inherit Value "No" from Processes). At the bottom, a table displays 13 rows of generated process sets, each named 'DB [orc] daemons starting with [letter]' and having a value of '10.87.110.103'.

| Name | Device | Min Threshold | Max Threshold | Monitored | Status |
|------------------------------------|---------------|---------------|---------------|-----------|---|
| DB [orc] daemons starting with [c] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [d] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [g] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [l] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [l] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [m] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [n] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [p] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [q] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [r] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [s] | 10.87.110.103 | | | true | Up U |
| DB [orc] daemons starting with [v] | 10.87.110.103 | | | true | Up U |

The process sets found on the host are displayed in the list at the bottom of the page.

Process class options

The process class page includes the options described in the following sections.

Process Count Threshold

You may set minimum and maximum values for the number of process instances included in a process set. The threshold values apply to all of the process sets generated by a class. The minimum and maximum values are inclusive. That is, if the minimum is 3 and the maximum is 5, then 3, 4, and 5 are all valid process instance counts.

You may define a threshold as an exclusive range. If the minimum is 5 and the maximum is 3, then 4 is an invalid process instance count.

Monitoring Options

■ Enable Monitoring (zMonitor)

To disable monitoring for all process sets generated by this class, set the local value to No.

■ Send Event on Restart (zAlertOnRestart)

To send an event when monitoring restarts, set the local value to Yes.

■ Failure Event Severity (zFailSeverity)

To specify a non-default event severity for the failure of process sets generated by this class, set a local value.

Process Set Locking

Process sets are generated at modeling time. Since modeling recurs regularly, a given modeling run may result in a missing process set, due to a transient absence of one or more process instances. To prevent this from happening, set the local value of the **Lock Process Components?** (**zModelerLock**) field to one of the following options.

- **Lock from Deletes**

Prevent deletion of process sets generated by this process class if modeling returns empty sets.

- **Lock from Updates**

Prevent updates to process sets generated by this process class if modeling returns new process sets.

The final option, **Send an event when action is blocked?** (**zSendEventWhenBlockedFlag**), is used only when a process set is locked. If you lock the process sets of a class, you may set the local value of this field to Yes, and an event will be created when a process set would have been either deleted or updated during modeling.

Monitoring IP services

The IP Services page (**Infrastructure > IP Services**) lets you manage and monitor IP services that are running on your network.

Figure 61: IP Services

| Name | Port | Count |
|---------------|------|-------|
| 1cli-smcs | 3091 | 0 |
| 3com-amp3 | 629 | 0 |
| 3com-net-mgmt | 2391 | 0 |
| 3Com-nsd | 1742 | 0 |
| 3com-tsmux | 106 | 0 |
| 3com-webview | 2339 | 0 |
| 3comfaxrpc | 3446 | 0 |
| 3d-nfsd | 2323 | 0 |
| 3ds-lm | 1538 | 0 |
| 3i-h1 | 1511 | 0 |
| 3m-image-lm | 1550 | 0 |

| Name | Device | Monitored | Status |
|-------------|-------------|-----------|--------|
| 3ds-lm | 3ds-lm | Yes | OK |
| 3i-h1 | 3i-h1 | Yes | OK |
| 3m-image-lm | 3m-image-lm | Yes | OK |

The tree view lists all monitored IP services. Filter this list by using the active search area at the top of the view.

The details area shows:

- Service class description
- TCP port

- Associated service keys

To add or change details for a service class, enter or change information, and then click **Save**.

The lower section of the page lists currently running services in this class (by device), and shows their monitoring status. You can also display Configuration Properties by selecting that from the drop-down list.

Enabling IP service monitoring

You can choose to monitor:

- Individual services
- Service classes

When monitoring a service class, you can choose not to monitor one or more individual services in the class. For example, the SMTP service class is monitored by default, but may not be a critical service on some devices. In this case, you can disable its monitoring on those devices.

Note If a service is configured to listen only on local host (127.0.0.1), then it is not monitored by default.

Note When adding a new IP service that uses a port value higher than 1024, you need to increase the value of `zIpServiceMapMaxPort` to a number higher than the port you are monitoring.

To enable monitoring for a service class or service:

- 1 In the tree view, select the service class or service to monitor.
- 2 Make one or more selections:

- **Enable Monitoring (zMonitor)** - By default, Inherit Value is selected for all services below the IPService node. When selected, the service class or service will inherit monitoring choices from its parent. If you want to individually enable monitoring choices, select the Set Local Value option, and then select a value.
- **Failure Event Severity (zFailSeverity)** - By default, Inherit Value is selected for all services below the IPService node. When selected, the service class or service will inherit severity level choices from its parent. If you want to individually select severity levels, select the Set Local Value option, and then select a value.

- 3 Click **Save** to save your choices.

Using the predefined /Server/Scan device class

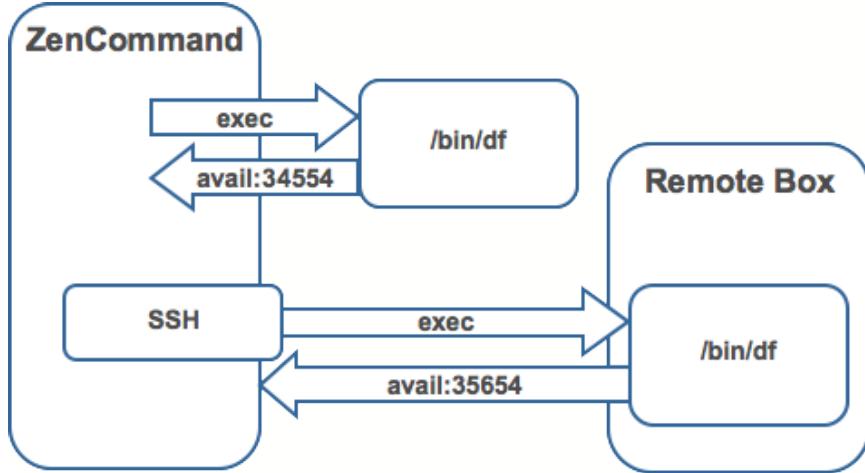
The predefined /Server/Scan device class is an example configuration for monitoring TCP services on devices using a port scan. If you have a device that you want to monitor for service availability alone, you can place it under this device class. The system will not collect performance data for devices in this class.

Monitoring Windows Services

The Windows Services page (**Infrastructure > Windows Services**) has been updated to use the WinService monitoring template.

Monitoring using ZenCommand

Zenoss Core has the ability to run Nagios® and Cacti plug-ins though the ZenCommand process. ZenCommand can run plugins locally and remotely by using a native SSH transport. When run, the system tracks the return code of each plug-in and then creates events with plug-in output. Additionally, it can track performance information from a plug-in.

Figure 62: Running ZenCommand

Plugin format for ZenCommands

Nagios® plugins are configured by using a command template.

A template named “Device” will bind to all devices below the template definition. Within each template is a list of commands that will run. The commands can be any program that follows the Nagios® plug-in standard. Inputs are command line arguments; output is the first line of stdout, plus a return code.

Note Zenoss Core return codes differ from Nagios® return codes, as follows:

| Value | Zenoss Core | Nagios |
|-------|---------------|----------|
| 0 | Clear | OK |
| 1 | Data Source | WARNING |
| 2 | Data Source+1 | CRITICAL |
| 3 | Data Source | UNKNOWN |

For comprehensive information about Nagios® plugins, refer to the [Nagios Plugin Development Guidelines](#).

A Nagios® command has several fields:

- name – Specifies the name of the command object.
- enabled – Indicates whether this command should be used on a given device.
- component – Specifies the component name to use when zencommand sends events to the system.
- event class – Specifies the event class to use when sending events to the system.
- severity – Sets the default severity to use when sending events to the system.
- cycle time – Sets the frequency a command should be run (in seconds).
- command template – Specifies the command to run.

The command template string is built by using Zope TALES expressions. Several variables are passed when evaluating the template. They are:

- zCommandPath – Path to the zencommand plug-ins on a given box it comes from the configuration property zCommandPath. zCommandPath is automatically added to a command if a path is absent from the beginning of the command.
- devname – Device name of the device against which the command is being evaluated.
- dev – Device object against which the command is being evaluated.

- here – Context of evaluation. For a device, this is equivalent to dev for a component (such as a file system or interface). This is the component object.
- compname – If this command evaluates against a component, specifies its name as a string.
- now – Current time.

Template values are accessed like shell variables. They are the same as the expression syntax used in the appendix titled TALES Expressions (in this guide).

Testing ZenCommands

You can test ZenCommand data sources by using the `zentestcommand` shell script.

- 1 Log in to the Control Center host as a user with `serviced` CLI privileges.
- 2 Attach to the `zenhub` service.

```
serviced service attach zenhub
```

- 3 Change to the `zenoss` user.

```
su - zenoss
```

- 4 Run the `zentestcommand` script.

```
zentestcommand -d DeviceID --datasource=DataSourceName
```

where `DeviceID` is the ID of the device on which you want to run the command, and `DataSourceName` is the name of a data source on a template associated with the device.

The `zentestcommand` script prints the results of the command to standard output.

SNMP monitoring

OID represent the data points where the data for the graphs comes from. Sometimes the reason that a graph is not appearing is because the OID for the particular graph is not valid for the device. You can test this validity using the command line to see if you can return a value. To test the validity of an OID data point giving performance data:

- 1 Log in to the Control Center host as a user with `serviced` CLI privileges.
- 2 Attach to the `Zenoss.core` service.

```
serviced service attach Zenoss.core
```

- 3 Change to the `zenoss` user.

```
su - zenoss
```

- 4 Run the `snmpget` command for one of the OIDs

In this case, use the command:

```
$ snmpget -v 2c -cpublic build .1.3.6.1.4.1.2021.4.14.0
```

where, `build` should be a valid server/ip address

If the OID is valid it will return a value.

Here are some basic SNMP commands to gather certain information.

- a Walk a basic system MIB.

```
snmpwalk -v 2c -cpublic <device_name_or_ip_address> system
```

- b Walk an interface description

```
snmpwalk -v 2c -cpublic <device_name_or_ip_address> ifDescr
```

- c Get a single value.

```
snmpget -v 2c -cpublic <device_name_or_ip_address> ifDescr.2
```

- d Detailed description of an OID value.

```
snmptranslate -Td RFC1213-MIB::ifDescr
```

- e Convert a name to a raw OID.

```
snmptranslate -On RFC1213-MIB::ifDescr
```

- f Convert a raw OID to a short name

```
snmptranslate -OS .1.3.6.1.2.1.2.2.1.2
```

Monitoring devices remotely through SSH

You can monitor devices remotely through SSH. Follow the steps in the following sections to set up remote monitoring.

Changing Zenoss Core to monitor devices remotely using SSH

You must edit system properties for the group where you want to collect remote information using SSH.

- 1 Navigate to the device class path that you want to monitor remotely. You can apply this monitoring for a device or a device class path.
- 2 Change the configuration properties value for the group. After selecting the device class, click **Details**, and then select **Configuration Properties**.
- 3 On the **Configuration Properties** page, change the properties that are listed in the following table.

The table includes sample values set up for remote devices. These have a pre-shared key (with no password) set up from the collector to the remote boxes. It also can use password authorization if the password is entered into zCommandPassword.

| Configuration properties | Sample value |
|--------------------------|---|
| zCollectorPlugins | snmp portscan |
| zCommandPassword | The SSH password for the remote machine. |
| zCommandPath | The path to zenplugin.py |
| zCommandUsername | The SSH user name for the remote machine. |

| Configuration properties | Sample value |
|--------------------------|--------------|
| zSnmpMonitorIgnore | True |

- 4 Two passes are required for full modeling. The first pass obtains the platform type (so that the system knows which plugins to run). The second pass provides detailed data on interfaces and file systems.

- a Log in to the Control Center host as a user with serviced CLI privileges.
- b Attach to the zenmodeler service.

```
serviced service attach zenmodeler
```

- c Change to the zenoss user.

```
su - zenoss
```

- d Run the zenmodeler command.

```
$ zenmodeler run -d DeviceName
```

where *DeviceName* is the fully qualified device name.

- e Run the zenmodeler command a second time to use the plugins the command gathered on the first pass.

Using the predefined /Server/Cmd device class

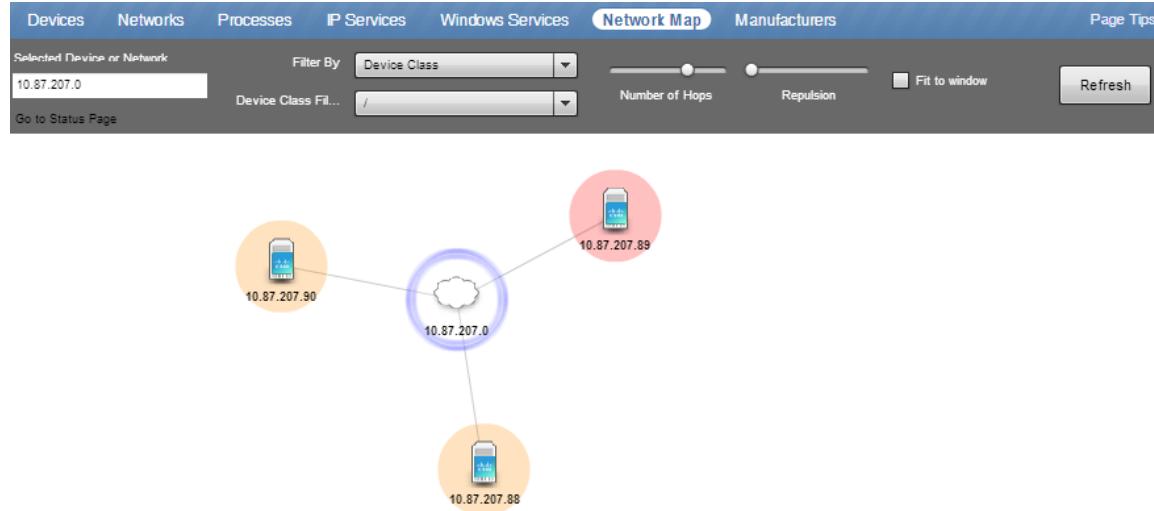
The /Server/Cmd device class is an example configuration for modeling and monitoring devices using SSH. The configuration properties have been modified (as described in the previous sections), and device, file system, and Ethernet interface templates that gather data over SSH have been created.

You can use this device class as a reference for your own configuration; or, if you have a device that needs to be modeled or monitored via SSH/Command, you can place it under this device class to use the pre-configured templates and configuration properties. You must set the zCommandUsername and zCommandPassword properties to the appropriate SSH login information for each device.

Network map

The network map represents your network's layer 3 topology. From the map, you can quickly determine the status of each device by its highlighted color.

To access the network map, select **INFRASTRUCTURE > Network Map**.

Figure 63: Network Map

Choosing the network to display

The network displayed is configured for each user. From user preferences, modify Network Map Start Object to select a network, and then click **Save**.

Viewing device and network details

Double-click a device or network icon in the map to focus on it. Focusing on a node:

- Centers it on the map
- Shows links from the node, based on the number of hops selected

Alternatively, you can type the name or IP address of a device or network in the Selected Device or Network field, and then click **Refresh** to focus on that node.

To see detailed information about a device or network, select it in the map, and then click **Go to Status Page**.

Note When you select a node, the network map displays only the links that are currently loaded into the map. It does not download and display new link data.

Loading link data

To load link data for a node:

- 1 Double-click the node on the map to focus on it, or enter the device name or IP address in the Selected Device or Network field.
- 2 Select the number of hops to download and display by sliding the counter.
- 3 Click **Refresh**.

Filtering by device type

You can filter the devices that appear on the network map. To do this, select a filter from the Device Class Filter list of options. For example, to show only Linux devices on the map, select /Server/Linux from the list of options, then click **Refresh**.

Adjusting viewable hops

You can adjust the number of hops that appear on the network map. Use the Number of Hops slider, which adjusts the number of hops from 1 to 4.

Adjusting the network map

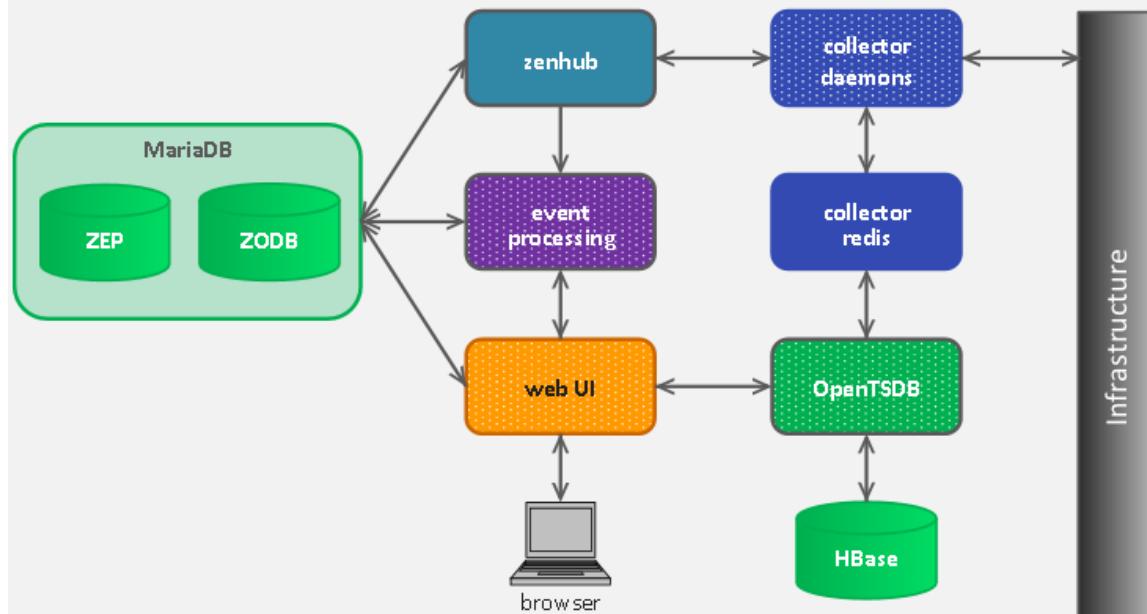
Use the Repulsion slider to expand or contract the icons that appear on the map. Move the slider right to expand the icons, or left to contract them.

Select the **Fit to Window** option to bring all displayed icons into the viewable area.

Performance monitoring

Zenoss Core stores device and daemon performance metrics directly in OpenTSDB, a time series database that runs on top of an HBase instance. Writing directly to OpenTSDB eliminates the need for RRD files to be stored on the collectors. The following image shows how the collector daemons fit into the data collection portion of the Zenoss Core architecture.

Figure 64: Data collection simplified architecture



Zenoss Core uses the following methods to monitor performance metrics of devices and device components:

- **ZenPerfSNMP-** Collects data through SNMP from any device that is correctly configured for SNMP monitoring.
- **ZenWinPerf-** ZenPack that allows performance monitoring of Windows servers.
- **ZenCommand-** Logs in to devices (by using telnet or ssh) and runs scripts to collect performance data.
- **Other ZenPacks-** Collect additional performance data. Examples include the ZenJMX ZenPack, which collects data from enterprise Java applications, and the HttpMonitor ZenPack, which checks the availability and responsiveness of Web pages.

Regardless of the monitoring method, the system stores performance monitoring configuration information in *monitoring templates*.

Monitoring templates

Monitoring templates determine how the system collects performance data for devices and device components. You can define monitoring templates for device classes and individual devices.

Templates comprise the following types of objects:

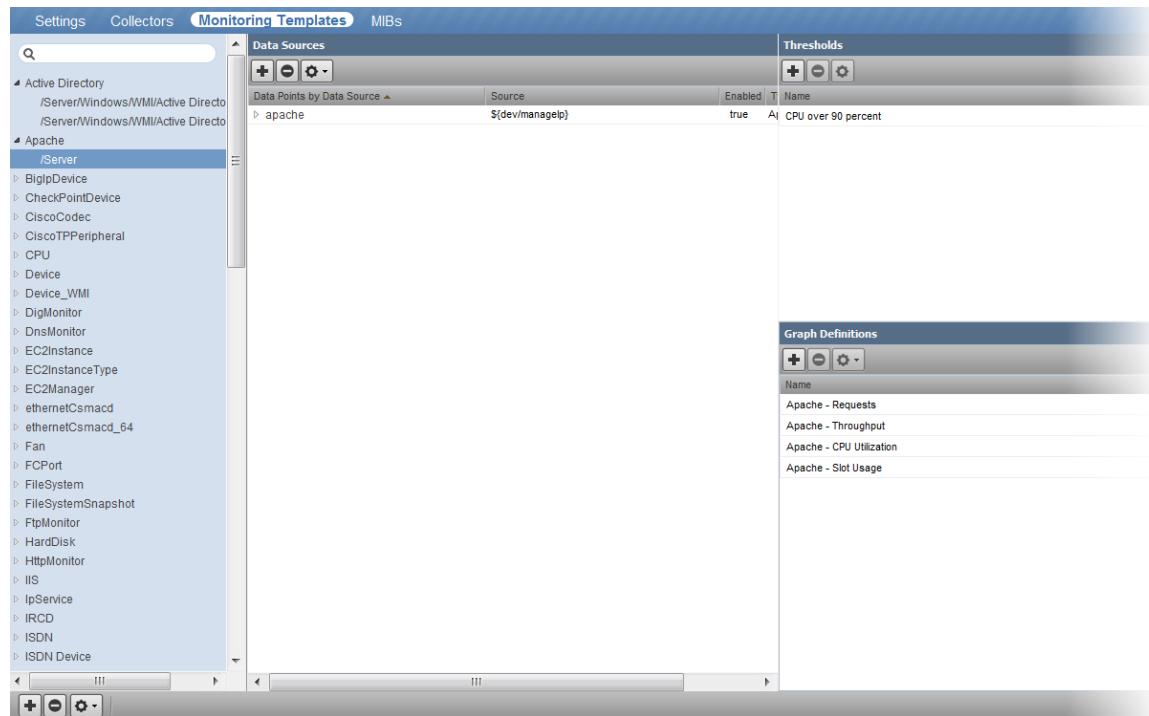
- **Data sources**- Specify the exact data points to collect and the collection method to use.
- **Thresholds**- Define expected bounds for collected data, and specify events to be created if the data does not match those bounds.
- **Graph definitions**- Describe how to graph the collected data on the device or device components.

Before the system can collect performance data for a device or component, it must use the *template binding* process to determine which monitoring templates apply.

Viewing monitoring templates

From the main navigation menu, choose **Advanced > Monitoring Templates**.

Figure 65: Monitoring template for Load Average graph



Template binding

The system determines the list of template names that apply to a device or component. For device components, the list is defined by the meta type of the component (for example, FileSystem, CPU, or HardDisk). For devices, the list is defined by the `zDeviceTemplates` configuration property.

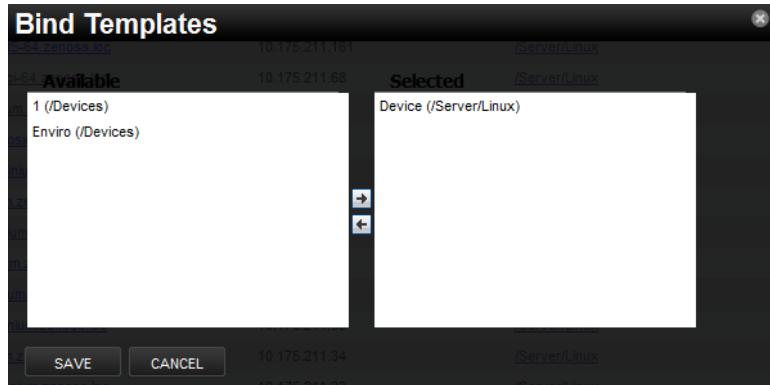
After defining the list, the system locates templates that match the names on the list. For each name, it searches the device and then searches the device class hierarchy. Zenoss Core uses the lowest template in the hierarchy

that it can locate with the correct name, ignoring others of the same name that might exist further up the device class hierarchy.

Editing templates bound to a device

- 1 From the main navigation menu, select **Infrastructure**.
- 2 From the device list in the left pane, choose a device.
- 3 Below the device list, click the context-sensitive actions menu and choose **Bind Templates**.
- 4 In the **Bind Templates** dialog box list of available templates, move one or more templates to the selected list.

Figure 66: Bind Templates dialog box



- 5 Click **Save**.

The templates are bound to the selected device.

Data sources

Data sources specify which data points to collect and how to collect them. Each monitoring template comprises one or more data sources. The system provides the following built-in data source types. ZenPacks provide other data source types.

- **SNMP**- Define data to be collected via SNMP by the ZenPerfSNMP daemon. They contain one additional field to specify which SNMP OID to collect. (Many OIDs must end in .0.) Because SNMP data sources specify only one performance metric, they contain a single data point.
- **Command**- Specify data to be collected by a shell command that is executed on the Zenoss Core server or on a monitored device. The ZenCommand daemon processes COMMAND data sources. A COMMAND data source can return one or more performance metrics, and usually has one data point for each metric.

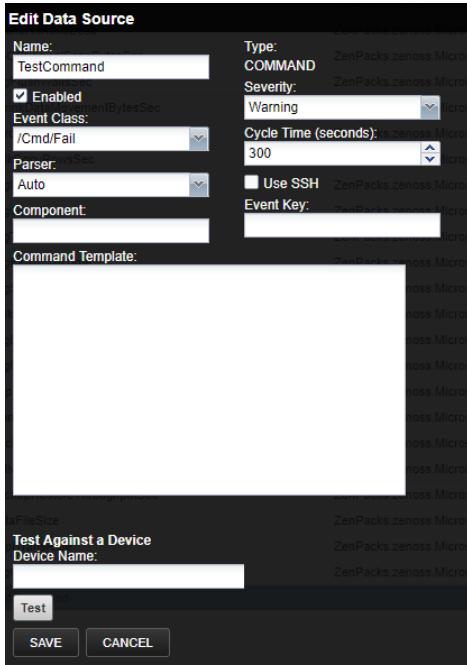
Shell commands that are used with COMMAND data sources must return data that conforms to the Nagios plug-in output specification. For more information, see [Monitoring using ZenCommand](#) on page 91.

Adding a data source

To add a data source to a monitoring template:

- 1 Select **Advanced** from the Navigation menu, and then select **Monitoring Templates**.
- 2 In the tree view, select the monitoring template to which you want to add a data source.
- 3 In the Data Sources area, click the **Add** button. The Add Data Source dialog box appears.
- 4 Enter a name for the data source and select the type, and then click **Submit**. The data source is added to the list in the Data Sources area.

- 5 Double-click the data source in the list. The Edit Data Source dialog box appears. This dialog box will vary based on the type of data source you are editing. For example, the following is the Edit Data Source dialog box for a COMMAND data source.



- 6 Enter or select values to define the data source.

In the example of the COMMAND data source, there are a few things to note in particular. Be sure to select the **Use SSH** check box. If you do not select this, the commands will only be run locally on the collector assigned to the device that has the monitoring template bound to it. The **Command Template** field is where you enter command to run. Be aware that any script you enter here is run through TALES before being run. You may have to escape certain characters. For more information, see [TALES expressions](#) on page 205.

Data points

Data sources can return data for one or more performance metrics. Each metric retrieved by a data source is represented by a data point.

You can define data points to data sources with all source types except SNMP and VMware. Because these data source types each rely on a single data point for performance metrics, additional data point definition is not needed.

To add a data point to a data source:

- 1 From the Navigation menu, select **Advanced > Monitoring Templates**.
- 2 In the Data Sources area, highlight the row containing a data source.
- 3 Select **Add Data Point** from the Action menu. The Add Data Point dialog box appears.
- 4 Enter a name for the data point, and then click **Submit**.

Note For COMMAND data points, the name should be the same as that used by the shell command when returning data.

- 5 Double-click the newly added data point to edit it. Enter information or make selections to define the data point:
- **Name**- Displays the name you entered in the Add a New DataPoint dialog.

- **RRD Type-** Specify the RRD data source type to use for storing data for this data point. Available options are:
 - **COUNTER-** Saves the rate of change of the value over a step period. This assumes that the value is always increasing (the difference between the current and the previous value is greater than 0). Traffic counters on a router are an ideal candidate for using COUNTER.
 - **GAUGE-** Does not save the rate of change, but saves the actual value. There are no divisions or calculations. To see memory consumption in a server, for example, you might want to select this value.

Note Rather than COUNTER, you can define a data point using DERIVE and with a minimum of zero. This approach creates the same conditions as COUNTER, with one exception. Because COUNTER is a "smart" data type, it can wrap the data when a maximum number of values is reached in the system. An issue can occur when there is a loss of reporting and the system (when looking at COUNTER values) thinks it should wrap the data. This creates an artificial spike in the system and creates statistical anomalies.

- **DERIVE-** Same as COUNTER, but additionally allows negative values. If you want to see the rate of change in free disk space on your server, for example, then you might want to select this value.
- **ABSOLUTE-** Saves the rate of change, but assumes that the previous value is set to 0. The difference between the current and the previous value is always equal to the current value. Thus, ABSOLUTE stores the current value, divided by the step interval.
- **Create Command-** Enter an RRD expression used to create the database for this data point. If you do not enter a value, then the system uses a default applicable to most situations.
- **RRD Minimum-** Enter a value. Any value received that is less than this number is ignored.
- **RRD Maximum-** Enter a value. Any value received that is greater than this number is ignored.

6 Click **Save** to save the defined data point.

Data point aliases

Performance reports pull information from various data points that represent a metric. The report itself knows which data points it requires, and which modifications are needed, if any, to put the data in its proper units and format.

The addition of a data point requires changing the report.

Figure 67: CPU Utilization Report**CPU Utilization Report**

*For the Linux data point:
Divide the total CPU idle percentage by the number of cores, and subtract from 100.*

*For the Windows data point:
Report the CPU utilization unmodified.*

Adding ESX requires adding another rule to fetch it and divide the value by 100, because the utilization is reported on a scale from 0 to 10000.

| Device | Utilization |
|---------------------|-------------|
| linux.example.com | 20% |
| windows.example.com | 34% |

Total CPU idle = 320%

linux.example.com
(Linux with 4 cores)

CPU utilization = 34%

windows.example.com
(Windows)

CPU utilization = 4322

esx.example.com
(ESX via VI SDK)

To allow for more flexibility in changes, some reports use *data point aliases*. Data point aliases group data points so they can be more easily used for reporting. In addition, if the data points return data in different units, then the plugin can normalize that data into a common unit.

An alias-based report looks up the data points that share a common alias string, and then uses them. This approach allows you to add data points without changing the report.

Figure 68: Alias-based CPU Utilization Report**CPU Utilization Report (using aliases)**

*Get values for all data points with the `cpu_pct` alias.
Adding ESX to the report is now just a matter of applying the alias `cpu_pct` with a formula for dividing by 100.*

| Device | Utilization |
|---------------------|-------------|
| linux.example.com | 20% |
| windows.example.com | 34% |

Total CPU idle = 320%

alias: `cpu_pct`
formula: divide idle by number of CPUs and subtract from 100.linux.example.com
(Linux)

CPU utilization = 34%

alias: `cpu_pct`
formula: nonewindows.example.com
(Windows)

CPU utilization = 4322

esx.example.com
(ESX via VI SDK)

In the simplest cases, data from the target data points are returned in units expected by a report. For cases in which data are not returned in the same units, an alias can use an associated formula at the data point. For example, if a data point returns data in kilobytes, but the report expects data in bytes, then the formula multiplies the value by 1024.

Alias formula evaluation

The system evaluates the alias formula in three passes.

Reverse polish notation

When complete, the alias formula must resolve to a Reverse Polish Notation (RPN) formula. For the simple conversion of kilobytes into bytes, the formula is:

```
1024, *
```

For more information about RPN formulas, refer to the [documentation](#).

Using TALES expressions in alias formulas

For cases in which contextual information is needed, the alias formula can contain a TALES expression that has access to the device as context (labeled as "here"). The result of the TALES evaluation should be an RRD formula.

For example, if the desired value is the data point value divided by total memory, the formula is:

```
 ${here/hw/totalMemory}, /
```

For more information about TALES, see [TALES expressions](#) on page 205.

Using Python in alias formulas

You also can embed full Python code in an alias formula for execution. The code must construct a string that results in a valid RRD formula. To signal the system to evaluate the formula correctly, it must begin with:

```
 __EVAL:
```

Using the same example as in the previous section (division by total memory), the formula is:

```
 __EVAL:here.hw.totalMemory + ", /"
```

Adding a data point alias

To add an alias to a data point:

- 1 Navigate to a data source on a monitoring template.
- 2 Double-click a data point in the list to edit it. The Edit Data Point dialog appears.
- 3 Enter the alias name and the formula.

Note If the data point returns values in the preferred units, then leave the Formula value blank.

Figure 69: Add data point alias

The dialog box shows the configuration of a data point alias. The 'Alias:' field contains 'memAvail__megabyte'. The 'ID / FORMULA' field contains '1024,*,1024,*'. There is a 'DELETE' button next to the formula field. Below these are 'ADD', 'SAVE', and 'CANCEL' buttons.

- 4 Click Save.

Reports that use aliases

For information about reports that use aliases, refer to the chapter titled "Reporting."

The following table shows performance reports that use aliases, and the aliases used. To add data points to a report, add the alias, and then ensure the values return in the expected units.

CPU utilization

| Alias | Expected units |
|-----------------|----------------|
| loadAverage5min | Processes |
| cpu_pct | Percent |

Thresholds

Thresholds define expected bounds for data points. When the value returned by a data point violates a threshold, the system creates an event. There are several threshold types available:

- MinMax
- ValueChange
- CiscoStatus
- PredictiveThreshold

There are many thresholds already defined in the system. You can see all the defined thresholds in the Defined Thresholds report which can be accessed on the **REPORTS > Enterprise Reports** menu.

The following sections describe each type of threshold and how to create a new one and edit an existing one.

Additional threshold types may be provided through installed ZenPacks.

MinMax threshold

MinMax thresholds inspect incoming data to determine whether it exceeds a given maximum or falls below a given minimum. You can use a MinMax threshold to check for these scenarios:

- *The current value is less than a minimum value.* To do this, you should set only a minimum value for the threshold. Any value less than this number results in creation of a threshold event.
- *The current value is greater than a maximum value.* To do this, you should set only a maximum value for the threshold. Any value greater than this number results in creation of a threshold event.
- *The current value is not a single, pre-defined number.* To do this, you should set the minimum and maximum values for the threshold to the same value. This will be the only "good" number. If the returned value is not this number, then a threshold event is created.

- *The current value falls outside a pre-defined range.* To do this, you should set the minimum value to the lowest value within the good range, and the maximum value to the highest value within the good range. If the returned value is less than the minimum, or greater than the maximum, then a threshold event is created.
- *The current value falls within a pre-defined range.* To do this, you should set the minimum value to the highest value within the bad range, and the maximum value to the lowest value within the bad range. If the returned value is greater than the maximum, and less than the minimum, then a threshold event is created.

ValueChange threshold

ValueChange thresholds inspect incoming data to determine whether a status change has occurred and if so issues an event based on the defined severity.

Adding thresholds

To define a threshold for a data point:

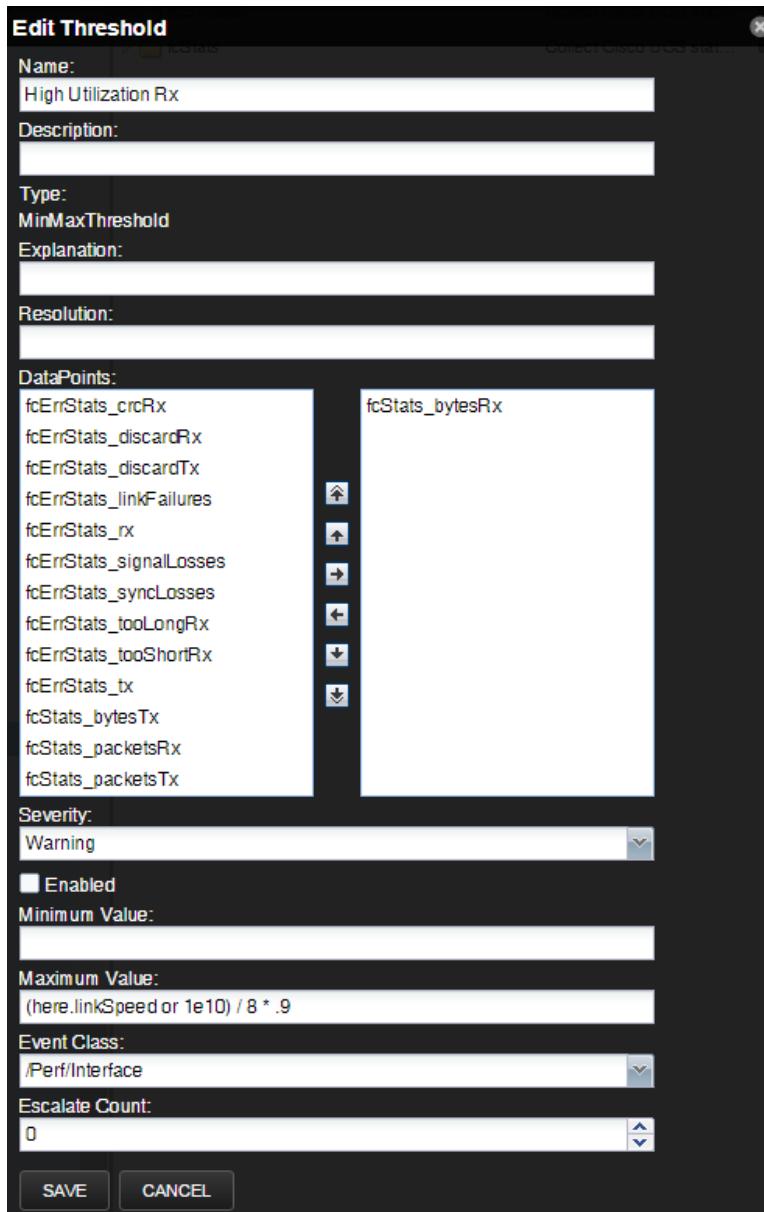
- 1 From the Navigation menu, select **ADVANCED > Monitoring Templates**.
- 2 Click on the template that contains the data point you want to use in your threshold. The Data Sources window is populated with folders containing all the data points being collected.
- 3 Select the data point by opening the appropriate folder in the data source and clicking on the data point row.
- 4 In the Thresholds area, click the **Add** icon. The Add Threshold dialog box appears.
- 5 Enter a name and select the threshold type, then click **Add**. The threshold name and type is displayed in the Thresholds window.

Editing MinMax thresholds

The threshold must be created by the Add Threshold functionality or already be defined in the system before you can edit it.

To edit a MinMax threshold:

- 1 Double-click the threshold in the list. The Edit Threshold dialog box appears.

Figure 70: Edit Threshold

2 Enter or select values to define the threshold:

- **Name**- Displays the value for the ID you entered on the Add a New Threshold dialog.
- **Description**- Description of the threshold that you entered on the Add a New Threshold dialog. The description is included in each event that is created from this threshold.
- **Type**- Type of threshold that you selected. You cannot change the type of threshold. If you want a different type of threshold, you need to create a new one and assign the correct type.
- **Explanation**- Information field where a user can enter information about what the event means. This field is included in each event that is created from this threshold
- **Resolution**- Information field where a user can enter information about what to do to resolve the event. This field is included in each event that is created from this threshold.
- **Data Points**- Select one or more data points to which this threshold will apply and click the right-arrow button to move them to the selected column.
- **Severity**- Select the severity level of the first event triggered when this threshold is breached.

- **Enabled**- Select the check box to enable the threshold, or clear the check box to disable it.
- **Minimum Value**- If this field contains a value, then each time one of the select data points falls below this value an event is triggered. This field may contain a number or a Python expression. When using a Python expression, the variable *here* references the device or component for which data is being collected. For example, a 90% threshold might be specified as:

```
(here.linkSpeed or 1e10) / 8 * .9
```

The division by 8 is needed because interface speed frequently is reported in bits/second, where the performance data is bytes/second.

- **Maximum Value**- If this field contains a value, then each time one of the selected data points goes above this value an event is triggered. This field may contain a number or a Python expression.
- **Event Class**- Select the event class of the event that will be triggered when this threshold is breached.
- **Escalate Count**- Enter the number of consecutive times this threshold can be broken before the event severity is escalated by one step. A value of zero (0) indicates that the severity will not escalate.

- 3 Click **Save** to confirm the edits.

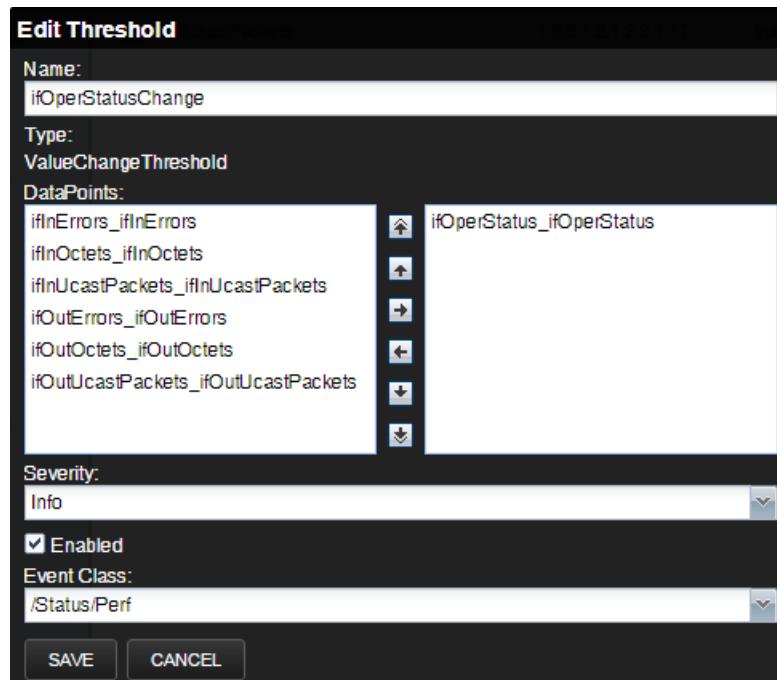
Editing ValueChange thresholds

The threshold must be created by the Add Threshold functionality or already be defined in the system before you can edit it.

To edit a ValueChange threshold:

- 1 Double-click the threshold in the list. The Edit Threshold dialog box appears.

Figure 71: Edit Threshold



- 2 Enter or select values to define the threshold:

- **Name**- Displays the value for the ID you entered on the Add a New Threshold dialog.
- **Type**- Type of threshold that you selected. You cannot change the type of threshold. If you want a different type of threshold, you need to create a new one and assign the correct type.

- **Data Points**- Select one or more data points to which this threshold will apply and click the right-arrow button to move them to the selected column. When the data point changes status, an event will be triggered.
 - **Severity**- Select the severity level of the first event triggered when this threshold is breached.
 - **Enabled**- Select the check box to enable the threshold, or clear the check box to disable it.
 - **Event Class**- Select the event class of the event that will be triggered when this threshold is breached.
- 3 Click **Save** to confirm the edits.

Performance Graphs

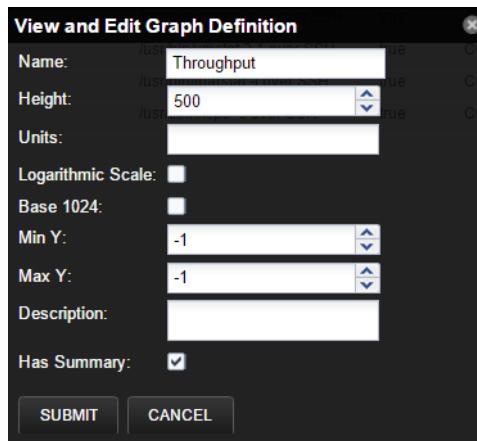
You can include any of the data points or thresholds from a monitoring template in a *performance graph*.

Graphs are no longer static images, but are rendered using the NVD3.js JavaScript library.

To define a graph:

- 1 From the navigation menu, select **ADVANCED > Monitoring Templates**.
- 2 In the left column, select the monitoring template in which you want to create a graph.
- 3 In the Graph Definitions area, click the **Add** icon. The Add Graph Definition dialog box appears.
- 4 Enter a name for the graph, and then click **Submit**.
- 5 Double-click the graph name in the list to edit it. Enter information or select values to define the graph:
 - **Name**- Optionally edit the name of the graph you entered in the Add a New Graph dialog. This name appears as the title of the graph.
 - **Height**- Enter the height of the graph, in pixels.
 - **Units**- Enter a label for the graph's vertical axis.
 - **Logarithmic Scale**- Select True to specify that the scale of the vertical axis is logarithmic. Select False (the default) to set the scale to linear. You might want to set the value to True, for example, if the data being graphed grows exponentially. Only positive data can be graphed logarithmically.
 - **Base 1024**- Select True if the data you are graphing is measured in multiples of 1024. By default, this value is False.
 - **Min Y**- Enter the bottom value for the graph's vertical axis.
 - **Max Y**- Enter the top value for the graph's vertical axis.
 - **Description**- Enter a description of the graph.
 - **Has Summary**- Select True (default) to display a summary of the data's current, average, and maximum values at the bottom of the graph.

Figure 72: Graph Definition



- 6 Click **Submit** to save the graph.

Graph points

Graph points represent each data point or threshold that is part of a graph. You can add any number of graph points to a graph definition by adding data points or thresholds.

From the Graph Definitions area of the Monitoring Templates page:

- 1 From the Action menu, select **Manage Graph Points**. The Manage Graph Points dialog box appears.
- 2 From the Add menu, add a data point, threshold, or custom graph point.
- 3 Select values, and then click **Submit**.

Note Thresholds are always drawn before other graph points.

Re-sequencing graph points

To re-sequence graph points, drag a graph point row in the Manage Graph Points dialog box. (Click-and-drag from an "empty" part of the row.)

DataPoint graph points

DataPoint graph points draw the value of data points from the template on a graph.

Adding DataPoint graph points

To define a DataPoint graph point:

- 1 From the **Add** menu on the Manage Graph Points dialog, select **Data Point**. The Add Data Point dialog box appears.
- 2 Select one or more data points defined in this template. One data point graph point is created for each data point you select from the list.
- 3 Optional: Select the **Include Related Thresholds** option. If selected, then any graph points are created for any thresholds that have been applied to the selected data points as well.
- 4 Click **Submit**.

Editing DataPoint graph points

Double-click the name of the graph point to go to its edit page. Enter information or select values to edit the graph point:

- **Name-** This is the name that appears on the Graph Definition page. By default, it appears in the graph legend.
- **Line Type-** Select Line to graph the data as a line. Select Area to fill the area between the line and the horizontal axis with the line color. Select None to use this data point for custom RRD commands and do not want it to be explicitly drawn.
- **Line Width-** Enter the pixel width of the line.
- **Stacked-** If selected, then the line or area is drawn above the previously drawn data. At any point in time on the graph, the value plotted for this data is the sum of the previously drawn data and the value of this data point now. You might set this value, for example, to assess total packets if measuring packets in and packets out.
- **Format-** Specify the RRD format to use when displaying values in the graph summary. For more information about formatting strings, refer to its [documentation](#).
- **RPN-** Optionally enter an RPN expression that alters the value of the data being graphed for the data point. For example, if the data is stored as bits, but you want to graph it as bytes, enter an RPN value of "8,/" to divide by 8. For more information about RPN notation, refer to the [RRDTool RPN tutorial](#).
- **Limit-** Optionally specify a maximum value for the data being graphed.

- **Consolidation-** Specify the RRD function used to graph the data point's data to the size of the graph. Most of the time, the default value of AVERAGE is appropriate.
- **Color-** Optionally specify a color for the line or area. Enter a six-digit hexadecimal color value with an optional two-digit hex value to specify an alpha channel. An alpha channel value is only used if 'stacked' is True.
- **Legend-** Name to use for the data in the graph legend. By default, this is a TALES expression that specifies the graph point name. The variables available in this TALES expression are here (the device or component being graphed) and graphPoint (the graph point itself).
- **Available RRD Variables-** Lists the RRD variables defined in this graph definition. These values can be used in the RPN field.

Editing threshold graph points

Threshold graph points graph the value of thresholds from the template.

To edit a threshold graph point, double-click it in the list:

You can edit values for Name, Color, and Legend for a threshold graph point. Refer to the definitions in the section titled [Editing DataPoint Graph Points](#) for more information.

Performance data retention

Zenoss Core stores all performance data (a.k.a., metrics) in HBase using OpenTSDB. The default retention policy saves performance data for 90 days. To change the default, the time to live (TTL) must be adjusted on the OpenTSDB column families in HBase.

Note TTL is defined in seconds.

Once a TTL value is changed, the data retention will adjust on the next major HBase compaction, which by default is once per day.

Changing the performance data retention time

To change the performance data retention time from the default value of 90 days:

- 1 Log in to the Control Center master host as a user with `sudo` and `docker` privileges.
- 2 Stop the openTSDB writer service.

```
serviced service stop opentsdb/writer
```

- 3 Execute the following command to list all the services and their SERVICEID values. Take note of the SERVICEID for the openTSDB reader service. It will be used as an argument in the following step.

```
serviced service list
```

- 4 Execute the following command, where `$id` is the openTSDB reader SERVICEID and `$ttl` is your TTL value, in seconds.

```
serviced service shell $id /opt/opentsdb/set-opentsdb-table-ttl.sh
$ttl
```

- 5 Start the openTSDB writer service.

```
serviced service start opentsdb/writer
```

Event management

Events, and the graphs generated from performance monitoring, are the primary operational tools for understanding the state of your environment.

Basic event fields

To enter the event management system, an event must contain values for the device, severity, and summary fields. Zenoss Core rejects events that are missing any of these fields.

Basic event fields are as follows:

- Summary
- Device
- Component
- Severity
- Event Class Key
- Event Class
- Collector

Device field

The device field is a free-form text field that allows up to 255 characters. Zenoss Core accepts any value for this field. If the device field contains an IP address or a hostname, then the system will automatically identify and add the event to the corresponding device.

Zenoss Core automatically adds information to incoming events that match a device. Fields added are:

- **prodState** - Specifies the device's current production state.
- **Location** - Specifies the location (if any) to which the device is assigned.
- **DeviceClass** - Classifies the device.
- **DeviceGroups** - Specifies the groups (if any) to which the device is assigned.
- **Systems** - Systems (if any) to which the device is assigned.
- **DevicePriority** - Priority assigned to the device.

For more information about these fields, refer to the following chapters:

- *Production states and maintenance windows* on page 136
- *Organizers and path navigation* on page 140

Status field

The Status field defines the current state of an event. This field is often updated after an event has been created. Values for this numeric field are 0-6, defined as follows:

| Number | Name | Description |
|--------|--------------|--|
| 0 | New | State given to an event when it is initially created in the system. |
| 1 | Acknowledged | State given to an event when a user has acknowledged the event. |
| 2 | Suppressed | State given to an event that has been suppressed via an event transform. |
| 3 | Closed | State given to an event that was closed as the result of a user action. |
| 4 | Cleared | State given to an event that was cleared by a corresponding clear event. |
| 5 | Dropped | State given to an event that was dropped via an event transform. These events are never persisted by the system. |
| 6 | Aged | State given to an event that was automatically closed by the system according to the severity and last seen time of the event. |

Severity field

The Severity field defines the severity of the event. Values for this numeric field are 0-5, defined as follows:

| Number | Name | Color |
|--------|----------|--------|
| 0 | Clear | Green |
| 1 | Debug | Grey |
| 2 | Info | Blue |
| 3 | Warning | Yellow |
| 4 | Error | Orange |
| 5 | Critical | Red |

Summary and message fields

The summary and message fields are free-form text fields. The summary field allows up to 255 characters. The message field allows up to 4096 characters. These fields usually contain similar data.

The system handles these fields differently, depending on whether one or both are present on an incoming event:

- If only summary is present, then the system copies its contents into message and truncates summary contents to 128 characters.
- If only message is present, then the system copies its contents into summary and truncates summary contents to 128 characters.

- If summary and message are both present, then the system truncates summary contents to 128 characters.

As a result, data loss is possible only if the message or summary content exceeds 65535 characters, or if both fields are present and the summary content exceeds 128 characters.

To ensure that enough detail can be contained within the 128-character summary field limit, avoid reproducing information in the summary that exists on other fields (such as device, component, or severity).

Other fields

Events include numerous other standard fields. Some control how an event is mapped and correlated; others provide information about the event.

The following table lists additional event fields.

| Field | Description |
|---------------|--|
| dedupid | Dynamically generated fingerprint that allows the system to perform de-duplication on repeating events that share similar characteristics. |
| component | Free-form text field (maximum 255 characters) that allows additional context to be given to events (for example, the interface name for an interface threshold event). |
| eventClass | Name of the event class into which this event has been created or mapped. |
| eventKey | Free-form text field (maximum 128 characters) that allows another specificity key to be used to drive the de-duplication and auto-clearing correlation process. |
| eventClassKey | Free-form text field (maximum 128 characters) that is used as the first step in mapping an unknown event into an event class. |
| eventGroup | Free-form text field (maximum 64 characters) that can be used to group similar types of events. This is primarily an extension point for customization. Currently not used in a standard system. |
| stateChange | Last time that any information about the event changed. |
| firstTime | First time that the event occurred. |
| lastTime | Most recent time that the event occurred. |
| count | Number of occurrences of the event between the firstTime and lastTime. |
| prodState | Production state of the device, updated when an event occurs. This value is not changed when a device's production state is changed; it always reflects the state when the event was received by the system. |
| agent | Typically the name of the daemon that generated the event. For example, an SNMP threshold event will have zenperfsnmp as its agent. |
| DeviceClass | Device class of the device that the event is related to. |
| Location | Location of the device that the event is related to. |
| Systems | Pipe-delimited list of systems that the device is contained within. |
| DeviceGroups | Pipe-delimited list of systems that the device is contained within. |
| facility | Only present on events coming from syslog. The syslog facility. |
| priority | Only present on events coming from syslog. The syslog priority. |
| ntevid | Only present on events coming from Windows event log. The NT Event ID. |

| Field | Description |
|-------------------|--|
| ownerid | Name of the user who acknowledged this event. |
| clearid | Only present on events in the archive that were auto-cleared. The evid of the event that cleared this one. |
| DevicePriority | Priority of the device that the event is related to. |
| eventClassMapping | If this event was matched by one of the configured event class mappings, contains the name of that mapping rule. |
| monitor | In a distributed setup, contains the name of the collector from which the event originated. |

Details

In addition to the standard fields, the system also allows events to add an arbitrary number of additional name/value pairs to events to give them more context.

De-duplication

Zenoss Core uses an event "de-duplication" feature, based on the concept of an event's fingerprint. Within the system, this fingerprint is the "dedupid." All of the standard events that the system creates as a result of its polling activities are de-duplicated, with no setup required. However, you can apply de-duplicating to events that arrive from other sources, such as syslog, SNMP traps, or a Windows event log.

The most important de-duplication concept is the *fingerprint*. An event's fingerprint (or dedupid) is composed of a pipe-delimited string that contains these event fields:

- device
- component (can be blank)
- eventClass
- eventKey (can be blank)
- severity
- summary (omitted from the dedupid if eventKey is non-blank)

When the component and eventKey fields are blank, a dedupid appears similar to:

```
www.example.com||/Status/Web||4|WebTx check failed
```

When the component and eventKey fields are present, a dedupid appears similar to:

```
router1.example.com|FastEthernet0/1||Perf/Interface|threshName
```

When a new event is received by the system, the dedupid is constructed. If it matches the dedupid for any active event, the existing event is updated with properties of the new event occurrence and the event's count is incremented by one, and the lastTime field is updated to be the created time of the new event occurrence. If it does not match the dedupid of any active events, then it is inserted into the active event table with a count of 1, and the firstTime and lastTime fields are set to the created time of the new event.

The following illustration depicts a de-duplication scenario in which an identical event occurs three times, followed by one that is different in a single aspect of the dedupid fingerprint.

Figure 73: Event de-duplication

| Event | dedupid | count |
|--|-----------------------------------|-------|
|  device: router1 eventClass: /Perf/CPU eventKey: cpuThresh severity: 3 | router1ll/Perf/CPU/cpuThreshold13 | 1 |
|  | router1ll/Perf/CPU/cpuThreshold13 | 2 |
|  | router1ll/Perf/CPU/cpuThreshold13 | 3 |
|  device: router1 eventClass: /Perf/CPU eventKey: cpuThresh severity: 4 | router1ll/Perf/CPU/cpuThreshold14 | 1 |

If you want to change the way de-duplication behaves, you can use an event transform to alter one of the fields used to build the dedupid. You also can use a transform to directly modify the dedupid field, for more powerful cross-device event de-duplication.

Auto-clear correlation

The auto-clearing feature is similar to the de-duplication feature. It also is based on the event's fingerprint. The difference is which event fields make up the fingerprint, and what happens when a new event matches an existing event's fingerprint.

All of the standard events created as a result of polling activities do auto-clearing by themselves. As with de-duplication, you would invoke auto-clearing manually only to handle events that come from other sources, such as syslog, a Windows event log, or SNMP traps.

If a component has been identified for the event, then the auto-clear fingerprint consists of these fields:

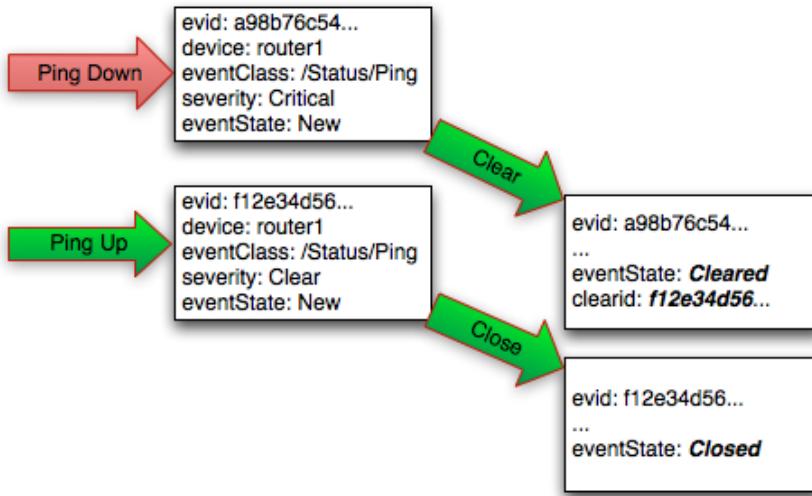
- If component UUID exists:
 - component UUID
 - eventClass (including zEventClearClasses from event class configuration properties)
 - eventKey (can be blank)
- If component UUID does not exist:
 - device
 - component (can be blank)
 - eventKey (can be blank)
 - eventClass (including zEventClearClasses from event class configuration properties)

When a new event comes into the system with a special 0 (Clear) severity, Zenoss Core checks all active events to see if they match the auto-clear fingerprint of the new event. All active events that match the auto-clear fingerprint are updated with a Cleared state, and the clearid field is set to the UUID of the clear event. After a configurable period of time, all events in a closed state (Closed, Cleared, and Aged) are moved from the active events table to the event archive.

If an event is cleared by the clear event, it is also inserted into the active events table with a status of Closed; otherwise, it is dropped. This is done to prevent extraneous clear messages from filling your events database.

The following illustration depicts a standard ping down event and its associated clear event.

Figure 74: Event auto-clear



If you need to manually invoke the auto-clearing correlation system, you can use an event transform to make sure that the clear event has the 0 (Clear) severity set. You also need to ensure that the device, component, and eventClass fields match the events you intend to clear.

Note To prevent inadvertently clearing a wider range of events than intend, avoid making clear events too generic.

Event consoles

Zenoss Core features multiple event consoles that allow you to view and manage events. Each console shows different events subsets, depending on your current context.

Event consoles are:

- **Master-** To access this console, click Events on the Navigation menu. You can view all events from this console.
- **Contextual-** Contextual event consoles are found throughout the system. Each time you see an Events selection for a device, device organizer, component, or event class, you can view event information that has been automatically filtered to show events specific to the current context.

Master event console

The master event console is Zenoss Core's central nervous system, enabling you to view and manage events. It displays the repository of all events that have been collected.

Figure 75: Event Console

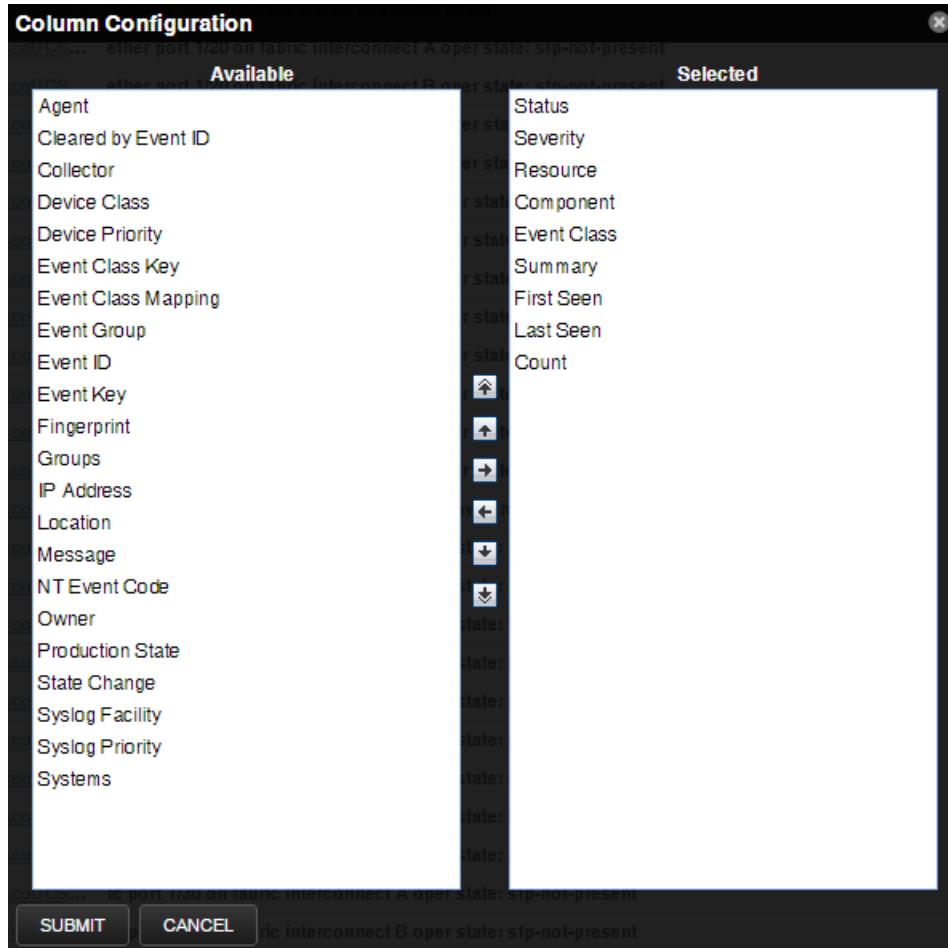
| Status | Severity | Resource | Component | Event Class | Summary | First Seen | Last Seen | Count |
|--------|----------|-------------------------------|-----------|-------------|--|------------------------|------------------------|-------|
| 0 | ucss1 | Fan 3/4/2 | | | Fan 2 in Fan Module 1-4 under chassis 3 speed: lower-non-recoverable | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 3 |
| 0 | ucss1 | extpol_regr_clients_client... | | | UCS Domain ucst-fabs is registered with UCS Central without a valid license. | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 4 |
| 0 | ucss1 | Fan Module 3/4 | | | Fan 2 in Fan Module 1-4 under chassis 3 speed: lower-non-recoverable | 2015-07-29 08:45:41 am | 2015-07-29 08:45:41 am | 1 |
| ! | ucss1 | fabric_san_A_phys-fcoes... | | | FCoE uplink port 1/8 is down | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 4 |
| ! | ucss1 | Fan 3/4/2 | | | Fan 2 in Fan Module 1-4 under chassis 3 operability: inoperable | 2015-07-29 08:44:33 am | 2015-07-29 10:29:40 am | 3 |

Customizing the event console

You can add or delete data columns to customize your event console view. The order of the selected column names determines the left-to-right display on the Event Console.

- 1 Navigate to **Events > Event Console**.
- 2 Click the **Configure** button and select **Adjust columns** from the drop-down list.
The Column Configuration window appears.

Figure 76: Event Console Column Configuration



- 3 To select a column, double-click the name in the **Available** list to move it to the **Selected** list.
- 4 In the **Selected** list, to re-order the columns, use the arrow keys.
- 5 Click **Submit**.

Selecting events

To select one or more events in the event console, you can:

- Click a row to select a single event
- Ctrl-click rows to select multiple events, or Shift-click to select a range of events

Sorting and filtering events

You can sort and filter events by any column that appears in the master event console.

To sort events, click a column header. Clicking the header toggles between ascending and descending sort order. Alternatively, hover over a column header to display its control, and then select Sort Ascending or Sort Descending.

Filter options appear below each column header.

Figure 77: Event Console filter options

The screenshot shows the Zenoss Event Console interface. At the top, there are tabs for Event Console, Event Archive, Event Classes, and Triggers. Below the tabs is a toolbar with various icons for filtering and configuration. The main area is a table displaying event logs. The columns are: Status, Severity, Resource, Component, Event Class, Summary, First Seen, Last Seen, and Count. There are dropdown menus for filtering these columns. A sidebar on the left shows filter criteria for Status (New, Acknowledged, Suppressed, Closed, Cleared, Aged), Resource (ucs1), Component (Fan 3/4/2, extpol_reg_clients_client..., Fan Module 3/4, fabric_san_A_phys-fcoes..., Fan 3/4/2, Fabric FC SAN Port Chan..., Fabric FC SAN Port Chan...), and Event Class (Summary). The table lists several events with their details and timestamps. The last event listed has a count of 4.

You can filter the events that appear in the list in several ways, depending on the field type. Date fields (such as First Seen and Last Seen) allow you to enter a value or use a date selection tool to limit the list. For other fields, such as Device, Component, and Event Class, enter a match value to limit the list.

The Count field allows you to filter the list when compared to a value. To search on count:

- N - Displays events with a count equal to N .
- $:N$ - displays events with a count less than or equal to N .
- $M:N$ - Displays events with a count between M and N (inclusive).
- $M:-$ - Displays events with a count greater than or equal to M .

To clear filters, select **Configure > Clear filters**.

Working with live search

By default, the system uses a "live search" feature to help you locate information. From the event console, you can search for information by:

- **Device (name) and Component** - Device name and Component searches:
 - Are case-insensitive.
 - Are tokenized on whitespace (meaning that any searches that span whitespace and do not start with a complete token will return no results).
 - If quoted, return only exact matches.
- **Summary** - Summary searches:
 - Are case-insensitive.
 - Are tokenized on whitespace (meaning that any searches that span whitespace and do not start with a complete token will return no results).
- **Event class** - Event class searches:
 - Are case-insensitive.
 - Are tokenized on / (slash). If the search begins with a slash, and ends with a slash or asterisk, then event classes are searched by using a "starts with" approach. If a search starts with a slash and ends with any other character, then event classes are searched by using an exact match for the event class. If a search does not begin with a slash, then event classes are searched by using a sub-string match on each event class.
- **IP Address** - IP address searches (for IPv4 and IPv6 values):

- Are tokenized by . (period) and : (colon). For example, the following searches would return a result of 129.168.1.100:
 - 168
 - 168.1
 - 129.16*
 - *29

■ Time fields

- **First Seen** - This is always the time of the first occurrence of the event and does not change.
- **Last Seen** - This is the most recent occurrence of the event, and is updated each time the event occurs.
- **State Change** - This is the time that the event state was modified, most commonly when the event is closed.

Entering a datetime in one of these filters formatted as YYYY-MM-DD HH:MM:SS displays events that have a timestamp that is equal to, or newer than the input datetime. Note that while the input field accepts a 24-hour format, the system displays it in 12-hour format by default (using am/pm).

Additionally you can also configure a time range to display events by using the following format 'start datetime TO end datetime': "YYYY-MM-DD HH:MM:SS TO YYYY-MM-DD HH:MM:SS". An example might look like: "2017-07-21 12:00:00 TO 2017-07-22 12:00:00". This would include all events that the timestamp occurred within a 24 hour period between 12:00:00 on July 21st through 12:00:00 on July 22nd.

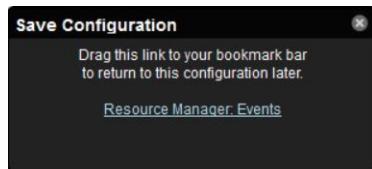
With live search enabled (the default behavior), the system filters available information immediately. It presents increasingly refined information with each character you type in the search window. When disabled, search responds only after you enter one or more characters and then press Enter.

Saving an event console view

Save a custom event console view by bookmarking it for quick access.

- 1 Select **Configure > Save this configuration**.
- 2 In the dialog box, select the link, and then drag it to the bookmarks area of the browser window.

Figure 78: Saving a custom view (bookmark)



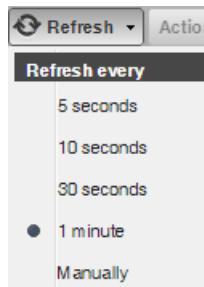
The browser adds a link to the bookmarks list.

- 3 Change the title of the bookmark to distinguish this event console view.

Refreshing the view

You can refresh the list of events manually or specify that they refresh automatically. To manually refresh the view, click **Refresh**. You can manually refresh at any time, even if you have an automatic refresh interval specified.

To set up automatic refresh, select one of the time increments from the Refresh list.

Figure 79: Automatic refresh selections

Viewing event details

You can view details for any event in the system.

- To view details, double-click an event row.

Figure 80: Event details

The screenshot shows an event details window for 'Server 4/6 (service profile:) has'. The window displays the following information:

- Resource:** [10.87.207.89](#)
- Component:** [chassis-4/blade-6](#)
- Event Class:** [/CiscoUCS/Fault](#)
- Status:** New
- Message:** Server 4/6 (service profile:) has ins number of DIMMs, CPUs and/or ad

Below the message, there are four links: [Event Management...](#), [Device State...](#), [Event Data...](#), and [Event Details...](#). A red box highlights the 'Open in new window' button, which is located next to the close icon (X) at the top right of the window.

- To display the event information in a new window, click the pop-out icon.
- To see more information about the event, click the link for **Event Management**, **Device State**, **Event Data**, or **Event Details**.
- In the log area, enter information about the event, and then click **Add**.

Acknowledging events

You may want to mark an event as "acknowledged" to indicate, for example, that you have taken action to remedy a problem. To mark events as acknowledged:

- Select one or more events in the event console view.
- Click the **Acknowledge Events** icon. A check mark appears for each acknowledged event.

Returning events to new status

Returning a previously acknowledged event to "new" status revokes its "acknowledged" status.

- Select one or more events in the event console view.

- 2 Click the **Unacknowledge Events** icon. A check mark no longer appears in the event row, and the event is returned to "new" status.

Classifying events

Classifying events lets you associate events shown as /Unknown with a specific event class. To classify an unknown event, an event class key must be specified for the event.

- 1 Select one or more /Unknown events in the event console view.
You can also classify events from the event archive.
- 2 Click the **Reclassify an Event** icon. The Classify Events dialog appears.
- 3 Select an event class from the list of options, and then click **Submit**.

Closing events

When you no longer want to actively monitor an event (after you acknowledge it, for example), you can specify to close the event and move it to the event archive according to a configured event archive interval. To do this:

- 1 Select one or more events in the event console view.
- 2 Click the **Close Events** icon.
The selected events are closed and moved to the archive at the specified interval.

To view events in the event archive, select **EVENTS > Event Archive**.

Note Users with no assigned role can view all events in the archive.

- 3 Click the **Refresh** icon to update the event list.
The closed events are removed from the display in the event console view.

Reopening events

You can reopen events in the active event console that are in the Closed, Cleared, or Aged state.

You cannot re-open a closed event if another active event with the same fingerprint exists. Before you can re-open the closed event, you must close the new event.

- 1 Select one or more Closed, Cleared, or Aged events.
- 2 Click the **Reopen Events** icon. The selected events are returned to active status.

Exporting event data

You can export data from the event console to a comma-separated value (.csv) or XML file. You can select individual events (to export only those events), or make no selections (to export all events that match the current filter criteria).

- 1 Select one or more events.
- 2 Select **Export > CSV** or **Export > XML**. By default, the exported file is named `events.Extension`.

Creating events

To create events from the event console, click the **Add an Event** icon.

For more information about manual event creation, see [Creating events manually](#) on page 123.

Event sources

Events enter the system as follows:

- *Generated events* are created as a result of active polling.
- *Captured events* are transmitted by external actions into the system.

Generated events

The following standard daemons are responsible for generating events. They automatically perform appropriate de-duplication and auto-clearing.

- **zenping** - Ping up/down events
- **zenstatus** - TCP port up/down events
- **zenperfsnmp** - SNMP agent up/down events, threshold events
- **zencommand** - Generic status events, threshold events
- **zenprocess** - Process up/down events, threshold events
- **zenwin** - Windows service up/down events

Captured events

Captured events are those events that the system does not specifically know will occur in advance. De-duplication is performed on these events, but might require tuning. By default, no auto-clearing is done on captured events. Event transforms must be used to create the auto-clear correlations.

The following standard daemons are responsible for collecting captured events:

- **zensyslog**- Events created from syslog messages.
- **zentrapp**- Events created from SNMP traps and informs.

ZenPacks that you install might include their own daemons.

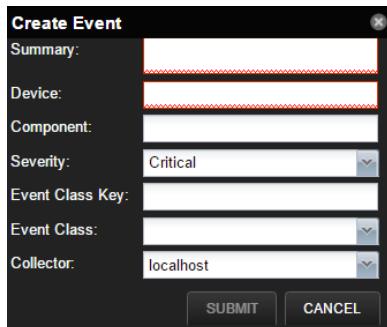
Creating events manually

You can manually create events. Though not required for normal system operation, creating events is useful when you have created mappings and transforms and need to test them.

Creating events in the browser interface

- 1 Navigate to **Events**, and then click the **Add an Event** icon.
- 2 In the **Create Event** dialog box, complete the event fields, and then click **SUBMIT**.

Figure 81: Create Event dialog box



Required fields: Summary, Device, Severity, and Collector.

Event class mappings are applied only for events that do not already have an event class.

Creating events from the command line interface

To send events from the command line, use the `zen sendevent` command.

Common options include:

- `-d DEVICE, --device=DEVICE`
- `-i IPADDRESS, --ipAddress=IPADDRESS`
- `-y EVENTKEY, --eventkey=EVENTKEY`
- `-p COMPONENT, --component=COMPONENT`
- `-k EVENTCLASSKEY, --eventclasskey=EVENTCLASSKEY`
- `-o OTHER, --other=OTHER`
- `-s SEVERITY, --severity=SEVERITY`
- `-c EVENTCLASS, --eventclass=EVENTCLASS`
- `-f INPUT_FILE, --file=INPUT_FILE`

- 1 Log in to the Control Center host as a user with `serviced` CLI privileges.
- 2 Attach to the `zenhub` service.

```
serviced service attach zenhub
```

- 3 Change to the `zenoss` user.

```
su - zenoss
```

- 4 Run the `zen sendevent` command.

```
zen sendevent Options "Event_Summary_Text"
```

Example: Simulate a ping down event

The following example shows how to use the `zen sendevent` script to simulate a ping down event:

```
zen sendevent -d router1.example.com -s Critical -c /Status/Ping "Router
down"
```

Event classes

Event classes are a simple organizational structure for the different types of events that the system generates and receives. This organization is useful for driving alerting and reporting. You can, for example, create an alerting rule that sends you an email or pages you when the availability of a Web site or page is affected by filtering on the `/Status/Web` event class.

Following is a subset of the default event classes. You can create additional event classes as needed.

- `/App` - Application-related events.
- `/Change` - Events created when the system finds changes in your environment.
- `/Perf` - Used for performance threshold events.
 - `/Perf/CPU` - CPU utilization events
 - `/Perf/Memory` - Memory utilization or paging events
 - `/Perf/Interface` - Network interface utilization events

- /Perf/Filesystem - File system usage events
- /Status - Used for events affecting availability.
- /Status/Ping - Ping up/down events
- /Status/Snmp - SNMP up/down events
- /Status/Web - Web site or page up/down events

Event class configuration properties

Just as device classes and devices have configuration properties, so do event classes and event class mappings. Configuration properties are applied hierarchically, with the most specific property being applied.

The following configuration properties are available on event classes and class mappings.

- **zEventAction**- How and where affected events are stored when they occur.
 - **status**- Active events table with original event state
 - **history**- Active events table with closed event state
 - **drop**- Events are not stored
- **zEventClearClasses**- Optional list of event class names whose active events will be cleared by clear events occurring in this class.
- **zEventMaxTransformFails** - Bad transforms are disabled from executing after the specified number of failures. The default value is 10.
- **zEventSeverity**- The severity of affected events is changed to this value unless the default value (-1) is used.
- **zFlappingIntervalSeconds**- Defines the time interval to check for event flapping (changing severity level repeatedly). Default value is 3600 seconds.
- **zFlappingSeverity**- Define the severity to check for event flapping. If the severity level on an event changes from this value a certain number of times (zFlappingThreshold) within a certain time range (zFlappingIntervalSeconds) then an event flapping event is generated. Possible values include: 5-Critical, 4-Error, 3-Warning, 2-Info, 1-Debug, and 0-Clear.
- **zFlappingThreshold**- Number of times an event severity must flap within an interval. One of the parameters to define in order to generate event flapping events.

A good example of how the system uses the event class configuration properties is found in the /Status event class. Within the /Status event class configuration properties, zEventAction is set to "history" and zEventSeverity is set to "Default". This means that events sent with this event class are sent into the active events table with an initial state of closed, and the event severity unchanged.

For more information about event manipulation techniques, see [Mapping and transformation](#) on page 125.

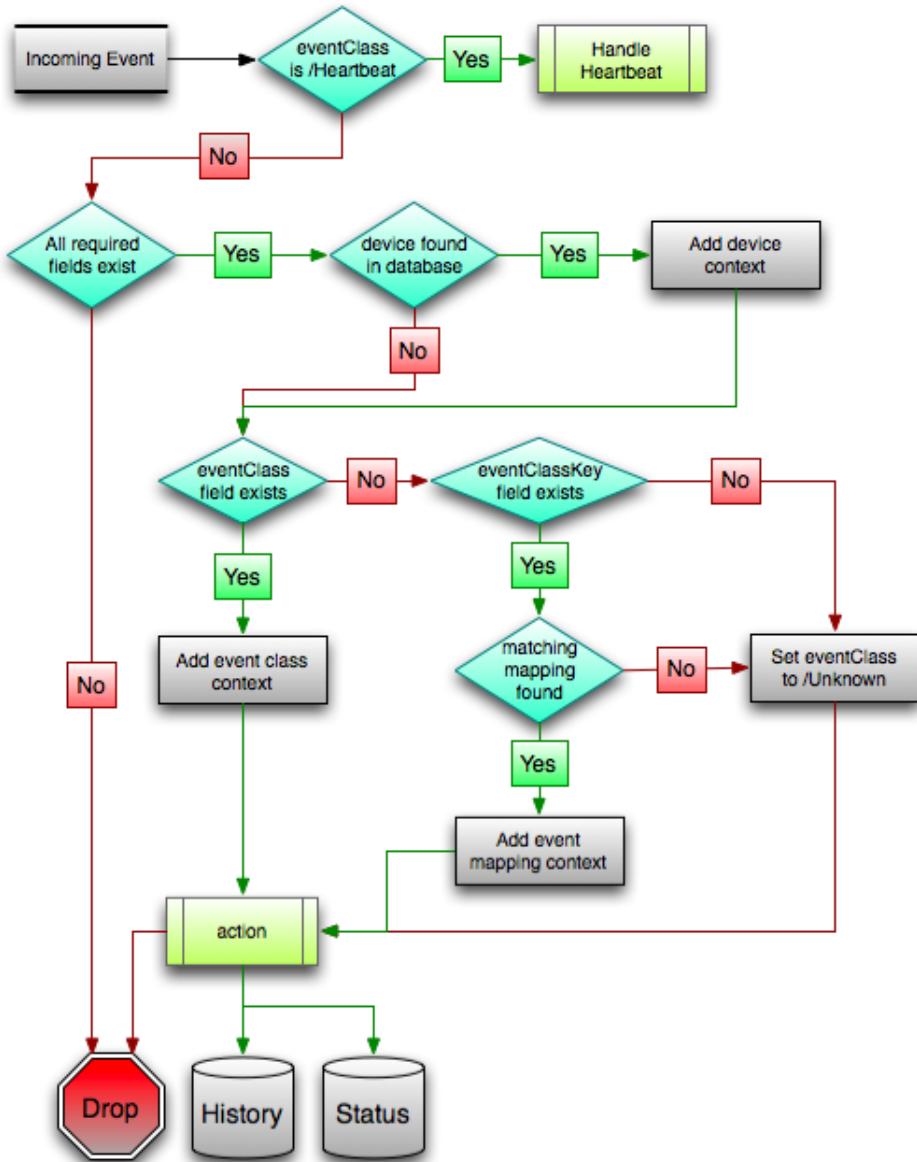
Mapping and transformation

The event mapping and transformation system allows you to perform a wide range of operations, from altering the severity of certain events to altering nearly every field on an event, based on complex rules.

You cannot alter the following fields through event transformation. (This is because they are set after transformation has been performed.)

- evid
- firstTime
- lastTime
- count

The following illustration shows the path followed by an incoming event in the event mapping system.

Figure 82: Event processing

The mapping and transformation process begins with the "eventClass field exists" decision. This also is one of the more important differentiators in how you must handle a particular type of event.

Event class mappings

To view event class mappings, select **EVENTS > Event Classes**, and then select **Mapping Instances** in the drop-down list. This allows you to see all event class mappings in a single location. The ID column shows the mapping's event class.

You can create event class mappings directly from the event classes, but this requires that you know the event class key. A simpler way to create event class mappings is through the event console:

- 1 Select an event that you want to match in the event console.
- 2 Click the **Reclassify an Event** icon. The Classify Events dialog appears.

- 3 Select the event class to which you want to map the event, and then click **Submit**. This creates the event class mapping with the correct event class key, and example text against which you can develop your regular expression.

When editing an event class mapping, you can control which events it will match, as well as other properties:

■ Matching tab

- **Event Class Key**- Must match the incoming event's Event Class Key field for this mapping to be considered as a match for events.
- **Rule**- Provides a programmatic secondary match requirement. It takes a Python expression. If the expression evaluates to True for an event, this mapping is applied.
- **Regex**- The regular expression match is used only in cases where the rule property is blank. It takes a Perl Compatible Regular Expression (PCRE). If the regex matches an event's message field, then this mapping is applied.
- **Explanation**- Free-form text field that can be used to add an explanation field to any event that matches this mapping.
- **Resolution**- Free-form text field that can be used to add a resolution field to any event that matches this mapping.
- **Transforms tab**- Takes Python code that will be executed on the event only if it matches this mapping. For more details on transforms, see the section titled "Event Class Transform."
- **Configuration Properties tab**- Listing of Configuration Properties defined for this event class.
- **Sequence tab**- Sequence number of this mapping. This number determines the order in which mappings with the same event class key are evaluated.

Mappings have the same configuration properties as event classes. Any configuration property set locally on a mapping will override the same property set on the event class. This works in the same hierarchical, most specific match, concept that device class and device configuration properties work.

When a captured event occurs, it will not have a pre-defined event class. For this type of event, you must create an event class mapping if you want to affect the event. If a captured event occurs and none of the event class mappings in the system match it, its event class will be set to /Unknown, and it will retain all of the default properties with which it began.

The next step of evaluation for events without an event class is to check the Event Class Key field. This controls which event class mapping the event will match. If the event has a blank event class key, or its event class key does not match any event class mappings in the system, the special "defaultmapping" event class key is searched for instead. This provides for a way to map events even if they have a blank or unpredictable event class key.

Event class mapping sequence

The sequence area of an event class mapping (select Sequence in the left panel) allows you to provide more than one mapping for the same event class key. In this case, the sequence is evaluated in ascending order until a full (rule or regex) match is found.

For example, suppose a router is sending in unclassified events that need to be mapped to two event classes:

- /Events/Router/fanDown
- /Events/Router/fanUnknown

The event class key for both has been sent to "router", but one has a message of "Fan Down" and the other has no message at all. The mapping on /Events/Router/fanDown has an event class key of "router" and a regex of "Fan Down." The mapping on /Events/Router/fanUnknown has only an event class key of "router" and (in this example) no regex. Because the fanUnknown mapping matches the fanDown events, the evaluation of fanDown needs to occur first.

You can modify the evaluation of mappings with the same event class key in the Sequence area of any of those event class mappings. In the previous example, you could go to either mapping, select Sequence, and both mappings would be displayed. You can set one to 0, and the other to 1. (You can enter other values, but they will be changed to the shortest list of integers, starting with 0.) Setting fanDown to 0 and fanUnknown to 1 will ensure that the events will be mapped properly.

Event class transform

When a generated event occurs, it has an event class assigned to it. This causes the event class mapping step to be skipped. The only way to affect the fields of one of these events is through the event class' configuration properties and transform.

To access the transform for an event class:

- 1 Navigate to the event class from **Events > Event Classes**.
- 2 From the drop-down list, select **Transforms**.
- 3 Enter information into the dialog box (as Python code), and then click the **Save** button in the upper-right corner. As you develop your transform, you can revert back to the last saved state by clicking the **Revert this Transform** button.

The objects available in this Python context are `evt` (the event); and, if the event matches a device that exists in the system database, a `device` object.

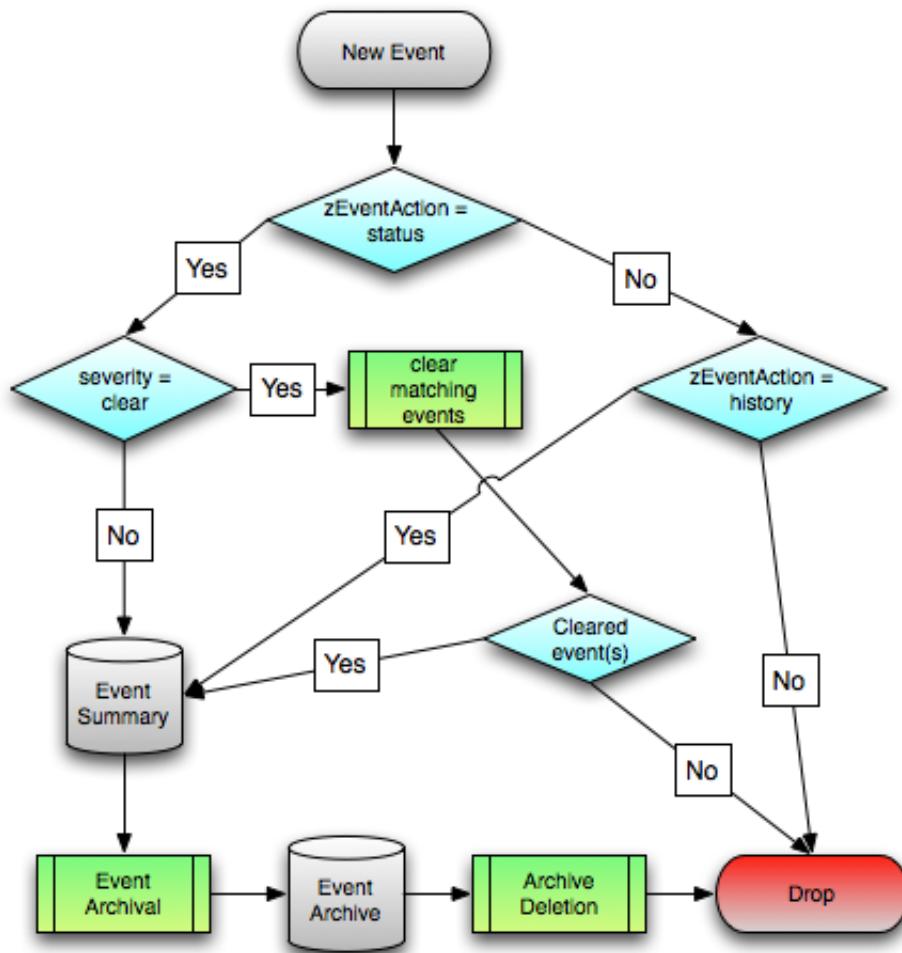
The following example shows how you can validate that a device object exists before using it to drop events from a particular location.

```
if device and "Hawaii" in device.getLocationName(): evt._action = "drop"
```

Event life cycle

In addition to manual methods for getting events into the status table or event archive, there are automated processes that move events from status into the archive. The *event life cycle* is defined as all of the ways that events can be added to, moved in, and deleted from the database.

The following illustration depicts the event life cycle.

Figure 83: Event life cycle

Automatic event aging

From the Event Configuration page (**ADVANCED > Settings > Events**), you can set up automatic aging of events. Aging of events will automatically update active events that match the severity and aging threshold to a status of Aged. After the configured event archive interval, all Closed, Aged, and Cleared events are moved to the event archive.

Properties that control this behavior are:

- **Don't Age This Severity and Above** - Options are Age All Events, Critical, Error, Warning, Info, Debug, and Clear. By default, this value is set to Error, meaning that all events with a status of Error or Critical are not aged.
- **Event Aging Threshold (minutes)** - Set the time value, in minutes, that an event must reach before it is aged. By default, this is 240 minutes.
- **Event Aging Interval (milliseconds)** - The interval when events are scanned to perform autoaging. By default, this is 60000 milliseconds (60 sec).
- **Event Aging Limit** - The maximum number of events to age in each interval. The limit should be kept relatively low to prevent large database transactions. By default, this is 1000 events.
- **Event Archive Threshold (minutes)** - Specify the number of minutes since a closed event was last seen before it is moved to the event archive. The minimum value is 1; the maximum value is 43200.
- **Event Archive Interval (milliseconds)** - The interval when events are scanned for moving to the archive. By default, this is 60000 milliseconds (60 sec).

- **Event Archive Limit** - The maximum number of events to archive in each interval. The limit should be kept relatively low to prevent large database transactions. By default, this is 1000 events.
- **Delete Archived Events Older Than (days)** - The number of days that events in the event archive are saved. By default, they are kept in the archive for 90 days. The minimum value is 1 and the maximum value is determined by the range of event archive partitions. With the default configuration, the maximum value is 1000 days.
- **Default Syslog Priority** - Specify the default severity level assigned to an event coming from zensyslog if no priority can be determined from the event.
- **Default Availability Report (days)** - Enter the number of days to include in the automatically generated Availability Report. This report shows a graphical summary of availability and status.
- **Max Event Size in Bytes** - The maximum size of an event that will be processed in bytes. Events that are too large will be logged and dropped. Events that will become too big will have their details overwritten with new details. By default, this is 32768 bytes.
- **Summary Index Interval (milliseconds)** - The default indexing interval of the event summary in milliseconds. By default, this is 1000 milliseconds (1 sec).
- **Archive Index Interval (milliseconds)** - The default indexing interval of the event archive in milliseconds. By default, this is 30000 milliseconds (30 sec).
- **Index Limit** - The number of events to index in each index interval. By default, this is 1000 events.
- **Event Time Purge Interval (days)** - The number of days that event occurrence time are kept. By default, they are kept for 7 days. The minimum value is 1 and the maximum value is determined by the range of event time partitions. With the default configuration, the maximum value is 7 days.
- **Enable Event Flapping Detection** - Select this check box if you wish to enable event flapping detection. If an event is created and then cleared *flapping_threshold* times in *event_flapping_interval* time then an event of event flapping event class is created.
- **Event Flapping Event Class** - The event class under which generated flapping events belong.
- **Clear Event Heartbeats** - Click **Clear** to clear the event heartbeats.

Automatic archived event cleanup

You can set up automatic purging of events from the event archive from the Event Configuration page (**ADVANCED > Settings > Events**). When events are purged, they can be recovered only from backups.

The property that controls this behavior is Delete Archived Events Older Than (days). Acceptable values are between 1 and 1000 (days).

Capturing email messages as events

ZenMail and ZenPop allow you to capture email messages as events. This capability can be useful for situations in which embedded systems (such as WAPs, NAS devices, or RAID controllers) rely on email notification for events.

ZenMail

ZenMail serves as an SMTP server that you can bind to a specific TCP port. You can then configure your embedded system to send mail to the Zenoss Core server explicitly by using the server's IP address as the relay.

ZenMail supports these configuration directives:

- `${ZENHOME}/bin/zenmail` (no arguments) - Default operation. Binds to port 25 on all ports and listens for email messages to arrive. Ignores the TO field in the email and uses the FROM address as the device IP address.
- `${ZENHOME}/bin/zenmail --listenPort` - Bind to the port provided. Useful in situations in which an SMTP server is already running on the Zenoss Core server and you do not want to interfere with the

existing mail delivery system. Semantics are the same as the no argument version (FROM address is used as the device IP).

Note To execute a command using \$ZENHOME (/opt/zenoss for the zenoss user), you must be attached to the container holding the Zenoss Core application. See the Control Center documentation for serviced commands.

ZenPop3

ZenPop3 allows you to retrieve event email from a POP server. ZenPop3 supports these configuration directives:

- --usessl- Issue the STARTTLS command to the POP server and attempt to transfer email messages using SSL encryption. This is required if retrieving mail from Google.
- --nodelete- Do not issue the DELE command after retrieving all messages. Typically this is used during initial testing so that you do not have to resend test messages to the POP account. Some email systems (such as Google) do not actually delete messages when the DELE command is issued.
- --pophost- The hostname or IP address of the POP server from which to retrieve messages.
- --popport- The TCP port the POP server listens on. Defaults to 110. Used in situations where the POP provider listens on another port (for example, Google on port 995).
- --popuser- The user name that contains email messages to retrieve.
- --poppass- The password to use for the user name provided.
- --cycletime- The time to sleep between polls. After all email is retrieved, ZenPop3 sleeps for this amount of time before waking up and attempting to pull new email.

Translating message elements to the event

Zenoss Core translates various message elements to the event, as follows:

- **FROM Field-** If the FROM field is an IP address, then the system associates the event with the device with the same IP address. If the FROM field is a fully qualified domain name, then the system resolves it to an IP address, and then performs the device association using the resolved IP address. The resolution of hostname uses "A" records rather than "MX" records.
- **TO Field-** The system ignores the TO field in the email message. ZenMail accepts email to any user and domain name combination. ZenPop also drops the TO field, and uses only the FROM field.
- **SUBJECT Field-** ZenMail and ZenPop use the SUBJECT as the event summary.
- **Message Body-** ZenMail and ZenPop use the first mime attachment as the event details. The system ignores secondary message bodies (typically HTML-encoded versions of the message). It also ignores attachments (such as files).

SNMP traps and event transforms

An SNMP trap is a message that is initiated by a network element and sent to the network management system. Often, traps indicate a failure of some sort, such as a router message indicating a power supply failure, or a printer message indicating an "out-of-ink" condition.

If an SNMP trap enters the system, and Zenoss Core cannot identify the event (the event is classified as "/Unknown"), then you can classify the event so that the system handles it consistently.

Classifying SNMP traps

To classify an SNMP trap event:

- 1 From the Event Console, select the unknown event or events.
- 2 Click the **Reclassify an event** icon. The Classify Events dialog appears.
- 3 Select /App, and then click **Submit**.

To edit this classification:

- a From the Navigation area, select **Events > Event Classes**.
- b Ensure Mapping Instances appears.
- c Select the event map you created.
- d In the left panel, select **Edit** from the **Action** icon.

The edit page appears. This page contains rules used to map the event to the /App category. This rule, since it matches the trap by a specific OID, is all that is needed.

In the Transform area, you can enter code to modify the summary. For example, if you want to set the summary string to "Spam Filter Detects Virus," then you can enter:

```
evt.summary = "Spam Filter Detects Virus"
```

A trap has a header with some standard information, followed by a sequence of attribute/values.

You have indicated you want the value for the OID ".1.3.6.1.4.1.9789.1500.2.5" as the summary. If you had the MIB loaded, you could do this:

```
evt.summary = evt.spamFilterDetectsVirus
```

However, the OID and the data is still in there. Instead, use the slightly more cryptic:

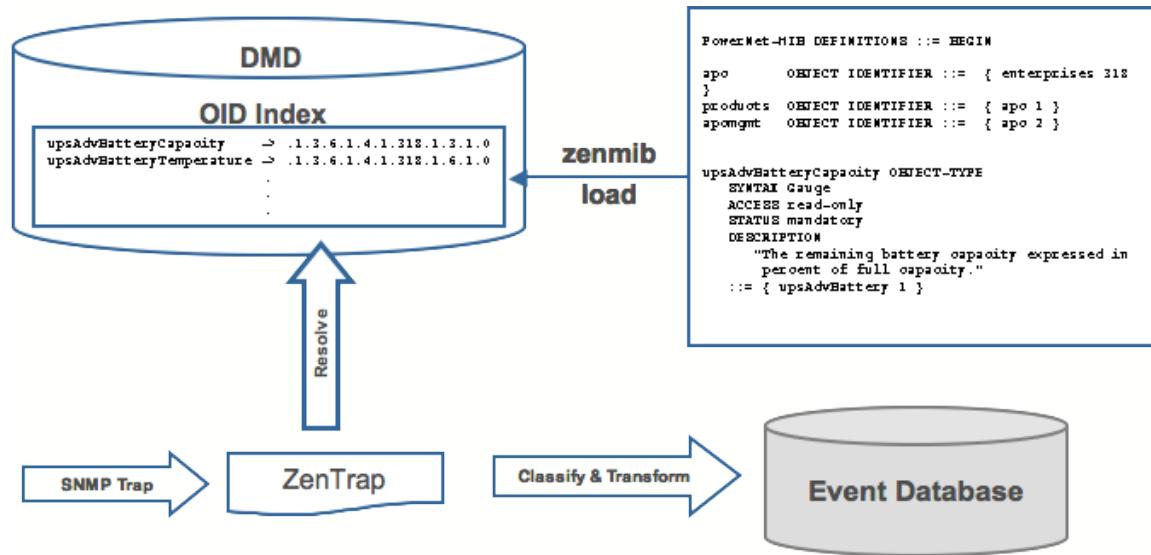
```
evt.summary = getattr(evt, ".1.3.6.1.4.9789.1500.2.5", "Unexpected missing OID")
```

The "device" object for the event has been made available, as well:

```
evt.summary = getattr(evt, ".1.3.6.1.4.9789.1500.2.5", "Unexpected missing OID") \ + " from device " + device.getId()
```

Zenoss Core uses MIBs to translate SNMP traps that contain raw OID values. Loading a MIB into the system allows it to translate numeric OIDs such as .1.3.6.1.2.1.1.6 into descriptive phrases like "sysLocation". It also makes it easier to manipulate the events in an event mapping.

Note After loading MIBs into the system, be sure to restart the zentrap service so that it can retrieve the new MIB information.

Figure 84: SNMP TRAP transform

Following is a small demonstration MIB.

```

NOTIFICATION-TEST-MIB DEFINITIONS ::= BEGIN
IMPORTS
  ucdavis FROM UCD-SNMP-MIB
  NOTIFICATION-TYPE FROM SNMPv2-SMI
;
demonotifs OBJECT IDENTIFIER
  ::= { ucdavis 991 }
demo-notif NOTIFICATION-TYPE
OBJECTS { sysLocation }
STATUS current
DESCRIPTION "Just a test notification"
  ::= { demonotifs 17 }
END
  
```

Example: Sending test traps

To send an SNMP trap:

- From the command line, enter the following command:

```
$ snmptrap -v 2c -c public localhost ''
1.3.6.1.4.1.2021.991.1.3.6.1.2.1.1.6 s \ "Device in Austin"
```

- Save this demonstration MIB into a file.
- Send the trap.
- Open the Event Console and find the trap you sent.
- Send this event to the event archive.
- Load some MIBs into the system so that this OID is translated into a better format:
 - Copy the demonstration MIB into \$ZENHOME/share/mibs/site.

Note To execute a command using \$ZENHOME (/opt/zenoss for the zenoss user), you must be attached to the container holding the Zenoss Core application. See the Control Center documentation for serviced commands.

- b** Run `zenmib` to load it:

```
$ zenmib run -v 10 DEBUG:zen.zenmib:TRAP-TEST-MIB.mib
INFO:zen.zenmib:Unable to find a file \ providing the MIB UCD-SNMP-
MIB ...
```

- c** The MIB loaded, but is missing some other definitions. Copy them:

```
$ cp /usr/share/snmp/mibs/SNMPv2-MIB.txt $ZENHOME/share/mibs/site \
$ cp /usr/share/snmp/mibs/UCD-SNMP-MIB.txt $ZENHOME/share/mibs/site
```

- d** Run `zenmib` again and load the definitions into the system:

```
$ zenmib run -v 10
```

- e** Restart the `zentrap` daemon to retrieve the new MIB information:

```
$ zentrap restart
```

- f** Send the trap a second time:

```
$ snmptrap -v 2c -c public localhost ''
1.3.6.1.4.1.2021.13.991 .1.3.6.1.2.1.1.6 s \ "Device in Austin"
```

- g** Check the event. Make sure the count is 1. If the count is 2, send the event to the event archive and send the trap again. Look at the Details tab. Now you should see something like this:

```
sysLocation Device in Austin
```

You should also see that the event summary changes from:

```
snmp trap 1.3.6.1.4.1.2021.13.991 from localhost
```

to:

```
snmp trap ucdExperimental from localhost
```

Transforming events with event mappings

To modify events as they arrive, create an event map through the user interface:

- 1 Create an event class.
- 2 Go to the event console and create an event mapping in this class from the existing event.
- 3 Edit the map.
- 4 In the Transform area, update the event with detail data. The entry field allows you to insert Python scripts. The event is provided as "evt" and the device as "dev." In this case, extract the sysLocation event detail and make it the summary with:

```
evt.summary = evt.sysLocation
```

- 5 Save the event mapping.

If you move the event to the event archive and resend the trap, the summary for the trap should now read the device name in the location you assigned.

If you encounter problems with the transform, check the `zentrap.log` file for errors that occurred.

Event transforms based on event class

When an event arrives in the system, you can change values (such as severity). For example, you can make the summary more informative, or change severity according to text within the summary.

Each event class allows for a short Python script to be executed when an event arrives.

Example

A user may want full file system threshold events on `/data` to be critical. Add the following Python script in the Threshold Transform of `/Events/Perf/Filesystem`:

```
if evt.component == '/data' and evt.severity != 0: evt.severity = 5
```

Like event mappings for event class keys, "evt" and "dev" objects are available in the script of the transform.

Production states and maintenance windows

Production state determines the level of monitoring and alerting applied to an individual device. Typically, alerting rules specify that the system will monitor and create events for devices that are in the "Production" production state.

Maintenance windows are planned time periods used to temporarily modify alerting rules so that event-generated alerts are temporarily halted during the window.

Production states

Production state determines whether a device is monitored, and can be used to control several elements of the event system, such as whether an event will produce a remote alert (email or page).

Choose a production state for a device based on whether you want:

- The device to be monitored
- The device to appear on the dashboard
- Alerting to occur

The following table lists production states and their characteristics.

| Production state | Devices monitored? | Appear on dashboard? |
|------------------|--------------------|----------------------|
| Production | yes | yes |
| Pre-Production | yes | no |
| Test | yes | no |
| Maintenance | yes | may appear |
| Decommissioned | no | no |

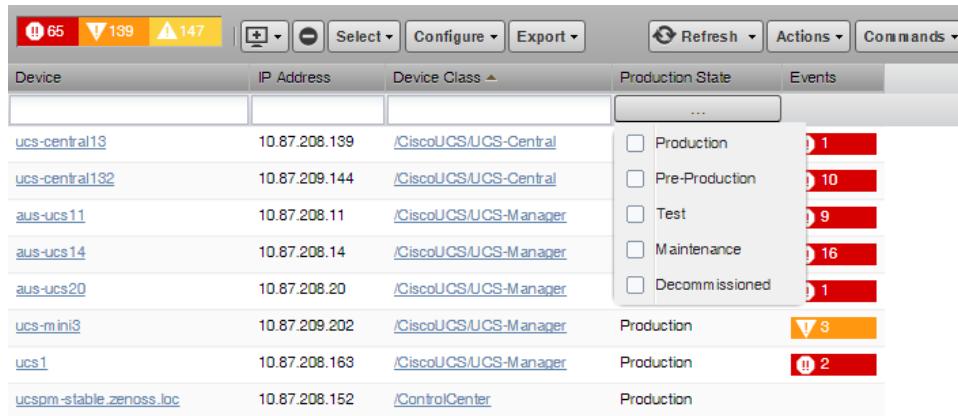
When you add a device to the system, its default state is Production. You may want to add triggers and notifications to alert you to various conditions that occur in the system, such as production state changes or a severity level being reached. For example, you can set up a trigger when a device is in either a production or a maintenance state and has a severity of Error or higher. You can then notify users when this trigger condition is met. For more information, see [Working with triggers](#) on page 22.

Setting the production state for devices

To set the production state for a device:

- 1 Click a device name in the list of devices. The Device Overview page appears.
- 2 Select a production state from the list of options, and then click **Save**.
- 3 Optional: To set the production state for a group of devices, perform the following:
 - a Select a category of devices (by class, group, system, or location) from the hierarchy.
 - b Click the **Actions** button and select **Set Production State** from the drop-down menu.
 - c Select a production state from the drop-down list and click **OK**.
 - d To filter the display of devices with a certain production state, click the button underneath the Production State column header and check the production states you want to see in the display.

Figure 85: Select production state (multiple devices)



The screenshot shows a table with columns: Device, IP Address, Device Class, Production State, and Events. The Production State column contains checkboxes for Production, Pre-Production, Test, Maintenance, and Decommissioned. The Events column shows counts for each state: Production (1), Pre-Production (10), Test (9), Maintenance (16), and Decommissioned (1). The Events column also includes a 'Production' button and a 'Decommissioned' button.

| Device | IP Address | Device Class | Production State | Events |
|-------------------------|---------------|-----------------------|--|--------|
| ucs-central13 | 10.87.208.139 | /CiscoUCS/UCS-Central | <input type="checkbox"/> Production <input type="checkbox"/> Pre-Production <input type="checkbox"/> Test <input type="checkbox"/> Maintenance <input type="checkbox"/> Decommissioned | 1 |
| ucs-central132 | 10.87.209.144 | /CiscoUCS/UCS-Central | <input type="checkbox"/> Production <input type="checkbox"/> Pre-Production <input type="checkbox"/> Test <input type="checkbox"/> Maintenance <input type="checkbox"/> Decommissioned | 10 |
| aus-ucs11 | 10.87.208.11 | /CiscoUCS/UCS-Manager | <input type="checkbox"/> Production <input type="checkbox"/> Pre-Production <input type="checkbox"/> Test <input type="checkbox"/> Maintenance <input type="checkbox"/> Decommissioned | 9 |
| aus-ucs14 | 10.87.208.14 | /CiscoUCS/UCS-Manager | <input type="checkbox"/> Production <input type="checkbox"/> Pre-Production <input type="checkbox"/> Test <input type="checkbox"/> Maintenance <input type="checkbox"/> Decommissioned | 16 |
| aus-ucs20 | 10.87.208.20 | /CiscoUCS/UCS-Manager | <input type="checkbox"/> Production <input type="checkbox"/> Pre-Production <input type="checkbox"/> Test <input type="checkbox"/> Maintenance <input type="checkbox"/> Decommissioned | 1 |
| ucs-mini3 | 10.87.209.202 | /CiscoUCS/UCS-Manager | Production | V 3 |
| ucs1 | 10.87.208.163 | /CiscoUCS/UCS-Manager | Production | !! 2 |
| ucspm-stable.zenoss.loc | 10.87.208.152 | /ControlCenter | Production | |

Maintenance windows

Maintenance windows allow scheduled production state changes of a device or all devices in a system, group, or location. You might want to set up a maintenance window, for example, to change a device's production state while you perform configuration changes or reboot a device.

Note In lieu of setting up a maintenance window, you can change the production state for a device manually at the time you want to make changes.

When the maintenance window starts, the production state of the device is set to the value of Start Production State (for example, Maintenance). When the maintenance window closes, the production state of the device reverts to the value of Stop Production State (the state the device was in prior to Maintenance).

Maintenance windows do not prevent notifications from being triggered on the device. If you want to define the notifications you receive during the maintenance window, you will need to set up an appropriate trigger for the device production state that you set during your maintenance window. For more information, see [Working with triggers](#) on page 22.

Maintenance window events

When a maintenance window starts, an event is created with the following information:

- depuid - zenactions | Resource | MaintenanceWindowName | TargetOrganizerOrDevice
- prodState - StartProductionState
- severity - Info
- summary/message - Maintenance window starting MaintenanceWindowName for TargetOrganizerOrDevice
- eventClass - /Status/Update
- eventClassKey - mw_change

- maintenance_devices - *TargetOrganizerOrDevice*
- maintenance_window - *MaintenanceWindowName*

When a maintenance window stops, an event is created with the following information:

- severity - Clear
- summary/message - Maintenance window stopping *MaintenanceWindowName* for *TargetOrganizerOrDevice*
- prodState - -99 (meaning "unknown.")

Maintenance window events auto-clear, meaning that stop events clear start events.

Creating and using maintenance windows

You can create a maintenance window for an individual device or group of devices (all devices, a device class, group, system, or location) in the devices hierarchy.

Create a maintenance window for a single device

Use this procedure to create a maintenance window for a device.

- 1 Log in to the Zenoss Core browser interface, and then navigate to **INFRASTRUCTURE > Devices**.
- 2 In the content area, click the name of the device.
- 3 In the sidebar, click **Device Administration**.
- 4 In the Maintenance Window toolbar, click **Add**.
- 5 In the **Add New Maintenance Window** dialog box, specify the attributes of the new maintenance window. The attributes allow you to specify whether or not to enable the window, the time of day and date when the maintenance window starts (in local time zone format), the duration of the window, and whether and how the window repeats. The **Window Production State** field allows you to categorize the production state of the device during the maintenance window. When the window ends, the device is returned to the production state it was in when it entered the maintenance window.
- 6 At the bottom of the **Add New Maintenance Window** dialog box, click **SUBMIT**.

Create a maintenance window for a group of devices

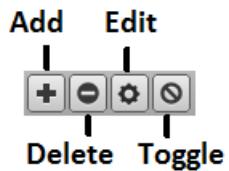
Use this procedure to create a maintenance window for a group of devices.

- 1 Log in to the Zenoss Core browser interface, and then navigate to **INFRASTRUCTURE > Devices**.
- 2 In the content area, select a group of devices, and then click **DETAILS**, located at the top of the sidebar.
- 3 In the sidebar, click **Device Administration**.
- 4 In the Maintenance Window toolbar, click **Add**.
- 5 In the **Add New Maintenance Window** dialog box, specify the attributes of the new maintenance window. The attributes allow you to specify whether or not to enable the window, the time of day and date when the maintenance window starts (in local time zone format), the duration of the window, and whether and how the window repeats. The **Window Production State** field allows you to categorize the production state of the device during the maintenance window. When the window ends, the device is returned to the production state it was in when it entered the maintenance window.
- 6 At the bottom of the **Add New Maintenance Window** dialog box, click **SUBMIT**.

Managing maintenance windows

Once you have created maintenance windows for your devices or groups of devices, you can quickly manage these instances on the Maintenance Windows screen.

- 1 Navigate to the Maintenance Window screen. This is the same place where you initially created the maintenance window (Device Administration link on Device Overview page). On this screen you can perform any of the following by clicking the appropriate icon:



- Add a new maintenance window
 - Delete the selected maintenance window
 - Edit the selected maintenance window (can also double-click a maintenance window row)
 - Toggle the selected maintenance window from enabled to disabled and vice-versa. The Enabled column will switch values.
- 2 Ensure that your changes are reflected in the Maintenance Window screen.

10

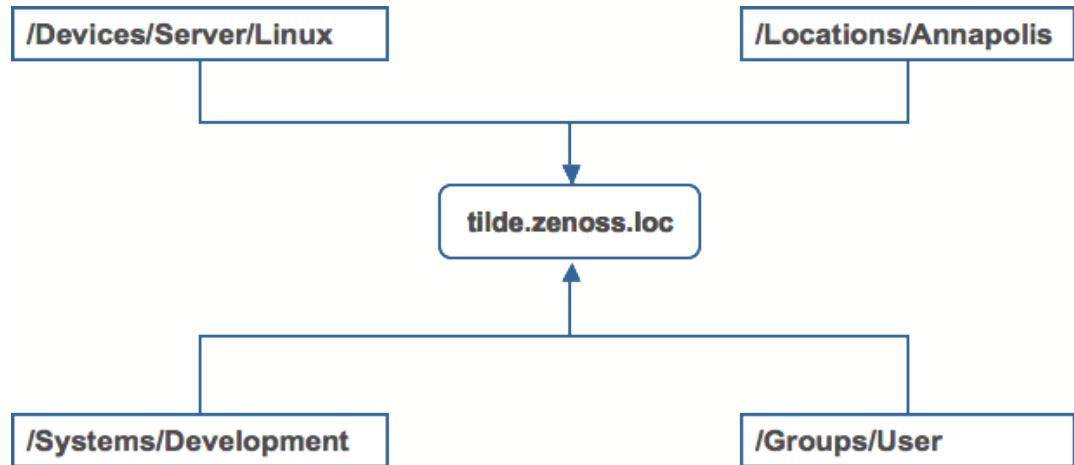
Organizers and path navigation

You can group system objects, including devices, sub-systems, components, configuration properties, and templates. Types of classes are as follows:

- Device classes
- Groups
- Systems
- Locations
- Component groups

In the following figure, the device `tilde.zenoss.loc` belongs to five different classifications. Configuration properties and monitoring settings for each of these groups are applied to this device.

Figure 86: Device groupings



Classes

The most important organizers are *classes*, as follows:

- Device classes
- Event classes
- Service classes
- Product classes

Templates and configuration properties can be inherited based on class. These attributes can be overwritten further down the class hierarchy, all the way down to the individual component level. The class hierarchy includes all defined and standard classes and sub-classes.

The following procedures are illustrated using device classes and sub-classes, but the same concepts apply to event classes, service classes, and product classes. When you add a device to the system, after providing the network name or IP address, specify its device class. Templates and configuration properties can be set at any level in the device class hierarchy.

Viewing device classes

To view device classes and the devices they contain, from the navigation menu, select **INFRASTRUCTURE**.

At the top level of the device hierarchy are device classes, such as CiscoUCS. To view devices in a class or to expand the organizer to show subclasses, click a name in the tree. Severity indicators show the most severe type of event that is associated with any device in that class.

Adding a class

After you add a device class, you can move devices into the class.

- 1 From the navigation menu, choose **INFRASTRUCTURE**.
- 2 In the left pane device class hierarchy, choose the parent class in which to add a child class; for example, CiscoUCS.
- 3 At the bottom of the left pane, click the **Add** icon. The Add Device Class dialog box appears.
- 4 Specify a name and description for the new device class, and then click **Submit**.
The new device class appears in the hierarchy under the parent device class.
- 5 To move devices to the new class, choose the devices in the device list, and then drag them to the new class.

Moving a class

You can move a class from one location in the hierarchy to another.

- 1 In the hierarchy, choose the class.
- 2 Drag the class to its new location.
- 3 In the **Move Organizer** confirmation dialog box, click **OK**.

Setting configuration properties at the class level

Definitions are applied to all devices currently in the class and those added to the class (unless overridden at a lower level in the hierarchy).

- 1 In the left pane device class hierarchy, choose **Devices**.
- 2 Click **DETAILS > Configuration Properties**. The **Configuration Properties** page for the selected device class appears.

Figure 87: Device class configuration properties

| Device class configuration properties | | | | | | | | | |
|---------------------------------------|--|----------|--|----------|------------------|-------------------------------|-------------------|-------------------------------------|--------|
| Devices | | SEE ALL | | | | | | | |
| | | Is Local | | Category | Name | Value | Label | Description | Path |
| Devices | | | | Yes | Misc | zAggregatorCollectionInterval | 300 | zAggregatorCollectionInterval | / |
| Events | | | | Yes | Cisco | zCiscoACEUseSSL | true | Use SSL for ACE XML API | En... |
| Modeler Plugins | | | | Yes | Cisco | zCiscoNXAPIInterval | 300 | NX-API Collection Interval | Ti... |
| Configuration Properties | | | | Yes | Cisco | zCiscoNXAPIPort | 443 | NX-API Port | Po... |
| Custom Properties | | | | Yes | Cisco | zCiscoNXAPIUseSSL | true | Use SSL for NX-API | En... |
| Device Administration | | | | Yes | Cisco | zCiscoRemodelEventClassKeys | | Remodeling Event Class Keys | Ev... |
| Overridden Objects | | | | Yes | Cisco UCS | zCiscoUCSCIMCEventsInterval | 60 | CIMC Event Collection Interval | Se... |
| Monitoring Templates | | | | Yes | Cisco UCS | zCiscoUCSCIMCPerfInterval | 300 | CIMC Stats Collection Interval | Se... |
| Device (/) | | | | Yes | Cisco UCS | zCiscoUCSCIMCSSLProtocol | SSLv23 | CIMC SSL/TLS Protocol | Th... |
| | | | | Yes | Cisco UCS | zCiscoUCSManagerPassword | ***** | Manager Password | Pa... |
| | | | | Yes | Cisco UCS | zCiscoUCSManagerPerfInterval | 300 | Manager Stats Collection Interval | Th... |
| | | | | Yes | Cisco UCS | zCiscoUCSManagerPort | 443 | Manager Port | Po... |
| | | | | Yes | Cisco UCS | zCiscoUCSManagerUseSSL | true | Manager SSL | Wh... |
| | | | | Yes | Cisco UCS | zCiscoUCSManagerUser | admin | Manager User | Us... |
| | | | | Yes | Modeler Controls | zCollectorClientTimeout | 180 | Collector Client Timeout (seconds) | All... |
| | | | | Yes | Modeler Controls | zCollectorDecoding | utf-8 | Collector Decoding | Co... |
| | | | | Yes | Misc | zCollectorLogChanges | false | Log Collector Changes? | Ind... |
| | | | | Yes | zencommand | zCommandCommandTimeout | 15 | Timeout for Commands (seconds) | Sp... |
| | | | | Yes | zencommand | zCommandExistenceTest | test -f % | Command Existence Test | / |
| | | | | Yes | zencommand | zCommandLoginTimeout | 10 | Timeout for Login (seconds) | Sp... |
| | | | | Yes | zencommand | zCommandLoginTries | 1 | Command Login Tries | Set... |
| | | | | Yes | zencommand | zCommandPassword | | Password | Sp... |
| | | | | Yes | zencommand | zCommandPath | \$ZENHOME/binexec | Command Path | Set... |
| | | | | Yes | zencommand | zCommandPort | 22 | Command Port | Sp... |
| | | | | Yes | zencommand | zCommandProtocol | ssh | Command Protocol | Est... |
| | | | | Yes | zencommand | zCommandSearchPath | | Command Search Path | Set... |
| | | | | Yes | zencommand | zCommandUserCommandTimeout | 15 | Timeout for User Commands (seconds) | Sp... |
| | | | | Yes | zencommand | zCommandUsername | | Username | Sp... |
| | | | | Yes | Control Center | zControlCenterHost | \$[here/manageip] | zControlCenterHost | / |
| | | | | Yes | Control Center | zControlCenterModelCycle | 3600 | zControlCenterModelCycle | / |
| | | | | Yes | Control Center | zControlCenterPassword | | zControlCenterPassword | / |
| | | | | Yes | Control Center | zControlCenterPerfCycle | 300 | zControlCenterPerfCycle | / |
| | | | | Yes | Control Center | zControlCenterPort | 443 | zControlCenterPort | / |
| | | | | Yes | Control Center | zControlCenterUser | | zControlCenterUser | / |

- 3 In the right pane, double-click a property to be edited.
- 4 In the **Edit Config Property** dialog box, change definitions and then click **SUBMIT**.

Groups

Groups are functional divisions that allow you to assign attributes to multiple objects with similar functions. Groups can be used, for example, to arrange objects along departmental lines. Groups do not appear on the Dashboard.

Adding a group

To add a group:

- 1 Select **Infrastructure** from the Navigation menu. The device list appears.
- 2 Select the parent location in the groups hierarchy where you want to create a child group.
- 3 Click the **Add** icon. The Add Group dialog box appears.
- 4 Enter a name and description for the group, and then click **Submit**. The group appears in the hierarchy. You can now move devices to this group. Select one or more devices in the device list, and then drag them to the new group.

Moving a group

To move a group:

- 1 Select the group in the hierarchy.
- 2 Drag the group to its new location. The Move Organizer confirmation dialog appears.
- 3 Click **OK** to confirm the action. The group appears at its new location in the hierarchy.

Systems

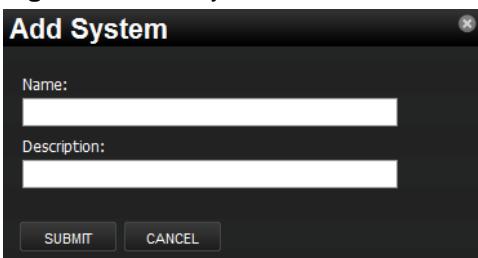
Systems are intended to follow virtual setups, such as those in a network setup or systems grouped by functionality.

Adding a systems

To add a system or sub-system:

- 1 Select **Infrastructure** from the navigation menu. The device list appears.
- 2 Select the parent location in the systems hierarchy where you want to create a new child system.
- 3 Click the **Add** icon. The Add System dialog box appears.

Figure 88: Add System



- 4 Enter a name and description for the system, and then click **Submit**. The system appears in the hierarchy. You can now move devices to this system. Select one or more devices in the device list, and then drag them to the new system.

Moving a system

To move a system:

- 1 Select the system in the hierarchy.
- 2 Drag the system to its new location. The Move Organizer confirmation dialog appears.
- 3 Click **OK** to confirm the action. The system appears at its new location in the hierarchy.

Locations

Locations are logical groupings for physical systems. They can be as general as city and state, or as specific as rack or closet. Locations do not appear on the Dashboard.

Adding a locations

To add a location:

- 1 Select **Infrastructure** from the Navigation menu. The device list appears.
- 2 Select the parent level in the Locations hierarchy where you want to create a child location.
- 3 Click the **Add** icon. The Add Location dialog box appears.
- 4 Enter a name and description for the location, and then click **Submit**. The location appears in the hierarchy. You can now move devices to this location. Select one or more devices in the device list, and then drag them to the new location.

Moving a location

To move a location:

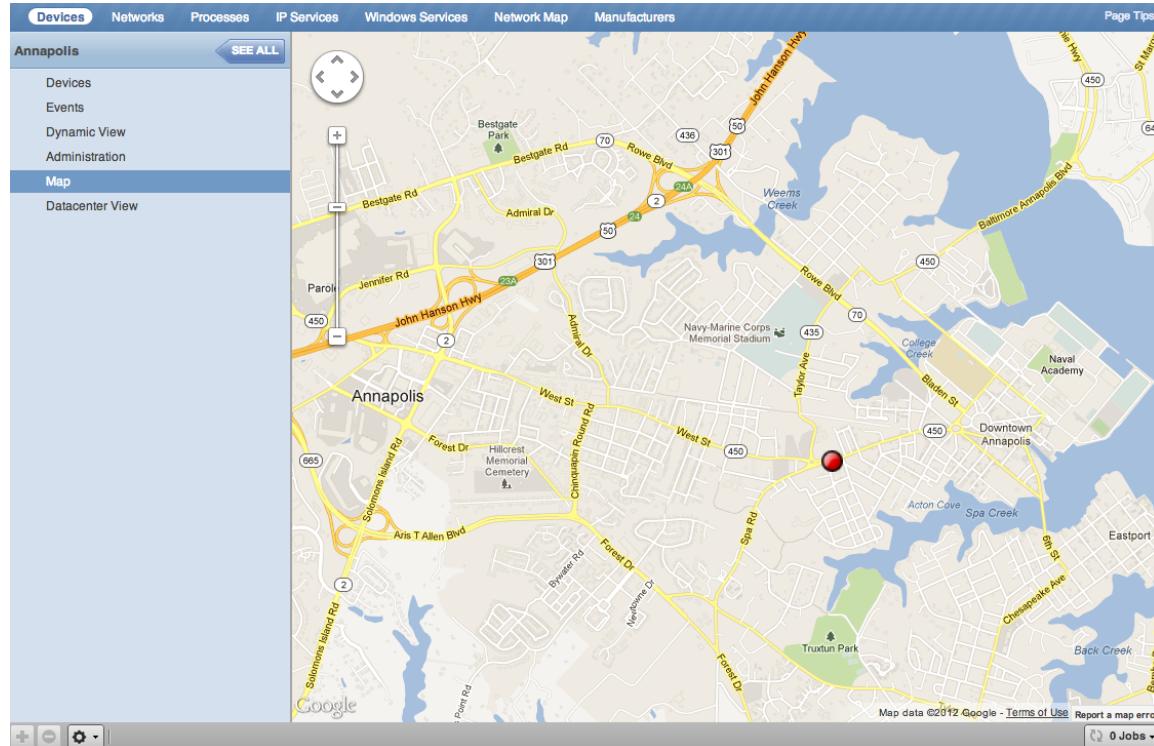
- 1 Select the location in the hierarchy.
- 2 Drag the location to its new place. The Move Organizer confirmation dialog appears.
- 3 Click **OK** to confirm the action.

Integration with Google Maps

The system can map locations by using a Google Maps mashup feature that sets the location's Address property to a valid Google Maps address. The selected location appears on the map as a dot. The color of the dot represents the highest severity of any event on any device in that location.

Network connections that span locations are represented on the map by lines. Each line color matches the status of the connection it represents.

Figure 89: Network location - Google Map



To view the Google Map for a network location:

- 1 From the Navigation menu, select Infrastructure.
- 2 The device list appears.
- 3 In the hierarchy, select a location.
- 4 Click **Details**.
- 5 Select **Map**.

The network map appears.

Note You also can view the network location map from the Google Maps portlet on the dashboard.

Enabling Google Maps

Before you can use the Google Maps feature, you must specify a Google Maps API key. For more information, refer to the [Google API Console](#). Enter the value in the **Google Maps API Key** field on the **Advanced > Settings** page.

Setting an address for a location

To set a location address:

- 1 Select **Infrastructure** from the navigation menu. The device list appears.
- 2 Select a location in the hierarchy.
- 3 Select **Edit** from the **Action** icon. The Edit Organizer dialog box appears.
- 4 In the Description field, enter a description for the location.
- 5 In the Address field, enter a complete address that can be resolved by [Google Maps](#). The address must include a valid zip code.
- 6 Click **Submit**. The selected address for the location is created. You must add at least one device to the location for the location "dot" to appear on the map.

Clearing the Google Maps cache

Sometimes there are issues with drawing the maps and seeing the network status of locations or connections. Clearing the Geocode cache will solve these problems. To clear the Geocode cache:

- 1 Select **Infrastructure** from the Navigation menu. The device list appears.
- 2 Select a location in the hierarchy.
- 3 Select **Clear Geocode Cache** from the **Action** icon. A confirmation dialog box appears.
- 4 Click **OK**.

Network links

If two devices in the same network are in different map-able locations, a line is on the map representing a network connection between the two. If there are multiple separate network connections between the same two locations, only one line is drawn. The color of the line represents the highest severity of any events affecting the connection. These are determined by:

- A ping down event on the device at either end of the connection; or
- Any event on the interface at either end of the connection.

Drawing map links (zDrawMapLinks configuration property)

Calculating network links on the fly is an time-intensive procedure. If you have a large number of devices that have been assigned locations, drawing those links on the map may take a long time.

To save time, you can tell the system not to attempt to draw links for specific networks. You might want to do this, for example, for a local network comprising many devices that you know does not span multiple locations.

To edit the value for this property:

- 1 Select **Infrastructure > Networks** from the Navigation menu. The Networks page appears.
- 2 Select a network or sub-network for which you want to disable map links.
- 3 Display configuration properties for the network.
- 4 Double-click the zDrawMapLinks configuration property in the list. The Edit Config Property dialog box appears.
- 5 De-select the value (uncheck the box), and then click **Submit**.

Note This setting will be inherited by networks or sub-networks below this selection in the hierarchy. If you have few networks for which links would be drawn, you might want to disable map links on /Networks, enabling it only on a network where you know a location-spanning WAN connection exists.

Google Maps example

This example will show you how to:

- Create and display Google map links of devices
- Send a test event to see how map links are affected by system changes

- 1 Disable map links. (Refer to the procedure in Drawing Map Links for instructions.)
- 2 Create two locations: "New York" and "Los Angeles." (Refer to the procedure in Adding Locations for instructions.)
- 3 Enter the following values in the Address field of the Add Location dialog: "New York, NY" and "Los Angeles, CA" respectively.
- 4 Set the location of a device to New York. Locate another device on the same network and set its location to Los Angeles.
- 5 Select Locations in the hierarchy, click **Details**, and then select Map.

New York and Los Angeles are represented by dots on the map; however, no link is drawn between these locations.

- 6 Select Networks and re-enable map linking.
- 7 Select Infrastructure, then select Locations in the hierarchy.
- 8 Click **Details**, and then select Map.

A green line is now drawn between New York and Los Angeles.

- 9 Send an event with a severity of Critical to the device in New York. (For information about creating events, refer to the chapter titled Event Management.) Do not specify a component.
- 10 Return to the Locations map.

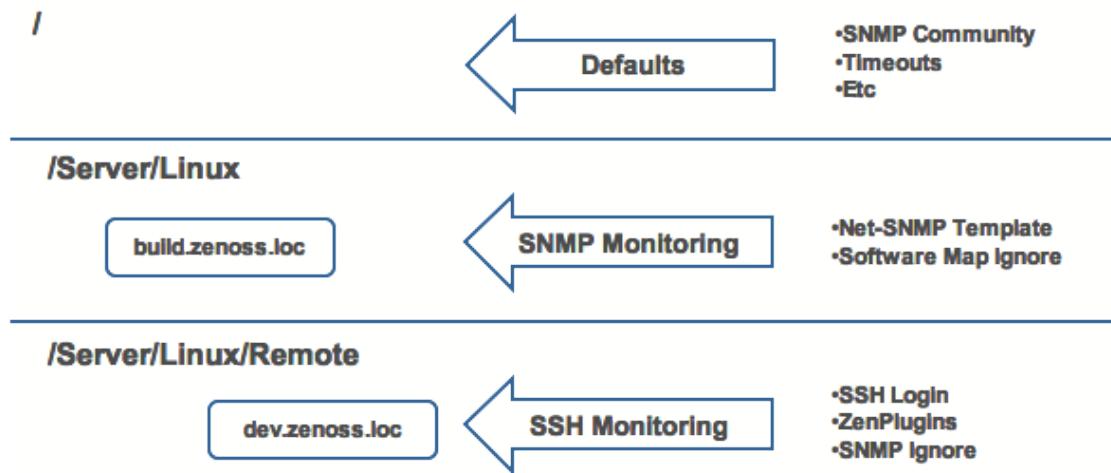
The dot representing New York is now red, but the link between New York and Los Angeles remains green.

- 11 Navigate to the New York device and determine the ID of the component that is connected to the network shared with the Los Angeles device.
- 12 Send another test event, this time specifying that component.
- 13 Return to the Locations map.

The link between New York and Los Angeles is now red.

Inheritance

Inheritance is defined by how many attributes are applied to a device at different levels in the device hierarchy. The following diagram shows an example of how and where configuration properties can be set throughout the device class tree.

Figure 90: Device class tree and inheritance

In this example, you can see that the default properties can be set at the highest level (/). However, as you travel further down the hierarchy, you see that you can override any of the configuration properties that are set at the root level.

The next two lines show how the device tree further defines properties for Linux servers. For example, to set up and use SNMP monitoring for all Linux servers (inclusive of) build.zenoss.loc, you could change these properties at the /Server/Linux level.

Further, if you wanted to change how you collect information for remote Linux servers, you could create a sub-group in /Server/Linux called /Server/Linux/Remote, setting these servers to use SSH monitoring and changing the associated properties for that sub-group.

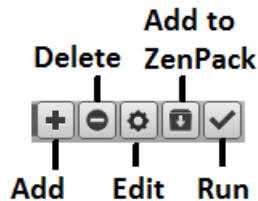
All of these configuration properties and groupings co-exist, with any changes made lower in the hierarchy taking priority.

User commands

User commands allow you to execute arbitrary shell commands from Zenoss Core. A user command is executed in its appropriate container rather than the remote device unless the command explicitly uses SSH to connect to the remote device.

You can define and run user commands on a device or organizer (device class, system, group, or location). You also can define commands globally. The User Commands menu bar shows the various functions that can be used in the User Commands screen. See the following sections for detailed instructions on adding and running user commands on specific devices or groups of devices.

Figure 91: User commands menu bar



Defining global user commands

Global commands appear in the Commands list of options located at the top of the Devices page.

To define global user commands:

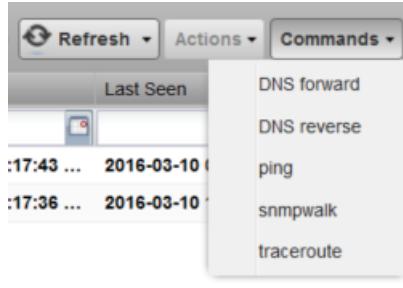
- 1 Select **Advanced > Settings** from the Navigation menu.
 - 2 In the left panel, select **Commands**.
 - 3 In the Define Commands area, select **Add User Command** from the Action menu. The Add User Command dialog box appears.
 - 4 Enter a name for the command, and then click **OK**. Only letters, numbers, and underscores are allowed in command names. Spaces are not allowed. The Define Commands page appears.
 - 5 In the Description field, enter a description of what the command will do.
 - 6 In the Command section, enter the TALES expression-based command you want to run on the device.
 - 7 Enter your system account password for confirmation, and then click **Save**.
- The command is saved and added to the commands menu.

Global commands also can be edited from a specific device. Changes to a global command from a device are not limited to that device.

Running global user commands

To run a global user command, select one or more devices in the devices list, and then select a command from the Commands list of options.

Figure 92: Global user commands

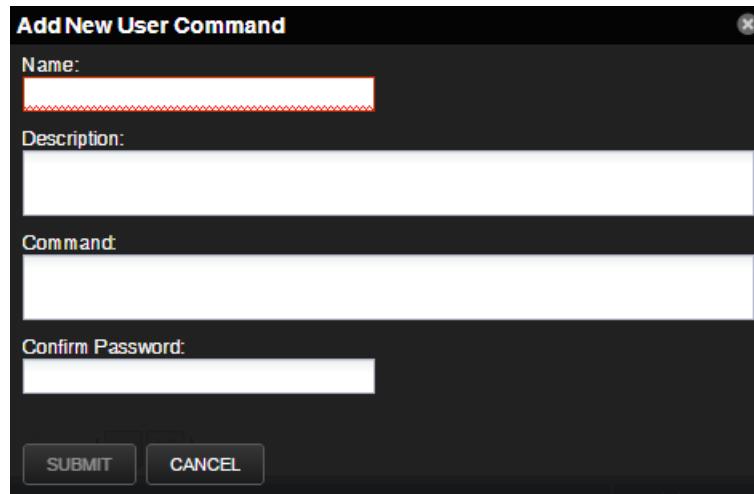


Defining user commands for a single device

To define a user command for a device:

- 1 Select **INFRASTRUCTURE** from the Navigation menu. The Devices page appears.
- 2 Click a device name from the list to open the Device Overview page.
- 3 In the left panel, select **Device Administration**.
- 4 In the User Commands area, click the **Add a User Command** icon. The Add New User Command dialog box appears.

Figure 93: Add New User Command



- 5 Enter the following information about the user command:
 - **Name** - Name of the user command.
 - **Description** - Description of what the command will do.
 - **Command** - TALES expression-based command you want to run.
 - **Confirm Password** - Enter your system account password for confirmation.
- 6 Click **Submit**. The command is saved and added to the user commands menu.

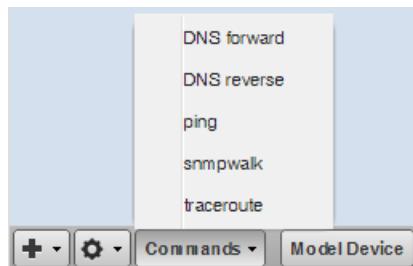
- 7 Optional: Test the command by selecting the command from the list and clicking the **Run** icon.

Running user commands for a single device

To run a command defined for a single device:

- 1 Navigate to the **Infrastructure > Devices** page.
- 2 Click the device name in the device list to open the Device Overview page.
- 3 Select the command from the Commands list of options located at the bottom of the page.

Figure 94: Global User Commands Menu

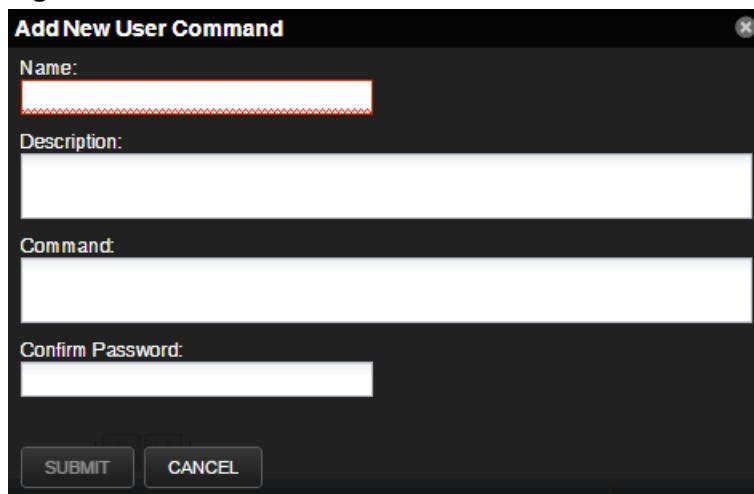


Defining user commands for all devices in an organizer

To define a user command for all devices in an organizer:

- 1 On the **INFRASTRUCTURE** page, select a device organizer in the devices hierarchy; for example, / Server/Linux.
- 2 Click **Details**.
- 3 In the left panel, select **Device Administration**.
- 4 In the User Commands area, click the **Add a User Command** icon. The Add New User Command dialog box appears.

Figure 95: Add New User Command



- 5 Enter the following information about the user command:
 - **Name** - Name of the user command.
 - **Description** - Description of what the command will do.
 - **Command** - TALES expression-based command you want to run.

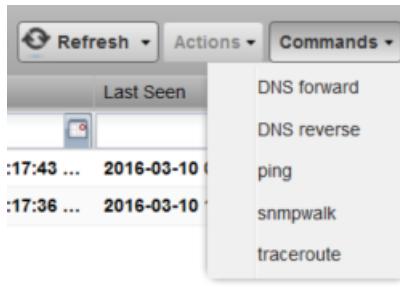
- **Confirm Password** - Enter your system account password for confirmation.
- 6 Click **Submit**. The command is saved and added to the user commands menu.
- 7 Optional: Test the command by selecting the command from the list and clicking the **Run** icon.

Running user commands for devices in an organizer

To run a command defined for devices in an organizer:

- 1 On the Infrastructure page, select a device organizer.
- 2 Select one or more devices in the filtered view.
- 3 Select the command from the Commands list located at the top of the page.

Figure 96: Global user commands



User command example: Echo command

This example shows how to create an echo user command. You can see the use of TALES expressions in the definition of this command.

- 1 Add a command called "echoDevice"
- 2 In the command definition, echo the name and IP address of the device:

```
echo name = ${here/id} ip = ${here/manageIp}
```

In a TALES expression, `here` is the object against which the expression is executed. Some TALES expressions in the system have other variables (such as `evt` for event, and `dev` or `device` for the device). See the Appendix titled TALES Expressions for more information about TALES expressions syntax.

- 3 Select a device and then run the command.
- 4 Edit the command to add more information:

```
echo name = ${here/id} ip = ${here/manageIp} hw = ${here/getHWProductName}
```

- 5 Run the command against a group of devices and view the command outputs.

Managing users

Every user within Zenoss Core has a unique user ID, which allows an administrator to assign group permissions and alerting rules that are unique to each user. Unique IDs also help ensure secure access to the system.

To create and manage user accounts, you must be logged in to the system `admin` account, or as a user with extended privileges.

Creating user accounts

To create a user account:

- 1 From the Navigation menu, select **ADVANCED**. The Settings page appears.
- 2 In the left panel, select **Users**. The users and groups administration page appears.
- 3 From the Action icon, select **Add New User**. The Add User dialog appears.
- 4 In the Username field, enter a unique name for the account.
- 5 In the Email field, enter the user account email address. Any alerts that you set up for this user will be sent to this address.
- 6 Click **OK**. The user appears in the User List.

After creating the account, edit the account to provide a password and additional user details.

Editing user accounts

To access and edit user account information:

- 1 In the Users list, click the name of the user you want to edit. The edit user page appears. The following example shows the admin user.

Figure 97: Edit user

The screenshot shows the 'Edit user' page for the 'admin' account. At the top, there's a message 'State at time: 2017-10-12 20:06:01' and a link to 'Reset Password'. Below that is a 'USER PREFERENCES' section with a link to 'Reset Preferences'. The main area is titled 'USER SETTINGS' and contains the following fields:

- Roles:** A dropdown menu showing 'Manager', 'ZenManager', 'ZenOperator', and 'ZenUser'. 'ZenUser' is selected.
- Groups:** A list box containing a single item.
- Email:** An input field containing an email address.
- Pager:** An input field containing a pager number.
- Default Page Size:** A dropdown menu set to '40'.
- Default Admin Role:** A dropdown menu set to 'ZenUser'.
- Network Map Start Object:** An input field.
- Time Zone:** A dropdown menu set to 'America/Chicago'.
- Date format:** A dropdown menu set to 'year/month/day'.
- Time format:** A dropdown menu set to '24h'.
- Set New Password:** An input field.
- Confirm New Password:** An input field.
- Current Password for admin:** An input field.

At the bottom right is a 'Save Settings' button.

2 Make changes to one or more settings:

- **Reset Password** - Facilitates user self-service by allowing a user to reset his or her own password. Click to reset and email the new password to the email address associated with the user's account.
- **User Preferences** - Resets all preferences such as grid columns and filters to their default values.
- **Roles** - Assign one or more roles (user privileges) to the user. To edit or assign roles, you must be a system Admin or be assigned the Manager role. For more information about user roles, and for a list of available roles and the privileges they provide, see [Roles](#) on page 156.
- **Groups** - Specify one or more groups to which this user belongs.
- **Email** - Enter the user's email address. To verify that the address is valid, click the test link.
- **Pager** - Enter the user's pager number.
- **Default Page Size** - Controls how many entries (by default) appear in tables. Enter a value for the default page size. The default value is 40.
- **Default Admin Role** - Select the default role that this user will have for administered objects associated with him or her.
- **Network Map Start Object** - Specify the default view for this user in the network map.
- **Time Zone** - Specify the time zone to be displayed on all charts and graphs within the product.
- **Date format** - Select the date format to use on all charts and graphs within the product. The default format is **year/month/day**.
- **Time format** - Select the time format. Options include 24-hour time (HH : mm : ss) or 12-hour time using a.m. and p.m. (hh : mm : ss a).
- **Set New Password / Confirm New Password** - Enter a new password for the user and confirm the entry.

3 Enter your password, and then click **Save Settings** to confirm and save the changes for the user.

Associating objects with specific users

You can associate any object in the system with a particular user, for monitoring or reporting purposes. Once associated with a user, you can then assign the user a specific role that applies to his privileges with respect to that object.

For more information about object-specific roles, see [Roles](#) on page 156.

To create an object association:

- 1 From **ADVANCED > Settings**, select **Users** in the left panel.
- 2 Click the name of a user.
- 3 From the Edit page, select **Administered Objects** in the left panel. The list of administered objects appears.

Figure 98: Administered objects - add object



- 4 Select an object type from the Administered Objects Action menu. You can add:
 - Device
 - Device class
 - System
 - Group
 - Location
- 5 Specify the component you want to add as an administered object, and then click **OK**. The object appears in the Administered Devices list for the user.

Figure 99: Administered Objects - Objects Added



- 6 Optional: Change the role that is associated for this user on this object.

Note The default role assigned to the user for an administered object is specified by the Default Admin Role field on the Edit page.

- 7 Click **Save** to save changes.

Adding administrators

You also can associate an object with a user by adding an administrator to the object. Perform the following:

- 1 Navigate to the object you want to add to the user's list of administered objects.
- 2 Select **Device Administration**.

Figure 100: Administered objects - add administrator

| Name | Role | Email | Pager |
|------------|---------|---------|-------|
| cgilchrist | Manager | cgil... | |

- 3 Click the **Add Administrator** icon in the Administrators area. The Add Administrator dialog box appears.
- 4 Select an administrator from the list and change the role if desired, then click **SUBMIT**. The administrator appears in the object's Administrators list. The object is added to the administrator's Administered Objects list.

User groups

Zenoss Core allows you to create user groups. By grouping users, you can aggregate rules and apply them across multiple user accounts.

Viewing user groups

To view user groups, select **ADVANCED > Settings**, and then select **Users** from the left panel.

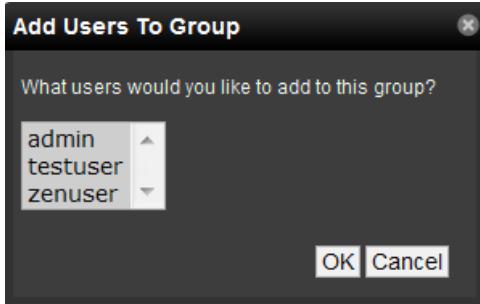
The groups area shows each user group and the users assigned to that group.

Creating user groups

You can create user groups to aggregate rules and apply them across multiple user accounts.

To create a user group:

- 1 Navigate to **ADVANCED > Settings**.
- 2 In the left panel, select **Users**. The Users page appears.
- 3 From the Groups area Action menu, select **Add New Group**. The Add Group dialog box appears.
- 4 In the Group field, enter a name for this user group, and then click **OK**. The group name appears in the Groups list.
- 5 Click the name of the group you created. The Users in Group page appears.
- 6 From the Action menu, select **Add User**. The Add User to Group dialog box appears.

Figure 101: Add User to Group

- 7 From the User list of selections, select one or more users you want to add to the group, and then click **OK**. The user or users you select appear in the list of users for this group.

You also can choose administered objects and alerting rules for this user group. These alerting rules will apply to all users in the group. The user's original alerting rules and objects will also apply.

Roles

A role is a group of permissions that you can assign to users or groups.

The following table lists available roles.

| Role | Permissions |
|------------|---|
| ZenUser | Provides global read-only access to system objects. |
| ZenManager | Provides global read-write access to system objects. |
| Manager | Provides global read-write access to system objects. Additionally provides read-write access to the Zope object database. |

Device access control lists

About device access control lists (ACL)

Zenoss Core supports fine-grained security controls. For example, this control can be used to give limited access to certain departments within a large organization or limit a customer to see only his own data. A user with limited access to objects also has a more limited view of features within the system. As an example, most global views, such as the network map, event console, and all types of class management, are not available. The device list is available, as are the device organizers: systems, groups, and locations. A limited set of reports can also be accessed.

Permissions and roles

Actions in the system are assigned permissions. For instance to access the device edit screen you must have the “Change Device” permission. Permissions are not assigned directly to a user; instead, permissions are granted to roles, which are then assigned to a user. A common example is the ZenUser role. Its primary permission is “View,” which grants read-only access to all objects. ZenManagers have additional permissions such as “Change Device,” which grants them access to the device edit screen. When you assign a role to a user using the Roles field (on the Edit page), it is global.

Administered objects

Device ACLs provide limited control to various objects within the system. Administered objects are the same as the device organizers: Groups, Systems, and Locations and Devices. If access is granted to any device organizer, it flows down to all devices within that organizer. To assign access to objects for a restricted user, you must have the Manager or ZenManager roles. The system grants access to objects is granted using the user's or user group's administered objects. To limit access, you must not assign a "global" role to the user or group.

Users and groups

Users and user groups work exactly as they would normally. See the section in the User Management section of this guide dealing with users and groups.

Assigning administered object access

For each user or group there is an Administered Objects selection, which lets you add items for each type of administered object. After adding an object you can assign it a role. Roles can be different for each object, so a user or group might have ZenUser on a particular device but ZenManager on a location organizer. If multiple roles are granted to a device though direct assignment and organizer assignment the resulting permissions will be additive. In the example above, if the device was within the organizer the user would inherit the ZenManager role on the device.

Portlet access control

Within Zenoss Core, portlet access can be controlled. This is important for device ACLs.

Example: Restricted user with ZenUser role

To create a restricted user with a ZenUser role:

- 1 As admin or any user account with Manager or ZenManager role, create a user named acltest. Set a password for the user.
- 2 Make sure that no role is assigned to the user.
- 3 Edit the user's administered objects.
- 4 Add an existing device to the user. The device's role will default to ZenUser.
- 5 Log out of your browser, or open a second browser and then log in as acltest.
- 6 Select **INFRASTRUCTURE**. You should see only the device you assigned to acltest.
- 7 Navigate to the device and notice that the edit capabilities are not available. This is because you are in read-only mode for this device.

Example: Restricted user with ZenManager role

To restrict a user to a specific role:

- 1 Change the acltest user's role to "ZenManager" on the device. (You must do this as a user with ZenManager global rights.)
- 2 Go back to the acltest user's administered objects and set the role on the device to ZenManager.
- 3 As acltest, navigate back to the device. You now have access to edit the device.

Example: Adding device organizers

To add a device organizer:

- 1 Go to Groups and create a group called "RestrictGroup."
- 2 Go to the acltest user's administered objects and add the group to the user.

- 3 Logged in as acltest, notice that groups can be added to a user.
- 4 Place a device within this group and as acltest you should not only see the device within the group but also in the device list.

Restricted user organizer management

- 1 Give the acltest user ZenManager on your restricted group.
- 2 As acltest, you can now add sub-organizers under the restricted group.

Viewing events

A user in restricted mode does not have access to the global event console. The available events for the user can be seen under his organizers.

Detailed restricted screen functionality

Dashboard

By default, the dashboard is configured with three portlets:

- Object Watch List
- Device Issues
- Production State

These have content that will be restricted to objects for a given user.

Device list

The device list is automatically filtered to devices of a restricted user scoped to accessible devices. No menu items are available.

Device organizers

Device organizers control groups of devices for a restricted user. Every device added to the group will be accessible to the user. Permissions will be inherited down multiple tiers of a device organizer.

Reporting

Reports are limited to device reports and performance reports.

Reporting

The **REPORTS** tab of the Zenoss Core browser interface provides summaries of monitored resources in a tree view. Expand an organizer to see reports in that category.

You can organize reports and the display order of the report organizers by drag-and-drop within the tree view.

Figure 102: Reports list

| Name | Class | First Seen | Collection | Change |
|--|--------------------------------|---------------------|---------------------|---------------------|
| qa-centos-7-events-ssh_zenoss_lab | /Server/SSH/Linux | 2016-10-20 18:16:45 | 2016-10-20 18:17:22 | 2016-10-20 18:17:20 |
| Juniper_J2350 | /Network/Juniper | 2016-10-20 17:50:00 | 2016-10-20 17:50:20 | 2016-10-20 17:52:14 |
| Solaris_5_10 | /Server/Linux | 2016-10-20 17:45:59 | 2016-10-20 17:46:18 | 2016-10-20 17:48:02 |
| ucs1_zenoss_loc | /CiscoUCS/UCS-Manager | 2016-10-20 16:19:18 | 2016-10-20 16:20:29 | 2016-10-20 16:20:18 |
| ucs1-4-7_zenoss_loc | /Server/Microsoft/Windows | 2016-10-20 15:15:21 | 1970-01-01 00:00:00 | 2016-10-20 15:15:41 |
| parf2-vcenter.zenoss_loc | /Sphere | 2016-10-20 15:09:45 | 2016-10-20 15:13:27 | 2016-10-20 18:22:31 |
| qa-ubuntu-12_zenoss_loc | /Server/Linux | 2016-10-20 15:06:16 | 2016-10-20 15:06:32 | 2016-10-20 15:06:32 |
| qa-suse-12_zenoss_lab | /Server/SSH/Linux | 2016-10-20 15:03:53 | 2016-10-20 15:05:13 | 2016-10-20 15:05:09 |
| qa-rhel7_zenoss_loc | /Server/SSH/Linux | 2016-10-20 15:02:02 | 2016-10-20 15:02:48 | 2016-10-20 15:02:42 |
| test-rhel6_zenoss_loc | /Server/Linux | 2016-10-20 15:00:37 | 2016-10-20 15:00:53 | 2016-10-20 15:00:52 |
| test-rhel5_zenoss_loc | /Server/Linux | 2016-10-20 14:59:18 | 2016-10-20 14:59:35 | 2016-10-20 14:59:35 |
| qa-cisco-Nexus-5000-snmp-v3_zenoss_loc | /Network/Cisco/Nexus/5000 | 2016-10-20 14:42:00 | 2016-10-20 14:44:54 | 2016-10-20 14:44:47 |
| qa-cisco-Nexus-7000-snmp-v3_zenoss_loc | /Network/Cisco/Nexus/7000 | 2016-10-20 14:40:11 | 2016-10-20 14:40:44 | 2016-10-20 14:40:40 |
| N9001-2 | /Network/Cisco/Networking/8000 | 2016-10-20 14:35:49 | 2016-10-20 14:38:52 | 2016-10-20 14:37:42 |
| nexus-5k_zenoss_loc | /Network/Cisco/Nexus/5000 | 2016-10-20 14:33:31 | 2016-10-20 14:34:12 | 2016-10-20 14:34:11 |
| CUSQU1001CN3B5 | /Network/Cisco/Nexus/3000 | 2016-10-20 14:30:55 | 2016-10-20 14:32:08 | 2016-10-20 14:31:45 |
| N1kv-Orc8 | /Network/Cisco/Nexus/1000V | 2016-10-20 14:29:18 | 2016-10-20 14:29:38 | 2016-10-20 14:29:38 |
| Cisco_10_171_100_14 | /Network/Cisco/6500 | 2016-10-20 14:26:51 | 2016-10-20 14:28:11 | 2016-10-20 14:27:49 |
| 10.87.254.1 | /Network/Cisco | 2016-10-20 14:24:23 | 2016-10-20 14:25:37 | 2016-10-20 14:25:32 |
| perf3-switch_zenoss_loc | /Network/Cisco | 2016-10-20 14:22:51 | 2016-10-20 14:23:18 | 2016-10-20 14:23:16 |
| perf2-switch_zenoss_loc | /Network/Cisco | 2016-10-20 14:21:09 | 2016-10-20 14:21:45 | 2016-10-20 14:21:43 |
| qa-centos-7-snmp-v3_zenoss_loc | /Server/Linux | 2016-10-20 14:16:01 | 2016-10-20 14:18:27 | 2016-10-20 14:18:27 |
| qa-centos-7_zenoss_loc | /Server/SSH/Linux | 2016-10-20 14:13:52 | 2016-10-20 14:58:49 | 2016-10-20 14:58:48 |
| qa-centos-7-snmp_zenoss_loc | /Server/Linux | 2016-10-20 14:13:06 | 2016-10-20 14:13:21 | 2016-10-20 14:13:21 |
| qa-centos-6_zenoss_loc | /Server/SSH/Linux | 2016-10-20 14:11:11 | 2016-10-20 14:11:55 | 2016-10-20 14:11:47 |
| qa-centos-6-snmp-v3_zenoss_loc | /Server/Linux | 2016-10-20 14:06:54 | 2016-10-20 14:09:04 | 2016-10-20 14:09:04 |
| qa-centos-6-snmp_zenoss_loc | /Server/Linux | 2016-10-20 14:05:27 | 2016-10-20 14:05:43 | 2016-10-20 14:05:42 |
| qa-centos-5_zenoss_loc | /Server/SSH/Linux | 2016-10-20 14:03:30 | 2016-10-20 14:04:12 | 2016-10-20 14:04:06 |
| 10.171.54.15 | /Server/Linux | 2016-10-20 14:00:05 | 1970-01-01 00:00:00 | 2016-10-20 14:00:05 |
| qa-centos-5-snmp_zenoss_loc | /Server/Linux | 2016-10-20 13:59:20 | 2016-10-20 13:59:41 | 2016-10-20 13:59:40 |
| resmar-510_zenoss_loc | /ControlCenter | 2016-10-20 13:56:32 | 2016-10-20 13:58:07 | 2016-10-20 13:57:22 |

Troubleshooting problems with report generation

If you experience stair-stepping in graphs, consider changing the reporting collection interval in Zenoss Core. For example, setting the reporting collection interval to 60 minutes tells Zenoss Core to update the API-driven reporting data at that interval, which is different from the native collection interval.

Organizing reports

You can organize reports by creating organizers and moving reports into them. You can create report organizers at multiple levels, even within another organizer. To create a report organizer:

- 1 Select an existing organizer or the top of the reports hierarchy, and then click **Add**.
- 2 Click **Add Report Organizer**.
- 3 In the **Create Report Organizer** dialog box, enter the name of the new report organizer, and then click **Submit**.
The report organizer appears in the tree view.
- 4 Move reports into the organizer, or create new reports.

Device reports

All Devices

A summary of each device that Zenoss Core is monitoring.

| Column | Content |
|----------------|---|
| Name | The name of the device. |
| Class | The Zenoss Core device class associated with the device. |
| Product | The hardware model information associated with the device, which is provided by the device's SNMP MIB, or entered manually. If the value in this column is an SNMP OID, the Zenoss Core database does not include a definition of the object. |
| State | The device's production state. Valid states include Production, Pre-Production, Test, and Maintenance. |
| Ping | The result of the most recent ping of the device. |
| SNMP | The result of the most recent attempt to gather data through the device's SNMP agent. |

All Monitored Components

A summary of each component Zenoss Core is monitoring.

| Column | Content |
|--------------------|--|
| Device | The name of the device which contains the component, with a link to its overview page. |
| Component | The name of the component, with a link to its overview page. |
| Type | The class associated with the component. |
| Description | A description of the component; typically, the component's name. |
| Status | The state of the component as of the most recent attempt to gather monitoring data. |

Device Changes

A summary of the devices in which changes were detected during the most recent collection of model data.

| Column | Content |
|-------------------|---|
| Name | The name of the changed device, with a link to its overview page. |
| Class | The Zenoss Core device class associated with the device. |
| First Seen | The timestamp of the initial collection of modeling data for the device. |
| Collection | The timestamp of the collection in which a change was detected before the most recent collection. |
| Change | The timestamp of the most recent collection in which a change was detected. |

MAC Addresses (MAC Address Inventory)

A list of the unique device name, interface ID, and MAC address combinations in the Zenoss Core database.

| Column | Content |
|---------------------|--|
| Device | The name of a device, with a link to its overview page. |
| Interface ID | The ID of a network interface, with a link to its overview page. |
| MAC address | A MAC address. |

Model Collection Age

A summary of devices that were not available for modeling data collection during the most recent 48 hour period.

| Column | Content |
|-------------------|--|
| Name | The name of the changed device, with a link to its overview page. |
| Class | The Zenoss Core device class associated with the device. |
| First Seen | The timestamp of the initial collection of modeling data for the device. |
| Collection | The timestamp of the most recent collection of modeling data. |
| Change | The timestamp of the most recent change in modeling data for the device. |

New Devices

The list of devices that were discovered and added to Zenoss Core recently.

| Column | Content |
|-------------------|--|
| Name | The name of the changed device, with a link to its overview page. |
| Class | The Zenoss Core device class associated with the device. |
| First Seen | The timestamp of the initial collection of modeling data for the device. |
| Collection | The timestamp of the most recent collection of modeling data. |
| Change | The timestamp of the most recent change in modeling data for the device. |

Ping Status Issues

A list of the devices which were down during the most recent collection of monitoring data.

| Column | Content |
|----------------|---|
| Name | The name of the device. |
| Class | The Zenoss Core device class associated with the device. |
| Product | The hardware model information associated with the device, which is provided by the device's SNMP MIB, or entered manually. If the value in this column is an SNMP OID, the Zenoss Core database does not include a definition of the object. |
| State | The device's production state. Valid states include Production, Pre-Production, Test, and Maintenance. |
| Ping | The result of the most recent ping of the device. |
| SNMP | The result of the most recent attempt to gather data through the device's SNMP agent. |

SNMP Status Issues

A list of the devices for which no SNMP agent responded during the most recent collection of monitoring data.

| Column | Content |
|----------------|---|
| Name | The name of the device. |
| Class | The Zenoss Core device class associated with the device. |
| Product | The hardware model information associated with the device, which is provided by the device's SNMP MIB, or entered manually. If the value in this column is an SNMP OID, the Zenoss Core database does not include a definition of the object. |
| State | The device's production state. Valid states include Production, Pre-Production, Test, and Maintenance. |
| Ping | The result of the most recent ping of the device. |
| SNMP | The result of the most recent attempt to gather data through the device's SNMP agent. |

Software Inventory

A list of the software installed in the devices and components which Zenoss Core monitors.

| Column | Content |
|---------------------|---|
| Manufacturer | The name of the company that makes the software product. |
| Product | The name of the software product. |
| Count | The total number of devices or components on which the software is installed. |

Event reports

All EventClasses (All Event Classes)

A list of each item in the event hierarchy in Zenoss Core. Each item (class) includes the total number of subclasses, instances, and events associated with the class.

| Column | Content |
|------------------|---|
| Name | The event class name. |
| Sublasses | The total number of subclasses associated with the event class. |
| Instances | The total number of instances of the class and its subclasses. |
| Events | The total number of events associated with the class. |

All EventMappings (All Event Mappings)

A list of each item in the event mapping hierarchy in Zenoss Core. Each item (event mapping) includes its key and example text, and a count of the events associated with the event mapping.

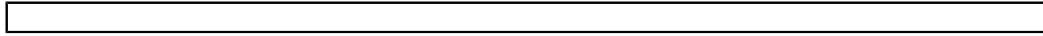
| Column | Content |
|----------------------|---|
| Name | The name of the event mapping, which includes its location in the event class hierarchy, and its key. |
| EventClassKey | The unique identifier of the event mapping. |
| Evaluation | A portion of the example associated with the event mapping. |
| Events | The total number of events associated with the event mapping. |

All Heartbeats

A list of all Zenoss Core daemons, showing the number of seconds elapsed since each daemon sent a heartbeat event.

| Column | Content |
|------------------|--|
| Device | The device on which the daemon is running. |
| Component | The name of the daemon. |
| Seconds | The number of seconds elapsed since the daemon sent a heartbeat event. |

Performance reports



Availability Report

Shows the percentage of time that a device is considered available. You can filter this report on a variety of criteria, including by a time period.

Report filtering

| | | | |
|-----------------|--------------|------------|------------|
| Device Class: | / | Systems: | / |
| Groups: | / | Locations: | / |
| Device Filter: | | Severity: | Error |
| Start Date: | 07/17/2017 | End Date: | 07/24/2017 |
| Event Class: | /Status/Ping | select | |
| Generate | | | |

Device Class

The device class to use for filtering. The default is / (all device classes).

Systems

Select the systems to filter by. The default is / (all systems).

Groups

Select the groups to filter by. The default is / (all groups).

Locations

Select the locations to filter by. The default is / (all locations).

Device Filter

Enter the name of the device to filter by.

Severity

The severity level used in the availability calculation described below. The default is Critical. If another level is wanted, select it from the drop-down list.

Start Date

End Date

The first and last dates of the range of dates to include in the report. To select a date from a calendar, click **select**. The default range is the week ending with the current date.

Event Class

The event class to use for filtering. The default is /Status/Ping.

To generate or refresh the report, click **Generate**.

Note If you export the report by clicking **Export all**, be sure to format the percentage columns to show percentages instead of decimal values.

The percent availability value is calculated by first summing the duration of all events of a particular class with a production state of Production and with a severity greater than or equal to a specified severity in the filter criteria. This sum is then divided by the total duration of the time range, and then subtracted from 1 and multiplied by 100 to get the percent available, as in the following equation:

$$1 - ((\text{Total event down time}) / (\text{total duration})) * 100$$

Note Events whose starttime and lasttime fields are the same are not used in the calculation. These could represent an event that occurs and is subsequently cleared by the next event, or an event that has happened only once in the specific date range.

Report contents

| Column | Content |
|---------|---|
| Device | Name of the device based on the filter parameters selected. |
| Systems | Systems name if applicable to the filter. |

| Column | Content |
|---------------------|---|
| Availability | Total availability of the selected devices. |

CPU Utilization

Shows monitored devices, load averages, % utilization, and forecasted exhaustion. You can customize start and end dates.

Report filtering

| | | | |
|---|------------|----------------|------------|
| Root Organizer: | /Devices | Device Filter: | |
| Start Date: | 07/17/2017 | End Date: | 07/24/2017 |
| Summary Type: | Average | Consolidation: | Average |
| Trendline Type: | Linear | | |
| <input type="button" value="Generate"/> | | | |

Root Organizer

The device class to use for filtering. The default is /Devices.

Device Filter

Enter the name of the device to filter by.

Start Date

End Date

The first and last dates of the range of dates to include in the report. To select a date from a calendar, click **Select**. The default range is the week ending with the current date.

Summary Type

Possible values include: Average, Maximum, Minimum, and Last.

Consolidation

Possible values include: Average and Max.

Trendline Type

Projection algorithm used in Forecasted % Util Exhaustion calculation. Possible value: Linear

To generate or refresh the report, click **Generate**.

Note If you export the report by clicking **Export all**, be sure to format the percentage columns to show percentages instead of decimal values.

This report uses data point aliases. (For more information about data point aliases, see [Data point aliases](#) on page 102.) To add data points to a report, add the alias, and then ensure the values return in the expected units.

| Alias | Expected Units |
|-----------------|----------------|
| loadAverage5min | Processes |
| cpu_pct | Percent |

Report contents

| Column | Content |
|--------|---|
| Device | Name of the device based on the filter parameters selected. |

| Column | Content |
|-------------------|--|
| Load Avg | Average load on the device. |
| % Util | % CPU utilization on the device |
| Forecasted | The amount of time before the exhaustion threshold will be breached. |
| % Util | |
| Exhaustion | |

Filesystem Util Report

Shows mount point, total bytes, used bytes, free bytes, and percentage of utilization for each device. You can customize start and end dates and summary type.

Report filtering

| | | | | | |
|---|------------|----------------|-----------|------------|--------|
| Root Organizer: | /Devices | Device Filter: | | | |
| Start Date: | 07/17/2017 | select | End Date: | 07/24/2017 | select |
| Summary Type: | Average | | | | |
| <input type="button" value="Generate"/> | | | | | |

Root Organizer

The device class to use for filtering. The default is /Devices.

Device Filter

Enter the name of the device to filter by.

Start Date

End Date

The first and last dates of the range of dates to include in the report. To select a date from a calendar, click **select**. The default range is the week ending with the current date.

Summary Type

Possible values include: Average, Maximum, Minimum, and Last.

To generate or refresh the report, click **Generate**.

Note If you export the report by clicking **Export all**, be sure to format the percentage columns to show percentages instead of decimal values.

This report uses data point aliases. (For more information about data point aliases, see [Data point aliases](#) on page 102.) To add datapoints to a report, add the alias, and then ensure the values return in the expected units.

| Alias | Expected Units |
|----------------------------|----------------|
| usedFilesystemSpace__bytes | bytes |

Report contents

| Column | Content |
|--------------------|--|
| Device | Name of the device based on the filter parameters selected. |
| Mount | File systems mount point. Click the link to be taken directly to the device's components page. |
| Total bytes | Amount of total bytes |

| Column | Content |
|------------|----------------------|
| Used bytes | Amount of used bytes |
| Free bytes | Amount of free bytes |
| % Util | Percent utilization |

Interface Utilization

Shows the traffic through all network interfaces monitored by Zenoss Core.

Report filtering

| | | | | | |
|---|------------|----------------|-----------|------------|--------|
| Root Organizer: | /Devices | Device Filter: | | | |
| Start Date: | 07/17/2017 | select | End Date: | 07/24/2017 | select |
| Summary Type: | Average | | | | |
| <input type="button" value="Generate"/> | | | | | |

Root Organizer

The device class to use for filtering. The default is /Devices.

Device Filter

Enter the name of the device to filter by.

Start Date

End Date

The first and last dates of the range of dates to include in the report. To select a date from a calendar, click **select**. The default range is the week ending with the current date.

Summary Type

Possible values include: Average, Maximum, Minimum, and Last.

To generate or refresh the report, click **Generate**.

Note If you export the report by clicking **Export all**, be sure to format the percentage columns to show percentages instead of decimal values.

This report uses data point aliases. (For more information about data point aliases, see [Data point aliases](#) on page 102.) To add data points to a report, add the alias, and then ensure the values return in the expected units.

| Alias | Expected Units |
|--------------------|----------------|
| inputOctets_bytes | bytes/sec |
| outputOctets_bytes | bytes/sec |

Report contents

| Column | Content |
|-----------|--|
| Device | Name of the device based on the filter parameters selected. |
| Interface | Name of interface. Click the link to be taken to the Device Components page. |
| Speed | The interface's rated bandwidth, in bits per second |
| Input | Average traffic coming in to the interface, in bits per second |

| Column | Content |
|---------------|--|
| Output | Average traffic going out of the interface, in bits per second |
| Total | Total average traffic across the interface, in bits per second |
| % Util | Percentage of the interface's bandwidth consumed |

Memory Utilization

Provides system-wide information about the memory usage for devices in Zenoss Core.

Report filtering

The screenshot shows a form for filtering a report. It includes dropdown menus for 'Root Organizer' (set to '/Devices'), 'Summary Type' (set to 'Average'), 'Trendline Type' (set to 'Linear'), and 'Consolidation' (set to 'Average'). There are also input fields for 'Start Date' (07/17/2017) and 'End Date' (07/24/2017). A 'Device Filter' field is present but appears empty. A 'Generate' button is at the bottom left.

Root Organizer

The device class to use for filtering. The default is /Devices.

Device Filter

Enter the name of the device to filter by.

Start Date

End Date

The first and last dates of the range of dates to include in the report. To select a date from a calendar, click **Select**. The default range is the week ending with the current date.

Summary Type

Possible values include: Average, Maximum, Minimum, and Last.

Consolidation

Possible values include: Average and Max.

Trendline Type

Projection algorithm used in Forecasted % Util Exhaustion calculation. Possible value: Linear

To generate or refresh the report, click **Generate**.

Note If you export the report by clicking **Export all**, be sure to format the percentage columns to show percentages instead of decimal values.

The report uses data point aliases. (For more information about data point aliases, see [Data point aliases](#) on page 102.) To add data points to the report, add the alias, and then ensure the values return in the expected units.

| Alias | Expected Units |
|------------------------|----------------|
| memoryAvailable__bytes | bytes |
| memoryBuffered__bytes | bytes |
| memoryCached__bytes | bytes |

Report contents

| Column | Content |
|--------------------------|--|
| Device | Name of the device based on the filter parameters selected. |
| Total | Amount of total memory |
| Available | Amount of available memory |
| Cache Memory | Amount of cache memory |
| Buffered Memory | Amount of buffered memory |
| % Util | % memory utilization on the device |
| Forecasted % Util | The amount of time before the exhaustion threshold will be breached. |
| Exhaustion | |

Threshold Summary

Provides information about the devices that are approaching or exceeding their thresholds.

Report filtering

| | | |
|------------------------|-----------------------------|-----------------------------|
| Start Date: 07/17/2017 | select End Date: 07/24/2017 | select Event Class: /Perf ▾ |
| Generate | | |

Start Date

End Date

The first and last dates of the range of dates to include in the report. To select a date from a calendar, click **select**. The default range is the week ending with the current date.

Event Class

The device class to use for filtering. The default is **/Perf**.

To generate or refresh the report, click **Generate**.

Note If you export the report by clicking **Export all**, be sure to format the percentage columns to show percentages instead of decimal values.

Report contents

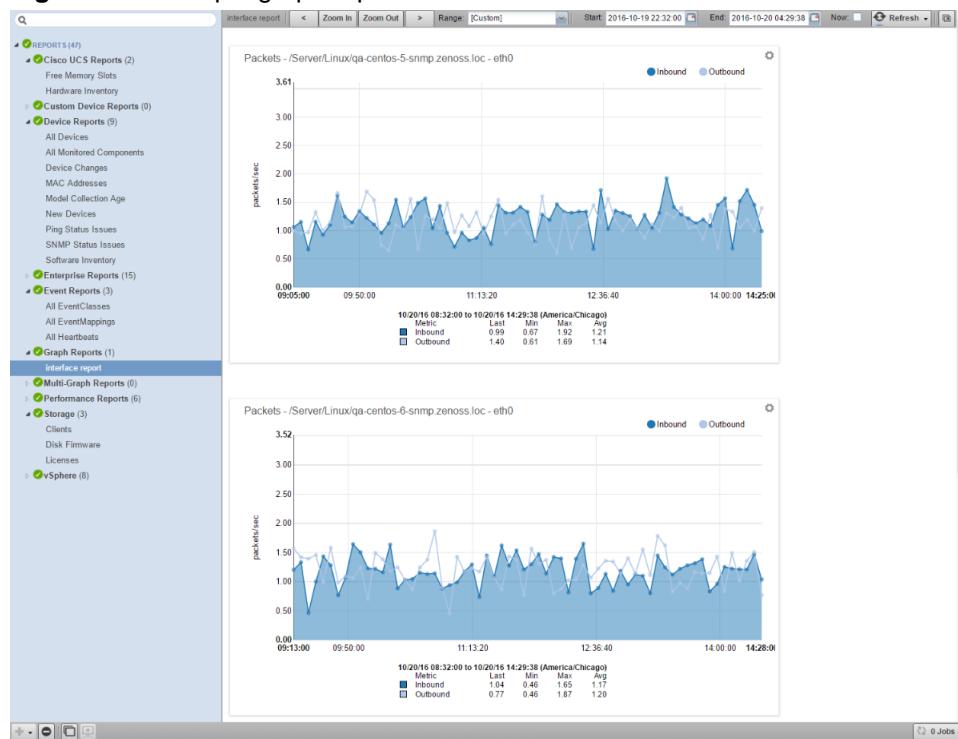
| Column | Content |
|--------------------|--|
| Device | Name of the device based on the filter parameters selected. |
| Component | Network interface component name, if applicable. |
| Event Class | Event class that had a threshold breach. |
| Count | Number of times a threshold was breached. |
| Duration | Amount of time that a threshold was breached. |
| % | Percentage of time that the threshold was breached for the report's time period. |

Graph reports

Graph reports allow you to assemble graphs from devices and device components into a single report. Graph reports only display those graphs that already exist on devices or components in the system. You cannot define or alter graphs in a graph report.

Graph reports are available in two views: a "normal" view (similar to the graph views for devices and device classes) and a print view.

Figure 103: Sample graph report



Creating a graph report

To create a graph report:

- 1 Navigate to **REPORTS > Graph Reports**
- 2 Click **Add** and select **Add Graph Report** from the popup menu.
- 3 Enter a name for the report in the Create Graph Report dialog box and click **SUBMIT**.
- 4 In the Edit Report screen, verify or edit the following information in the Graph Report section:
 - **Name:** The name of the report as defined in the Create Graph Report dialog box.
 - **Title:** Enter a descriptive title to display in the list of reports for the report organizer.
 - **Number of Columns:** Specify the number of columns (1-3) in which graphs will be displayed on the report.
 - **Comments:** Enter comments to display at the top of the printable version of the report. This is a TALES evaluated string that can contain HTML formatting. The variables available to the TALES expression are:
 - `now` (current date and time)
 - `report` (report object)
- 5 Click **Save** to save the new graph report.

- 6 In the Add New Graph section, fill in the appropriate information to add a graph to the report:
 - **Device:** Select the one or more devices. You can narrow the list of devices by entering a search string and clicking **Filter**.
 - **Component:** Optionally, select one or more components from the **Component** list. This list displays the names of all components defined on at least one of the selected device. If you want to see the complete component path in the system to help with identification, select the **Show component path** check box.
- 7 Select one or more graphs from the Graph list. This list displays the names of all the graphs available for the selected devices; or, if you have selected one or more components, the graphs available for the components.
- 8 Click **Add Graph to Report**. The selected graphs are displayed in the Graphs section. You can resequence or delete graphs using the **Action** icon.

The screenshot shows two parts of a software interface for managing graphs in a report.

Add New Graph Dialog:

- Device:** A list box containing device names: 10.87.254.1, Cisco_10.171.100.14, CUSQU001CN3B5, N1kv-Orch, N6001-2, nexus-5k.zenoss.loc, perf2-switch.zenoss.loc, perf3-switch.zenoss.loc. A checkbox labeled "Show component path" is present.
- Component:** A list box containing component names: 32 port Modular Universal Port Supervisor in Fixed Module-1, bts-esx1, bts-esx2, default, default, ESXi_4_8_SAN.
- Graph:** A list box containing graph names: CPU Utilization, Memory Utilization, Power Consumption.

A button labeled "Add Graph to Report" is located at the bottom of this dialog.

Graphs Section:

| Seq | Name | Device | Component | Graph |
|-----|---|---------------------|---|-----------------|
| 0 | nexus-5k.zenoss.loc 32 port Modular Universal Port Supervisor in Fixed Module-1 CPU Utilization | nexus-5k.zenoss.loc | 32 port Modular Universal Port Supervisor in Fixed Module-1 | CPU Utilization |

Graph reports maintain a static list of graphs. This list does not automatically change when graphs are added or deleted from monitoring templates. For example, you have two devices with associated graphs:

- DeviceA: Has a single graph (Graph1)
- Device B: Has two graphs (Graph1 and Graph2)

If you select DeviceA and DeviceB on the Graph Report edit page, the list of graphs will include Graph1 and Graph2. If you select both graphs and add them to the report, you will see three graphs:

- DeviceA - Graph1
- Device B - Graph1
- Device B - Graph2

If, at a later time, you create a second graph (Graph2) on DeviceA's monitoring templates, that new graph will not automatically appear on the graph report. You must edit the report to add it. Similarly, if you later remove a graph from DeviceB's template (or even delete DeviceB from the system), you must manually remove the graph (or device) from the graph report.

Working with graph reports

You can customize the display of the graphs contained in any graph report you've created. You can change the graph name by clicking on the graph name in the Graphs section of the Edit Report screen. You can also edit the text that appears with the graph when viewing the report:

- **Summary:** Displays above the graph in the normal report view. It may contain TALES expressions with these variables:
 - dev - Device
 - comp - Component
 - graph - Graph
- **Comments:** Displays to the left of the graph in the printable view. May contain TALES expressions with these variables:
 - dev - Device
 - comp - Component
 - graph - Graph

On the Graph report, you can control the information displayed by selecting the Range of data to display or by clicking the **Zoom In/Zoom Out** controls. Using this latter method, automatically adjusts the data range to Custom and you can fine-tune your date range.

Multi-Graph reports

Multi-graph reports combine data from different devices and components into a single report. You can create a graph definition and have it drawn once for each of a group of devices and components that you define. Alternatively, you can combine the data for those graphs into a single graph.

The groups of devices and components you assemble are called "collections". Specifying the graph definition to apply to collections is done through graph group objects. Multi-graph reports include their own graph definitions, and thus do not use the graph definitions that are defined in monitoring templates. To create a report that includes graphs defined on templates, use a Graph report instead.

Creating a multi-graph report

- 1 Navigate to **REPORTS > Multi-Graph Reports**
- 2 Click **Add** and select **Add Multi-Graph Report** from the popup menu.
- 3 Enter a name for the report in the Create Multi-Graph Report dialog box and click **SUBMIT**.
- 4 In the report edit page, enter or select values for the following:
 - **Name:** The name of the report as defined in the Create Multi-Graph Report dialog box.
 - **Title:** Enter a descriptive title to display in the list of reports for the report organizer.
 - **Number of Columns:** Specify the number of columns (1-3) in which graphs will be displayed on the report.
- 5 Click **Save**.
- 6 Add one or more of the following to define the source you want to graph:
 - **Collections:** Contain the devices and components you want to graph.
 - **Graph Definitions:** Describe the graphs you want on the report.
 - **Graph Groups:** Specify the collections and graph definition to use.

Figure 104: Multi-Graph Report Edit page

The screenshot shows the 'Multi-Graph Report Edit' interface. At the top, it displays the state at time: 2016/10/20 21:05:39. Below this, there are fields for 'Name' (set to 'test_multi_graph_report'), 'Title', and 'Number of Columns' (set to 1). A 'Save' button is located below these fields. The interface is divided into several sections: 'Collections' (with a dropdown menu), 'Graph Definitions' (with columns for Name, Graph Points, Units, Height, and Width), and 'Graph Groups' (with columns for Seq, Name, Collection, and Graph Definition).

Adding collections

A collection comprises one or more collection items. A collection item can be a list of device classes, systems, groups, locations, or specific devices or components. A single collection may contain as many collection items as desired. A multi-graph report must contain at least one collection. Collections are shown in the Collections area of the report's Edit page.

To create a collection:

- 1 In the **Collections** area of the multi-graph report Edit page, click the **Action** icon and select **Add Collection**. The Add a Collection dialog box appears.
- 2 Enter a name for the collection, then click **OK**. The Multi-Graph Report Collection dialog box appears.

Figure 105: Multi-Graph report collection

The screenshot shows the 'Multi-Graph Report Collection' dialog box. It starts with a 'Name' field containing 'test_collection' and a 'Save' button. Below this is the 'Add To Collection' section, which includes an 'Item Type' dropdown set to 'Device Class' with a list of sub-options like /Discovered, /Network, /Server, etc. There is also a 'Include Suborganizers?' dropdown set to 'True'. An 'Add to Collection' button is located at the bottom of this section. The final section is 'Collection Items', which lists items by Seq, Name, Item Description, and Number of Devices/Components.

- 3 In the Add To Collection area, select collection items to add to the collection:
 - a Select a value for Item Type. If you select either Device Class, System, Group, or Location, then you can select one or more of the organizers to include in the collection. If you select Specific Device/Component, you will be able to choose from a list of all the devices in the system. You can use the Filter field to narrow the selection process. Selecting one or more devices will display a list of component names that apply to the selected devices.
 - b Select a value for **Include Suborganizers?**. If true, the collection will also include all organizers recursively beneath the selected organizer. These collection items are dynamic, when devices are added or removed from the organizers, they will appear or disappear from the report.

- 4 Click **Add to Collection** to create a new collection item for each of the selected organizers or specific device. The collection item appears in the Collection Items area. If desired, you can re-order collection items. Their listed order determines the order in which the graphs are drawn, or the order that data is drawn on a combined graph.

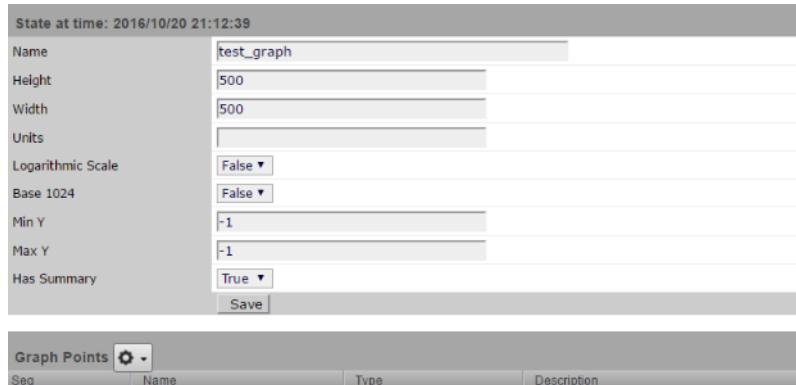
Adding graph definitions

In the context of multi-graph reports, graph definitions are very similar to those in monitoring templates. Settings on the graph definition define basic parameters. Graph points are added to specify which data should be drawn. (For more information on creating graph definitions, see *Performance Graphs* on page 109.)

The most significant differences between graph definitions in the two contexts is how data point graph points and threshold graph points are added. When adding a data point graph point to a graph definition in a performance template, you can select from a list of data points that are defined on that template. In the context of a multi-graph report, there are no graph point definitions listed. You must enter the name of the data point on the data point graph point dialog.

- 1 Click the **Action** icon in the Graph Definitions area of the Graph edit page and select **Add Graph**. The Add a New Graph dialog box appears.
- 2 Enter a name for the graph, then click **OK**. The Edit Graph Definition page appears.

Figure 106: Multi-Graph Report Graph Definition



- 3 Make any changes to the graph definition values displayed, then click **Save**.
- 4 Click the **Action** icon in the Graph Points section to perform the following:
 - Add data points (see next topic)
 - Add thresholds
 - Add a custom graph point
 - Delete a graph point
 - Re-sequence graph points

Adding data points

To add a data point to a graph definition:

- 1 Ensure that you have created a graph definition as part of your multi-graph report.
- 2 On the Graph Definition page, click the **Action** icon in the Graph Points section and select **Add DataPoint**.
- 3 Enter a DataPoint Name. The field will auto-populate based on your entry. Click **OK** to save the name.
- 4 Click the name of the graph point you want to define. The Edit screen appears.

Figure 107: Edit DataPoint

State at time: 2016/11/03 16:40:51

| | |
|-------------------------------------|--|
| Name | usedBlocks |
| Type | DataPoint |
| DataPoint | disk_usedBlocks |
| Consolidation | AVERAGE |
| | <code> \${here/getTotalBlocks},/,100,*</code> |
| RPN | |
| Limit | -1 |
| Line Type | Line ▾ |
| Line Width | 1 |
| Stacked | False ▾ |
| Color (Hex value RRGGBB) | |
| Format | %5.2lf%s |
| Legend | <code> \${here/name} here/id} \${graphPoint/id}</code> |
| <input type="button" value="Save"/> | |
| Available RRD Variables | None |

- 5 Edit the fields based on your datapoint and the way you want data displayed. You can enter a custom RPN expression on this screen if needed.
- 6 Click Save.

Adding graph groups

Graph groups combine a graph definition with a collection to produce graphs for the report. In order for the report to show graphs, at least one graph group must be created.

To create a graph group:

- 1 Click the **Action** icon in the Graph Groups area of the Graph edit page and select **Add Group**. The Add A New Graph Group dialog box appears.
- 2 Enter a name for the graph group, then click **OK**. The Edit Graph Group Definition page appears.

Figure 108: Multi-Graph Report Graph Group

State at time: 2016/10/20 21:14:44

| | |
|-------------------------------------|----------------------------------|
| Name | test_graph_group |
| Collection | test_collection ▾ |
| Graph Definition | test_graph ▾ |
| Method | Separate graph for each device ▾ |
| <input type="button" value="Save"/> | |

- 3 Make any changes to the graph group values displayed:
 - **Name:** Identifies the graph group on the multi-graph report page. It does not appear on the report.
 - **Collection:** Select a collection that has been defined for this report.
 - **Graph Definition:** Select a graph definition that has been defined for this report.
 - **Method:** Choose between having the graph drawn once for each device and component in the collection or combining the data from all devices and components into a single graph. Options are:

- **Separate graph for each device:** The graph definition is used to draw one graph for each device and component in the collection. Graphs will appear in the list in the same order they are specified in the collection.
- **All devices on a single graph:** Draws one graph with the data from all devices and components included.

4 Click **Save** to save the graph group.

Re-sequencing graph group order

Graph groups are drawn in the order listed on the multi-graph report Edit page. To change the order of the graph groups:

- 1 On the Multi-Graph Edit Report page, edit the sequence order numbers (0, 1, 2, etc.) next to the graph groups that you've defined.
- 2 From the **Action** icon, select **Re-sequence items**.
The page refreshes and displays the graph groups in the re-sequenced order.

Note If a graph group results in multiple graphs, the graphs are drawn in the order that the collection items are listed in the corresponding collection. If a collection item specifies a device organizer, the order of devices drawn from that collection item is indeterminate.

Creating a custom device report

To create a custom device report:

- 1 Navigate to **REPORTS > Custom Device Reports**
- 2 Click **Add** and select **Add Custom Device Report** from the popup menu.
- 3 Enter a name for the report in the Create Custom Device Report dialog box and click **SUBMIT**.
- 4 Define the following report parameters:
 - **Name:** Edit the report name if needed.
 - **Title:** Enter the report title. This title is displayed in the report and is distinct from the report name.
 - **Path:** Specify the path in the hierarchy where you want the system to store the report.
 - **Query:** Specify the actual query string for the report. For example, if you want to limit the report to just those devices with a serial number, you can set the query value to:

```
here.hw.serialNumber != ""
```

- **Sort Column:** Specify the column on which you want to sort the report by default.
- **Sort Sense:** Specify the sense that the system uses to sort: `asc` (ascending sort) or `desc` (descending sort)
- **Columns:** Specify the data to be retrieved and displayed in the report. For example you could specify:
 - `getID`: Gets the name of any device.
 - `getManageIp`: Gets the IP addresses of the devices.
 - `getHWSerialNumber`: Gets the serial numbers of the devices.

Note For a complete list of valid options, refer to the information on the Device schema in [TALES expressions](#) on page 205.

Scheduling reports

By default, all reports run on demand, presenting information in the browser interface when you run the report. You can also schedule a report to be run using the `reportmail` command line tool. You can select a report to generate and email its output to a list of recipients. Ensure that an SMTP server is configured for your environment.

To schedule a report using `reportmail`:

- 1 Log in to the Zenoss Core browser interface and click **REPORTS**.
- 2 Take note of the report name you want to schedule and the folder it is in. You will need this information later in this procedure. For example, `MAC Addresses` report in the `Device Reports` folder.
- 3 Log in to the Control Center master host as a user with `serviced` CLI privileges.
- 4 Adapt the following command for your environment:

```
serviced service run zope reportmail run -u "http://localhost:8080/
zport/dmd/Reports/FolderName/ReportName" -U user -p password -
a emailaddress -f emailaddress
```

The URL must use `localhost:8080`. Use `%20` for a space in the URL. A more real-world example:

```
serviced service run zope reportmail run -u "http://localhost:8080/
zport/dmd/Reports/Device%20Reports/MAC%20Addresses" -U admin -p Pa$%
$w@rd -a managers@example.com -f craig@example.com
```

The following table lists all the arguments available for the `reportmail` command:

Table 7: Reportmail command line arguments

| Argument | Description |
|---|--|
| <code>-u URL, --url=URL</code> | Uniform Resource Locator of the report to send. This can also be the URL of any other page in the system. Use <code>localhost:8080</code> as the domain. Use <code>%20</code> for the space character. |
| <code>-U USER, --user=USER</code> | User to log in to the system. This user must have permission to view the supplied URL. |
| <code>-p PASSWD, --passwd=PASSWD</code> | Password to log in to the system. |
| <code>-a ADDRESS, --address=ADDRESS</code> | Email address for report delivery (may be given more than once). Default value comes from the user's profile. |
| <code>-s SUBJECT, --subject=SUBJECT</code> | Subject line for email message. Default value is the title of the page. |
| <code>-f FROMADDRESS, --from=FROMADDRESS</code> | Origination address for the email being sent. |
| <code>-d DIV, --div=DIV</code> | DIV to extract from the HTML at URL. The default value is <code>contentPane</code> , which works for all default reports. |

ZenPacks

ZenPacks extend and modify the system to add new functionality. This can be as simple as adding new device classes or monitoring templates, or as complex as extending the data model and providing new collection daemons.

You can use ZenPacks to add:

- Monitoring templates
- Data sources
- Graphs
- Event classes
- User commands
- Reports
- Model extensions
- Product definitions

Simple ZenPacks can be created completely within the user interface. More complex ZenPacks require development of scripts or daemons, using Python or another programming language.

ZenPacks can be distributed for installation on other Zenoss Core systems.

Displaying the list of installed ZenPacks

You can display installed ZenPacks by using the browser interface or the command-line interface (CLI).

Displaying installed ZenPacks in the browser interface

- 1 In the Zenoss Core browser interface, select the **ADVANCED** tab.

Note The action menu in the left column does not include an option to create a ZenPack. For more information, see [Creating a ZenPack](#) on page 182.

- 2 In the left column, select **ZenPacks**.

The following figure shows an example list of ZenPacks.

The screenshot shows the Zenoss Control Center's main menu at the top with tabs for Settings, Control Center, Monitoring Templates, Jobs, MIBs, Page, and Tips. The 'Settings' tab is selected. In the left sidebar, under the 'ZenPacks' heading, there is a list of categories: Settings, Commands, Users, ZenPacks, Portlets, Versions, Events, User Interface, and Support. The 'ZenPacks' item is highlighted with a blue background. To the right of the sidebar, a table titled 'Loaded ZenPacks' is displayed. The table has a header row with columns: Pack, Package, Author, Version, and Egg. Below the header, there are ten rows, each representing a different ZenPack. Each row contains a link to the ZenPack documentation, the package name (zenoss), the author (Zenoss), the version (e.g., 2.1.4, 1.2.0dev, etc.), and a 'Yes' or 'No' value for the Egg column.

| Pack | Package | Author | Version | Egg |
|---|---------|--------|----------|-----|
| ZenPacks zenoss ApacheMonitor | zenoss | Zenoss | 2.1.4 | Yes |
| ZenPacks zenoss Dashboard | zenoss | Zenoss | 1.2.0dev | Yes |
| ZenPacks zenoss DeviceSearch | zenoss | Zenoss | 1.2.1 | Yes |
| ZenPacks zenoss HttpMonitor | zenoss | Zenoss | 2.1.0 | Yes |
| ZenPacks zenoss LinuxMonitor | zenoss | Zenoss | 1.4.1 | Yes |
| ZenPacks zenoss Microsoft Windows | zenoss | Zenoss | 2.5.7 | Yes |
| ZenPacks zenoss MySqlMonitor | zenoss | Zenoss | 3.0.7 | Yes |
| ZenPacks zenoss NtpMonitor | zenoss | Zenoss | 2.2.2 | Yes |
| ZenPacks zenoss PythonCollector | zenoss | Zenoss | 1.7.3 | Yes |
| ZenPacks zenoss ZenMail | zenoss | Zenoss | 5.0.1 | Yes |

Displaying installed ZenPacks in the CLI

To perform this procedure, you need a user account with `serviced` command-line interface (CLI) privileges on the Control Center master host.

- 1 Log in to the Control Center master host as a user with `serviced` CLI privileges.
- 2 Display the list of installed ZenPacks:

```
serviced service run zope zenpack list
```

ZenPack information resources

Zenoss provides numerous ZenPacks that add and extend system functionality. In the Zenoss Core browser interface, the question mark icon provides a link to the documentation of the included ZenPacks. The [ZenPack catalog](#) provides detailed descriptions of all ZenPacks that are developed by Zenoss.

You can create your own ZenPacks or download and install ZenPacks that are developed by others. For more information, see the following ZenPack resources:

- [ZenPack SDK](#)
- [Zenoss Community](#), which includes a ZenPack development forum
- [Public Zenoss repositories on GitHub](#)

Preparing to install or upgrade a ZenPack

Perform this procedure to minimize the amount of time that Zenoss Core is unavailable during a ZenPack installation or upgrade.

- 1 Log in to your workstation and start a web browser.
- 2 Download the ZenPack to install or upgrade from the [ZenPack catalog](#) site.
- 3 Copy the ZenPack egg file to a local directory on the Control Center master host.
 - a Create a directory for the ZenPack egg file.
The directory must be local (not mounted).

The following command creates a directory in /tmp:

```
mkdir /tmp/zenpack
```

- b** Use a file transfer command or utility to copy the file.
- c** Set full permissions on the directory and files:

```
chmod -R 0777 /tmp/zenpack
```

4 Optional: Install ZenPack dependencies.

A ZenPack might require packages or other software not included in the ZenPack egg file. To ensure that the dependencies are available, perform the following substeps:

- a** Log in to the Control Center master host as a user with Control Center CLI privileges.
- b** Start an interactive shell in a Zope service container.

In the following command, the `-s` flag saves and tags the changes that you make. Replace *MyTag* with a short name that describes the dependencies that you are installing.

```
serviced service shell -i -s MyTag zope bash
```

The serviced daemon starts a Bash shell and logs you in as the root user.

- c** Install required dependencies.

For example, to install the terminus font for X Windows in Ubuntu Linux, enter the following command:

```
apt-get install xfonts-terminus
```

Enter any number of commands to install the required dependencies.

- d** Return to the Control Center master host shell session:

```
exit
```

- e** Create a snapshot and commit your changes:

```
serviced snapshot commit MyTag
```

- f** Restart all Zenoss Core application services:

```
serviced service restart Zenoss.core/Zenoss
```

Installing or upgrading a ZenPack

Before you begin, review the following requirements and considerations:

- Complete the steps in [Preparing to install or upgrade a ZenPack](#) on page 179.
- Do not use this procedure to install or upgrade the Zenoss Service Impact ZenPacks, *ZenPacks.zenoss.ImpactServer* and *ZenPacks.zenoss.Impact*. For more information, refer to the *Zenoss Core Installation Guide*.

- 1** Log in to the Control Center master host as a user with Control Center CLI privileges.
- 2** Create a snapshot:

```
serviced service snapshot Zenoss.core
```

On completion, the `serviced` command returns the ID of the new snapshot. If the installation of the ZenPack ends up failing, you can restore the snapshot you took in this step. For detailed instructions on restoring a snapshot, refer to the *Control Center Reference Guide*.

- 3 Change directory to the directory in which the ZenPack egg file is located.
For example:

```
cd /tmp/zenpack
```

- 4 Install the ZenPack:

```
serviced service run zope zenpack-manager install ZenPack-File.egg
```

Daemons that a ZenPack provides are packaged in Docker containers and installed as child services of the current instance of Zenoss Core.

- 5 Restart all Zenoss services:

```
serviced service restart Zenoss.core/Zenoss
```

Removing a ZenPack

Removing a ZenPack can have unexpected consequences, and often, the safest choice is not to remove a ZenPack. Before you begin, review the following requirements and considerations:

- Removing a ZenPack removes all objects provided by the ZenPack and all objects that depend on code provided by the ZenPack.
- Removing a newer version of a ZenPack to install an older version fails if the newer version includes migration code.
- Removing a ZenPack that installs a device class removes the device class, any contained device classes, and all devices in that class.
- Some ZenPacks provide services upon which other ZenPacks rely. Make sure the service you remove is not needed by another ZenPack.
- Do not use this procedure to remove the Zenoss Service Impact ZenPacks, *ZenPacks.zenoss.ImpactServer* and *ZenPacks.zenoss.Impact*. For more information, refer to the *Zenoss Core Installation Guide*.
- Review the documentation of the ZenPack that you want to remove for information about classes and daemons (services) associated with it.
- Delete data sources provided by the ZenPack that you want to remove.

- 1 Log in to the Control Center master host as a user with Control Center CLI privileges.
- 2 Create a snapshot:

```
serviced service snapshot Zenoss.core
```

On completion, the `serviced` command returns the ID of the new snapshot. If the installation of the ZenPack ends up failing, you can restore the snapshot you took in this step. For detailed instructions on restoring a snapshot, refer to the *Control Center Reference Guide*.

- 3 Obtain the exact name of the ZenPack to remove:

```
serviced service run zope zenpack list
```

The first item of each line of output is the full name of an installed ZenPack.

- 4 Stop services that are associated with the ZenPack.

Daemons that a ZenPack provides are packaged in Docker containers and installed as child services of the current instance of Zenoss Core. For example, the `zenwebtx` service is provided by the `ZenPacks.zenoss.ZenWebTx` ZenPack.

- a Log into the Zenoss Core browser interface as the Zenoss Core user.
- b From the navigation menu, select **ADVANCED > Control Center**.
The Control Center **All Services** page appears.
- c Select the daemon for the service and click **Stop**.
For example, select the `zenwebtx` daemon.

5 Remove the ZenPack.

In the Control Center CLI, replace `ZenPack-Name` with the full name of the ZenPack to remove.

```
serviced service run zope zenpack-manager uninstall ZenPack-Name
```

The ZenPack and any daemons that it provides are removed.

Creating a ZenPack

To perform this procedure, you need a user account with `serviced` CLI privileges on the Control Center master host.

This procedure demonstrates how to create a ZenPack for customized monitoring templates, customized event classes and mappings, device MIBs, and similar items which require no customized Python code. For more advanced ZenPack development, refer to the [ZenPack SDK](#) site.

- 1 Log in to the Control Center master host as a user with `serviced` CLI privileges.
- 2 Create a ZenPack.

Replace `ZenPacks.myOrg.myPackName` with the name of the ZenPack to create.

```
serviced service run zope zenpack-manager create \
ZenPacks.myOrg.myPackName
```

- 3 Restart the Zope service.

```
serviced service restart zope
```

General administration and settings

Use the information and procedures in this section for troubleshooting and performance improvement purposes.

Events settings

You can adjust events settings for

- Events database connection
- Event maintenance

Changing events database connection information

To edit events database connection settings, make changes in the `zeneventserver.conf` file. You can edit the file directly, or run a configuration script.

Configurable database connection settings are:

- **JDBC Hostname (zep.jdbc.hostname)** - Specify the IP address of the host.
- **JDBC Port (zep.jdbc.port)** - Specify the port to use when accessing the events database.
- **JDBC Database Name (zep.jdbc.dbname)** - Specify the database name.
- **JDBC Username (zep.jdbc.username)** - Specify the user name for the database.
- **JDBC Password (zep.jdbc.password)**- Specify the password for the database.

To edit these values, run the `zeneventserver` configuration script, as follows:

```
zeneventserver-config -u zep.jdbc.Name=Value
```

Where *Name* is the partial setting name and *Value* is the value you want to specify for the setting.

Changing events maintenance settings

To edit maintenance settings, make changes to one or more fields on the Event Configuration page (**ADVANCED > Settings > Events**):

- **Don't Age This Severity and Above** - Options are Age All Events, Critical, Error, Warning, Info, Debug, and Clear. By default, this value is set to Error, meaning that all events with a status of Error or Critical are not aged.

- **Event Aging Threshold (minutes)** - Set the time value, in minutes, that an event must reach before it is aged. By default, this is 240 minutes.
- **Event Aging Interval (milliseconds)** - The interval when events are scanned to perform autoaging. By default, this is 60000 milliseconds (60 sec).
- **Event Aging Limit** - The maximum number of events to age in each interval. The limit should be kept relatively low to prevent large database transactions. By default, this is 1000 events.
- **Event Archive Threshold (minutes)** - Specify the number of minutes since a closed event was last seen before it is moved to the event archive. The minimum value is 1; the maximum value is 43200.
- **Event Archive Interval (milliseconds)** - The interval when events are scanned for moving to the archive. By default, this is 60000 milliseconds (60 sec).
- **Event Archive Limit** - The maximum number of events to archive in each interval. The limit should be kept relatively low to prevent large database transactions. By default, this is 1000 events.
- **Delete Archived Events Older Than (days)** - The number of days that events in the event archive are saved. By default, they are kept in the archive for 90 days. The minimum value is 1 and the maximum value is determined by the range of event archive partitions. With the default configuration, the maximum value is 1000 days.
- **Default Syslog Priority** - Specify the default severity level assigned to an event coming from zensyslog if no priority can be determined from the event.
- **Default Availability Report (days)** - Enter the number of days to include in the automatically generated Availability Report. This report shows a graphical summary of availability and status.
- **Max Event Size in Bytes** - The maximum size of an event that will be processed in bytes. Events that are too large will be logged and dropped. Events that will become too big will have their details overwritten with new details. By default, this is 32768 bytes.
- **Summary Index Interval (milliseconds)** - The default indexing interval of the event summary in milliseconds. By default, this is 1000 milliseconds (1 sec).
- **Archive Index Interval (milliseconds)** - The default indexing interval of the event archive in milliseconds. By default, this is 30000 milliseconds (30 sec).
- **Index Limit** - The number of events to index in each index interval. By default, this is 1000 events.
- **Event Time Purge Interval (days)** - The number of days that event occurrence time are kept. By default, they are kept for 7 days. The minimum value is 1 and the maximum value is determined by the range of event time partitions. With the default configuration, the maximum value is 7 days.
- **Enable Event Flapping Detection** - Select this check box if you wish to enable event flapping detection. If an event is created and then cleared *flapping_threshold* times in *event_flapping_interval* time then an event of event flapping event class is created.
- **Event Flapping Event Class** - The event class under which generated flapping events belong.
- **Clear Event Heartbeats** - Click Clear to clear the event heartbeats.

Rebuilding the events index

If you encounter inconsistent search results, you can rebuild the events index.

- 1 Log in to an account on the Control Center master host that has permission to use the Control Center command-line interface.
- 2 Stop zeneventserver:

```
serviced service stop zeneventserver
```

- 3 Delete the index data:

```
export SERVICE_ID=$(serviced service status Zenoss.core | sed -n '2p'
| awk '{print $2}')
export SVCROOT=/opt/serviced/var/volumes/$SERVICE_ID
```

```
rm -rf $SVCROOT/zeneventserver/index
```

- 4 Start zeneventserver:

```
serviced service start zeneventserver
```

Depending on the number of events in the database, it may take a significant amount of time for indexing to complete. Until every event is indexed, the number of events shown in the event console may be inconsistent.

Working with the job manager

The job manager runs background tasks, such as discovering a network or adding a device. When you ask the system to perform one of these tasks, it adds a job to the queue. Jobs are run by the `zenjobs` daemon.

Not all actions are performed in the job manager. Some jobs are run automatically in the foreground. Others, such as moving devices, depend on user interface configuration settings.

When running jobs in the foreground, do not navigate away from the current page until the action completes.

Viewing the job manager

- From the Navigation menu, select **ADVANCED**.
- On the **Settings** page, select **Jobs**.

Figure 109: Job manager

The screenshot shows the Zenoss web interface for managing jobs. At the top, there are navigation links: Settings, Control Center, Monitoring Templates, Jobs (which is the active tab), and Page Tips. Below the navigation is a toolbar with buttons for Delete and Abort, and a Refresh dropdown.

The main area displays a table titled "Background Jobs" with the following columns: Status, Description, Scheduled, Started, Finished, and Created By. The table lists 13 rows of job entries. The first row is highlighted with a blue selection bar. The last row is also highlighted with a blue selection bar.

| Status | Description | Scheduled | Started | Finished | Created By |
|---------|---|------------|------------|------------|------------|
| Success | Create 10.171.100.14 under /Network/Cisco/6500 | 2 days ago | 2 days ago | 2 days ago | admin |
| Failure | Discover and model device 10.171.100.14 as /Netw... | 2 days ago | 2 days ago | 2 days ago | admin |
| Success | Create 10.171.100.92 under /Network/Cisco/Nexus/70... | 2 days ago | 2 days ago | 2 days ago | admin |
| Failure | Discover and model device 10.171.100.92 as /Netw... | 2 days ago | 2 days ago | 2 days ago | admin |
| Success | Create 10.171.100.88 under /Network/Cisco/Nexus/10... | 2 days ago | 2 days ago | 2 days ago | admin |
| Failure | Discover and model device 10.171.100.88 as /Netw... | 2 days ago | 2 days ago | 2 days ago | admin |
| Success | Create 10.171.54.9 under /Network/Cisco | 2 days ago | 2 days ago | 2 days ago | admin |
| Success | Discover and model device 10.171.54.9 as /Network/... | 2 days ago | 2 days ago | 2 days ago | admin |
| Success | Create 10.171.100.107 under /Network/Cisco/Nexus/6... | 2 days ago | 2 days ago | 2 days ago | admin |
| Failure | Discover and model device 10.171.100.107 as /Netw... | 2 days ago | 2 days ago | 2 days ago | admin |
| Success | Create 10.171.53.1 under /Network/Cisco | 2 days ago | 2 days ago | 2 days ago | admin |
| Success | Discover and model device 10.171.53.1 as /Network/... | 2 days ago | 2 days ago | 2 days ago | admin |

At the bottom right of the table, it says "DISPLAYING 1 • 13 of 33 ROWS".

Below the table is a section titled "Job Log" with a scrollable log file content. The log file path is listed as </opt/zenoss/log/jobs/c4583592-2506-4466-b909-854ebfa20258.log>.

```

2015-08-04 17:59:20,643 INFO zen.Job: Job c4583592-2506-4466-b909-854ebfa20258 (Products.ZenModel.ZDeviceLoader.CreateDeviceJob)
received
2015-08-04 17:59:20,657 INFO zen.Job: Starting job c4583592-2506-4466-b909-854ebfa20258
(Products.ZenModel.ZDeviceLoader.CreateDeviceJob)
2015-08-04 17:59:21,051 INFO zen.Job: Job c4583592-2506-4466-b909-854ebfa20258 finished with result
/zport/dmd/Devices/Network/Cisco/Nexus/7000/devices/10.171.100.92

```

The jobs list displays the following information about the jobs:

- **Status** - Shows the current job status. Status options are Pending (waiting for zenjobs to begin running), Running, Succeeded, and Failed.
- **Description** - Provides a description of the job.
- **Scheduled** - Shows when the job was scheduled to begin.
- **Started / Finished** - Provide information about the time period in which the job ran.
- **Created By** - User that created the job.

The lower section of the page displays the job log for the job that you select in the list. You can also view the information in the log file.

Stopping and deleting jobs

To stop a job, select it in the list, and then click **Abort**. The zenjobs daemon will not run the job.

To remove a job from the system, select it and then click **Delete**.

Configuring jobs

When you move devices, you can choose whether the action is performed immediately or as a job. By default, if you select five or more devices, the move action is performed as a job. To adjust this setting:

- 1 Select **Advanced > Settings**.
- 2 Select **User Interface** in the left panel.
- 3 Enter a value for **Device Move Job Threshold**, and then click **Save**.

Running the zenjobs daemon

You can stop and start the zenjobs daemon from the command line, and from **Advanced > Settings**(Daemons selection).

Using the Appliance Administration menu

A

This appendix describes the curses-based Appliance Administration menu, a text user interface (TUI).

Configure Network and DNS

The **Configure Network and DNS** option invokes *nmtui*, the *NetworkManager* text user interface (TUI) tool. The *nmtui* utility provides submenus for editing and activating network connections, and for changing the hostname.

Note Zenoss recommends using only the **Configure Network and DNS** option to change connection properties or the hostname, and always rebooting after making changes.

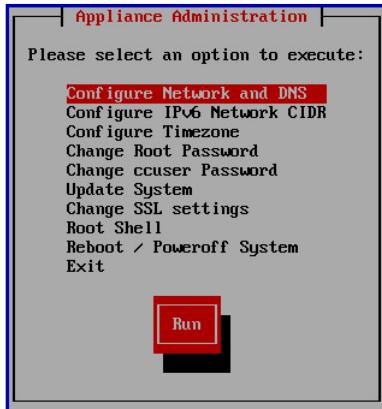
Editing a connection to configure static IPv4 addressing

The default configuration for network connections is DHCP. To configure static IPv4 addressing, perform this procedure.

To navigate in the text user interface (TUI):

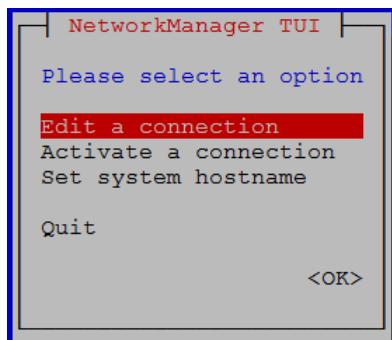
- To move forward or backward through options, press the arrow keys.
- To display a menu or choose an option, press **Enter**.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PuTTY*.
- 2 Log in as the **root** user.



- 3 Select the **NetworkManager TUI** menu as follows:

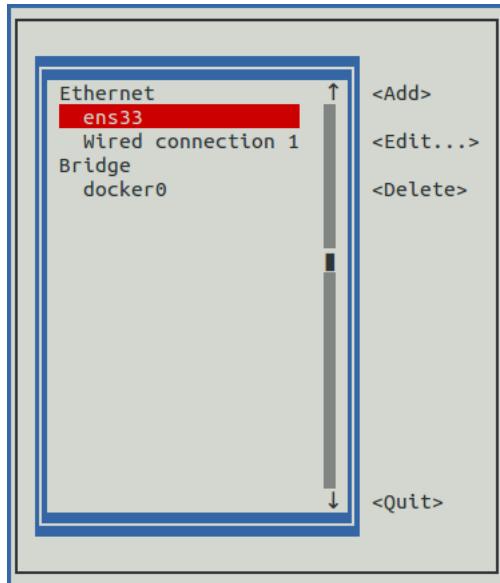
- a In the **Appliance Administration** menu, select **Configure Network and DNS**, and then press **Enter**.



- 4 On the **NetworkManager TUI** menu, select **Edit a connection**, and then press **Enter**.

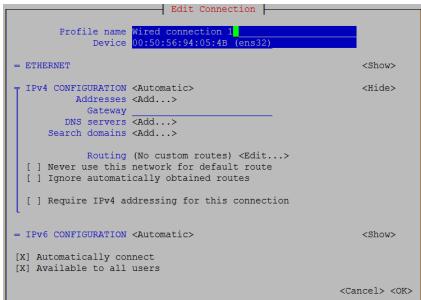
The TUI displays the connections that are available on the host.

Figure 110: Example: Available connections

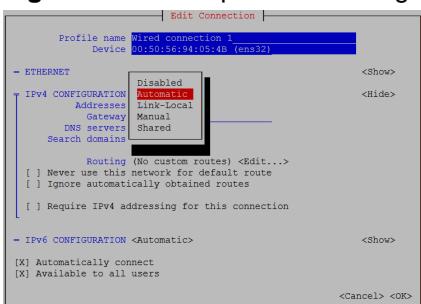


Note Do not use this procedure to modify the docker0 connection.

- 5 Select the virtual connection, and then press **Enter**.

Figure 111: Example: Edit Connection screen

- 6 Optional: If the **IPv4 CONFIGURATION** area is not visible, select its display option (<Show>), and then press **Enter**.
- 7 In the **IPv4 CONFIGURATION** area, select <Automatic>, and then press **Enter**.

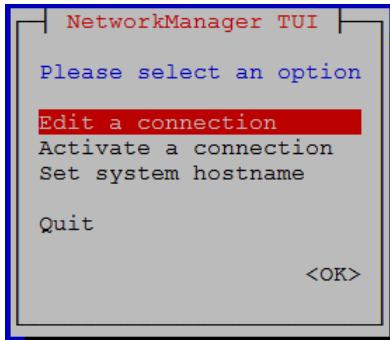
Figure 112: Example: IPv4 Configuration options

- 8 Configure static IPv4 networking as follows:
 - a Select **Manual**, and then press **Enter**.
 - b Beside **Addresses**, select <Add>, and then press **Enter**.
 - c In the **Addresses** field, enter an IPv4 address for the virtual machine, and then press **Enter**.
 - d Repeat the preceding two steps for the **Gateway** and **DNS servers** fields.
- 9 Tab to the bottom of the **Edit Connection** screen to select <OK>, and then press **Enter**.
- 10 Return to the **Appliance Administration** menu: On the **NetworkManager TUI** screen, select <Quit>, and then press **Enter**.
- 11 Reboot the operating system as follows:
 - a In the **Appliance Administration** menu, select **Reboot / Poweroff System**.
 - b Select **Reboot**.
 - c Select **OK**, and then press **Enter**.

Edit a connection (Docker virtual bridge)

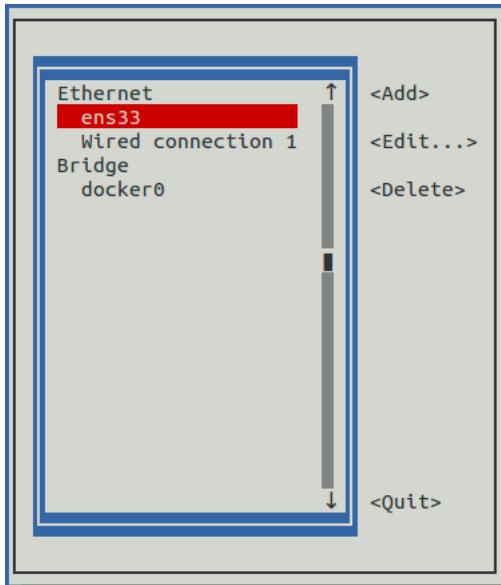
The default IP address space of the Docker virtual bridge is 172.17.0.1/16. To configure a different address space, perform this procedure.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PutTY*.
- 2 Log in as the **root** user.
- 3 Select the **NetworkManager TUI** menu as follows:
 - a In the **Appliance Administration** menu, select **Configure Network and DNS**, and then press **Enter**.

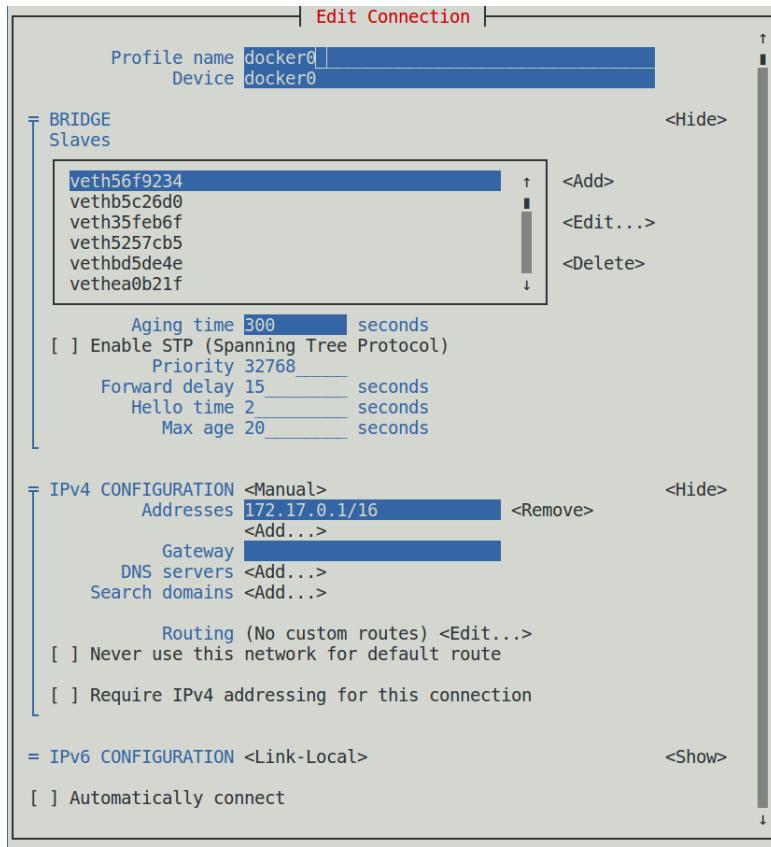


- 4 On the **NetworkManager TUI** menu, select **Edit a connection**, and then press **Enter**.
The TUI displays the connections that are available on this host.

Figure 113: Example: Available connections



- 5 Use the down-arrow key to select **docker0**, and then press **Enter**.

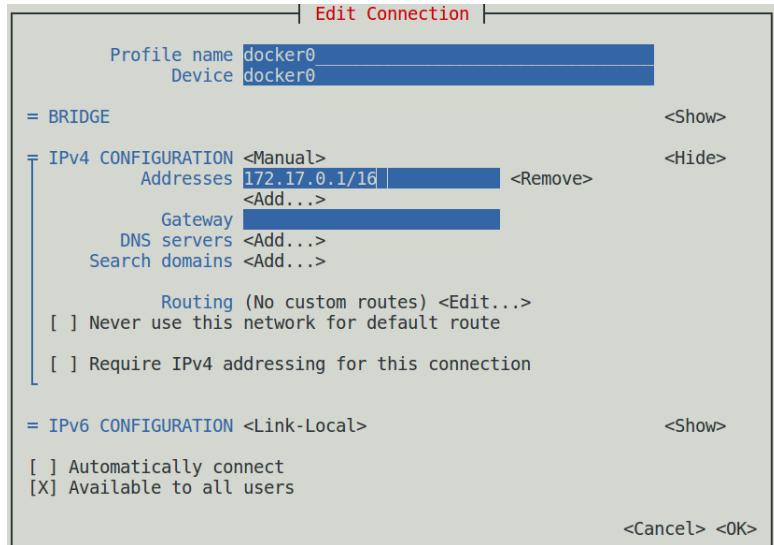
Figure 114: Example: Edit Connection screen

Use the **Tab** key and the arrow keys to navigate among options in the **Edit Connection** screen, and use **Enter** to toggle an option or to display a menu of options.

- If the **BRIDGE** area is visible, select its display option (**<Hide>**), and then press **Enter**.

Note Do not edit any of the entries in the BRIDGE area.

- If the **IPv4 CONFIGURATION** area is not visible, select its display option (**<Show>**), and then press **Enter**.
- In the **IPv4 CONFIGURATION** area, navigate to **Adresses**, and then enter a new IPv4 address in CIDR notation.

Figure 115: Example: IPv4 Configuration options

- 9 Use **Tab** or the Down Arrow key to select the <OK> option at the bottom of the **Edit Connection** screen, and then press **Enter**.
- 10 In the available connections screen, use **Tab** to select the <Quit> option, and then press **Enter**.
- 11 Reboot the operating system as follows:
 - a In the **Appliance Administration** menu, select **Reboot / Poweroff System**.
 - b Press **Tab** to select **OK**, and then press **Enter**.

CIDR prefix lengths for common subnet masks

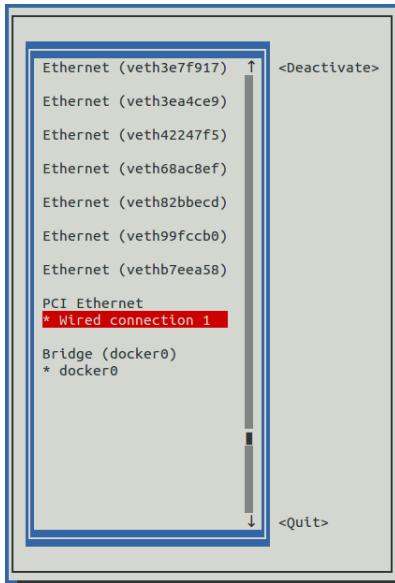
| Subnet mask | CIDR prefix length | Subnet mask | CIDR prefix length |
|---------------|--------------------|-----------------|--------------------|
| 255.255.0.0 | /16 | 255.255.254.0 | /23 |
| 255.255.128.0 | /17 | 255.255.255.0 | /24 |
| 255.255.192.0 | /18 | 255.255.255.128 | /25 |
| 255.255.224.0 | /19 | 255.255.255.192 | /26 |
| 255.255.240.0 | /20 | 255.255.255.224 | /27 |
| 255.255.248.0 | /21 | 255.255.255.240 | /28 |
| 255.255.252.0 | /22 | 255.255.255.248 | /29 |

Activate a connection

The **Activate a connection** submenu provides options for activating and deactivating network connections.

Note Do not deactivate the `docker0` connection.

On selection, the **Activate a connection** submenu displays the available connections. The asterisk character (*) at the beginning of a connection name indicates that the connection is active.

Figure 116: Example: Available connections

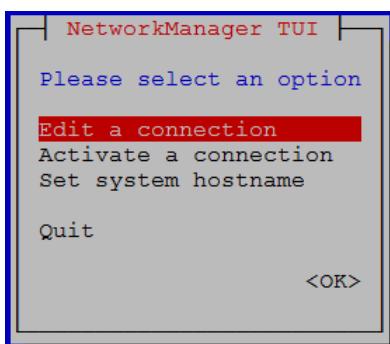
Use the arrow keys to select a connection, and then use **Tab** to navigate the options at the right side of the list. Use **Enter** to choose an option.

Note Always reboot after activating or deactivating a connection.

Setting the system hostname

The default hostname is `core-master` for the Zenoss Core master host and is `core-delegate` for Zenoss Core delegate hosts. To change the default hostname, perform this procedure.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PuTTY*.
- 2 Select the **NetworkManager TUI** menu as follows:
 - a In the **Appliance Administration** menu, select **Configure Network and DNS**, and then press **Enter**.



- 3 Display the hostname entry field.
 - a In the **NetworkManager TUI** menu, select **Set system hostname**.
 - b Select **OK**, and then press **Enter**.
- 4 In the **Hostname** field, enter the hostname or a fully qualified domain name.
- 5 Press **Tab** twice to select **OK**, and then press **Enter**.

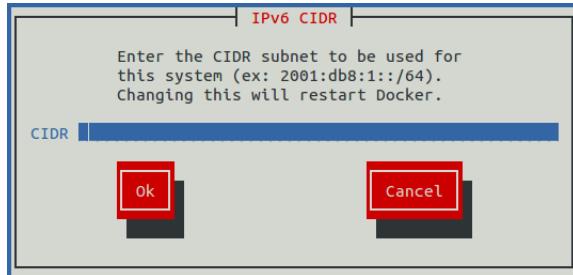


- 6 In the confirmation dialog box, press **Enter**.
- 7 Return to the **Appliance Administration** menu: On the **NetworkManager TUI** screen, select <Quit>, and then press **Enter**.
- 8 Reboot the operating system as follows:
 - a In the **Appliance Administration** menu, select **Reboot / Poweroff System**.
 - b Select **Reboot**.
 - c Select **OK**, and then press **Enter**.

Configure IPv6 Network CIDR

The version of Docker included in the Zenoss Core virtual appliance needs to know at startup the address prefix of the IPv6 network it will use. To enable monitoring of devices that use IPv6, perform this procedure on the Control Center master host, and all delegate hosts.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PuTTY*.
- 2 Log in as the **root** user.
- 3 In the **Appliance Administration** menu, select the **Configure IPv6 Network CIDR** option.

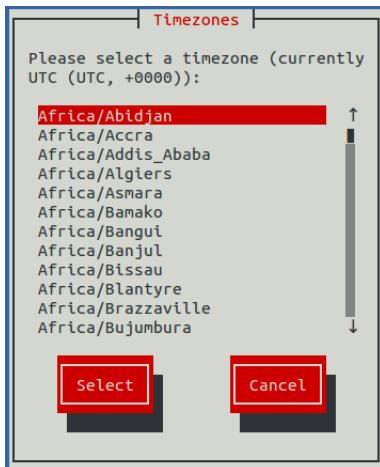


- 4 In the **IPv6 CIDR** screen, enter the address prefix of your IPv6 network in the **CIDR** field.
- 5 User **Tab** to select the **Ok** button, and then press **Enter**.
The Docker daemon restarts, and the **Appliance Administration** disappears briefly before returning. This is normal.

Configure Timezone

The default timezone of the Zenoss Core virtual appliance is UTC. This procedure changes the timezone setting of a single host. All hosts in a multi-host deployment must use the same timezone.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PuTTY*.
- 2 Log in as the **root** user.
- 3 In the **Appliance Administration** menu, select the **Configure Timezone** option.



- 4 Use the **Down Arrow** key to select the desired timezone.
- 5 Press **Tab** to highlight **Select**, and then press **Enter**.

Note Always reboot after changing the timezone.

Change Root Password

This option invokes the `passwd` command to change the password of the `root` user account.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PuTTY*.
- 2 Log in as the `root` user.
- 3 In the **Appliance Administration** menu, select the **Change Root Password** option.
The **Appliance Administration** menu disappears, and the system prompts for a new password:

```
Changing password for user root.
New password:
```

- 4 **Note** Passwords must include a minimum of eight characters, with at least one character from three of the following character classes: uppercase letter, lowercase letter, digit, and special.
-
- Enter a new password, and then press **Enter**.
- 5 Enter the password again, and then press **Enter**.
The **Appliance Administration** menu reappears.

Change ccuser Password

This option invokes the `passwd` command to change the password of the `ccuser` user account.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PuTTY*.
- 2 Log in as the `root` user.
- 3 In the **Appliance Administration** menu, select the **Change Root Password** option.
The **Appliance Administration** menu disappears, and the system prompts for a new password:

```
Changing password for user ccuser.
New password:
```

-
- 4 **Note** Passwords must include a minimum of eight characters, with at least one character from three of the following character classes: uppercase letter, lowercase letter, digit, and special.

Enter a new password, and then press **Enter**.

- 5 Enter the password again, and then press **Enter**.
The **Appliance Administration** menu reappears.

Update System

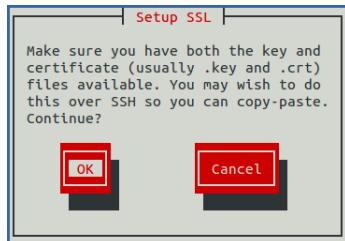
This option updates the Control Center and Zenoss Core software on a host. For more information, refer to the *Zenoss Core Upgrade Guide*.

Change SSL settings

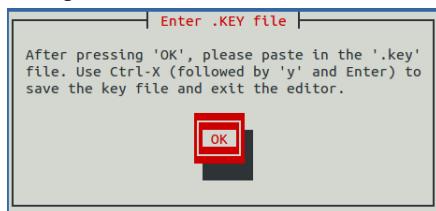
To perform this procedure, you need to be able to display the contents of the SSL certificate and key files that you want to install on the Control Center master host, and you need a copy of the root certificate file (`rootCA.pem`). In addition, Zenoss recommends logging in to the master host through SSH, rather than the hypervisor console, so that you can copy and paste content.

This option allows you to provide new content for SSL certificate and key files.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as *PUTTY*.
- 2 Log in as the **root** user.
- 3 Use the **Down Arrow** key to select **Change SSL settings**, and then press **Enter**.



- 4 When you are ready to add the contents of your SSL certificate and key files to the Control Center master host, press **Enter**.



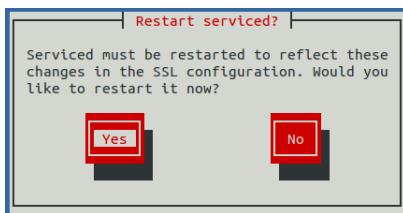
- 5 Press **Enter**.
The **Appliance Administration** menu is replaced with the `nano` text editor.
- 6 Enter the contents of your SSL key file, and then save the file and exit the editor.
 - a Press **Ctrl-O**.
 - b Press **Ctrl-X**.
 - c Press **y**, and then press **Enter**.



- 7 Press **Enter**.

The **Appliance Administration** menu is replaced with the `nano` text editor.

- 8 Enter the contents of your SSL certificate file, and then save the file and exit the editor.
 - a Press **Ctrl-O**.
 - b Press **Ctrl-X**.
 - c Press **y**, and then press **Enter**.



- 9 Restart the Control Center daemon (`serviced`) now or later.

Restarting `serviced` pauses Zenoss Core services briefly.

- To restart `serviced` now, press **Enter**.
- To restart `serviced` later, press **Tab** to select **No**, and then press **Enter**.

- 10 Install the root certificate into browser clients.

The procedures for installing a root certificate into a browser client varies by browser and client operating system. For more information, refer to your browser documentation or articles such as [this one](#).

Root Shell

This option starts a command-line session as the `root` user.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as [PuTTY](#).
 - 2 Log in as the `root` user.
 - 3 Use the **Down Arrow** key to select **Root Shell**, and then press **Enter**.
- The menu is replaced by a command prompt similar to the following example:

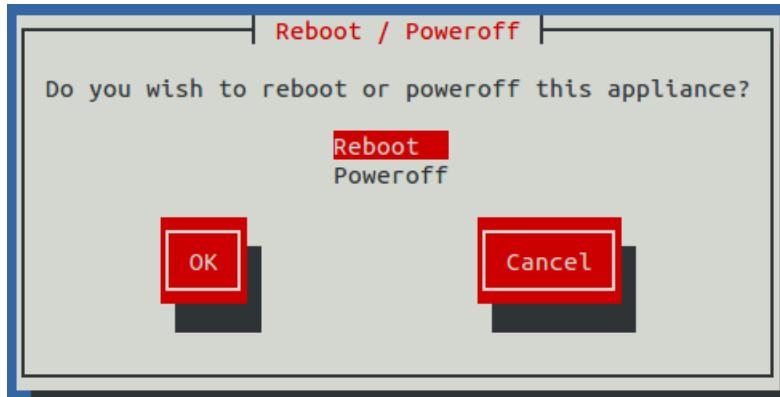
```
[root@core-master ~] #
```

To return to the **Appliance Administration** menu, enter the `exit` command.

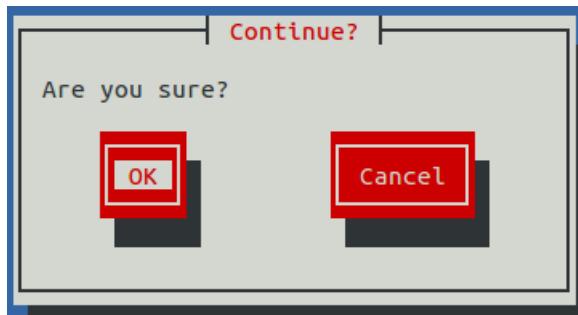
Reboot / Poweroff System

This option reboots or shuts down and turns off a Control Center host.

- 1 Gain access to the Control Center host, through the console interface of your hypervisor, or through a remote shell utility such as [PuTTY](#).
- 2 Log in as the `root` user.
- 3 In the **Appliance Administration** menu, select the **Reboot / Poweroff System** option.



- 4 Use the **Down Arrow** key to select **Reboot** or **Poweroff System**.
- 5 Press **Tab** to highlight **OK**, and then press **Enter**.



- 6 Use **Tab** to select **OK** or **Cancel**, and then press **Enter**.
The system reboots or shuts down and powers off.

B

SNMP device preparation

This section provides information about SNMP support and lists Net-SNMP configuration settings that are required by the system.

Net-SNMP

By default, Net-SNMP does not publish the full SNMP tree. Check to see if that is currently the case on a device and configure it correctly.

- 1 Confirm snmpd is running:

```
> snmpwalk -v 2c -cp public <your device name> system
```

- 2 Retrieve the IP table for the device with snmpwalk:

```
> snmpwalk -v 2c -cp public <your device name> ip
```

Typical SNMP View:

```
view systemview included .1 view systemview included .1.3.6.1.2.1.25.1
access notConfigGroup "" any noauth exact systemview none none
```

SNMP v3 support

Zenoss Core provides support for SNMP v3 data collection.

The following configuration properties control the authentication and privacy of these requests:

- **zSnmpAuthType**- Use "MD5" or "SHA" signatures to authenticate SNMP requests. If only zSnmpAuthType and zSnmpAuthPassword are set, then the message is sent with authentication but no privacy.
- **zSnmpAuthPassword**- Shared private key used for authentication. Must be at least 8 characters long.
- **zSnmpPrivType**- "DES" or "AES" cryptographic algorithms. If zSnmpPrivType and zSnmpPrivPassword are set, then the message is sent with privacy and authentication. You cannot set a PrivType and PrivPassword without also setting an AuthType and AuthPassword. If neither Priv nor Auth values are set, then the message is sent with no authentication or privacy.
- **zSnmpPrivKey**- Shared private key used for encrypting SNMP requests. Must be at least 8 characters long.
- **zSnmpSecurityName**- Security Name (user) to use when making SNMPv3 requests.

If monitoring SNMPv3 devices, make sure that msgAuthoritativeEngineID (also known as snmpEngineID or Engine ID) is not shared by two devices. It must be unique for each device.

Advanced Encryption Standard

SNMPv3 encryption using the Advanced Encryption Standard (AES) algorithm is supported only if the host platform net-snmp library supports it.

You can determine whether your platform supports AES by using the following test:

```
$ snmpwalk -x AES 2>&1 | head -1
```

If the response is:

```
"Invalid privacy protocol specified after -x flag: AES"
```

then your platform does not support AES encryption for SNMPv3.

If the response is:

```
"No hostname specified."
```

Then your platform supports AES.

Community information

Add these lines to your `snmp.conf` file.

This line will map the community name "public" into a "security name":

```
# sec.name source community  
  
com2sec notConfigUser default public
```

This line will map the security name into a group name:

```
# groupName securityModel securityName  
  
group notConfigGroup v2c notConfigUser
```

This line will create a view for you to let the group have rights to:

```
# Make at least snmpwalk -v 1 localhost -c public system fast again.  
  
# name incl/excl subtree mask(optional)  
  
view systemview included .1
```

This line will grant the group read-only access to the systemview view.

```
# group context sec.model sec.level prefix read write notif access
notConfigGroup "" any noauth exact systemview none none
```

System contact information

It is also possible to set the `sysContact` and `sysLocation` system variables through the `snmpd.conf` file:

```
syslocation Unknown (edit /etc/snmp/snmpd.conf)
```

```
syscontact Root <root@localhost> (configure /etc/snmp/snmp.local.conf)
```

```
# Added for support of bcm5820 cards. pass .1 /usr/bin/ucd5820stat
```

Extra information

For more information, see the `snmpd.conf` manual page, and the output of the `snmpd -H` command.

```
trapcommunity public
```

```
trapsink default
```



Syslog device preparation

Forwarding syslog messages from UNIX/Linux devices

Zenoss Core has its own syslog server (`zensyslog`). Managed devices should point their syslog daemons to the system.

To do this, edit the `/etc/rsyslog.conf` file and add an entry, where 1.2.3.4 is the zensyslog IP:

- 1 Log in to the target device as a super user.
- 2 Open the `/etc/rsyslog.conf` file with a text editor (such as `vi`).
- 3 Enter `* .debug`, and then press the Tab key.
- 4 Enter the host name or IP address of the server. For example:

```
* .debug @192.168.X.X
```

- 5 Save the file and exit the file editor program.
- 6 Restart the Syslog service using the command below:

```
/etc/init.d/syslog restart
```

Forwarding syslog messages from a Cisco IOS router

Here are some links to Cisco commands to turn on syslog. Typically, it is easier to use syslog than SNMP traps from network devices. The most basic IOS command to send syslog messages is:

```
logging 1.2.3.4
```

Other Cisco syslog configurations

Following are additional configurations for other Cisco devices. To set up these configurations:

- 1 Log in to the target router.
- 2 Type the command `enable` at the prompt.
- 3 Once you are prompted for a password, enter the correct password.
- 4 Type the command `config` at the prompt.
- 5 Type the command `terminal` at the configuration prompt.

- 6** At the prompt, Set the Syslog forwarding mechanism. See example below:

```
logging <IP address of the server>
```

- 7** Exit out all the prompts to the main router prompt.

Catalyst

```
set logging server enable set logging server 192.168.1.100 set logging
level all 5 set logging server severity 6
```

Local Director

```
syslog output 20.5 no syslog console syslog host 192.168.1.100
```

PIX Firewalls

```
logging on logging standby logging timestamp logging trap notifications
logging facility 19 logging host inside 192.168.1.100
```

Forwarding syslog messages from a Cisco CatOS switch

To forward a syslog message from a Cisco CatOS switch:

- 1** Log in to the target switch.
- 2** Type the command enable at the prompt.
- 3** Enter the password when prompted.
- 4** Set the Syslog forwarding mechanism; for example:

```
set logging server <IP address of the server>
```

- 5** You can set the types of logging information that you want the switch to provide with the commands below as examples:

```
set logging level mgmt 7 default set logging level sys 7 default set
logging level filesys 7 default
```

Forwarding syslog messages using syslog-ng

Here is an example for FreeBSD and Linux platforms.

- 1** Log in to the target device as a super user.
- 2** Open /etc/syslog-ng/syslog-ng.conf file with a text editor (e.g vi).
- 3** Add source information to file. See the following examples:

FreeBSD:

```
source src { unix-dgram("/var/run/log"); internal () ;};
```

Linux: (will gather both system and kernel logs)

```
source src { internal(); unix-stream("/dev/log" keep-alive(yes) max-  
connections(100)); pipe("/proc/kmsg"); udp(); };
```

- 4 Add destination information (in this case, the server). For example:

```
log { source(src); destination(zenoss); };
```

TALES expressions

D

Use TALES syntax to retrieve values and call methods on Zenoss Core objects. Several areas accept TALES syntax; these include:

- Command templates
- User commands
- Notifications
- zLinks

Commands (those associated with devices and those associated with events) can use TALES expressions to incorporate data from the related devices or events. TALES is a syntax for specifying expressions that let you access the attributes of certain objects, such as a device or an event.

For additional documentation on TALES syntax, see the TALES section of the [Zope Page Templates Reference](#).

Depending on context, you may have access to a device, an event, or both. Following is a list of the attributes and methods you may want to use on device and event objects. The syntax for accessing device attributes and methods is \${dev/attributename}. For example, to get the manageIp of a device you would use \${dev/manageIp}. For events, the syntax is \${evt/attributename}.

A command to ping a device might look like this. (The \${...} is a TALES expression to get the manageIp value for the device.)

```
ping -c 10 ${device/manageIp}
```

Examples

- DNS Forward Lookup (assumes device/id is a resolvable name)

```
host ${device/id}
```

- DNS Reverse Lookup

```
host ${device/manageIp}
```

- SNMP Walk

```
snmpwalk -v 2c -c${device/zSnmpCommunity} ${device/manageIp} system
```

To use these expressions effectively, you must know which objects, attributes, and methods are available, and in which contexts. Usually there is a device that allows you to access the device in a particular context. Contexts related to a particular event usually have event defined.

TALES device attributes

The following table lists available device attributes.

| Attribute | Description |
|-----------------------------|---|
| getId | The primary means of identifying a device within the system |
| getManageIp | The IP address used to contact the device in most situations |
| productionState | The production status of the device: Production, Pre-Production, Test, Maintenance or Decommissioned. This attribute is a numeric value, use getProductionStateString for a textual representation. |
| getProductionStateString | Returns a textual representation of the productionState |
| snmpAgent | The agent returned from SNMP collection |
| snmpDescr | The description returned by the SNMP agent |
| snmpOid | The oid returned by the SNMP agent |
| snmpContact | The contact returned by the SNMP agent |
| snmpSysName | The system name returned by the SNMP agent |
| snmpLocation | The location returned by the SNMP agent |
| snmpLastCollection | When SNMP collection was last performed on the device. This is a DateTime object. |
| getSnmpLastCollectionString | Textual representation of snmpLastCollection |
| rackSlot | The slot name/number in the rack where this physical device is installed |
| comments | User entered comments regarding the device |
| priority | A numeric value: 0 (Trivial), 1 (Lowest), 2 (Low), 3 (Normal), 4 (High), 5 (Highest) |
| getPriorityString | A textual representation of the priority |
| getHWManufacturerName | Name of the manufacturer of this hardware |
| getHWProductName | Name of this physical product |
| getHWProductKey | Used to associate this device with a hardware product class |
| getOSManufacturerName | Name of the manufacturer of this device's operating system. |
| getOSProductName | Name of the operating system running on this device. |
| getOSProductKey | Used to associate the operating system with a software product class |
| getHWSerialNumber | Serial number for this physical device |
| getLocationName | Name of the Location assigned to this device |
| getLocationLink | Link to the system page for the assigned Location |
| getSystemNames | A list of names of the Systems this device is associated with |
| getDeviceGroupNames | A list of names of the Groups this device is associated with |

| Attribute | Description |
|-----------------------|---|
| getLastChangeString | When the last change was made to this device |
| getLastPollSnmpUpTime | Uptime returned from SNMP |
| uptimeStr | Textual representation of the SNMP uptime for this device |
| getPingStatusString | Textual representation of the ping status of the device |
| getSnmpStatusString | Textual representation of the SNMP status of the device |

TALES event attributes

The following table lists available event attributes.

| Attribute | Description |
|---------------|--|
| agent | Collector name from which the event came (such as zensyslog or zentrap). |
| component | Component of the associated device, if applicable. (Examples: eth0, httpd.) |
| count | Number of times this event has been seen. |
| dedupid | Key used to correlate duplicate events. By default, this is: device, component, eventClass, eventKey, severity. |
| device | ID of the associated device, if applicable. |
| DeviceClass | Device class from device context. |
| DeviceGroups | Device systems from device context, separated by . |
| eventClass | Event class associated with this device. If not specified, may be added by the rule process. If this fails, then will be /Unknown. |
| eventClassKey | Key by which rules processing begins. Often equal to component. |
| eventGroup | Logical group of event source (such as syslog, ping, or nteventlog). |
| eventKey | Primary criteria for mapping events into event classes. Use if a component needs further de-duplication specification. |
| eventState | State of event. 0 = new, 1 = acknowledged, 2 = suppressed. |
| evid | Unique ID for the event. |
| facility | syslog facility, if this is a syslog event. |
| firstTime | UNIX timestamp when event is received. |
| ipAddress | IP Address of the associated device, if applicable. |
| lastTime | Last time this event was seen and its count incremented. |
| Location | Device location from device context. |
| manager | Fully qualified domain name of the collector from which this event came. |
| message | Full message text. |
| ntevid | nt event ID, if this is an nt eventlog event. |
| priority | syslog priority, if this is a syslog event. |

| Attribute | Description |
|----------------|---|
| prodState | prodState of the device context. |
| severity | the severity of the event expressed as a number (0, 1, 2, 3, 4, or 5) |
| severityString | the severity of the event expressed as a string (Clear, Debug, Info, Warning, Error, or Critical) |
| stateChange | Time the MySQLrecord for this event was last modified. |
| summary | Text description of the event. Limited to 255 characters. |
| suppid | ID of the event that suppressed this event. |
| Systems | Device systems from device context, separated by . |

Configuration properties and custom properties

Configuration properties and custom properties also are available for devices, and use the same syntax as shown in the previous sections.



Monitoring large file systems

By default, Zenoss Core uses the Host Resources MIB to monitor file systems. A defect in the implementation of the Host Resources MIB in net-snmp causes file systems larger than 16TB to report incorrect utilization metrics, such that you might observe file system utilization values greater than 100%.

Note Zenoss Core uses the Host Resources MIB in the ethernetCsmacd template (rather than the UCD dskTable MIB) by default because most of the systems Zenoss Core monitors do not have the UCD dskTable MIB enabled.

To work around this deficiency, Zenoss Core can instead use the UCD dskTable MIB to monitor file system utilization.

Procedure

To use the UCD dskTable MIB, you must modify the configuration of your data sources, thresholds, and graphs in your FileSystem template:

- 1 Create an SNMP data source named dskPercent.
- 2 Set the OID of the new data source to:

```
1.3.6.1.4.1.2021.9.1.9
```

- 3 Modify your thresholds to use the dskPercent data point. Remove any calculations that were associated with the Host Resource MIB data point. The dskPercent data point is reported as an integer from 0 to 100.
- 4 Modify your graphs to use the dskPercent data point and thresholds.
- 5 Enable the UCD dskTable MIB on your managed hosts:

- a Add the following line to your /etc/snmp/snmpd.conf file:

```
includeAllDisks 0%
```

- b Restart the snmpd daemon.

The UCD dskTable MIB associates different indexes with the file systems than the Host Resources MIB. As a result, you must remodel the devices to fetch the UCD dskTable indexes, and to begin plotting data on the dskPercent graph.

Glossary

aggregation pools

A logical bundling of multiple physical network interfaces, commonly known as a port channel. For example, the Per Chassis Ethernet Pools includes all links from all chassis to all fabric interconnects which is used for chassis bandwidth balance comparison. For more examples, see the Aggregation Pools component section of CiscoUCS devices.

bandwidth utilization

The total amount of bandwidth being used by an aggregation pool, a port, a fabric interconnect, a FEX, etc.

component

Object contained by a device. Components include interfaces, OS processes, file systems, CPUs, and hard drives.

data point

Data returned from a data source. In many cases, there is only one data point for a data source (such as in SNMP); but there may also be many data points for a data source (such as when a command results in the output of several variables).

data source

Method used to collect monitoring information. Example data sources include SNMP OIDs, SSH commands, and perfmon paths.

device

Primary monitoring object in the system. Generally, a device is the combination of hardware and an operating system.

device class

Special type of organizer used to manage how the system models and monitors devices (through configuration properties and monitoring templates).

discovery

Process by which Zenoss Core gathers detailed information about devices in the infrastructure. Results of discovery are used to populate the model.

event

Manifestation of important occurrence within the system. Events are generated internally (such as when a threshold is exceeded) or externally (such as through a syslog message or SNMP trap).

event class

Categorization system used to organize event rules.

event rules

Controls how events are manipulated as they enter the system (for example, changing the severity of an event). Configuration properties configure event rules.

graph

Displays one or more data points, thresholds, or both.

headroom

The unused bandwidth in an aggregation pool, a port, a fabric interconnect (FI), a FEX, etc. For example, if an aggregation pool including 4 ports between a chassis and the FIs has 40 GB of capacity and if the bandwidth use of that pool is 25 GB, then the headroom is 15 GB.

managed resource

Servers, networks, virtual machines, and other devices in the IT environment.

model

Representation of the IT infrastructure. The model tells the system "what is out there" and how to monitor it.

monitoring template

Description of what to monitor on a device or device component. Monitoring templates comprise four main elements: data sources, data points, thresholds, and graphs.

notification

Sends email or pages to system users or groups when certain events occur.

organizer

Hierarchical system used to describe locations and groups within the application. Zenoss Core also includes special organizers, which are classes that control system configuration.

out of balance

Indicates that the bandwidth use is quite different among the ports in an aggregation pool. This can often be corrected by reconfiguration, for example, by moving a service profile from one chassis to another.

resource component

Interfaces, services and processes, and installed software in the IT environment.

service definition

A service definition contains the information that Control Center needs to start and manage a service, in JavaScript Object Notation (JSON) format.

service profile

A service profile is a software definition of a server and its LAN and SAN network connectivity, in other words, a service profile defines a single server and its storage and networking characteristics.

service template

A service template contains one or more service definitions, in JavaScript Object Notation (JSON) format.

threshold

Defines a value beyond which a data point should not go. When a threshold is reached, the system generates an event. Typically, threshold events use the /Capacity event class.

trigger

Determines how and when notifications are sent. Specifies a rule comprising a series of one or more conditions.