



Zenoss Core Configuration Guide

Release 5.1.8

Zenoss, Inc.

www.zenoss.com

Zenoss Core Configuration Guide

Copyright © 2016 Zenoss, Inc. All rights reserved.

Zenoss 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 VMware, 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: 1041.16.291

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

Contents

About this guide.....	4
Chapter 1: Enabling access to browser interfaces.....	5
Creating public endpoints.....	5
Configuring name resolution for virtual hosts.....	14
Chapter 2: Configuring Zenoss Core.....	16
Starting Zenoss Core.....	16
Deleting the RabbitMQ guest user account.....	17
Creating a weekly maintenance script.....	17
MariaDB database utilities.....	18
Optional: Assigning a virtual IP address to a resource pool.....	20
Optional: Replacing the default digital certificate.....	20
Optional: Enabling monitoring on IPv6 networks.....	21
Optional: Customization management.....	22
Optional: Configuring OpenTSDB compaction.....	23
Chapter 3: Preparing for monitoring.....	25
Extending monitoring with ZenPacks.....	25
Preparing network devices.....	26
Preparing storage devices.....	27
Preparing server devices.....	29
Preparing hypervisor devices.....	29
Chapter 4: Modeling Devices.....	30
Configuring Windows Devices to Provide Data Through SNMP.....	30
Configuring Linux Devices to Provide Data Through SNMP.....	31
Modeling Devices Using SSH/COMMAND.....	31
Using Device Class to Monitor Devices Using SSH.....	32
Using the /Server/Scan Device Class to Monitor with Port Scan.....	32
Modeling Devices Using Port Scan.....	32
About Modeler Plugins.....	33
Debugging the Modeling Process.....	34
Next Steps.....	34
Appendix A: External HBase configuration.....	35
Configuring OpenTSDB for an external HBase cluster.....	35
Configuring the OpenTSDB service startup command.....	36
Disabling the Zenoss Core HBase cluster.....	37

About this guide

Zenoss Core Configuration Guide describes how to set up Zenoss Core and to prepare your environment for monitoring. Use this guide after completing all of the steps required for your deployment in *Zenoss Core Installation Guide*.

Related 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.

1

Enabling access to browser interfaces

Control Center and Zenoss Core have independent browser interfaces served by independent web servers. Both web servers are configured to use SSL/TLS communications.

The Control Center web server listens at the hostname of the Control Center master host and port 443. For a Control Center master host with the fully-qualified domain name (FQDN) `cc-master.example.com`, the hostname URL is `https://cc-master`. As always, you may substitute an IP address for the hostname portion of the URL.

The Zenoss Core web server can listen at *port public endpoints* and *virtual host public endpoints*.

- A *port public endpoint* is a combination of the IP address or hostname of the Control Center master host and a port number. The default configuration of Zenoss Core does not include any port public endpoints. If the Control Center master host has more than one interface, you can configure port public endpoints with different hostnames. Also, you can disable TLS communications for a port public endpoint, if desired.

To use a port public endpoint to gain access to the Zenoss Core browser interface, no additional network name resolution entries are required. The default entries for the network interface(s) of the Control Center master host are sufficient.

- The default *virtual host public endpoint* is the text `zenoss5` prepended to the hostname of the Control Center master host and port 50443. For the FQDN `cc-master.example.com`, the URL of the default virtual host public endpoint is `https://zenoss5.cc-master:50443`. You can change the name of the default virtual host and configure additional virtual host public endpoints, if desired.

To use a virtual host public endpoint to gain access to the Zenoss Core browser interface, you must add name resolution entries for the virtual host to the DNS servers in your environment or to the hosts files of individual client systems.

The following sections provide additional information about public endpoints, and instructions for creating public endpoints and for configuring virtual host name resolution.

Creating public endpoints

This section describes how to create a port public endpoint and a virtual host public endpoint. The following table outlines the process for each.

Port public endpoint	Virtual host public endpoint
1 Create the endpoint	1 Create the endpoint
2 Configure the Zope service	2 Configure the Zope service

Port public endpoint

Virtual host public endpoint

3 Configure virtual host name resolution

To change an existing public endpoint, create a new endpoint and then delete the existing endpoint.

Note Virtual host public endpoints must use SSL/TLS communications. Port public endpoints can communicate with or without SSL/TLS.

Creating a port public endpoint

Use this procedure to create a new port public endpoint. Port public endpoints can communicate with or without SSL/TLS.

- 1 Log in to the Control Center browser interface.
- 2 In the **Application** column of the **Applications** table, click the application name (**Zenoss.core**).

The screenshot shows the Zenoss Control Center interface. At the top, there's a navigation bar with 'Control Center' logo and menu items: Applications, Resource Pools, Hosts, Logs, Backup / Restore. Below that, the breadcrumb is 'Applications / Zenoss.resmgr'. The main content area shows the application 'Zenoss.resmgr (v5.1.4)' with buttons for 'Edit Service', 'Edit Variables', 'Start', 'Stop', and 'Restart'. Below this is a 'Public Endpoints' table with a '+ Add Public Endpoint' button on the right. The table has columns: Service, Endpoint, Type, Protocol, URL, and Actions. It lists four endpoints: HMaster, reader, RabbitMQ, and Zenoss.resmgr.

Service	Endpoint	Type	Protocol	URL	Actions
HMaster	hbase-masterinfo-1	vhost	https	https://hbase.resmgr:443	▶ Start ■ Stop ○ Delete
reader	opentsdb-reader	vhost	https	https://opentsdb.resmgr:443	▶ Start ■ Stop ○ Delete
RabbitMQ	rabbitmq_admin	vhost	https	https://rabbitmq.resmgr:443	▶ Start ■ Stop ○ Delete
Zenoss.resmgr	zproxy	vhost	https	https://zenoss5.resmgr:443	▶ Start ■ Stop ○ Delete

At the bottom right of the table, it says 'Last Update: a few seconds ago' and 'Showing 4 Results'.

- 3 Click **+ Add Public Endpoint**, located above the **Public Endpoints** table, on the right side.

The 'Add Public Endpoint' dialog box is shown. It has a title bar and a close button. The main content area contains instructions: 'VHost public endpoints are accessible by hostname (eg, https://zenoss.mckraken), while Port public endpoints are accessible by ip port or hostname:port (eg, myhost:54321 or 10.07.1.100:54321)'. Below this, there are two tabs: 'Port' (selected) and 'VHost'. The 'Service - Endpoint' dropdown is set to 'HMaster - hbase-master-1'. The 'Host' field contains 'cc-master'. The 'Port' field contains '54321'. The 'Protocol' dropdown is set to 'HTTPS'. At the bottom, there are two buttons: 'Cancel' and 'Add and Restart Service'.

The default view of the **Add Public Endpoint** dialog displays the fields for creating a port public endpoint.

- 4 Define a new port public endpoint.
 - a In the **Type** area, click **Port**.
 - b From the **Service - Endpoint** list, select **Zenoss.core - zproxy**.
The selection is the last entry in the list.
 - c In the **Host** field, enter a hostname or IP address that is assigned to a network interface on the Control Center master host.

The default value is the hostname that was added with the Deployment Wizard when Zenoss Core was initially deployed. If the Control Center master host has more than one network interface, you may add the hostname or IP address assigned to one of the other interfaces.

- d In the **Port** field, enter a safe, unused port number greater than or equal to 1024 and less than or equal to 65535.

For a list of ports considered unsafe, see [Unsafe ports on Chrome](#). For the list of ports used by the Control Center master host, refer to the [Zenoss Core Planning Guide](#).

- e In the **Protocol** field, select **HTTPS** or **HTTP**.

You can set up a secure proxy server to handle HTTP requests sent to a port public endpoint, if desired.

- f Click **Add and Restart Service**.

Configure the **Zope** service to use the new port public endpoint. Choose one of the configuration options in the following table.

Zope configuration	Procedure
HTTPS and the default secure proxy server	Configuring Zope for HTTPS and the default secure proxy server on page 7
HTTP and no proxy server	Configuring Zope for HTTP and no proxy server on page 9
HTTP and a secure proxy server other than the default	Configuring Zope for HTTP and a secure proxy server on page 10

Note When you configure Zope for HTTP protocol and no proxy server, you can only gain access to the Zenoss Core browser interface through port public endpoints configured for HTTP. Virtual host public endpoints must use HTTPS protocol, so any existing virtual host public endpoints stop working.

Configuring Zope for HTTPS and the default secure proxy server

Before performing this procedure, create a port public endpoint or a virtual host public endpoint to use the HTTPS protocol.

Use this procedure to configure the **Zope** service for SSL/TLS communications and the secure proxy server that is included in Zenoss Core.

- 1 Log in to the Control Center browser interface.
- 2 In the **Application** column of the **Applications** table, click the application name (**Zenoss.core**).
- 3 Scroll down to the bottom of the **Services** table, and then click **Zope**.
The Control Center browser interface displays the details page of the **Zope** service.
- 4 Scroll to the top of the **Zope** service details page.

The screenshot shows the Zenoss Control Center interface for the Zope service. The top navigation bar includes 'Applications', 'Resource Pools', 'Hosts', 'Logs', and 'Backup / Restore'. The main content area is titled 'Zope' and includes buttons for 'Edit Service', 'Edit Variables', 'Start', 'Stop', and 'Restart'. Below this are three tables: 'Public Endpoints', 'IP Assignments', and 'Configuration Files'. The 'Configuration Files' table has two entries: '/opt/zenoss/etc/global.conf' and '/opt/zenoss/etc/zope.conf', each with an 'Edit' button in the 'Actions' column.

- 5 In the **Actions** column of the **Configuration Files** table, click the **Edit** control of the `/opt/zenoss/etc/zope.conf` file.

The screenshot shows the 'Edit Configuration' dialog for the file `/opt/zenoss/etc/zope.conf`. The dialog contains a text editor with the following content:

```

1 # This is the proposed version for SC / Zope 2.12.1
2
3 #####
4 # Welcome to Zope 2.
5 #####
6 #
7 # This is the Zope configuration file. The Zope configuration file
8 # shows what the default configuration directives are, and show
9 # examples for each directive. To declare a directive, make sure that
10 # you add it to a line that does not begin with '#'. Note that comments
11 # are only allowed at the beginning of a line: you may not add comments
12 # after directive text on the same line.
13 #
14 # Note for Developers
15 # -----
16 #
17 # This file is *not* auto-generated. If you create a new directive you
18 # very likely want to include an example of how to use the new
19 # directive in this file.
20 #
21 # You shouldn't modify 'zope.conf.in' to change
22 # configuration. Instead, you should make a copy into 'zope.conf' and
23 # modify that to avoid checking in changes to this file by mistake.
24 #
25 # ZConfig "defines" used for later textual substitution
26 #
27 #define INSTANCE /opt/zenoss
28 #
29 # this needs to match the encoding in the sitecustomize.py file
30 # in $ZENHOME/lib/python
31 default-zpublisher-encoding utf-8
32
33 # Directive: instancehome
34 #
35 # Description:
36 # This sets the data files, local product files, format directories
  
```

At the bottom of the dialog, there are 'Cancel' and 'Save' buttons. The 'Save' button is highlighted.

- 6 Configure Zope for secure communications with the proxy server.
 - a In the **Edit Configuration** dialog, scroll down to the `cgi-environment` directive. The directive is about one-third of the way down from the top of the file, on or near line 380.
 - b Configure the proxy server for SSL/TLS communications:

```

<cgi-environment>
  HTTPS ON
</cgi-environment>
  
```

- 7 Configure the Beaker add-on product to use secure communications.
 - a In the **Edit Configuration** dialog, scroll down to the `product-config` directive. The directive is at the bottom the file, on or near line 1122.
 - b Set the value of the `session.secure` key to `True`.
- 8 At the bottom of the **Edit Configuration** dialog, click **Save**.

Next steps:

- If you created a port public endpoint before performing this procedure, the endpoint is ready to use.
- If you created a virtual host public endpoint before performing this procedure, proceed to [Configuring name resolution for virtual hosts](#) on page 14.

Configuring Zope for HTTP and no proxy server

Before performing this procedure, create a port public endpoint to use the HTTP protocol. For more information, see [Creating a port public endpoint](#) on page 6.

Use this procedure to configure the **Zope** service for insecure communications with Zenoss Core browser interface clients.

Note When you configure Zope for insecure communications, all existing virtual host public endpoints stop working.

- 1 Log in to the Control Center browser interface.
- 2 In the **Application** column of the **Applications** table, click the application name (**Zenoss.core**).
- 3 Scroll down to the bottom of the **Services** table, and then click **Zope**.
The Control Center browser interface displays the details page of the **Zope** service.
- 4 Scroll to the top of the **Zope** service details page.

The screenshot shows the Zenoss Control Center interface for the Zope service. The breadcrumb trail is: Applications / Zenoss.core / Zenoss / User Interface / Zope. The page title is "Zope" with subtext "Zope server". There are action buttons: Edit Service, Edit Variables, Start, Stop, and Restart. Below this are three tables:

- Public Endpoints:** A table with columns: Service, Endpoint, Type, Protocol, URL, Actions. It shows "No Data Found".
- IP Assignments:** A table with columns: Service, Assignment Type, Host, Resource Pool, IP, Actions. It shows "No Data Found".
- Configuration Files:** A table with columns: Path, Actions. It lists two files:

Path	Actions
/opt/zenoss/etc/global.conf	Edit
/opt/zenoss/etc/zope.conf	Edit

- 5 In the **Actions** column of the **Configuration Files** table, click the **Edit** control of the **/opt/zenoss/etc/zope.conf** file.

```

1 # This is the proposed version for SC / Zope 2.12.1
2
3 #####
4 # Welcome to Zope 2.
5 #####
6 #
7 # This is the Zope configuration file. The Zope configuration file
8 # shows what the default configuration directives are, and show
9 # examples for each directive. To declare a directive, make sure that
10 # you add it to a line that does not begin with '#'. Note that comments
11 # are only allowed at the beginning of a line: you may not add comments
12 # after directive text on the same line.
13 #
14 # Note for Developers
15 # =====
16 #
17 # This file is 'not' auto-generated. If you create a new directive you
18 # very likely want to include an example of how to use the new
19 # directive in this file.
20 #
21 # You shouldn't modify 'zope.conf.in' to change
22 # configuration. Instead, you should make a copy into 'zope.conf' and
23 # modify that to avoid checking in changes to this file by mistake.
24 #
25 # ZConfig "defines" used for later textual substitution
26 #
27 %define INSTANCE /opt/zenoss
28 #
29 # this needs to match the encoding in the sitecustomize.py file
30 # in $ZENHOME/lib/python
31 default-publisher-encoding utf-8
32 #
33 # Directive: instancehome
34 #
35 # Description:
36 #   The path to the data files (local product files, forest directory

```

6 Configure Zope for insecure communications with the proxy server.

a In the **Edit Configuration** dialog, scroll down to the `cgi-environment` directive.

The directive is about one-third of the way down from the top of the file, on or near line 380.

b Configure the proxy server for insecure communications:

```

<cgi-environment>
  HTTPS OFF
</cgi-environment>

```

7 Configure the Beaker add-on product to use insecure communications.

a In the **Edit Configuration** dialog, scroll down to the `product-config` directive.

The directive is at the bottom the file, on or near line 1122.

b Set the value of the `session.secure` key to `False`.

8 At the bottom of the **Edit Configuration** dialog, click **Save**.

Configuring Zope for HTTP and a secure proxy server

Before performing this procedure, create a port public endpoint to use the HTTP protocol. For more information, see [Creating a port public endpoint](#) on page 6.

Use this procedure to configure the **Zope** service for SSL/TLS communications and a secure proxy server that is available on your network.

1 Log in to the Control Center browser interface.

2 In the **Application** column of the **Applications** table, click the application name (**Zenoss.core**).

3 Scroll down to the bottom of the **Services** table, and then click **Zope**.

The Control Center browser interface displays the details page of the **Zope** service.

4 Scroll to the top of the **Zope** service details page.

The screenshot shows the Zenoss Control Center interface for the Zope service. The top navigation bar includes 'Applications', 'Resource Pools', 'Hosts', 'Logs', and 'Backup / Restore'. The main content area is titled 'Zope' and includes a 'Zope server' status bar with 'Edit Service', 'Edit Variables', 'Start', 'Stop', and 'Restart' buttons. Below this are three tables: 'Public Endpoints' (No Data Found), 'IP Assignments' (No Data Found), and 'Configuration Files'. The 'Configuration Files' table has two entries: '/opt/zenoss/etc/global.conf' and '/opt/zenoss/etc/zope.conf', each with an 'Edit' button in the 'Actions' column.

- 5 In the **Actions** column of the **Configuration Files** table, click the **Edit** control of the `/opt/zenoss/etc/zope.conf` file.

The screenshot shows the 'Edit Configuration' dialog for the file `/opt/zenoss/etc/zope.conf`. The dialog contains a text editor with the following content:

```

1 # This is the proposed version for SC / Zope 2.12.1
2
3 #####
4 # Welcome to Zope 2.
5 #####
6 #
7 # This is the Zope configuration file. The Zope configuration file
8 # shows what the default configuration directives are, and show
9 # examples for each directive. To declare a directive, make sure that
10 # you add it to a line that does not begin with '#'. Note that comments
11 # are only allowed at the beginning of a line: you may not add comments
12 # after directive text on the same line.
13 #
14 # Note for Developers
15 # -----
16 #
17 # This file is *not* auto-generated. If you create a new directive you
18 # very likely want to include an example of how to use the new
19 # directive in this file.
20 #
21 # You shouldn't modify 'zope.conf.in' to change
22 # configuration. Instead, you should make a copy into 'zope.conf' and
23 # modify that to avoid checking in changes to this file by mistake.
24 #
25 # ZConfig "defines" used for later textual substitution
26 #
27 #define INSTANCE /opt/zenoss
28 #
29 # this needs to match the encoding in the sitecustomize.py file
30 # in $ZENHOME/lib/python
31 default-zpublisher-encoding utf-8
32
33 # Directive: instancehome
34 #
35 # Description:
36 #   This sets up the data files, local product files, format directives
  
```

At the bottom right of the dialog, there are 'Cancel' and 'Save' buttons.

- 6 Configure Zope for secure communications with your proxy server.
 - a In the **Edit Configuration** dialog, scroll down to the `cgi-environment` directive. The directive is about one-third of the way down from the top of the file, on or near line 380.
 - b Configure the proxy server for SSL/TLS communications:

```

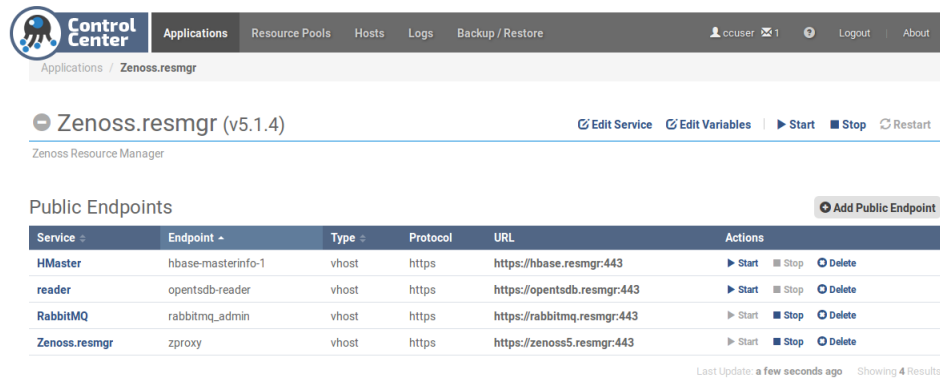
<cgi-environment>
  HTTPS ON
</cgi-environment>
  
```

- 7 Configure the Beaker add-on product to use secure communications.
 - a In the **Edit Configuration** dialog, scroll down to the `product-config` directive. The directive is at the bottom the file, on or near line 1122.
 - b Set the value of the `session.secure` key to `True`.
- 8 At the bottom of the **Edit Configuration** dialog, click **Save**.

Creating a virtual host public endpoint

Use this procedure to create a new virtual host public endpoint. Virtual host public endpoints must use SSL/TLS communications.

- 1 Log in to the Control Center browser interface.
- 2 In the **Application** column of the **Applications** table, click the application name (**Zenoss.core**).

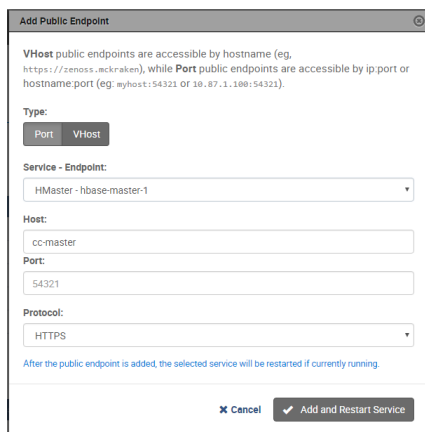


The screenshot shows the Zenoss Control Center interface. The top navigation bar includes 'Control Center', 'Applications', 'Resource Pools', 'Hosts', 'Logs', 'Backup / Restore', and user information. The main content area displays the 'Zenoss.resmgr (v5.1.4)' application page. Below the application name, there are buttons for 'Edit Service', 'Edit Variables', 'Start', 'Stop', and 'Restart'. A 'Public Endpoints' table is shown with the following data:

Service	Endpoint	Type	Protocol	URL	Actions
HMaster	hbase-masterinfo-1	vhost	https	https://hbase.resmgr:443	Start Stop Delete
reader	opentsdb-reader	vhost	https	https://opentsdb.resmgr:443	Start Stop Delete
RabbitMQ	rabbitmq_admin	vhost	https	https://rabbitmq.resmgr:443	Start Stop Delete
Zenoss.resmgr	zproxy	vhost	https	https://zenoss5.resmgr:443	Start Stop Delete

At the bottom right of the table, there is a '+ Add Public Endpoint' button and a status message: 'Last Update: a few seconds ago Showing 4 Results'.

- 3 Click **+ Add Public Endpoint**, located above the **Public Endpoints** table, on the right side.



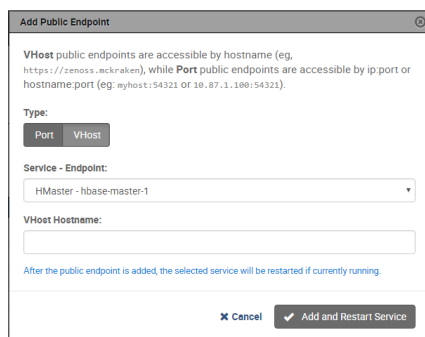
The 'Add Public Endpoint' dialog box is shown. It contains the following fields and options:

- Type:** Port (selected), VHost
- Service - Endpoint:** HMaster - hbase-master-1
- Host:** cc-master
- Port:** 54321
- Protocol:** HTTPS

At the bottom, there are 'Cancel' and 'Add and Restart Service' buttons. A note at the bottom states: 'After the public endpoint is added, the selected service will be restarted if currently running.'

The default view of the **Add Public Endpoint** dialog displays the fields for creating a port public endpoint.

- 4 Define a new virtual host public endpoint.
 - a In the **Type** area, click **VHost**.



The 'Add Public Endpoint' dialog box is shown. It contains the following fields and options:

- Type:** Port, VHost (selected)
- Service - Endpoint:** HMaster - hbase-master-1
- VHost Hostname:** (empty field)

At the bottom, there are 'Cancel' and 'Add and Restart Service' buttons. A note at the bottom states: 'After the public endpoint is added, the selected service will be restarted if currently running.'

- b From the **Service - Endpoint** list, select **Zenoss.core - zproxy**.

The selection is the last entry in the list.

- c In the **VHost Hostname** field, enter a virtual hostname.

The following strings of text are valid in this field:

- A fully-qualified domain name (FQDN). Any string of text that includes one or more full stop characters (.) is treated as an FQDN.
- A string of text that contains only letters and one or more hyphen characters (-). The string is prepended to the hostname of the Control Center master host, with a full stop character (.) separating the string and the hostname.

- d Click **Add and Restart Service**.

Configuring Zope for HTTPS and the default secure proxy server

Before performing this procedure, create a port public endpoint or a virtual host public endpoint to use the HTTPS protocol.

Use this procedure to configure the **Zope** service for SSL/TLS communications and the secure proxy server that is included in Zenoss Core.

- 1 Log in to the Control Center browser interface.
- 2 In the **Application** column of the **Applications** table, click the application name (**Zenoss.core**).
- 3 Scroll down to the bottom of the **Services** table, and then click **Zope**.
The Control Center browser interface displays the details page of the **Zope** service.
- 4 Scroll to the top of the **Zope** service details page.

The screenshot shows the Zenoss Control Center interface for the Zope service. At the top, there is a navigation bar with tabs for Applications, Resource Pools, Hosts, Logs, and Backup / Restore. Below this, the breadcrumb trail reads: Applications / Zenoss.core / Zenoss / User Interface / Zope. The main content area is titled 'Zope' and includes a 'Zope server' label. There are three tables displayed: 'Public Endpoints' (No Data Found), 'IP Assignments' (No Data Found), and 'Configuration Files'. The 'Configuration Files' table has two rows: one for '/opt/zenoss/etc/global.conf' and another for '/opt/zenoss/etc/zope.conf'. The second row has an 'Edit' button in the 'Actions' column.

- 5 In the **Actions** column of the **Configuration Files** table, click the **Edit** control of the **/opt/zenoss/etc/zope.conf** file.

- 6 Configure Zope for secure communications with the proxy server.
 - a In the **Edit Configuration** dialog, scroll down to the `cgi-environment` directive.

The directive is about one-third of the way down from the top of the file, on or near line 380.
 - b Configure the proxy server for SSL/TLS communications:

```
<cgi-environment>
  HTTPS ON
</cgi-environment>
```

- 7 Configure the Beaker add-on product to use secure communications.
 - a In the **Edit Configuration** dialog, scroll down to the `product-config` directive.

The directive is at the bottom the file, on or near line 1122.
 - b Set the value of the `session.secure` key to `True`.
- 8 At the bottom of the **Edit Configuration** dialog, click **Save**.

Next steps:

- If you created a port public endpoint before performing this procedure, the endpoint is ready to use.
- If you created a virtual host public endpoint before performing this procedure, proceed to [Configuring name resolution for virtual hosts](#) on page 14.

Configuring name resolution for virtual hosts

To enable access to browser interfaces by virtual hosts, add name resolution entries to the DNS servers in your environment or to the hosts files of individual client systems.

- On Windows client systems, the hosts file is `C:\Windows\System32\drivers\etc\hosts`.
- Linux and OS/X client systems, the hosts file is `/etc/hosts`.

Name resolution syntax

The following line shows the syntax of the entry to add to a name resolution file:

```
IP-Address FQDN Hostname zenoss5.Hostname
```

For example, the following entry identifies a Control Center master host at IP address 192.0.2.12, hostname `cc-master`, in the `example.com` domain.

```
192.0.2.12 cc-master.example.com cc-master zenoss5.cc-master
```

Configuring name resolution on a Windows 7 system

To perform this procedure, you need Windows Administrator privileges.

- 1 Log in to the Windows 7 system as a user with Administrator privileges.
- 2 From the **Start** menu, highlight **All Programs > Accessories > Notepad**.
- 3 Right click, and then select **Run as administrator**.
- 4 From the Notepad **File** menu, select **Open**.
- 5 In the **File name** field of the **Open** window, enter `C:\Windows\System32\drivers\etc\hosts`.
- 6 Add a name resolution entry to the end of the file.
For more information, see [Name resolution syntax](#) on page 14.
- 7 Save the file, and then exit Notepad.

Configuring name resolution on a Linux or OS/X system

To perform this procedure, you need superuser privileges on the client system.

- 1 Log in to the client system as `root` or as a user with `sudo` privileges.
- 2 Open the `/etc/hosts` file in a text editor.
- 3 Add a name resolution entry to the end of the file.
For more information, see [Name resolution syntax](#) on page 14.
- 4 Save the file, and then close the editor.

2

Configuring Zenoss Core

This chapter contains configuration procedures that you perform after Zenoss Core is installed. Some of the procedures are optional, and indicated as such in the section title. For installation and deployment instructions, refer to the *Zenoss Core Installation Guide*.

Starting Zenoss Core

You can start Zenoss Core from the Control Center browser interface or from the command-line interface.

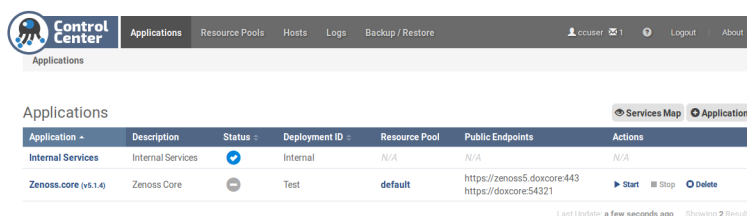
Using the Control Center browser interface to start Zenoss Core

To perform this procedure, you need:

- A supported client system and browser
- A user account on the Control Center master host with access privileges for the Control Center browser interface

For more information, refer to the *Zenoss Core Installation Guide*.

- 1 Log in to the Control Center browser interface.



- 2 In the **Actions** column of the **Applications** table, click **Start** for **Zenoss.core**.
- 3 In the **Start Service** dialog, click **Start Service** and all 40 child services.
- 4 Optional: Monitor the startup, if desired.
 - a In the **Applications** table, click **Zenoss.core**.
 - b Scroll down to the **Services** table and review the **Instances** icon for each service.

As services are started the **Instance** icon changes from a minus sign to a check mark.

Using the command line to start Zenoss Core

To perform this procedure, you need serviced CLI privileges. For more information, refer to the *Zenoss Core Installation Guide*.

- 1 Log in to the Control Center master host as a user with `serviced` CLI privileges.
- 2 Start Zenoss Core:

```
serviced service start Zenoss.core
```

- 3 Optional: Monitor the startup, if desired:

```
serviced service status Zenoss.core
```

Deleting the RabbitMQ guest user account

By default, RabbitMQ distributions include the `guest` user account. To prevent security issues, Zenoss recommends deleting the account.

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

```
serviced service attach rabbitmq
```

- 3 Delete the guest user account.

```
rabbitmqctl delete_user guest
```

- 4 Exit the container session.

```
exit
```

- 5 Restart the RabbitMQ service.

```
serviced service restart rabbitmq
```

Creating a weekly maintenance script

The Zenoss Core databases require regular maintenance to perform optimally. This procedure creates a script for weekly invocation, to perform the required maintenance.

- 1 Log in to the Control Center master host as `root`, or as a user with superuser privileges.
- 2 Determine whether the `crontie-anacron` package is installed.
 - a Search the list of installed packages.

```
rpm -qa | grep -i anacron
```

- If the command returns no result, perform the next substep.
- If the command returns a result, skip to the next step.

- b Install the package, if necessary.

```
yum install -y crontie-anacron
```

- 3 Determine whether a script is already installed.
Some packages of Zenoss Core install the script.

```
ls -l /etc/cron.weekly
```

- If the result includes `serviced-zenossdbpack`, the maintenance script is installed. Stop performing this procedure.
 - If the result does not include `serviced-zenossdbpack`, the maintenance script is installed. Complete this procedure.
- 4 Create a shell script for weekly invocation.
 - a Open `/etc/cron.weekly/serviced-zenossdbpack` with a text editor.
The file is empty.
 - b Add the following content to the file.

```
#!/bin/sh
${SERVICED:=/opt/serviced/bin/serviced} service run zope
zenossdbpack
```

- c Save the file, and then close the text editor.
- 5 Set file permissions.

```
chmod 0755 /etc/cron.weekly/serviced
```

MariaDB database utilities

The *Percona Toolkit* is a collection of helpful utilities for MySQL and MariaDB databases. For licensing reasons, Zenoss can not distribute it. Zenoss strongly recommends that all installations of Zenoss Core install the Percona Toolkit.

Installing the Percona Toolkit with internet access

To perform this procedure, you need one of the following:

- a login account on the master host that is a member of the `docker` group
- the password of the `root` user account

For more information, refer to the *Zenoss Core Installation Guide*.

- 1 Log in to the Control Center master host.
- 2 Install the package.

```
serviced service run zope install-percona
```

At the end of the installation process, the message `Container not committed` is displayed. This is normal. The tools are installed in the distributed file system, not in an image.

Installing the Percona Toolkit without internet access

To perform this procedure, you need one of the following:

- a login account on the master host that is a member of the `docker` group
- the password of the `root` user account

In addition, you need the Percona Toolkit package file. This procedure includes steps for downloading it to a client system, and then copying it to the Control Center master host.

- 1 On a client system, use a web browser to download *the latest version of the Percona Toolkit package*.
- 2 Log in to the Control Center master host.

- 3 Prepare the package for installation.
 - a On the Control Center master host, create a directory for the package, and then change directory.

```
mkdir /tmp/percona && cd /tmp/percona
```

- b Copy the package to the temporary location.
You may use a file transfer utility such as *WinSCP*.
- c Update the access permissions of the file and directory.

```
chmod -R 777 /tmp/percona
```

- 4 Start a shell as the zenoss user in a Zope container.
 - a Change directory to the location of the Percona Toolkit file.

```
cd /tmp/percona
```

- b Start an interactive shell in a Zope container and save a snapshot named PerconaToolkit.

```
mySnap=InstallPerconaToolkit  
serviced service shell -i -s $mySnap zope bash
```

- c Switch user to zenoss.

```
su - zenoss
```

- 5 Install the package and exit the Zope container.
 - a Create a directory for the package.

```
PERCONADIR=/var/zenoss/percona  
mkdir -p $PERCONADIR
```

- b Extract the package files.

Replace *Version* with the version number of the package file:

```
tar --strip-components=1 -C $PERCONADIR -xzvf \  
/mnt/pwd/percona-toolkit-Version.tar.gz
```

- c Exit the zenoss shell.

```
exit
```

- d Exit the Zope container.

```
exit
```

- 6 Commit the named snapshot.

```
serviced snapshot commit $mySnap
```

- 7 Restart the zeneventserver service.

```
serviced service restart zeneventserver
```

Optional: Assigning a virtual IP address to a resource pool

The **zentrap** and **zensyslog** services are designed to receive data from devices in your environment at a specific IP address. Typically, the address is assigned to a specific host. However, if the host fails, then no data is received. To avoid this issue, you can assign a virtual IP address to a resource pool, and then Control Center can create a virtual IP interface on any host in the pool. Zenoss recommends using a virtual IP with resource pools that include Zenoss Core collection services as a best practice.

To perform this procedure, you need an unused IPv4 address in the same subnet as the other hosts in the resource pool to modify. To avoid conflicts, ask your networking specialist to assign or reserve the address. In addition, all of the hosts in the resource pool to modify must have the same network interface names.

- 1 Log in to the Control Center browser interface.
- 2 At the top of the page, click **Resource Pools**.
- 3 In the **Resource Pool** column of the **Resource Pools** table, click the name of the resource pool to modify.
- 4 At the right side of the **Virtual IPs** table, click **Add Virtual IP**.
- 5 In the **Add Virtual IP** dialog, specify the virtual IP.

- a In the **IP** field, enter an IPv4 address.

The address must be in the same subnet as the other hosts in the current resource pool.

- b In the **Netmask** field, enter an IPv4 subnet mask.

The mask must match the range of addresses in the current resource pool. The following table associates commonly-used subnet masks with the number of addresses they include.

Subnet mask	Addresses in subnet
255.255.255.192	64
255.255.255.224	32
255.255.255.240	16
255.255.255.248	8

- c In the **Interface** field, enter the name of the network interface that is used on all hosts in the resource pool.
- d At the bottom of the **Add Virtual IP** dialog, click **Add Virtual IP**.

When you configure devices to send `syslog` or SNMP trap messages, use the virtual IP address assigned to a resource pool.

Optional: Replacing the default digital certificate

The default configuration of the Zenoss Core web server uses a Zenoss self-signed certificate for SSL/TLS communications. Use this procedure to install your own digital certificate.

To perform this procedure, you need:

- the certificate and key files of a digital certificate from a certificate authority or from a digital certificate created with a utility such as OpenSSL

Note Currently, certificates that require a passphrase are not supported.

- superuser privileges on the Control Center master host

- 1 Log in to the Control Center master host.
- 2 Copy the certificate and key files of your digital certificate to `/etc` on the master host.
You can store the files in any location that remains unchanged during operating system upgrades.
- 3 Configure Control Center to use your digital certificate.

- a Open `/etc/default/serviced` with a text editor.
 - b Locate the `SERVICED_CERT_FILE` declaration, and then replace its value with the absolute path of your certificate file.
 - c Remove the number sign character (`#`) from the beginning of the line.
 - d Locate the `SERVICED_KEY_FILE` declaration, and then replace its value with the absolute path of your key file.
 - e Remove the number sign character (`#`) from the beginning of the line.
 - f Save the file, and then close editor.
- 4 Reload the Control Center service.

```
systemctl reload serviced
```

Optional: Enabling monitoring on IPv6 networks

This procedure describes how to configure Zenoss Core to enable monitoring of devices that are located on an IPv6 network. The network must be reachable from the IPv4 network environment in which Control Center is deployed.

Use this procedure to route an IPv6 address block to Control Center using Docker's virtual bridge interface, `docker0`. Zenoss Core can monitor IPv6 devices that have addresses in the routed block.

To perform this procedure, you need the IPv6 subnet to route to Control Center, in CIDR notation. The subnet must have a minimum routing prefix length of 80 bits.

The following network parameters provide an example:

- Router IP address: `2001:0db8:200b::1/64`
 - Resource pool IP address: `2001:0db8:200b::2/64`
 - Routed subnet (to resource pool's IP): `2001:0db8:dce3::/80`
- 1 Log on to the Control Center master host as `root`, or as a user with superuser privileges.
 - 2 Configure IPv6 packet forwarding.
 - a Open `/etc/sysctl.d/ipv6.conf` with a text editor.
 - b Add or edit the following line:

```
net.ipv6.conf.all.forwarding=1
```

- c Save the file, and then close the text editor.
- 3 Enable IPv6 packet forwarding without rebooting the host.

```
sysctl -w net.ipv6.conf.all.forwarding=1
```

- 4 Configure Docker for IPv6 communications.
 - a Open `/etc/sysconfig/docker` with a text editor.
 - b Add the following flags to the end of the `OPTIONS` declaration.
Replace `Subnet-Block` with the IPv6 subnet to route to Control Center, in CIDR notation:

```
--ipv6 --fixed-cidr-v6="Subnet-Block"
```

- c Change the delimiter of the `OPTIONS` declaration to the apostrophe character (`'`).
The default delimiter of the `OPTIONS` declaration is the quotation mark character (`"`), which is the same delimiter used with the `--fixed-cidr-ipv6` flag.
- d Save the file, and then close the text editor.

- Restart the Docker service.

```
systemctl restart docker
```

- Use the Docker container of the **zenping** service to ping a known IPv6 address.

```
serviced service attach zenping ping6 -c 1 ipv6.google.com
```

If the ping is successful, Docker is able to resolve IPv6 addresses and you can monitor devices on the IPv6 network.

Optional: Customization management

Zenoss Core software is distributed as Docker images. Upgrades often replace images, so customizations of Zenoss Core services are lost, unless customizations are installed with a change management system.

Quilt is a utility for managing software changes, and Zenoss recommends installing it to manage customizations.

Installing Quilt with internet access

To perform this procedure, you need superuser privileges on the Control Center master host.

Use this procedure to add the Quilt patch management system to Zenoss Core.

- Log in to the Control Center master host.
- Install the Quilt package.

```
serviced service run zope install-quilt
```

Installing Quilt without internet access

To perform this procedure, you need superuser privileges on the Control Center master host and the Quilt package file. This procedure includes steps for downloading the package to a client system, and then copying it to the Control Center master host.

Use this procedure to add the Quilt patch management system to Zenoss Core.

- On a client system, use a web browser to download *the latest version of the Quilt package*.
- Log in to the Control Center master host.
- Prepare the package for installation.
 - On the Control Center master host, create a directory for the package, and then change directory.

```
mkdir /tmp/quilt && cd /tmp/quilt
```

- Copy the package to the temporary location.
You may use a file transfer utility such as *WinSCP*.
- Update the access permissions of the file and directory.

```
chmod -R 777 /tmp/quilt
```

- Start a shell as the `zenoss` user in a Zope container.
 - Change directory to the location of the Quilt package file.

```
cd /tmp/quilt
```

- b Start an interactive shell in a Zope container and save a snapshot named InstallQuilt.

```
mySnap=InstallQuilt
serviced service shell -i -s $mySnap zope bash
```

- c Switch user to zenoss.

```
su - zenoss
```

- 5 Extract the package files, and then compile and install Quilt.

- a Extract the package files.

```
tar xzvf /mnt/pwd/quilt-*.tar.gz -C /tmp
```

- b Compile and install the package.

```
cd /tmp/quilt-* && ./configure --prefix=/opt/zenoss/var/ext \
&& make && make install
```

- 6 Exit the container.

- a Exit the zenoss shell.

```
exit
```

- b Exit the Zope container.

```
exit
```

- 7 Commit the named snapshot.

```
serviced snapshot commit $mySnap
```

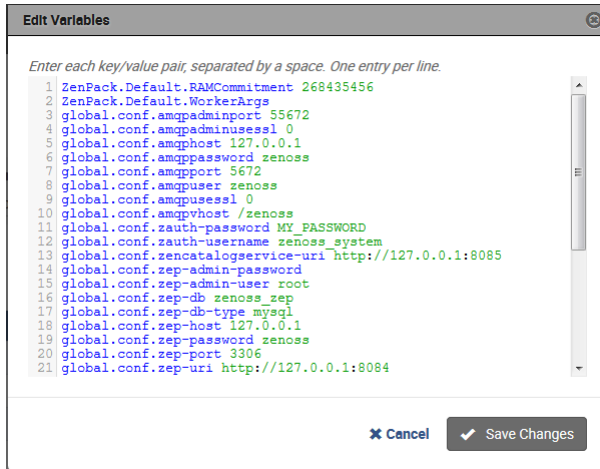
Optional: Configuring OpenTSDB compaction

Zenoss Core uses OpenTSDB to store the monitoring data it collects. When OpenTSDB compaction is enabled, multiple columns in an HBase row are merged into a single column, to reduce disk space. In testing, Zenoss has observed that these merges result in duplicate data points, so by default, compaction is disabled. Duplicate data points do not affect data integrity.

Note Enabling compaction slows performance and is not recommended.

- 1 Log in to the Control Center browser interface.
- 2 In the **Applications** table, click **Zenoss.core**.
- 3 In the application title line, click **Edit Variables**.

Initially, the application title line appears immediately below the Control Center banner at the top of the page. When you scroll down the page, the application title line persists at the top of the page.

Figure 1: Edit Variables dialog

- 4 In the **Edit Variables** dialog, scroll to the bottom of the list.
- 5 Change the value of the `tsd.storage.enable_compaction` variable from `False` to `True`.
- 6 Click **Save Changes**.
- 7 Restart the OpenTSDB services.
 - a Scroll down the page to the **Services** table, and then locate the **opentsdb** service.
 - b In the **Actions** column of the **opentsdb** service, click the **Restart** control.

3

Preparing for monitoring

Zenoss Core uses standard management APIs to collect performance data, and therefore does not install proprietary agents on your infrastructure devices to collect monitoring data. However, Zenoss recommends that you review the information in this chapter to verify that the devices to you want to monitor are ready to respond to requests for data.

Note This chapter describes how to prepare the most common IT infrastructure. If the infrastructure you want to monitor is not described here, please refer to the corresponding ZenPack documentation in the [Zenpack Catalog](#).

When your infrastructure is ready to monitor, the Zenoss Core Setup Wizard guides you through the process of discovering devices on your network and adding devices by category and type.

Extending monitoring with ZenPacks

Data centers typically contain many different types of hardware, software, and cloud services from a long list vendors. To keep your company's data secure, all devices, network infrastructure, and services must be monitored.

Zenoss Core is ready to monitor a large number of common devices and network infrastructure as soon it is installed. However, you can monitor an even larger number of devices in Zenoss Core through the use of ZenPacks. A ZenPack is a plug-in that extends not only monitoring capabilities, but also adds new capabilities to the Zenoss Core itself. 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.

There are hundreds of ZenPacks available, some developed, supported, and maintained by Zenoss, and many others that are developed and maintained by the Zenoss user community.

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 Zenoss Core. More complex ZenPacks require development of scripts or daemons, using Python or another programming language. ZenPacks can also be distributed for

installation on other Zenoss Core systems. For information on how to create a new ZenPack, refer to *Zenoss Core Administration Guide*.

ZenPack information resources

Zenoss Core includes a link (the question mark icon) to the documentation of the ZenPacks that are included in your installation of Zenoss Core. It also provides access to the [ZenPack Catalog](#), which provides detailed descriptions of all of the ZenPacks developed by Zenoss.

You may also create your own ZenPacks, or download and install ZenPacks developed by others. The following list identifies ZenPack resources:

- [Zenoss Community](#)
- [Public Zenoss repositories on GitHub](#)

Displaying installed ZenPacks in Zenoss Core

To display the pre-installed ZenPacks on Zenoss Core:

- 1 In the browser interface, select the **ADVANCED** tab.
- 2 In the left column, select **ZenPacks**.

The following figure shows an example list of ZenPacks.

Pack	Package	Author	Version	Egg
ZenPacks.zenoss.AdvancedSearch	zenoss	Zenoss	1.1.4	Yes
ZenPacks.zenoss.AixMonitor	zenoss	Zenoss	1.3.0	Yes
ZenPacks.zenoss.ApacheMonitor	zenoss	Zenoss	2.1.4	Yes
ZenPacks.zenoss.AuditLog	zenoss	Zenoss	1.3.0	Yes
ZenPacks.zenoss.BigIpMonitor	zenoss	Zenoss	2.6.3	Yes
ZenPacks.zenoss.BrocadeMonitor	zenoss	Zenoss	2.1.1	Yes
ZenPacks.zenoss.CatalogService	zenoss	Zenoss	3.0.9	Yes
ZenPacks.zenoss.CheckPointMonitor	zenoss	Zenoss	2.0.0	Yes
ZenPacks.zenoss.CiscoMonitor	zenoss	Zenoss	5.3.1	Yes
ZenPacks.zenoss.CiscoUCS	zenoss	Zenoss	1.9.1	Yes
ZenPacks.zenoss.ControlCenter	zenoss	Zenoss	1.0.0	Yes
ZenPacks.zenoss.Dashboard	zenoss	Zenoss	1.0.3	Yes
ZenPacks.zenoss.DellMonitor	zenoss	Zenoss	2.2.0	Yes

- 3 To monitor infrastructure that does not appear in the **Loaded ZenPacks** list, download the required ZenPack from the [ZenPack Catalog](#).

Once the ZenPack is installed, you can then add the infrastructure to Zenoss Core.

Preparing network devices

Preparing Switches and Routers

To prepare a switch or router device for monitoring, verify that an SNMP agent is installed and currently running on the device.

Note This rest of this section describes how to prepare Cisco network devices for monitoring. For other device types, refer to the [ZenPack catalog](#) documentation.

Preparing Cisco UCS Network Devices

Zenoss Core uses SNMP to provide customized or generalized support for many Cisco products.

The following table associates Cisco products with the customized Zenoss Core device types that support them. Device types are listed in the **Network** area of the **Add Infrastructure** wizard, which is both part of the setup wizard and available through the Zenoss Core browser interface.

Note Some supported devices, such as the Cisco Nexus 7000 and 9000 switches, represent a large number of discrete monitoring endpoints. If you are unsure which Zenoss Core deployment size supports the number of high-density devices you wish to monitor, contact your Zenoss representative.

Note In order to monitor Cisco Nexus 9000 Series devices, you must first enable NX-API with the `feature manager CLI` command on the device. For detailed instructions on performing this task, refer to the [Cisco documentation](#) for the Nexus 9000.

Cisco product	Device type
Cisco Catalyst 6500 and 3560 Series Switches	Cisco 6500 (SNMP)
Cisco Nexus 5000 Series Switches	Cisco Nexus 5000 (SNMP + Netconf)
Cisco Nexus 7000 Series Switches	Cisco Nexus 7000 (SNMP + Netconf)
Cisco Nexus 1000v Series Switches	Cisco Nexus 1000V (SNMP + Netconf)
Cisco Nexus 3000 Series Switches	Cisco Nexus 3000 (SNMP + Netconf)
Cisco Nexus 9000 Series Switches	Cisco Nexus 9000 (NX-API)
Cisco Catalyst 6500 Series Virtual Switching Systems	Cisco VSS (SNMP)
Cisco MDS 9000 Series Multilayer Switches	Cisco MDS 9000 (SNMP)

In addition, Zenoss Core provides two generalized device types.

Cisco product	Device type
Cisco CatOS-based switches or routers	Generic Switch/Router (SNMP)
Cisco IOS-based switches or routers	Cisco IOS (SNMP)

Preparing storage devices

Note This section describes how to prepare NetApp and EMC storage devices for monitoring. For other device types, refer to the [ZenPack catalog](#) documentation.

Legacy NetApp Filers

Zenoss Core uses SNMP to monitor legacy NetApp Filers that do not support the Data ONTAP® API (ZAPI).

Note The data gathered are approximate, because the values for many objects (Aggregate, Volume, Plex, and RAID group) are not exposed by the NetApp MIB.

To prepare a legacy NetApp Filer for monitoring, verify that SNMPv2 is installed, and then start an SNMP agent.

Recent NetApp Filers

Zenoss Core uses HTTP to monitor NetApp Filers that support the Data ONTAP® API (ZAPI).

To prepare a recent NetApp Filers for monitoring, verify the following conditions:

- The Filer is running in 7-Mode or C-Mode.
- ZAPI is installed and enabled. Version 8.x, or a more recent version, is required.

Also, you need the username and password of an account on the Filer that is authorized to use ZAPI.

EMC Storage Arrays

Zenoss Core uses the Web-Based Enterprise Management (WBEM) protocol to send queries to EMC Storage Management Initiative Specification (SMI-S) providers associated with EMC VMAX and VNX storage arrays.

To prepare EMC arrays for monitoring, verify that at least one EMC SMI-S provider is running for each type of array to monitor. (The VMAX and VNX data models are different.) In addition, you need the following information:

- The username and password of a user account that is authorized to collect data on each SMI-S provider.
- The IP address of each SMI-S provider.
- The port number at which each SMI-S provider listens for requests.
- Whether or not to use SSL.

Zenoss recommends verifying that an SMI-S provider is responding to requests before adding it to Zenoss Core.

Note Many of the graphs for components types of EMC arrays display NaN when statistics logging is disabled on the EMC device. The logging feature has a low default timeout value, and must be set to a higher value or turned on again periodically.

Verifying an SMI-S provider on EMC devices

To perform this procedure, you need a Linux host that has a network path to the SMI-S providers of the arrays to monitor.

Note Do not perform this procedure on the Zenoss Core host.

Perform this procedure to verify that the SMI-S providers associated with EMC arrays are configured correctly, and are responding to WBEM queries from command line tools.

- 1 Log in to a Linux host as `root`, or as a user with superuser privileges.
- 2 Install a WBEM command-line interface package, such as `wbemcli`.
- 3 Verify the SMI-S provider. Replace the variables with values that are valid in your environment.

```
wbemcli IP-Address:Port -u admin \  
-p 'Password' -n root/emc --no-sslei('EMC_DiskDrive')
```

The expected result is a list of Disk Drive classes.

Preparing server devices

Note This section describes how to prepare Linux and Windows servers for monitoring. For other device types, refer to the [ZenPack catalog](#) documentation.

Linux Servers

Zenoss Core uses SNMP or SSH to monitor Linux servers.

To prepare a Linux server for SNMP monitoring, install an SNMP package on the server (for example, [Net-SNMP](#)) and start the agent.

To prepare a Linux server for SSH monitoring, install an SSH server package (for example, [OpenSSH](#)) and start the SSH daemon. Also, obtain the username and password of a user account on the server that has standard user privileges (root privileges are not required).

Windows Servers

Zenoss Core uses SNMP or WinRM to monitor the following Microsoft Windows systems:

- Microsoft Windows Server 2012 and 2012 R2
- Microsoft Windows Server 2008 R2

To prepare a Windows system for SNMP monitoring, start the SNMP service.

To prepare a Windows system for WinRM monitoring, refer to the [support article](#) that describes the options and provides the procedures for configuring your systems.

Preparing hypervisor devices

Note This section describes how to prepare vSphere and Hyper-V hypervisors for monitoring. For other device types, refer to the [ZenPack catalog](#) documentation.

vSphere EndPoint

Zenoss Core uses SOAP to monitor VMware vSphere servers running vSphere 4.1, 5.0, 5.1, 5.5, or 6.0.

To prepare a VMware vSphere server for monitoring, verify the software version, and obtain the username and password of an account on the server that is authorized to use the vSphere API and determine whether or not to use SSL.

Hyper-V

Zenoss Core uses WinRM to monitor the following Microsoft Hyper-V systems:

- Microsoft Hyper-V Server 2012 and 2012 R2
- Microsoft Hyper-V Server 2008 and 2008 R2

To prepare a Hyper-V system for WinRM monitoring, refer to the [support article](#) that describes the options and provides the procedures for configuring your systems.

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 stop the `zenmodeler` daemon and run the modeling process once daily from a cron job.

Configuring Windows Devices to Provide Data Through SNMP

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. Go to <http://www.snmp-informant.com> and download SNMP for Windows.

To collect Windows event logs or log files from a Windows box using syslog, you can use the SyslogAgent Windows add-on, available from:

<http://syslogserver.com/syslogagent.html>

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 De-select 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.)

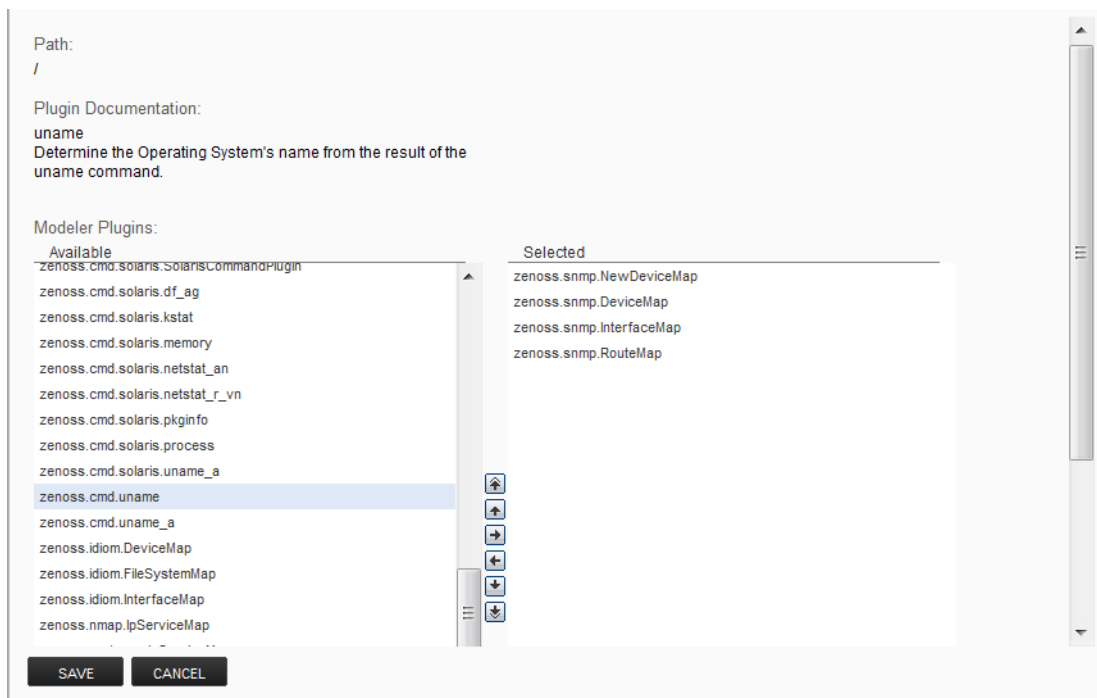
Note If using RSA keys for a device or device class, change the value of the `zKeyPath` configuration property to:

```
~/.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 2: 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.

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.

Modeling Devices Using Port Scan

You can model IP services by doing a port scan, using the Nmap Security Scanner (<http://nmap.org/>). 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 In order 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.

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.
The Device summary page appears.
- 2 In the left panel, 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 Login into 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.
- 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.

Next Steps

Zenoss Core is now configured to monitor your IT infrastructure. Your next steps in the monitoring process may include some of the following items:

- Customize the Dashboard
- Review events in the Events Console
- Organize devices and infrastructure in to logical groupings on the Infrastructure page
- View data collection graphs from the Infrastructure page
- Generate and review reports on the Reports page
- Refine data collection on the Advanced page

For more information on these and other procedures, refer to the *Zenoss Core Administration Guide*.

A

External HBase configuration

Zenoss Core may be configured to use an external HBase cluster, rather than the cluster that is included in the application.

Note If you do not already have an external HBase cluster, there is no need to create one. The procedures in this section are for customers who wish to use an existing HBase cluster for Zenoss Core data.

The version of HBase installed in your external HBase cluster must be compatible with the version of OpenTSDB used by the Zenoss Core application. For Zenoss Core versions 5.0.1 through 5.1.1, the required version of HBase is 0.92.

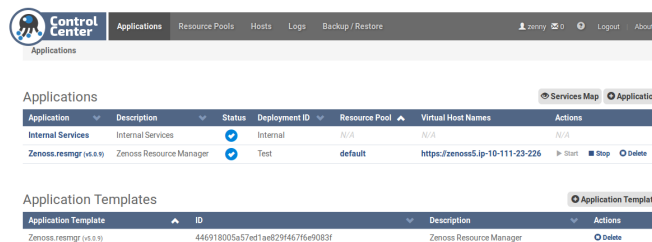
Perform the procedures in the following sections in order.

Configuring OpenTSDB for an external HBase cluster

To perform this procedure, install and start Zenoss Core.

This procedure configures OpenTSDB to use an external HBase cluster, rather than the HBase cluster that is included in the Zenoss Core application.

- 1 Log in to the Control Center browser interface.



- 2 In the **Applications** table, click **Zenoss.resmgr**.
- 3 Scroll down to the **Services** table, and then locate the OpenTSDB service.
Depending on your version of Control Center, the service is either **opentsdb** or two separate services, **reader** and **writer**.
- 4 Click **opentsdb**, or one of **reader** or **writer**.
- 5 On the service details page, scroll down to the **Configuration Files** table.

The screenshot shows the Zenoss Control Center interface for the 'reader' service. At the top, there are navigation tabs for Applications, Resource Pools, Hosts, Logs, Backup/Restore, and user information. Below the service name, there are buttons for Edit Service, Edit Variables, Start, Stop, and Restart. The 'Virtual Host Names' section contains a table with columns for Virtual Host Name, Service, Endpoint, URL, and Actions. The 'IP Assignments' section shows a table with columns for Service, Assignment Type, Host, Resource Pool, IP, and Actions, with a note that no data was found. The 'Configuration Files' section shows a table with columns for Path and Actions, listing the file /opt/zenoss/etc/opentsdb/opentsdb.conf.

- 6 In the **Actions** column of the **Configuration Files** table, click **Edit**.
- 7 In the **Edit Configuration** dialog, replace the value of the `tsd.storage.hbase.zk_quorum` key with the ZooKeeper quorum of the external HBase cluster.
 - a Delete the existing value.

The default value is a *Go language template* expression.
 - b Specify the ZooKeeper quorum of the external HBase cluster.

To specify a ZooKeeper quorum, create a comma-separated list of all quorum members. Specify each member of the quorum with a hostname or IP address, the colon character (:), and then the port number on which the ZooKeeper service is listening.

Note If you use hostnames, the Control Center master host must be able to resolve them to IPv4 addresses, either through a nameserver on the network or through entries in `/etc/hosts`.

The following example shows the correct syntax for a 3-member ZooKeeper quorum:

```
zk-1.example.com:2181, zk-2.example.com:2181, zk-3.example.com:2181
```

- 8 In the **Edit Configuration** dialog, click **Save**.
- 9 At the top of the page, click the **Stop** button, and then click the **Start** button.
- 10 If your version of Zenoss Core includes two OpenTSDB services (**reader** and **writer**) repeat the preceding steps for the other service.

Configuring the OpenTSDB service startup command

This procedure configures the OpenTSDB service to use the external HBase cluster on startup.

- 1 Log in to the Control Center browser interface.
- 2 In the **Applications** table, click **Zenoss.resmgr**.
- 3 Scroll down to the **Services** table, and then locate the OpenTSDB service.

Depending on your version of Control Center, the service is either **opentsdb** or two separate services, **reader** and **writer**.
- 4 Click **opentsdb**, or one of **reader** or **writer**.
- 5 On the service details page, click the **Edit Service** link at the top of the page.

This screenshot is identical to the one above, showing the Zenoss Control Center interface for the 'reader' service configuration page.

- 6 In the **Edit Service** dialog, change the value of the **Startup Command** field.

- a Delete the *Go language template* expression.

The expression is everything after `start-opentsdb.sh`.

- b Specify the ZooKeeper quorum of the external HBase cluster.

To specify a ZooKeeper quorum, create a comma-separated list of all quorum members. Specify each member of the quorum with a hostname or IP address, the colon character (:), and then the port number on which the ZooKeeper service is listening.

Note If you use hostnames, the Control Center master host must be able to resolve them to IPv4 addresses, either through a nameserver on the network or through entries in `/etc/hosts`.

The following example shows the correct syntax for a 3-member ZooKeeper quorum:

```
zk-1.example.com:2181, zk-2.example.com:2181, zk-3.example.com:2181
```

The result should include at least one empty space between `start-opentsdb.sh` and the ZooKeeper quorum list.

- 7 In the **Edit Service** dialog, click **Save Changes**.
- 8 At the top of the page, click the **Stop** button, and then click the **Start** button.
- 9 If your version of Zenoss Core includes two OpenTSDB services (**reader** and **writer**) repeat the preceding steps for the other service.

Disabling the Zenoss Core HBase cluster

This procedure disables the HBase cluster that is included in the Zenoss Core application.

- 1 Log in to the Control Center master host as `root`, or as a user with superuser privileges.
- 2 Stop the Zenoss Core HBase cluster.

```
serviced service stop HBase
```

- 3 Disable automatic start of the HBase services.

- a Change the configuration of each service.

```
for svc in hmaster regionserver zookeeper
do
    serviced service list $svc | \
    sed -e 's/"Launch": "auto"/"Launch": "manual"/' | \
    serviced service edit $svc
done
```

The `serviced` command displays the new configuration after each edit.

- b Verify that each service is set to manual start.

```
for svc in hmaster regionserver zookeeper
do
    serviced service list $svc | egrep '"Launch":'
done
```

- 4 Remove the OpenTSDB prerequisite for the Zenoss Core HBase cluster.

Depending on your version of Control Center, the OpenTSDB service is either `opentsdb` or two separate services, `reader` and `writer`.

- a Edit `opentsdb`, or one of `reader` or `writer`.

```
serviced service edit reader
```

The `serviced` command opens the service's configuration in the default text editor.

- b** Locate the `Prereqs` section, and then remove everything between the left square bracket (`[`) and the right square bracket (`]`) characters.

The following lines show an example `Prereqs` section:

```
"Prereqs": [  
  {  
    "Name": "HBase Regionserver up",  
    "Script": "{{with $rss := (child (child (parent) \"HBase  
\"))}.Instances }}"  
  }  
],
```

After editing, the section should look like the following example:

```
"Prereqs": [],
```

- c** Save the file, and then exit the text editor.
- d** If your version of Zenoss Core includes two OpenTSDB services (**reader** and **writer**) repeat the preceding substeps for the other service.