Zenoss®

Transforming IT Operations

Zenoss Control Center

Release 1.0.0

Zenoss, Inc.

www.zenoss.com

Zenoss Control Center

Part Number: 29.14.xxx

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

Preface

Zenoss Control Center provides...

Additional information and comments

If you have technical questions about this product that are not answered in this guide, visit the *Zenoss Support* site.

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.

Configuration and maintenance 1

Zenoss Control Center defaults

Zenoss Control Center is a single binary that behaves differently based on the variables defined in `/etc/default/serviced`.

HOME

Default: `/root`

The path Zenoss Control Center uses to locate the `.dockercfg` file to which it refers.

SERVICED_AGENT

Default: `0`

One of two variables that determine whether an instance assumes the role of agent or master. The other variable is SERVICED_MASTER.

- To configure an instance as an agent:

```
SERVICED_AGENT=1
SERVICED_MASTER=0
```

- To configure an instance as the master:

```
SERVICED_AGENT=1
SERVICED_MASTER=1
```

SERVICED_MASTER

Default: `0`

One of two variables that determine whether an instance assumes the role of agent or master. The other variable is SERVICED_AGENT.

SERVICED_MASTER_POOLID

Default: `default`

The name of the resource pool in which the instance configured as the master is located.

SERVICED_MASTER_IP

Default: `172.17.42.1`

The IP address of the instance configured as master. This variable simplifies the configuration of multi-host deployments.

SERVICED_ZK

Default: `$SERVICED_MASTER_IP:2181`

The list of endpoints in the *ZooKeeper* ensemble of Zenoss Control Center, separated by the comma character (`,`). Currently, the instance configured as the master is the only host in the ZooKeeper ensemble.

SERVICED_REGISTRY

Default: `1`

Determines whether Docker uses a local registry to store images.

SERVICED_DOCKER_REGISTRY

Default: `$SERVICED_MASTER_IP:5000`

The IP address and port number of the Docker registry host.

SERVICED_STATIC_IPS

Default: `10.0.0.30,10.0.0.31`

The list of static IP addresses reserved for Zenoss Control Center to use when it advertises external IP addresses on behalf of services it manages, separated by the comma character (`,`).

SERVICED_ENDPOINT

Default: `$SERVICED_MASTER_IP:4979`

The IP address and port number of the default Zenoss Control Center RPC endpoint.

SERVICED_RPC_PORT

Default: `4979`

The port on which an instance listens for RPC requests.

SERVICED_UI_PORT

Default: `443`

The port on which an instance listens for HTTPS requests. (The port for the Zenoss Control Center web interface.)

SERVICED_MUX_PORT

Default: `22250`

The port an instance uses for multiplexing on its private subnet.

SERVICED_VARPATH

Default: `/opt/serviced/var`

The path of the local directory in which an instance stores its data files.

SERVICED_KEY_FILE

Default: `/etc/....`

The path of a TSL key file. By default, no key file is installed.

SERVICED_CERT_FILE

Default: `/etc/....`

The path of a TLS certificate file. By default, no certificate file is installed.

SERVICED_VFS

Default: `rsync`

The driver for virtual file system volumes. The supported drivers are `rsync` and `btrfs`.

SERVICED_VHOST_ALIASES

Default: `foobar.com,example.com`

The list of virtual host aliases to use in virtual host multiplexing, separated by the comma character (`,`).

SERVICED_MAX_CONTAINER_AGE

Default: `60`

The maximum number of days the instance waits before removing a stopped container.

SERVICED_VIRTUAL_ADDRESS_SUBNET

Default: `10.3`

The 16-bit private subnet to use for virtual IPv4 addresses.

Note This value affects the values of SERVICED_STATIC_IPS.

SERVICED_LOG_LEVEL

Default: `0`

The relative amount of information to include in the `/var/log/upstart/serviced.log` file. The range is 0 (minimum) to 5 (maximum).

SERVICED_LOG_ADDRESS

Default: `$SERVICED_MASTER_IP:5042`

The endpoint of the *logstash* service.

SERVICED_STATS_PORT

Default: `$SERVICED_MASTER_IP:8443`

The endpoint of the OpenTSDB reader for Zenoss Control Center statistics.

SERVICED_STATS_PERIOD

Default: `10`

The number of seconds to wait between polls of hosts in resource pools.

SERVICED_OPTS

Default: (none)

Arbitrary options for the `serviced` daemon startup command of an instance.

Defining services 2

Service template

A service template contains service definitions, in JavaScript Object Notation (JSON) format.

Atttribute	Value	Description
`ID`	String	The identifier of this template.
`Name`	String	The name of this template.
`Description`	String	The description of this tempalte.
`Services`	Array of objects	One or more *ServiceDefinition* objects.
`ConfigFile`	Object	The configuration file for this service template (a *ConfigFile* object).

`ServiceDefinition` object

The information that Zenoss Control Center needs to start and manage a service, in JavaScript Object Notation (JSON) format.

Atttribute	Value	Description
`Name`	String	The service name.
`Command`	String	The command that starts and runs the service.
`Description`	String	The service description.
`Tags`	Array of strings	Searchable words or phrases for the service.
`ImageID`	String	The Docker image ID for this service.
`Instances`	Object	The number of instances of the service to run at the same time, in different Docker containers. This object has two members. **`Min`** The minimum number of instances to run. If the value is zero, no instances are started. If non-zero, the value must be greater than or equal to zero.

Atttribute	Value	Description
		Max The maximum number of instances to run. If the value is zero, only Min is used. If non-zero, the value must be greater than or equal to Min.
ChangeOptions	Array of strings	Control options for what happens when a running instance of the service is changed.
Launch	String	Determines whether the service starts when its parent service starts. **AUTO** (Default) Start the service when the parent service starts. **MANUAL** Do not start the service when the parent service starts.
HostPolicy	String	Determines how to schedule instances of the service on hosts in the resource pool. **LeastCommitted** (Default) Select the host with the smallest amount of memory committed to other services. **PreferSeparate** Attempt to schedule instances of the service on separate hosts. **RequireSeparate** Schedule instances of the service on separate hosts only.
Hostname	String	The virtual host name to use when an instance of the service is run.
Privileged	true or false	Determines whether Docker containers (in which instances of the service run) have extended privileges.
ConfigFiles	Objects	The configuration file templates to install in Docker containers (in which instances of the service run). Each member of this object is a string containing the absolute path of a configuration file template and a *ConfigFile* object.
Context	Object	
Endpoints	Array of objects	The network endpoints that the service uses, as *Endpoint* objects.
Services	Array of objects	The subservices that are part of the service, as *ServiceDefinition* objects.
Tasks	Array of objects	
LogFilters	Object	Pairs of strings and values mapping log filter names to log filter definitions.
Volumes	Array of objects	The list of file system directories to bind mount in Docker containers (in which instances of the service run). Each item is a *Volume* object.
LogConfigs	Array of objects	A list of logstash configurations for the log files of the service, as *LogConfig* objects.

Atttribute	Value	Description
Snapshot	Object	The bash commands to pause and resume the service. Between commands, Zenoss Control Center makes a copy of the container's file system. This object has two members: **Pause** (string) The bash command to stop file system writes, or stop the service. **Resume** (string) The bash command to return the service to normal processing.
RAMCommitment	Integer	The amount of main memory the service requires, in bytes. This value is used to help schedule the service into the best available resource pool.
CPUCommitment	Integer	The number of CPU cores the service requires. This value is used to help schedule the service into the best available resource pool.
Runs	Object	The commands the service executes inside its container when invoked with serviced run *Command*. This object contains one or more command names, each paired with the bash command that is run inside the container.
Actions	Object	The commands the service executes inside its container when invoked with serviced action *Command*. This object contains one or more command names, each paired with the bash command that is run inside the container.
HealthChecks	Object	The commands to invoke at regular intervals to gather data about health of the service. Each member of this object is a health check name paired with a *HealthCheck* object.
Prereqs	Array of objects	A list of scripts to run successfully inside a container before starting the service in the container. Each item is an object with two members: **Name** (string) The name of the prerequisite. **Script** (string) The bash command to invoke inside the container.
MonitoringProfile	Object	Metadata descriptions of the metrics, graphs, and thresholds this service makes available to external monitoring services. The members of this object are the following pairs, in order: **MetricConfigs** An array of *MetricConfig* objects. **GraphConfigs** An array of *GraphConfig* objects. **ThresholdConfigs** An array of *ThresholdConfig* objects.
MemoryLimit	float64	
CPUShares	int64	

Atttribute	Value	Description
`PIDFile`	String	An optional path or command to generate a path for a PID file to which signals are relayed

`ConfigFile` object

Atttribute	Value	Description
`Name`	String	The service name.
`Filename`	String	The absolute path of the configuration file.
`Owner`	String	The user and group of the configuration file's owner inside Docker containers, in a format accepted by the `chown` command. For example, for a file owned by user and group `root`, `root:root` or `0:0`.
`Permissions`	String	The file mode bits for the configuration file inside Docker containers, in a format accepted by the `chmod` command. For example, for read, write, and execute permission for the owner and read permission for group and others, `go+r,u+rwx` or `0744`.
`Content`	String	The content of the configuration file template, in a single JSON string.

`Endpoint` object

Atttribute	Value	Description
`Name`	String	The human-readable name of the endpoint. This value must be unique in its parent service definition.
`Purpose`	String	
`Protocol`	String	The networking protocol to use, `tcp` or `udp`.
`PortNumber`	Integer	The port number of the endpoint.
`PortTemplate`	String	???
`VirtualAddress`	String	An address by which an imported endpoint may be accessed within its container. For example, `mysqlhost:1336`.
`Application`	String	
`ApplicationTemplate`	String	Deprecated. `Application` accepts templates.
`AddressConfig`	Object	The definition of an external-facing port. This object has two members: **`Port`** The port number of the endpoint. **`Protocol`** The networking protocol to use, `tcp` or `udp`.
`VHosts`	Array of strings	Reserve a virtual hostname for the endpoint. The string or strings are used as subdomain names. A virtual hostname allows users to find a service quickly, without manually searching for its IP address.

Task object

Atttribute	Value	Description
Name	String	The name of the task.
Schedule	String	???
Command	String	???
LastRunAt	time	timestamp
TotalRunCount	int	???

Volume object

A Volume object defines a file system directory that is exported by Docker containers.

Atttribute	Value	Description
Owner	String	The user and group of the directory's owner inside Docker containers, in a format accepted by the chown command. For example, for a file owned by user and group root, root:root or 0:0.
Permission	String	The file mode bits for the directory inside Docker containers, in a format accepted by the chmod command. For example, for read, write, and execute permission for the owner and read permission for group and others, go+r,u+rwx or 0744.
ResourcePath	String	Resource Pool Path, shared across all hosts in a resource pool
ContainerPath	String	The absolute path of the bind mount directory inside Docker containers.
Type	String	The use of the path; for example, dfs or tmp.

LogConfig object

Atttribute	Value	Description
Path	String	The absolute path of a log file on a Docker container's file system. The path can be a directory or a file, and bash pathname expansion (globbing) is supported.
Type	String	A string that identifies the "types" of logs that come from this source. This will be
Filters	Array of strings	A list of filters to apply to the log file. The filters listed here must be defined in the LogFilters object of the current service, or a parent service.
LogTags	Array of objects	A list of tags to send to logstash with all log file entries. Each object has two members: **Name (string)** The name of the tag to send. **Value (integer)** The value of the tag to send, which may be expressed as a Zenoss Control Center template.

`HealthCheck` object

Atttribute	Value	Description
`Script`	String	The `bash` command to invoke inside the service's container, to gather data about the health of the service.
`Interval`	Float	The number of seconds to wait between incovations of the data-gathering command.

`MetricConfig` object

Atttribute	Value	Description
`ID`	String	The unique identifier of the metrics.
`Name`	String	The canonical name of the metrics.
`Description`	String	The description of the metrics.
`Query`	Object	The components of an HTTP query to request metrics. This object includes the following pairs: **`RequestURI`** The HTTP request URI. **`Method`** The HTTP method to use. **`Headers`** The HTTP headers require to make a request. **`Data`** The HTTP request body.
`Metrics`	Array of objects	One or more *Metric* objects.

`Metric` object

The metadata description of a single metric.

Atttribute	Value	Description
`ID`	String	The unique identifier of the metric
`Name`	String	The canonical name of the metric.
`Description`	String	The description of this metric.
`Counter`	`true` or `false`	Determines whether this metric is a counter.
`CounterMax`	Integer	The maximum value this metric may provide, if it is a counter.
`ResetValue`	Integer	
`Unit`	String	The type of unit this metric provides; for example, `seconds`.

GraphConfig object

Atttribute	Value	Description
ID	String	The unique identifier of the graph.
Name	String	The canonical name of the graph.
Footer	true or false	Determines whether the graph includes a footer.
Format	String	The format specification for graph data, as defined in the *Go fmt package*.
ReturnSet	String	
Type	String	
Tags		
MinY	Integer	
MaxY	Integer	
YAxisLabel	String	
TimeZone	String	
DownSample	String	
Description	String	
Range	Object	This object has two members: **Start** The minimum value for the range. **End** The maximum value for the range.
DataPoints	Array of objects	A list of *DataPoint* objects.

DataPoint object

Atttribute	Value	Description
Aggregator	String	
Color	String	
Expression	String	
Fill	true or false	
Format	String	
Legend	String	
Metric	String	
MetricSource	String	
ID	String	
Name	String	

Atttribute	Value	Description
`Rate`	`true` or `false`	

`RateOptions`	Object	This object has 3 members:
		`Counter`
		(`true` or `false`)
		`CounterMax`
		`ResetThreshold`

`Type`	String	

`ThresholdConfig` object

Atttribute	Value	Description
`ID`	String	
`Name`	String	
`Type`	String	type of threshold (MinMax, Duration, ValueChange, or HoltWinters)
`Description`	String	
`AppliedTo`	Integer	how should this threshold be applied 0=everything, 1=services only, 2=running services only
`MetricSource`	Arrays of strings	
`DataPoints`	String	
`Threshold`		
`EventTags`		

Text templates in service definitions

Some service definition values may be expressed in *Go text template* syntax with Zenoss Control Center functions and contexts.

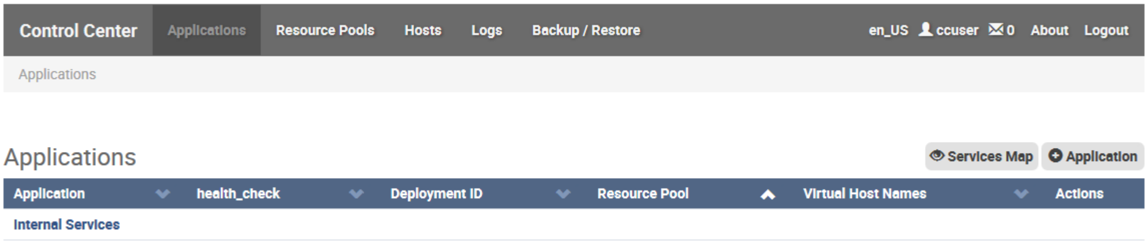
User interface reference 3

Initial page

The first time the Zenoss Control Center interface is started, the **Applications** page includes only **Internal Services** in the **Applications** table.

The menu bar is at the top of each page. Click an item in the menu bar to display the page or dialog box associated with the item, or to log out of the Zenoss Control Center interface.

Figure 1: Initial **Applications** page

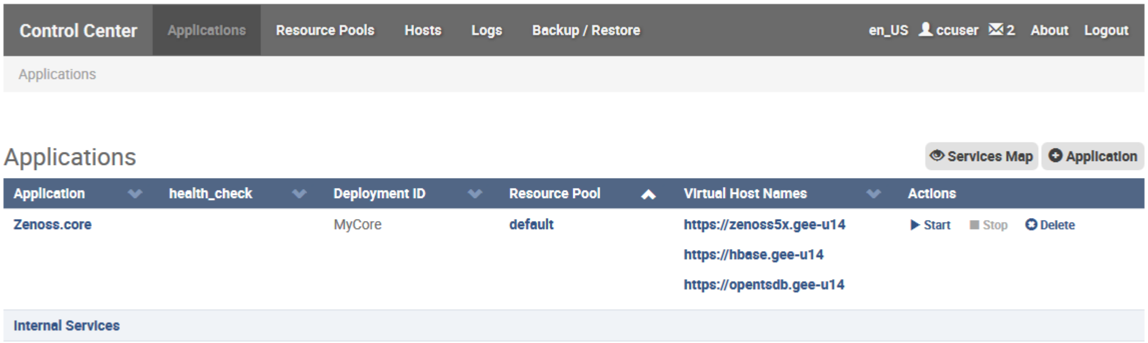


Application	health_check	Deployment ID	Resource Pool	Virtual Host Names	Actions
Internal Services					

Applications page

The **Applications** page displays upon login, and features a table listing both deployed applications and the services that support Zenoss Control Center.

Figure 2: Applications page



Application	health_check	Deployment ID	Resource Pool	Virtual Host Names	Actions
Zenoss.core		MyCore	default	https://zenoss5x.gee-u14 https://hbase.gee-u14 https://opentsdb.gee-u14	▶ Start ■ Stop Delete
Internal Services					

Resource Pools page

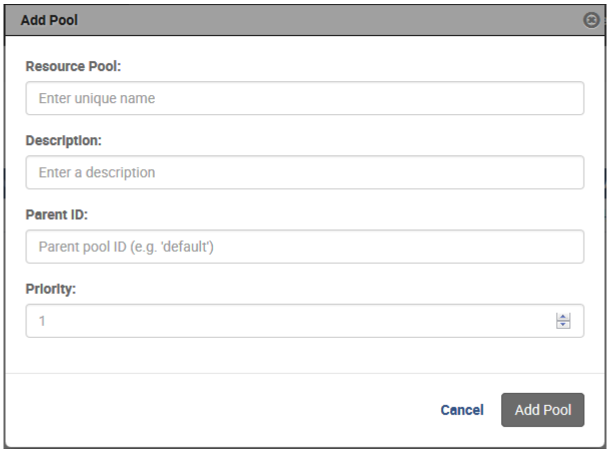
The **Resource Pools** page lists the resource pools Zenoss Control Center is employing, with aggregate information about the hosts in each pool.

The **Resource Pools** table shows the total resources available in each pool.

The **Resource Pool** button, located immediately above the right side of the table, displays the **Add Pool** dialog.

Add Pool dialog

The **Add Pool** dialog creates a new resource pool.



Add Pool

Resource Pool:

Enter unique name

Description:

Enter a description

Parent ID:

Parent pool ID (e.g. 'default')

Priority:

1

Cancel Add Pool

Field	Type	Description
Resource Pool	Required	The unique name of the pool. UTF-8 characters are accepted. Leading and trailing spaces are ignored.
Description	Optional	A description of the pool.
Parent ID	Optional	The identifier of the parent resource pool, if this pool is embedded.
Priority	Optional	The priority of the pool relative to other pools, as a postive integer. This value is used for CPU priority.

Hosts page

The **Hosts** page displays information about the individual hosts that are included in all of the resource pools that Zenoss Control Center is employing.

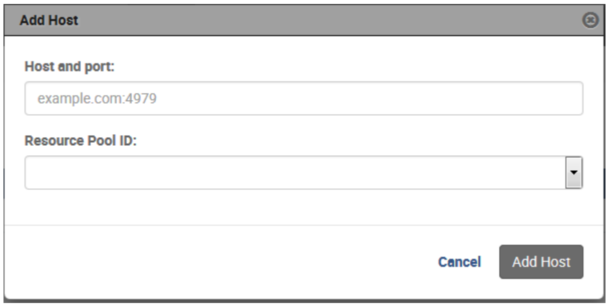
The **Hosts** table lists each machine that is assigned to a resource pool.

Two buttons are located immediately above the right side of the table.

- The **Hosts Map** button displays a graphical representation of the relative sizes of the hosts in the table, either by the amount of RAM or the number of CPUs in each host.
- The **Host** button displays the **Add Host** dialog.

Add Host dialog

The **Add Host** dialog adds machines to resource pools.



Field	Type	Description
Host and port	Required	The hostname or IP address of the machine to add, followed by a colon character (`:`), and then the port number Zenoss Control Center uses to communicate among hosts. The default port is 4979.
Resource Pool ID	Required	The identifier of the resource pool for the machine to add. Enter the identifier or select it from the list.

Logs page

The **Logs** page displays the log information that Zenoss Control Center collects.

The log information is presented in *the Kibana JavaScript interface*, which provides a variety of methods for sorting, querying, and analyzing log data.