electric CLOUD

ElectricAccelerator cmtool Reference Guide

for version 7.1

Electric Cloud, Inc.
676 W. Maude Avenue
Sunnyvale, CA 94085
www.electric-cloud.com

Document Rev. 1

Contents

Chapter 1: Overview

cmtool is the ElectricAccelerator command-line tool. **cmtool** provides access to the Cluster Manager through a command-line interface instead of using the web interface. With cmtool, you can write Perl scripts to access Cluster Manager information or manage builds. Almost all ElectricAccelerator® operations and tasks can be implemented with cmtool—with the exception of a few reports that are generated only from the web interface.

cmtool is used primarily for build and agent management, including commands for build class management, agent testing, and adding comments automatically.

Topics:

Logging In

If you use cmtool outside of a job, you *must* invoke the ***cmtool login*** command to log in to the server. After logging in, cmtool saves information about the login session for use in future cmtool invocations. If you run cmtool as part of an ElectricAccelerator job, you do not need to log in because `--cmtool` uses the login session (and credentials) for that job.

To Log In to cmtool:

```
cmtool login <username> <password>
```

To specify a session file, use the `--sessionFile=<fileName>` option, so you can use the same session for subsequent cmtool invocations.

Using cmtool

An invocation of cmtool identifies the Cluster Manager to contact, using the `--server` command-line option, followed by a list of commands to execute. Certain commands may have optional or required arguments.

For example, the following invocation receives all build requests that ran fewer than 10 jobs and orders the list [that ran the build] by host name.

```
cmtool --server easerver getBuilds-filter "job_count <10" --order host_name
```

General syntax for cmtool command usage:

```
cmtool [optional global argument(s)] <command> <required arguments> [optional
arguments]
```

Return Codes

0 = success (the command was correct; if no data meets the criteria, return is still 0)
1 = failure (command was invalid)

Using runAgentCmd

IMPORTANT: Exercise caution when using the `runAgentCmd` command. Electric Cloud recommends using this command for documented scenarios only or under the direction of Electric Cloud Technical Support.

The `runAgentCmd` command enables you to run agent commands against the cluster.

Use this format: `cmtool --cm=<cm> runAgentCmd "agent command to run"`)

where <cm> is the IP address or name of your Cluster Manager.

Some of the possible reasons for using `runAgentCmd` include:

- Setting agent-side breakpoints (see the Using Breakpoints topic in online help)
- Configuring agent log rotation (see the Installation and Configuration Guide)
- Getting and setting agent and EFS debug levels (Knowledge Base article KBEA-00020)
- Configuring the stalled job killer (Knowledge Base article KBEA-00031)
- Troubleshooting builds that appear to hang (Knowledge Base article KBEA-00036)

Global Arguments (Optional)

Global arguments supply general information quickly, including cmtool online help.

Note: Global arguments support using the “=” sign character.

- `--help [`*`command`*`]` - Prints this message and exits. If a command is specified, prints the help text for that command.
- `--help-commands` - Prints the list of available commands with a short description.
- `--help-fields <`*`command`*`>` - Displays a list of fields for a command—requires the <*command*> argument.
- `--version` - Prints cmtool version number.
- `--server <`*`hostname`*`>` - ElectricAccelerator server address. Defaults to the ACCELERATOR_SERVER environment variable. If this variable does not exit, default is to the localhost.
- `--port <`*`port`*`>` - HTTP listener port on the ElectricAccelerator server. Defaults to port 8030.
- `--securePort <`*`securePort`*`>` - HTTPS listener port on the ElectricAccelerator server. Defaults to port 8031.
- `--secure` - Uses HTTPS to communicate with the ElectricAccelerator server.
- `--timeout <`*`s`*`>` - cmtool waits for a response from the server for a specified amount of time. Timeout for server communication defaults to 180 seconds (3 minutes) if no other time is specified. After the timeout, cmtool exits but the server will continue to process the command.
- `--output <`*`style`*`>` - Set output style—default is 'xml'. ‘xml’ for an XML document; ‘csv’ for comma separated values; ‘simple’ for no formatting; ‘silent’ for no output
- `--fields <`*`list`*`>` - *List* is a comma separated list of fields to emit when using ‘csv’ or ‘simple’ output styles. Default is *all* fields.
- `--sessionFile <`*`path`*`>` - Overrides the location where session information will be stored.

Chapter 2: Agent Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

changeAgentsEnabled

Description

Changes the agent enabled status of one or more agents.

Required Arguments

- `<enabled>` - Possible values are `true` or `false`.

Optional Arguments

Note: If no agent name, agent ID, or filter is specified, all agents are changed.

- `--agentId` *`<unique, internal number that can change; assigned by the Cluster Manager>`*
- `--agentName` *`<name defined by the host where the agent resides [numbers and/or letters]>`*
- `--filter` *`<SQL query used to limit the result set>`* - For a list of possible SQL values, see the `getAgents` command.

  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

Examples

```
cmtool changeAgentsEnabled false
```

Disables all agents in the cluster.

```
cmtool changeAgentsEnabled true --agentName linagent1
```

Enables the agent named “linagent1”.

```
cmtool changeAgentsEnabled true --filter "agent_name LIKE 'winbuild1-%'"
```

Enables all agents with a name that begins with “winbuild1-”.

createAgentComment

Description

Creates a new agent comment.

Required Arguments

Note: Either `agentId` or `agentName` must also be specified.

- *`<text>`* - The text of the item.

Optional Arguments

- `--agentId` *`<unique, internal number that can change; assigned by the Cluster Manager>`*

- `--agentName` *<name defined by the host where the agent resides [numbers and/or letters]>*

Example

```
cmtool createAgentComment --agentName linagent "Agent has been running great"
```

Creates a comment for an agent named “linagent”.

createResource

Description

Creates a new resource definition. After creating a resource, ensure the server is configured to support resource management. You can use the `modifyServer` command to enable resource management.

Required Arguments

- *<resourceName>* - This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class. It is used in the `ea_resource` table and also matches the resource requirement string for eMake.
- *<hostMasks>* - This is a quote-enclosed, semi-colon delimited list of host name masks, used to identify the list of hosts that support a resource. “*” is the wildcard character.

Optional Arguments

- `--description` *<a quote-enclosed text description for your reference only>*

Example

```
cmtool createResource R29 "rs*; rt*" --description "rs or rt hosts"
```

Creates a new resource named R29 that only uses hosts whose names start with ’rs’ or ’rt’.

createResourceComment

Description

Creates a new resource comment.

Required Arguments

- *<resourceId>* - A unique number that identifies each resource.
- *<text>* - The text of the item.

Optional Arguments

None

Example

```
cmtool createResourceComment 2 "This resource identifies production servers"
```

Creates a comment for resource 2.

deleteAgentComment

Description

Deletes an agent comment.

Required Arguments

Note: Either `agentId` or `agentName` must also be specified.

- *`<commentId>`* - The unique key that identifies a comment.
  Use `getAgentComments` to get a list of `commentId` numbers.

Optional Arguments

- `--agentId` *`<unique, internal number that can change; assigned by the Cluster Manager>`*
- `--agentName` *`<name defined by the host where the agent resides [numbers and/or letters]>`*

Example

```
cmtool deleteAgentComment 1008 --agentId 14
```

Deletes comment 1008 from agent 14 (14 is the Cluster Manager internal ID for the agent). To find out what the appropriate comment ID is, use the `getAgentComments` command, which will list the comments attached to a particular agent.

deleteAgents

Description

Deletes one or more agents, including all dependent records.

Required Arguments

None

Optional Arguments

- `--agentId` *`<unique, internal number that can change; assigned by the Cluster Manager>`*
- `--agentName` *`<name defined by the host where the agent resides [numbers and/or letters]>`*
- `--filter` *`<SQL query used to limit the result set>`* - For a list of possible SQL values, see the `getAgents` command.

Example

```
cmtool deleteAgents --agentName winbuild1
```

Deletes agent “winbuild1” and all associated comments.

deleteResource

Description

Deletes a resource definition.

Required Arguments

- `<resourceId>` - A unique number that identifies each resource.
  Use the `getResources` command to get a list of resource IDs.

Optional Arguments

None

Example

```
cmtool deleteResource 3
```

Deletes the resource definition for resource 3.

deleteResources

Description

Deletes multiple resource definitions.

Required Arguments

None

Optional Arguments

- `--filter <SQL query used to limit the result set>` - For a list of possible SQL values, see the `getResources` command.

Example

```
cmtool deleteResources
```

Deletes all resource definitions.

deleteResourceComment

Description

Deletes a resource comment.

Required Arguments

- `<resourceId>` - A unique number that identifies each resource.
- `<commentId>` - The unique key that identifies a comment.
  Use the `getResourceComments` command to get a list of comment IDs.

Optional Arguments

None

Example

```
cmtool deleteResourceComment 3 49
```

Deletes comment 49 from resource 3.

getAgentComments

Description

Retrieves a list of related agent comments, or a specific comment (by using the `--commentId` option).

Required Arguments

None

Optional Arguments

- `--agentId` *<unique, internal number that can change; assigned by the Cluster Manager>*
- `--agentName` *<name defined by the host where the agent resides [numbers and/or letters]>*

Result Tags

- `commentId` - The unique key that identifies a comment.
- `createTime` - The time when the item was created.
- `lastModifiedBy` - The user who last modified the item.
- `modifyTime` - The time when the item was last modified.
- `text` - The text of the item.

Example

```
cmtool getAgentComments --agentName ahost-3
```

Retrieves all comments for agent “ahost-3”.

getAgentPerformance

Description

Retrieves the performance log of one or more agents.

Required Arguments

None

Optional Arguments

- `--agentId` *<unique, internal number that can change; assigned by the Cluster Manager>*
- `--agentName` *<name defined by the host where the agent resides [numbers and/or letters]>*
- `--agents` *<a list of agents whose performance you want to see>*

- `--buildId` *<this further restricts the returned agents to those running a specific build ID>*
- `--status <1|0>` - Choose active or inactive agents.
- `--enabled <1|0>` - Choose enabled or disabled agents only.

Result Tags

- `agentName` - This is the name of the agent as it appears on the web page (product UI).
- `result` - This is the performance information of the agent.

Example

```
cmtool getAgentPerformance --agentName SOL1-1
```

Returns the performance log of the agent named “SOL1-1”.

getAgents

Description

Retrieves a list of agents.

Required Arguments

None

Optional Arguments

- `--agentId` *<unique, internal number that can change; assigned by the Cluster Manager>*
- `--agentName` *<name defined by the host where the agent resides [numbers and/or letters]>*
- `--filter` *<SQL query used to limit the result set>* - See possible values in the table below.
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--maxResults` *<maximum number of elements to run from a query>*
- `--firstResult` *<starting index for the query result set>*
  Note: `--firstResult` takes values beginning with `0`. A negative value indicates a record starting from the end of the set, counting backwards, so `-1` is the last record, `-2` is the next to last, and so on.
- `--order` *<SQL order by clause>* - Used to specify ordering for the query result set.
- `--profile <details|info>` - This is the level of detail to return from a query; `details` gets all information and `info` gets a reduced information set.

Result Tags and SQL Query Names

Tag	Description	SQL Query Name for `--filter` and `--order`

`a2aPort`	The agent to agent protocol communication port.	`a2a_port`
`agentId`	A unique, internal number assigned to each agent by the Cluster Manager; this number can change.	`id`
`agentName`	A name defined by the host where the agent resides [numbers and/or letters].	`agent_name`
`agentVersion`	The agent version string.	`agent_version`
`availableResults`	This is a count of 'max' or 'first' results if `--maxResults` or `--firstResult` is specified.	N/A
`buildId`	A unique number assigned by the Cluster Manager for each build.	`current_build_id`
`buildName`	The build name that is the expanded build class tag.	N/A
`consolePort`	The agent console port.	`console_port`
`efsVersion`	The EFS version string.	`efs_version`
`enabled`	The flag indicating if an agent is enabled or not.	`enabled`
`errorCount`	The number of internal agent errors.	`error_count`
`hostName`	The name of the machine where Electric Make was invoked.	`host_name`
`inPenaltyBox`	A flag indicating Electric Make had a recent problem with this agent.	N/A
`ipAddress`	The agent IP address.	`ip_address`
`lastErrorTime`	The last time the agent experienced an error.	`last_error_time`
`lastPingTime`	The last time the agent was pinged to determine its status.	`last_ping_time`
`platform`	The operating system being used or supported. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.	`platform`

`port`	The agent protocol communication port.	`port`
`restartCount`	The number of agent restarts.	`restart_count`
`status`	The agent status. 1= OK, but anything else is an error code.	`status`
`statusDetail`	If the last status update resulted in an error, it contains the error string (or the "OK" string if no error occurred).	`status_detail`
`webPort`	The agent web server port.	`web_port`

Example

```
cmtool getAgents --filter "agent_name like '%SOL%'"
```

Retrieves a list of all agents whose names start with "SOL".

getAgentStatus

Description

Retrieves the state of one or more agents. By default, only active agents are returned. Use `--status 0` to list inactive agents.

Required Arguments

None

Optional Arguments

- `--agentId` *<unique, internal number that can change; assigned by the Cluster Manager>*
- `--agentName` *<name defined by the host where the agent resides [numbers and/or letters]>*
- `--agents` *<this can be a list of agents whose status you want to see>*
- `--buildId` *<this further restricts the returned agents to those running a specific build ID>*
- `--status <1|0>` - Choose active or inactive agents.
- `--enabled <1|0>` - Choose enabled or disabled agents only.

Result Tags

- `agentName` - This is the name of the agent as it appears on the web page (product UI).
- `result` - This is the text string that describes the current state of the agent.

Example

```
cmtool getAgentStatus --agentName SOL1-1
```

Returns the status of the agent named "SOL1-1".

getLsfInformation

Description

Retrieves current information about the LSF interface.

Note: LSF must be enabled to retrieve information.

Required Arguments

None

Optional Arguments

None

Result Tags

- `clusterName` - The name of the LSF grid cluster.
- `lsfAvailable` - “1” if LSF is available to the Cluster Manager.
- `masterName` - The LSF Master Host name.
- `numPendingAgentJobs` - The number of LSF jobs submitted by the Cluster Manager that are waiting to run.
- `numRunningAgentJobs` - The number of LSF jobs Cluster Manager submitted that are running now.
- `statusMessage` - A message.

Example

```
cmtool --output csv --fields getLsfInformation
```

Retrieves a Boolean value to indicate whether LSF is available or not.

getLsfJobs

Description

Retrieves information about all jobs submitted to LSF.

Required Arguments

None

Optional Arguments

None

Result Tags

- `agentHostName` - The machine name where the agent is running.
- `jobNumber` - The job number referencing a batch job submitted to LSF.
- `jobStatus` - The current status of an LSF job.
- `resourceRequest` - A request to the resource manager for a particular type of agent.
- `submitTime` - The time the job was submitted to LSF.

Example

`cmtool getLsfJobs`

Retrieves all LSF job information.

getResource

Description

Finds a resource with full detail by the resource ID number.

Required Arguments

- <*`resourceId`*> - A unique number that identifies each resource.
  Use `getResources` to retrieve a list of resource IDs.

Optional Arguments

None

Result Tags

- `hostMasks` - This is a semi-colon delimited list of host name masks, used to identify the list of hosts that support a resource. “*” is the wildcard character.
- `resourceId` - A unique number that identifies each resource.
- `resourceName` - This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class.

Example

`cmtool getResource 7`

Retrieves resource 7.

getResources

Description

Retrieves a list of all resources.

Required Arguments

None

Optional Arguments

- `--filter` <*`SQL query used to limit the result set`*>
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--maxResults` <*`maximum number of elements to run from a query`*>

- `--firstResult` *<starting index for the query result set>*
  Note: `--firstResult` takes values beginning with `0`. A negative value indicates a record starting from the end of the set, counting backwards, so `-1` is the last record, `-2` is the next to last, and so on.
- `--order` *<SQL order by clause>* - Used to specify ordering for the query result set.
- `--profile <details|info>` - This is the level of detail to return from a query; `details` gets all information and `info` gets a reduced information set.

Result Tags and SQL Query Names

Tag	Description	SQL Query Name for `--filter` and `--order`
`availableResults`	This is a count of 'max' or 'first' results if `--maxResults` or `--firstResult` is specified.	N/A
`hostMasks`	This is a semi-colon delimited list of host name masks, used to identify the list of hosts that support a resource. "*" is the wildcard character.	`host_masks`
`resourceId`	A unique number that identifies each resource.	`id`
`resourceName`	This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class.	`resource_name`

Example

```
cmtool getResources --order resource_name
```

Retrieves a list of resources ordered by the resource name.

getResourceComments

Description

Retrieves resource comments.

Required Arguments

- *<resourceId>* - A unique number that identifies each resource.

Optional Arguments

- *<commentId>* - The unique key that identifies a comment.

Example

```
cmtool getResourceComments 29
```

Retrieves comments for resource 29.

modifyAgentComment

Description

Modifies an agent comment.

Required Arguments

Note: Either `agentId` or `agentName` must also be specified.

- `<commentId>` - The unique key that identifies a comment.
- `<text>` - The text of the item.

Optional Arguments

- `--agentId <unique, internal number that can change; assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`

Example

```
cmtool modifyAgentComment 1037 "changed comment" --agentName SOL1-1
```

Changes comment number 1037 on agent SOL1-1 to “changed comment”.

modifyBuildClassComment

Description

Modifies a build class comment.

Required Arguments

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class.
- `<commentId>` - The unique key that identifies a comment.
- `<text>` - The text of the item.

Optional Arguments

None

Example

```
cmtool modifyBuildClassComment 1037 1129 "This is a low-priority class"
```

modifyResource

Description

Modifies a resource definition.

Required Arguments

- `<resourceId>` - A unique number that identifies each resource.

Optional Arguments

- `--hostMasks` *<a semi-colon delimited list of host name masks>* - This is used to identify the list of hosts that support a resource. "*" is the wildcard character.
- `--resourceName` *<the unique name of the resource>*
- `--description` *<a text description for your reference only>*

Example

```
cmtool modifyResource 27 --hostMasks "SOL*; SRL*"
```

Sets the host masks for resource 27 to "SOL*; SRL*".

modifyResourceComment

Description

Modifies a resource comment. Use `getResources` to retrieve a list of resource IDs.

Required Arguments

- *<resourceId>* - A unique number that identifies each resource.
- *<commentId>* - The unique key that identifies a comment.
- *<text>* - The text of the item.

Optional Arguments

None

Example

```
cmtool modifyResourceComment 1 1015 "new xxx"
```

Changes comment 1015 for resource 1.

setAgentDebug

Description

Sets the agent debug level (see `getAgentStatus`). This command sends a message to the agent(s) in real time; therefore, the agents must be up and connected to the Cluster Manager to have any effect.

Required Arguments

- *<level>* - Possible values are:

all	registry
commands	requests
environment	state
fileinfo	test
log	usage
other	nothing
profile	

Optional Arguments

- `--agentId` *<unique, internal number that can change; assigned by the Cluster Manager>*
- `--agentName` *<name defined by the host where the agent resides [numbers and/or letters]>*
- `--status <1|0>` - Chooses active or inactive agents.
- `--buildId` *<a specific build ID>* - This further restricts the returned agents to those running a specific build ID.
- `--enabled <1|0>` - Choose enabled or disabled agents only.
- `--agents` *<host>*`[:<port>[:`*<agentKey>*`]]` - This specifies individual agents based on their host name and listening port.

Result Tags

- `agentName` - The name of the configured agent.
- `result` - The configuration result.

Example

```
cmtool setAgentDebug profile --agentName SOLAgent-1
```

Sets SOLAgent-1’s debug level to “profile”.

Chapter 3: Build Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

createBuildClass

Description

Creates a build class.

Required Arguments

- `<buildClassName>` - A name assigned by the user for the build class.

Optional Arguments

- `--tagDefinition <a format string that defines the resulting build name>`
- `--annotationLevels <a comma-separated list of values>` - Possible values are: `basic`, `history`, `file`, `lookup`, and `waiting`.
- `--maxAgents <maximum number of agents to request for this build>`
- `--minAgents <minimum number of agents required for this build to run>`
- `--platform <operating system being used/supported>` - Must be either `Windows`, `Linux`, or `Solaris`. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.
- `--priority <priority value>` - The default priority value is `120` (normal). `220` is high and `20` is low. Priority value can be adjusted up or down by 1-10 to “boost” the priority to give certain build classes preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- `--annoUpload <Y|N>` - If set to true, the annotation file is uploaded to Cluster Manager. Values can also be `1`, `0`, `true`, or `false`.
- `--resourceRequest <value>` - This is a request to the resource manager for a particular type of agent. Value is the name of a pre-existing resource.

Example

```
cmtool createBuildClass batch --minAgents 5 --maxAgents 12 --priority 30 --
resourceRequest blades
```

Creates a build class named batch that requires a minimum of 5 agents and a maximum of 12 agents. The priority is relatively low and the requested resource is named blades.

createBuildClassComment

Description

Creates a new build class comment.

Required Arguments

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class. Use `getBuildClass` to retrieve a list of build class IDs.
- `<text>` - The text of the item.

Optional Arguments

None

Example

```
cmtool createBuildClassComment 7 "This build class is for QA builds."
```

Creates a comment for build class 7.

createBuildComment

Description

Creates a new build comment.

Required Arguments

- *<buildId>* - A unique number assigned by the Cluster Manager for each build.
  Use getBuilds to retrieve a list of build IDs.
- *<text>* - The text of the item.

Optional Arguments

None

Example

```
cmtool createBuildComment 1044 "This is our gold build for release 7.0"
```

Creates a comment for build 1044.

deleteBuild

Description

Deletes a build, including all dependent records.

Required Arguments

- *<buildId>* - A unique number assigned by the Cluster Manager for each build.
  Use getBuilds to retrieve a list of build IDs.

Optional Arguments

None

Example

```
cmtool deleteBuild 1037
```

Deletes build 1037.

deleteBuildClass

Description

Deletes a build class, including all dependent records.

Required Arguments

- *<buildClassId>* - A unique number assigned by the Cluster Manager for each build class.
  Use `getBuildClasses` to retrieve a list of build class IDs.

Optional Arguments

None

Example

```
cmtool deleteBuildClass 7
```

Deletes build class 7.

deleteBuildClasses

Description

Deletes a set of build classes, including all dependent records.

Required Arguments

None

Optional Arguments

Note: If no filter is provided, all build classes (except the default) will be deleted.

- `--filter` *<SQL query used to limit the result set>* - For a list of possible SQL values, see the getBuildClasses command.

Example

```
cmtool deleteBuildClasses --filter "max_agents >20"
```

Deletes all build classes with more than 20 maximum agents.

deleteBuildClassComment

Description

Deletes a build class comment.

Required Arguments

- *<buildClassId>* - A unique number assigned by the Cluster Manager for each build class.
- *<commentId>* - The unique key that identifies a comment.

Optional Arguments

None

Example

```
cmtool deleteBuildClassComment 6 1018
```

Deletes comment 1018 for build class 6.

deleteBuildComment

Description

Deletes a build comment.

Required Arguments

- *<buildId>* - A unique number assigned by the Cluster Manager for each build.
- *<commentId>* - The unique key that identifies a comment.
  Use `getBuildComments` to retrieve a list of comment IDs.

Optional Arguments

None

Example

```
cmtool deleteBuildComment 1037 1019
```

Deletes build comment 1019 for build 1037.

deleteBuilds

Description

Deletes a set of builds, including all dependent records.

It is important to remove build logs periodically so they do not fill up the Cluster Manager's available disk space. Uploaded annotation is also considered part of build logs, so remember to clean up build logs regularly if annotation is frequently uploaded to the Cluster Manager.

You can also manage build logs using the Cluster Manager web interface. Select the Builds tab, and then create and run a "Builds by Date" filter to display the set of builds that you want to remove. Click **Delete Filtered Builds** to remove the build logs from disk and from the database.

Required Arguments

If no argument is provided, all builds will be deleted.

Optional Arguments

- `--filter` *<SQL query used to limit the result set>* - For a list of possible SQL values, see the `getBuilds` command.

Example

```
cmtool deleteBuilds --filter "start_time <date_sub(curdate( ), interval 20 day)"
```

Deletes all builds more than 20 days old.

Note: This example is valid for MySQL only. If you use a different database, use syntax that is appropriate for your respective database.

getBuild

Description

Finds a build with full detail by the build's ID number.

Required Arguments

- *<buildId>* - A unique number assigned by the Cluster Manager for each build.
  Use `getBuilds` to retrieve a list of build IDs.

Optional Arguments

None

Result Tags

See getBuilds for descriptions.

`allocatedAgents`	`ipAddress`
`buildClassId`	`jobCount`
`buildClassName`	`lastRequestTime`
`buildId`	`maxAgents`
`buildLogDir`	`minAgents`
`buildName`	`osUserName`
`commandLine`	`platform`
`conflicts`	`priority`
`cwd`	`resourceRequest`
`duration`	`result`
`effectiveAgentAlloc`	`requestedAgents`
`emakeVersion`	`startTime`
`historyExists`	`userLabel`
`historyFile`	`userName`
`hostName`	`waitTime`

Example

```
cmtool getBuild 1000
```

Retrieves build 1000.

getBuilds

Description

Retrieves a list of builds.

Required Arguments

None

Optional Arguments

- `--filter` *`<SQL query used to limit the result set>`* - See the possible values below.
  There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--maxResults` *`<maximum number of elements to run from a query>`*
- `--firstResult` *`<starting index for the query result set>`*
  Note: `--firstResult` takes values beginning with `0`. A negative value indicates a record starting from the end of the set, counting backwards, so `-1` is the last record, `-2` is the next to last, and so on.
- `--order` *`<SQL order by clause>`* - Used to specify ordering for the query result set.
- `--profile <details|info>` - This is the level of detail to return from a query; `details` gets all information and `info` gets a reduced information set.
  Note: You must set `--profile details` in order to print fields that are part of the details category.

Result Tags and SQL Query Names

Tag	Description	SQL Query Name for `--filter` and `--order`
`allocatedAgents`	The number of currently assigned agents for this build.	N/A
`availableResults`	This is a count of 'max' or 'first' results if `--maxResults` or `--firstResult` is specified.	N/A
`buildClassId`	A unique number assigned by the Cluster Manager for each build class.	`build_class_id`
`buildClassName`	A name assigned by the user for the build class.	`build_class_name`
`buildId`	A unique number assigned by the Cluster Manager for each build.	`id`
`buildLogDir`	The directory containing uploaded build logs.	N/A
`buildName`	The build name that is the expanded build class tag.	`build_name`
`commandLine`	The original command-line invocation of Electric Make.	`command_line`
`conflicts`	The number of conflicts in the build.	`conflicts`
`cwd`	The current working directory where Electric Make was invoked.	`cwd`
`duration`	The number of milli-seconds the build has been running. **Note:**`duration` for running builds is always 0.	`duration`
`effectiveAgentAlloc`	The effective agent allocation percentage. 100% means eMake had all the hosts it needed all the time, while a lesser percentage means eMake had the hosts it needed for that percent of time. **Note:**`effective_agent_alloc` for running builds is always 0.	`effective_agent_alloc`
`emakeVersion`	The Electric Make version used for this build.	`emake_version`

`historyExists`	True means the history file existed and was used by the build.	`history_exists`
`historyFile`	The name of the Electric Make history file.	`history_file`
`hostName`	The name of the machine where Electric Make was invoked.	`host_name`
`ipAddress`	The IP address of the machine where Electric Make was invoked.	`ip_address`
`jobCount`	The total number of jobs that ran for the build. **Note:**`job_count` for running builds is always 0.	`job_count`
`lastRequestTime`	The last time Electric Make requested agents for this build.	N/A
`maxAgents`	The maximum number of agents to request for this build.	`max_agents`
`minAgents`	The minimum number of agents required for this build to run.	`min_agents`
`osUserName`	The OS-level name for the user who started Electric Make.	`os_user_name`
`platform`	The operating system being used/supported. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.	`platform`
`priority`	The build priority level. When assigning resources, an optional priority boost value can be selected to give a build class preference over other builds of the same priority level. Higher boost values correspond to greater preference.	`priority`
`resourceRequest`	A request to the resource manager for a particular type of agent.	`resource_request`
`result`	The build result code. -1 means the build is still running, 0-254 are actual exit codes, 256 means the build timed out, and 257 means the build was stopped.	`result`

`requestedAgents`	The number of agents Electric Make requested.	N/A
`startTime`	The time the build was started.	`start_time`
`userLabel`	The user-supplied label (via the eMake command-line), attached to the build.	`user_label`
`userName`	The unique name of the user.	`user_name`
`waitTime`	The number of seconds Electric Make was stalled because it had to wait for agents. **Note:**`wait_time` for running builds is always 0.	`wait_time`

Example

```
cmtool --output simple --fields "startTime,buildName,userId,duration" getBuilds --
filter "duration >10000"
```

Returns the start time, build name, userid, and duration of all builds that ran more than 10 seconds.

getBuildComments

Description

Retrieves a list of related build comments.

Required Arguments

- `<buildId>` - A unique number assigned by the Cluster Manager for each build.

Optional Arguments

- `--commentId <unique key that identifies a comment>`

Result Tags

- `commentId` - The unique key that identifies a comment.
- `createTime` - The time when the item was created.
- `lastModifiedBy` - The user who last modified the item.
- `modifyTime` - The time when the item was last modified.
- `text` - The text of the item.

Example

```
cmtool getBuildComments 1000 --commentId 1039
```

Retrieves comment 1039 for build 1000.

getBuildClass

Description

Finds a build class with full detail by its ID.

Required Arguments

- *`<buildClassId>`* - A unique number assigned by the Cluster Manager for each build class. Use `getBuildClasses` to retrieve a list of build class IDs.

Optional Arguments

None

Result Tags

See `getBuildClasses` for descriptions.

```
annotationLevels
annoUpload
buildClassId
buildClassName
defaultClass
maxAgents
minAgents
notifyOnBuildEnd
platform
priority
resourceRequest
tagDefinition
```

Example

```
cmtool getBuildClass 1
```

Retrieves build class 1.

getBuildClasses

Description

Retrieves a list of build classes with limited detail.

Required Arguments

None

Optional Arguments

- `--filter` *`<SQL query used to limit the result set>`*
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--maxResults` *`<maximum number of elements to run from a query>`*

- `--firstResult` *`<starting index for the query result set>`*
  Note: `--firstResult` takes values beginning with `0`. A negative value indicates a record starting from the end of the set, counting backwards, so `-1` is the last record, `-2` is the next to last, and so on.
- `--order` *`<SQL order by clause>`* - Used to specify ordering for the query result set.
- `--profile <details|info>` - This is the level of detail to return from a query; `details` gets all information and `info` gets a reduced information set.

Result Tags and SQL Query Names

Tag	**Description**	**SQL Query Name for** `--filter` **and** `--order`
`annotationLevels`	Annotation choices to include in the annotation file. Possible values are basic, history, file, lookup, and waiting.	`annotation_levels`
`annoUpload`	If set to true, the annotation file is uploaded to Cluster Manager.	`anno_upload`
`availableResults`	This is a count of ’max’ or ’first’ results if `--maxResults` or `--firstResult` is specified.	N/A
`buildClassId`	A unique number assigned by the Cluster Manager for each build class.	`id`
`buildClassName`	A name assigned by the user for the build class.	`build_class_name`
`defaultClass`	If set, this is the default build class and cannot be deleted.	`default_class`
`maxAgents`	The maximum number of agents to request for this build.	`max_agents`
`minAgents`	The minimum number of agents required for this build to run.	`min_agents`
`notifyOnBuildEnd`	If set to true, the currently logged-in user will receive an email when the build is finished.	`notify_on_build_end`
`platform`	The operating system being used/supported. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.	`platform`

`priority`	The build priority level. When assigning resources, an optional priority boost value can be selected to give a build class preference over other builds of the same priority level. Higher boost values correspond to greater preference.	`priority`
`resourceRequest`	A request to the resource manager for a particular type of agent.	`resource_request`
`tagDefinition`	A format string that defines the resulting build name.	`tag_definition`

Example

```
cmtool getBuildClasses --filter "min_agents <5"
```

Retrieves a list of build classes that require less than 5 agents.

getBuildClassComments

Description

Retrieves a list of related build class comments.

Required Arguments

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class. Use `getBuildClasses` to retrieve a list of build class IDs.

Optional Arguments

- `--commentId <unique key that identifies a comment>`

Result Tags

- `commentId` - The unique key that identifies a comment.
- `createTime` - The time when the item was created.
- `lastModifiedBy` - The user who last modified the item.
- `modifyTime` - The time when the item was last modified.
- `text` - The text of the item.

Example

```
cmtool getBuildClassComments 12
```

Retrieves all build class comments for build class 12.

getBuildUserStats

Description

Retrieves a list of user build statistics, grouped by user name, IP address, or host name.

Required Arguments

- *<groupBy>* - Possible values are `hostName`, `ipAddress`, and `userName`.

Optional Arguments

- `--filter` *<SQL query used to limit the result set>*
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--order` *<SQL order by clause>* - Used to specify ordering for the query result set.

Result Tags and SQL Query Names

Tag	Description	SQL Query Name for `--filter` and `--order`
`duration`	The total number of milli-seconds of all builds, filtered by the value specified in the `groupBy` argument.	`duration`
`entryName`	The value specified in the `groupBy` argument. If `groupBy` is "userName", the entry name is the user name.	N/A
`numOfBuilds`	The number of builds.	N/A
`waitTime`	The number of seconds Electric Make was stalled because it had to wait for agents.	`wait_time`
`workload`	The total number of seconds used by the agents for all of the filtered builds.	`workload`

Example

```
cmtool getBuildUserStats hostName --filter "duration >30000" --order "waitTime desc,
entryName asc"
```

Retrieves build user statistics for builds longer than 30 seconds, grouped by host name and ordered by wait time in a descending order and by entry name (in this case host name) in an ascending order.

modifyBuild

Description

Modifies a build.

Required Arguments

- *<buildId>* - A unique number assigned by the Cluster Manager for each build.
- *<priority>* - The build priority level. Value can be `Low` or `Normal`, but ***not*** High.

Optional Arguments

None

Example

```
cmtool modifyBuild 1137 20
```

Changes build 1137 to priority 20.

modifyBuildClass

Description

Modifies a build class.

Required Arguments

- *<buildClassId>* - A unique number assigned by the Cluster Manager for each build class.

Optional Arguments

- `--buildClassName` *<name assigned by the user for the build class>*
- `--tagDefinition` *<format string that defines the resulting build name>*
- `--annotationLevels` *<a comma-separated list of values>* - Possible values are: `basic`, `history`, `file`, `lookup`, and `waiting`.
- `--maxAgents` *<maximum number of agents to request for this build>*
- `--minAgents` *<minimum number of agents required for this build to run>*
- `--platform` *<the operating system being used/supported>* - If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class. The value can be `Windows`, `Linux`, or `Solaris`.
- `--priority` *<priority value>* - The default priority value is `120` (normal). `220` is high and `20` is low. Priority value can be adjusted up or down by 1-10 to “boost” the priority to give certain build classes preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- `--annoUpload <Y|N>` - If set to true, the annotation file is uploaded to Cluster Manager. Values can also be `1`, `0`, `true`, or `false`.
- `--resourceRequest` *<value>* - This is a request to the resource manager for a particular type of agent. Value is the name of a pre-existing resource.

Example

```
cmtool modifyBuildClass 1 --annoupload true
```

Changes build class 1 to upload annotation files.

modifyBuildContent

Description

Modifies a build comment.

Required Arguments

- *<buildClassId>* - A unique number assigned by the Cluster Manager for each build class.
- *<commentId>* - The unique key that identifies a comment.

- *<text>* - The text of the item.

Optional Arguments

None

Example

```
cmtool modifyBuildComment 16975 1137 "This is not a usable build"
```

setDatabaseConfiguration

Description

Modifies database configuration settings.

Required Arguments

- *<databaseName>* - The database instance name.
- *<databaseType>* - The database type. Possible values are `mariadb`, `mysql`, `oracle`, and `sqlserver`.
- *<hostName>* - Machine name where the database is installed.
- *<port>* - Database port number.
- *<userName>* - Unique name of the user that is used to access the database.
- *<password>* - Secret value used to identify an account for a particular user.

Optional Arguments

None

stopBuild

Description

Stops a running build. (This command has no effect on completed builds.)

Required Arguments

Note: Use `getBuilds --filter "result <0"` to retrieve a list of running builds.

- *<buildId>* - A unique number assigned by the Cluster Manager for each build.

Optional Arguments

None

Example

```
cmtool stopBuild 16937
```

Chapter 4: Cluster Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

createServerComment

Description

Creates a new server comment. Server comments are displayed on the Home page of the Cluster Manager machine.

Required Arguments

- `<text>` - The text of the item.

Optional Arguments

None

Example

```
cmtool createServerComment "cluster needs more servers to handle production builds"
```

deleteLicense

Description

Deletes a license.

Required Arguments

- `<productName>` - ElectricAccelerator, the name of the license.
- `<featureName>` - Feature name of the license, which is currently the `server`.

Optional Arguments

None

Example

```
cmtool deleteLicense ElectricAccelerator Server
```

Deletes the license stored in the server.

deleteMessage

Description

Deletes a specific message, including all dependent records. Messages are listed in the Cluster Manager interface Messages tab and generally are notifications about issues with agents or the Cluster Manager.

Required Arguments

- `<messageId>` - The numeric value that uniquely identifies each message.

Optional Arguments

None

Example

```
cmtool deleteMessage 501
```

deleteMessages

Description

Deletes a set of messages, including all dependent records.

Required Arguments

None

Optional Arguments

- `--filter` *`<SQL query used to limit the result set>`* - For a list of possible SQL values, see the getMessages command.

Example

```
cmtool deleteMessages --filter "create_time <date_sub(curdate( ), interval 200 day)"
```

Removes all messages more than 200 days old.

Note: This example is valid for MySQL only. If you use a different database, use syntax that is appropriate for your respective database.

deleteServerComment

Description

Deletes a server comment.

Required Arguments

- *`<commentId>`* - The unique key that identifies a comment.

Optional Arguments

None

Example

```
cmtool deleteServerComment 1396
```

exportData

Description

Exports Cluster Manager data to a file.

Note: This is a full database dump, which may take an extended period of time to complete depending on the size of the database.

Required Arguments

- *`<fileName>`* - The filename or path to export to. If you use a filename, the destination is the current working directory of the Java process, for example, `/opt/ecloud/i686_Linux` or `C:\ECloud\i686_win32`. If you use a path, the Cluster Manager Java user must have execute and write access to the destination path.

Optional Arguments

None

Example

```
cmtool exportData fileabc
```

getLicense

Description

Retrieves information for one license.

Required Arguments

- *<productName>* - ElectricAccelerator, the name of the license.
- *<featureName>* - Feature name of the license.

Optional Arguments

None

Example

```
cmtool getLicense ElectricAccelerator Server
```

getLicenses

Description

Retrieves all license data.

Required Arguments

None

Optional Arguments

None

Example

```
cmtool getLicenses
```

getMessage

Description

Retrieves a particular message.

Required Arguments

- *<messageId>* - The numeric value that uniquely identifies each message.

Optional Arguments

None

Result Tags

See `getMessages` for descriptions.

```
agentId
agentName
buildId
buildName
createTime
messageId
severity
text
```

Example

```
cmtool getMessage 47
```

getMessages

Description

Retrieves a list of messages

Required Arguments

None

Optional Arguments

- `--filter` *`<SQL query used to limit the result set>`* - See the possible values in the table below.
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--maxResults` *`<maximum number of elements to run from a query>`*
- `--firstResult` *`<starting index for the query result set>`*
  Note: `--firstResult` takes values beginning with `0`. A negative value indicates a record starting from the end of the set, counting backwards, so `-1` is the last record, `-2` is the next to last, and so on.
- `--order` *`<SQL order by clause>`* - Used to specify ordering for the query result set.
- `--profile <details|info>` - This is the level of detail to return from a query; `details` gets all information and `info` gets a reduced information set.

Result Tags and SQL Query Names

Tag	**Description**	**SQL Query Name for** `--filter` **and** `--order`
`agentId`	A unique, internal number assigned to each agent by the Cluster Manager; this number can change.	N/A

`agentName`	A name defined by the host where the agent resides [numbers and/or letters].	`agent_name`
`buildId`	A unique number assigned by the Cluster Manager for each build.	`build_id`
`buildName`	The build name that is the expanded build class tag.	N/A
`createTime`	The time when the item was created.	`create_time`
`messageId`	The numeric value that uniquely identifies each message.	`id`
`severity`	The severity level of the event: Info, Warning, or Error. For `--filter` and `--order`, use the following numerical values: `1` = Info `2` = Warning `3` = Error	`severity`
`text`	The text of the item.	`text`

Example

```
cmtool --output csv --fields buildId,severity,text getMessages --filter "text like
'%I/O%'"
```

Lists all messages in the Cluster Manager that contain the string 'I/O'.

getResourceStats

Description

Retrieves resource usage statistics.

Required Arguments

None

Optional Arguments

- `--filter` *<SQL query used to limit the result set>* - See the possible values in the table below.
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--maxResults` *<maximum number of elements to run from a query>*

- `--firstResult` *`<starting index for the query result set>`*
  Note: `--firstResult` takes values beginning with `0`. A negative value indicates a record starting from the end of the set, counting backwards, so `-1` is the last record, `-2` is the next to last, and so on.
- `--order` *`<SQL order by clause>`* - Used to specify ordering for the query result set.
- `--profile <details|info>` - This is the level of detail to return from a query; `details` gets all information and `info` gets a reduced information set.

Result Tags and SQL Query Names

Tag	Description	SQL Query Name for `--filter` and `--order`
`agentClusterShortage`	The difference between the maximum number of agents requested by all builds and the number of agents that were assigned.	`agent_cluster_shortage`
`agentDemand`	The total maximum number of requests for agents by all running builds.	`agent_demand`
`agentLicenseShortage`	The difference between the maximum request for agents by all builds and the number of agents the license allows.	`agent_license_shortage`
`agentsAvailable`	The total number of active agents in the cluster.	`agents_available`
`agentsInUse`	The total number of agents assigned to builds.	`agents_in_use`
`availableResults`	This is a count of ’max’ or ’first’ results if `--maxResults` or `--firstResult` is specified.	N/A
`buildsDuration`	The average amount of time the current builds have been running.	`builds_duration`
`buildsRunning`	Average number of simultaneous builds running during a specific time period.	`builds_running`
`createTime`	The time when the item was created.	`create_time`

`duration`	The number of milli-seconds the build has been running.	`duration`
`resourceName`	This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class. It is used in the `ea_resource` table and also matches the resource requirement string for eMake.	`resource_name`
`resourceStatId`	The resource ID number that uniquely identifies every resource.	`id`

Example

```
cmtool getResourceStats --maxResults 100 --order "id desc" --filter "resource_
name='Cluster'"
```

Retrieves the 100 most current resource statistic records for the entire cluster.

getServer

Description

Retrieves server configuration.

Required Arguments

None

Optional Arguments

None

Result Tags

- `agentAllocationPolicy` - Defined as either `exclusive` or `shared`.
- `agentLockTimerSec` - When jobs run beyond this number of seconds, the agent should be locked.
- `badAgents` - The number of enabled agents with a bad status.
- `disabledAgents` - The number of disabled agents.
- `emailInterval` - The number of minutes between email notifications.
- `emailItemLimit` - Maximum number of messages per email notification.
- `goodAgents` - The number of enabled agents with a good status.
- `logDaysToKeep` - The number of days to keep message log entries.
- `lsfAvailable` - True if LSF is available to the Cluster Manager.
- `mailFrom` - The value to use in the From header element.
- `mailPrefix` - The string used to prefix subject lines.
- `maxAgents` - The maximum number of agents to request for this build.
- `maxClockSkew` - The maximum clock skew (in seconds) allowed between the Electric Make client and agents in the cluster.

- `minAgents` - The minimum number of agents required for this build to run.
- `preemptionPolicy` - The allocation preemption policy.
- `priority` - The build priority level. When assigning resources, an optional priority boost value can be selected to give a build class preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- `resourceManagerType` - The type of resource manager that Cluster Manager should employ.
- `resourceStatInterval` - In minutes, the interval to collect stats on resource usage.
- `resourceStatKeep` - The number of minutes of resource usage statistics to keep.
- `runningBuilds` - The number of incomplete builds in the system.

Example

```
cmtool getServer
```

getServerComments

Description

Retrieves a list of related server comments.

Required Arguments

None

Optional Arguments

- `--commentId` *`<unique key that identifies a comment>`*

Result Tags

- `commentId` - The unique key that identifies a comment.
- `createTime` - The time when the item was created.
- `lastModifiedBy` - The user who last modified the item.
- `modifyTime` - The time when the item was last modified.
- `text` - The text of the item.

Example

```
cmtool getServerComments
```

Returns all comments related to the server.

getVersion

Description

Retrieves server version information.

Required Arguments

None

Optional Arguments

None

Result Tags

- `label` - The Electric Cloud build label for the server.
- `protocolVersion` - The server protocol version.
- `schemaVersion` - The server database schema version.
- `version` - The string identifying a component version.

Example

```
cmtool getVersion
```

importData

Description

Imports Cluster Manager data from a file.

Note: Because this command imports a full database dump, be advised of the following:

- The import may take an extended period of time to complete depending on the size of the database.
- You must manually delete any old/unused agents from the agents list.
- You must update the license file after import if it previously expired.

Required Arguments

- <*fileName*> - The name of the file to import. The file's path is relative to the current working directory of the Java process, for example, `/opt/ecloud/i686_Linux` or `C:\ECloud\i686_win32`.

Optional Arguments

None

Example

```
cmtool importData fileabc
```

importLicenseData

Description

Imports one or more licenses.

Required Arguments

- <*licenseFile*> - Name of the file containing the license with the path.

Optional Arguments

None

Example

```
cmtool importLicenseData ./license.xml
```

logMessage

Description

Creates a custom message on the Cluster Manager Messages page.

Required Arguments

- `“message text”`

Optional Arguments

Note: If `--buildId` and `--agentName` are on the same line, the message is applied to the build and the agent name.

- `--severity <Debug|Info|Warning|Error>` - You can also use `0`, `1`, `2`, or `3`.
- `--buildId` *`<buildId>`* - The message applies to the specified build only.
- `--agentName` *`<agentName>`* - The message applies to the specified agent name only.

Example

```
cmtool logMessage “some text”
```

modifyServer

Description

Modifies the server configuration.

Required Arguments

None

Optional Arguments

- `--priority` *`<priority value>`* - The default priority value is `120` (normal). `220` is high and `20` is low. Priority value can be adjusted up or down by 1-10 to “boost” the priority to give certain build classes preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- `--emailInterval` *`<number of minutes between email notifications>`*
- `--emailItemLimit` *`<maximum number of messages per email notification>`*
- `--agentAllocationPolicy <exclusive|shared>`
- `--preemptionPolicy` *`<the allocation preemption policy>`*
- `--maxClockSkew` *`<the maximum clock skew (in seconds) allowed between the Electric Make client and agents in the cluster>`*
- `--maxAgents` *`<maximum number of agents to request for this build>`*
- `--minAgents` *`<minimum number of agents required for this build to run>`*
- `--resourceManagerType <none|ea|lsf|cloud|prioritypool>` - Define which resource manager Cluster Manager should employ.
- `--mailFrom` *`<value to use in the From header element>`*
- `--mailPrefix` *`<string used to prefix subject lines>`*

- `--logDaysToKeep` *`<the number of days to keep message log entries>`*
- `--resourceStatInterval`
- `--resourceStatKeep`
- `--wideDeepAllocationPolicy <deep|wide>`- Deep means the agent allocation algorithm favors assigning more agents on the same host to a build. Wide means the algorithm favors assigning more agents from different hosts. If wide, be sure `--agentAllocationPolicy` is set to `shared`.

Example

```
cmtool modifyServer --mailFrom "cm@ourhost.com" --mailPrefix "cm message:"
```

Changes the mail “from” and mail prefix values used for mail notifications sent by the server.

modifyServerComment

Description

Modifies a server comment.

Required Arguments

- *`<commentId>`* - The unique key that identifies a comment.
- *`<text>`* - The text of the item.

Optional Arguments

None

Example

```
cmtool modifyServerComment 1178 "Server is fine"
```

shutdownServer

Description

Stops the server.

IMPORTANT: Use with caution.

Required Arguments

None

Optional Arguments

- `--restart <true|false>`

Example

```
cmtool shutdownServer
```

testAgents

Description

Instructs the Cluster Manager to contact each active agent and update its status.

Required Arguments

None

Optional Arguments

- `--agentId` *<unique, internal number that can change; assigned by the Cluster Manager>*
- `--agentName` *<name defined by the host where the agent resides [numbers and/or letters]>*
- `--filter` *<SQL query used to limit the result set>* For a list of possible SQL values, see the `getAgents` command

Example

```
cmtool testAgents --filter "agent_name like ’%bl%’"
```

This command contacts all agents whose name contains ’bl’ and updates their status.

Chapter 5: Reporting

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

createFilter

Description

Creates a named filter for a specific table.

Note: Non-global filters are stored by user ID; therefore, the same name can be used by more than one user.

Required Arguments

- `<`*`tableName`*`>` - A short string that uniquely identifies the table being filtered.
  Possible table names are: `ec_agent, ec_build, ec_build_class, ec_filter, ec_message, ec_resource, ec_resource_stat.`
- `<`*`filterName`*`>` - A short string that uniquely identifies the filter.
- `<`*`filterQuery`*`>` - A SQL order by clause for the associated table.

Optional Arguments

- `--global <true|false>` - If true, this is a globally visible filter.
- `--order <`*`SQL order by clause`*`>` - Used to specify ordering for the query result set.

Example

```
cmtool createFilter ec_agents linuxAgents ""platform = 'linux'" --global true
```

Creates a global filter that selects Linux agents only.

deleteFilter

Description

Deletes a named filter for a specific table.

Required Arguments

- `<`*`tableName`*`>` - A short string that uniquely identifies the table being filtered.
  Possible table names are: `ec_agent, ec_build, ec_build_class, ec_filter, ec_message, ec_resource, ec_resource_stat.`
- `<`*`filterName`*`>` - A short string that uniquely identifies the filter.

Optional Arguments

- `--global <true|false>` - If true, this is a globally visible filter.

Usage Example

```
cmtool deleteFilter ec_agents linuxAgents --global true
```

getCurrentServerLoad

Description

Retrieves information about the current resource load.

Required Arguments

None

Optional Arguments

None

Result Tags

- `agentsAvailable` - The total number of active agents in the cluster.
- `agentClusterShortage` - The difference between the maximum number of agents requested by all builds and the number of agents that were assigned.
- `agentDemand` - The total maximum number of requests for agents by all running builds.
- `agentLicenseShortage` - The difference between the maximum request for agents by all builds and the number of agents the license allows.
- `agentsInUse` - The total number of agents assigned to builds.
- `buildsDuration` - The average amount of time the current builds have been running.
- `buildsRunning` - Average number of simultaneous builds running during a specific time period.
- `createTime` - The time when the item was created.
- `duration` - The number of milli-seconds the build has been running.
- `resourceName` - This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class. It is used in the `ea_resource` table and also matches the resource requirement string for eMake.
- `resourceStatId` - The resource ID number that uniquely identifies every resource.

Example

```
cmtool getCurrentServerLoad
```

getFilter

Description

Retrieves a named filter for a specific table.

Required Arguments

- `<`*`tableName`*`>` - A short string that uniquely identifies the table being filtered.
  Possible table names are: `ec_agent, ec_build, ec_build_class, ec_filter, ec_message, ec_resource, ec_resource_stat.`
- `<`*`filterName`*`>` - A short string that uniquely identifies the filter.

Optional Arguments

- `--global <true|false>` - If true, this is a globally visible filter.

Example

```
cmtool getFilter ec_agent agentFilter
```

getFilters

Description

Retrieves a list of saved filters for the current user.

Required Arguments

None

Optional Arguments

- `--filter` *`<SQL query used to limit the result set>`* - For a list of possible SQL values, see the getAgentscommand.
- `--maxResults` *`<maximum number of elements to run from a query>`*
- `--firstResult` *`<starting index for the query result set>`*
  Note: `--firstResult` takes values beginning with `0`. A negative value indicates a record starting from the end of the set, counting backwards, so `-1` is the last record, `-2` is the next to last, and so on.
- `--order` *`<SQL order by clause>`* - Used to specify ordering for the query result set.

Example

```
cmtool getFilters --filter "table_name = ’ec_agent’ && user_name is null"
```

Retrieves a list of all global filters for the agent table.

modifyFilter

Description

Updates a named filter for a specific table.

Required Arguments

- *`<tableName>`* - A short string that uniquely identifies the table being filtered.
  Possible table names are: `ec_agent, ec_build, ec_build_class, ec_filter, ec_message, ec_resource, ec_resource_stat.`
- *`<filterName>`* - A short string that uniquely identifies the filter.
- *`<filterQuery>`* - A SQL order by clause for the associated table.

Optional Arguments

- `--global <true|false>` - If true, this is a globally visible filter. This parameter is required for global filters.
- `--order` *`<SQL order by clause>`* - Used to specify ordering for the query result set.

Example

```
cmtool modifyFilter ec_agent agentFilter "id 750" --order agent_name
```

Chapter 6: User Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

addGroupMember

Description

Adds a user name to the member list for a specific group.

Required Arguments

- `<groupName>` - The unique name of the group.
- `<userName>` - The unique name of the user.

Optional Arguments

None

Example

```
cmtool addGroupMember DevGroupA ec123
```

Adds user 'ec123' to group DevGroupA.

changeOwnUser

Description

Modifies the settings for the currently logged-in user.

Required Arguments

- `<userName>` - The unique name of the user.

Optional Arguments

- `--fullUserName <real world name of the user>`
- `--email <the associated user email address>`
- `--password <password for a particular user>`
- `--passwordFile <path to password file>` - If `--password` is also specified, `--passwordFile` overrides its value in the command line.

Example

```
cmtool ec123 --fullUserName "Mary Smith"
```

createGroup

Description

Creates a new local group.

Required Arguments

- `<groupName>` - The unique name of the group.

Optional Arguments

None

Example

```
cmtool createGroup DevGroupA
```

createUser

Description

Creates a new local user.

Required Arguments

- `<userName>` - The unique name of the user.
- `<password>` - The password for a particular user.

Optional Arguments

- `--fullUserName <real world name of the user>`
- `--email <the associated user email address>`
- `--passwordFile <path to password file>` - If `--password` is also specified, `--passwordFile` overrides its value in the command line.

Example

```
cmtool createUser ec123 psword --fullUserName "Bob Smith" --email "ec123@ourhost.com"
```

Creates a new user named “ec123” whose real-world name is Bob Smith; with “psword” as his password.

Note: If you do not wish to expose passwords on the command line, you can omit the password from the example above. Press the Enter key after typing the command string (without the password) and you will be prompted for the password.

deleteGroup

Description

Deletes a local group.

Required Arguments

- `<groupName>` - The unique name of the group.

Optional Arguments

None

Example

```
cmtool deleteGroup DevGroupA
```

Removes the ’DevGroupA’ group from the Cluster Manager.

deleteUser

Description

Deletes a local user.

Required Arguments

- `<userName>` - The unique name of the user.

Optional Arguments

None

Example

```
cmtool deleteUser ec123
```

getAccessEntries

Description

Retrieves permissions for all users and groups that were granted server access.

Required Arguments

None

Optional Arguments

None

Result Tags

- `entityName` - A user or group name in an access entry.
- `permissions` - The list of permission flags for a particular entity.

Example

```
cmtool getAccessEntries
```

getGroupMembers

Description

Retrieves a list of users in a specific group.

Required Arguments

- `<groupName>` - The unique name of the group.

Optional Arguments

None

Result Tags

- `userName` - The unique name of the user.

Example

```
cmtool getGroupMembers
```

Retrieves a list of user name elements.

getGroups

Description

Finds all groups known to the server. If “local” is true, returns local groups only.

Required Arguments

- `<userName>` - The unique name of the user.

Optional Arguments

- `--local <true|false>` - If true, returns local users only.

Result Tags

- `groupName` - The unique name of the group.
- `mutable` - True if the associated user or group record is modifiable.
- `providerName` - The human-readable name configured for the directory provider of a specific user or group.

Example

```
cmtool getGroups
```

Returns a list of `groupInfo` elements.

getEffectivePermissions

Description

Retrieves the permissions for the currently logged-in user.

Required Arguments

None

Optional Arguments

None

Result Tags

- `permissions` - The list of permission flags for a particular entity.

Possible Results

```
AgentsDelete                    MaintenanceWrite
AgentsRead                      MessageLogDelete
AgentsWrite                     MessageLogRead
BuildsDelete                    MessageLogWrite
BuildsRead                      ReportsDelete
BuildsWrite                     ReportsRead
ClassesDelete                   ReportsWrite
ClassesRead                     ResourcesDelete
ClassesWrite                    ResourcesRead
EMakeImpersonate                ResourcesWrite
EMakeInvoke                     ServerAccess
MaintenanceDelete               UserModify
MaintenanceRead
```

Example

`cmtool getEffectivePermissions`

Retrieves the permissions for the currently logged-in user.

getPermissions

Description

Retrieves permissions for a particular user or group.

Required Arguments

- *<principalType>* - Can be `user` or `group`.
- *<entityName>* - A user or group name in an access entry.

Optional Arguments

None

Result Tags

- `permissions` - The list of permission flags for a particular entity.

Possible Results

```
AgentsDelete                    MaintenanceWrite
AgentsRead                      MessageLogDelete
AgentsWrite                     MessageLogRead
BuildsDelete                    MessageLogWrite
BuildsRead                      ReportsDelete
BuildsWrite                     ReportsRead
ClassesDelete                   ReportsWrite
ClassesRead                     ResourcesDelete
ClassesWrite                    ResourcesRead
EMakeImpersonate                ResourcesWrite
EMakeInvoke                     ServerAccess
MaintenanceDelete               UserModify
MaintenanceRead
```

Example

```
cmtool getPermissions group DevGroupA
```

Retrieves permissions for group DevGroupA.

getUser

Description

Finds a specific user known to the server.

Required Arguments

- `<userName>` - The unique name of the user.

Optional Arguments

None

Result Tags

- `email` - The associated user email address.
- `fullUserName` - The real world name of the user.
- `groupName` - The unique name of the group.
- `mutable` - True if the associated user or group record is modifiable.
- `providerName` - The human-readable name configured for the directory provider of a specific user or group.
- `userName` - The unique name of the user.

Example

```
cmtool getUser ec123
```

Retrieves the attributes for user ec123.

getUsers

Description

Finds all users known to the server. If “local” is true, returns local users only.

Required Arguments

None

Optional Arguments

- `--pattern <pattern>` - <pattern> is a wildcard pattern for a user name where “*” matches any character or SQL “like” string. If LDAP is set up for getting users, the * is the preferred wildcard, as % is not understood by LDAP (this limits the result set to records in the local database).
- `--local <true|false>` - If true, returns local users only.

Result Tags

See `getUser` for descriptions.

```
email
fullUserName
mutable
providerName
userName
```

Example

```
cmtool getUsers --pattern ec*
```

Retrieves information on all user IDs that begin with ’ec’.

getUserSettings

Description

Retrieves settings for the currently logged-in user.

Required Arguments

None

Optional Arguments

None

Example

```
cmtool getUserSettings
```

login

Description

Logs in to the client with the appropriate credentials and creates a session file in the users home directory, which allows subsequent calls to cmtool to connect to the Cluster Manager.

Required Arguments

- `<`*`userName`*`>` - The unique name of the user.
- `<`*`password`*`>` - The password for a particular user.

Optional Arguments

- `--passwordFile` *`<path to password file>`* - If `--password` is also specified, `--passwordFile` overrides its value in the command line.

Result Tags

- `sessionId` - This is a session "cookie."

Example

```
cmtool login ec123 bobs
```

Logs in a user named “ec123” whose password is “bobs”.

Note: If you do not wish to expose passwords on the command line, you can omit the password from the example above. Press the Enter key after typing the command string (without the password) and you will be prompted for the password.

logout

Description

Logs out of the client session.

Required Arguments

None

Optional Arguments

None

modifyGroup

Description

Modifies a local group.

Required Arguments

- `<`*`groupName`*`>` - The unique name of the group.

Optional Arguments

- `--newName` `<`*`new group name`*`>`

Example

```
cmtool modifyGroup DevGroupA --newName GroupDevA
```

modifyUser

Description

Modifies a local user.

Required Arguments

- `<`*`userName`*`>` - The unique name of the user.

Optional Arguments

- `--fullUserName` `<`*`real world name of the user`*`>`
- `--email` `<`*`the associated user email address`*`>`
- `--password` `<`*`password for a particular user`*`>`
- `--passwordFile` `<`*`path to password file`*`>` - If `--password` is also specified, `--passwordFile` overrides its value in the command line.

Example

```
cmtool modifyUser ec123 --fullUserName "Mary Smith"
```

removeGroupMember

Description

Deletes a user name from a specific group member list.

Required Arguments

- *`<groupName>`* - The unique name of the group.
- *`<userName>`* - The unique name of the user.

Optional Arguments

None

Example

```
cmtool removeGroupMember DevGroupA ec123
```

setBuildEndNotification

Description

Enables/disables notification when builds of this class end for the currently logged-in user.

Required Arguments

- *`<buildClassId>`* - A unique number assigned by the Cluster Manager for each build class. Use `getBuildClasses` to retrieve a list of build class IDs.
- *`<enabled>`* - Set this to true to enable notification and to false to disable it.

Optional Arguments

None

Example

```
cmtool setBuildEndNotification 1 true
```

Enables build ’end notification’ for build class 1.

setPermissions

Description

Creates or modifies permissions for a user or group. The permissions are a space-separated list of permission names.

Required Arguments

- *`<principalType>`* - Can be `user` or `group`.
- *`<entityName>`* - A user or group name in an access entry.
- *`<permissions>`* - The list of permission flags for a particular entity. See the available permissions flags below.

Optional Arguments

None

Available Permission Flags

```
AgentsDelete
AgentsRead
AgentsWrite
BuildsDelete
BuildsRead
BuildsWrite
ClassesDelete
ClassesRead
ClassesWrite
EMakeImpersonate
EMakeInvoke
MaintenanceDelete
MaintenanceRead
MaintenanceWrite
MessageLogDelete
MessageLogRead
MessageLogWrite
ReportsDelete
ReportsRead
ReportsWrite
ResourcesDelete
ResourcesRead
ResourcesWrite
ServerAccess
UserModify
```

Example

```
cmtool setPermissions user ec123 "BuildsRead AgentsRead"
```

Restricts user ec123 to read-only privileges for builds and agents.

setUserSettings

Description

Updates settings for the currently logged-in user.

Required Arguments

- `<`*`watchMessages`*`>` - Indicates whether you want to receive notifications when messages of the specified notification level arrive. Values are `Y`, `N`, `y`, `n`, `yes`, `no`, `Yes`, or `No`.

Optional arguments

- `--notificationLevel <Info|Warning|Error>`

Example

```
cmtool setUserSettings yes --notificationLevel Info
```

Sets the current user to receive notifications for ’Info’ level messages.