ElectricAccelerator Version 8.1
Copyright © 2002–2015 Electric Cloud, Inc. All rights reserved.
Published 9/28/2015
Electric Cloud® believes the information in this publication is accurate as of its publication date. The information is subject to change without notice and does not represent a commitment from the vendor.
THE INFORMATION IN THIS PUBLICATION IS PROVIDED “AS IS.” ELECTRIC CLOUD, INCORPORATED MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
Use, copying, and distribution of any ELECTRIC CLOUD software described in this publication requires an applicable software license.
Copyright protection includes all forms and matters of copyrightable material and information now allowed by statutory or judicial law or hereinafter granted, including without limitation, material generated from software programs displayed on the screen such as icons and screen display appearance.
The software and/or databases described in this document are furnished under a license agreement or nondisclosure agreement. The software and/or databases may be used or copied only in accordance with terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or nondisclosure agreement.

Trademarks
Electric Cloud, ElectricAccelerator, ElectricAccelerator Huddle, ElectricCommander, ElectricFlow, ElectricInsight, and Electric Make are registered trademarks or trademarks of Electric Cloud, Incorporated.
All other trademarks used herein are the property of their respective owners.
## Contents

### Chapter 1: Overview
- Logging In ............................................................................. 1-1
- Using cmtool ........................................................................... 1-1
- Using runAgentCmd ................................................................. 1-2
- Global Arguments (Optional) .................................................... 1-2

### Chapter 2: API Requests
- Agent Management .................................................................. 2-1
  - changeAgentsEnabled .......................................................... 2-2
  - createAgentComment ........................................................... 2-2
  - createResource ................................................................. 2-3
  - createResourceComment ..................................................... 2-3
  - deleteAgentComment ........................................................... 2-4
  - deleteAgents ...................................................................... 2-4
  - deleteResource ................................................................. 2-5
  - deleteResources .................................................................. 2-5
  - deleteResourceComment ..................................................... 2-6
  - getAgentComments ............................................................ 2-6
  - getAgentPerformance ......................................................... 2-7
  - getAgents ............................................................................ 2-8
  - getAgentStatus ................................................................... 2-10
  - getLsfInformation ............................................................... 2-11
  - getLsfJobs .......................................................................... 2-12
  - getResource .......................................................................... 2-13
  - getResources ......................................................................... 2-13
  - getResourceComments .......................................................... 2-15
  - modifyAgentComment ........................................................... 2-15
  - modifyBuildClassComment ................................................... 2-15
  - modifyResource ................................................................. 2-16
  - modifyResourceComment .................................................... 2-17
  - setAgentDebug ...................................................................... 2-17
- Build Management ................................................................... 2-18
  - createBuildClass ................................................................. 2-18
  - createBuildClassComment .................................................... 2-20
  - createBuildComment ........................................................... 2-21
  - deleteBuild ......................................................................... 2-21
  - deleteBuildClass .............................................................. 2-22
  - deleteBuildClasses ........................................................... 2-22
  - deleteBuildClassComment .................................................... 2-23
  - deleteBuildComment ........................................................... 2-23
deleteBuilds ................................................................. 2-23
getBuild ................................................................. 2-24
getBuilds ................................................................. 2-25
getBuildComments ....................................................... 2-28
getBuildClass ............................................................ 2-29
getBuildClasses ......................................................... 2-30
getBuildClassComments .............................................. 2-32
getBuildUserStats ....................................................... 2-32
modifyBuild ............................................................. 2-33
modifyBuildClass ......................................................... 2-34
modifyBuildComment .................................................. 2-36
setDatabaseConfiguration .......................................... 2-37
stopBuild ................................................................. 2-37
Cluster Management .................................................... 2-38
createServerComment ................................................. 2-38
deleteLicense ............................................................ 2-38
deleteMessage ........................................................... 2-38
deleteMessages .......................................................... 2-39
deleteServerComment .................................................. 2-39
exportData ............................................................... 2-40
getLicense ................................................................. 2-40
getLicenses ............................................................... 2-41
getMessage ............................................................... 2-41
getMessages ............................................................. 2-42
getServerResourceStats .............................................. 2-43
getServer ................................................................. 2-45
getServerComments .................................................... 2-46
getVersion ............................................................... 2-47
importData ............................................................... 2-48
importLicenseData ...................................................... 2-48
logMessage .............................................................. 2-49
modifyServer ........................................................... 2-49
modifyServerComment ................................................ 2-51
shutdownServer ........................................................ 2-51
testAgents .............................................................. 2-51
Reporting ................................................................. 2-52
createFilter ............................................................. 2-52
deleteFilter ............................................................. 2-52
getCurrentServerLoad ............................................... 2-53
getFilter ................................................................. 2-54
getFilters ............................................................... 2-55
modifyFilter ............................................................ 2-55
User Management ....................................................... 2-56
addGroupMember ........................................................ 2-56
changeOwnUser ........................................................ 2-56
createGroup ............................................................ 2-57
createUser .............................................................. 2-57
deleteGroup ........................................................................................................ 2-58
deleteUser ........................................................................................................ 2-58
getAccessEntries .............................................................................................. 2-59
getGroupMembers ............................................................................................. 2-59
getGroups .......................................................................................................... 2-60
getEffectivePermissions .................................................................................... 2-60
getPermissions .................................................................................................. 2-61
getUser .............................................................................................................. 2-62
getUsers ............................................................................................................ 2-63
getUserSettings ................................................................................................. 2-63
login .................................................................................................................. 2-64
logout ................................................................................................................. 2-64
modifyGroup ..................................................................................................... 2-64
modifyUser ......................................................................................................... 2-65
removeGroupMember ......................................................................................... 2-65
setBuildEndNotification ..................................................................................... 2-66
setPermissions .................................................................................................. 2-66
setUserSettings ................................................................................................. 2-67
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
- Using cmtool
- Using runAgentCmd
- Global Arguments (Optional)

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 might 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 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 cmttool online help.

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

--help [command]
Description: Prints this message and exits. If a command is specified, prints the help text for that command.

--help-commands
Description: Prints the list of available commands with a short description.

--help-fields <command>
Description: Displays a list of fields for a command—requires the <command> argument.

--version
Description: Prints cmttool version number.

--server <hostname>
Description: ElectricAccelerator server address. Defaults to the ACCELERATOR_SERVER environment variable. If this variable does not exist, default is to the localhost.

--port <port>
Description: HTTP listener port on the ElectricAccelerator server. Defaults to port 8030.

--securePort <securePort>
Description: HTTPS listener port on the ElectricAccelerator server. Defaults to port 8031.

--secure
Description: Uses HTTPS to communicate with the ElectricAccelerator server.

--timeout <seconds>
**Description:** 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>
```

**Description:** 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>
```

**Description:** List is a comma separated list of fields to emit when using ‘csv’ or ‘simple’ output styles. Default is all fields.

```
--sessionFile <path>
```

**Description:** Overrides the location where session information will be stored.
Chapter 2: API Requests

This section describes cmtool API requests.

Topics:

- Agent Management
- Build Management
- Cluster Management
- Reporting
- User Management
Agent Management

This section describes agent management-related requests.

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.

changeAgentsEnabled

Changes the agent enabled status of one or more agents.

Required Arguments

derived

Description: Possible values are true or false.

Optional Arguments

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

agentId

Description: Unique, internal number that can change; assigned by the Cluster Manager.

agentName

Description: Name defined by the host where the agent resides [numbers and/or letters].

filter

Description: A 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 this argument for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

Syntax

cmtool changeAgentsEnabled <enabled> [optionals...]

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

Creates a new agent comment.

Required Arguments

Note: Either agentId or agentName must also be specified.
**createAgentComment**

Description: The comment text.

Optional Arguments

agentId

Description: Unique, internal number that can change; assigned by the Cluster Manager.

agentName

Description: Name defined by the host where the agent resides [numbers and/or letters].

Syntax

cmtool createAgentComment <text> [optionals...]

Example

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

Creates a comment for an agent named “linagent”.

**createResource**

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

Description: 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

Description: 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

Description: A quote-enclosed text description for your reference only.

Syntax

cmtool createResource <resourceName> <hostMasks> [optionals...]

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**

Creates a new resource comment.
Required Arguments

resourceId
   Description: A unique number that identifies each resource.

text
   Description: The comment text.

Optional Arguments

None

Syntax

cmtool createResourceComment <resourceId> <text>

Example

cmtool createResourceComment 2 "This resource identifies production servers"
Creates a comment for resource 2.

deleteAgentComment

Deletes an agent comment.

Required Arguments

Note: Either agentId or agentName must also be specified.

commentId
   Description: The unique key that identifies a comment. Use getAgentComments to get a list of comment ID numbers.

Optional Arguments

agentId
   Description: Unique, internal number that can change; assigned by the Cluster Manager.

agentName
   Description: Name defined by the host where the agent resides [numbers and/or letters].

Syntax

cmtool deleteAgentComment <commentId> [optionals...]

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

Deletes one or more agents, including all dependent records.
Chapter 2: API Requests

**Required Arguments**
None

**Optional Arguments**

agentId
- **Description:** Unique, internal number that can change; assigned by the Cluster Manager.

agentName
- **Description:** Name defined by the host where the agent resides [numbers and/or letters].

filter
- **Description:** A SQL query used to limit the result set. For a list of possible SQL values, see the `getAgents` command.

**Syntax**
cmtool deleteAgents [optionals...]

**Example**
cmtool deleteAgents --agentName winbuild1
Deletes agent “winbuild1” and all associated comments.

**deleteResource**
Deletes a resource definition.

**Required Arguments**

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

**Optional Arguments**
None

**Syntax**
cmtool deleteResource <resourceId>

**Example**
cmtool deleteResource 3
Deletes the resource definition for resource 3.

**deleteResources**
Deletes multiple resource definitions.

**Required Arguments**
None
Optional Arguments

filter
  Description: A SQL query used to limit the result set. For a list of possible SQL values, see the getResources command.

Syntax
cmtool deleteResources [optionals...]

Example
cmtool deleteResources
Deletes all resource definitions.

deleteResourceComment

Deletes a resource comment.

Required Arguments

resourceId
  Description: A unique number that identifies each resource.

commentId
  Description: The unique key that identifies a comment. Use the getResourceComments command to get a list of comment IDs.

Optional Arguments

None

Syntax
cmtool deleteResourceComment <resourceId> <commentId>

Example
cmtool deleteResourceComment 3 49
Deletes comment 49 from resource 3.

getAgentComments

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

Required Arguments

None

Optional Arguments

agentId
  Description: Unique, internal number that can change; assigned by the Cluster Manager.

agentName
  Description: Name defined by the host where the agent resides [numbers and/or letters].
Chapter 2: API Requests

Result Tags

- **commentId**
  - **Description:** The unique key that identifies a comment.

- **createTime**
  - **Description:** The time when the item was created.

- **lastModifiedBy**
  - **Description:** The user who last modified the item.

- **modifyTime**
  - **Description:** The time when the item was last modified.

- **text**
  - **Description:** The text of the item.

Syntax

```cmtool getAgentComments [optionals...]```

Example

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

Retrieves all comments for agent "ahost-3".

**getAgentPerformance**

Retrieves the performance log of one or more agents.

**Required Arguments**

None

**Optional Arguments**

- **agentId**
  - **Description:** Unique, internal number that can change; assigned by the Cluster Manager.

- **agentName**
  - **Description:** Name defined by the host where the agent resides [numbers and/or letters].

- **agents**
  - **Description:** A list of agents whose performance you want to see.

- **buildId**
  - **Description:** Further restricts the returned agents to those running a specific build ID.

- **status**
  - **Description:** Can be 1 or 0. Choose active or inactive agents only.

- **enabled**
  - **Description:** Can be 1 or 0. Choose enabled or disabled agents only.
Result Tags

agentName
Description: This is the name of the agent as it appears on the web page (product UI).

result
Description: This is the performance information of the agent.

Syntax

cmtool getAgentPerformance [optionals...]

Example

cmtool getAgentPerformance --agentName SOL1-1

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

getAgents

Retrieves a list of agents.

Required Arguments
None

Optional Arguments

agentId
Description: Unique, internal number that can change; assigned by the Cluster Manager.

agentName
Description: Name defined by the host where the agent resides [numbers and/or letters].

filter
Description: A SQL query used to limit the result set. See the possible values below.
Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using this argument for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

maxResults
Description: The maximum number of elements to return from a query.

firstResult
Description: The starting index for the query result set.
Note: This argument 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
Description: A SQL order by clause. Used to specify ordering for the query result set.

profile
Description: Can be details or 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 this argument to details in order to print fields that are part of the details category.
Result Tags and SQL Query Names

a2aPort

Description: The agent to agent protocol communication port.
SQL query name for --filter and --order:a2a_port

agentId

Description: A unique, internal number assigned to each agent by the Cluster Manager; this number can change.
SQL query name for --filter and --order:id

agentName

Description: A name defined by the host where the agent resides [numbers and/or letters].
SQL query name for --filter and --order:agent_name

agentVersion

Description: The agent version string.
SQL query name for --filter and --order:agent_version

availableResults

Description: This is a count of 'max' or 'first' results if --maxResults or --firstResult is specified.
SQL query name for --filter and --order:N/A

buildId

Description: A unique number assigned by the Cluster Manager for each build.
SQL query name for --filter and --order:current_build_id

buildName

Description: The build name that is the expanded build class tag.
SQL query name for --filter and --order:N/A

consolePort

Description: The agent console port.
SQL query name for --filter and --order:console_port

efsVersion

Description: The EFS version string.
SQL query name for --filter and --order:efs_version

enabled

Description: The flag indicating if an agent is enabled or not.
SQL query name for --filter and --order:enabled

errorCount

Description: The number of internal agent errors.
SQL query name for --filter and --order:error_count

hostName

Description: The name of the machine where eMake was invoked.
SQL query name for --filter and --order:host_name

inPenaltyBox
**Description**: A flag indicating eMake had a recent problem with this agent.
SQL query name for --filter and --order: N/A

**ipAddress**
**Description**: The agent IP address.
SQL query name for --filter and --order: ip_address

**lastErrorTime**
**Description**: The last time the agent experienced an error.
SQL query name for --filter and --order: last_error_time

**lastPingTime**
**Description**: The last time the agent was pinged to determine its status.
SQL query name for --filter and --order: last_ping_time

**platform**
**Description**: 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.
SQL query name for --filter and --order: platform

**port**
**Description**: The agent protocol communication port.
SQL query name for --filter and --order: port

**restartCount**
**Description**: The number of agent restarts.
SQL query name for --filter and --order: restart_count

**status**
**Description**: The agent status. 1 = OK, but anything else is an error code.
SQL query name for --filter and --order: status

**statusDetail**
**Description**: If the last status update resulted in an error, it contains the error string (or the “OK” string if no error occurred).
SQL query name for --filter and --order: status_detail

**webPort**
**Description**: The agent web server port.
SQL query name for --filter and --order: web_port

**Syntax**
cmtool getAgents [optionals...]

**Example**
cmtool getAgents --filter "agent_name like ’%SOL%’"
Retrieves a list of all agents whose names start with “SOL”.

getAgentStatus
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
   **Description:** Unique, internal number that can change; assigned by the Cluster Manager.

agentName
   **Description:** Name defined by the host where the agent resides [numbers and/or letters].

agents
   **Description:** A list of agents whose status you want to see.

buildId
   **Description:** Further restricts the returned agents to those running a specific build ID.

status
   **Description:** Can be 1 or 0. Choose active or inactive agents only.

enabled
   **Description:** Can be 1 or 0. Choose enabled or disabled agents only.

**Result Tags**

agentName
   **Description:** This is the name of the agent as it appears on the web page (product UI).

result
   **Description:** This is the text string that describes the current state of the agent.

**Syntax**
cmtool getAgentStatus [optionals...]

**Example**
cmtool getAgentStatus --agentName SOL1-1

Returns the status of the agent named “SOL1-1”.

**getLsfInformation**

Retrieves current information about the LSF interface.

**Note:** LSF must be enabled to retrieve information.

**Required Arguments**
None

**Optional Arguments**
None
Result Tags

clusterName
  Description: The name of the LSF grid cluster.

lsfAvailable
  Description: “1” if LSF is available to the Cluster Manager.

masterName
  Description: The LSF Master Host name.

numPendingAgentJobs
  Description: The number of LSF jobs submitted by the Cluster Manager that are waiting to run.

numRunningAgentJobs
  Description: The number of LSF jobs Cluster Manager submitted that are running now.

statusMessage
  Description: A message.

Syntax

cmtool getLsfInformation

Example

cmtool --output csv --fields getLsfInformation

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

getLsfJobs

Retrieves information about all jobs submitted to LSF.

Required Arguments
None

Optional Arguments
None

Result Tags

agentHostName
  Description: The machine name where the agent is running.

jobNumber
  Description: The job number referencing a batch job submitted to LSF.

jobStatus
  Description: The current status of an LSF job.

resourceRequest
  Description: A request to the resource manager for a particular type of agent.
submitTime
   **Description:** The time the job was submitted to LSF.

**Syntax**

`cmtool getLsfJobs`

**Example**

`cmtool getLsfJobs`

Retrieves all LSF job information.

**getResource**

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

**Required Arguments**

`resourceId`

   **Description:** A unique number that identifies each resource. Use `getResources` to retrieve a list of resource IDs.

**Optional Arguments**

None

**Result Tags**

`hostMasks`

   **Description:** 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`

   **Description:** A unique number that identifies each resource.

`resourceName`

   **Description:** This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class.

**Syntax**

`cmtool getResource <resourceId>`

**Example**

`cmtool getResource 7`

Retrieves resource 7.

**getResources**

Retrieves a list of all resources.

**Required Arguments**

None
Optional Arguments

**filter**

**Description:** A SQL query used to limit the result set. See the possible values below.

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

**maxResults**

**Description:** The maximum number of elements to return from a query.

**firstResult**

**Description:** The starting index for the query result set.

**Note:** This argument 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**

**Description:** A SQL order by clause. Used to specify ordering for the query result set.

**profile**

**Description:** Can be details or 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 this argument to details in order to print fields that are part of the details category.

and SQL Query Names

**availableResults**

**Description:** This is a count of 'max' or 'first' results if --maxResults or --firstResult is specified.

**Note:** SQL query name for --filter and --order: N/A

**hostMasks**

**Description:** 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.

**Note:** SQL query name for --filter and --order: host_masks

**resourceId**

**Description:** A unique number that identifies each resource.

**Note:** SQL query name for --filter and --order: id

**resourceName**

**Description:** This name is used on the eMake parameter: --emake-resource, and can be specified in a build class.

**Note:** SQL query name for --filter and --order: resource_name

Syntax

cmtool getResources [optionals...]

Example

cmtool getResources --order resource_name

Retrieves a list of resources ordered by the resource name.
**getResourceComments**
Retrieves resource comments.

**Required Arguments**

resourceId
Description: A unique that identifies each resource.

**Optional Arguments**

commentId
Description: A unique key that identifies a comment.

**Syntax**

cmtool getResourceComments <resourceId> [optionals...]

**Example**

cmtool getResourceComments 29
Retrieves comments for resource 29.

**modifyAgentComment**
Modifies an agent comment.

**Required Arguments**

Note: Either agentId or agentName must also be specified.

commentId
Description: A unique key that identifies a comment.

text
Description: The comment text.

**Optional Arguments**

agentId
Description: Unique, internal number that can change; assigned by the Cluster Manager.

agentName
Description: Name defined by the host where the agent resides [numbers and/or letters].

**Syntax**

cmtool modifyAgentComment <commentId> <text> [optionals...]

**Example**

cmtool modifyAgentComment 1037 "changed comment" --agentName SOL1-1
Changes comment number 1037 on agent SOL1-1 to “changed comment”.

**modifyBuildClassComment**
Modifies a build class comment.

**Required Arguments**

- **buildClassId**
  
  **Description:** A unique number assigned by the Cluster Manager that identifies each build class.

- **commentId**
  
  **Description:** A unique key that identifies a comment.

- **text**
  
  **Description:** The comment text.

**Optional Arguments**

- **None**

**Syntax**

cmtool modifyBuildClassComment <buildClassId> <commentId> <text>

**Example**

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

### modifyResource

Modifies a resource definition.

**Required Arguments**

- **resourceId**
  
  **Description:** A unique number that identifies each resource.

**Optional Arguments**

- **hostMasks**
  
  **Description:** A semi-colon delimited list of host name masks used to identify the list of hosts that support a resource. "*" is the wildcard character.

- **resourceName**
  
  **Description:** The unique name of the resource.

- **description**
  
  **Description:** A text description for your reference only.

**Syntax**

cmtool modifyResource <resourceId> [optionals...]

**Example**

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

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

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

**Required Arguments**

- **resourceId**
  - **Description**: A unique number that identifies each resource.

- **commentId**
  - **Description**: A unique key that identifies a comment.

- **text**
  - **Description**: The comment text.

**Optional Arguments**

None

**Syntax**

```
cmtool modifyResourceComment <resourceId> <commentId> <text>
```

**Example**

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

Changes comment 1015 for resource 1.

setAgentDebug

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**
  - **Description**: The debug level value. Can be:

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

**Optional Arguments**

- **agentId**
  - **Description**: Unique, internal number that can change; assigned by the Cluster Manager.

- **agentName**
  - **Description**: Name defined by the host where the agent resides [numbers and/or letters].

- **status**
**Description:** Can be 1 or 0. Choose active or inactive agents only.

buildId

**Description:** Further restricts the returned agents to those running a specific build ID.

enabled

**Description:** Can be 1 or 0. Choose enabled or disabled agents only.

agents

**Description:** Specifies individual agents based on their host name and listening port using this format: `<host>[:<port>][:<agentKey>]`

**Result Tags**

agentName

**Description:** The name of the configured agent.

result

**Description:** The configuration result.

**Syntax**

cmtool setAgentDebug <level> [optionals...]

**Example**

cmtool setAgentDebug profile --agentName SOLAgent-4

Sets SOLAgent-4’s debug level to “profile”.

---

**Build Management**

This section describes build management-related requests.

**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.

**createBuildClass**

Creates a build class.

**Required Arguments**

buildClassName

**Description:** Name for the build class.

**Optional Arguments**

**Syntax**

```
tool createBuildClass <buildClassName> [optionals...]
```

**Example**

cmtool createBuildClass testclass

Create a build class named testclass.
Platform

minAgents

maxAgents

annotationLevels

Description: Comma-separated list of values that indicates which levels of information to include in the annotation file. The possible values are basic, env, history, file, lookup, waiting, or registry (Windows only). An annotation file is not created until you specify at least one annotation level.

Basic annotation includes annotation for the JobCache feature. (JobCache is an add-on option. It is available only on Linux and Solaris platforms and requires separately-purchased licensing. For more information about JobCache, see the ElectricAccelerator Electric Make User Guide at http://docs.electric-cloud.com/accelerator_doc/AcceleratorIndex.html.)

maxAgents

Description: Maximum number of agents that can be assigned to this build. The default is 64.

minAgents

Description: Minimum number of agents required for this build to run. The default is 2.

platform
**Description:** OS being used or supported. The possible values are Windows, Linux, or Solaris. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class. The default is that no platforms are specified.

**priority**

**Description:** Priority for builds in this class. You can use one of three levels of priority: high, normal, and low. The default is 120 (middle of the normal priority range). The priority can be adjusted up or down by 1-10 to “boost” the priority to give certain classes preference over other builds of the same priority level. Higher boost values mean greater preference.

The value must be a number in one of the following three ranges:

- 230 to 210 (high priority range). 220 is high priority with no boost
- 130 to 110 (normal priority range). 120 is normal priority with no boost
- 30 to 10 (low priority range). 20 is low priority with no boost

**Description:**

**annoUpload**

**Description:** Specifies whether to upload the annotation file to the Cluster Manager. The possible values are Y, 1, or true (upload) or N, 0, or false (do not upload). The default is N.

**jobcacheAllowed**

**Description:** Specifies whether the JobCache feature is allowed for this build class. The possible values are Y, 1, or true (allow) or N, 0, or false (do not allow). The default is N, except for the default build class and for build classes that exist when you upgrade to Accelerator 8.0. (JobCache is an add-on option. It is available only on Linux and Solaris platforms and requires separately-purchased licensing. For more information about JobCache, see the ElectricAccelerator Electric Make User Guide.)

**resourceRequest**

**Description:** Name of an existing resource. This requests a particular type of agent from the resource manager.

**Syntax**

cmtool createBuildClass <buildClassName> [optionals...]

**Example**

cmtool createBuildClass batch --minAgents 5 --maxAgents 12 --priority 30 --resource Request 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**

Creates a new build class comment.

**Required Arguments**

**buildClassId**

**Description:** A unique number assigned by the Cluster Manager for each build class. Use `getBuildClasses` to retrieve a list of build class IDs.
Chapter 2: API Requests

createBuildClassComment

Creates a comment for build class.

Optional Arguments
None

Syntax

cmtool createBuildClassComment <buildClassId> <text>

Example

cmtool createBuildClassComment 7 "This build class is for QA builds."
Creates a comment for build class 7.

decreaseBuildComment

Creates a new build comment.

Required Arguments

buildId

Description: A unique number assigned by the Cluster Manager for each build. Use getBuilds to retrieve a list of build IDs.

text

Description: The comment text.

Optional Arguments
None

Syntax

cmtool createBuildComment <buildId> <text>

Example

cmtool createBuildComment 1044 "This is our gold build for release 7.0"
Creates a comment for build 1044.

deleteBuild

Deletes a build, including all dependent records.

Required Arguments

buildId

Description: A unique number assigned by the Cluster Manager for each build. Use getBuilds to retrieve a list of build IDs.

Optional Arguments
None
**Syntax**
```
cmtool deleteBuild <buildId>
```

**Example**
```
cmtool deleteBuild 1037
```
Deletes build 1037.

### deleteBuildClass

Deletes a build class, including all dependent records.

**Required Arguments**

*buildClassId*

**Description:** 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

**Syntax**
```
cmtool deleteBuildClass <buildClassId>
```

**Example**
```
cmtool deleteBuildClass 7
```
Deletes build class 7.

### deleteBuildClasses

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*

**Description:** A SQL query used to limit the result set. See the possible values below. For a list of possible SQL values, see the `getBuildClasses` command.

**Syntax**
```
cmtool deleteBuildClasses [optionals...]
```

**Example**
```
cmtool deleteBuildClasses --filter "max_agents >20"
```
Deletes all build classes with more than 20 maximum agents.
deleteBuildClassComment

Deletes a build class comment.

Required Arguments

buildClassId

Description: A unique number assigned by the Cluster Manager for each build class.

commentId

Description: The unique key that identifies a comment.

Optional Arguments

None

Syntax

cmtool deleteBuildClassComment <buildClassId> <commentId>

Example

cmtool deleteBuildClassComment 6 1018

Deletes comment 1018 for build class 6.

deleteBuildComment

Deletes a build comment.

Required Arguments

buildId

Description: A unique number assigned by the Cluster Manager for each build.

commentId

Description: The unique key that identifies a comment. Use getBuildComments to retrieve a list of comment IDs.

Optional Arguments

None

Syntax

cmtool deleteBuildComment <buildId> <commentId>

Example

cmtool deleteBuildComment 1037 1019

Deletes build comment 1019 for build 1037.

deleteBuilds

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*

**Description:** SQL query used to limit the result set. For a list of possible SQL values, see the `getBuilds` command.

**Syntax**

cmtool deleteBuilds [optionals...]

**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

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

**Required Arguments**

*buildId*

**Description:** 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   userEmail
historyFile     userName
hostName        waitTime

Syntax

cmtool getBuild <buildId>

Example

cmtool getBuild 1000
Retrieves build 1000.

getBuilds

Retrieves a list of builds.

Required Arguments
None

Optional Arguments

filter

   Description: A SQL query used to limit the result set. See the possible values below.
   Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when
   using this argument for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single
   quotes.

maxResults

   Description: The maximum number of elements to run from a query.

firstResult

   Description: The starting index for the query result set.
   Note: This argument 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

   Description: A SQL order by clause. Used to specify ordering for the query result set.

profile
**Description:** Can be details or 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 this argument to details in order to print fields that are part of the details category.

### Result Tags and SQL Query Names

- **allocatedAgents**
  - **Description:** The number of currently assigned agents for this build.
  - SQL query name for **--filter** and **--order**: N/A

- **availableResults**
  - **Description:** This is a count of 'max' or 'first' results if **--maxResults** or **--firstResult** is specified.
  - SQL query name for **--filter** and **--order**: N/A

- **buildClassId**
  - **Description:** A unique number assigned by the Cluster Manager for each build class.
  - SQL query name for **--filter** and **--order**: build_class_id

- **buildClassName**
  - **Description:** A name assigned by the user for the build class.
  - SQL query name for **--filter** and **--order**: build_class_name

- **buildId**
  - **Description:** A unique number assigned by the Cluster Manager for each build.
  - SQL query name for **--filter** and **--order**: id

- **buildLogDir**
  - **Description:** The directory containing uploaded build logs.
  - SQL query name for **--filter** and **--order**: N/A

- **buildName**
  - **Description:** The build name that is the expanded build class tag.
  - SQL query name for **--filter** and **--order**: build_name

- **commandLine**
  - **Description:** The original command-line invocation of eMake.
  - SQL query name for **--filter** and **--order**: command_line

- **conflicts**
  - **Description:** The number of conflicts in the build.
  - SQL query name for **--filter** and **--order**: conflicts

- **cwd**
  - **Description:** The current working directory where eMake was invoked.
  - SQL query name for **--filter** and **--order**: cwd

- **duration**
  - **Description:** The number of milli-seconds the build has been running.
  - **Note:** duration for running builds is always 0.
  - SQL query name for **--filter** and **--order**: duration

- **effectiveAgentAlloc**
**Description:** 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:** The effectiveAgentAlloc for running builds is always 0.

SQL query name for --filter and --order: effective_agent_alloc

**emakeVersion**

**Description:** The eMake version used for this build.

SQL query name for --filter and --order: emake_version

**historyExists**

**Description:** True means the history file existed and was used by the build.

SQL query name for --filter and --order: history_exists

**historyFile**

**Description:** The name of the eMake history file.

SQL query name for --filter and --order: history_file

**hostName**

**Description:** The name of the machine where eMake was invoked.

SQL query name for --filter and --order: host_name

**ipAddress**

**Description:** The IP address of the machine where eMake was invoked.

SQL query name for --filter and --order: ip_address

**jobCount**

**Description:** The total number of jobs that ran for the build.

**Note:** job_count for running builds is always 0.

SQL query name for --filter and --order: job_count

**lastRequestTime**

**Description:** The last time eMake requested agents for this build.

SQL query name for --filter and --order: N/A

**maxAgents**

**Description:** The maximum number of agents to request for this build.

SQL query name for --filter and --order: max_agents

**minAgents**

**Description:** The minimum number of agents required for this build to run.

SQL query name for --filter and --order: min_agents

**osUserName**

**Description:** The OS-level name for the user who started eMake.

SQL query name for --filter and --order: os_user_name

**platform**

**Description:** 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.

SQL query name for --filter and --order: platform

**priority**
**Description:** 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.
SQL query name for --filter and --order: priority

**resourceRequest**

**Description:** A request to the resource manager for a particular type of agent.
SQL query name for --filter and --order: resource_request

**result**

**Description:** 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.
SQL query name for --filter and --order: result

**requestedAgents**

**Description:** The number of agents eMake requested.
SQL query name for --filter and --order: N/A

**startTime**

**Description:** The time the build was started.
SQL query name for --filter and --order: start_time

**userLabel**

**Description:** The user-supplied label (via the eMake command-line), attached to the build.
SQL query name for --filter and --order: user_label

**userName**

**Description:** The unique name of the user.
SQL query name for --filter and --order: user_name

**waitTime**

**Description:** The number of seconds eMake was stalled because it had to wait for agents.
*Note:* wait_time for running builds is always 0.
SQL query name for --filter and --order: wait_time

**Syntax**

cmtool getBuilds [optionals...]

**Example**

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

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

**getBuildComments**

Retrieves a list of related build comments.

**Required Arguments**

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

**Optional Arguments**

- `commentId`
  - **Description**: The unique key that identifies a comment.

**Result Tags**

- `commentId`
  - **Description**: The unique key that identifies a comment.

- `createTime`
  - **Description**: The time when the item was created.

- `lastModifiedBy`
  - **Description**: The user who last modified the item.

- `modifyTime`
  - **Description**: The time when the item was last modified.

- `text`
  - **Description**: The text of the item.

**Syntax**

cmtool getBuildComments <buildId> [optionals...]

**Example**

cmtool getBuildComments 1000 --commentId 1039

Retrieves comment 1039 for build 1000.

**getBuildClass**

Finds a build class with full detail by its ID.

**Required Arguments**

- `buildClassId`
  - **Description**: 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**

**Syntax**

cmtool getBuildClass <buildClassId>

**Example**

cmtool getBuildClass 1
Retrieves build class 1.

**getBuildClasses**

Retrieves a list of build classes with limited detail.

**Required Arguments**
None

**Optional Arguments**

**filter**

**Description:** A SQL query used to limit the result set. See the possible values below.
**Note:** There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using this argument for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

**maxResults**

**Description:** The maximum number of elements to run from a query.

**firstResult**

**Description:** The starting index for the query result set.
**Note:** This argument 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**

**Description:** SQL order by clause. Used to specify ordering for the query result set.

**profile**

**Description:** Can be details or 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**

**annotationLevels**

**Description:** Annotation choices to include in the annotation file. Possible values are basic, history, file, lookup, and waiting.
SQL query name for --filter and --order: annotation_levels
annoUpload

**Description:** If set to true, the annotation file is uploaded to Cluster Manager.
SQL query name for --filter and --order: anno_upload

availableResults

**Description:** This is a count of 'max' or 'first' results if --maxResults or --firstResult is specified.
SQL query name for --filter and --order: N/A

buildClassId

**Description:** A unique number assigned by the Cluster Manager for each build class.
SQL query name for --filter and --order: id

buildClassName

**Description:** A name assigned by the user for the build class.
SQL query name for --filter and --order: build_class_name

defaultClass

**Description:** If set, this is the default build class and cannot be deleted.
SQL query name for --filter and --order: default_class

maxAgents

**Description:** The maximum number of agents to request for this build.
SQL query name for --filter and --order: max_agents

minAgents

**Description:** The minimum number of agents required for this build to run.
SQL query name for --filter and --order: min_agents

notifyOnBuildEnd

**Description:** If set to true, the currently logged-in user will receive an email when the build is finished.
SQL query name for --filter and --order: notify_on_build_end

platform

**Description:** 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.
SQL query name for --filter and --order: platform

priority

**Description:** 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.
SQL query name for --filter and --order: priority

resourceRequest

**Description:** A request to the resource manager for a particular type of agent.
resource_request

tagDefinition

**Description:** A format string that defines the resulting build name.
SQL query name for --filter and --order: tag_definition

**Syntax**
cmtool getBuildClasses [optionals...]
Example

cmtool getBuildClasses --filter "min_agents <5"

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

**getBuildClassComments**

Retrieves a list of related build class comments.

**Required Arguments**

buildClassId

**Description:** A unique number assigned by the Cluster Manager for each build class. You can use `getBuildClasses` to retrieve a list of build class IDs.

**Optional Arguments**

commentId

**Description:** The unique key that identifies a comment.

**Result Tags**

 commentaire:

**Description:** The unique key that identifies a comment.

createTime

**Description:** The time when the item was created.

lastModifiedBy

**Description:** The user who last modified the item.

modifyTime

**Description:** The time when the item was last modified.

text

**Description:** The text of the item.

**Syntax**

cmtool getBuildComments <buildId> [optionals...]

**Example**

cmtool getBuildComments 1000 --commentId 1039

Retrieves comment 1039 for build 1000.

**Example**

cmtool getBuildClassComments 12

Retrieves all build class comments for build class 12.

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

**Required Arguments**

`groupBy`

- **Description**: Can be `hostName`, `ipAddress`, or `userName`.

**Optional Arguments**

`filter`

- **Description**: A SQL query used to limit the result set. See the possible values below.
- **Note**: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using this argument for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

`order`

- **Description**: A SQL order by clause. Used to specify ordering for the query result set.

**Result Tags and SQL Query Names**

`duration`

- **Description**: The total number of milli-seconds of all builds, filtered by the value specified in the `groupBy` argument.
- SQL query name for `--filter` and `--order`: N/A `duration`

`entryName`

- **Description**: The value specified in the `groupBy` argument. If `groupBy` is "userName", the entry name is the user name.
- SQL query name for `--filter` and `--order`: N/A

`numOfBuilds`

- **Description**: The number of builds.
- SQL query name for `--filter` and `--order`: N/A

`waitTime`

- **Description**: The number of seconds eMake was stalled because it had to wait for agents.
- SQL query name for `--filter` and `--order": wait_time`

`workload`

- **Description**: The total number of seconds used by the agents for all of the filtered builds.
- SQL query name for `--filter` and `--order": workload`

**Syntax**

`cmtool getBuildUserStats <groupBy> [optionals...]`

**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.
Modifies a build.

**Required Arguments**

buildId

*Description:* A unique number assigned by the Cluster Manager for each build.

priority

*Description:* Can be Low or Normal, but *not* High.

**Optional Arguments**

None

**Syntax**

cmtool modifyBuild <buildId> <priority>

**Example**

cmtool modifyBuild 1137 20

Changes build 1137 to priority 20.

**modifyBuildClass**

Modifies a build class.

**Required Arguments**

buildClassId

*Description:* Unique number assigned by the Cluster Manager for the build class. You can use *getBuildClasses* to retrieve a list of build class IDs.

**Optional Arguments**

buildClassName

*Description:* User-defined name of the build class.

tagDefinition

*Description:* Format string that defines the resultant build name. The default is default_%GC%_%DATE%. This string generally consists of a generic build name appended with build-specific data that you construct from the following variables:

- **GC**—Globally unique number (Global Counter)
- **LC**—Number unique to the build class (Local Counter; the build serial number within the class)
- **BUILD_CLASS**—User-defined build class name
- **BUILD_CLASS_ID**—System-generated number that the Cluster Manager uses to identify each class
- **USER_NAME**—Name of the user who invoked eMake
- **MACHINE_NAME**—Name of the machine where eMake was invoked
- **USER_BUILD_LABEL**—Label specified at the eMake command line. For example, --emake-build-label=my_build
BUILD_OS_ID—Operating system ID under which the build was invoked (0 = undefined, 1 = Windows, 2 = Solaris, and 3 = Linux)

DATE—Build start date and time using variables Y, y, m, d, H, M, and S (for example, 2005-01-18 10:14:32 is 20050118101432)

Y—Year at build start time (YYYY)
y—Year at build start time (YY)

m—Sequential month number at build start time (1-12)
d—Sequential day of month at build start time (1-31)
H—Hour of the day at build start time (0-23)
M—Minutes at build start time (0-59)
S—Seconds at build start time (0-60)
a—Abbreviated day of week at build start time (Wed)
A—Full name day of week at build start time (Wednesday)
b—Abbreviated month name at build start time (Aug)
B—Full month name at build start time (August)
c—Build start date and time using the variables A, B, d, H, M, S, and Y (for example, 2005-01-18 10:14:32 means 18/01/05 10:14:32)

For information about constructing tag definitions, see the "Tag Definitions" section in Chapter 4, Additional Electric Make Settings and Features, of the ElectricAccelerator Electric Make User Guide.

annotationLevels

Description: Comma-separated list of values that indicates which levels of information to include in the annotation file. The possible values are basic, env, history, file, lookup, waiting, or registry (Windows only). An annotation file is not created until you specify at least one annotation level.

Basic annotation includes annotation for the JobCache feature. (JobCache is an add-on option. It is available only on Linux and Solaris platforms and requires separately-purchased licensing. For more information about JobCache, see the ElectricAccelerator Electric Make User Guide.)

maxAgents

Description: Maximum number of agents that can be assigned to this build. The default is 64.

minAgents

Description: Minimum number of agents required for this build to run. The default is 2.

platform

Description: OS being used or supported. The possible values are Windows, Linux, or Solaris. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class. The default is that no platforms are specified.

priority

Description: Priority for builds in this class. You can use one of three levels of priority: high, normal, and low. The default is 120 (middle of the normal priority range). The priority can be adjusted up or down by 1-10 to “boost” the priority to give certain classes preference over other builds of the same priority level. Higher boost values mean greater preference.

The value must be a number in one of the following three ranges:
- 230 to 210 (high priority range). 220 is high priority with no boost
- 130 to 110 (normal priority range). 120 is normal priority with no boost
- 30 to 10 (low priority range). 20 is low priority with no boost

**annoUpload**

**Description:** Specifies whether to upload the annotation file to the Cluster Manager. The possible values are Y, 1, or true (upload) or N, 0, or false (do not upload). The default is N.

**jobcacheAllowed**

**Description:** Specifies whether the JobCache feature is allowed for this build class. The possible values are Y, 1, or true (allow) or N, 0, or false (do not allow). The default is N, except for the default build class and for build classes that exist when you upgrade to Accelerator 8.0. (JobCache is an add-on option. It is available only on Linux and Solaris platforms and requires separately-purchased licensing. For more information about JobCache, see the ElectricAccelerator Electric Make User Guide.)

**resourceRequest**

**Description:** Name of an existing resource. This requests a particular type of agent from the resource manager.

**Syntax**

cmtool modifyBuildClass <buildClassId> [optionals...]

**Example**

cmtool modifyBuildClass 1 --annoUpload true

Changes build class 1 to upload annotation files.

**modifyBuildComment**

Modifies a build comment.

**Required Arguments**

- **buildId**
  
  **Description:** A unique number assigned by the Cluster Manager for each build.

- **commentId**
  
  **Description:** The unique key that identifies a comment. Use `getBuildComments` to retrieve a list of comment IDs.

- **text**
  
  **Description:** The text of the item.

**Optional Arguments**

None

**Syntax**

cmtool modifyBuildComment <buildId> <commentId> <text>

**Example**

cmtool modifyBuildComment 16975 1137 "This is not a usable build"
**setDatabaseConfiguration**

Modifies database configuration settings.

**Required Arguments**

- databaseName
  - **Description:** The database instance name.

- databaseType
  - **Description:** The database type. Can be mariadb, mysql, oracle, or sqlserver.

- hostName
  - **Description:** Machine name where the database is installed.

- port
  - **Description:** Database port number.

- userName
  - **Description:** Unique name of the user that is used to access the database.

- password
  - **Description:** Secret value used to identify an account for a particular user.

**Optional Arguments**

None

**Syntax**

```bash
cmtool setDatabaseConfiguration <databaseName> <databaseType> <hostName> <port> <userName> <password>
```

**stopBuild**

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

**Required Arguments**

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

**Optional Arguments**

None

**Syntax**

```bash
cmtool stopBuild <buildId>
```

**Example**

```bash
cmtool stopBuild 16937
```
Cluster Management

This section describes cluster management-related requests.

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.

createServerComment

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

Required Arguments

text

Description: The text of the comment.

Optional Arguments
None

Syntax

cmtool createServerComment <text>

Example

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

Creates the server comment "cluster needs more servers to handle production builds".

deleteLicense

Deletes a license.

Required Arguments

productName

Description: The name of the license, which is ElectricAccelerator.

featureName

Description: Feature name of the license, which is Server.

Optional Arguments
None

Syntax

cmtool deleteLicense <productName> <featureName>

Example

cmtool deleteLicense ElectricAccelerator Server

Deletes the license stored in the server.

deleteMessage
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

Description: The numeric value that uniquely identifies each message.

Optional Arguments

None

Syntax

cmtool deleteMessage <messageId>

Example

cmtool deleteMessage 501

Deletes the message with ID 501.

deleteMessages

Deletes a set of messages, including all dependent records.

Required Arguments

None

Optional Arguments

filter

Description: A SQL query used to limit the result set. For a list of possible SQL values, see the getMessages command.

Syntax

cmtool deleteMessages [optionals...]

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

Deletes a server comment.

Required Arguments

commentId

Description: The unique key that identifies a comment.
Optional Arguments
None

Syntax
cmtool deleteServerComment <commentId>

Example
cmtool deleteServerComment 1396
Deletes the server comment with ID 1396.

exportData
Exports Cluster Manager data to a file. This is a full database dump, which might take substantial time for a large database.

Note: Manual migration using the exportData and importData commands is recommended only to replicate data between Cluster Manager instances running the same version of the Cluster Manager. For example, you could use it to make a backup dump and then restore it to a new instance of the Cluster Manager of the same version.

Required Arguments
cFileName

Description: Target file name or path. If you use a file name, the destination is the current working directory of the Java process (/opt/ecloud/i686_Linux or C:\ECloud\i686_win32 by default). If you use a path, the Cluster Manager Java user (eacmuser) must have execute and write access to that path.

Optional Arguments
None

Syntax
cmtool exportData <filename>

Example
cmtool exportData fileabc

getLicense
Retrieves information for one license.

Required Arguments
cproductName

Description: The name of the license, which is ElectricAccelerator.

cfeatureName

Description: The name of the feature, which is Server.

Optional Arguments
None
Chapter 2: API Requests

Syntax

cmtool getLicense <productName> <featureName>

Example

cmtool getLicense ElectricAccelerator Server

getLicenses

Retrieves all license data.

Required Arguments
None

Optional Arguments
None

Syntax

cmtool getLicenses

Example

cmtool getLicenses

getMessage

Retrieves a particular message.

Required Arguments

messageId

Description: 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

Syntax

cmtool getMessage <messageId>

Example

cmtool getMessage 47
**getMessages**

Retrieves a list of messages

**Required Arguments**

None

**Optional Arguments**

*filter*

**Description:** A SQL query used to limit the result set. See the possible values below.

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

*maxResults*

**Description:** The maximum number of elements to return from a query.

*firstResult*

**Description:** The starting index for the query result set.

**Note:** This argument 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*

**Description:** A SQL order by clause. Used to specify ordering for the query result set.

*profile*

**Description:** Can be details or 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**

*agentId*

**Description:** A unique, internal number assigned to each agent by the Cluster Manager; this number can change.

SQL query name for **--filter** and **--order: N/A**

*agentName*

**Description:** A name defined by the host where the agent resides [numbers and/or letters].

SQL query name for **--filter** and **--order: agent_name**

*buildId*

**Description:** A unique number assigned by the Cluster Manager for each build.

SQL query name for **--filter** and **--order: build_id**

*buildName*

**Description:** The build name that is the expanded build class tag.

SQL query name for **--filter** and **--order: N/A**

*createTime*

**Description:** The time when the item was created.

SQL query name for **--filter** and **--order: create_time**
messageId

**Description:** The numeric value that uniquely identifies each message.
SQL query name for --filter and --order:id

severity

**Description:** 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
SQL query name for --filter and --order:severity

text

**Description:** The text of the item.
SQL query name for --filter and --order:text

**Syntax**

cmtool [optionals...]

**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**

Retrieves resource usage statistics.

**Required Arguments**

None

**Optional Arguments**

**filter**

**Description:** A SQL query used to limit the result set. See the possible values below.
**Note:** There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using this argument for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

**maxResults**

**Description:** The maximum number of elements to return from a query.

**firstResult**

**Description:** The 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**

**Description:** A SQL order by clause. Used to specify ordering for the query result set.

**profile**
**Description:** Can be details or 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**

**agentClusterShortage**
- **Description:** The difference between the maximum number of agents requested by all builds and the number of agents that were assigned.
- SQL query name for `--filter` and `--order:agent_cluster_shortage`

**agentDemand**
- **Description:** The total maximum number of requests for agents by all running builds.
- SQL query name for `--filter` and `--order:agent_demand`

**agentLicenseShortage**
- **Description:** The difference between the maximum request for agents by all builds and the number of agents the license allows.
- SQL query name for `--filter` and `--order:agent_license_shortage`

**agentsAvailable**
- **Description:** The total number of active agents in the cluster.
- SQL query name for `--filter` and `--order:agents_available`

**agentsInUse**
- **Description:** The total number of agents assigned to builds.
- SQL query name for `--filter` and `--order:agents_in_use`

**availableResults**
- **Description:** This is a count of 'max' or 'first' results if `--maxResults` or `--firstResult` is specified.
- SQL query name for `--filter` and `--order:N/A`

**buildsDuration**
- **Description:** The average amount of time the current builds have been running.
- SQL query name for `--filter` and `--order:builds_duration`

**buildsRunning**
- **Description:** Average number of simultaneous builds running during a specific time period.
- SQL query name for `--filter` and `--order:builds_running`

**createTime**
- **Description:** The time when the item was created.
- SQL query name for `--filter` and `--order:create_time`

**duration**
- **Description:** The number of milli-seconds the build has been running.
- SQL query name for `--filter` and `--order:duration`

**resourceName**
- **Description:** 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.
- SQL query name for `--filter` and `--order:resource_name`

**resourceStatId**
**Description**: The resource ID number that uniquely identifies every resource. SQL query name for `--filter` and `--order:id`

**Syntax**

cmtool getResourceStats [optionals...]

**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**

Retrieves server configuration.

**Required Arguments**

None

**Optional Arguments**

None

**Result Tags**

agentAllocationPolicy
  **Description**: Defined as either exclusive or shared.

agentLockTimerSec
  **Description**: When jobs run beyond this number of seconds, the agent should be locked.

badAgents
  **Description**: The number of enabled agents with a bad status.

disabledAgents
  **Description**: The number of disabled agents.

eMailInterval
  **Description**: The number of minutes between email notifications.

eMailItemLimit
  **Description**: Maximum number of messages per email notification.

goodAgents
  **Description**: The number of enabled agents with a good status.

logDaysToKeep
  **Description**: The number of days to keep message log entries.

lsfAvailable
  **Description**: True if LSF is available to the Cluster Manager.

mailFrom
**Description**: The value to use in the From header element.

**mailPrefix**

**Description**: The string used to prefix subject lines.

**maxAgents**

**Description**: The maximum number of agents to request for this build.

**maxClockSkew**

**Description**: The maximum clock skew (in seconds) allowed between the eMake client and agents in the cluster.

**minAgents**

**Description**: The minimum number of agents required for this build to run.

**preemptionPolicy**

**Description**: The allocation preemption policy.

**priority**

**Description**: 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**

**Description**: The type of resource manager that Cluster Manager should employ.

**resourceStatInterval**

**Description**: In minutes, the interval to collect stats on resource usage.

**resourceStatKeep**

**Description**: The number of minutes of resource usage statistics to keep.

**runningBuilds**

**Description**: The number of incomplete builds in the system.

**Syntax**

cmtool getServer

**Example**

cmtool getServer

g**getServerComments**

Retrieves a list of related server comments.

**Required Arguments**

None

**Optional Arguments**

commentId
Description: The unique key that identifies a comment.

Result Tags

commentId
  Description: The unique key that identifies a comment.

createTime
  Description: The time when the item was created.

lastModifiedBy
  Description: The user who last modified the item.

modifyTime
  Description: The time when the item was last modified.

text
  Description: The text of the item.

Syntax

cmtool getServerComments [optionals...]

Example

cmtool getServerComments

Returns all comments related to the server.

getVersion

Retrieves server version information.

Required Arguments

None

Optional Arguments

None

Result Tags

label
  Description: The Electric Cloud build label for the server.

protocolVersion
  Description: The server protocol version.

schemaVersion
  Description: The server database schema version.

version
  Description: The string identifying a component version.
Syntax

cmtool getVersion

Example

cmtool getVersion

importData
Imports Cluster Manager data from a file. This command imports a full database dump, which might take substantial time for a large database.

Notes:

- You must manually delete any old or unused agents from the agents list.
- You must update the license file after importing it, if it has expired.
- Manual migration using the exportData and importData commands is recommended only to replicate data between Cluster Manager instances running the same version of the Cluster Manager. For example, you could use it to make a backup dump and then restore it to a new instance of the Cluster Manager of the same version.

Required Arguments

fileName

Description: Name of the file to import. The file path is relative to the current working directory of the Java process (/opt/ecloud/i686_Linux or C:\ECloud\i686_win32 by default).

Optional Arguments

None

Syntax

cmtool importData <filename>

Example

cmtool importData fileabc

importLicenseData
Imports one or more licenses.

Required Arguments

licenseFile

Description: Name of the file containing the license with the path.

Optional Arguments

None

Syntax


cmtool importLicenseData <licenseFile>
Example
cmtool importLicenseData ./license.xml

logMessage
Create a custom message on the Cluster Manager Messages page.

Required Arguments
text
   Description: 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
      Description: Can be Debug, Info, Warning, or Error. You can also use 0, 1, 2, or 3.

   buildId
      Description: The message applies to this specified build only.

   agentName
      Description: The message applies to this specified agent name only.

Syntax
cmtool logMessage <text> [optionals...]

Example
cmtool logMessage "some text"

modifyServer
Modify the server configuration.

Required Arguments
None

Optional Arguments

   priority
      Description: 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
      Description: The number of minutes between email notifications.

   emailItemLimit
      Description: The maximum number of messages per email notification.
agentAllocationPolicy

**Description:** Can be exclusive or shared. Exclusive means all agents on a specific machine are assigned to the same build. Shared means all agents on the same machine can be assigned to different builds. This policy requires that eMake client and agent machines have synchronized clocks. You must choose this policy if using Priority Pools.

preemptionPolicy

**Description:** The allocation preemption policy.

maxClockSkew

**Description:** The maximum clock skew (in seconds) allowed between the eMake client and agents in the cluster.

maxAgents

**Description:** The maximum number of agents to request for this build.

minAgents

**Description:** The minimum number of agents required for this build to run.

resourceManagerType

**Description:** Can be none, ea, lsf, cloud, or prioritypool. Define which resource manager Cluster Manager should employ.

mailFrom

**Description:** The value to use in the From header element.

mailPrefix

**Description:** The string used to prefix subject lines.

logDaysToKeep

**Description:** The number of days to keep message log entries.

resourceStatInterval

**Description:** In minute units, this is the interval to collect statistics on resource usage.

resourceStatKeep

**Description:** The number of days of Resource usage statistics to keep.

wideDeepAllocationPolicy

**Description:** Can be deep or 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.

Syntax

cmtool modifyServer [optionals...]

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

Modifies a server comment.

Required Arguments

commentId
Description: The unique key that identifies a comment.

text
Description: The comment text.

Optional Arguments
None

Syntax

cmtool modifyServerComment <commentId> <text>

Example

cmtool modifyServerComment 1178 "Server is fine"

shutdownServer

Stops the server.

IMPORTANT: Use with caution.

Required Arguments
None

Optional Arguments

restart
Description: Restart the server. Can be true or false.

Syntax

cmtool shutdownServer [optionals...]

Example

cmtool shutdownServer

testAgents

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

Required Arguments

None

Optional Arguments

agentId
**Description:** A unique, internal number that can change; assigned by the Cluster Manager.

**agentName**

**Description:** The name defined by the host where the agent resides [numbers and/or letters].

**filter**

**Description:** A SQL query used to limit the result set. For a list of possible SQL values, see the `getAgents` command

**Syntax**

`cmtool testAgents [optionals...]`

**Example**

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

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

### Reporting

This section describes reporting-related requests.

**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.

**createFilter**

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**

**Description:** 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**

**Description:** A short string that uniquely identifies the filter.

**filterQuery**

**Description:** A SQL order by clause for the associated table.

**Optional Arguments**

**global**

**Description:** Can be true or false. If true, this is a globally visible filter. This parameter is required for global filters.

**order**

**Description:** A SQL order by clause. Used to specify ordering for the query result set.

**Syntax**

`cmtool createFilter <tableName> <filterName> <filterQuery> [optionals...]`
Example

cmtool createFilter ec_agents linuxAgents ""platform = ‘linux’" --global true

Creates a global filter that selects Linux agents only.

deleteFilter

Deletes a named filter for a specific table.

Required Arguments

tableName

**Description:** 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

**Description:** A short string that uniquely identifies the filter.

Optional Arguments

global

**Description:** Can be true or false. If true, this is a globally visible filter. This parameter is required for global filters.

Syntax

cmtool deleteFilter <tableName> <filterName> [optionals...]

Example

cmtool deleteFilter ec_agents linuxAgents --global true

getCurrentServerLoad

Retrieves information about the current resource load.

Required Arguments

None

Optional Arguments

None

Result Tags

agentsAvailable

**Description:** The total number of active agents in the cluster.

agentClusterShortage

**Description:** The difference between the maximum number of agents requested by all builds and the number of agents that were assigned.

agentDemand

**Description:** The total maximum number of requests for agents by all running builds.
agentLicenseShortage
  
  **Description:** The difference between the maximum request for agents by all builds and the number of agents the license allows.

agentsInUse
  
  **Description:** The total number of agents assigned to builds.

buildsDuration
  
  **Description:** The average amount of time the current builds have been running.

buildsRunning
  
  **Description:** Average number of simultaneous builds running during a specific time period.

createTime
  
  **Description:** The time when the item was created.

duration
  
  **Description:** The number of milli-seconds the build has been running.

resourceName
  
  **Description:** 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
  
  **Description:** The resource ID number that uniquely identifies every resource.

Example

```
cmtool getCurrentServerLoad
```

getFilter
  
  Retrieves a named filter for a specific table.

**Required Arguments**

tableName
  
  **Description:** 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
  
  **Description:** A short string that uniquely identifies the filter.

**Optional Arguments**

global
  
  **Description:** Can be true or false. If true, this is a globally visible filter. This parameter is required for global filters.

**Syntax**

```
cmtool getFilter <tableName> <filterName> [optionals...]
```
Example

cmtool getFilter ec_agent agentFilter

getFilters

Retrieves a list of saved filters for the current user.

**Required Arguments**

None

**Optional Arguments**

- **filter**
  
  **Description:** The query to use to limit the result set. For a list of possible SQL values, see the `getAgents` command.

- **maxResults**
  
  **Description:** The maximum number of elements to return from a query.

- **firstResult**
  
  **Description:** The starting index for the query result set. The argument 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**
  
  **Description:** A SQL order by clause. Used to specify ordering for the query result set.

**Syntax**

cmtool getFilters [optionals...]

**Example**

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

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

modifyFilter

Updates a named filter for a specific table.

**Required Arguments**

- **tableName**
  
  **Description:** 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**
  
  **Description:** A short string that uniquely identifies the filter.

- **filterQuery**
  
  **Description:** A SQL order by clause for the associated table.
Optional Arguments

- **global**
  
  **Description:** Can be true or false. If true, this is a globally visible filter. This parameter is required for global filters.

- **order**
  
  **Description:** A SQL order by clause. Used to specify ordering for the query result set.

**Syntax**

cmtool modifyFilter <tableName> <filterName> <filterQuery> [optionals...]

**Example**

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

**User Management**

This section describes user management-related requests.

**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.

**addGroupMember**

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

**Required Arguments**

- **groupName**
  
  **Description:** The unique name of the group.

- **userName**
  
  **Description:** The unique name of the user.

**Optional Arguments**

None

**Syntax**

cmtool addGroupMember <groupName> <userName>

**Example**

cmtool addGroupMember DevGroupA ec123

Adds user 'ec123' to group DevGroupA.

**changeOwnUser**

Modifies the settings for the currently logged-in user.

**Required Arguments**

- **userName**
  
  **Description:** The unique name of the user.
Optional Arguments

fullUserName
   Description: The real world name of the user.

email
   Description: The associated user email address.

password
   Description: The password for a particular user.

passwordFile
   Description: The path to a password file. If --password is also specified, --passwordFile overrides its value in the command line.

Syntax
cmtool <userName> [optionals...]

Example
cmtool ec123 --fullUserName "Mary Smith"

createGroup

Creates a new local group.

Required Arguments

groupName
   Description: The unique name of the group to create.

Optional Arguments
None

Syntax
cmtool createGroup <groupName>

Example
cmtool createGroup DevGroupA

createUser

Creates a new local user.

Required Arguments

userName
   Description: The unique name of the user.

password
   Description: The password for a particular user.
**Optional Arguments**

fullUserName

**Description:** The real world name of the user.

e-mail

**Description:** The associated user email address.

passwordFile

**Description:** The path to a password file. If `--password` is also specified, `--passwordFile` overrides its value in the command line.

**Syntax**

cmtool createUser <userName> <password> [optionals...]

**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**

Deletes a local group.

**Required Arguments**

groupName

**Description:** The unique name of the group.

**Optional Arguments**

None

**Syntax**

cmtool deleteGroup <groupName>

**Example**

cmtool deleteGroup DevGroupA

Removes the ‘DevGroupA’ group from the Cluster Manager.

**deleteUser**

Deletes a local user.

**Required Arguments**

userName
**Description:** The unique name of the user.

**Optional Arguments**
None

**Syntax**
cmtool deleteUser <userName>

**Example**
cmtool deleteUser ec123

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

**Required Arguments**
None

**Optional Arguments**
None

**Result Tags**
- **entityName**
  - **Description:** A user or group name in an access entry.
- **permissions**
  - **Description:** The list of permission flags for a particular entity.

**Example**
cmtool getAccessEntries

### getGroupMembers
Retrieves a list of users in a specific group.

**Required Arguments**
- **groupName**
  - **Description:** The unique name of the group.

**Optional Arguments**
None

**Result Tags**
- **userName**
  - **Description:** The unique name of the user.

**Example**
cmtool getGroupMembers
Retrieves a list of user name elements.

**getGroups**

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

**Required Arguments**

- **userName**
  
  **Description:** The unique name of the user.

**Optional Arguments**

- **local**
  
  **Description:** Can be true or false. If true, returns local users only.

**Result Tags**

- **groupName**
  
  **Description:** The unique name of the group.

- **mutable**
  
  **Description:** True if the associated user or group record is modifiable.

- **providerName**
  
  **Description:** The human-readable name configured for the directory provider of a specific user or group.

**Syntax**

cmtool getGroups [optionals...]

**Example**

cmtool getGroups

Returns a list of groupInfo elements.

**getEffectivePermissions**

Retrieves the permissions for the currently logged-in user.

**Required Arguments**

None

**Optional Arguments**

None

**Result Tags**

- **permissions**
  
  **Description:** The list of permission flags for a particular entity.
## Possible Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgentsDelete</td>
<td>MaintenanceWrite</td>
</tr>
<tr>
<td>AgentsRead</td>
<td>MessageLogDelete</td>
</tr>
<tr>
<td>AgentsWrite</td>
<td>MessageLogRead</td>
</tr>
<tr>
<td>BuildsDelete</td>
<td>MessageLogWrite</td>
</tr>
<tr>
<td>BuildsRead</td>
<td>ReportsDelete</td>
</tr>
<tr>
<td>BuildsWrite</td>
<td>ReportsRead</td>
</tr>
<tr>
<td>ClassesDelete</td>
<td>ReportsWrite</td>
</tr>
<tr>
<td>ClassesRead</td>
<td>ResourcesDelete</td>
</tr>
<tr>
<td>ClassesWrite</td>
<td>ResourcesRead</td>
</tr>
<tr>
<td>EMakeImpersonate</td>
<td>ResourcesWrite</td>
</tr>
<tr>
<td>EMakeInvoke</td>
<td>ServerAccess</td>
</tr>
<tr>
<td>MaintenanceDelete</td>
<td>UserModify</td>
</tr>
<tr>
<td>MaintenanceRead</td>
<td></td>
</tr>
</tbody>
</table>

## Example

```shell
cmtool getEffectivePermissions
```

Retrieves the permissions for the currently logged-in user.

## getPermissions

Retrieves permissions for a particular user or group.

### Required Arguments

- **principalType**
  - **Description:** Can be user or group.

- **entityName**
  - **Description:** A user or group name in an access entry.

### Optional Arguments

- None

### Result Tags

- **permissions**
  - **Description:** The list of permission flags for a particular entity.
**Possible Results**

<table>
<thead>
<tr>
<th>Agent operation</th>
<th>Maintenance operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgentsDelete</td>
<td>MaintenanceWrite</td>
</tr>
<tr>
<td>AgentsRead</td>
<td>MessageLogDelete</td>
</tr>
<tr>
<td>AgentsWrite</td>
<td>MessageLogRead</td>
</tr>
<tr>
<td>BuildsDelete</td>
<td>MessageLogWrite</td>
</tr>
<tr>
<td>BuildsRead</td>
<td>ReportsDelete</td>
</tr>
<tr>
<td>BuildsWrite</td>
<td>ReportsRead</td>
</tr>
<tr>
<td>ClassesDelete</td>
<td>ReportsWrite</td>
</tr>
<tr>
<td>ClassesRead</td>
<td>ResourcesDelete</td>
</tr>
<tr>
<td>ClassesWrite</td>
<td>ResourcesRead</td>
</tr>
<tr>
<td>EMakeImpersonate</td>
<td>ServerAccess</td>
</tr>
<tr>
<td>EMakeInvoke</td>
<td></td>
</tr>
<tr>
<td>MaintenanceDelete</td>
<td>UserModify</td>
</tr>
<tr>
<td>MaintenanceRead</td>
<td></td>
</tr>
</tbody>
</table>

**Syntax**

```cmtool getPermissions <principalType> <entityName>```

**Example**

```cmtool getPermissions group DevGroupA```

Retrieves permissions for group DevGroupA.

**getUser**

Finds a specific user known to the server.

**Required Arguments**

- `userName`

  **Description**: The unique name of the user.

**Optional Arguments**

None

**Result Tags**

- `email`

  **Description**: The associated user email address.

- `fullUserName`

  **Description**: The real world name of the user.

- `groupName`

  **Description**: The unique name of the group.

- `mutable`

  **Description**: True if the associated user or group record is modifiable.

- `providerName`

  **Description**: The human-readable name configured for the directory provider of a specific user or group.

- `userName`
**Description:** The unique name of the user.

**Syntax**

```
cmtool getUser <userName> [optionals...]
```

**Example**

```
cmtool getUser ec123
```

Retrieves the attributes for user ec123.

---

**getUsers**

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

**Required Arguments**

None

**Optional Arguments**

- `pattern`

  **Description:** 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`

  **Description:** Can be true or false. If true, returns local users only.

**Result Tags**

See `getUser` for descriptions.

- `email`
- `fullUserName`
- `mutable`
- `providerName`
- `userName`

---

**getUserSettings**

Retrieves settings for the currently logged-in user.

**Required Arguments**

None

**Optional Arguments**

None
Example

cmtool getUserSettings

login

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

Description: The unique name of the user.

password

Description: The password for a particular user.

Optional Arguments

passwordFile

Description: The path to a password file. If --password is also specified, --passwordFile overrides its value in the command line.

Result Tags

sessionId

Description: This is a session "cookie."

Syntax

cmtool login <userName> <password> [optionals...]

Example

cmtool login ecl23 bobs

Logs in a user named “ecl23” 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

Logs out of the client session.

Required Arguments

None

Optional Arguments

None

modifyGroup

Modifies a local group.
Required Arguments

groupName
   Description: The unique name of the group.

Optional Arguments

newName
   Description: The new group name.

Syntax

cmtool modifyGroup <groupName> [optionals...]

Example

cmtool modifyGroup DevGroupA --newName GroupDevA

modifyUser

Modifies a local user.

Required Arguments

userName
   Description: The unique name of the user.

Optional Arguments

fullUserName
   Description: The real world name of the user.

email
   Description: The associated user email address.

password
   Description: The password for a particular user.

passwordFile
   Description: The path to a password file. If --password is also specified, --passwordFile overrides its value in the command line.

Syntax

cmtool modifyUser <userName> [optionals...]

Example

cmtool modifyUser ec123 --fullUserName "Mary Smith"

removeGroupMember

Deletes a user name from a specific group member list.
**Required Arguments**

- **groupName**
  - *Description:* The unique name of the group.

- **userName**
  - *Description:* The unique name of the user.

**Optional Arguments**

None

**Syntax**

```shell
cmtool removeGroupMember <groupName> <userName>
```

**Example**

```shell
cmtool removeGroupMember DevGroupA ec123
```

### setBuildEndNotification

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

**Required Arguments**

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

- **enabled**
  - *Description:* Set this to true to enable notification and to false to disable it.

**Optional Arguments**

None

**Syntax**

```shell
cmtool setBuildEndNotification <buildClassId> <enabled>
```

**Example**

```shell
cmtool setBuildEndNotification 1 true
```

Enables build 'end notification' for build class 1.

### setPermissions

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

**Required Arguments**

- **principalType**
  - *Description:* Can be user or group.
entityName

**Description:** A user or group name in an access entry.

permissions

**Description:** The list of permission flags for a particular entity. See the available permissions flags below.

<table>
<thead>
<tr>
<th>AgentDelete</th>
<th>MaintenanceWrite</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgentsRead</td>
<td>MessageLogDelete</td>
</tr>
<tr>
<td>AgentsWrite</td>
<td>MessageLogRead</td>
</tr>
<tr>
<td>BuildsDelete</td>
<td>MessageLogWrite</td>
</tr>
<tr>
<td>BuildsWrite</td>
<td>ReportsDelete</td>
</tr>
<tr>
<td>ClassesDelete</td>
<td>ReportsWrite</td>
</tr>
<tr>
<td>ClassesRead</td>
<td>ResourcesDelete</td>
</tr>
<tr>
<td>ClassesWrite</td>
<td>ResourcesRead</td>
</tr>
<tr>
<td>EMMakeImpersonate</td>
<td>ResourcesWrite</td>
</tr>
<tr>
<td>EMMakeInvoke</td>
<td>ServerAccess</td>
</tr>
<tr>
<td>MaintenanceDelete</td>
<td>UserModify</td>
</tr>
<tr>
<td>MaintenanceRead</td>
<td></td>
</tr>
</tbody>
</table>

Optional Arguments

None

Available Permission Flags

Syntax

cmtool setPermissions <principalType> <entityName> <permissions>

Example

cmtool setPermissions user ec123 "BuildsRead AgentsRead"

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

**setUserSettings**

Updates settings for the currently logged-in user.

Required Arguments

watchMessages

**Description:** Indicates whether you want to receive notifications when messages of the specified notification level arrive. Can be Y, N, y, n, yes, no, Yes, or No.

Optional Arguments

notificationLevel

**Description:** Can be Info, Warning, or Error.

Syntax

cmtool setUserSettings <watchMessages> [optionals...]
Example

cmtool setUserSettings yes --notificationLevel Info

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