tsuru-admin

Release 1.0.0

July 03, 2016

Contents

tsuru-admin is the command line utility used by cloud operators to execute administrative operations on a tsuru server.

Note: This documentation is a reference of **tsuru-admin** command line interface. If you want know about how to use tsuru, you should see the tsuru documentation.

There are several ways to install **tsuru-admin**:

CHAPTER 1

Downloading binaries (Mac OS X and Linux)

We provide pre-built binaries for OS X and Linux, only for the amd64 architecture. You can download these binaries directly from the releases page:

- **tsuru-admin**: https://github.com/tsuru/tsuru-admin/releases

CHAPTER 2

Using homebrew (Mac OS X only)

If you use Mac OS X and homebrew, you may use a custom tap to install **tsuru-admin**. First you need to add the tap:

```
$ brew tap tsuru/homebrew-tsuru
```

Now you can install **tsuru-admin**:

```
$ brew install tsuru-admin
```

Whenever a new version of **tsuru-admin** is out, you can just run:

```
$ brew update
$ brew upgrade tsuru-admin
```

For more details on taps, check homebrew documentation.

Note: **tsuru-admin** requires Go 1.4. Make sure you have the last version of Go installed in your system.

CHAPTER 3

Using the PPA (Ubuntu only)

Ubuntu users can install tsuru clients using `apt-get` and the tsuru PPA. You'll need to add the PPA repository locally and run an `apt-get update`:

```
$ sudo apt-add-repository ppa:tsuru/ppa
$ sudo apt-get update
```

Now you can install **tsuru-admin** clients:

```
$ sudo apt-get install tsuru-admin
```

CHAPTER 4

Using AUR (ArchLinux only)

Archlinux users can build and install tsuru admin from AUR repository, Is needed to have installed yaourt program.

You can run:

```
$ yaourt -S tsuru
```

CHAPTER 5

Build from source (Linux and Mac OS X)

Note: If you're feeling adventurous, you can try it on other systems, like FreeBSD, OpenBSD or even Windows. Please let us know about your progress!

tsuru admin source is written in Go, so before installing tsuru from source, please make sure you have installed and configured Go.

With Go installed and configured, you will need to install godep and then download and compile tsuru-admin source. You can do that with the following commands:

```
$ GO15VENDOREXPERIMENT=1 go get github.com/tsuru/tsuru-admin
```

CHAPTER 6

Reference

6.1 Managing remote tsuru server endpoints

In tsuru, a target is the address of the remote tsuru server.

Each target is identified by a label and a HTTP/HTTPS address. The client requires at least one target to connect to, there's no default target. A user may have multiple targets, but he/she will be able to use only per session.

6.1.1 Add a new target

```
$ tsuru-admin target-add <label> <target> [--set-current|-s]
```

Adds a new entry to the list of available targets

Flags:

-s, --set-current (= false) Add and define the target as the current target

Minimum # of arguments: 2

6.1.2 List existing targets

```
$ tsuru-admin target-list
```

Displays the list of targets, marking the current.

Other commands related to target:

- target-add: adds a new target to the list of targets
- target-set: defines one of the targets in the list as the current target
- target-remove: removes one target from the list

6.1.3 Set a target as current

```
$ tsuru-admin target-set <label>
```

Change current target (tsuru server)

Minimum # of arguments: 1

6.1.4 Removes an existing target

```
$ tsuru-admin target-remove
```

Remove a target from target-list (tsuru server)

Minimum # of arguments: 1

6.2 Check current version

```
$ tsuru-admin version
```

display the current version

6.3 Container management

All the **container** commands below only exist when using the docker provisioner.

6.3.1 Moves single container

```
$ tsuru-admin container-move <container id> <to host>
```

Move specified container to another host. This command allow you to specify a container id and a destination host, this will create a new container on the destination host and remove the container from its previous host.

Minimum # of arguments: 2

6.3.2 Moves all containers from on node

```
$ tsuru-admin containers-move <from host> <to host>
```

Move all containers from one host to another. This command allows you to move all containers from one host to another. This is useful when doing maintenance on hosts. <from host> and <to host> must be host names of existing docker nodes.

This command will go through the following steps:

- Enumerate all units at the origin host;
- For each unit, create a new unit at the destination host;
- Erase each unit from the origin host.

Minimum # of arguments: 2

6.3.3 Rebalance containers in nodes

```
$ tsuru-admin containers-rebalance [--dry] [-y/--assume-yes] [-m/--metadata <metadata>=<value>]... [-
```

Move containers creating a more even distribution between docker nodes. Instead of specifying hosts as in the containers-move command, this command will automatically choose to which host each unit should be moved, trying to distribute the units as evenly as possible.

The –dry flag runs the balancing algorithm without doing any real modification. It will only print which units would be moved and where they would be created.

Flags:

-a, --app	(= []) Filter by app name
--dry	(= false) Dry run, only shows what would be done
-m, --metadata	(= {}) Filter by host metadata
-y, --assume-yes	(= false) Don't ask for confirmation.

6.4 Node management

6.4.1 Add a new docker node

```
$ tsuru-admin docker-node-add [param_name=param_value]... [--register]
```

Creates or registers a new node in the cluster. By default, this command will call the configured IaaS to create a new machine. Every param will be sent to the IaaS implementation.

IaaS providers should have been previously configured in the `tsuru.conf` file. See tsuru.conf reference docs for more information.

If using an IaaS to create a node is not wanted it's possible to simply register an existing docker node with the `--register` flag.

Parameters with special meaning:

iaas=<iaas name> Which iaas provider should be used, if not set tsuru will use the default iaas specified in tsuru.conf file.

template=<template name> A machine template with predefined parameters, additional parameters will override template ones. See 'machine-template-add' command.

address=<docker api url> Only used if `--register` flag is used. Should point to the endpoint of a working docker server.

pool=<pool name> Mandatory parameter specifying to which pool the added node will belong. Available pools can be lister with the `pool-list` command.

Flags:

--register	(= false) Registers an existing docker endpoint, the IaaS won't be called.

6.4.2 List docker nodes in cluster

```
$ tsuru-admin docker-node-list [--filter/-f <metadata>=<value>]...
```

Lists nodes in the cluster. It will also show you metadata associated to each node and the IaaS ID if the node was added using tsuru IaaS providers.

Using the `-f/--filter` flag, the user is able to filter the nodes that appear in the list based on the key pairs displayed in the metadata column. Users can also combine filters using `-f` multiple times.

Flags:

-f, --filter	(= {}) Filter by metadata name and value
-q	(= false) Display only nodes IP address

6.4.3 Update a docker node

```
$ tsuru-admin docker-node-update <address> [param_name=param_value...] [--disable] [--enable]
```

Modifies metadata associated to a docker node. If a parameter is set to an empty value, it will be removed from the node's metadata.

If the `--disable` flag is used, the node will be marked as disabled and the scheduler won't consider it when selecting a node to receive containers.

Flags:

--disable	(= false) Disable node in scheduler.
--enable	(= false) Enable node in scheduler.

Minimum # of arguments: 1

6.4.4 Remove a docker node

```
$ tsuru-admin docker-node-remove <address> [--no-rebalance] [--destroy] [-y]
```

Removes a node from the cluster.

By default tsuru will redistribute all containers present on the removed node among other nodes. This behavior can be inhibited using the `--no-rebalance` flag.

If the node being removed was created using a IaaS provider tsuru will NOT destroy the machine on the IaaS, unless the `--destroy` flag is used.

Flags:

--destroy	(= false) Destroy node from IaaS
--no-rebalance	(= false) Do not rebalance containers from removed node.
-y, --assume-yes	(= false) Don't ask for confirmation.

Minimum # of arguments: 1

6.5 Node Containers management

6.5.1 Add a new node container

```
$ tsuru-admin node-container-add <name> [-p/--pool poolname] [-r/--raw path=value]... [docker run fla
```

Add new node container or overwrite existing one. If the pool name is omitted the node container will be valid for all pools.

Flags:

-e, --env	(= []) Set environment variables
--image	(= “”) Image that will be used
--log-driver	(= “”) Logging driver for container
--log-opt	(= {}) Log driver options
--net	(= “”) Connect a container to a network
-o, --pool	(= “”) Pool to add container config. If empty it’ll be a default entry to all pools.
-p, --publish	(= []) Publish a container’s port(s) to the host
--privileged	(= false) Give extended privileges to this container
-r, --raw	(= {}) Add raw parameter to node container api call
--restart	(= “”) Restart policy to apply when a container exits
-v, --volume	(= []) Bind mount a volume

Minimum # of arguments: 1 Maximum # of arguments: 1

6.5.2 Delete an existing node container

```
$ tsuru-admin node-container-delete <name> [-p/--pool poolname] [-y]
```

Delete existing node container.

Flags:

-p, --pool	(= “”) Pool to remove container config. If empty the default node container will be removed.
-y, --assume-yes	(= false) Don’t ask for confirmation.

Minimum # of arguments: 1 Maximum # of arguments: 1

6.5.3 Update an existing node container

```
$ tsuru-admin node-container-update <name> [-p/--pool poolname] [-r/--raw path=value]... [docker run
```

Update an existing node container. If the pool name is omitted the default configuration will be updated. When updating node containers the specified configuration will be merged with the existing configuration.

Flags:

-e, --env	(= []) Set environment variables
--image	(= “”) Image that will be used
--log-driver	(= “”) Logging driver for container
--log-opt	(= {}) Log driver options
--net	(= “”) Connect a container to a network
-o, --pool	(= “”) Pool to update container config. If empty it’ll be a default entry to all pools.
-p, --publish	(= []) Publish a container’s port(s) to the host

--privileged (= false) Give extended privileges to this container

-r, --raw (= {}) Add raw parameter to node container api call

--restart (= "") Restart policy to apply when a container exits

-v, --volume (= []) Bind mount a volume

Minimum # of arguments: 1 Maximum # of arguments: 1

6.5.4 List existing node containers

```
$ tsuru-admin node-container-list
```

List all existing node containers.

Flags:

-q (= false) Show only names of existing node containers.

6.5.5 Show information abort a node container

```
$ tsuru-admin node-container-info <name>
```

Show details about a single node container.

Minimum # of arguments: 1 Maximum # of arguments: 1

6.5.6 Upgrade node container version on docker nodes

```
$ tsuru-admin node-container-upgrade <name> [-p/--pool poolname] [-y]
```

Upgrade version and restart node containers.

Flags:

-y, --assume-yes (= false) Don't ask for confirmation.

Minimum # of arguments: 1 Maximum # of arguments: 1

6.6 Machine management

6.6.1 List IaaS machines

```
$ tsuru-admin machine-list
```

Lists all machines created using an IaaS provider. These machines were created with the `docker-node-add` command.

6.6.2 Destroy IaaS machine

```
$ tsuru-admin machine-destroy <machine id>
```

Destroys an existing machine created using a IaaS.

Minimum # of arguments: 1

6.6.3 List machine templates

```
$ tsuru-admin machine-template-list
```

Lists all machine templates.

6.6.4 Add machine template

```
$ tsuru-admin machine-template-add <name> <iaas> <param>=<value>...
```

Creates a new machine template.

Templates can be used with the `docker-node-add` command running it with the `template=<template name>` parameter. Templates can contain a list of parameters that will be sent to the IaaS provider.

Minimum # of arguments: 3

6.6.5 Remove machine template

```
$ tsuru-admin machine-template-remove <name>
```

Removes an existing machine template.

Minimum # of arguments: 1

6.7 Pool management

6.7.1 Add a new pool

```
$ tsuru-admin pool-add <pool> [-p/--public] [-d/--default] [-f/--force]
```

Adds a new pool.

Each docker node added using `docker-node-add` command belongs to one pool. Also, when creating a new application a pool must be chosen and this means that all units of the created application will be spawned in nodes belonging to the chosen pool.

Flags:

-d, --default	(= false) Make pool default (when none is specified during `app-create` this pool will be used)
-f, --force	(= false) Force overwrite default pool
-p, --public	(= false) Make pool public (all teams can use it)

Minimum # of arguments: 1

6.7.2 Update pool attributes

```
$ tsuru-admin pool-update <pool> [--public=true/false] [--default=true/false] [-f/--force]
```

Updates attributes for a pool.

Flags:

--default	(= not set) Make pool default (when none is specified during `app-create` this pool will be used)
-f, --force	(= false) Force pool to be default.
--public	(= not set) Make pool public (all teams can use it)

Minimum # of arguments: 1

6.7.3 Remove a pool

```
$ tsuru-admin pool-remove <pool> [-y]
```

Remove an existing pool.

Flags:

-y, --assume-yes	(= false) Don't ask for confirmation.

Minimum # of arguments: 1

6.7.4 Add team to a pool

```
$ tsuru-admin pool-teams-add <pool> <teams>...
```

Adds teams to a pool. This will make the specified pool available when creating a new application for one of the added teams.

Minimum # of arguments: 2

6.7.5 Remove a team from a pool

```
$ tsuru-admin pool-teams-remove <pool> <teams>...
```

Removes teams from a pool. Listed teams will be no longer able to use this pool when creating a new application.

Minimum # of arguments: 2

6.8 Healer

6.8.1 List latest healing events

```
$ tsuru-admin docker-healing-list [--node] [--container]
```

List healing history for nodes or containers.

Flags:

--container (= false) List only healing process started for containers

--node (= false) List only healing process started for nodes

6.8.2 Show node healing config information

```
$ tsuru-admin docker-healing-info
```

Show the current configuration for active healing nodes.

6.8.3 Update node healing configuration

```
$ tsuru-admin docker-healing-update [-p/--pool pool] [--enable] [--disable] [--max-unresponsive <seco
```

Update node healing configuration

Flags:

--disable (= false) Disable active node healing

--enable (= false) Enable active node healing

--max-unresponsive (= -1) Number of seconds tsuru will wait for the node to notify it's alive

--max-unsuccessful (= -1) Number of seconds tsuru will wait for the node to run successful checks

-p, --pool (= "") The pool name to which the configuration will apply. If unset it'll be set as default for all pools.

6.8.4 Delete node healing configuration

```
$ tsuru-admin docker-healing-delete [-p/--pool pool] [--enabled] [--max-unresponsive] [--max-unsucces
```

Delete a node healing configuration entry.

If `--pool` is provided the configuration entries from the specified pool will be removed and the default value will be used.

If `--pool` is not provided the configuration entry will be removed from the default configuration.

Flags:

--enabled (= false) Remove the 'enabled' configuration option

--max-unresponsive (= false) Remove the 'max-unresponsive' configuration option

--max-unsuccessful (= false) Remove the 'max-unsuccessful' configuration option

-p, --pool (= "") The pool name from where the configuration will be removed. If unset it'll delete the default healing configuration.

-y, --assume-yes (= false) Don't ask for confirmation.

6.9 Platform management

Warning: All the **platform** commands below only exist when using the docker provisioner.

6.9.1 Add a new platform

```
$ tsuru-admin platform-add <platform name> [--dockerfile/-d Dockerfile] [--image/-i image]
```

Adds a new platform to tsuru.

The name of the image can be automatically inferred in case you're using an official platform. Check https://github.com/tsuru/platforms for a list of official platforms and instructions on how to create a custom platform.

Examples:

```
tsuru-admin platform-add java # uses official tsuru/java
image from docker hub                tsuru-admin platform-add java -i
registry.company.com/tsuru/java # uses custom Java image
tsuru-admin platform-add java -d /data/projects/java/Dockerfile
# uses local Dockerfile              tsuru-admin platform-add java -d
https://platforms.com/java/Dockerfile # uses remote Dockerfile
```

Flags:

-d, --dockerfile	(= "") URL or path to the Dockerfile used for building the image of the platform
-i, --image	(= "") Name of the prebuilt Docker image

Minimum # of arguments: 1

6.9.2 Update an existing platform

```
$ tsuru-admin platform-update <platform name> [--dockerfile/-d Dockerfile] [--disable/--enable] [--ir
```

Updates a platform in tsuru.

The name of the image can be automatically inferred in case you're using an official platform. Check https://github.com/tsuru/platforms for a list of official platforms.

The flags –enable and –disable can be used for enabling or disabling a platform.

Examples:

```
tsuru-admin platform-update java # uses official tsuru/java image from docker
hub        tsuru-admin platform-update java -i registry.company.com/tsuru/java
# uses custom Java image                  tsuru-admin platform-update java -d
/data/projects/java/Dockerfile # uses local Dockerfile            tsuru-admin
platform-update java -d https://platforms.com/java/Dockerfile # uses remote
Dockerfile
```

Flags:

-d, --dockerfile	(= "") URL or path to the Dockerfile used for building the image of the platform
--disable	(= false) Disable the platform

--enable (= false) Enable the platform

-i, --image (= "") Name of the prebuilt Docker image

Minimum # of arguments: 1

6.9.3 Remove an existing platform

```
$ tsuru-admin platform-remove <platform name> [-y]
```

Remove a platform from tsuru. This command will fail if there are application still using the platform.

Flags:

-y, --assume-yes (= false) Don't ask for confirmation.

Minimum # of arguments: 1

6.10 Plan management

6.10.1 Create a new plan

```
$ tsuru-admin plan-create <name> -c cpushare [-m memory] [-s swap] [-r router] [--default]
```

Creates a new plan for being used when creating apps.

Flags:

-c, --cpushare (= 0) Relative cpu share each unit will have available. This value is unitless and relative, so specifying the same value for all plans means all units will equally share processing power.

-d, --default (= false) Set plan as default, this will remove the default flag from any other plan. The default plan will be used when creating an application without explicitly setting a plan.

-m, --memory (= "0") Amount of available memory for units in bytes or an integer value followed by M, K or G for megabytes, kilobytes or gigabytes respectively.

-r, --router (= "") The name of the router used by this plan.

-s, --swap (= "0") Amount of available swap space for units in bytes or an integer value followed by M, K or G for megabytes, kilobytes or gigabytes respectively.

Minimum # of arguments: 1

6.10.2 Remove an existing plan

```
$ tsuru-admin plan-remove <name>
```

Removes an existing plan. It will no longer be available for newly created apps. However, this won't change anything for existing apps that were created using the removed plan. They will keep using the same value amount of resources described by the plan.

Minimum # of arguments: 1

6.10.3 List available routers

```
$ tsuru-admin router-list
```

List all routers available for plan creation.

6.11 Auto Scale

6.11.1 List auto scale events

```
$ tsuru-admin docker-autoscale-list [--page/-p 1]
```

List node auto scale history.

Flags:

-p, --page (= 1) Current page

6.11.2 Run auto scale process algorithm once

```
$ tsuru-admin docker-autoscale-run [-y/--assume-yes]
```

Run node auto scale checks once. This command will work even if `docker:auto- scale:enabled` config entry is set to false. Auto scaling checks may trigger the addition, removal or rebalancing of docker nodes, as long as these nodes were created using an IaaS provider registered in tsuru.

Flags:

-y, --assume-yes (= false) Don't ask for confirmation.

6.11.3 Show auto scale rules

```
$ tsuru-admin docker-autoscale-info
```

Display the current configuration for tsuru autoscale, including the set of rules and the current metadata filter.

The metadata filter is the value that defines which node metadata will be used to group autoscale rules. A common approach is to use the "pool" as the filter. Then autoscale can be configured for each matching rule value.

6.11.4 Set a new auto scale rule

```
$ tsuru-admin docker-autoscale-rule-set [-f/--filter-value <pool name>] [-c/--max-container-count 0]
```

Creates or update an auto-scale rule. Using resources limitation (amount of container or memory usage).

Flags:

-c, --max-container-count (= 0) The maximum amount of containers on every node. Might be zero, which means no maximum value. Whenever this value is reached, tsuru will trigger a new auto scale event.

-d, --scale-down-ratio (= 1.33) The ratio for triggering an scale down event. The default value is 1.33, which mean that whenever it gets one third of the resource utilization (memory ratio or container count).

--disable (= false) A boolean flag indicating whether the rule should be disabled

--enable (= false) A boolean flag indicating whether the rule should be enabled

-f, --filter-value (= "") The pool name matching the rule. This is the unique identifier of the rule.

-m, --max-memory-ratio (= 0) The maximum memory usage per node. 0 means no limit, 1 means 100%. It is fine to use values greater than 1, which means that tsuru will overcommit memory in Docker nodes. Keep in mind that container count has higher precedence than memory ratio, so if –max-container-count is defined, the value of –max-memory-ratio will be ignored.

--no-rebalance-on-scale (= false) A boolean flag indicating whether containers should NOT be rebalanced after running an scale. The default behavior is to always rebalance the containers.

6.11.5 Remove an auto scale rule

```
$ tsuru-admin docker-autoscale-rule-remove [rule-name] [-y/--assume-yes]
```

Removes an auto-scale rule. The name of the rule may be omited, which means "remove the default rule".

Flags:

-y, --assume-yes (= false) Don't ask for confirmation.

6.12 Application Logging

6.12.1 Update logging configuration

```
$ tsuru-admin docker-log-update [-r/--restart] [-p/--pool poolname] --log-driver <driver> [--log-opt
```

Set custom configuration for container logs. By default tsuru configures application containers to send all logs to the tsuru/bs container through syslog.

Setting a custom log-driver allow users to change this behavior and make containers send their logs directly using the driver bypassing tsuru/bs completely. In this situation the 'tsuru app-log' command will not work anymore.

The –log-driver option accepts either the value 'bs' restoring tsuru default behavior or any log-driver supported by docker along with their –log-opt. See https://docs.docker.com/engine/reference/logging/overview/ for more details.

If –pool is specified the log-driver will only be used on containers started on the chosen pool.

Flags:

--log-driver (= "") Chosen log driver. Supported log drivers depend on the docker version running on nodes.

--log-opt (= {}) Log options send to the specified log-driver

-p, --pool (= "") Pool name where log options will be used.

-r, --restart (= false) Whether tsuru should restart all apps on the specified pool.

6.12.2 Show logging configuration

```
$ tsuru-admin docker-log-info
```

Prints information about docker log configuration for each pool.

6.13 Quota management

Quotas are handled per application and user. Every user has a quota number for applications. For example, users may have a default quota of 2 applications, so whenever a user tries to create more than two applications, he/she will receive a quota exceeded error. There are also per applications quota. This one limits the maximum number of units that an application may have.

tsuru-admin can be used to see and change quota data.

6.13.1 Change application quota

```
$ tsuru-admin app-quota-change <app-name> <new-limit>
```

Changes the limit of units that an app can have.

The new limit must be an integer, it may also be "unlimited".

Minimum # of arguments: 2

6.13.2 Change user quota

```
$ tsuru-admin user-quota-change <user-email> <new-limit>
```

Changes the limit of apps that a user can create.

The new limit must be an integer, it may also be "unlimited".

Minimum # of arguments: 2

6.13.3 View application quota

```
$ tsuru-admin app-quota-view <app-name>
```

Displays the current usage and limit of the given app.

Minimum # of arguments: 1

6.13.4 View user quota

```
$ tsuru-admin user-quota-view <user-email>
```

Displays the current usage and limit of the user.

Minimum # of arguments: 1

6.14 Other commands

6.14.1 Unlock an application

```
$ tsuru-admin app-unlock -a <app-name> [-y]
```

Forces the removal of an application lock. Use with caution, removing an active lock may cause inconsistencies.

Flags:

-a, --app	(= “”) The name of the app.
-y, --assume-yes	(= false) Don’t ask for confirmation.