Consular Documentation

Release 1.3.0

Simon de Haan

February 20, 2017

Contents

Consular is a micro-service that relays information between Marathon and Consul. It registers itself for HTTP event callbacks with Marathon and makes the appropriate API calls to register applications that Marathon runs as services in Consul. De-registrations of applications happens in the same way.

Marathon is always considered the source of truth.

If Marathon application definitions contain labels (application metadata) they will be added to the Consul key/value store. This can be especially useful when Consul is combined with Consul Template to automatically generate configuration files for proxies such as HAProxy and Nginx.

CHAPTER 1

Usage

```
$ pip install consular
$ consular --help
```

CHAPTER 2

Installing for local dev

```
$ git clone https://github.com/universalcore/consular.git
$ cd consular
$ virtualenv ve
$ source ve/bin/activate
(ve)$ pip install -e .
(ve)$ pip install -r requirements-dev.txt
```

CHAPTER 3

Running tests

```
$ source ve/bin/activate
(ve)$ py.test consular
```

CHAPTER 4

Consular Class Documentation

class consular.main.**Consular**(*consul_endpoint*, *marathon_endpoint*, *enable_fallback*, *registration_id*)

Parameters

- **consul_endpoint** (*str*) – The HTTP endpoint for Consul (often http://example.org:8500).
- **marathon_endpoint** (*str*) – The HTTP endpoint for Marathon (often http://example.org:8080).
- **enable_fallback** (*bool*) – Fallback to the main Consul endpoint for registrations if unable to reach Consul running on the machine running a specific Marathon task.
- **registration_id** (*str*) – A unique parameter for this Consul server. It is used for house-keeping purposes such as purging tasks that are no longer running in Marathon.

app_id_tag(*app_id*)
Get the app ID tag for the given app ID.

check_apps_namespace_clash(*apps*)
Checks if app names in Marathon will cause a namespace clash in Consul. Throws an exception if there is a collision, else returns the apps.

Param apps: The JSON list of apps from Marathon's API.

clean_consul_app_labels(**args*, ***kwargs*)
Delete app labels stored in the Consul k/v store under the given app name that aren't present in the given set of labels.

deregister_consul_service(*agent_endpoint*, *service_id*)
Deregister a service from a Consul agent.

Parameters

- **agent_endpoint** (*str*) – The HTTP endpoint of the Consul agent.
- **service_id** (*str*) – The ID of the Consul service to be deregistered.

deregister_task_service(*task_id*, *host*)
Deregister a Marathon task's service from Consul.

Parameters

- **task_id** (*str*) – The ID of the task, this will be used as the Consul service ID.
- **host** (*str*) – The host address of the machine the task is running on.

events(*request*)
Listens to incoming events from Marathon on `/events`.

Parameters **request** (*klein.app.KleinRequest*) – The Klein HTTP request

get_app_id_from_tags(*tags*)
Get the app ID from the app ID tag in the given tags, or None if the tag could not be found.

get_consul_app_keys(*app_name*)
Get the Consul k/v keys for the app with the given name.

get_consul_consular_keys()
Get the next level of Consul k/v keys at 'consular/', i.e. will return 'consular/my-app' but not 'consular/my-app/my-label'.

handle_status_update_event(*request*, *event*)
Handles status updates from Marathon.

The various task stages are handled as follows:

TASK_STAGING: ignored TASK_STARTING: ignored TASK_RUNNING: task data updated on Consul TASK_FINISHED: task data removed from Consul TASK_FAILED: task data removed from Consul TASK_KILLED: task data removed from Consul TASK_LOST: task data removed from Consul

Parameters

- **request** (*klein.app.KleinRequest*) – The Klein HTTP request
- **event** (*dict*) – The Marathon event

purge_dead_app_labels(**args*, ***kwargs*)
Delete any keys stored in the Consul k/v store that belong to apps that no longer exist.

Param apps: The list of apps as returned by the Marathon API.

put_consul_app_labels(*app_name*, *labels*)
Store the given set of labels under the given app name in the Consul k/v store.

put_consul_kvs(**args*, ***kwargs*)
Store the given key/value set in the Consul k/v store.

reg_id_tag()
Get the registration ID tag for this instance of Consular.

register_marathon_event_callback(**args*, ***kwargs*)
Register Consular with Marathon to receive HTTP event callbacks. To use this ensure that Marathon is configured to send HTTP event callbacks for state changes in tasks.

Parameters **events_url** (*str*) – The HTTP endpoint to register with Marathon for event callbacks.

register_task_service(*app_id*, *task_id*, *host*, *ports*)
Register a Marathon task as a service in Consul.

Parameters

- **app_id** (*str*) – The ID of the Marathon app that the task belongs to.
- **task_id** (*str*) – The ID of the task, this will be used as the Consul service ID.
- **host** (*str*) – The host address of the machine the task is running on.
- **ports** (*list*) – The port numbers the task can be accessed on on the host machine.

run(*host*, *port*)
Starts the HTTP server.

Parameters

- **host** (*str*) – The host to bind to (example is `localhost`)
- **port** (*int*) – The port to listen on (example is `7000`)

schedule_sync(*interval*, *purge=False*)
Schedule a recurring sync of apps, starting after this method is called.

Parameters

- **interval** (*float*) – The number of seconds between syncs.
- **purge** (*bool*) – Whether to purge old apps after each sync.

Returns A tuple of the LoopingCall object and the deferred created when it was started.

sync_app_labels(**args*, ***kwargs*)
Sync the app labels for the given app by pushing its labels to the Consul k/v store and cleaning any labels there that are no longer present.

Param app: The app JSON as return by the Marathon HTTP API.

sync_apps(**args*, ***kwargs*)
Ensure all the apps in Marathon are registered as services in Consul.

Set `purge` to `True` if you automatically want services in Consul that aren't registered in Marathon to be purged. Consular only purges services that have been registered with the same `registration-id`.

Parameters **purge** (*bool*) – To purge or not to purge.

update_task_running(**args*, ***kwargs*)
Use a running event to register a new Consul service.

consular.main.**get_app_name**(*app_id*)
Get the app name from the marathon app ID. Separators in the ID ('/') are replaced with '-'s while the leading separator is removed.

consular.main.**handle_not_found_error**(**args*, ***kwargs*)
Perform a request and catch the not found (404) error if one occurs.

Param f: The function to call to perform the request. The function may return a deferred.

Param args: The arguments to call the function with.

Param kwargs: The keyword arguments to call the function with.

Returns The return value of the function call or None if there was a 404 response code.

C

A

C

D

E

G

H

P

R

S

U