kongui Documentation

Release 0.1.0

Thomas Farla

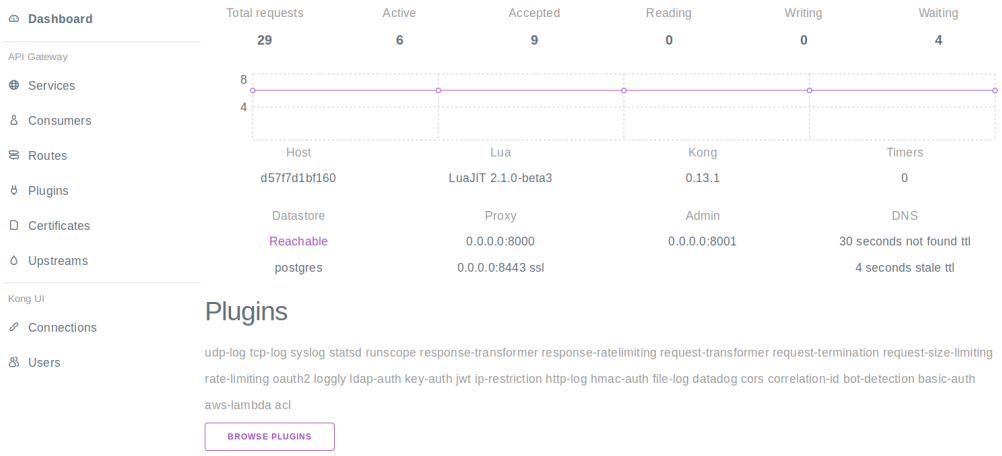
Jul 01, 2018

Contents:

CHAPTER 1

Installation

Kongui is a user interface to manage the Kong Microservice API Gateway. If you are not familiar with Kong then please visit their website to gain a general understanding of the problems that Kong solves.



CHAPTER 2

Why another user interface for kong?

Deploying kong into a production environment is simple. The difficult part is monitoring, access control and configuration management. The latter one becomes especially difficult when Kong is shared with multiple teams. Kongui enables the maintainers to configure templates, ACL (via LDAP) and a basic notification service when an anomaly occurres.

CHAPTER 3

Status

Kongui supports configuration for most of the objects in the Kong admin API but is far from production ready. The following objects can be configured using Kongui:

- consumers
- certificates
- plugins
- routes
- services
- upstreams
- connections towards the Kong admin API

At the time of writing the following objects have not been implemented:

- targets
- snis
- apis (deprecated since kong v0.13.0)

CHAPTER 4

Quickstart

Kongui needs a database to store information about the connections towards the Kong admin api. Only postgresql is supported at the moment which might be okay if you are already using Kong with postgresql as its primary database. Support for the following databases may follow in future releases:

- mysql
- mssql
- sqlite
- in memory using mnesia

The rest of this quickstart will assume that posgres is being used as the database but the migration step will be mostly the same for other databases. The recommended way to start Kongui is with the official docker image which is based on ubuntu bionic. Execute the following command to run the Kongui docker container:

```
docker run --rm \
   -e KONGUI_DB_HOSTNAME=localhost \
   -e KONGUI_DB_USERNAME=kongui \
   -e KONGUI_DB_PASSWORD=kongui \
   -e KONGUI_DB_NAME=kongui \
   -e KONGUI_DB_PORT=5432 \
   -p 4000:4000 \
   tfarla/kongui bin/kongui migrate && bin/kongui foreground
```

This command will:

1. Set several environment variables with the -e flag which allows Kongui to connect with the database
2. Creates a binding on port 4000 from *host:guest* if you'd like to run Kongui on a different part then change *-p 4000:4000* to *-p <the desired port on the host machine>:4000*
3. Migrates the database and starts Kongui on the foreground so we are able to see all Kongui's log messages in our terminal

Please look at the output carefully to see if any error has occurred. If not, then Kongui should be running on http://localhost:4000.

The bin/kongui application allows the following commands:

```
Usage: kongui <task>

Service Control
=======================
start                          # start kongui as a daemon
start_boot <file>              # start kongui as a daemon, but supply a custom .boot␣
↪file
foreground                     # start kongui in the foreground
console                        # start kongui with a console attached
console_clean                  # start a console with code paths set but no apps␣
↪loaded/started
console_boot <file>            # start kongui with a console attached, but supply a␣
↪custom .boot file
stop                           # stop the kongui daemon
restart                        # restart the kongui daemon without shutting down the␣
↪VM
reboot                         # restart the kongui daemon
reload_config                  # reload the current system's configuration from disk

Upgrades
=======================
upgrade <version>              # upgrade kongui to <version>
downgrade <version>            # downgrade kongui to <version>
install <version>              # install the kongui-<version> release, but do not␣
↪upgrade to it

Utilities
=======================
attach                         # attach the current TTY to kongui's console
remote_console                 # remote shell to kongui's console
pid                            # get the pid of the running kongui instance
ping                           # checks if kongui is running, pong is returned if␣
↪successful
pingpeer <peer>                # check if a peer node is running, pong is returned if␣
↪successful
escript <file>                 # execute an escript
rpc <mod> <fun> [<args..>]     # execute an RPC call using the given MFA
rpcterms <mod> <fun> [<expr>]  # execute an RPC call using the given Erlang␣
↪expression for args
eval <expr>                    # execute the given Erlang expression on the running␣
↪node
command <mod> <fun> [<args..>] # execute the given MFA
describe                       # print useful information about the kongui release

Custom Commands
=======================
/kongui/releases/0.1.0/commands/migrate.sh
```