topologist Documentation

Release alpha0

Paul Colomiets

November 21, 2013

Contents

Sources: http://github.com/nanomsg/topologist

Contents:

CHAPTER 1

Topologist Tutorial

Topologist configuration is a YAML file, that is parsed using quire. So you might refer to the documentation of both for more expret info.

1.1 Simple Example

Here is a simple configuration file for one service:

```
server:
  config-addr: [tcp://10.0.0.1:10001]
  updates-addr: [tcp://10.0.0.2:10002]

layouts:
  simple:
    server <- client:
      port: 10010

topologies:
  simple: !Topology
    type: reqrep
    layout: simple
    assign:
      server:
      - host: 10.0.0.3
```

In consists of the following sections.

1.1.1 The Server Section

```
server:
  config-addr: [tcp://10.0.0.1:10001]
  updates-addr: [tcp://10.0.0.2:10002]
```

It consists of addresses where to listen to for requests and some other server-wide settings. The addresses in example are in square brackets because they are actually YAML lists, for the case you want multiple addresses to listen to (such as on different network interfaces).

1.1.2 The Layouts Section

This section represents graph of the connections with connection settings. Given the example:

```
layouts:
  simple:
    server <- client:
      port: 10010
```

There are two roles `client` and `server`. Where server is bound on the port `10010` and client connects to it.

1.1.3 The Topologies Section

CHAPTER 2

Topologist Reference

Topologist configuration is a YAML file, that is parsed using quire. So you might refer to the documentation of both for more expret info.

2.1 Sections

2.1.1 Layouts

The `layouts` section describes the layouts in form of mapping. Layouts are reusable parts of the topology. Layouts represented in graph-like structure.

2.1.2 Topologies

The `topologies` section is a directory of the real topologies that might be resolved by end client. There are currently two kinds of topologies:

- normal topology (the `!Topology` tag) is represented by hirarchy of layouts and define the connections between nodes of the topology
- extern topology (the `!Extern` tag) is used to "extract" some public node addresses from other normal topology. So that clients, that connect to the topology never connect to some internal addresses regardless of what role and other parameters they specify.

2.2 Tips And Tricks

2.2.1 Including files

You can put every layout and every topology into it's own file and include them in the following way:

```
layouts: !GlobMap "./layouts/(*).yaml"
topologies: !GlobMap "./topologies/(*).yaml"
```

Note the following things:

- The name of the layout and topology is derived from the name of the file, in particular the part that is matched inside parenthesis.
- The path to file is relative to the directory where original configuration file is, not relative to the current working directory as one may expect