Documentation for the sf1r-module

Paolo D'Apice

March 30, 2012

Abstract

This document contains the documentation for the Nginx sf1r-module enabling the communication with the SF1.

1 Overview

The objective of the sf1r-module is to allow the SF1 to handle requests via HTTP, as shown in Figure 1 on the following page. The sf1r-module relies on the libsf1r [1], which implements two driver clients for the SF1:

single allowing the connection to an SF1 instance, hence implementing a direct HTTP front-end to the SF1

distributed allowing the connection to a cluster of SF1 instances managed with ZooKeeper, acting as an active reverse proxy to the cluster

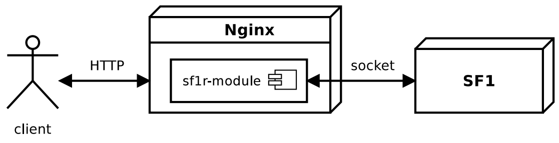
The diagram in Figure 2 on the next page shows how a single request is processed:

1. HTTP request received
2. sf1r-module request handler
3. request sent to SF1
4. response received from the SF1
5. response handler
6. HTTP response sent

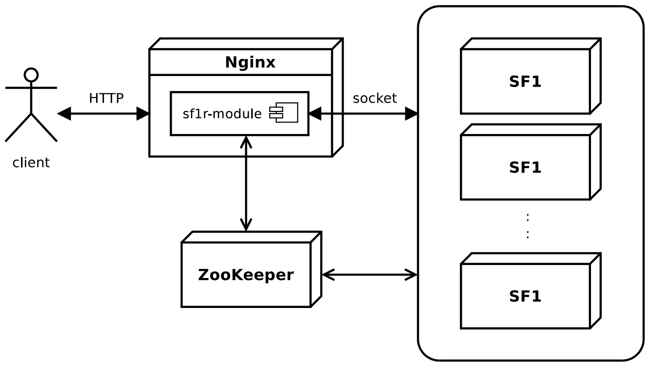
Implementation Notes

Logging The libsf1r uses the Google Logging Library. Within Nginx it is possible to partially control the logging behavior by properly defining environment variables, such as `GLOG_log_dir` or `GLOG_minloglevel` [5]. However, there are still some open issues about the initialization/finalization of the logging system:

- logging from a library
  `http://code.google.com/p/google-glog/issues/detail?id=113`



(a) Single driver.



(b) Distributed driver.

Figure 1: Deployment diagram.

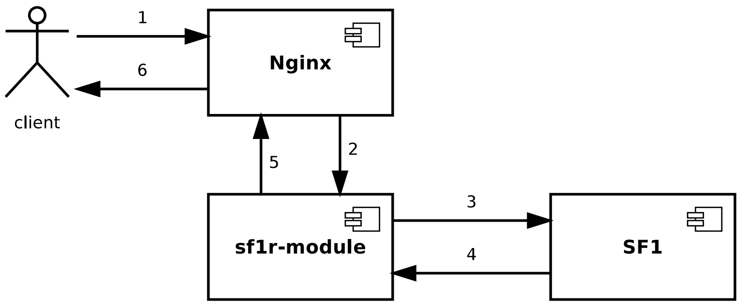


Figure 2: Communication diagram.

Listing 1: Sample HTTP request

```
HTTP POST /path/documents/search
content-type: application/json
{ "collection":"example", "header":{"check_time":true},
  "search":{"keywords":"america"}, "limit":10 }
```

- calling `ShutdownGoogleLogging()` after `InitGoogleLogging()` does not work `http://code.google.com/p/google-glog/issues/detail?id=83`

Such issues can be workaround-ed until they get fixed upstream simply by not explicitly initializing/finalizing the logging system.

Threads The ZooKeeper client used within libsf1r spawns two thread: one for I/O and one for event handling. Nginx implementation, on the other hand, is totally asynchronous, event-based and does not use threads [4]. It instead uses the fork() system call for creating the master and workers processes. Hence problems arise because threads are not inherited across forks[1] The lazy initialization of the distributed driver in the libsf1r workarounds this issue.

2 Communication protocol

HTTP requests for the SF1 are required to specify in the URI the controller and the action parameters for the underlying SF1 request, as in the sample Listing 1. The HTTP response body will then contain the SF1 response in the specified format.

3 Configuration

The Listing 2 on the next page contains a sample configuration snippet containing the follow directives:

- `underscores_in_headers [on|off]` Nginx directive enabling HTTP headers in nonstandard format
- `rewrite` enables URI rewriting in order to replace /sf1r/controller/action to /controller/action as required by the driver[2]
- `sf1r` enables the sf1r-module
- `sf1r_addr host:port[,host:port] [single|distributed]` defines the target host; if the flag `distributed` is used, it is possible to define multiple hosts
- `sf1r_poolSize n` defines the connection pool initial size
- `sf1r_poolResize [on|off]` enables the pool auto-resize

[1]It is discouraged to use fork() and threads within the same program (see the glibc documentation at `http://www.imodulo.com/gnu/glibc/Threads-and-Fork.html`).

[2]See Nginx HttpRewriteModule [2].

Listing 2: Sample configuration

```
underscores_in_headers on;

location /sf1r/ {
    rewrite ^/sf1r(/.*)$ $1 break;

    sf1r;
    sf1r_addr server1:2181,server2:port2 distributed;
    sf1r_poolSize 5;
    sf1r_poolResize on;
    sf1r_poolMaxSize 10;
    sf1r_broadcast ^test/\w+$;
    sf1r_broadcast ^recommend/visit_item$;

    more_set_headers 'Access-Control-Allow-Origin: *';
    more_set_headers 'Access-Control-Allow-Methods: POST,
        GET, PUT, DELETE, OPTIONS';
    more_set_headers 'Access-Control-Allow-Headers: CONTENT-
        TYPE';
    more_set_headers 'Access-Control-Max-Age: 1728000';
    more_set_headers 'Access-Control-Allow-Credentials:
        false';
}
```

- `sf1r_poolMaxSize n` defines the maximum number of connections for the pool if the `sf1r_poolResize` has been set
- `sf1r_broadcast regex` define URI pattern for broadcasted requests
- `more_set_headers` additional HTTP response headers needed in order to support Ajax requests[3]

4 References

References

[1] libsf1r documentation, `git@izenesoft.cn:izenelib.git`

[2] Nginx Wiki, `http://wiki.nginx.org/Main`

[3] Evan Miller, *Emiller's Guide To Nginx Module Development*, `http://www.evanmiller.org/nginx-modules-guide.htm`

[4] Joshua Zhou, *Nginx Internals Talk in Guangzhou, China*, `http://www.slideshare.net/joshzhu/nginx-internals`

[5] Google Logging Library documentation, `http://google-glog.googlecode.com/svn/trunk/doc/glog.html`

[3]See third party HttpHeadersMoreModule [2].