hi-nginx 多语言通用服务器

version 1.0.3

使用手册

hi-nginx@webcpp.net

2017 年 6 月 16 日

目录

摘要

hi-nginx 是一款基于 nginx 写成的通用服务器。它既是 web server，也是 application server；它不仅继承了 nginx 的全部功能，完全兼容 nginx，而且支持多种语言混合开发 web 应用。它性能强劲，易于开发，部署方便。目前，hi-nginx 支持混合使用 c++，python 以及 lua 同时进行 web 应用开发。用户应该根据应用场景的实际需要，细粒度地选择最为合适的开发语言，最大限度的发挥 hi-nginx 的潜能。

1 导言

hi-nginx 既是 web server，也是 application server。这是它区别于 nginx 的最主要的特点。作为前者，它跟 nginx 一样，可作静态资源服务器，可作反向代理服务器，还可作负载均衡服务器，一切都一如 nginx。作为后者，它让 c++ 程序员，python 程序员，以及 lua 程序员写的 web application 完全运行在 nginx 服务器内部，从而可以轻松提供高性能的、支持大并发的 web application。

为什么 hi-nginx 要同时支持三种编程语言？其原因有三。第一，我最常用的编程语言是 c++，它必须被支持；而且它非常快，非常适合处理“热点”业务。第二，python 库资源极为丰富，非常适合处理常规业务，几乎没有它未曾涉猎开发领域，因而它能够极大地加快开发速度。第三，lua 比 python 快，但是库资源不及后者，支持它是为了方便处理那些介于”热点”业务和“常规”业务之间的业务。

因此，使用 hi-nginx，让其发挥出最大潜能，需要开发者同时熟知 c++、python、lua 三种编程语言。这并不是非常高的要求。实际上，这三种语言都非常易学易用，尽管并不是所有人都认同这一点。当然，用户只熟知其中的某一种编程语言也无妨——即便是对 python 程序员而言，hi-nginx 也能提供非常高效的并发处理能力。

目前，把 c 或者 c++ 运用于 web 应用开发的最主要的方式是 script 加 c 或者 c++ extension。这样做的目的其实主要地还是为了解决 script 可能无法胜任“热点”业务的问题。首先是脚本，然后是 c 或者 c++ 扩展，最后再回到脚本。这条性能优化路线在 web 应用开发中极为常见。hi-nginx 不仅支持这条路线——用户照样可以为 python 和 lua 开发 c 或 c++ 扩展——而且还支持另一条路线，即直接用 c++ 写 web application。这条路线省去了“脱裤子”的麻烦；对于能写 c 或 c++ 扩展的程序员而言，这是极为便利的。当然，如果用户仅仅能写脚本，hi-nginx 也保证提供比已有的反向代理方案更强大的并发能力。

hi-nginx 致力于增强用户的工作，而不是改变用户的工作。当用户不满意它时，用户可以安全地“回滚”至之前的工作状态，而不会产生任何损失。

2 快速部署

hi-nginx 目前仅仅支持 Linux 系统。

2.1 下载源码包

hi-nginx 是一个开源在https://github.com/webcpp/hi-nginx上的一个开源项目。用户可以直接从该地址下载 hi-nginx 的源码包。建议下载最新的正式发布版，前往https://github.com/webcpp/hi-nginx/releases查看并点击下载最新版本即可。

不建议直接使用 git clone 命令下载未正式分布的源码包。

2.2 编译及安装

要编译 hi-nginx，需要安装一些依赖软件。包括：

- gcc
- gcc-c++
- make
- pcre-devel
- zlib-devel
- openssl-devel

- hiredis-devel
- python-devel
- boost-devel
- luajit-devel

假设用户使用的 Linux 是 CentOS，那么很简单，执行:

```
sudo yum install gcc gcc-c++ make pcre-devel zlib-devel openssl-devel hiredis-devel python-devel boost-devel luajit-devel
```

即可。

安装以上依赖软件之后，解压缩 hi-nginx 源码包。进入源码目录后，会看到一个演示性的安装脚本 install-demo.sh。如果用户仅仅是尝试 hi-nginx，可以直接运行该该脚本。它的内容很简单:

```
#!/bin/bash
./configure  --with-http_ssl_module \
             --with-http_v2_module \
             --prefix=/home/centos7/nginx \
             --add-module=ngx_http_hi_module
```

这个脚本告诉用户，编译 hi-nginx 与编译 nginx 没有什么不同，只需使用配置选项--add-module 指定 ngx_http_hi_module，其他则一如后者。

执行完以上步骤之后,就只剩下 make && make install 了。安装目录是--prefix 选项指定的/home/centos7/nginx。

2.3 基本测试

现在，hi-nginx 已经安装到了/home/centos7/nginx 目录中。用户可以从该目录启动 hi-nginx。启动方法与 nginx 无异。

但是，既然已经安装好了 hi-nginx，在正式使用它之前，就应该测试一下它 application server 功能，确保安装的成功。为此，用户需要做两件事情。第一件事情是安装 redis 服务器，因为 hi-nginx 的会话功能需要它。安装的方法也很简单: sudo yum install redis。第二件事情是下载https://github.com/webcpp/hi_demo上的演示代码。git clone https://github.com/webcpp/hi_demo.git 即可。演示代码假定 hi-nginx 的安装目录为/home/centos7/nginx，执行 make && make install 即可。如果安装目录不同于此，需修改 hi_demo 目录下 Makefile 中 NGINX_INSTALL_DIR 变量的值。

hi_demo 目录下有两个特殊文件，一个是 demo.html，一个是 demo.conf。前者可带领用户测试 hi-nginx，后者则用户配置 hi-nginx 以正确加载 hi_demo。用户执行以下命令就可以正确安装这两个文件:

```
install demo.html /home/centos7/nginx/html
install demo.conf /home/centos7/nginx/conf
```

完成以上步骤之后，就可以正式测试 hi-nginx 了。

进入 hi-nginx 安装目录，执行 sbin/nginx -c conf/demo.conf，然后访问http://localhost:8765/demo.html，按链接指引点击即可。如无意外，hi-nginx 会通过所有测试。如果遇到问题，用户可查看/home/centos7/nginx/conf/demo.conf 文件：

```
        hi_need_cache on;       #缓存开关
        hi_cache_size 10;       #缓存容器大小
        hi_cache_expires 300s;  #缓存过期时间
        hi_need_headers off;    #http header 开关
        hi_need_cookies off;    #http cookie 开关
        hi_need_session off;    #http session 开关
        hi_session_expires 300s; #http session 过期时间
        hi_redis_host 127.0.0.1; #redis 主机
        hi_redis_port 6379;  #redis 端口

        expires 10s;
        location = /hello {
            hi_need_cache off;
            hi hi/hello.so;
        }



        location ^~ /form {
            rewrite ^/form/(\d+)$ /form/?item=$1 break;
            hi_need_cache off;
            hi_need_headers on;
            hi_need_cookies on;
            hi hi/form.so;
        }

        location = /error {
            hi hi/error.so;
        }

        location = /redirect {
            hi hi/redirect.so;
        }

        location = /empty {
            hi hi/empty.so;
        }

        location = /math {
            hi_cache_expires 5s;
            hi hi/math.so;
        }

        location = /session {
            hi_need_cache off;
            hi_need_session on;
            hi_session_expires 30s;
            hi hi/session.so;
        }

```

```
51      location = /pyecho {
52             hi_need_cache off;
53             hi_python_content "hi_res.status(200)\nhi_res.content('hello,world')";  #运行python内容块
54
55      }
56
57      location ~ \.py$ {
58             hi_need_cache off;
59             hi_need_headers on;
60             hi_need_session on;
61             hi_session_expires 30s;
62             hi_python_script python;                                    #运行python脚本
63      }
64
65      location = /luaecho {
66             hi_need_cache off;
67             hi_lua_content "hi_res:status(200)\nhi_res:content('hello,world')";  #运行lua内容块
68
69      }
70
71      location ~ \.lua$ {
72             hi_need_cache off;
73             hi_need_headers on;
74             hi_need_session on;
75             hi_session_expires 30s;
76             hi_lua_script lua;                                          #运行lua脚本
77      }
78
79         location / {
80             root  html;
81             index index.html index.htm;
82         }
```

对照以上配置检查哪里出现不意状态。也可查看logs/error.log，看看有何种提示信息。

3 起步

3.1 hi-project

hi-project 是一个辅助脚本，安装在/home/centos7/nginx/hi 目录中。它的用途是为用户创建 “起步” 代码模板。运行它可以使用三个选项，依次是：

- 工程名，可选，默认 demo
- 工程类型，可选，支持 cpp，python 和 lua，默认 cpp
- hi-nginx 安装路径，可选，默认/home/centos7/nginx

用户可以通过-h 或者--help 参看使用说明。

3.2 hello world

`hello world` 工程包含了 `hi-nginx web application` 开发的最基本要素。

3.2.1 cpp

首先使用 `hi-project` 脚本创建一个 `cpp` 工程，名为 hello：

```
/home/centos7/nginx/hi/hi-project hello cpp /home/centos7/nginx
```

后面两个参数是可省的。这时，`hi-project` 会创建一个名为 hello 的目录，并在该目录中创建两个文件，一个是`Makefile`，一个是`hello.cpp`。前者帮助用户在执行 `make && make install` 时把 `web application` 编译、安装至正确位置；后者则帮助用户正确创建合乎 `hi-nginx` 要求的 `class`：

```
#include "servlet.hpp"


namespace hi {

    class hello : public servlet {
    public:

        void handler(request& req, response& res) {
            res.headers.find("Content-Type")->second = "text/plain;charset=UTF-8";
            res.content = "hello,world";
            res.status = 200;
        }

    };
}

extern "C" hi::servlet* create() {
    return new hi::hello();
}

extern "C" void destroy(hi::servlet* p) {
    delete p;
}
```

以上代码一目了然，无需过多解释，任何熟知 `c++` 的程序员都能看懂。没错，`hi-nginx` 并不要求 `cpp` 程序员“精通”自己的工具，只需熟知即可。当然，熟知 `http` 协议是必要的，否则很难正确地使用 `request` 类和 `response` 类。

`request` 类包含了客户端访问 `hi-nginx` 时所携带的信息。

3.2.2 python

3.2.3 lua