第 3 章 Chapter 3

OpenResty

一个典型的互联网系统或云计算系统大量使用到关系型数据库、非关系型数据库、缓存、内存数据库等，以提供高速、高扩展性的服务，通常组成集群。以前协调这些系统开发是非常麻烦的，现在 OpenResty 提供的 Nginx+Lua+Module 的机制，使我们可以实现快速开发，让开发者着眼于应用，使用同一语言开发。在这种架构的应用领域里，其效率是其他语言和技术不能比拟的。

本章介绍 OpenResty 的组成及安装、配置方法。

3.1 OpenResty：概述

OpenResty 是一个基于 Nginx 与 Lua 的高性能 Web 平台，集成了大量精良的 Lua 库、第三方模块以及大多数的依赖项，用于方便地搭建能够处理超高并发、扩展性极高的动态 Web 应用、Web 服务和动态网关。

OpenResty 通过汇聚各种设计精良的 Nginx 模块（主要由 OpenResty 团队自主开发），从而将 Nginx 有效地变成一个强大的通用 Web 应用平台。这样，Web 开发人员和系统工程师可以使用 Lua 脚本语言调动 Nginx 支持的各种 C 以及 Lua 模块，快速构造出足以胜任 10K 乃至 1000K 以上单机并发连接的高性能 Web 应用系统。

OpenResty 致力于将服务器端应用完全运行于 Nginx 服务器中，充分利用 Nginx 的事件模型进行非阻塞 I/O 通信，不仅仅和 HTTP 客户端间的网络通信是非阻塞的，与 MySQL、PostgreSQL、Memcached 以及 Redis 等众多后端之间的网络通信也是非阻塞的。

因为 OpenResty 软件包的维护者也是其中打包的许多 Nginx 模块的作者，所以 Open-

Resty 可以确保所包含的所有组件可以可靠地协同工作。

使用 Lua 在 Nginx 下开发，需要安装很多支撑库，例如：

- Lua 解释器：标准 Lua 5.1 或 LuaJIT 2.0/2.1，用于对 Lua 语言进行解析。
- Lua 核心模块：lua_nginx_module，是 Lua 语言和 Nginx 的桥梁，我们的脚本全部是通过 ngx_lua 模块和 Nginx 协调起来工作的。其中 Lua 的 VM 也在 ngx_lua 中工作。
- MySQL 库：异步访问 MySQL 的 Lua 库。
- Redis 库：异步访问 MySQL 的 Lua 库。
- Memcached 库：异步访问 Memcached 的 Lua 库。
- PostgreSQL 库：异步访问 PostgreSQL 的 Lua 库。
- JSON 库 ：Lua 上的 CJSON 库。
- MySQL RDS 库：MySQL 结果集处理 RDS 库。
- Redis RDS 库：Redis 的结果集处理 RDS 库。
- Drizzle Nginx Module：一个和 MySQL 或 Drizzle 通信的上游服务器。
- ……

这些库都需要分别安装和配置，通过 OpenResty 可以把这些库和 Nginx 打包到一起，让研发者或使用者直接使用，从而省去配置和匹配的麻烦，所以我们推荐使用 OpenResty 进行 Nginx 下 Lua 开发环境的搭建。

3.2 OpenResty 的组成

OpenResty 由下面的组件组成。

- 标准 Lua 5.1 解释器；
- Drizzle Nginx 模块；
- Postgres Nginx 模块 ；
- Iconv Nginx 模块 。

所有组件均可以方便地被激活或禁止。绝大部分组件已内置在 OpenResty 安装包中，但也有一部分不包含在内。

上面 4 个模块默认并未启用，需要分别加入 --with-lua51、--with-http_drizzle_module、--with-http_postgres_module 和 --with-http_iconv_module 编译选项开启它们。

其余各组件编译选项，可对照 OpenResty 安装说明，按需启用。非必要时，不推荐启用标准 Lua 5.1 解释器，而应尽量使用 LuaJIT 组件。

在 1.5.8.1 版本之前，OpenResty 默认使用标准 Lua 5.1 解释器。所以对于老版本，需要显式地加入 --with-luajit 编译选项（1.5.8.1 以后的版本已默认开启）来启用 LuaJIT 组件。

OpenResty 支持的模块如下：

- LuaJIT；

- ArrayVarNginxModule；
- AuthRequestNginxModule；
- CoolkitNginxModule；
- DrizzleNginxModule；
- EchoNginxModule；
- EncryptedSessionNginxModule；
- FormInputNginxModule；
- HeadersMoreNginxModule；
- IconvNginxModule；
- StandardLuaInterpreter；
- MemcNginxModule；
- Nginx；
- NginxDevelKit；
- LuaCjsonLibrary；
- LuaNginxModule；
- LuaRdsParserLibrary；
- LuaRedisParserLibrary；
- LuaRestyCoreLibrary；
- LuaRestyDNSLibrary；
- LuaRestyLockLibrary；
- LuaRestyLrucacheLibrary；
- LuaRestyMemcachedLibrary；
- LuaRestyMySQLLibrary；
- LuaRestyRedisLibrary；
- LuaRestyStringLibrary；
- LuaRestyUploadLibrary；
- LuaRestyUpstreamHealthcheckLibrary；
- LuaRestyWebSocketLibrary；
- LuaRestyLimitTrafficLibrary；
- LuaUpstreamNginxModule；
- PostgresNginxModule；
- RdsCsvNginxModule；
- RdsJsonNginxModule；
- RedisNginxModule；
- Redis2NginxModule；

- RestyCLI；
- OPM；
- SetMiscNginxModule；
- SrcacheNginxModule；
- XssNginxModule。

3.3 OpenResty 的安装

本章介绍在 CentOS 6.x 上使用 yum 安装 OpenResty 方法，其他平台的安装方法请到官方网站（https://openresty.org/cn/installation.html）查看：

对于下列 Linux 发行版的种类和版本号，OpenResty 提供官方的预编译包。

1）RHEL/CentOS。版本号支持的体系结构：

5.x	x86_64, i386
6.x	x86_64, i386
7.x	x86_64

2）Fedora。版本号支持的体系结构：

23	x86_64, i386
24	x86_64, i386
25	x86_64, i386
26	x86_64, i386

1. 添加资源库

在 CentOS 上使用 yum 安装 OpenResty，需要首先安装资源库，这样就可以方便地安装 OpenResty，以后也可以更新（通过 yum update 命令）。

创建一个名为 /etc/yum.repos.d/OpenResty.repo 的文件，内容如下：

```
    [openresty]
    name=Official OpenResty Repository
    baseurl=https://copr-be.cloud.fedoraproject.org/results/openresty/openresty/
epel-$releasever-$basearch/
    skip_if_unavailable=True
    gpgcheck=1
    gpgkey=https://copr-be.cloud.fedoraproject.org/results/openresty/openresty/
pubkey.gpg
    enabled=1
    enabled_metadata=1
```

也可以直接运行下面命令添加仓库：

```
    sudo yum-config-manager --add-repo https://openresty.org/yum/centos/OpenResty.
repo
```

国内用户可以把 baseurl 改成下面的链接，速度会更快：

```
baseurl=https://openresty.org/yum/openresty/openresty/epel-$releasever-
$basearch/
```

或者运行下面命令直接添加仓库：

```
sudo yum-config-manager --add-repo https://openresty.org/yum/cn/centos/OpenResty.
repo
```

2. 列出所有包

使用下面命令列出资源库中所有的 OpenResty 包：

```
sudo yum --disablerepo="*" --enablerepo="openresty" list available
```

3. 安装

使用下面命令进行安装：

```
sudo yum install openresty
```

使用 yum 安装 OpenResty 可能会因为缺少 GeoIP 库失败，所以需要先运行下面命令安装 GeoIP：

```
yum install GeoIP-devel
```

GeoIP 库的安装可能会因为仓库里没有 Extra 库而失败，所以需要首先添加 Extra 库：

```
yum install epel-release
```

4. 测试

运行下面命令启动 Nginx：

```
/usr/local/openresty/nginx/sbin/nginx -p /usr/local/openresty/nginx/
```

在浏览器里输入 http://127.0.0.1 (或主机 IP)，看到“ Welcome to OpenResty !”表示已经启动成功。

可以进一步修改 /usr/local/openresty/nginx/conf/nginx.conf，测试 Lua 是否正常工作，nginx.conf 内容如下：

```
worker_processes  1;
error_log logs/error.log;
events {
    worker_connections 1024;
}
http {
    server {
        listen 8080;
        location / {
            default_type text/html;
            content_by_lua '
                ngx.say("<p>hello, world</p>")
            ';
        }
```

```
    }
}
```

然后运行下面命令重载配置文件：

```
/usr/local/openresty/nginx/sbin/nginx -p /usr/local/openresty/nginx/ -s reload
```

重载之前可以先测试一下配置文件的正确性：

```
/usr/local/openresty/nginx/sbin/nginx -p /usr/local/openresty/nginx/ -t
```

在浏览器里输入 http://127.0.0.1:8080，如果看到了“hello world”就表示可以正常工作了。也可以使用 CURL 工具测试：

```
curl http://localhost:8080/

<p>hello, world</p>
```

3.4 Nginx 多实例

OpenRestry 安装成功后，包里的 Nginx 可以部署多个实例，可以实例化多个不同的服务：或用于对外提供服务，或用于不同的开发任务，或用于学习。

只需要把 OpenRestry 中的 Nginx 目录复制一份就可以启动不同的实例：

```
cp -r /usr/local/openrestry/nginx  /usr/local/openrestry/nginx_9090
```

然后修改 nginx_9090/conf/nginx.conf，把端口从 8080 修改为 9090，把“hello world”修改为“hello world2”，修改完成后启动实例。

```
/usr/local/openrestry/nginx_9090/sbin/nginx -p /usr/local/openrestry/nginx_9090/
```

在浏览器里输入 http://127.0.0.1:9090，可以得到：

```
hello world2
```

表示新实例启动成功。

3.5 小结

本章介绍了 OpenResty 应用，并介绍了 OpenResty 的组成、安装方法；另外，为了方便应用，介绍了在 OpenResty 下多 Nginx 实例的方法。OpenResty 是一个流行的 Nginx 下 Lua 开发解决方案，使用非常广泛。通过本章的介绍，读者可以在后续的学习和工作中掌握 OpenResty 的使用方法，感受其带来的便利性。