Centos7 安装 Tengine

# Tengine介绍

Tengine是淘宝开发的基于nginx的应用服务器，反向代理服务器。它兼容nginx的所有配置，并且在其上增加了很多实用的功能，例如动态模块加载，集成lua语言进行扩展等，在性能方面有了比较大的改进。下面是Tengine独有的部分功能，查看全部等前往官网，地址是 [http://tengine.taobao.org/](http://tengine.taobao.org/" \t "_blank) 。

## Tengine部分特性

1. nginx -m 查看所有已经加载的模块，static说明已经安装的，shared说明是动态安装的；
2. nginx**[架构](http://lib.csdn.net/base/architecture" \t "_blank" \o "大型网站架构知识库)**是模块化的，除了主模块，其他模块都需在安装时进行编译安装，也可在安装后进行编译安装；
3. tengine完全兼容nginx的配置；
4. tengine实现动态加载模块，不需重新编译，就可以实现安装新的模块。动态模块功能会被默认安装。而nginx不支持模块动态加载，需要编译重启；
5. lua语言扩展功能，通过编程可以有效为应用服务器分流请求；
6. 强大的负载均衡功能，可以检测应用服务器的健康情况，动态的上下线。

# 安装Tengine

## 安装必要的编译环境好

# yum update

# yum install gcc gcc-c++ autoconf automake

## 安装需要的组件

### PCRE

PCRE(Perl Compatible Regular Expressions)是一个Perl库，包括 perl 兼容的正则表达式库。nginx rewrite依赖于PCRE库，所以在安装Tengine前一定要先安装PCRE，最新版本的PCRE可在官网（[http://www.pcre.org/](http://www.pcre.org/" \t "_blank)）获取。具体安装流程为：

cd /usr/local/src

wget ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/pcre-8.40.tar.gz

tar zxvf pcre-8.40.tar.gz

cd pcre-8.40

./configure --prefix=/usr/local/pcre

make && make install

附加信息：

源码的安装一般由3个步骤组成：配置(configure)、编译(make)、安装(make install)。

Configure是一个可执行脚本，它有很多选项，在待安装的源码路径下使用命令./configure –help输出详细的选项列表。其中–prefix选项是配置安装的路径，如果不配置该选项，安装后可执行文件默认放在/usr /local/bin，库文件默认放在/usr/local/lib，配置文件默认放在/usr/local/etc，其它的资源文件放在/usr /local/share，比较凌乱。

如果配置–prefix，如：./configure –prefix=/usr/local/test，可以把所有资源文件放在/usr/local/test的路径中，不会杂乱。

用了—prefix选项的另一个好处是卸载软件或移植软件。当某个安装的软件不再需要时，只须简单的删除该安装目录，就可以把软件卸载得干干净净；移植软件只需拷贝整个目录到另外一个机器即可（相同的**[操作系统](http://lib.csdn.net/base/operatingsystem" \t "_blank" \o "操作系统知识库)**）。当然要卸载程序，也可以在原来的make目录下用一次make uninstall，但前提是make文件指定过uninstall。

### OpenSSL

OpenSSL 是一个强大的安全套接字层密码库，囊括主要的密码**[算法](http://lib.csdn.net/base/datastructure" \t "_blank" \o "算法与数据结构知识库)**、常用的密钥和证书封装管理功能及SSL协议，并提供丰富的应用程序供**[测试](http://lib.csdn.net/base/softwaretest" \t "_blank" \o "软件测试知识库)**或其它目的使用。，安装OpenSSL（[http://www.openssl.org/source/](http://www.openssl.org/source/" \t "_blank)）主要是为了让tengine支持Https的访问请求。具体是否安装看需求。

复制代码 代码如下:

cd /software/tools

wget http://www.openssl.org/source/openssl-1.0.1h.tar.gz

tar zxvf openssl-1.0.1h.tar.gz

cd openssl-1.0.1h.tar.gz

./config --prefix=/usr/local/openssl -fPIC --shared

make

make install

mv /usr/bin/openssl /usr/bin/openssl.1.0.1e

mv /usr/include/openssl /usr/include/openssl.bak

ln -s /usr/local/openssl/bin/openssl /usr/bin/openssl

ln -s /usr/local/openssl/include/openssl /usr/include/openssl

echo "/usr/local/openssl/lib" >> /etc/ld.so.conf

ldconfig -v

### Zlib

Zlib是提供资料压缩之用的函式库，当Tengine想启用GZIP压缩的时候就需要使用到Zlib（[http://www.zlib.net/](http://www.zlib.net/" \t "_blank)）。

cd /usr/local/src

wget http://zlib.net/zlib-1.2.11.tar.gz

tar zxvf zlib-1.2.11.tar.gz

cd zlib-1.2.11.tar.gz

./configure --prefix=/usr/local/zlib

make && make install

### jemalloc

jemalloc（[http://www.canonware.com/jemalloc/](http://www.canonware.com/jemalloc/" \t "_blank)）是一个更好的内存管理工具，使用jemalloc可以更好的优化Tengine的内存管理。

cd /usr/local/src

wget http://www.canonware.com/download/jemalloc/jemalloc-3.6.0.tar.bz2

tar jxvf jemalloc-3.6.0.tar.bz2

cd jemalloc-3.6.0.tar.bz2

./configure --prefix=/usr/local/jemalloc

make && make install

## 安装Tengine

在主要核心的组件安装完毕以后就可以安装Tegine了，最新版本的Tegine可从官网([http://tengine.taobao.org/](http://tengine.taobao.org/" \t "_blank))获取。   
在编译安装前还需要做的一件事是添加一个专门的用户来执行Tengine。当然你也可以用root（不建议）。   
复制代码 代码如下:

groupadd tengine

useradd -s /sbin/nologin -g tengine tengine

接下来才是进行安装：

cd /usr/local/src

wget http://tengine.taobao.org/download/tengine-2.2.0.tar.gz

tar -zxvf tengine-2.2.0.tar.gz

cd tengine-2.2.0

./configure --prefix=/software/tengine --user=tengine --group=tengine --with-pcre=/software/tools/pcre-8.40 --with-openssl=/software/tools/openssl-1.0.2 --with-jemalloc=/software/tools/jemalloc-3.6.0 --with-zlib=/software/tools/zlib-1.2.11 --with-http\_gzip\_static\_module --with-http\_realip\_module --with-http\_access\_module=shared --with-http\_footer\_filter\_module=shared --with-http\_split\_clients\_module=shared --with-http\_addition\_module=shared --with-http\_limit\_conn\_module=shared --with-http\_referer\_module=shared --with-http\_sub\_module=shared --with-http\_concat\_module=shared --with-http\_limit\_req\_module=shared --with-http\_rewrite\_module=shared --with-http\_sysguard\_module=shared --with-http\_stub\_status\_module --with-http\_reqstat\_module=shared

make && make install

注意配置的时候 –with-pcre 、–with-openssl、–with-jemalloc、–with-zlib的路径为源文件的路径。

# Tengine启动

## 设置tengine开机自启

* 系统用户登录系统后启动的服务的目录

/usr/lib/systemd/system

* 如需要开机没有登陆情况下就能运行的程序在系统目录内

/usr/lib/systemd/system

* 希望系统开机就启动目录，把文件放在系统目录内。

vim /lib/systemd/system/nginx.service 创建文件

[Unit]

Description=The nginx HTTP and reverse proxy server

After=syslog.target network.target remote-fs.target nss-lookup.target

[Service]

Type=forking

PIDFile=/usr/local/nginx/logs/nginx.pid

ExecStartPre=/software/tengine/sbin/nginx -t

ExecStart=/software/tengine/sbin/nginx -c /software/tengine/conf/nginx.conf

ExecReload=/bin/kill -s HUP $MAINPID

ExecStop=/bin/kill -s QUIT $MAINPID

PrivateTmp=true

[Install]

WantedBy=multi-user.target

* 修改文件权限

chmod 745 nginx.service

* 设置为开机启动

systemctl enable nginx.service

* 设置开机自启动

systemctl enable nginx.service

* 停止开机自启动

systemctl disable nginx.service

## 启动nginx服务

systemctl start nginx.service

## 查看服务当前状态

systemctl status nginx.service

## 重新启动服务

systemctl restart nginx.service

## 查看所有已启动的服务

systemctl list-units --type=service

# tengine配置

## 全局配置

user nobody;

worker\_processes 2;

worker\_rlimit\_nofile 65535;

dso {

# load ngx\_pagespeed.so;

load ngx\_http\_rewrite\_module.so;

load ngx\_http\_access\_module.so;

load ngx\_http\_concat\_module.so;

load ngx\_http\_limit\_conn\_module.so;

load ngx\_http\_limit\_req\_module.so;

load ngx\_http\_sysguard\_module.so;

}

error\_log logs/error.log;

#google\_perftools\_profiles /var/tmp/tcmalloc;

events {

worker\_connections 65535;

use epoll;

}

http {

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log logs/access.log main;

server\_tokens off;

server\_tag off;

autoindex off;

access\_log off;

include mime.types;

default\_type application/octet-stream;

server\_names\_hash\_bucket\_size 128;

client\_header\_buffer\_size 32k;

large\_client\_header\_buffers 4 32k;

client\_max\_body\_size 10m;

client\_body\_buffer\_size 256k;

sendfile on;

tcp\_nopush on;

keepalive\_timeout 60;

tcp\_nodelay on;

## 压缩配置

gzip on;

gzip\_min\_length 1k;

gzip\_buffers 4 16k;

gzip\_http\_version 1.0;

gzip\_comp\_level 2;

gzip\_types text/plain application/x-javascript text/css application/xml;

gzip\_vary on;

## 反向代理配置

proxy\_redirect off;

proxy\_set\_header Host $http\_host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_connect\_timeout 90;

proxy\_send\_timeout 90;

proxy\_read\_timeout 90;

proxy\_buffer\_size 16k;

proxy\_buffers 4 64k;

proxy\_busy\_buffers\_size 128k;

proxy\_temp\_file\_write\_size 128k;

###反向代理缓存

proxy\_cache\_path /software/tengine/proxy\_cache levels=1:2 keys\_zone=http-cache:500m max\_size=10g inactive=30d;

proxy\_temp\_path /tmp/proxy\_cache;

client\_body\_temp\_path /tmp/client\_body\_temp 3 2;

## 集群配置

upstream jdbsbzs {

###基于sticky的会话保持

session\_sticky;

###后端服务器

server 172.18.11.60:8080 max\_fails=3 fail\_timeout=5s;

server 172.18.11.70:8080 max\_fails=3 fail\_timeout=5s;

###后端服务状态检测

check interval=3000 rise=2 fall=5 timeout=1000 type=http;

check\_http\_send "HEAD / HTTP/1.0";

check\_http\_expect\_alive http\_2xx http\_3xx;

}

## 虚拟主机配置

server {

listen 80; # default\_server;

server\_name gdcsgjdjb.jchl.com;

# Load configuration files for the default server block.

include /etc/nginx/default.d/\*.conf;

access\_log logs/gdcsgjdjb.access.log main;

###通用匹配

location /sbzs-cjpt-web {

proxy\_buffer\_size 64k;

proxy\_buffers 32 32k;

proxy\_busy\_buffers\_size 128k;

proxy\_pass http://jdbsbzs ;

proxy\_next\_upstream http\_502 http\_504 error timeout invalid\_header;

proxy\_set\_header Host $host:$server\_port;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

client\_max\_body\_size 10m;

}

###正则表达式，静态图片等使用本地目录，缓存7天

location ~/sbzs-cjpt-web/resources/.\*\.(gif|jpg|jpeg|png|bmp|swf|eot|ttf|woff)$ {

root html;

expires 7d;

}

###正则表达式，js、html等使用本地目录，缓存1天

location ~/sbzs-cjpt-web/resources/.\*\.(js|css|htm|html|json)$ {

root html;

expires 1d;

}

｝