```
## set uid and gid of process
user webproxy ngboss;
## how many process will be started
worker_processes 10;
## worker_cpu_affinity define
worker_cpu_affinity   000000000100   000000001000   000000010000   000000100000
000001000000 000010000000 000100000000 001000000000 010000000000 100000000000;
## how many open files will be allowd of each process
worker_rlimit_nofile 51200;
## error log defind
error_log  logs/error.log crit;
## save master process-id in file
pid        logs/nginx.pid;
events {
    ## powered by epoll, good!
    use epoll;
    worker_connections 51200;
}
http {
    include       mime.types;
    default_type  text/html;
    ## access log format defind
    log_format      main      '$remote_addr   [$time_local]   $request   $status
$body_bytes_sent';
    ## access log defind
    access_log  logs/access.log  main;
    ## fast send file system call, good!
    sendfile        on;
    tcp_nopush      on;
    tcp_nodelay         on;
    client_body_buffer_size     1024k;
    proxy_connect_timeout       600;
    proxy_read_timeout      600;
    proxy_send_timeout      600;
    proxy_buffer_size       8k;
    proxy_buffers           4 32k;
    proxy_busy_buffers_size     64k;
    proxy_temp_file_write_size  1024k;

    ## set connection timeout (by second)
    keepalive_timeout  30;
    ## gzip setting begin.
    gzip            on;
    gzip_min_length     1k;
```

```
gzip_buffers        4 16k;
gzip_http_version   1.1;
gzip_comp_level     9;
gzip_vary       off;
gzip_types      text/plain text/javascript text/css text/xml application/xml;
## gzip setting end.
output_buffers      4 32k;
postpone_output     1460;
client_header_buffer_size       128k;
large_client_header_buffers     4 256k;
## default encoding
    # charset GBK;
## app redirect && load-balancer start
# ngboss cluster
upstream ngboss_cluster {
    ip_hash;
    server 10.238.15.65:7101;
    server 10.238.15.66:7201;
    server 10.238.15.67:7301;
    server 10.238.15.68:7401;
}
# saleserv cluster
upstream saleserv_cluster {
    ip_hash;
    server 10.238.15.65:8181;
    server 10.238.15.66:8281;
    server 10.238.15.67:8381;
    server 10.238.15.68:8481;
    server 10.238.15.65:8182;
    server 10.238.15.66:8282;
    server 10.238.15.67:8382;
    server 10.238.15.68:8482;
}
# acctmanm cluster
upstream acctmanm_cluster {
    ip_hash;
    server 10.238.15.65:8191;
    server 10.238.15.66:8291;
    server 10.238.15.67:8391;
    server 10.238.15.68:8491;
    server 10.238.15.65:8192;
    server 10.238.15.66:8292;
    server 10.238.15.67:8392;
    server 10.238.15.68:8492;
```

```
}
# custmanm cluster
upstream custmanm_cluster {
    ip_hash;
    server 10.238.15.65:8111;
    server 10.238.15.66:8211;
}
# groupserv cluster
upstream groupserv_cluster {
    ip_hash;
    server 10.238.15.65:8183;
    server 10.238.15.66:8283;
}
# salemanm cluster
upstream salemanm_cluster {
    ip_hash;
    server 10.238.15.65:8121;
    server 10.238.15.66:8221;
}
# chnlmanm cluster
upstream chnlmanm_cluster {
    ip_hash;
    server 10.238.15.65:8101;
    server 10.238.15.66:8201;
}
# resmanm cluster
upstream resmanm_cluster {
    ip_hash;
    server 10.238.15.65:8131;
    server 10.238.15.66:8231;
}
# prodmcrm prodmbil bilmanm cluster
upstream prodmanm_cluster {
    server 10.238.15.66:8261;
}
# copmanm cluster
upstream copmanm_cluster {
    server 10.238.15.66:8271;
}
# sysmanm cluster
upstream sysmanm_cluster {
    ip_hash;
    server 10.238.15.65:8141;
    server 10.238.15.66:8241;
```

```
    }
    # statmanm cluster
    upstream statmanm_cluster {
        ip_hash;
        server 10.238.15.65:8151;
        server 10.238.15.66:8251;
    }
    ## app redirect && load-balancer end
    server {
        listen       10.238.15.101:18080;
        server_name  10.238.15.101;
        proxy_set_header   X-Forwarded-For  $remote_addr;
                # charset GBK;
        location /download {
            root html;
            proxy_redirect off;
        }
        location /saleserv {
                        if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                                proxy_pass http://saleserv_cluster;
                                expires max;
                                break;
                        }
            proxy_pass  http://saleserv_cluster;
            proxy_redirect  off;
        }
        location /acctmanm {
                        if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                                proxy_pass http://acctmanm_cluster;
                                expires max;
                                break;
                        }
            proxy_pass  http://acctmanm_cluster;
            proxy_redirect  off;
        }
        location /custmanm {
                        if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                                proxy_pass http://custmanm_cluster;
                                expires max;
                                break;
                        }
```

```
            proxy_pass  http://custmanm_cluster;
            proxy_redirect  off;
        }
        location /groupserv {
                        if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                                proxy_pass http://groupserv_cluster;
                                expires max;
                                break;
                        }
            proxy_pass  http://groupserv_cluster;
            proxy_redirect  off;
        }
        location /salemanm {
                        if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                                proxy_pass http://salemanm_cluster;
                                expires max;
                                break;
                        }
            proxy_pass  http://salemanm_cluster;
            proxy_redirect  off;
        }
        location /chnlmanm {
                        if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                                proxy_pass http://chnlmanm_cluster;
                                expires max;
                                break;
                        }
            proxy_pass  http://chnlmanm_cluster;
            proxy_redirect  off;
        }
        location /resmanm {
                        if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                                proxy_pass http://resmanm_cluster;
                                expires max;
                                break;
                        }
            proxy_pass  http://resmanm_cluster;
            proxy_redirect  off;
        }
        location /prodmcrm {
```

```
                if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                        proxy_pass http://prodmanm_cluster;
                        expires max;
                        break;
                }
            proxy_pass  http://prodmanm_cluster;
            proxy_redirect  off;
        }
        location /prodmbil {
                if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                        proxy_pass http://prodmanm_cluster;
                        expires max;
                        break;
                }
            proxy_pass  http://prodmanm_cluster;
            proxy_redirect  off;
        }
        location /bilmanm {
                if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                        proxy_pass http://prodmanm_cluster;
                        expires max;
                        break;
                }
            proxy_pass  http://prodmanm_cluster;
            proxy_redirect  off;
        }
        location /copmanm {
                if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                        proxy_pass http://copmanm_cluster;
                        expires max;
                        break;
                }
            proxy_pass  http://copmanm_cluster;
            proxy_redirect  off;
        }
        location /sysmanm {
                if                      ($request_uri                   ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                        proxy_pass http://sysmanm_cluster;
                        expires max;
```

```
                        break;
                    }
            proxy_pass  http://sysmanm_cluster;
            proxy_redirect  off;
        }
        location /statmanm {
                    if                    ($request_uri                    ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                        proxy_pass http://statmanm_cluster;
                        expires max;
                        break;
                    }
            proxy_pass  http://statmanm_cluster;
            proxy_redirect  off;
        }
        location /nginxstatus {
            stub_status on;
            access_log off;
            allow all;
        }
        location / {
                    if                    ($request_uri                    ~*
".*\.(js|css|gif|jpg|jpeg|png|bmp|swf)$") {
                        proxy_pass http://ngboss_cluster;
                        expires max;
                        break;
                    }
            proxy_pass  http://ngboss_cluster;
            proxy_redirect  off;
        }

        # redirect server error pages to the static page /50x.html
        error_page 500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        } <br>
```

Nginx 介绍

-与 weblogic 集群

2014 年 5 月

版本历史

日期	版本号	作者/修改者	描述	审核人

目　录

1 前言

1.1 文档说明

本文主要介绍什么是 nginx，nginx 在 linux 系统上源码安装、部署，优化、启用、停用操作，及与 weblogic 集群部署。

2 Nginx 介绍

Nginx ("engine x") 是一个高性能的 HTTP 和 反向代理 服务器，也是一个 IMAP/POP3/SMTP 代理服务器。

它具有有很多非常优越的特性:

作为 **Web** 服务器：相比 Apache，Nginx 使用更少的资源，支持更多的并发连接，体现更高的效率，这点使 Nginx 尤其受到虚拟主机提供商的欢迎。能够支持高达 50,000 个并发连接数的响应。

作为负载均衡服务器：Nginx 既可以在内部直接支持 Rails 和 PHP，也可以支持作为 HTTP 代理服务器 对外进行服务。Nginx 用 C 编写, 不论是系统资源开销还是 CPU 使用效率都比高。

作为邮件代理服务器**:** Nginx 同时也是一个非常优秀的邮件代理服务器（最早开发这个产品的目的之一也是作为邮件代理服务器）。

Nginx 安装非常的简单，配置文件 非常简洁（还能够支持 **perl** 语法），**Bugs** 非常少的服务器**:** Nginx 启动特别容易，并且几乎可以做到 7*24 不间断运行，即使运行数个月也不需要重新启动。你还能够在 不间断服务的情况下进行软件版本的升级。

3 Nginx 安装

3.1 前期准备

序号	名称	作用	备注
1	Linux	操作系统	
2	gcc g++	编译软件	系统自带
3	openssl	SSL 功能	需要下载安装
4	Pcre 库	Rewrite 模块	需要下载安装
5	Zlib 库	Gzlib 模块	系统自带
6	Nginx 源代码		

3.2 安装 openssl

1、进入网址 http://www.openssl.org/source，下载需要安装的 openssl 版本，并存放到 linux 中的/usr/local 目录下，本文以 openssl-1.0.0a.tar.gz 作安装源文件介绍。

2、进入控制命令台

```
cd /usr/local                                    //存放要下载程序的路径
tar -zxvf openssl-1.0.0a.tar.gz                  //解压程序
cd openssl-1.0.0a                                //进入程序文件夹内
./config                                                    // 安 装
./config                                                          -t
make                                                           depend
make
make                                                              test
make install
```

安装之后在/usr/local 下生成一个 ssl 目录。

3、配置环境变量

设置环境变量，在/etc/profile 的 PATH 中增加如下内容：

```
PATH=/usr/local/ssl/bin:/sbin/:$PATH:/usr/sbin
export PATH
```

4、重启系统。

3.3 安装 pcre

1、进入网址 http://sourceforge.net/，下载需要安装的 pcre 版本，并存放到 linux 中的/usr/local 目录下，本文以 pcre-8.02.tar.gz 作安装源文件介绍。

2、进入控制台

```
cd /usr/local
tar -zxvf pcre-8.02.tar.gz                    //解压程序
cd pcre-8.02                                  //进入安装目录
./configure --prefix=/usr/local/pcre          //安装程序到/usr/local/pcre 目录
make
make install
```

3.4 安装 nginx 源代码

1、进入网址 http://nginx.org/en/download.html，下载需要安装的 nginx 版本，并存放到 linux 中的/usr/local 目录下，本文以 nginx-1.6.0.tar.gz 作安装源文件介绍。

2、进入控制台

```
cd /usr/local/
tar -zxvf nginx-1.6.0.tar.gz
cd nginx-1.6.0.
cd /usr/local/nginx-0.8.36
./configure --prefix=/usr/local/nginx --with-pcre=/usr/local/pcre-8.02 --with-http_ssl_module
--with-openssl=/usr/local/openssl-0.9.8o
make
make install
```

3、更多的安装配置说明

```
./configure                                    --prefix=/usr/local/nginx
--with-openssl=/usr/include                    (启用ssl)
--with-pcre=/usr/include/pcre/                 (启用正规表达式)
--with-http_stub_status_module                 (安装可以查看nginx状态的程序)
--with-http_memcached_module                   (启用memcache缓存)
--with-http_rewrite_module (启用支持 url 重写)
```

4 Nginx 启动、停止、重启

4.1 启动

1、进入 nginx 安装目录

```
cd /usr/local/nginx/sbin                                  //进入程序的目录
./nginx   -t                                              //检查配置文件是否正确
//或者 nginx -t -c /usr/nginx/conf/nginx.conf             //检查配置文件是否正确
./nginx                                                   //运行程序
```

4.2 关闭

停止操作停止操作是通过向 nginx 进程发送信号来进行的

进入控制台

```
ps -ef | grep nginx                                       //查询 nginx 的主进程号
kill -QUIT  主进程号                                      //从容停止 Nginx
kill -TERM  主进程号                                      //快速停止 Nginx
kill -9  主进程号                                         //强制停止 Nginx
```

另外， 若在 nginx.conf 配置了 pid 文件存放路径则该文件存放的就是 Nginx 主进程号，如果没指定则放在 nginx 的 logs 目录下。有了 pid 文 件，我们就不用先查询 Nginx 的主进程号，而直接向 Nginx 发送信号了，命令如下：

```
kill -信号类型  '/usr/nginx/logs/nginx.pid'
```

4.2 重启

进入控制命令台

```
ps -ef | grep nginx                                       //查询 nginx 的主进程号
kill -HUP  主进程号或进程号文件路径                       //重启命令
//或者
/usr/nginx/sbin/nginx -s reload
```

5 Nginx 配置说明

```
#运行用户
user nobody;      #nginx  启动使用的用户，配置 fastcgi 时，需要改为有权限执行 fastcgi 的用
```

```
户
#启动进程,通常设置成和 cpu 的数量相等
worker_processes  1;
#全局错误日志及 PID 文件
#error_log  logs/error.log;                    #错误日志，相对于/usr/local/nginx
#error_log  logs/error.log  notice;            #记录警告日志，相对于/usr/local/nginx，可改为
logs/notice.log
#error_log  logs/error.log  info;              #记录信息日志，相对于/usr/local/nginx，可改为
logs/info.log
#pid        logs/nginx.pid;                    #进程文件，最好不要改
#工作模式及连接数上限
events {
    #epoll 是多路复用 IO(I/O Multiplexing)中的一种方式,
    #仅用于 linux2.6 以上内核,可以大大提高 nginx 的性能
    use   epoll;
    #单个后台 worker process 进程的最大并发链接数
    worker_connections  1024;
    # 并发总数是 worker_processes 和 worker_connections 的乘积
    # 即 max_clients = worker_processes * worker_connections
    # 在设置了反向代理的情况下，max_clients = worker_processes * worker_connections / 4
为什么
    # 为什么上面反向代理要除以 4，应该说是一个经验值
    # 根据以上条件，正常情况下的 Nginx Server 可以应付的最大连接数为：4 * 8000 =
32000
    # worker_connections 值的设置跟物理内存大小有关
    # 因为并发受 IO 约束，max_clients 的值须小于系统可以打开的最大文件数
    # 而系统可以打开的最大文件数和内存大小成正比，一般 1GB 内存的机器上可以打开
的文件数大约是 10 万左右
    # 我们来看看 360M 内存的 VPS 可以打开的文件句柄数是多少：
    # $ cat /proc/sys/fs/file-max
    # 输出 34336
    # 32000 < 34336，即并发连接总数小于系统可以打开的文件句柄总数，这样就在操作系
统可以承受的范围之内
    # 所以，worker_connections 的值需根据 worker_processes 进程数目和系统可以打开的
最大文件总数进行适当地进行设置
    # 使得并发总数小于操作系统可以打开的最大文件数目
    # 其实质也就是根据主机的物理 CPU 和内存进行配置
    # 当然，理论上的并发总数可能会和实际有所偏差，因为主机还有其他的工作进程需要
消耗系统资源。
    #ulimit -n 查看
    # ulimit -SHn 65535
}

http {
```

```
#设定 mime 类型,类型由 mime.type 文件定义
include       mime.types;
default_type    application/octet-stream;
#设定日志格式
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;
#sendfile 指令指定 nginx 是否调用 sendfile 函数（zero copy 方式）来输出文件，
#对于普通应用，必须设为 on,
#如果用来进行下载等应用磁盘 IO 重负载应用，可设置为 off，
#以平衡磁盘与网络 I/O 处理速度，降低系统的 uptime.
sendfile        on;
#tcp_nopush        on;
#连接超时时间
#keepalive_timeout    0;
keepalive_timeout    65;
tcp_nodelay        on;
#开启 gzip 压缩
gzip    on;
gzip_disable "MSIE [1-6].";
#设定请求缓冲
client_header_buffer_size       128k;
large_client_header_buffers    4 128k;

#设定虚拟主机配置
server {
#侦听 80 端口
listen       80;
#定义使用 www.nginx.cn 访问       服务器名，可以是主机 IP 地址
server_name    www.nginx.cn;
#定义服务器的默认网站根目录位置
root html;
#charset koi8-r;    #默认字符集
      #设定本虚拟主机的访问日志
access_log    logs/nginx.access.log    main;
      #默认请求
location / {
            #定义首页索引文件的名称
index index.php index.html index.htm;
      }
      # 定义错误提示页面
error_page      500 502 503 504 /50x.html;
location = /50x.html {
```

```
        }

        #静态文件，nginx 自己处理
        location ~ ^/(images|javascript|js|css|flash|media|static)/ {
            #过期 30 天，静态文件不怎么更新，过期可以设大一点，
    #如果频繁更新，则可以设置得小一点。
    expires 30d;
    }
        #禁止访问 .htxxx 文件
    location ~ /.ht {                deny all;            }
    }
}
```

6 Nginx 与 weblogic 集群

6.1 前期准备

准备两台 weblogic 服务器，同时部署相同的应用，只是主页显示文字有些区别。

服务器	IP 地址与端口号	部署应用	备注
Weblogic_1	172.16.0.156:7001	nginxtest	
Weblogic_2	172.16.0.154:7001	nginxtest	
Nginx	172.16.0.156:80		作为集群主 web 服务器

6.2 配置 nginx

1、进入 nginx 的安装目录,修改/nginx/conf/nginx.conf 文件，在“http{……}”区域内增加如下内容

```
#设定负载均衡的服务器列表
upstream mysvr {
#weigth 参数表示权值,权值越高被分配到的几率越大
   server 172.16.0.156:7001 weight=5;
   server 172.16.0.154:7001 weight=5;
}
```

在“server{……}”区域内增加如下内容

```
location /nginxtest {
    proxy_pass http://mysvr;  #以这种格式来使用后端的 web 服务器
    proxy_redirect off;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    client_max_body_size 10m;
    client_body_buffer_size 128k;
    proxy_connect_timeout 90;
    proxy_send_timeout 90;
    proxy_read_timeout 90;
    proxy_buffer_size 4k;
    proxy_buffers 4 32k;
    proxy_busy_buffers_size 64k;
    proxy_temp_file_write_size 64k;
}
```

6.3 运行测试环境

1、启动 weblongic 服务器与 nginx 服务器，在浏览器中分别输入如下网址。

```
http://172.16.0.156:80/nginxtest
http://172.16.0.156:7001/nginxtest
http://172.16.0.154:7001/nginxtest
```

2、关闭当前 nginx 访问的 weblogic 服务器，再次使用浏览器方位 ngingx 服务器。

相关命令：

```
./configure --help
nohup  运行程序  > info.log &
tail -f info.log
```

```
date --set ‘20140508 18:15:30’
./nginx -V
./nginx -v
ps -ef | grep nginx
./nginx -t
./nginx -h
```

参考网址：

http://baike.baidu.com/view/926025.htm?fr=aladdin

http://www.nginx.cn/