Nginx 默认端口号80，与http协议端口号相同

nginx/sbin/nginx 根目录，直接从根目录启动nginx，ps aux|grep nginx命令查看nginx是否启动，tail –f 动态的查看文件内容，vim编辑文件

keepalived，nginx监听软件，每个nginx服务器需要单独配一个keepalived，nginx的主从由keepalived进行管理，在其config文件中配置state属性，主节点为master，从节点为backup

keepalived中配置一个虚拟ip，集群内所有nginx都使用相同的虚拟ip，访问时由虚拟ip转移到对应的master节点，如果主节点挂掉可以直接使用从节点而不用手动改写ip地址

keepalived的心跳配置，通过脚本检测nginx的进程个数，如果进程数为零则nginx宕机，会自动启动nginx，然后重新检测nginx的进程数，如果启动失败，则关闭keepalived

# 1、nginx的安装与集群搭建

## 1、1nginx的安装

安装环境准备：

1、linux内核2.6及以上版本

只有2.6之后才支持epool，在此之前使用select或pool多路复用的IO模型，无法解决高并发压力的问题。通过命令uname –a 即可查看linux内核

2、GCC编译期

GCC（GNU Compiler Collection）可用来编译C语言程序。Nginx不会直接提供二进制可执行程序,只能下载源码进行编译。

3、PCRE库

PCRE（Perl Compatible Regular Expressions，Perl兼容正则表达式）是由Philip Hazel开发的函数库，目前为很多软件所使用，该库支持正则表达式。

4、zlib库

zlib库用于对HTTP包的内容做gzip格式的压缩，如果我们在nginx.conf里配置了gzip on，并指定对于某些类型（content-type）的HTTP响应使用gzip来进行压缩以减少网络传输量

5、OpenSSL开发库

如果我们的服务器不只是要支持HTTP，还需要在更安全的SSL协议上传输HTTP，那么就需要拥有OpenSSL了。另外，如果我们想使用MD5、SHA1等散列函数，那么也需要安装它。

上面几个库都是Nginx 基础功能所必需的，为简单起见我们可以通过yum 命令统一安装。

#yum 安装nginx 环境

yum -y install make zlib zlib-devel gcc-c++ libtool openssl openssl-devel pcre pcre-devel

**获取nginx源码：**

nginx 下载页：http://nginx.org/en/download.html 。

# 下载nginx 最新稳定版本

wget http://nginx.org/download/nginx-1.14.0.tar.gz

#解压

cd /usr/local/nginx

tar -zxvf nginx-1.14.0.tar.gz

最简单的安装：

# 全部采用默认安装

./configure & make & make install

make & make install

执行完成之后 nginx 运行文件 就会被安装在 /usr/local/nginx 下。

**模块更新：**

# 添加状态查查看模块

./configure --with-http\_stub\_status\_module

# 重新创建主文件

make

# 将新生成的nginx 文件覆盖 旧文件。

cp objs/nginx /usr/local/nginx/sbin/

# 查看是否更新成功 显示了 configure 构建参数表示成功

/usr/local/nginx/sbin/nginx –V

**控制命令：**

#查看命令帮助

./sbin/nginx -?

#默认方式启动：

./sbin/nginx

#指定配置文件启动

./sbin/nginx -c /tmp/nginx.conf

#指定nginx程序目录启动

./sbin/nginx -p /usr/local/nginx/

#快速停止

./sbin/nginx -s stop

#优雅停止

./sbin/nginx -s quit

# 热装载配置文件

./sbin/nginx -s reload

# 重新打开日志文件

./sbin/nginx -s reopen

# 设置全局命令，如下表示设置启动用户为root

./sbin/nginx -g "user root;"

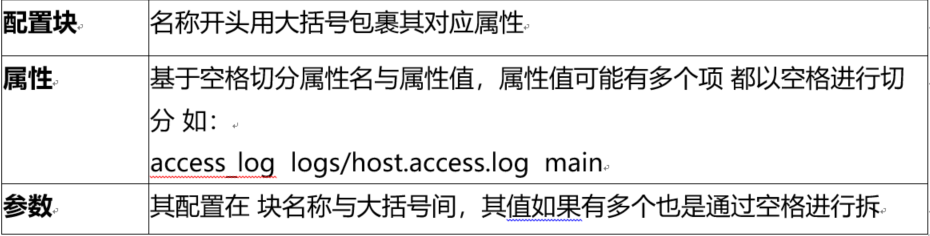
## 1.2 nginx的集群搭建

# 2、nginx的使用

## 2.1 nginx的配置

Nginx可实现的功能：动静分离、防盗链、多域名站点、下载限速、IP黑名单、基于user-agent分流

配置中events、http、server、location、upstream等属于配置块。而worker\_processes、worker\_connections、include、listen属于配置块中的属性。其中server块嵌套于http块，其可以直接访问继承http块当中的参数



注意 如果配置项值中包括语法符号，比如空格符，那么需要使用单引号或双引号括住配置项值，否则Nginx会报语法错误。例如：

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

                     '$status $body\_bytes\_sent "$http\_referer" ' '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

基本配置介绍：

1、监听端口

语法：listen address

默认：listen 80

配置块：server

2、主机名称

语法：server\_name name[……]

默认：server\_name “”;

Server\_name [www.testweb.com](http://www.testweb.com) download.testweb.com;支持通配符与正则

配置块：server

3、location

语法：[=|~|~\*|^~|@]/uri/{……}

配置块：server

1. =表示把URI作为字符串，以便与参数中的uri做完全匹配

2. /基于uri目录匹配

3. ~表示正则匹配URI时字母大小写敏感

4. ~\*表示正则匹配URI时忽略大小写问题

5. ^~表示匹配URI时只需要前半部分与uri参数匹配即可

匹配的优先规则：

a、精确匹配优先= b、正则匹配优先^~ c、前缀最大匹配优先 d、配置靠前优化

4、root指定站点目录

可配置在server与location中，基于root路径+url中路径去寻找指定文件

5、alias指定站点别名

只能配置于location中，基于alias路径+URL移除location前缀后的路径来寻找文件

如下示例：

location /V1 {

      alias  /www/old\_site;

      index  index.html index.htm;

}

#防问规则如下

URL：http://xxx:xx/V1/a.html

最终寻址：/www/old\_site/a.thml

6、**动静分离实现**

A、创建静态站点 b、配置location/static c、配置~\*\.(gif|png|css|js)$

server {

        listen 80;

        server\_name \*.luban.com;

        root /usr/www/luban;

        location / {

                index luban.html;

        }

        location /static {

         alias /usr/www/static;

        }

 }

基于正则动静分离

location ~\* \.(gif|jpg|png|css|js)$ {

      root /usr/www/static;

}

7、防盗链配置

# 加入至指定location 即可实现

valid\_referers none blocked \*.luban.com;

 if ($invalid\_referer) {

       return 403;

}

8、**下载限速**

location /download {

    limit\_rate 1m; //限制每S下载速度

limit\_rate\_after 30m; // 超过30 m 后下载限速

}

9、创建IP黑名单

#封禁指定IP

deny 192.168.0.1;

allow 192.168.0.1;

#开放指定IP 段

allow 192.168.0.0/24;

#封禁所有

deny all;

#开放所有

allow all;

# 创建黑名单文件

echo 'deny 192.168.0.132;' >> black.ip

#http 配置块中引入 黑名单文件

include       black.ip;

3、日志配置：

日志格式（log\_format 日志格式，main日志格式名称 具体的日志格式参数）：

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;

$remote\_addr：客户端ip地址

$remote\_user：客户端用户名

$request：记录请求的URL与http协议（get，post，del，put等）

$status：记录请求状态（如200）

$body\_bytes\_sent：发送给客户端的字节数，不包括响应头的大小

$ bytes\_sent：发送给客户端的总字节数

$connection:连接的序列号

$connection\_requests:当前通过一个连接获得的请求数量

$msec：日志写入时间。单位为s，精度是毫秒

$pipe：如果请求是通过HTTP流水线（pipelined）发送，pipe值为“p”，否则为“.”

$http\_referer:记录从哪个页面连接访问过来的

$http\_user\_agent:记录客户端浏览器相关信息

$request\_length 请求的长度（包括请求行，请求头和请求正文）

$request\_time:请求处理时间，单位秒，精度毫秒；从读入客户端的第一个字节开始，直到把最后一个字符发送给客户端后进行日志写入为止

$time\_iso8601 ISO8601 标准格式下的本地时间

$time\_local 通用日志格式下的本地时间

#基于域名打印日志

access\_log logs/$host.access.log main;

error日志的设置

语法：error\_log /path/file level;

默认：error\_log logs/error.log error;

level是日志的输出级别，取值范围是debug、info、notice、warn、error、crit、alert、emerg，

针对指定的客户端输出debug级别的日志

语法：debug\_connection[IP|CIDR]

events {

debug\_connection 192.168.0.147;

debug\_connection 10.224.57.0/200;

}

注意：debug 日志开启 必须在安装时 添加 --with-debug (允许debug)

#### open\_log\_file\_cache

使用open\_log\_file\_cache来设置日志文件缓存(默认是off)。

* max:设置缓存中的最大文件描述符数量，如果缓存被占满，采用LRU算法将描述符关闭。
* inactive:设置存活时间，默认是10s
* min\_uses:设置在inactive时间段内，日志文件最少使用多少次后，该日志文件描述符记入缓存中，默认是1次
* valid:设置检查频率，默认60s
* off：禁用缓存

