爱维 Linux 公开课

企业 Web 架构应用案例分享（2）

主讲人：南非蚂蚁

课程安排

- **LNMP 的安装以及排错经验**
- **Nginx 企业常用功能实战**
- **Nginx 作为 web 缓存服务器应用案例**
- **Nginx 作为负载均衡服务器应用案例**
- **Nginx 应用架构经验谈**

一、LNMP 架构的部署注意事项

1、nginx 的安装过程

建议安装 pcre, zlib，前者为了重写 rewrite，后者为了 gzip 压缩
依赖库安装：

```
yum -y install libmcrypt-devel mhash-devel libxslt-devel \
libjpeg libjpeg-devel libpng libpng-devel freetype freetype-devel libxml2 libxml2-devel \
zlib zlib-devel glibc glibc-devel glib2 glib2-devel bzip2 bzip2-devel \
ncurses ncurses-devel curl curl-devel e2fsprogs e2fsprogs-devel \
krb5 krb5-devel libidn libidn-devel openssl openssl-devel
```

安装 nginx

```
[root@localhost nginx-1.8.1]# ./configure --with-http_stub_status_module
--prefix=/usr/local/nginx --with-pcre=/app/lamp/pcre-8.10 --with-zlib=/app/lamp/zlib-1.2.5
```

其中，/app/lamp/pcre-8.10、/app/lamp/zlib-1.2.5 为 pcre 和 zlib 源码包路径。

2、mysql 安装过程（略）
3、Php 的安装过程

```
[root@localhost /app]#tar xjvf php-5.6.18.tar.bz2
[root@localhost /app]#t cd php-5.6.18
./configure --prefix=/usr/local/php --enable-fpm --with-fpm-user=www-data
--with-fpm-group=www-data --with-curl --with-mcrypt --enable-mbstring --enable-pdo
--with-pdo-mysql=mysqlnd --with-mysqli=mysqlnd --with-mysql=mysqlnd --with-openssl
--with-imap-ssl --with-gd --with-jpeg-dir=/usr/lib/ --with-png-dir=/usr/lib/ --enable-exif
```

--enable-zip

上面的写法中，mysql 是通过 yum 安装的，如果 mysql 是源码安装的话，需要修改 --with-mysql=/usr/loca/mysql，

而--with-mysqli 为--with-mysqli=/usr/local/mysql/bin/mysql_config

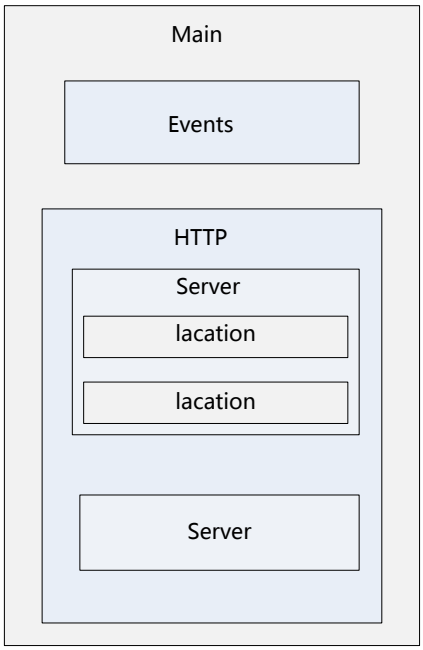
[root@localhost /app]#make

[root@localhost /app]#make install

二、nginx 的安装与配置文件结构

Nginx 配置文件主要分成四部分：

main（全局设置）
server（主机设置）
upstream（负载均衡服务器设置）
location（URL 匹配特定位置的设置）



二、Nginx 常用功能实战

1、Nginx 反向代理应用实例

多域名跳转：

```
server www.tb.com
location / {
proxy_pass http://192.168.66.90:8080/web/
}
location /admin {
proxy_pass http://192.168.66.90:8080/admin
}
server m.tb.com
location / {
proxy_pass http://192.168.66.90:8080/wap/
}
```

新旧域名过渡：

```
server{
server_name www.taob.com;
rewrite ^/(.*)$  http://www.tb.com/$1 permanent;
}
```

alias 与 root 功能

```
location  /i {
  alias  /var/www/html/images/;
}
```

对比：

```
location  /i {
  root  /var/www/html/images/;
}
```

2、location 应用实例

location 匹配规则优先级：

```
location  = / {
  [ config A ]
}
```

```
location ^~ /images/ {
    [ config C ]
}

location ~* \.(gif|jpg|jpeg|swf)$ {
    [ config D ]
}

location    / {
    [ config B ]
}
```

location 实现访问控制：

```
location / {
        deny 192.168.66.80;
        allow 192.168.66.0/24;
        allow 192.16.88.0/16;
        deny all;
}


location ~ ^/(WEB-INF)/ {
        deny all;
}
```

3、URL 重写应用实例

正则表达式匹配：

- ~ 表示区分大小写匹配
- ~* 表示不区分大小写匹配
- !~和!~*分别表示区分大小写不匹配及不区分大小写不匹配

文件及目录匹配：

- -f 和!-f 用来判断是否存在文件
- -d 和!-d 用来判断是否存在目录

- -e 和!-e 用来判断是否存在文件或目录
- -x 和!-x 用来判断文件是否可执行

在 Nginx 配置文件中，有很多内置变量，这些变量经常和 if 指令一起使用。常见的内置变量有如下几种：

- $args，此变量与请求行中的参数相等
- $document_root，此变量等同于当前请求的 root 指令指定的值
- $uri，此变量等同于当前 request 中的 URI。
- $document_uri，此变量与$uri 含义一样。
- $host，此变量与请求头部中“Host”行指定的值一致。
- $limit_rate，此变量用来设置限制连接的速率。
- $request_method，此变量等同于 request 的 method，通常是“GET”或“POST”。
- $remote_addr，此变量表示客户端 IP 地址。
- $remote_port，此变量表示客户端端口。
- $remote_user，此变量等同于用户名，由 ngx_http_auth_basic_module 认证。
- $request_filename，此变量表示当前请求的文件的路径名，由 root 或 alias 和 URI request 组合而成。
- $request_uri，此变量表示含有参数的完整的初始 URI。
- $query_string，此变量与$args 含义一致。
- $server_name，此变量表示请求到达的服务器名。
- $server_port，此变量表示请求到达的服务器的端口号。

（1）、301 重定向技术：

```
server{
        listen          80;
        server_name   www.taobao.com;
        #charset koi8-r;
        access_log   logs/taobao.access.log   main;
        rewrite ^/(.*)$   http://www.taob.com/$1   permanent;
```

```
        }
```

（2）if 指令应用实战

语法：if (condition) { … }
默认值：none
使用字段：server, location

```
server{
        listen          80;
        server_name www.tb.com;
        access_log   logs/host.access.log   main;
        location / {
        root     /var/www/html;
        index   index.html index.htm;
        }
        location ~*\.(gif|jpg|jpeg|png|bmp|swf|htm|html|css|js)$   {
                  root       /usr/local/nginx/www/img;
                if (!-f $request_filename)
                {
                    root       /var/www/html/img;
                 }
                 if (!-f $request_filename)
                {
                    root       /apps/images;
                 }
        }
        location ~*\.(jsp)$   {
                  root       /webdata/webapp/www/ROOT;
                if (!-f $request_filename)
                {
                    root       /usr/local/nginx/www/jsp;
                 }
                proxy_pass http://127.0.0.1:8888;
        }
}
```

（3）rewrite 指令实战

语法：rewrite regex flag

默认值：none
使用字段：server, location, if

```
location ~ ^/best/ {
        rewrite ^/best/(.*)$ /test/$1  break;
        proxy_pass  http://www.taob.com;
 }
```

三、Nginx 作为 Web 缓存服务器应用案例

从 http://labs.frickle.com/nginx_ngx_cache_purge/ 下载 ngx_cache_purge 插件，这里下载的文件是 ngx_cache_purge-2.1.tar.gz，然后进行解压即可，过程如下：

```
[root@ngxserver app]#wget http://labs.frickle.com/files/ngx_cache_purge-2.1.tar.gz
[root@ngxserver app]#tar zxvf -C /app/ngx_cache_purge-2.1.tar.gz
```

接着，开始编译安装 Nginx，过程如下：

```
[root@ngxserver app]# tar zxvf nginx-1.4.7.tar.gz
[root@ngxserver app]# cd nginx-1.4.7/
[root@ngxserver app]# ./configure --user=www --group=www  --prefix=/usr/local/nginx\
> --add-module=/app/ngx_cache_purge-2.1
> --with-http_stub_status_module --with-http_ssl_module
[root@ngxserver app]# make
[root@ngxserver app]# make install
```

完成安装后，可以通过“/usr/local/nginx/sbin/nginx -V”命令查看已安装的 Nginx 的版本和加载的模块信息。

```
proxy_cache_path  /backup/proxy_cache_dir  levels=1:2  keys_zone=cache_one:4096m inactive=1d max_size=3g;
```

- proxy_cache_path：用于设置缓存的目录,后面跟缓存路径。最好将缓存目录放在一个独立的硬盘上。
- levels=1:2：levels 用来设置目录深度，这里是两层目录深度，第一层是一个字符，第二层#是两个字符。
- keys_zone：用来设置 web 缓存区名称，这里是 cache_one，后面的”4096m“表示内存缓存空间大小为 4G。
- inactive：表示自动清除缓存文件的时间，这里的”1d“表示 1 天没有被访问的内容自动清除，还可以使用分钟或小时计时，例如”5m“表示 5 分钟后自动清除，”5h“表示 5 小时后自动清除。
- max_size：表示硬盘缓存空间可使用的最大值，默认情况下经常访问的文件将被放到内存中进行缓存，而在内存缓存空间不足时，nginx 会将不经常访问的数据从内存写到磁

盘

proxy_temp_path /backup/proxy_temp_dir;

- proxy_temp_path 用于指定临时缓存文件的存储路径，这里需要注意的是，proxy_temp_path 和 proxy_cache_path 指定的路径必须在同一磁盘分区

```
server{
        listen          80;
        server_name   www.tb.com www.taob.com;
        charset UTF8;
        access_log   logs/cms.access.log   main;


        location /{
            proxy_cache cache_one;
```

- 反向代理缓存设置指令，语法为"proxy_cache zone|off",默认值为 off，需要将 proxy_cache 指令放到 location 字段，这样匹配此 location 的 url 才能被缓存

```
          proxy_cache_valid   200 304 12h; #对不同的 HTTP 状态码设置不同的缓存时间
            proxy_cache_key  $host$uri$is_args$args;   #这个指令是设置以什么样的参数得到缓存的文件名，默认为 "$scheme$proxy_host$request_uri"，表示以协议、主机名、请求 uri(包含参数)作 md5 得出缓存的文件名。这里是以域名、URI、参数组合成 web 缓存的 Key 值，Nginx 根据 Key 值哈希，存储缓存内容到二级缓存目录内
            proxy_set_header Host   $host;
            proxy_set_header X-Forwarded-For   $remote_addr;
            proxy_pass http://127.0.0.1:8080;
            expires          1d;
        }
```

下面这段用于配置手动清除缓存策略，清除的方法为：如果一个 URL 为 http://www.tb.com/2014/0413/3.html ，那么通过访问 http://www.tb.com/purge/2014/0413/3.html 即可清除该 URL 的缓存

```
        location ~ /purge(/.*)
        {
                allow               127.0.0.1;          #表示只允许指定的IP或IP段才可以清除 URL 缓存。
                allow               192.168.88.0/24;
                deny        all;
                proxy_cache_purge     cache_one     $host$1$is_args$args;
        }
```

```
        location ~ .*\.(jsp|php|jspx)?$    #设置不做缓存的内容，这里设置扩展名以.jsp、.php、.jspx 结尾的动态应用程序不缓存
        {
                proxy_set_header Host   $host;
                proxy_set_header X-Forwarded-For   $remote_addr;
                proxy_pass http://127.0.0.1:8080;
        }

    access_log   off;
  }
}
```

四：Nginx 作为负载均衡服务器应用案例

负载均衡算法：

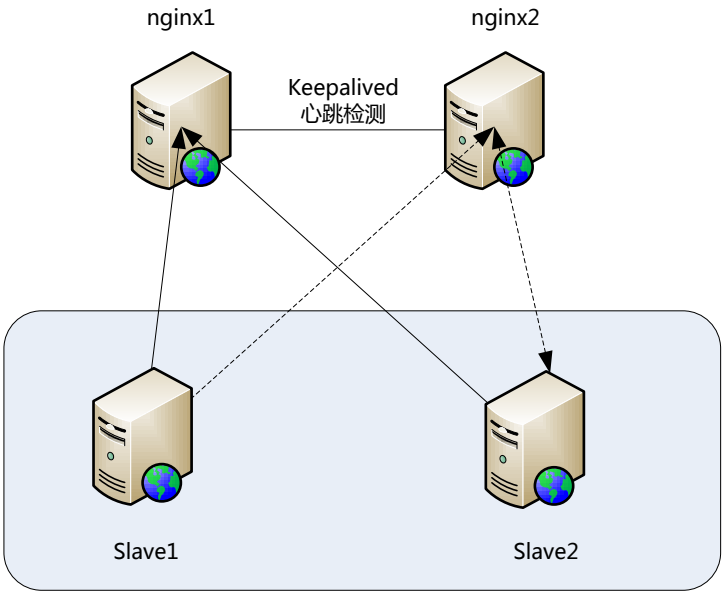
- 轮询
- Weight
- ip_hash

```
http
{
  upstream   myserver{
    server    192.168.12.181:80  weight=3 max_fails=3 fail_timeout=20s;
    server    192.168.12.182:80  weight=1 max_fails=3 fail_timeout=20s;
    server    192.168.12.183:80  weight=4 max_fails=3 fail_timeout=20s;
  }

  server
  {
    listen          80;
    server_name   www.domain.com 192.168.12.189;
    index index.htm index.html;
    root   /ixdba/web/wwwroot;

location / {
    proxy_pass http://myserver;
    proxy_next_upstream http_500 http_502 http_503 error timeout invalid_header;
    include      /opt/nginx/conf/proxy.conf;
```

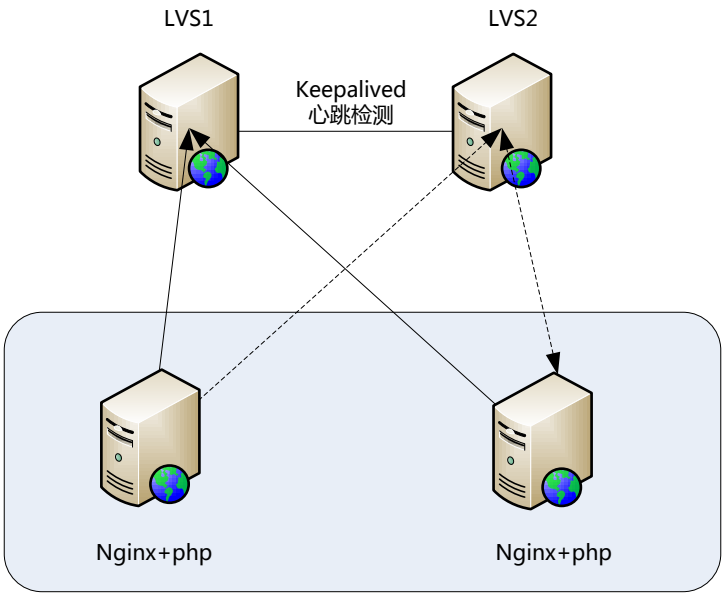
```
    }
  }
}
```

五、常见的 nginx 应用架构以及几个故障排除案例

1、nginx+tomcat 常见架构



2、Nginx+php 常见架构



3、如何排除 nginx 故障

实例演示

下期公开课预告

主题：Mysql 服务器企业应用架构经验分享

内容：

1、Mysql 企业常见应用架构经验分享

2、线上 Mysql 调优经验技巧

3、典型的 Mysql 企业应用架构案例

参与方式：

加入 Linux 运维专家（134896298），然后关注群公告，每周会定期进行公开课技术分享，具体开课时间会进行提前通知，欢迎大家届时参加。

参与方式：打开爱维 Linux 腾讯课堂：http://ke.qq.com/course/117291，点击报名，即可免费参加，如果你没来及参加线上直播，那么可以通过访问爱维 Linux 交流论坛：http://i.iivey.com/ 观看公开课的录播视频。

广而告之

开班介绍

爱维 Linux，专注 Linux 运维实战教育，我们开设了两个班级：

- 高薪运维入门提高班　详情：http://www.iivey.com/666-2
- 高薪运维实战提升班　详情：http://www.iivey.com/archives/66

授课模式

课程汇聚了以**南非蚂蚁**领衔的行业顶尖技术专家 10 年一线工作经验和培训心得，课程由浅入深，循序渐进，能够帮助学员们系统学习 Linux 一线经验，并迅速掌握 Linux 的各种应用技能。

我们的授课方法：

理论结合实际+实战技巧+经验分享+实时互动+专业学习教材

开课时间

入门提高班将在 3 月 12 开课，而实战提高班将在 5 月份开课，5 个月的授课时间，现在入门提高班接受报名，有意向的朋友可通过如下方式联系我们：

QQ：397824870（蚂蚁老师） 3335603751（章老师） 18966929688（王老师）

微信：ixdba8