Nginx进阶

By IT崖柏图
Mail | jyk0213@163.com

第1章 安装介绍

① 准备工作

> *yum -y install pcre pcre-devel*

为什么要安装nginx?

我们的nginx服务器可能要用到rewrite重写,重写又涉及到了正则匹配,说到正则,你就能够想到正则的库,PCRE库.

② 下载解压编译nginx

1: 下载nginx

官网: http://nginx.org/

> *wget http://nginx.org/download/nginx-1.10.3.tar.gz*

2: 解压nginx-1.10.3.tar.gz

> *tar zxvf nginx-1.10.3.tar.gz*

3: 配置编译参数编译

> *./configure --prefix=/usr/local/nginx*

如果提示你缺少 *pcre* 库,你要看看你的nginx版本, *1.6.x* 以上版本,要求指定 *pcre* 的源码目录.

下载pcre源码库:

> *wget ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/pcre-8.39.tar.gz*

重新配置 *nginx* 编译参数:

> *./configure --prefix=/usr/local/nginx --with-pcre=/usr/local/src/pcre-8.39*

如果还提示你缺少其他库,再就是重复上面的步骤了.

4: 编译

> *make*

5: 安装

> *make install*

③ 启动nginx

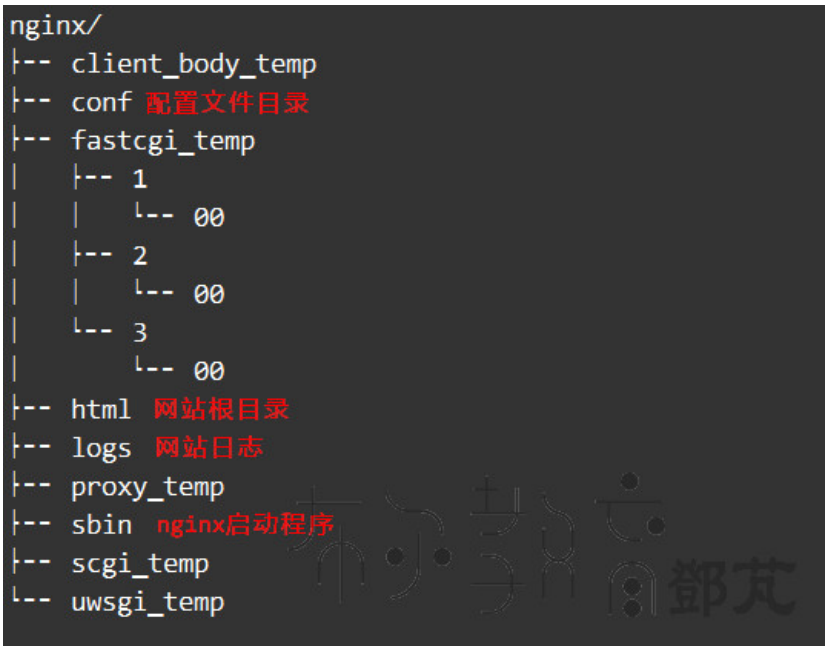
> *./sbin/nginx*

问题解决:

1: 如果从局域网中连接本机服务器失败,请关闭防火墙试试.

> *service iptables stop*

2: nginx目录分布



第2章 控制命令

我们之前对 *nginx* 的开启和停止都是使用 *kill* 和 *pkill* 命令直接杀死进程.这种方法很暴力,暴力的后果是,如果一批用户正在请求你的服务器,突然服务器不响应了,即使你很快重新开启了服务器,但是用户体验很不好,很容易遗失潜在的用户群.

就比如,你开出租车拉客,正拉着客呢,你爸爸突然打电话说你老婆生娃了,你这啪地一下把客人撵下车,自己往医院赶.这用户体验能好吗?

怎么办?

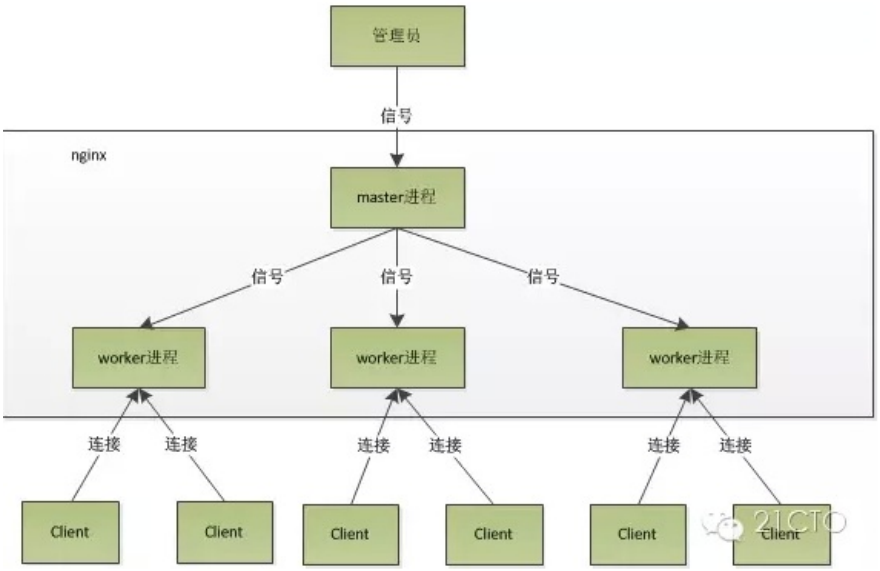
那能不能让服务器先把要服务的用户,服务完成后,再去执行关机或者重启命令呢?

这是可以的,这节课咱们要学习的就是这样一个手艺 *软重启*

nginx常用的命令参数:

命令	作用
nginx -t	测试配置是否正确
nginx -s reload	重新加载配置到内存,重启nginx
nginx -s reopen	重新打开日志
nginx -s stop	立即停止
nginx -s quit	优雅停止(等待该响应的响应完成后才停止)

nginx原理示意图



nginx -s reopen 重新打开日志的解释:

先重命名access.log日志文件,并重新创建新的access.log文件,再次访问查看变化.

3.1 全局配置

```
# 配置worker的数目
# worker的数量 = cpu的数量*cpu的核数
# 没必要设置太大,因为这些worker会去抢夺cpu
worker_processes  1;

# 配置1个worker进程最多能接受多少个连接
events {
    worker_connections  1024;
}

http {
    # 配置虚拟主机
    server {
        listen       80;    # 监听端口号
        server_name  localhost;    # 虚拟主机的域名

        location / {    # 虚拟主机的站点目录,/表示就是nginx服务器下的html目录
            root   html;
            index  index.php index.html index.htm;    # 默认文件,优先级从前到后
        }
    }

    server {
        ......
    }
}
```

3.2 nginx配置虚拟主机

示例1: 基于域名的虚拟主机

```
server {
    listen       80;    # 监听80端口
    server_name  itbool.com;    # 监听域名,如果有多个域名,用空格隔开

    location / {
        root   html;    # 网站根目录定位
        index  index.php index.html index.htm;  # 默认索引页
```

```
    }
}
```

示例2: 基于端口号的虚拟主机配置

```
server {
    listen       8080;    # 监听8080端口
    server_name  blog.itbool.com;    # 监听域名,如果有多个域名,用空格隔开

    location / {
        root   html;    # 网站根目录定位
        index  index.php index.html index.htm;  # 默认索引页
    }
}
```

第4章 日志管理

我们通过查看nginx的配置文件 `nginx.conf` ,在server配置部分可以看到如下配置:

```
#access_log logs/host.access.log main;
```

这一行很明显,被注释了.那么这一行配置是用来做什么的呢?

配置我们该台Server的访问日志,日志文件叫做 `host.access.log` ,存放在nginx目录下的 `logs/` 文件夹下,日志的格式是 *main* 格式.

默认main日志格式

那么main格式是什么样的格式呢?

```
log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                  '$status $body_bytes_sent "$http_referer" '
                  '"$http_user_agent" "$http_x_forwarded_for"';
```

```
[root@centos68 logs]# tail -10 access.log
192.168.30.105 - - [18/Feb/2017:11:43:26 +0800] "GET /index.php HTTP/1.1" 502 537 "-" "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleW
ebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36"
```

默认的main格式就只有这几行,具体每个参数代表什么,请参考下方:

$remote_addr 与$http_x_forwarded_for用以记录客户端的ip地址;
$remote_user 用来记录客户端用户名称;
$time_local 用来记录访问时间与时区;
$request 用来记录请求的url与http协议;
$status 用来记录请求状态,成功是200;
$body_bytes_sent 记录发送给客户端文件主体内容大小;
$http_referer 用来记录从那个页面链接访问过来的;
$http_user_agent 记录客户端浏览器的相关信息;

我们既然能够知道每个变量所代表的含义,那么我们自己就能根据自己的需求去自定义自己需要的日志格式.

自定义日志格式

```
log_format  mystyle  '$remote_addr - $remote_user [$time_local] '
                     '$status "$http_referer" '
                     '"$http_user_agent" "$http_x_forwarded_for"';
```

既然定义了自己的日志格式,那么你就要在Server配置段中启用:

```
server {
    listen       80;    # 监听8080端口
    server_name  zixue.it;    # 监听域名,如果有多个域名,用空格隔开

    access_log  logs/zixue.it.access.log  mystyle;
```

```
    location / {
        root   html;    # 网站根目录定位
        index  index.php index.html index.htm;  # 默认索引页
    }
}
```

重启nginx,再次访问zixue.it,看看 `./nginx/logs/` 目录下是否生成了一个 `zixue.it.access.log` ,风格又是否是我们定制的样式呢?

第5章 nginx与php的整合

在整合nginx和php之前,我们要弄清楚这两者的关系:

我们知道apache与php的关系就是,php作为apache的一个模块来启动.然而nginx跟php却不是,他们两者是互相独立的.独立如何工作呢?

nginx每次接到php的脚本请求时,则将请求转发给php进程,php进程处理完再返回给nginx,这种通信方式,业内称之为fastcgi模式.

注意:

我们在编译php的时候要有如下功能:

连接MySQL,GD,TTF,以fpm(fastcgi)方式运行

> *./configure --prefix=/usr/local/fastphp*
> *--with-mysql=mysqlnd*
> *--enable-mysqlnd*
> *--with-gd*
> *--enable-gd-native-ttf*
> *--enable-gd-jis-conv*
> *--enable-fpm*

编译完毕后去php的安装路径下的 `./etc/php-fpm.conf.default` 复制一份为 `./etc/php-fpm.conf`

nginx + php 配置很简单

核心: 让nginx把脚本请求信息转发给监听ip:9000端口号的php进程,让php处理指定目录下的脚本.

示例:

```
location ~ \.php$ {
    root           html;
    fastcgi_pass   127.0.0.1:9000;
    fastcgi_index  index.php;
    fastcgi_param  SCRIPT_FILENAME  $DOCUMENT_ROOT$fastcgi_script_name;
    include        fastcgi_params;
}
```

参数释义:

1: 遇到请求 `html/` 根目录下的 `.php` 结尾的脚本时,将请求转发到监听 `127.0.0.1:9000` 端口的php进程.

2: 并告诉php进程,当前要解释的脚本路径是 `$DOCUMENT_ROOT$fastcgi_script_name`

注意: 所以你一定要指定正确要解释的php脚本的正确路径,否则php进程会找不到哦.

第6章 rewrite规则

网站根目录 `./html/` 下，新建脚本 `user.php`

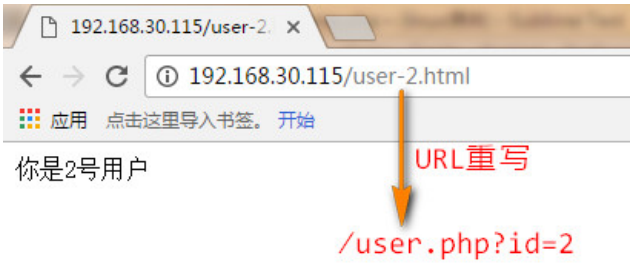
```
<?php

$info = [
    '1' => '你是1号用户',
    '2' => '你是2号用户',
    '3' => '你是3号用户',
```

```
];

$uid = $_GET['id'];

echo $info[$uid];
```

编辑 `./conf/nginx.conf` 配置文件

```
location / {
    root   html;
    index  index.html index.htm;
    rewrite /user-(\w+)\.html /user.php?id=$1;
}
```

访问 `user-2.html`



这种通过url重写的方法,也被SEO界称之伪静态.

第6.1章 pathinfo支持

```
# 典型配置
location ~ \.php$ {
    root           html;
    fastcgi_pass   127.0.0.1:9000;
    fastcgi_index  index.php;
    fastcgi_param  SCRIPT_FILENAME  $DOCUMENT_ROOT$fastcgi_script_name;
    include        fastcgi_params;
}

# 修改第1,6行,支持pathinfo

location ~ \.php(.*)$ { # 正则匹配.php后的pathinfo部分
    root html;
    fastcgi_pass   127.0.0.1:9000;
    fastcgi_index  index.php;
    fastcgi_param  SCRIPT_FILENAME  $DOCUMENT_ROOT$fastcgi_script_name;
    fastcgi_param  PATH_INFO $1; # 把pathinfo部分赋给PATH_INFO变量
    include        fastcgi_params;
}
```

第6.2章 try_files

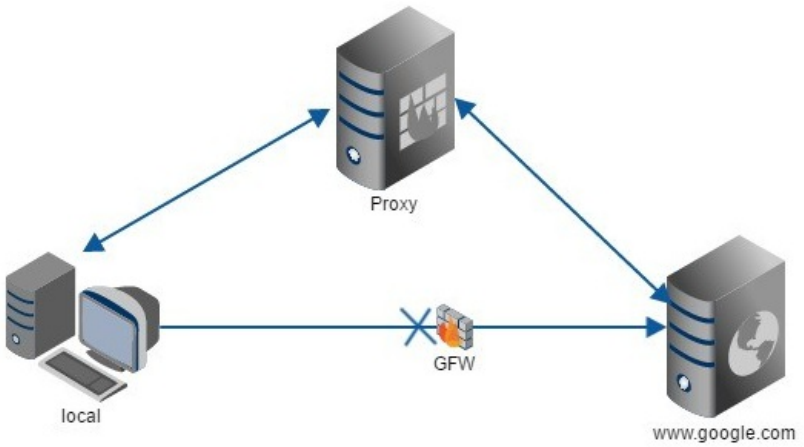
```
try 试试

try_files $uri/ $uri /index.php?$query_string;

以 xx.com/goods/1 为例
是指,
先试试 有没有 /goods/1/ 这个目录
再试试 有没有 /goods/1 这个文件
最后试试 /index.php?goods/1 这个 URL
```

第7章 反向代理

代理(Proxy)?

什么是代理?娱乐圈的"经纪人"就是代理.所以说对代理的配置问题,我们一定要学好,配置不好约束不好代理就容易出问题.去年炒得比较热的马王事件就是配置不当,导致代理不好好工作出现的一些列问题.

何为正向代理?



何为反向代理?
nginx作为web服务器一个重要的功能就是反向代理.

为啥要用反向代理?
1: 保护网站安全

任何来自Internet的请求都必须先经过代理服务器

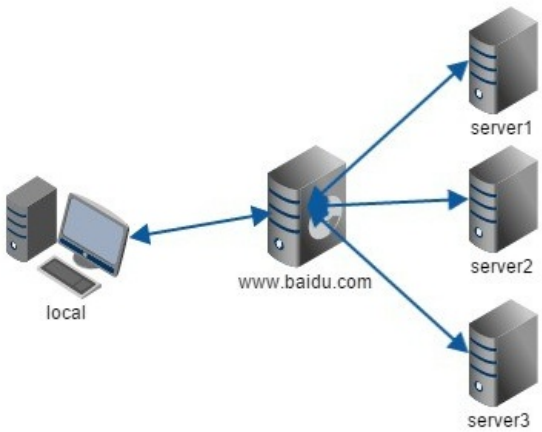
2: 通过配置缓存功能加速Web请求

可以缓存真实Web服务器上的某些静态资源,减轻真实Web服务器的负载压力

3: 实现负载均衡

充当负载均衡服务器均衡地分发请求,平衡集群中各个服务器的负载压力

反向代理原理:



两者区别
正向代理代理的对象是客户端，反向代理代理的对象是服务端

动静分离之配置proxy_pass

我们这节课给大家讲一个动静分离的案例,其实质运用的就是反向代理,我们一台服务器代理代理布尔服务器上的图片资源.

我们说了你想使用代理,必然就要配置代理.配置反向代理,必须要用到 *proxy_pass* 命令来配置.

打开nginx的配置文件 `./conf/nginx.conf` 在你的Server虚拟主机段中添加如下配置:

```
location ~ \.(jpg|gif|png)$ {
    proxy_pass IP:port;
}
```

格式:

```
location ~ \.(jpg|gif|png)$ {
    proxy_pass http://www.itbool.com;
}             协议://ip地址:端口号;
              不给端口号,默认是80
```

思考:
1: 反向代理导致了后端服务器接到的客户端 IP,为前端服务器的 IP,而不是客户真正的 IP,怎么办?
答: 代理服务器通过设置头信息字段,把用户 IP 传到后台服务器去.

```
location ~ \.(jpg|jpeg|png|gif)$ {
 proxy_set_header X-Forwarded-For $remote_addr;
 proxy_pass IP:port;
}
```

第8章 负载均衡

By The Way

负载均衡:

一听这词很多小伙伴吓坏了,前人就喜欢搞一些看起来很高大上的词,好让后生望而敬畏.那我们一起来捋一捋. *负载* 就是 *负担* . *均衡* 就是 *平均分* .这样一说就是 *负担平均分* .

南方洗脚城比较多,大家都去过,里面有专门给人洗脚按摩,修脚啥的.假如你在这里面打工,来了一个客户,老板让小明上,再来一个客户让你上.但是吧,你这人干活干净利落,不拖泥带水,客户一致五❤好评.这时候又来了第三位客户,老板又让你上.唉,这就是负载均衡.小花擅长洗脚不擅长修脚,凡是只要求洗脚的客户,全都给小花去服务.这也是一种负载均衡. *其宗旨就是不把负担压在一个人身上的同时,提高服务质量和效率.*

服务器也要减减压

我们都知道服务器是第三产业,服务行业,客户来了,你不能不服务.客户少还行,客户多了一台服务器就顶不住了,怎么办?我们找来多台服务器,让这些服务器去均分客户然后服务响应.或者让某些服务器只干一类时,来提高效率.

那么如何让我们的代理服务器知道哪些服务器是可以去干活的,而且擅长干什么活呢?这个时候就需要我们去配置了.

编辑 `./conf/nginx.conf` 文件(注意 upstream {} 配置信息必须在 Server {} 配置的外部,不是 Server {} 的里面)

```
# 集群服务器组

upstream itbool { # 服务器集群的组名
    server 192.168.3.110:80 weight=1 max_fails=2 fail_timeout=30s;
    server 192.168.3.111:80 weight=1 max_fails=2 fail_timeout=30s;
    #server 服务器的ip:端口号 权重 最大失败次数 最大连接时间
}

# 指定代理请求的集群组名

Server {
    location ~ \.(jpg|png|jpeg|gif)$ {
```

```
        proxy_pass   http://itbool;
    }
}
```

说明:

1: weight的值越大,表明这台服务器办事效率高,老板喜欢,有事了,找他的概率大.
2: max_fails要说明的是,你找这台服务器办事,叫他2次如果还不理你,你不要对他报以希望了.
3: fail_timeout要说的是,给这台服务器一件小事让它办,30s还没办完,算了,不靠谱,不要等了,找其他人吧.

Nginx 中的几种负载均衡方式

> 1、轮询（默认）
> 每个请求按时间顺序逐一分配到不同的后端服务器，如果后端服务器down掉，能自动剔除。
> 2、weight
> 指定轮询几率，weight和访问比率成正比，用于后端服务器性能不均的情况。
> 2、ip_hash
> 每个请求按访问ip的hash结果分配，这样每个访客固定访问一个后端服务器，可以解决session的问题。
> 3、fair（第三方）
> 按后端服务器的响应时间来分配请求，响应时间短的优先分配。
> 4、url_hash（第三方）
> 按访问url的hash结果来分配请求，使每个url定向到同一个后端服务器，后端服务器为缓存时比较有效。