NGINX服务企业应用

惨绿少年 Linux运维, Web应用, 运维基本功 0评论 来源：本站原创 48℃ 字体：小 中 大

1.1 常用来提供静态服务的软件

Apache :这是中小型Web服务的主流，Web服务器中的老大哥，

Nginx :大型网站Web服务的主流，曾经Web服务器中的初生牛犊，现已长大。

Nginx 的分支 Tengine (http://tengine.taobao.org/)目前也在飞速发展•

Lighttpd :这是一个不温不火的优秀 Web软件，社区不活跃，静态解析效率很高.在 Nginx 流行前，它是大并发静态业务的首选，国内百度贴吧、豆瓣等众多网站都有Lighttpd奋斗的身彩”

1.2 常用来提供动态服务的软件

> * PHP (FastCGI):大中小型网站都会使用，动态网页语言PHP程序的解析容器。它可配合Apache解析动态程序，不过，这里的PHP不是FastCGI守护进程横式，而是mod_php5.so (module）也可配合Nginx解析动态程序，此时的PHP常用FastCGI守护进程模式提供服务。
> * Tomcat :中小企业动态Web服务主流，互联网Java容器主流（如jsp、do ）
> * Resin :大型动态Web服务主流，互联网Java容器主流（如jsp、do ）
> * IIS (Internet information services):微软 windows 下的 Web 服务软件（如 asp、aspx ）

第2章 nginx 软件

2.1 软件介绍

如果你听说或使用过 Apache软件，那么很快就会熟悉 Nginx软件，与 Apache软件类似， Nginx (“engme x”)是一个开源的，支持高性能、高并发的 WWW服务器和代理服务软件。它是由俄罗斯人Igor Sysoev开发的，最初被应用在勘罗斯的大型网站www.rambler.ru 上,后来作者将源代码以类BSD许可证的形式开源出来供全球使用。

Nginx因具有高并发（特别是静态资源）占用系统资源少等特性，且功能丰富而逐渐流行起来。

在功能应用发面，Nginx不但是一个优秀的Web服务软件，还具有反向代理负载均衡功能和缓存服务功能。在反向代理负载均衡功能方面，它类似于大名鼎鼎的LVS负载均衡及Haproxy等专业代理软件，但是Nginx部署起来更为简单、方便；在缓存服务功能方面，它又类似于Squid等专业的缓存服务软件。

Nginx 可以运行在 UNIX、Linux、BSD、Mac 0S X、Solaris,以及Microsoft Windows 等操作系统中。随着Nginx在国内很多大型网站中的稳定高效运行，近两年它也逐渐被越来越多的中小型网站所使用。当前流行的Nginx Web组合被称为LNMP或LEMP(即Linux Nginx MySQL PHP),其中 LNMP 里的 N 取自Nginx (“engine x”)

Nginx 的官方介绍见 http://nginx.org/en

2.2 NGINX 软件特性

2.2.1 HTTP服务器的特色及优点:

- 支持高并发：能支持几万并发连接（特别是静态小文件业务环境）
- 资源消耗少：在3万并发连接下，开启10个Nginx线程消耗的内存不到200MB
- 可以做HTTP反向代理及加速缓存、即负载均衡功能，内置对RS节点服务器健康检查功能，这相当于专业的Haproxy软件或LVS的功能。
- 具备Squid等专业缓存软件等的缓存功能。
- 支持异步网络I/O事件横型epoll(Linux2.6+)

2.2.2 nginx功能特性

- web网站服务
- 反向代理负载均衡(nginx /lvs /haproxy)
- nginx缓存服务（memcache /redis /mongodb）

2.3 nginx软件的企业功能应用

业务类型	应用方案
静态业务	若是高并发场景，尽量采用Nginx或Lighttpd，二者首选Nginx
动态业务	理论上采用Nginx和Apache均可，建议选择Nginx，为了避免相同业务的服务软件多样化，增加额外维护成本。动态业务可以由Nginx兼做前端代理，再根据页面元素的类型或目录，转发到后端相应的服务器处理进程。 —首选tomcat
既有静态业务又有动态业务	采用Nginx 利用nginx软件是无法处理动态业务请求，要让nginx结合php软件处理动态业务请求，在加上mysql 即**LNMP**架构

2.4 nginx软件的动态访问瓶颈

2.4.1 与其他软件的对比

先来看看**Apache软件的特点**，如下

- ✧ Apache2.2版本非常稳定强大，据官方说，Apache2.4版本性能更强。
- ✧ Prefork模式取消了进程创建开销，性能很高。
- ✧ 处理动态业务数据时，因关联到后端的引擎和数据库，瓶颈不在Apache上。
- ✧ 高并发时消耗系统资源相对多一些。
- ✧ 基于传统的select模型，高并发能力有限。
- ✧ 支持扩展库，可通过DSO、apxs方法编译安装额外的插件功能，不需要重新编译Apache
- ✧ 功能多，更稳定，更安全，插件也多。
- ✧ 市场份额在逐年递减

再来看看**Nginx软件**的特点，如下：

- ■ 基于异步网结I/O模 型（epollk kqueue）
- ■ 具备支持高性能，高并发的特性，并发连接可达数万。
- ■ 对小文件（小于1 MB的静态文件）高并发支持很好，性能很高
- ■ 不支持类似 Apache的DSO模式、扩展库必须编译进主程序（缺点）
- ■ 进程占用系统资源比较低。
- ■ 支持Web、反向Proxy、Cache三大重点功能，并且都很优秀。
- ■ 市场份额在逐年快速增加。

最后是**Lighttpd的特点**，如下：

- ✧ 基于异步网络 I/ O模型，性能、并发都与 Nginx相近。
- ✧ 扩展库是 SO模式，比Nginx灵活
- ✧ 目前国内的使用率比较低，安全性没有 Apache和Nginx好。
- ✧ 通过插件(mod_secdownload)可实现文件 URL地址加密（优点）
- ✧ 社区不活跃，市场份额较低，

2.4.2 最主要的区别（select & epoll）

NGINX 使用的是epoll 和Kqueue 异步网络I/O模型，而apache使用的是传统的select模型

比喻：

> 第一个比喻：

> 假设你在大学读书，住的宿舍楼有很多房间，你的朋友要来找你。select版宿管大妈就会带着你的朋友到各房间挨个去找，直到找到你为止。而epoll版宿管大妈会先记下每位入住同学的房间号，你的朋友来找你时，只需告诉你的朋友你住在哪个房间即可，不用亲自带着你的朋友满宿舍楼找人了。如果同时来了100个人，都要找自己住这栋楼的同学，select版和epoll版宿管大妈,谁的效率更高，就很明显了。

第二个比喻:

> select的调用复杂度是线性的，即O(n)。举个例子，一个保姆照看照看一群孩子，如果把孩子是否需要尿尿比作网络I/O事件，select的作用就好比这个保姆挨个询问每个孩子”你要尿尿吗？”如果孩子回答是，保姆则把孩子领出来放到另外一个地方。当所有孩子询问完之后，保姆领着这些要尿尿的孩子去上厕所（处理网络I/O事件).在epoll机制下，保姆不再需要挨个询问每个孩子是否需要尿尿。取而代之的是，如果孩子需要尿尿，他就自己主动站到事先约定好的地方，而保姆的职责就是查看事先约定好的地方是否有孩子。如果有小孩，则领着孩子去上厕所（网络事件处理)。因此，epoll的这种机制，能够高效地处理成千上万的并发连接，并且性能不会随着连接数增加而下降太多。

2.4.3 apache select和nginx epoll技术对比图

指标	select	epoll
性能	随着连接数的增加性能急剧下降。处理成千上万的并发连接数，性能很差	随着连接数的增加，性能基本上没有下降。处理成千上万连接时性能很好
连接数	连接数有限制，处理的最大连接数不超过1024，如果要处理的连接数超过1024个，则需要修改FD_SETSIZE宏，并重新编译	连接数无限制
内在处理机制	线性轮询	回调callback
开发复杂性	低	中

第3章 nginx的安装与使用

3.1 nginx软件的编译安装步骤

3.1.1 检查软件安装的系统环境

```
[root@web01 ~]# cat /etc/redhat-release
CentOS release 6.9 (Final)
[root@web01 ~]# uname -r
2.6.32-696.el6.x86_64
```

3.1.2 安装nginx的依赖包（pcre-devel openssl-devel）

```
yum install -y pcre-devel openssl-devel
```

> pcre：兼容perl语言正则表达式，perl compatible regular expressions
>
> rewirte模块 参数信息（perl方式定义正则表达式）
>
> openssl：ssh—openssh/openssl—https
>
> **总结：所有安装依赖软件，后面都要加上-devel**

3.1.3 下载nginx软件

```
wget http://nginx.org/download/nginx-1.10.2.tar.gz
```

说明：软件很小，用心查看一下

解压软件

```
tar xf nginx-1.10.2.tar.gz
```

3.1.4 创建管理用户 www

```
useradd -M -s /sbin/nologin www
```

3.1.5 nginx软件编译安装过程

3.1.5.1 注意

> 软件编译安装步骤
>
> a>软件解压配置（将软件程序安装到哪个目录中 开启nginx软件的哪些功能）
>
> b>软件编译过程
>
> c>软件编译安装过程

注意顺序，顺序不对软件安装会出错

3.1.5.2 编译安装软件

1、配置软件，在软件的解压目录中

```
[root@web01 nginx-1.10.2]# ./configure --prefix=/application/nginx-1.10.2 --user=www --group=www
```

编译参数说明:

> -prefix 表示指定软件安装到哪个目录中，指定目录不存在会自动创建
>
> -user/-group nginx工作进程由哪个用户运行管理
>
> -with-http_stub_status_module 启动nginx状态模块功能（用户访问nginx的网络信息）
>
> -with-http_ssl_module 启动https功能模块

通过软件编译过程中的返回值是否正确，确认配置是否正确

```
[root@web01 nginx-1.10.2]# echo $?
0
```

2、编译软件

```
[root@web01 nginx-1.10.2]# make
```

3、编译安装

```
[root@web01 nginx-1.10.2]# make install
```

3.1.6 创建软连接

```
[root@web01 application]# ln -s /application/nginx-1.10.2/ /application/nginx
```

3.1.7 精简化nginx.conf 主配置文件内容

```
[root@web01 conf]# egrep -v "#|^$" nginx.conf.default >nginx.conf
```

3.1.8 启动程序

```
[root@web01 application]# /application/nginx/sbin/nginx
[root@web01 application]#
```

检查是否启动

```
[root@web01 application]# ps -ef |grep nginx
root      26548      1  0 20:13 ?        00:00:00 nginx: master process /application/nginx/sbin/
www       26549  26548  0 20:13 ?        00:00:00 nginx: worker process
root      26551  23431  3 20:13 pts/0    00:00:00 grep --color=auto nginx
```

检查端口信息

```
[root@web01 application]# netstat -lntup |grep 80
tcp        0      0 0.0.0.0:80                  0.0.0.0:*                   LISTEN      26548/ng
```

服务部署完成

至此软件安装完毕!

nginx命令简化方法

```
echo 'export PATH=/application/nginx/sbin:$PATH'>>/etc/profile
source /etc/profile
which nginx
```

3.1 nginx 目录结构

```
[root@web01 nginx]# ll
total 36
drwxr-xr-x 2 root root 4096 Oct 21 19:34 conf    #配置文件保存目录
drwxr-xr-x 2 root root 4096 Oct 21 19:34 html    #站点目录
drwxr-xr-x 2 root root 4096 Oct 21 20:26 logs    #nginx 服务相关日志文件保存目录（错误日志访问日
drwxr-xr-x 2 root root 4096 Oct 21 19:34 sbin    # 服务命令目录（只有一个nginx文件）
```

3.2 nginx.conf 配置文件说明

这样的配置文件是通过精简化配置文件得到!

```
[root@web01 conf]# cat nginx.conf
worker_processes  1;                        ← worker 进程数量
events {                                    ←事件区块
    worker_connections  1024;               ←每个worker进程可以处理的连接数
}                                            ←事件区块结束
http {                                      ← HTTP 区块
    include       mime.types;               ←支持的媒体文件
    default_type  application/octet-stream;←默认的媒体类型
    sendfile        on;                     ←高效传输模式
    keepalive_timeout  65;                  ←超时时间
    server {                                 ← server 区块
        listen       80;                    ←端口
        server_name  localhost;             ←域名
        location / {                        ←第一个location区块
            root   html;                     ←站点目录
            index  index.html index.htm;     ←首页文件
        }                                    ←第一个location区块结束
        error_page   500 502 503 504  /50x.html;  ← 错误信息配置
        location = /50x.html {                  文件位置
            root   html;                       在哪找：路径
        }
    }                                        ← server 区块结束
}                                           ← HTTP 区块结束
```

3.2.1 站点目录与首页文件概

3.2.2 配置文件详解

3.3 【常见错误】nginx软件的编译安装常见错误说明

3.3.1 nginx软件安装过程中遇到的问题

软件依赖包未正确安装问题—PCRE依赖包没有安装

```
./configure: error: the HTTP rewrite module requires the PCRE library.
You can either disable the module by using --without-http_rewrite_module
option, or install the PCRE library into the system, or build the PCRE library
statically from the source with nginx by using --with-pcre= option.
```

解决方法：yum install pcre pcre-devel -y

软件依赖包未正确安装问题—OPENSSL依赖包没有安装

```
./configure: error: SSL modules require the OpenSSL library.
You can either do not enable the modules, or install the OpenSSL library
into the system, or build the OpenSSL library statically from the source
with nginx by using --with-openssl= option.
```

解决方法：yum install openssl openssl-devel -y

3.3.2 nginx软件启动过程中遇到的问题

nginx软件重复启动产生的错误信息

```
[root@web01 nginx-1.10.2]# /application/nginx/sbin/nginx
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] still could not bind()
```

解决方法：

nginx软件已经启动无需反复启动，如果需要重新启动需要停止nginx进程或者用reload方式进行重启

3.3.3 启动 Nginx 时如下报错“nginx:[emerg]getpwnam(“nginx”〉failed”

解答这是因为没有对应的Nginx服务用户，执行useradd nginx-s/sbin/no | ogin-M创建 Nginx

用户即可。为了让读者理解问题，重现上述错误过程，命令如下：

```
[root@web01 tools]# pkill nginx
[root@web01 tools]# userdel nginx
```

```
[root@web01 tools]# /application/nginx/sbin/nginx
nginx: [emerg] getpwnam(Mnginx") failed
[root@web01 tools]# useradd nginx -s /sbin/nologin -M
[root@web01 tools]# /application/nginx/sbin/nginx
```

3.3.4 编译安装pcre编译软件时，gcc不全导致报错（本文使用yum安装不存在此问题)。

报错信息如下:

```
[root@gjlin2 pcre-8.30]# make && make install
make all-am
make[l] : Entering directory 7home/gjlin/tools/pcre-8.30'
CXX pcrecpp.lo
libtool : compile : unrecognized option '-DHAVE_CONFIG_H'
libtool : compile : Try 'libtool --help* for more information.
make[l】: *** [pcrecpp.lo】错误 1
make[l] : Leaving directory Vhome/gjlin/tools/pcre-8.30'
make : *** [all]错误
```

解答：执行“yum -y install gcc-c++”命令安装gcc-c++依赖包。

3.3.5 nginx软件编译安装后，看不到程序目录（/application)

说明：编译安装步骤不对（配置 编译 编译安装生成/appliation）

3.3.6 nginx软件排查问题三部曲说明

a 在客户端上ping服务器端IP，检查链路是否通畅

b 在客户端上telnet服务器端IP、端口，检查链路访问是否通畅

c 在客户端上wget检测模拟页面访问是否正常

3.3.7 【注意】403状态码出现情况原因

01. 服务阻止客户端访问

02. 服务端站点目录中，没有指定首页文件信息

3.4 nginx软件使用命令参数

3.4.1 nginx 启动方法

```
[root@web01 application]# /application/nginx/sbin/nginx
```

3.4.2 nginx 停止方法

```
[root@web01 application]# /application/nginx/sbin/nginx -s stop
```

3.4.3 nginx 重启方法 （平滑重启）

```
[root@web01 application]# /application/nginx/sbin/nginx -s reload
```

3.4.4 检查配置文件语法是否正确

```
[root@web01 application]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
```

3.4.5 显示配置参数 -V （大写V）

```
[root@web01 application]# /application/nginx/sbin/nginx -V
nginx version: nginx/1.10.2
built by gcc 4.4.7 20120313 (Red Hat 4.4.7-18) (GCC)
built with OpenSSL 1.0.1e-fips 11 Feb 2013
TLS SNI support enabled
configure arguments: --prefix=/application/nginx-1.10.2 --user=www --group=www --with-http_stub_
```

3.4.6 nginx软件使用过程中深入说明

①. nginx软件语法检查方法:

```
    nginx -t
```

②. nginx软件访问测试过程:

```
curl -v www.baidu.com
```

③. nginx软件编译参数查看:

```
    nginx -V                <--- 查看原有的编译参数信息
```

3.5 nginx软件静态页面编写过程

编写配置文件

```
[root@web02 www]# cat ../../conf/nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    server {
        listen       80;
        server_name www.zinx.top;
        location / {
            root   html/www;
```

```
            index  clsn.html index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
}
```

编写静态访问页面文件信息

```
[root@web1 www]# cat clsn.html

"utf-8">




惨绿少年
01
02
03


"http://blog.znix.top">
nginx服务企业应用"znix.png" />
```

第4章 nginx进阶 --虚拟主机配置

4.1 【企业要求】需要按照以前nginx服务编译安装过程安装

1、参看已安装服务的配置参数信息

```
[root@web01 sbin]# /application/nginx/sbin/nginx -V
nginx version: nginx/1.10.2
built by gcc 4.4.7 20120313 (Red Hat 4.4.7-18) (GCC)
built with OpenSSL 1.0.1e-fips 11 Feb 2013
TLS SNI support enabled
configure arguments: --prefix=/application/nginx-1.10.2 --user=www --group=www --with-http_stub_
```

-V 参数能显示软件的详细详细，安装配置参数

2、按照配置参数进行部署

4.1.1 【语法检查】检查配置文件

```
[root@web01 nginx]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
```

```
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
```

4.2 首页文件不存在--利用nginx服务搭建文件共享服务器

通过配置 autoindex on; 参数

使用 autoindex参数，nginx能识别的直接显示，不识别的直接下载

配置完 autoindex on; 参数以后 会显示站点下的文件信息

对于nginx可以解析的资源会解析相应的内容

对于nginx不可以解析的资源会直接下载

4.2.1 进行curl时，报403错误，因为没有首页文件信息

```
[root@web02 ~]# echo 'web01 www' > /application/nginx/html/www/index.html
```

<- 在虚拟主机指定的站点目录中创建首页文件

```
[root@web02 ~]# curl www.nmtui.com
<- 利用curl命令本地检测nginx配置是否成功；已经存在首页文件，测试成功
```

4.2.2 autoindex on参数实践

1）修改配置文件

```
[root@web01 www]# cat ../../conf/nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    server {
        listen       10.0.0.8:80;
        server_name  www.nmtui.com;
        location / {
            root   html/www;
            autoindex on;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
}
```

2）重启服务

```
[root@web01 conf]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
[root@web01 conf]# /application/nginx/sbin/nginx -s reload
```

3）访问测试

4.3 【概念】虚拟主机的概念和类型

虚拟主机使用的是特殊的软硬件技术，它把一台运行在因特网上的服务器主机分成一台台“虚拟”的主机，每台虚拟主机都可以是一个独立的网站，可以具有独立的域名，具有完整的Internet服务器功能（WWW、FTP、Email等），同一台主机上的虚拟主机之间是完全独立的。从网站访问者来看，每一台虚拟主机和一台独立的主机完全一样。

利用虚拟主机，不用为每个要运行的网站提供一台单独的Nginx服务器或单独运行一组Nginx进程。虚拟主机提供了在同一台服务器、同一组Nginx进程上运行多个网站的功能。

4.3.1 虚拟主机概念

所谓虚拟主机，在Web服务里就是一个独立的网站站点，这个站点对应独立的域名（也可能是ip或端口.具有独立的程序及资源目录，可以独立地对外提供服务供用户访问。

这个独立的站点在配置里是由一定格式的标签段标记的，对于Apache软件来说，一个虚拟主机的标签段通常被包含在以的此，而Nginx软件则使用一个server{}标签来标示一个虚拟主机。一个Web服务里可以有多个虚拟主机标签对，即可以同时支持多个虚拟主机站点。

4.3.2 虚拟主机类型

常见的虚拟主机类型有如下几种

1）基于域名的虚拟主机

所谓基于域名的虚拟主机，意思就是通过不同的域名区分不同的虚拟主机，基于域名的虚拟主机是企业应用最广的虚拟主机类型，几乎所有对外提供服务的网站使用的都是基于域名的虚拟主机，例如: www.znix.top。

2）基于端口的虚拟主机

同理，所谓基于端口的虚拟主机，意思就是通过不同的端口来区分不同的虚拟主机，此类虚拟主机对应的企业应用主要为公司内部网站，例如：一些不希望直接对外提供用户访问的网站后台等，访问基于端口的虚拟主机，地址里要带有端口，例如：http://blog.znix.top:80

3）基于IP的虚拟主机

所谓基于IP的虚拟主机，意思是通过不同的IP区分不同的虚拟主机，

4.3.3 Nginx配置虚拟主机的步骤如下（适合各类虚拟主机类型）

1）增加一个完整的server标签段到结尾处。注意，要放在http的结束大括号前，也就是将server标签段放入http标签。

2)更改server_name及对应网页的root根目录，如果需要其他参数，可以增加或修改。

3)创建Seever_name域名对应网页的根目录，并且建立测试文件，如果没有index首页，访问会出现403错误。

> 如果是apache软件，没有首页文件，默认会把站点目录下面的信息显示出来
>
> nginx出403错误解决方式：http://clsn.blog.51cto.com/2561410/1633952
>
> autoindex on;#<==当找不到首页文件时，会展示目录结构，这个功能一般不要用除非有需求。
>
> PS:显示的目录结构中，有些信息点击就是下载，有的点击就是显示，因为扩展名称不一样
>
> 根本在于nginx软件是否能够进行解析
>
> nginx是否解析：
>
> 1.html jpg认识显示出内容
>
> 2.不认识不解析便直接下载

4)检查Nginx配置文件语法，平滑重启Nginx服务，快速检查启动结果。

5)在客户端对server_name处配置的域名做host解析或DNS配置，并检查（ping域名看返回的IP是否正确）。

6)在Win32浏览器中输入地址访问，或者在Linux客户端做hosts解析，用wget或curl接地址访问。

7）在服务重启或关闭之前先进行一次配置文件检查 /application/ngnix/sbin/nginx -t

Nginx虚拟主机的官方帮助网址为：http://Nginx.org/en/docs/http/requestj3rocessing.html

4.4 【实践】虚拟主机配置

4.4.1 基于域名的虚拟主机

修改配置文件

```
[root@web01 ~]# cat  /application/nginx/conf/nginx.conf
worker_processes  1;
events {
```

```
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    server {
        listen       80;
        server_name  www.nmtui.com;
        location / {
            root   html/www;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
    server {
        listen       80;
        server_name  bbs.nmtui.com;
        location / {
            root   html/bbs;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
    server {
        listen       80;
        server_name  blog.nmtui.com;
        location / {
            root   html/blog;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
}
```

检查配置信息是否正确

```
[root@web01 conf]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
```

重启服务

```
[root@web01 conf]# /application/nginx/sbin/nginx -s reload
```

注意：服务没有启动的时候不能使用平滑重启

创建站点目录

```
[root@web01 conf]# for name in www blog bbs ;do  mkdir ../html/$name -p ;done
```

创建主页文件

```
[root@web01 conf]# for name in www blog bbs ;do echo "web01 $name" >../html/$name/index.html  ;d
```

检查主页内容信息

```
[root@web01 conf]# for name in www blog bbs ;do  cat ../html/$name/index.html ;done
web01 www
web01 blog
web01 bbs
```

修改主机hosts文件

```
[root@web01 ~]# vim /etc/hosts
172.16.1.8    web01 www.nmtui.com blog.nmtui.com bbs.nmtui.com
```

测试

```
[root@web01 ~]# curl  www.nmtui.com
web01 www
[root@web01 ~]# curl  blog.nmtui.com
web01 blog
[root@web01 ~]# curl  bbs.nmtui.com
web01 bbs
```

4.4.2 基于端口的虚拟主机

修改配置文件内容

```
[root@web01 conf]# vim nginx.conf
    server {
        listen       81;
        server_name  bbs.nmtui.com;
        location / {
            root   html/bbs;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
"nginx.conf" 46L, 1098C written
```

重启服务

```
[root@web01 conf]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
[root@web01 conf]# /application/nginx/sbin/nginx -s reload
```

检查端口信息

```
[root@web01 conf]# netstat -lntup |grep ng
tcp        0      0 0.0.0.0:80                  0.0.0.0:*                   LISTEN      40110/ng
tcp        0      0 0.0.0.0:81                  0.0.0.0:*                   LISTEN      40110/ng
```

测试访问

```
[root@web01 ~]# curl  bbs.nmtui.com:81
web01 bbs
```

4.4.3 基于IP的虚拟主机

注意:

> 采用基于IP的虚拟主机，配置文件修改后要重启（-s stop）
>
> 配置和IP地址配置相关的都要采用（-s stop）重启，不能够使用软重启的方式

修改配置文件

```
[root@web01 conf]# cat nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    server {
        listen       10.0.0.8:80;
        server_name  www.nmtui.com;
        location / {
            root   html/www;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
}
```

检查配置文件格式

```
[root@web01 conf]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
```

重启服务，注意使用的是直接重启的方式

只要nginx配置文件中涉及到IP地址的更改只能正真的重启才生效

```
[root@web01 conf]# /application/nginx/sbin/nginx -s stop
[root@web01 conf]# /application/nginx/sbin/nginx
[root@web01 conf]# netstat -lntup |grep ng
tcp        0      0 10.0.0.8:80                 0.0.0.0:*                   LISTEN      40592/ng
```

访问测试

4.5 【规范化配置】nginx配置文件企业规范化

4.5.1 第一个里程碑： 创建虚拟主机配置文件存储目录

```
[root@web01 conf]# pwd
/application/nginx/conf
[root@web01 conf]# mkdir extra
```

4.5.2 第二个里程碑： 生产虚拟主机配置文件

```
[root@web01 conf]# sed -n '10,21p' nginx.conf > extra/www.conf
[root@web01 conf]# sed -n '22,33p' nginx.conf > extra/bbs.conf
[root@web01 conf]# sed -n '34,45p' nginx.conf > extra/blog.conf
```

4.5.3 第三个里程碑： 修改nginx配置文件使之加载识别虚拟主机配置文件

```
[root@web01 conf]# cat  nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    include extra/*;
}
```

4.5.4 重启服务

```
[root@web01 conf]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
```

4.5.5 检查监听端口

```
[root@web01 conf]# netstat -lntup |grep ng
tcp        0      0 0.0.0.0:80              0.0.0.0:*       LISTEN      40714/nginx
```

4.5.6 查看配置文件的加载顺序

```
[root@web01 logs]# /application/nginx/sbin/nginx -T
```

参数说明:

-T : test configuration, dump it and exit

测试配置文件，并且加载一遍，并显示加载的顺序

4.5.7 【优化】调整 inculde的加载顺序，指定第一个加载为conf

```
[root@web01 conf]# vim nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    include extra/www.conf;
    include extra/bbs.conf;
    include extra/blog.conf;
}
```

4.5.8 重启服务

```
[root@web01 conf]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
[root@web01 conf]# /application/nginx/sbin/nginx -s reload
```

4.5.9 说明;

这样的配置能够让用户通过IP访问的时候，访问到的网站是www的网站。

4.6 别名的配置

在配置文件中添加别名

```
[root@web01 conf]# vim extra/www.conf
    server {
        listen       80;
        server_name  www.nmtui.com nmtui.cn;
        location / {
            root   html/www;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
```

```
        }
    }
}
```

重启服务

```
[root@web01 conf]# /application/nginx/sbin/nginx -s reload
```

修改hosts 进行访问测试

```
[root@web01 www]# curl  nmtui.cn
web01 www
```

4.7 status 状态模块

4.7.1 状态模块的配置

修改配置文件，添加三status模块

```
[root@web01 conf]# vim nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
          server{
          listen  80;
          server_name  status.nmtui.com;
          location / {
            stub_status on;
            access_log   off;
          }
        }
    include extra/www.conf;
    include extra/bbs.conf;
    include extra/blog.conf;
}
```

检查配置文件是否正确

```
[root@web01 conf]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
```

重启服务

```
[root@web01 conf]# /application/nginx/sbin/nginx -s reload
```

访问测试

说明：

以上页面内容信息主要会被zabbix监控服务调取，形成图像信息
根据图形信息，从而判断nginx网站服务用户访问量情况

4.7.2 状态模块说明

参数	参数说明
Active connections	当前的活动客户端连接数量
accepts	接受客户端连接的总数
handled	处理的连接总数
requests	客户端请求的总数
Reading	nginx正在读请求头的当前连接数。
Writing	nginx正在将响应写回客户端的当前连接数。
Waiting	当前空闲客户端连接数等待一个请求。

4.8 nginx的日志功能

nginx的两种日志种类

错误日志：记录nginx运行错误情况信息

访问日志：记录用户访问日志信息

官方说明： http://nginx.org/en/docs/ngx_core_module.html#error_log

4.8.1 定义错误日志信息

系统默认配置

```
#error_log  logs/error.log;
#error_log  logs/error.log  notice;
#error_log  logs/error.log  info;
```

配置错误日志，修改主配置文件

```
[root@web01 logs]# vim ../conf/nginx.conf
worker_processes  1;
error_log  logs/error.log  error;
events {
    worker_connections  1024;
}
```

```
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
          server{
          listen  80;
          server_name  status.nmtui.com;
          location / {
            stub_status on;
            access_log   off;
          }
        }
    include extra/www.conf;
    include extra/bbs.conf;
    include extra/blog.conf;
}
```

重启服务

```
[root@web01 logs]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
[root@web01 logs]# /application/nginx/sbin/nginx -s reload
```

查看错误日志信息

```
[root@web01 logs]# tail  error.log
2017/10/25 11:41:55 [error] 40842#0: *7 open() "/application/nginx-1.10.2/html/www/favicon.ico"
```

4.8.2 访问日志配置

系统默认配置

```
#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;
```

修改访问日志配置--修改主配置文件

```
[root@web01 www]# cat ../../conf/nginx.conf
worker_processes  1;
error_log  logs/error.log  error;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
```

```
                  '$status $body_bytes_sent "$http_referer" '
                  '"$http_user_agent" "$http_x_forwarded_for"';

          server{
          listen  80;
          server_name  status.nmtui.com;
          location / {
            stub_status on;
            access_log   off;
          }
        }
    include extra/www.conf;
    include extra/bbs.conf;
    include extra/blog.conf;
}
```

修改访问日志配置--修改虚拟主机配置文件

```
[root@web01 www]# cat /application/nginx/conf/extra/www.conf
    server {
        listen       80;
        server_name  www.nmtui.com nmtui.cn;
        location / {
            root   html/www;
            index  index.html index.htm;
        }
         access_log  logs/access_www.log  main;
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
```

4.8.3 【重要】访问日志信息说明

日志内容

```
10.0.0.1 - - [22/Oct/2017:16:04:54 +0800] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10
```

配置文件

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

日志信息说明

参数	日志内容	含义
$remote_addr	10.0.0.1	客户端ip地址
-	-	

$remote_user	-	显示远程访问者用户信息
[$time_local]	[22/Oct/2017:16:04:54 +0800]	显示访问时间
$request	GET / HTTP/1.1"	请求行信息
$status	304	状态码
$body_bytes_sent	0	响应报文主体内容大小
$http_referer	-	
$http_user_agent	Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.100 Safari/537.36	客户端浏览网页工具信息
$http_x_forwarded_for	-	反向代理转发

详细日志详细说明

Nginx日志变量	说明
$remote_addr	记录访问网站的客户端地址；即源IP地址
$http_x_forwarded_for	当前端有代理服务器时，设置web节点记录客户端地址的配置，此参数生效的 前提是代理服务器上也进行了相关的x_forwarded_for设置 可以记录用户真实的IP地址信息
$remote_user	远程客户端用户名称
$time_local	记录访问时间与时区
$request	用户的http请求起始行信息
$status	http状态码，记录请求返回的状态，例如：200 , 404 , 301等
$body_bytes_sents	服务器发送给客户端的响应body字节数
$http_referer	记录此次请求是从哪个链接访问过来的，可以根据referer进行防盗链设置 即表示是哪个网站介绍过来的

$http_user_agent	记录客户端访问信息，例如：浏览器、手机客户端等

在没有特殊要求的情况下，采用默认的配置即可，更多可以设置的记录日志信息的变量见：
http://nginx.org/en/docs/http/ngx_httpJog_module.html1

4.8.4 日志的切割

切割日志方式

01：利用脚本实现日志切割

简单脚本

```
[root@web01 logs]# cat /server/scripts/log_cut.sh
#!/bin/bash

mv /application/nginx/logs/access_www.log  /application/nginx/logs/access_www_`date +%F`.log
/application/nginx/sbin/nginx -s reload
```

包含判断的脚本

```
[root@web01 logs]# cat /server/script/cut_nginx_log.sh
# ! /bin/sh
Dateformat='date +%Y%m%d'
Basedir= "/application/nginx"
Nginxlogdir="$Basedir/logs"
Logname="access_www"
[-d $Nginxlogdir ] && cd $Nginxlogdir || exit 1
[-f ${Logname}.log ] || exit 1
/bin/mv ${Logname}.log ${Dateformat}_${Logname}.log
$Basedir/sbin/nginx -s reload
```

02.利用系统自带切割软件进行切割

```
cat /etc/logrotate.conf
```

4.9 企业需求解决（location应用）

1、搭建好一台nginx的web服务器，配置好内网网卡地址与外网网卡地址

2、web服务器的网站域名为www.nmtui.com 站点目录为 html/www

3、要求内网可以访问 www.nmtui.com/AV 资源

4、要求外网不可以访问 www.nmtui.com/AV 资源

4.9.1 需求处理 --location的应用

1、定位需要控制的资源

location == if

修改配置文件

```
[root@web01 extra]# cat www.conf
    server {
        listen       80;
        server_name  www.nmtui.com;
        location / {
            root   html/www;
            index  index.html index.htm;
        }
        location /AV  {
            root  html/www;
            index index.html index.htm;
            allow 172.16.1.0/24;
            deny  all;
        }
    }
```

4.9.2 location 语法

location 指令的作用是根据用户请求的URI来执行不同的应用。

locationn使用的语法为

```
location [=|~|~*|^~] uri {
  ....
}
```

location 语法说明表

location	[=|~|~*|^~]	uri	{....}
指令	匹配标识	匹配的网站地址	匹配URI后要执行的配置段

~ 与~* 的区别

- ~ 匹配内容区分大小写
- ~* 匹配内容不区分的小写
- !~ 取反
- ^~ 但多个匹配同时存在，优先匹配 ^~匹配的内容;不做正则表达式的检查 （优先处理）

4.9.3 官方配置示例

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

location / {
    [ configuration B ]
}
```

```
location /documents/ {
    [ configuration C ]
}

location ^~ /images/ {
    [ configuration D ]
}

location ~* \.(gif|jpg|jpeg)$ {
    [ configuration E ]
}
```

说明：

> "/"请求将匹配配置A，
>
> "/index.html"请求将匹配配置B，
>
> "/documents/document.html"请求将匹配配置C，
>
> "/images/1.gif"请求将匹配配置D，
>
> "/documents/1.jpg"请求将匹配配置E.

不同uri及特殊字符组合匹配的顺序说明

顺序	不用URI及特殊字符组合匹配	匹配说明
1	location = / {}	精确匹配 /
2	location ^~ /image/{	匹配常规字符串，不做正则表达式匹配检查
3	location ~* \.(gif|jpg|jpeg)$ {	正则匹配
4	location /documents/ {	匹配常规字符串，如果有正则，则优先匹配正则
5	location / {	所有location 都不能匹配后的默认匹配

4.9.4 【测试】测试location的访问

4.9.4.1 修改配置文件

定义不同的location返回不同的数值

```
[root@web01 extra]# vim www.conf
server {
    listen       80;
    server_name  www.nmtui.com nmtui.com;
    root   html/www;
    location / {
       return 401;
```

```
    }
    location = / {
        return 402;
    }
    location /documents/ {
        return 403;
    }
    location ^~ /images/ {
        return 404;
    }
    location ~* \.（gif|jpg|jpeg）$ {
    return 500;
    }
    access_log logs/access_www.log main;
}
```

4.9.4.2 访问测试

根据请求不同uri的返回值验证 location的配置。

```
[root@nfs01 ~]# curl -I -w "%{http_code}\n" -o /dev/null -s  www.nmtui.com/documents
401
[root@nfs01 ~]# curl -I -w "%{http_code}\n" -o /dev/null -s  www.nmtui.com
402
[root@nfs01 ~]# curl -I -w "%{http_code}\n" -o /dev/null -s  www.nmtui.com/documents/ss.jpg
500
[root@nfs01 ~]# curl -I -w "%{http_code}\n" -o /dev/null -s  www.nmtui.com/documents
401
[root@nfs01 ~]# curl -I -w "%{http_code}\n" -o /dev/null -s  www.nmtui.com/documents/
403
[root@nfs01 ~]# curl -I -w "%{http_code}\n" -o /dev/null -s  www.nmtui.com/images/1.jpg
404
```

4.10 rewrite 模块的使用--地址重写

4.10.1 rewrite 重写模块

将地址信息进行重写

rewrite 语法格式

```
rewrite regex replacement [flag]
```

rewrite应用标签：server、location、if

4.10.2 rewrite模块两个功能

1. 实现网站地址信息跳转
2. 实现伪静态

4.10.3 方法一 使用if判断

```
[root@web01 extra]# cat www.conf
    server {
        listen       80;
        server_name  www.nmtui.com nmtui.cn;
if ($host ~* "^nmtui.com$") {
rewrite ^/(.*) http://www.nmtui.com/$1 permanent;
}
        location / {
            root   html/www;
            index  index.html index.htm;
        }
        access_log  logs/access_www.log  main;
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
```

测试

```
[root@web01 www]# curl  nmtui.com -L
web01 www
```

4.10.4 方法二 再添加上一个区块

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

4.10.5 nginx 的rwite重写企业应用场景

可以调整用户浏览的URL，使其看起来更规范，合乎开发及产品人员的需求。

为了让搜索引擎收录网站内容，并让用户体验更好，企业会将动态URL地址伪装成静态地址提供服务。

网站换新域名后，让旧域名的访问跳转到新的域名上，例如：让京东的360buy换成了jd.com。

根据特殊变量、目录、客户端的信息进行URL跳转等。

说明：开源软件类似wordpress的，官方都会对伪静态配置进行说明

4.11 nginx 的访问认证

4.11.1 修改nginx的相关配置文件

```
vim extra/www.conf
location / {
        root   html/www;
        index  index.html index.htm;
        auth_basic           "clsn training";
        auth_basic_user_file   /application/nginx/conf/htpasswd;
 }
```

4.11.2 创建密码文件

注： 这里使用的htpasswd命令默认是没有的，需要通过yum install httpd-tools -y 安装

```
[root@web01 extra]# htpasswd -c /application/nginx/conf/htpasswd clsn
New password:
Re-type new password:
Adding password for user clsn
```

参数说明：

-c Create a new file.

创建一个新的密码文

-b Use the password from the command line rather than prompting for it.

采用免交互的方式输入用户的密码信息

htpasswd参数说明

参数	参数说明
-c	创建一个新文件。
-n	不更新文件; 显示结果。
-m	强制MD5密码加密。
-d	强制CRYPT加密密码（默认）。
-p	不加密密码（明文）。
-s	强制SHA加密密码。
-b	使用命令行中的密码，而不是提示。（免交互）
-D	删除指定的用户。

4.11.3 更改密码文件权限

```
[root@web01 extra]#  chmod 400 /application/nginx/conf/htpasswd
[root@web01 extra]#  chown -R www.www /application/nginx/conf/htpasswd
[root@web01 extra]# cat /application/nginx/conf/htpasswd
clsn:e3OfMiJThE0Qg
```

4.11.4 重启服务： 配置修改后要重启服务

```
[root@web01 extra]# /application/nginx/sbin/nginx -t
nginx: the configuration file /application/nginx-1.10.2/conf/nginx.conf syntax is ok
```

```
nginx: configuration file /application/nginx-1.10.2/conf/nginx.conf test is successful
[root@web01 extra]# /application/nginx/sbin/nginx -s reload
```

4.11.5 访问测试

```
# 使用交互式输入密码
[root@web01 www]# curl www.nmtui.com -uclsn
Enter host password for user 'clsn':123456
web01 www
# 使用免交互输入密码
[root@web01 www]# curl www.nmtui.com -uclsn:123456
web01 www
```

401错误说明： 需要认证，但是没有认证

赞1

如无特殊说明，文章均为本站原创，转载请注明出处

- 转载请注明来源：nginx服务企业应用
- 本文永久链接地址：https://www.nmtui.com/clsn/lx892.html

该文章由 惨绿少年 发布

惨绿少年Linux www.nmtui.com