# WEB

**APACHE:**是世界使用排名第一的Web服务器软件。它可以运行在几乎所有广泛使用的计算机平台上，由于其跨平台和安全性被广泛使用，是最流行的Web服务器端软件之一。

Web网站服务端：软件包httpd、系统服务httpd

传输协议及端口：TCP 80

Web网站服务端配置文件：

/etc/httpd/conf/httpd.conf /etc/httpd/conf.d/\*.conf

默认首页文件：index.html

httpd网站文档的默认根目录:/var/www/html

Apache的三种工作模式：Prefork、Worker 、Event

Prefork优点：成熟，兼容所有新老模块。进程之间完全独立，使得它非常稳定。同时，不需要担心线程安全的问题。（我们常用的mod\_php，PHP的拓展不需要支持线程安全）

缺点：一个进程相对占用更多的系统资源，消耗更多的内存。而且，它并不擅长处理高并发请求，在这种场景下，它会将请求放进队列中，一直等到有可用进程，请求才会被处理。

Worker 优点：占据更少的内存，高并发下表现更优秀。

缺点：必须考虑线程安全的问题，因为多个子线程是共享父进程的内存地址的。如果使用keep-alive的长连接方式，也许中间几乎没有请求，这时就会发生阻塞，线程被挂起，需要一直等待到超时才会被释放。如果过多的线程，被这样占据，也会导致在高并发场景下的无服务线程可用。（该问题在prefork模式下，同样会发生）

Event

这个是 Apache中最新的模式，在现在版本里的已经是稳定可用的模式。event MPM在遇到某些不兼容的模块时，会失效，将会回退到worker模式，一个工作线程处理一个请求。官方自带的模块，全部是支持event MPM的。

## **虚拟Web主机的部署**

<VirtualHost \*:80>

ServerName 网站1的FQDN

DocumentRoot 网站1的网页根目录

</VirtualHost>

<VirtualHost \*:80>

ServerName 网站2的FQDN

DocumentRoot 网站2的网页根目录

</VirtualHost>

## **配置网页内容访问**

<Directory "父目录路径">

Require all denied //上层目录拒绝任何访问

</Directory>

<Directory "子目录1路径">

Require all granted //子目录1允许任何访问

</Directory>

<Directory "子目录2路径">

Require ip IP或网段地址 .. .. //子目录2允许少数客户机

</Directory>

## **部署WSGI站点**

安装页面解释程序：mod\_wsgi

配置文件：

Listen 8909 //监听端口

<VirtualHost \*:8909>

DocumentRoot //网页根目录

ServerName

WSGIScriptAlias / /var/www/webapp0/webinfo.wsgi

</VirtualHost>

调整SELinux策略，允许Web服务使用8909端口

semanage port -a -t http\_port\_t -p tcp 8909

## **部署安全web服务器**

https协议：443端口

安装加密软件 mod\_ssl

部署网站证书：/etc/pki/tls/certs .crt

部署根证书：/etc/pki/tls/certs .crt

部署私钥：/etc/pki/tls/private .key

配置文件 ServerName 域名：443

**Nginx：**是一款轻量级的Web 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器，并在一个BSD-like 协议下发行。其特点是占有内存少，并发能力强，事实上nginx的并发能力确实在同类型的网页服务器中表现较好。

使用源码包安装nginx软件包

yum -y install gcc pcre-devel openssl-devel //安装依赖

解压缩

./configure \

>--without-http\_autoindex\_module \ //禁用自动索引文件目录模块

>--without-http\_ssi\_module //禁用支持服务器端的include

> --prefix=/usr/local/nginx \ //指定安装路径

> --user=nginx \ //指定用户

> --group=nginx \ //指定组

> --with-http\_ssl\_module //开启SSL加密功能

make && make install

[root@proxy nginx-1.12]# vim +48 src/http/ngx\_http\_header\_filter\_module.c

//注意：vim这条命令必须在nginx源码包目录下执行！！！！！！

//该文件修改前效果如下：

static u\_char ngx\_http\_server\_string[] = "Server: nginx" CRLF;

static u\_char ngx\_http\_server\_full\_string[] = "Server: " NGINX\_VER CRLF;

static u\_char ngx\_http\_server\_build\_string[] = "Server: " NGINX\_VER\_BUILD CRLF;

//下面是我们修改后的效果：

static u\_char ngx\_http\_server\_string[] = "Server: Jacob" CRLF;

static u\_char ngx\_http\_server\_full\_string[] = "Server: Jacob" CRLF;

static u\_char ngx\_http\_server\_build\_string[] = "Server: Jacob" CRLF;

//修改完成后，再去编译安装Nignx，版本信息将不再显示为Nginx，而是Jacob

[root@proxy nginx-1.12]# ./configure

[root@proxy nginx-1.12]# make && make install

[root@proxy nginx-1.12]# killall nginx

[root@proxy nginx-1.12]# /usr/local/nginx/sbin/nginx //启动服务

# 最小化安装永远是对的！

/usr/local/nginx/sbin/nginx //启动服务

/usr/local/nginx/sbin/nginx -s stop //关闭服务

/usr/local/nginx/sbin/nginx -s reload //重新加载配置文件

/usr/local/nginx/sbin/nginx –V //查看软件信息

ln -s /usr/local/nginx/sbin/nginx /sbin/ //方便后期使用

Nginx Web服务默认首页文档存储目录为/usr/local/nginx/html/

配置文件：/usr/local/nginx/conf/nginx.conf

## **基于域名的虚拟主机**

yum -y install httpd-tools

htpasswd -c /usr/local/nginx/pass tom //创建密码文件

htpasswd /usr/local/nginx/pass jerry //追加用户，不使用-c选项

/usr/local/nginx/sbin/nginx -s reload //重新加载配置文件

修改版本信息，并隐藏具体的版本号

… …

http{

server\_tokens off; //在http下面手动添加这么一行

… …

}

虚拟主机一般可用分为：基于域名、基于IP和基于端口的虚拟主机。

vim /usr/local/nginx/conf/nginx.conf

.. ..

server {

listen 80; //端口

server\_name www.a.com; //域名

auth\_basic "Input Password:"; //认证提示符

auth\_basic\_user\_file "/usr/local/nginx/pass"; //认证密码文件

location / {

root html; //指定网站根路径

index index.html index.htm;

}

}

… …

server {

listen 80; //端口

server\_name www.b.com; //域名

location / {

root www; //指定网站根路径

index index.html index.htm;

}

}

## **SSL虚拟主机**

源码安装Nginx时必须使用--with-http\_ssl\_module参数，启用加密模块，对于需要进行SSL加密处理的站点添加ssl相关指令（设置网站需要的私钥和证书）。

加密算法一般分为对称算法、非对称算法、信息摘要。

对称算法有：AES、DES，主要应用在单机数据加密。

非对称算法有：RSA、DSA，主要应用在网络数据加密。

信息摘要：MD5、sha256，主要应用在数据完整性校验、数据秒传等

1）生成私钥与证书

cd /usr/local/nginx/conf

openssl genrsa > cert.key //生成私钥

openssl req -new -x509 -key cert.key > cert.pem //生成证书

2）修改Nginx配置文件，设置加密网站的虚拟主机

vim /usr/local/nginx/conf/nginx.conf

… …

server {

listen 443 ssl;

server\_name www.c.com;

ssl\_certificate cert.pem; #这里是证书文件

ssl\_certificate\_key cert.key; #这里是私钥文件

ssl\_session\_cache shared:SSL:1m; #设置存储session参数的缓存的类型和大小

ssl\_session\_timeout 5m; #指定客户端可以重用会话参数的时间（超时后不可使用）

ssl\_protocols SSLv2 SSLv3 TLSv1; #协议配置

ssl\_ciphers HIGH:!aNULL:!MD5; #说明密码

ssl\_prefer\_server\_ciphers on;

location / {

root html;

index index.html index.htm;

}

}

ssl\_session\_cache:

Off:严格禁止使用会话缓存：nginx明确告知客户端会话不可重用。

None:会话缓存是不允许的：nginx告知客户端会话可以重用，但并没有在缓存中存储会话参数。

Builtin:在OpenSSL中构建缓存；只能被一个工作进程使用。缓存的大小在会话中指定，如果没有指定大小，默认20480个会话。使用内置缓存会导致内存碎片化。

Shared:缓存在所有工作进程之间共享。缓存大小按照字节为单位指定；1MB可以存储4000个会话。每块共享内存都应该起个名字。同一块缓存可以在多个虚拟服务中使用。

ssl\_protocols 却属于全局配置，而 ssl\_ciphers 却针对特定的虚拟主机起作用

**优化Nginx服务(部分)**

Nginx安全优化包括：删除不要的模块、修改版本信息、限制并发、拒绝非法请求、防止buffer溢出

优化Nginx并发量、优化Linux内核参数（最大文件数量）、优化Nginx数据包头缓存、定义对静态页面的缓存时间、对页面进行压缩处理、服务器内存缓存

**限制并发量**

DDOS攻击者会发送大量的并发连接，占用服务器资源（包括连接数、带宽等），这样会导致正常用户处于等待或无法访问服务器的状态。

Nginx提供了一个ngx\_http\_limit\_req\_module模块，可以有效降低DDOS攻击的风险，操作方法如下：

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

… …

http{

… …

limit\_req\_zone $binary\_remote\_addr zone=one:10m rate=1r/s;

server {

listen 80;

server\_name localhost;

limit\_req zone=one burst=5;

}

}

//备注说明：

//limit\_req\_zone语法格式如下：

//limit\_req\_zone key zone=name:size rate=rate;

//上面案例中是将客户端IP信息存储名称为one的共享内存，内存空间为10M

//1M可以存储8千个IP信息，10M可以存储8万个主机连接的状态，容量可以根据需要任意调整

//每秒中仅接受1个请求，多余的放入漏斗

//漏斗超过5个则报错

**拒绝非法的请求**

网站使用的是HTTP协议，该协议中定义了很多方法，可以让用户连接服务器，获得需要的资源。但实际应用中一般仅需要get和post。

通过如下设置可以让Nginx拒绝非法的请求方法：

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

http{

server {

listen 80;

#这里，!符号表示对正则取反，~符号是正则匹配符号

#如果用户使用非GET或POST方法访问网站，则retrun返回444的错误信息

if ($request\_method !~ ^(GET|POST)$ ) {

return 444;

}

}

}

[root@proxy ~]# /usr/local/nginx/sbin/nginx -s reload

**防止buffer溢出**

当客户端连接服务器时，服务器会启用各种缓存，用来存放连接的状态信息。

如果攻击者发送大量的连接请求，而服务器不对缓存做限制的话，内存数据就有可能溢出（空间不足）。

修改Nginx配置文件，调整各种buffer参数，可以有效降低溢出风险。

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

http{

client\_body\_buffer\_size 1K;

client\_header\_buffer\_size 1k;

client\_max\_body\_size 1k;

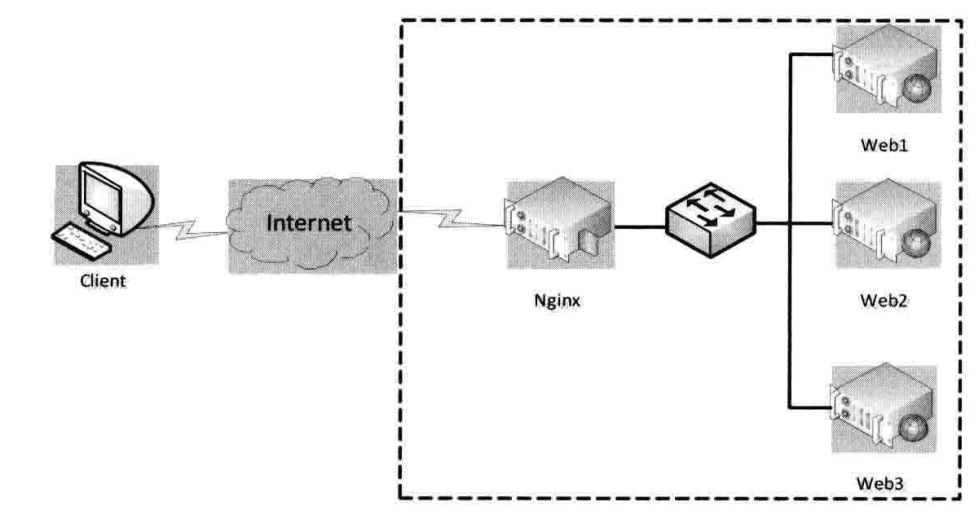
large\_client\_header\_buffers 2 1k;

… …

}

[root@proxy ~]# /usr/local/nginx/sbin/nginx -s reload

**Nginx负载均衡**



[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

.. ..

http {

.. ..

#使用upstream定义后端服务器集群，集群名称任意(如webserver)

#使用server定义集群中的具体服务器和端口

upstream webserver {

#通过ip\_hash设置调度规则为：相同客户端访问相同服务器

ip\_hash;

server 192.168.2.100:80 weight=1 max\_fails=1 fail\_timeout=30;

server 192.168.2.200:80 weight=2 max\_fails=2 fail\_timeout=30;

}

#weight设置服务器权重值，默认值为1

#max\_fails设置最大失败次数

#fail\_timeout设置失败超时时间，单位为秒

#down标记服务器已关机，不参与集群调度

.. ..

server {

listen 80;

server\_name localhost;

location / {

#通过proxy\_pass将用户的请求转发给webserver集群

proxy\_pass http://webserver;

}

}

Nginx的面试问题：

1、请解释一下什么是Nginx?

Nginx是一个web服务器和反向代理服务器，用于HTTP、HTTPS、SMTP、POP3和IMAP协议。

2、请列举Nginx的一些特性。

Nginx服务器的特性包括：

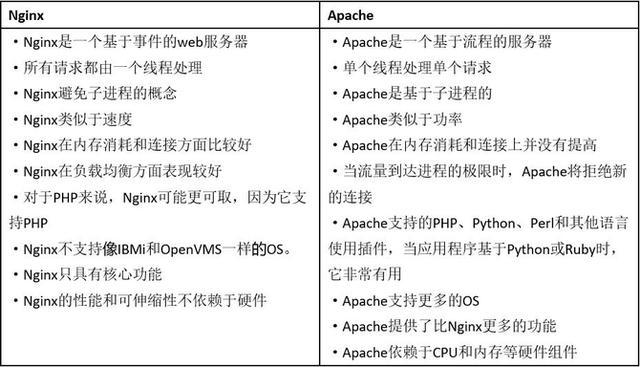
反向代理/L7负载均衡器

嵌入式Perl解释器

动态二进制升级

可用于重新编写URL，具有非常好的PCRE支持

3、请列举Nginx和Apache 之间的不同点。



4、请解释Nginx如何处理HTTP请求。

Nginx使用反应器模式。主事件循环等待操作系统发出准备事件的信号，这样数据就可以从套接字读取，在该实例中读取到缓冲区并进行处理。单个线程可以提供数万个并发连接。

6、 使用“反向代理服务器”的优点是什么?

反向代理服务器可以隐藏源服务器的存在和特征。它充当互联网云和web服务器之间的中间层。这对于安全方面来说是很好的，特别是当您使用web托管服务时。

7、请列举Nginx服务器的最佳用途。

Nginx服务器的最佳用法是在网络上部署动态HTTP内容，使用SCGI、WSGI应用程序服务器、用于脚本的FastCGI处理程序。它还可以作为负载均衡器。

8、请解释Nginx服务器上的Master和Worker进程分别是什么?

Master进程：读取及评估配置和维持

Worker进程：处理请求

9、请解释你如何通过不同于80的端口开启Nginx?

为了通过一个不同的端口开启Nginx，你必须进入/etc/Nginx/sites-enabled/，如果这是默认文件，那么你必须打开名为“default”的文件。编辑文件，并放置在你想要的端口：

Like server { listen 81; }

10、请解释是否有可能将Nginx的错误替换为502错误、503?

502 =错误网关

503 =服务器超载

有可能，但是您可以确保fastcgi\_intercept\_errors被设置为ON，并使用错误页面指令。

Location / {

fastcgi\_pass 127.0.01:9001;

fastcgi\_intercept\_errors on;

error\_page 502 =503/error\_page.html;

#…

}

12、请解释ngx\_http\_upstream\_module的作用是什么?

ngx\_http\_upstream\_module用于定义可通过fastcgi传递、proxy传递、uwsgi传递、memcached传递和scgi传递指令来引用的服务器组。

13、请解释什么是C10K问题?

C10K问题是指无法同时处理大量客户端(10,000)的网络套接字。

14、请陈述stub\_status和sub\_filter指令的作用是什么?

Stub\_status指令：该指令用于了解Nginx当前状态的当前状态，如当前的活动连接，接受和处理当前读/写/等待连接的总数

Sub\_filter指令：它用于搜索和替换响应中的内容，并快速修复陈旧的数据

15、解释Nginx是否支持将请求压缩到上游?

您可以使用Nginx模块gunzip将请求压缩到上游。gunzip模块是一个过滤器，它可以对不支持“gzip”编码方法的客户机或服务器使用“内容编码:gzip”来解压缩响应。

16、解释如何在Nginx中获得当前的时间?

要获得Nginx的当前时间，必须使用SSI模块、$date\_gmt和$date\_local的变量。

Proxy\_set\_header THE-TIME $date\_gmt;

17、用Nginx服务器解释-s的目的是什么?

用于运行Nginx -s参数的可执行文件。

18、解释如何在Nginx服务器上添加模块?

在编译过程中，必须选择Nginx模块，因为Nginx不支持模块的运行时间选择。

19、nginx负载均衡的算法

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。

3)、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。

20、Nginx优秀模块 模块设计：

高度模块化设计，除了少量核心代码，其他一切接模块。官方Nginx共有五大类型模块：核心模块、配置模块、事件模块、HTTP模块、mail模块。

要注意的是：nginx的模块是静态的，添加和删除模块都要对nginx进行重新编译，这一点与Apache的动态模块完全不同。

21、为什么要用Nginx？

优点：

跨平台、配置简单

非阻塞、高并发连接：处理2-3万并发连接数，官方监测能支持5万并发

内存消耗小：开启10个nginx才占150M内存，Nginx采取了分阶段资源分配技术

nginx处理静态文件好,耗费内存少

内置的健康检查功能：如果有一个服务器宕机，会做一个健康检查，再发送的请求就不会发送到宕机的服务器了。重新将请求提交到其他的节点上。

节省宽带：支持GZIP压缩，可以添加浏览器本地缓存

稳定性高：宕机的概率非常小

master/worker结构：一个master进程，生成一个或者多个worker进程

接收用户请求是异步的：浏览器将请求发送到nginx服务器，它先将用户请求全部接收下来，再一次性发送给后端web服务器，极大减轻了web服务器的压力

一边接收web服务器的返回数据，一边发送给浏览器客户端

网络依赖性比较低，只要ping通就可以负载均衡

可以有多台nginx服务器

事件驱动：通信机制采用epoll模型

22、为什么Nginx性能这么高？

得益于它的事件处理机制：

异步非阻塞事件处理机制：运用了epoll模型，提供了一个队列，排队解决

Nginx响应码

1XX代表提示信息

2XX代表成功信息

3XX代表重定向信息

4XX代表客户端错误信息

5XX代表服务器错误信息



**Tomcat:** 是一个免费的开放源代码的Web 应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP 程序的首选。

端口：8080

**部署Tomcat服务器软件**

1. 使用RPM安装JDK环境

yum –y install java-1.8.0-openjdk java-1.8.0-openjdk-headless //安装JDK

1. 安装Tomcat （解压、拷贝）

mv apache-tomcat-8.0.30 /usr/local/tomcat

1. 启动服务

/usr/local/tomcat/bin/startup.sh

**修改Tomcat配置文件**

1. 创建测试JSP页面

[root@web1 ~]# vim /usr/local/tomcat/webapps/ROOT/test.jsp

<html>

<body>

<center>

Now time is: <%=new java.util.Date()%> //显示服务器当前时间

</center>

</body>

</html>

2）重启服务

[root@web1 ~]# /usr/local/tomcat/bin/shutdown.sh

[root@web1 ~]# /usr/local/tomcat/bin/startup.sh

## **使用Tomcat部署虚拟主机**

修改server.xml配置文件，创建两个域名的虚拟主机，修改如下两个参数块：

# cat /usr/local/tomcat/conf/server.xml

<Server>

<Service>

<Connector port=8080 />

<Connector port=8009 />

<Engine name="Catalina" defaultHost="localhost">

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

<Context path="/test" docBase="/var/www/html/" />

#当用户访问http://www.a.com/test打开/var/www/html目录下的页面

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs" prefix=" a\_access" suffix=".txt" pattern="%h %l %u %t &quot;%r&quot; %s %b" />

#配置Tomcat日志

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

<Context path="" docBase="base" reloadable="true"/>

#修改www.b.com网站的首页目录为base

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs" prefix=" a\_access" suffix=".txt" pattern="%h %l %u %t &quot;%r&quot; %s %b" />

#配置Tomcat日志

</Host>

**配置Tomcat支持SSL加密网站**

<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"

maxThreads="150" SSLEnabled="true" scheme="https" secure="true"

keystoreFile="/usr/local/tomcat/keystore" keystorePass="123456" clientAuth="false" sslProtocol="TLS" />

//备注，默认这段Connector被注释掉了，打开注释，添加密钥信息即可

**Tomcat 面试问题**

1）解释什么是Jasper?

　　Jasper是Tomcat的JSP引擎

　　它解析JSP文件，将它们编译成JAVA代码作为servlet

　　在运行时，Jasper允许自动检测JSP文件的更改并重新编译它们

2）tomcat支持多大的连接？怎样集群？怎样保证唯一session?

tomcat 默认maxThreads=200 不建议修改超过500，太大性能下降的很大，集群话的 可以通过nginx, 配置多个tomcat搞集群，session 的话 niginx 提供同一域ip\_hash 配置，如果互相网应用的话，建议吧用户状态拉出来，其他服务弹性部署更容易，用户状态缓存到redis中！

3）Tomcat的缺省是多少，怎么修改

Tomcat的缺省端口号是8080.

修改Tomcat端口号：

1.找到Tomcat目录下的conf文件夹

2.进入conf文件夹里面找到server.xml文件

3.打开server.xml文件

4.在server.xml文件里面找到下列信息

maxThreads=”150″ minSpareThreads=”25″ maxSpareThreads=”75″

enableLookups=”false” redirectPort=”8443″ acceptCount=”100″

connectionTimeout=”20000″ disableUploadTimeout=”true” />

5.把port=”8080″改成port=”8888″，并且保存

6.启动Tomcat，并且在IE浏览器里面的地址栏输入http://127.0.0.1:8888/

LNMP:LINUX+NGINX+MYSQL+PHP

LAMP:LINUX+APACHE+MYSQL+PHP

LNMP安装：

nginx

mariadb、mariadb-server、mariadb-devel、php-mysql

php、php-fpm（需要自己下载）

**1.修改Nginx配置文件并启动服务**

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

location / {

root html;

index index.php index.html index.htm;

#设置默认首页为index.php，当用户在浏览器地址栏中只写域名或IP，不说访问什么页面时，服务器会把默认首页index.php返回给用户

}

location ~ \.php$ {

root html;

fastcgi\_pass 127.0.0.1:9000; #将请求转发给本机9000端口，PHP解释器

fastcgi\_index index.php;

#fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

include fastcgi.conf;

}

[root@proxy ~]# /usr/local/nginx/sbin/nginx -s reload

#请先确保nginx是启动状态才可以执行命令成功，否则报错,报错信息如下：

#[error] open() "/usr/local/nginx/logs/nginx.pid" failed (2: No such file or directory)

**php-fpm配置文件**

[root@proxy etc]# vim /etc/php-fpm.d/www.conf

[www]

listen = 127.0.0.1:9000 //PHP端口号

pm.max\_children = 32 //最大进程数量

pm.start\_servers = 15 //最小进程数量

pm.min\_spare\_servers = 5 //最少需要几个空闲着的进程

pm.max\_spare\_servers = 32 //最多允许几个进程处于空闲状态

**LNMP常见问题**

Nginx的默认访问日志文件为/usr/local/nginx/logs/access.log

Nginx的默认错误日志文件为/usr/local/nginx/logs/error.log

PHP默认错误日志文件为/var/log/php-fpm/www-error.log

如果动态网站访问失败，可用参考错误日志，查找错误信息。

**DNS：**

万维网上作为域名和IP地址相互映射的一个分布式数据库，能够使用户更方便的访问互联网，而不用去记住能够被机器直接读取的IP数串。通过域名，最终得到该域名对应的IP地址的过程叫做域名解析（或主机名解析）。DNS协议运行在UDP协议之上，使用端口号53。

DNS服务器的功能

– 正向解析:根据注册的域名查找其对应的IP地址

– 反向解析:根据IP地址查找对应的注册域名,不常用

**.**  根域

一级域名： .cn .us .hk .tw .kr .com .net

二级域名: .com.cn .net.cn .org.cn .edu.cn ......

三级域名: nb.com.cn dawai.com.cn haha.com.cn haxi.com.cn

完整的主机名： www.nb.com.cn ftp.nb.com.cn tts.nb.com.cn

• Full Qualified Domain Name,完全合格主机名(FQDN)

– 系统服务: named

– 默认端口: 53

– 运行时的虚拟根环境:/var/named/chroot/

• 主配置文件:/etc/named.conf #设置本机负责解析的域名 tedu.cn

• 地址库文件:/var/named/ # 主机名与IP地址的对应记录

DNS解析过程：

递归解析：指DNS服务器与其他DNS服务器交互，最终将解析结果带回来的过程

迭代解析：指DNS服务器与其他DNS服务器交互，最终将告知下一个DNS服务

验证迭代查询

• dig命令,更专业的DNS测试工具

**缓存DNS**

作用：缓存解析结果，提高解析速度

搭建方式：

1.全局转发，所有的DNS解析请求全部转发给公网DNS

2.根域迭代，所有的DNS解析请求全部发给根域DNS服务器

真机上实现缓存DNS服务器

1.搭建Yum仓库

# mkdir /dvd

# mount /iso/CentOS-7-x86\_64-DVD-1708.iso /dvd

# ls /dvd

# rm -rf /etc/yum.repos.d/\*

# vim /etc/yum.repos.d/dvd.repo

[dvd]

name=dvd

baseurl=file:///dvd

enabled=1

gpgcheck=0

[root@room9pc01 ~]# yum -y install bind bind-chroot

2.确认真机的DNS服务器

[root@room9pc01 ~]# cat /etc/resolv.conf

nameserver 172.40.1.10

3.真机修改配置文件/etc/named.conf

options {

directory "/var/named";

forwarders { 172.40.1.10; }; #转发给172.40.1.10

};

[root@room9pc01 ~]# systemctl restart named

4.在虚拟机上解析

[root@svr7 /]# nslookup www.qq.com 192.168.4.254

[root@svr7 /]# nslookup www.360.com 192.168.4.254