day01

192.168.4.5 :nginx服务器

web服务器: (httpd\nginx\lighttpd

tomcat\weblogic\websphere\jboss)

nginx(taobao)-->tengine

F12 server: taobao (tengine)

./configure --help // 帮助 with 安装 without 不安装

netstat命令可以查看系统中启动的端口信息，该命令常用选项如下：

-a显示所有端口的信息

-n以数字格式显示端口号

-t显示TCP连接的端口

-u显示UDP连接的端口

-l显示服务正在监听的端口信息，如httpd启动后，会一直监听80端口

-p显示监听端口的服务名称是什么（也就是程序名称）

相对的是nginx 的安装目录(/usr/local/nginx/)

index index.html index.htm; //默认首页,默认页面

1)源码安装nginx

1.安装依赖包 gcc pcre-devel openssl-devel

2.安全 创建用户

3.解包 cd

4.configure --user --group --prefix --with-名称

make make install

5.升级 手动拷贝nginx ,重启

2)默认有一个网站(允许所有人访问)

用户认证

vim nginx.conf

server {

auth\_basic "字符";

auth\_basic\_user\_file "文件";

}

yum -y install httpd-tools

htpasswd -c 文件 用户名

>密码

nginx -s reload //重启

3)虚拟主机(做多个网站)域名

vim nginx.conf

server {

charset utf-8; //使网站支持中文

server\_name www.a.com:

}

server {

server\_name www.b.com

}

#mkdir /usr/local/nginx/www

#echo 内容 > 主页

#yum -y install \*fonts\*

4)httpd加密网站

对称 123 123

非对称 123 abc

//机制问题网站不能使用对称加密

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

day02

LNMP(linux+nginx+mysql+php), LAMP

静态

动态

web:/var/www/html

/usr/local/nginx/html

yum安装软件

nginx

mariadb maria-server maria-devel

php php-fpm php-mysql

1）启动MySQL服务

[root@proxy ~]# systemctl start mariadb //启动服务器

[root@proxy ~]# systemctl status mariadb //查看服务状态

[root@proxy ~]# systemctl enable mariadb //设置开机启动

2）启动PHP-FPM服务

[root@proxy ~]# systemctl start php-fpm //启动服务

[root@proxy ~]# systemctl status php-fpm //查看服务状态

[root@proxy ~]# systemctl enable php-fpm //设置开机启动

实现动静分离(nginx.conf),做判断

如果用户访问的静态页面,则直接返回

如果用户访问的是php脚本,则把脚本转发给9000端口

location语句是匹配用户的地址栏(从域名后面开始)

location 支持正则

location ~ /abc {

}

php是多进程软件

################################

程序 (program)

-进程(process)

-线程(thread)

-线程

-进程

-线程

################################

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

使用tailf (与tail区别,动态查看)

创建新的动态页面:

[root@proxy conf]# cd /root/lnmp\_soft/php\_scripts/

[root@proxy php\_scripts]# cp mysql.php /usr/local/nginx/html/

[root@client ~]# firefox www.a.com/mysql.php

查看变化:

mysql创建用户

grant all on \*.\* to 'dc'@'1.1.1.1' identified by '123';

####################################################

地址重写:

rewrite 旧地址 新地址 [选项]

rewrite 旧地址 新地址 [选项];

last 不再读其他rewrite

break 不再读其他语句，结束请求

redirect 临时重定向

permament 永久重定向

配置默认安装,没安装的话要按升级步骤重新启用

访问192.168.4.5/下面子页面，重定向至www.tmooc.cn/下相同的页面

rewrite /(.\*) www.tmooc.cn/$1; //$1代表前面括号里的值

1.安装软件 7个

2.启动服务 nginx,php-fpm,mariadb

3.修改配置 location ~ \.php${ }

4.nginx -s reload

5.firefox http://www.a.com/test.php

day03

nginx web服务器

代理服务器

功能:

1.调度(轮询)

2.健康检查

upstream abc{

server 192.168.2.100 weight=2;

server 192.168.2.200 max\_fails= fail\_timeout=;

}

server{

proxy\_pass http://abc;

proxy\_pass abc;

}

#####################################################

修改默认404页面:主配置文件下的error..404(去#号)

error\_page 404 /404.html;

error\_page 402 /402.html;

error\_page 401 /401.html;

error\_page 401 402 403 404 405 /a.html;

#####################################################

web代理服务器/web调度器(http协议)

默认不安装

./configure --with-stream

nginx服务器:

关心的问题:

谁在访问你

访问量多少

并发量多少

PV是多少(page view 页面被点击多少次)

UV是多少(user view 有多少用户点击)

有没有等待用户,是多少

统计信息模块:

--with-http\_stub\_status\_module

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

… …

location /status {

stub\_status on;

#allow IP地址;

#deny IP地址;

}

… …

[root@proxy ~]# nginx

ss命令可以查看系统中启动的端口信息，该命令常用选项如下：

-a显示所有端口的信息

-n以数字格式显示端口号

-t显示TCP连接的端口

-u显示UDP连接的端口

-l显示服务正在监听的端口信息，如httpd启动后，会一直监听80端口

-p显示监听端口的服务名称是什么（也就是程序名称）

########################################################

tcp/udp:

3次握手,4次断开

http 一次连接,一次请求

keepalive 一次连接多次请求

#############################################

PV是多少(page view 页面被点击多少次)

UV是多少(user view 有多少用户点击)

/usr/local/nginx/logs/access.log

awk '{IP[$1]++} END{for (i in IP){print IP[i],i}}' access.log

#############################################

客户端的电脑(客户端)--------------------服务器

缓存的功能

存什么,存多久

动态数据

静态数据(图片视频音频)

location ~\* \.(jpg|png) {

expires 30d; //如果访问图片,则缓存30天

}

#################################

kill 作用是给程序发送 信号

kill -l //查看

####################################################

cd /var/lib/php/session/ //查看服务器本地的Session信息

web页面更改背景颜色:

<body bgcolor=blue>

####################################################

PHP实现session共享

[root@web1 ~]# yum -y install php-pecl-memcache

[root@web1 ~]# vim /etc/php-fpm.d/www.conf //修改该配置文件的两个参数

//文件的最后2行

php\_value[session.save\_handler] = files

php\_value[session.save\_path] = /var/lib/php/session

//原始文件，默认定义Sessoin会话信息本地计算机（默认在/var/lib/php/session）

php\_value[session.save\_handler] = memcache

php\_value[session.save\_path] = "tcp://192.168.2.5:11211"

session,cookie

HTTP 无状态协议

web1

client--------proxy memcached(数据库)

web2

web1创建session (文件名=状态)

web1创建session (连接memcached(234df89=状态))

day04

day05

静态数据(图片视频音频文档)

动态数据(脚本)

PHP,java,python,perl,ruby,shell

java{跨平台的开发语言}

Windows{jdk.JVM java虚拟机}

linux{jdk,JVM,java虚拟机}

MacOS{jdk,JVM,java虚拟机}

电视盒子l{jdk,JVM,java虚拟机}

tomcat 就是用java写的代码

######################################################

web虚拟主机:

apache:

<viryualhost \*:80>

servername

documentroot

</virtualhost>

nginx:

server{

listen 80;

server\_name;

root html;

}

ls /dev/random

ln -s /dev/urandom /dev/random //tomcat启动需要大量随机字符

tomcat:

<engin ...defaulthost=www.a.com> //输ip默认打开页面

<host name=www.a.com appBase=a> //appBase=a(路径)

</Host> //与上面对应,首字母大写

<host name=www.b.com appBase=b> //appBase=b(路径)

<Context path="" docBase="base"/ > //指定网页文件路径

</Host> //与上面对应

</Eigine> //与上面对应

##########################################################

二:修改www.b.com网站的首页目录为base

1）使用docBase参数可以修改默认网站首页路径

[root@web1 ~]# vim /usr/local/tomcat/conf/server.xml

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

</Host>

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

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

</Host>

… …

[root@web1 ~]# mkdir /usr/local/tomcat/b/base

[root@web1 ~]# echo "BASE" > /usr/local/tomcat/b/base/index.html

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

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

##################################################

三:跳转

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

[root@web1 ~]# vim /usr/local/tomcat/conf/server.xml

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

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

</Host>

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

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

</Host>

[root@web1 ~]# echo "Test" > /var/www/html/index.html

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

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

测试查看页面是否正确

################################################

四:配置Tomcat支持SSL加密网站:

1）创建加密用的私钥和证书文件

[root@web1 ~]# keytool -genkeypair -alias tomcat -keyalg RSA -keystore /usr/local/tomcat/keystore //提示输入密码为:123456

//-genkeypair 生成密钥对

//-alias tomcat 密钥别名

//-keyalg RSA 定义密钥算法为RSA算法

//-keystore 定义密钥文件存储在:/usr/local/tomcat/keystore

2)再次修改server.xml配置文件，创建支持加密连接的Connector

###############################################################

五：配置Tomcat日志

vim /usr/local/tomcat/conf/server.xml

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

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

#从默认localhost虚拟主机中把Valve这段复制过来，适当修改下即可

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

prefix=" a\_access" suffix=".txt"

pattern="%h %l %u %t &quot;%r&quot; %s %b" />

</Host>

重启Tomcat服务器

nginx[代理加软件]

varnish代理软件

varnish[代理加软件] CDN(content delivery network) 内容分发网络

day06

版本控制软件

版本控制软件(共享仓库) a1 a2 a3 a4 [a.txt]

集中版本控制: SVN(subversion)

git init /仓库名称 --bare

分布式版本控制: git

##################################

全览:

web1: /var/git/project(仓库)

web2: git clone ip:/var/git/project

project(工作目录)

project/.git(本地仓库)

project/新建文件a.txt

git add . 建索引

git commit -m "注释" 把a.txt拷贝到.git(提交本地仓库)

project/新建文件a.txt(修改文件)()v2

git add . 建索引

git commit -m "注释" .git (a.txt-v1 a.txt-v2 a.txt-v3)

......

git push //提交到网络服务器

git pull //下载服务器的新版本

最常用四个命令:

git add .

git commit

git push

git pull

###########################################################

一：部署Git服务器

1）YUM安装Git软件。

[root@web1 ~]# yum -y install git

[root@web1 ~]# git --version //查看版本号

2)初始化一个空仓库。

[root@web1 ~]# mkdir /var/git //创建一个普通目录

[root@web1 ~]# git init /var/git/project --bare 初始化仓库

[root@web1 ~]# ls /var/git/ /查看

二:客户端

1) clone克隆服务器仓库到本地

[root@web2 ~]# yum -y install git

[root@web2 ~]# git clone root@192.168.2.100:/var/git/project //克隆服务器仓库

2) 修改git配置

[root@web2 project]# git config --global user.email "you@example.com" //邮箱

[root@web2 project]# git config --global user.name "Your Name" //用户名

[root@web2 project]# cat ~/.gitconfig //查看自己配置

3） 本地工作区对数据进行增删改查(必须要先进入仓库再操作数据)。

root@web2 project]# echo "init date" > init.txt //创建文档

[root@web2 project]# mkdir demo

[root@web2 project]# cp /etc/hosts demo

4） 查看仓库中数据的状态。

[root@web2 project]# git status

5） 将工作区的修改提交到暂存区。

[root@web2 project]# git add . //. 代表所有,可写文档名单加

6) 将暂存区修改提交到本地仓库。

[root@web2 project]# git commit -m "注释，可以为任意字符" //提交到仓库

[root@web2 project]# git status

查看日志方法:

git log //查看提交日志

git log --pretty=oneline //在一行显示提交日志

git log --oneline //查看版本码前几位

git reflog //查看版本码前几位

Windows需要安装git和tortoiseGit

#######################################################

Git分支操作

[root@web2 project]# git status //查看当前工作分支

[root@web2 project]# git branch -v //查看当前工作分支

[root@web2 project]# git branch hotfix //创建后提醒分支(git branch 分支名)

[root@web2 project]# git checkout hotfix //切换hotfix 分支

在hotfix修改代码

[root@web2 project]# git checkout master //切回master分支

[root@web2 project]# git merge hotfix //合并分支

[root@web2 project]# git push //提交到服务器

#####################################################################

git的几种服务器:###

SSH协议服务器（支持读写操作）

1）创建基于密码验证的SSH协议服务器（web1主机操作）。

[root@web1 ~]# git init --bare /var/git/base\_ssh

Initialized empty Git repository in /var/git/base\_ssh/

2)客户端访问的方式（web2主机操作）。

[root@web2 ~]# git clone root@192.168.2.100:/var/git/base\_ssh

3）客户端生成SSH密钥，实现免密码登陆git服务器（web2主机操作）。

设置免密登录:

ssh-keygen -f /root/.ssh/id\_rsa -N '' //生成秘钥,无交互

ssh-copy-id 192.168.2.100 //传递秘钥

[root@web2 ~]# git clone root@192.168.2.100:/var/git

Git协议服务器（只读操作的服务器）

1）安装git-daemon软件包（web1主机操作）。

[root@web1 ~]# yum -y install git-daemon

2）创建版本库（web1主机操作）。

[root@web1 ~]# git init --bare /var/git/base\_git

Initialized empty Git repository in /var/git/base\_git/

3）修改配置文件，启动git服务（web1主机操作）。

[root@web1 ~]# vim /usr/lib/systemd/system/git@.service

修改前内容如下：

ExecStart=-/usr/libexec/git-core/git-daemon --base-path=/var/lib/git

--export-all --user-path=public\_git --syslog --inetd –verbose

修改后内容如下：

ExecStart=-/usr/libexec/git-core/git-daemon --base-path=/var/git

--export-all --user-path=public\_git --syslog --inetd –verbose

[root@web1 ~]# systemctl start git.socket

4）客户端访问方式（web2主机操作）

[root@web2 ~]# git clone git://192.168.2.100/base\_git

步骤三：HTTP协议服务器（只读操作的服务器）

1）安装gitweb、httpd软件包（web1主机操作）。

[root@web1 ~]# yum -y install httpd gitweb

2）修改配置文件，设置仓库根目录（web1主机操作）。

[root@web1 ~]# vim +11 /etc/gitweb.conf

$projectroot = "/var/git"; #添加一行

3) 创建版本仓库（web1主机操作）

[root@web1 ~]# git init --bare /var/git/base\_http

4）启动httpd服务器

[root@web1 ~]# systemctl start httpd

5）客户端访问方式（web2主机操作）

注意：调用虚拟机中的firefox浏览器，需要在远程时使用ssh -X 服务器IP，并且确保真实主机的firefox已经关闭。

[root@web2 ~]# firefox http://192.168.2.100/git/

###############################################################################

制作nginx的RPM包

RPM是一个压缩包

把/usr/local/nginx 打包压缩nginx.rpm

rpm -ivh nginx.rpm

/usr/local.nginx

yum -y ins install nginx

步骤一：安装rpm-build软件

1)[root@web1 ~]# yum -y install rpm-build

2）生成rpmbuild目录结构

[root@web1 ~]# rpmbuild -ba nginx.spec //会报错，没有文件或目录

[root@web1 ~]# ls /root/rpmbuild //自动生成的目录结构

BUILD BUILDROOT RPMS SOURCES SPECS SRPMS

3）准备工作，将源码软件复制到SOURCES目录

[root@web1 ~]# cp nginx-1.12.2.tar.gz /root/rpmbuild/SOURCES/

4）创建并修改SPEC配置文件

[root@web1 ~]# vim /root/rpmbuild/SPECS/nginx.spec

Name:nginx #源码包软件名称

Version:1.12.2 #源码包软件的版本号

Release: 10 #制作的RPM包版本号

Summary: Nginx is a web server software. #RPM软件的概述

License:GPL #软件的协议

URL: www.test.com #网址

Source0:nginx-1.12.2.tar.gz #源码包文件的全称

#BuildRequires: #制作RPM时的依赖关系

#Requires: #安装RPM时的依赖关系

%description

nginx [engine x] is an HTTP and reverse proxy server. #软件的详细描述

%post

useradd nginx #非必需操作：安装后脚本(创建账户)

%prep

%setup -q #自动解压源码包，并cd进入目录

%build

./configure

make %{?\_smp\_mflags}

%install

make install DESTDIR=%{buildroot}

%files

%doc

/usr/local/nginx/\* #对哪些文件与目录打包

%changelog