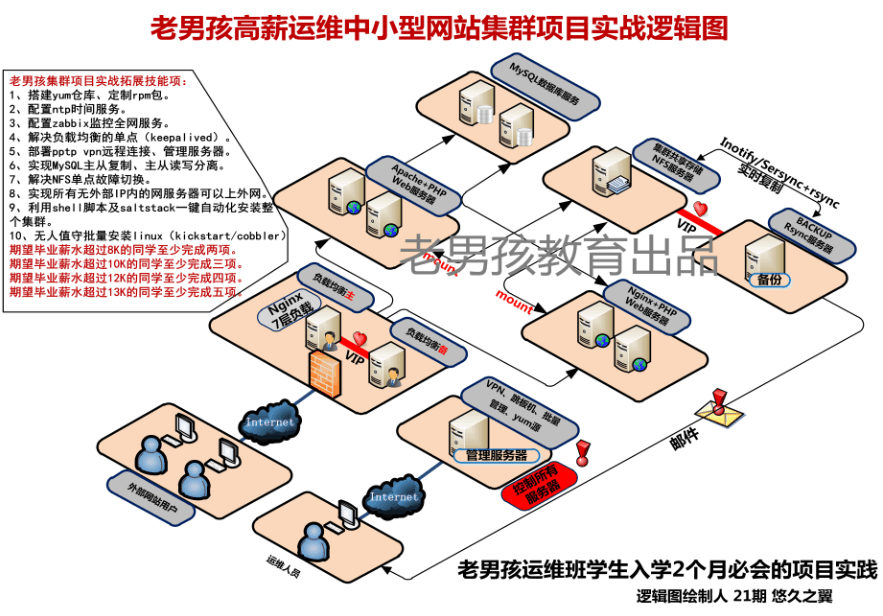
服务器架构

## 服务器架构图



1. lb01-lb02使用nginx + keepalived 实现高可用，
2. apache, nginx web 服务器的站点目录使用git 从x管理服务器推送过来，实现版本控制。web服务器上的uploads目录挂载到NFS服务器的/uploads上，
3. NFS服务器下的/uploads 使用inotify+rsync实时同步到backup服务器下的/uploads 。NFS 与 backup keepalived 实现高可用，当nfs服务器挂了，web服务的uploads 自动切换挂载到backup 服务器的/uploads下，
4. mysql 主备同步。 mysql 数据库文件每天凌晨1点自动备份到本地/mysql-backup然后使用rsync推送到backup服务器里的mysql-backup文件中
5. backup服务器每天早上8点检查备份包是否异常，并且发送邮件给运维人员

## 服务器主机名及IP规划

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 服务器 | 外网IP | 内网IP | 名称规划 | 域名规划 |
| A1-nginx 负载均衡服务器 | 10.0.0.72/24 | 172.16.1.72/24 | lb-master01 | lb01.etiantian.org |
| A2-nginx 负载均衡服务器 | 10.0.0.78/24 | 172.16.1.78/24 | lb-backup01 | lb02.etiantian.org |
| B1-apache软件web服务器 |  | 172.16.1.73/24 | apache01 | lamp01.etiantian.org |
| B2-nginx软件web服务器 |  | 172.16.1.74/24 | nginx01 | lnmp01.etiantian.org |
| C3-My Sql 数据库服务器 |  | 172.16.1.75/24 | mysql01 | db01.etiantian.org |
| C1-NFS 存储服务器 |  | 172.16.1.76/24 | nfs01 | nfs.etiantian.org |
| C2-rsync 数据库服务器 |  | 172.16.1.77/24 | backup01 | rsync.etiantian.org |
| X-管理服务器 | 10．0．0．80/24 | 172.16.1.80/24 | Xmanager | m.etiantian.org |
| yum 仓库 | 10．0．0．71/24 | 172.16.1.71/24 | yum | yum.etiantian.org |

**环境准备：**

**内核：2.6.32-573.el6.x86\_64**

**版本信息：CentOS release 6.7 (Final)**

## 目录规划

|  |  |  |
| --- | --- | --- |
| 服务器说明 | 目录结构 | 目录结构说明 |
| 初始化目录结构 | /application  /server/scripts  /home/oldboy/tools | 软件程序的安装目录  存放脚本的目录  存放编译软件源码目录 |
| Web服务器 | /data/www  /app/logs | web服务器网页站点目录包含www, bbs, blog目录  web服务器访问日志存放目录 |
| Rsync服务器 | /data/nfs  /data/uploads  /data/mysql  /data/apache  /data/nginx | nfs 服务器定时备份数据目录  nfs 服务器实时无差异同步数据目录  mysql 服务器定时备份数据目录  apache web服务器定时备份数据目录  nginx web服务器定时备份数据目录 |
| NFS 存储服务器 | /uploads  /nfs-backup | web服务器共享存储目录  nfs 服务器重要数据备份目录 |
| mysql 存储服务器 | /mysql-backup | mysql 服务器重要数据备份目录 |
| yum 仓库服务器 | /application/yum/centos6.7/x86\_64 | 用于存储 rpm包目录 |

# yum 仓库搭建

出于安全考虑架构中仅负载均衡有外网，其它服务器都仅有内网IP，所有需要安装软件的机子都需要从此服务器来下载安装，并且这样做也具有节省宽带，方便自己制作yum源，制作yun仓库十分简单，不过rpm包的收集算是一个难事，还好曾经安装的时候已经下载下来了。

yum 仓库的搭建：

$ mkdir -p /application/yum/centos6.7/x86\_64/

$ cd /application/yum/centos6.7/x86\_64/

$ yumdownloader zlib-devel

$ yum -y install createrepo

$ createrepo -pdo /application/yum/centos6.7/x86\_64/ /application/yum/centos6.7/x86\_64/

$ python -m SimpleHTTPServer 80 &>/dev/null &

$ echo "/usr/bin/python -m SimpleHTTPServer 80 &>/dev/null & " >>/etc/rc.local

之后如果如果增加了新的包，我们需要再次更新rpm包

$ createrepo --update /application/yum/centos6.7/x86\_64/

在其它服务器上配置

vi /etc/yum.repos.d/etiantian.repo

[etiantian]

name=Server

baseurl=http://yum.etiantian.org

enabled=1

gpgcheck=0

打包软件此次我们用fpm，该包是ruby下的一个模块,

yum install -y ruby ruby-devel rubygems

gem install fpm

这样所有内网服务器便可以使用本地yum仓库来安装软件了。

yum install nginx –y

vi /usr/local/nginx/conf/nginx.conf #编辑配置文件，在http {下面添加以下内容：

autoindex on; #开启nginx目录浏览功能

autoindex\_exact\_size off; #文件大小从KB开始显示

autoindex\_localtime on; #显示文件修改时间为服务器本地时间

:wq! #保存，退出

service nginx reload #重新加载配置

二、确定以上三个yum源上游源的同步镜像地址

1、CentOS官方标准源：rsync://mirrors.ustc.edu.cn/centos/

2、rpmforge源：rsync://mirrors.ispros.com.bd/repoforge/

3、epel源：rsync://mirrors.ustc.edu.cn/epel/

备注：上游yum源必须要支持rsync协议，否则不能使用rsync进行同步。

参考：

CentOS官方标准源：

rsync://mirrors.kernel.org/centos

rpmforge源：

http://apt.sw.be/

rsync://ftp-stud.fht-esslingen.de/dag

epel源：

http://mirrors.fedoraproject.org/publiclist/EPEL/

rsync://mirrors.kernel.org/fedora-epel

三、创建以上三个yum源同步脚本，并且设定脚本自动执行

mkdir -p /home/crontab #创建目录

vi /home/crontab/yum\_rsync.sh #添加以下代码

#!/bin/sh

/usr/bin/rsync -avrt rsync://mirrors.ustc.edu.cn/centos/ --exclude-from=/usr/local/nginx/html/exclude\_centos.list /usr/local/nginx/html/centos/

/usr/bin/rsync -avrt rsync://mirrors.ispros.com.bd/repoforge/ --exclude-from=/usr/local/nginx/html/exclude\_repoforge.list /usr/local/nginx/html/repoforge/

/usr/bin/rsync -avrt rsync://mirrors.ustc.edu.cn/epel/ --exclude-from=/usr/local/nginx/html/exclude\_epel.list /usr/local/nginx/html/epel/

# X 管理服务器

主要用于批量分发密钥，批量管理及维护服务器，使用git对web服务器的站点目录实行版本控制，下面便是用来自动生成密钥，并且自动发送公钥给其它服务器并且把backup.sh发送到要备份的服务器上，然后做好定时任务每天01点自动备份并且发送到backup服务器上，

## 主管理脚本

下面这个脚本用来批量分发sshkey,文件，以及分发安装脚本，远程执行脚本

[root@Xmanager scripts]# cat sendkey.sh

#!/bin/sh

. /etc/init.d/functions

#Msg函数用来控制输出信息

function Msg () {

if [ $? -eq 0 ];then

action "$1" /bin/true

else

action "$1" /bin/false

fi

}

#scpfile函数用来批量分发文件，下面的函数会调用此模块来分发文件到指定目录，并且也支持传参

function scpfile () {

for a in ${ip[@]}

do

scp -rp $1 172.16.1.$a:$2 >/dev/null

Msg "send $1 to ip $a $2"

done

}

# sshcommend函数用来远程登录并执行命令，同样其它函数会调用此函数，并且也支持传参

function sshcommend () {

for a in ${ip[@]}

do

/usr/bin/ssh 172.16.1.$a $\* >/dev/null

Msg "ssh to ip $a $\*"

done

}

# gekey用于生成sshkey，

function gekey () {

[ ! -d ~/.ssh ] && ssh-keygen -t rsa -P "" -f ~/.ssh/id\_rsa

Msg "generate key"

}

#sendkey用于发送公钥到所有指定IP的服务器，这里使用的是expect，我觉得这个expect脚本需要再优化下

#因为如果是第二次发送默认是不会输yes的，而且我又不想每次删除known\_host文件，这里以后再论吧

function sendkey () {

for a in ${ip[@]}

do

/usr/bin/expect /server/scripts/autoSendKey.exp ~/.ssh/id\_rsa.pub 172.16.1.${a} &>/dev/null

Msg "sendkey to 172.16.1.${a}"

done

}

#sendfile 用于分发指定文件到指定目录，有自己制作的yum源，以及host文件

function sendfile () {

sshcommend /bin/mkdir -p /server/scripts

scpfile /etc/hosts /etc/

scpfile "/server/scripts/installRpm.sh /server/scripts/backup.sh" /server/scripts/

scpfile "/etc/yum.repos.d/etiantian.repo /etc/yum.repos.d/CentOS-Base.repo" /etc/yum.repos.d/

}

#installRpm，用于对每一台服务器安装对应的rpm包，这里需要说明的是，每台服务器的hostname必须按上面的规划来修改

function installRpm () {

sshcommend "/bin/sh /server/scripts/installRpm.sh"

}

# webdeploy这个函数用于把站点目录发送过去，并且把相应的图片挂载到nfs服务器的uplods上

function webdeploy () {

scpfile /data/\* /data/

sshcommend "chown -R nginx.nginx /data"

sshcommend "mount -t nfs -o noatime,nodiratime,nosuid,noexec,nodev,rsize=131072,wsize=131072 172.16.1.76:/uploads /data/www/blog/wp-content/uploads/"

}

#main函数用来调用其它函数，这是

function main () {

gekey

sendkey

sendfile

installRpm

}

function helpp () {

echo "this shell include three function you cant user agument sendkey send file installRpm to run this function"

echo "the defalut ip is 72 73 74 75 76 77 78 you can use number parameter to custome ip, ip is effective in all function"

echo "function sendkey, send sshkey to remote server this function do not accepte any parameter"

echo "function scpfile , this function accept two parameter [sorc disc] will be send sorc file to remote disc like scp "

echo "function installRpm, install rpm to remote server, and remote server hostname must be setup "

echo "function sshcommend is ssh remote server to run command like ssh remote mkdir /data"

}

#默认IP是下面几个，当然也可以在执行脚本的时候传参进去，多少IP都行，只要用空格隔开，而且也可以单独执行上面的任何函数，自定义任何IP

ip=(77 76 75 73 74 72 78)

num=$(echo $@ |sed -r 's#[a-zA-Z/.]\*##g')

co=$(echo $@ |sed 's/[0-9]\*//g')

numc=$(echo $num|wc -w)

coc=$(echo $co|wc -w)

if [ $numc -gt 0 ];then

unset ip

ip=$num

fi

if [ $coc -gt 0 ];then

$co

else

main

fi

## expect脚本

#下面是expect脚本，借用的别人的，虽然可以优化，但也不想再玩啦。

[root@Xmanager scripts]# cat autoSendKey.exp

#!/usr/bin/expect

set file [lindex $argv 0]

set host [lindex $argv 1]

set password "123456"

spawn ssh-copy-id -i $file " $host"

expect {

"yes/no" {send "yes\r"; exp\_continue}

"\*password" {send "$password\r"}

}

expect eof

## 通用备份脚本

#backup 脚本主要用于对每个服务器定期对指定文件生成备份文件，并且使用rsync推送到backup服务器

[root@Xmanager scripts]# cat backup.sh

#!/bin/bash

. /etc/profile

IP=$(ifconfig eth0|awk -F '[: ]+' 'NR==2 {print $4}')

hostn=$(hostname|egrep -o '[a-z-]+')

Path="/data/$hostn/$IP"

time=$(date +%F)

##backup

[ ! -d $Path ] && mkdir $Path -p

cd / && \

tar zcf $Path/${hostn}\_backup\_$(date +%F).tar.gz etc/rc.local var/spool/cron/root server/scripts app/logs && \

find $Path/ -type f -name "\*$(date +%F).tar.gz" |xargs md5sum >$Path/flag\_$(date +%F).txt

## backup to rsync server

rsync -az /data/$hostn/ rsync\_backup@172.16.1.77::$hostn --password-file=/etc/rsync.password

##local delete before 3days age

find $Path/ -type f -name "\*.tar.gz" -mtime +3 |xargs rm –f

## 安装rpm包

#下面是自动化安装rpm包

[root@Xmanager scripts]# cat installRpm.sh

#!/bin/bash

. /etc/profile

. /etc/init.d/functions

hostn=$(/bin/hostname|egrep -o "[a-z-]+")

case "$hostn" in

lb-master)

yum install nginx-lb-master -y >/etc/null

;;

lb-backup)

yum install nginx-lb-backup -y >/etc/null

;;

apache)

yum install lamp -y >/etc/null

/etc/init.d/rpcbind start >/etc/null

mkdir -p /var/spool/cron/

echo "00 3 \* \* \* /bin/sh /server/scritpt/backup.sh &>/dev/null" >>/var/spool/cron/root

/bin/sh /server/scripts/backup.sh

;;

nginx)

yum install lnmp -y >/etc/null

/etc/init.d/rpcbind start >/etc/null

mkdir -p /var/spool/cron/

echo "00 3 \* \* \* /bin/sh /server/scritpt/backup.sh &>/dev/null" >>/var/spool/cron/root

/bin/sh /server/scripts/backup.sh

;;

nfs)

yum install nfs01 -y >/etc/null

;;

rsync)

yum install backup -y >/etc/null

;;

\*)

exit

esac

## 跳板机

#然后是跳板机的代码，主要用于远程登录到其它服务器

[root@Xmanager scripts]# cat tiaoban.sh

#!/bin/bash

function trapper() {

trap ':' INT EXIT TSTP TERM HUP

}

while true

do

trapper

clear

cat <<menu

72) lb01.etiantian.org

78) lb02.etiantian.org

73) lamp01.etiantian.org

74) lnmp01.etiantian.org

75) db01.etiantian.org

76) nfs.etiantian.org

77) rsync.etiantian.org

q) exit

menu

read -p "please select:" num

case "$num" in

72) ssh lb01.etiantian.org

;;

78) ssh lb02.etiantian.org

;;

73) ssh lamp01.etiantian.org

;;

74) ssh lnmp01.etiantian.org

;;

75) ssh db01.etiantian.org

;;

76) ssh nfs.etiantian.org

;;

77) ssh rsync.etiantian.org

;;

key) sh /server/scripts/sendkey.sh sendkey

;;

q) exit

esac

done

# backup 服务器

backup用于备份web服务器，NFS服务器，MYSQL服务器定时备份，并且早上检查并发邮件，并且nfs服务器实时同步。与nfs服务器使用keepalived实现高可用，

**安装软件：rsync-3.0.6-12.el6.x86\_64**

**软件介绍**

rsync, (remote synchronize) 是一个实现同步功能的软件，它在同步文件的同时，可以保持原来文件的权限，时间，软硬链接等附加信息。 rsync是用rsync 算法提供了一个客户机和远程文件服务器的文件同步的快速方法，而且可以通过ssh方式来传输文件，这样保密也非常好。主要用于远程数据备份与同步。 可以实现全备及增量备份，也可以本地数据同步。 在Centos5中rsync的版本是2.X，是把所有文件比对一遍后进行同步。在CentOs6中rsync的版本是3.0，是一边比对差异一边进行同步。而这里我们用的是3.0

rsync有三种模式，分别是本地，ssh远程同步，daemon模式

* Local: rsync [OPTION...] SRC... [DEST]
* Access via remote shell:
  + Pull: rsync [OPTION...] [USER@]HOST:SRC... [DEST]
  + Push: rsync [OPTION...] SRC... [USER@]HOST:DEST
* Access via rsync daemon:
* Pull:  rsync [OPTION...] [USER@]HOST::SRC... [DEST]
* rsync [OPTION...] rsync://[USER@]HOST[:PORT]/SRC... [DEST]
* Push: rsync [OPTION...] SRC... [USER@]HOST::DEST
* rsync [OPTION...] SRC... rsync://[USER@]HOST[:PORT]/DEST

rsync 参数说明：

* –delete; delete extraneous files from dest dirs 删除目标文件中多余的文件
* –exclude=PATTERN ; exclude files matching PATTERN 排除特定文件 –exclude={a,b}
* –exclude-from=FILE; read exclude patterns from FILE ,–exclude-from=/tmp/exclude.txt
* -z, –compress ; compress file data during the transfer
* -v, –verbose increase verbosity
* -a, –archive archive mode; equals -rlptgoD (no -H,-A,-X)
* –bwlimit=KBPS limit I/O bandwidth; KBytes per second
* –partial keep partially transferred files 断点续传

具体用法可以参见<http://gsandow.com/inotify_rsync/>。这里主要用的是deamon模式在其它服务器来推送数据到backup。

## 配置rsync

增加适用于rsync的用户

$ useradd rsync -M -s /sbin/nologin

$ mkdir /app/logs/ -p

$ mkdir /data/uploads /data/mysql /data/nfs /data/nginx /data/apache -p

$ chown -R rsync.rsync /data/

$ echo "rsync\_backup:sandow" >>/etc/rsync.password

$ chmod 600 /etc/rsync.password

[root@rsync01 scripts]# cat /etc/rsyncd.conf

###rsync server

uid = rsync

gid = rsync

use chroot = no

max connections = 200

timeout = 300

strict modes=yes

port=873

pid file = /var/run/rstbcd.pid

lock file = /var/run/rsync.lock

log file = /app/logs/rsyncd.log

hosts allow = 172.16.1.0/24

hosts deny = 0.0.0.0/32

secrets file = /etc/rsync.password

auth users = rsync\_backup

read only = false

ignore errors

list = false

[backup]

path = /backup

[uploads]

path = /data/uploads

[mysql]

path = /data/mysql

[nfs]

path = /data/nfs

[nginx]

path = /data/nginx

[apache]

path = /data/apache

每天晚上从其它服务器推送过来的数据都需要执行md5校验，是否打包正常然后发邮件告诉运维人员

[root@backup scripts]# cat check\_backup.sh

#!/bin/bash

. /etc/profile

flag\_num=$(find /data/ -type f -name "flag\_$(date +%F).txt"|wc -l)

if [ $flag\_num -eq 4 ];then

echo "$(date +%U%T) total number of package is correct" >/opt/mail\_$(date +%F)

else

echo "$(date +%U%T) there missing some packege" >/opt/mail\_$(date +%F)

fi

echo ================================================ >>/opt/mail\_$(date +%F)

failed=$(find /data/ -type f -name "flag\_$(date +%F).txt"|xargs md5sum -c|grep -v OK)

count=$(find /data/ -type f -name "flag\_$(date +%F).txt"|xargs md5sum -c|grep -v OK|wc -l)

if [ $count -eq 0 ];then

echo "$(date +%U%T) 备份服务器上的备份全部正常" >>/opt/mail\_$(date +%F)

else

echo "$(date +%U%T) 备份服务器上的备份有异常" >>/opt/mail\_$(date +%F)

echo " $failed" >>/opt/mail\_$(date +%F)

fi

mail -s "data info $(date +%F\ %T)" 921070747@qq.com </opt/mail\_$(date +%F)

####delete

find /data/ -type f -name "\*.tar.bz2" -mtime +180|xargs rm -f

## 实时同步

上面提到要nfs服务器与backup服务器实时同步，在nfs上可以用多种方法来实时同步 inotify + rsync, sersync，lsync, drbd。 下面说下前三个的区别

1. **lsyncd**是luo语言封装了inotify和rsync工具，采用了linux2.6.13里的inotify触发机制，然后通过rsync去差异同步，达到实时的效果，它完美解决了 inotify + rsync 海量文件同步带来的文件频繁发送文件列表的问题—通过时间延迟或累计触发事件次数实现，另外，配置简单，lua本身就是一种配置语言，可读性强，而且lsyncd的工作模式可以选择本地目录cp,本地目录rsync， 远程目录rsyncssh。下面是 inotify + rsync 与 sersync 的区别
2. **inotify + rsync**，随着文件数量的增大到100W+，目录下的文件列表就达20M，在网络状况不佳或者限速的情况下，变更的文件可能10来个才几M，却因此要发送的文件列表就达20M，严重减低的带宽的使用效率以及同步效率；更为要紧的是，加入inotifywait在5s内监控到10个小文件发生变化，便会触发10个rsync同步操作，结果就是真正需要传输的才2-3M的文件，比对的文件列表就达200M。使用这两个组合的好处在于，它们都是最基本的软件，可以通过不同选项做到很精确的控制，比如排除同步的目录，同步多个模块或同步到多个主机。
3. **sersync** 这么个工具可以提高同步的性能，也解决了同步大文件时出现异常的问题。sersync是国内的一个开发者开源出来的，使用c++编写，采用多线程的方式进行同步，失败后还有重传机制，对临时文件过滤，自带crontab定时同步功能。性能不错，seync只是减少了监听事件，源码中都是在拼接rsync命令，后面的目的地址始终是针对module，只要执行rsync命令，就会对整个目遍历，发送要比对的文件列表，然后再发送变化的文件。   
   * 国产开源，文档不是很全，而且2011年后再没更新
   * 采用xml配置文件方式，可读性比较好，但有些原生功能没有实现就没法使用了
   * 无法实现多目录同步，只能通过多个配置文件启动多个进程
   * 文件排除功能太弱
   * 插件功能虽有，不过有些鸡肋

此次在nfs上使用的软件为lsyncd,安装方法与详细参数见<http://gsandow.com/lsnycd/>在nfs服务器上需要配置的文件只有一个，然后配置好之后启动便好。下面是配置文件的详细内容

[root@buff buffer]# cat /etc/lsyncd.conf

settings {

logfile ="/app/logs/lsyncd.log",

statusFile ="/app/logs/lsyncd.status",

inotifyMode = "CloseWrite",

maxProcesses = 7,

-- nodaemon =flase,

}

sync {

default.rsync,

source = "/data/uploads",

target = "rsync\_backup@172.16.1.77::nfs",

delete="running",

exclude = { ".tmp", ".swap" },

-- excludeFrom = "/server/scripts/rsync\_exclude.lst",

delay = 30,

rsync = {

password\_file = "/etc/rsync.password",

archive = true,

compress = true,

verbose = true,

\_extra = {"--bwlimit=200"}

}

}

然后在nfs服务器上启动lsync便可以达到实时同步的目的

[root@buff buffer]# /etc/init.d/lsyncd start

Starting lsyncd: [ OK ]

目录规划

|  |  |  |
| --- | --- | --- |
| 目录 | 说明 | 备注 |
| /data/nginx | nginx web服务器备份数据 | /data/目录属主权限必须为rsync |
| /data/apache | apache web 服务器备份数据 |  |
| /data/mysql | mysql 服务器备份数据 |  |
| /data/nfs | nfs 服务器备份数据 |  |
| /data/uploads | nfs 服务器实时同步目录 |  |

## 制作yum包

制作成yum包的主要目的就是为了方便新机子上一键安装

fpm -s dir -t rpm -n backup -v 1.0 --description 'rsync' --post-install /server/scripts/rsync.sh /server/scripts/ /app/ /etc/rsync\*

[root@rsync01 scripts]# cat rsync.sh

#!/bin/bash

/usr/sbin/useradd rsync -M -s /sbin/nologin

/bin/mkdir /app/logs/ -p

mkdir /data/uploads /data/mysql /data/nfs /data/nginx /data/apache -p

chown -R rsync.rsync /data/

rsync --daemon

mkdir -p /var/spool/cron/

echo "00 08 \* \* \* /bin/sh /server/scripts/check\_backup.sh >/dev/null 2>&1" >

>/var/spool/cron/root

echo "rsync --daemon" >>/etc/rc.local

echo "set from=921070747@qq.com smtp=smtp.qq.com smtp-auth-user=921070747 smtp-auth-password=adminqwe smtp-auth=login" >>/etc/mail.rc

# nfs 服务器

network file sysctem 是网络文件系统，它的主要功能是通过网络让i同的主机系统之间可以共享文件或目录，NFS客户端一般为应用服务器，RPC (remote procedure call) 是能使客户端执行其他系统中程序的一种机制。NFS本身是没有提供信息传输的协议和功能的，但NFS却能让我们通过网络进行资料的分享，这是因为NFS使用了一些其它的传输协议。而这些传输协议用到这个RPC功能的。。所以只要用到NFS的地方都要启动RPC服务，不论是NFS SERVER或者NFS CLIENT。这样SERVER和CLIENT才能通过RPC来实现PROGRAM PORT的对应。可以这么理解RPC和NFS的关系：NFS是一个文件系统，而RPC是负责负责信息的传输。在大企业可以用 MFS, FASTDFS, GLUSTERFS, CEPH。

在些次架构中所有的web服务器为NFS客户端，NFS在服务端需要先启动rpcbind然后再启动nfs。在客户端只启rpcbind.

yum install nfs-utils rpcbind inotify-tools -y

需要安装些软件的服务器有web服务器nfs服务器

$ cat /etc/exports

/data 172.16.1.1/24(rw,rsync,all\_squash,anonuid=888,anongid=888)

在每台web服务器上把/data/www/blog/wp-content/uploads 挂载到 /uploads 上面

mount -t nfs -o noatime,nodiratime,nosuid,noexec,nodev,rsize=131072,wsize=131072 172.16.1.31:/uploads /data/www/blog/wp-content/uploads

## inotify+rsync实时同步

inotify 是一种强大的细粒度的，异步的文件系统事件监控机制， linux内核从2.6.13起，加入了inotify支持。其本身没有推送的功能，只 有监控文件系统的任务. 我们可以用inotify + rsync 做到实时复制

以下是使用inotify-tools实时同步脚本，上面的区别已经看到，而且效率很低所以最后还是换成lsyncd同步大文件，多文件都OK

[root@nfs01 scripts]# cat Iwait.sh

#!/bin/bash

inotify\_fun(){

hostn=$(hostname|egrep -o "[a-z-]+")

IEXCLUDE='(.\*/\*\.log|.\*/\*\.swp)$|^/tmp/src/mail/(2014|20.\*/.\*che.\*)'

/usr/bin/inotifywait -mrq --format '%w %f' --exclude ${IEXCLUDE} -e modify,delete,create,move,attrib /uploads | \

while read line

do

[ ! -e "$line" ] && \

/usr/bin/rsync -az --delete --bwlimit=200 /uploads/ rsync\_backup@172.16.1.77::uploads --password-file=/etc/rsync.password && continue

/usr/bin/rsync -auvrtzopgP --delete --progress --bwlimit=200 --password-file=/etc/rsync.password /uploads/ rsync\_backup@172.16.1.77::uploads

done

}

function main () {

log\_file=/app/logs/rsync\_client.log

inotify\_fun >>${log\_file} 2>&1 &

}

main

然后打包成rpm包

fpm -s dir -t rpm -n nfs01 -v 1.0 --description 'nfs' -d 'inotify-tools,nfs-utils,rpcbind' --post-install /server/scripts/nfs.sh /server/scripts/ /app/ /var/spool/cron/root /uploads/ /etc/rsync.password

[root@nfs01 scripts]# cat nfs.sh

#!/bin/bash

/bin/mkdir /app/logs/ /uploads/ -p

echo "/uploads 172.16.1.0/24(rw,sync,all\_squash)" >/etc/exports

chown -R nfsnobody.nfsnobody /uploads

/etc/init.d/rpcbind start

/etc/init.d/nfs start

/bin/sh /server/scripts/Iwait.sh

/bin/sh /server/scripts/backup.sh

cat >>/etc/rc.local<<EOF

/etc/init.d/rpcbind start

/etc/init.d/nfs start

EOF

# nginx负载均衡

架设服务器均衡方式有多种，Nginx, LVS, HAProxy + Keepalived , 是目前最广泛使用的三种方案。下面说说三者的比较

* **Nginx**做负载均衡， 由于Nginx工作在第7层，所以其效率不及LVS，但对于日PV小于1000万，nginx是完全够用的。而且它的与正则规则比haproxy更为强大和灵活，对网络稳定性的依赖非常小，只要能ping通理论上就可以访问页面，安装配置简单，可以通过端口检测到服务器内部的故障，但不支持url来检测，而且nginx还可以作为中层反向代理使用，可以作为作为静态见面和图片服务器。
* **LVS**，抗负载能力强，是工作在第四层仅作分发之用，没有流量的产生，这也使得在负载软件里性能最强，对内存和CPU资源消耗也比较低，但是其不支持正则表达式处理，不能动静分离，而且配置及维护过程复杂。工作稳定，自身有完整的双机热备方案。
* **HAProxy** 支持虚拟主机，而且补充了nginx的一些缺点： 支持session的保持，cookie的引导，同时支持通过获取指定的url来检测后端服务器的状态。其本身仅是一个负载均衡软件，并发优于Nginx,低于LVS.其负载策略多

在本次的构架中，我们选择使用Nginx + keepalived, 实现双机高可用负载均衡。Nginx 采用nginx-1.6.3，这里我们使

## 脚本安装nginx

#!/bin/sh

. /etc/init.d/functions

Dir="/application"

SoftDir="/home/oldboy/tools"

function Msg () {

if [ $? -eq 0 ];then

action "$1" /bin/true

else

action "$1" /bin/false

exit 1

fi

}

function User () {

use=`grep "nginx" /etc/passwd|wc -l`

[ -d $SoftDir ] ||/bin/mkdir $SoftDir -p && \

if [ $use -ne 1 ];then

/usr/sbin/useradd -u 888 nginx -s /sbin/nologin -M &>/dev/null

fi

Msg "mkdir&user"

}

function installNginx () {

yum -y install pcre-devel openssl-devel >/dev/null && \

cd $SoftDir && \

/usr/bin/wget http://nginx.org/download/nginx-1.6.3.tar.gz && \

/bin/tar zxf nginx-1.6.3.tar.gz && \

cd nginx-1.6.3 && \

./configure --prefix=/application/nginx-1.6.3 \

--user=nginx \

--group=nginx \

--with-http\_ssl\_module \

--with-http\_stub\_status\_module &>/dev/null ||

make >/dev/null && make install >/dev/null && \

Msg "make and install"

}

function deployN () {

ln -s $Dir/nginx-1.6.3/ $Dir/nginx &>/dev/null && \

}

function startN (){

/bin/mkdir -p /app/logs && \

/bin/mkdir -p /data/www/www /data/www/blog /data/www/bbs && \

chown -R nginx.nginx /data

net=`netstat -lntup |grep 80|grep -v "grep" |wc -l`

[ "$net" -ne 0 ] && /usr/bin/killall `netstat -lntup |grep 80|awk -F '/' '{print $NF}'` &>/dev/null && $Dir/nginx/sbin/nginx || $Dir/nginx/sbin/nginx &>/dev/null

Msg "Nginx start"

}

function main () {

User

installNginx

deployN

startN

}

main

## 配置nginx

然后我们来配置nginx, 配置路径为/application/nginx-1.6.3/conf/nginx.conf

worker\_processes auto;

error\_log /app/logs/nginx\_error.log crit;

pid /application/nginx/logs/nginx.pid;

worker\_rlimit\_nofile 51200;

events {

use epoll;

worker\_connections 51200;

multi\_accept on;

}

http {

include mime.types;

default\_type application/octet-stream;

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 /app/logs/access.log main;

sendfile on;

keepalive\_timeout 65;

upstream web\_pools {

ip\_hash;

server 172.16.1.73:80 weight=5 max\_fails=10 fail\_timeout=10s;

server 172.16.1.74:80 weight=5;

#server 10.0.0.10:80 weight=5 backup;

}

server {

listen 80;

server\_name blog.etiantian.org;

location / {

root html;

index index.php index.html index.htm;

proxy\_pass http://web\_pools;

proxy\_headers\_hash\_max\_size 51200;

proxy\_headers\_hash\_bucket\_size 6400;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

include proxy.conf;

}

}

server {

listen 80;

server\_name www.etiantian.org;

location / {

root html;

index index.php index.html index.htm;

proxy\_pass http://web\_pools;

proxy\_headers\_hash\_max\_size 51200;

proxy\_headers\_hash\_bucket\_size 6400;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

include proxy.conf;

}

}

server {

listen 80;

server\_name bbs.etiantian.org;

location / {

root html;

index index.php index.html index.htm;

proxy\_pass http://web\_pools;

proxy\_headers\_hash\_max\_size 51200;

proxy\_headers\_hash\_bucket\_size 6400;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

include proxy.conf;

}

}

}

# cat proxy.conf

proxy\_redirect off;

proxy\_set\_header Host $host;

proxy\_set\_header X-Forwarded-For $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

client\_max\_body\_size 10m;

client\_body\_buffer\_size 128k;

proxy\_connect\_timeout 90;

proxy\_send\_timeout 90;

proxy\_read\_timeout 90;

proxy\_buffer\_size 4k;

proxy\_buffers 4 32k;

proxy\_busy\_buffers\_size 64k;

proxy\_temp\_file\_write\_size 64k;

配置好后我们习惯性的先检查是否配置正确

[root@lb02 ~]# /application/nginx/sbin/nginx -t

nginx: the configuration file /application/nginx-1.6.3/conf/nginx.conf syntax is ok

nginx: configuration file /application/nginx-1.6.3/conf/nginx.conf test is successful

然后再平滑重启

$ /application/nginx/sbin/nginx -s reload

我们使用Keepalived,进行高可用(High Avaiability) 安装方法十分简单

yum install keepalived -y

## keepalived

## keepalived 介绍

Keepalived是一个基于VRRP协议来实现的服务高可用方案，可以利用其来避免IP单点故障，类似的工具还有heartbeat、corosync、pacemaker。但是它一般不会单独出现，而是与其它负载均衡技术（如lvs、haproxy、nginx）一起工作来达到集群的高可用。

### VRRP协议

VRRP全称 Virtual Router Redundancy Protocol，即 [虚拟路由冗余协议](http://en.wikipedia.org/wiki/VRRP)。可以认为它是实现路由器高可用的容错协议，即将N台提供相同功能的路由器组成一个路由器组(Router Group)，这个组里面有一个master和多个backup，但在外界看来就像一台一样，构成虚拟路由器，拥有一个虚拟IP（vip，也就是路由器所在局域网内其他机器的默认路由），占有这个IP的master实际负责ARP相应和转发IP数据包，组中的其它路由器作为备份的角色处于待命状态。master会发组播消息，当backup在超时时间内收不到vrrp包时就认为master宕掉了，这时就需要根据VRRP的优先级来选举一个backup当master，保证路由器的高可用。

在VRRP协议实现里，虚拟路由器使用 00-00-5E-00-01-XX 作为虚拟MAC地址，XX就是唯一的 VRID （Virtual Router IDentifier），这个地址同一时间只有一个物理路由器占用。在虚拟路由器里面的物理路由器组里面通过多播IP地址 224.0.0.18 来定时发送通告消息。每个Router都有一个 1-255 之间的优先级别，级别最高的（highest priority）将成为主控（master）路由器。通过降低master的优先权可以让处于backup状态的路由器抢占（pro-empt）主路由器的状态，两个backup优先级相同的IP地址较大者为master，接管虚拟IP。

+-------------------+

| uplink |

+-------------------+

|

+

MASTER keep|alived BACKUP

172.29.88.224 172.29.88.222 172.29.88.225

+-------------------+ +----------------+ +----------------+

| nginx01 |--------- | virtualIP |--- -----| nginx02 |

+-------------------+ +-----------------+ +-----------------+

|

+-----------------------------+-----------------------------+

| | |

+--------------------+ +---------------------+ +------------------+

| web01 | | web02 | | web03 |

+--------------------+ +---------------------+ +--------------------+

check\_nginx 用于监控nginx进程，如果nginx结束会试着重启，如果启动不成功就把keepalived进程也结束掉，这样做的好处便是，如果这台机子的nginx挂了并且又启动不了，vip就会漂移到keepalived的从服务器上

# cat /server/scripts/check\_nginx.sh

#!/bin/bash

counter=$(ps -C nginx --no-heading|wc -l)

if [ "${counter}" = "0" ]; then

/application/nginx/sbin/nginx

sleep 2

counter=$(ps -C nginx --no-heading|wc -l)

if [ "${counter}" = "0" ]; then

/etc/init.d/keepalived stop

fi

fi

cat /etc/keepalived/keepalived.conf

global\_defs {

notification\_email {

921070747@qq.com

}

notification\_email\_from Alexandre.Cassen@firewall.loc

smtp\_server 192.168.200.1

smtp\_connect\_timeout 30

router\_id LVS\_01

}

vrrp\_script chk\_nginx {

# script "killall -0 nginx"

script "/server/scripts/check\_nginx.sh"

interval 2

weight -5

fall 3

rise 2

}

vrrp\_instance VI\_1 {

state MASTER

interface eth0

mcast\_src\_ip 10.0.0.72

virtual\_router\_id 51

priority 150

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

10.0.0.3/24

}

track\_script {

chk\_nginx

}

}

keepalived从服务器仅需要把 state MASTER 改成 stat BACKUP 把mcast\_src\_ip 10.0.0.72 改成 mcast\_src\_ip 10.0.0.78 把priority 150 改成 priority 100 具体详细说明请参见<http://gsandow.com/keepalived/>

然后打包做成yum源

fpm -s dir -t rpm -n nginx-lb-master -v 1.1 --description 'nginx-keepalived' -d 'pcre,pcre-devel,keepalived,openssl,openssl-devel' --post-install /server/scripts/nginx-lb-master.sh /server/scripts/ /application/

[root@lb01 scripts]# cat nginx-lb-master.sh

#!/bin/bash

/usr/sbin/useradd -u 888 nginx -M -s /sbin/nologin

/bin/mkdir /app/logs/ -p

\cp /server/scripts/keepalived.conf /etc/keepalived/

/application/nginx/sbin/nginx

/etc/init.d/keepalived start

chkconfig keepalived on

cat >>/etc/rc.local<<EOF

#nginx+php-fpm by oldboy at 2011

/application/nginx/sbin/nginx

EOF

配置备负载均衡仅需要把上面三个地方改动下

state MASTER

mcast\_src\_ip 10.0.0.72

priority 150

其它配置与主服务器一样

然后我们打包成rpm包

fpm -s dir -t rpm -n nginx-lb-backup -v 1.1 --description 'nginx-keepalived' -d 'pcre,pcre-devel,keepalived,openssl,openssl-devel' --post-install /server/scripts/nginx-lb-backup.sh /server/scripts/ /application/

# nginx web服务器

nginx后台进程一个master进程，多个worker进程。master进程主要用来管理worker进程，包含接收来自外界的信号，然后向各worker进程发信号，并监控worker的运行状态。worker的数量可以有多个，一般会设置与机器CPU核数一致。如果reload，执行命令时我们启动一个新的nginx进程，新的nginx进程在解析到reload参数后，就知道我们的目的是控制nginx重新加载配置文件，它会向master进程发送信号，然后master重新加载配置文件，然后再启动新的worker进程，并向所有老的worker进程发送信号，告诉他们可以光荣退休了。新worker开始接收新请求，老的worker收到master的信号后不再接收新的请求，并且当进程中所有未处理完的请求处理完成后，再退出。 nginx采用了异步非阻塞的方式来处理请求，也就是说nginx可以现时处理成千上万个请求。 apache 也有异步非阻塞，每个请求会独占一个工作线程，当并发数上到几升时，就同时有几升个线和在处理请求，对操作系统来说，是不小的挑战，线程带来的内存占用非常大，纯种的上下文切换带来的cpu开销很大，性能就上不去。

nginx 三大功能 web服务（邮件服务） 负载均衡（反向代理proxy)，缓存服务器（相当于squid，cnd主要用于SQUID)

### WEB服务

配合动态PHP服务实现

1. 提供静态服务（图片，视频）另一个是lighttpd,并发，几万并发，
2. 提供动态服务 nginx+ fastcgi的方式运行php, jsp, 动态并发。nginx通过fastcgi客户端和fastcgi服务端(PHP)通途。 fastcgi(通用风头接口)
3. 反向代理（proxy）服务，或者称为负载均衡。 PV2000万以下，并发1万以下，都可以用Nginx做反向代理
4. 提供缓存服务，类似squid, varnish,

www 服务软件虚拟主机分类

1. 基于域名，就是通过域名 来区分不同的虚拟主机 (互联网的网站都会用这个)
2. 基于IP，（不用）
3. 基于端口 （内部网站，外部后台，安全考虑）

安装方法上面已经介绍过了现在来配置nginx,下面来查看配置文件

[root@nginx01 ~]# cat /application/nginx/conf/nginx.conf

worker\_processes 1;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

sendfile on;

access\_log /app/logs/access.log;

error\_log /app/logs/error.log;

keepalive\_timeout 65;

server {

listen 80;

server\_name www.etiantian.org;

root /data/www/www;

index index.php index.html index.htm;

location ~ .\*\.(php|php5)?$ {

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

include fastcgi.conf;

}

}

server {

listen 80;

server\_name bbs.etiantian.org;

root /data/www/bbs;

index index.php index.html index.htm;

location ~ .\*\.(php|php5)?$ {

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

include fastcgi.conf;

}

}

server {

listen 80;

server\_name blog.etiantian.org;

root /data/www/blog;

location / {

index index.php index.html index.htm;

}

location ~ .\*\.(php|php5)?$ {

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

include fastcgi.conf;

}

}

}

安装并配置php, 下面是自动化安装脚本

#!/bin/sh

. /etc/init.d/functions

Dir="/application"

SoftDir="/home/oldboy/tools"

function Msg () {

if [ $? -eq 0 ];then

action "$1" /bin/true

else

action "$1" /bin/false

exit 1

fi

}

function yumInstall () {

yum install zlib-devel libxml2-devel libjpeg-devel libiconv-devel -y >/dev/null && \

yum install freetype-devel libpng-devel gd-devel curl-devel libxslt-devel -y >/dev/null && \

yum install -y openssl-devel libtool libtool-ltdl-devel libmcrypt-devel mhash mhash-devel mcrypt libxslt-devel >/dev/null && \

wget -O /etc/yum.repos.d/epel.repo http://mirrors.aliyun.com/repo/epel-6.repo && \

/bin/mkdir $SoftDir -p && \

cd $SoftDir && \

/usr/bin/wget http://ftp.gnu.org/pub/gnu/libiconv/libiconv-1.14.tar.gz && \

tar zxf libiconv-1.14.tar.gz && \

cd libiconv-1.14 && \

./configure --prefix=/usr/local/libiconv >/dev/null && \

make >/dev/null && \

make install >/dev/null && \

Msg "yumInstall"

}

function installPhp () {

cd $SoftDir && \

tar xf php-5.5.30.tar.bz2 && \

cd php-5.5.30 && \

./configure \

--prefix=/application/php5.5.30 \

--with-mysql=mysqlnd \

--with-iconv-dir=/usr/local/libiconv \

--with-freetype-dir \

--with-jpeg-dir \

--with-png-dir \

--with-zlib \

--with-libxml-dir=/usr \

--enable-xml \

--disable-rpath \

--enable-bcmath \

--enable-shmop \

--enable-sysvsem \

--enable-inline-optimization \

--with-curl \

--enable-mbregex \

--enable-fpm \

--enable-mbstring \

--with-mcrypt \

--with-gd \

--enable-gd-native-ttf \

--with-openssl \

--with-mhash \

--enable-pcntl \

--enable-sockets \

--with-xmlrpc \

--enable-soap \

--enable-short-tags \

--enable-static \

--with-xsl \

--with-fpm-user=nginx \

--with-fpm-group=nginx \

--enable-ftp >/dev/null && \

make >/dev/null && \

make install >/dev/null && \

ln -s /application/php5.5.30/ /application/php && \

cp php.ini-production /application/php/lib/php.ini && \

Msg "install Php"

}

function main () {

yumInstall

installPhp

}

main

配置php

$ egrep -v "^;.\*|^$|^ $" /application/php/etc/php-fpm.conf

21:[global]

25:pid = /app/logs/php-fpm.pid

32:error\_log = /app/logs/php-fpm.log

50: log\_level = error

92:rlimit\_files = 32768

107:events.mechanism = epoll

121:[www]

139:user = nginx

140:group = nginx

150:listen = 127.0.0.1:9000

161:listen.owner = nginx

162:listen.group = nginx

204:pm = dynamic

215:pm.max\_children = 1024

220:pm.start\_servers = 16

225:pm.min\_spare\_servers = 5

230:pm.max\_spare\_servers = 20

235:pm.process\_idle\_timeout = 15s;

241:pm.max\_requests = 2048

421:slowlog = /app/logs/$pool.log.slow

427:request\_slowlog\_timeout = 10

504:php\_admin\_value[sendmail\_path] = /usr/sbin/sendmail -t -i -f 49000448@qq.com

nginx 服务器LNMP打包

fpm -s dir -t rpm -n lnmp -v 1.0 --description 'nginx-php' -d 'libtool,libtool-ltdl-devel,pcre,pcre-devel,openssl-devel,zlib,zlib-devel,libxml2-devel,libjpeg-devel,freetype,freetype-devel,libpng-devel,gd,gd-devel,curl,curl-devel,libxslt-devel,libmcrypt-devel,mhash-devel,mcrypt,nfs-utils,rpcbind' --post-install /server/scripts/LNMP.sh /server/scripts/ /application /usr/local/libiconv/

[sandow@nginx01 scripts]$ cat lnmp.sh

#!/bin/bash

/usr/sbin/useradd -u 888 nginx -M -s /sbin/nologin

/bin/mkdir /app/logs/ -p

/bin/mkdir -p /data/www/www /data/www/blog /data/www/bbs

chown -R nginx.nginx /data

/application/php/sbin/php-fpm

/application/nginx/sbin/nginx

cat >>/etc/rc.local<<EOF

#nginx+php-fpm by oldboy at 2011

/application/php/sbin/php-fpm

/application/nginx/sbin/nginx

EOF

# apache web服务器

[root@apache01 conf]# egrep -nv "#|^$" httpd.conf

30:ServerRoot "/application/apache2.2.31"

41:Listen 80

54:LoadModule php5\_module modules/libphp5.so

57:<IfModule !mpm\_netware\_module>

58:<IfModule !mpm\_winnt\_module>

67:User nginx

68:Group nginx

70:</IfModule>

71:</IfModule>

90:ServerAdmin you@example.com

99:ServerName 172.0.0.1:80

106:DocumentRoot "/data/www"

116:<Directory />

117: Options FollowSymLinks

118: AllowOverride None

119: Order deny,allow

120: Deny from all

121:</Directory>

133:<Directory "/data/www">

146: Options Indexes FollowSymLinks

153: AllowOverride None

158: Order allow,deny

159: Allow from all

161:</Directory>

167:<IfModule dir\_module>

168: DirectoryIndex index.php index.html

169:</IfModule>

175:<FilesMatch "^\.ht">

176: Order allow,deny

177: Deny from all

178: Satisfy All

179:</FilesMatch>

188:ErrorLog "/app/logs/error.log"

195:LogLevel warn

197:<IfModule log\_config\_module>

202: LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined

203: LogFormat "%h %l %u %t \"%r\" %>s %b" common

205: <IfModule logio\_module>

207: LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O" combinedio

208: </IfModule>

217: CustomLog "/app/logs/access.log" common

224:</IfModule>

226:<IfModule alias\_module>

253: ScriptAlias /cgi-bin/ "/application/apache2.2.31/cgi-bin/"

255:</IfModule>

257:<IfModule cgid\_module>

263:</IfModule>

269:<Directory "/application/apache2.2.31/cgi-bin">

270: AllowOverride None

271: Options None

272: Order allow,deny

273: Allow from all

274:</Directory>

285:DefaultType text/plain

287:<IfModule mime\_module>

292: TypesConfig conf/mime.types

309: AddType application/x-compress .Z

310: AddType application/x-gzip .gz .tgz

311: AddType application/x-httpd-php .php .phtml

312: AddType application/x-httpd-php-source .phps

334:</IfModule>

380:Include conf/extra/httpd-mpm.conf

398:Include conf/extra/httpd-vhosts.conf

416:<IfModule ssl\_module>

417:SSLRandomSeed startup builtin

418:SSLRandomSeed connect builtin

419:</IfModule>

站点配置文件

[root@apache01 extra]# egrep -nv "#|^$" /application/apache/conf/extra/httpd-vhosts.conf

19:NameVirtualHost \*:80

27:<VirtualHost \*:80>

28: ServerAdmin sandow@qq.com

29: DocumentRoot "/data/www/blog"

30: ServerName blog.etiantian.org

31: ErrorLog "/app/logs/www-access\_log"

32: CustomLog "/app/logs/www-access\_log" common

33:</VirtualHost>

34:<VirtualHost \*:80>

35: ServerAdmin sandow@qq.com

36: DocumentRoot "/data/www/bbs"

37: ServerName bbs.etiantian.org

38: ErrorLog "/app/logs/www-access\_log"

39: CustomLog "/app/logs/www-access\_log" common

40:</VirtualHost>

41:<VirtualHost \*:80>

42: ServerAdmin sandow@qq.com

43: DocumentRoot "/data/www/www"

44: ServerName www.etiantian.org

45: ServerAlias etiantian.org

46: ErrorLog "/app/logs/www-access\_log"

47: CustomLog "/app/logs/www-access\_log" common

48:</VirtualHost>

fpm -s dir -t rpm -n lamp -v 1.1 --description 'apache-php' -d 'openssl-devel,zlib,zlib-devel,libxml2-devel,libjpeg-devel,freetype,freetype-devel,libpng-devel,gd,gd-devel,curl,curl-devel,libxslt-devel,libmcrypt-devel,mhash-devel,mcrypt,nfs-utils,rpcbind' --post-install /server/scripts/lamp.sh /application

vim /server/scripts/lamp.sh

#!/bin/bash

/usr/sbin/useradd -u 888 nginx -M -s /sbin/nologin

#ln -s /application/apache2.2.31 /application/apache

#ln -s /application/php5.3.27 /application/php

/bin/mkdir /app/logs/ -p

/bin/mkdir -p /data/www/www /data/www/blog /data/www/bbs

chown -R nginx.nginx /data

/application/php/sbin/php-fpm

/application/apache/bin/apachectl start

cat >>/etc/rc.local<<EOF

#nginx+php-fpm by oldboy at 2011

/application/php/sbin/php-fpm

/application/apache/bin/apachectl start

EOF

# mysql

#!/bin/sh

. /etc/init.d/functions

Dir="/application"

SoftDir="/home/oldboy/tools"

function Msg () {

if [ $? -eq 0 ];then

action "$1" /bin/true

else

action "$1" /bin/false

exit 1

fi

}

function addUser () {

useradd -M -s /sbin/nologin mysql && \

/bin/mkdir -p $Dir $SoftDir && \

Msg "addUser"

}

function installMysql () {

cd $SoftDir && \

tar xf mysql-5.5.32-linux2.6-x86\_64.tar.gz && \

mv mysql-5.5.32-linux2.6-x86\_64 /application/mysql-5.5.32 && \

ln -s /application/mysql-5.5.32/ /application/mysql && \

/application/mysql/scripts/mysql\_install\_db --basedir=/application/mysql/ --datadir=/application/mysql/data/ --user=mysql && \

chown -R mysql.mysql /application/mysql/ && \

cp /application/mysql/support-files/my-small.cnf /etc/my.cnf && \

Msg "installMysql"

}

function conf () {

sed -i 's#/usr/local/mysql#/application/mysql#g' /application/mysql/bin/mysqld\_safe && \

/application/mysql/bin/mysqld\_safe & >/dev/null && \

\cp /application/mysql/support-files/mysql.server /etc/init.d/mysqld && \

sed -i 's#/usr/local/mysql#/application/mysql#g' /etc/init.d/mysqld && \

/etc/init.d/mysqld start && \

echo 'PATH="/application/mysql/bin:$PATH"' >>/etc/profile && \

source /etc/profile && \

Msg "conf"

}

function main () {

addUser

installMysql

conf

}

main

然后我们使用mysqldump来备份

我们配置mysql。

/application/mysql/bin/mysqladmin -u root password '123456'

/application/mysql/bin/mysqladmin -uroot password 123456

mysql 创建wordpress数据库

create database wordpress;

grant all on wordpress.\* to wordpress@'localhost' identified by '123456';

flush privileges;

mysql> create database bbs;

Query OK, 1 row affected (0.00 sec)

mysql> grant all on bbs.\* to bbs@'172.16.1.%' identified by '123456';

Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;

Query OK, 0 rows affected (0.00 sec)

mysql> create database dedecms;

Query OK, 1 row affected (0.08 sec)

mysql> grant all on dedecms.\* to dedecms@'172.16.1.%' identified by '123456';

Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;

Query OK, 0 rows affected (0.00 sec)

fpm -s dir -t rpm -n mysql -v 1.0 --description "mysql" --post-install /server/scripts/mysql.sh /etc/init.d/mysqld /server/scripts/ /application/

[root@db01 scripts]# cat mysql.sh

#!/bin/bash

/usr/sbin/useradd -u 888 mysql -M -s /sbin/nologin

ln -s /application/mysql-5.5.32/ /application/mysql

/application/mysql/scripts/mysql\_install\_db --basedir=/application/mysql/ --datadir=/application/mysql/data/ --user=mysql

chown -R mysql.mysql /application/mysql/

cp /application/mysql/support-files/my-small.cnf /etc/

echo "sandow" >>/etc/rsync.password

chmod 600 /etc/rsync.password

/bin/mkdir /mysql-backup

echo "00 01 \* \* \* /bin/sh /server/scripts/backupDB.sh &>/dev/null" >>/var/spool/cron/root

/etc/init.d/mysqld start

cat >>/etc/rc.local<<EOF

/etc/init.d/mysqld start

EOF