**backup服务器搭建**

使用rsync, ssh, 做到批量分发与实时同步

**RSYNC 数据库服务器**

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

首先检查rsync看是否安装，及安装的版本

[root@backup ~]$ rpm -qa rsync  
rsync-3.0.6-12.el6.x86\_64

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

都是把SRC的数据同步到DEST. 如果没有DEST，但会把SRC的文件列出来而不copy

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

#排除单个文件：  
rsync -auvrtzopgP --progress --exclude=a /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password  
  
#排除多个文件：  
rsync -avz --exclude={a,b} /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password  
  
rsync -avz --exclude-from=paichu.log /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password

**本地备份**

本地 模式，相当于copy 或者rm   
比如

$ rsync -auvrtzopgP /tmp/ /backup/

此命令就会把tmp目录下的文件全部复制到backup目录下面， 如果加上--delete，此参数的意思是把目标目录与源目录同步，当目标目录里有多于源目录的文件就会删除，也就是源目录是啥样，目标目录就是啥样。如果源目录为空，目标目录也会被清空。我们一般都会在目录后面加/意思是目录下面，如果不加就直接把目录复制过去了，在远程的时候也是一样的。

保持属性： -avz 相当于 cp -rp

**通过shell远程传输(remote)**

远程拷贝；相当于ssh带的scp命令，但又优于scp命令的功能。scp每次都是全量拷贝。rsync可以增量拷贝。 当源路径，或目的路径后面包含一个卡号

$ rsync -avz -e 'ssh -p 22' root@172.16.1.41:/tmp/ /tmp/  
The authenticity of host '172.16.1.41 (172.16.1.41)' can't be established.  
RSA key fingerprint is 97:b9:1f:95:8b:7d:ae:59:27:4a:d4:b3:98:ae:c1:77.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '172.16.1.41' (RSA) to the list of known hosts.  
root@172.16.1.41's password:   
receiving incremental file list  
created directory /temp  
./  
hosts  
.ICE-unix/  
  
sent 37 bytes received 184 bytes 9.02 bytes/sec  
total size is 220 speedup is 1.00

此命令便是把ip为172.16.1.41的主机里tmp目录下的文件传到本地的/tmp/下。这里 ssh -p 22命令用于指定端口，默认就是22，-e 这个参数可以不写直接 rsync -avz root@172.16.1.41:/tmp/ /tmp/便可以把41的文件复制过来。且这里是交互式，手动备份与推送，十分不方便。

这里有个问题，如果我们用其它用户推或者拉，的时候需要共享文件的所属主与所属组都对该用户对应

[root@NFS tmp]$ rsync -avz /data/ sandow@172.16.1.41:/data/

此命令是把本地data目录下的文件推到服务器IP为：41下的data目录下。这里两个data目录都要具有对应主机用户的读取执行写入权限，最好的办法就是把该目录的所属主，所属组改为对应的用户

**daemon 模式，**

daemon模式需要配置文件，使用rsync --daemon启动进程，使用独立的进程方式传输文件，默认端口是873。

这里需要两台服务器做测试环境，服务器端名为backup, ip为172.16.1.41主要用于存放数据. 客户端(NFS01)IP为172.16.1.31 主要用于向服务器推送（备份）拉回（还原）数据。然后结合crontab实现实时自动备份同步。

**服务端的配置**

首先我们来配置rsync的主配置文件 rsyncd.conf，默认情况下该文件不存在需要创建，我们也可以使用man rsyncd.conf来查看主配置文件里的参数与说明。

[root@backup ~]$ vim /etc/rsyncd.conf  
#Rsync server  
#created by sandow at 2015-11-05  
#rsyncd.conf start#  
uid = rsync //相当于nfsnobody 客户端连过来具备什么样的权限默认是不存在的  
gid = rsync //相当于nfsnobody  
use chroot = no // 程序出现问题就会开启 开启给个空目录就行  
max connections = 200 //客户端连接数 可以同时  
timeout = 300 //超时 这是服务端 客户端连过来最多多久后就断掉  
strict modes=yes  
port=873  
pid file = /var/run/rstbcd.pid //PID就是进程号唯一标识进程,进程号放文件里面  
lock file = /var/run/rsync.lock //锁文件  
log file = /var/log/rsyncd.log //日志文件，出问题在这里看  
hosts allow = 172.16.1.0/24 //允许ip  
hosts deny = 0.0.0.0/32 //拒绝  
secrets file = /etc/rsync.password //存放用户和密码文件  
  
[backup] //模块，调用下面服务  
comment=backup server by liuchunhao at 2015-11-05 //注释  
path = /backup //共享目录  
read only = false //可读写  
ignore errors //忽略错误  
auth users = rsync\_backup //远程连接的用户，虚拟用户和系统用户没关系  
uid=rsync\_backup  
gid=rsync\_backup  
list = false //不让远程列表，不让看服务端有什么  
secrets file=/etc/rsync.password

配置好rsyncd.conf后便接下来就可以创建对应用户、目录与密码文件

[root@backup ~]$ useradd rsync -s /sbin/nologin -M  
[root@backup ~]$ id rsync  
uid=504(rsync) gid=504(rsync) groups=504(rsync)  
[root@backup ~]$ mkdir -p /backup  
[root@backup ~]$ chown -R rsync.rsync /backup/  
[root@backup ~]$ ls -ld /backup/   
drwxr-xr-t. 3 rsync rsync 4096 Nov 12 06:53 /backup/

创建密码文件，并且修改属性为600

[root@backup ~]$ echo "rsync\_backup:oldboy" >/etc/rsync.password  
[root@backup ~]$ cat /etc/rsync.password  
rsync\_backup:oldboy  
[root@backup ~]$ chmod 600 /etc/rsync.password  
[sandow@backup ~]$ ll /etc/rsync.password   
-rw------- 1 root root 20 11月 6 16:12 /etc/rsync.password

修改/etc/xinetd.d/rsync 文件，把disable 改为no,这应该是最新版本的linux才会有此文件。所以没有的可以不配置

# default: off  
# description: The rsync server is a good addition to an ftp server, as it \  
# allows crc checksumming etc.  
service rsync  
{  
 disable = no  
 flags = IPv6  
 socket\_type = stream  
 wait = no  
 user = root  
 server = /usr/bin/rsync  
 server\_args = --daemon  
 log\_on\_failure += USERID  
}

配置好后便可以启动rsync daemon守护进程了

[sandow@backup tmp]$ rsync --daemon  
  
[sandow@backup tmp]$ ps -ef|grep rsync |grep -v grep  
root 2370 1 0 01:57 ? 00:00:00 rsync --daemon  
  
[sandow@backup tmp]$ netstat -lntup|grep rsync  
tcp 0 0 0.0.0.0:873 0.0.0.0:\* LISTEN 2370/rsync   
tcp 0 0 :::873 :::\* LISTEN 2370/rsync

然后让rsync开机自动启动：

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

这里需要注意的是要让端口873通过防火墙

$ iptables -A INPUT -p tcp -m state --state NEW -m tcp --dport 873 -j ACCEPT  
$ iptables -L 查看一下防火墙是不是打开了 873端口

**客户端的配置**

客户端的配置很简单，仅需要增加一个密码文件并把权限改为600

[root@NFS tmp]$ echo "oldboy" >/etc/rsync.password  
[root@NFS tmp]$ chmod 600 /etc/rsync.password   
[root@NFS tmp]$ ll /etc/rsync.password   
-rw-------. 1 root root 7 Nov 12 02:03 /etc/rsync.password  
[root@NFS tmp]$ cat /etc/rsync.password   
oldboy

**备份数据以及自动备份**

向服务器推送数据：

rsync -auvrtzopgP --progress /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password

或者

rsync -auvrtzopgP --progress /backup/ rsync://rsync\_backup@172.16.1.41/backup --password-file=/etc/rsync.password

从服务器拉回数据

rsync -avz rsync\_backup@172.16.1.41::backup /backup/ --password-file=/etc/rsync.password

或者

rsync -avz rsync://rsync\_backup@172.16.1.41/backup /backup/ --password-file=/etc/rsync.password

**小结**

配置步骤回顾

查看rsync 安装包>> 添加rsync用户 >> 生成配置文件 >> 配置账户，密码文件 >> 密码文件配置权限 600 >> 创建共享的目录并授权rsync >>启动rsync rsync --daemon >> 开机启动   
排错小结：查看推送或者拉取报错信息， 查看日志 tail /var/log/rsyncd.log

重启rsync就必须杀掉进程

[root@backup backup]$ kill `cat /var/run/rsyncd.pid`  
[root@backup backup]$ lsof -i :873  
[root@backup backup]$ pkill rsync  
[root@backup backup]$ killall rsync

rsync 优点

1. 增量备份， 支持socket 集中备份，（支持推拉，都是以客户端参照物）
2. 远程shell 通道模式还可以加密(ssh) 传输 。 socket 需要加密传输，可以利用vpn 服务或ipsec服务

rsync 缺点

1. 大量小文件时候同步的时候，比对时间长，有的时候，rsync进程可能会停止
2. 同步大文件，10G 这样的大文件有时也会问题，中断。 未完整同步前，是隐藏文件

实战作业：   
5.1.4.2 备份全网服务器数据生产架构方案案例模型   
企业案例：rsync上机实战考试题：   
某公司里有一台Web服务器，里面的数据很重要，但是如果硬盘坏了，数据就会丢失，现在领导要求你把数据在其他机器上做一个周期性定时备份。要求如下：   
每天晚上00点整在Web服务器A上打包备份网站程序目录并通过rsync命令推送到服务器B上备份保留（备份思路可以是先在本地按日期打包，然后再推到备份服务器上）。   
具体要求如下：   
1)Web服务器A和备份服务器B的备份目录必须都为/backup。   
2)Web服务器站点目录假定为(/var/www/html)。   
3)Web服务器本地仅保留7天内的备份。   
4)备份服务器上检查备份结果是否正常，并将每天的备份结果发给管理员信箱（选做）。   
5)备份服务器上每周六的数据都保留，其他备份仅保留180天备份（选做）。

拔锚必备思想：   
部署流程步骤熟练   
rsync原理理解

**crontab + rsync 定时推送**

定时任务一般都新建一个目录专门用来存放定时任务的sh.我一般呢会存放在**/server/scripts/**目录下面。 新建一个sh 命名为rsync.dailyBackup.sh. 增加好下内容

$ cat rsync.dailyBackup.sh  
#! /bin/sh  
# backup /backup to 172.16.1.41:/backup  
path=/backup  
hostn=$(hostname)  
dir=${hostn}-172.16.1.31  
ldate=$(date +%F)  
mkdir ${path}/${dir} &&\  
/bin/cp /etc/rc.local ${path}/${dir}/rc.local\_${ldate} &&\  
rsync -az ${path}/ rsync\_backup@172.16.1.41::backup/ --password-file=/etc/rsync.password

首先测试是否成功执行任务，并推送到rsync服务器。然后便写入crontab

$ crontab -l  
#######time sync by lzy at 2015-11-02  
\*/5 \* \* \* \* /usr/sbin/ntpdate time.nist.gov >/dev/null 2>&1  
  
####rsync by lzy at 2015-11-07  
00 01 \* \* \* /bin/sh /server/scripts/rsync.sh >/dev/null 2>&1

**实时同步**

inotify+rsync   
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，但是   
sersync 国人周洋编辑的   
inotify + rsync 实时复制 osersync   
每秒只能同步200张100K的图片 可以用drbd （备节点不可用） 双协

查看系统是否支持inotify，如果没有下列文件便表示不支持

$ ll /proc/sys/fs/inotify/  
total 0  
-rw-r--r-- 1 root root 0 Nov 9 17:39 max\_queued\_events  
-rw-r--r-- 1 root root 0 Nov 9 17:39 max\_user\_instances  
-rw-r--r-- 1 root root 0 Nov 9 17:39 max\_user\_watches

* max\_queued\_events,表示调用inotify\_init时分配给inotify instance中可排队的event的数目的最大值，超出这值的事件被丢弃，触发IN\_Q\_OVERFLOW事件
* max\_user\_instances 每一个user ID可创建的inotify instatnces的数量上限
* max\_user\_watches表示每个inotify instatnces可监控的最大目录数量。

安装inotify-tools   
方法一

$ rpm -ivh /apps/crm/soft\_src/inotify-tools-3.14-1.el6.x86\_64.rpm   
warning: /apps/crm/soft\_src/inotify-tools-3.14-1.el6.x86\_64.rpm: Header V3 DSA/SHA1 Signature, key ID 4026433f: NOKEY  
Preparing... ########################################### [100%]  
 1:inotify-tools ########################################### [100%]  
$ rpm -qa|grep inotify  
inotify-tools-3.14-1.el5.x86\_64

方法二

wget http://github.com/downloads/rvoicilas/inotify-tools/inotify-tools-3.14.tar.gz  
tar zxvf inotify-tools-3.14.tar.gz  
cd inotify-tools-3.14  
./configure --prefix=/usr/local/inotify  
make  
make install

**inotifywait 介绍**

如果文件或者目录变更，那么就会调用inotify命令。是一个非常高效的工具。

wget -O /etc/yum.repos.d/epel.repo http://mirrors.aliyun.com/repo/epel-6.repo  
yum install inotify-tools

其语法格式是

inotifywait [-hcmrq] [-e <event> ] [-t <seconds> ] [--format <fmt> ] [--timefmt <fmt> ] <file> [ ... ]

参数：

* -m 监听
* -r, –recursive 递归监控
* -q, –quiet，不输出信息
* –exclude <pattern\>
* -e <event\>, –event <event\>   
  + delete ,create,close\_write,modify
* –timefmt 设置一个时间样式

inotifywait -mrq --timefmt '%d/%m/%y %H:%M' --format '%T %w %f' -e create /backup   
inotifywait -mrq --timefmt '%d/%m/%y %H:%M' --format '%T %w %f' -e create,delete,close\_write /backup

执行第二个命令后的结果格式结果是

09/11/15 17:47 /backup/ a  
09/11/15 17:47 /backup/ a  
09/11/15 17:47 /backup/ a.txt  
09/11/15 17:47 /backup/ a.txt  
09/11/15 17:47 /backup/ .test.txt.swp  
09/11/15 17:47 /backup/ .test.txt.swx  
09/11/15 17:47 /backup/ .test.txt.swx  
09/11/15 17:47 /backup/ .test.txt.swx  
09/11/15 17:47 /backup/ .test.txt.swp  
09/11/15 17:47 /backup/ .test.txt.swp  
09/11/15 17:47 /backup/ .test.txt.swp  
09/11/15 17:48 /backup/ test.txt  
09/11/15 17:48 /backup/ test.txt  
09/11/15 17:48 /backup/ .test.txt.swp  
09/11/15 17:48 /backup/ .test.txt.swp

刚才注意到 创建一个文件产生的临时文件也都会被记录下来。而且还有重复。为了同步时减少带宽与CPU消耗，就需要使用排除，来排除指定的文件。排除一般有两种方法：1，使用rsync排除; 2,使用inotify排除，同时加入rsync排除,我们一般使用第二种   
inotifywati排队监控目录有 –exclude <pattern\> –formfile <file\> 前面可以用正则，后者只能是具体的目录或文件

vi inotify\_exclude.lst  
/tmp/src/pdf  
@/tmp/src/2014

–formfile 的file只能使用绝对路径，@表示排除。   
用正则排除 --exclude '(.\*/\*\.log|.\*/\*\.swp)$|^/tmp/src/mail/(2014|201.\*/cache.\*)'   
正则很明子了吧。 排除所有目录下以**.log**, **.swp** 结尾的文件，和所有在/tmp/src/mail下面2014这个目录和201\*目录下的cache开头的文件或目录   
在rsync里排除，这里指定file排除时用相对路径。

写一个角本**inotifywait.sh**放到backup目录下面,执行便可以实时同步，以下是几个版本   
粗糙版，每次都会让rsync比对所有文件然后再同步

#!/bin/bash  
/usr/bin/inotifywait -mrq --format '%w%f' -e create,close\_write,delete /backup \  
|while read file  
do  
 cd /backup &&  
 rsync -az ./ --delete rsync\_backup@172.16.1.41::backup \  
 --password-file=/etc/rsync.password  
done

精准版

fullPath="/usr/bin/inotifywait"  
  
$fullPath -mrq --format '%w%f' -e create,close\_write,delete /backup |\  
while read line  
do  
 [ ! -e "$line" ] &&\  
 rsync -az --delete ./ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password && continue  
 rsync -az --delete $line rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password  
done

上面的是不是很简单，那么再来一个更全面的客户端配置文件

#rsync auto sync script with inotify  
#2015-11-09 sandow  
#variables  
current\_date=$(date +%Y%m%d\_%H%M%S)  
source\_path=/backup/  
log\_file=/var/log/rsync\_client.log  
#rsync  
rsync\_server=172.16.1.41  
rsync\_user=rsync\_backup  
rsync\_pwd=/etc/rsync.password  
rsync\_module=backup  
INOTIFY\_EXCLUDE='(.\*/\*\.log|.\*/\*\.swp)$|^/tmp/src/mail/(2014|20.\*/.\*che.\*)'  
RSYNC\_EXCLUDE='/etc/rsyncd.d/rsync\_exclude.lst'  
#rsync client pwd check  
if [ ! -e ${rsync\_pwd} ];then  
 echo -e "rsync client passwod file ${rsync\_pwd} does not exist!"  
 exit 0  
fi  
#inotify\_function  
inotify\_fun(){  
 /usr/bin/inotifywait -mrq --timefmt '%Y/%m/%d-%H:%M:%S' --format '%T %w %f' \  
 --exclude ${INOTIFY\_EXCLUDE} -e modify,delete,create,move,attrib ${source\_path} \  
 | while read file  
 do  
 /usr/bin/rsync -auvrtzopgP --exclude-from=${RSYNC\_EXCLUDE} --progress --bwlimit=200 --password-file=${rsync\_pwd} ${source\_path} ${rsync\_user}@${rsync\_server}::${rsync\_module}   
 done  
}  
#inotify log  
inotify\_fun >> ${log\_file} 2>&1 &

来点链接，深入研究的话可以上去查看   
[How Rsync Works A Practical Overview](https://rsync.samba.org/how-rsync-works.html)   
[用 inotify 监控 Linux 文件系统事件](https://www.ibm.com/developerworks/cn/linux/l-inotify/)   
[inotify: 高效、实时的Linux文件系统事件监控框架](http://www.infoq.com/cn/articles/inotify-linux-file-system-event-monitoring)   
[rsync 的核心算法](http://coolshell.cn/articles/7425.html)

**sersync**

听同事说 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，所以就尝试了一下。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。网上看到有人说性能还不错，但是我觉得：

国产开源，文档不是很全，在2011年之后就没更新了。采用xml配置文件的方式，可读性比较好，但是有些原生的有些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。

虽然提供插件的功能，但很鸡肋，因为软件本身没有持续更新，也没有看到贡献有其它插件出现（可能是我知识面不够，还用不到里面的refreshCDN plugin）。虽然不懂c++，但大致看了下源码 FileSynchronize，拼接rsync命令大概在273行左右，最后一个函数就是排除选项，简单一点可以将–exclude=改成–eclude-from来灵活控制。有机会再改吧。

另外，在作者的文章 Sersync服务器同步程序 项目简介与设计框架 评论中，说能解决上面 rsync + inotify中所描述的问题。阅读了下源码，这个应该是没有解决，因为在拼接rsync命令时，后面的目的地址始终是针对module的，只要执行rsync命令，就会对整个目录进行遍历，发送要比对的文件列表，然后再发送变化的文件。sersync只是减少了监听的事件，减少了rsync的次数——这已经是很大的改进，但每次rsync没办法改变。（如有其它看法可与我讨论）

其实我们也不能要求每一个软件功能都十分健全，关键是看能否满足我们当下的特定的需求。所谓好的架构不是设计出来的，而是进化来的。目前使用sersync2没什么问题，而且看了它的设计思路应该是比较科学的，特别是过滤队列的设计。双向同步看起来也是可以实现。

**rsync常见错误与问题**

Q：如何通过ssh进行rsync，而且无须输入密码？   
A：可以通过以下几个步骤

1. 通过ssh-keygen在server A上建立SSH keys，不要指定密码，你会在~/.ssh下看到identity和identity.pub文件
2. 在server B上的home目录建立子目录.ssh
3. 将A的identity.pub拷贝到server B上
4. 将identity.pub加到~[user b]/.ssh/authorized\_keys
5. 于是server A上的A用户，可通过下面命令以用户B ssh到server B上了。e.g. ssh -l userB serverB。这样就使server A上的用户A就可以ssh以用户B的身份无需密码登陆到server B上了。

Q：如何通过在不危害安全的情况下通过防火墙使用rsync?   
A：解答如下：   
这通常有两种情况，一种是服务器在防火墙内，一种是服务器在防火墙外。无论哪种情况，通常还是使用ssh，这时最好新建一个备份用户，并且配置sshd 仅允许这个用户通过RSA认证方式进入。如果服务器在防火墙内，则最好限定客户端的IP地址，拒绝其它所有连接。如果客户机在防火墙内，则可以简单允许防 火墙打开TCP端口22的ssh外发连接就ok了。

Q：我能将更改过或者删除的文件也备份上来吗？   
A：当然可 以。你可以使用如：rsync -other -options -backupdir = ./backup-2000-2-13 …这样的命令来实现。这样如果源文件:/path/to/some/file.c改变了，那么旧的文件就会被移到./backup- 2000-2-13/path/to/some/file.c，这里这个目录需要自己手工建立起来

Q：我需要在防火墙上开放哪些端口以适应rsync？   
　A：视情况而定。rsync可以直接通过873端口的tcp连接传文件，也可以通过22端口的ssh来进行文件传递，但你也可以通过下列命令改变它的端口：

$ iptables -A INPUT -p tcp -m state --state NEW -m tcp --dport 873 -j ACCEPT

rsync --port 8730 otherhost::或者rsync -e 'ssh -p 2002' otherhost:

Q：我如何通过rsync只复制目录结构，忽略掉文件呢？   
A：rsync -av --include '\*/' --exclude '\*' source-dir dest-dir

Q：为什么我总会出现”Read-only file system”的错误呢？   
A：看看是否忘了设”read only = no”了

Q：为什么我会出现’@ERROR: invalid gid’的错误呢？   
A：rsync使用时默认是用uid=nobody;gid=nobody来运行的，如果你的系统不存在nobody组的话，就会出现这样的错误，可以试试gid = ogroup或者其它

Q：绑定端口873失败是怎么回事？   
A：如果你不是以root权限运行这一守护进程的话，因为1024端口以下是特权端口，会出现这样的错误。你可以用–port参数来改变。

Q：为什么我认证失败？   
A：从你的命令行看来：你用的是

bash$ rsync -a 144.16.251.213::test test  
Password:  
@ERROR: auth failed on module *test*

应该是没有以你的用户名登陆导致的问题，试试rsync -a max@144.16.251.213::test test

Q: 出现以下这个讯息, 是怎么一回事?

@ERROR: auth failed on module xxxxx  
rsync: connection unexpectedly closed (90 bytes read so far)  
rsync error: error in rsync protocol data stream (code 12) at io.c(150)

A: 这是因为密码设错了, 无法登入成功, 请再检查一下 rsyncd.secrets 中的密码设定, 二端是否一致?

Q: 出现以下这个讯息, 是怎么一回事?

password file must not be other-accessible   
continuing without password file   
Password:

A: 这表示 rsyncd.secrets 的档案权限属性不对, 应设为 600。请下 chmod 600 rsyncd.secrets

Q: 出现以下这个讯息, 是怎么一回事?

@ERROR: chroot failed  
rsync: connection unexpectedly closed (75 bytes read so far)  
rsync error: error in rsync protocol data stream (code 12) at io.c(150)

A: 这通常是您的 rsyncd.conf 中的 path 路径所设的那个目录并不存在所致.请先用 mkdir开设好备份目录.

Q：下面的error是什么原因呢

@ERROR: invalid uid rsync\_backup  
rsync error: error starting client-server protocol (code 5) at main.c(1503) [sender=3.0.6]

A: 这是因为服务器上面的目录未创建或者推送用户在该目录没有权限

高并发数据实时同步方案小结   
1。inotify+rsync 文件级别   
2。drbd 文件系统级别   
3。 第三方软件的同步功能， mysql, oracle, mongodb   
4.。 程序双写   
5。业务逻辑解决。（读写分离，备读不到，读主）