

北门吹雪

认清规律，看清大势，顺势而为，乘势而上

linux_rsync定时备份

在linux系统中，需要注意空格使用，有着整体性原则，并且注意大小写问题

Rsync数据同步工具

开源、快速、多功能、可实现全量和增量的本地或远程

具有本地和远程两台主机之间数据快速同步镜像、远程备份的功能类似ssh带的scp命令，还可以实现删除文件和目录的功能，同步内容和属性，还可以同步一个文件里有变化的内容部分

全量：全部备份

增量：差异化备份，效率更高

```
1 | rsync --version # 查看版本信息
```

Rsync的特性

- 1. 支持拷贝特殊文件如链接文件、设备等
- 2. 可以有排除指定文件或目录同步功能，相当于tar的排除功能
- 3. 可以做到保持原文件或，目录的权限、时间、软硬链接、主、组等所有属性均不变
- 4. 可以实现增量同步，即只同步发送变化的数据，因此数据传输效率很高
- 5. 可以通过socket(进程方式)传输文件和数据（服务端和客户端），远程数据同步
- 6. 支持匿名或认证（无需系统用户）的进程模式传输，可实现方便安全的数据备份及镜像

Rsync企业工作场景

两台服务器之间同步数据

方案1：cron + rsync 定时备份针对内部人员，配置信息，发布代码

方案2：sersync+rsync 或 inotify+rsync 实时备份，用户所有数据

rsync工作方式

- 1. 本地模式，相当于cp

```
1 | cp -a /etc/hosts /tmp/ # 把hosts文件拷贝到 /tmp目录下
2 |
3 | rsync -vzrtpg /etc/hosts /mnt/ # 等价于上一个
4 |
5 | rsync -avz --delete /tmp/ /mnt/ # 删除两个文件中单个存在的文件，源和目标保持一致，这
```

--delete 生产中不用和慎用删除相关的操作，一旦删除数据不可挽回

Rsync 作为客户端详细选项

公告

大音希声，大道至简
顶层设计：认识规律，运用规律
最简单的言语，叙说最复杂的实现
始终要思考为什么这样，而不是应该这样

昵称：北门吹雪
园龄：2年3个月
粉丝：25
关注：3
[+加关注](#)

2019年2月						
日	一	二	三	四	五	六
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	1	2
3	4	5	6	7	8	9

搜索

常用链接

[我的随笔](#)
[我的评论](#)
[我的参与](#)
[最新评论](#)
[我的标签](#)

我的标签

[python 购物车小程序\(1\)](#)
[pyth多级菜单\(1\)](#)
[佛学\(1\)](#)

随笔分类

[CentOS\(112\)](#)
[Django\(17\)](#)
[Errors\(31\)](#)
[Git](#)
[Golang\(9\)](#)
[MySQL\(17\)](#)

进行备份时候，有末尾带斜线和不带斜线区别，带斜线只备份该目录下所有内容，不带斜线包括目录本身和目录下所有内容

如： /data/mysql/ /data/mysql ,前者只备份mysql目录下内容， 后者备份 mysql 目录和目录下所有内容

- v 详细输出，传输时的进度等信息
 - z 传输是进行压缩以提高传输效率
 - a 归档模式，表示以递归方式传输文件，保持所有文件属性，把下面的选项包了
 - r 对子目录以递归模式，即目录下的所有目录都是同样传输
 - t 保持文件时间信息
 - o 保持文件主信息
 - p 保持文件权限
 - g 保持文件组信息
 - P 显示同步的过程与传输时的进度等信息
 - D 保持设备文件信息
 - l 保留软链接
- 一般来讲， -avz 就可以了
- e 使用信息协议，指定替代rsh和shell程序上
- exclude=PATTERN 指定排除必须的文件模式(和tar参数一样)
- exclude-from=file(文件名所在的目录文件)（和tar参数一样）

- bwlimit= 限速，单位为k
- delete 让目标目录SRC和源目录数据DST一致

备份需要考虑带宽进行限速，时间选择晚上用户访问少，rsync scp ftp 都有限速功能

遇到的工作故障：

- 1. 某DBA做数据同步，导致用户无法访问网站的问题。

问题原因：

白天时候进行数据库同步备份，并没有对备份进行带宽限速，造成备份时候，占满带宽，用户打不开网页

解决方法：

选择夜间用户访问量少的时候进行备份，以及备份时候进行备份带宽限制，让其不能占满服务带宽

```
1 dd if=/dev/zero of=test1 bs=1M count=128
2 # 当前目录下创建 128M大小的 test1文件， 模拟测试数据
3
4 rsync -avz --bwlimit=10 /backup/ rsync_backup@172.16.1.41::backup/ --password-file=/etc/rs
5
6 # --bwlimit= 限速，单位是k/s
```

第二个模式：使用远端的shell

```
1 rsync -avz /etc/hosts -e 'ssh -p 22' root@10.0.0.31:/mnt
2 # 把 /etc/hosts 文件通过 ssh 服务，推送到远端服务器 10.0.0.31 以root角色接收，存放在/mnt下， 用户@ip
```

- Python(78)
- Tornado(1)
- Web-html+css+js(37)
- 计算机基础(6)
- 思想(26)
- 微信小程序(1)
- 中国传统文化(4)

随笔档案

- 2018年6月 (27)
- 2018年5月 (98)
- 2018年2月 (5)
- 2017年12月 (32)
- 2017年11月 (35)
- 2017年10月 (10)
- 2017年9月 (24)
- 2017年8月 (34)
- 2017年7月 (43)
- 2017年6月 (47)
- 2017年5月 (1)
- 2017年4月 (23)
- 2017年3月 (10)
- 2016年12月 (3)
- 2016年11月 (3)

文章分类

酒酒

最新评论

- 1. Re:Python-去除字符串中不想要的字符
删除左右空字符的lstrip () 和rstrip () 写反了，博主留意更改。
--断北风
- 2. Re:jinja2.exceptions.TemplateNotFound:
home/index.html
你的模板拼写错啦，templates
--Arish
- 3. Re:Python-去除字符串中不想要的字符
删除字符串左右那个弄反了吧
--xxddpac
- 4. Re:Python-去除字符串中不想要的字符
多谢 ,用到了
--呦吼吼吼~
- 5. Re:Python-去除字符串中不想要的字符
不错,学习了,谢谢!
--Ryan_code

阅读排行榜

- 1. Python-去除字符串中不想要的字符 (104618)
- 2. Python-IndexError: list index out of range(29356)
- 3. Python-判断字符串是否以某个字符串开头或结尾? (22957)
- 4. Python-如何拆分含有多重分隔符的字符串?(10612)
- 5. Python-在列表、字典中筛选数据(8562)

评论排行榜

- 1. Python-去除字符串中不想要的字符(5)
- 2. Python-函数式编程-map reduce filter lambda 三元表达式 闭包(3)
- 3. Python-面向对象(1)
- 4. 技术思想(1)

第三个模式，以守护进程的方式传输数据（重点），采用这种模式

rsync 端口 873

守护进程：持续运行的进程

demon 搭在备份服务器上，其他机器把备份文件推送到备份服务器，这个是大多数企业采用的方案

如何搭建demon，服务端？

1. 默认配置文件不存在，可以创建一个配置文件

```
1 | touch /etc/rsyncd.conf # 在linux系统中，几乎所有的配置文件都是这个格式
```

2. 编辑配置文件，并写入以下配置信息， vim /etc/rsyncd.conf

```
1  ##rsyncd.conf start##
2
3  uid = rsync
4
5  # 主，在linux中进程和文件必须属于主和组，远端连接过来时候使用该主访问文件
6
7  gid = rsync # 组
8
9  use chroot = no # 安全机制
10
11 max connections = 200 # 最大连接数
12
13 timeout = 300 # 超时时间，断掉无意义连接
14
15 pid file = /var/run/rsyncd.pid # 进程文件
16
17 lock file = /var/run/rsync.lock # 锁文件
18
19 log file = /var/log/rsyncd.log # 日志文件
20
21 [backup] # 配置模块名
22
23 path = /backup # 服务器提供的访问目录
24
25 ignore errors # 忽略错误
26
27 read only = false # 可写
28
29 list = false # 不可列表
30
31 host allow = 172.16.1.0/24 # 允许远端连接的主机网段
32
33 host deny = 0.0.0.0/32 # 拒绝主机
34
35 auth users = rsync_backup # 独立于系统的虚拟用户，用于验证
36
37 secrets file = /etc/rsync.password # 虚拟用户名密码
38
39 #rsync_config_____end
```

3. 添加用户：

```
1 | useradd rsync -s /sbin/nologin -M # 不允许登录并不需要家目录
```

4. 启动rsync demon

```
1 | rsync --daemon
```

5. 查看是否有这个进程：

```
1 | ps -ef|grep rsync|grep -v grep # 进程是用root存在
```

5. Python-全局解释器锁GIL原理和多线程产生原因与原理-多线程通信机制(1)

推荐排行榜

1. Python-去除字符串中不想要的字符(4)
2. Python-面向网络编程-socket原理(3)
3. Python-函数式编程-map reduce filter lambda 三元表达式 闭包(1)
4. Python-面向对象(1)
5. Python-全局解释器锁GIL原理和多线程产生原因与原理-多线程通信机制(1)

6. 创建 /backup 目录，更改用户和组为配置文件中设定的主和组，不然远端推送不过来

```
1 # low方法: 把 这个目录权限位 777, 太危险, 把配置文件中uid和gid都改为root
2
3 mkdir /backup          # 创建服务器提供的访问目录
4
5 chown rsync.rsync /backup/    # 用户远端推送使用 rsync身份
6
7 ll -d /backup/          # 检查一下
```

7. 创建 /etc/rsync.password 文件，写入：resync_backup:beimenchuixue

```
1 chmod 600 /etc/rsync.password    # 存放密码文件，缩小权限
2
3 # 这个文件是客户端连接过来时候带来的密码beimenchuixue, 用户就可以不需要root密码实现远程推送文件，验证用户
```

8. 检查端口：

```
1 lsof -i :873          # 查看端口是否开启 或 netstat -lntup|grep 873
```

9. 让其开机自启动

```
1 echo '/usr/bin/rsync --daemon' >> /etc/rc.local
2
3 cat /etc/rc.local    # 检查
```

客户端：

1. 检查是否有rsync服务

```
1 rpm -qa *rsync*
2
3 rsync --version
```

2. 只需要生成密码文件即可

```
1 echo 'beimenchuixue' >> /etc/rsync.password    # 设置连接备份服务端密码
2
3 cat /etc/rsync.password    # 检查
4
5 chmod 600 /etc/rsync.password    # 最小化权限
6
7 ll /etc/rsync.password    # 检查
```

3. 创建 /backup 目录，对备份文件，首先要打包到这个目录下，统一推送过去

```
1 mkdir -p /backup
```

测试：

客户端

```
1 touch stu{01..100}    # 在/backup下创建100个文件
```

推送1：

```
1 rsync -avz /backup/ rsync_backup@172.16.1.41::backup/ --password-file=/etc/rsync.password
```

推送2：

```
1 rsync -avz /backup/ rsync://rsync_backup@172.16.1.41/backup/ --password-file=/etc/rsync.pa
```

backup/ 模块名，如果推到某个子目录，直接在模块后面跟子目录，依赖于客户端配置

在linux系统中，配置文件会加载到内存，修改配置文件需要重启服务

```
1 pkill rsync    # 结束进程
```

```

2
3 | lsof -i :873      # 检查进程是否结束
4
5 | rsync --daemon   # 启动
6
7 | lsof -i :873      # 检查服务是否启动

```

问题排错：

1. @ERROR: chdir failed

服务器端对应模块下没有备份目录

解决方法：

```
1 | mkdir /backup    # 创建这个模块对应的目录
```

2. rsync: mkstemp ".stu100.JWpxhq" (in backup) failed: Permission denied (13)

权限不够

解决方法：

```

1 | chown rsync.rsync /backup/          # 更改为 rsync需要的主和组
2
3 | ll -d /backup/                      # 检查

```

3. @ERROR: invalid uid rsync

备份服务器rsync这个用户不存在

解决方法：

```

1 | useradd rsync -s /sbin/nologin -M    # 添加这个用户，不需要家目录和不允许登录
2
3 | id rsync # 检查

```

4. @ERROR: auth failed on module backup

模块验证失败

解决方法：

客户端和服务端：

检查客户端 /etc/rsync.password ,注意文件中是有多余空格和敲错的字符

检查客户端和服务端的 /etc/rsync.password 是否授权为600

```
1 | cat -A /etc/rsync.password    # 查看所有字符，包括空格结尾符号
```

5. rsync: failed to connect to 172.16.1.22: Connection refused (111)

解决方法： 服务端 rsync --daemon 没开

```

1 | ps -ef | grep rsync          # 检查
2 | rsync --daemon              # 启动

```

如何对应多个模块？

更改 /etc/rsyncd.conf配置文件，把共有的部分提取放到全局

```

1 | uid = rsync
2
3 | gid = rsync
4
5 | use chroot = no
6
7 | max connections = 200
8
9 | timeout = 300

```

```

10
11 pid file = /var/run/rsyncd.pid
12
13 lock file = /var/run/rsync.lock
14
15 log file = /var/log/rsyncd.log
16
17 path = /backup
18
19 ignore errors
20
21 read only = false
22
23 list = false
24
25 host allow = 172.16.1.0/24
26
27 #host deny = 0.0.0.0/32
28
29 auth users = rsync_backup
30
31 secrets file = /etc/rsync.password
32
33 [backup]
34
35 path = /backup
36
37 [beimen]
38
39 path = /beimen
40
41 # 每个单独的模块，单独有一个path，对应不同的推送过来的数据

```

修改配置的时候，需要时刻备份，出了问题可以快速复原

```
1 cp /etc/rsyncd.conf{,.$(date +%F)_v1} # 专业备份格式，加上备份时间和更改次数版本
```

假如想要排除一些文件如何做？

```

--exclude=文件名 # 排除单个文件

--exclude={filename1, filename2,...} 排除多个文件

--exclude={a..z} 排除连续的

--exclude=paichu.log 按文件排除

```

无差异同步

```
--delete
```

服务器端放了文件，推送的时候，以客户端推送目录为依据，客户端有啥服务端就有啥，多余的删除

拉取，以客户端/backup为依据

与这个无差异同步发生的血案：

视频网站，推到服务器上上线发布，本地/backup只有当天发布的内容，服务器上却有以前的所有文件，执行含 --delete 的rsync推送命令

结果：服务器端删除以前的所有，只有当天的了

相当于 rm -rf

提示：非常危险，慎用

rsync三种工作模式：

1. 本地模式，相当于cp

2. 通道模式 -e指定用什么协议传输

```
1 | rsync -avz /etc/hosts -e 'ssh -p 22' root@10.0.0.31:/mnt, 配合ssh密钥进行免密码传输
```

3. daemon模式

提示：内网不需要加密，加密性能能有损失

跨机房，使用vpn(pptp, openvpn, ipsec)

rsync

优点：

1. 增量备份，支持socket(daemon)，集中备份（支持推拉，以客户端为参照物）
2. 可以利用ssh, vpn服务等加密传输远程数据

缺点：

1. 大量小文件同步时候，对比时间长，有时候rsync进程会停止
2. 同步大文件，比如10G会出现中断问题，未同步完成前，是隐藏文件，通过续传等参数实现传输
3. 一次性远程拷贝可以用scp

如何检查服务端防火墙是否阻挡？

telnet ip地址 端口

rsync服务配置过程总结：

服务器端：

1. 检查rsync安装包

```
1 | rpm -qa rsync
```

2. 添加rsync服务用户，管理本地目录的

```
1 | useradd rsync -s /sbin/nologin -M
```

3. 生成rsyncd.conf配置文件

```
1 | vim rsyncd.conf 放入已经配置好的配置文本
```

4. 根据 rsyncd.conf 的auth users 配置账号，用于远程连接，并根据 secrets file 参数生成密码文件

5. 为密码文件配置权限

```
1 | chmod 600 /etc/rsync.password
2 |
3 | ll /etc/rsync.password
```

6. 创建备份目录，并授权rsync服务管理

```
1 | mkdir -p /backup
2 |
3 | chown -R rsync.rsync /backup
4 |
5 | ll /backup
```

7. 启动 rsync服务的daemon模式，接收其他服务器推送过来的数据

```
1 | rsync --daemon
2 |
3 | lsof -i :873
```

8. 加入开机自启动

```
1 echo `/usr/bin/rsync --daemon` >> /etc/rc.local
2
3 tail -1 /etc/rc.local
```

客户端：

1. 生成服务器端需要的密码文件

```
1 echo `beimenchuixue` > /etc/rsync.password
2
3 cat /etc/rsync.password
```

2. 为密码文件配置权限

```
1 chmod 600 /etc/rsync.password
2
3 ll /etc/rsync.password
```

测试：

往服务器推送文件，看能不能成功

```
1 rsync -avz /backup/ rsync_backup@172.16.1.41::backup --password-file=/etc/rsync.password
```

出错误排错方式：

1. 看输出结果
2. 看日志 /var/log/rsync.log
3. 熟悉部署流程

总结过程：

服务端：

1. 添加对备份文件操作的用户，为了统一，客户端和服务端统一名字

```
1 useradd rsync -s /sbin/nologin -M
2 id rsync
```

2. 创建备份目录 /backup

```
1 mkdir /backup
```

3. 把1步骤建立的用户，授权 /backup目录

```
1 chown -R rsync.rsync /backup/
```

4. 写配置文件， /etc/rsyncd.conf

```
1 vim /etc/rsyncd.conf
2
3 uid = rsync
4 gid = rsync
5 use chroot = no
6 max connections = 200
7 timeout = 300
8 pid file = /var/run/rsyncd.pid
9 lock file = /var/run/rsync.lock
10 log file = /var/log/rsyncd.log
11 path = /backup
12 ignore errors
13 read only = false
14 list = false
15 host allow = 172.16.1.0/24
16 #host deny = 0.0.0.0/32
17 auth users = rsync_backup
18 secrets file = /etc/rsync.password
19 [backup]
```



```
20 | path = /backup
```

5. 写密码文件 /etc/rsync.password，rsync_backup这个只是用于客户端验证的用户，让其他人不可见

```
1 | echo 'rsync_backup:beimenchuixue' > /etc/rsync.password<br>cat /etc/rsync.password<br>chmod
```

6. 启动daemon，并将这条启动写入/etc/rc.local

```
1 | rsync --daemon
2 | echo 'rsync --daemon' >> /etc/rc.local
3 | tail -1 /etc/rc.local
```

客户端：

1. 编写 /etc/rsync.password 文件，写入密码，让其他人不可见

```
1 | echo 'beimenchuixue' > /etc/rsync.password<br>chmod 600 /etc/rsync.password
```

2. -avz 备份上传，--password-file指定密码文件，本地文件和服务器的顺序决定是推还是拉

```
1 | mkdir /backup
2 | echo 'echo hello' > /backup/hello.sh # 创建测试文件
3 | rsync -avz /backup/ rsync_backup@172.16.1.22::backup/ --password-file=/etc/rsync.password
4 | # v 具体脚本可以省略，表示显示信息
```

3. 去backup服务器查看是否备份

过程程序化：

1. 一键配置rsync客户端

```
1 | #!/bin/sh
2 |
3 | # author: beimenchuixue
4 | # email: 42283556@qq.com
5 | # blog:Warning: http://www.cnblogs.com/2bjuijiu/
6 |
7 | rsyncConf="/etc/rsyncd.conf"
8 | rsyncPid="/var/run/rsyncd.pid"
9 | rsyncPath="/backup"
10 | ServerPwdFile="/etc/rsync.password"
11 | rsyncUser="rsync"
12 | loginUser="beimenchuixue"
13 | loginPwd="123456"
14 |
15 | . /etc/init.d/functions
16 |
17 | function sureOK {
18 |     [ $? -eq 0 ] && {
19 |         action "$2 is" /bin/true
20 |     } || {
21 |         action "$2 is" /bin/false
22 |         exit $?
23 |     }
24 | }
25 |
26 | function hasInstallRsync {
27 |     rsync --version &> /dev/null
28 |     sureOK $? "hasInstallRsync"
29 | }
30 | # hasInstallRsync
31 |
32 | function rsyncConf {
33 |     [ -f $rsyncConf ] && {
34 |         cat /dev/null > $rsyncConf
```

```

35     sureOK $? "init rsyncConf"
36 }
37 cat >>$rsyncConf<<EOF
38 uid = $rsyncUser
39 gid = $rsyncUser
40 use chroot = no
41 max connections = 200
42 timeout = 300
43 pid file = $rsyncPid
44 lock file = /var/run/rsync.lock
45 log file = /var/log/rsyncd.log
46 ignore errors
47 read only = false
48 list = false
49 # host allow = 172.16.1.0/24
50 # host deny=0.0.0.0/32
51 auth users = $loginUser
52 secrets file = $ServerPwdFile
53 [backup]
54 path = $rsyncPath
55 EOF
56     sureOK $? "rsyncConf"
57 }
58 # rsyncConf
59
60 function addRsyncUser {
61     id $rsyncUser &> /dev/null
62     [ $? -eq 0 ] || {
63         useradd $rsyncUser -s /sbin/nologin -M
64     }
65     sureOK $? "addRsyncUser"
66 }
67 # addRsyncUser
68
69 function initRsyncPath {
70     [ -d $rsyncPath ] || {
71         mkdir -p $rsyncPath
72         sureOK $? "create $rsyncPath"
73     }
74     chown -R ${rsyncUser}.${rsyncUser} $rsyncPath
75     sureOK $? "initRsyncPath"
76 }
77 # initRsyncPath
78
79 function initServerRsyncPwdFile {
80     [ -f $ServerPwdFile ] && {
81         cat /dev/null > $ServerPwdFile
82         sureOK $? "clear ServerRsyncPwdFile"
83     }
84     echo "$loginUser:$loginPwd" > $ServerPwdFile
85     sureOK $? "write rsyncPwd"
86     chmod 000 $ServerPwdFile
87 }
88 # initServerRsyncPwdFile
89
90 function main_BeiMenChuiXue {
91     hasInstallRsync
92     rsyncConf
93     addRsyncUser
94     initRsyncPath
95     initServerRsyncPwdFile
96 }
97 main_BeiMenChuiXue

```

2. rsync启动脚本

```

1  #!/bin/sh
2
3  # author: beimenchuiXue
4  # email: 42283556@qq.com

```

```
5 # blog:Warning: http://www.cnblogs.com/2bjuijiu/
6
7 # chkconfig: 2345 98 25
8
9 rsyncPid="/var/run/rsyncd.pid"
10
11 . /etc/init.d/functions
12
13 function sureOK {
14     [ $1 -eq 0 ] && {
15         action "$2 is" /bin/true
16     } || {
17         action "$2 is" /bin/false
18         exit $1
19     }
20 }
21
22 [ $# -eq 1 ] || {
23     echo "$0 (start|stop|restart)"
24     exit 1
25 }
26
27 function startRsyncDaemon {
28     [ -f $rsyncPid ] && {
29         echo "rsync is running"
30         sureOK 1 "start rsync"
31     }
32     rsync --daemon
33     sureOK $? "start rsync"
34 }
35 # RsyncDaemon
36
37 function stopRsyncDaemon {
38     if [ ! -f $rsyncPid ]; then
39         echo "rsync is stoping"
40         sureOK 1 "stop rsync"
41     else
42         kill $(cat $rsyncPid)
43         sleep 1
44         if [ -f $rsyncPid ]; then
45             kill -9 $(cat $rsyncPid)
46             sureOK $? "stop rsync"
47         else
48             sureOK 0 "stop rsync"
49         fi
50     fi
51 }
52 # stopRsyncDaemon
53
54 function restartRsyncDaemon {
55     stopRsyncDaemon
56     startRsyncDaemon
57 }
58
59 case $1 in
60     start)
61         startRsyncDaemon
62         ;;
63     stop)
64         stopRsyncDaemon
65         ;;
66     restart)
67         restartRsyncDaemon
68         ;;
69     *)
70         echo "$0 (start|stop|restart)"
71         exit 3
72 esac
```

分类: CentOS


好文置顶

关注我

收藏该文







北门吹雪

关注 - 3

粉丝 - 25

+加关注

« 上一篇: [linux 软件安装策略和升级策略](#)
» 下一篇: [linux NFS](#)

posted @ 2017-12-19 20:03 北门吹雪 阅读(708) 评论(0) 编辑 收藏

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】超50万VC++源码: 大型组态工控、电力仿真CAD与GIS源码库！
【推荐】专业便捷的企业级代码托管服务 - Gitee 码云

相关博文：

- [RSync实现文件备份同步](#)
- [rsync 备份](#)
- [linux定时备份Mysql](#)
- [rsync从linux到linux的文件同步备份](#)
- [rsync定时自动增量备份远程服务器数据](#)

最新新闻：

- “墨子号”科研团队获美国2018年度克利夫兰奖
- 苹果失去“美国人最亲密品牌”称号 迪斯尼取而代之
- 英伟达第四季净利润5.67亿美元 同比下滑49%
- 大疆等在美被诉专利侵权 起诉方疑似NPE机构
- 特朗普的《AI 倡议》存在一个致命问题：海外AI人才的政策依然欠缺
- » [更多新闻...](#)

