一　SVN

Subversion

1 安装Subversion服务器

1）YUM安装subversion软件

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

[root@web1 ~]# rpm -q subversion

2)创建版本库

[root@web1 ~]# mkdir /var/svn/

[root@web1 ~]# svnadmin create /var/svn/project

[root@web1 ~]# ls /var/svn/project/

conf/ db/ format hooks/ locks/ README.txt

3）本地导入初始化数据

[root@web1 ~]# cd /usr/lib/systemd/system/

[root@web1 ~]# svn import . file:///var/svn/project/ -m "Init Data"

-m:后面跟注释

4）修改配置文件，创建账户与密码

提示：所有配置文件，要求顶头写，开头不要有空格。

[root@web1 ~]# vim /var/svn/project/conf/svnserve.conf

[general]

### These options control access to the repository for unauthenticated

### and authenticated users. Valid values are "write", "read",

### and "none". The sample settings below are the defaults.

anon-access = none

//19行，匿名无任何权限

auth-access = write

//20行，有效账户可写

password-db = passwd

//27行，密码文件，当前目录下的密码文件

authz-db = authz

//34行，ACL访问控制列表文件，当前目录下的访问控制列表文件

[root@web1 ~]# vim /var/svn/project/conf/passwd

… …

[users]

harry = 123456

//用户名和密码

tom = 123456

//用户名和密码

[root@web1 ~]# cat /var/svn/project/conf/authz

[/] //定义ACL访问控制

harry = rw //用户对项目根路径可读可写

tom = rw

5）启动服务

[root@web1 ~]# svnserve -d -r /var/svn/project　　＃－d：启动服务放后台，以守护进程的方式运行　-r：共享那个目录　--listen-port=port：设置监听的端口，默认端口3690

[root@web1 ~]# svnserve -d 　　＃共享svn服务下的所有仓库文件

[root@web1 ~]# netstat -nutlp |grep svnserve

tcp 0 0 0.0.0.0:3690 0.0.0.0:\* LISTEN 4043/svnserve

备注：启动服务也可以使用svnserve -d启动，但客户端访问时需要指定绝对路径（svn://服务器IP/var/svn/project）。

6）客户端测试

将服务器上的代码下载到本地

[root@web2 ~]# cd /tmp

[root@web2 ~]# svn --username harry --password 123456 \

co svn://192.168.2.100/ code

//建立本地副本,从服务器192.168.2.100上co下载代码到本地code目录

//用户名harry,密码123456

＃co：下载　　ci:上传

-----------------------------------------------------------------------

ATTENTION! Your password for authentication realm:

<svn://127.0.0.1:3690> b72f45f0-bbe5-4a0c-ad4a-37f52704f0b1

can only be stored to disk unencrypted! You are advised to configure

your system so that Subversion can store passwords encrypted, if

possible. See the documentation for details.

You can avoid future appearances of this warning by setting the value

of the 'store-plaintext-passwords' option to either 'yes' or 'no' in

'/root/.subversion/servers'.

-----------------------------------------------------------------------

Store password unencrypted (yes/no)? yes //提示是否保存密码

[root@web2 ~]# cd /tmp/code

[root@web2 code]# ls

[root@web2 code]# vim user.slice //挑选任意文件修改其内容

[root@web2 code]# svn ci -m "modify user" //将本地修改的数据同步到服务器

[root@web2 code]# svn update //将服务器上新的数据同步到本地

[root@web2 code]# svn info svn://192.168.2.100 //查看版本仓库基本信息

[root@web2 code]# svn log svn://192.168.2.100 //查看版本仓库的日志

[root@web2 code]# echo "test" > test.sh //本地新建一个文件

[root@web2 code]# svn ci -m "new file" //提交失败，该文件不被svn管理

[root@web2 code]# svn add test.sh //将文件或目录加入版本控制

[root@web2 code]# svn ci -m "new file" //再次提交，成功

[root@web2 code]# svn mkdir subdir //创建子目录

[root@web2 code]# svn rm timers.target //使用svn删除文件

[root@web2 code]# svn ci -m "xxx" //提交一次代码

[root@web2 code]# vim umount.target //任意修改本地的一个文件

[root@web2 code]# svn diff //查看所有文件的差异

[root@web2 code]# svn diff umount.target //仅查看某一个文件的差异

[root@web2 code]# svn cat svn://192.168.2.100/reboot.target //查看服务器文件的内容

[root@web2 code]# sed -i 'd' tmp.mount

//删除文件所有内容，但未提交

[root@web2 code]# svn revert tmp.mount

//还原tmp.mount文件

[root@web2 code]# rm -rf \*.target

//任意删除若干文件

[root@web2 code]# svn update

//还原

[root@web2 code]# sed -i '1a #test###' tuned.service

//修改本地副本中的代码文件

[root@web2 code]# svn ci -m "xxx"

//提交代码

[root@web2 code]# svn merge -r7:2 tuned.service

//将文件从版本7还原到版本2

使用svn命令测试svnserver服务时可以使用的命令列表如表-1所示。

表－1 svn命令列表



2多人协同工作

1）远程连接两个终端，每个人下载代码本地副本，注意web1(192.168.2.100)和web2（192.168.2.200）代表了两个不同的主机，看清楚操作是在哪一台计算机上执行！

[root@web1 ~]# cd /tmp

[root@web1 ~]# svn --username tom --password 123456 \

> co svn://192.168.2.100/ code

[root@web2 ~]# cd /tmp

[root@web2 ~]# svn --username harry --password 123456 \

> co svn://192.168.2.100/ code

[root@web1 ~]# cd code

[root@web2 ~]# cd code

2) harry和tom修改不同的文件

[root@web1 mycode]# sed -i "3a ###tom modify#####" tmp.mount

[root@web1 mycode]# svn ci -m "has modified"

[root@web2 mycode]# sed -i "3a ###harry modify#####" umount.target

[root@web2 mycode]# svn ci -m "has modified"

[root@web2 mycode]# svn update

[root@web1 mycode]# svn update

3）harry和tom修改相同文件的不同行

[root@srv5 ~]# cd harry

[root@web1 mycode]# sed -i "3a ###tom modify#####" user.slice

[root@web1 mycode]# svn ci -m "modified"

[root@web2 mycode]# sed -i "6a ###harry modify#####" user.slice

[root@web2 mycode]# svn ci -m "modified" //提交失败

Sending svnserve

Transmitting file data .svn: Commit failed (details follow):

svn: File '/user.slice' is out of date（过期）

[root@web2 mycode]# svn update //提示失败后，先更新再提交即可

[root@web2 mycode]# svn ci -m "modified" //提交成功

Sending user.slice

Transmitting file data .

4) harry和tom修改相同文件的相同行

[root@web1 mycode]# sed -i '1c [UNIT]' tuned.service

[root@web1 mycode]# svn ci -m "modified"

[root@web2 mycode]# sed -i '1c [unit]' tuned.service

[root@web2 mycode]# svn ci -m "modified"

Sending tuned.service

Transmitting file data .svn: Commit failed (details follow):

svn: File '/tuned.service' is out of date(过期)

[root@web2 mycode]# svn update //出现冲突，需要解决

Conflict(冲突) discovered in 'tuned.service'.

Select: (p) postpone, (df) diff-full, (e) edit,

(mc) mine-conflict, (tc) theirs-conflict,

(s) show all options:p //选择先标记p，随后解决

[root@web2 mycode]# ls

tuned.service tuned.service.mine tuned.service.r10 tuned.service.r9

[root@web2 mycode]# mv tuned.service.mine tuned.service

[root@web2 mycode]# rm -rf tuned.service.r10 tuned.service.r9

[root@web2 mycode]# svn ci -m "modified" //解决冲突

3使用dump指令备份版本库数据

[root@web1 ~]# svnadmin dump /var/svn/project > project.bak //备份

\* Dumped revision 0.

\* Dumped revision 1.

\* Dumped revision 2.

\* Dumped revision 3.

\* Dumped revision 4.

\* Dumped revision 5.

\* Dumped revision 6.

\* Dumped revision 7.

\* Dumped revision 8.

\* Dumped revision 9.

\* Dumped revision 10.

\* Dumped revision 11.

[root@web1 ~]# svnadmin create /var/svn/project2 //新建空仓库

[root@web1 ~]# svnadmin load /var/svn/project2 < project.bak //还原

3，svn命令

1)创建svn仓库

svnadmin create 目录名称　　　　#创建svn仓库

svn import . <file://仓库路径> -m　＂描述＂

svnserve -d -r svn仓库路径

svnadmin dump svn仓库路径 > 备份文件名　　　#备份svn仓库内容

svnadmin　load 新建的svn空仓库路径 < 备份的文件名　　#还原svn仓库内容

1. 客户端下载

svn --username 用户名　--password 密码　co svn://svn仓库ip地址/ code

##把仓库数据下载到本地的code目录上 ,co:下载

svn ci -m “描述信息”　　#提交信息

svn add 当前文件名　　　#向svn添加该文件

svn update #将服务器上新的数据同步到本地

svn log svn://服务器svnip地址　　＃查看版本仓库的日志

svn　info svn://服务器svnip地址 #查看仓库基本信息

svn diff #查看所有文件的差异

svn cat svn://svn仓库地址/文件名 #查看服务器文件的内容

svn revert 文件名　　　　#还原该文件(前提：未提交svn仓库是可以还原)

svn merge -r 7:2 文件名　#还原该文件(前提：该文件的名称还在时可以还原)