**Operation Day06**

Subversion 是一个自由/开源的版本控制系统

-在Subversion 管理下，文件和目录可以超越时空

-Subversion 允许你数据恢复到早期版本

-允许你和别恶人协作文档并跟踪所做的修改

**Subversion架构**

●客户端

-命令行

-图形

●通信方式

-本地访问

-SVN服务器

-Web服务

svnserve -d（systemctl start svnserver) 所有仓库贡献出去,客户端访问某个仓库要写绝对路径

svn --username harry --password pass \

co svn://192.168.2.100/ code // co 下载

svn add test.sh //将文件或目录加入版本控制

svn ci -m "new file" //提交，成功(成功一次之后，不用再加入版本控制）

[root@room9pc01 ~]# cd /root/aa

[root@room9pc01 aa]# git clone https://github.com/mazhengwei/file

[root@room9pc01 aa]# cd file/

[root@room9pc01 file]# echo aaa > a.sh

[root@room9pc01 file]# git add . ////添加文件

[root@room9pc01 file]# ls

a.sh foo.sh README.md

[root@room9pc01 file]# git commit -m "aa"

[root@room9pc01 file]# git push //上传文件，输入帐号和密码

**案例1：Subversion基本操作**

本案例要求先快速搭建好一台Subversion服务器，并测试该版本控制软件：

●创建版本库

●导入初始化数据

●检出数据至用户本地副本

●对本地副本进行增删改查等操作

步骤一：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 //创建一个新的版本库

[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" //名字随意写

-import指令执行导入操作

-m 选项设置注释性的字符串

4）修改配置文件，创建账户与密码（可以有效增强客户端访问的安全性）

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

anon-access = none //设置拒绝匿名账户访问，此处不允许匿名访问

auth-access = write //设置认证的账户权限为可写权限

password-db = passwd //账户名称与密码的存放文件名，该文件在conf目录下

authz-db = authz //基于路径的访问控制文件名（可以对文件或目录设置权限）

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

… …

[users]

harry = pass //设置用户名和密码

tom = pass //设置用户名和密码

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

[/] //根路径设置权限，其他路径也可设置

harry = rw //用户对项目根路径可读可写

tom = rw

\* = r //其他人只读

5）启动服务

[root@web1 ~]# svnserve -d -r /var/svn/project

//-d放入后台 -r设置虚拟路径 仅把一个目录共享出去

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

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

**步骤二：客户端测试(192.168.2.200)**

1）将服务器上的代码下载到本地

[root@web2 ~]# cd /tmp //找内容少些的目录，便于读取

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

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pass

[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

// -m commit（提交修改） add不会自动提交版本库服务器，需要加-m

[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 xxxxx' 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

## **案例2：使用Subversion协同工作**

**步骤一：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 harry --password pass \

> co svn://192.168.2.100/ code

[root@web2 ~]# cd /tmp

[root@web2 ~]# svn --username tom --password pass \

> 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 xxxx" tmp.mount

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

[root@web2 mycode]# sed -i "3a xxxx" 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@web1 code]# sed -i "3a xxxx" user.slice

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

[root@web2 code]# sed -i "6a xxxx" user.slice

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

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

4) harry和tom修改相同文件的相同行

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

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

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

[root@web2 code]# svn ci -m "modified" //提示过期

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

Conflict 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 code]# ls

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

[root@web2 code]# mv tuned.service.mine tuned.service //选择需要的版本

[root@web2 code]# rm -rf tuned.service.r10 tuned.service.r9 //删除其他的

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

**步骤二：使用dump指令备份版本库数据**

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

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

## **案例3：制作nginx的RPM包**

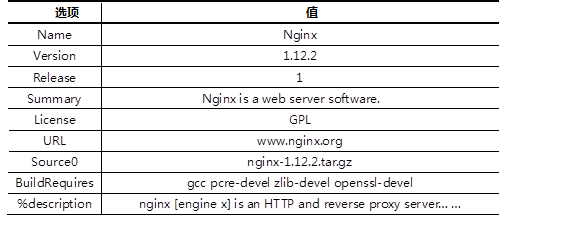
本案例使用nginx-1.12.2版本的源码软件，生产对应的RPM包软件，具体要求如下：

●软件名称为nginx

●软件版本为1.12.2

●RPM软件包可以查询描述信息

●RPM软件包可以安装及卸载



**步骤一：安装rpm-build软件**

1）安装rpm-build软件包

[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

Version:1.12.2 //必填

Release: 10

Summary: Nginx is a web server software.

License:GPL //必填

URL: www.test.com

Source0:nginx-1.12.2.tar.gz //必填

#BuildRequires: //备注，需添加的依赖软件包，写了也安装不了

#Requires:

%description

nginx [engine x] is an HTTP and reverse proxy server.

%prep

%setup –q //自动解压源码包，并cd进入目录

%build

./configure //解压

make %{?\_smp\_mflags}

%install

make install DESTDIR=%{buildroot}

cp /root/rpmbuild/SPECS/nginx.sh %{buildroot}/usr/local/nginx/

##//非必须操作，注意，这里是将一个脚本拷贝到安装目录，必须提前准备该文件

%files

%doc

/usr/local/nginx/\* //对哪些文件与目录打包

%changelog

**步骤二：使用配置文件创建RPM包**

1）安装依赖软件包

[root@web1 ~]# yum –y install gcc pcre-devel zlib-devel openssl-devel

2）rpmbuild创建RPM软件包

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

[root@web1 ~]# cd /root/rpmbuild/RPMS/x86\_64/

[root@web1 ~]# scp nginx-1.12.2-10.x86\_64.rpm 192.168.2.200:/root

**步骤三：安装、卸载软件**

[root@web2 ~]# yum -y install nginx-1.12.2-10.x86\_64.rpm

[root@web2 ~]# rpm -qi nginx