第 3 章 SVN 版本管理实战

3.1 Subversion 服务器简介

Subversion（简称 SVN）是一个自由、开源的版本控制系统。

在实际使用分布式版本控制系统的时候，其实很少在两台计算机之间推送版本库的修改，这两台计算机因为可能不在一个局域网内而互相访问不了，也可能今天你的同事病了，他的计算机压根没有开机。因此，分布式版本控制系统通常也有一台充当“中央服务器”的计算机，但这个服务器的作用仅仅是方便“交换”大家的修改，没有它大家也一样工作，只是不方便交换修改而已。

在 Subversion 管理下，文件和目录可以超越时空。Subversion 允许数据恢复到早期版本，或者是检查数据修改的历史。正因为如此，许多人将版本控制系统当作一种神奇的“时间机器”。

Subversion 的版本库可以通过网络访问，从而使用户可以在不同的计算机上进行操作。从某种程度上来说，允许用户在各自的空间里修改和管理同一组数据可以促进团队协作。因为修改不再是单线进行，开发速度会更快。

此外，由于所有的工作都已经版本化，也就不必担心由于错误的更改而影响软件质量问题，即使出现不正确的更改，只要撤销那一次更改操作（回滚）即可。

3.2 Subversion 的功能特性

Subversion 版本库有如下特点：

（1）版本化的目录。CVS 只能跟踪单个文件的变更历史，但是 Subversion 实现的“虚拟”版本化文件系统可以跟踪目录树的变更。在 Subversion 中，文件和目录都是版本化的。

（2）真实的版本历史。由于只能跟踪单个文件的变更，CVS 无法支持如文件复制和改名这些常见的操作——这些操作改变了目录的内容。同样地，在 CVS 中，一个目录下的文件只要名字相同即拥有相同的历史，即使这些同名文件在历史上毫无关系。而在 Subversion 中，可以对文件或目录进行增加、复制和改名操作，也解决了同名而无关的文件之间的历史联系问题。

（3）原子提交。一系列相关的更改，要么全部提交到版本库，要么一个也不提交。这样用户就可以将相关的更改组成一个逻辑整体，防止出现只有部分修改提交到版本库的情况。

（4）版本化的元数据。每一个文件和目录都有自己的一组属性——键和对应的值。可以根据需要建立并存储任何键/值对。和文件本身的内容一样，属性也在版本控制之下。

（5）可选的网络层。Subversion 在版本库访问的实现上具有较高的抽象程度，有利于人们实现新的网络访问机制。Subversion 可以作为一个扩展模块嵌入 Apache。这种方式在稳定性和交互性方面有很大的优势，可以直接使用服务器的成熟技术——认证、授权和传输压缩等。此外，Subversion 自身也实现了一个轻型的、可独立运行的服务器软件，这个服务器使用了一个自定义协议，可以轻松使用 SSH 封装。

（6）一致的数据操作。Subversion 用一个二进制差异算法描述文件的变化，对文本（可读）和二进制（不可读）文件的操作方式一致。这两种类型的文件压缩存储在版本库中，而差异信息则在网络上双向传递。

（7）高效的分支和标签操作。在 Subversion 中，分支与标签操作的开销与工程的大小无关。Subversion 的分支和标签操作只是一种类似于硬链接的机制复制整个工程。因而，这些操作通常只会花费很少且相对固定的时间。

（8）可修改性。Subversion 没有历史负担，它以一系列优质的共享 C 程序库的方式实现，具有定义良好的 API。这使得 Subversion 非常容易维护，和其他语言的互操作性很强。

3.3　Subversion 的架构剖析

如图 3-1 所示，一端是保存所有版本数据的 Subversion 版本库，另一端是 Subversion 的客户程序，管理着所有版本数据的本地影射（称为“工作拷贝”），在这两极之间是各种各样的版本库访问（RA）层，某些通过网络服务器访问版本库，某些则绕过网络服务器直接访问版本库。

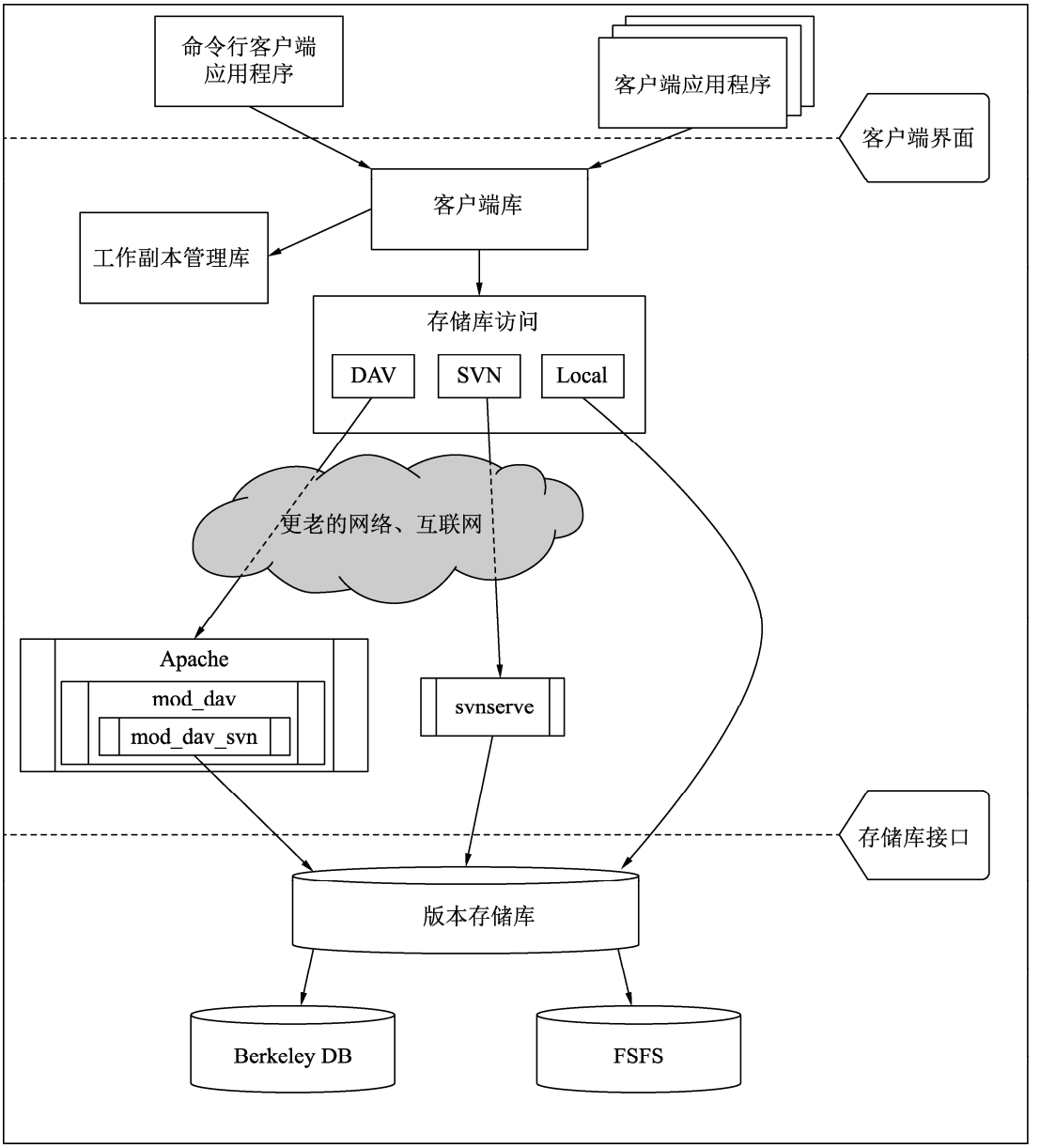

图 3-1　Subversion 拓扑结构图

Subversion 版本管理主要以文件变更列表的方式存储信息，Subversion 将保存的信息看作一组基本文件和每个文件随时间逐步累积的差异，如图 3-2 所示。

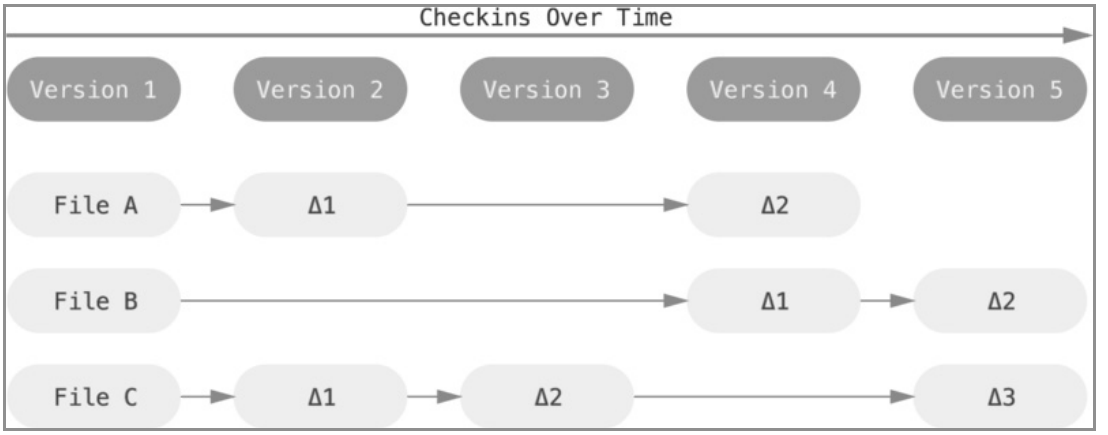

图 3-2　Subversion 版本管理

3.4　Subversion 的组件模块

Subversion 由以下几部分组成。

（1）svn。

命令行客户端程序。

（2）svnversion。

用来显示工作拷贝的状态（即当前项目的修订版本）。

（3）svnlook。

直接查看 Subversion 版本库的工具。

（4）svnadmin。

建立、调整和修复 Subversion 版本库的工具。

（5）svndumpfilter。

过滤 Subversion 版本库转储数据流的工具。

（6）mod_dav_svn。

Apache HTTP 服务器的插件，使版本库可以通过网络访问。

（7）svnserve。

单独运行的服务器程序，可以作为守护进程或由 SSH 调用。这是另一种使版本库可以通过网络访问的方式。

（8）svnsync。

通过网络增量镜像版本库的程序。

3.5 Subversion 分支概念剖析

假设你的工作是维护本公司一个部门的手册文档，一天，另一个部门需要近乎相同的手册，但一些地方会有区别，因为他们部分需求不同。

这种情况下你会怎样做？显而易见的方法是：做一个版本的拷贝，然后分别维护两个版本，需要时在对应的版本进行更改即可。

你也许希望在两个版本同时作修改，举个例子，你在第一个版本发现了一个拼写错误，很显然这个错误也会出现在第二个版本里。毕竟两份文档几乎相同，只有特定的微小区别。

这是分支的基本概念——正如它的名字，开发的一条线独立于另一条线，如果回顾历史，可以发现两条线分享共同的历史，一个分支总是从一个备份开始的，然后发展自己独有的历史。

Subversion 允许并行维护文件和目录的分支，允许通过复制数据建立分支。注意，分支互相联系，它从一个分支复制修改到另一个分支。最终，它可以让工作拷贝反映到不同的分支上，所以日常工作中可以“混合和比较”不同的开发线。

3.6 基于 YUM 构建 SVN 服务器

（1）基于 CentOS 7.x Linux，通过 YUM 安装 Subversion，操作指令如下：

```
yum install subversion -y
```

（2）建立版本库目录，操作指令如下：

```
mkdir -p /data/svn/
```

（3）建立 SVN 版本库，操作指令如下：

```
svnadmin create /data/svn
```

（4）修改版本库配置文件，在/data/svn/conf/svnserve.conf 文件中加入如下代码：

```
cat>/data/svn/conf/svnserve.conf<<EOF
[general]
#使非授权用户无法访问
anon-access = none
##使授权用户有写权限
```

```
auth-access = write
##指明密码文件路径
password-db = passwd
##访问控制文件
authz-db = authz
##认证命名空间,Subversion 会在认证提示里显示,并作为凭证缓存的关键字
realm = /data/svn
EOF
```

（5）配置用户及权限，修改/data/svn/conf/passwd，添加两个用户访问 SVN 服务端，操作方法和指令如下：

```
cat>/data/svn/conf/passwd<<EOF
[users]
jfedu1 = 123456
jfedu2 = 123456
EOF
```

（6）配置用户及权限，修改/data/svn/conf/authz，添加两个用户访问 SVN 服务端，操作方法和指令如下：

```
cat>/data/svn/conf/authz<<EOF
[/]
jfedu1 = rw
jfedu2 = rw
EOF
```

（7）启动 Subversion 服务并设置系统服务，操作指令如下：

```
svnserve -d -r /data/svn/ --listen-port=8001
```

（8）查看本地 8001 端口是否启动，操作指令如下：

```
netstat -ntl|grep 8001
```

（9）SVN 平台部署成功，可以在客户端使用 SVN 如下指令，所有操作如图 3-3 所示。

```
svn co svn://101.34.116.235:8001/
```

```
  Verifying  : pakchois-0.4-10.el7.x86_64
  Verifying  : gnutls-3.3.29-9.el7_6.x86_64

Installed:
  subversion.x86_64 0:1.7.14-16.el7

Dependency Installed:
  gnutls.x86_64 0:3.3.29-9.el7_6     neon.x86_64 0:0.30.0-4.el7
  pakchois.x86_64 0:0.4-10.el7       subversion-libs.x86_64 0:1

Complete!
```

（a）

图 3-3　在客户端使用 SVN 命令

```
[root@www-jfedu-net ~]# cat>/data/svn/conf/passwd<<EOF
> [users]
> jfedu1 = 123456
> jfedu2 = 123456
> EOF
[root@www-jfedu-net ~]# cat>/data/svn/conf/authz<<EOF
> [/]
> jfedu1 = rw
> jfedu2 = rw
> EOF
[root@www-jfedu-net ~]#
```

（b）

```
[root@www-jfedu-net ~]# ps -ef|grep -aiE svn
root     25037     1  0 16:10 ?        00:00:00 svnserve -d -r /data
root     25049 23751  0 16:10 pts/0    00:00:00 grep --color=auto -a
[root@www-jfedu-net ~]#
[root@www-jfedu-net ~]# netstat -tnlp|grep -aiwE 8001
tcp        0      0 0.0.0.0:8001            0.0.0.0:*
[root@www-jfedu-net ~]#
[root@www-jfedu-net ~]# svn co svn://101.34.116.235:8001/
Authentication realm: <svn://101.34.116.235:8001> /data/svn
Password for 'root':
Authentication realm: <svn://101.34.116.235:8001> /data/svn
Username: jfedu1
```

（c）

图 3-3 （续）

3.7 SVN 二进制+Apache 整合实战

（1）安装 httpd 软件包，操作指令如下：

```
yum install httpd httpd-devel mod_dav_svn -y
```

（2）检测 SVN 和 Apache 整合模块是否安装成功，操作指令如下：

```
ls -l/etc/httpd/modules/ | grep svn
```

（3）修改 httpd 主配置文件/etc/httpd/conf/httpd.conf，加入如下代码：

```
LoadModule dav_module modules/mod_dav.so
LoadModule dav_svn_module modules/mod_dav_svn.so
```

（4）修改 httpd 主配置文件，末行添加如下代码：

```
<Location /svn>
       DAV svn
```

```
        SVNPath /data/svn
        AuthType Basic
        AuthName "svn for project"
        AuthUserFile  /etc/httpd/conf/passwd
        AuthzSVNAccessFile /data/svn/conf/authz
        Satisfy all
        Require valid-user
</Location>
```

（5）生成 HTTP 访问密钥，操作指令如下：

```
htpasswd -c /etc/httpd/conf/passwd jfedu1
htpasswd /etc/httpd/conf/passwd jfedu2
```

（6）重启 Apache httpd 服务，操作指令如下：

```
service httpd restart
```

（7）如果没有权限，需要授权给/data/svn 目录，操作指令如下：

```
chown -R  apache /data/svn
```

（8）根据如上操作，SVN 和 Apache 整合成功，通过浏览器访问 SVN，输入用户名和密码，如图 3-4 所示。

图 3-4　浏览器访问 SVN

3.8　基于 MAKE 构建 SVN 服务器

（1）基于 CentOS 7.x Linux，通过 MAKE 源码编译安装 Subversion，操作指令如下：

```
#下载 SVN 相关软件包
wget https://archive.apache.org/dist/subversion/subversion-1.8.9.tar.bz2
```

```
wget http://www.sqlite.org/sqlite-amalgamation-3071502.zip
wget https://mirrors.cnnic.cn/apache/httpd/httpd-2.4.37.tar.gz
#解压 SVN 软件包
tar -xzvf subversion-1.8.9.tar.bz2
unzip sqlite-amalgamation-3071502.zip
mv sqlite-amalgamation-3071502  subversion-1.8.9/sqlite-amalgamation
#切换至 SVN 源代码包
cd subversion-1.8.9
#SVN 预编译（提前编译安装 Apache）
./configure --prefix=/usr/local/subversion --with-apxs=/usr/local/apache/
bin/apxs --enable-mod-activation
#Apache 预编译参数
#./configure --prefix=/usr/local/apache/ --enable--so --enable-dav
--enable-maintainer-mode --enable-rewrite
#编译
make
#安装
make install
```

（2）在/etc/profile 添加如下内容，操作指令如下：

```
cat>>/etc/profile<<EOF
export PATH = /usr/local/subversion/bin:\$PATH
EOF
source /etc/profile
```

（3）安装完成后，查看 SVN 版本信息是否已经安装，操作指令如下：

```
/usr/local/subversion/bin/svn --version
svn --version
```

（4）建立版本库目录，操作指令如下：

```
mkdir -p /data/svn/
```

（5）建立 SVN 版本库，操作指令如下：

```
svnadmin create /data/svn
```

（6）修改版本库配置文件，在/data/svn/conf/svnserve.conf 文件中加入如下代码：

```
cat>/data/svn/conf/svnserve.conf<<EOF
[general]
#使非授权用户无法访问
```

```
anon-access = none
##使授权用户有写权限
auth-access = write
##指明密码文件路径
password-db = passwd
##访问控制文件
authz-db = authz
##认证命名空间,Subversion 会在认证提示里显示,并作为凭证缓存的关键字
realm = /data/svn
EOF
```

（7）配置用户及权限，修改/data/svn/conf/passwd，添加两个用户访问 SVN 服务端，操作方法和指令如下：

```
cat>/data/svn/conf/passwd<<EOF
[users]
jfedu1 = 123456
jfedu2 = 123456
EOF
```

（8）配置用户及权限，修改/data/svn/conf/authz，添加两个用户访问 SVN 服务端，操作方法和指令如下：

```
cat>/data/svn/conf/authz<<EOF
[/]
jfedu1 = rw
jfedu2 = rw
EOF
```

（9）启动 Subversion 服务并设置系统服务，操作指令如下：

```
svnserve -d -r /data/svn/ --listen-port=8001
```

（10）查看本地 8001 端口有没有启动，操作指令如下：

```
netstat -ntl|grep 8001
```

（11）SVN 平台部署成功，可以在客户端使用 SVN 如下指令，所有操作如图 3-5 所示。

```
svn co svn://101.34.116.235:8001/
```

```
  Verifying  : pakchois-0.4-10.el7.x86_64
  Verifying  : gnutls-3.3.29-9.el7_6.x86_64

Installed:
  subversion.x86_64 0:1.7.14-16.el7

Dependency Installed:
  gnutls.x86_64 0:3.3.29-9.el7_6     neon.x86_64 0:0.30.0-4.el7
  pakchois.x86_64 0:0.4-10.el7       subversion-libs.x86_64 0:1

Complete!
```

（a）

```
[root@www-jfedu-net ~]# cat>/data/svn/conf/passwd<<EOF
> [users]
> jfedu1 = 123456
> jfedu2 = 123456
> EOF
[root@www-jfedu-net ~]# cat>/data/svn/conf/authz<<EOF
> [/]
> jfedu1 = rw
> jfedu2 = rw
> EOF
[root@www-jfedu-net ~]#
```

（b）

```
[root@www-jfedu-net ~]# ps -ef|grep -aiE svn
root     25037     1  0 16:10 ?        00:00:00 svnserve -d -r /data
root     25049 23751  0 16:10 pts/0    00:00:00 grep --color=auto -a
[root@www-jfedu-net ~]#
[root@www-jfedu-net ~]# netstat -tnlp|grep -aiwE 8001
tcp        0      0 0.0.0.0:8001            0.0.0.0:*
[root@www-jfedu-net ~]#
[root@www-jfedu-net ~]# svn co svn://101.34.116.235:8001/
Authentication realm: <svn://101.34.116.235:8001> /data/svn
Password for 'root':
Authentication realm: <svn://101.34.116.235:8001> /data/svn
Username: jfedu1
```

（c）

图 3-5　在客户端使用 SVN 命令

3.9　SVN 源码+Apache 整合实战

（1）复制 SVN 模块至 Apache modules 模块目录（如果 httpd.conf 配置文件存在，无须添加以下代码）。

```
cp /usr/local/subversion/libexec/mod_dav_svn.so /usr/local/apache/modules/
cp /usr/local/subversion/libexec/mod_authz_svn.so /usr/local/apache/modules/
```

（2）修改 httpd.conf 配置文件/usr/local/apache/conf/httpd.conf，添加如下代码：

```
LoadModule dav_module modules/mod_dav.so
LoadModule dav_svn_module modules/mod_dav_svn.so
```

（3）在 httpd.conf 配置文件末行添加如下代码：

```
<Location /svn>
       DAV svn
       SVNPath /data/svn
       AuthType Basic
       AuthName "svn for project"
       AuthUserFile  /data/svn/conf/.passwd
       AuthzSVNAccessFile /data/svn/conf/authz
       Satisfy all
       Require valid-user
</Location>
```

（4）生成 HTTP 访问密钥，操作指令如下：

```
/usr/local/apache/bin/htpasswd -c /data/svn/conf/.passwd jfedu1
/usr/local/apache/bin/htpasswd  /data/svn/conf/.passwd jfedu2
```

（5）重启 Apache httpd 服务，操作指令如下：

```
/usr/local/apache/bin/apachectl restart
```

（6）如果没有权限，需要授权给/data/svn 目录，操作指令如下：

```
chown -R  apache /data/svn
```

（7）SVN 和 Apache 整合成功，通过浏览器访问 SVN，输入用户名和密码，如图 3-6 所示。

图 3-6　通过浏览器访问 SVN

3.10 Subversion 客户端命令实战

作为运维人员，要想维护好 SVN 服务器，必须掌握常见的 SVN 操作指令，以下为常见的 SVN 操作指令和含义详解。

（1）迁出 SVN 代码至 www 目录，操作指令如下：

```
svn  co  svn://101.34.116.235:8001/ www
svn  co  http://106.12.133.186/svn/ www
```

（2）创建文件并添加和提交，如图 3-7 所示。

```
svn add jfedu1.txt jfedu2.txt
svn commit -m v1
```

```
[root@www-jfedu-net www]# touch jfedu1.txt
[root@www-jfedu-net www]#
[root@www-jfedu-net www]# touch jfedu2.txt
[root@www-jfedu-net www]#
[root@www-jfedu-net www]# svn add jfedu1.txt jfedu2.txt
A         jfedu1.txt
A         jfedu2.txt
[root@www-jfedu-net www]#
[root@www-jfedu-net www]# svn commit -m v1
Adding         jfedu1.txt
Adding         jfedu2.txt
Transmitting file data ..
Committed revision 3.
[root@www-jfedu-net www]#
```

图 3-7　SVN 服务器提交文件

（3）从 SVN 服务器更新软件至本地目录，如图 3-8 所示。

```
svn up
```

（4）增加本地所有修改，操作指令如下：

```
svn add . --no-ignore --force
```

（5）提交本地所有修改，操作指令如下，如图 3-9 所示。

```
svn commit . -m vv1 --force-log
```

```
Adding         jfedu2.txt
Transmitting file data ..
Committed revision 3.
[root@www-jfedu-net www]#
[root@www-jfedu-net www]# ls
jfedu1.txt  jfedu2.txt
[root@www-jfedu-net www]# rm -rf *
[root@www-jfedu-net www]#
[root@www-jfedu-net www]# svn up
Updating '.':
Restored 'jfedu1.txt'
Restored 'jfedu2.txt'
At revision 3.
[root@www-jfedu-net www]#
```

图 3-8　SVN 更新代码文件

```
[root@www-jfedu-net www]# svn commit . -m vv1
svn: E205005: The log message is a pathname (was -F intend
[root@www-jfedu-net www]#
[root@www-jfedu-net www]# svn commit . -m vv1 -F
svn: missing argument: F
Type 'svn help' for usage.
[root@www-jfedu-net www]# svn commit . -m vv1 --force-log
Adding         vv1
Adding         vv1/jfedu1.txt
Adding         vv1/jfedu2.txt
Transmitting file data ..
Committed revision 6.
[root@www-jfedu-net www]#
```

图 3-9　SVN 提交代码文件

（6）更多 SVN 常见命令如下：

```
#将文件 checkout 到本地目录
svn checkout path                    #path 是服务器上的目录
svn checkout svn://101.34.116.235:8001 www
svn co svn://101.34.116.235:8001/ www
#往版本库中添加新的文件
svn add file
svn add jfedu.txt                    #添加 jfedu.txt
svn add *.txt                        #添加当前目录下所有的 txt 文件
#将改动的文件提交到版本库
svn commit -m "LogMessage" [-N] [--no-unlock] PATH #如果选择了保持锁,就使用
                                                    #--no-unlock 开关
svn commit -m "add test file for my test" jfedu.txt
```

```
svn ci -m "add test file for my test" jfedu.txt
#SVN 加锁/解锁
svn lock -m "LockMessage" [--force] PATH
svn lock -m "lock test file" jfedu.txt
svn unlock PATH
#更新到某个版本
svn update -r m path
#svn update 如果后面没有目录,默认将当前目录以及子目录下的所有文件都更新到最新版本
svn update -r 200 jfedu.txt     #将版本库中的文件 jfedu.txt 还原到版本 200
svn update jfedu.txt    #更新,与版本库同步。如果在提交的时候提示过期,是因为冲突,
                        #需要先更新,修改文件,然后清除 svn resolved,最后再提交
                        #查看文件或者目录状态
#(1)
svn status path         #目录下的文件和子目录的状态,正常状态不显示
#【?:不在 svn 的控制中;M:内容被修改;C:发生冲突;A:预定加入版本库;K:被锁定】
#(2)
svn status -v path      #显示文件和子目录状态
#第一列保持相同,第二列显示工作版本号,第三和第四列显示最后一次修改的版本号和修改人
#注:svn status、svn diff 和 svn revert 这三条命令在没有网络的情况下也可以执行,原
#因是 SVN 在本地的.svn 文件中保留了本地版本的原始拷贝
#简写为 svn st
#SVN 删除文件
svn delete path -m "delete test fle"
#例如 svn delete svn://101.34.116.235:8001/pro/domain/jfedu.txt -m "delete
#test file"
#或者直接 svn delete jfedu.txt 然后再 svn ci -m 'delete test file',推荐使用
#简写为 svn (del, remove, rm)
#SVN 查看日志
svn log path
#例如,svn log jfedu.txt 显示这个文件的所有修改记录及其版本号的变化
#查看文件详细信息
svn info path
svn info jfedu.txt
#比较差异
svn diff path                   #将修改的文件与基础版本比较
svn diff jfedu.txt
svn diff -r m:n path            #对版本 m 和版本 n 比较差异
svn diff -r 200:201 jfedu.txt
svn di
#将两个版本之间的差异合并到当前文件
svn merge -r m:n path
```

```
svn merge -r 200:205 jfedu.txt  #将版本 200 与版本 205 之间的差异合并到当前文件,但
                                #是一般都会产生冲突,需要处理一下
#SVN 查看帮助
svn help
svn help ci
```

3.11　Svnserve.conf 配置参数剖析

Svnserve 是 SVN 服务器核心主配置文件，主要用于对 SVN 服务器进行访问权限、目录定义等的配置。该文件由一个[general]配置段组成，格式为<配置项>=<值>，常见参数详解如下。

（1）anon-access。

控制非鉴权用户访问版本库的权限。取值范围为 write、read 和 none，其中 write 为可读可写，read 为只读，none 表示无访问权限。默认值为 read。

（2）auth-access。

控制鉴权用户访问版本库的权限。取值范围为 write、read 和 none，其中 write 为可读可写，read 为只读，none 表示无访问权限。默认值为 write。

（3）password-db。

指定用户名口令文件名。除非指定绝对路径，否则文件位置为相对 conf 目录的相对路径。默认值为 passwd。

（4）authz-db。

指定权限配置文件名，通过该文件可以实现以路径为基础的访问控制。除非指定绝对路径，否则文件位置为相对 conf 目录的相对路径。默认值为 authz。

（5）Realm。

指定版本库的认证域，即在登录时提示的认证域名称。若两个版本库的认证域相同，建议使用相同的用户名口令数据文件。默认值为 UUID（Universal Unique IDentifier，全局唯一标示）。

在企业生产环境中，Svnserve.conf 配置文件案例代码如下：

```
[general]
#使非授权用户无法访问
anon-access = none
##使授权用户有写权限
auth-access = write
##指明密码文件路径
```

```
password-db = passwd
##访问控制文件
authz-db = authz
##认证命名空间,Subversion 会在认证提示里显示,并且作为凭证缓存的关键字
realm = /data/svn
```

3.12 Passwd 文件参数剖析

Passwd 是 SVN 用户名和密码配置文件,该文件由一个[users]配置段组成,格式为<用户名>=<口令> （注：口令为未经过任何处理的明文）。在企业生产环境中，Passwd 配置文件案例代码如下：

```
[users]
jfedu1 = 123456
jfedu2 = 123456
```

3.13 Authz 文件参数剖析

Authz 是 SVN 用户名和密码配置文件，该文件由[groups]配置段和若干版本库路径权限段组成，常见参数详解如下。

（1）[groups]配置段格式：<用户组>=<用户列表>。

用户列表由若干用户组或用户名构成，用户组或用户名之间用逗号“,”分隔，引用用户组时要使用前缀“@”。

（2）版本库路径权限段格式：[<版本库名>:<路径>]。

如版本库 abc 路径/tmp 的版本库路径权限段的段名为"[abc:/tmp]"，可省略段名中的版本库名。若省略版本库名，则该版本库路径权限段对所有版本库中相同路径的访问控制都有效，如[/tmp]。

版本库路径权限段中配置行格式有如下 3 种：

（1）<用户名> = <权限>；

（2）<用户组> = <权限>；

（3）* = <权限>。

其中，“*”表示任何用户；权限的取值范围为空值、r 和 rw，空值表示对该版本库路径无任

何权限，r 表示具有只读权限，rw 表示有读写权限。

在企业生产环境中，authz 配置文件案例代码如图 3-10 所示。

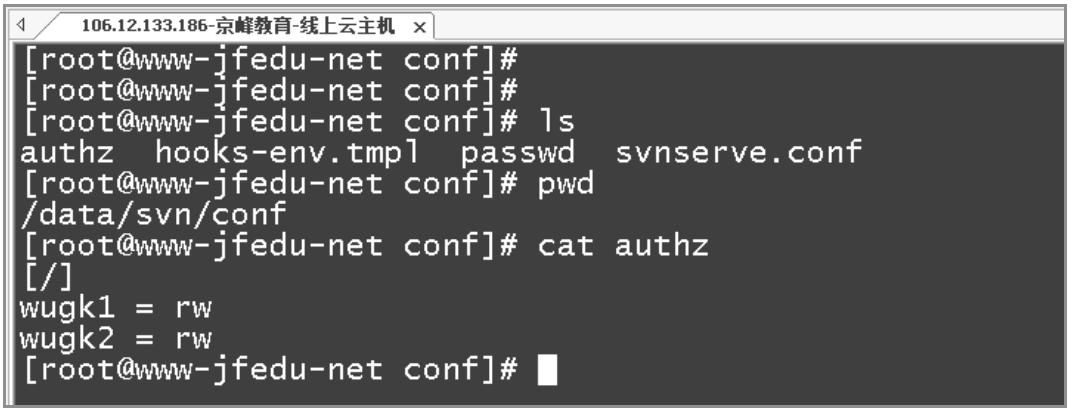

图 3-10　SVN 服务配置密码文件