**Morning：**

1. SVN概述
2. **SVN（Subversion）：版本控制系统**

-特点：记录每一次改变

-可以随时将文件、目录还原到早期旧版本；

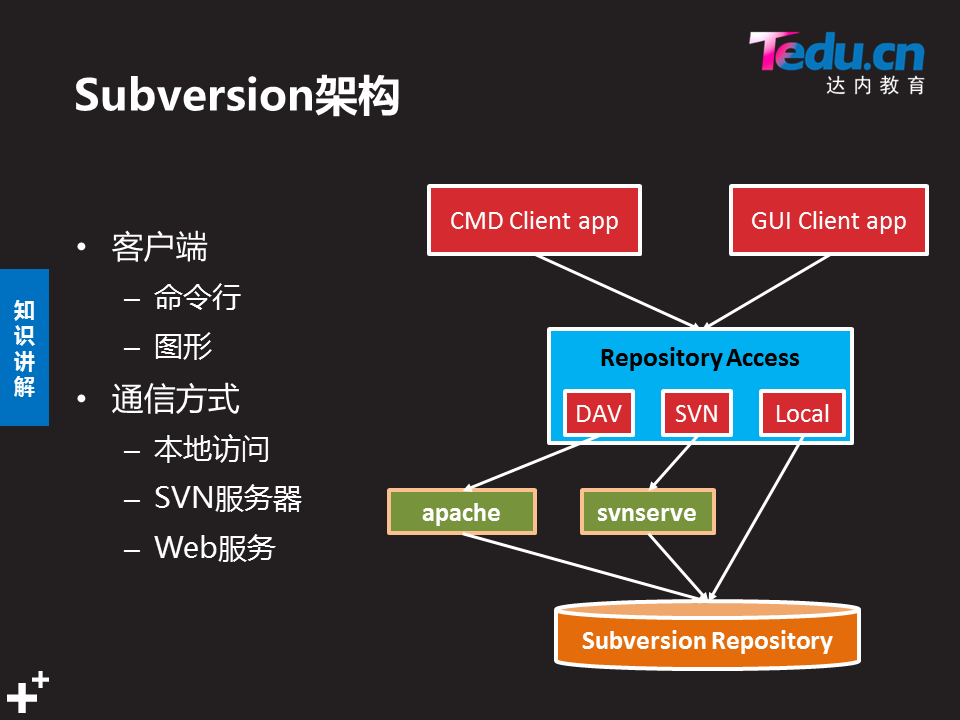
-记录数据修改的历史，便于检查

-允许和别人协作文档并跟踪所做的修改

**2）SVN架构**

-必须具备svn Repository版本仓库；

-可以通过http、SVN、本地访问



2、SVN基本操作

**步骤1：搭建Subversion服务器**

1. yum安装subversion软件

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

1. 创建版本仓库

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

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

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

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

1. 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 "Initdt"

#-m：注释，说明版本更新原因。导入后，以数据库密文的形式存放

4）修改svn配置文件，创建账户密码

注意：

-配置文件存放位置为svn仓库下的conf/，从右往左配置3个文件

-配置文件的开头不能有空格

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

anon-access = none #19行，匿名无任何权限

#如果保持为read，svn log则无任何信息

auth-access = write #20行，有效账户可读写

password-db = passwd #27行，账户密码配置文件

authz-db = authz #34行，ACL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 ~]# vim /var/svn/project/conf/authz

#作用：给某些特殊目录单独设置权限，无设置的默认继承/

[/] #/指的是svn仓库根目录

harry = rw

tom = rw

1. 启动svn服务

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

#-d启动后放后台，-r指定共享目录

#如果共享多个目录，svnserve -d -r /var/svn/

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

#查看3690端口

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）。

**步骤2：SVN客户端操作**

注意：对本地数据作增删、并提交到服务器，必须使用svn开头的指令

1）将服务器数据下载到本地

[root@web2 ~]# yum -y install subversion

[root@web2 ~]# cd /tmp

[root@web2 tmp]# svn --username harry --password 123456 co svn://192.168.2.100/ code

#服务器192.168.2.100上co下载代码到本地/tmp/code目录

#用户名harry,密码123456

1. 修改文件内容，上传到服务器

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

[root@web2 code]# vim user.slice

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

#将本地修改的数据同步到服务器

3）将服务器数据更新到本地

[root@web2 code]# svn update

[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

1. 本地新建文件，提交到服务器

[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

1. 创建目录、删除文件

[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" #上传代码

1. 查看文件的差异

[root@web2 code]# svn diff

#查看所有文件的差异，无差异输出为空

[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

1. 还原文件的内容

[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

8）还原已删除的文件

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

[root@web2 code]# svn update

9）还原到历史版本

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

#修改本地副本中的代码文件

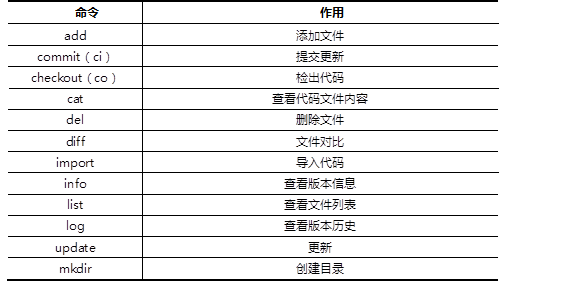
[root@web2 code]# svn ci -m "xxx"

#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

**-SVN命令列表**



Windows图形界面SVN：TortoiseSVN

**Afternoon：**

1、使用Subversion协同工作

**步骤1：多人协同工作**

1）A和B分别下载代码到本地

[root@web1 ~]# svn --user... co svn://192.168.2.100/ code

[root@web2 ~]# svn --user... co svn://192.168.2.100/ code

2）A和B修改不同的文件

[root@web1 code]# sed ... tmp.mount && svn ci -m ”tom ”

[root@web2 code]# sed ... umount.target && svn ci -m “harry”

[root@web2 code]# svn update

[root@web1 code]# svn update

#结果无任何冲突

3）A和B修改相同文件的不同行

[root@web1 code]# sed ...3a user.slice && svn ci -m ”tom”

[root@web2 code]# sed ...6a user.slice && svn ci -m “harry”

#后面用户提交失败，提示已经过期

[root@web2 code]# svn update && svn ci -m "modified"

[root@web1 code]# svn update

4）A和B修改相同文件的相同行

#后面用户提交失败，提示已经过期

[root@web2 code]# svn update #出现冲突，需要手动解决

#更新版本后，会把服务器版本与本地版本内容合并，提交成功

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

[root@web2 code]# ls tuned\*

tuned.service(两者比较) tuned.service.mine(我的版本) tuned.service.r10(他的版本) tuned.service.r9(修改前版本)

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

[root@web2 code]# rm -rf tuned.ser.r10 tuned.ser.r9

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

#线下确认后，留下正确版本文件，解决冲突

**步骤2：版本库数据的备份与还原**

1. 备份

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

2）还原

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

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

2、Github

https://github.com，面向公网的版本控制软件。

1. 登陆网站，点击sign up（注册）。
2. 填写注册信息（用户名，邮箱，密码）。
3. 选择初始化操作，完成后到邮箱中激活Github账户。

4）创建仓库、使用仓库

-点击Start a project

-填写项目（仓库）名称

-勾选初始化，点击创建

-Create new file（新建）、Upload file（上传 ）、Download（下载）

1. 命令行git操作

[root@pc001 ~]# yum -y install git

[root@pc001 ~]# git clone https://github.com/账户名/仓库名

#clone指令用于将服务器仓库中的资料打包下载到本地

[root@pc001 ~]# cd 仓库名称

[root@pc001 ~]# 任意修改文件，或新建文件

[root@pc001 ~]# git add .

#add添加新文件

[root@pc001 ~]# git commit -m "test"

[root@pc001 ~]# git push

#commit和push实现提交代码的功能

[root@pc001 ~]# git pull

#pull更新，类似于svn update(必须进入git目录)

3、制作Nginx的RPM包

**-什么是RPM?**

a）RPM是一个压缩包（将源码安装后的二进制文件压缩成rpm包）

b）包含描述信息

**步骤1：安装rpm-build软件并配置**

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 #执行命令后生成的目录结构

BUILD

BUILDROOT（源码安装的根）

RPMS（生成的RPM）

SOURCES（存放源码包）

SPECS（管理员配置文件）

SRPMS（spec+rpm包的打包）

1. 将源码包复制到SOURCES目录

[root@web1 ~]# cp nginx-1.12.2.tar.gz /root/rpmbuild/SOURCES/

1. 创建并修改SPEC配置文件

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

Name:nginx #必须与SOURCES软件名+版本一致

Version:1.12.2

Release: 10

Summary: Nginx is a web server software.

#Group: #yum组包。 yum grouplist

License:GPL #允许随意复制、修改、发布

URL: www.test.com

Source0:nginx-1.12.2.tar.gz #与SOURCES软件名一致

#BuildRequires:

#Requires:

%description

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

%post

useradd nginx #非必需操作：安装后脚本(创建账户)

%prep

%setup –q #自动解压源码包，并cd进入目录

%build

./configure

make %{?\_smp\_mflags}

%install

make install DESTDIR=%{buildroot}

#默认的根为/root/rpmbuild/BUILDROOT

%files

%doc

/usr/local/nginx/\*

#对哪些文件与目录打包成RPM，依据./configure的安装路径

%changelog

**步骤2：使用配置文件创建RPM包**

1. 安装依赖包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 ~]# rpm -qpi RPMS/x86\_64/nginx-1.12.2-10.x86\_64.rpm

#-p未安装的包，-qi RPM的详细描述

[root@web1 ~]# rpm -qpl nginx-1.12.2-10.x86\_64.rpm

#-ql RPM的安装清单

**步骤3：安装软件**

[root@web1 ~]# rpm -ivh RPMS/x86\_64/nginx-1.12.2-10.x86\_64.rpm

#执行安装，安装在/usr/local/nginx/下