

# NSD OPERATION DAY06

1. [案例1：Subversion基本操作](#)
2. [案例2：使用Subversion协同工作](#)
3. [案例3：制作nginx的RPM包](#)

## 1 案例1：Subversion基本操作

### 1.1 问题

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

### 1.2 方案

使用YUM安装subversion软件，使用svn客户端工具连接svnserver服务器并测试版本控制软件。

### 1.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：安装Subversion服务器

##### 1) YUM安装subversion软件

- ```
01. [root@web1 ~] # yum -y install subversion
02. [root@web1 ~] # rpm -q subversion
```

##### 2)创建版本库

- ```
01. [root@web1 ~] # mkdir /var/svn/
02. [root@web1 ~] # svnadmin create /var/svn/project
03. [root@web1 ~] # ls /var/svn/project/
04. conf/ db/ format hooks/ locks/ README.txt
```

##### 3) 本地导入初始化数据

- ```
01. [root@web1 ~] # cd /usr/lib/systemd/system/
```

[Top](#)

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

#### 4) 修改配置文件，创建账户与密码

所有配置文件，要求顶头写，开头不要有空格。

```
01. [root@web1 ~]# vim /var/svn/project/conf/svnserve.conf
02. [general]
03. ### These options control access to the repository for unauthenticated
04. ### and authenticated users. Valid values are "write", "read",
05. ### and "none". The sample settings below are the defaults.
06. anon-access = none
07. //19行，匿名无任何权限
08. auth-access = write
09. //20行，有效账户可写
10. password-db = passwd
11. //27行，密码文件
12. authz-db = authz
13. //34行，ACL访问控制列表文件
14.
15. [root@web1 ~]# vim /var/svn/project/conf/passwd
16. ... ..
17. [users]
18. harry = 123456
19. //用户名和密码
20. tom = 123456
21. //用户名和密码
22.
23. [root@web1 ~]# cat /var/svn/project/conf/authz
24. [/] //定义ACL访问控制
25. harry = rw //用户对项目根路径可读可写
26. tom = rw
```

#### 5) 启动服务

```
01. [root@web1 ~]# svnserve -d -r /var/svn/project
02. [root@web1 ~]# netstat -nltlp | grep svnserve
03. tcp      0      0 0.0.0.0:3690 0.0.0.0:* LISTEN    4043/svnserve
```

[Top](#)

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`）。

## 步骤二：客户端测试(192.168.2.200)

### 1) 将服务器上的代码下载到本地

```

01. [root@web2 ~] # cd /tmp
02. [root@web2 ~] # svn --username harry --password 123456 \
03. co svn://192.168.2.100/ code
04. //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
05. //用户名harry,密码123456
06.
07. Store password unencrypted (yes/no)? yes //提示是否保存密码
08.
09. [root@web2 ~] # cd /tmp/code
10. [root@web2 code] # ls
11. [root@web2 code] # vim user.slice //挑选任意文件修改其内容
12. [root@web2 code] # svn ci -m "modify user" //将本地修改的数据同步到服务器
13.
14. [root@web2 code] # svn update //将服务器上新的数据同步到本地
15. [root@web2 code] # svn info svn://192.168.2.100 //查看版本仓库基本信息
16. [root@web2 code] # svn log svn://192.168.2.100 //查看版本仓库的日志
17.
18. [root@web2 code] # echo "test" > test.sh //本地新建一个文件
19. [root@web2 code] # svn ci -m "new file" //提交失败,该文件不被svn管理
20. [root@web2 code] # svn add test.sh //将文件或目录加入版本控制
21. [root@web2 code] # svn ci -m "new file" //再次提交,成功
22.
23. [root@web2 code] # svn mkdir subdir //创建子目录
24. [root@web2 code] # svn rm timers.target //使用svn删除文件
25. [root@web2 code] # svn ci -m "xxx" //提交一次代码
26.
27. [root@web2 code] # vim umount.target //任意修改本地的一个文件
28. [root@web2 code] # svn diff //查看所有文件的差异
29. [root@web2 code] # svn diff umount.target //仅查看某一个文件的差异
30. [root@web2 code] # svn cat svn://192.168.2.100/reboot.target //查看服务器文件的p
31.
32. [root@web2 code] # sed -i 'd' tmp.mount
33. //删除文件所有内容,但未提交
34. [root@web2 code] # svn revert tmp.mount
35. //还原tmp.mount文件

```

[Top](#)

```
36.
37. [root@web2 code] # rm -rf *.target
38. //任意删除若干文件
39. [root@web2 code] # svn update
40. //还原
41.
42. [root@web2 code] # sed -i '1a #test###' tuned.service
43. //修改本地副本中的代码文件
44. [root@web2 code] # svn ci -m "xxx"
45. //提交代码
46. [root@web2 code] # svn merge -r7:2 tuned.service
47. //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

| 命令              | 作用       |
|-----------------|----------|
| add             | 添加文件     |
| commit ( ci )   | 提交更新     |
| checkout ( co ) | 检出代码     |
| cat             | 查看代码文件内容 |
| del             | 删除文件     |
| diff            | 文件对比     |
| import          | 导入代码     |
| info            | 查看版本信息   |
| list            | 查看文件列表   |
| log             | 查看版本历史   |
| update          | 更新       |
| mkdir           | 创建目录     |

2 案例2：使用Subversion协同工作

2.1 问题

沿用练习一，通过svn工具，对subversion版本库进行多人协同工作测试，要求如下：

- 该版本库支持多个账户同时协作编辑文件
- 测试演示多人协作编辑的具体操作
- 手动解决版本冲突问题
- 备份版本库数据
- 注册使用Github

2.2 方案

[Top](#)

使用svn客户端工具连接subversion服务器并测试多人协同工作以及如何手动解决冲突问题，账户名称分别为harry和tom，最后使用svnadmin dump指令对版本库进行备份工作。

## 2.3 步骤

实现此案例需要按照如下步骤进行。

### 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！

```
01. [ root@web1 ~] # cd /tmp
02. [ root@web1 ~] # svn -- username tom -- password 123456 \
03. > co svn://192.168.2.100/ code
04. [ root@web2 ~] # cd /tmp
05. [ root@web2 ~] # svn -- username harry -- password 123456 \
06. > co svn://192.168.2.100/ code
07. [ root@web1 ~] # cd code
08. [ root@web2 ~] # cd code
```

2) harry和tom修改不同的文件

```
01. [ root@web1 my code] # sed -i "3a ###tom modify #####" tmp.mount
02. [ root@web1 my code] # svn ci -m "has modified"
03. [ root@web2 my code] # sed -i "3a ###harry modify #####" umount.target
04. [ root@web2 my code] # svn ci -m "has modified"
05. [ root@web2 my code] # svn update
06. [ root@web1 my code] # svn update
```

3) harry和tom修改相同文件的不同行

```
01. [ root@srv5 ~] # cd harry
02. [ root@web1 my code] # sed -i "3a ###tom modify #####" user.slice
03. [ root@web1 my code] # svn ci -m "modified"
04. [ root@web2 my code] # sed -i "6a ###harry modify #####" user.slice
05. [ root@web2 my code] # svn ci -m "modified" //提交失败
06. Sending      svnserve
07. Transmitting file data .svn: Commit failed (details follow):
08. svn: File '/user.slice' is out of date (过期)
09. [ root@web2 my code] # svn update //提示失败后，先更新再提交即可
10. [ root@web2 my code] # svn ci -m "modified" //提交成功
11. Sending      user.slice
```

[Top](#)

## 12. Transmitting file data .

## 4) harry和tom修改相同文件的相同行

```

01. [ root@web1 my code] # sed -i '1c [UNIT]' tuned.service
02. [ root@web1 my code] # svn ci -m "modified"
03. [ root@web2 my code] # sed -i '1c [unit]' tuned.service
04. [ root@web2 my code] # svn ci -m "modified"
05. Sending tuned.service
06. Transmitting file data .svn: Commit failed (details follow):
07. svn: File '/tuned.service' is out of date(过期)
08. [ root@web2 my code] # svn update //出现冲突，需要解决
09. Conflict(冲突) discovered in 'tuned.service'.
10. Select: (p) postpone, (df) diff-full, (e) edit,
11.          (mc) mine-conflict, (tc) theirs-conflict,
12.          (s) show all options: p //选择先标记p，随后解决
13. [ root@web2 my code] # ls
14. tuned.service tuned.service.mine tuned.service.r10 tuned.service.r9
15. [ root@web2 my code] # mv tuned.service.mine tuned.service
16. [ root@web2 my code] # rm -rf tuned.service.r10 tuned.service.r9
17. [ root@web2 my code] # svn ci -m "modified" //解决冲突

```

## 步骤二：使用dump指令备份版本库数据

```

01. [ root@web1 ~] # svnadmin dump /var/svn/project > project.bak //备份
02. * Dumped revision 0.
03. * Dumped revision 1.
04. * Dumped revision 2.
05. * Dumped revision 3.
06. * Dumped revision 4.
07. * Dumped revision 5.
08. * Dumped revision 6.
09. * Dumped revision 7.
10. * Dumped revision 8.
11. * Dumped revision 9.
12. * Dumped revision 10.
13. * Dumped revision 11. Top
14. [ root@web1 ~] # svnadmin create /var/svn/project2 //新建空仓库
15. [ root@web1 ~] # svnadmin load /var/svn/project2 < project.bak //还原

```

### 步骤三：注册使用Github

1. 登陆网站<https://github.com>，点击Sign up（注册），如图-1所示。

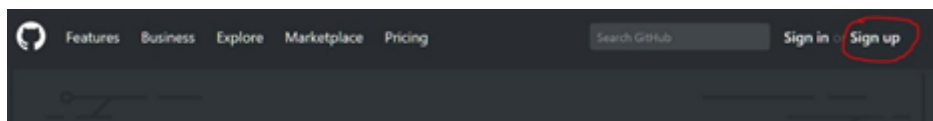


图-1

2. 填写注册信息（用户名，邮箱，密码），如图-2所示。

A screenshot of the 'Create your personal account' form on GitHub. The form has three input fields: 'Username' with the value 'testthera', 'Email address' with the value 'testthera11@163.com', and 'Password' with masked characters. Each field has a green checkmark on the right. Below the fields, there is a green button labeled 'Create an account' which is circled in red. The form also includes a disclaimer about terms of service and privacy.

图-2

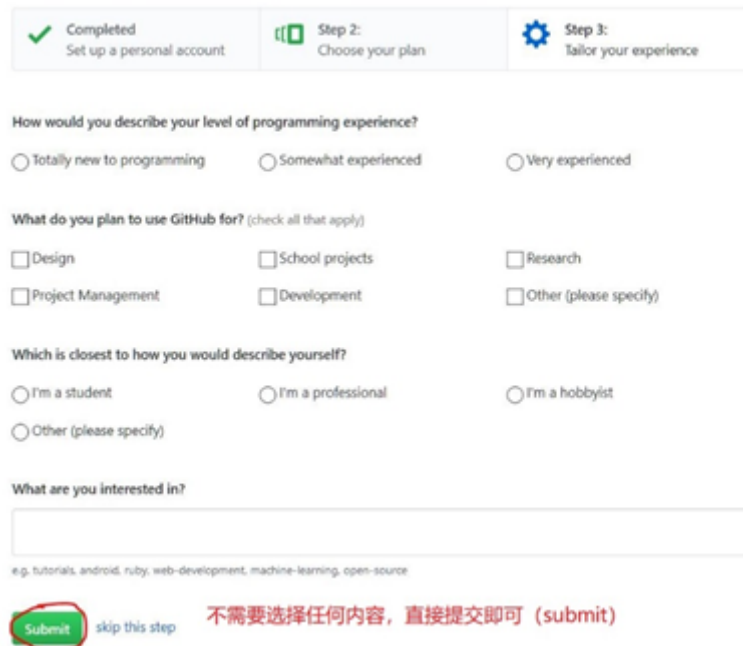
3. 初始化操作，如图-3和图-4所示。

A screenshot of the 'Welcome to GitHub' page. It shows a progress bar with 'Step 1: Set up a personal account' completed and 'Step 2: Choose your plan' in progress. Under 'Choose your plan', there are two radio button options: 'Unlimited public repositories for free.' (selected and circled in red) and 'Unlimited private repositories for \$7/month. (view in CNY)'. Below the plan selection, there are checkboxes for 'Help me set up an organization next' and 'Send me updates on GitHub news, offers, and events'. At the bottom, there is a green button labeled 'Continue' which is circled in red.

图-3

[Top](#)

You'll find endless opportunities to learn, code, and create, @testthera.



The form shows a progress bar with three steps: Step 1 (Completed: Set up a personal account), Step 2 (Choose your plan), and Step 3 (Tailor your experience). Below the progress bar are several questions with radio button and checkbox options. The questions are: 'How would you describe your level of programming experience?' (Totally new to programming, Somewhat experienced, Very experienced), 'What do you plan to use GitHub for? (check all that apply)' (Design, School projects, Research, Project Management, Development, Other (please specify)), 'Which is closest to how you would describe yourself?' (I'm a student, I'm a professional, I'm a hobbyist, Other (please specify)), and 'What are you interested in?' (with a text input field and examples like tutorials, android, ruby, web-development, machine-learning, open-source). At the bottom, there is a 'Submit' button circled in red, a 'skip this step' link, and a red text note: '不需要选择任何内容，直接提交即可 (submit)'.

图-4

注意，初始化完成后，到邮箱中去激活Github账户。

#### 4. 创建仓库、使用仓库

点击Start a project (如图-5所示)，

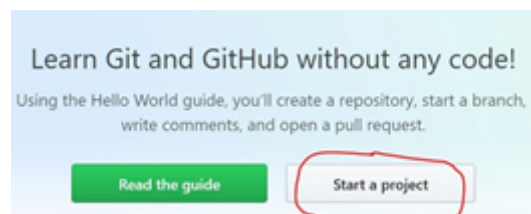
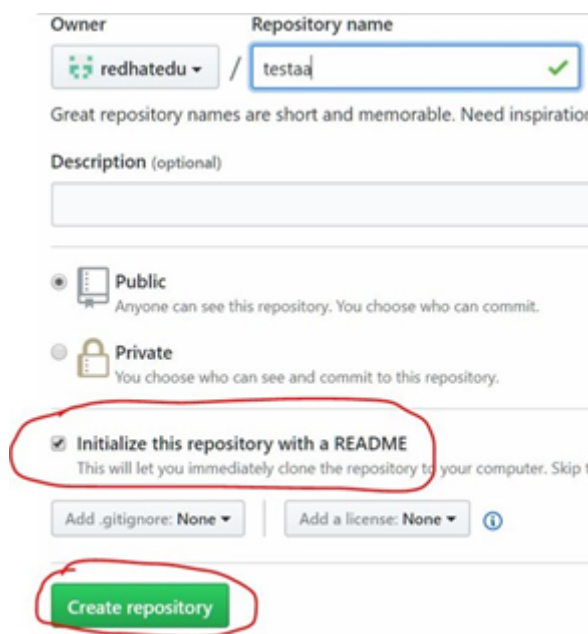


图-5

填写项目名称（项目名称任意），如图-6所示。



The form shows the 'Create repository' process. The 'Owner' is 'redhatedu' and the 'Repository name' is 'testaa', which is circled in red and has a green checkmark. Below the name is a note: 'Great repository names are short and memorable. Need inspiration'. The 'Description (optional)' field is empty. The 'Public' option is selected. Below the visibility options, the checkbox 'Initialize this repository with a README' is checked and circled in red. Below this are two dropdown menus: 'Add .gitignore: None' and 'Add a license: None'. At the bottom, the 'Create repository' button is circled in red.

[Top](#)

图-6



往仓库中上传文件或新建文件，如图-7所示

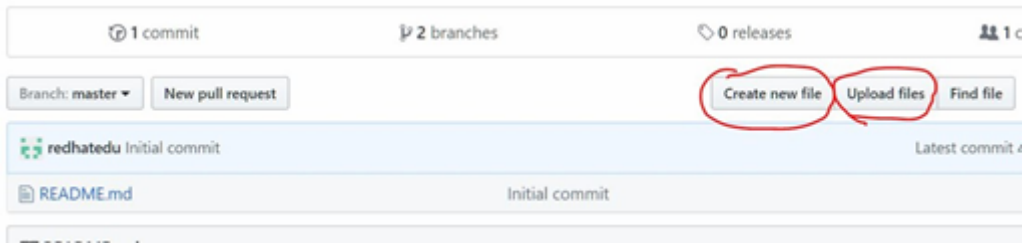


图-7

下载仓库中的代码，如图-8所示。

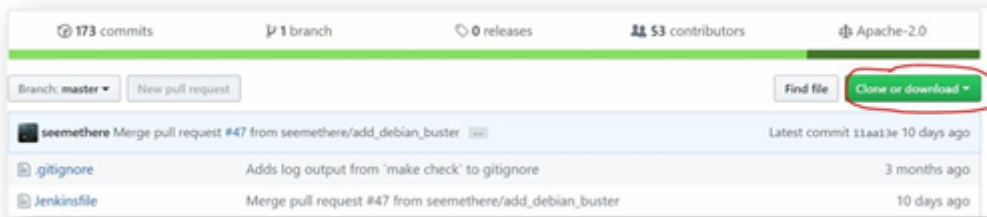


图-8

5. 命令行操作（需要联网的主机，如真实机）

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

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

#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

## 3 案例3：制作nginx的RPM包

### 3.1 问题

本案例使用nginx-1.12.2版本的源码软件，生成对应的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

### 3.2 方案

安装rpm-build软件包，编写SPEC配置文件，创建新的RPM软件包。

[Top](#)

配置文件中的描述信息如表-2：

表 - 2 SPEC描述信息

| 选项            | 值                                                          |
|---------------|------------------------------------------------------------|
| Name          | Nginx                                                      |
| Version       | 1.12.2                                                     |
| Release       | 1                                                          |
| Summary       | Nginx is a web server software.                            |
| License       | GPL                                                        |
| URL           | www.nginx.org                                              |
| Source0       | nginx-1.12.2.tar.gz                                        |
| BuildRequires | gcc pcre-devel zlib-devel openssl-devel                    |
| %description  | nginx [engine x] is an HTTP and reverse proxy server... .. |

### 3.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：安装rpm-build软件

##### 1) 安装rpm-build软件包

```
01. [root@web1 ~]# yum -y install rpm-build
```

##### 2) 生成rpmbuild目录结构

```
01. [root@web1 ~]# rpmbuild -ba nginx.spec           //会报错，没有文件或目录
02. [root@web1 ~]# ls /root/rpmbuild                //自动生成的目录结构
03. BUILD BUILDROOT RPMS SOURCES SPECS SRPMS
```

##### 3) 准备工作，将源码软件复制到SOURCES目录

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

##### 4) 创建并修改SPEC配置文件

```
01. [root@web1 ~]# vim /root/rpmbuild/SPECS/nginx.spec
02. Name: nginx
03. Version: 1.12.2
04. Release: 10
```

[Top](#)

```

05.  Summary : Nginx is a web server software.
06.
07.  License: GPL
08.  URL:   www.test.com
09.  Source0: nginx- 1.12.2.tar.gz
10.
11.  #BuildRequires:
12.  #Requires:
13.
14.  %description
15.  nginx [ engine x] is an HTTP and reverse proxy server.
16.
17.  %post
18.  useradd nginx                //非必需操作：安装后脚本( 创建账户)
19.
20.  %prep
21.  %setup - q                  //自动解压源码包，并cd进入目录
22.
23.  %build
24.  ./configure
25.  make %?_smp_mflags}
26.
27.
28.  %install
29.  make install DESTDIR=%{ buildroot}
30.
31.
32.  %files
33.  %doc
34.  /usr/local/nginx/*        //对哪些文件与目录打包
35.
36.  %changelog

```

## 步骤二：使用配置文件创建RPM包

### 1 ) 安装依赖软件包

```
01.  [ root@web1 ~] #yum -y install gcc pcre-devel openssl-devel
```

[Top](#)

### 2 ) rpmbuild创建RPM软件包

```

01. [ root@web1 ~] # rpmbuild - ba /root/rpmbuild/SPECS/nginx.spec
02. [ root@web1 ~] # ls /root/rpmbuild/RPMS/x86_64/nginx- 1.12.2- 10.x86_64.rpm
03. [ root@web1 ~] # rpm - qpi RPMS/x86_64/nginx- 1.12.2- 10.x86_64.rpm
04. Name      : nginx      Relocations: ( not relocatable)
05. Version   : 1.12.2      Vendor: ( none)
06. Release   : 10          Build Date: Mon 02 May 2016 02: 30: 53 AM PDT
07. Install Date: ( not installed)      Build Host: localhost
08. Group      : Applications/Internet      Source RPM: nginx- 1.8.0- 1.src.rpm
09. Size       : 721243          License: GPL
10. Signature  : ( none)
11. URL        : www.nginx.org
12. Summary    : Nginx is a web server software.
13. Description :
14. nginx [ engine x] is an HTTP and reverse proxy server.
15. [ root@web1 ~] # rpm - qpl nginx- 1.12.2- 10.x86_64.rpm
16. /usr
17. /usr/local
18. /usr/local/nginx
19. /usr/local/nginx/conf
20. /usr/local/nginx/conf /fastcgi.conf
21. /usr/local/nginx/conf /fastcgi.conf .def ault
22. /usr/local/nginx/conf /fastcgi_params
23. /usr/local/nginx/conf /fastcgi_params.def ault
24. /usr/local/nginx/conf /koi- utf
25. /usr/local/nginx/conf /koi- win
26. /usr/local/nginx/conf /mime.ty pes
27. /usr/local/nginx/conf /mime.ty pes.def ault
28. /usr/local/nginx/conf /nginx.conf
29. /usr/local/nginx/conf /nginx.conf .def ault
30. /usr/local/nginx/conf /scgi_params
31. /usr/local/nginx/conf /scgi_params.def ault
32. /usr/local/nginx/conf /uwsgi_params
33. /usr/local/nginx/conf /uwsgi_params.def ault
34. /usr/local/nginx/conf /win- utf
35. /usr/local/nginx/html
36. /usr/local/nginx/html/50x.html
37. /usr/local/nginx/html/index.html
38. /usr/local/nginx/logs
39. /usr/local/nginx/sbin
40. /usr/local/nginx/sbin/nginx

```

[Top](#)

### 步骤三：安装、卸载软件

01. [root@web1 ~] # rpm -ivh RPMS/x86\_64/nginx-1.12.2-10.x86\_64.rpm
02. [root@web1 ~] # rpm -qa | grep nginx
03. [root@web1 ~] # /usr/local/nginx/sbin/nginx
04. [root@web1 ~] # curl http://127.0.0.1/

[Top](#)