operation day 07

================================================

**git服务的使用方式：**

**1，使用ssh方式**

[root@web1 ~]# git init /var/lib/git/project --bare //如果web1没有仓

库就创建

[root@web2 ~]# ssh-keygen -f /root/.ssh/id\_rsa -N '' //在客户端web2

创建秘钥秘钥，且无密码

[root@web2 ~]# ssh-copy-id 192.168.2.100 //传入git服务端

cd project //进入仓库

echo abc > abc.txt //创建测试文件

git add . //提交到暂存区

git commit -m "abc" //提交到仓库保存

git push //推送到git服务器，也不需要密码了

2,使用git协议

[root@web1 ~]# yum -y install git-daemon

[root@web1 ~]# vim /usr/lib/systemd/system/git@.service

--enable=receive-pack //在第7行末尾添加这段, 可以实现对仓库

的写入,否则是只读

[root@web1 ~]# systemctl start git.socket //启动服务

关闭防火墙与selinux

[root@web1 ~]# chmod -R 777 /var/lib/git/

[root@web2 ~]# git clone git://192.168.2.100/project //使用git协议方式克隆仓库

cd project //进入仓库

echo xyz > xyz //创建测试文件

git add . //提交到暂存区

git commit -m "xyz" //提交到本地仓库

git push //推送到远程服务器

3, 使用http 访问git , 只读

[root@web1 ~]# yum -y install httpd gitweb

[root@web1 ~]# vim +10 /etc/gitweb.conf //去掉第10行的注释

[root@web1 ~]# systemctl restart httpd

<http://192.168.2.100/git/> //使用浏览器查看仓库上级目录

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

**制作rpm**

**rpm包 安装便利 更新慢**

**源码包 安装繁琐 更新快**

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

[root@web1 ~]# rpmbuild -ba nginx.spec //制作rpm包，但是没有配置文件

会报错，报错也需要敲，会产生所需的目录

[root@web1 ~]# cp lnmp\_soft/nginx-1.17.6.tar.gz rpmbuild/SOURCES/ //拷贝

源码包到指定目录

[root@web1 ~]# vim rpmbuild/SPECS/nginx.spec //编写转换nginx为rpm的配置文件

Name:nginx //软件包名

Version:1.17.6 //软件包版本号

Release:1 //发布rpm包的版本号

Summary:test //简单描述

#Group:

License:GPL //授权协议

URL:www.abc.com //网站地址

Source0:nginx-1.17.6.tar.gz //源码包完整文件名

#BuildRequires:

#Requires:

%description //详细描述

test test test

%post //可选项,分配额外任务

useradd nginx -s /sbin/nologin //安装rpm的时候创建nginx用户

yum -y install pcre-devel openssl-devel

%prep

%setup -q

%build

./configure --user=nginx //修改% , --user=nginx是配置过程中添加的选项,不需要的话可以不写,包括需要其他模块的话也可以添加

make %{?\_smp\_mflags}

%install

make install DESTDIR=%{buildroot}

%files

%doc

/usr/local/nginx/\* //定义打包的文件所在目录

%changelog

[root@web1 ~]# yum -y install gcc pcre-devel openssl-devel //安装nginx依赖

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

rpm -qpi rpmbuild/RPMS/x86\_64/nginx-1.17.6-1.x86\_64.rpm //查看rpm

包信息

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

**vpn 虚拟专用网**

1：使用gre技术实现vpn(两台linux)

Web1主机配置：

modprobe ip\_gre //在内核中开启gre模块功能

[root@web1 ~]# lsmod | grep ip\_gre //查询

ip tunnel add tun0 mode gre remote 192.168.2.200 local 192.168.2.100 //创

建vpn隧道，名字叫tun0，使用gre技术，与2.200连接，自身ip是2.100

[root@web1 ~]# ip addr show //查看

[root@web1 ~]# ip tunnel del tun0 //如果配置错误,要删除重来

ip addr add 10.10.10.10/8 peer 10.10.10.5/8 dev tun0 //在tun0隧道

中使用私有ip地址，本机是10.10，远程主机是10.5

[root@web1 ~]# ip link set tun0 up //激活之前的配置

**web2主机也按照上述配置创建vpn，但是ip地址要反着写**

测试：

selinux、防火墙都要关闭，相互ping 10网段的ip

2, 使用pptp搭建vpn (支持windows)

cd ~/lnmp\_soft/vpn

yum -y install pptpd-1.4.0-2.el7.x86\_64.rpm //安装软件包

[root@web1 vpn]# vim /etc/pptpd.conf //修改102、103行

localip 192.168.2.100 //本机ip,如果是实际环境要写公网ip

remoteip 10.10.10.10-18 //客户机分配的ip范围,自定义

[root@web1 vpn]# vim /etc/ppp/options.pptpd //修改66行，添加dns

地址，此处的dns地址可以是公共网络中任意一个可用dns即可

ms-dns 8.8.8.8

[root@web1 vpn]# vim /etc/ppp/chap-secrets //添加用户tom和密码123456

tom \* 123456 \*

systemctl restart pptpd //开启服务

[root@web1 vpn]# ss -ntulp | grep pptpd //检查

3, 使用l2tp+IPsec 配置vpn ,相比pptp安全性更好

[root@web1 vpn]# yum -y install libreswan

[root@web1 vpn]# cp myipsec.conf /etc/ipsec.d/

[root@web1 vpn]# vim /etc/ipsec.d/ myipsec.conf //修改第16vim行为本机ip

left=192.168.2.100

[root@web1 vpn]# vim /etc/ipsec.secrets

192.168.2.100 %any: PSK "randpass" //2.100是本机ip，%any:是任何客

户端可以连接本机，PSK预共享秘钥，密码是randpass

[root@web1 vpn]# yum - y install xl2tpd-1.3.8-2.el7.x86\_64.rpm //安装l2tp软件包

[root@web1 vpn]# vim /etc/xl2tpd/xl2tpd.conf

ip range = 9.9.9.9-9.9.9.18 //分配给客户的ip

local ip = 192.168.2.100 //vpn服务器本身的ip

[root@web1 vpn]# vim /etc/ppp/options.xl2tpd //修改配置，把10与16行注释

掉，21行删除注释与空格

[root@web1 vpn]# vim /etc/ppp/chap-secrets //如果添加账户，修改该文件

[root@web1 vpn]# systemctl start ipsec

[root@web1 vpn]# systemctl start xl2tpd

[root@web1 vpn]# ss -ntulp | grep xl2tpd

[root@web1 vpn]# ss -ntulp | grep :500

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

**配置Unit文件实现自定义systemctl控制的服务**

**在linux系统中，有些程序无法利用systemctl控制，就需要添加Unit文件**

1. **使用一个脚本测试**

[root@web1 opt]# cat test.sh

#!/bin/bash

while :

do

echo abc

echo xyz

sleep 1

done

cd /usr/lib/systemd/system

cp crond.service test.service //拷贝某个其他服务的配置文件作为模板

vim test.service

[Unit]

Description=test //描述

After=time-sync.target //在哪个服务启动之后再启动test服务，此处如果是

复制于其他服务的配置可以保持不变，目前仅测试test.sh脚本

[Service]

ExecStart=/opt/test.sh //如果执行了systemctl start test之后就运行/opt/test.sh程

序，提前要设置x权限

ExecReload=/bin/kill -HUP $MAINPID //如果执行了systemctl reload test 之后，就利

用kill程序发送不停止服务仅仅更新配置的信号，其效果相当于重新加载配置

KillMode=process //如果关闭服务，就停止主进程，比如在输入systemctl stop test 时。

[Install]

WantedBy=multi-user.target //支持开机自启

上述文件修改好之后保存退出 ,再执行systemctl daemon-reload 刷新

systemctl start test //已经可以控制test脚本（服务）

**2，配置systemctl管理nginx服务**

cp httpd.service nginx.service //拷贝某个其他服务的配置文件作为模板

vim nginx.service //修改

[Unit]

Description=nginx //描述

After=network.target remote-fs.target nss-lookup.target //在网络服务、远程文件系统服务、域名相关服务启动之后再开启nginx

[Service]

Type=forking //由于nginx是多进程服务，要设置forking

ExecStart=/usr/local/nginx/sbin/nginx //当执行systemctl start nginx时，所执行的命令

ExecReload=/usr/local/nginx/sbin/nginx -s reload //当执行systemctl reload nginx时，所执行的命令

ExecStop=/bin/kill -s QUIT ${MAINPID} //当执行systemctl stop nginx时，所执行的命令,-s QUIT是发送退出信号，${MAINPID}是nginx的主进程号

[Install]

WantedBy=multi-user.target //支持开机自启

[root@web1 system]# systemctl start nginx //开启nginx如果不好用，就执行systemctl daemon-reload命令或者重启服务器

[root@web1 system]# systemctl status nginx //查看状态