第二课：搭建企业私有Git服务

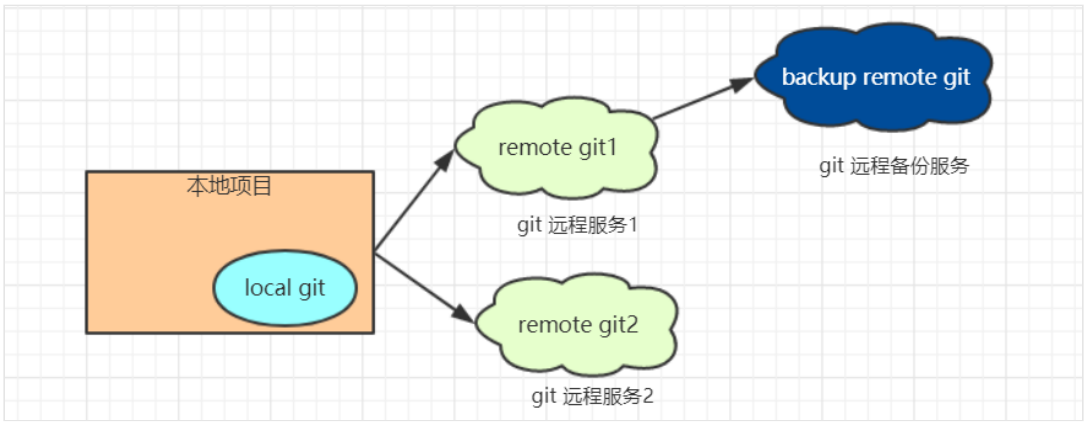
概要:

1. GIT远程通信协议详解
2. 基于gogs 搭建WEB管理服务

讲师介绍

一、GIT服务器搭建方式

上一节课我们讲过GIT是一个分布式版本管理系统，既然是分布那么必定会涉及远程通信，那么GIT是采用什么协议进行远程通信的呢?



git支持的四种通信协议：

1. Local(本地协议)
2. ssh
3. http(Dumb、Smart)
4. git

1、Local(本地协议)

基于本地文件系统或共享（NFS）文件系统进行访问，
优点：简单，直接使用了现有的文件权限和网络访问权限，小团队小项目建立一个这样的版本管理系统是非常轻松的一件事。
缺点：这种协议缺陷就是本身共享文件系统的局限，只能在局域网，而且速度也慢。
适应场景：小团队，小项目临时搭建版本服务。

☐ 演示本地协议使用方式：

```
# 从本地 f/git/atals 目录克隆项目
git clone /f/git/atals/
# 即使是 bare仓库也可以正常下载
git clone /f/git/atals.git
# 基于file 协议克隆本地项目
git clone file:///f/git/atals/
```

如果在 URL 开头明确的指定 file://，那么 Git 的行为会略有不同。 如果仅是指定路径，Git 会尝试使用硬链接（hard link）或直接复制所需要的文件。 如果指定 file://，Git 会触发平时用于网路传输资料的进程，传输过来的是打包好的文件,更节约硬盘空间。

☐ 演示通过 file:///协议与直接拷贝的区别。

2、ssh 协议

git 支持支持利用ssh 协议进行通信，这是绝大部分linux、uninx系统都支持的，所以利用该协议架设GIT版本服务是非常方便的
优点：首先SSH 架设相对简单、其次通过 SSH 访问是安全的、另外SSH 协议很高效，在传输前也会尽量压缩数据。
缺点：权限体系不灵活，必须提供操作系统的帐户密码，哪怕是只需要读取版本。
适应场景：小团队、小项目、临时项目

☐ 演示基于 ssh 协议

这里我们把git 服务必须先安装到linux 系统上，然后才能使用ssh 协议跟git 服务进行通信。

☐ linux 安装git服务

```
# 1.安装依赖环境
yum -y install curl-devel expat-devel gettext-devel openssl-devel zlib-devel gcc perl-ExtUtils-MakeMaker

# 2.下载并解压源码
$ wget https://github.com/git/git/archive/v2.3.0.zip
```

```
# 备用下载链接: https://pan.baidu.com/s/1czwEz8 密码: y3hn
$ unzip v2.3.0.zip
$ cd git-2.3.0

#3 编译 安装（如果没有权限就加上sudo）
make prefix=/usr/local/git all
make prefix=/usr/local/git install

#4、添加环境变量
vim /etc/profile
export PATH=/root/svr/git/bin:$PATH
source /etc/profile

#如果成功显示版本号表示添加成功
git --version
git version 2.3.0
```

☐ **#创建一个裸项目**

```
git --bare init luban.git

#本地基于远程克隆仓库
git clone root@192.168.0.147:/data/git-repository/luban.git
cd luban/
#添加文件
echo "this is luban" >> README.MF
# 本地添加、提交、并推送至远程
git add -A; git commit -am 'first commit'; git push;

```

可能的错误：

```
git-upload-pack: command not found
```

原因是 ssh 协议下只能访问/usr/bin 下的目录，解决办法如下

```
ln -s /usr/local/git/bin/git-upload-pack /usr/bin/git-upload-pack
ln -s /usr/local/git/bin/git-receive-pack /usr/bin/git-receive-pack
```

3、http(Dumb、Smart)

Git http 协议实现是依懒 WEB容器(apache、nginx)及cgi 组件进行通信交互，并利用WEB容器本身权限体系进行授权验证。在 Git 1.6.6 前只支持http Dumb（哑）协议，该协议只能下载不能提交，通常会配合ssh 协议一起使用，ssh 分配提交帐号，http dumb提供只读帐号。1.6.6 之后git 提供了git-http-backend 的 CGI 用于实现接收远程推送等功能。

优点：解决了local 与ssh 权限验证单一的问题、可基于http url 提供匿名服务，从而可以放到公网上去。而local 与ssh 是很难做到这一点,必如实现一个类似github 这样的网站。
缺点：架设复杂一些需要部署 WEB服务器，和https 证书之类的配置
场景：大型团队、需要对权限精准控制、需要把服务部署到公网上去

☐ 演示 http Dumb 配置与使用

1、创建服务端版本仓库

```
cd /data/git-repository
git --bare init luban.git
cd luban.git/hooks/mv
// 版本更新钩子，当有版本提交的时候会执行更新
post-update.sample post-update
./post-update
```

nginx 静态访问配置

```
server {
        listen       80;
        server_name  git.tl.com;
        location / {
            root  /data/git-repository;
        }
}
```

本地克隆远程服务

```
git clone http://git.tl.com/luban.git
```

注：http Smart 协议 是基于 CGI 配合GIT git-http-backend 脚本进行使用，配置较复杂，现在一般不会这么去做，而是采用gitlab 、gogs 之类的web管理进行代替，在此就不在演示。

4、GIT 协议

Git 协议是包含在 Git 里的一个特殊的守护进程；它监听在一个特定的端口（9418），类似于 SSH 服务，但是访问无需任何授权。

优点

目前，Git 协议是 Git 使用的网络传输协议里最快的。 如果你的项目有很大的访问量，或者你的项目很庞大并且不需要为写进行用户授权，架设 Git 守护进程来提供服务是不错的选择。它使用与 SSH 相同的数据传输机制，但是省去了加密和授权的开销。

缺点

Git 协议缺点是缺乏授权机制。 而且9418是一个非标准端口，一般防火墙不会开放。

☐ 演示GIT协议的使用

```
cd luban.git/
# 创建一个空文件，表示开放该项目
touch git-daemon-export-ok
# 启动守护进程
$nohub git daemon --reuseaddr --base-path=/data/git-repository/ /data/git-repository/ &
#本地克隆远程项目
git clone git://192.168.0.147:9418/luban.git
```

二、基于gogs快速搭建企业私有GIT服务

概要：

1. gogs 介绍与安装
2. gogs 基础配置
3. gogs 定时备份与恢复

1、gogs 介绍安装

Gogs 是一款开源的轻量级Git web服务，其特点是简单易用完档齐全、国际化做的相当不错。其主要功能如下：

1. 提供Http 与ssh 两种协议访问源码服务
2. 提供可WEB界面可查看修改源码代码
3. 提供较完善的权限管理功能、其中包括组织、团队、个人等仓库权限
4. 提供简单的项目viki功能
5. 提供工单管理与里程碑管理。

下载安装

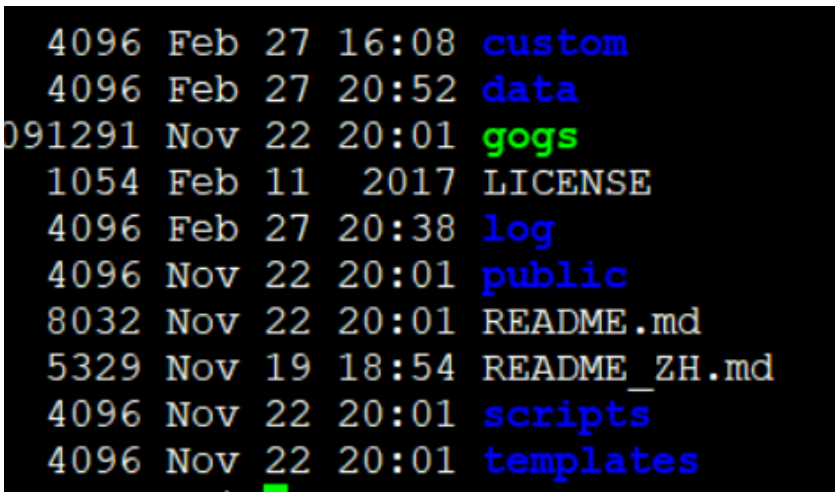
官网：https://gogs.io

下载：https://gogs.io/docs/installation 选择 linx amd64 下载安装

文档：https://gogs.io/docs/installation/install_from_binary

安装：

解压之后目录：



运行：

```
#前台运行
./gogs web
#后台运行
$nohup ./gogs web &
```

默认端口：3000

初次访问http://<host>:3000 会进到初始化页,进行引导配置。

可选择mysql 或sqlite 等数据。这里选的是sqllite

注：mysql 索引长度的问题没有安装成功,需要用mysql5.7 以上版本

2、gogs 基础配置

邮件配置说明：

邮件配置是用于注册时邮件确认，和找回密码时候的验证邮件发送。其配置分为两步：

第一：创建一个开通了smtp 服务的邮箱帐号，一般用公司管理员邮箱。我这里用的是QQ邮箱。

第二：在{gogs_home/custom/conf/app.ini 文件中配置。

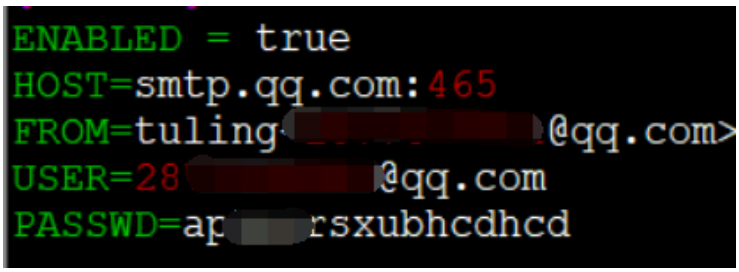
QQ邮箱开通smtp服务

1、点击设置

2、开启smtp

邮件设置

设置文件：{gogs_home/custom/conf/app.ini



ENABLED =true 表示启用邮件服务

host 为smtp 服务器地址，（需要对应邮箱开通smtp服务 且必须为ssl 的形式访问）

from 发送人名称地址

user 发送帐号

passwd 开通smtp 帐户时会有对应的授权码

重启后可直接测试

管理员登录==》控制面版==》应用配置管理==》邮件配置==》发送测试邮件

3、gogs定时备份与恢复

备份与恢复：

```
#查看备份相关参数
```

```
./gogs backup -h
#默认备份,备份在当前目录
./gogs backup
#参数化备份  --target 输出目录 --database-only 只备份 db
./gogs backup --target=./backupes --database-only --exclude-repos
#恢复。执行该命令前要先删除 custom.bak
./gogs restore --from=gogs-backup-20180411062712.zip

```

#自动备份脚本

```
#!/bin/sh -e
gogs_home="/home/apps/svr/gogs/"
backup_dir="$gogs_home/backups"

cd `dirname $0`
# 执行备份命令
./gogs backup --target=$backup_dir

echo 'backup sucess'
day=7
#查找并删除 7天前的备份
find $backup_dir -name '*.zip' -mtime +7 -type f |xargs rm -f;
echo 'delete expire back data!'
```

#添加定时任务 每天4：00执行备份

```
# 打开任务编辑器
crontab -e
# 输入如下命令 00 04 * * * 每天凌晨4点执行 do-backup.sh 并输出日志至
#backup.log
00 04 * * * /home/apps/svr/gogs/do-backup.sh >>
/home/apps/svr/gogs/backup.log 2>&1
```

4、客户端公钥配置与添加

Git配置

```
#Git安装完之后，需做最后一步配置。打开git bash，分别执行以下两句命令
git config --global user.name “用户名”
git config --global user.email “邮箱”
#git 自动记住用户和密码操作
git config --global credential.helper store
```

SSH公钥创建

```
1、打开git bash
2、执行生成公钥和私钥的命令：ssh-keygen -t rsa 并按回车3下
3、执行查看公钥的命令：cat ~/.ssh/id_rsa.pub
```