git 的缺点：

- Git 没有严格的权限管理控制，一般通过系统设置文件读写权限的方式来做权限控制。

就是那种,一个项目,前段只能看到前段代码,后端只能看到后端代码,git是做不到这种按路径授权， 如果允许按照路径授权，则各个克隆的关系将不再是平等的关系，有的内容多，有的内容少，分布式的理念被破坏

- 工作目录只能是整个项目。比如 checkout，建分支，都是基于整个项目的。而 svn 可以基于项目中的某一个目录

来自〈https://blog.csdn.net/hellow__world/article/details/72529022〉

https://www.zhihu.com/question/22363313/answer/142703190

关于git　　密钥:(主要用于免密登录)

git用的是密钥,由公钥和私钥组成(公钥加密,私钥解密,所以说公钥给别人,私钥自己留　=> 两个人想通信,则公钥要互给　)

我们要和github传代码,就要互相通信,我们把公钥上传到github上,就要能访问github了,第一次访问时github就将指纹系统保存下来了(仿佛就像将github的公钥保存在自己电脑上),以后就能互相通信了

登录过程和使用 rlogin 或 telnet 建立的会话非常类似。 在连接时， SSH 会利用一个密钥指纹系统来验证服务器的真实性。 只有在第一次连接时， 用户会被要求输入 yes。 之后的连接将会验证预先保存下来的密钥指纹。 如果保存的指纹与登录时接收到的不符， 则将会给出警告。 指纹保存在 ~/.ssh/known_hosts 中，对于 SSH v2 指纹， 则
是 ~/.ssh/known_hosts2。

> 来自 <https://www.freebsd.org/doc/zh_CN/books/handbook/openssh.html>
> 来自 <https://www.cnblogs.com/dzblog/p/6930147.html>

误解:以前以为git中的密钥是用来加密代码传输的,其实是用来验证电脑和github间的信任

认证流程：

1. Client端用户TopGun将自己的公钥存放在Server上，追加在文件authorized_keys中。
2. Server收到登录请求后，随机生成一个字符串str1，并发送给Client。
3. Client用自己的私钥对字符串str1进行加密。
4. 将加密后字符串发送给Server。
5. Server用之前存储的公钥进行解密，比较解密后的str2和str1。
6. 根据比较结果，返回客户端登陆结果。

来自〈https://www.cnblogs.com/dzblog/p/6930147.html〉