由Git的SSH公钥到非对称加密

Git协议

Git主要使用四种协议来进行数据的传输：

1）本地传输

2）Git协议

3）SSH协议

4）HTTP协议

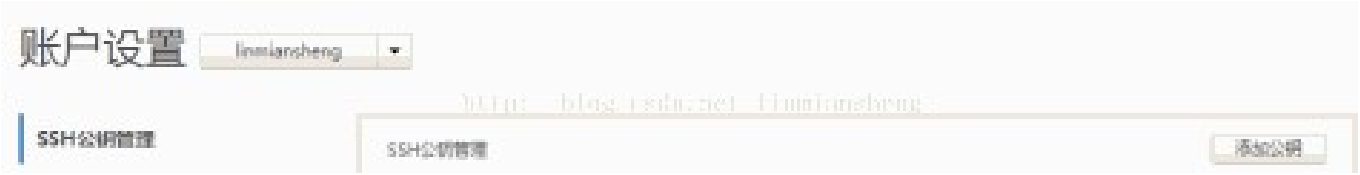
其中SSH协议和HTTP协议是最常见的两种协议了。

SSH协议则是唯一同时支持读写操作的协议，很多企业内部在架设Git服务器的时候，都会提供SSH协议来传输数据。

SSH协议的安全性体现在其使用了公钥加密，而其实用性和安全性的另外一个体现，则是提供了公钥登陆的机制。

只要将我们的公钥放上服务器，由Git服务器进行管理，我们就不用每一次推送都要输入密码，方便了我们的推送操作。

在GitLab或者CSDN的Code上面，都有一个SSH公钥管理页面，如下：



那么SSH公钥到底是什么，为什么要添加SSH公钥呢，怎么生成SSH公钥呢？

希望能通过这一篇文章，将我理解的学习到的知识跟大家分享一下。

公钥私钥，是非对称加密中的概念，是相对于对称加密而言的。

所谓加密，就是将人人都能看懂的内容变成了莫名奇妙的内容，但是你跟我能够去解读它，而其他人则不行。

这就等于给内容加上了一把锁。

对称加密

而你跟我，手上都有一把钥匙，所以我们都能去打开这把锁，从而看到里面的里容

比如下面这个例子：

我：wygdlkd pejf ktqn 。 你：。。。 我：五笔！ 你：哦！

在这里，“五笔”就是一种密钥。你只要知道了这个密钥，你马上就能够知道我在说什么了。

如果加密解密都是用同一个密钥，就叫做对称加密。

不过在网络传输中，如果你不把这个密钥也传给对方的话，对方也就无法知道你在说什么了。

但如果你将密钥也通过网络传输出去，密钥就有可能被别人截取，那么你的内容也就有可能被破解了。

非对称加密

而在非对称加密中，则会产生一对密钥，比如说KeyA和KeyB，满足下面两条规则：

1）用KeyA加密过的内容，只有KeyB能够解密。

2）用KeyB加密过的内容，则只有KeyA能够解密。

而我们会将其中一个密钥（比如KeyA）公开给外面的人使用，这就叫公钥。

另外一个密钥（比如KeyB）则由自己私人保存着，这就叫私钥。

我们假设每个人都有这样一对公钥跟私钥，我们都知道彼此的公钥。

那么，如果我要传数据给你，我就会拿你的公钥来对这些数据进行加密。这样，这些加密过的数据就只有你能解开了，因为只有你有私钥。

同样的，如果你要传数据给我，你就可以拿我的公钥来加密，然后再传给我，这样也不怕别人截取，因为它们也解不开。

利用非对称加密的公钥私钥，我们就可以只传输数据，而不用去传输解密的密钥，从而避免了对称加密的不足。

SSH的安全性就在于其利用了公钥加密这种机制。

身份验证（数字签名）

上面所说的，是用公钥加密来保证数据在传输中的安全性。而用私钥加密则可以帮助我们进行一个身份的验证，而这也就是SSH公钥登陆的原理。

根据非对称加密的规则，用其中一个密钥加密的数据只有另外一个密钥才能解开。

据是用你的私钥来加密的，而这个私钥只有你一个人有，那么就说明这份数据的确就是你的。

这就正好验证了你的身份。

Git服务器的SSH公钥管理机制正是利用这一点来实现身份的检查了。

1）首先我们会将公钥保存在Git服务器上。

2）当客户端连接Git服务器的时候，服务器会随机生成一串字符串，并将其传送给客户端。

3）我们的私钥是保存在本地的。当客户端收到服务器传回来的随机字符串的时候，客户端会利用本地的私钥进行加密，并将加密后的数据传回给服务器。

4）服务器收到客户端加密的数据，会利用该用户事先存储的公钥进行解密，如果解密成功，并且跟原先的字符串一致，则验证了该用户的身份，连接建立。

上面这几步，其实就是SSH公钥登陆的机制。

而从其他的角度来说，这种利用私钥加密的过程，其实就是在对数据进行签名，所以也叫数字签名。

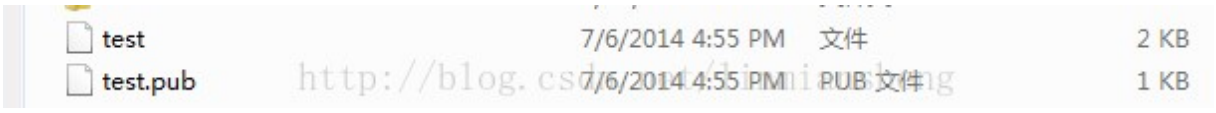
生成SSH公钥

我们可以利用ssh-keygen命令来生成一对密钥：

```
linmiansheng@LINMIANSHENG-PC /F/test_workspace
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/linmiansheng/.ssh/id_rsa): test
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in test.
Your public key has been saved in test.pub.
The key fingerprint is:
cc:ac:96:2a:a0:6a:4b:eb:cc:08:54:d4:f6:0c:57:bc linmiansheng@LINMIANSHENG-PC
```


passphrase是对使用这个密钥的密码，可以留空，也可以输入。

而命令完成之后，就会在当前目录下生成一对key，其中以.pub结尾的文件，则是我们对应的公钥了，可以将其放到Git服务器上了。



关于非对称加密，其实还涉及到很多其他方面的知识，比如我现在想传一些数据给你，想拿你的公钥来加密，但是我应该去哪拿你的公钥呢？我又怎么知道我拿到的公

结束。

作者: foolsheep