**双机互信配置**

**一、实现原理**

使用一种被称为"公私钥"认证的方式来进行ssh登录。"公私钥"认证方式简单的解释是：

首先在客户端上创建一对公私钥（公钥文件：~/.ssh/id\_rsa.pub；私钥文件：~/.ssh/id\_rsa），然后把公钥放到服务器上（~/.ssh/authorized\_keys），自己保留好私钥。当ssh登录时，ssh程序会发送私钥去和服务器上的公钥做匹配。如果匹配成功就可以登录了。

**二、实验环境**

A机：192.168.1.77

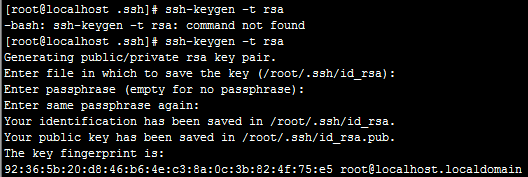
B机：192.168.1.87

**三、Linux/Unix双机建立信任**

3.1 在A机生成证书

在A机root用户下执行ssh-keygen命令，在需要输入的地方，直接回车，生成建立安全信任关系的证书。

# ssh-keygen -t rsa



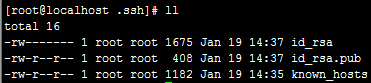
**注意：**在程序提示输入passphrase时直接输入回车，表示无证书密码。

　　　上述命令将生成私钥证书id\_rsa和公钥证书id\_rsa.pub，存放在用户root目录的.ssh子目录中。

3.2 查看~/.ssh生成密钥的文件

# cd ~/.ssh

# ll



3.3 A对B建立信任关系

将公钥证书id\_rsa.pub复制到机器B的root家目录的.ssh子目录中，同时将文件名更换为authorized\_keys，此时需要输入B机的root用户密码（还未建立信任关系）。建立了客户端到服务器端的信任关系后，客户端就可以不用再输入密码，就可以从服务器端拷贝数据了。

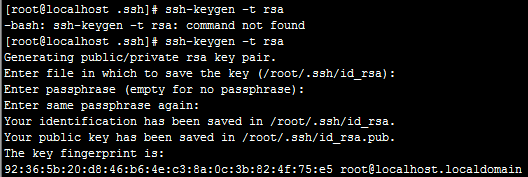
# scp -r id\_rsa.pub 192.168.1.87:/root/.ssh/authorized\_keys



3.4 B对A建立信任关系

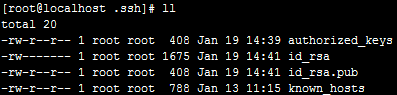
在B机上执行同样的操作，建立B对A的信任关系。

# ssh-keygen -t rsa



# cd ~/.ssh/

# ll



# scp -r id\_rsa.pub 192.168.1.77:/root/.ssh/authorized\_keys



**四、测试**

在A机上：

# scp -r /text/text.txt 192.168.1.87:/text/



在B机上：



**注：**如果想让B，C同时可以scp不输入密码，传输A中的数据；

则要把B、C的公钥都给 A；  
操作步骤：把两机器的id\_rsa.pub中的数据都拷贝到A的/root/.ssh/authorized\_keys文件中，一行表示一条；

**五、远程执行命令**

**命令格式：**ssh 远程用户名@远程主机IP地址 '远程命令或者脚本'

# ssh root@192.168.1.77 'hostname'



上述命令执行后，终端输出的是对端主机的主机名，而不是当前登录的主机的主机名。说明 hostname 这个命令其实是在对端主机上运行的。

双机信任关系已经建立！