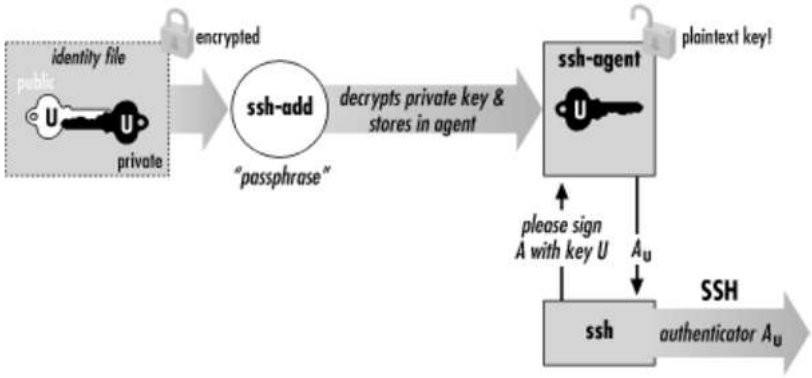
A+

ssh代理转发

上一篇文章中，我们初步的介绍了ssh-agent(ssh代理)的使用方法，ssh-agent可以帮助我们管理私钥。

ssh-agent的工作原理如下图所示，下图来自[ssh权威指南]



如图所示，我们有一对儿密钥对，我们通过ssh-add把私钥添加到了ssh-agent中，在ssh-agent中，私钥是明文保存的，当ssh客户端需要与ssh服务端进行认证时过来一些用于验证客户端身份的数据，此时，ssh客户端会跟ssh-agent进行交互，从而通过agent中的私钥对服务端发送过来的数据进行处理，然后再将经过私钥处发送到远端的服务器，服务器通过对应公钥检验对应的数据，验证成功后，建立连接。

ssh-agent始终在本机运行，完成管理私钥的任务，并且帮助我们完成认证。

那么，我们来看一个稍微复杂一点场景。

假设，我们有三台服务器，ServerA，ServerB，ServerC，它们的IP地址如下

serverA IP:10.1.0.1

serverB IP:10.1.0.2

serverC IP:10.1.0.3

目前，ServerB上有一个文件，文件路径为/testdir/ssh/test，我们需要将test文件从ServerB中拷贝到ServerC上，但是如果此时，你正处于ServerA中，你该怎么在ServerA中执行如下命令

```
A# scp root@10.1.0.2:/testdir/ssh/test root@10.1.0.3:/testdir/ssh/
root@10.1.0.2's password:
root@10.1.0.3's password:
```

执行上述命令，即可在serverA中直接将ServerB中的/testdir/ssh/test文件拷贝到ServerC的/testdir/ssh目录中，但是，由于你正处于serverA中，所以在使用上述要输入ServerB与ServerC的密码，同时完成两个机器的密码认证，才能正常的执行上述命令。

那么，如果我们同时将ServerA的公钥放置到ServerB与ServerC的对应的账户中，在执行上述命令时，能不能免去输入两次密码的操作呢？我们来试试。

我们已经在ServerA中生成了默认名称的密钥对id_rsa与id_rsa.pub，我们将id_rsa.pub分别推送到ServerB与ServerC中。

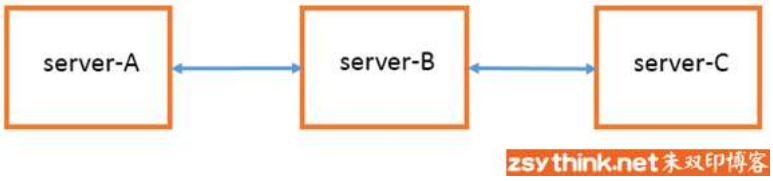
```
1 ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.1.0.2
2 ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.1.0.3
```

此时，通过ServerA已经可以免密码登录到ServerB与ServerC，环境已经准备好了，我们再次使用之前的scp命令测试一下

```
A# scp root@10.1.0.2:/testdir/ssh/test root@10.1.0.3:/testdir/ssh/
root@10.1.0.3's password:
test                                          100%   13     0.0KB/s   00:00
Connection to 10.1.0.2 closed.
A#
```

如上图所示，结果与我们想象中情况并不相同，即使通过ServerA能够免密码登录ServerC与ServerB，但是执行上述scp命令时，仍然会提示我们输入ServerC的密预期是，在ServerA中执行上述scp命令时不用输入任何密码即可将test文件从ServerB中拷贝到ServerC中，为什么会出现上述情况呢？

这是因为，当我们执行上述SCP命令时，认证的过程是这样的



如上图所示，当在ServerA中执行上述Scp命令时，ServerB会对ServerA的身份进行验证，要求ServerA输入ServerB的密码，由于我们已经将ServerA的公钥放置中，所以，输入ServerB的密码的操作即可省略，因为已经基于密钥认证，可以免密码。

但是，ServerC并不会直接向ServerA提出验证要求，而是向ServerB提出验证要求，所以，A与C之间基于密钥的认证并不会生效，同时，由于ServerB与ServerC之密钥的认证，所以，ServerC仍然会要求ServerB输入C的密码，因为我们是在ServerA上执行的操作，所以，我们需要在serverA中替ServerB输入密码，替Server

后，整个认证过程完成。

在了解了上述认证过程后，聪明如你，一定想到了一些办法，能够完全免密码的在ServerA中执行上述scp命令。

没错，我们可以在A与B之间基于密钥认证，同时，在B与C之间基于密钥认证，换句话说就是，A可以免密码登录到B，B可以免密码登录到C，完成上述设置后，即可在A中执行上述SCP命令时完全实现免密码验证，具体的操作就不再赘述了，可以自己动手试试。

但是…

在实际的工作中，我们往往不会这样做，因为我们不可能为了执行次数非常少的命令，而为服务器之间添加不必要的基于密钥的认证，这样会降低安全性。

在实际的工作中，运维人员可能会使用一台服务器作为配置管理的专用主机，所有被管理服务器都通过这一台机器进行统一管理，而这台用于统一管理的机器的公钥所有被管理服务器中，以便可以通过这台机器免密码的管理那些受管服务器，假设，上文中的ServerA就是这台用于配置管理的服务器，ServerB与ServerC就是受管常的情况就是，A的公钥会分发到B与C中，以便A可以免密码登录B或者C，方便管理，但是，B或者C之间可能并没有什么关系，B与C之间也不需要基于密钥进行认证们在ServerA中执行上述SCP命令的可能性就非常高了，这与上文中的场景完全相同，也就是说，当A能够免密码登录B或C，但是B与C之间却没有进行基于密钥的认果我们在A中执行类似上述scp命令时，仍然会提示我们输入ServerC的密码，而在实际的生产环境中，密码往往是无规律的，有位数长度要求的，所以，管理员往往服务器的密码，这时，我们还需要去查找ServerC的密码，还是很麻烦的。

那么，在刚才描述的场景中，有没有更便捷的方法，能够在不添加额外多余的密钥认证的情况下，在ServerA中完全免密码的执行上述操作呢？这个问题就会引出此理转发。

但是在描述代理转发之前，我们再来总结一下刚才的场景以及问题所在

ServerA可以基于密钥免密码登录ServerB

ServerA可以基于密钥免密码登录ServerC

ServerB与ServerC之间没有任何基于密钥的认证

我们需要在ServerA中执行如下命令，并完全免密码认证

```
1 scp root@10.1.0.2:/testdir/ssh/test root@10.1.0.3:/testdir/ssh/
```

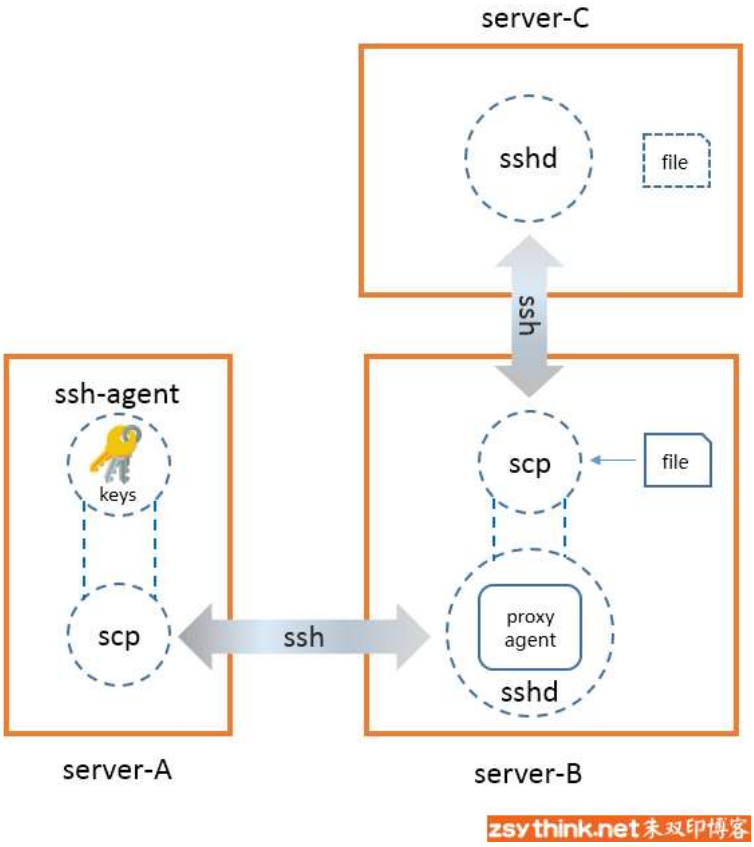
但是，在执行上述命令时，ServerC并不会与ServerA进行认证，而是与ServerB进行认证

由于ServerB与ServerC之间没有任何基于密钥的认证，导致我们需要输入ServerC的密码

现在问题的根源在于，虽然ServerC持有ServerA的公钥，但是无法直接与ServerA进行认证，而是会去找ServerB进行认证

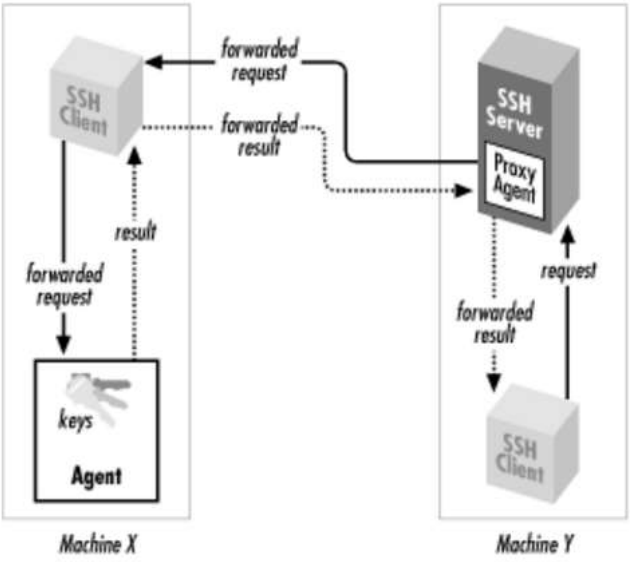
如果有一种方法，能够让ServerC通过ServerB与ServerA进行认证，那样问题就能够解决了

没错，这种方法就是代理转发，ServerB可以"充当一个代理（ssh-agent）"，当ServerC对ServerB提出认证要求时，ServerC的验证要求会发送到"ServerB的代理B的代理"会将ServerC的验证要求转发到ServerA中，这样，ServerC的验证要求就传达到了ServerA中，ServerA使用自己本地的代理（ServerA的ssh-agent）处验证要求，当处理完毕后，再将处理结果返回给ServerB充当的代理，最终，再通过ServerB返回给ServerC，这就是代理转发的过程，一图胜千言，整个过程如下图



上图中，ServerB的sshd（ssh服务端）会充当一个代理，对于ServerB中的Scp进程来说，它就是一个ssh-agent，只不过它会将ServerC的验证要求转发到ServerA本地的ssh-agent处理完成后，将结果返回给ServerB，serverB再将结果返回给ServerC，完成转发。

我们还可以参考下图，下图中的X主机与ServerA对应，下图中的Y主机与ServerB对应，下图来自[ssh权威指南]



其实，代理转发就是将验证要求通过别的机器转发到本机，通过本机的代理处理完成后，再将验证结果原路转发回去

配置代理转发

好了，原理理解的差不多了，我们可以开始动手试试了

如果想要能够实现上述代理转发，首先要在ServerA的ssh客户端配置中允许客户端启用代理转发功能。

所以，我们需要打开**ServerA的/etc/ssh/ssh_config**配置文件（**注意**：ssh_config与sshd_config不要搞混了）

将ForwardAgent的值设置为yes，默认情况下ForwardAgent配置项是被注释的，默认值为no

ForwardAgent yes表示允许ssh客户端进行代理转发

主机A中的代理转发配置已经完毕，但是我们还没有启动代理，使用如下命令启动代理，并将对应私钥添加到ssh代理中

```
ssh-agent bash
ssh-add ~/.ssh/id_rsa
```

因为上例中，我只使用了一个默认名称的密钥，无论是ServerB还是ServerC，都是使用公钥id_rsa.pub进行认证的，如果你对ServerB与ServerC分别使用了不同的
你需要将这些公钥对应的私钥都加入到ServerA的ssh-agent中。

ServerA的配置完成了，就是这么简单，剩下的配置就是在ServerB中了。

我们需要让ServerB的ssh服务端允许代理转发，因为在上述代理转发的过程中，ServerB的ssh服务端扮演了中间代理的角色，所以，我们要对ServerB的ssh服务端
功能。

我们需要编辑**ServerB的/etc/ssh/sshd_config**配置文件（**注意**：ssh_config与sshd_config不要搞混了）

将AllowAgentForwarding的值设置为yes，表示允许进行代理转发， openssh中AllowAgentForwarding默认值即为yes，所以，如果配置没有修改过，保持默

完成上述配置后，在之前描述的场景中，即可在ServerA中完全免密码的复制文件了，是不是很简单，而且完全没有多余的密钥认证配置。

其他场景

除了刚才描述的场景，还有一些场景，也会用到代理转发的功能，比如

你想通过家里的主机A连接到公司的主机C，但是，主机C位于公司办公网络的防火墙内，不能直接从互联网上访问，不过，公司为了提供访问入口，提供了一台堡垒
要连接到公司内的主机，则需要通过堡垒机B作为跳板，你可以通过堡垒机B连接到主机C，为了增加安全性，公司不允许通过密码连接到主机C，只允许通过密钥认
接，同时，由于主机B暴露在互联网上，为了安全起见，公司不允许在主机B上存放任何私钥，因为万一主机B被攻破，存放在主机B上的私钥都会被窃取，那么在这
需要在主机A中通过主机B连接主机C，而且必须通过密钥认证的方式连接主机C，同时，主机B中又不能存放任何私钥，这时，我们就可以使用代理转发解决问题，