任务背景

经过一段时间后，开发人员和运维人员都觉得使用密码SSH登录的方式太麻烦（每次登录都需要输入密码，难记又容易泄露密码）。为了安全和便利性方面考虑，要求运维人员给所有服务器实现免密码登录。

任务要求

所有开发人员通过远程管理用户code登录生产服务器实现免密码登录。

任务拆解

1. ==理解免密登录原理==
2. ==根据需求针对不同用户配置免密登录==

涉及知识点

- 免密登录原理（==理解==）
- 用户生成秘钥对（公钥和私钥）
- 免密码登录配置（==**重点**==）

课程目标

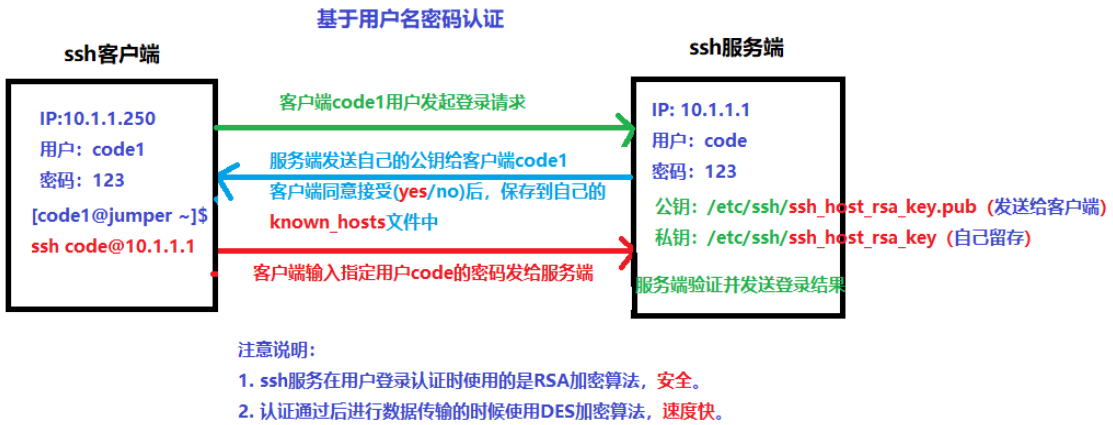
- 了解sshd服务的认证方式
- 理解免密等原理
- **==能够根据需求对用户进行免密码登录配置==**

理论储备

SSH两种认证方式

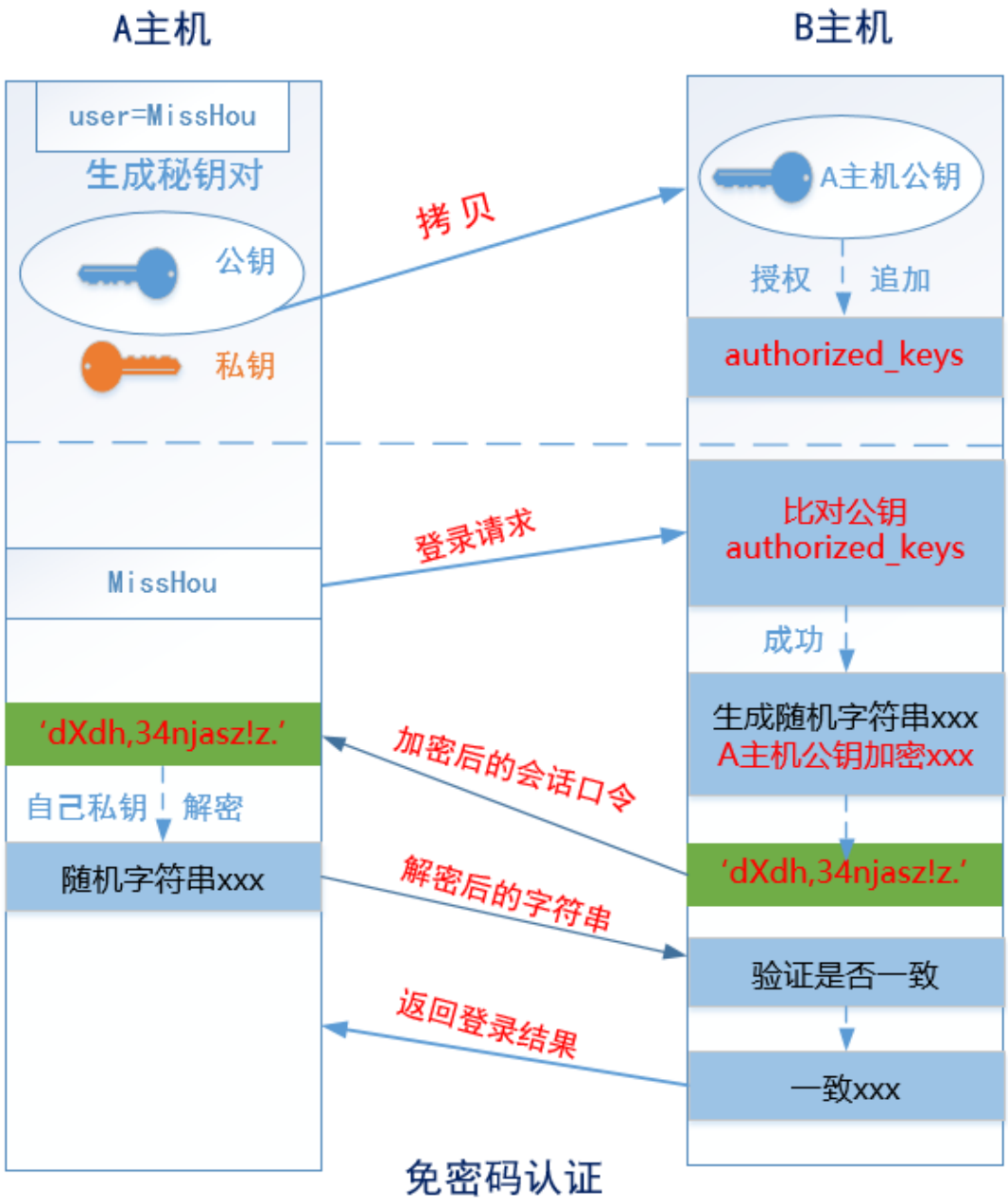
1、基于用户名密码的认证(精简版)

JumpServer => ssh code@RealServer的IP地址



2、基于密钥对的认证

基于密钥对认证，也就是所谓的免密码登录，理解免密登录原理：



任务解决方案

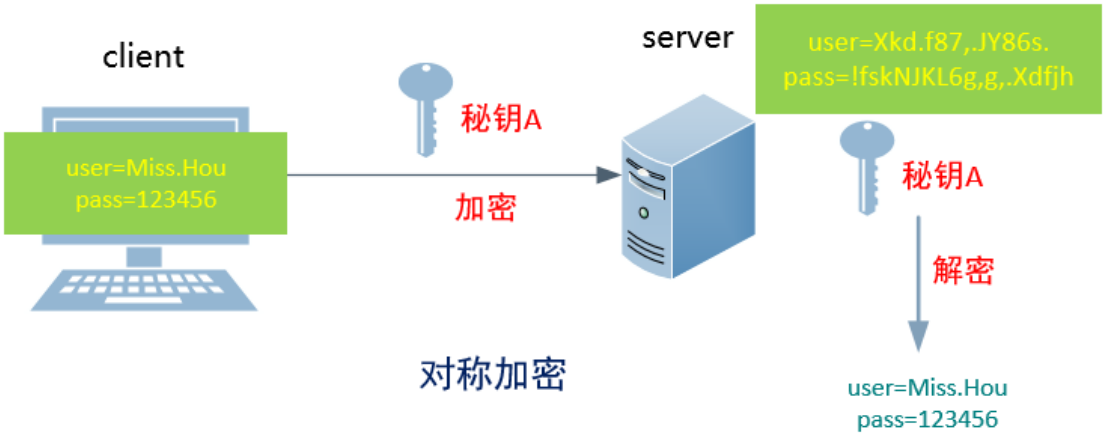
```
1. 跳板机上的开发人员自己生成一对秘钥
code1为例：
[code1@MissHou ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/code1/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/code1/.ssh/id_rsa.
Your public key has been saved in /home/code1/.ssh/id_rsa.pub.
The key fingerprint is:
14:78:f6:70:9f:48:64:7e:19:c3:cb:c3:7a:52:1e:d8 code1@MissHou.itcast.cc
The key's randomart image is:
+--[ RSA 2048]----+
```

```
|       ...o.o    |
|      . ++o .+   |
|       o.=.Boo   |
|       .  +.E    |
|        S  + o   |
|          o o    |
|           o     |
|                 |
|                 |
+-----------------+
[code1@MissHou ~]$ ll -a .ssh/
total 16
drwx------ 2 code1 coding 4096 Dec 28 09:33 .
drwx------ 5 code1 coding 4096 Dec 27 11:49 ..
-rw------- 1 code1 coding 1675 Dec 28 09:33 id_rsa
-rw-r--r-- 1 code1 coding  405 Dec 28 09:33 id_rsa.pub

2. 将code1用户的公钥远程拷贝到生产服务器上指定用户的指定目录
[code1@MissHou ~]$ ssh-copy-id code@10.1.1.1
The authenticity of host '10.1.1.1 (10.1.1.1)' can't be established.
RSA key fingerprint is 30:c8:1a:67:55:22:33:26:e5:fb:44:56:4d:8b:26:40.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.1.1.1' (RSA) to the list of known hosts.
code@10.1.1.1's password:
Now try logging into the machine, with "ssh 'code@10.1.1.1'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.
或者
[code1@MissHou ~]$ scp -P22 ~/.ssh/id_rsa.pub
code@10.1.1.1:/home/code/.ssh/authorized_keys
code@10.1.1.1's password:
id_rsa.pub
3. 测试验证
[code1@MissHou ~]$ ssh -lcode 10.1.1.1
Last login: Fri Dec 28 09:38:17 2018 from 10.1.1.250
[code@server ~]$
```

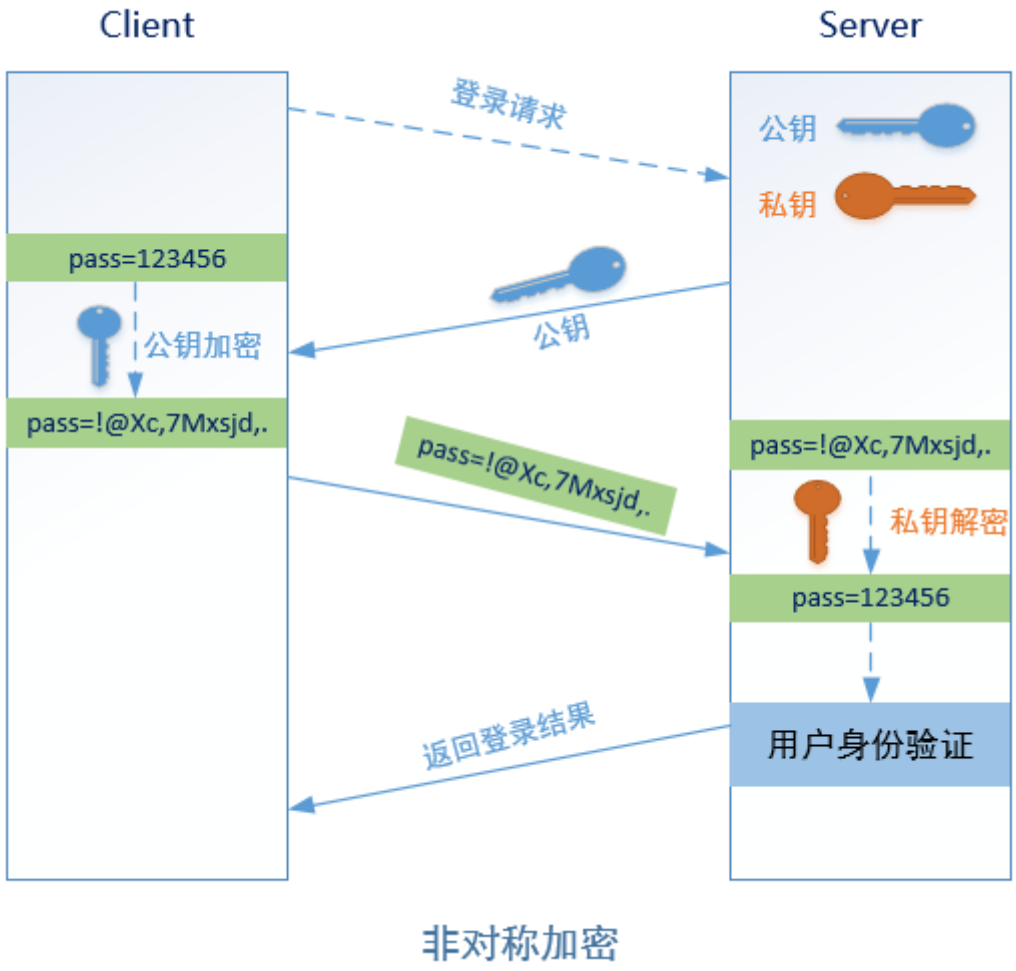
扩展总结

图解SSH加密算法

- des 对称的公钥加密算法,安全低，数据传输速度快；使用同一个秘钥进行加密或解密
- rsa 非对称的公钥加密算法,安全,数据传输速度慢， SSH默认的加密算法

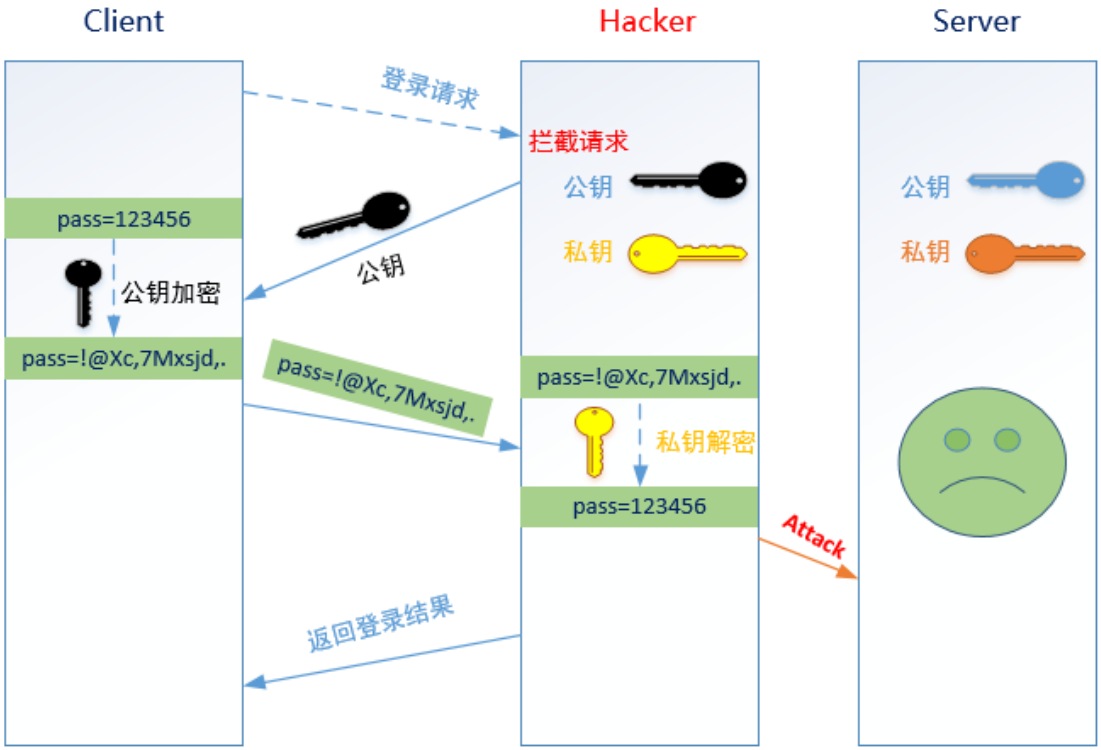


思考1： 用户信息加密了，但如何安全的保存密钥呢?



1、远程Server收到Client端用户的登录请求后，Server端把自己的公钥发给用户
2、Client端使用这个公钥，将密码进行加密
3、Client将加密的密码发送给Server端
4、远程Server用自己的私钥，解密登录密码，然后验证其合法性
5、根据验证结果，给Client相应的响应。

思考2： 非对称加密就绝对安全吗?



中间人劫持

问题： SSH中是如何解决这个问题的呢?

答：基于用户名密码认证和密钥对认证。

- **==基于用户密码的认证==**

```
[root@MissHou ~]# ssh 192.168.10.171
The authenticity of host '192.168.10.171 (192.168.10.171)' can't be established.
RSA key fingerprint is 9f:71:de:3c:86:25:dd:f0:06:78:ab:ba:96:5a:e4:95.
Are you sure you want to continue connecting (yes/no)?
```

提示信息：无法确认主机192.168.10.171的真实性，指纹是`9f:71:de:3c:86:25:dd:f0:06:78:ab:ba:96:5a:e4:95.`，你确定想要继续吗?

说明:

1. 理论上应该是对公钥的确认，由于公钥通过RSA算法加密，太长，不好直接比较，所以给公钥生成一个hash的指纹，方便比较。
2. 当客户端输入yes确认对方的公钥指纹后，server端的公钥就会被存放到客户机的用户家目录里~/.ssh/known_hosts文件中，下次再访问就直接通过密码登录，不需要再确认公钥。

- **==基于秘钥对的认证（免密码登录）==**

相关文件解读:

1. id_rsa：保存私钥
2. id_rsa.pub：保存公钥
3. authorized_keys：保存已授权的客户端公钥
4. known_hosts：保存已认证的远程主机公钥