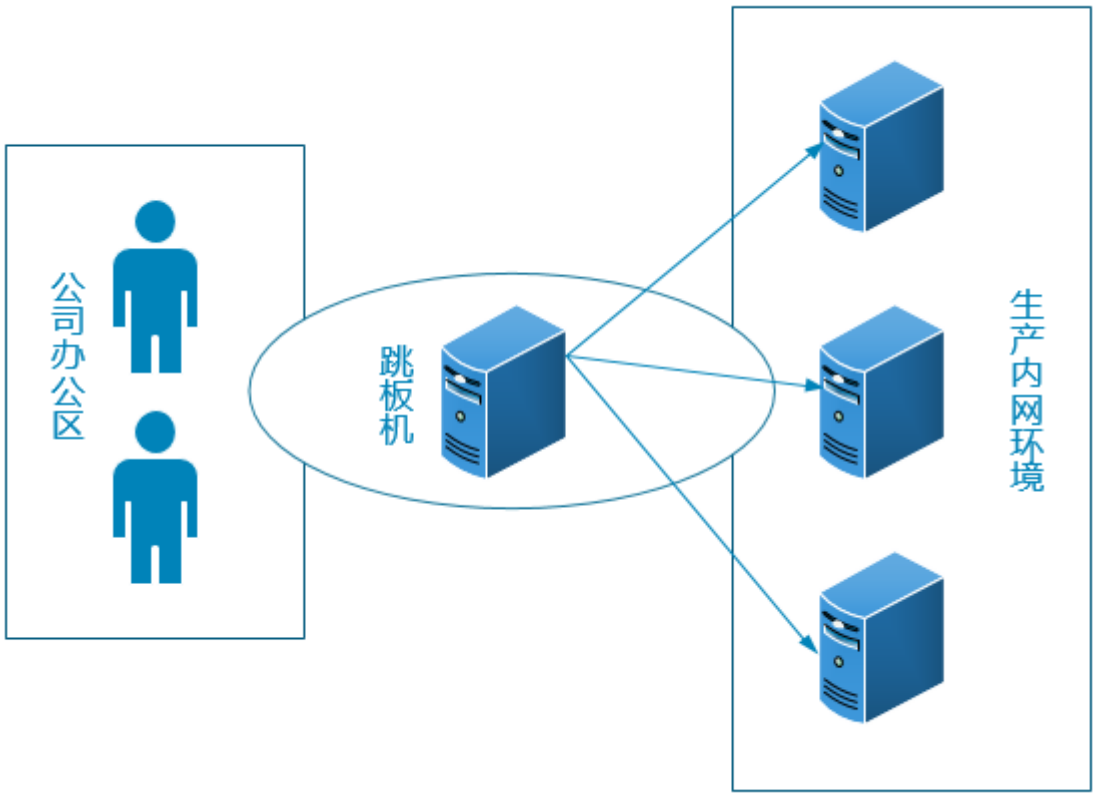
任务背景

为了最大程度的保护公司内网服务器的安全，公司内部有一台服务器做跳板机。运维人员在维护过程中首先要统一登录到这台服务器，然后再登录到目标设备进行维护和操作。由于开发人员有时候需要通过跳板机登录到线上生产环境查看一些业务日志，所以现在需要运维人员针对不同的人员和需求对账号密码进行统一管理，并且遵循**权限最小化**原则。



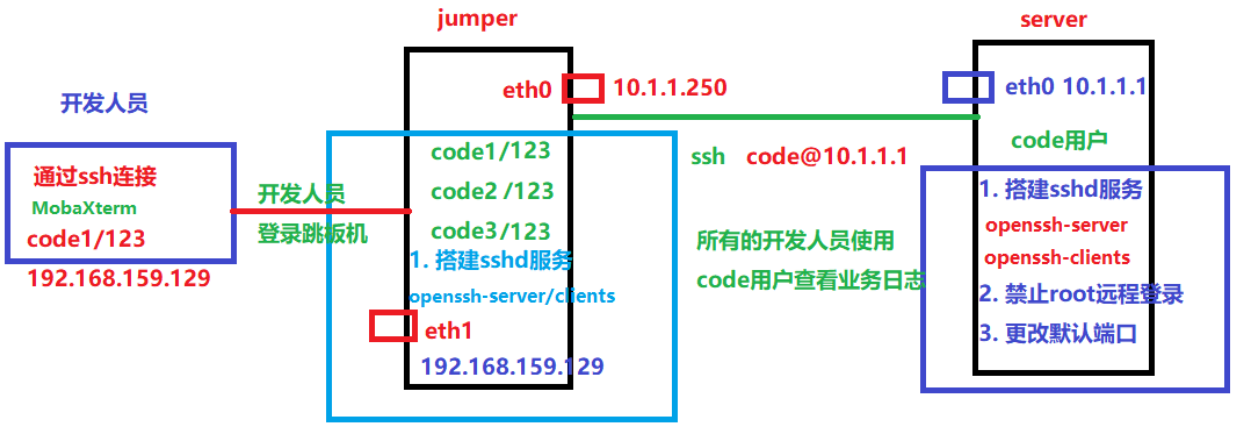
任务要求

1. 跳板机上为每个开发人员创建一个账号，并且只能在指定的目录里管理自己的文件。
2. 线上生产服务器，禁止使用root用户远程登录。
3. 线上生产服务器sshd服务不允许使用默认端口，防止黑客通过端口扫描。
4. 线上生产服务器上开发人员使用的账号code用户的密码使用工具随机生成。

任务拆解

1. 跳板机上为开发人员创建用户及公共目录供开发人员使用，并做好权限控制（粘滞位）
2. 所有线上生产服务器搭建ssh服务
3. 对于ssh服务根据需求进行配置
    - 禁止root用户远程登录

    - 更改默认端口
4. 线上生产服务器创建code用户，并安装工具来生成随机密码



涉及知识点

- 用户权限管理（旧知识点）
- **ssh服务配置（新知识点）**
- 生成随机密码工具（新知识点）

课程目标

- [ ] 了解ssh服务的认证方式
- [ ] 能够禁止root远程登录
- [ ] 能够更改ssh服务的默认端口
- [ ] 熟练使用相关客户端工具，如ssh远程登录，scp远程拷贝文件

知识储备

一、什么是服务

- 运行在操作系统后台的一个或者多个程序，为系统或者用户提供特定的服务
- 可靠的，并发的，连续的不间断的运行，随时接受请求
- 通过交互式提供服务

二、服务架构模型

(一) B/S架构

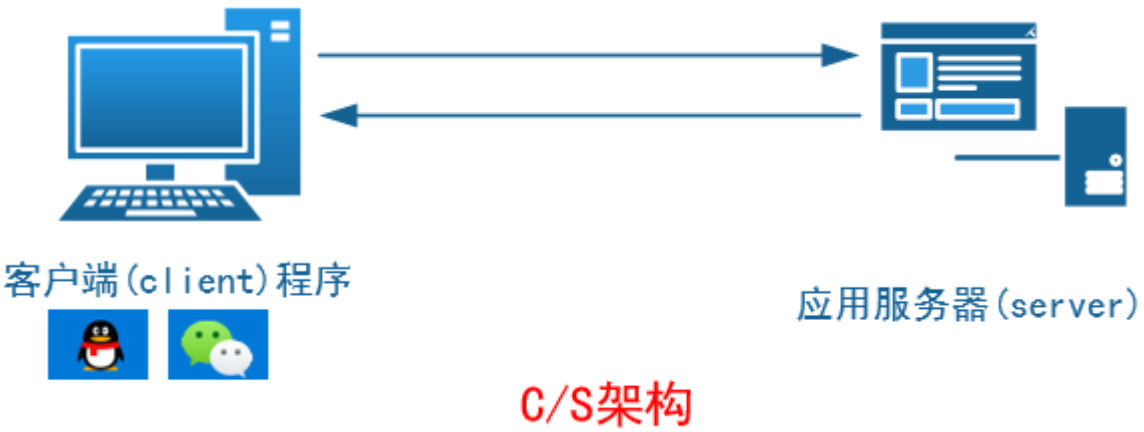
- B/S(browser/server) 浏览器/服务器

概念：这种结构用户界面是完全通过浏览器来实现，使用http协议 优势：节约开发成本

㈡ C/S架构

- C/S（client/server）客户端/服务器

概念：指的是客户端和服务端之间的通信方式，客户端提供用户请求接口，服务端响应请求进行对应的处理，并返回给客户端 优势：安全性较高，一般面向具体的应用



㈢ 两者区别

B/S： 1、广域网，只需要有浏览器即可 2、一般面向整个互联网用户，安全性低 3、维护升级简单

C/S： 1、专用网络、小型局域网，需要具体安装对应的软件 2、一般面向固定用户，安全性较高

思考1:

我们通过网络是如何找到我们想要访问的服务的?

IP(提供服务的服务器)+Port(找到相应的服务)

三、端口号设定

说明:端口号只有整数，范围是从0 到65535：1～255：**一般是知名端口号**，如:ftp 21号、web 80、ssh 22、telnet 23号等；256～1023：通常都是由Unix系统占用来提供特定的服务；1024~5000：客户端的临时端口，随机产生；大于5000：为互联网上的其他服务预留

思考2:

如何查看系统默认的注册端口?

```
/etc/services
```

四、常见的网络服务

- 文件共享服务：FTP、SMB、NFS
- 域名管理服务：DNS
- 网站服务：Apache(httpd)、Nginx、Lighttpd、IIS
- 邮件服务: Mail
- 远程管理服务：SSH、telnet
- 动态地址管理服务:DHCP

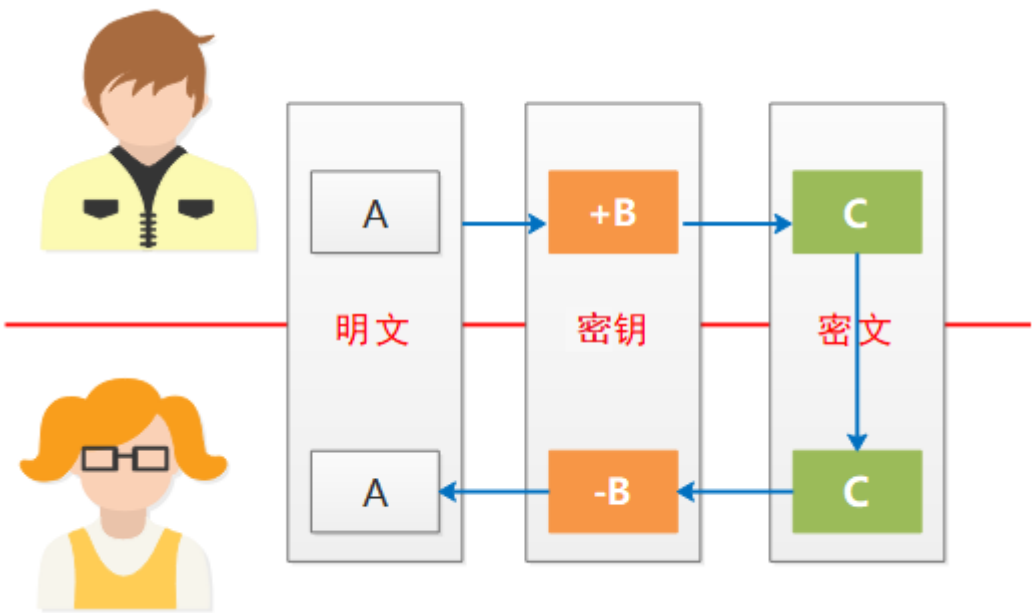
五、SSH服务概述

熟悉Linux的人那肯定都对SSH不陌生。ssh是一种用于安全访问远程服务器的协议，远程管理工具。它之所以集万千宠爱为一身，就是因为它的安全性。那么它到底是怎么样来保证安全的呢？到底是如何工作的呢?

首先，在讲SSH是如何保证安全的之前，我们先来了解以下几个密码学相关概念：

㈠ 加密算法（了解）

1、对称加密算法(DES)



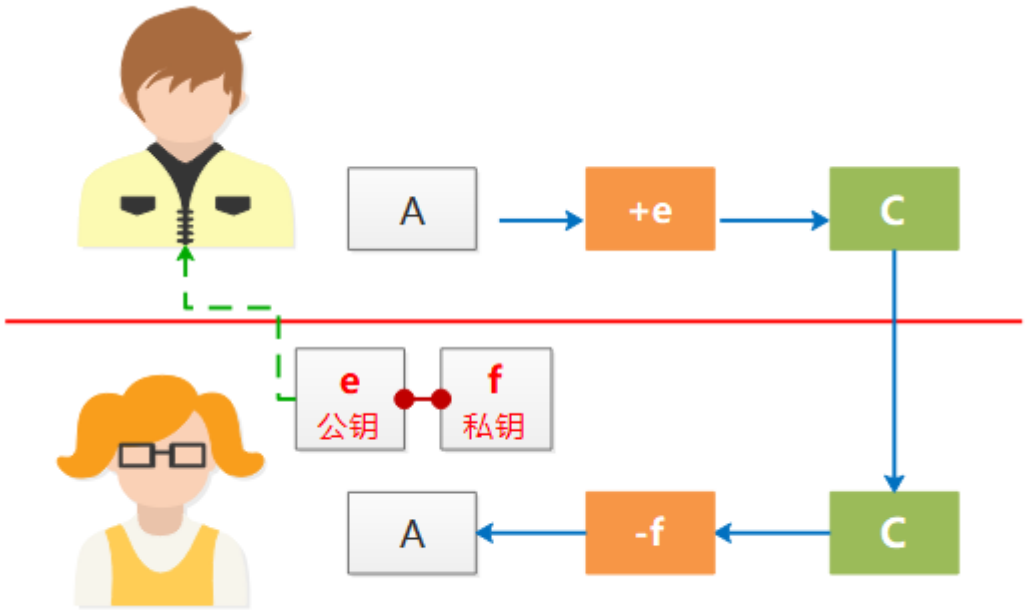
1. Jack想要给Harry发送信息一个信息A，为了安全起见，Jack使用一种加密算法，比如给信息通过加一个数字B得到一个新的数字C，然后以公开的方式发送给Harry

2. Harry接受到数字C后，通过减去一个数字B得到最终的真正的信息A
3. Jack发送给Harry的信息A称为明文；加密后的信息C称为密文；加密用的B称之为密钥
4. 加密算法（方法）可以很复杂，不一定是加和减，也可以是乘和除等等
5. 以上过程中，加密和解密的秘钥是同一个密钥B

总结:

1. 发送方使用**密钥**将**明文数据**加密成**密文**，然后发送出去
2. 接收方收到密文后，使用同一个密钥将密文解密成明文进行读取

2、非对称加密算法(RSA)



1. 首先Harry生成一对有相互关系的密钥对，比如e（公钥）和f（私钥）；其中公钥是可以公开给所有人的，私钥必须Harry本人私自留存，不得泄露。
2. 当Jack发送请求时，Harry会把自己的公钥e发送给Jack
3. Jack拿着Harry的公钥e通过一种加密算法将信息A加密成密文C，以公开的方式发送给Harry
4. Harry收到密文C后，通过自己本地留存的私钥 f 将密文解密成最终的信息A
5. 以上过程中，加密使用的是公钥e，解密使用的是私有f；使用不同的秘钥加解密

总结:

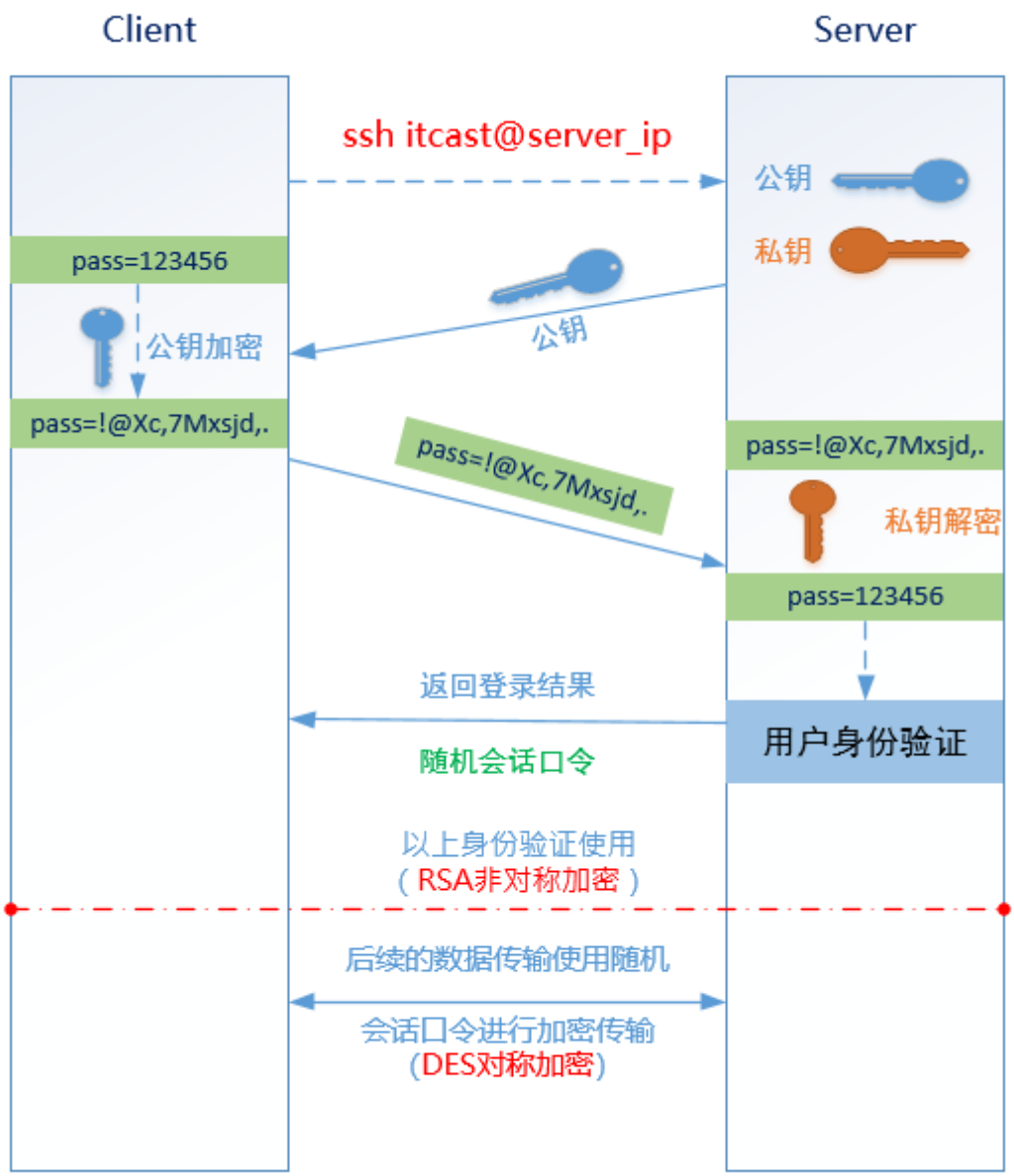
1. 发送方使用接收方发送过来的**公钥**将**明文数据**加密成**密文**，然后发送出去
2. 接收方收到密文后，使用自己本地留存的**私钥**将密文解密成明文进行读取

- **对称加密算法与非对称加密算法区别**
  - 对称加密
    1. 使用同一个密钥进行加密和解密，密钥容易泄露
    2. 加密速度快，效率高，数据传输速度快，安全性较低
  - 非对称加密
    1. 使用不同的密钥（公钥和私钥）进行加密和解密
    2. 加密速度远远慢于对称加密，数据传输速度慢，安全性较高

问： 了解了加密算法，那SSH到底是如何保证数据安全的呢？

答：SSH服务有两种认证方式，分别是基于用户名密码认证和密钥对认证

㈡ SSH基于用户名密码认证原理



1. SSH客户端向SSH服务端发起一个登录请求
2. SSH服务端将自己的公钥发送给SSH客户端

   注意：如果是第一次访问，则提示以下内容：

```
[root@MissHou ~]# ssh 192.168.10.171
The authenticity of host '192.168.10.171 (192.168.10.171)' can't be established.    //
无法确认主机192.168.10.171的真实性
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)?yes
说明:
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。
```

3. SSH客户端使用服务端发过来的公钥将自己的密码加密并且发送给SSH服务端
4. SSH服务端收到SSH客户端发过来的加密密码后使用本地留存的私钥进行解密
5. SSH服务端将解密出来的密码和`/etc/shadow`文件里的用户密码对比认证
6. SSH服务端认证成功，则返回登录成功结果，并发送一个随机会话口令给客户端，该口令用于后面两台主机进行数据传输的一个临时加密会话口令

㈢ SSH介绍总结

- SSH是Linux下远程管理的工具，相比Telnet安全，运维人员必备的神器！
- SSH的全称Secure Shell，安全的shell，是Client/Server架构，默认端口号为22，TCP协议
- SSH其实用于商业，而OpenSSH即为开源的，在Linux中默认安装

六、SSH服务配置

㈠ 搭建所有服务的思路

1. 关闭防火墙和selinux（实验环境）
2. 配置yum源（本地yum源或根据需求选择配置网络源）
3. 软件三步曲
   1）安装软件
   2）确认成功安装
   3）查看软件的文件列表（配置文件、程序本身、说明手册...）
4. 了解配置文件（语法、参数配置 man 5 xxx.conf）
5. 根据需求通过修改配置文件来完成服务的搭建
6. 启动服务（开机自启动）
7. 测试验证

㈡ 搭建SSH服务

server：10.1.1.2 搭建好SSH服务

clinet：10.1.1.1 远程访问server端的SSH服务

1、关闭防火墙和selinux

```
略
```

2、配置yum源（本地）

```
1）挂载光盘到本地
# mount -o ro /dev/sr0 /mnt
# echo "mount -o ro /dev/sr0 /mnt" >> /etc/rc.local
2）修改配置文件
# cd /etc/yum.repos.d/
# rm -f Centos-*
# cat local.repo
[local]
name=xxxx
baseurl=file:///mnt
enabled=1
gpgcheck=0

3）清空yum缓存重新创建
# yum clean all
# yum makecache
```

3、软件三步曲

```
1）安装软件
默认系统已经安装完毕
2）确认是否安装
# rpm -q openssh-server
openssh-server-5.3p1-94.el6.x86_64
# rpm -q openssh-clients
openssh-clients-5.3p1-94.el6.x86_64
3）查看软件包的文件列表
# rpm -ql openssh-server
/etc/rc.d/init.d/sshd                    启动脚本
/etc/ssh/sshd_config                     配置文件
/usr/sbin/sshd                           程序本身（二进制文件）
/usr/share/man/man5/sshd_config.5.gz     配置文件的man手册
/usr/share/man/man8/sshd.8.gz            sshd程序的man手册

# rpm -ql openssh-clients
/etc/ssh/ssh_config                客户端配置文件
以下是客户端相关命令：scp、sftp、slogin|ssh、ssh-copy-id
/usr/bin/scp
/usr/bin/sftp
/usr/bin/slogin
/usr/bin/ssh
/usr/bin/ssh-copy-id
客户端命令以及配置文件的man手册
/usr/share/man/man1/scp.1.gz
/usr/share/man/man1/sftp.1.gz
/usr/share/man/man1/slogin.1.gz
/usr/share/man/man1/ssh-add.1.gz
/usr/share/man/man1/ssh-agent.1.gz
/usr/share/man/man1/ssh-copy-id.1.gz
/usr/share/man/man1/ssh-keyscan.1.gz
```

```
/usr/share/man/man1/ssh.1.gz
/usr/share/man/man5/ssh_config.5.gz
```

4、了解配置文件

```
# man 5 sshd_config

PermitRootLogin   指定是否允许root远程登录，yes or  no
Port              指定服务的监听端口，默认是22
```

5、根据需求修改配置文件

需求：禁止root用户远程登录

```
# vim /etc/ssh/sshd_config
#PermitRootLogin yes
PermitRootLogin no
```

6、启动服务开机自启动

```
[root@server ~]# service sshd restart
Stopping sshd:                                             [  OK  ]
Starting sshd:                                             [  OK  ]
[root@server ~]# chkconfig --list|grep sshd
sshd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

7、测试验证

```
[root@client ~]# ssh root@10.1.1.2
root@10.1.1.2's password:
Permission denied, please try again.
root@10.1.1.2's password:
Permission denied, please try again.
root@10.1.1.2's password:
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
以上说明不允许root用户远程登录。
```

任务解决方案

1. 创建用户并授权

注意：在跳板机上完成jumper-server

```
环境：开发部门3个人code1~code3
1) 创建组和相应用户
groupadd coding
useradd -G coding code1
useradd -G coding code2
```

```
useradd -G coding code3
2) 使用非交互式设置密码
echo 123|passwd --stdin code1
echo 123|passwd --stdin code2
echo 123|passwd --stdin code3
3) 为开发人员创建数据目录并且设置相应的权限
mkdir /code/data -p
[root@jumper ~]# ll -d /code/data/
drwxr-xr-x. 2 root root 4096 Apr 15 08:47 /code/data/
[root@jumper ~]# chgrp coding /code/data/
[root@jumper ~]# chmod 1770 /code/data/
[root@jumper ~]# ll -d /code/data/
drwxrwx--T. 2 root coding 4096 Apr 15 08:50 /code/data/

4) 测试验证
code1人员不能删除其他人员创建的文件；并且coding组所有成员都可以在该目录下创建文件
```

2. 禁止root远程登录

注意：在生产服务器server端完成

```
1. 搭建sshd服务
默认系统已经安装好，并且已经成功启动
2. 根据需求修改配置文件
vim /etc/ssh/sshd_config
#PermitRootLogin yes
PermitRootLogin no
3. 重启服务
service sshd restart
或者
service sshd reload
4. 测试验证
```

3. 更改默认端口

端口默认更改为：10086

```
1. 确定当前服务器端口没有被占用
[root@server ~]# netstat -nltup|grep 10086
[root@server ~]# ss -nalp|grep 10086
[root@server ~]# lsof -i :10086

2. 修改配置文件
vim /etc/ssh/sshd_config
#Port 22
Port 10086

3. 重启服务
service sshd restart
```

```
或者
service sshd reload


4.测试验证
[code1@jumper-server ~]$ ssh stu1@10.1.1.1
ssh: connect to host 10.1.1.1 port 22: Connection refused
原因：端口号不对
解决：指定端口 -p参数
[code1@jumper-server ~]$ ssh -p 10086 stu1@10.1.1.1
stu1@10.1.1.1's password:
Last login: Thu Dec 27 11:50:09 2018 from 10.1.1.250


补充：
更改客户端配置文件，不想验证指纹：
# vim /etc/ssh/ssh_config
#   StrictHostKeyChecking ask
   StrictHostKeyChecking no
```

4. 用户密码随机

注意：在线上生产环境中创建一个开发人员专用的账号

```
思路：
0. 需要在线上生产环境服务器中，创建一个公共账号（开发人员）
[root@server ~]# useradd code
1. 安装pwgen工具
分析：
   1）pwgen工具本地镜像没有，需要通过外网epel源下载安装
   2）生产环境不能访问互联网
   3）跳板机可以访问互联网，所以通过跳板机配置epel网络源来缓存软件包
   4）缓存下来的软件包再远程拷贝到server端进行安装
步骤：


2. 使用pwgen工具生成随机密码
[root@server ~]# pwgen -cnsB1 15 1
RTMU4scsh7e7vAi

3. 给线上code用户设置随机密码
[root@server ~]# echo RTMU4scsh7e7vAi |passwd --stdin code
Changing password for user code.
passwd: all authentication tokens updated successfully.

命令的使用：
# pwgen --help
Usage: pwgen [ OPTIONS ] [ pw_length ] [ num_pw ]
参数：
-c or -capitalize
密码中至少包含一个大写字母
-A or -no-capitalize
```

```
密码中不包含大写字母
-n or –numerals
密码中至少包含一个数字
-0 or –no-numerals
密码中不包含数字
-y or –symbols
密码中至少包含一个特殊符号
-s or –secure
生成完全随机密码
-B or –ambiguous
密码中不包含模糊字符（如1,l,O,0）
-C
在列中打印生成的密码
-1
不要在列中打印生成的密码，即一行一个密码

# pwgen -cnBs1 10 3
生成长度为10，包含大写、数字、不包含模糊字符完全随机的3个密码

# pwgen -cnBs1 10 5
J9xREhKP9b
CJXoJffW4F
nTi4mAzjPx
UiJTiVcz9i
xTFK7KLbRc

# echo 'J9xREhKP9b' |passwd --stdin yunwei
Changing password for user yunwei.
passwd: all authentication tokens updated successfully.
注意:
在实际的工作中，以shell脚本形式定期修改用户的密码，并且用户名和密码对应会保存到一个密码文件里!
```

补充扩展

1. ssh客户端工具

```
C/S

openssh-clients/openssh-server

ssh --help
man ssh

[user@]hostname [command]

-l: 指定连接用户
-p: 指定端口

注意:
当前本机用户是root，如果不指定连接用户，那么会让你输入远程主机的root密码。
如果指定了连接用户，那么不管你当前是什么用户就只让你输入远程主机的指定用户密码。
```

2. scp客户端工具

```
cp 本地文件的拷贝
scp 远程文件拷贝

用法:
将本地文件拷贝到远程: push 推

scp 需要拷贝的文件  远程服务器
scp file1 pos1@10.1.1.1:/tmp/
scp -r dir1 pos1@10.1.1.1:/tmp

将远程文件拷贝到本地: pull 拉

scp 远程文件  本地路径
scp -r pos1@10.1.1.1:/tmp/dir1 /data/code

-r:递归拷贝目录
-P:指定远程服务器的端口
# scp -P10022 1.txt pos@10.1.1.2:/tmp
```

3. sftp客户端工具

```
[root@localhost ~]# sftp 192.168.159.101
Connecting to 192.168.159.101...
root@192.168.159.101's password:
sftp> help

[root@localhost ~]# sftp 192.168.159.101:/root/anaconda-ks.cfg /tmp
```