

# ssh服务

☁ 作者：牟建波 (1353429820@qq.com)

时间：2025-05-28

描述：日常自学笔记

## 1.ssh基本概念



☁ SSH (Secure Shell, 安全外壳)：是一种网络安全协议，旨在通过加密和认证机制实现安全的远程访问和文件传输等业务

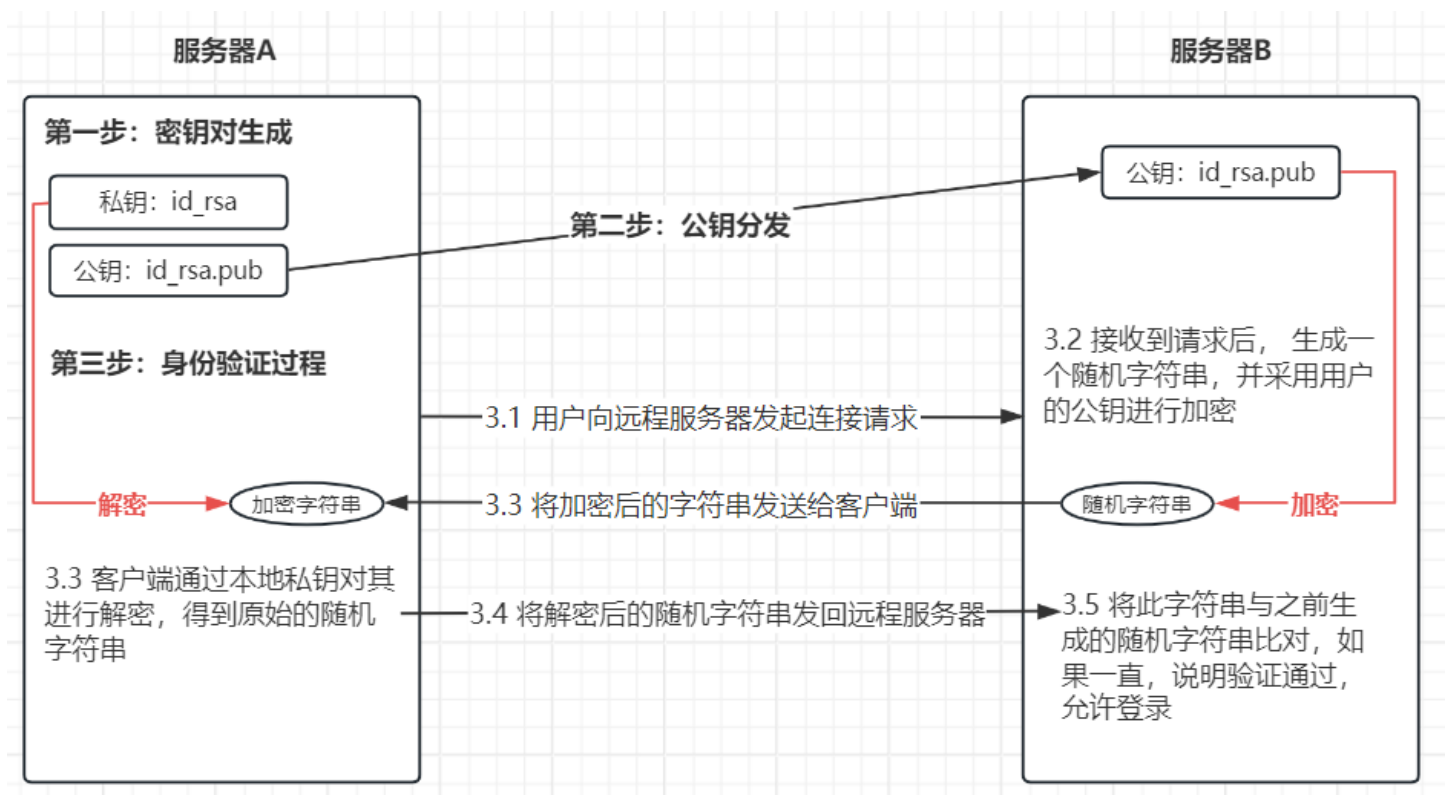
SSH支持多种身份验证方法，包括**密码认证**和**密钥认证**。密码认证是将用户名和密码发送给服务器进行认证，而密钥认证则使用**公钥**和**私钥**对进行身份验证，实现安全的免密登录

### 代码块

- 1 公钥：
  1. 公钥是可以公开分享的密钥，用于加密信息和验证数字签名
  2. 公钥需要被放置在远程服务器上用户账号的`~/.ssh/authorized_keys`文件中
- 4
- 5 私钥：
  1. 私钥是必须保密的密钥，用于解密信息和创建数字签名
  2. 私钥保存在客户端，通常位于用户主目录下的隐藏文件夹中，如`~/.ssh/id_rsa`
- 7
- 8

- 9 注意：私钥非常重要，不能泄露或分享给其他人。一旦私钥被泄露，任何人都可以使用它来冒充你进行SSH连接

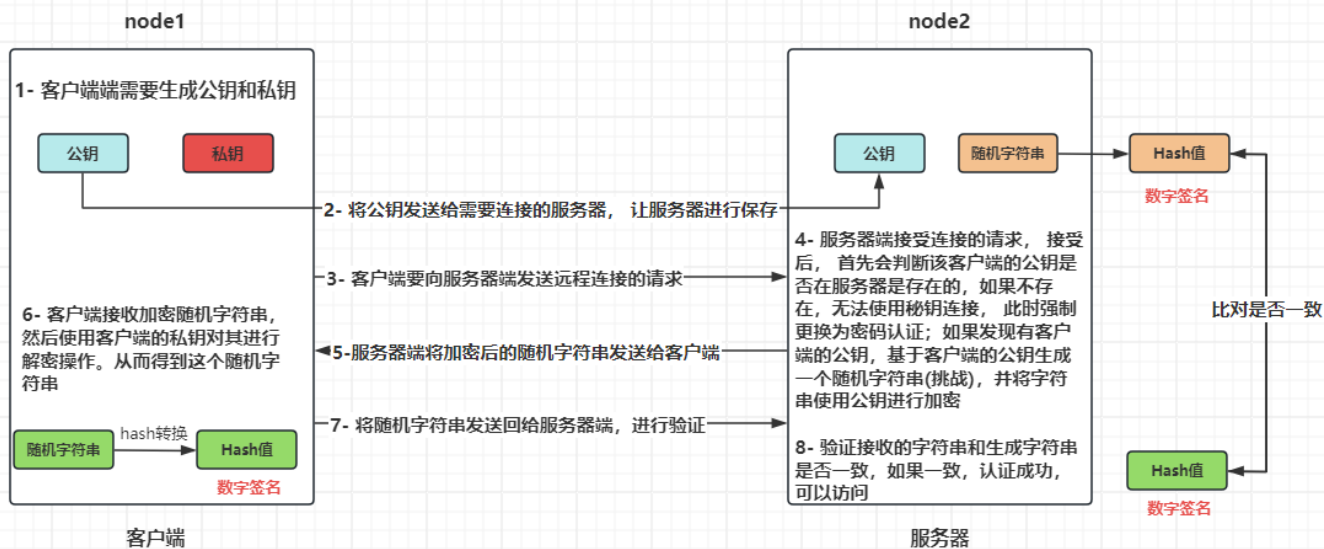
## 2.ssh工作原理



### 代码块

- 1 1. 密钥对生成:
  - 2 1. 用户在本地机器上使用ssh-keygen命令生成一对密钥，即公钥和私钥
  - 3 2. 私钥必须保密，并存储在本地机器的~/.ssh/id\_rsa文件中（或其他用户指定的位置）
  - 4 3. 公钥则可以公开，并存储在本地机器的~/.ssh/id\_rsa.pub文件中
- 5
- 6 2. 公钥分发:
  - 7 1. 用户需要将生成的公钥文件（id\_rsa.pub）的内容复制到远程服务器的~/.ssh/authorized\_keys文件中
  - 8 2. 这通常通过使用ssh-copy-id命令或其他安全方式完成
- 9
- 10 3. 身份验证过程:
  - 11 1. 当用户尝试通过SSH登录到远程服务器时，SSH客户端会向远程服务器发起连接请求
  - 12 2. 远程服务器接收到连接请求后，会生成一个随机字符串（称为“挑战”），并使用用户的公钥对其进行加密，然后将加密后的字符串发送给客户端
  - 13 3. 客户端接收到加密后的字符串后，使用本地存储的私钥对其进行解密，得到原始的随机字符串
  - 14 4. 客户端将解密后的随机字符串发送回远程服务器
  - 15 5. 远程服务器接收到客户端发送的解密后的字符串后，会与之前生成的随机字符串进行比对。如果两者一致，说明客户端拥有与公钥对应的私钥，验证通过，允许用户登录

需求：假设有二台服务器，node1和node2，需要让node1在连接node2服务器的时候，使用秘钥认证方式，不需要输入密码即可连接成功



## 3.ssh配置

### 3.1 安装ssh服务并配置

代码块

```
1  # 检查SSH是否已经安装
2  ssh -v
3
4  # 安装SSH服务
5  dnf install -y openssh-server openssh-clients
6
7  # 启动SSH相关服务
8  systemctl start sshd          # 启动服务
9  systemctl enable --now sshd   # 开机启动
10
11 # 关于SSH配置文档相关参数: /etc/ssh/sshd_config
12 vim /etc/ssh/sshd_config
13
14 核心配置说明:
15  Port: 指定SSH服务监听的端口号，默认为22，可以修改为其他端口以增强安全性
16  PermitRootLogin: 控制是否允许root用户通过SSH登录，建议设置为no以提高安全性
17  PasswordAuthentication: 控制是否允许使用密码进行认证，如果启用了密钥认证，可以将其设置为no
18  ChallengeResponseAuthentication和UsePAM: 通常与密码认证相关，如果禁用了密码认证，可以将其设置为no
```

```
19 AllowUsers 和 DenyUsers: 分别指定允许和拒绝通过SSH登录的用户列表
20
21 # 如果修改了相关配置, 需要重新加载SSH服务配置, 使其生效:
22 systemctl reload sshd
```

## 3.2 配置ssh免密登录

☁ 举例: 有node1和node2两台服务器, 实现node2访问node1时免密访问

代码块

```
1 # node2中生成密钥对
2 [node2]
3 ssh-keygen -f ~/.ssh/id_rsa -P '' -q
4 说明:
5     -f: 指定私钥文件的保存位置 (默认为: ~/.ssh/id_rsa)
6     -P: 指定设置私钥的密码 (一般为空, 不设置)
7     -q: 表示静默执行, 此时大部分信息不显示, 仅显示必要信息
8
9 cat ~/.ssh/id_rsa      # 查看生成的密钥
```

```
[root@node2 ~]# ssh-keygen -f ~/.ssh/id_rsa -P '' -q
[root@node2 ~]# ll .ssh/
total 8
-rw----- 1 root root 2610 Nov  8 18:50 id_rsa      私钥
-rw-r--r-- 1 root root  574 Nov  8 18:50 id_rsa.pub  公钥
[root@node2 ~]#
```

代码块

```
1 # 发送公钥到node1
2 ssh-copy-id 192.168.88.101      # node1的IP地址
```

```
[root@node2 ~]# ssh-copy-id 192.168.88.101
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host '192.168.88.101 (192.168.88.101)' can't be established.
ED25519 key fingerprint is SHA256:s5GZ8+ZC/UCjH0UPcoaZfJ8NTXtvp7dncmPHvf9B0k8.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  询问是否继续连接 输入 yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@192.168.88.101's password: 输入密码 由于此时免密配置未完成 仍需要输入密码使用

Number of key(s) added: 1

Now try logging into the machine, with: "ssh '192.168.88.101'"
and check to make sure that only the key(s) you wanted were added.

[root@node2 ~]#
```

1 在node1中查看是否发送成功: `~/.ssh/authorized_keys`

2 `cat ~/.ssh/authorized_keys`

```
[root@node1 ~]# cd .ssh/
[root@node1 .ssh]# ll
total 4
-rw-r----- 1 root root 574 Nov  8 18:54 authorized_keys
[root@node1 .ssh]# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAceKI3emdh9dR0n9AsfvhpWgISQnQPCrtV2e+LI2Ri8IS6gFVnmIkZwUgCI6D953FVBIPx2zco0ZtNIXTW92+XvbJxMJ/h/muPDp0GfxlJ5PZYwPlJhPP8QY976J+9GiuZB0HgyGrpdSc0Yt+j9D0/yy50Z5S9s739yTH9fTE
qhHwXL0ElV44ojB3vP/39DmHggqfUhmKwY0Yy6DF1bH4ZrdeNk69aovAXwPzwyjAvkL1LPgQ/wX34MeYe=Xt0sdLnXE1IX10jx2VRax500eRLumIj3Vyy+avUf5UCybk0Hk8X51EX16Xn05j+sPwXzo3GSHCKQow0PD/PdIV7A9tvsmsCJ7KEh8HXEE/o6Zs+T6ENzV4Rgr
Fcqj0cwS1cz6P0MIWSz7M4NACn767v1h7x/Ni7+/pog1BoAgTq+qebfpzqbdA3A5U0F9a9kTlfvpYQZEaj06u5AGolM6yLho9FhtsjICKgt1kpNQIsm0TmNSasNDUFLZyEdnfuV7W= root@node2.litcast.cn
[root@node1 .ssh]#
```

代码块

```
1 # 在node2通过ssh命令连接node1, 尝试是否免密访问
2 ssh root@192.168.88.101 # 或者 ssh 192.168.88.101, 默认用当前用户
```

```
[root@node2 ~]# ssh 192.168.88.101
Last login: Fri Nov  8 18:25:09 2024 from 192.168.88.1
[root@node1 ~]#
```

直接就连接成功, 不需要任何密码

## 4.基于ssh发送命令到远程服务器

SSH远程执行命令是通过SSH协议连接到远程服务器, 并在远程服务器上执行指定的命令或脚本, 而无需直接登录到服务器的终端


- **高效管理**: 无需登录每台服务器, 即可快速执行命令, 特别适合管理多台服务器
- **自动化运维**: 结合脚本工具, 实现批量操作、定时任务和自动化部署
- **快速故障排查**: 实时查看日志、监控状态、快速定位和解决问题
- **安全性**: 通过加密通信执行命令, 避免敏感信息泄露
- **节省时间**: 无需物理接触服务器, 即可完成维护任务, 特别适合分布式或云环境

代码块

```
1 # ssh语法结构
2 ssh [选项] username@remote_host "command [&]"
3
4 说明:
5 username: 远程服务器的用户名
6 remote_host: 远程服务器的IP地址或域名
7 command: 要在远程服务器上执行的命令
8
9 &: 异步执行, 在远程服务器后台运行, SSH不会等待命令完成, 会立即返回并关闭连接
```

```
10
11 选项：
12  -t: 如果远程命令需要交互式终端（例如 sudo 命令），可以使用 -t 选项
13  -p: 如果远程服务器的SSH服务运行在非默认端口（默认是22），可以使用 -p 选项指定端口
14  -o: 使用 -o 选项来指定SSH配置选项，例如禁用主机密钥检查（StrictHostKeyChecking=no）
15  -A: 需要转发认证代理连接（例如使用SSH密钥进行跳板机连接），可以使用 -A 选项
16
17  # 举例
18  # 在node1中远程查看node2的主机名
19  ssh root@192.168.88.102 "hostname"
20
21  # 在node1中，远程执行进入root用户家目录，并在此目录下创建 a.txt文件
22  ssh root@192.168.88.102 "cd ~ && touch a.txt"
```

## 5.基于ssh免密数据传输(scp)

 **SCP (Secure Copy Protocol)**：是基于 SSH 协议的安全文件传输工具，可用于在本地和远程服务器之间传输文件或目录。通过 SCP，数据在传输过程中会被加密，确保安全性

### 代码块

```
1  # scp语法结构
2  scp [选项] [源文件路径] [目标路径]
3
4  源文件路径：本地文件或远程服务器上的文件
5  目标路径：本地路径或远程服务器上的路径
6
7  常用选项：
8  -r: 递归复制整个目录（传输目录时必用）
9  -P: 指定 SSH 连接的端口号（默认是 22）
10 -i: 指定 SSH 使用的私钥文件（用于免密登录）
11 -q: 静默模式，不显示进度条和调试信息
12 -C: 开启压缩传输，提高传输速度（适合大文件或低速网络）
13 -l: 限制传输速度（单位为 Kbit/s，例如 -l 1000 表示限制为 1Mbps）
14
15 # scp基本用法
16 scp /local/path/file.txt user@remote:/remote/path/ # 从本地传输到远程
17 scp user@remote:/remote/path/file.txt /local/path/ # 从远程传输到本地
18 scp -r /local/path/dir user@remote:/remote/path/ # 递归传输整个目录
19
20 # 举例
```

```
21  # 在node1的/root目录下，构建ssh_test目录，并在此目录下创建a.txt b.txt文件，向文件中随
    机添加一些数据
22  cd ~
23  mkdir -p ./ssh_test
24  cd /root/ssh_test
25  touch a.txt b.txt
26  echo "a.txt 11111111" > a.txt
27  echo "b.txt 22222222" > b.txt
28
29  # 通过SCP命令，在node1完成将该目录拷贝到node2的/目录下
30  scp -r /root/ssh_test/ root@192.168.88.102:/
31
32  # 通过SCP命令，在node2的家目录完成将远端/root/ssh_test/a.txt拷贝到当前目录操作
33  cd ~
34  scp root@192.168.88.101:/root/ssh_test/a.txt ./
```