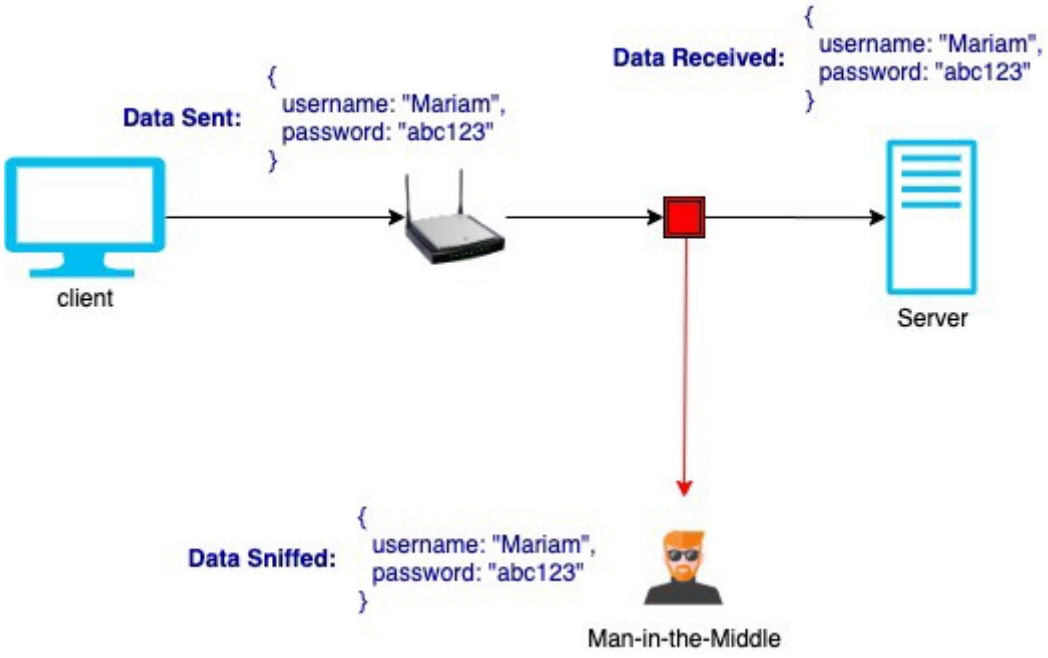
什么是SSH?

首先让我们来认识一下什么是SSH。

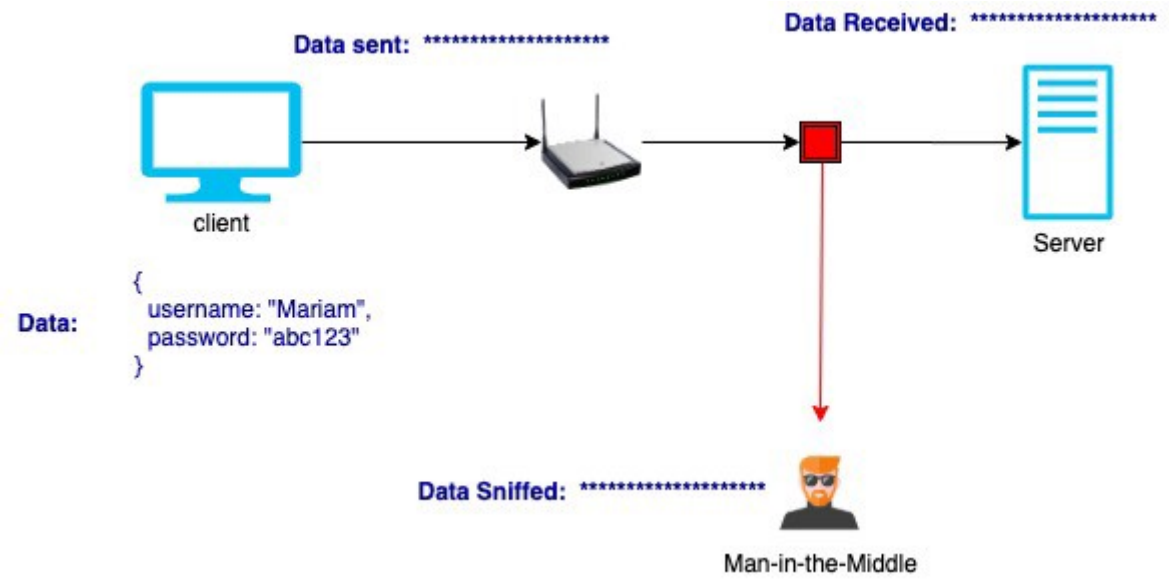
> SSH或Secure Shell是一种网络协议，它允许一台计算机通过共享通信方式协议，通过不安全的网络(例如Internet)安全地连接到另一台计算机。 SSH是应用程序层协议，它是OSI模型的第七层。
>
> SSH非常有用，因为不必物理访问另一台计算机。 可以简单地通过Internet连接到它。 这使我们可以远程控制服务器。
>
> SSH最早出现于90年代中期，被设计为Telnet的替代，后者也是一种无需加密即可传输数据的应用层协议。 如果不加密，数据将以纯文本格式在Internet上传输。 任何人在您与远程计算机之间使用数据包嗅探器时，都可以看到您正在传输的所有数据以及正在执行的所有操作。

以纯文本格式传输的数据



> 加密是一种隐藏数据的方法，除非知道如何解码或解密数据，否则它是不可读的。 SSH被创建为一种安全的通信方式，可以通过隧道对数据进行加密，因此不良行为者无法在传输过程中检索数据。 使用SSH，您仍然可以看到正在传输数据以及正在传输多少数据，但是看不到数据是什么。

中间人无法读取加密的数据



SSH通常使用**客户端-服务器模型来实现**。 一台计算机称为**SSH客户端**，另一台计算机称为**SSH服务器**或**主机**。

> 连接到SSH服务器时，您将进入外壳。 该外壳可以是Linux终端外壳，也可以是Windows命令提示符外壳，您可以在其中连接的计算机上执行命令。 当您使用终端或命令行时，您正在与操作系统进行通信。 使用SSH，您也可以与远程操作系统通信。

总之呢，**SSH就是是一个用来替代TELNET、FTP以及R命令的工具包**，主要是想**解决数据在网上明文传输的问题。最大特点就是安全。 通过使用SSH，你可以把所有传输的数据进行加密**，这样"中间人"这种攻击方式就不可能实现了，而且也能够防止DNS欺骗和IP欺骗。总的来说就是一种**安全的进行远程登陆的方式**，也是现在最常用的登陆服务器的方式。

通过SSH传输什么?

SSH可用于传输:

- Data

  数据
- Commands

  指令
- Text

  文本
- Files (Using SFTP: Secure File Transfer Protocol, basically an encrypted version of FTP that makes it that man-in-the-middle attacks are not possible)

  文件(使用SFTP：安全文件传输协议，**基本上是FTP的加密版本，无法进行中间人攻击**)

SSH如何工作?

如上图所示，加密数据包

为了保持SSH的安全性，SSH在传输过程中的各个时间点使用三种不同类型的数据处理技术。 SSH中使用的三种技术是：

1. Symmetrical Encryption

   对称加密

2. Asymmetrical Encryption

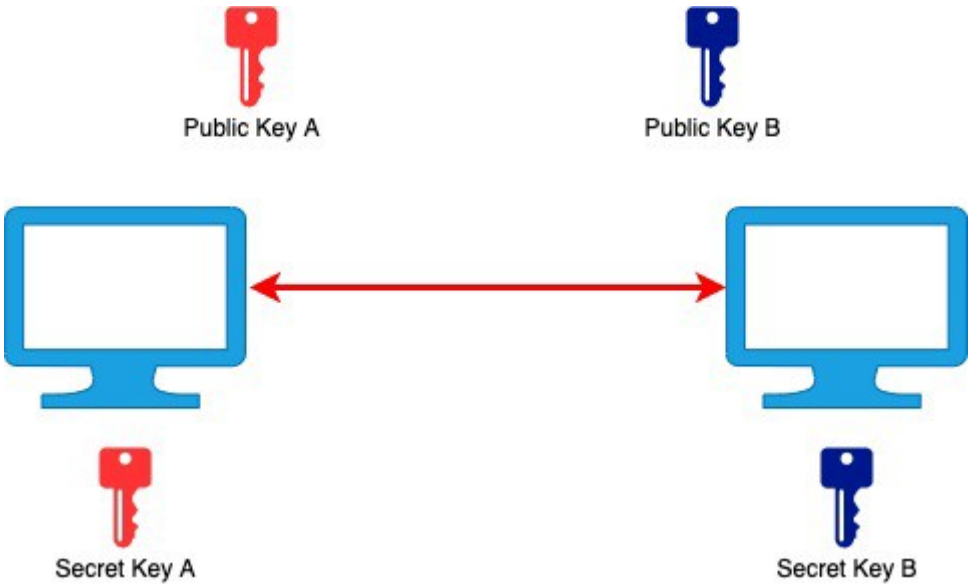
   非对称加密

3. Hashing

   散列

1.*对称加密*

- **一个密钥可用于加密发送到目的地的消息，也可以解密在目的地接收的消息。 此加密方案也称为共享秘密加密或共享密钥加密**



2.*非对称加密*

- **使用两个单独的密钥进行加密和解密** 即公共密钥和私有密钥
- **可以与任何人共享公共密钥，但永远不会共享私有密钥**
- **公钥存储在SSH服务器上，私钥存储在SSH客户端本地。**
- “ 如果我给您我的公共密钥，您可以通过使用我的公共密钥加密它来向我发送消息。 然后，我将可以通过使用我的私钥对其解密来读取它。“

3.*散列 (Hashing)*

> 总之就是使用散列函数，使得传输的每个消息都必须包含称为MAC的内容。 该MAC是根据对称密钥，数据包序列号和已发送的消息内容生成的哈希。 使用这三个元素组合作为哈希函数的输入，此哈希函数将输出一些没有意义的字符串。 该字符串或**签名**被发送到主机。

SSH分为两部分：客户端部分和服务端部分

服务端是一个守护进程(demon)，他在后台运行并响应来自客户端的连接请求。服务端一般是sshd进程，提供了对远程连接的处理，一般包括公共密钥认证、密钥交换、对称密钥加密和非安全连接。

```
sshd
openssh软件套件中的服务器守护进程

补充说明
sshd命令 是openssh软件套件中的服务器守护进程。

语法
sshd(选项)
选项
-4：强制使用IPv4地址；
-6：强制使用IPv6地址；
-D：以后台守护进程方式运行服务器；
-d：调试模式；
-e：将错误发送到标准错误设备，而不是将其发送到系统日志；
-f：指定服务器的配置文件；
-g：指定客户端登录时的过期时间，如果在此期限内，用户没有正确认证，则服务器断开次客户端的连接；
```

-h: 指定读取主机key文件;
-i: ssh以inetd方式运行;
-o: 指定ssh的配置选项;
-p: 静默模式，没有任何信息写入日志;
-t: 测试模式。

客户端包含ssh程序以及像scp（远程拷贝）、slogin（远程登陆）、sftp（安全文件传输）等其他的应用程序。这些也都是我们比较常用的命令。

scp
加密的方式在本地主机和远程主机之间复制文件

补充说明
scp命令 用于在Linux下进行远程拷贝文件的命令，和它类似的命令有cp，不过cp只是在本机进行拷贝不能跨服务器，而且scp传输是加密的。可能会稍微影响一下速度。当你服务器硬盘变为只读read only system时，用scp可以帮你把文件移出来。另外，scp还非常不占资源，不会提高多少系统负荷，在这一点上，rsync就远远不及它了。虽然 rsync比scp会快一点，但当小文件众多的情况下，rsync会导致硬盘I/O非常高，而scp基本不影响系统正常使用。

语法
scp(选项)(参数)
选项
-1: 使用ssh协议版本1;
-2: 使用ssh协议版本2;
-4: 使用ipv4;
-6: 使用ipv6;
-B: 以批处理模式运行;
-C: 使用压缩;
-F: 指定ssh配置文件;
-i: identity_file 从指定文件中读取传输时使用的密钥文件（例如亚马逊云pem），此参数直接传递给ssh;
-l: 指定宽带限制;
-o: 指定使用的ssh选项;
-P: 指定远程主机的端口号;
-p: 保留文件的最后修改时间，最后访问时间和权限模式;
-q: 不显示复制进度;
-r: 以递归方式复制。

实例
从远程机器复制文件到本地目录
```
scp root@10.10.10.10:/opt/soft/nginx-0.5.38.tar.gz /opt/soft/
```
从10.10.10.10机器上的/opt/soft/的目录中下载nginx-0.5.38.tar.gz 文件到本地/opt/soft/目录中。
上传本地文件到远程机器指定目录
```
scp /opt/soft/nginx-0.5.38.tar.gz root@10.10.10.10:/opt/soft/scptest
# 指定端口 2222
scp -rp -P 2222 /opt/soft/nginx-0.5.38.tar.gz root@10.10.10.10:/opt/soft/scptest
```
复制本地/opt/soft/目录下的文件nginx-0.5.38.tar.gz到远程机器10.10.10.10的opt/soft/scptest目录。

值得一提的是我们现在常用的SSH并不是真正的SSH，而是另一种替代的版本OPENSSH，毕竟这里涉及到加密算法和版权的限制。

在自己的电脑上搭建一个SSH服务器

1. **安装SSH 服务器**

```
sudo apt-get install openssh-server
```

2. 启动

```
sudo service ssh start
```

它会告诉你正在运行，第一步就ok了（一定要记得sudo，否则他会骗你说没有这个服务的0.0）。接下来就需要处理配置文件了，openssh的配置文件是/etc/ssh/sshd_config，（不是ssh_config）

3. **搞配置文件**

```
gongna@gongna-Ubuntu:/etc/ssh$ cat sshd_config
# Package generated configuration file
# See the sshd_config(5) manpage for details
# What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes
# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024
# Logging
SyslogFacility AUTH
LogLevel INFO
# Authentication:
LoginGraceTime 120
PermitRootLogin without-password
StrictModes yes
RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile %h/.ssh/authorized_keys
# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# For this to work you will also need host keys in /etc/ssh_known_hosts
RhostsRSAAuthentication no
# similar for protocol version 2
HostbasedAuthentication no
# Uncomment if you don't trust ~/.ssh/known_hosts for RhostsRSAAuthentication
#IgnoreUserKnownHosts yes
# To enable empty passwords, change to yes (NOT RECOMMENDED)
PermitEmptyPasswords no
# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
```

```
ChallengeResponseAuthentication no
# Change to no to disable tunnelled clear text passwords
#PasswordAuthentication yes
# Kerberos options
#KerberosAuthentication no
#KerberosGetAFSToken no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
# GSSAPI options
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes
X11Forwarding yes
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes
#UseLogin no
#MaxStartups 10:30:60
#Banner /etc/issue.net
# Allow client to pass locale environment variables
AcceptEnv LANG LC_*
Subsystem sftp /usr/lib/openssh/sftp-server
# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes
```

呃.....有点长，我们捡几个重（jian）要（dan）的研究下。

1. Port：这个很明显就是记录SSH的端口啦，默认的是22，自己也可以改，（虽然并不晓得改完会不会出问题）
2. Protocol：这个指的是SSH的版本，众所周知有两个版本1和2，据说兼容行不好，如果确实要兼容的话，值应该设置为2,1。我这里是2。
3. HostKey：这里有好几个路径，指的就是private-key的保存路径，闲着没事可以打开看看。
4. ServerKeyBits：就是密钥的长度啦，估计随着时间的推移这个数字会越来越大吧。
5. PermitRootLogin： 是否允许 root 登入！预设是允许的，但是很明显这是不安全的，我们通常把这个改为no，但是我知道我在干什么所以我一般把这一栏改成yes
6. StrictModes： 当使用者的 host key 改变之后，Server 就不接受联机 当使用者的 host key 改变之后，Server 就不接受联机，为了安全一般是

然后呢，就可以通过远程计算机输入“ssh 用户名@主机名” 进行远程登陆了。

画重点！！！但我们一般都是团队直接买服务器，所以如果是你的项目要上线，部署在团队的服务器上，直接找组长要团队的服务器IP啥的就阔以了。

Linux下使用 SSH 进行登陆

```
ssh user@192.168.1.2
```

更简单的方法是打开你的Linux电脑，打开文件管理

点击其他位置，右下方就有连接到服务器，然后输入连接就好了。

over～