32 连接远程终端和对称加密，安全快捷

更新时间：2019-07-26 09:34:30

> 富贵必从勤苦得。
>
> ——杜甫

1. 前言

上一课是 带你玩转Linux和Shell编程 | 第四部分第一课：压缩文件，解压无压力，比较轻松。

> 我想：这一课和下一课的内容将让你大呼过瘾，也许是本系列课程到目前为止最有意思的课之一。

我们会聊很多内容，会学习如何连接到远程的一台安装有 Linux 的机器。

在本系列课程的开头，我们略微提到过：可以配置安装 Linux 的机器，只要机器开着，就可以远程连接它。

我们不仅会学习如何远程连接到 Linux，还会学习其原理。

而且也会学习 SSH 协议，以及数据如何通过 SSH 加密，以及如何用 SSH 连接到远程电脑。

因此，我们可以一窥网络和安全（密码学）的奇幻世界。

> 设想一下：
> 你家里的电脑安装了 Linux 系统，处于开机状态，你在工作的时候想起来要用家里的电脑进行某项下载任务或者提取某个文件。
> 此时，你可以远程连接到家里的 Linux 系统，打开一个终端，就好像你坐在家里的电脑面前一样。
> 我们到目前为止所学关于终端的所有操作，你都可以在远程连接到的机器上实现，哪怕这台机器是在世界的另一个角落。

这一课对各位站长特别有用，因为我们可以远程连接到自己网站的服务器。

现在越来越多的人购买了服务器，在服务器上可以架设自己的网站，或提供服务（比如游戏、文件服务器）。

一般来说，对服务器的操作都需要远程连接（因为我们一般都是租用服务器提供商的远端电脑，除非你有钱“任性”，那你可以买几台电脑来做服务器；或者省钱一些的，可以买树莓派之类的来搭建自己的服务器），用终端来进行，而且基本都是用 SSH 协议来连接。

2. 连接到远程终端

到目前为止，我们本课程中所学的 Linux 的操作，都是我们本人坐在电脑前去操作自己面前的电脑。可能与我们原先还没学编程，使用 Windows 玩游戏没什么区别。

> 怎奈我们不甘寂寞（“一生有一种大海的气魄，岁月一页页无情翻过。那乾坤留在我心中的一刻，就已经注定我不甘寂寞…”），在一个地方待久了，总想去外面闯闯、看看。

Linux 的一个强大之处就是可以很方便地操作很远之处的另一台安装有 Linux 的电脑。这个功能早在 Unix 时代就已经有了。

当然了，Windows 系统也可以远程连接到另一个 Windows 系统，我们也可以从 Linux 系统连接到 Windows 系统、macOS 系统，或者从 Windows 连接到 Linux、macOS等等，只不过使用起来没有 Linux 下那么顺手。

今天，比如我住在中国，我可以很方便地远程连接到位于美国纽约的一台电脑。也可以连接到日本东京的一台电脑。我也可以操作美国纽约的那台电脑给日本东京的电脑发送文件。

因为 Internet（互联网或英特网）的出现，这样的事情在今天看来已经很平常了。

远程连接技术大大节约了时间和精力。假如没有这个技术，专门负责维护远程服务器的技术人员（通常称为 System Administrator，系统管理员）要给东京机房的电脑安装一个软件，那还得买飞机票去东京…

服务器，英语是 server。一般是指 7 天 24 小时不关机的电脑，这些电脑和你家里的电脑类似，但又不一样（通常更强劲，噪音更大）：也有一个处理器，一个或多个硬盘等等。服务器的主要特色是保持开机，始终连接 Internet，提供服务。

我们把连接到服务器的机器称为客户机，英语是 client，我们本课之后的图片中也会像下图这样来表述：

客户机

服务器

目前，你家里的电脑应该还不能被称为一台服务器。当然，如果你愿意，你也可以将其“变为”一台服务器，只需要安装一些软件，做一些配置。

还有就是：不关机。

因为一台关机的服务器，就不是能“服务”的机器了。作为服务器必须“任劳任怨，随叫随到；宕机重启，永不言弃”。

接下来，我们将有请今天的主角：**SSH** 上场。

不过我们会先讲很多其它的相关知识点。

> 前方密集知识点高能预警。准备好了吗？

我们将会以下面的顺序来介绍 SSH：

- 为什么要保护网络通信
- SSH 又是如何保护网络通信的
- SSH 的具体使用方法
- 从 Telnet 协议到 SSH 协议

协议

两台机器在互联网上通信的时候，需要遵循相同的协议。这不难理解，生活中，两家公司要合作，要协商，也需要拟定一份协议，双方都同意，并且签署，遵守。一旦违反协议，交流合作即会结束。

比如我们中国的主席访问外国，和他们签署一系列商业协议；只有双方都遵循此协议，合作交流才能继续。

同样的，对于我们的互联网（Internet），要在其上相互通信，也需要一定的协议。互联网毕竟是人创立的嘛，肯定引入了人的惯有思维。

或者我们也可以把互联网协议（参看 网络协议 ）比作一种语言：只有都说英语或都说中文才能相互交流。

互联网协议有很多，就不一一列举了，可以看上面百度百科链接里的列表。因此，两台机器之间可以有不少种方式相互通信（每种协议对应一种通信方式）。

身为读者的你肯定已经见过其中的一种协议了，那就是 HTTP 协议。只要你上网浏览网页，那么你其实一直在使用这种协议，例如 Github 的官网地址：

https://github.com

看到最前面的 https 这几个英文字母了吗？这就说明它遵循的是 HTTPS 协议。

因此，你的电脑能看到 Github 的官方网站，是因为你的电脑和 Github 的服务器之间使用了 HTTPS 协议来互相通信。

HTTP 是英语 Hyper Text Transfer Protocol 的缩写，表示“超文本传输协议”，protocol 是英语“协议”的意思。

HTTP 协议是 Web（Web 可以简单翻译为“网络”）最常用的传输网页协议。

HTTPS 是 Hyper Text Transfer Protocol Secure 的缩写，表示“安全的超文本传输协议”，简单来说就是 HTTP 协议的安全版本。

HTTPS 是 HTTP 的扩展。HTTP 的所有传输的内容都是明文。而 HTTPS 使用 TLS（Transport Layer Security 的缩写，表示“传输层安全性协议”）或其前身 SSL（Secure Sockets Layer 的缩写，表示“安全套接字层”），所有传输的内容都经过加密。

目前大部分网站都已经从使用 HTTP 协议改为 HTTPS 协议了。例如 Chrome 这样的浏览器，如果你的网址是以 http 开头，则会被标记为“不安全”。

其它常见的网络协议有 FTP，是英语 File Transfer Protocol 的简称，表示“文件传输协议”。一般站长常用 FTP，因为需要把文件（比如网站的源代码）传输到远程的网站服务器上。以前在大学时代，学校也会有公共的 FTP 站点，供学生们下载需要的学习资料。

还有我们每天使用电子邮件（Email）所需要遵循的协议，常见的有 IMAP 和 POP3。IMAP 是 Internet Message Access Protocol（互联网消息访问协议）的缩写，POP3 是 Post Office Protocol Version 3（邮局协议第三版）的缩写。一般我们使用 Office Outlook、Thunderbird、Gmail、163 邮箱、QQ 邮箱等，都需要遵循相应的邮件协议。

Telnet 协议：简单易用危险多

既然我们这课的重点是讲远程连接，那么我们就来谈谈远程连接的协议好了。首先，介绍一个老古董：

Telnet

这个协议简单、易用，在 20 世纪 80 年代就被创立了，它的功用就是在机器间传输简单信息。

理论上，我们就可以使用 Telnet 来与远端机器通信啦，比如与我们的服务器通信。

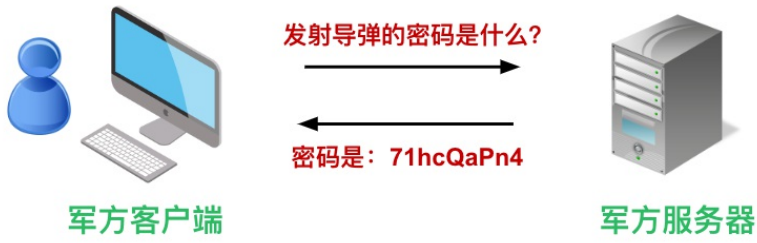
但是这个协议有什么缺点呢？那就是它太简单太基础了，因此传输的信息并没有经过加密，而是明文传输。

在密码学（Cryptography）中，明文（英语是 plaintext 或 cleartext）是指传送方想要接收方获得的可读信息（比如“我爱你”，“Hello World”）。

明文经过加密所产生的信息被称为密文（英语是 ciphertext 或 cyphertext），而密文经过解密还原得来的信息就是明文。

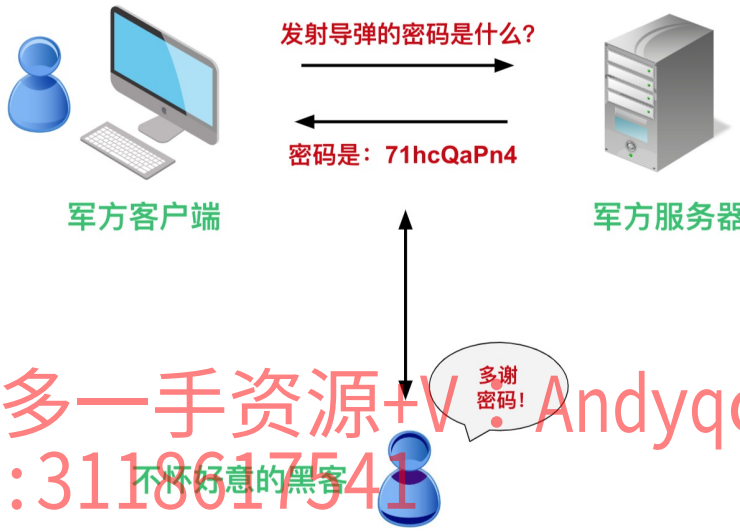
如果在互联网上传输明文，那是很危险的。假设以下场景：

一台军方的客户端电脑请求军方的服务器提供发射导弹（可能是核弹...）的密码，服务器用明文将密码传输给客户端。如下图所示：



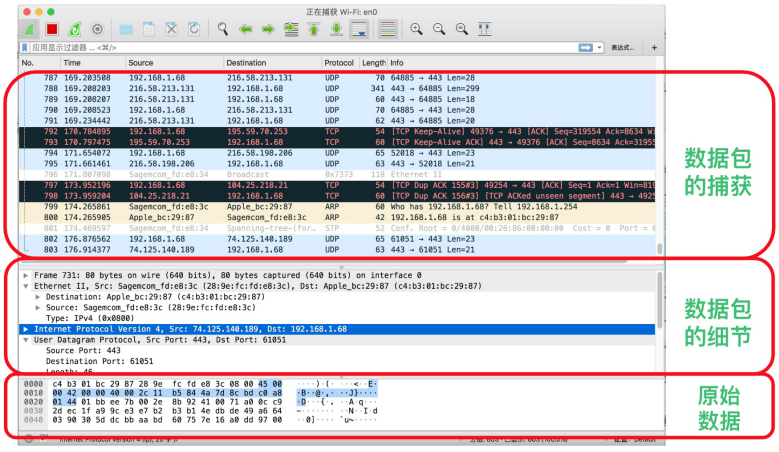
从上图看来，并没有什么风险不是吗？因为信息只传递给了发出请求的客户机。

但是，很有可能一个不怀好意的黑客，监听了上面两者通信的信息，就可以截取军方服务器发给军方客户机的明文信息了。如下图所示：



要阻止这样的黑客行为是很难的。虽说拦截数据这样的操作本身有难度，但是水平高的黑客是可以轻易做到的。

比如，像 Wireshark 这样的软件就可以被用于监听网络（尤其是本地网络），可以截获数据。如下图所示（是我用 Wireshark 来嗅探我的无线网连接）：



你也许会说：“呃… 等等。我只是想远程连接到我的电脑或服务器以便访问终端控制台。我可不会共享核弹密码！况且我在电脑上只是运行 grep 之类的命令，被人知道也无妨啊。”

你的意思是你并不介意别人窥探你的电脑咯？好吧，你赢了...

但是当你连接到服务器时，你需要提供你的登录名和密码，假如被人窃取就麻烦了。

所以，必须加密网络间传输的信息，你应该不希望自己的密码被人知道吧。

SSH 协议：保护信息的好方法

既然我们并不能阻止心怀不轨的人试图截取 Internet 上传输的信息，那么我们就要采取一些措施，使客户机和服务器之间以安全的方式通信。加密技术就是专门替我们做这等差事的：假如黑客获取了加密后的密码，他便不能做什么。

但是怎么加密数据呢？客官莫急，待我把主角 SSH 请上来~

3. 使用 SSH 的信息交换是如何加密的

SSH 是英语 Secure SHell 的缩写，直译过来就是“安全的 Shell”，shell 是英语“壳”的意思。

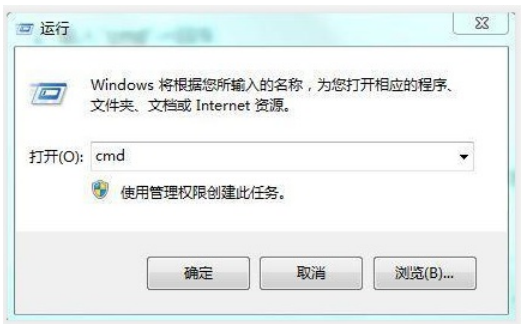
> 在计算机科学中，shell 俗称“壳”（自然界中的贝壳的英语也是 shell。“壳”用来区别于“核”（kernel），一个是外壳，一个是核心），是指“提供用户使用界面”的软件（命令解析器）。
> 它类似于 DOS 下的 command（英语“命令”的意思）和后来的 cmd.exe。
> 它接收用户命令，然后调用相应的应用程序。

Linux 下有 Bash 等 Shell 程序，之后我们有专门的一大部分会讲 Shell 程序，不要担心。

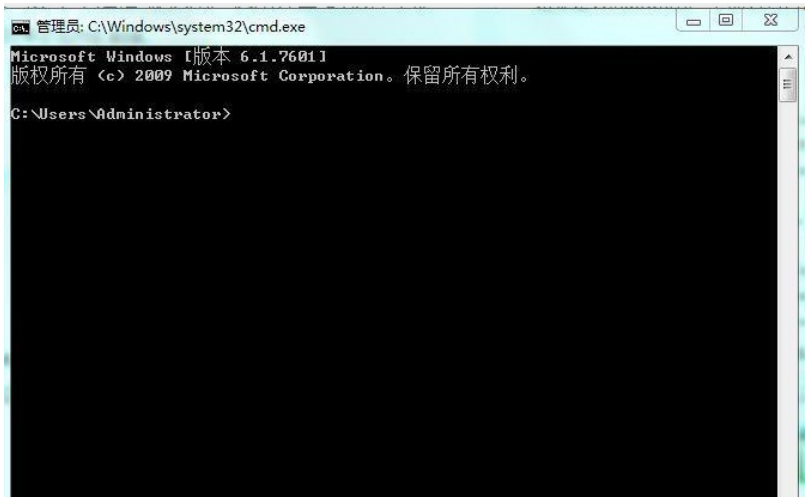
如果是 Windows 用户，那很有可能知道 cmd（是 command 的缩写）这个命令吧。我们有时会使用到它。如何运行这个命令呢？

- 点击：开始 -> 运行

- 输入“cmd” -> 回车



- 在出现的黑窗口里输入命令，回车运行



> cmd 确实比较低端... 不过微软终于在 2019 年 5 月推出了全新的终端，叫 Windows Terminal，还是不错的（虽然更新得有点晚...）。大家可以去看一下：https://github.com/microsoft/Terminal

不可讳言，SSH 协议本身比较复杂，但是如果能对其大致原理有所了解，岂不快哉？

如果知其所以然，那使用的时候也不会那么盲目了。所以我们从以下两条主线来了解 SSH：

1. 有哪些不同的加密方法
2. SSH 又是如何运用加密方法来保护数据的

不同的加密方法

说到加密方法，就要涉及到相应的算法了。什么是算法呢？

> 百度百科的解释：
> 算法（Algorithm）是指解题方案的准确而完整的描述，是一系列解决问题的清晰指令，算法代表着用系统的方法描述解决问题的策略机制。

因此，程序员需要掌握的 “数据结构与算法”里的算法，是很重要的。

所以下面我们就直接用“加密算法”这个词了。

要说不同的加密算法，那实在是三天三夜讲不完啊，所以我就不在这给大家“一千零一夜”了。

虽说我们不能了解所有的加密算法（也没那个必要），但是我们需要知道加密算法大致分两类：

- 对称加密
- 非对称加密

对称加密

对称加密，英语是 Symmetric-Key Encryption，symmetric 是“对称的”意思，key 是“钥匙/密钥”的意思，encryption 是“加密”的意思。所以全称其实是“对称密钥加密”。

对称加密是比较简单的加密算法，但简单并不意味着不保险（有很安全的对称加密算法）。简单意味着功能比较好理解。

对称加密算法用一个密钥（就是　key，是在明文转换为密文或将密文转换为明文的算法中输入的参数）来加密信息。

举个例子，假如此密钥叫 superkey（super + key，意为“超级密钥”。super 是“超级的、特级的”意思），而需要被加密的信息是 message（英语“信息”的意思），那么加密过程如下图所示：

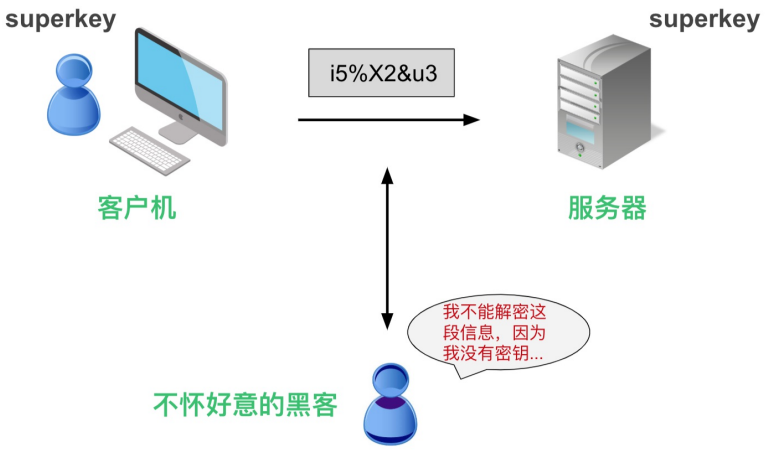
上图是对称加密的加密过程的简单演示。传输方用 superkey 这个密钥将 message 这个明文信息进行加密，成为 i5%X2&u3 这样的密文。

之后，接收方用同样的密钥 superkey，对 i5%X2&u3 这个密文进行解密，就重新得到了明文 message。解密的过程如下图所示：

上图就是对称加密中用同样的密钥 superkey 进行解密的过程。

因此，不难理解，对称加密中“对称”的意思就是指加密和解密使用的是同一个密钥。因此加密方和解密方都须要知道这个密钥。

这样，假如黑客截获了 i5%X2&u3 这个传输的密文，他没有解密的密钥 superkey，他就不知道究竟是什么明文信息了。如下图所示：

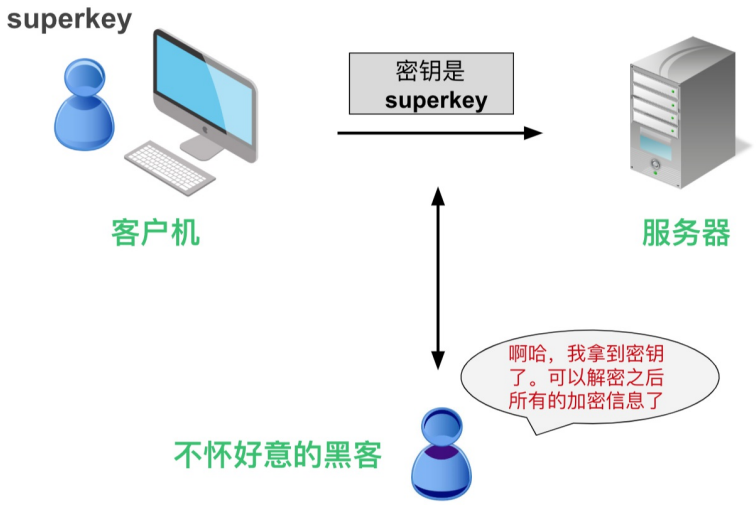


你会说：“这样很不错啊！”

但是，客户机和服务器都须要知道这个密钥。因此客户机首先要把这个密钥传给服务器，为了让服务器可以解密那些加密过的信息。

实际上，为了达到上图的目的，客户机和服务器必须事先传递那个密钥 superkey。

但是他们怎么传递呢？假如他们传递密钥用的是明文，那么黑客照样可以截获密钥，接下来就可以解密任何传递的加密信息了，不是吗？如下图所示：



因此，对称加密虽然强大，但是有一个致命的缺陷：必须谨慎地传递密钥。但这几乎是不可能的：因为首先得把密钥传递过去。

你也许会问：为什么我们不干脆把传递的密钥也加密呢？

好问题，你的想法是对的。看来你很聪明~

为了加密用于对称加密的密钥（有点绕...），我们将用另一种方法：

非对称加密

只要用了这个方法，我们就不用担心密钥被黑客获取了。

篇幅关系，我们下一课再讲解非对称加密。

4. 总结

1. 我们可以远程连接到 Linux 系统，进入它的终端，我们一般都是这样操作远程服务器的；
2. 我们一般把连接到远程 Linux 服务器的电脑称为 client，就是客户。我们可以从多种操作系统远程连接到 Linux 的终端，比如从 Windows、Linux、macOS、Unix 系统。

今天的课就到这里，一起加油吧！

31 压缩文件，解压无压力
33 非对称加密和使用SSH建立安全连接