如何在 Linux 中配置基于密钥认证的 SSH

Linux爱好者 昨天

> 英文：SK，翻译：Linux中国/LuMing
> linux.cn/article-10086-1.html

什么是基于 SSH 密钥的认证?

众所周知，Secure Shell，又称 SSH，是允许你通过无安全网络（例如 Internet）和远程系统之间安全访问/通信的加密网络协议。无论何时使用 SSH 在无安全网络上发送数据，它都会在源系统上自动地被加密，并且在目的系统上解密。SSH 提供了四种加密方式，基于密码认证，基于密钥认证，基于主机认证和键盘认证。最常用的认证方式是基于密码认证和基于密钥认证。

在基于密码认证中，你需要的仅仅是远程系统上用户的密码。如果你知道远程用户的密码，你可以使用 ssh user@remote-system-name 访问各自的系统。另一方面，在基于密钥认证中，为了通过 SSH 通信，你需要生成 SSH 密钥对，并且为远程系统上传 SSH 公钥。每个 SSH 密钥对由私钥与公钥组成。私钥应该保存在客户系统上，公钥应该上传给远程系统。你不应该将私钥透露给任何人。希望你已经对 SSH 和它的认证方式有了基本的概念。

这篇教程，我们将讨论如何在 Linux 上配置基于密钥认证的 SSH。

在 Linux 上配置基于密钥认证的 SSH

为方便演示，我将使用 Arch Linux 为本地系统，Ubuntu 18.04 LTS 为远程系统。

本地系统详情：

- OS: Arch Linux Desktop
- IP address: 192.168.225.37/24

远程系统详情：

- OS: Ubuntu 18.04 LTS Server
- IP address: 192.168.225.22/24

本地系统配置

就像我之前所说，在基于密钥认证的方法中，想要通过 SSH 访问远程系统，需要将公钥上传到远程系统。公钥通常会被保存在远程系统的一个 ~/.ssh/authorized_keys 文件中。

注意事项：不要使用 root 用户生成密钥对，这样只有 root 用户才可以使用。使用普通用户创建密钥对。

现在，让我们在本地系统上创建一个 SSH 密钥对。只需要在客户端系统上运行下面的命令。

```
$ ssh-keygen
```

上面的命令将会创建一个 2048 位的 RSA 密钥对。你需要输入两次密码。更重要的是，记住你的密码。后面将会用到它。

样例输出：

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/sk/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sk/.ssh/id_rsa.
Your public key has been saved in /home/sk/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:wYOgvdkBgMFydTMCUI3qZaUxvjs+p2287Tn4uaZ5KyE [email protected]
The key's randomart image is:
+---[RSA 2048]----+
|+=+*= + |
|o.o=.* = |
|.oo * o + |
|. = + . o |
|. o + . S |
| . E . |
| + o |
| +.*o+o |
| .o*=OO+ |
+----[SHA256]-----+
```

如果你已经创建了密钥对，你将看到以下信息。输入 y 就会覆盖已存在的密钥。

```
/home/username/.ssh/id_rsa already exists.
Overwrite (y/n)?
```

请注意密码是可选的。如果你输入了密码，那么每次通过 SSH 访问远程系统时都要求输入密码，除非你使用了 SSH 代理保存了密码。如果你不想要密码（虽然不安全），简单地敲两次回车。不过，

我建议你使用密码。从安全的角度来看，使用无密码的 ssh 密钥对不是什么好主意。这种方式应该限定在特殊的情况下使用，例如，没有用户介入的服务访问远程系统。（例如，用 rsync 远程备份……）

如果你已经在个人文件 ~/.ssh/id_rsa 中有了无密码的密钥，但想要更新为带密码的密钥。使用下面的命令：

```
$ ssh-keygen -p -f ~/.ssh/id_rsa
```

样例输出：

```
Enter new passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved with the new passphrase.
```

现在，我们已经在本地系统上创建了密钥对。接下来，使用下面的命令将 SSH 公钥拷贝到你的远程 SSH 服务端上。

```
$ ssh-copy-id sk@192.168.225.22
```

在这里，我把本地（Arch Linux）系统上的公钥拷贝到了远程系统（Ubuntu 18.04 LTS）上。从技术上讲，上面的命令会把本地系统 ~/.ssh/id_rsa.pub 文件中的内容拷贝到远程系统 ~/.ssh/authorized_keys 中。明白了吗？非常棒。

输入 yes 来继续连接你的远程 SSH 服务端。接着，输入远程系统用户 sk 的密码。

```
/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
sk@192.168.225.22's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'sk@192.168.225.22'"
and check to make sure that only the key(s) you wanted were added.
```

如果你已经拷贝了密钥，但想要替换为新的密码，使用 -f 选项覆盖已有的密钥。

```
$ ssh-copy-id -f sk@192.168.225.22
```

我们现在已经成功地将本地系统的 SSH 公钥添加进了远程系统。现在，让我们在远程系统上完全禁用掉基于密码认证的方式。因为我们已经配置了密钥认证，因此不再需要密码认证了。

在远程系统上禁用基于密码认证的 SSH

你需要在 root 用户或者 sudo 执行下面的命令。

禁用基于密码的认证，你需要在远程系统的终端里编辑 /etc/ssh/sshd_config 配置文件：

```
$ sudo vi /etc/ssh/sshd_config
```

找到下面这一行，去掉注释然后将值设为 no：

```
PasswordAuthentication no
```

重启 ssh 服务让它生效。

```
$ sudo systemctl restart sshd
```

从本地系统访问远程系统

在本地系统上使用命令 SSH 你的远程服务端：

```
$ ssh sk@192.168.225.22
```

输入密码。

样例输出：

```
Enter passphrase for key '/home/sk/.ssh/id_rsa':
Last login: Mon Jul 9 09:59:51 2018 from 192.168.225.37
sk@ubuntuserver:~$
```

现在，你就能 SSH 你的远程系统了。如你所见，我们已经使用之前 ssh-keygen 创建的密码登录进了远程系统的账户，而不是使用当前账户实际的密码。

如果你试图从其它客户端系统 ssh（远程系统），你将会得到这条错误信息。比如，我试图通过命令从 CentOS SSH 访问 Ubuntu 系统：

样例输出：

```
The authenticity of host '192.168.225.22 (192.168.225.22)' can't be established.
ECDSA key fingerprint is 67:fc:69:b7:d4:4d:fd:6e:38:44:a8:2f:08:ed:f4:21.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.225.22' (ECDSA) to the list of known hosts.
Permission denied (publickey).
```

如你所见，除了 CentOS（LCTT 译注：根据上文，这里应该是 Arch）系统外，我不能通过其它任何系统 SSH 访问我的远程系统 Ubuntu 18.04。

为 SSH 服务端添加更多客户端系统的密钥

这点非常重要。就像我说过的那样，除非你配置过（在之前的例子中，是 Ubuntu），否则你不能通过 SSH 访问到远程系统。如果我希望给更多客户端予以权限去访问远程 SSH 服务端，我应该怎么做？很简单。你需要在所有的客户端系统上生成 SSH 密钥对并且手动拷贝 ssh 公钥到想要通过 ssh 访问的远程服务端上。

在客户端系统上创建 SSH 密钥对，运行：

```
$ ssh-keygen
```

输入两次密码。现在，ssh 密钥对已经生成了。你需要手动把公钥（不是私钥）拷贝到远程服务端上。

使用以下命令查看公钥：

```
$ cat ~/.ssh/id_rsa.pub
```

应该会输出类似下面的信息：

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCt3a9tIeK5rPx9p74/KjEVXa6/OODyRp0QLS/sLp8W6iTxFL+UgALZlupV
NgFjvRR5luJ9dLHWwc+d4umavAWz708e6Na9ftEPQtC28rTFsHwmyLKvLkzcGkC5+A0NdbiDZLaK3K3wgq1jzYYKT
5k+IaNS6vtrx5LDObcPNPEBDt4vTixQ7GZHrDUUk5586IKeFfwMCWguHveTN7ykmo2EyL2rV7TmYq+eY2ZqqcsoK0
fzXMK7iifGXVmuqTkAmZLGZK8a3bPb6VZd7KFum3Ezbu4BXZGp7FVhnOMgau2kYeOH/ItKPzpCAn+dg3NAAziCCx
nII9b4nSSGz3mMY4Y7 ostechnix@centosserver
```

拷贝所有内容（通过 USB 驱动器或者其它任何介质），然后去你的远程服务端的终端，像下面那样，在 $HOME 下创建文件夹叫做 .ssh。你需要以 root 身份执行命令（注：不一定需要 root）。

```
$ mkdir -p ~/.ssh
```

现在，将前几步创建的客户端系统的公钥添加进文件中。

```
echo {Your_public_key_contents_here} >> ~/.ssh/authorized_keys
```

在远程系统上重启 ssh 服务。现在，你可以在新的客户端上 SSH 远程服务端了。

如果觉得手动添加 ssh 公钥有些困难，在远程系统上暂时性启用密码认证，使用 ssh-copy-id 命令从本地系统上拷贝密钥，最后禁用密码认证。

【关于投稿】

如果大家有原创好文投稿，请直接给公号发送留言。

① 留言格式：
【投稿】+《 文章标题》+ 文章链接

② 示例：
【投稿】《不要自称是程序员，我十多年的 IT 职场总结》：http://blog.jobbole.com/94148/

③ 最后请附上您的个人简介哈~