SSH: Secure Shell

A possibilidade de conexão por terminal remoto é essencial para a administração de sistemas. O comando `telnet` permite estabelecer conexões de terminal texto com hosts UNIX. Dentro de uma sessão `telnet`, todos os comandos são executados na máquina-alvo, mas as interações com o usuário são apresentadas localmente, na máquina onde o usuário se encontra.

O serviço `telnet` é considerado inseguro, sobretudo em redes públicas. Isso porque as informações de uma `telnet` circulam pela rede de forma aberta, sem criptografia. Algum usuário que esteja "ouvindo" a rede pode capturar pacotes com logins, senhas, etc, comprometendo a segurança do sistema. Por isso, recomenda-se o uso de alternativas criptografadas para efetuar conexões de terminais remotos. Uma boa alternativa ao `telnet` é o serviço *Secure Shell* (SSH).

O serviço SSH - Secure Shell (porta 22/TCP) substitui o telnet com diversas vantagens, sendo a principal o fato de criar um "túnel" criptografado para o tráfego das informações da sessão de trabalho. O SSH faz uso do serviço SSL - *Secure Sockets Layer*, uma camada de software que cria a noção de sessão de trabalho segura sobre sockets TCP convencionais.

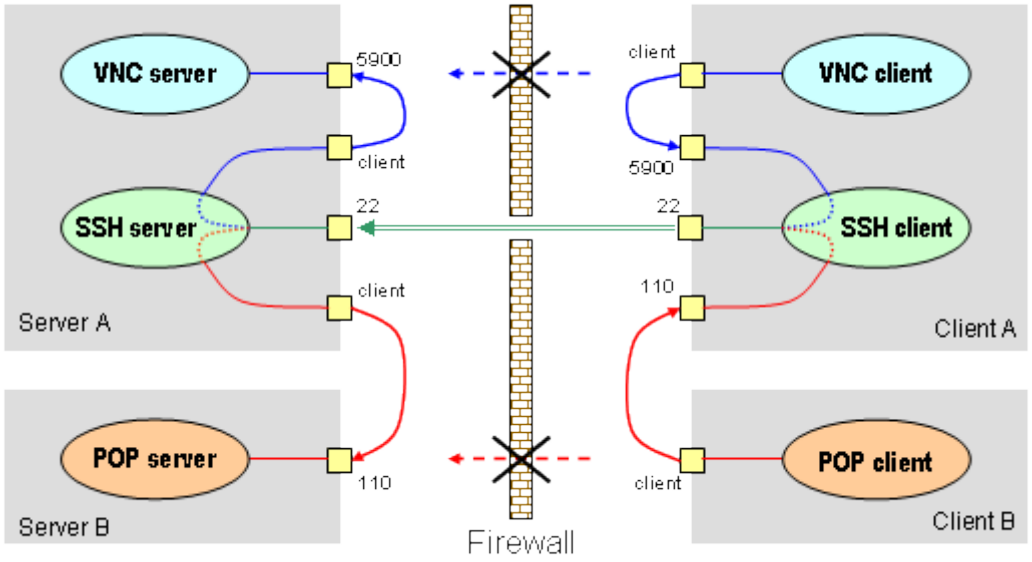
SCP: Secure Copy

Além de substituir o telnet em terminais de modo texto, o SSH também substitui o serviço FTP autenticado, provendo uma forma segura de transferência de arquivos entre computadores. Isso é feito através dos serviços SCP e SFTP:

- SCP é mais usado em servidores SSH versão 1 (SSH1). Permite a transferência cifrada de arquivos em ambas as direções (*upload*, *download*).
- SFTP é mais usado em servidores SSH2. Ao contrário do SCP, implementa um sistema de arquivo remoto, permitindo mais operações sobre os arquivos remotos, como mover, copiar, renomear, etc.

Tunelamento

Outra característica interessante do protocolo SSH é a capacidade de "tunelar" outros protocolos. Ao criar um túnel, o servidor e o cliente SSH permitem transportar pacotes de outros protocolos através do canal SSH. Essa característica é útil para proteger protocolos não-criptografados (como POP3 e VNC), fazendo-os passar por dentro do túnel cifrado. Outro uso freqüente é "burlar" *firewalls*, passando através do túnel os protocolos que seriam filtrados pelo *firewall*. Veja exemplos de tunelamento das portas 5900 (VNC) e 110 (POP3):



Na figura acima estão presentes dois tipos de túneis:

- **túnel privado**: quando somente processos da máquina cliente podem fazer uso do túnel (túnel VNC em azul na figura)
- **túnel público**: quando processos clientes em outros hosts podem usar o túnel (túnel POP em vermelho na figura)

Autenticação por chaves

Normalmente, as sessões SSH/SCP são autenticadas por senha: ao se conectar, o usuário informa seu nome de login e sua senha; caso os dados sejam válidos, a sessão é iniciada. Todavia, outras formas de autenticação também podem ser usadas, como chaves criptográficas, NIS, Kerberos, etc. Esses métodos evitam que o usuário tenha de se autenticar a cada conexão, e também são úteis no caso de conexões ativadas por programas ou scripts, sem intervenção do usuário.

O uso de autenticação por chaves criptográficas é particulamente interessante, por ser um método simples de configurar e relativamente robusto. Esse método é baseado em um par de chaves criptográficas assimétricas, ou seja, uma chave pública e sua correspondente privada. Para gerar esse par de chaves, pode ser usado o comando `ssh-keygen` no computador cliente (as solicitações do programa podem ser respondidas com `<enter>`):

```
(client)$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/myuser/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/myuser/.ssh/id_rsa.
Your public key has been saved in /home/myuser/.ssh/id_rsa.pub.
The key fingerprint is:
11:1d:9e:b0:68:1d:2d:3a:0a:41:be:51:db:67:1f:05 myuser@client
```

Esse comando gerará dois arquivos no diretório `$HOME/.ssh`:

```
(client)$ ls -l .ssh/
total 2
-rw------- 1 myuser mygroup 1675 2006-07-08 10:26 id_rsa
```

```
-rw-r--r-- 1 myuser mygroup  411 2006-07-08 10:26 id_rsa.pub
```

- O arquivo `id_rsa` contém uma chave privada, que só deve estar acessível ao seu dono;
- O arquivo `id_rsa.pub` contém a chave pública correspondente, que será distribuída.

Uma vez gerado o par de chaves criptográficas, a chave pública (o arquivo `$HOME/.ssh/id_rsa.pub`) deve ser transferida para todas as contas (os servidores SSH remotos) onde se deseja efetuar a autenticação por chaves. Em cada conta, a chave pública deve ser copiada no final do arquivo `$HOME/.ssh/authorized_keys`. Se esse arquivo não existir, ele deve ser criado, com permissões 0644 (`rw-r--r--`).

```
(server)$ cat id_rsa.pub >> $HOME/.ssh/authorized_keys
(server)$ chmod 0644 $HOME/.ssh/authorized_keys
```

A cópia da chave pública para o servidor remoto também pode ser feita através do comando `ssh-copy-id`, que automatiza o procedimento acima (basta substituir `myuser` pela sua identificação no servidor):

```
(client)$ ssh-copy-id myuser@server
```

A mesma chave pública (arquivo `id_rsa.pub`) pode ser copiada para todos os servidores/contas nos quais se deseja efetuar autenticação por chaves. Se tudo tiver sido feito corretamente, já deve ser possível efetuar conexões SSH com autenticação por chaves, sem a necessidade de senhas. Isso pode ser facilmente testado a partir da máquina cliente:

```
(client)$ ssh myuser@server
Last login: Fri Jul 18 10:10:02 2008 from 189.34.123.54

(server)$
```

A autenticação por chaves torna muito simples e prática a execução de comandos remotos em scripts, por dispensar a digitação das senhas. Por exemplo, para verificar os usuários conectados ao *host* `dainf.ct.utfpr.edu.br` (comando `w`), basta digitar:

```
(client)$ ssh dainf.ct.utfpr.edu.br w
 17:10:47 up 23 days, 22:42,  1 user,  load average: 0,00, 0,00, 0,00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
root     tty1                      24Apr12 23days  0.43s  0.27s -bash
maziero  pts/0    200.134.10.56     17:13    5.00s  0.16s  0.16s -bash
(client)$
```

Caso a autenticação por chaves não funcione, deve-se verificar a configuração do **servidor SSH** (arquivo `/etc/ssh/sshd_config`), mais especificamente as opções `RSAAuthentication`, `PubkeyAuthentication` e `AuthorizedKeysFile`.