Apostila SSH

Laboratório de Macroevolução e Macroecologia

Maio/2015

Contents

Introdução

O que é SSH?

SSH é a sigla para 'Secure SHell', e é basicamente um protocolo de rede que permite computadores se conectarem via rede de forma a permitir que um computador execute comandos em unidades remotas de maneira segura (criptografada). Essa segurança é garantida por uma chave de segurança que é armazenada tanto em no computador local quanto na máquina remota. Além disso, na maioria das vezes esse acesso é somente autorizado através de senhas de acesso.

Acesso

O acesso básico a uma máquina remota via SSH é bastante simples:

```
$ ssh usuario@ip
```

No entanto, alguns argumentos podem ser usados conjuntamente com esse comando básico quando a tarefa requer especificações extras. Uma lista destes argumentos e suas funções pode ser facilmente obtida usando o comando

```
$ man ssh
```

que abre o manual deste programa. (Esta sintaxe pode ser usada para qualquer programa na linha de comando. *Ex. man cp, man emacs, etc.*)

Apesar de vários desses argumentos serem importantes, para nosso acesso aos servidores *labmeme1*, *jabba* e *leia* usaremos a sintaxe básica. Os servidores do **LabMeMe** estão configurados no endereço:

```
labmeme1: 143.107.246.251
jabba: 143.107.246.253
leia: 192.168.0.1 (ip interno, acesso via jabba)
```

Sendo assim, para acessar o servidor *jabba*, por exemplo, o usuário deve digitar:

```
$ ssh usuario@143.107.246.253
```

No primeiro acesso, uma mensagem de aviso aparecerá perguntando se você tem certeza se quer continuar.

```
# The authenticity of host 'github.com (207.97.227.239)' can't be established.
# RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
# Are you sure you want to continue connecting (yes/no)?
```

Isso acontece pois o protocolo SSH é baseado no reconhecimento entre as chaves públicas do computador local e do computador remoto. No primeiro acesso tanto a chave local quanto a chave remota serão armazenadas nos dois locais, evitando que a identidade das máquinas precise ser verificada pelo usuário em todo acesso. Por fim, basta digitar *yes* que sua senha será pedida. (*Nota: ao digitar sua senha, a maioria dos sistemas operacionais não mostrará nenhum caracter representando sua senha, ou seja, você digitará a senha e nada vai aparecer na tela. Não se preocupe, isso é mais uma medida de segurança.*)

Rotinas Comuns

Armazenando os endereços de IP das máquinas

Lembrar dos endereços numéricos de IP das máquinas é algo que não precisa ser feito. Em sistemas UNIX, é possível atribuir nomes a esses endereços ao editar o arquivo /etc/hosts utilizando seu editor de preferência (nano, vi, emacs, gedit, etc.)

```
$ emacs /etc/hosts

##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
fe80::1%lo0     localhost
```

Ao editar esse arquivo, basta inserir na primeira coluna o endereço de IP da máquina desejada e na segunda coluna o nome que você quer atribuir à ela.

```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
fe80::1%lo0     localhost
143.107.246.251 labmeme1
143.107.246.253 jabba
```

Após atribuir um *alias* (nome que damos a qualquer "codinome" de um objeto ou função em UNIX) aos servidores, podemos nos conectar a eles tanto através do endereço de IP quanto através de seu *alias*.

```
$ ssh usuario@labmeme1
$ ssh usuario@jabba
```

Neste exemplo, o mesmo nome do servidor foi atribuído ao seu , porém isso pode ser modificado de acordo com sua vontade/necessidade.

```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
```

```
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
fe80::1%lo0     localhost
143.107.246.251 servidor-mac
143.107.246.253 servidor-linux
```

Aqui o acesso seria feito da seguinte maneira:

```
$ ssh usuario@servidor-mac
$ ssh usuario@servidor-linux
```

Acessando nós escravos

Da mesma forma como você pode fazer no seu computador, o nó escravo do nosso servidor Linux (*leia*) pode ser acessado via ssh através do nó *jabba*

```
$ *dentro do nó jabba* ssh usuario@leia
```

Tunneling por ssh

Em alguns casos, como no serviço de computação em nuvem da USP, para maior segurança o acesso aos servidores não é feito de forma direta. Nesse caso, o acesso é inicialmente feito a uma máquina (virtual nesse caso, podendo ser física) que media a "conversa" entre a máquina local e os servidores de processamento. Sendo assim, o acesso aos servidores de processamento deve ser feito em duas etapas: 1. Acesso via ssh: **ssh usuario@shark.lcca.usp.br**; 2. Acesso via ssh a partir dessa máquina aos servidores *aguia* e *jaguar*: **ssh usuario@aguia** ou **ssh usuario@jaguar**.

Sendo assim, a cópia de arquivos de/para o servidor (capítulo 2) precisaria também ser feita seguindo essas duas etapas. Porém, as máquinas intermediárias costumam ter limitação de recursos (principalmente de armazenamento), além de ser um processo bastante inconveniente copiar arquivos para um determinado lugar para em seguida copiá-los novamente para uma nova localidade. Para evitar esse tipo de problemas, é possível estabelecer uma conexão de *tunneling* por ssh, que cria literalmente um túnel de acesso direto entre a máquina local e os servidores de processamento. Essa conexão é criada da seguinte forma (usando uma conexão com o servidor *aguia* como exemplo:

```
$ ssh -2 -L 8020:aguia.lcca.usp.br:22 usuario@shark.lcca.usp.br
```

onde **-L** indica que o link entre sua máquina e o servidor deve ser feito através da porta 8020 (a porta 22 é a porta padrão de comunicação via ssh). Essa conexão deve ser mantida aberta para que os arquivos possam ser copiados diretamente da máquina local para os servidores de processamento e vice-versa.

Encerrando uma conexão

Para encerrar uma conexão ssh, basta digitar **exit** ou apertar as teclas **Ctrl+d**

Copiando arquivos *via* ssh

A rotina de trabalho em servidores remotos envolve muitas etapas de cópia de arquivos tanto do computador local para o remoto quanto no caminho inverso. A cópia de arquivos via ssh se assemelha bastante à cópia de arquivos localmente, porém usa-se o comando **scp** e seus argumentos.

Os argumentos mais comumente usados são:

Argumento	Função
-r	**recursive**: necessário para copiar múltiplos arquivos e/ou pastas
-v	**verbose**: imprime mensagens de progresso
-P	**port**: indica a porta pela qual os arquivos devem ser copiados
-q	**quiet**: não imprime mensagens nem avisos
-l	**limit**: limita o uso de banda de conexão (em bytes) para a cópia dos arquivos

Copiando um arquivo

Neste exemplo, vamos copiar o arquivo **foo.txt** do computador local para a pasta */home* do servidor. (Para os exemplos a seguir, assumiremos que você já domina os comandos básicos de shell).

```
$ scp foo.txt usuario@jabba:~/
```

Copiando vários arquivos ou pasta

Agora, vamos copiar a pasta *phylogenies* localizada na pasta */home/usuario/* para a pasta *~/Dropbox*.

```
$ scp -r usuario@143.107.246.251:~/usuario/phylogenies ~/Dropbox
```

Lembre-se de tomar cuidado com os caminhos para os arquivos. Verifique se você está na pasta desejada para usar a forma simplificada de caminho, ou certifique-se de passar o caminho completo do(s) arquivo(s)/pasta(s) a ser(em) copiados(as).

Copiando arquivos usando o *tunneling*

Como mencionado anteriormente, em servidores como o da nuvem da USP podemos copiar arquivos diretamente para os servidores de processamento sem ter que copiar primeiro os arquivos para o gatekeeper e em seguida copiar de lá para os servidores. Para isso, o primeiro passo é ativar a conexão de *tunneling* como indicado na seção **2.3**.

Em seguida, utilizando o protocolo scp, podemos copiar um ou mais arquivos ou pastas diretamente para os servidores. IMPORTANTE: lembrar a porta usada no *tunneling*! Ela será necessária nesta etapa.

```
$ scp -r -P 8020 ~/Dropbox/phylogenies usuario@localhost:~/projects
```

Este comando copiou a pasta de filogenias diretamente para o servidor ao qual nos conectamos usando a porta 8020 que foi determinada na conexão de *tunneling*. O comando para copiar arquivos do servidor para o computador local é similar:

```
$ scp -r -P 8020 usuario@localhost:~/projects/phylogenies ~/backup/
```