SSH: Uma abordagem geral

Renê de Souza Pinto

Abril / 2013

Agenda

Histórico

O que é SSH?

SSH = Secure Shell: É um protocolo de arquitetura cliente/servidor para troca de dados, execução de shell e comandos remotos de maneira segura, criptografada.

- **SSH Versão 1**: Primeira versão foi desenvolvida em 1995 por Tatu Ylönen, pesquisador da Universidade de Tecnologia de Helsinki (Finlândia)
- Tinha o intuito de substituir ferramentas de acesso remoto (telnet, rlogin, rcp) por versões mais seguras, que não transmitiam senhas em texto puro pela rede
- Utilizava código CRC-32 para verificação de integridade, o que tornava o protocolo vulnerável a inserção de dados forjados

Histórico

- **SSH Versão 2**: Versão oficial padronizada pela IETF (Internet Engineering Task Force)
- Padrão descrito em cinco RFCs:
  - SSH Assigned Numbers (RFC 4250)
  - SSH Protocol Architecture (RFC 4251)
  - SSH Authentication Protocol (RFC 4252)
  - SSH Transport Layer Protocol (RFC 4253)
  - SSH Connection Protocol (RFC 4254)
- Possui maior segurança, padrão atual
- Apesar de ser suportada por ferramentas atuais, a versão 1 está ultrapassada e deve ser evitada.

Funcionamento Básico

Funcionamento Básico:

- Primeiro passo: Identificar e promover a conexão com o host remoto
  - Algoritmos de chave assimétrica (RSA, DSA) são utilizados para a troca de uma chave simétrica
- Comunicação
  - Após a troca de chaves a comunicação é criptografada por algoritmos simétricos (AES, 3DES, Blowfish)

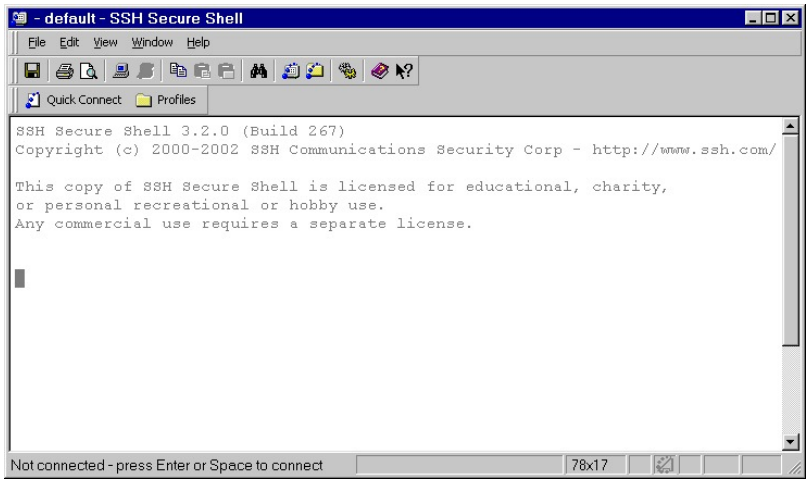
Ferramentas

OpenSSH: Padrão em diversos sistemas Unix *like* (GNU/Linux, OpenBSD, etc)

- http://www.openssh.org/
- Provê diversos utilitários, tanto cliente quanto servidor: ssh, scp, sftp, sshd (servidor), etc

E no Windows?

Clientes:



SSH Secure Shell

PUTTY

Servidores:

- COPSSH: https://www.itefix.no/i2/copssh
- freeSSHd: http://www.freesshd.com/
- MobaSSH: http://mobassh.mobatek.net/

Comandos Básicos

Ambiente de exemplo:

- Máquina Local:
  - Host: terra
  - Usuário: *foo*
- Máquina Remota:
  - Host: jupiter.net
  - Usuário: *bar*

Login na máquina remota:

```
foo@terra# ssh bar@jupiter.net
```

Ou:

```
foo@terra# ssh -l bar jupiter.net
```

Comandos Básicos

Ambiente de exemplo:

- Máquina Local:
  - Host: terra
  - Usuário: *foo*
- Máquina Remota:
  - Host: jupiter.net
  - Usuário: *bar*

Login na máquina remota:

```
foo@terra# ssh bar@jupiter.net
```

Ou:

```
foo@terra# ssh -l bar jupiter.net
```

Comandos Básicos

- Porta padrão do SSH: **22**
- Outras portas podem ser utilizadas
- Login na máquina remota executando o servidor SSH na porta 1030:

```
foo@terra# ssh -p1030 bar@jupiter.net
```

Comandos Básicos

- O cliente faz um *fingerprint* de cada servidor, que é um HASH aplicado a chave do servidor para verificar a "assinatura" do mesmo mais rapidamente do que utilizar uma chave completa
- Se o endereço do servidor é desviado para outra máquina, o cliente é proibido de logar, a não ser que o *fingerprint* seja atualizado pelo próprio cliente

Arquivos de configuração SSH

- Os arquivos de configuração do usuário ficam na pasta ∼/**.ssh**
- *known_hosts*: *Fingerprints* dos servidores
- *authorized_keys*: Chaves públicas de clientes
- *config*: Arquivo que contém configurações do usuário (hosts, etc)
- Chaves pública e privada do usuário (*id_rsa/id_rsa.pub*, *id_dsa/id_dsa.pub*)

Geração das chaves

Geração das chaves pública e privada no cliente

```
foo@terra# ssh-keygen -t dsa
```

Chaves:

- id_dsa: Privada
- id_dsa.pub: Pública

```
foo@terra# ls ~/.ssh/
id_dsa  id_dsa.pub
```

Autenticação por chaves

Para autenticação somente por chaves (sem a necessidade de fornecer senha):

- Gerar as chaves no cliente
- Copiar a chave pública do cliente para o arquivo *authorized_keys* no servidor
- Exemplo:

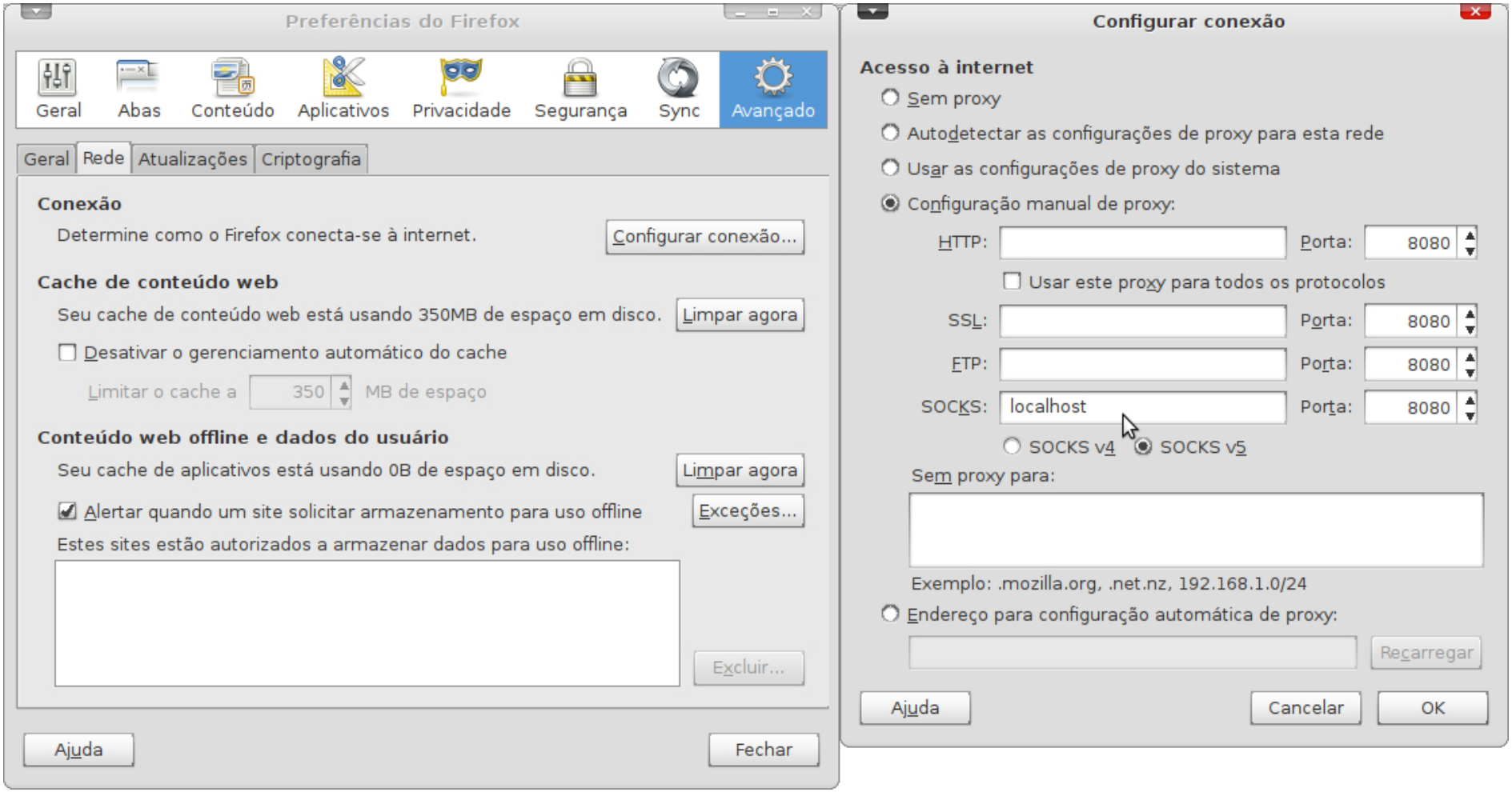
```
cat .ssh/authorized_keys
ssh-dss AAAAB3NzaC1kc3MAAACBAIAqiQO/+Oz/hWOD/alJT8V5L3H1lXOfmFYcLcTpr9LjHnlDMdET/OkxxN4pvHqaoe+Uejxszi4Brt
ZS9wLodGhOG4QayV1uKWfdv83cdp2ObnOW9LJBHwUn929B+FkqbWog35MkG2FUfYKwWixSHQWX4MOq5vVOJZZJ6XUBJx8FAAAAFQDNY8p5
RELukxoplZ7HpVFB+5HnyQAAAIBSaOqNPXV+eB+bBA1eWbpgTlUNM4I6nGS7h3z6B1JOAE/XHXI5+hiBd42LnOf3D14zOLmheqPOxRPDYh
v/aq/T3yc9wYWPflbCdZ64iYqfgkikWtz8tpMsVDwA6Ns16uHKRPccxO+y189pDhkSwCM9HPpFUcFKOOZRMWOLZUrtSgAAAIBrnCxIWOii
c4O2I+Lbbbp43DfjIreAMAj3AQ8c4zBxcf8c4bThdVfxyU7B3cIveUh+WEeEXtNQHqpAzEE3gCGsixnYah2iOIIZ1+V6J2AXPQLy/5CZJI
eSjl44wfe+U5H5V3N/FRzTdPvXWVcfUyhVdzntbE1DQRMEbYno3kJ6LA== foo@terra
```

Tunelamento SSH

- Redireciona as conexões em uma porta local para a máquina remota, atuando como um proxy
- Exemplos de uso: Transpassar proxy, fazer conexões seguras (criptografadas) em redes inseguras (públicas, abertas, etc)

Tunelamento SSH

```
foo@terra# ssh -D8080 bar@jupiter.net
```



Configuração do Proxy no host terra.

Tunelamento Reverso

Sintaxe:

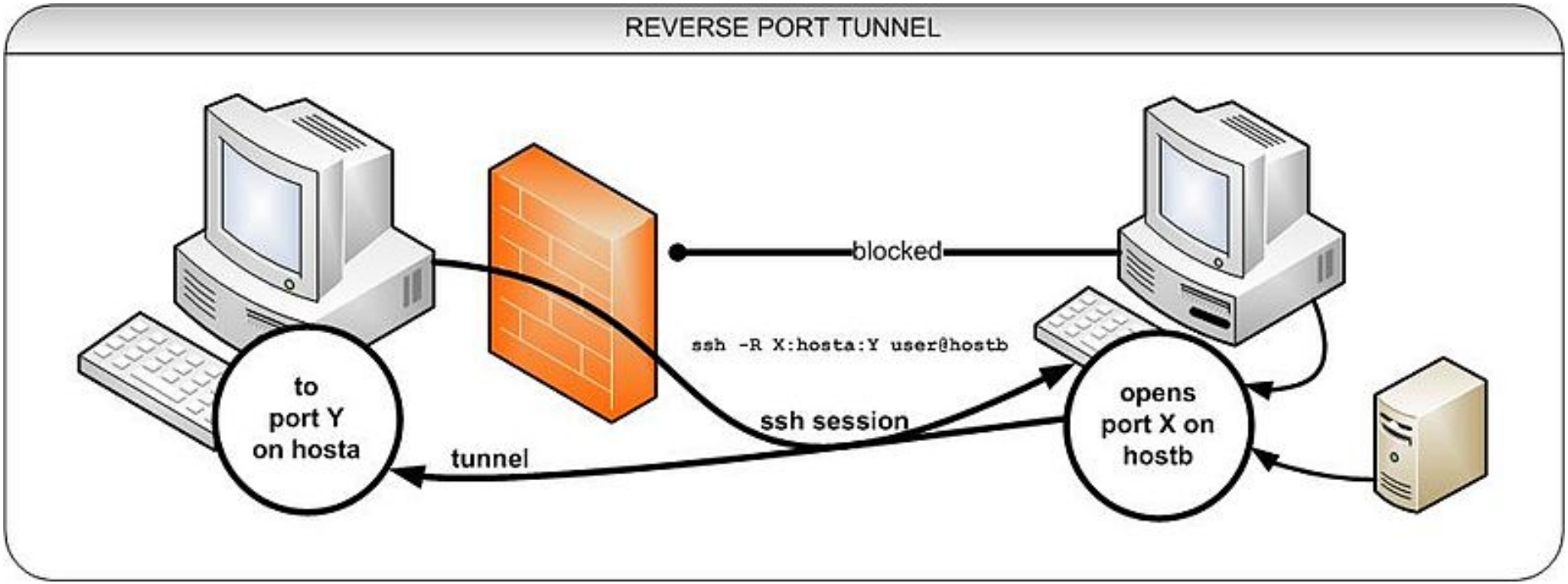
```
ssh -R port:host:hostport
```

Especifica a porta (port) do servidor que será redirecionada para a porta (hostport) do host especificado (host). Útil para acessar máquinas abaixo de Firewalls/NATs, etc

Funcionamento:

- Suponha a máquina hosta que está abaixo de um NAT, e que se deseja acessar a partir de outra máquina (hostb)
- Não é possível acessar hosta diretamente, pois a mesma está abaixo do NAT. Porém, é possível acessar hostb a partir de hosta.
- Criando um túnel reverso a partir de uma conexão SSH entre hosta↔hostb poderemos posteriormente acessar hosta a partir de hostb

Tunelamento Reverso



Tunelamento Reverso

- hosta: SSH na porta 22, host: terra, usuário: foo
- hostb: SSH na porta 22, host: jupiter.net, usuário: bar
- Criaremos o túnel na porta 20022 do hostb

- Abrindo o Túnel Reverso através da conexão de hosta com hostb:

```
ssh -R 20022:localhost:22 bar@jupiter.net
```

- Acessando hosta a partir de hostb via túnel:

```
ssh -p20022 foo@localhost
```

Outras ferramentas

scp: Copia arquivos entre cliente/servidor

```
foo@terra# scp <arquivo_local> user@host:<path>
```

```
foo@terra# scp teste.txt bar@jupiter.net:/home/bar
foo@terra# scp bar@jupiter.net:/home/bar/teste.txt
teste2.txt
```

Se o servidor estiver sendo executado em outra porta (Por ex: 1022):

```
foo@terra# scp -P1022 bar@jupiter.net:/home/bar/teste.txt
teste2.txt
```

Cópias de diretórios, recursiva, etc, também são permitidas

Outras ferramentas

sftp: Comandos interativos iguais ao FTP, porém o protocolo é seguro

```
foo@terra# sftp user@host
```

Obrigado. Dúvidas?

rene@renesp.com.br rene@dc.ufscar.br rene@icmc.usp.br