SSH Keys

William Stearns
http://www.stearns.org/
william.l.stearns@dartmouth.edu
6-0647

What you'll need:

- Ssh command line client: `ssh`
  - Graphical clients fine too, but you may have to look through a menu or two.
- Ssh-agent running: `set | grep '^SSH'`
- Ssh-keyinstall: http://www.stearns.org/ ssh-keyinstall/
- At least one account on a remote server
- USB flash drive

Why ssh?

- Encrypted connections
- Remote terminal
- File transfer
- Tunneling traffic
  - TCP connections
  - X Windows applications
  - Full VPN

Why keys?

- Password problems
  - Crackable
  - Remembering them vs reuse
  - No easy and safe way to automate connections
- Key can be loaded and used for multiple connections
- Private key can be moved onto a flash drive

Key basics

- Private key stays on machine where you type
- Public key copied out to servers to which you log in
- Connection can only be made *from* machine with private *to* machine with public
- Server can accept keys from some users, passwords from others
  - May later lock down to keys only

Goals – ssh for sysadmins

- Set up an ssh keypair on a token
- Install public key on remote machine
- Learn how to load and unload
- Use keys for automated tasks
- A few advanced tricks at the end :-)

Basic ssh connections

- `ssh {user@}hostname`
- Problems? Be verbose:
  - `ssh -v {user@}hostname`
- Emergency disconnect:
  - `<Enter> ~ .`

Mounting flash drives

- As root: `mkdir -p /media/flash/`
- Insert token
  - Check if automounted with `mount`
- `tail --lines=30 /var/log/messages`
- Find device, such as “/dev/sdc1”
- Find your uid: `id -u`
- `mount -t vfat -o uid={uid} /dev/sdc1 /media/flash/`

Setup

- (Do this once)
- `cd`
- `mkdir -p .ssh`
- `chmod 700 .ssh`
- `mkdir -p /media/flash/.ssh/`
- `chmod 700 /media/flash/.ssh/`

Get key install script

- (Do once)
- `rpm -Uvh http://www.stearns.org/ssh-keyinstall/ssh-keyinstall-1.0.0-0.noarch.rpm`
  - or
- `cd /usr/bin`
- `wget http://www.stearns.org/ssh-keyinstall/ssh-keyinstall`
- ssh-keyinstall needs netcat (nc)

Create key

- (Do this once)
- `cd ~/.ssh`
- `ls -al`
  - If no id_dsa_sysadmin, continue
- `ssh-keygen -t dsa -b 1024 -C {YourHostname}_sysadmin -f ~/.ssh/id_dsa_sysadmin`
- Enter long passphrase
- Private: id_dsa_sysadmin
- Public: id_dsa_sysadmin.pub

Move to usb token

- (Do once)
- `mv id_dsa_sysadmin /media/flash/.ssh/`
- `ln -sf /media/flash/.ssh/id_dsa_sysadmin id_dsa_sysadmin`
- `cp -p id_dsa_sysadmin.pub /media/flash/.ssh/`
- Backup token

Load Keys

- (Do this each morning)
- Insert USB key and mount if necessary
- `ssh-add ~/.ssh/id_dsa_sysadmin`
- `umount /media/flash/`
- `set | grep '^SSH' >~/agent`

Install public key on server

- (Once for each server)
- `ssh-keyinstall -s {ServerName}.dartmouth.edu -u {AccountOnServer}`
- Enter password for that account multiple times
- Details in O'Reilly “SSH, The Secure Shell”, Chapter 6

Connect to account

- `ssh {user@}servername`
- Can jump from server to server
  - `ssh {user@}gateway`
  - From inside that terminal:
  - `ssh {user@}internal_box`
  - Agent on your laptop provides the key to connect to both remote servers
  - Gateway acts as a middleman for login to internal_box

Port Forwarding

- `ssh -L 8306:sql1:3306 sql1`
- ssh client listens on 8306 on your machine; check with `netstat -pant`
- If connection comes into that port, data pushed through ssh tunnel to port 3306 on sql11
- Only ssh traffic seen on wire

Copy file

- Local file to remote server
- `scp -p /path/to/source.txt {user@}ServerName:/dest/path/`
- Remote file to local
- `scp -p {user@}ServerName:/source/path/file.txt /local/dest/path/`

Copy a file tree

- Local tree to remote system
- `rsync -av -e ssh /local/path/ {user@}ServerName:/dest/path/`
- Remote tree to local
- `rsync -av -e ssh {user@}ServerName:/dest/path/ /local/path/`
- End all paths with “/”
- Use `-avz` to compress as well

Cron shell scripts

- `if [ -f $HOME/agent ]; then`
  - `. $HOME/agent`
  - `export SSH_AUTH_SOCK SSH_AGENT_PID SSH_ASKPASS`
- `else`
  - `logger  Missing $HOME/agent`
- `fi`
- #Rest of shell script here

Run commands remotely

- `ssh wstearns@ford 'df' >ford-df.txt`
- `cat shell_commands | ssh user@server`
- `cat local_file | ssh server 'egrep -ai confidential information ' | less`
- `ssh root@sniffbox ' tcpdump -i eth1 -w - not tcp port 22 ' | passer.py -r /proc/self/fd/0`

Run graphical commands

- `ssh -X Server`
  - `xclock &`
  - `firefox &`
  - `/home/tripwire/manager/TW_Manager &`

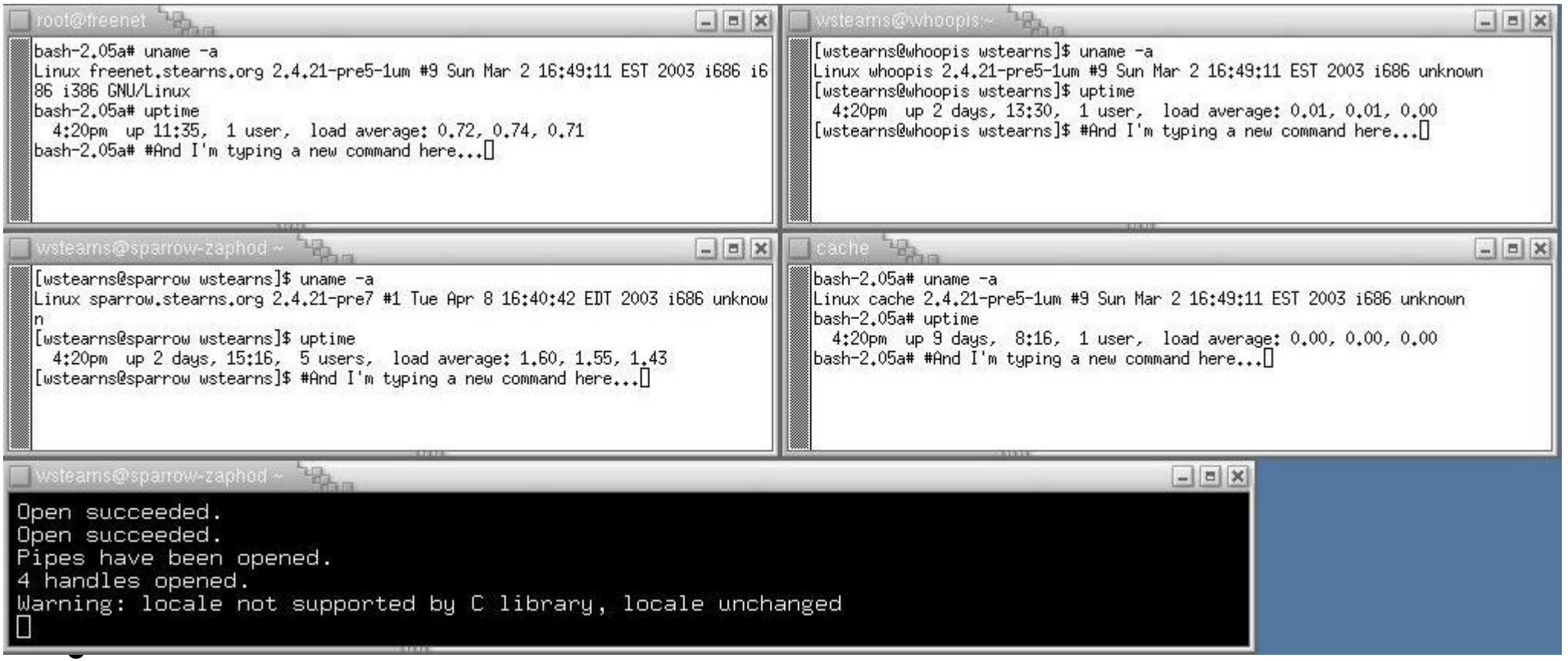
Disconnectable session

- `ssh -t {user@}server 'screen -S user@server -R'`
  - Executes the “screen” program
  - screen lets you disconnect (`<Ctrl>-a,d`)
  - Reconnect later, even from a different computer, with above command
  - Excellent for long-running jobs
  - Screen quickstart at the end

Run commands on multiple servers

- `rpm -Uvh http://www.stearns.org/fanout/fanout-0.6.1-0.noarch.rpm`
- `fanout "localhost Web1 AnotherBox" "uname -a ; rpm -qa | egrep -i '(fedora|redhat-release)' ; uptime ; df -P / ; netstat -a | grep '*:*'" | less`
- `fanterm box1 ftp mail`

Fanterm run



- http://www.stearns.org/fanout/ fanterm-v0.6-50.jpg

Forget key

- At lunch or end of day
- `ssh-add -D`
- For shorter breaks, lock screen (System menu) or `vlock -a` from a text console

Thanks!

- Questions?
-

- William Stearns
- http://www.stearns.org/
- william.l.stearns@dartmouth.edu
- 6-0647

Appendix - Screen hints

- `screen -S sessionname -R`
  - To create or connect to existing
- `<Ctrl>-a,c`
  - Add a shell inside screen
- `<Ctrl>-a,n`
  - Go to next shell
- `<Ctrl>-a,d`
  - Disconnect but leave running
- `screen -S sessionname -R`
  - Reconnect later