Working remotely in shell environments

Hans-Martin von Gaudecker

Why care?

- Need to interact with remote computers
- Only one standard way that (almost) alway works
- A **S**ecure **SH**ell connection
- First and foremost, SSH is a protocol

How

- In form of the OpenSSH implementation, SSH is built into essentially all shells on Linux and MacOS
- Apparently some form also available on recent Powershell
- Else, use PuTTY on Windows - https://www.putty.org

Usage

- ssh [user]@[hostname]
- [user] is your standard login at machine [hostname]
- [hostname] could be server name or IP address
- Hit enter and type in your password
- Get standard shell there, usually Bash
- Do your work
- Close connection by Ctrl-D or at the will of physics / your internet provider

Tip 1: Set up alias for frequently used connections

- Tip 0: keep aliases consistent across machines

```
$ tail ~/.bashrc -n1
source $HOME/Dropbox/bashrc_common
```

- Tip 1:

```
$ tail ~/Dropbox/bashrc_common -n1
alias computeserver="ssh xyz@some.fast.machine.de"
```

Tip 2: Use `screen` for persistent shell sessions

- My most frequent use case for remote machines are long-running computations
- Good luck keeping the connection open ...
- Similar: Might want to check progress from a different computer than the one I used to connect to the remote machine initially
- `screen` to the rescue https://www.gnu.org/software/screen/screen.html
- Usually pre-installed, else (have administrator) use package manager

Screen introduction

- Start a new screen session using:

```
$ screen
```

- Some keybindings change.
- Screen-internal stuff always starts with `ctrl-a`
- Completely exit screen session with `ctrl-d`
- Multiple sessions in parallel possible

Screen introduction

- The keybindings I remember from the top of my head
- `ctrl-a d` (note the difference to `ctrl-d` !!!)

  Detach from the screen, so you can re-attach to it later using

  ```
  $ screen -r
  ```

  Might need to add identifier, see below.
- `ctrl-a [` or `ctrl-a ESC`

  Enter "copy" mode = ability to scroll.

  Also works while another command is running.

Screen in "copy" mode

- `ctrl-u` / `ctrl-d` to move up (half) pages
- Arrow up / down keys to move single lines.
- Pretty much any other key to get back command prompt.
- Buffer may get exhausted - so do not rely exclusively on screen output for checking jobs but perform some persistent logging (screen seems to have options for that, too)

Interacting with screens

- Say connection was closed and you want to get back:

  ```
  $ screen -r
  There is a screen on:
    10874.pts-1.[hostname]  (08.07.2019 10:32:53)   (Attached)
  There is no screen to be resumed.
  ```
- Need to remote detach and then re-attach:

  ```
  $ screen -D 10874
  [10874.pts-1.hmg-desktop power detached.]
  $ screen -r 10874
  ```
- Explicitly specifying screen pid is necessary when you have multiple detached screens
- Use `screen -list` to show all running screen sessions, whether detached, attached, or whatever

Tip 3: Set up public / private key pair

- Saves you typing in your password every time
- Only if you trust your local machine
- `ssh-keygen` https://www.ssh.com/ssh/keygen/ on the commandline, PuTTYgen on Windows https://www.ssh.com/ssh/putty/windows/puttygen
- Generate a key with modern algorithm and strong encryption

```
$ ssh-keygen -t ecdsa -b 521
Generating public/private ecdsa key pair.
Enter file in which to save the key (https://www.ssh.com/ssh/copy-id):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/xyz/.ssh/id_ecdsa.
Your public key has been saved in /home/xyz/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:wx4M98k72DEBgDijfdbnVa7DVYYRLrZHg4H9PY0Zntk xyz@local-machine
The key's randomart image is:
+---[ECDSA 521]---+
|   . ....o. o+   |
|  + .   ...+o +  |
| o o .. . =++= O |
|. . o .=.+.*+.O E|
|   o   oSoBo.  . |
|       ..=+=     |
|        o +.     |
|           .     |
|                 |
+----[SHA256]-----+
```

Tip 3: Set up public / private key pair

- Copy the key over to the remote machine (https://www.ssh.com/ssh/copy-id):

```
$ ssh-copy-id -i /home/xyz/.ssh/id_ecdsa [user]@[hostname]
```

- This will be the last time you are prompted for the password of [hostname]
- You might get prompted for the passphrase - if so, check `ssh-agent` and friends. https://www.ssh.com/ssh/agent

Tip 4: Transferring files, command line only

- The tool `sftp` gives you a shell that is mainly useful for transferring files
- Remote → local:

```
$ sftp [user]@[hostname]:/home/[user]
$ get relpath/on/remote.txt relpath/on/local
$ ctrl+d
```

- Other way round:

```
$ sftp [user]@[hostname]:/home/[user]
$ get relpath/on/local.txt relpath/on/remote
$ ctrl+d
```

- Add `-r` tp get for directories. Pretty much the same as standard `cp`

Tip 4a: Transferring files, GUIs

- All platforms: FileZilla https://filezilla-project.org
- Windows: WinSCP https://winscp.net/eng/
- MacOS: FUSE for MacOS https://osxfuse.github.io
- Ubuntu:
  - In Nautilus sidebar, click on "+ Other Locations"
  - Type in `sftp://[user]@[hostname]/home/[user]` (note the `://` after the sftp and the missing colon relative to command line)

Trick: Jupyter notebooks via SSH tunnels

- Google the above if something goes wrong - that will also tell you about PuTTY on Windows
- I would not be able to remember this command to create an **ssh tunnel**

```
$ tail ~/.bashrc -n2

alias computeserver_notebook_7799="ssh -N -f -L localhost:7799:localhost:7799   xyz@som
alias computeserver="ssh xyz@some.fast.machine.de"
```

- If you are more interested in the details of what this does than me, check https://www.ssh.com/ssh/tunneling/example
- I use different ports for each remote machine I regularly use and all different from 8888

Trick: Jupyter notebooks via SSH tunnels

- Local machine:

```
$ computeserver_notebook_7799
$ computeserver
```

- [Thanks to key pair, logged in remote machine now]

```
$ jupyter-lab --port=7799 --no-browser
[...]
Or copy and paste one of these URLs:
    http://localhost:7799/?token=836cef197b87684a466d8e5a69ceac21ebbf9205069c1850
```

- Do the latter and watch your computations
- Repeat for estimagic dashboard.
- If connection turns stale, use `killall ssh` or try to make sense of results in https://lmgtfy.com/?q=kill+ssh+tunnel
- If you need to get jupyterlab url again, hit `ctrl-c` in the screen window on remote machine once

License for the course material

- [Links to the full legal text and the source text for this page.] You are free:
  - to Share to copy, distribute and transmit the work
  - to Remix to adapt the work

Under the following conditions:

- Attribution You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

With the understanding that:

- Waiver Any of the above condition scan be waived if you get permission from the copyright holder.
- Public Domain Where the work or any of its elements in the public domain under applicable law, that status is in no way affected by the license.
- Other Rights In no way are any of the following rights affected by the license:
  - Your fair dealing or fair use rights, or other applicable copyright exceptions and limitations;
  - The author's moral rights;
  - Rights other persons may have either in the work itself or in how the work is used, such as publicity or privacy rights.

Notice For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page.