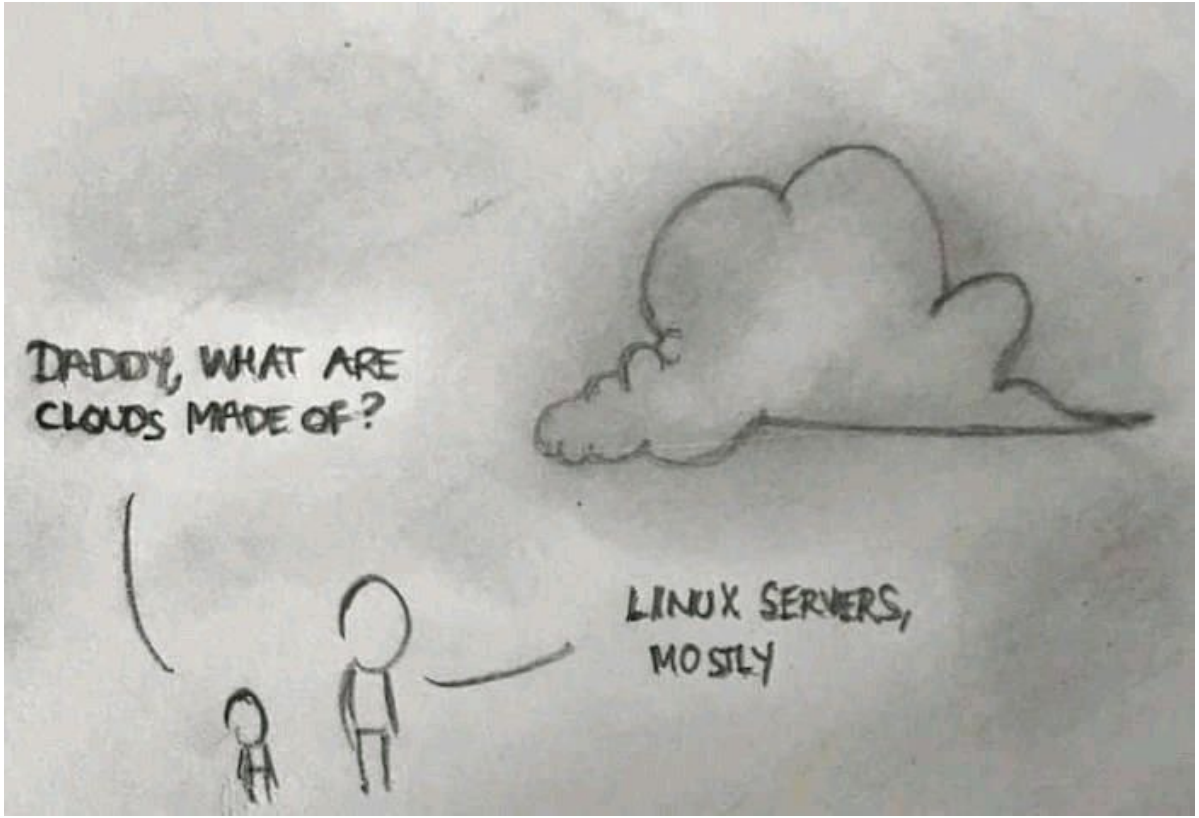
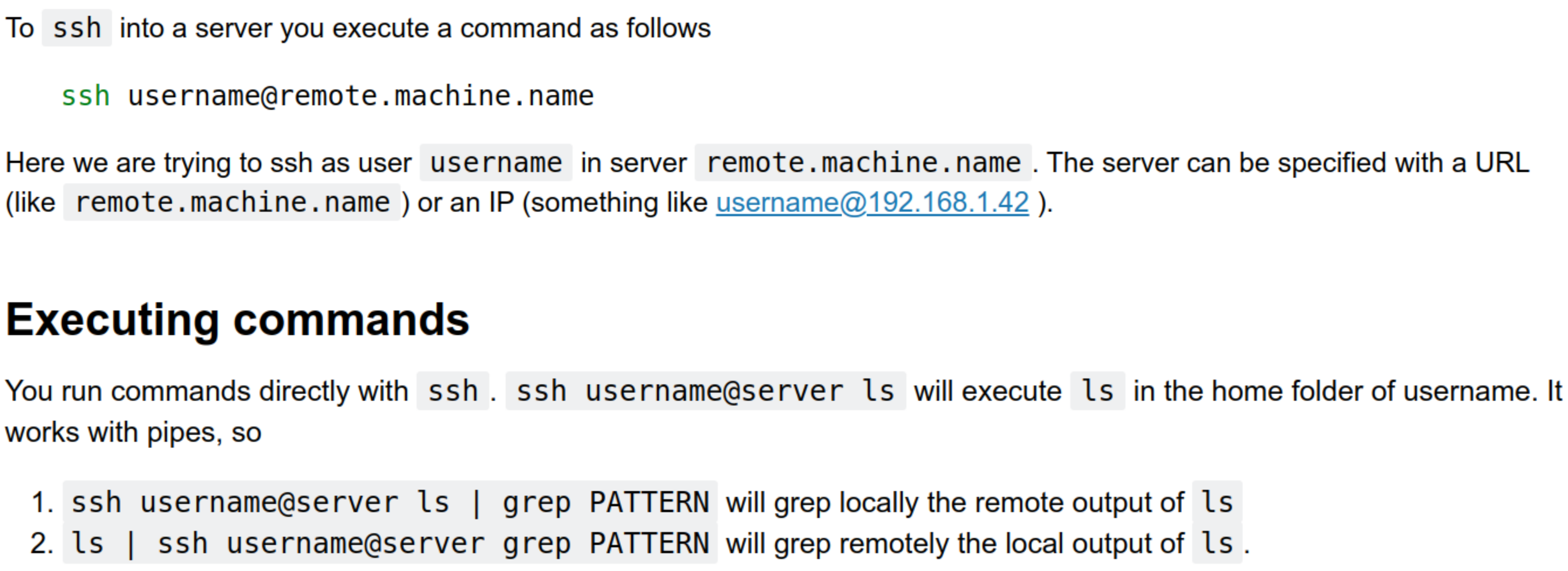
LECTURE:
UNIX ON A REMOTE MACHINE

FIRST STEPS ON A REMOTE MACHINE



CONNECT VIA SSH & EXECUTING COMMANDS



To `ssh` into a server you execute a command as follows

```
ssh username@remote.machine.name
```

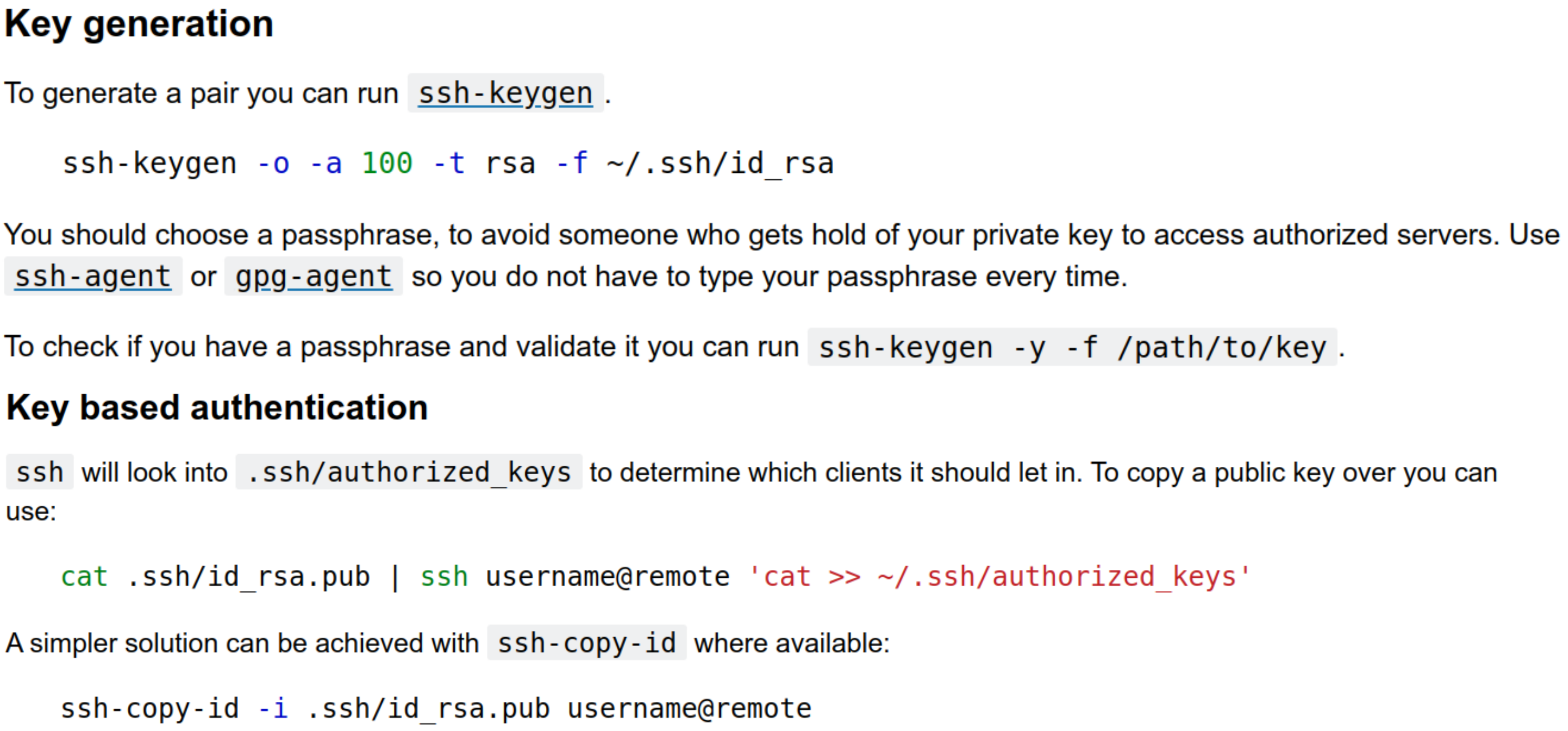
Here we are trying to ssh as user `username` in server `remote.machine.name`. The server can be specified with a URL (like `remote.machine.name`) or an IP (something like username@192.168.1.42).

Executing commands

You run commands directly with `ssh`. `ssh username@server ls` will execute `ls` in the home folder of username. It works with pipes, so

1. `ssh username@server ls | grep PATTERN` will grep locally the remote output of `ls`
2. `ls | ssh username@server grep PATTERN` will grep remotely the local output of `ls`.

SSH KEYS



Key generation

To generate a pair you can run `ssh-keygen`.

```
ssh-keygen -o -a 100 -t rsa -f ~/.ssh/id_rsa
```

You should choose a passphrase, to avoid someone who gets hold of your private key to access authorized servers. Use `ssh-agent` or `gpg-agent` so you do not have to type your passphrase every time.

To check if you have a passphrase and validate it you can run `ssh-keygen -y -f /path/to/key`.

Key based authentication

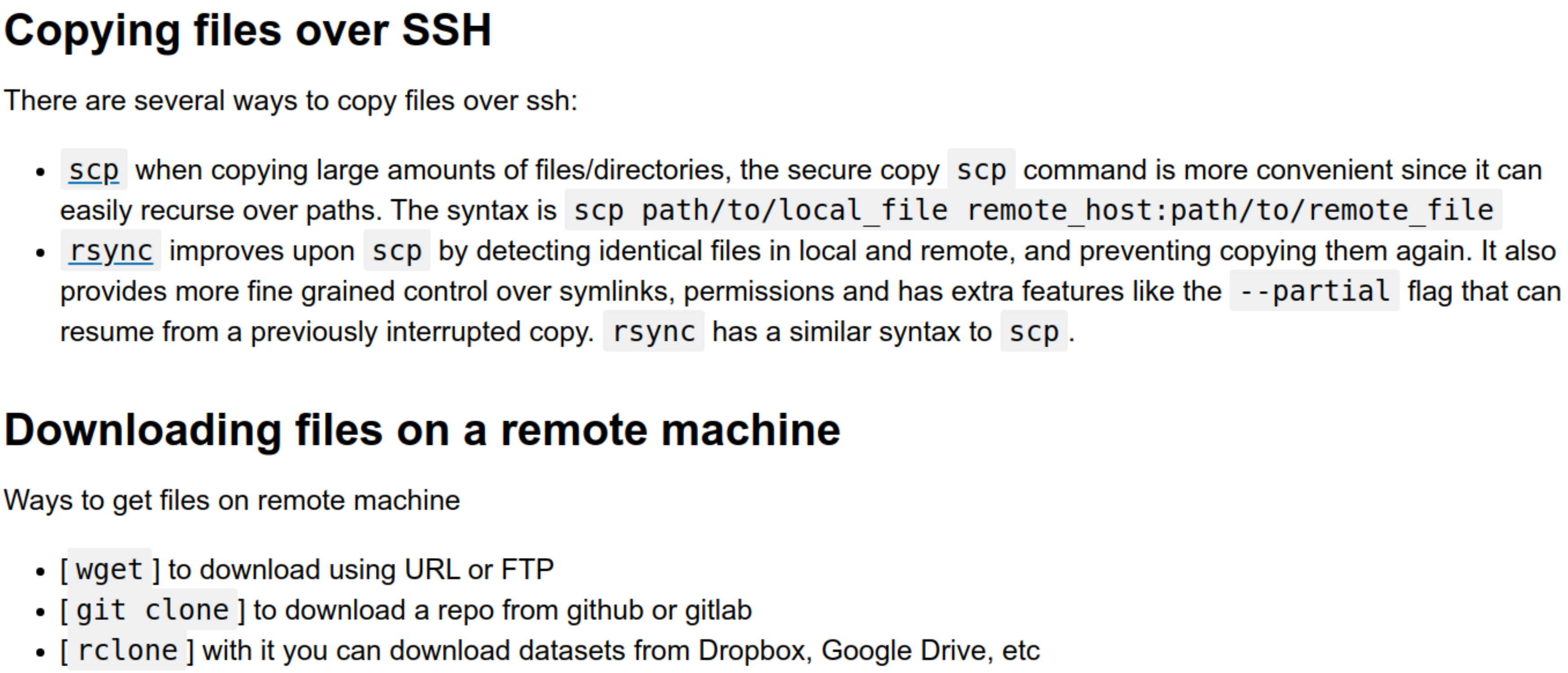
`ssh` will look into `.ssh/authorized_keys` to determine which clients it should let in. To copy a public key over you can use:

```
cat .ssh/id_rsa.pub | ssh username@remote 'cat >> ~/.ssh/authorized_keys'
```

A simpler solution can be achieved with `ssh-copy-id` where available:

```
ssh-copy-id -i .ssh/id_rsa.pub username@remote
```

COPYING AND DOWNLOADING FILES ON A REMOTE



Copying files over SSH

There are several ways to copy files over ssh:

- `scp` when copying large amounts of files/directories, the secure copy `scp` command is more convenient since it can easily recurse over paths. The syntax is `scp path/to/local_file remote_host:path/to/remote_file`
- `rsync` improves upon `scp` by detecting identical files in local and remote, and preventing copying them again. It also provides more fine grained control over symlinks, permissions and has extra features like the `--partial` flag that can resume from a previously interrupted copy. `rsync` has a similar syntax to `scp`.

Downloading files on a remote machine

Ways to get files on remote machine

- [`wget`] to download using URL or FTP
- [`git clone`] to download a repo from github or gitlab
- [`rclone`] with it you can download datasets from Dropbox, Google Drive, etc

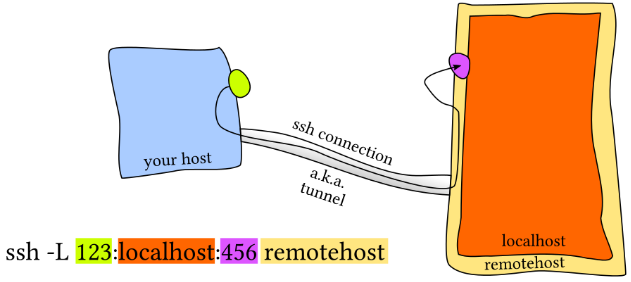
PORT FORWARDING

For example, if we execute

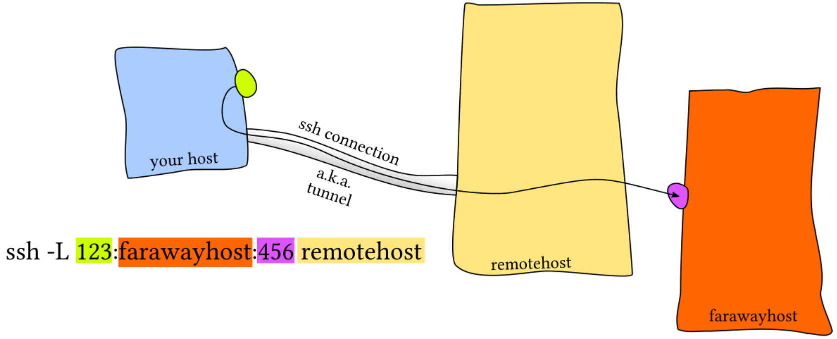
`jupyter notebook` on port `8888`.

We would do `ssh -L 9999:localhost:8888 username@remote_server`

and then navigate to `locahost:9999` in our local machine.



```
ssh -L 123:localhost:456 remotehost
```

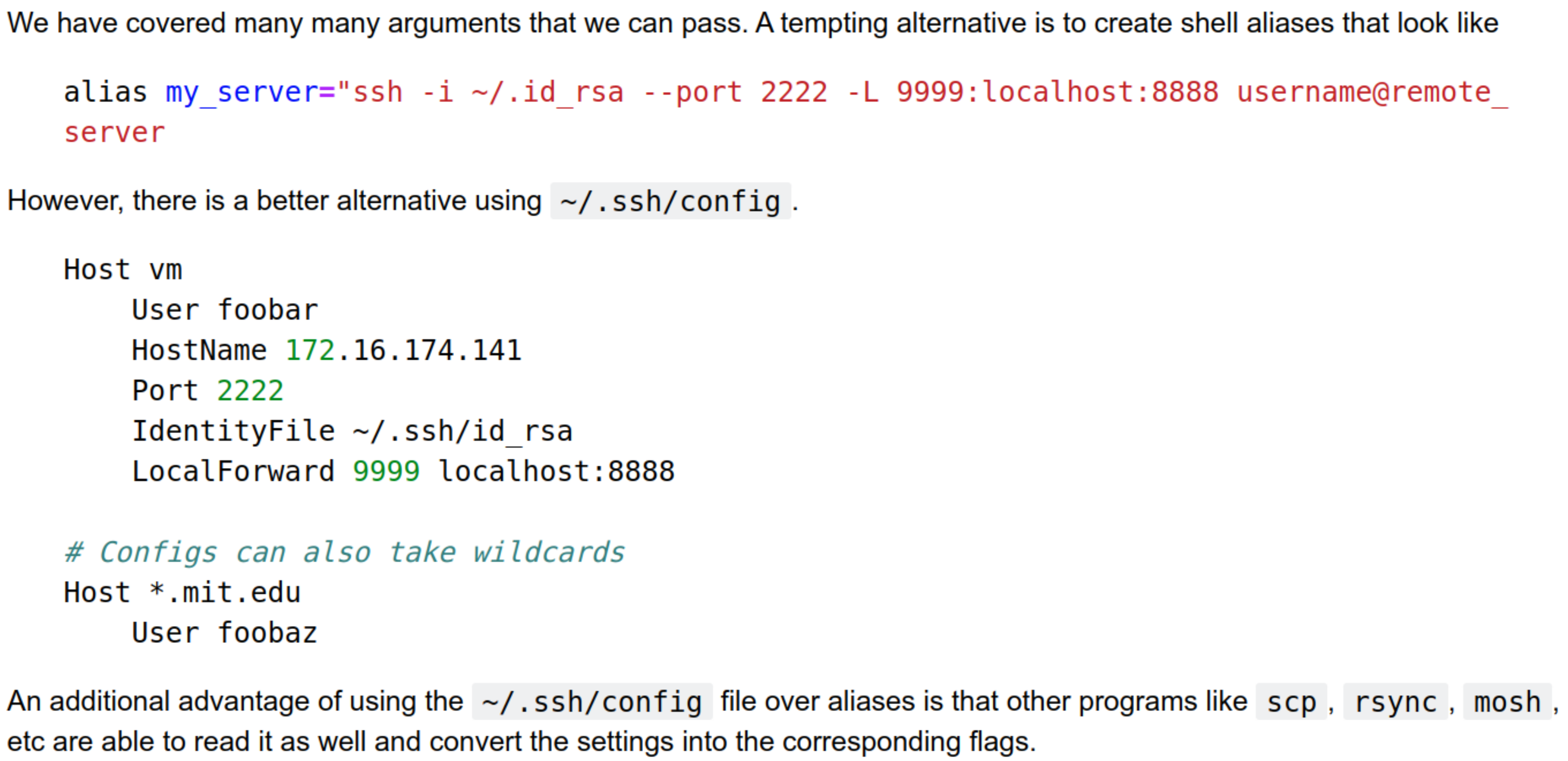


```
ssh -L 123:farawayhost:456 remotehost
```

Picture from [this StackOverflow post]

ENVIRONMENT AND CONFIGURATIONS

SSH CONFIGURATION



We have covered many many arguments that we can pass. A tempting alternative is to create shell aliases that look like

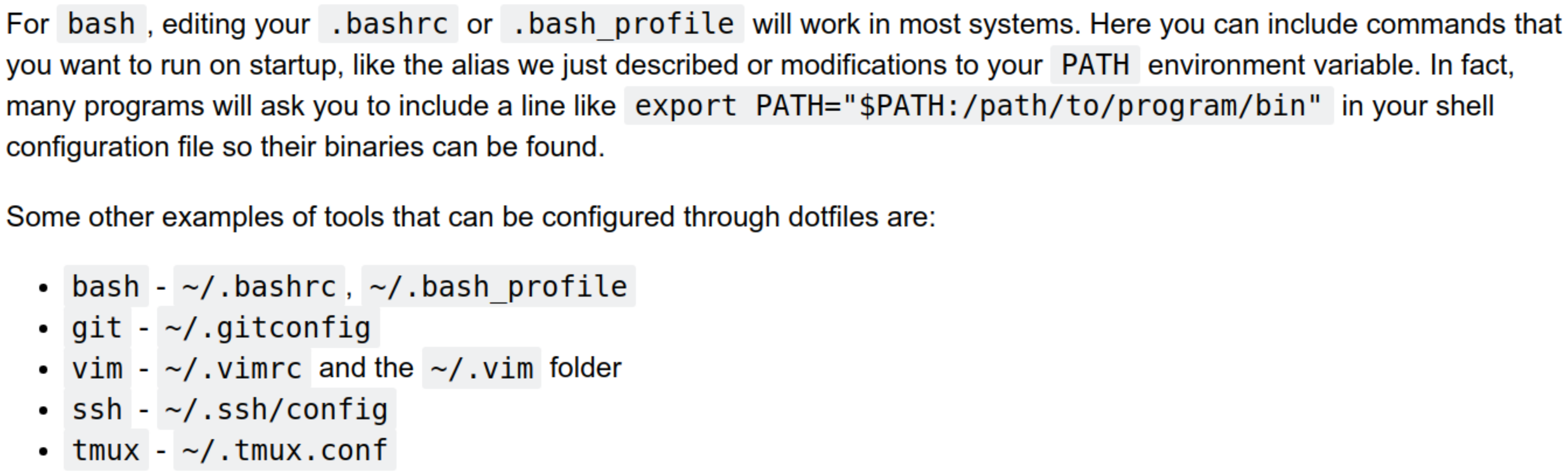
```
alias my_server="ssh -i ~/.id_rsa --port 2222 -L 9999:localhost:8888 username@remote_server
```

However, there is a better alternative using `~/.ssh/config`.

```
Host vm
    User foobar
    HostName 172.16.174.141
    Port 2222
    IdentityFile ~/.ssh/id_rsa
    LocalForward 9999 localhost:8888

# Configs can also take wildcards
Host *.mit.edu
    User foobaz
```

An additional advantage of using the `~/.ssh/config` file over aliases is that other programs like `scp`, `rsync`, `mosh`, etc are able to read it as well and convert the settings into the corresponding flags.

DOTFILES



For `bash`, editing your `.bashrc` or `.bash_profile` will work in most systems. Here you can include commands that you want to run on startup, like the alias we just described or modifications to your `PATH` environment variable. In fact, many programs will ask you to include a line like `export PATH="$PATH:/path/to/program/bin"` in your shell configuration file so their binaries can be found.

Some other examples of tools that can be configured through dotfiles are:

- `bash` - `~/.bashrc`, `~/.bash_profile`
- `git` - `~/.gitconfig`
- `vim` - `~/.vimrc` and the `~/.vim` folder
- `ssh` - `~/.ssh/config`
- `tmux` - `~/.tmux.conf`

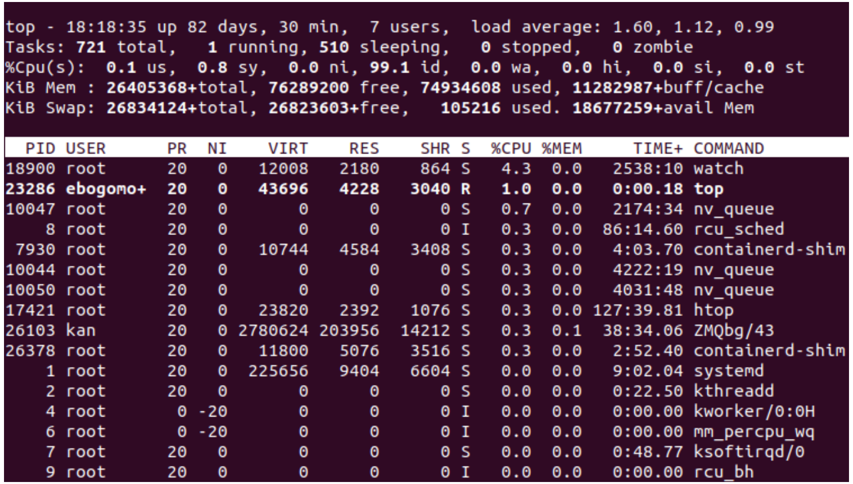
Way to learn about customizations is to look through other people's dotfiles: you can find tons of dotfiles repositories on Github --- see the most popular one here (we advise you not to blindly copy configurations though).

Here is another good resource on the topic.

INSPECTING RESOURCES

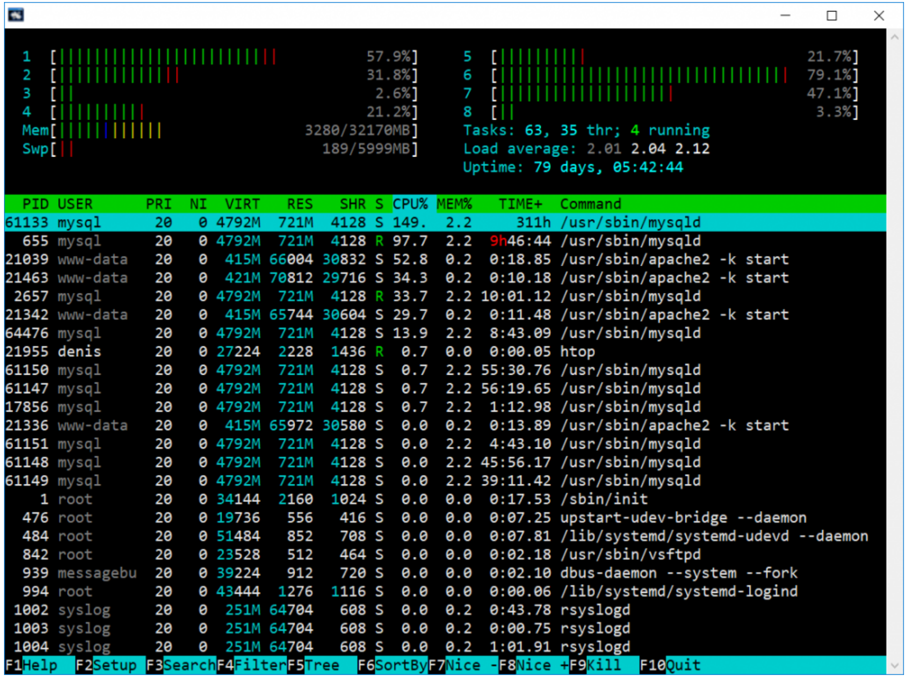
CPU RAM AND DISK USAGE

- top



```
top - 18:18:35 up 82 days, 30 min,  7 users,  load average: 1.60, 1.12, 0.99
Tasks: 721 total,   1 running, 510 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.8 sy,  0.0 ni, 99.1 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 26405368+total, 76289200 free, 74934608 used, 11282987+buff/cache
KiB Swap: 26834124+total, 26823603+free,   105216 used. 18677259+avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
18900 root      20   0   12008   2180    864 S   4.3  0.0   2538:10 watch
23286 ebogomo+  20   0   43696   4228   3040 R   1.0  0.0   0:00.18 top
10047 root      20   0       0      0      0 S   0.7  0.0   2174:34 nv_queue
    8 root      20   0       0      0      0 I   0.3  0.0  86:14.60 rcu_sched
 7930 root      20   0   10744   4584   3408 S   0.3  0.0   4:03.70 containerd-shim
10044 root      20   0       0      0      0 S   0.3  0.0   4222:19 nv_queue
10050 root      20   0       0      0      0 S   0.3  0.0   4031:48 nv_queue
17421 root      20   0   23820   2392   1076 S   0.3  0.0 127:39.81 htop
26103 kan       20   0 2780624 203956  14212 S   0.3  0.1  38:34.06 ZMQbg/43
26378 root      20   0   11800   5076   3516 S   0.3  0.0   2:52.40 containerd-shim
    1 root      20   0  225656   9404   6604 S   0.0  0.0   9:02.04 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:22.50 kthreadd
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0:0H
    6 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 mm_percpu_wq
    7 root      20   0       0      0      0 S   0.0  0.0   0:48.77 ksoftirqd/0
    9 root      20   0       0      0      0 I   0.0  0.0   0:00.00 rcu_bh
```

- htop



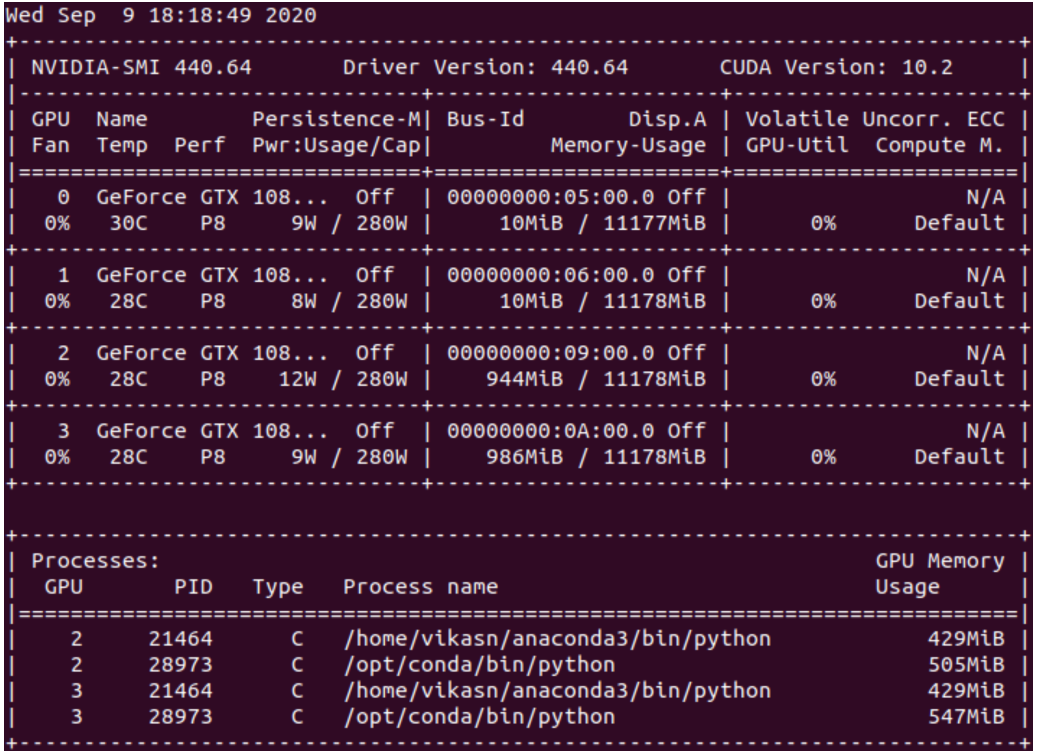
```
  1  [|||||||||||||||||||||||||                57.9%]     5  [||||||||||                          21.7%]
  2  [||||||||||||||                          31.8%]     6  [|||||||||||||||||||||||||||||||||   79.1%]
  3  [||                                       2.6%]     7  [||||||||||||||||||||                47.1%]
  4  [||||||||||                              21.2%]     8  [||                                   3.3%]
  Mem[||||||||||||                      3280/32170MB]     Tasks: 63, 35 thr; 4 running
  Swp[||                                  189/5999MB]     Load average: 2.01 2.04 2.12
                                                          Uptime: 79 days, 05:42:44

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
61133 mysql      20   0 4792M  721M  4128 S 149.  2.2     311h /usr/sbin/mysqld
  655 mysql      20   0 4792M  721M  4128 R 97.7  2.2  9h46:44 /usr/sbin/mysqld
21039 www-data   20   0  415M 66004 30832 S 52.8  0.2  0:18.85 /usr/sbin/apache2 -k start
21463 www-data   20   0  421M 70812 29716 S 34.3  0.2  0:10.18 /usr/sbin/apache2 -k start
 2657 mysql      20   0 4792M  721M  4128 R 33.7  2.2 10:01.12 /usr/sbin/mysqld
21342 www-data   20   0  415M 65744 30604 S 29.7  0.2  0:11.48 /usr/sbin/apache2 -k start
64476 mysql      20   0 4792M  721M  4128 S 13.9  2.2  8:43.09 /usr/sbin/mysqld
21955 denis      20   0 27224  2228  1436 R  0.7  0.0  0:00.05 htop
61150 mysql      20   0 4792M  721M  4128 S  0.7  2.2 55:30.76 /usr/sbin/mysqld
61147 mysql      20   0 4792M  721M  4128 S  0.7  2.2 56:19.65 /usr/sbin/mysqld
17856 mysql      20   0 4792M  721M  4128 S  0.7  2.2  1:12.98 /usr/sbin/mysqld
21336 www-data   20   0  415M 65972 30580 S  0.0  0.2  0:13.89 /usr/sbin/apache2 -k start
61151 mysql      20   0 4792M  721M  4128 S  0.0  2.2  4:43.10 /usr/sbin/mysqld
61148 mysql      20   0 4792M  721M  4128 S  0.0  2.2 45:56.17 /usr/sbin/mysqld
61149 mysql      20   0 4792M  721M  4128 S  0.0  2.2 39:11.42 /usr/sbin/mysqld
    1 root       20   0 34144  2160  1024 S  0.0  0.0  0:17.53 /sbin/init
  476 root       20   0 19736   556   416 S  0.0  0.0  0:07.25 upstart-udev-bridge --daemon
  484 root       20   0 51484   852   708 S  0.0  0.0  0:07.81 /lib/systemd/systemd-udevd --daemon
  842 root       20   0 23528   512   464 S  0.0  0.0  0:02.18 /usr/sbin/vsftpd
  939 messagebu  20   0 39224   912   720 S  0.0  0.0  0:02.10 dbus-daemon --system --fork
  994 root       20   0 43444  1276  1116 S  0.0  0.0  0:00.06 /lib/systemd/systemd-logind
 1002 syslog     20   0  251M 64704   608 S  0.0  0.2  0:43.78 rsyslogd
 1003 syslog     20   0  251M 64704   608 S  0.0  0.2  0:00.75 rsyslogd
 1004 syslog     20   0  251M 64704   608 S  0.0  0.2  1:01.91 rsyslogd
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Skoltech

Skolkovo Institute of Science and Technology

GPU USAGE

- nvidia-smi



```
Wed Sep  9 18:18:49 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 440.64       Driver Version: 440.64       CUDA Version: 10.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX 108...  Off  | 00000000:05:00.0 Off |                  N/A |
|  0%   30C    P8     9W / 280W |     10MiB / 11177MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  GeForce GTX 108...  Off  | 00000000:06:00.0 Off |                  N/A |
|  0%   28C    P8     8W / 280W |     10MiB / 11178MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   2  GeForce GTX 108...  Off  | 00000000:09:00.0 Off |                  N/A |
|  0%   28C    P8    12W / 280W |    944MiB / 11178MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   3  GeForce GTX 108...  Off  | 00000000:0A:00.0 Off |                  N/A |
|  0%   28C    P8     9W / 280W |    986MiB / 11178MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+


+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    2     21464      C   /home/vikasn/anaconda3/bin/python            429MiB |
|    2     28973      C   /opt/conda/bin/python                        505MiB |
|    3     21464      C   /home/vikasn/anaconda3/bin/python            429MiB |
|    3     28973      C   /opt/conda/bin/python                        547MiB |
+-----------------------------------------------------------------------------+
```

DOCKER:
HOW TO USE A CONTAINER AS A SANDBOX

Skoltech
Skolkovo Institute of Science and Technology

CREATE A SANDBOX

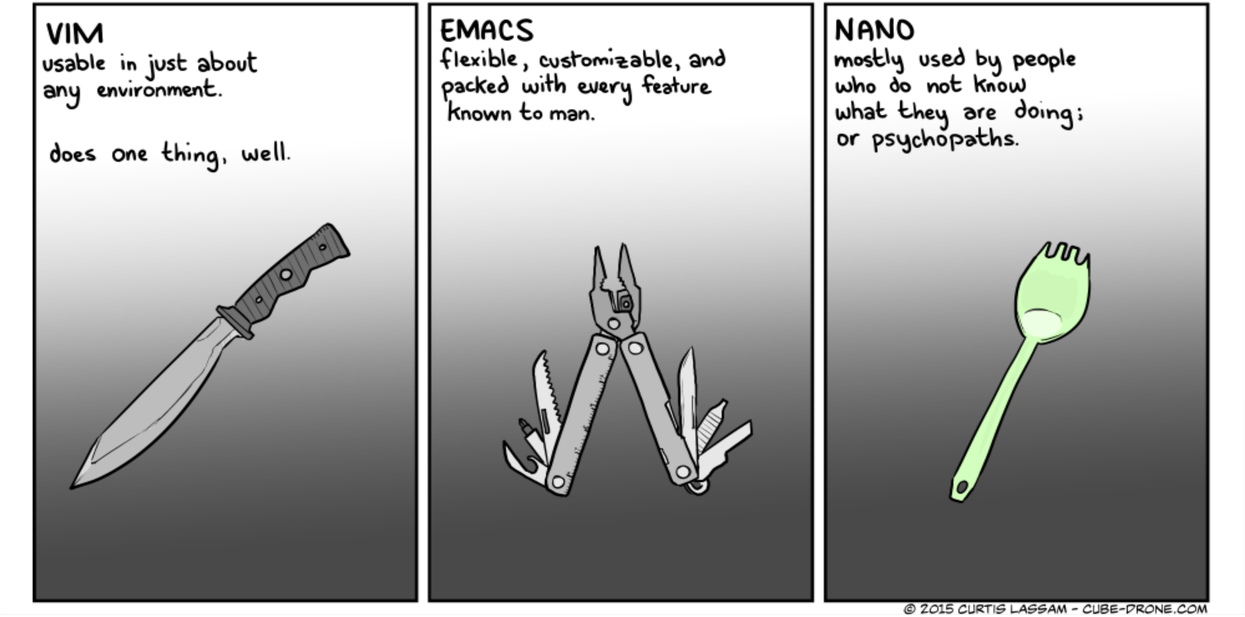
- Docker provides Containers
- A Container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

- Example of sandbox creation command

```
docker run --rm -it --name fse/student-n \
-v /home/fsestudent/student-n:/root \
--memory 24g --memory-swap 24g \
--cpuset-cpus '0-3' \
--gpus '"device=0,1"' \
nvidia/cuda:8.0-cudnn7-devel-ubuntu16.04
```

WRITE A CODE: TEXT EDITORS

VIM, EMACS, NANO


VIM
usable in just about any environment.
does one thing, well.
EMACS
flexible, customizable, and packed with every feature known to man.
NANO
mostly used by people who do not know what they are doing; or psychopaths.
© 2015 CURTIS LASSAM - CUBE-DRONE.COM

- Links on the tutorials
  - Vim Tutorial
  - Nano Tutorial
  - EMACS Tutorial

JUPYTER, PYCHARM, VSCODE

- Set it up on the remote
- Use it for the coarse file management on your remote machine and to write interactive notebooks

- Set it up on the remote
- Use it for the for code editing using your browser

- Connect your local installation to the remote
- Use it for the for code editing in your local PyCharm

CONCLUSIONS

CONCLUSIONS

- How to perform first steps on the server
- How to configure your environment
- How to inspect resources of the remote machine
- How to set up a sandbox
- How to write a code