Evolutionary Genetics

LV 25600-01 | Lecture with exercises | 4KP

Bioinformatics

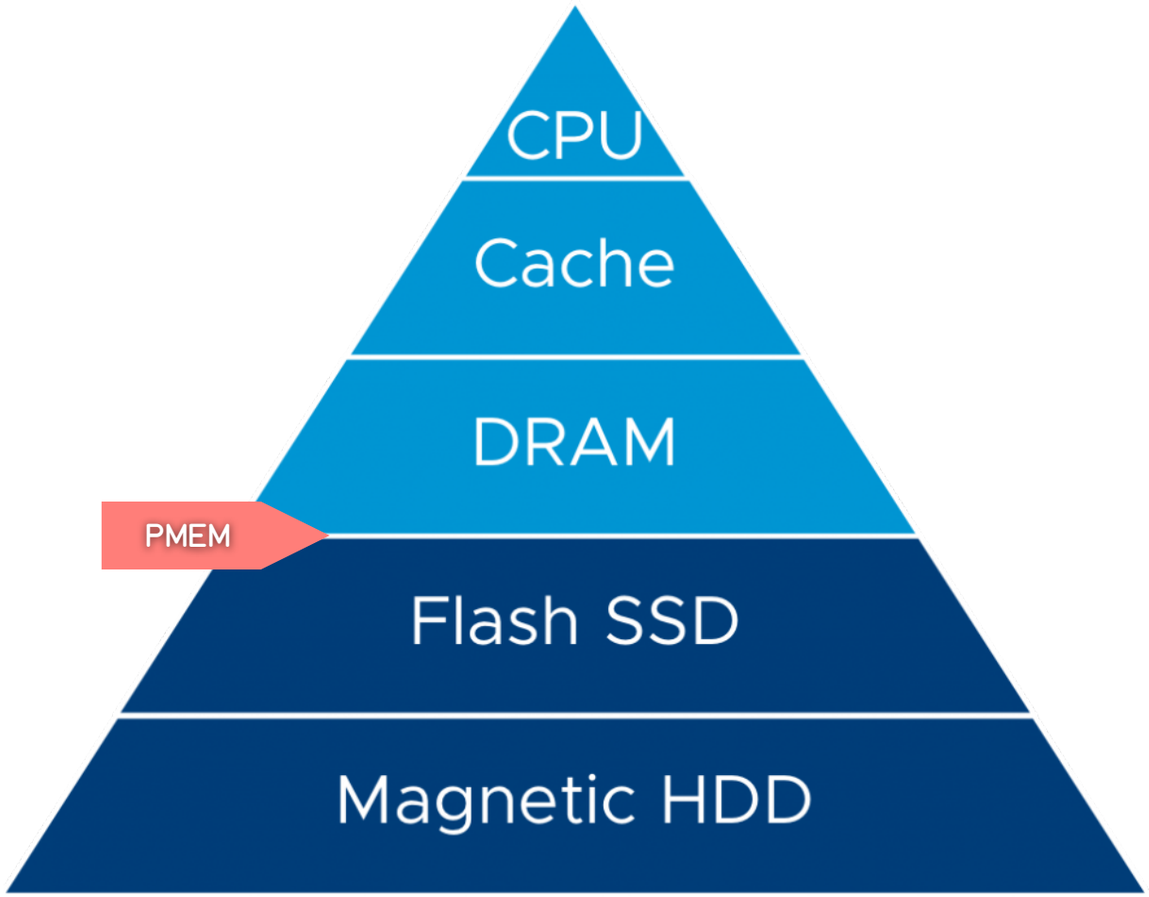
Jean-Claude Walser
jean-claude.walser [at] env.ethz.ch

HS2024

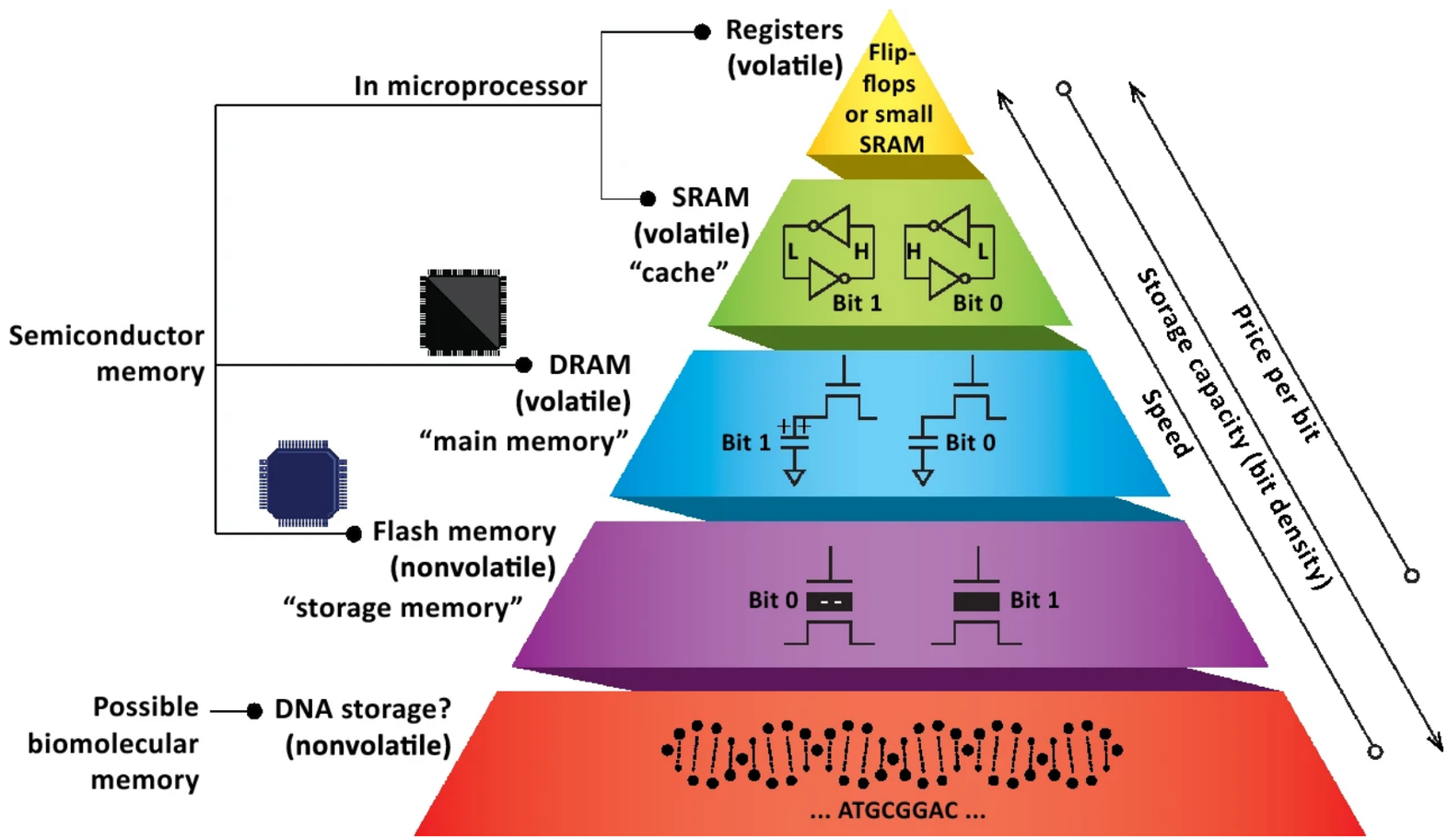
> To err is human, but to really foul things up you need a computer.
>
> — Paul R. Ehrlich

Remote Terminal

Model Identifier: MacBookPro
Number of Processors: 1
Total Number of Cores: 8
Memory: 32 GB (LPDDR5*)

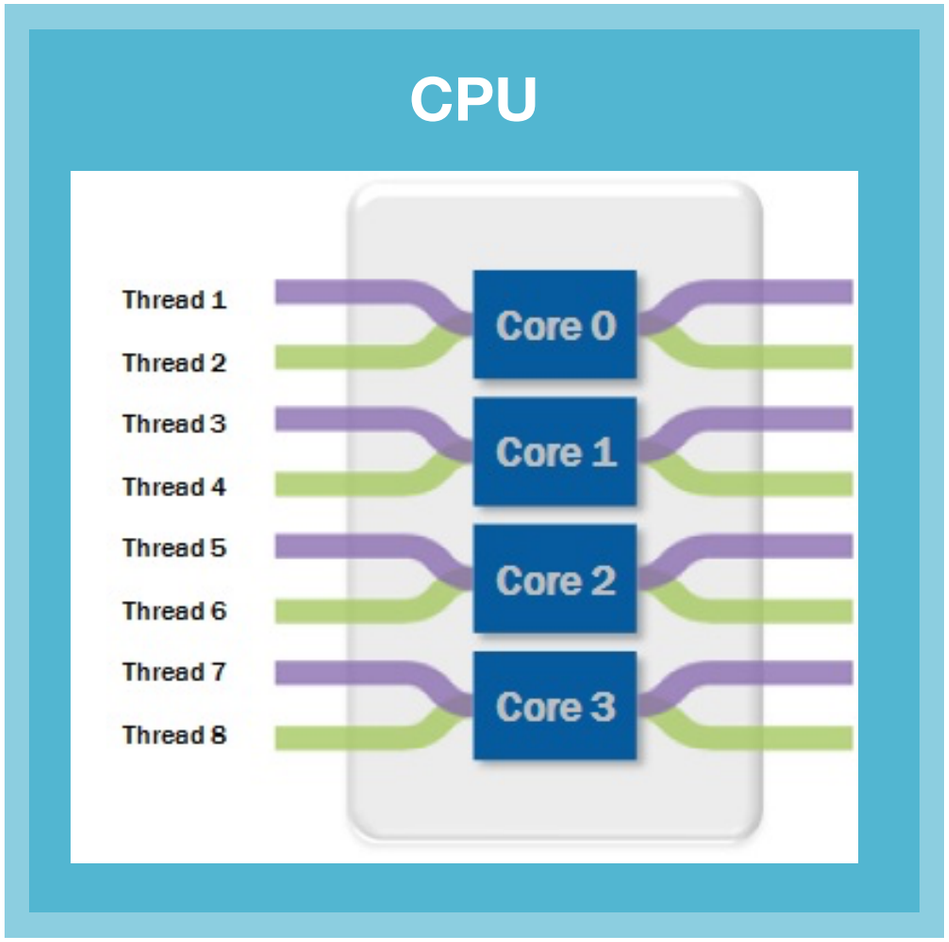


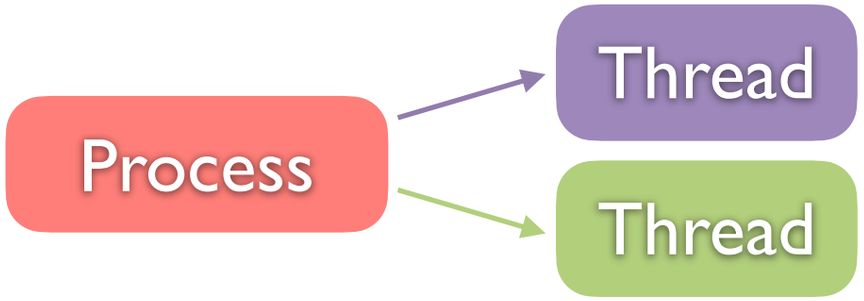
*Low-Power Double Data Rate (LPDDR), also known as LPDDR SDRAM, is a type of synchronous dynamic random-access memory that consumes less power and is targeted for mobile computers and devices such as mobile phones. Older variants are also known as Mobile DDR, and abbreviated as mDDR. (Source: Wikipedia)


Registers
(volatile)
Flip-
flops
or small
SRAM
In microprocessor
SRAM
(volatile)
"cache"
L
H
H
L
Bit 1
Bit 0
Semiconductor
memory
DRAM
(volatile)
"main memory"
Bit 1
Bit 0
Flash memory
(nonvolatile)
"storage memory"
Bit 0
Bit 1
Possible
biomolecular
memory
DNA storage?
(nonvolatile)
... ATGCGGAC ...
Speed
Storage capacity (bit density)
Price per bit

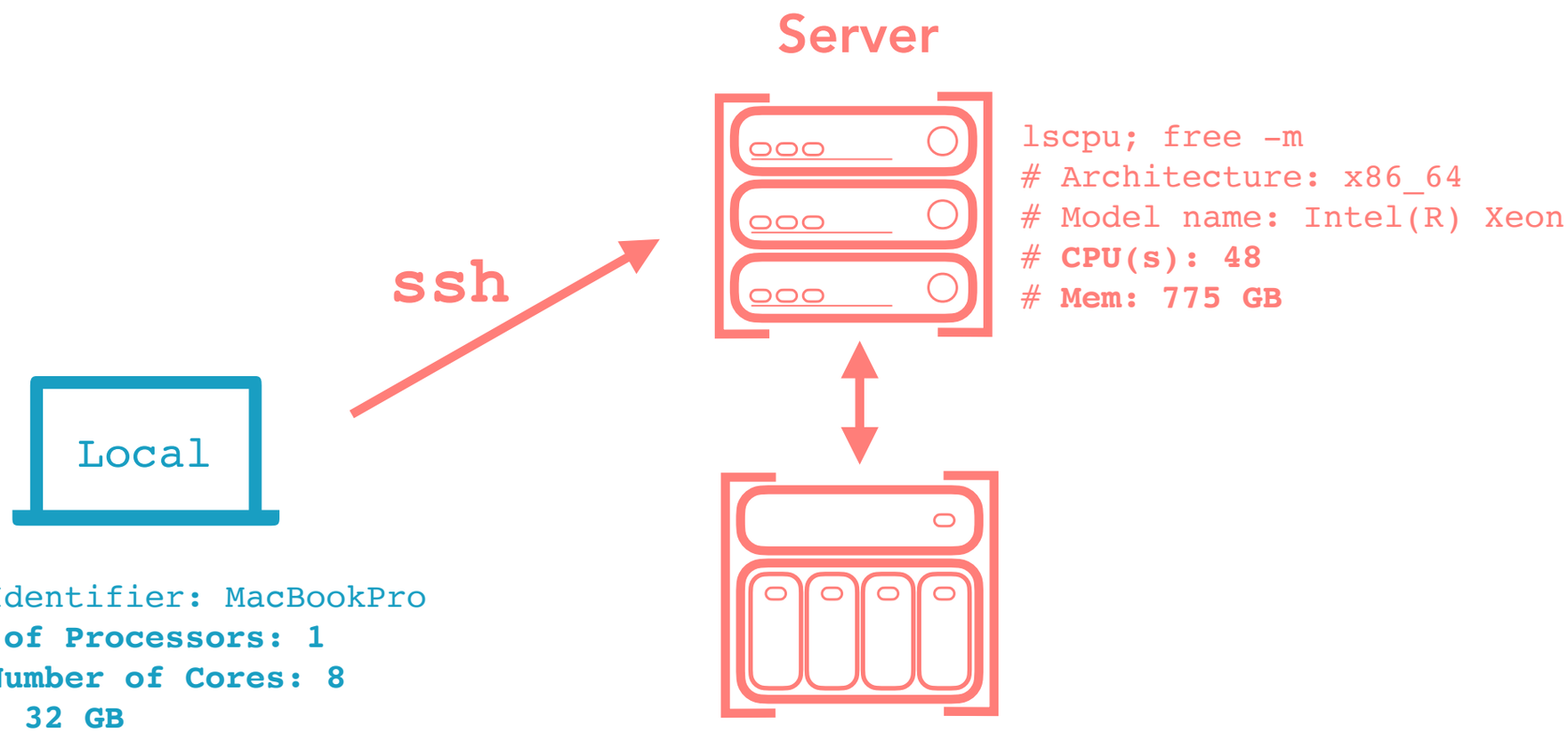
Local

```
Model Identifier: MacBookPro
Number of Processors: 1
Total Number of Cores: 8
Memory: 32 GB
```





A process can do more than one unit of work concurrently by creating one or more threads.

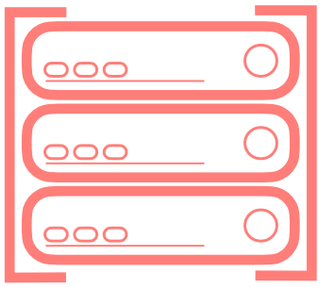

Server
lscpu; free -m
Architecture: x86_64
Model name: Intel(R) Xeon
CPU(s): 48
Mem: 775 GB
ssh
Local
Model Identifier: MacBookPro
Number of Processors: 1
Total Number of Cores: 8
Memory: 32 GB

A **remote** (compute) **server** is a (more powerful) computer or cluster of computers accessible over a network, typically the Internet, that allows users to perform computational tasks from a remote location. Managed using secure protocols such as **SSH**, these servers offer significant processing power, scalability and resource allocation, enabling multiple users to perform demanding tasks such as data processing, simulation or application hosting. They offer centralised management, security features and flexibility, making them essential for cloud computing, scientific research, software development and business operations where high performance and remote access are critical.

Secure Shell (**SSH**) is a cryptographic network **protocol for operating network services** securely over an unsecured network. Typical applications include **remote command-line**, login, and remote command execution, but any network service can be secured with SSH.

Source: Wikipedia

Server



```
lscpu; free -m
# Architecture: x86_64
# Model name: Intel(R) Xeon
# CPU(s): 48
# Mem: 775 GB
```

Local

```
Model Identifier: MacBookPro
Number of Processors: 1
Total Number of Cores: 8
Memory: 32 GB
```

ssh

A Virtual Machine is a software-based environment that emulates a physical computer, allowing it to run its own operating system and applications independently of the host system.

A **virtual machine** (VM) is a software-based emulation of a physical computer. Managed by a hypervisor, the VM is assigned virtual hardware, including CPU, memory and storage, allowing it to operate independently as a standalone server. VMs provide resource efficiency by enabling multiple isolated environments on a single physical machine, making them ideal for testing, development and scaling operations. They increase security through isolation, offer easy backup and recovery, and allow users to experiment with Linux without the need for dedicated hardware. VMs are essential for hosting web servers, databases and development environments, providing flexibility, scalability and efficient use of resources.

```
> ssh guest99@gdc-vserver.ethz.ch
# guest99@gdc-vserver.ethz.ch's password:🔑
> pwd
# /home/guest99
> users
jwalser guest99 guest03
```

ssh guest??@gdc-vserver.ethz.ch

```
## Monitoring server activity:
> top # press Q to leave top
```

```
top - 15:34:03 up 7 days,  5:00,  1 user,  load average: 0.00, 0.00, 0.00
Tasks: 326 total,   1 running, 325 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni, 99.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  63791.5 total,  61624.5 free,   1648.0 used,   1241.4 buff/cache
MiB Swap:  32216.0 total,  32216.0 free,      0.0 used.  62143.5 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    980 guest01   20   0  456568  10748   7424 S   0.7   0.0  10:11.96 mapstats.pl
    991 guest03   20   0  173396  16672  10664 S   0.1   0.2   0:04.28 grep
    804 guest11   20   0  696555  34532  13421 S   1.5   3.1  12:00.08 kronaplot.ph
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
```

CPU state percentages

us: user
sy: system
ni: nice
wa: IO-wait
hi: hardware interrupts
si: software interrupts

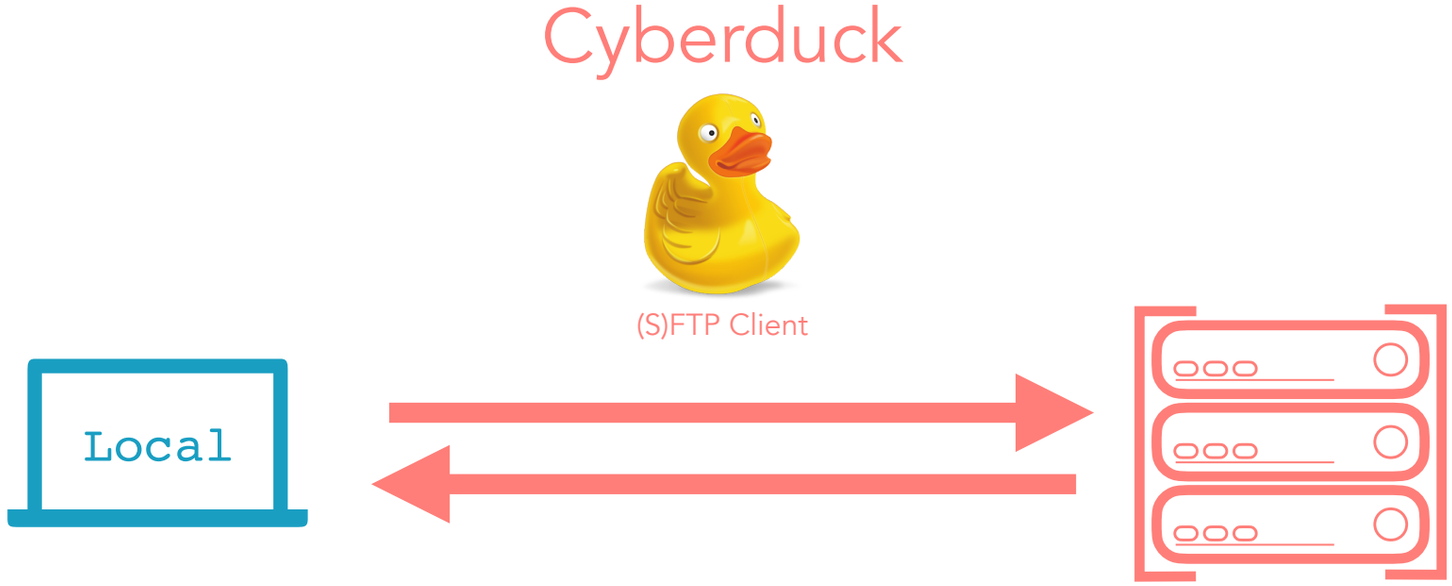
PID : Process ID
USER: USER
%CPU: 100% == 1 CPU
%MEM: Memory Usage
CND : Process

File Exchange

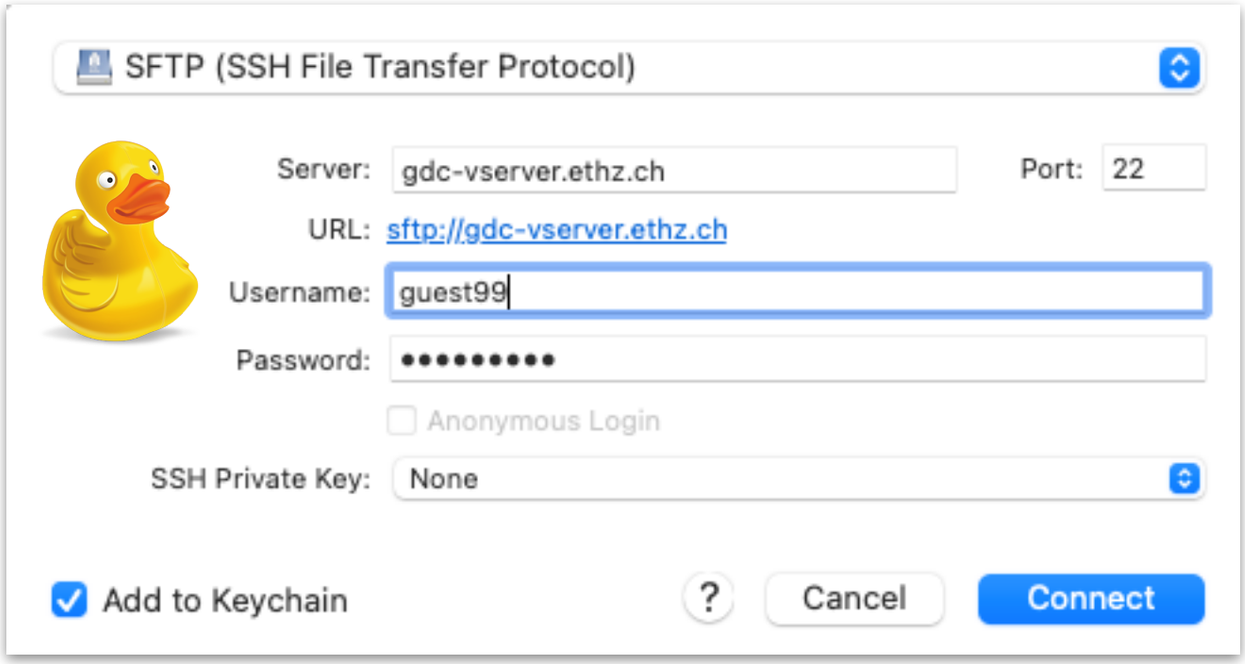

```
# Create a text files
> echo "Let me see the world" > go.txt
# Send the file to the server
> scp go.txt guest01@gdc-vserver.ethz.ch:/home/guest01
```


```
# Get the file back but rename it
> scp guest01@gdc-vserver.ethz.ch:/home/guest01/go.txt back.txt
> cat back.txt
```

A convenient way to upload or download (exchange) files from or to a remote server is via a (S)FTP client like Cyberduck.



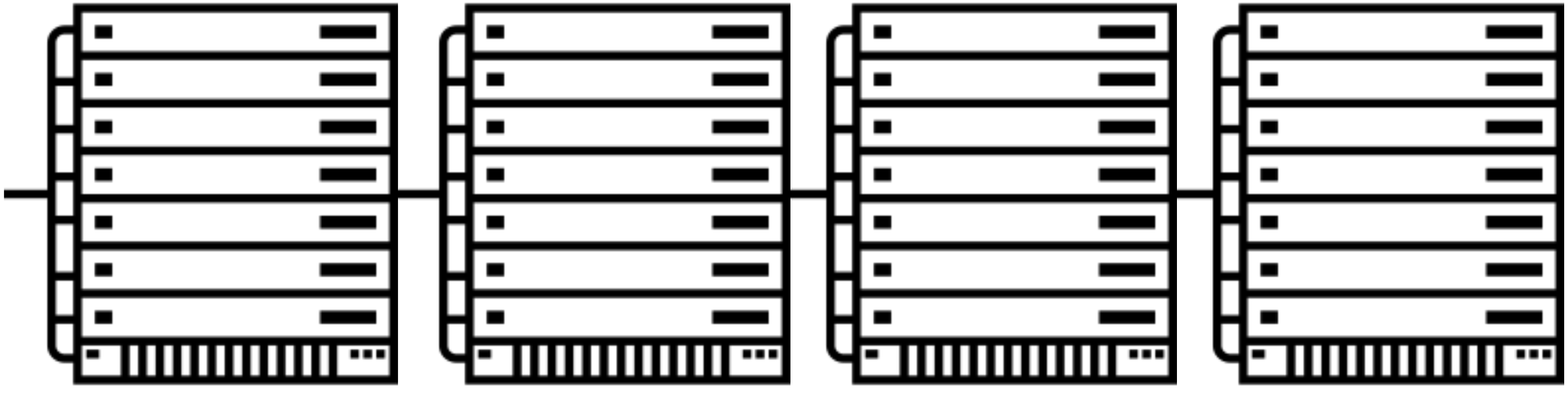
Settings for GDC Teaching VM-Server



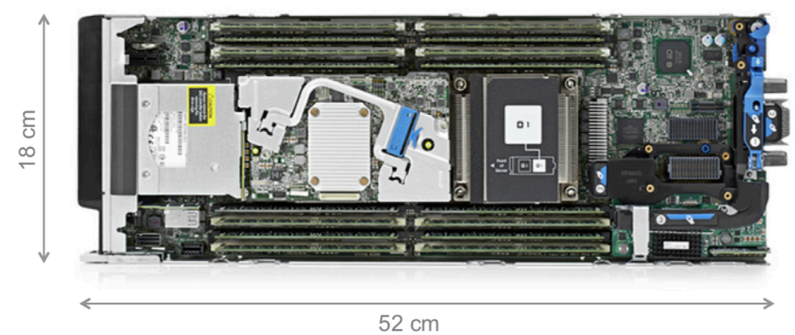
When you are finished, you should properly close the connection to the remote server using the *exit* command:

```
> exit
# logout
# Connection to gdcvserver.ethz.ch closed.
```

High Performance Cluster (HPC)




Euler VII


18 cm
52 cm

Euler (Erweiterbarer, Umweltfreundlicher, Leistungsfähiger ETH-Rechner)

Euler VI

Euler VI contains **216** compute nodes from Swiss company Dalco AG, each equipped with:

- Two **64-core** AMD EPYC 7742 processors (2.25 GHz nominal, 3.4 GHz peak)
- 512 GB of DDR4 memory clocked at 3200 MHz
- Local scratch: 920,618.0 MB

All these nodes are connected together via a dedicated 100 Gb/s InfiniBand HDR network.

216 nodes x 2 CPUs x 64 cores = 27’648

Euler VII — phase 1

The first phase of Euler VII contains **292** compute nodes — HPE ProLiant XL225n Gen10 Plus —, each equipped with:

- Two **64-core** AMD EPYC 7H12 processors (2.6 GHz nominal, 3.3 GHz peak)
- 256 GB of DDR4 memory clocked at 3200 MHz

All these nodes are connected together via a dedicated 100 Gb/s InfiniBand HDR network.

292 nodes x 2 CPUs x 64 cores = 37’376

Euler VII — phase 2

The 2nd phase of Euler VII contains **248** compute nodes — HPE ProLiant XL225n Gen10 Plus —, each equipped with:

- Two **64-core** AMD EPYC 7763 processors (2.45 GHz nominal, 3.5 GHz peak)
- 256 GB of DDR4 memory clocked at 3200 MHz

All these nodes share the same network as Euler VII phase 1.

248 nodes x 2 CPUs x 64 cores = 31’744

Euler VIII

Euler VIII contains **192** compute nodes from Swiss company Dalco AG, each equipped with:

- Two **64-core** AMD EPYC 7742 processors (2.25 GHz nominal, 3.4 GHz peak)
- 512 GB of DDR4 memory clocked at 3200 MHz
- Local scratch: 920,618.0 MB

All these nodes are connected to the cluster's 100 Gb/s Ethernet network.

192 nodes x 2 CPUs x 64 cores = 24’576

Euler IX

Euler VIII contains **192** compute nodes from Swiss company Dalco AG, each equipped with:

- Two **96-core** AMD EPYC 9654 processors (2.4 GHz nominal, 3.7 GHz peak)
- 384 GB of DDR5 memory clocked at 4800 MHz
- Local scratch: 1.8 TB

All these nodes are connected to the cluster's 100 Gb/s Ethernet network and will be connected to a dedicated 100 Gb/s InfiniBand HDR network later this year.

192 nodes x 2 CPUs x 96 cores = 36’864

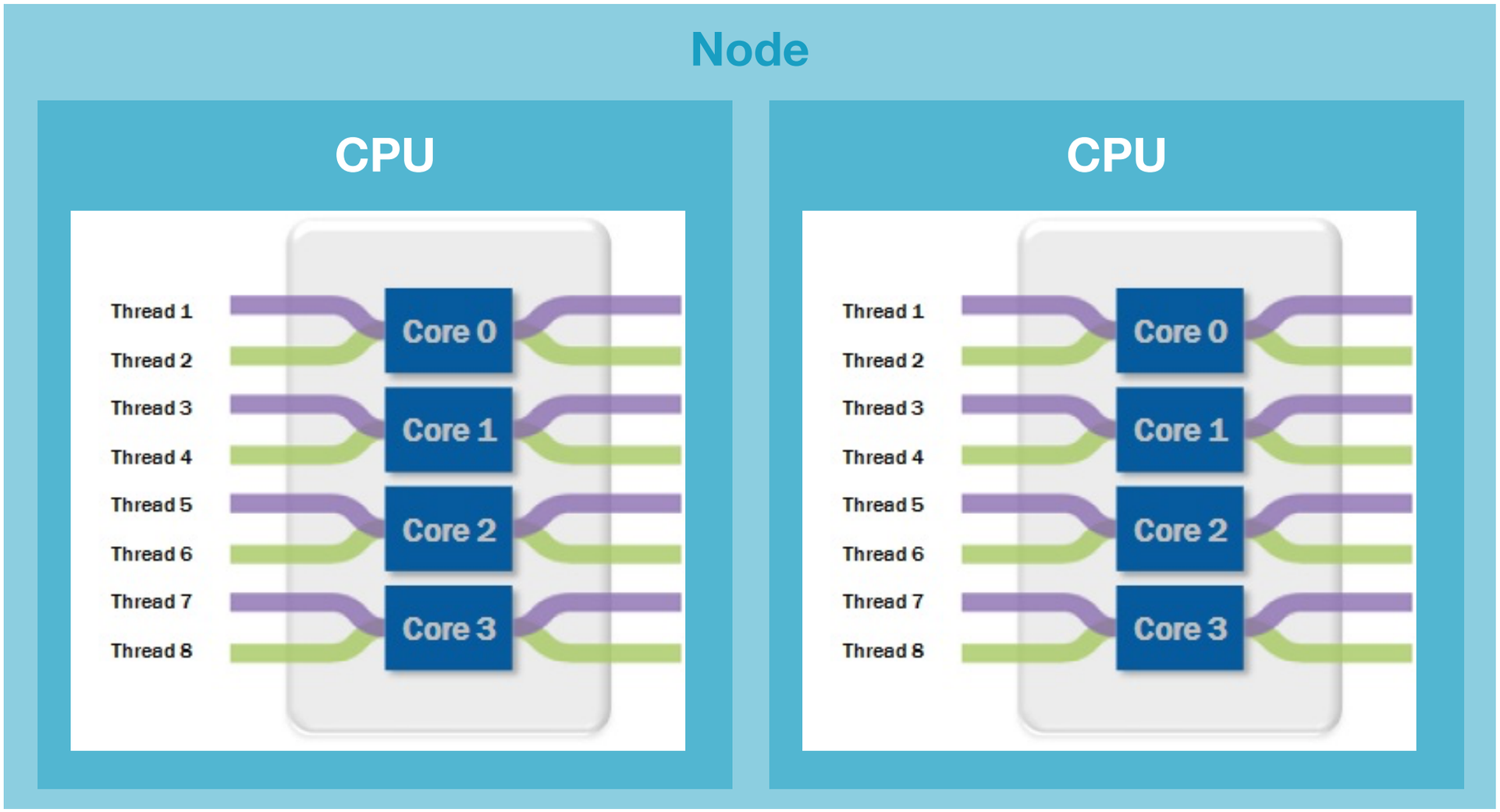
Total = 158’208

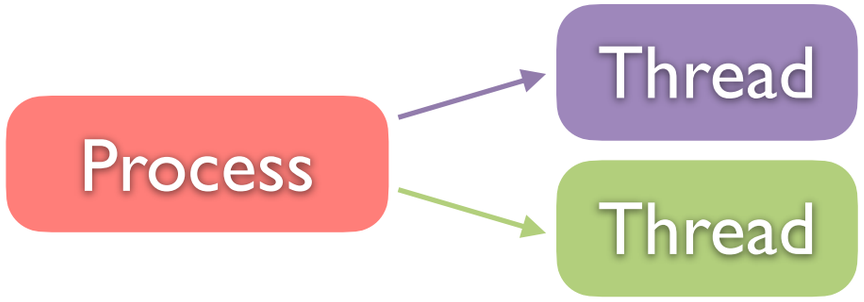
GPU nodes

Euler contains dozens of GPU nodes equipped with different types of GPUs:

- ~~9 nodes with 8 x Nvidia GTX 1080 (formerly in Leonhard Open)~~ *decommissioned in 2023*
- ~~47 nodes with 8 x Nvidia GTX 1080 Ti (formerly in Leonhard Open)~~ *decommissioned in 2023-2024*
- 4 nodes with 8 x Nvidia Tesla V100 (including some formerly in Leonhard Open)
- 93 nodes with 8 x Nvidia RTX 2080 Ti (including some formerly in Leonhard Open)
- 16 nodes with 8 x Nvidia Titan RTX
- 20 nodes with 8 x Nvidia Quadro RTX 6000
- 33 nodes with 8 x Nvidia RTX 3090
- 3 nodes with 8 x Nvidia Tesla A100 (40 GB PCIe)
- 3 nodes with 10 x Nvidia Tesla A100 (80 GB PCIe)
- 2 nodes with 8 x Nvidia Tesla A100 (80 GB PCIe)
- 40 nodes with 8 x Nvidia RTX 4090

https://scicomp.ethz.ch/wiki/Euler

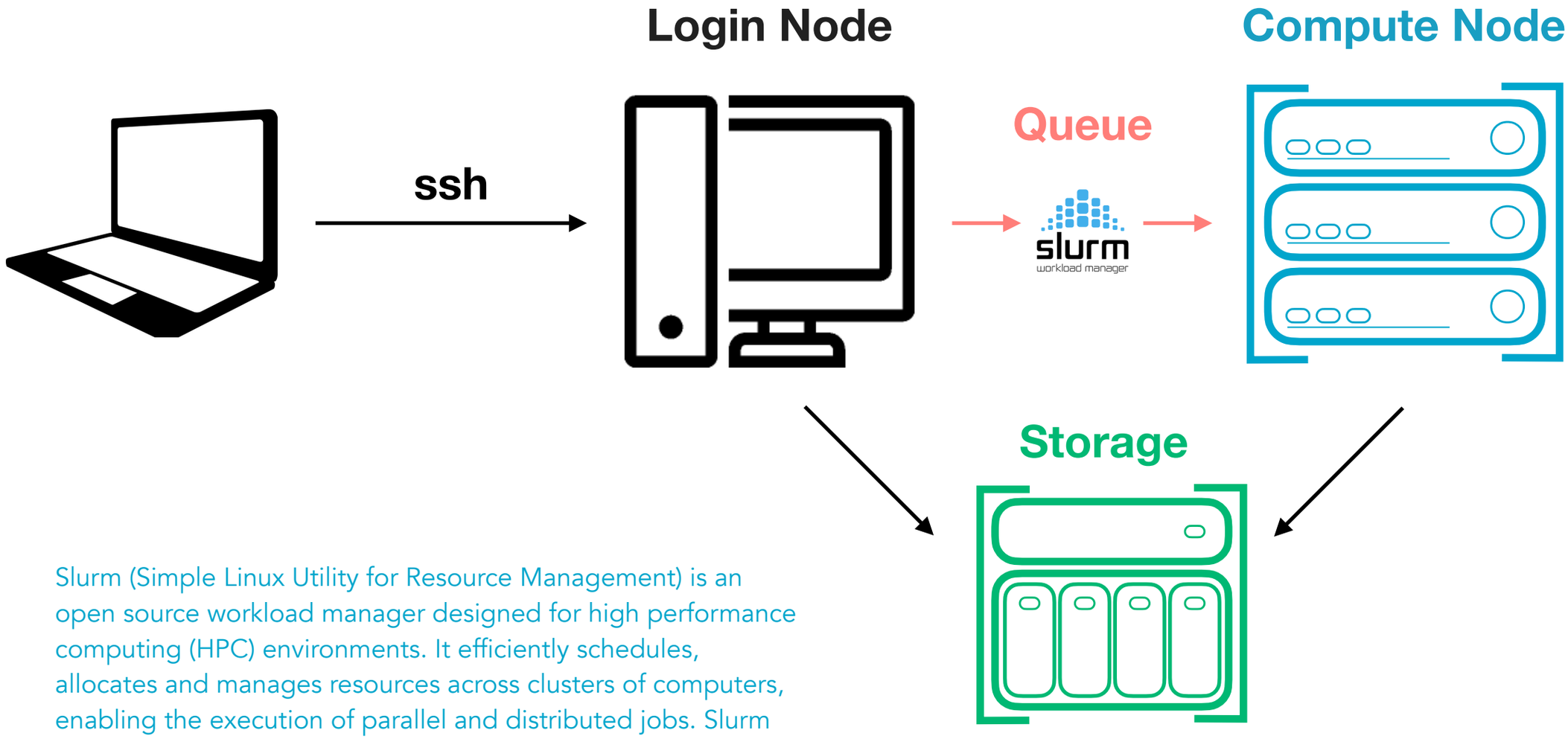




A process can do more than one unit of work concurrently by creating one or more threads.

A Few important terms:

- **HPC cluster**: relatively tightly coupled collection of compute nodes. Access to the cluster is provided through a login node. A resource manager and scheduler provide the logic to schedule jobs efficiently on the cluster.
- **Compute node**: an individual computer, part of an HPC cluster. Currently most compute node have two sockets, each with a single CPU, volatile working memory (RAM), a hard drive, typically small, and only used to store temporary files, and a network card.
- **CPU**: Central Processing Unit, the chip that performs the actual computation in a compute node. A modern CPU is composed of numerous cores, typically 8 or 10. It has also several cache levels that help in data reuse.
- **Core**: part of a modern CPU. A core is capable of running processes, and has its own processing logic and floating point unit. Each core has its own level 1 and level 2 cache for data and instructions. Cores share last level cache.
- **Threads**: a process can perform multiple computations, i.e., program flows, concurrently. In scientific applications, threads typically process their own subset of data, or a subset of loop iterations.



Slurm (Simple Linux Utility for Resource Management) is an open source workload manager designed for high performance computing (HPC) environments. It efficiently schedules, allocates and manages resources across clusters of computers, enabling the execution of parallel and distributed jobs. Slurm allows users to submit, monitor and control jobs via commands, managing resources such as CPUs, memory and GPUs to optimise performance. It also handles job prioritisation, queue management and fair resource allocation. Widely used in research institutions and supercomputing centres, Slurm is essential for running complex computations and simulations, ensuring that resources are used effectively.

Syntax

```
sbatch --mem-per-cpu=2G
       --time=04:00:00
       --ntasks=2
       --tmp=1G
       --wrap="<command(s)>"
```

Submission Example

```
module load bwa/0.7.17 fastq-screen/0.15.3

sbatch --job-name=FS \
       --output=FS-%j.log \
       --mem-per-cpu=1G \
       --time=00:10:00 \
       --ntasks=1 \
       --tmp=512 \
       --wrap="fastq_screen --threads 4 \
                            -conf fastq_screen.conf \
                            --aligner bwa \
                            s123x.fq.gz"
```

Submission Script

```
sbatch myscript.slurm
```

```
#!/bin/bash

## MyScript

#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --time=04:00:00
#SBATCH --mem-per-cpu=10G
#SBATCH --tmp=2G
#SBATCH --output=<JobTitle>_log.%j
#SBATCH —open-mode=append

<load module(s)>

<command(s)>
```

It is recommended that you monitor submitted SLURM jobs to ensure that they are running as expected and to avoid potential problems such as resource over-consumption, errors or job failures. Monitoring helps to identify inefficiencies early, such as long wait times in queues or incorrect resource allocations, so that adjustments can be made to improve overall job performance and system utilisation.

You can observe submitted SLURM jobs using several built-in commands:

Check job status: Use squeue to view the status of jobs in the queue, including their state (e.g., running, pending).

```bash
squeue -u <username>
Monitor resource usage: Use scontrol show job <job_id> to see detailed information about a specific
```

Track resource consumption: Use sacct to review resource usage (CPU, memory) of completed or running jobs.

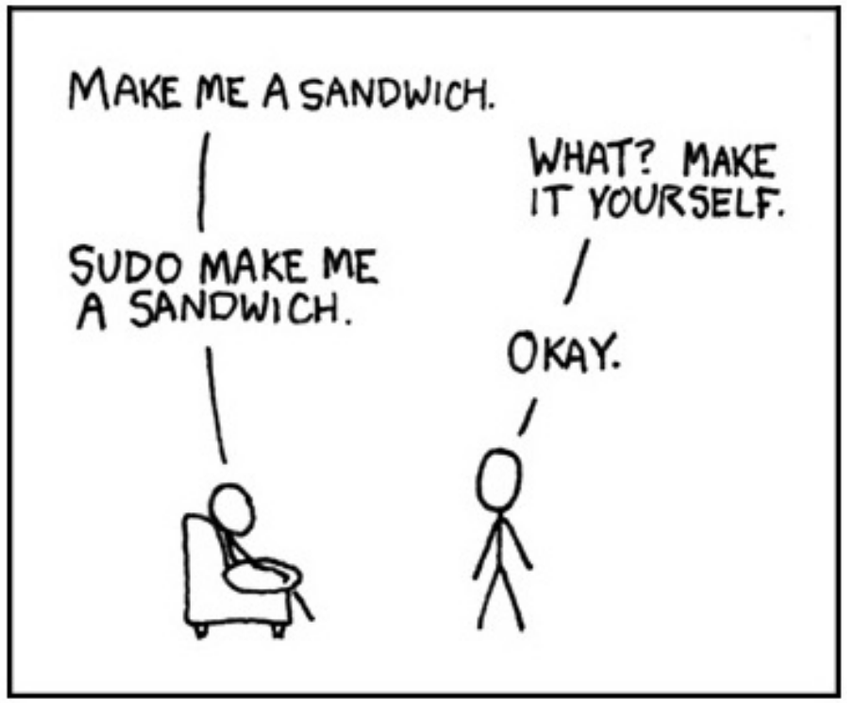
```bash
sacct -j <job_id>
```

Live monitoring: Use sstat to track real-time performance of running jobs.

```bash
sstat -j <job_id>
```

These commands help you monitor and optimize job execution, avoiding inefficiencies and errors.

Terminal
Command Line

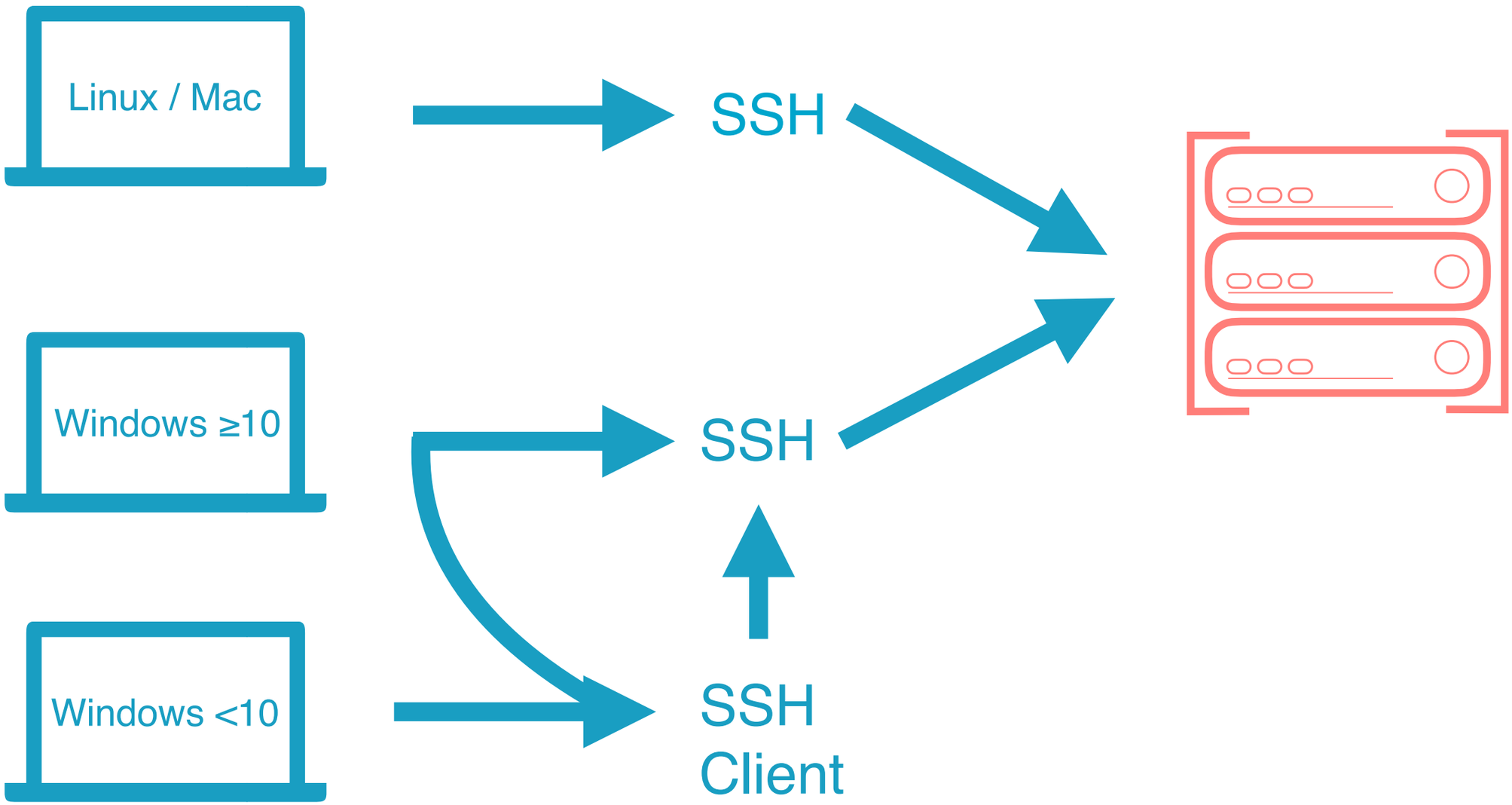


The war of the OS and the conflict of the Vs

Mac

Linux

Windows


Linux / Mac
SSH
Windows ≥10
SSH
Windows <10
SSH
Client

A **graphical user interface (GUI)** - often pronounced gooey - an interface that allows the user (you) to interact with programs in more ways than typing. GUIs were introduced in reaction to the steep learning curve of **command-line interfaces (CLI)**, which require commands to be typed on the keyboard. Since the commands available in command line interfaces can be numerous, complicated operations can be completed using a short sequence of words and symbols. **This allows for greater efficiency, productivity once many commands are learned, and better reproducibility.**

Where there is a shell, there is a (reproducible) way.

Terminal

Shell is a UNIX term for the interactive user interface with an operating system. The shell is the layer of programming that understands and executes the commands a user enters.

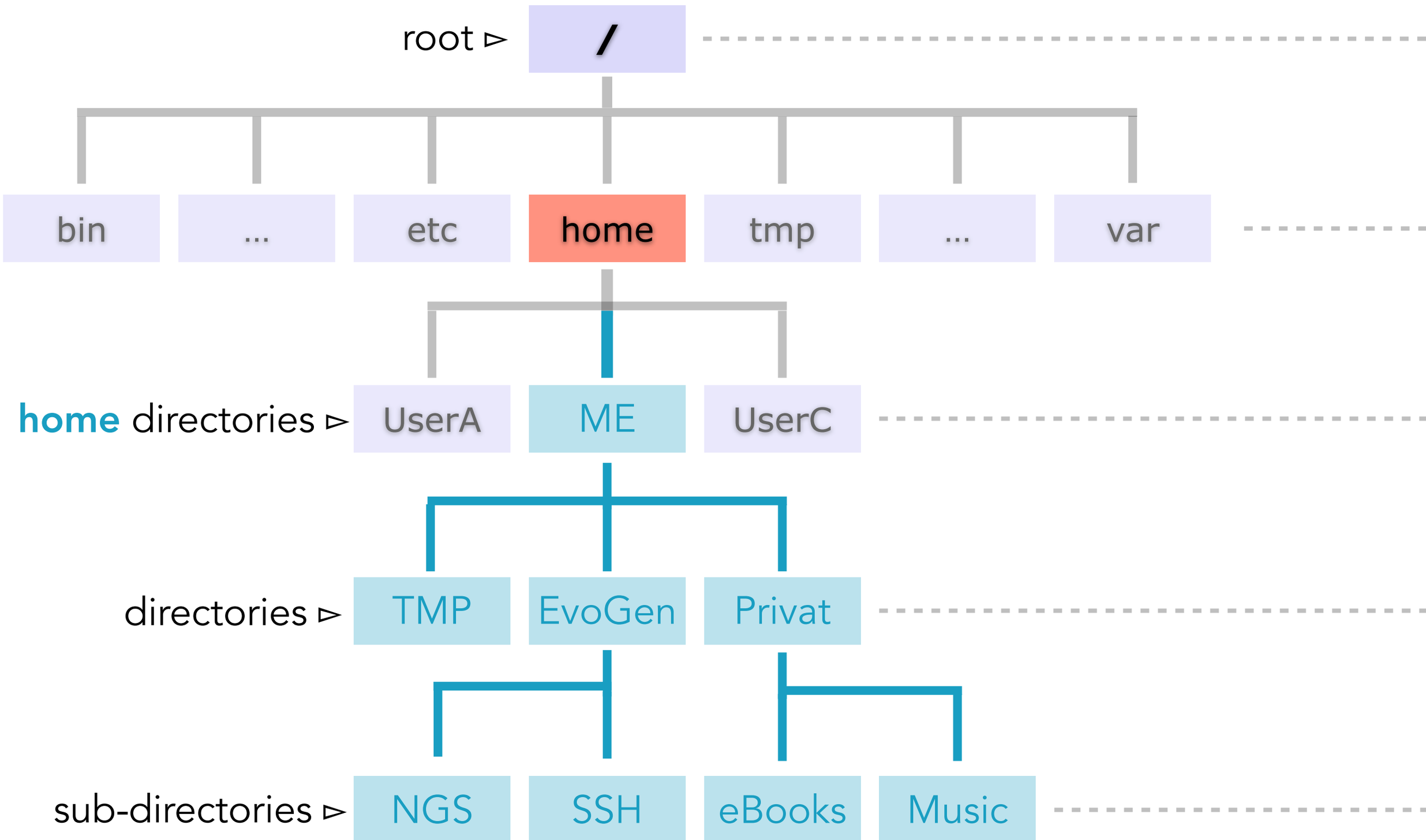
Bourne-Shell (sh)
Korn-Shell (ksh)
C-Shell (csh)
TC-Shell (tcsh)
Bourne-Again-Shell (bash)
Debian Almquist Shell (dash)
Z-Shell (zsh)
A-Shell (ash)
PowerShell / cmd.exe

What do I have?
```
$> echo ${SHELL}
```

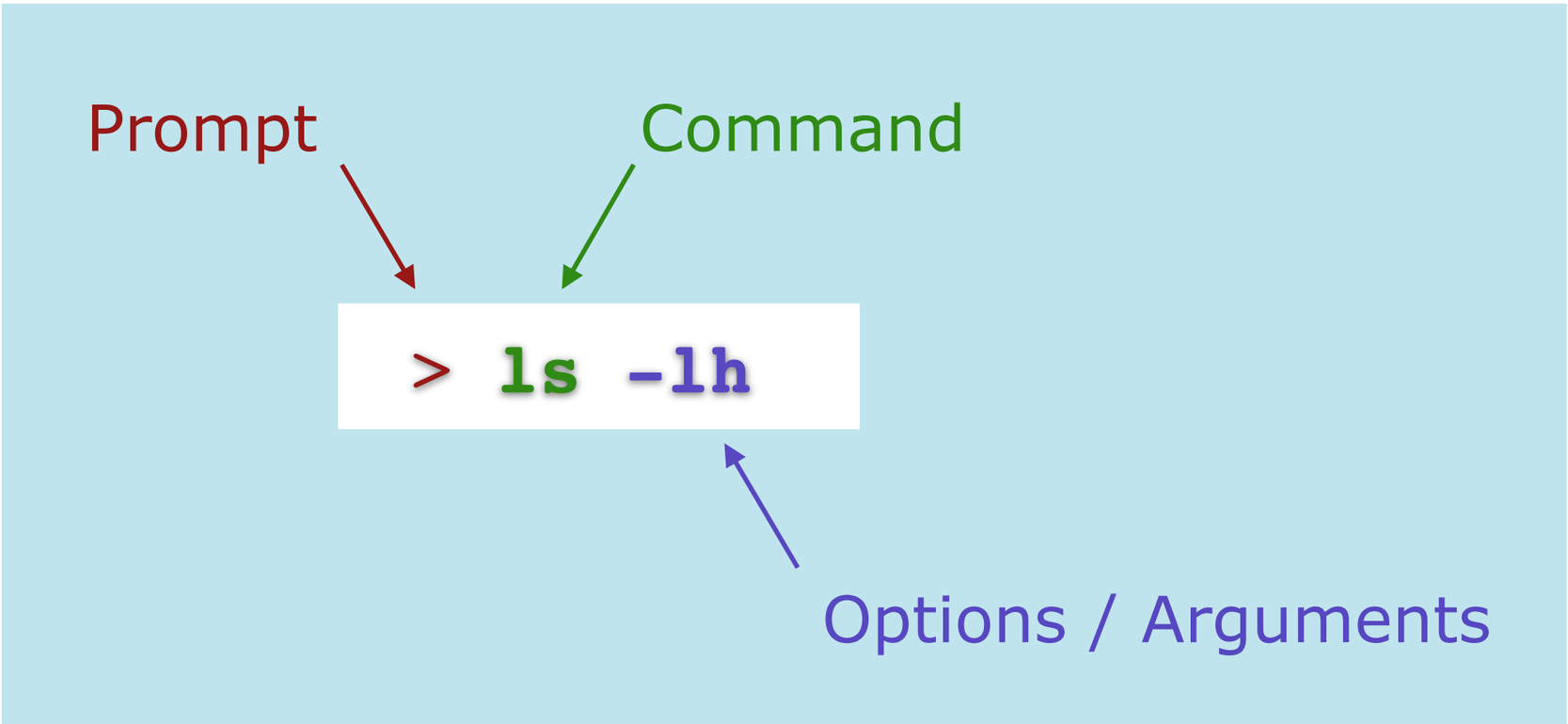
Check / Change Shell

```
echo ${SHELL}
#which zsh
#which bash

## Bash > Zsh
chsh -s /bin/bash

## Zsh > Bash
chsh -s /bin/zsh
```

Which one is better? Both shells will get the job done. The bash is a bit outdated (version 3.2 - 2006). Many of the conveniences provided by zsh can be made available in bash. It seems zsh is more helpful to newer shell user.


root ▻
/
bin
...
etc
home
tmp
...
var
home directories ▻
UserA
ME
UserC
directories ▻
TMP
EvoGen
Privat
sub-directories ▻
NGS
SSH
eBooks
Music

Command - Line



Built-in Help

```
> info <command>
```

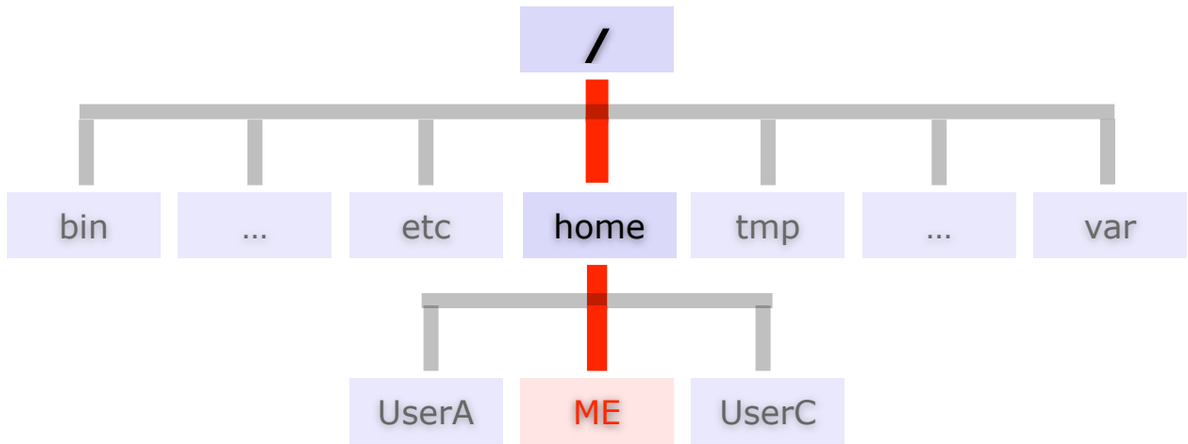
```
> info ls
```

```
> man <command>
```

```
> man ls
```

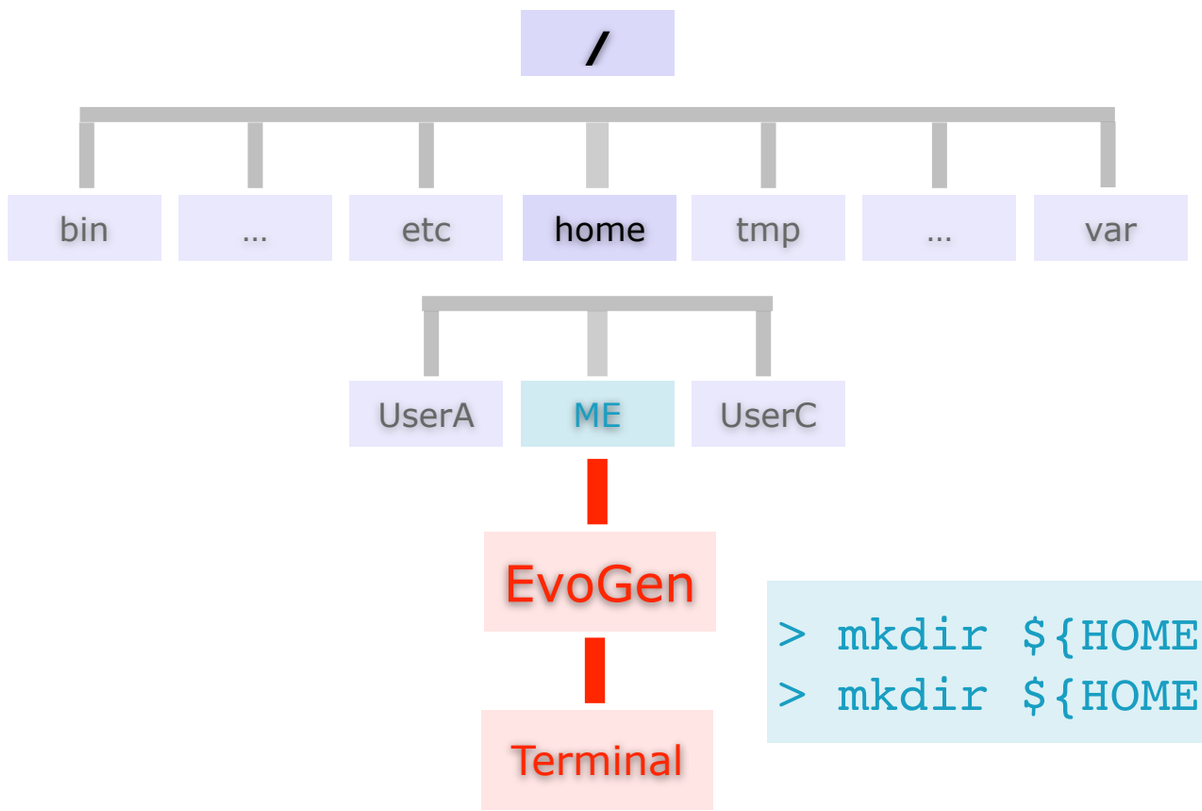
* press Q to leave info or manual

pwd - **p**rint name of current/**w**orking **d**irectory




```
> pwd
/home/ME
/root/ME
```

mkdir - creating/**ma**king **dir**ectories

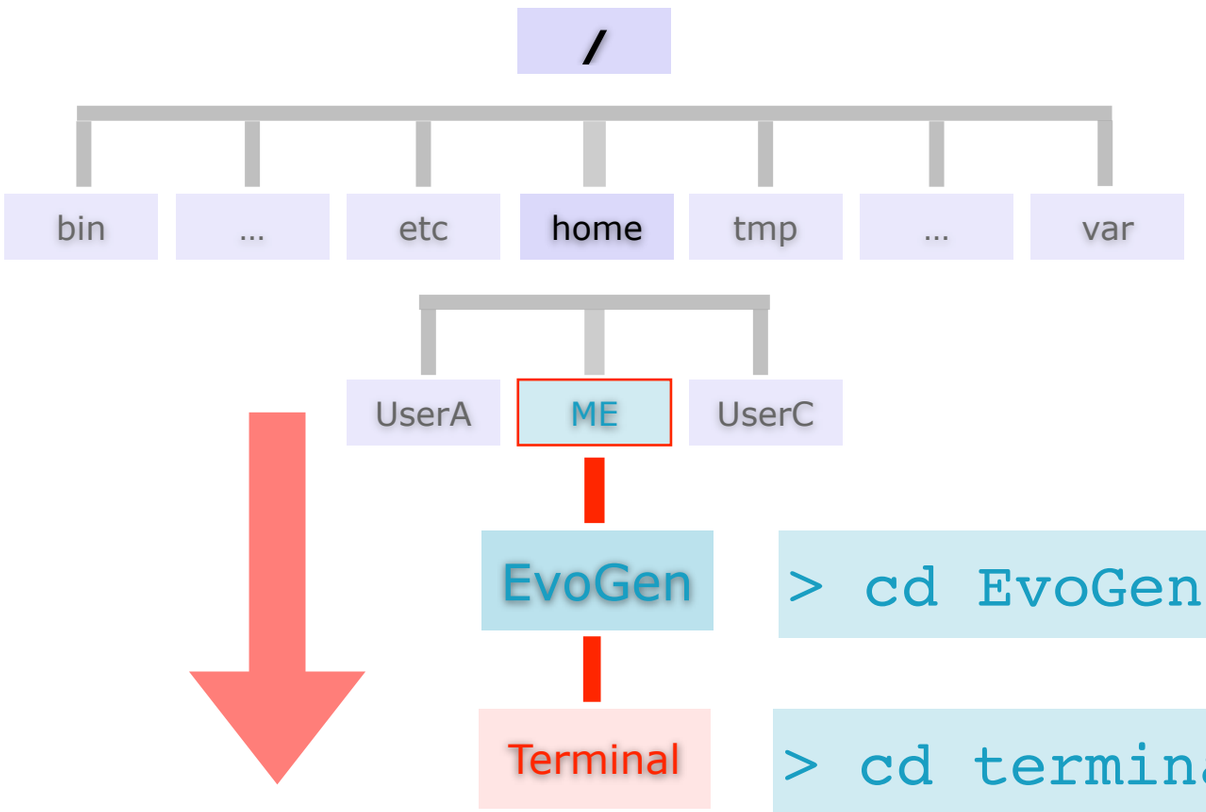



```
> mkdir ${HOME}/EvoGen
> mkdir ${HOME}/EvoGen/Terminal
```

Alternatively, use option p to create directories and sub-directories

```
> mkdir -p ${HOME}/EvoGen/Terminal
```

cd - **c**hange **d**irectory




```
> cd EvoGen
```

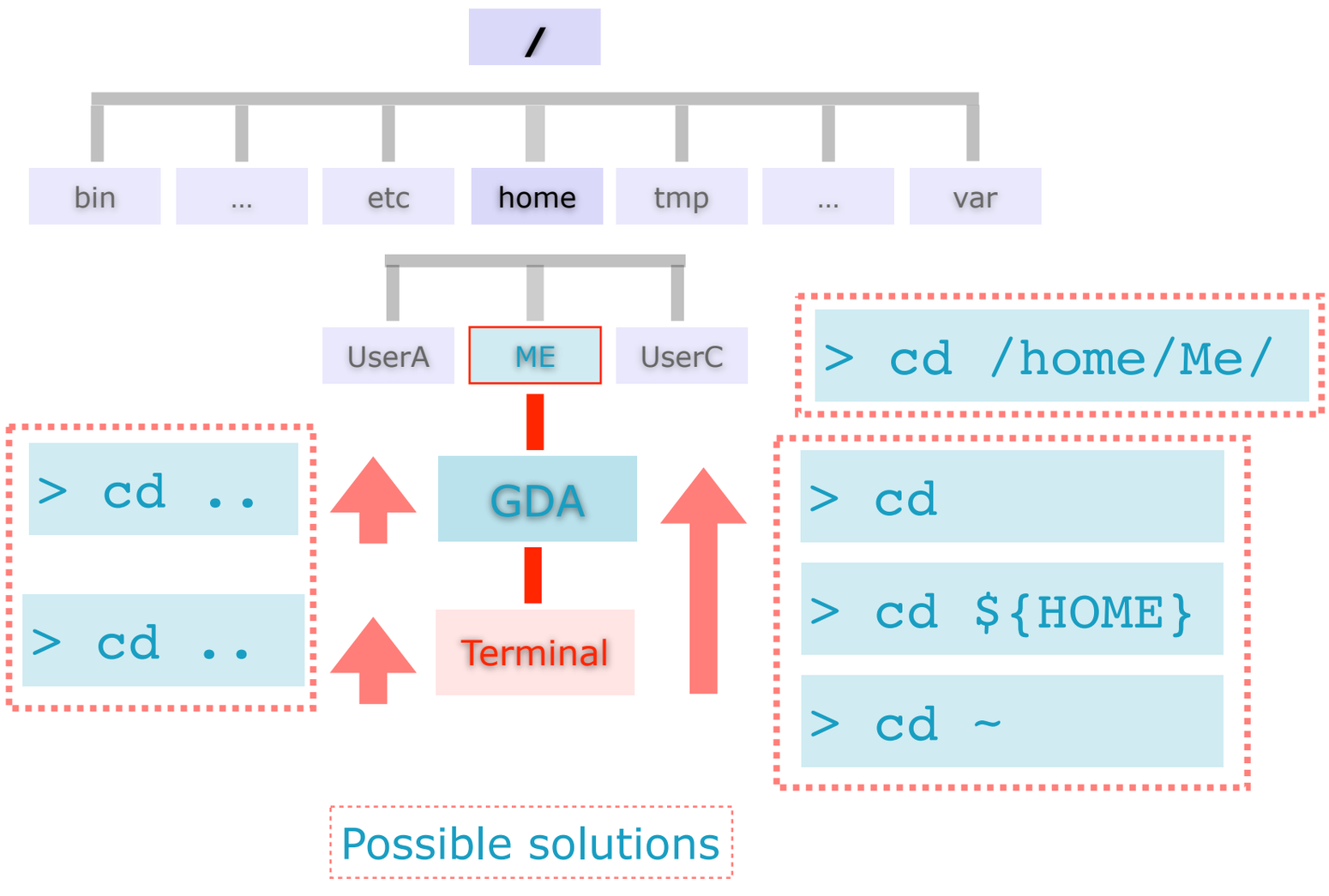
```
> cd terminal
```

One-Step Alternative

```
> cd EvoGen/terminal
```

Tip: Use [Tab] completion

ways to go ${**HOME**}



```
# Open an application (e.g., web browser Firefox)
> open /Volumes/Mac*/Applications/Firefox.app

# Open text file with a text editor (e.g., BBEdit)
> bbedit ${HOME}/TMP/test.txt
```

Local
Software
Management

Software
Dependencies

Parallel
Versions

Version Control

Application **A**
Requirments: Python 2.7

Application **B**
Requirments: Python >3.0

CONDA

*Package, **dependency and environment management** for any language—Python, R, Ruby, Lua, Scala, Java, JavaScript, C/ C++, FORTRAN. Conda as a package manager helps you find and install packages. If you need a package that requires a different version of Python, you do not need to switch to a different environment manager, because conda is also an environment manager. With just a few commands, you can set up a totally separate environment to run that different version of Python, while continuing to run your usual version of Python in your normal environment.*

BIOCONDA®

Bioconda is a channel for the conda package manager specializing in bioinformatics software. The conda package manager makes installing software a vastly more streamlined process. Conda is a combination of other package managers you may have encountered, such as pip, CPAN, CRAN, Bioconductor, apt-get, and homebrew. Conda is both language- and OS-agnostic, and can be used to install C/C++, Fortran, Go, R, Python, Java etc programs on Linux, Mac OSX, and Windows.

```
python --version
# Python 2.7.15
bwa
# -bash: bwa: command not found
blast -help
# -bash: blast: command not found

conda info --envs
source activate aligners
python --version
# Python 3.6.7
bwa
blastn -help
```

note: aligners = environment name

Evolutionary Genetics

LV 25600-01 | Lecture with exercises | **4**KP