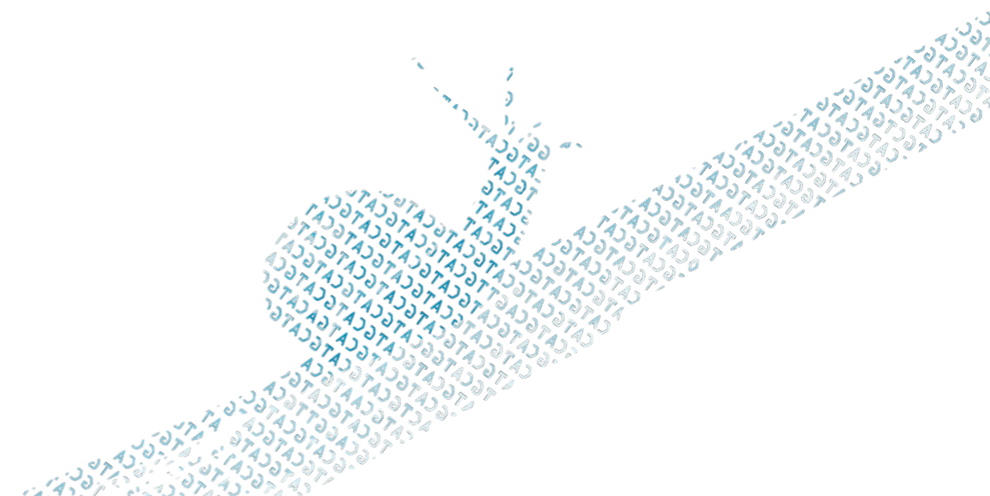


SSH / Remote Access

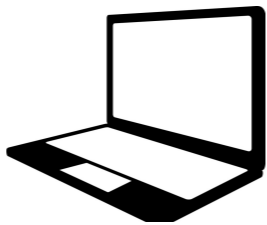
Tuesday, June 18, 2024



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

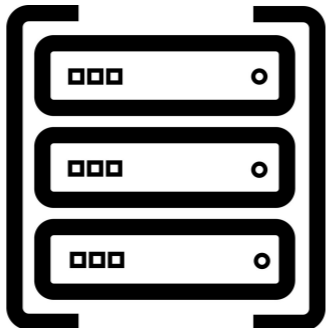
Local



Model Identifier: MacBookPro
Number of Processors: 1 (M1 Pro)
Total Number of Cores: 8
Memory: 32 GB

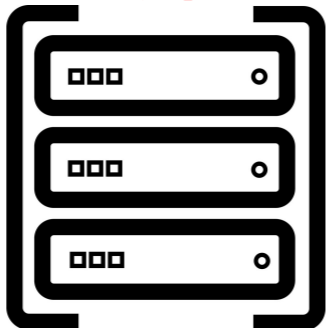


Remote



Abacus (Compute-Server)

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



Box (Data/Backup-Server)

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

```
ssh guest99@gdc-vserver.ethz.ch
```

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

```
top - 15:31:08 up 198 days, 5:44, 11 users, load average: 2.28, 2.95, 2.74  
Tasks: 4418 total, 3 running, 4414 sleeping, 1 stopped, 0 zombie  
Cpu(s): 1.4%us, 0.1%sy, 0.0%ni, 98.5%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st  
Mem: 926346512k total, 915660416k used, 10686096k free, 1139248k buffers  
Swap: 41943036k total, 3202716k used, 38740320k free, 858950540k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
101909	cstritt	20	0	145m	12m	2216	R	99.5	0.0	1670:03	python2
101912	cstritt	20	0	145m	12m	2216	R	99.1	0.0	1670:03	python2
47027	smrtanal	20	0	85.2g	1.8g	7224	S	39.5	0.2	77856:07	java
61899	jwalser	20	0	18404	4652	948	R	4.7	0.0	0:01.10	top
45866	smrtanal	20	0	65.9g	8.3g	8548	S	0.6	0.9	1050:29	java
525	root	20	0	0	0	0	S	0.3	0.0	1:33.86	ksoftirqd/130
649	root	20	0	0	0	0	S	0.3	0.0	49:55.41	events/6
683	root	20	0	0	0	0	S	0.3	0.0	123:26.83	events/40
8717	root	20	0	0	0	0	S	0.3	0.0	525:50.39	kondemand/41
8826	root	20	0	0	0	0	S	0.3	0.0	389:55.32	kondemand/150
61910	root	20	0	98.4m	3908	2944	S	0.3	0.0	0:00.02	sshd
1	root	20	0	19368	1136	916	S	0.0	0.0	66:56.88	init
2	root	20	0	0	0	0	S	0.0	0.0	0:18.69	kthreadd
3	root	RT	0	0	0	0	S	0.0	0.0	3613:39	migration/0
4	root	20	0	0	0	0	S	0.0	0.0	3:51.22	ksoftirqd/0
5	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	stopper/0
6	root	RT	0	0	0	0	S	0.0	0.0	128:44.45	watchdog/0
7	root	RT	0	0	0	0	S	0.0	0.0	2585:53	migration/1
8	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	stopper/1
9	root	20	0	0	0	0	S	0.0	0.0	2:22.84	ksoftirqd/1
10	root	RT	0	0	0	0	S	0.0	0.0	105:29.73	watchdog/1
11	root	RT	0	0	0	0	S	0.0	0.0	2263:28	migration/2
12	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	stopper/2

CPU state percentages

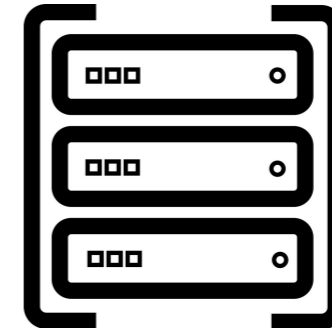
- us: user
- sy: system
- ni: nice
- wa: IO-wait
- hi: hardware interrupts
- si: software interrupts

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

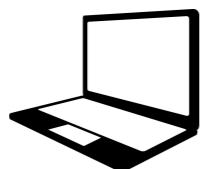
File Exchange



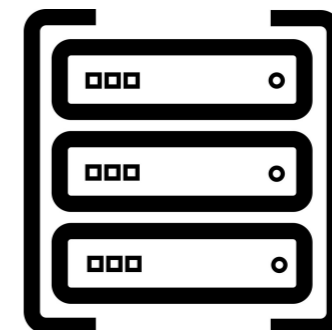
Upload



```
# 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
```

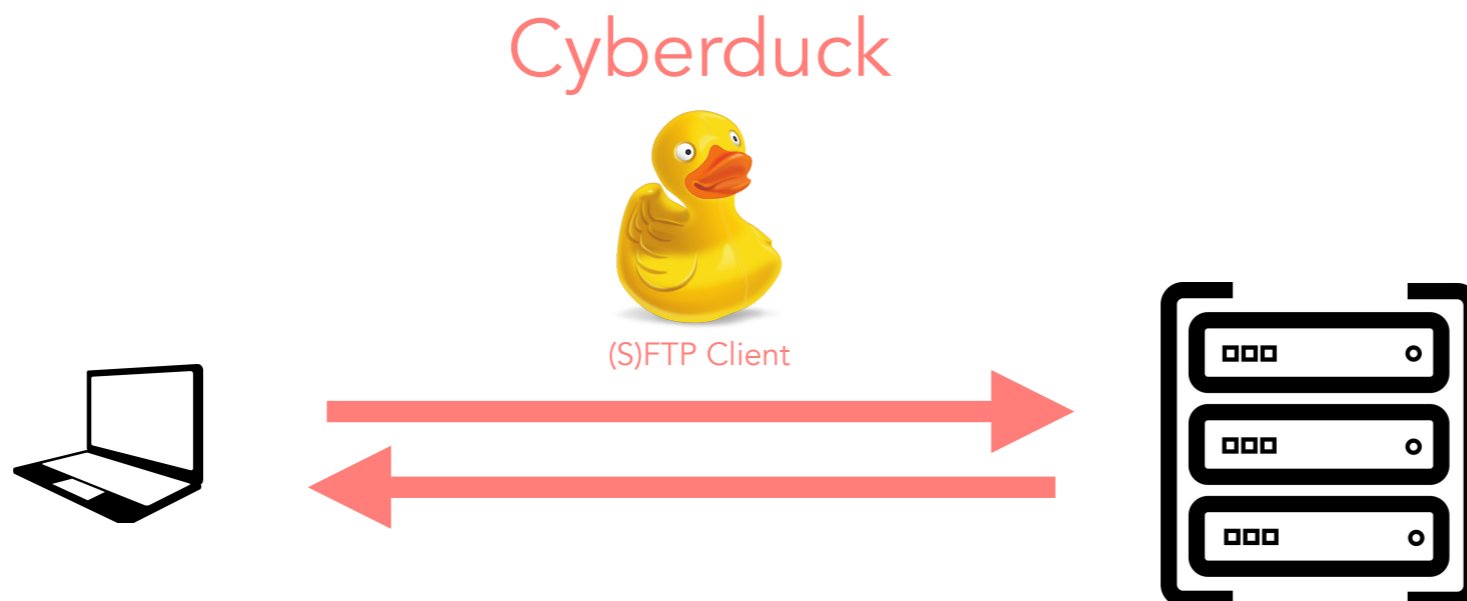


Download



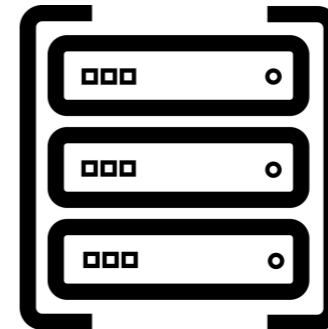
```
# Get the file back but rename it  
> scp guest01@gdc-vserver.ethz.ch:/home/guest01/go.txt back.txt  
> cat back.txt
```

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





Uploads



```
File01.fa  
File02.fa  
File03.fa  
File04.fa  
File05.fa
```

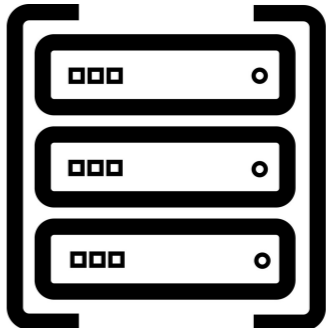
```
scp *.fa guest01@gdc-vserver.ethz.ch:/home/guest01
```

```
zip Files.fa.zip *.fa
```

```
scp *.zip guest01@gdc-vserver.ethz.ch:/home/guest01
```



Verify Upload



File01.fa
File02.fa
File03.fa
File04.fa
File05.fa

File01.fa
File02.fa
File03.fa
File04.fa
File05.fa

md5sum *.fa

md5sum *.fa

File01.fa: 81f4c5b9205a18c4c47307d189846e3b
File02.fa: 3bc3be114fb6323adc5b0ad7422d193a
File03.fa: 18dfa68a1bdd6e4329ab2d7227687d91
File04.fa: 5561612324670105ad9b52d88b37ede6
File05.fa: b45b0e608ef277ed8b530408d111efa5



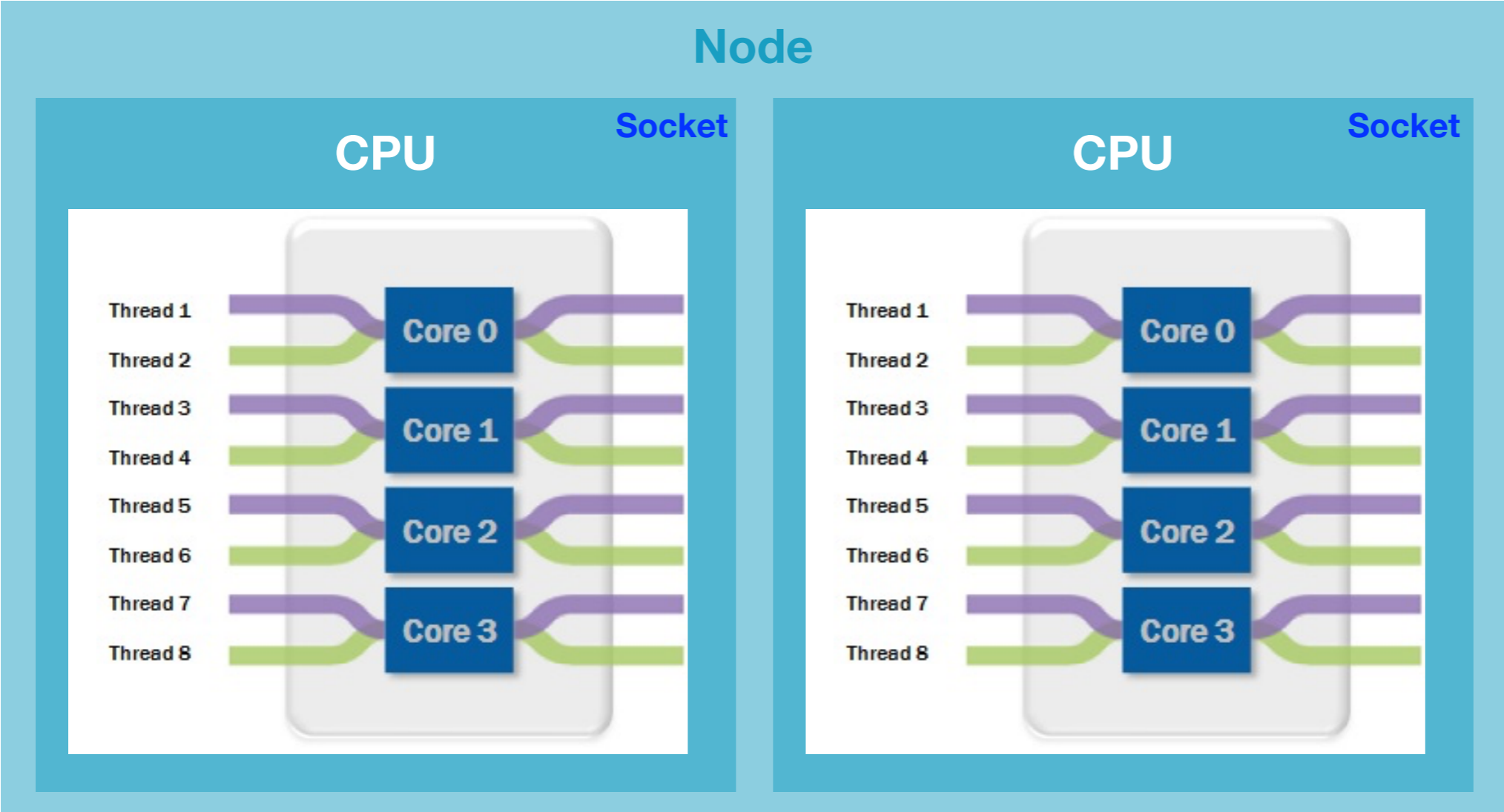
File01.fa: 81f4c5b9205a18c4c47307d189846e3b
File02.fa: 3bc3be114fb6323adc5b0ad7422d193a
File03.fa: e355950460780d84ca94a28885cb05e8
File04.fa: 5561612324670105ad9b52d88b37ede6
File05.fa: b45b0e608ef277ed8b530408d111efa5

```
md5sum [option] [file(s)]
```

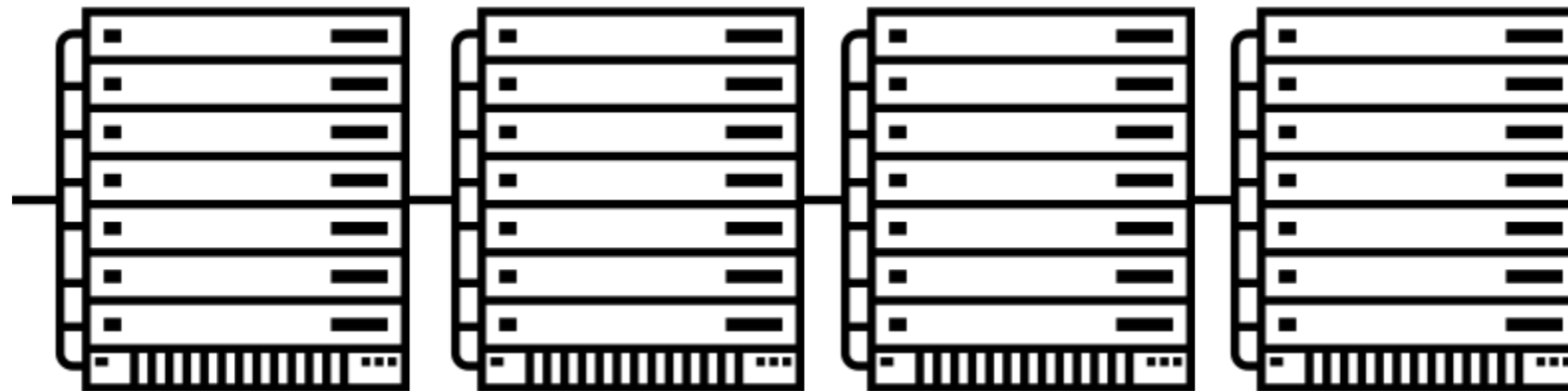
Compute or check 128-bit MD5 checksums is used to ensure that no change has been made to a file, e.g. ensure that a file was not corrupted during transfer. With no files or - specified, read from standard input. The exit status is 0 for success and nonzero for failure.

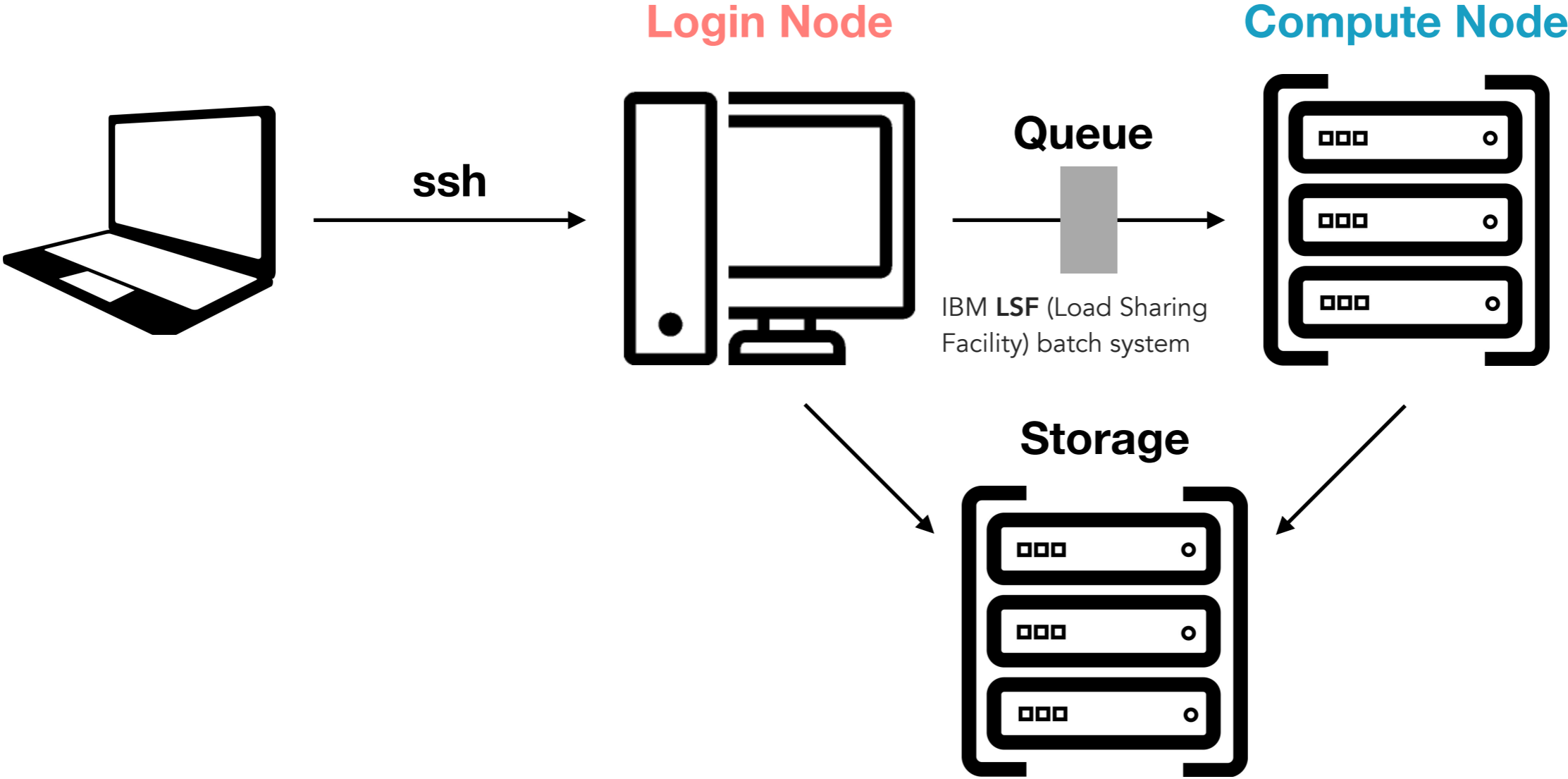
A Few important terms:

- **Compute node:** 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.



High Performance Cluster (HPC)





Euler II

Euler II contains **768** compute nodes of a newer generation — [BL460c Gen9](#) —, each equipped with:

- Two 12-core [Intel Xeon E5-2680v3](#) processors (2.5-3.3 GHz)
- Between 64 and 512 GB of DDR4 memory clocked at 2133 MHz (32 × 512 GB; 32 × 256 GB; 32 × 128 GB; 672 × 64 GB)

Euler II also contains **4** very large memory nodes — Hewlett-Packard [DL580 Gen9](#) —, each equipped with:

- Four 16-core [Intel Xeon E7-8867v3](#) processors (2.5 GHz)
- **3072** GB of DDR4 memory clocked at 2133 MHz

Euler III

Euler III contains **1215** compute nodes — [Hewlett-Packard m710x](#) —, each equipped with:

- A quad-core [Intel Xeon E3-1585Lv5](#) processor (3.0-3.7 GHz)
- 32 GB of DDR4 memory clocked at 2133 MHz
- A 256 GB [NVMe](#) flash drive

All these nodes are connected to the rest of the cluster via 10G/40G Ethernet.

Euler IV

Euler IV contains **288** high-performance nodes — [Hewlett-Packard XL230k Gen10](#) —, each equipped with:

- Two **18-core** [Intel Xeon Gold 6150](#) processors (2.7-3.7 GHz)
- 192 GB of DDR4 memory clocked at 2666 MHz

All these nodes are connected together via a new 100Gb/s InfiniBand EDR network.

Euler V

Euler V contains **352** compute nodes — Hewlett-Packard BL460c Gen10 —, each equipped with:

- Two **12-core** [Intel Xeon Gold 5118](#) processors (2.3 GHz nominal, 3.2 GHz peak)
- 96 GB of DDR4 memory clocked at 2400 MHz

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

Basic job submission

```
bsub -W 2:00 -n number_of_procs -R "rusage[mem=2048,scratch=5000]" <command>  
<parameters>  
  
# -n request multiple cores (or threads)  
# -R mem default the batch system allocates 1024 MB (1 GB) of memory per  
processor core  
# -R scratch for temporary data
```

Submission script

```
#!/bin/bash  
#BSUB -J "MyScript"          ## Job Title  
#BSUB -n 10                  ## Number of Cores  
#BSUB -R "rusage[mem=2048]"  ## Memory Request  
#BSUB -W 2:00                ## Running Time  
  
## Load environment  
module load gcc/4.8.2 gdc perl/5.18.4  
  
## ...
```

Job monitoring

```
[leonhard@euler08 ~]$ bjobs 31989961
Job information
  Job ID           : 31989961
  Status           : RUNNING
  Running on node  : e1268
  User             : leonhard
  Queue            : normal.4h
  Command          : compute_pq.py
  Working directory: $HOME/testruns
Requested resources
  Requested cores   : 1
  Requested memory  : 1024 MB per core
  Requested scratch : not specified
  Dependency        : -
Job history
  Submitted at     : 08:45 2016-11-15
  Started at       : 08:48 2016-11-15
  Queue wait time  : 140 sec
Resource usage
  Updated at       : 08:48 2016-11-15
  Wall-clock       : 34 sec
  Tasks            : 4
  Total CPU time    : 5 sec
  CPU utilization   : 80.0 %
  Sys/Kernel time  : 0.0 %
  Total resident memory : 2 MB
  Resident memory utilization : 0.2 %
```

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.