جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

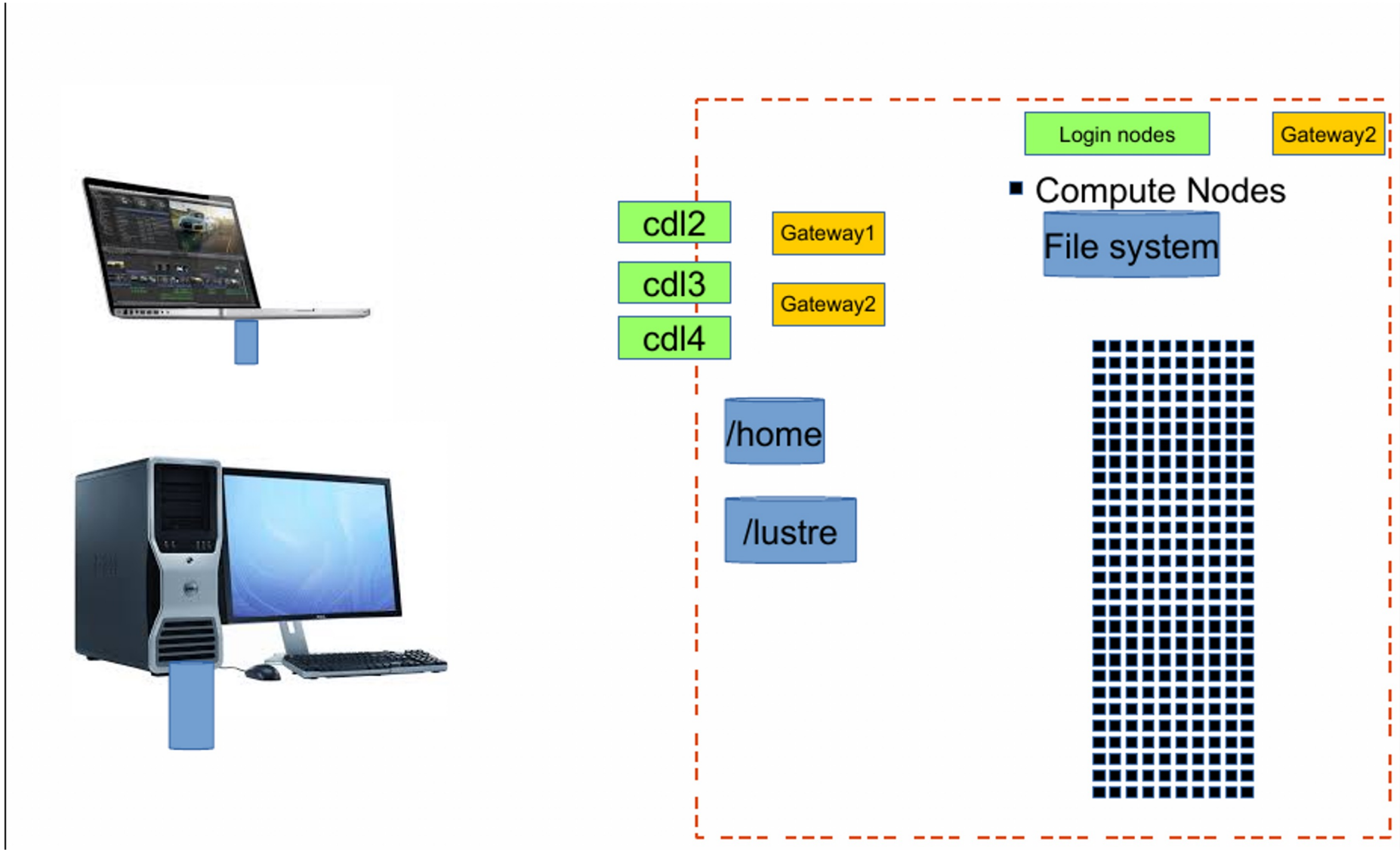
Core Labs and
Research Infrastructure

Accessing Shaheen

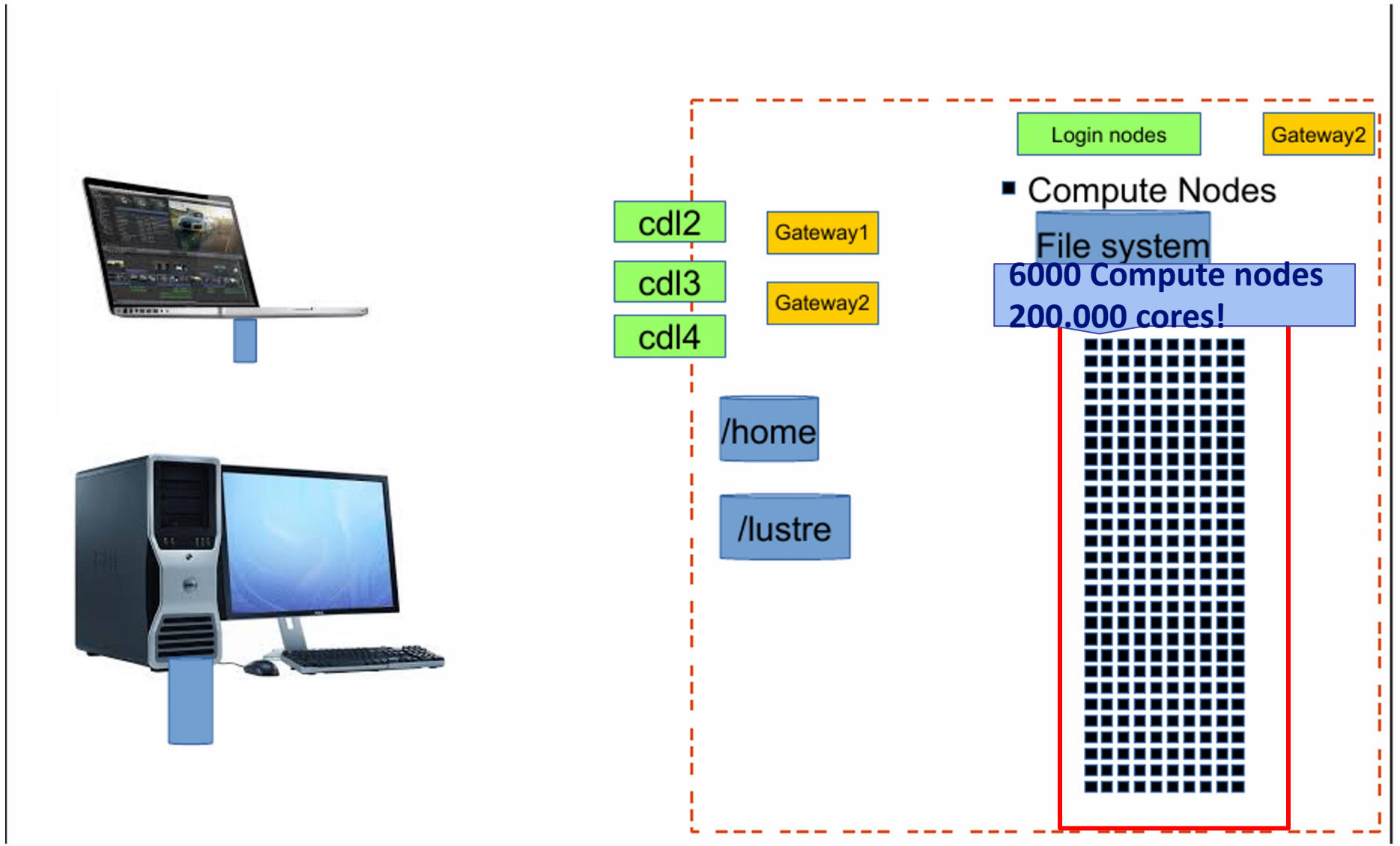
Accessing Shaheen

1. Login nodes are how you interact with Shaheen
2. Once account is created you can login
3. You need a **Terminal** or **SSH client** to execute SSH command
    1. SSH from your machine to Shaheen login node
    2. Use your KAUST credentials (email username and password)
4. Shaheen has two factor authentication
    1. Your email password
    2. One Time Password (OTP) (refreshed every time you login)

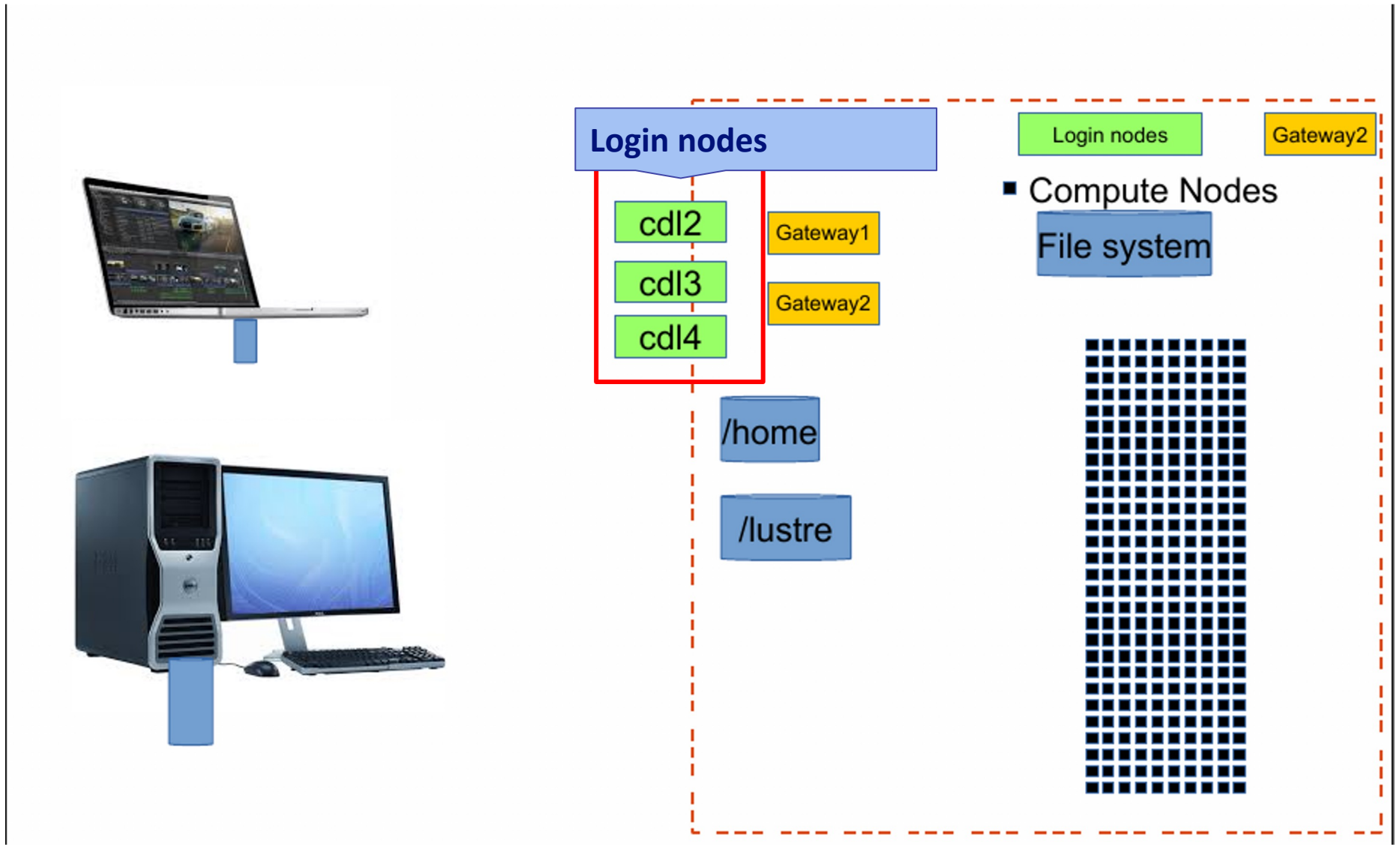
Shaheen's architecture



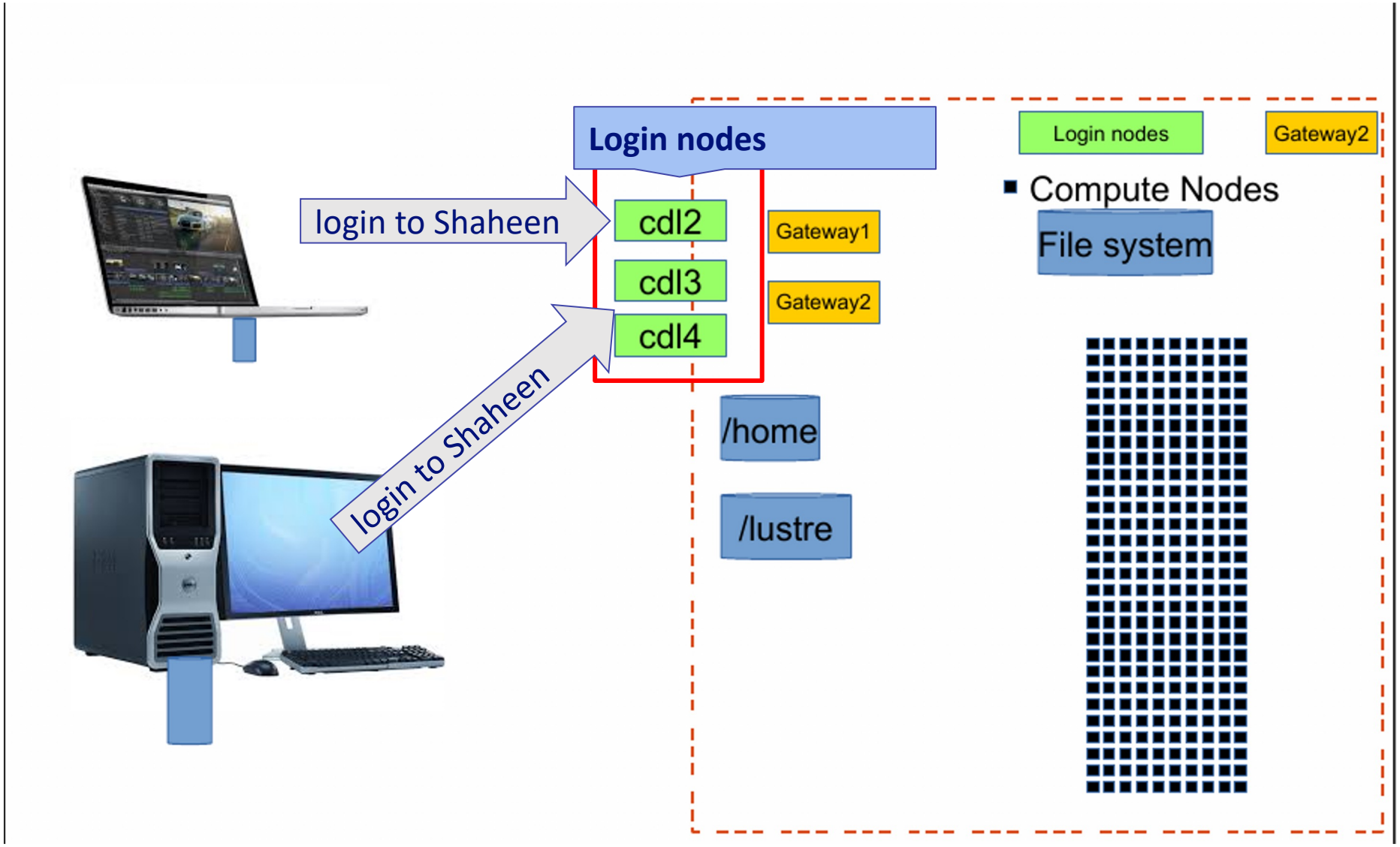
Shaheen's architecture



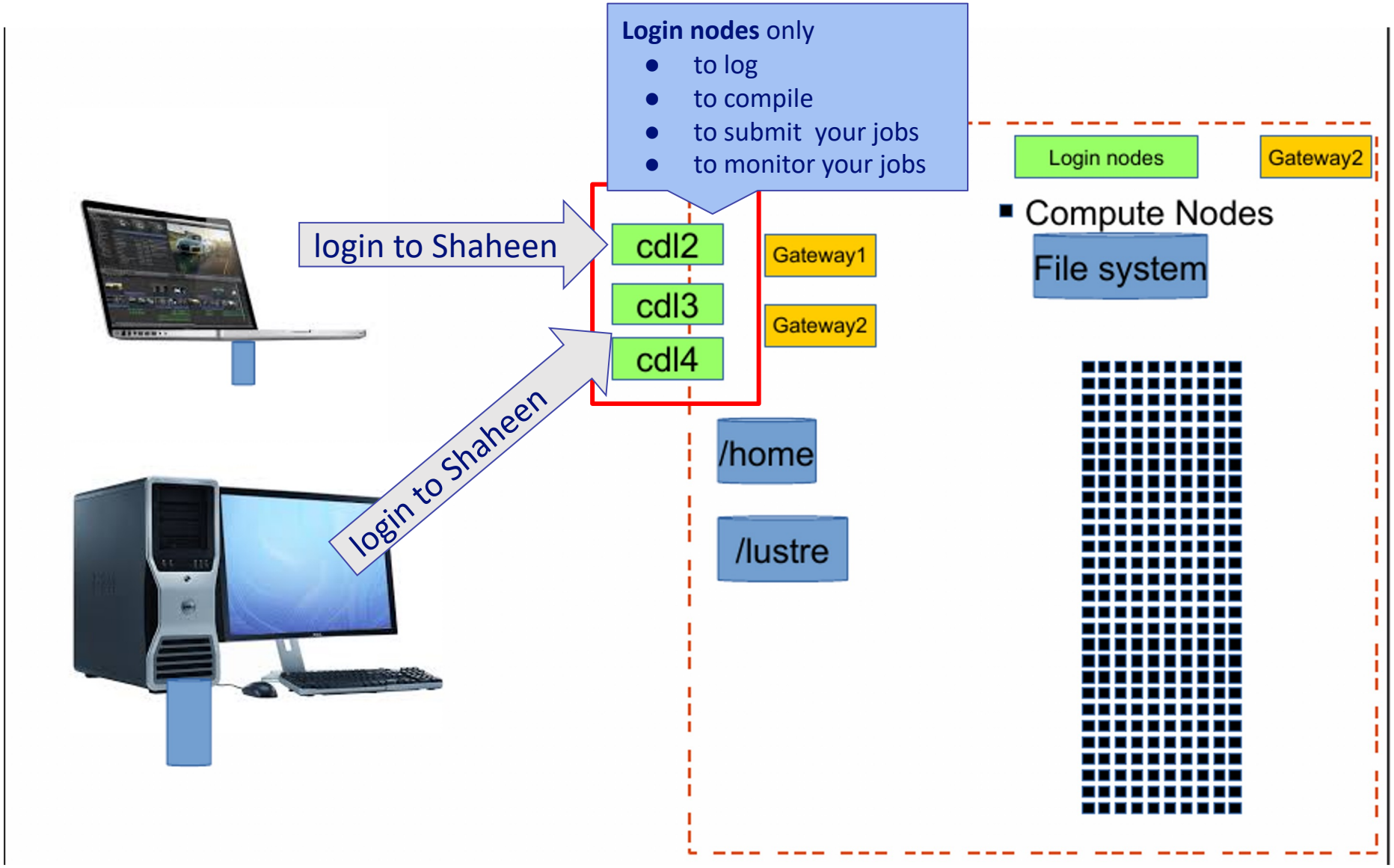
Shaheen's architecture



Shaheen's architecture

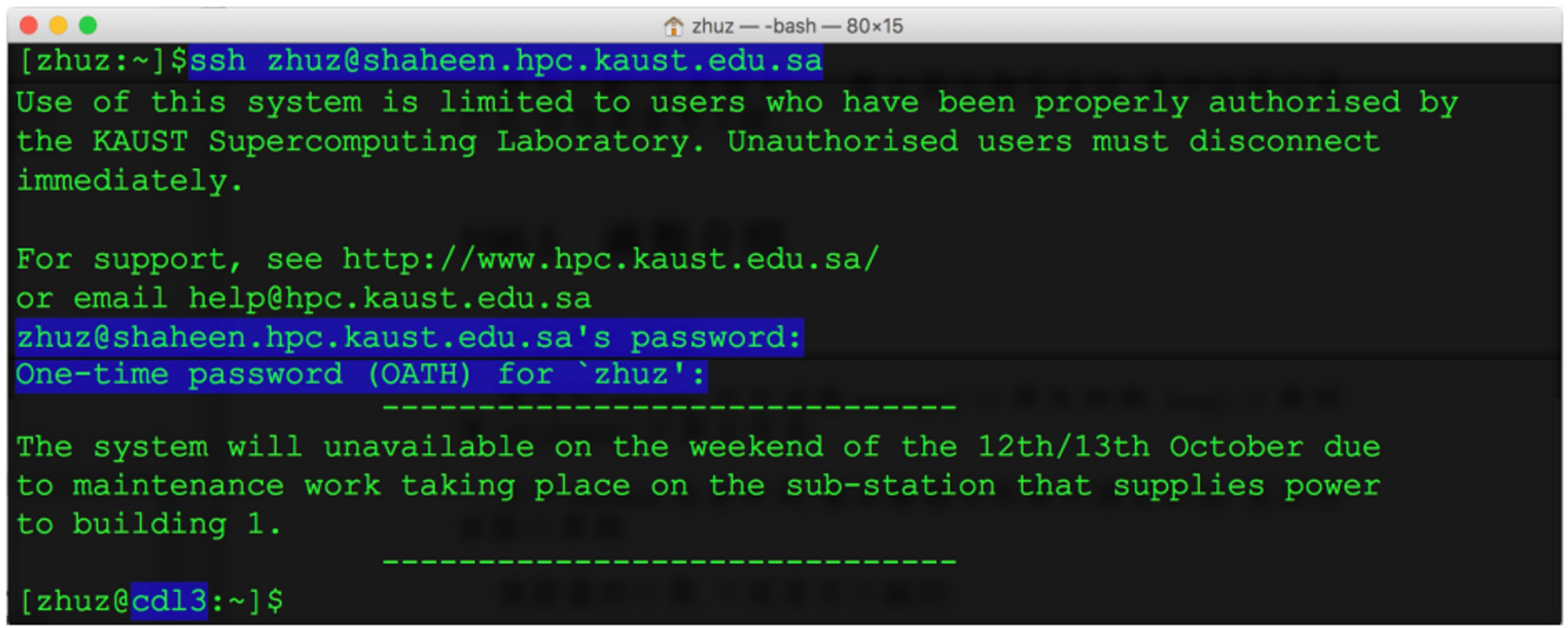


Shaheen's architecture

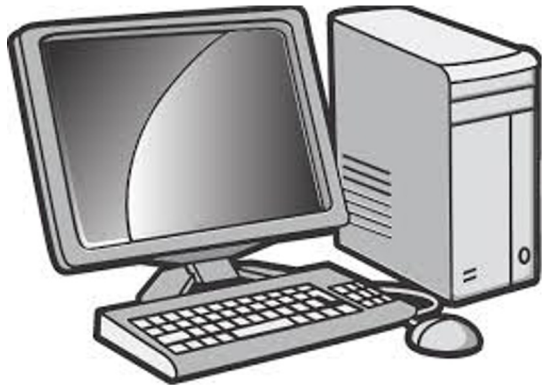


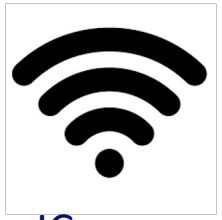
Login Shaheen

- Logins
  - ssh -X zhuz@shaheen.hpc.kaust.edu.sa

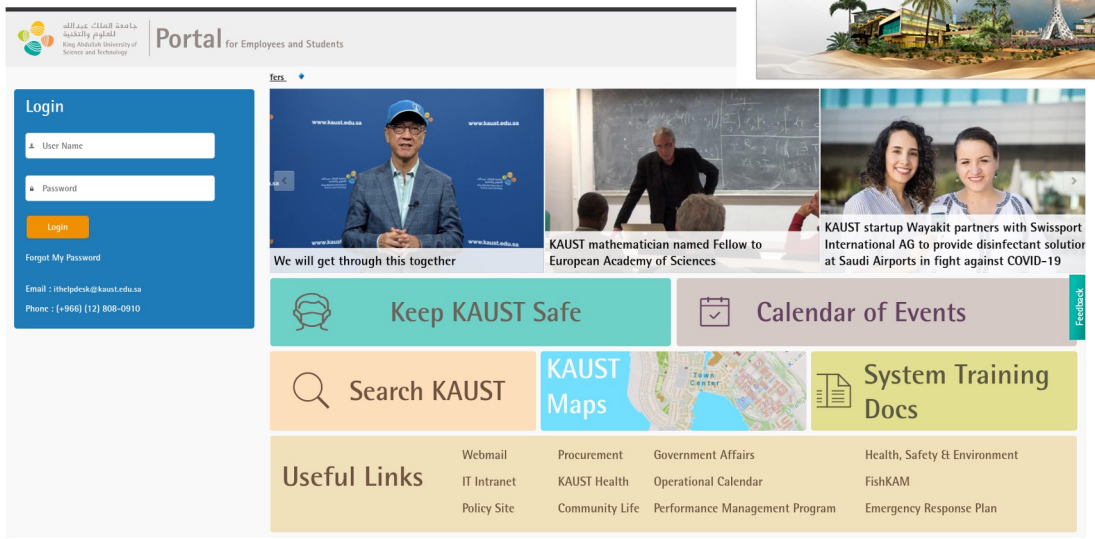


first password / two-factor authentication

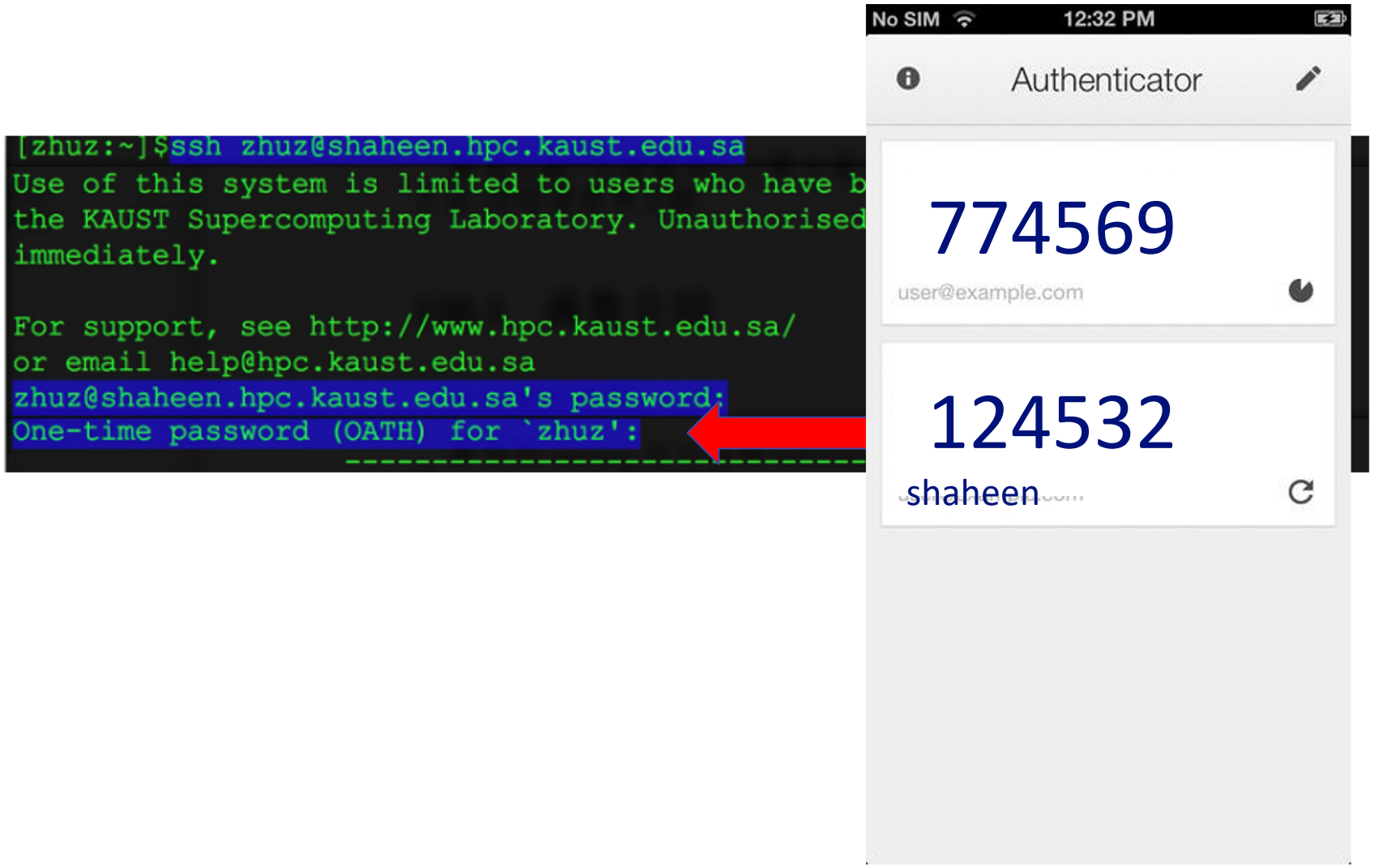




ICampus

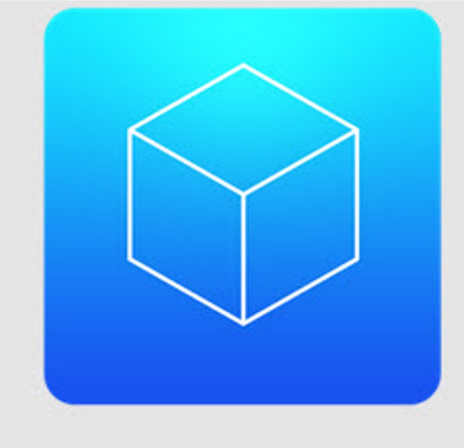


second password / two-factor authentication



Application running on smartphone

App installation and Initialization



FreeOTP

OTP Auth

Google Authenticator

Initial seed you get from us
Taken from your first login
at
http://hpc.kaust.edu.sa

Your password best practices

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Core Labs and
Research Infrastructure

Accessing Shaheen from Windows

Setting environment on a Windows machine

- Check https://www.hpc.kaust.edu.sa/windows for helpful instructions

- A video tutorial on installing bash terminal with ssh, scp inside follow

  https://www.youtube.com/watch?v=339AEqk9c-8

X11 forwarding from different OS

- X Windows allow you to open e.g. GUIs as windows on remote machines.
- X11 client needed on your workstation/laptop
- How to enable X11 on?
  - For windows : https://youtu.be/WUrj5V6mTNI
  - For Mac Osx : https://youtu.be/vzcBWcf9sm0
  - For Ubuntu: https://fabianlee.org/2018/10/14/ubuntu-x11-forwarding-to-view-gui-applications-running-on-server-hosts/
    Use **`ssh -X username@shaheen.hpc.kaust.edu.sa`**

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Core Labs and
Research Infrastructure

Job scheduling

On Shaheen

SLURM

- A resource manager
  - Manages more work than the resource by scheduling queues of work
- Supports complex scheduling of algorithms

- Provides:
  - Way to describing and submitting a resource request
  - Way to prescribing how to run workload on allocated resource
  - Way to monitoring the state of the submitted ”job”
  - Way to account for the resources used (charging system)

Six important commands

sinfo	Querying system resources and their state
squeue	Querying queue and status of jobs in it
sbatch	Submitting a batch job
salloc	Submitting resource request for an interactive allocation
scancel	Cancelling an allocation, batch or internactive
sacct	Query report of usage metrics of a job or jobs

Querying system resources

sinfo

A concise view of the system resources and their state/availability

```
> sinfo
PARTITION   AVAIL JOB_SIZE  TIMELIMIT   CPUS  S:C:T NODES STATE
workq*      up    1-6158   1-00:00:00     64 2:16:2    24 drained*
workq*      up    1-6158   1-00:00:00     64 2:16:2    46 down*
workq*      up    1-6158   1-00:00:00     64 2:16:2    36 drained
workq*      up    1-6158   1-00:00:00     64 2:16:2  2563 allocated
workq*      up    1-6158   1-00:00:00     64 2:16:2  3489 idle
72hours     up    1-512    3-00:00:00     64 2:16:2    24 drained*
72hours     up    1-512    3-00:00:00     64 2:16:2    46 down*
72hours     up    1-512    3-00:00:00     64 2:16:2    36 drained
72hours     up    1-512    3-00:00:00     64 2:16:2  2563 allocated
72hours     up    1-512    3-00:00:00     64 2:16:2  3485 idle
debug       up    1-4           30:00     64 2:16:2    16 idle
```

squeue

Shows the list of jobs in the queue along with information about the request and its current state

```
> squeue
JOBID       USER ACCOUNT           NAME  ST REASON    START_TIME                TIME  TIME_LEFT NODES
27438611    bob   k1335          -10kV  SO burst_buf 2022-09-07T12:58:48      24:28   23:35:32    65
27431642    john  k1415            SP   R None       2022-09-06T14:27:12   23:52:43      7:17     1
27431644    mary  k1415            SP   R None       2022-09-06T14:27:56   23:51:59      8:01     1
27431665    chris k1416    DNS_2D_1e13   R None       2022-09-06T14:36:34   23:43:21     16:39    64
```

You can use filters on the list:

- -u $USERNAME – show jobs by a user
- -p PARTITION – show jobs on a specific partition (workq, debug, 72hour)
- -j $JOBID -- show status of a particular job

Specifying resources

Requestable resources

- CPUs
- Memory
- Burst Buffer
- Wall time

Requesting resource

- You can either run your work as a batch job or interactive job:
- Implications:
- Batch job
  - You will need a script with resource request and the commands to run on that resource once allocated
  - Scheduler will run the script on your behalf once the requested resources are available
  - Resources can be requests for longer durations (several hours)
- Interactive job
  - Resource requests are usually small and short
  - You run each command by typing it interactively
  - Useful for prototyping and debugging

sbatch

- Command to submit your **jobscript** to SLURM:
- Upon successful submission a unique job ID is assigned
- Job is queued and awaits allocation of the requested resources
- On Shaheen where the usage is accounted for, a priority is assigned to each job based on first come basis
- In general, shorter and smaller jobs are easier to scheduler

JobScript (Shaheen)

```
--------- jobscirpt.slurm ------
#!/bin/bash
#SBATCH --account=prj01
#SBATCH --job-name=myfirstjob
#SBATCH --time=04:00:00
#SBATCH --partition=workq
#SBATCH --ntasks=64
#SBATCH --hint=nomultithread
#DW jobdw type=scratch access_mode=striped capacity=2TiB
module load vasp
srun --hint=nomultithread -n 64 vasp
------------------------------
```

```
> sbatch jobscript.slurm
```

salloc

Command to request allocation of resource for interactive use:

- Primarily be used for prototyping and/or debugging your workflow

```
shaima0d@cdl2:> salloc --partition=debug -N 2
salloc: Granted job allocation 12130189
shaima0d@gateway2:> srun -n 64 --hint=nomultithread ./helloworld

Hello from rank  0 of 64
Hello from rank  1 of 64
….
Hello from rank 63 of 64

shaima0d@gateway2:> exit
shaima0d@cdl2:>
```

scancel

- SLURM command to cancel a queued job:

```
> squeue -u bob
JOBID       USER  ACCOUNT   NAME    ST REASON    START_TIME  TIME  TIME_LEFT NODES
13723548    bob     k01     job.sh PD Priority  N/A         0:00      10:00     2
> scancel 13723548
> squeue -u bob
JOBID       USER  ACCOUNT   NAME    ST REASON    START_TIME  TIME  TIME_LEFT NODES
```

Using allocated resources

srun

- **Once allocated**, `srun` command can be used to launch your application on to the compute resources

```
> user123@cdl2:~> salloc --partition=debug -N 2
salloc: Granted job allocation 12140840
> export OMP_NUM_THREADS=4
> srun  --hint=nomultithread --ntasks=16 --cpus-per-task=4
./hello_world
```

- Each srun command is considered as "job step" for the corresponding allocation by SLURM
- When a job step completes, the allocation does not automatically terminate
- This means you can run multiple job steps with different configurations.

srun (in jobscript)

```
--------- jobscirpt.slurm ------
#!/bin/bash --login
#SBATCH --account=k01
#SBATCH --job-name=myfirstjob
#SBATCH --time=04:00:00
#SBATCH --partition=workq
#SBATCH --ntasks=32

srun -n 1 ./serial_step          # quick but sequential preprocessing step
srun -n 8 -c 4 ./parallel_step   # computationally expensive/parallel step

----------------------------
```

Monitoring and accounting

Of your jobs

sacct

Displays accounting command which tell about the resources used by the job and its job steps.

```
> sacct -j 12130040
       JobID    JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
12130040     profiling+      workq        k01         64  COMPLETED      0:0
12130040.ba+      batch                   k01         64  COMPLETED      0:0
12130040.0       Rscript                  k01         12  COMPLETED      0:0
```

The accounting information persists after the life of the job

Common filters include `-u` for "by user" and `-j` for "by jobID"

squeue

- You can query the state of your job using squeue
- Common filters include
  - by user
  - by job ID

```
> squeue -u bob
             JOBID    USER ACCOUNT NAME           ST REASON    START_TIME                 TIME  TIME_LEFT NODES
          12127214     bob   k1320 IMe-H-RuCl3-fr   R None      2019-12-08T20:05:20   21:13:44    2:46:16     1
          12128314     bob   k1320 IMe-H-RuCl3-fr   R None      2019-12-09T08:32:51    8:46:13   15:13:47     1
          12129938     bob   k1320 IMe-H-RuCl3-fr  PD AssocGrpC N/A                       0:00 1-00:00:00     1
          12129939     bob   k1320 IMe-H-RuCl3-fr  PD AssocGrpC N/A                       0:00 1-00:00:00     1

> squeue -j 12127214
             JOBID    USER ACCOUNT NAME           ST REASON    START_TIME                 TIME  TIME_LEFT NODES
          12127214     bob   k1320 IMe-H-RuCl3-fr   R None      2019-12-08T20:05:20   21:14:44    2:45:16     1
```

scontrol

scontrol command, amongst other things, allows user to show parameters of request and allocated resource for a job in queue (in any state i.e running, pending, etc)

```
> scontrol show job 12130305
   JobId=12130305 JobName=profiling_jags
   UserId=bob(123456) GroupId=g-bob(123456) MCS_label=N/A
   Priority=1 Nice=0 Account=k01 QOS=normal
   JobState=RUNNING Reason=None Dependency=(null)
   Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
   RunTime=00:55:25 TimeLimit=10:00:00 TimeMin=N/A
   SubmitTime=2019-12-09T15:04:12 EligibleTime=2019-12-09T15:04:12
   AccrueTime=2019-12-09T15:04:12
   StartTime=2019-12-09T15:04:12 EndTime=2019-12-10T01:04:12 Deadline=N/A
   SuspendTime=None SecsPreSuspend=0 LastSchedEval=2019-12-09T15:04:12
   Partition=workq AllocNode:Sid=cdl2:38945
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=nid00107
   BatchHost=nid00107
   NumNodes=1 NumCPUs=64 NumTasks=1 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
   TRES=cpu=64,mem=128448M,node=1,billing=64
```

Example Jobscripts

Example – OpenMP jobs on Shaheen

- A jobscript running an OpenMP code on a single Shaheen node with 4 OpenMP threads

```
#!/bin/bash
#SBATCH --account=k01
#SBATCH --time=00:10:00
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=4
#SBATCH --hint=nomultithread

export OMP_NUM_THREADS=4
srun --hint=nomultithread -n 1 -c 4 ./my_omp_application
```

Example – MPI jobs on Shaheen

- A jobscript running MPI code on a Shaheen with 32 MPI tasks

```
#!/bin/bash -l
#SBATCH --account=k01
#SBATCH --time=00:10:00
#SBATCH --ntasks=32
#SBATCH --hint=nomultithread

# load your modules here …
srun --hint=nomultithread -n 32 ./my_mpi_application
```

Example – MPI+OpenMP jobs on Shaheen

- A jobscript running a hybrid code on Shaheen parallelized with both OpenMP and MPI (requires 32 x 4 = 128 core in total)

```
#!/bin/bash
#SBATCH --account=k01
#SBATCH --time=00:10:00
#SBATCH --ntasks=32
#SBATCH --cpus-per-task=4
#SBATCH --hint=nomultithread

export OMP_NUM_THREADS=4
srun --hint=nomultithread -n 32 -c 4 ./my_hybrid_application
```