Introduction to HPC-Cluster

Training Cluster HLRS

TASC (HLRS)

Training Cluster HLRS Authentication

- **Login– Linux, Mac**
  - Default path to **private key**: `$ /~/.ssh/id_ed25519`
  - If your **private key** has a **different path**, you need to specify the path when you login using `-i`
  - Login with **X11 forwarding** (if required):

```
$ ssh -X username@training.hlrs.de -i /path/to/private/key
```

  - Advanced option: Use a configuration file (**Client** uses "alias"):

```
$ /~/.ssh/config
$ ssh trainingscluster

Host trainingscluster
        HostName training.hlrs.de
        User username
        ForwardX11 yes
```

Your access
username: **sca509xx**

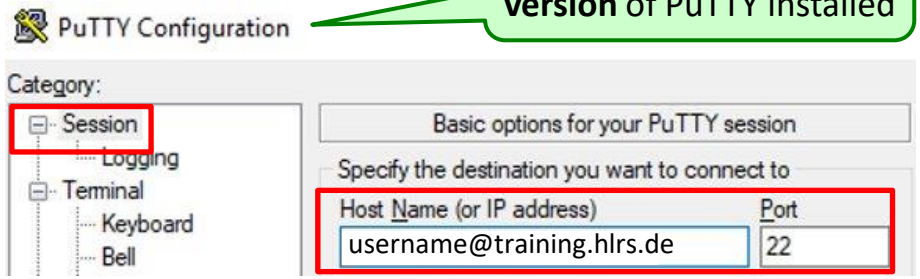
Training Cluster HLRS Authentication

Please have **the current version** of PuTTY installed

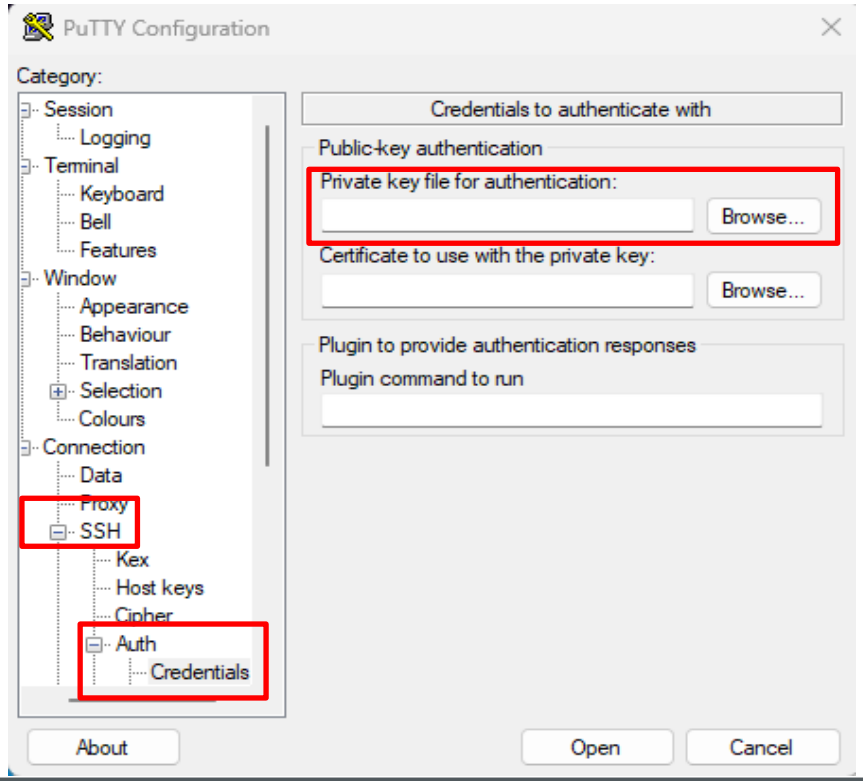
- **Login in Windows (GUI) - PuTTY**
  - Host name:

```
username@training.hlrs.de
```

Your access
username: **sca509xx**



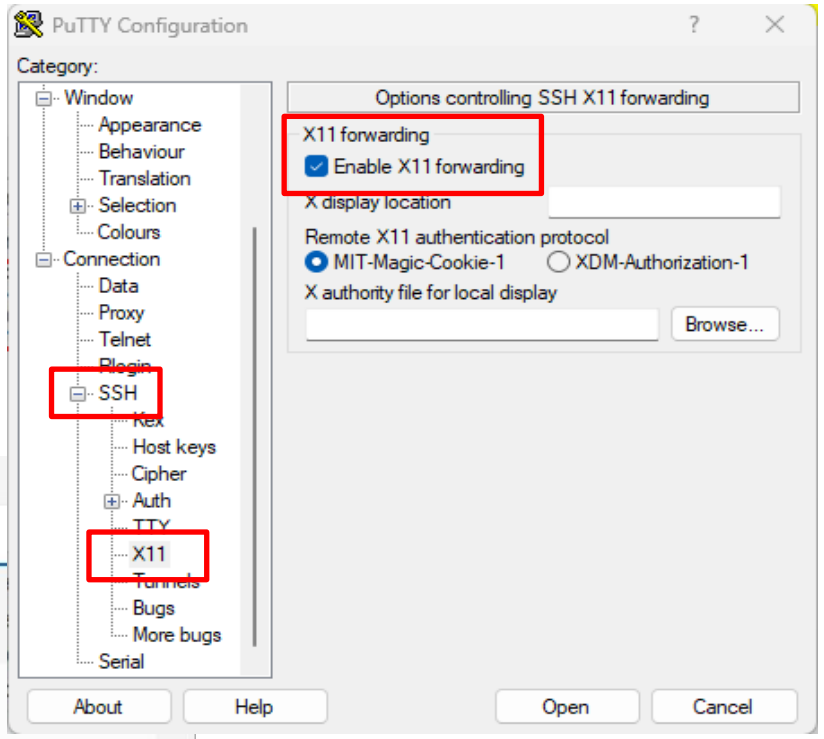
- Provide the full path to the **private key** file **(.ppk)**
  - If needed, **use PuTTY Gen** to convert your private key into a **.ppk** one

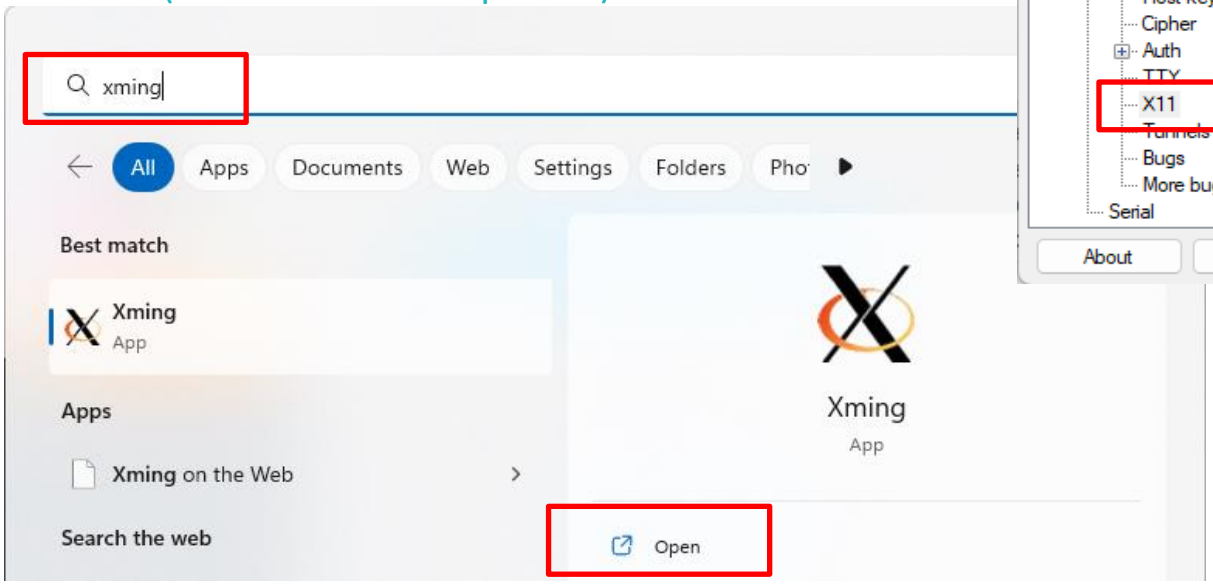


Training Cluster HLRS Authentication

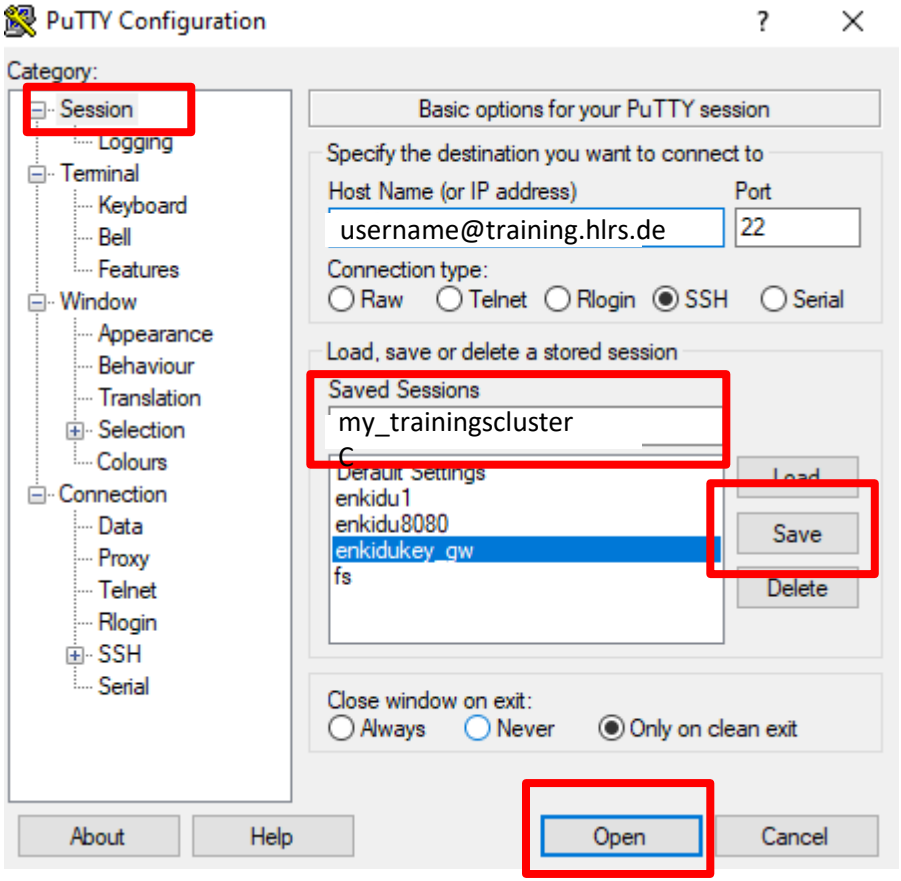
- Select “**Enable X11 forwarding**”

- Launch **Xming**
  - Download and install it if not already available (e.g. from here “Public Domain Releases”)
  - A small icon in the task bar should appear (no other event expected)





Training Cluster HLRS Authentication



- Enter a name in "**Saved Sessions**"
- **Save** for the next logins
- Click on **Open** to start the session

- You can use **Load** to edit the session

Only for X11-forwarding:

- Do not forget to **launch Xming before** clicking on **Open**!
- **Test** successful X11 e.g. by typing
  `xterm`
  after login. A terminal should pop up.

Training Cluster Data Sheet

- Data Server:
  - 16x 2TB disks HDD
  - Capacity usable: 16TB (since all RAID10)
  - Connection: NFSv4 -> TCP via Infiniband
  - QDR (40Gbit)
- 1 Chassis (4 Nodes):
  - 1x Frontend (Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz, SandyBridge EP, 16 Cores, 32GB RAM, 20 MB Intel® Smart Cache, 2 QPI-Links, Lithographie 32 nm)
  - 1x Clustermanagement
  - 1x Compute Node smp, Intel(R) Xeon(R) CPI E5-2670 0 @ 2.60GHz, SandyBridge EP, 16 Cores, 32GB RAM, 20 MB Intel® Smart Cache, 2 QPI-Links, Lithographie 32 nm
  - 1x NFS-Server
- 1 Chassis Graphics Server:
  - 1x Vis-Node
  - Intel(R) Xeon(R) CPU E5540 @ 2.53GHz, Nehalem EP, 8 Cores, 48 GB RAM, 8 MB Intel® Smart Cache, 2 QPI-Links, Lithographie 45 nm
  - Graphics Card AMD Radeon Vega56 with 8GB graphic mem *
  - smd (shared access)

Training Cluster Data Sheet

- 4 Chassis à 4 Nodes:
  - **16 Compute Nodes**
  - each **2x** Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz Skylake EP, **20 Cores**, **192 GB RAM**, 27,5 MB L3 Cache, 3 UPI-Links,Lithographie 14 nm
  - smd (shared access)
- Hardware-Interface:
  - InfiniBand (IB)
  - FDR
- Network Connection
  - 1 Gbit to University network
  - 1 Gbit cluster internal
- Operating System: Rocky Linux 8.5
- Batch-System: PBSPro
- Home-Directory: Quota 2GB, 10.000 Files (default, is course dependent)
- NFS Storage with Workspace-Mechanism: **Quota: 20GB, 200.000 files** (default, is course dependent)
- Similar to the NEC-Cluster Vulcan

https://kb.hlrs.de/platforms/index.php/Vulcan

Training Cluster Data Sheet

- Storage of your data
  - Name, Login Name, E-Mail Address, PubKey saved in the LDAP-Server
  - The account will be deactivated 7 days after course end
  - Final disposal after 2 years

Training Cluster HLRS

- Hostname
  - The main hostname required to connect to training cluster is **training.hlrs.de**
  - The system has **one login node**
- Usernames
  - Your username is composed of the prefix sca and your user-id, for instance sca50949
- **Only the secure shell SSH is allowed to login.** Other protocols like telnet or rlogin are not allowed for security reasons.

File Systems

- Usally there are 4 different spaces on a cluster
  - $HOME
    - Personal space
  - $PROJECT or $SHARED
    - Shared space with Colleagues/Project-Partners
  - $TMP
    - Temporary space on nodes for calculations
  - $WORKSPACE
    - The workspace mechanism provides temporary scratch space, called workspaces, for your computation on a central file store.
    - They are intended to hold data **for a limited time** - but usually longer than the time of a single job run.
    - They are not intended for permanent storage, so **data in workspaces is not backed up** and may be lost if there are problems on the storage system.
    - Please copy/move important results to $HOME or to some disks outside the cluster.

Exercise Workspace & Quota

- Commands for Workspace

```
$ ws_allocate my_workspace 1
$ ws_list
$ ws_extend my_workspace 2
$ ws_release my_workspace
$ cd $(ws_find my_workspace)
```

- Quota Limitation
  - Home-Directory

    ```
    $ quota -us $(whoami) $HOME
    ```

  - Workspace

    ```
    $ quota -us $(whoami) /shared/training/<your-workspace-name>
    ```

Workspace - Summary

- For compute jobs
  - **do not use your home directory**
  - but **create a workspace**
- Workspace mechanism
  - allows to store data outside the home directory for several days
  - reserves storage space for a certain number of days
  - workspace uniquely identified by name
  - **After the duration expires, the data is deleted and thus lost!**

Enviroment Module System

- Environment modules, or modules for short, are the way by which most of the installed software is deployed on a cluster.
- The use of various compilers (and multiple versions), libraries and software packages requires the user to set up a specific session environment suitable for the program to be executed.
- The cluster provides users with the ability to load and unload complete environments for compilers, libraries and software packages with a single command.
- The Training cluster uses the Lua based module system **Lmod**.

https://lmod.readthedocs.io/en/latest/

Exercise Enviroment Modules

- Load Module
  ```
  $ module load compiler/gcc # default version
  $ module load compiler/gcc/9.2.0 # specific version
  ```

- Show all loaded modules
  ```
  $ module list
  ```
- Show all available modules
  ```
  $ module avail
  ```
- Show all possible modules and dependencies
  ```
  $ module spider
  ```
- Show all available versions of the module
  ```
  $ module spider compiler/gcc
  ```
- Outputting the help text of a module
  ```
  $ module help compiler/gcc
  ```
- Output all "whatis" information about a module
  ```
  $ module whatis compiler/gcc
  ```

- Show all environment variables and paths of a module
  ```
  $ module show compiler/gcc
  ```
  (=shows the commands in the module file)
- Unload module
  ```
  $ module unload compiler/gcc
  $ module del compiler/gcc
  ```
- Unload all modules
  ```
  $ module purge
  ```

Batch System - Summary

- Jobs need to be queued and then executed on time and that is what the batch system does.
- The batch system is responsible for the distribution of all resources in the cluster
  - **working time**
  - **nodes and cores**
  - **memory**
  - **diskspace**

  You can request any of these resources.

- The **training cluster** uses the batch system **PBS Pro**
- The components of the Management System – PBS Pro are
  - **Resource Manager**
    - Control over jobs and distributed nodes
  - **Scheduler**
    - Scheduling
    - Managing
    - Monitoring
    - Reporting

EXT https://www.altair.de/pbs-works-documentation/

Submit Job on Training Cluster HLRS

- If you want to run a your code on a cluster, then you need to submit it as a job.
- Batch jobs are submitted by using the command **qsub**.
- The main purpose of the qsub command is to specify the resources that are needed to run the job.
- qsub will then queue the batch job.
- However, starting a batch job depends on the availability of the requested resources.

Exercise Submit / View Job

Specific for each course!

- Submit a batch job, e.g.

```
$ qsub -I -l select=1:node_type=skl:ncpus=8:mpiprocs=8:mem=14gb,walltime=00:15:00 -q smp
```

Do not forget!

- If you want to view information about submitted jobs, use the command squeue
  - View job information

    ```
    $ qstat
    ```

  - View job information of specific user

    ```
    $ qstat –u <username>
    ```

  - View jobs in queue

    ```
    $ qstat –q <Queue-Name>
    ```

- If you want to cancel submitted jobs, use the command qdel
  - Cancel specific job

    ```
    $ qdel <job-id>
    ```

Example batch script

```
#!/bin/bash
sleep 120;
```

- See:
  https://kb.hlrs.de/platforms/index.php/Batch_System_PBSPro_(vulcan)