Advanced Topics: Usage of bwHPC & HoreKa Clusters

Peter Weisbrod, SCC, KIT

www.bwhpc.de / www.nhr.kit.edu
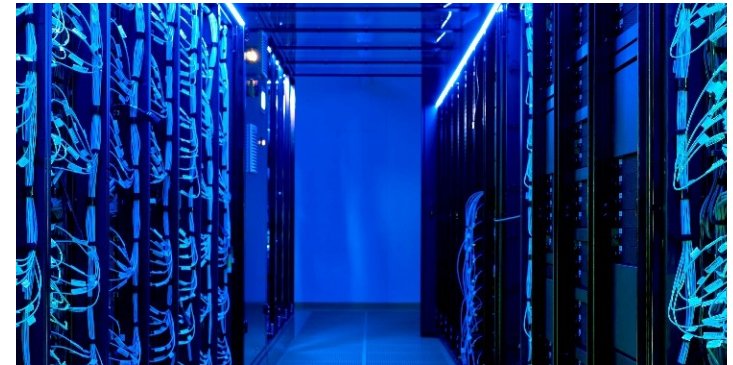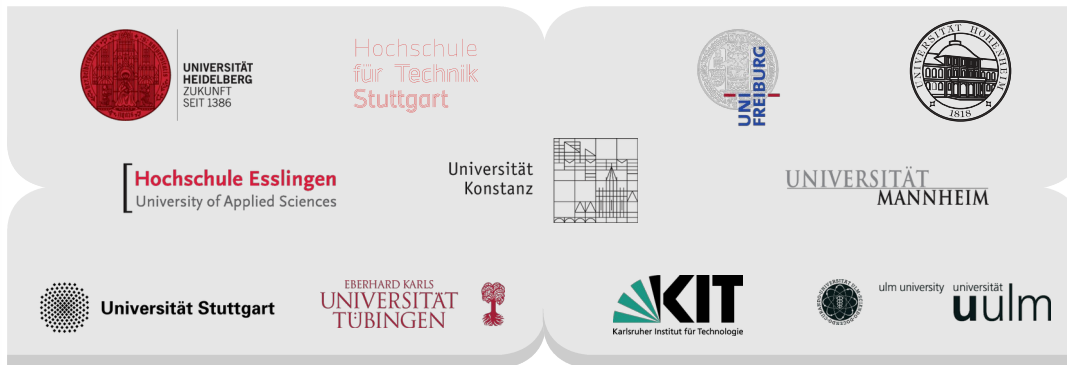
Outline

- Access and Data Transfer
  - Registration + Access
  - SSH Keys
  - Remote Visualization
  - Best Practice: Data Sharing
  - Data Transfer
  - Visual Studio Code remote development
  - WSL2
- Filesystem Topics
  - On-demand File System
- Software Topics
  - Best practice: installing own software
  - Best practice: compiling code
  - Using Containers
- Questions from participants

Reference: bwHPC + NHR Best Practices Repository

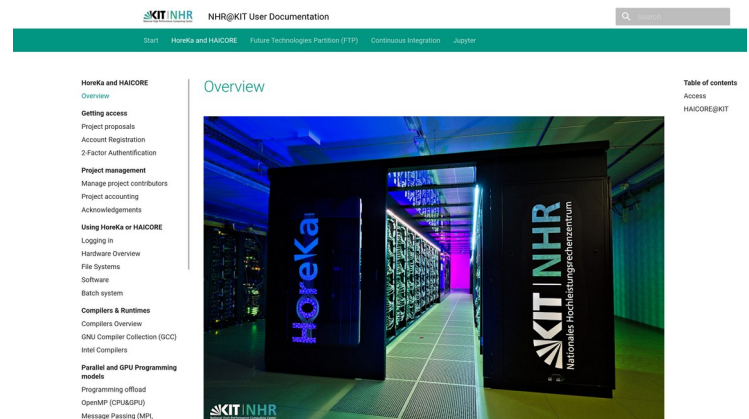- Most information given by this talk can be found at

  - bwHPC Wiki:
    https://wiki.bwhpc.de/e/Main_Page

  - NHR@KIT Wiki:
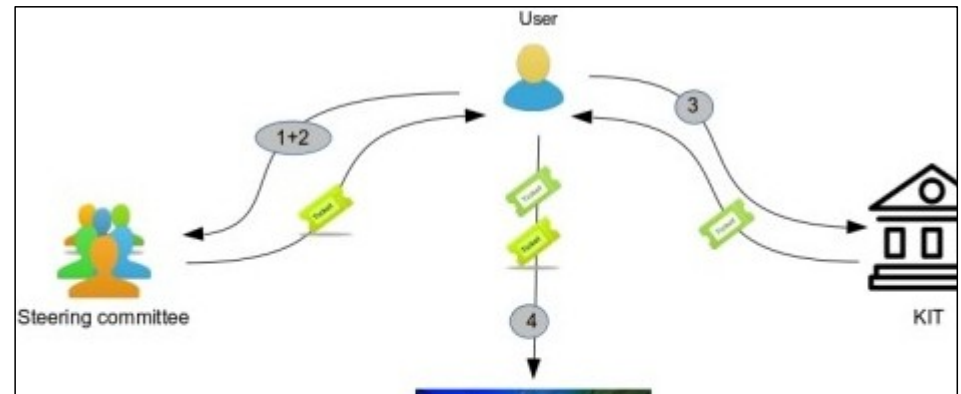   https://www.nhr.kit.edu/userdocs
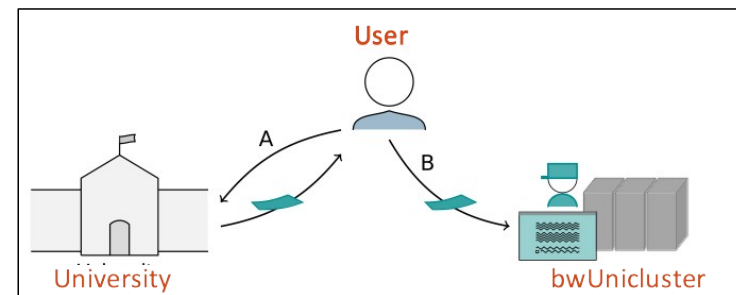
Access and Data transfer

Registration Processes per Cluster

HoreKa:

1. Online Proposal Form (Jards)
2. Peer reviewed proposal
3. HoreKa access form
4. Register on web page
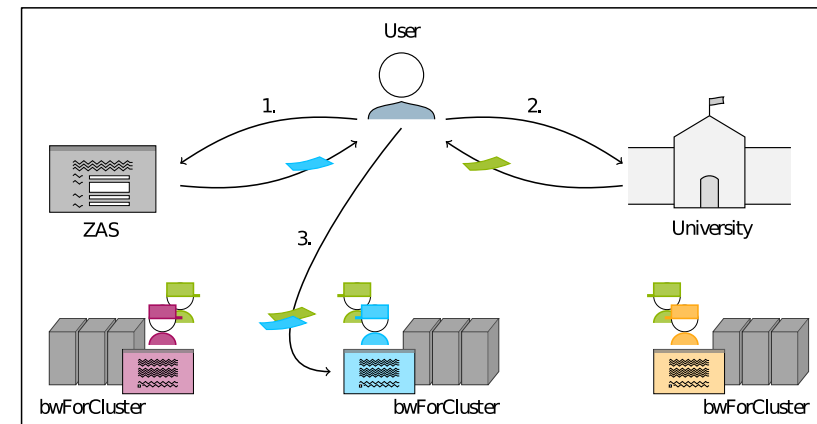   https://fels.scc.kit.edu

bwUniCluster 2.0:

Step A: Get bwUniCluster entitlement

Step B: Web registration + questionnaire

bwForClusters

Step 1: Registration at ZAS
         Approval by CAT

Step 2: Get bwForCluster entitlement

Step 3: Web registration at bwForCluster site

SSH Keys

- **It is not possible to self-manage your SSH Keys by adding them to the ~/.ssh/authorized_keys file**
- Deposit and Register Keys
  - UC2: https://bwidm.scc.kit.edu/
  - HoreKa: https://fels.scc.kit.edu/
- Types of Keys:
  - *Interactive*, for login
  - *Command*, for automation
- Use the Key
  - `ssh/scp -i … / rsync -e "ssh -i $HOME/.ssh/somekey" …`
  - Or simply define it here `~/.ssh/config`
- Documentation:
  - https://wiki.bwhpc.de/e/BwUniCluster_2.0_User_Access/SSH_Keys
  - https://www.nhr.kit.edu/userdocs/common/sshkeys/

Remote Visualization (1)

- The Linux 3D graphics stack is based on X11 and OpenGL. This has some drawbacks in conjunction with remote visualization

    - Rendering takes place on the client, not the cluster
    - Whole 3D model must be transferred via network to the client
    - Many round trips in the X11 protocol negatively influence interactivity
    - X11 is not available on non-Linux platforms
    - Compatibility problems between client and cluster can occur

- To avoid all those problems the module `start_vnc_desktop` is provided on bwUniCluster and HoreKa for remote visualization.
  More details at bwHPC wiki
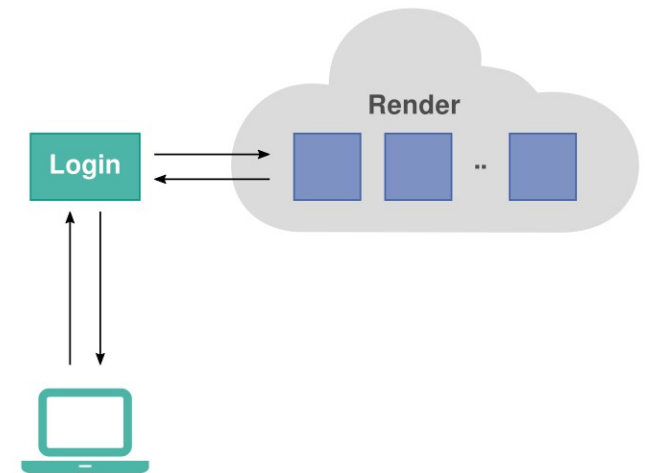
Remote Visualization (2)


```
hk:~$ start_vnc_desktop --hw-rendering
```

```
* Allocating 76 processors per node (Use command line option --ppn to change)
* Allocating 4 GPUs per node (Use command line option --num-gpu to change)
* Using queue 'accelerated' to allow hardware rendering
* Using wall clock limit 02:00:00 (Use command line option --walltime to change)
Allocating resources to launch a VNC desktop:
Launching the VNC desktop on the resources granted by the batch system.
salloc: Granted job allocation 1498539
salloc: Waiting for resource configuration
salloc: Nodes hkn0802 are ready for job

Desktop 'TurboVNC: hkn0802.localdomain:1 (ej4555)' started on display hkn0802.localdomain:1

Starting applications specified in vglrun -d :0.0 -c proxy +wm /usr/bin/gnome-session
Log file is /home/hk-project-scs/ej4555/.vnc/hkn0802.localdomain:1.log

Hint for TurboVNC Viewer users (command line):
=============================================
  vncviewer ExtSSH=1 Via=ej4555@hk.scc.kit.edu Server=hkn0802:1 Password=olbmW1+9 SecurityTypes=VNC

Hint for TurboVNC Viewer users (GUI)
=============================================
  Fill in the following entry field:
  VNC server:    hkn0802:5901

  Click "Options" and choose tab "Security".
  Fill in the following entry fields:

  Gateway (SSH server or UltraVNC repeater)
```
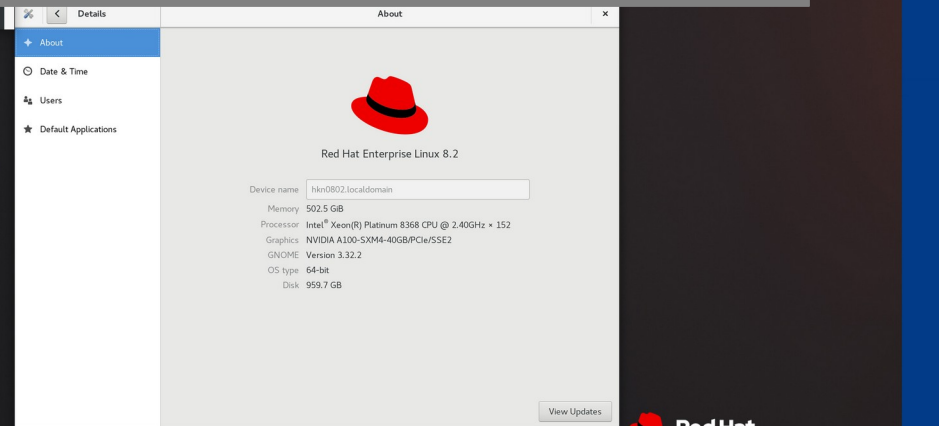
```
local:~$ /opt/TurboVNC/bin/vncviewer
```

Best Practice: Data Sharing

- How to share data with another person on the same cluster?

```
$ ws_allocate sharing 30
...
$ ls -ld $(ws_find sharing)
drwx------ 2 ab1234 xyz 4096 Oct  9 00:42 /pfs/work7/workspace/scratch/ab1234-sharing-0
```

→ workspace is private!

- Allow full permission for user xy*3456*

```
$ setfacl -Rm u:xy3456:rwX,d:u:xy3456:rwX,other::-,group::-,d:other::-,d:group::- $(ws_find sharing)
```

→ Access Control Lists (ACLs) are used

→ Default ACLs define permissions for newly created files and directories

- Further information and examples:

→ https://wiki.bwhpc.de/e/Workspace#Sharing_Workspace_Data_within_your_Workgroup

Best Practice: Data Sharing (2)

- Module for easily setting ACLs: `ws_share`

```
$ module load system/ws_addon
$ ws_share

 USAGE     : ws_share [-options <argument>] <workspace_1> ... <workspace_N>

 -h, --help              Print this message
 -v, --verbose           Verbose printout
 -n, --dryrun            Do a dry run

 -u, --user=<usernames>  Comma separated list of users
                         that given workspace(s) has to be shared with
 -g, --group=<groupnames> Comma separated list of groups
                         that given workspace(s) has to be shared with

 -t, --sharetype=<option> Type of sharing workspaces with following options:
                         All added permissions will be NEVER MORE then owner's permission on files/directory in workspace(s)!
                             default  = All directories can be accessed (r-x) but new user/group can not add files
                             dir-w    = puts write permissons on all directories (rwx) if owner's directory has write permissions
                                        while permission of owner's files are unaltered
                                        and new user/group can add files
                             replicate = replicate owner's permissions on dirs/files.
                                        Depending on owner's dir permissions new user/group can add files

 -s, --status            Print sharing (.i.e., ACL) permissions
 -r, --revoke-all        Revoke recursively all sharing (.i.e., ACL) permissions
```
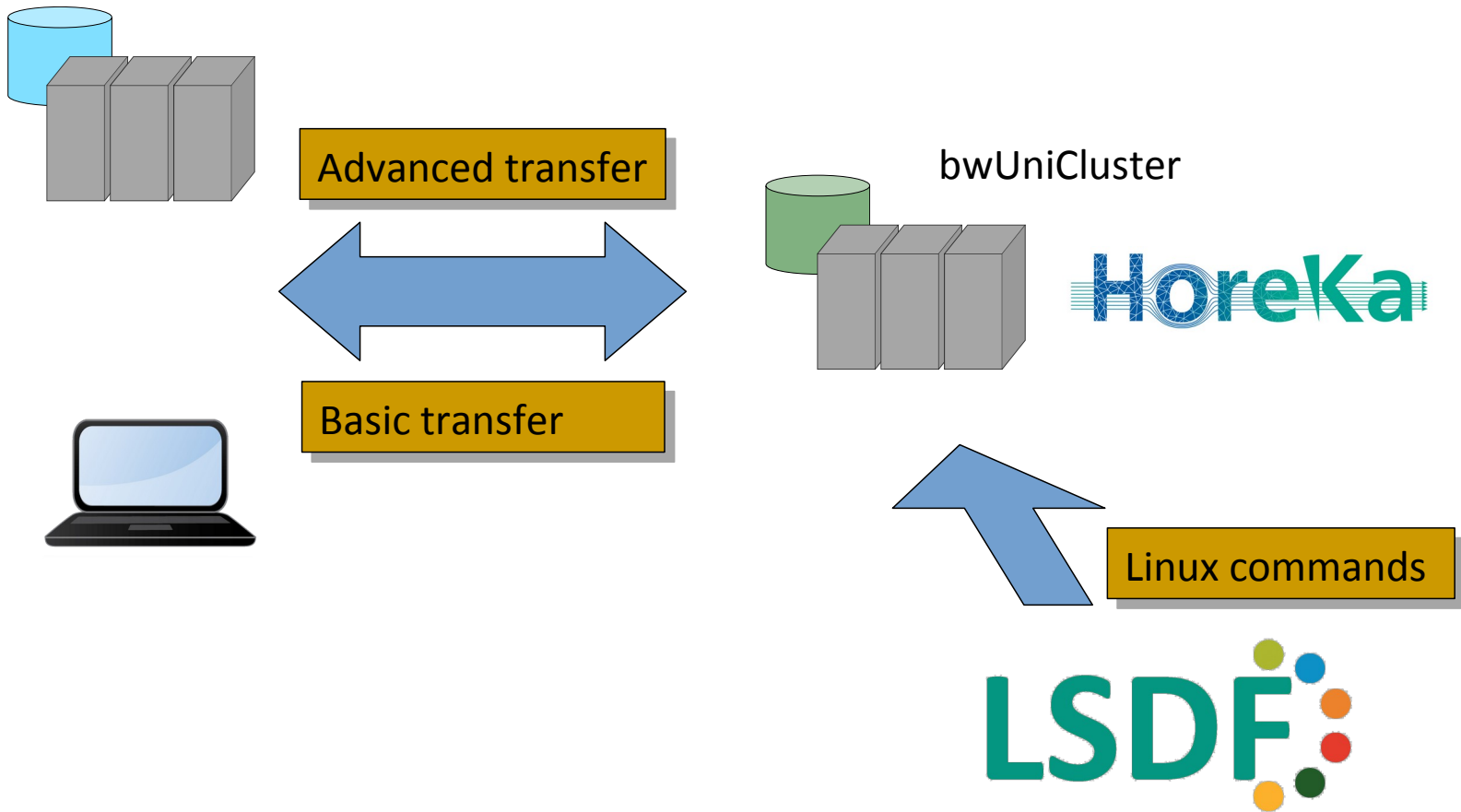
Data Transfer bwUniCluster and Horeka

Advanced transfer

Basic transfer

bwUniCluster

HoreKa

Linux commands

LSDF

Basic File Transfer – Linux und Windows

- **`scp`** = OpenSSH secure file copy (Linux)

```
Push: $ scp [options] SRC [USER@]HOST:DEST
Pull:  $ scp [options]     [USER@]HOST:SRC  [DEST]
```

- **`rsync`** = fast file-copying tool (Linux)
  - superior to scp, sending only the differences between the source files and the existing files at the destination

```
Push: $ rsync [options] SRC [USER@]HOST:DEST
Pull:  $ rsync [options]     [USER@]HOST:SRC  [DEST]
```

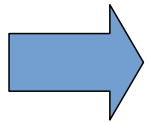- **MobaXterm + MS File Explorer** (Windows)
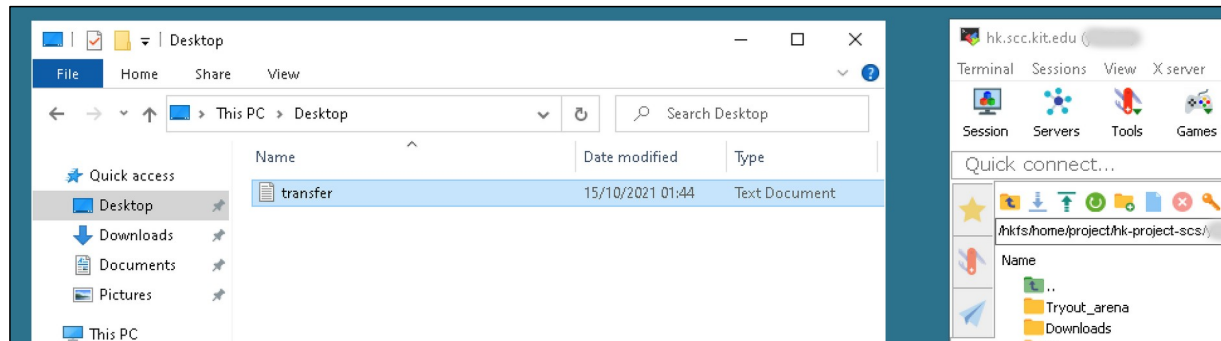
Advanced File Transfer with SSH Key

- **Interactive key**
  - Valid for one hour after 2FA login, limited lifetime
  - Useful for small amount of data

- **Command key**
  - Valid w/o 2FA login, limited lifetime
  - Restriction to a single command
  - For transfer with rsync register the following command:
    ```
    /usr/bin/rrsync -ro / -rw /
    ```

    ➡️ https://wiki.bwhpc.de/e/BwUniCluster_2.0_User_Access/SSH_Keys

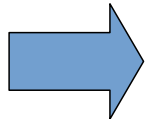# Advanced File Transfer via Interactive Job

- **Login + tmux**
  - Do the transfer in the background of your login session

```
$ tmux
$ scp [options] SRC [USER@]HOST:DEST
```

- **Interactive job**
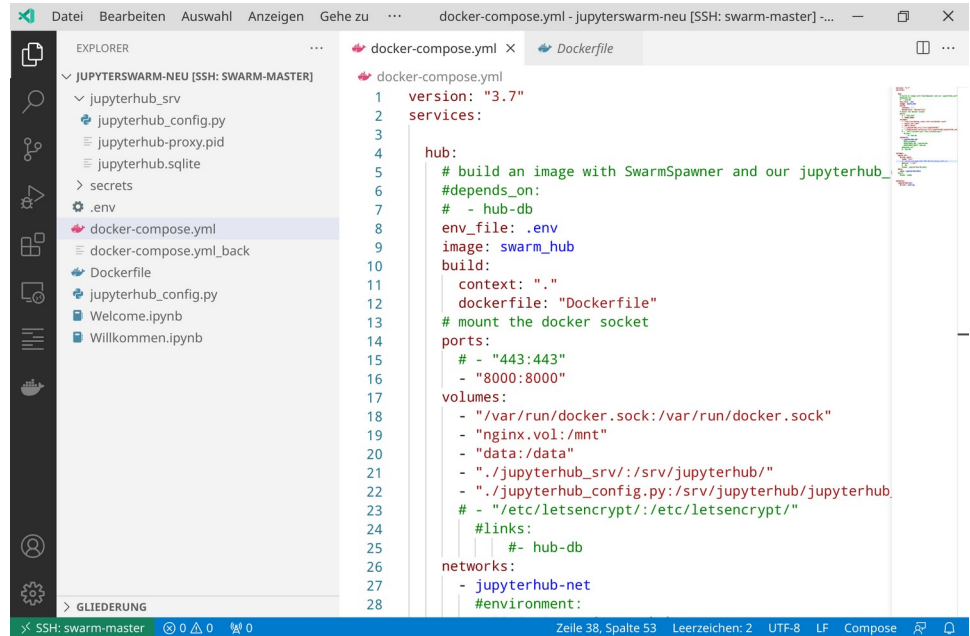 - Use an interactive job for data transfer

```
$ salloc –p single –n 1 –t 120 –mem=5000
```

https://wiki.bwhpc.de/e/BwUniCluster_2.0_Batch_Queues

Visual Studio Code – Remote SSH

- IDE, code editor
- VS Code has become very popular
- Windows, macOS and Linux
- Extension: Remote SSH
  - SSH to HPC systems
  - Connection to local WSL2
  - Supports 2FA, respects `~/.ssh/config`
- Documentation:
  - https://code.visualstudio.com/docs/remote/remote-overview
  - Remote SSH

Visual Studio Code – code-server

- Server-side component of VS Code
- Access VS Code in web browser
- Development and Debugging on <u>compute nodes</u>!

Usage:

- Start code-server
  ```
  module load devel/code-server
  code-server --bind-addr 0.0.0.0:8081
  ```

- SSH-Tunnel
  ```
  ssh -L 8081:<uc2nXXX>:8081 \
  ab1234@uc2.scc.kit.edu
  ```

- Open web browser at http://127.0.0.1:8081

- Documentation:
  - code-server
  - https://github.com/cdr/code-server/tree/main/docs
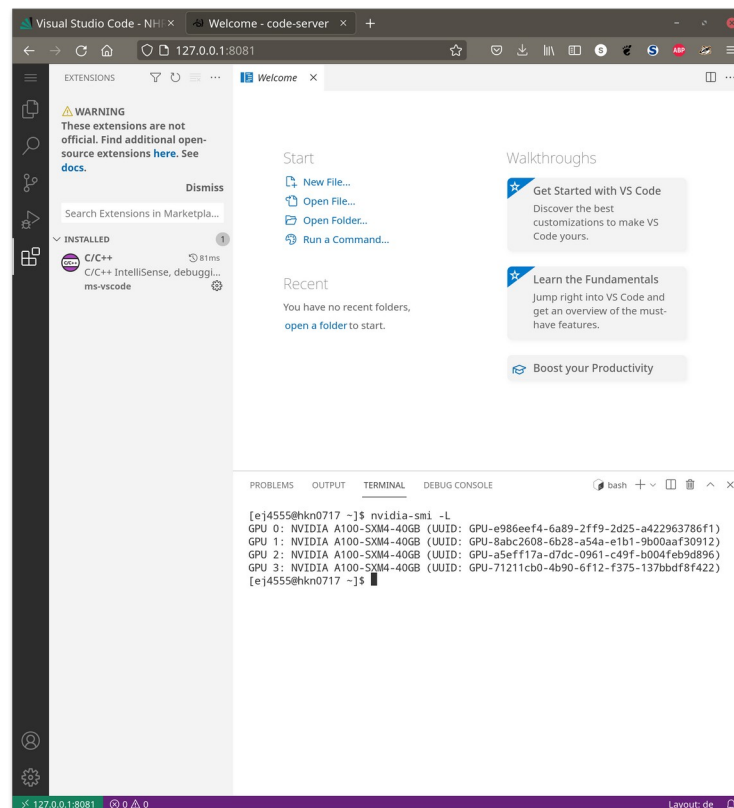
Windows Subsystem for Linux (WSL2)

- Linux environment directly on Windows

- Different distros

- Very well integrated VM

  - Access to file systems

  - Windows Terminal

  - GUIs (sic!)

- **All** Linux tools at hand:

  - `ssh`

  - `rsync`

  - `...`

- Documentation:
  https://docs.microsoft.com/de-de/windows/wsl/

https://docs.microsoft.com/de-de/windows/wsl/tutorials/gui-apps
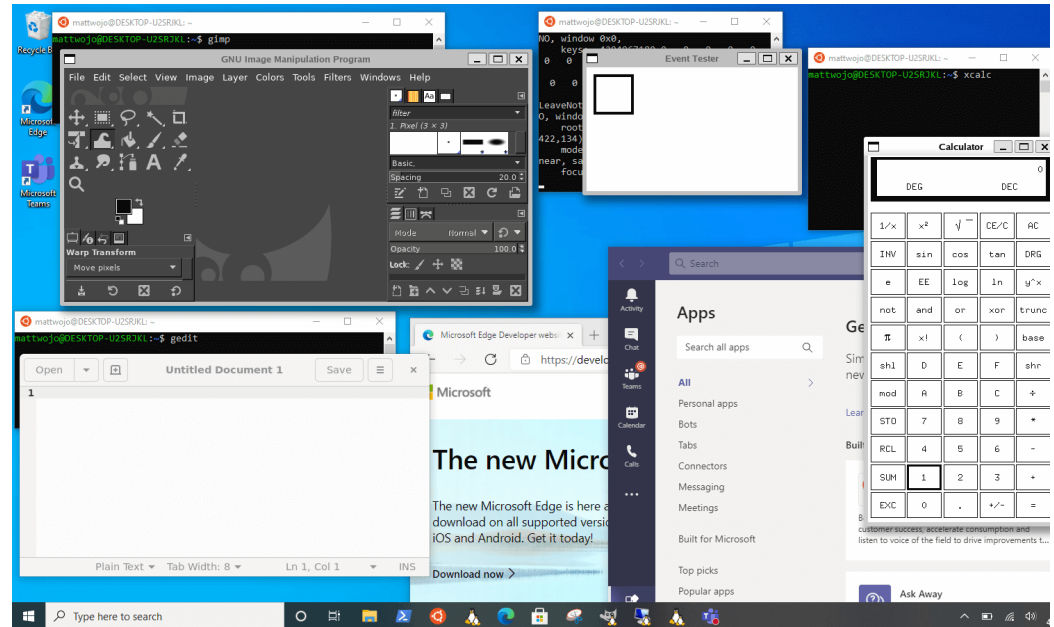
Filesystem Topics

On-demand File System (BeeGFS)

What is a On-Demand File System (ODFS)

- It is On-Demand ;)
  - A file system created on-demand with a short lifetime – Job runtime
    - Started exclusive for your job (shortly before job start)
    - Only available on the Compute nodes allocated for your job
    - Removed/deleted after job ends (shortly after job end)
- Why is there a ODFS?
  - Global file systems like $HOME, $WORKSPACE are shared medium
  - Performance is shared with all users
  - Interference with other Users and Jobs
- Who should use the ODFS feature
  - If your application does a lot of I/O (millions of operations in "short" time)
    - create/remove files/directories
    - open/close read/write files
  - If you need help with using it – contact us

ODFS Overview (1)
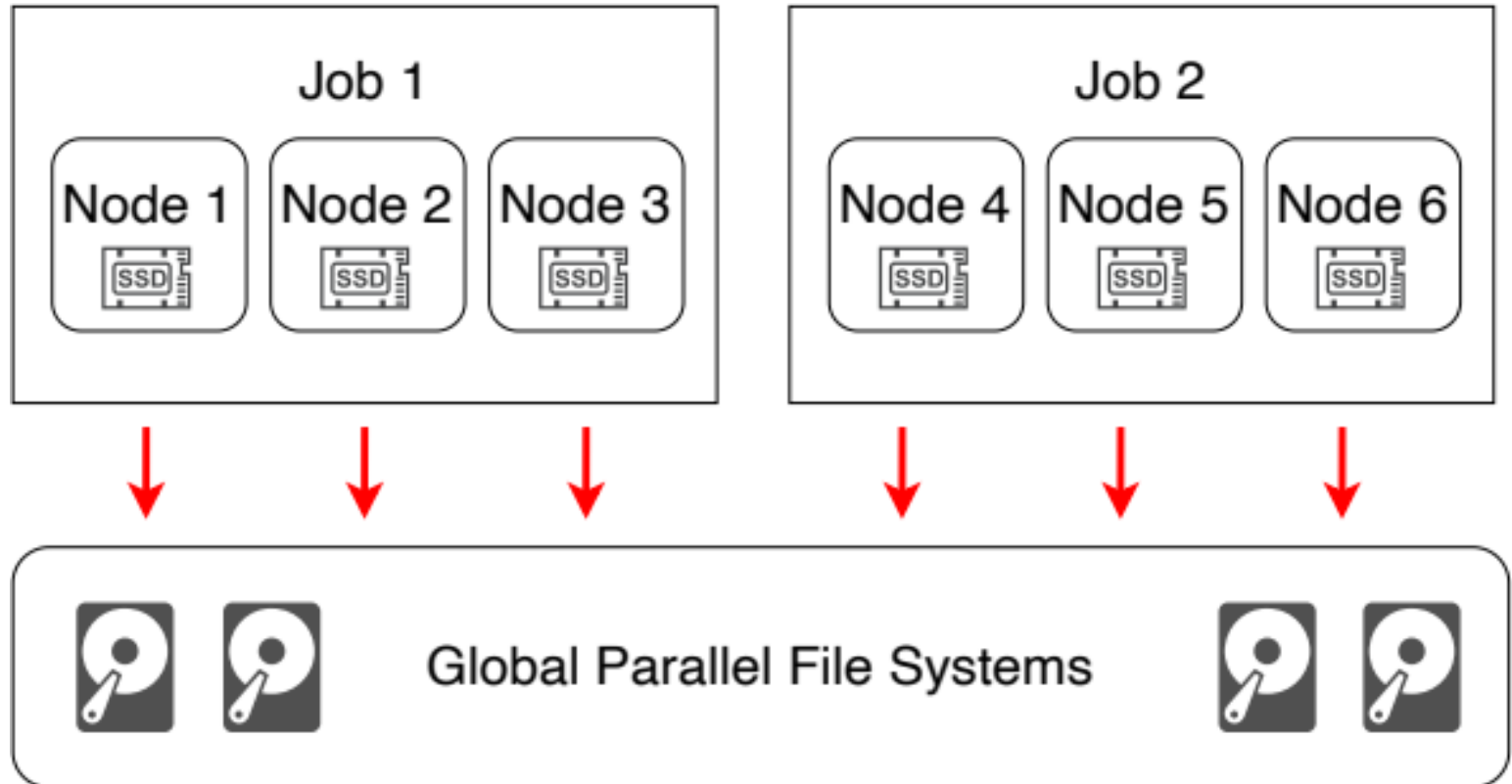
ODFS Overview (2)
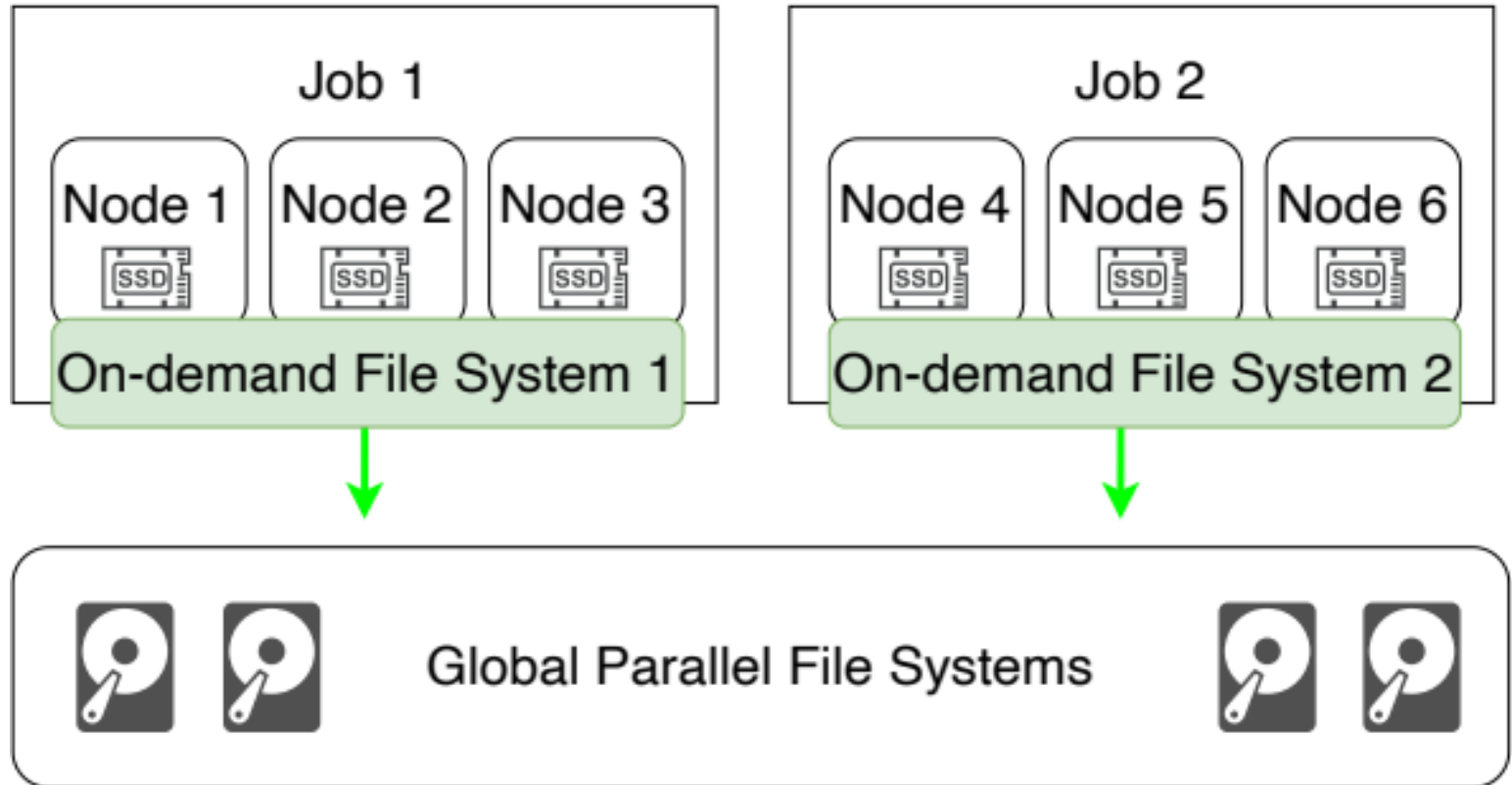
ODFS Workflow
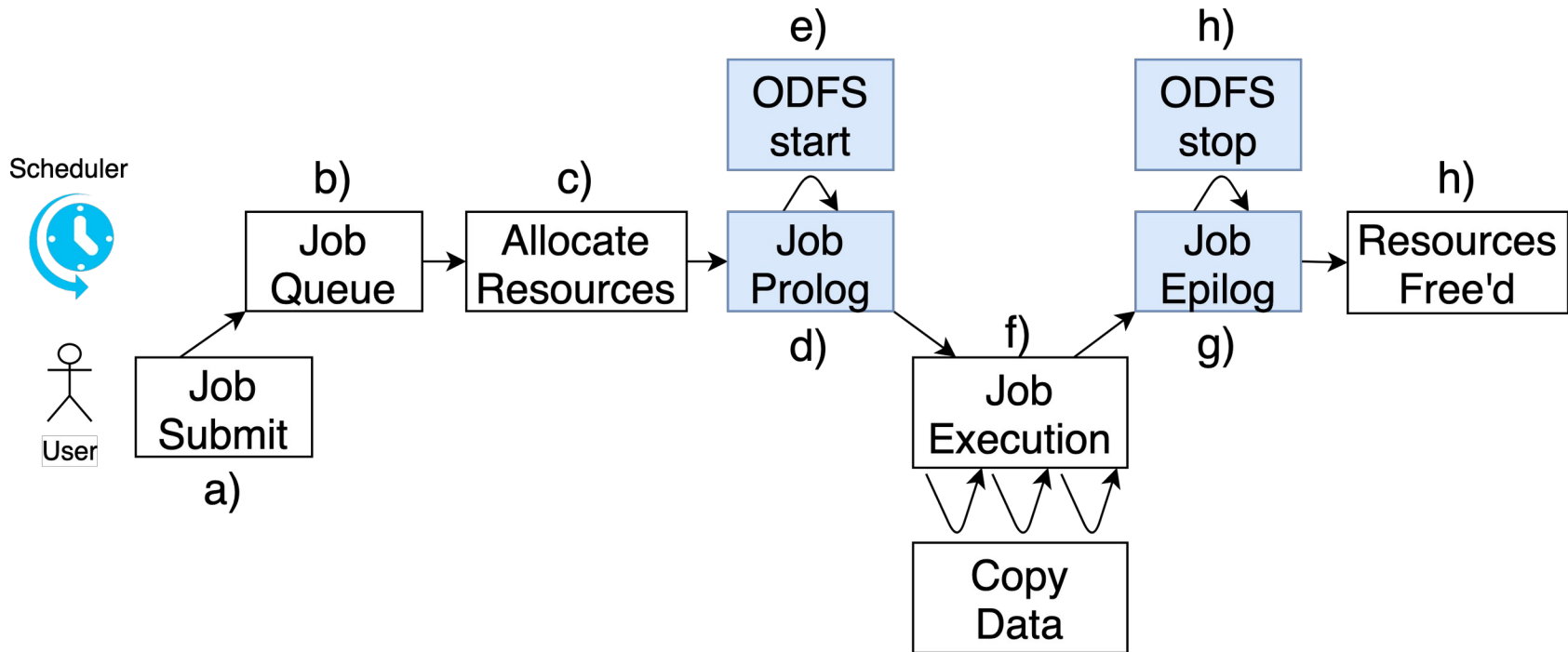
ODFS Workflow
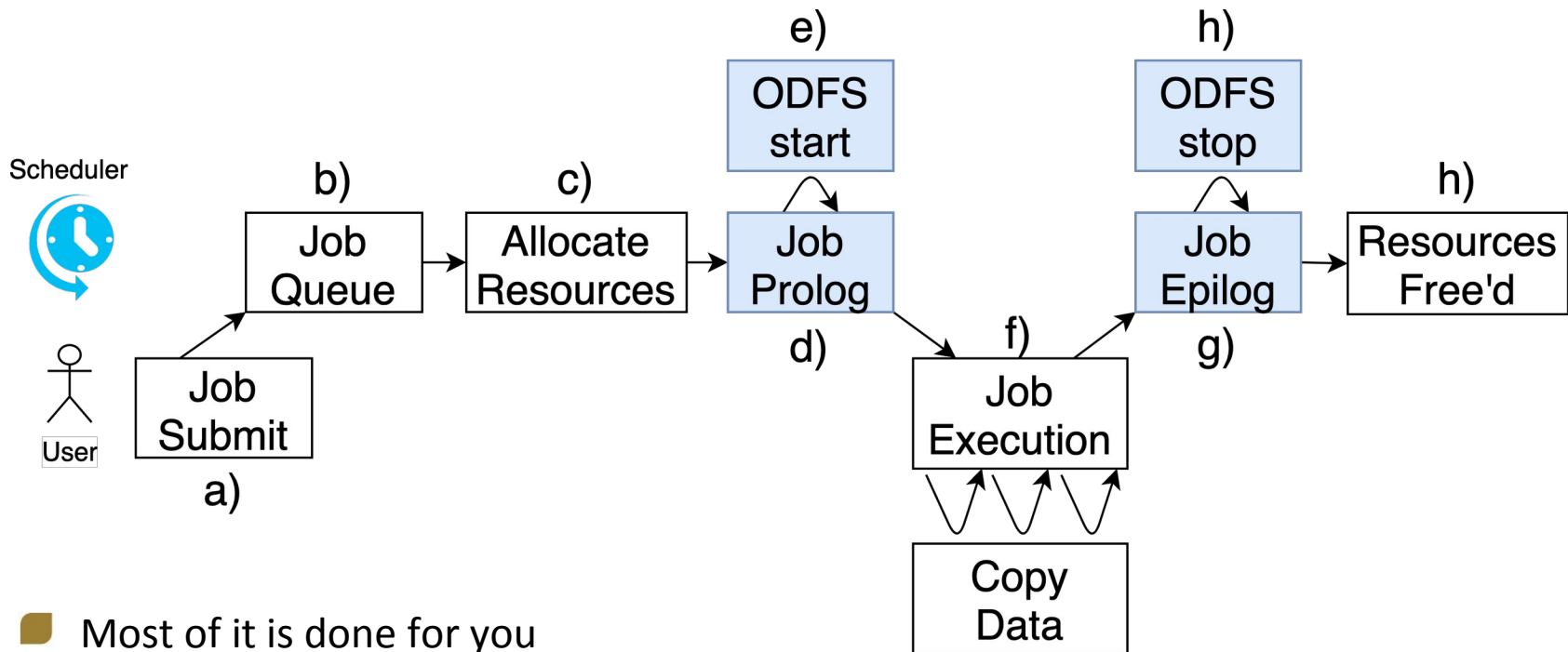
- Most of it is done for you
- You have to request it c)
- You have to manage copy in/out – f)

(Dis-)advantages when using ODFS

- Using NVMe-(SSD)
- SSDs have huge random I/O performance
- 500/750 GB per node (UC2/HoreKa)
  - 10 Nodes → 7.5 TB
- You can do as much I/O as you want
  - You hurt no one else (good for us/everyone)
  - No one can hurt you (good for you)
  - BUT: You can only hurt yourself (its ok for us/everyone)
- You have to copy your data in/out by yourself
- Only useful if you use more than one node
  - More nodes → more performance (Metadata performance vs. throughput)

How to use it

- Request ODFS
    - `#SBATCH --constraint=BEEOND` (within Jobscript)
    - `-C BEEOND` on the command-line
- ODFS is mounted/available here: `/mnt/odfs/${SLURM_JOB_ID}`

- Some Tips & Tricks (ask us here for best practice)
    - https://www.nhr.kit.edu/userdocs/horeka/batch_slurm_beeond/
    - If your job does very heavy I/O, request one additional node
        - Use `mpirun -nolocal` (leave first node for ODFS)
    - For convenience we create multiple directories for `stripe_1` to `stripe_32`
        - Usually: put <u>small</u> files → `stripe_1`
        - <u>Larger</u> files into higher stripes, e.g. `stripe_8` …
        - If you are unsure, just use the <u>default</u>: `stripe_default` (`stripe_4`)

Hands on ODFS

- Create Jobscript

```
$ mkdir odfs_test
$ cd odfs_test
# Create "testfile.sh" with following content
#!/bin/bash

#SBATCH -N 2
#SBATCH --time=00:05:00
#SBATCH --constraint=BEEOND
#SBATCH -p cpuonly
echo "My JobID is: ${SLURM_JOB_ID}"
echo "This Script is executed on node: ${SLURMD_NODENAME}"
echo "This job is running on nodes: ${SLURM_NODELIST}"
echo "ODFS mounted here: /mnt/odfs/${SLURM_JOB_ID}"
echo "list of directories"
ls /mnt/odfs/${SLURM_JOB_ID}
echo "---------------------"
echo  "List of available filesystem"
echo "Take a closer look at a line with /mnt/odfs"
echo "-----------------"
df -h

$ sbatch testfile.sh
# Check outputfile slurm-<jobid>.out
```

Hands on ODFS (interactive)

```
# Request interactive job
$ salloc -p cpuonly -N 4 -t 20 -C BEEOND
salloc: Granted job allocation 1495874
salloc: Waiting for resource configuration
salloc: Nodes hkn[1009,1011-1013] are ready for job

# Check the prompt
$ ls -l /mnt/odfs/${SLURM_JOB_ID}
$ touch /mnt/odfs/${SLURM_JOB_ID}/onlyatest
$ ls -l /mnt/odfs/${SLURM_JOB_ID}

# Create bash script
$ cat touch_hostname.sh
#!/bin/bash
MYHOSTNAME=`/bin/hostname`
touch /mnt/odfs/${SLURM_JOB_ID}/${MYHOSTNAME}

$ chmod 755 touch_hostname.sh
$ mpirun -npernode 1 touch_hostname.sh
$ ls -l /mnt/odfs/1495874/

# Check stripe counts
$ beegfs-ctl --mount=$(pwd) --getentryinfo /mnt/odfs/1495874/stripe_default
$ rm /mnt/odfs/1495874/hkn*

# Check the missing file
$ mpirun -nolocal touch_hostname.sh
$exit
```

KIT | NHR
National High-Performance Computing Center
bw|HPC

ODFS – Summary

- Stay up-to-date (read wiki for changes)
    - https://wiki.bwhpc.de/e/BwUniCluster_2.0_Slurm_common_Features#BeeOND_.28BeeGFS_On-Demand.29
    - https://www.nhr.kit.edu/userdocs/horeka/batch_slurm_beeond/
- Don't hesitate to contact support
    - We can check if it is useful for your use case
    - We are happy to advise you
- Many more options available for more advanced usage
    - Multi Metadata servers (BEEOND_4MDS, BEEONDS_MAXMDS)

Software

Best Practice: Installing Own Software

- Check list:
  - Legal issues: do you have licence for your software?
  - Installation procedure?
    - If compilation exceeds 10 min
      - Install via interactive batch job on a compute node
    - Never use: `make -j` → but: `make -j <number>`
    - Never simply use binaries on different architecture
      - But: recompile or compile supporting multiple architecture
    - Use guides stored in software module files
- Help:
  - Contact support
    - https://bw-support.scc.kit.edu
    - https://support.nhr.kit.edu/
  - Tiger Team Support
    https://zas.bwhpc.de/shib/en/call4tt.php

Best Practice: Compiling code

- Details to compilation and Makefile tutorial:
  - See today's talk „Compile, Makefiles"

- Clusters with different architectures / generations
  - e.g. HoreKA (Intel and AMD, AVX512)

    ```
    hk-login1:~$ icc/ifort -xHost ...
    ```
    ⟶ May crash on AMD node

  - Compile code on corresponding login node or interactively (salloc)
  - Compile code including multiple, feature-specific code paths

    ```
    hk-login1:~$ icc/ifort -xCORE-AVX2 -axCORE-AVX512 ...
    ```

Using Containers: Enroot and Singularity

- Containers @ HPC
  - Security concerns
  - Performance (?)
  - Work in progress
- Singularity
  - Documetation
- Enroot
  - Enroot project: https://github.com/NVIDIA/enroot
  - Pyxis project: https://github.com/NVIDIA/pyxis
  - NHR@KIT Wiki:
    https://www.nhr.kit.edu/userdocs/horeka/containers/

Enroot: How-To Use

- **Import a container image**

  - `enroot import docker://alpine`
    - Pulls the latest alpine image from dockerhub (default registry)
      → `alpine.sqsh`

    - `enroot import docker://nvcr.io#nvidia/pytorch:21.04-py3`
    - Pulls the latest pytorch image from NVIDIA's NGC registry
      → `nvidia+pytorch+21.04-py3.sqsh`

    - `enroot import dockerd://myalpine`
    - `enroot import podman://myalpine`
    - Get image from a running Docker daemon / local Podman registry
    - → `myalpine.sqsh`

Enroot: How-To Use

- **Create a container**
  - `enroot` `create` `--name alpine alpine.sqsh`
  - Create a container named `alpine` by unpacking the `.sqsh`-file

  - Creating: unpack squashed container image file
    - `$ENROOT_DATA_PATH/` (default: `$HOME/.local/share/enroot/`)

Enroot: How-To Use

- **Start a container**

  - `enroot start` `--rw` `nvidia+pytorch+21.04-py3 bash`
  - Start the container in read-write mode (`--rw`) and run bash inside the container

  - `enroot start` `--root` `--rw nvidia+pytorch+21.04-py3 bash`
  - Start container with root privileges (`--root`) inside the container

  - `enroot start` `-m <localDir>:/work` `-rw \`
    `        nvidia+pytorch+21.04-py3 bash`
  - Start container and mount (`-m`) a local directory to `/work` inside the container.

  - `enroot start -m <localDir>:/work -rw \`
    `        nvidia+pytorch+21.04-py3` `jupyter lab`
  - Start container and start the application `jupyter lab`

Enroot+Pyxis: How-To Use
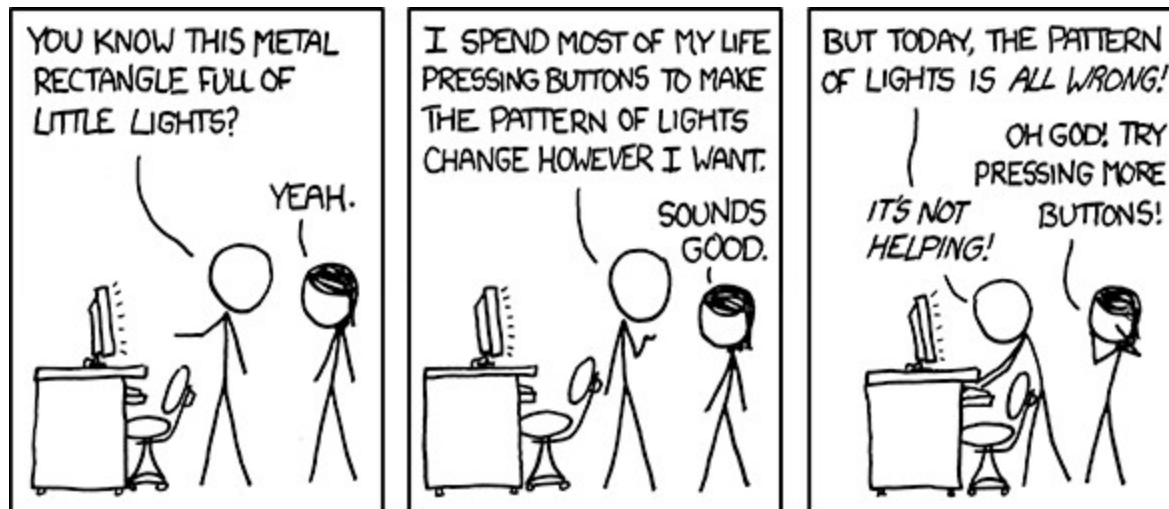
- Pyxis

  - ```
    srun --container-image=centos grep PRETTY /etc/os-release
    PRETTY_NAME="CentOS Linux 8"
    ```

  - ```
    srun --container-image nvcr.io#nvidia/pytorch:20.03-py3 …
    ```

  - ```
    srun --container-image ~/ubuntu.sqsh …
    ```

Enroot+Pyxis: How-To Use

- `srun --help`
    - …
    - `--container-image=[USER@][REGISTRY#]IMAGE[:TAG]|PATH`
    - `--container-mounts=SRC:DST[:FLAGS][,SRC:DST...]`
    - `--container-workdir=PATH`
    - `--container-name=NAME`
    - `--container-save=PATH`
    - `--container-mount-home`
    - `--no-container-mount-home`
    - `--container-remap-root`
    - `--no-container-remap-root`
    - `--container-entrypoint`
    - `--no-container-entrypoint`

KIT | NHR
National High-Performance Computing Center   bw|HPC

Thank you for your attention!

Questions?

https://xkcd.com/722