Effective use of batch computing and storage in Zeuthen

A guide to success ;-)

Andreas Haupt – DV –
Data Science Seminar, Zeuthen

Motivation

… what you should take home from this talk

- The Computing Centre hosts lots of (expensive) compute/storage resources
  - Aim to get the most output and value out of them
- The talk will be divided into two main parts: storage & compute
  - Both are closely connected
  - Incorrect usage of central resources can badly affect performance & availability
    - not just only for you, but for every other user!
    - so it helps all of us to avoid bad usage patterns ;-)
- Disclaimer …
  - Amount of information is quite huge and was shortened at some points
  - In case of missing details, another presentation (one topic, more detailed) could be provided

Overview

Numbers, numbers, numbers ...

- Compute resources

Cluster	use case	Cores	HS06
PAX	HPC	~1800	40k
Local farm	High Throughput	~3500	70k
Grid farm	Grid Computing	~4500	92k

Overview

Numbers, numbers, numbers ...

- Central storage resources

Storage product	Use case	capacity / servers	Notes
Lustre	mass storage with fast parallel data access (scratch space)	~2.8PB / 36	
dCache	data archive, mass storage with fast parallel data access	~7.3PB / 100	
AFS	$HOME, software, scratch	~260TB / 22	
Sync&Share	Document share		Provided by DESY-HH

Storage

Central storage systems

… some general remarks

- Network-attached storage is supposed to provide:
  - Data access on whichever client system you are running your stuff
  - Fast, aggregated data throughput by accumulating the power of many storage servers

- However, this flexibility has some drawbacks
  - Latency matters!
    - metadata i/o operations are comparable slow due to complex operations in background
    - open/close ops, recursive find, compiling software!
  - File locks are usually only enforced on the same client
  - Your notebook ssd will perform significantly better for some usage patterns
    - Keep this in mind whenever you are migrating software/workflows to our central compute environment

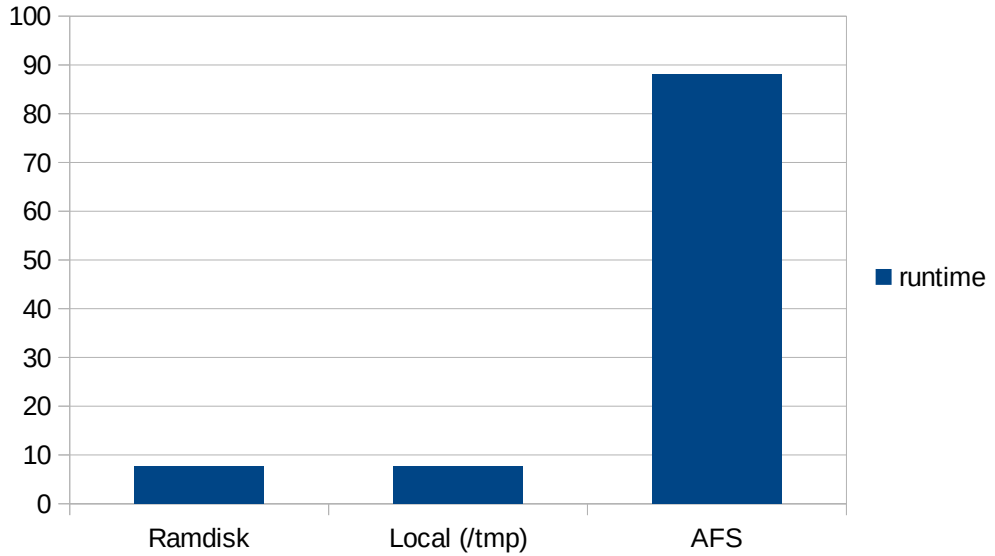
Central storage systems

… even more general advises

- Avoid i/o as much as possible
  - example: avoid reading the same files again and again
- Avoid too many clients writing into the same directory
- Avoid concurrent writes from multiple clients to the same file
  - in many cases the result will not be the expected one ...
- Try to move i/o operations which are affected by latency to local scratch (/tmp, $TMPDIR)
  - Example: untar software in /tmp, compile in /tmp but install to AFS
- In extreme cases: consider usage of ramdisks (/dev/shm)
  - but do not forget to clean up afterwards

Extract Linux kernel (>70k files)
Runtime in seconds:

```
tar -xJf linux-5.6.15.tar.xz
```



In case of problems ...

when things go wrong ...

- In case of data access issues (slow ops, errors, etc.) contact uco-zn@desy.de
- Please be as much precise as possible in describing your problem!
  - It really helps us identifying the possible source of the problem
- A typical complaint: “My access to cluster is slow!”
  - … but what does that mean?
  - Therefore here some details we always would like to here from you:
    - Which operations do not perform as desired?
    - Which files/directories are affected?
    - Which client(s) are you using when hitting the problem?
    - In case you use many clients: Are all of them affected? Or just some of them? Which ones?

Getting storage resources

I need more space!

- Please keep in mind that we are not allowed to purchase large amounts of storage “on spec”
  - mainly: financial restrictions ...

- Unfortunately, especially for Lustre storage, we are not as flexible in providing new space as we want
  - Lustre is not designed to get easily extended once is has been established
    - “just put in another machine” does not really work :-(
  - We usually establish **one Lustre file system per year**
    - This should cover all the year's demands
  - So, we need to know yearly storage demands **well in advance**!

- Therefore: the first person to contact should always be your group leader!
  - He (alternatively: your group's computing contact) should have an overview of your group's resources
  - … and should announce further demands directly to us or via the Computing Board

AFS

… still the work horse for personal data

- Organized in volumes with quotas
  - Your $HOME directory on WGS is one volume
  - Volumes have mountpoints (the place they are accessible for the client)
  - Partly self-service management via group admins
- Access restrictions via ACLs (access control lists) per directory
- Centrally managed backup
  - … and last day's snapshot of $HOME available in ~/.OldFiles !
- Authentication based on a so-called „AFS token”
  - Will be generated automatically during the login process
  - … but can and will finally expire if not renewed in time (lifetime 25 hours)
  - Expiring AFS tokens are the main source of access problems!
- Unfortunately AFS not designed to work flawlessly on mobile devices (like notebooks)
  - use sshfs or your client's “connect to server” (connection type: ssh) instead

AFS

… basic management

- Group‘s AFS space managed by groups themselves:
  - First person to contact: your group’s computing contact
  - Create, increase, … volumes beyond **/afs/ifh.de/group/<group>**
- Please do not mess with ACLs directly in $HOME
  - Use subdirectories and adapt ACLs there
  - Classic Unix access rights are mainly ignored, so something like this is useless:
  - `[wgs34] ~ % chmod 0600 my-private-file`
- Some subdirectories in AFS $HOME are automatically created during account initialisation:
  - ~/private : your private space, ACLs adjusted so that only you can access files stored there
  - ~/public : the opposite – any other user with AFS access can read files
  - ~/public/www : content visible on our public webserver (https://www.zeuthen.desy.de/~<user>)

AFS ACLs explained

- AFS rights:

l	lookup - read entries in this directory
r	read – read files in this directory
w	write – write files in this directory
i	insert – add new files to this directory
d	delete – delete files from this directory
a	administer – modify ACLs in this directory

- Some pre-defined groups are often set:

system:anyuser	any user in the world
system:administrators	an administrator
ifh-hosts	all hosts at DESY Zeuthen
desy-hosts	all hosts at DESY (Hamburg + Zeuthen)
group:<group name>	members of a DESY group

example ACL:

```
[wgs1d] /project/singularity % fs la .
Access list for . is
Normal rights:
  desy-hosts rl
  ifh-hosts rl
  system:administrators rlidwka
  group:usg_zn rlidwka
  system:anyuser l
  znasw rlidwka
```

AFS usage examples

… yes, you can do this at home :-)

- Show quota:

```
[wgs34] ~ % fs lq ~
Volume Name                    Quota       Used %Used   Partition
user.ahaupt                  2000000    1549680   77%         16%
```

- List ACLs:

```
[wgs34] ~ % fs la ~
Access list for /afs/ifh.de/user/a/ahaupt is
Normal rights:
  system:administrators rlidwka
  system:anyuser l
  ahaupt rlidwka
```

- Add an ACL to a directory:

```
[wgs34] ~ % fs sa ~/friends group:cta rl
[wgs34] ~ % fs la ~/friends
Access list for /afs/ifh.de/user/a/ahaupt/friends is
Normal rights:
  group:cta rl
  system:administrators rlidwka
  system:anyuser l
  ahaupt rlidwka
```

AFS usage examples

… continued

- Instant (world-wide) data sharing:

```
[wgs34] ~ % echo 'very cool content' > ~/public/www/my-share.txt
[wgs34] ~ % curl https://www.zeuthen.desy.de/~ahaupt/my-share.txt
very cool content
```

- Check if your AFS token is still valid:

```
[wgs34] ~ % tokens

Tokens held by the Cache Manager:

Tokens for afs@ifh.de [Expires May 29 09:20]
   --End of list--
```

- Fetch a new AFS token (including a new Kerberos ticket):

```
[wgs34] ~ % kinit
ahaupt@IFH.DE's Password:
```

Lustre

… put your huge datasets here

- Large and fast “scratch space”
  - Suitable for any bulk data (in large files preferably!)
  - NOT designed to host source trees, executables, etc.
    - minimum internal request size is 1MB – so if you think you read a 5kByte file, you read 1MB instead!
  - There is NO BACKUP!
- Currently six independent instances active:
  - /lustre/fs{18..23}
- Every Lustre file system lives for 5-6 years and gets replaced by a new one after that
  - Group's task to migrate data still needed
  - A bit of an effort, but also an easy way to clean up – by just doing nothing ;-)

dCache

… the data archive

- Biggest data provider at DESY (not just only in Zeuthen)
- Storage solution to share huge datasets with collaborations anywhere in the world
- Transparent tape backend possible
- Many access protocols
  - dCap (deprecated)
  - NFS-4.1 (makes it feel just like a „normal“ file system)
  - GSIFTP
  - WebDAV (https)
- Many authentication methods exist
  - GSI (Grid security infrastructure with X.509 certificates)
  - Kerberos
  - Macaroons (bearer tokens) – only for WebDAV access

dCache usage

- dCache only partly behaves like a normal Posix file system
  - no file modifications possible – modification means: deletion / re-creation
- Two mountpoints exist:
  - /acs : Old, legacy entry point
    - metadata operations (deletions, chmod, ...)
    - dccp entry (dccp /acs/… /tmp/file)
    - writing/reading data only possible with dCap client (dccp) here!
  - /pnfs/ifh.de/acs : nowadays preferred NFS-4.1 based access
    - handle files just like on any other file system
- Beware of bulk data operations on areas with a tape backend
  - Unfortunately these areas are not easily identifiable from user's perspective
    - … but had been agreed with group’s contact “some time ago”
  - Nevertheless, usually only group's computing admin (or your collaboration) will modify content
  - Massive staging from tape (i.e. >1000 files) should be pre-announced to dCache admins

Sync&Share

… aka „DESYCloud”

- Provided by colleagues at DESY Hamburg
- Based on the file hosting service “NextCloud”
- In theory without any quota!
- Data accessible via web browser, mobile app as well as sync client on any client device
  - Unfortunately data access on centrally managed nodes (wgs, farm, etc.) not available, yet!
    - unclear, when this will change :-(
- Anyway, the perfect place to store and share your documents!

Storage summary

- Unfortunately a “silver bullet” that perfectly fits all storage use cases does not exist
  - It's also questionable whether it will ever exist ...
- Some rules of thumb where to put your data:

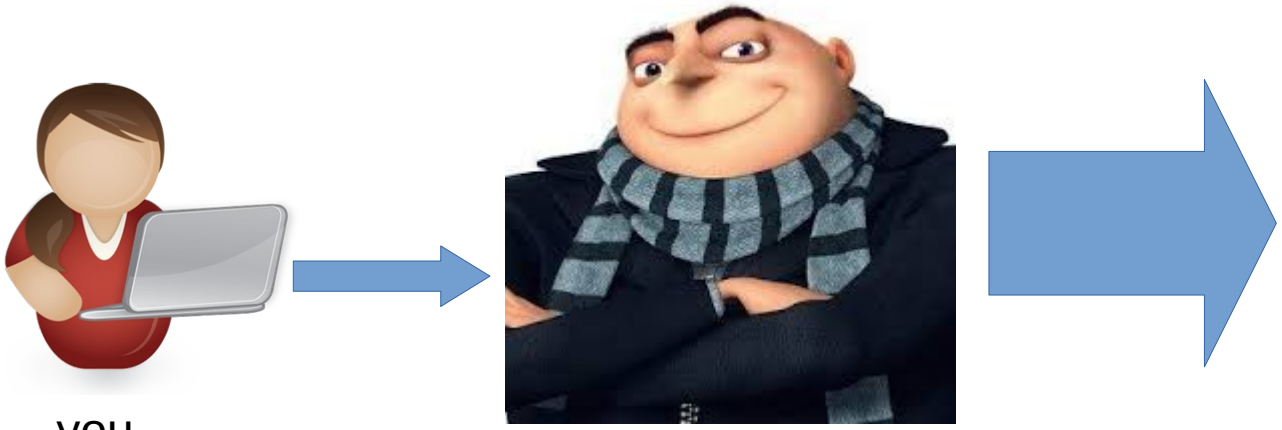
Kind of data	Where to put
Your thesis	AFS $HOME / Sync&Share
Collaboration's, group's or private documents	AFS $HOME / Sync&Share
Collaboration or group (raw) data	dCache
private or group's simulated / derived data	Lustre
Software, histograms, ...	AFS group space

Batch

Batch computing

… heat the computing centre, efficiently ;-)

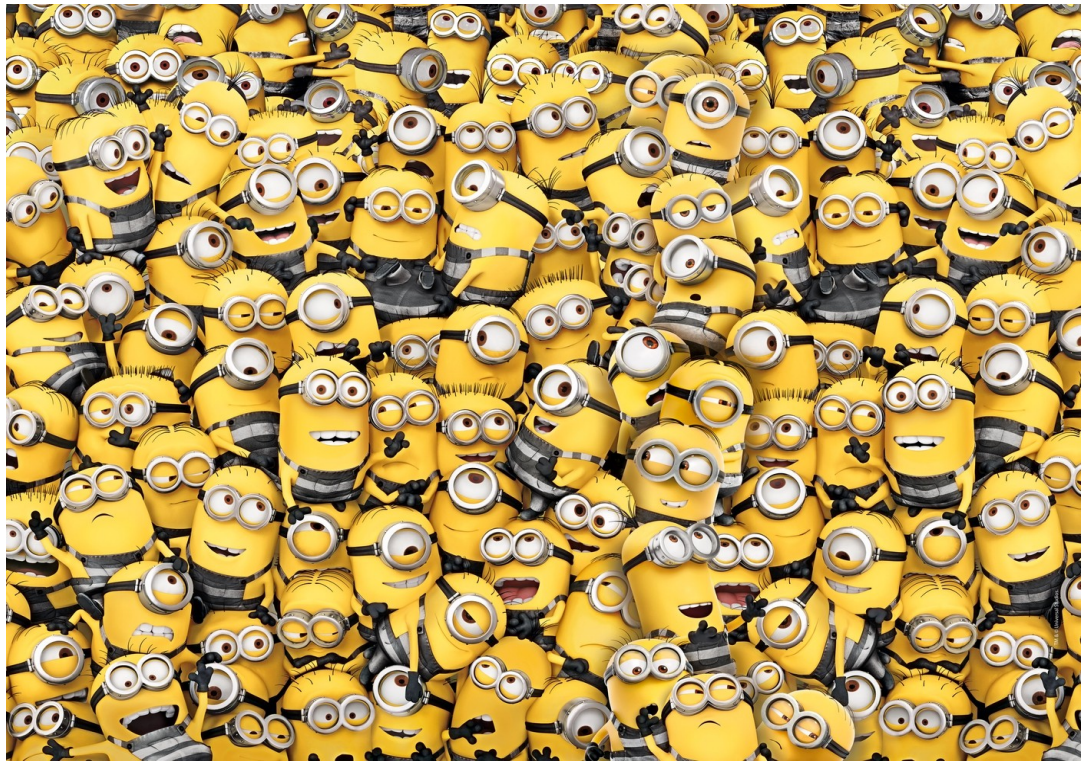
- Have to run too many tasks for a single node?
  - spread the work load over many minions …
- A batch system takes care to schedule your task/job to the next free node as soon as possible



you

batch master

- DESY, Zeuthen uses Univa Grid Engine as batch queueing system



Structure your work / tasks

- Divide your problem into smaller (or even: atomic) tasks
  - every task is a potential single job
  - In case tasks depend on each other, think of building depency graphs
    - … and look for the 'hold_jid' option in the qsub man page
- There's unfortunately no such thing as a “patent remedy”
  - Think about your problem, if in doubt ask for help
- Keep job run times in mind
  - **Prefererred** from admin's point of view: **1-12 hours per job**
    - moderately high job turnaround eases scheduler decisions
    - little lost time in case of node failures
    - and most important: eases admin's maintenance tasks (reboot campaigns, etc.) ;-)
  - Too many too short jobs stress the scheduler
- Jobs running for 2 days or even more?
  - It is highly suggested to split those into smaller tasks

Example job script

- A batch job is actually just a shell script
  - … with some additional information for the scheduler (lines starting with #$)

```
#!/bin/bash
#
# job runtime
#$ -l h_rt=05:30:00
# maximum memory usage (resident set size) of this job
#$ -l h_rss=2G
# scratch space needed in $TMPDIR
#$ -l tmpdir_size=2G
# job's STDOUT/STDERR directory (needs to exist!)
#$ -o /afs/ifh.de/group/mygroup/logfiles/
#$ -j y

# change to scratch directory
cd $TMPDIR
# copy input data to scratch directory
cp /afs/ifh.de/group/mygroup/rawdata/file file
# write calculated output also to scratch first
/path/to/executable/which/works/on/file -in file -out output
# copy the output back into afs
cp output /afs/ifh.de/group/mygroup/mydir/output/
```

Example job workflow

- Submit a job:

```
[wgs34] ~ % qsub <my-job-script>
Your job 61082956 ("myjob") has been submitted
```

- Query the job:

```
[wgs34] ~ % qstat -j 61082956
… lots of job information …
```

- Delete the job (in case you realise you made a mistake during submission):

```
[wgs34] ~ % qdel 61082956
user has registered the job 61082956 for deletion
```

- Query error reason, if job failed to start (ended in status: Eqw)

```
[wgs34] ~ % sge-job-error 61212300
Task: 1, Error: 05/28/2020 13:34:42 [9132:10151]: can't make directory "/bla/bla" as stdout_path:
Permission denied
```

Know your (job) limits!

- Your jobs will get assigned a part of the execution node's resources
- Resources (memory, disk space, ...) are **NOT** unlimited
  - of course, you knew it before, but better remind too much than too little … ;-)
- Overestimation will result in decreased throughput
  - and wasted resources!
  - use the job dashboard (slide 34) to validate
- If in doubt: run a sample job and measure its demands
- Many jobs request way too much h_rss!
  - … compared to actual usage

Specialities

Array jobs, AVX, etc.

- Array jobs provide a way to submit hundreds of identical jobs with a single „qsub“ statement
  - Submit a job with 500 tasks: qsub -t 1-500 <job>
  - Inside a running job you can query your task-id via the env variable $SGE_TASK_ID
  - hint: $SGE_TASK_ID could be e.g. used as start seed in these identical jobs
- By default a job slot gets assigned a single, physical cpu core
  - All of our execution nodes run with hyperthreading activated – so a job can consume two (virtual) cpus
  - Core binding is active by default – a job can only run on the assigned cpu core
- Multi-threaded jobs (e.g. OpenMP) should use parallel environment “multicore”.
  - request 4 cpu cores on a single node: **qsub -pe multicore 4 <job>**
- Execution nodes are up to 6 years old → not all new cpu features available on all nodes
  - request a cpu providing AVX2 instruction set: **qsub -l avx2 <job>**
  - request a cpu providing AVX512 instruction set: **qsub -l avx512 <job>**

GPGPUs

the bitcoin factory ;-)

- Thanks to Icecube our farm also hosts ~150 nVidia General Purpuse GPU devices
  - freely available for opportunistic usage!
  - just in case you need to run some Machine Learing, TensorFlow, etc.

- Different models available:
  - Tesla K20m : 34 devices
  - Tesla K80 : 44 devices
  - Tesla P4 : 36 devices
  - GeForce RTX 2080 Ti : 42 devices

- Request a GPGPU for your job: **qsub -l gpu=1 <job>**
  - The device to use is set in $SGE_GPU_DEVICE
  - $CUDA_VISIBLE_DEVICES is automatically set, too

- Request a special GPGPU model: **qsub -l gpu=1,gpu_type=nvidia_geforce_rtx_2080_ti <job>**

Tips and tricks

dos and don'ts

- As mentioned, jobs should be self-contained
  - Avoid exporting the whole submission environment to the job!
    - This is NOT recommended: **qsub -V <job>**
- You can get a mail every time the status of your job changes
  - **qsub -m [b|e|a] <job>** (start, end, abortion)
  - Use with care!!! Really, I mean it!
    - Job mass failures can easily result in mail storms!
    - You can even block your mail box for legitimate mail temporarly
- Batch farm only hosts SL7-based nodes these days
  - Singularity containers exist to run workload not compiled or available for this Linux distribution
  - It's even partly, transparently integrated
  - Run job in Ubuntu 18.04 Container: **qsub -l singularity_os=u18 <job>**

Some rules of thumb

… for mass productions

- Use array jobs
  - Much faster to submit many tasks
  - Less overhead for the batch system
- Keep the number of log files per output directory reasonable
  - better not exceed 1000 files per directory
    - AFS cannot even store more than 64k files per directory (results in misleading error “file too large”, though)
  - and do NOT use $HOME for them – better a scratch directory in group's AFS space
- Again: avoid sending mails from thousands of jobs!
  - Especially if you read your mail on non-DESY providers!
    - Spam protections can/will temporary lock your mail account for legitimate mail, too!

Usage of mass storage

… for mass productions

- Mass storage is extremely vulnerable to inappropriate usage patterns
  - limiting factor rather iops than (network) throughput
  - e.g. recursive finds are a guarantee for trouble
- General rule of thumb: minimise I/O where possible
  - your mileage may vary i.e. if your workflow is badly affected by data access latency
  - try testing the different options

Shares, priorities

… scheduling explained in a nutshell

- A scheduler run consists of multiple steps:

  1) Filter all jobs in the queue which can be started based on requested and available resources
  2) Prioritize these remaining jobs based on other group's current/past jobs usage and configured share
  3) Start all jobs from the top of this list until available resources are exceeded

- A „share” is globally configured and describes the portion of compute resources a group should get
  - Configured share distribution is to be decided by Computing Board
  - A “buy in” of extra group resources is supported, too

- In case a group does not use its share, others can step in
  - “opportunistic usage”

Reservations

… when classic scheduling hits the limit

- Jobs with huge resource demands can „starve” in the queue
  - Filtered out during each scheduling run as requested resources unavailable at this moment
    - Occupied by other jobs
    - … and “smaller” jobs fill the little gaps again and again
- Reservation: scheduler predicts a future point in time where the resource are available
  - and keeps the small gaps free in the meantime
- Reservations are an expensive operation!
  - Nevertheless automatically enabled for:
    - Multicore / smp jobs
    - Jobs requesting h_rss > 8GB
    - Can also be enabled manually:

```
[wgs34] ~ % qsub -R y <job>
```

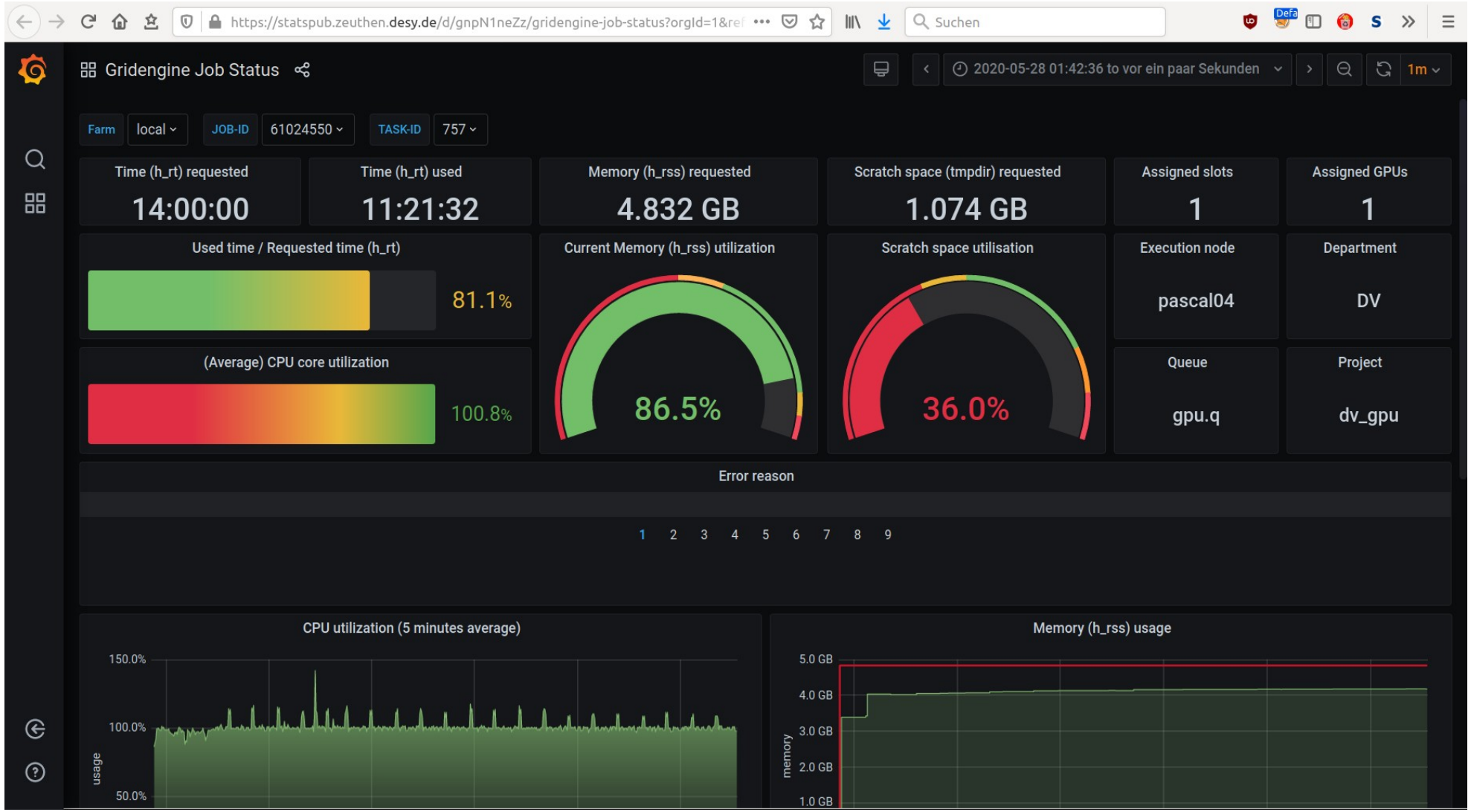
Monitoring ...

… a big mess at the moment :-(

- Old monitoring page still works, but developer left DESY (or: retired) two years ago...
  - https://www-zeuthen.desy.de/macbat/mon
- Nevertheless a fine-granular monitoring of your job's behaviour can pin (potential) problems
- We recently introduced a new computing centre monitoring infrastructure based on Prometheus, ElasticSearch & Grafana
  - … and started collecting job and batch system metrics into it
- During runtime a job can be queried with: **qstat -j <job-id>**
  - many rather useless information displayed here, though …
- Alternatively, query the url to a Grafana-based dashboard with: **sge-job-url <job-id> [task-id]**

```
[wgs1d] ~ % sge-job-url 61024550 757
https://statspub.zeuthen.desy.de/d/gnpN1neZz/gridengine-job-status?orgId=1&refresh=1m&var-
farm=local&var-job_id=61024550&var-task_id=757&from=1590622956000&to=now
```

A job dashboard ...



A batch system dashboard ...

https://statspub.zeuthen.desy.de/d/T28VV4_Wk/gridengine-status?orgId=1&refresh=1m&var-farm=local

- Keep an eye on CPU & Memory utilisation
  - Farm is full if one of them is close to 100%
- Due to special treatment of GPU jobs cpu utilization can exceed 100% temporary

That's it folks ...

- Please also read our docs:
  - Storage: https://dv-zeuthen.desy.de/services/storage_recources/
  - Batch: https://dv-zeuthen.desy.de/services/batch/
- … as well as some hints to set up your notebook for easier remote access:
  - https://dvinfo.zeuthen.desy.de/BYOD/User-Info