HTCondor and Containers for Batch and Interactive use

(Mostly) a success story

Oliver Freyermuth, Peter Wienemann

University of Bonn
{freyermuth,wienemann}@physik.uni-bonn.de

24th September, 2019

Physikalisches Institut
UNIVERSITÄT BONN

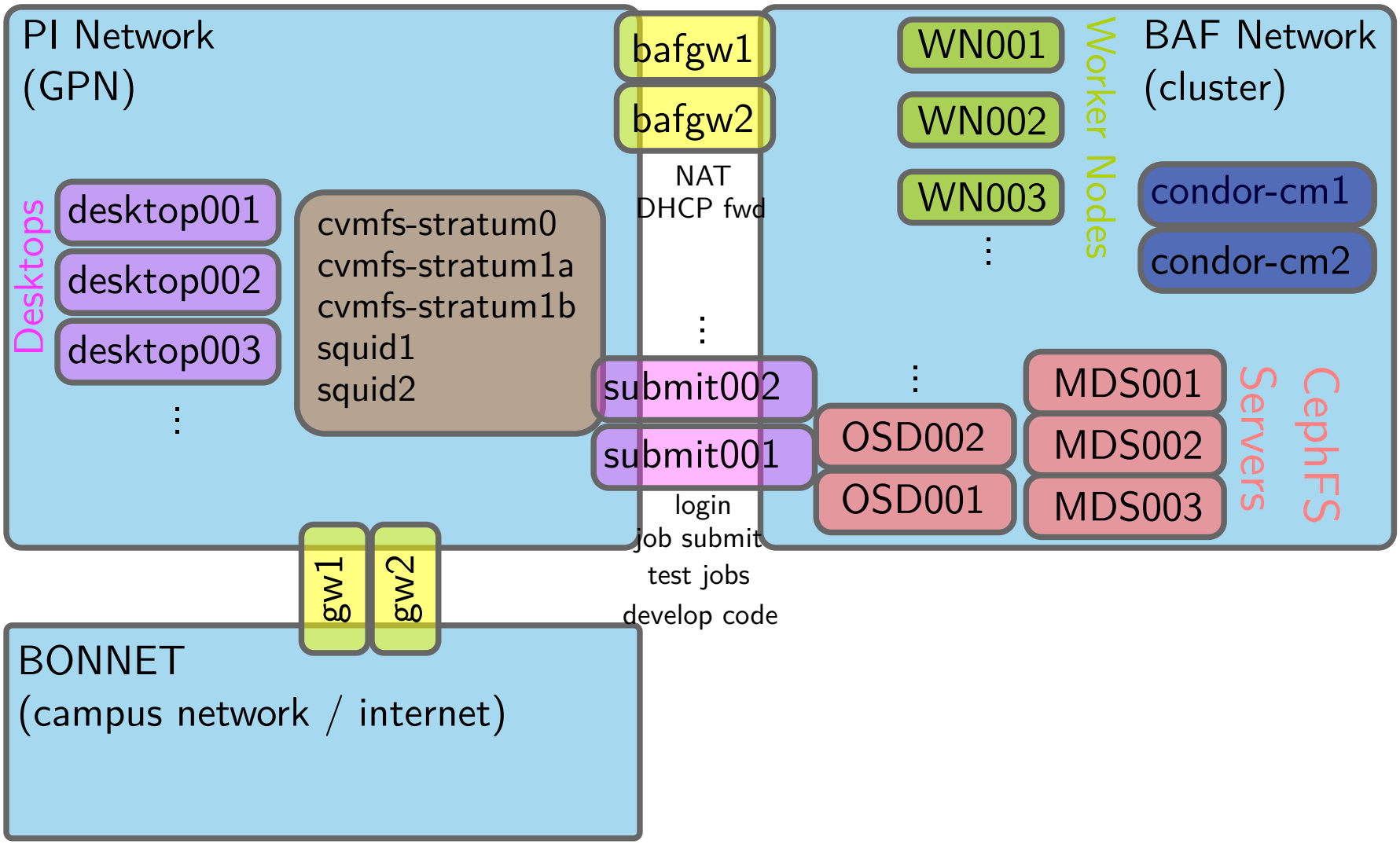
Physics Institute at University of Bonn

- 240 members
- Biggest particle accelerator run by a German university ('ELSA', 164.4 m circumference) with two experiments ($\approx$ 50 people)
- Groups from:
  - High Energy Physics (ATLAS, Belle II)
  - Hadron physics
  - detector development
  - photonics
  - theory groups

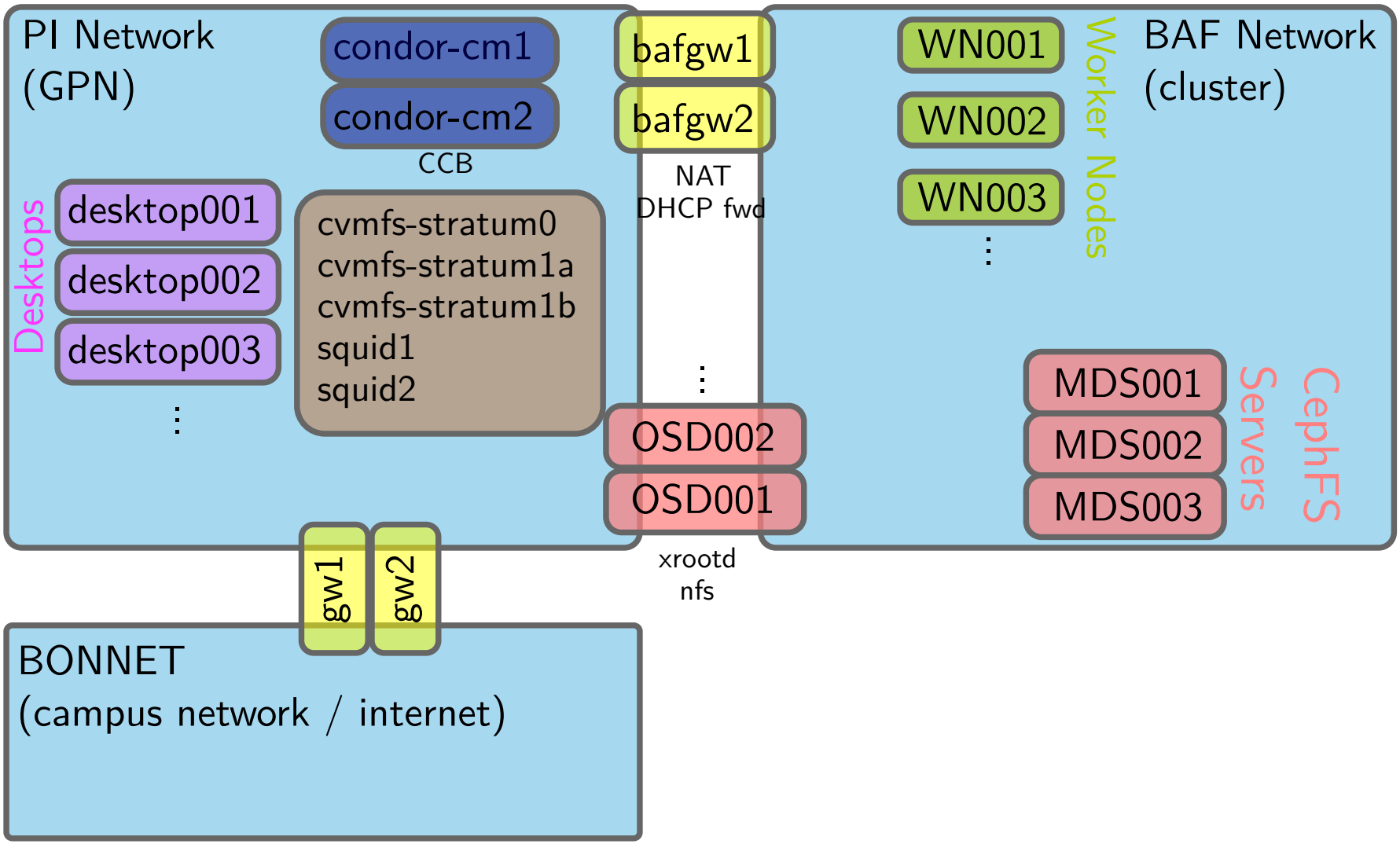
Extremely diverse requirements on software environments & job resources.

Old cluster used PBS / Maui, everything SL 6, mostly HEP usage.
Chance to start over in 2017 => **HTCondor**!

Classical Cluster Setup



Our setup: 'Submit Locally, Run Globally'



Key changes in our new setup

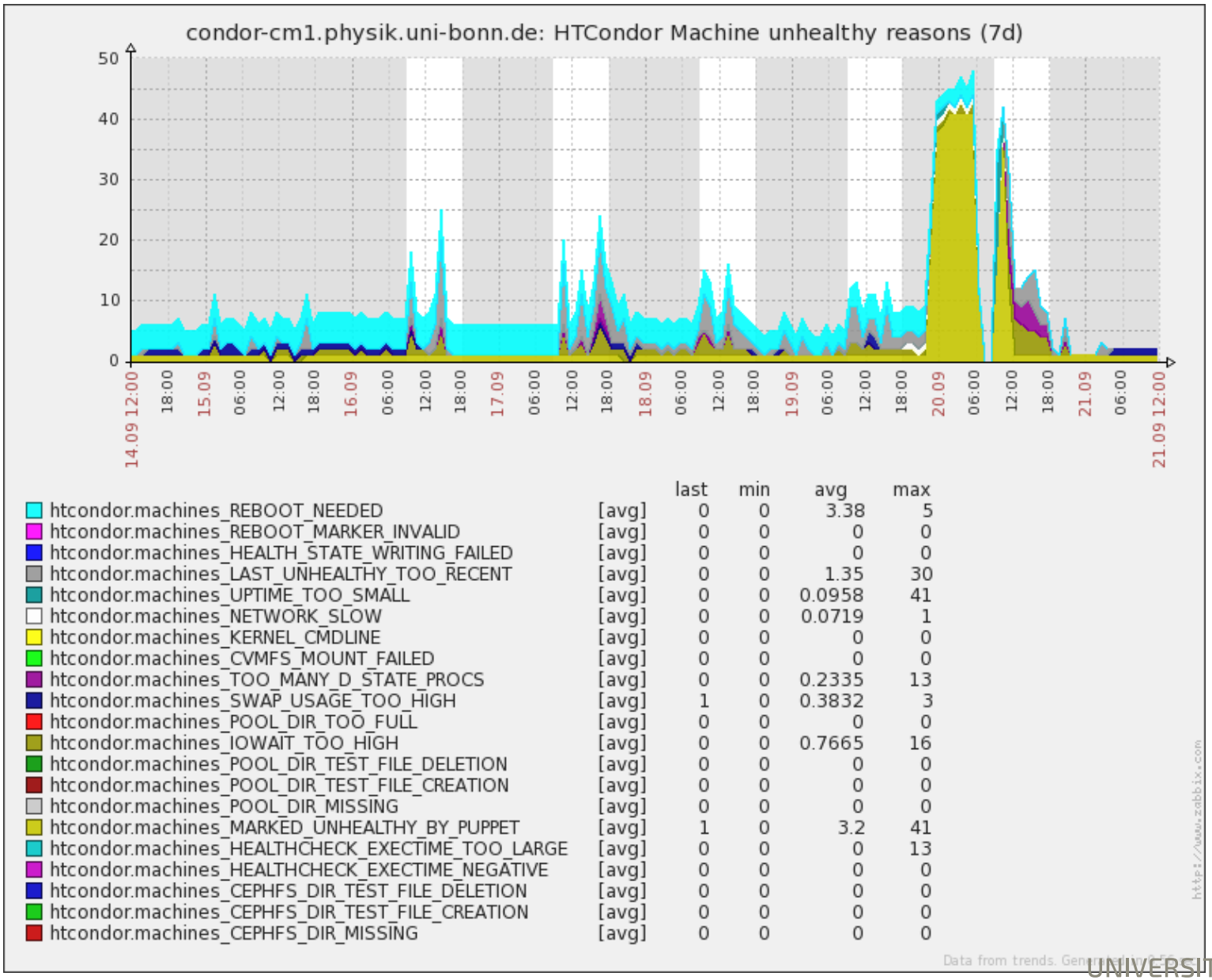
- All desktops, worker nodes, condor central managers fully puppetized, for HTCondor: HEP-Puppet/htcondor
  Module allows to set up queue super-users, block users from submission, set up HTCondor for Singularity,. . .
- **No login / submission nodes** (‘use your desktop’)
- Condor central managers in desktop network
- Desktops running Ubuntu 18.04 LTS
- Cluster nodes running CentOS 7.7
- Full containerization (all user jobs run in containers)
- Containerization decouples OS upgrades from user jobs
- Cluster file system (CephFS) directly accessible from Desktop machines via NFS.
- Cluster worker nodes interconnected with InfiniBand (56 $^{\text{Gbit}}/_{\text{s}}$) instead of Gigabit ethernet

HTCondor Configuration

- Authentication via Kerberos / LDAP
  - Issues with ticket lifetime don't hit us heavily — ***yet***
    (mostly short jobs, Kerberos only needed on submit machine)
  - Hit by some HTCondor bugs (no ticket caching on Collector overloading KDC servers, `dagman` authentication issue)

  ⇒ Looking forward to HTCondor prolonging tickets!
- Node health script:
  - run via `STARTD_CRON`
  - can pick up admin-enforced state via Puppet
    (e.g. for maintenance)
  - picks up state from 'reboot-needed' cronjob
  - Captures common node overload issues:
    - Heavy I/O on local disks (`iowait`)
    - Heavy swapping (HTCondor cannot limit swap usage!)

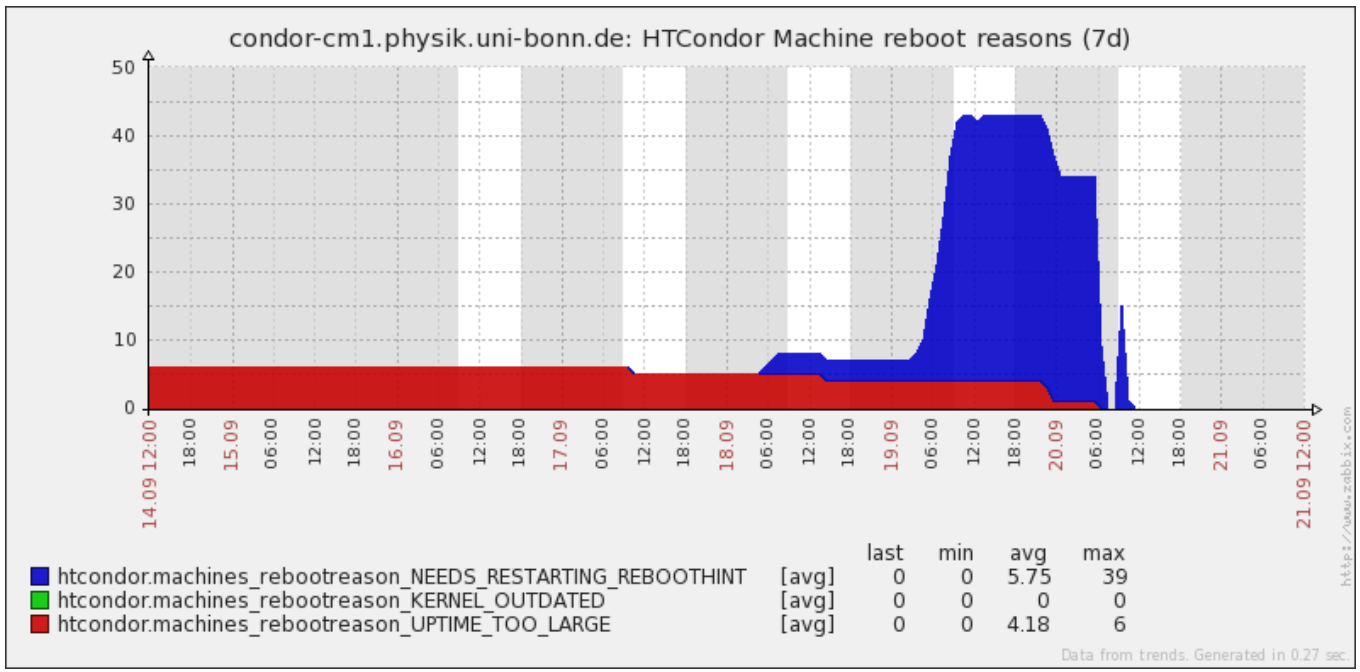
Node health checking



		last	min	avg	max
htcondor.machines_REBOOT_NEEDED	[avg]	0	0	3.38	5
htcondor.machines_REBOOT_MARKER_INVALID	[avg]	0	0	0	0
htcondor.machines_HEALTH_STATE_WRITING_FAILED	[avg]	0	0	0	0
htcondor.machines_LAST_UNHEALTHY_TOO_RECENT	[avg]	0	0	1.35	30
htcondor.machines_UPTIME_TOO_SMALL	[avg]	0	0	0.0958	41
htcondor.machines_NETWORK_SLOW	[avg]	0	0	0.0719	1
htcondor.machines_KERNEL_CMDLINE	[avg]	0	0	0	0
htcondor.machines_CVMFS_MOUNT_FAILED	[avg]	0	0	0	0
htcondor.machines_TOO_MANY_D_STATE_PROCS	[avg]	0	0	0.2335	13
htcondor.machines_SWAP_USAGE_TOO_HIGH	[avg]	1	0	0.3832	3
htcondor.machines_POOL_DIR_TOO_FULL	[avg]	0	0	0	0
htcondor.machines_IOWAIT_TOO_HIGH	[avg]	0	0	0.7665	16
htcondor.machines_POOL_DIR_TEST_FILE_DELETION	[avg]	0	0	0	0
htcondor.machines_POOL_DIR_TEST_FILE_CREATION	[avg]	0	0	0	0
htcondor.machines_POOL_DIR_MISSING	[avg]	0	0	0	0
htcondor.machines_MARKED_UNHEALTHY_BY_PUPPET	[avg]	1	0	3.2	41
htcondor.machines_HEALTHCHECK_EXECTIME_TOO_LARGE	[avg]	0	0	0	13
htcondor.machines_HEALTHCHECK_EXECTIME_NEGATIVE	[avg]	0	0	0	0
htcondor.machines_CEPHFS_DIR_TEST_FILE_DELETION	[avg]	0	0	0	0
htcondor.machines_CEPHFS_DIR_TEST_FILE_CREATION	[avg]	0	0	0	0
htcondor.machines_CEPHFS_DIR_MISSING	[avg]	0	0	0	0

Node reboot handling

- Detection mainly via `needs-restarting -r`
- Start of drain smeared out over 10 days
- Marks nodes as ‘unhealthy’



This functionality is there (one way or another) in many clusters – but how do we survive without login / submit nodes?

Choice of Container Runtime

- Aiming for unprivileged lightweight runtime
- Needs working HTCondor support including interactive jobs
- Allow image distribution via CernVM FS

CernVM FS

- Read-only file system with aggressive caching and deduplication
- Ideal for many small files and high duplication factor
- Perfect match for unpacked containers
- 'Unpacked' is a requirement for rootless operation

⇒ Settled on Singularity for now, but wishing for support for off-the-shelf solutions such as `Podman` / `runc`.

Singularity

- Supports privileged and unprivileged operation
- Developed at LBNL, optimized for HPC applications: `http://singularity.lbl.gov`
- Process and file isolation, optional network isolation (no kernel isolation)
- Commonly used in HEP community
- Still works with old kernels (e.g. CentOS 6), *privileged only*

However...

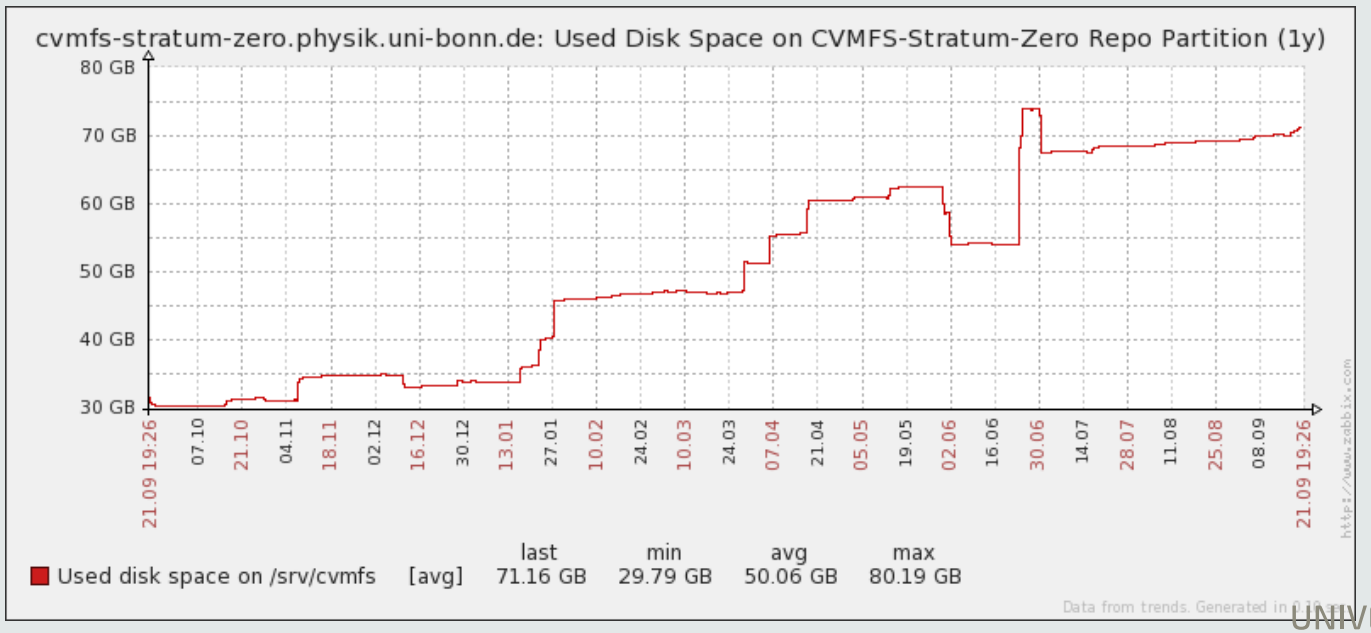
- Young project with non-negligible rate of CVEs (version 3.0 was a full rewrite in Go)
- Focus on SIF™ (Singularity Image Format) requiring root
- Reproduces a lot of existing, standardized infrastructure in a non-standard way (cloud builders, container library etc.)

⇒ **Use it, but avoid a lock-in as far as possible.**

Container Build Workflow

- All containers based on official DockerHub base images
- Offering Ubuntu 18.04, CentOS 7 and SL 6 with site-specifics
- Rebuilt at least daily with Singularity recipe
- Deployed to our own CVMFS, kept there for at least 30 days
- Unpacked images also work with other runtimes (only site-specifics in Singularity recipes slightly builder-dependent)

CVMFS usage over a year, Containers (daily) & Software



Container Site-Specifics

- Compatibility with HEP experiments' requirements (HEP_OSlibs, ALRB)
- User data directory in environment variable, quote check tool
- DBUS hacks for X11 applications in containers
- HTCondor resource requests (login message, environment)
- lmod environment modules integration:

```
module load mathematica/12.0.0
```

- Source user-defined `.bashrc`, potentially OS-specific, from shared file system
- Allow users to relay mail
- Timezone setup
- Add packages requested by users

HTCondor Integration

- All jobs forced into Singularity

```
SINGULARITY_JOB = true
```

- Users can select from pre-build containers ('choose your OS')

```
CHOSEN_IMAGE = "$(SL6_DEFAULT_IMAGE)"
CHOSEN_IMAGE = ifThenElse(TARGET.ContainerOS is
↪   "CentOS7", "$(CENTOS7_DEFAULT_IMAGE)",
↪   $(CHOSEN_IMAGE))
CHOSEN_IMAGE = ifThenElse(TARGET.ContainerOS is
↪   "Ubuntu1804", "$(UBUNTU1804_DEFAULT_IMAGE)",
↪   $(CHOSEN_IMAGE))
SINGULARITY_IMAGE_EXPR = $(CHOSEN_IMAGE)
```

- Paths to most recent image per OS and available OSes provided by `include command : someScript.sh`

‘Choose your OS’

- Users add to their Job ClassAd:

```
+ContainerOS = "CentOS7"
```

- Their jobs run in a container
- Same for interactive jobs (‘login-node experience’!)
- Small fractions of worker nodes exclusively for interactive jobs
  But: Interactive jobs can go to any slot!
- Resource-request specific tuning via `/etc/profile` possible:

```
REQUEST_CPUS=$(awk '/^RequestCpus/{print $3}'
↪  ${_CONDOR_JOB_AD})
export NUMEXPR_NUM_THREADS=${REQUEST_CPUS}
export MKL_NUM_THREADS=${REQUEST_CPUS}
export OMP_NUM_THREADS=${REQUEST_CPUS}
export CUBACORES=${REQUEST_CPUS}
export JULIA_NUM_THREADS=${REQUEST_CPUS}
```

Necessary hacks for interactive jobs

- As of HTCondor 8.6, interactive jobs use an `sshd` running inside the container (i.e. `singularity` is a 'job-wrapper' command)
- Need to have `sshd` installed inside the container
- We only got this to work privileged (potentially could tweak `groups` file to not contain `tty` group to go unprivileged)
- Need some obscure extra bind mounts:

```
SINGULARITY_BIND_EXPR =
 ↪  "/pool,/usr/libexec/condor/,/cephfs,/cvmfs"
```

⇒ Need to include `EXECUTE` directory (`/pool`) and `/usr/libexec/condor` here!

Remaining issues in 8.6...

- `singularity` is only a 'job-wrapper' command
  - ⇒ `sshd` runs in a *new* container
  - ⇒ Interactive works 'fine' (two containers started...), but `condor_ssh_to_job` does not!
- Killing jobs takes long in some cases...
- Difference between batch and interactive (`source /etc/profile` needed in batch)

However...

- We have been running with this for two years now.
- Users are delighted by the new choices, and `ssh -X` works!
- There's light on the horizon...!

The `nsenter` approach

- Enter the namespaces the container runtime has created
  ⇒ Essentially, 'attach' to the container!
- Compatible with *any* container runtime (with potential quirks)
- Other container runtimes one could think of:
  - Charliecloud (https://hpc.github.io/charliecloud/)
    - Even more lightweight (no PID / network namespaces)
      PID namespace could be handled by HTCondor
    - Code is short and easily auditable
  - Podman / runc (https://podman.io/)
    - Included in RHEL 7.6 and 8 with official support
    - Can be used with `alias docker=podman`
    - Can run rootless
    - CRIU integration (freeze, live-migrate)
    - Still requires bind-mount target directories to exist for rootless (GitHub issue 1671)

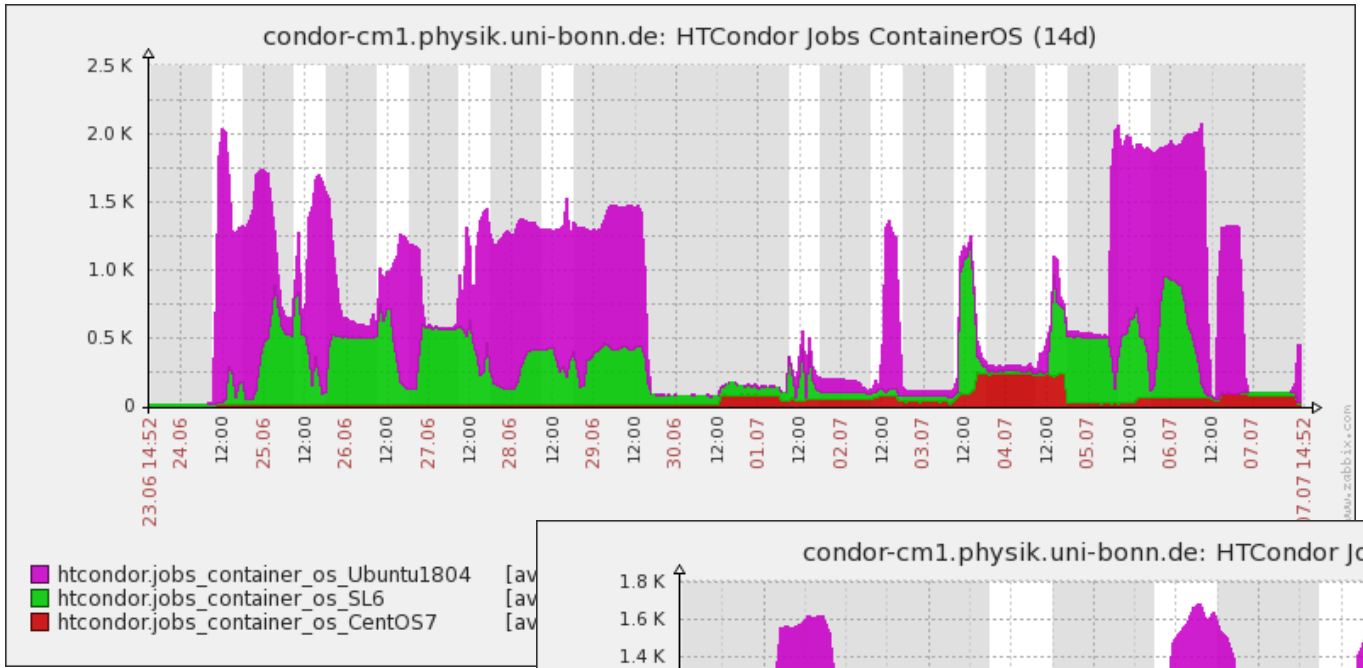
Here comes HTCondor 8.8!

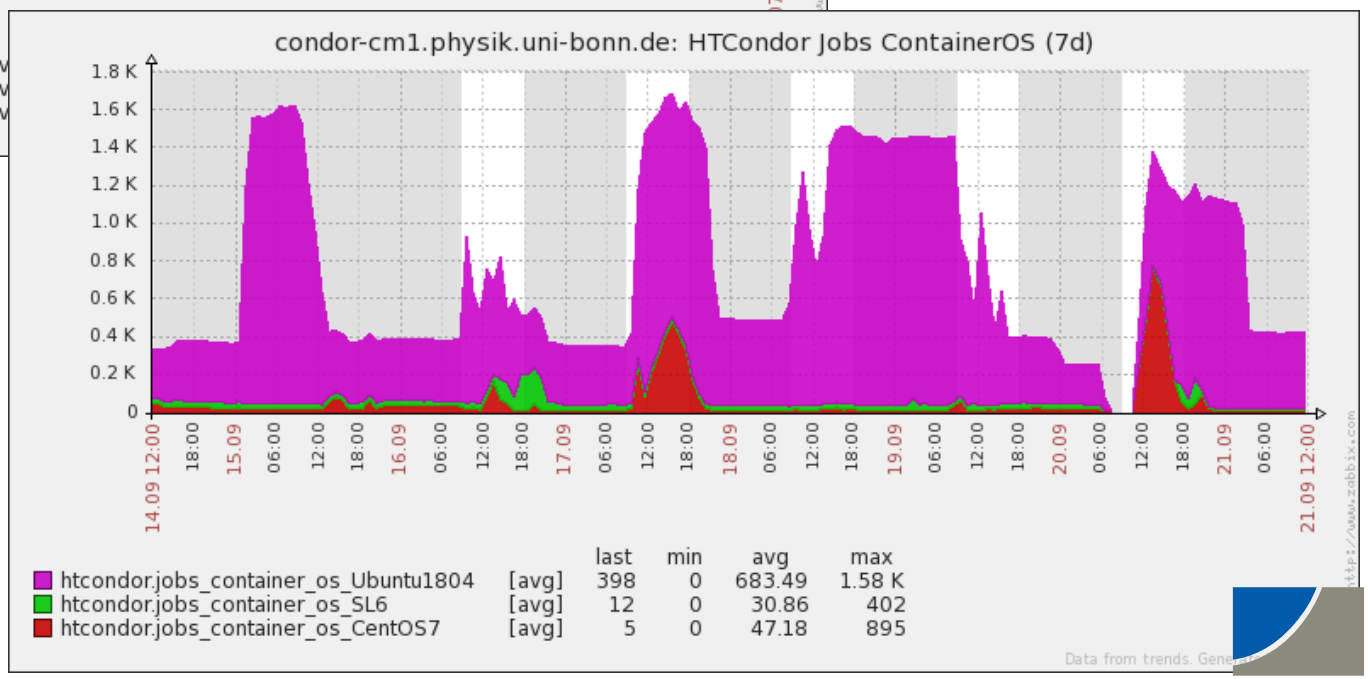
HTCondor 8.8

- **`sshd` now running outside of the container!**
- However, lots of issues in 8.8.0:
  - Too modern `nsenter` required (not in any LTS distro)
    ⇒ fixed in 8.8.2
  - Support for rootless broken
    ⇒ fixed in 8.8.2
  - Interactive jobs closed after 3 minutes
    ⇒ partially fixed in 8.8.3
  - Environment in interactive jobs / `condor_ssh_to_job` unset
    ⇒ maybe fixed in 8.8.5 (and have workaround)
  - Interactive jobs / `condor_ssh_to_job` do not get a `pty`
    ⇒ not fixed yet
- Now running 8.8.5 everywhere but startd machines (8.6.13)
  ⇒ This requires some dirty hacks (interactive jobs never close).
  ⇒ This causes jobs to die on short network connection loss.

Looking forward to future fixes making 8.8 usable for us!

Container Usage





Container Usage: Well accepted!

Instead of `ssh` to a login node, users run:

```
freyermu@exp199:~$ condor_submit -interactive -append
↪  '+ContainerOS="CentOS7"'
Submitting job(s).
1 job(s) submitted to cluster 1008.
/usr/bin/xauth:  file /jwd/.Xauthority does not exist
Welcome to sloti_2_2@wn004.baf.physik.uni-bonn.de!
You will be logged out after 7200 seconds of inactivity.
You requested 1 core(s), 512 MB RAM, 125 kB disk space.
freyermu@wn004(CentOS7) /pool/condor/dir_14973 $
```

- Well accepted by users.
- Rarely, new users still try to run SL 6 code on CentOS 7...
- No good way to run an IDE in the same environment (but this is also true for login nodes).

Conclusions

- New cluster setup works very well for us!
- Getting rid of login nodes solved a lot of issues and headaches
- HTCondor does a very good job and ClassAd system is extremely flexible both for administrators and users
- Containers with different software environments well-accepted and heavily used
- Still, we hit a list of bugs and hope for further improvement along the way...

Thank you!

Thank you

for your attention!