Updates from JAliEn

Site services on v2.0.0, GPU and new features

Maxim Storetvedt | US-ALICE Grid Operations Review | Sept.17, 2025

Evolving JAliEn

- Milestone release, **JAliEn 2.0.0**, tagged on **14th** of **April 2025**
- Stark contrast to original **1.0.0** tag from **14th** of **June 2018**
  - Full role / permission separation via authentication tokens
  - Multi-layered, sandboxed, execution via containers
  - Compatibility matching
  - Sanity checks and execution protections
  - Isolation mechanisms, cgroups v2
  - 64bit, GPUs, multiarch
  - Fully migrated central services
- And several larger changes **leading up to v2.0**

Changes to Java and the JDK

- Previously
  - Source target: **JDK 11**
  - Runtime: **JDK 17** (LTS)
- Changes for v2.0 series:
  - Source target: **JDK 15**
  - Runtime: **JDK 21** (LTS)
- Source target needs to remain < JDK 17 for
  - To remain binary compatible when embedded with other applications
    - Some still depend on feature removed by Oracle (RMI) in JDK17
- All runtimes taken directly from **CVMFS**, independent of *alienv*:

```
/cvmfs/alice.cern.ch/java/JDKs/<arch>/jdk-latest
```

Changes to Java and the JDK

- Previously
  - Source target: **JDK 11**
  - Runtime: **JDK 17** (LTS)
- Changes for v2.0 series:
  - Source target: **JDK 15**
  - Runtime: **JDK 21** (LTS)
- Source target needs to remain < JDK 17 for
  - To remain binary compatible when embed
    - Some still depend on feature removed by
- All runtimes taken directly from **CVMFS**, i

`/cvmfs/alice.cern.c`

.... and now subscribed to Azul's JDK reports to quickly catch CVEs

Current production JDK:
`zulu21.44.17-ca-jdk21.0.8-linux_x64`

Removal of Docker

- No longer any Docker dependencies in production
- VOBox now available as a self-contained **Apptainer** instance
  - Also compatible with ***Podman***

https://gitlab.cern.ch/jalien/jalien-jobcontainer

- May also run directly from CVMFS, using the same binaries as for Jobs

```
/cvmfs/alice.cern.ch/containers/bin/apptainer
```

- **Note**: Apptainer version remains at **v1.3.2** for now
  - Last version to also support EL7
  - To be updated once last EL7 sites migrate, or if later security patch needed

Removal of Docker

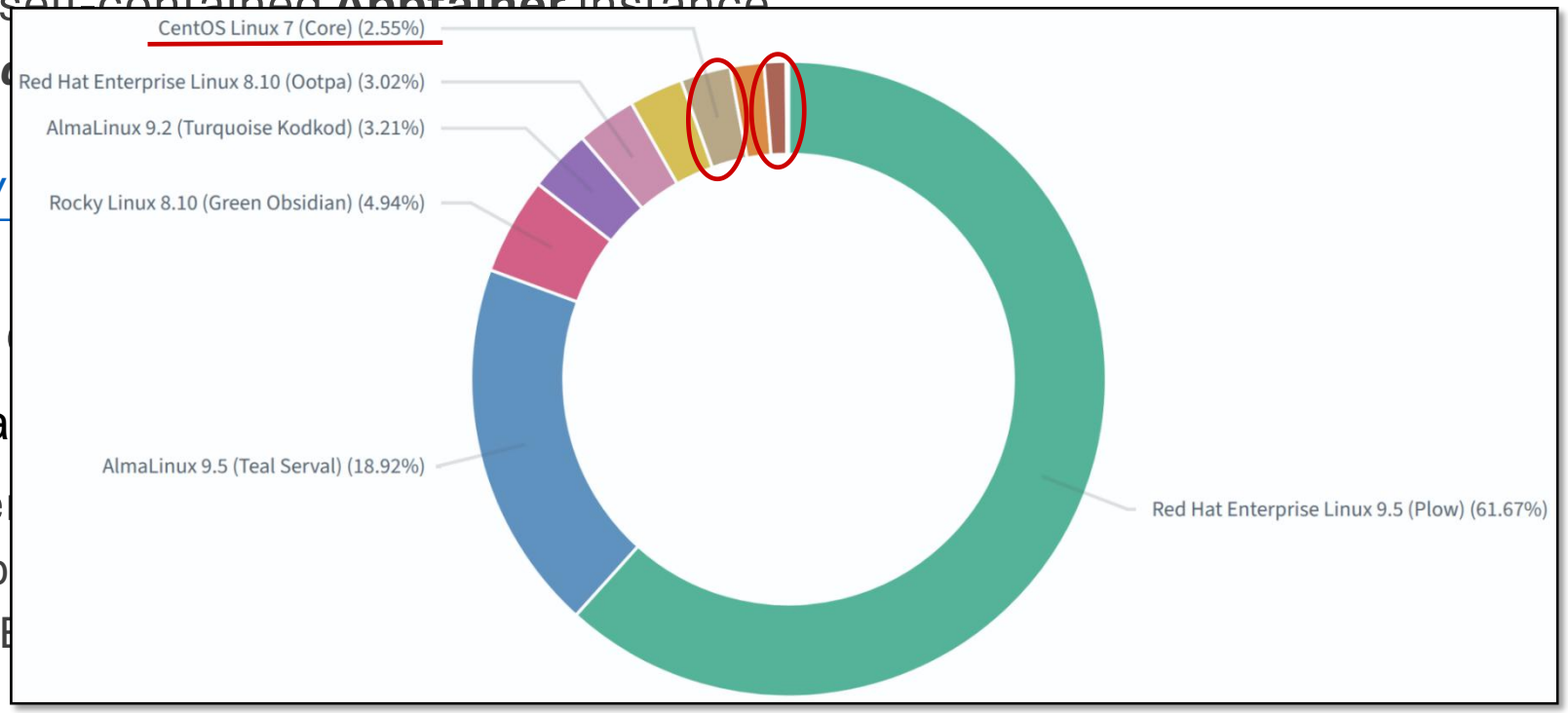
- No longer any Docker dependencies in production
- VOBox now available as a self-contained **Apptainer** instance
  - Also compatible with ***Pod***

https://

- May also run directly from

/cvmfs/a

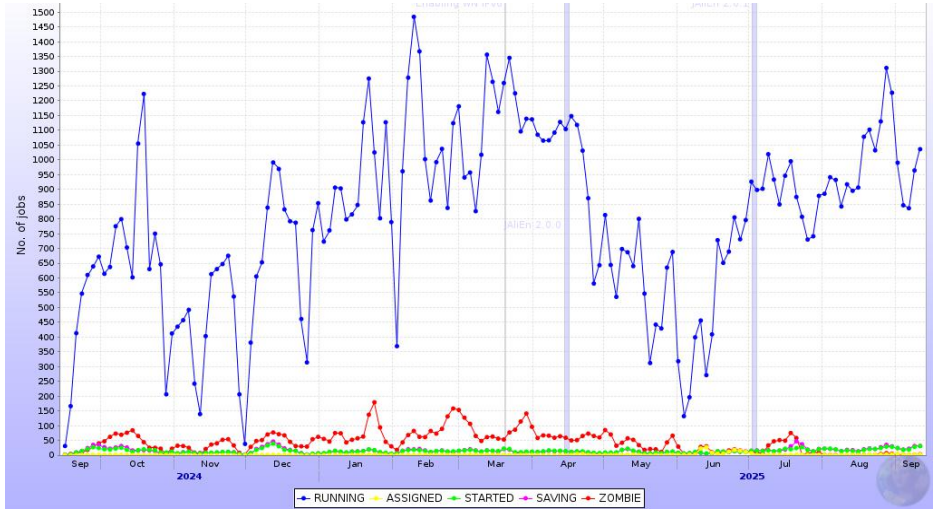
- **Note**: Apptainer version re
  - Last version to also supp
  - To be updated once last E



EL7 is now at **<4%!**

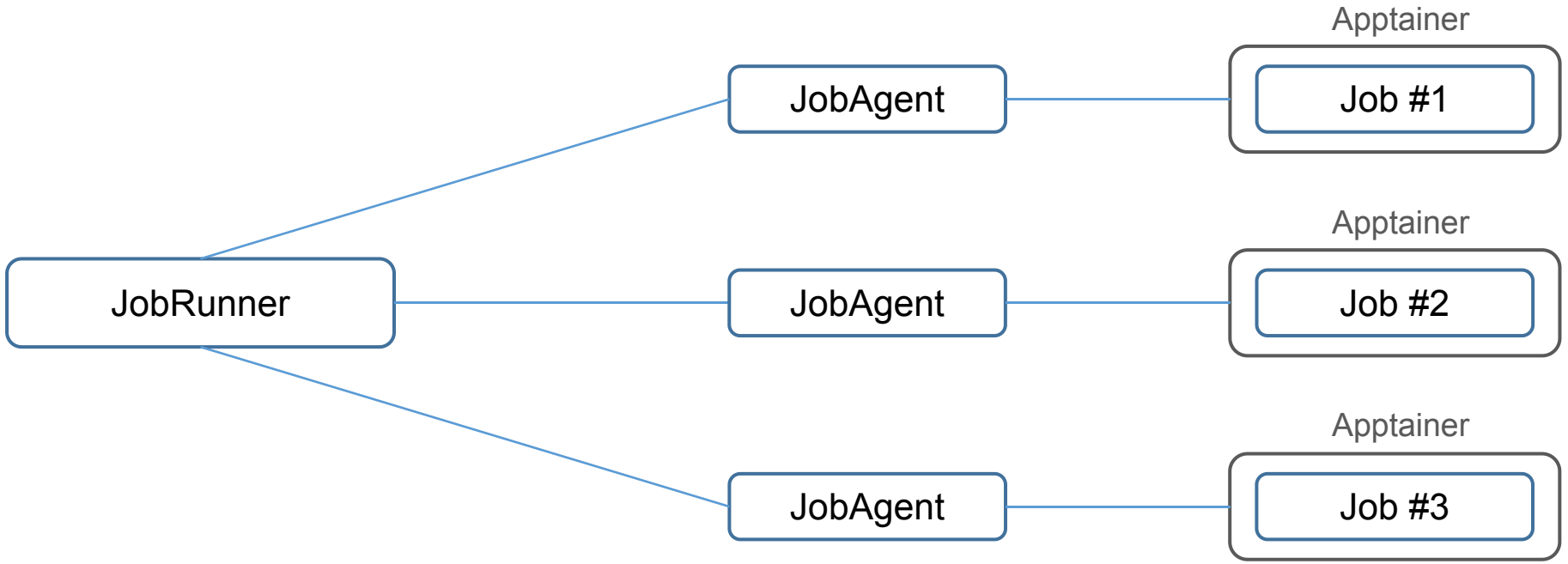
Towards running jobs in Podman

- For ~1 year, KISTI was plagued by jobs constantly going to ZOMBIE

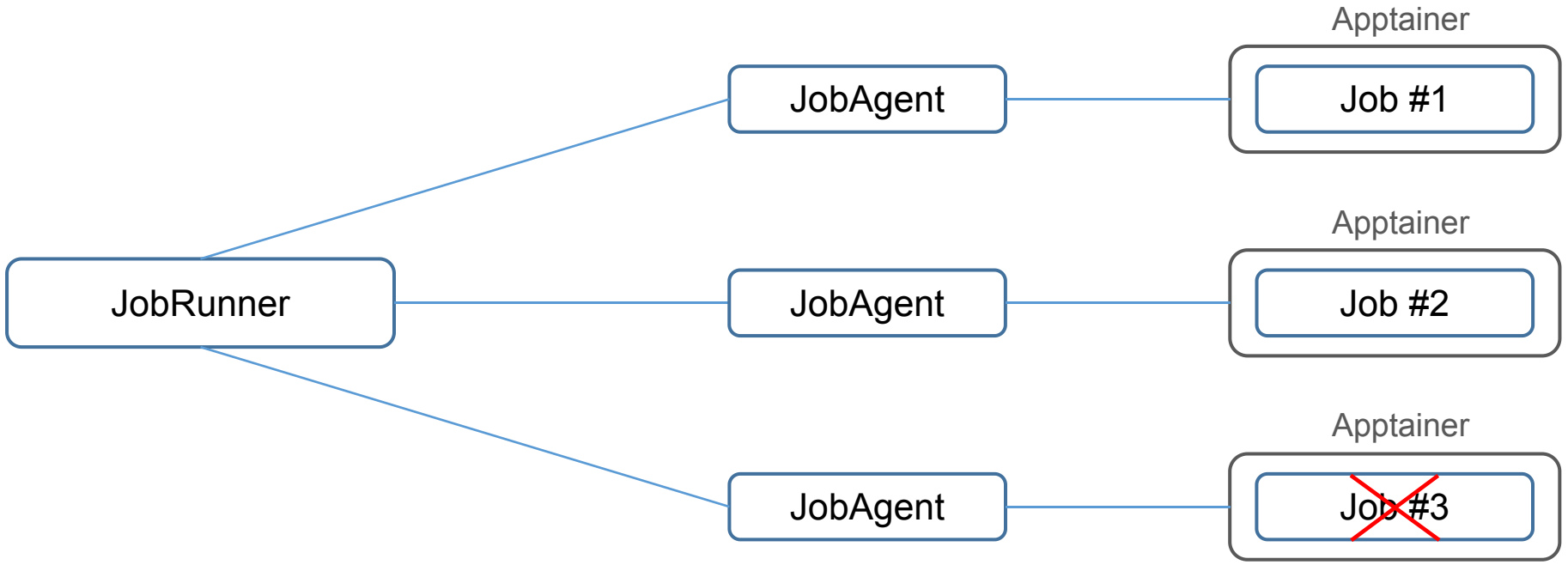


- Investigations revealed connection issues, but also **misbehaving jobs**
  - Several production jobs were calling `kill -9 0` when terminating
  - The signal would propagate down the process tree, and eventually reach the JobRunner
    - Forcing not only the JobRunner to abruptly exit, but also **all other jobs in the slot**
- Reason: apptainer runs every jobs **in the same PGID**, despite `containall` being set

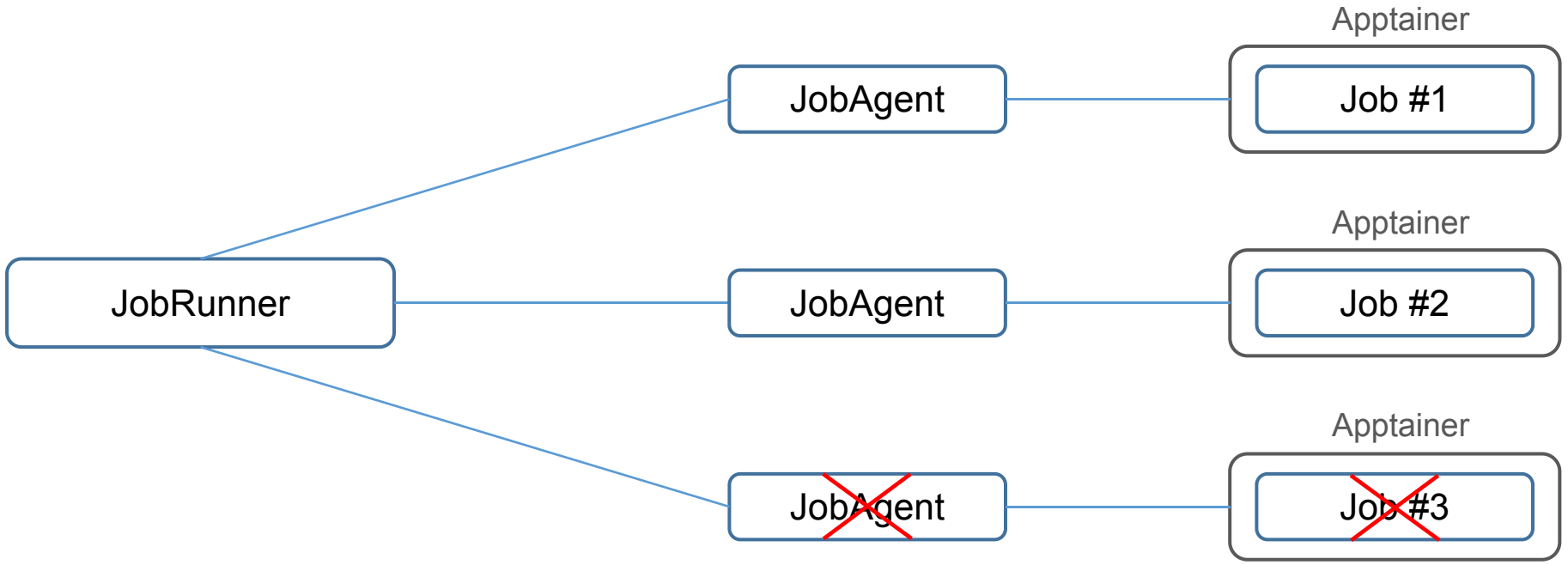
When jobs misbehave...



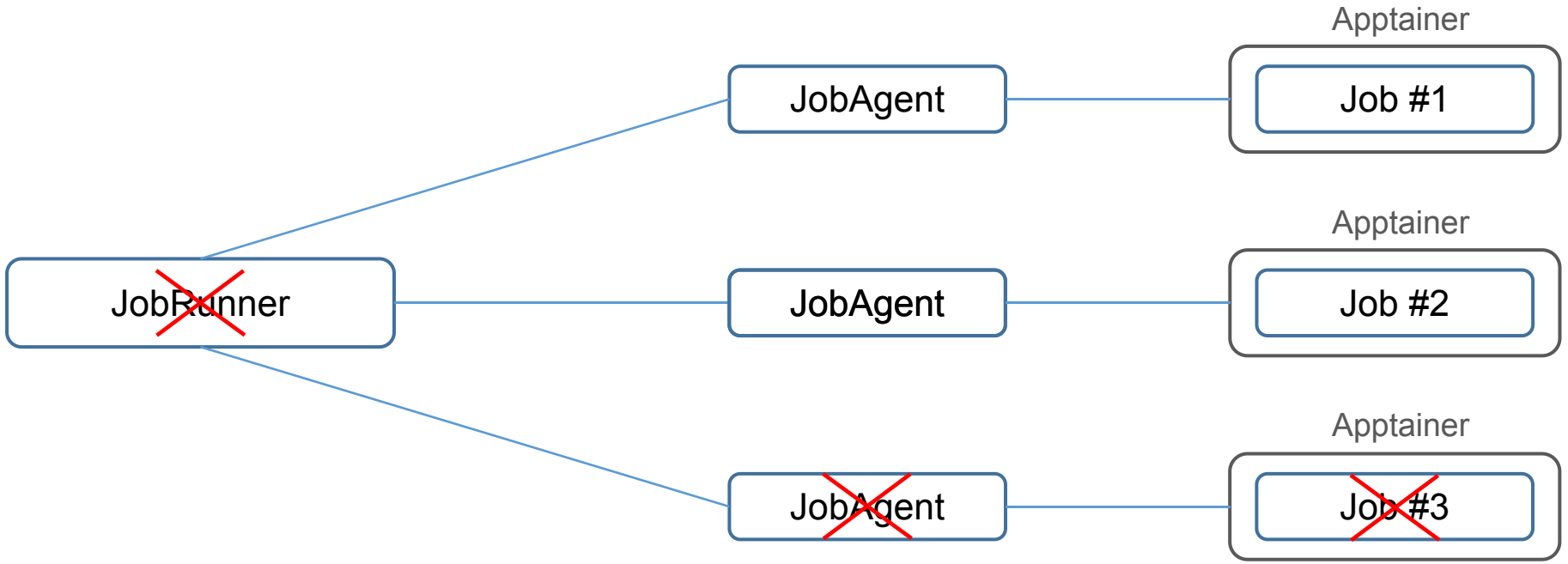
When jobs misbehave...



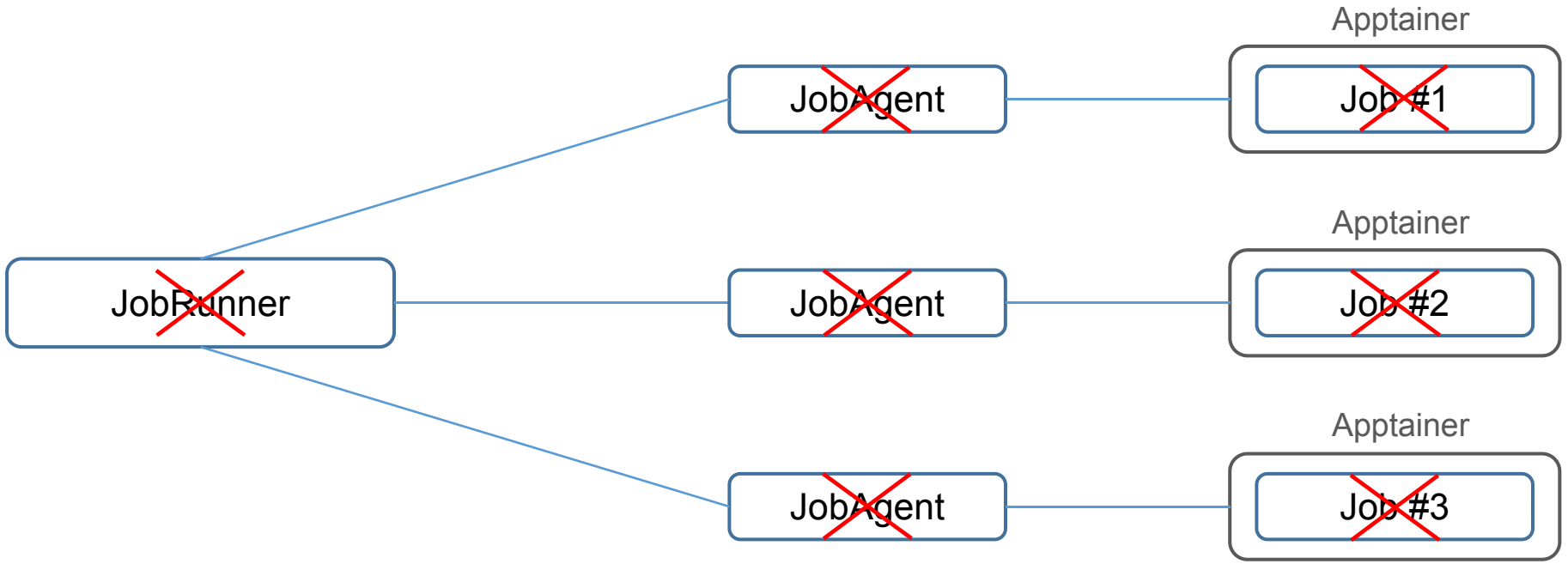
When jobs misbehave...



When jobs misbehave...

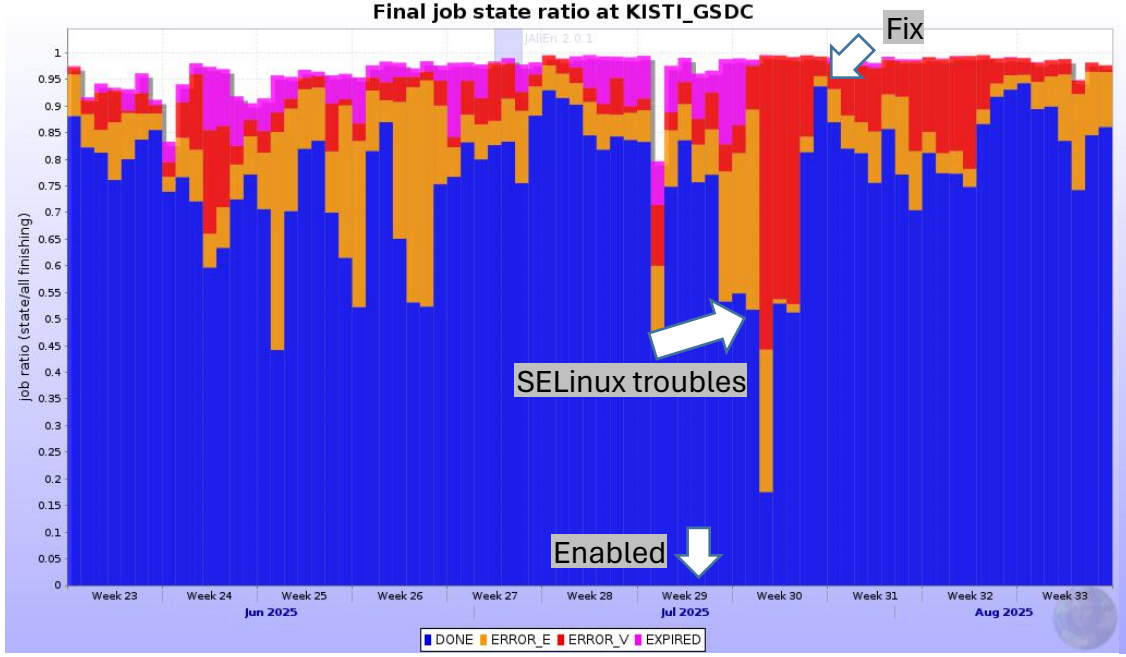


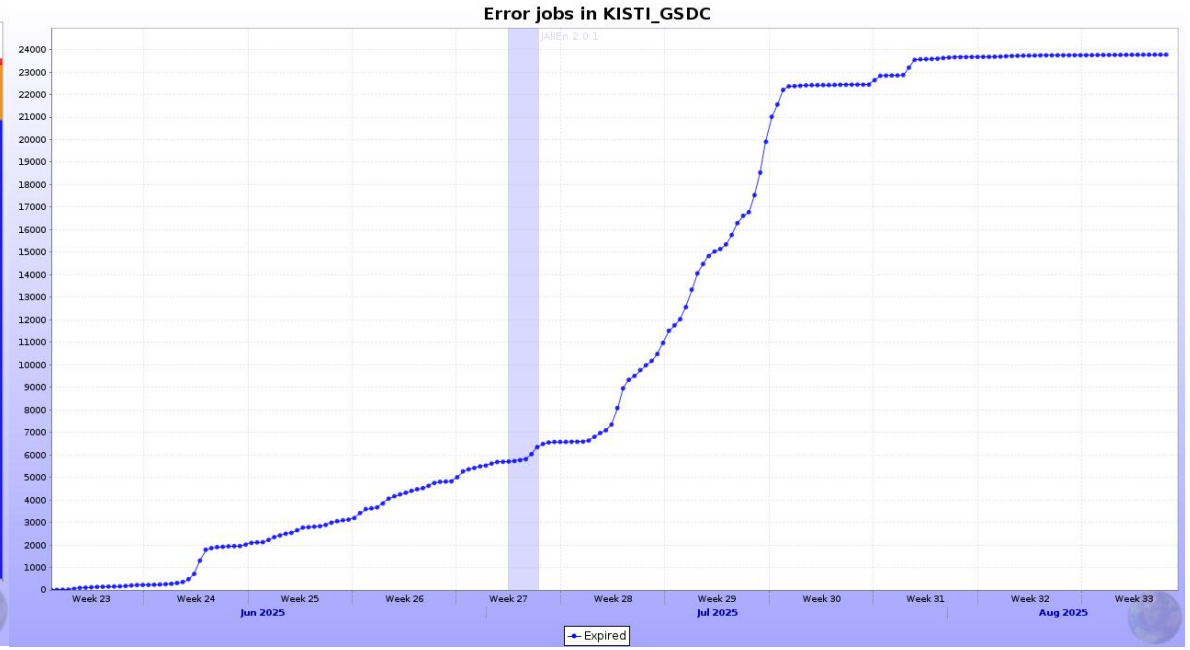
When jobs misbehave...



Towards running jobs in Podman (2)

- Problem solved by changing containers to **Podman**





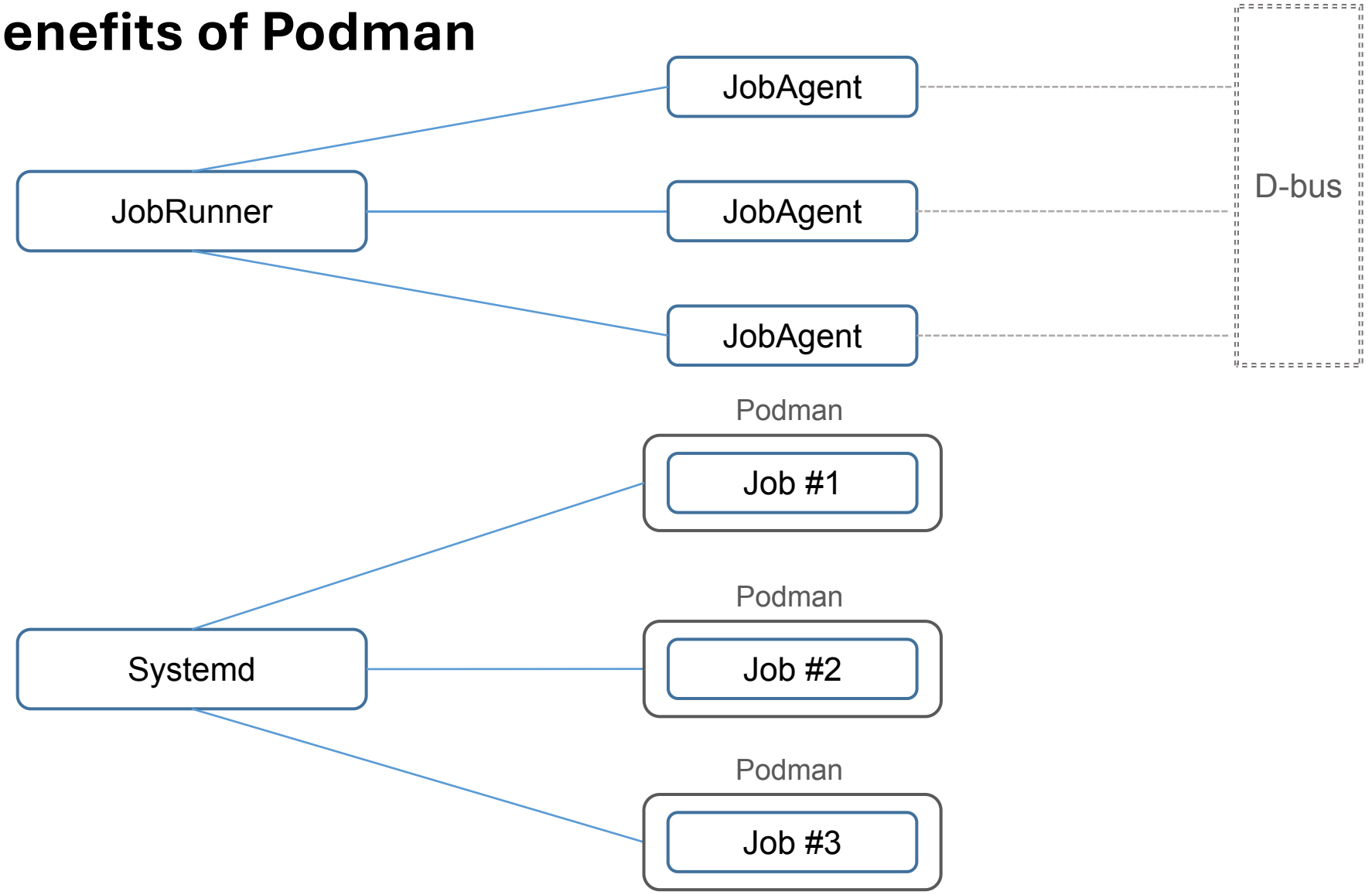
- Improved support for Podman coming in **JAliEn 2.0.3**
  - Includes fixes for shared memory, SELinux and monitoring

Benefits of Podman

- Unlike Apptainer, Podman provides **proper PGID isolation**
  - `kill -9` signals unable to propagate up the tree
- Podman has also an ***additional*** protection
  - The container/child process are not owned by the JobRunner/JobAgent!

podman

Benefits of Podman



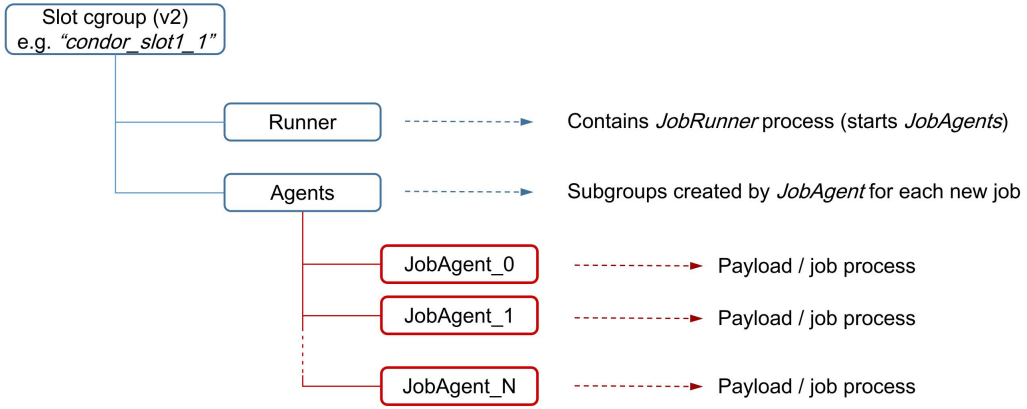
Benefits of Podman (2)

- Unlike Apptainer, Podman provides **proper PGID isolation**
  - `kill -9` signals unable to propagate up the tree
- Podman has also an ***additional*** protection
  - The container/child process are not owned by the JobRunner/JobAgent!
  - Kill signals will never reach the JobRunner
- Switching other sites have shown similar decrease in ZOMBIE/EXPIRED jobs
  - Hiroshima switched a few weeks after KISTI
- **LBL/ORNL** next potential candidates
  - Receive many of the same jobs with similar behaviour
  - Note: SLURM has limited Systemd integration, and requires **lingering** to be enabled
- Still not giving up on Apptainer completely
  - To be retried on latest upstream once no more CentOS 7 sites

podman

Support for cgroups v2 on Slurm

- Alternative approach discussed during T1/T2 in Seoul
  - Puts the Slurm slot cgroup in **user.slice**
  - Still requires **lingering** enabled for the running user
  - But no additional software/plugins/hacks needed!
- Included directly within JAliEn since **1.9.7**
  - Initially for **whole node** and **Slurm 22.05+** only
    - Non-whole sites from **JAliEn 2.0.1**
- Derived from previous workaround on VOBox (now fully in CE):




```
export XDG_RUNTIME_DIR=/run/user/$(id -u)
export DBUS_SESSION_BUS_ADDRESS="unix:path=${XDG_RUNTIME_DIR}/bus"

export DISABLE_NESTED_CONTAINERS=true
export META_VARIABLES='DISABLE_NESTED_CONTAINERS'

export JALIEN_JOBAGENT_CMD='/cvmfs/alice.cern.ch/java/JDKs/x86_64/jdk-latest/bin/java -client -Xms16M -Xmx128M
-Djdk.lang.Process.launchMechanism=vfork -Dalien.config.ConfigUtils.loggingConfigFile=remote_logging -XX:
+UseSerialGC -cp /cvmfs/alice.cern.ch/java/jar/jalien/alien-users-20240924-1554.jar alien.site.JobRunner'

export JALIEN_JOBAGENT_CMD_FULL="/cvmfs/alice.cern.ch/containers/bin/apptainer/current/bin/apptainer run -B /
cvmfs:/cvmfs,/tmp:/tmp,/sys:/sys,/scratch:/scratch --memory 700G /cvmfs/alice.cern.ch/containers/fs/
singularity/el9 /bin/bash -c $JALIEN_JOBAGENT_CMD"

eval $JALIEN_JOBAGENT_CMD_FULL
exit $?
```

Support for cgroups v2 on Slurm (2)

- Additional improvements (included in **2.0.1**)
  - Startup needs to connect to dbus for moving processes to `user.slice`
  - May occasionally fail

```
(output: "Failed to connect to bus: No such file or directory\n"): exit status 1
```

  - Needs to be more **reliable** as monitoring has shifted to cgroups v2
    - More on this in Marta's talk
  - Fallbacks to v1 mode exist for now, but will not be maintained forever
  - **Solution**: will now instead retry if initial call fails
    - Solved issue on UiB where it was first observed

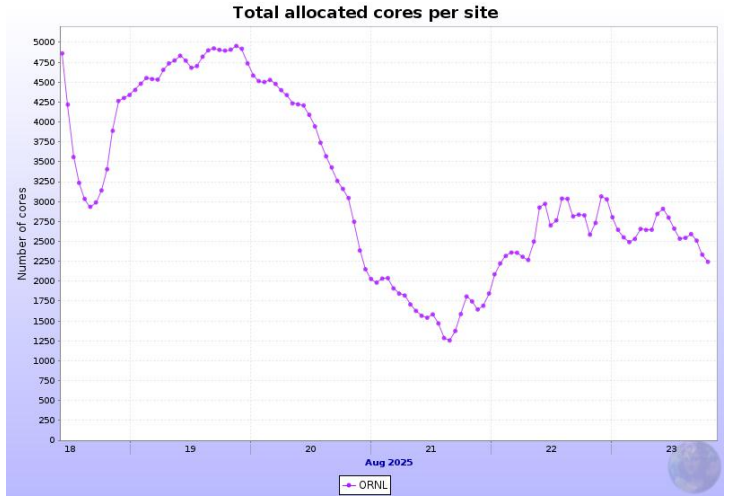
Running jobs with cgroups v2 on ORNL

- Lingering now **enabled** on ORNL, allowing cgroups v2 partitioning on Slurm
  - First v2 jobs started August 19

Trace of job 3393853519

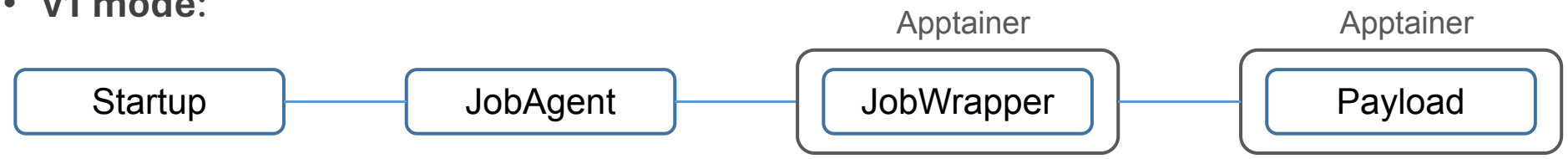
Aug 19 21:10:01 [trace]: Job inserted by **aliendb7.cern.ch** [Masterjob is 3393853223]
Aug 19 21:10:01 [state]: Job state transition to WAITING
Aug 19 22:06:13 [state]: Job ASSIGNED to: ALICE::ORNL::ORNL
Aug 19 22:06:13 [trace]: This job has requested packages available on the following platforms: [el9-x86_64].
Aug 19 22:06:13 [trace]: Slot memory hard limit set to 3.94318848E8 kB memsw hard limit set to 0.0 kB by cgroup configuration.
Parsed cgroupV2: /sys/fs/cgroup/user.slice/user-501.slice/user@501.service/user.slice/apptainer-1852825.scope
Aug 19 22:06:13 [trace]: BatchId CONDOR_PARENT_ID: slurm9r:2214:1754601796

  - But had to be **disabled** again before long, as the site was operating at only “half” capacity

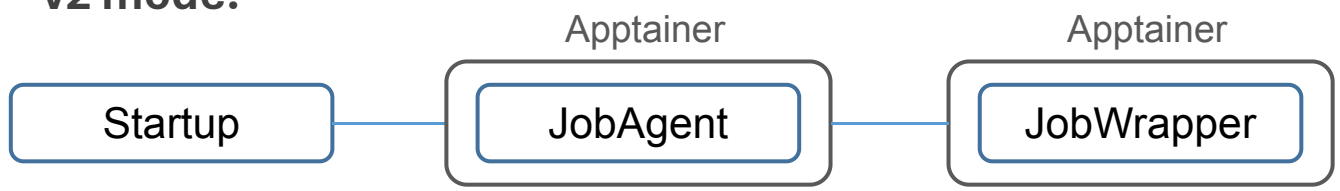


Running jobs with cgroups v2 on ORNL (2)

- From debug: many jobs were no longer matching due to insufficient **storage** (workdir size)
- Comes from the container hierarchy when on cgroups v2 in Slurm:
  - **v1 mode**:



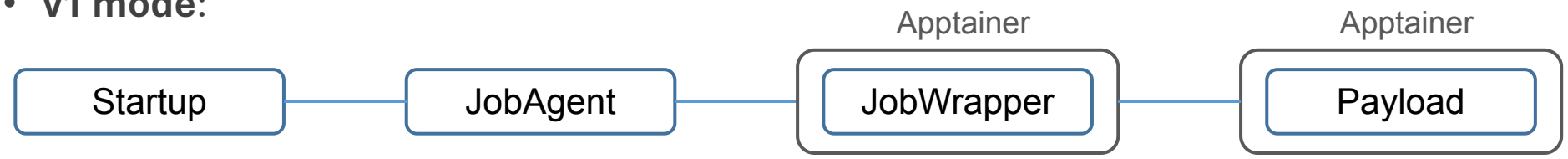
  - **v2 mode:**



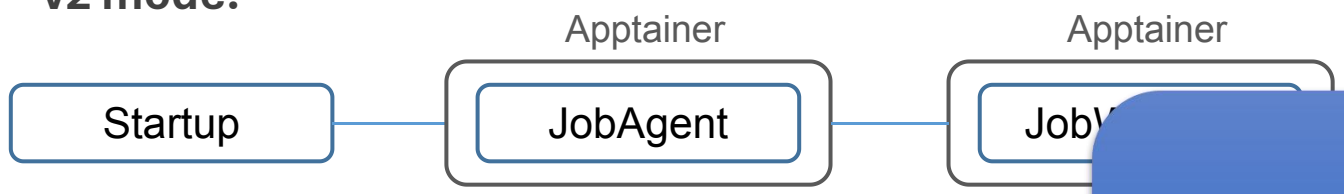
- In other words jobmatching happens from **inside** the container
  - But the working directory **changes** after Apptainer starts!
  - And it is the size of *this* directory that is passed for matching
    - Even if the correct directory is used when the job finally starts

Running jobs with cgroups v2 on ORNL (2)

- From debug: many jobs were no longer matching due to insufficient **storage** (workdir size)
- Comes from the container hierarchy when on cgroups v2 in Slurm:
  - **v1 mode**:



  - **v2 mode:**



- In other words jobmatching happens from **inside** t
  - But the working directory **changes** after Apptainer s
  - And it is the size of *this* directory that is passed for r
    - Even if the correct directory is used when the job finat

Fix being tested:
Explicitly set workdir in Apptainer to match that of LDAP

Running jobs with cgroups v2 on ORNL (3)

- **Potential fix:** explicitly set Apptainer cwd to match that of LDAP
  - But containers seemingly failed to start when including the appropriate flag
    - SELinux complaint or invalid dir(?)
    - No logs available
  - Manually retried at ACAT with Irakli
    - …but this time, it ***worked(?!)***
    - Correct storage size and working directory reported by SiteMap
  - Cgroups v2 to be **re-enabled** at ORNL under supervision for a new test run
- **Side-note**: status on **LBL**
  - Majority of site down due to earlier power work
  - But should have **lingering** enabled once back
  - Should be possible to enable running with cgroups v2 for the site
    - **EL8**, but v2 is enabled. Slurm also 22.05+

Improving cleanup

- **Process** cleanup
  - Attempts kill using Java → fallback to script in CVMFS if fails
    - Allows separating between SIGKILL/SIGTERM
    - Several safeguards in place against “kill loops”, in case wrong process is attempted killed
  - Will trigger payloads own cleanup first
    - 60s wait before forcibly killing
    - Default value can be extended in JDL up to 10 minutes (if needed)
      - Via `ForcedKillTimeout`
- **File** cleanup
  - Attempted file cleanup using Java
    - Fallback to cmd in case of IO errors
    - Always fallback in case of errors
  - Cleanup directories added to trace

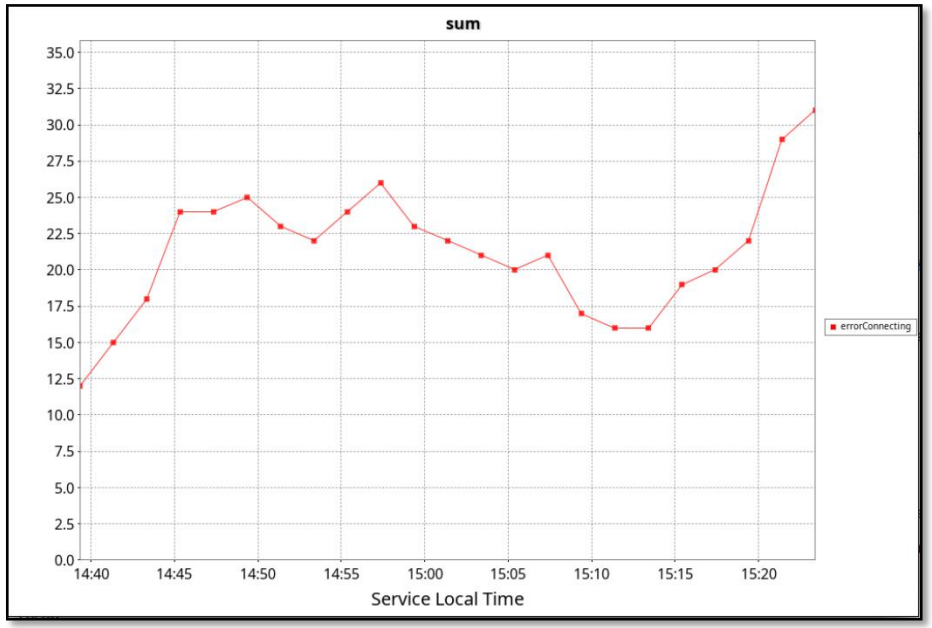
```
Oct 29 13:19:11 [trace ]: JobWrapper exit code: 0
Oct 29 13:19:11 [trace ]: Cleaning up after execution...
Oct 29 13:19:11 [trace ]: Deleting directory /var/lib/condor/execute/dir_1819317/alien-job-3193729
Oct 29 13:19:12 [trace ]: Done!
```

Multiarch improvements

- **Mixed** clusters are now supported
  - One site for multiple architectures (e.g. x86_64, aarch64)
  - Binaries checked/matched on the agent side instead of VOBox
- Support for **additional** architectures (`riscv64`)
  - Others can be “slot-in” as needed by providing compatible JDK/container in CVMFS
- Stability improvements
  - Prevent crash attempting x86 specifics on `aarch64`
- No more MacOS on x86_64

Heartbeat monitor “2.0”

- Originally, separate process monitoring the **main loop**
  - Warns in case something on the host is blocking its execution
- Extended to warn when **connection** is failing
  - Also report to ML (via the VOBox) in case traces are lost
  - Flagged as “**errorConnecting**”
- Additionally, warn if specific hosts have **problems executing** jobs
  - Avoid WNs occupying slot while idling
  - In this case WN flagged and reported to ML as “**nodeBad**”
  - Includes check if **CVMFS** is falling behind on individual workers



Towards GPU brokering and resource controls

- Currently required to match against **site** if needing GPU resources
- Should be changed to **match on GPU** resources just as for CPU
  - With possibility to match against specific GPU model via JDL
  - E.g. `GPU="Radeon Instinct MI50 32GB"`
- Also, introduce resource control for GPUs on **whole-node** slots
  - Avoid multiple jobs claiming the same GPU
  - Block access if already claimed by another job
- However…
  - **Compatibility** remains recurring issue, e.g. EL8 -> EL9 containers not supported
  - Not to mention there is no standardised way for reading GPU information **across vendors**

Towards GPU brokering and resource controls (2)

- New SiteSonar probe added to list GPUs on system across vendors:

`/cvmfs/alice.cern.ch/sitesonar/sitesonar.d/gpu.sh`

- Uses **`lspci`** and vendor ID classes to list all relevant devices

```
for CLASS in 0300 0380 0302 0301; do
        (lspci -d $VENDOR::$CLASS -mm) | cut -d\" -f3,4,6
done
```

- Fallback to `lspci` via **Apptainer** if unavailable on system
  - And use Apptainer to block other jobs from accessing GPU resources…

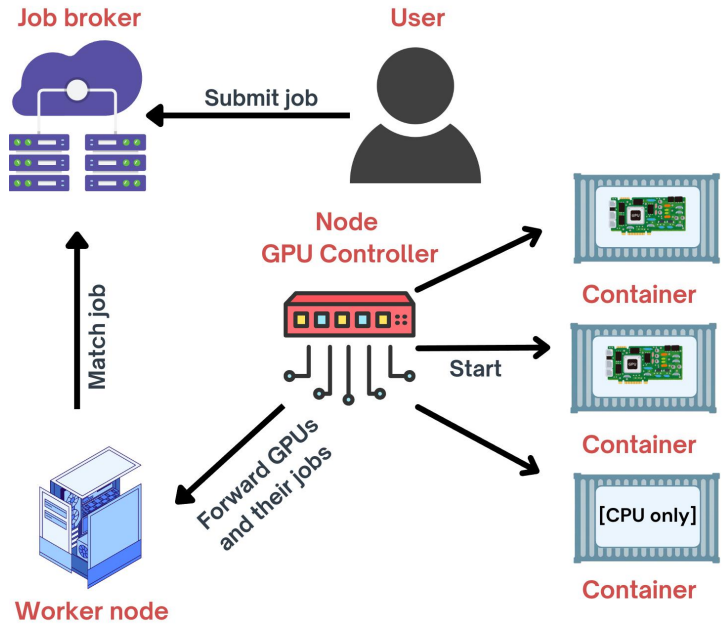
A GPU controller for JAliEn

- Now that we have a universal way of reading system GPU info, let's use it!
- Introduction of a new **GPU controlle**r extension to JAliEn
  - Provides GPU information for **brokering** decisions alongside the existing resource monitoring
  - Additionally, track and **allocate GPU resources** on each WN to jobs requesting them
  - Performs a "**sanity check**" before assigning a device, to ensure the GPU workload is compatible with the available drivers / host OS
  - Uses **containers** to expose GPU resources only to jobs assigned to them

Match fields
Vendor
Device model
GPU Architecture
Availability

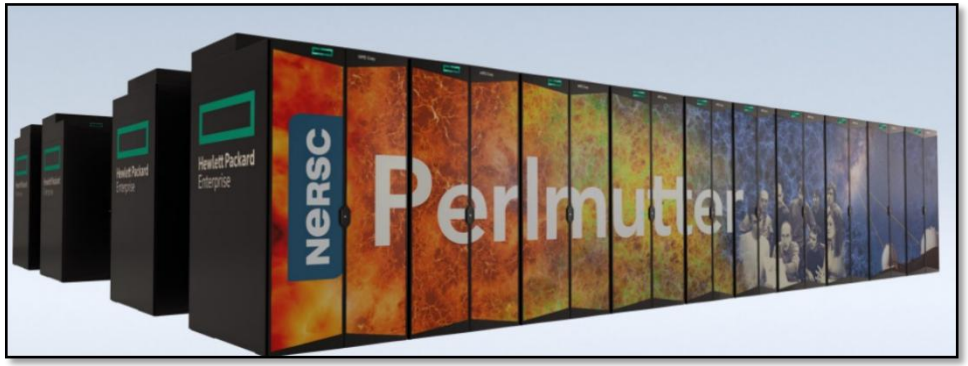
A GPU controller for JAliEn (2)

- Workflow:
  - User jobs are **centrally brokered** to match suitable nodes
  - Broker examines SiteSonar, alongside **SiteMap** provided by each node
  - Each map is supplied with information by the **GPU Controller**
  - Controller maintains **which jobs**, if any, have **reserved GPU(s)**, and on **which WN**
  - Once a job is assigned, a **container** instance **exposing only the given GPU** is provided, **blocking** other jobs from accessing it
  - Provides **all libraries needed**, and runs a sanity check within the container before job execution
  - If no more GPU resources, remaining job containers will be **CPU only**



Matching GPUs on Perlmutter

- The new GPU controller will be able to pick up Perlmutter's **Nvidia GPUs**
  - If there are GPU slots available
- Users can specify **vendor** in JDL, or match on **GPU architecture**
- Initial tests by David successful using O2 GPU builds
  - Sanity checks will **pass**, despite EL9 builds + container on top of SUSE Enterprise 15
  - Tested on 1GPU+64core+220GB with 2024 Pb-Pb async reco
  - **Similar GPU speedup expected** as on **EPNs**, since same fraction of CPU workload can be offloaded and A100 GPU fast enough
  - Universal EPN + Perlmutter builds possible



More to come...

- Initial GPU brokering set for **JAliEn 2.0.4**
- With GPU monitoring to come after
  - Relies on `amdgpu_top/nvidia-smi/nvtop`
- Also still to come
  - Display more **fine grained** codes beyond current killReason set
  - Less time needed to dig through logs for common issues
    - In some cases, log access may be limited
  - Additionally, the following job states are still **unused**:
    - FAULTY
    - INCORRECT
    - ERROR_VER

```
OK(0, "Finished OK or no kill decision taken"),
OOM(10, "JW JVM out of memory"),
TTL(20, "Job running for longer than the TTL"),
KILLED(30, "Job was killed"),
OOM_PREEMPTED(40, "Killed by JAliEn due to overuse of memory"),
IDLE_CPU(50, "Payload idling for more than 15 minutes"),
CORE_DUMP(60, "Core file detected"),
DISK_USAGE(70, "Too much disk space used"),
MEMORY_USAGE(80, "Too much memory used"),
OOM_SYSTEM(90, "Killed by executing system due to overuse of
memory"),
SANDBOX_CREATE(201, "Cannot create the sandbox directory"),
INPUT_FILES(202, "Failed to download input files"), or it
returns an error
PACKAGES(203, "Cannot set up the packages environment"),
JOBBROKER(301, "Cannot assign the job");
```

Other key changes in the meantime

- **General improvements**
  - Remove timestamp duplicates
  - Switch to ‘localhost’ instead of ‘127.0.0.1’
    - For IPv6 only
  - Use WN memsize as Slurm top limit, instead of hardcoding
    - Otherwise interferes with monitoring
  - Bump dependencies (TomCat, LazyJ, BouncyCastle)
  - Bump test images to include JDK17
    - And avoid Docker Hub
  - Remove deprecated methods for JDK11 --> JDK15
  - Adjust monitoring clock based on CS
  - Do not attempt upload for killed jobs / abort immediately
    - Would otherwise start preparing upload
  - Tuned upstream idle connection timeout for JBox / Wrapper
  - Central fixes for DB queries to reduce contention and recovery from it
- **New features**
  - Allow disabling checking for core files via JDL
    - Via AbortOnCore=true/false
  - Option to temporarily disable cgroups v2
    - For debugging only, as upcoming monitoring will depend on it
  - Oversubscribed cores computation merged
    - From Marta’s branch
- **Fixes**
  - Prevent some trace strings from being flagged as errors
  - Fix potential NPE when fetching JDL flag
  - Fix memory leak on long-running connections
  - Fix leak on long running clients
  - Fix potentially stuck agent when large masterjobs killed centrally
    - Issue seen on ORNL
  - Fix for moving oversubscribed job to regular pool
  - Fix JW reporting to the correct cluster name
  - Use correct timeout value for killed jobs
  - Fixes to the split-by-input-size constraint
  - Fix bug reading from broken field on Slurm sites
  - Remove misleading message in case of OOM kill
  - Prevent LDAP returning multiple values from voboxes with extended names
  - Central fixes for DB queries to reduce contention and recovery from it
  - Fix CE bug where certificates would be renewed too late
    - Persistent connection preventing cert reload, despite <48hrs remaining
  - Workaround Slurm dbus issues on Apptainer

Summary

- **JAliEn** has reached milestone release **2.0**
  - Major changes since its initial 1.0 tag in 2018
  - Complete independence from **Docker**
- Includes several **JDK** upgrades
  - Source target will remain at **<JDK17** for now
- Further improvements to **cgroups v2** support
  - Whole-node no longer requirement for Slurm
  - More reliability fixes for **2.0.1**
- Further improvements to multiarch support
  - With support for **mixed resource** clusters
- **Cleanup** allowing for graceful shutdown of payloads
- Heartbeat monitoring for **connection**
  - With **ML** integration and tagging of **bad WN**s
- With more **changes** on the way
  - Fine-grained error codes
  - **GPU Brokering**
    - But more reliable means for identifying GPU resources needed
      - Dedicated GPUController class(?)