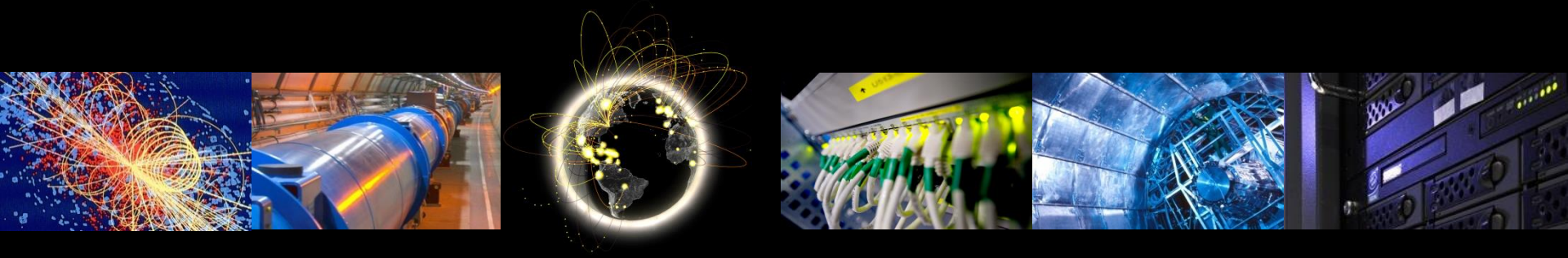
Traceability & Isolation WG update

Vincent BRILLAULT, CERN/EGI-CSIRT

WLCG Traceability and Isolation WG, July 2019



WLCG
Worldwide LHC Computing Grid

CERN

A bit of history…

- Working group created in March 2016. Goals:
  - Explore new paradigms, esp. containers
  - Maintain traceability level with ongoing changes
    - VMs, containers, gLExec deployment frozen, etc
- Technology quickly identified and validated
  - Containers, in particular *Singularity*, validated
  - Container WG created to follow deployment
  - Already used in production!
- Agreeing on a policy proposal took longer…

Agreeing on a policy…

- Technical uncertainty:
  - How to deploy containers? Dedicated WG
  - Can Singularity run within Singularity? Yes* if ≥ 3.2
- Covering all sites & workflows:
  - HPCs
  - Sites running Singularity internally
  - Different VO workflows & approaches
- No immediate operational benefits, less interest…

WG Current situation

- Two documents produced by the working group
  - Both now agreed upon by all participants

- Recommendations:
  - Recommendations for a new policy (VO pilots)
  - Isolation recommendations, with examples
  - Traceability recommendations

- Current Status:
  - What's the current situation and plan for every VO

Isolation recommendations

- Files: user payloads MUST NOT be able to steal or manipulate local files of other users' payloads, or files from their caller (e.g. Pilot) if those contain credentials or other means to affect other users' payloads.
- Processes: user payloads MUST NOT be able to manipulate (attach to, inject into, etc) processes of other users' payloads, or their caller (e.g. Pilot) if those contain credentials or other means to affect other users' payloads

Isolation recommendations

- Very high level recommendations
  - Designed to be technology agnostic
  - Also compatible with gLExec
- No dependency on user credentials
  - No certificates, no Argus, no emergency suspension
  - Required for storage accesses (and direct submission)
- Two scenarios interpreting recommendations:
  - Using containers to isolate user payloads from VO pilots
  - Using VO pilots without VO framework capabilities

Privileged VO pilot, isolated payload (like gLExec)

- [MUST] File: dedicated Mount namespace
  - Only exposes part of file system used by payload
  - Not exposing any pilot or other user files
- [MUST] Processes: dedicated PID namespace
  - Hiding any other process but the payload itself
- [SHOULD] dedicated IPC namespace
  - Further process isolation (might clash with MPI)

Using VO pilots without VO framework capabilities

Unprivileged and isolated pilot

- Pilots MUST be isolated from each other
  - Similar to isolation mentioned before
- Limited Pilots credentials
  - No capability to pull new job of a different user
- Shared file systems precautions
  - E.g. no shared writeable folder

Isolation Scenarios: VOs

- ALICE plans:
  - Use containers to isolate payload from pilot
    - With dedicated token for the payload itself
  - When not possible: production workload only
- ATLAS plans:
  - Use containers to isolate payload from pilot
    - With new user robot certificate for the payload itself
  - Sites without container capabilities: “Push mode”
  - Sites without network on WNs: no credentials

Isolation Scenarios: VOs

- CMS plans:
  - Use containers to isolate payload from pilot
    - With user credentials isolated from other users
  - HPC exception: production workload only
- LHCb plans:
  - Use containers to isolate payload from pilot
    - Efforts currently focused on running the pilot in Singularity, to provide a controlled environment (e.g. switching OS)
    - Isolating the payload itself will be a 2nd step

Traceability considerations

- "Payload" (VM, container, binary) complex
  - Sites usually cannot easily inspect them
  - Sites can only reliably monitor external activity
- Recommendations: split responsibilities
  - Site: Identify VO jobs from activity from their systems
  - VO: Identify user & payload from VO job details
- No overall changes in requirements
  - Same log retention periods
  - Same logs collected

Traceability Examples

- Expected queries for Sites:
  - Time, host/IP → VOs & local job IDs
  - Time, host/IP, activity (e.g. remote connection) → VO & local job ID
  - Time, VO, [local job IDs] → hosts/IPs
- Expected queries for VOs:
  - Time, host/IP → Users & payloads
  - Time, host/IP, local job ID → User & payload
  - Time, user → Sites, payloads, local job IDs

Incident Response & Emergency suspension

- VOs MUST take appropriate action
  - Within time frames in policies (TBD, same as sites?)
  - In case of delays, sites MUST NOT be penalized for suspending the whole VO
  - If sites propose a standard for validating users, VOs SHOULD make reasonable effort to use it
    - Not implemented yet, left open for sites to agree on a solution
- VOs SHOULD connect their pilot factories to central emergency suspension mechanisms

Next steps

- Documents now agreed upon within WG
  - More feedback, esp. from sites, welcome
- Documents presented to policy groups
  - Well received and accepted as task to work on
- Management Board approval
  - On WG outcome/recommendations?
  - For the final policy?
- Close the Working Group ☺

Conclusion – Take-away

- Designed a new traceability framework
  - Keeping full traceability of user actions
  - Adjusting to reality of pilot/payload transparency
- Designed a new isolation framework
  - Benefiting from new technologies
  - Decoupled from authentication/authorization
- Still need to be made into an actual policy
  - All VO planning to follow new recommendations
  - Already used in production (e.g. replacing gLExec)