Science & Technology Facilities Council

www.stfc.ac.uk

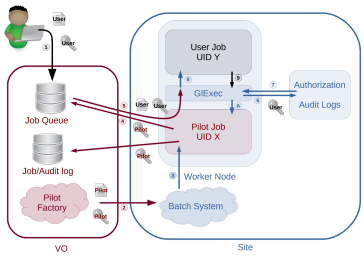
Developing the Traceability Model to meet the Requirements of an Evolving Distributed Computing Infrastructure

Ian Collier, STFC, Vincent Brillault, CERN, Brian Bockelmann, University of Nebraska

ian.collier@stfc.ac.uk

Introduction

- Security incidents are an operational reality in distributed infrastructure.
- *When* things go wrong we need, at a minimum, to know:
  - *WHO* did *WHAT*, *WHEN* they did it, and *WHERE* they did it.
- This allows us to contain the impact of incidents, preserve reputations and ensure that resources are available for their intended purposes.
- As our infrastructure evolves we should ensure we make the best use of all available technologies to maintain the traceability we depend upon.



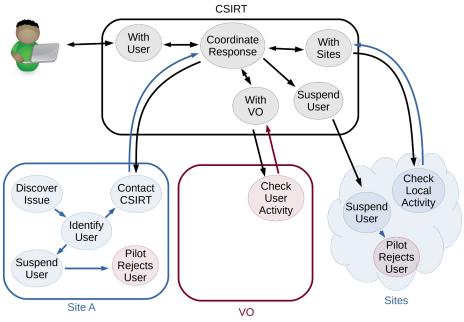
1. GLExec execution and traceability model

Current GLExec model

- GLExec manages authentication, authorization & isolation and logging
- UID changes can confuse batch systems:
  - Jobs of the same user might not be isolated, esp. for (shared) storage
  - Job tracking could get confused: payload running as a separate UID
- Exposes the final user to the site, allowing *direct* trust:
  - But rely on the VO to provide a proxy matching the payload!
- Cannot provide isolation without full authentication
- Complex to deploy & operate – adoption by VOs remains low

Separating Isolation & Traceability:

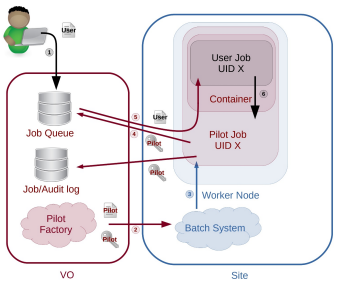
- Containers (namespaces) can provide isolation between processes (jobs):
- Unprivileged namespaces allow users to create their own containers:
  - No root privilege required, no SUID required
  - Users (and jobs) are isolated: one cannot another's container
- No need for a trusted gatekeeper within the site:
  - We already trust VO to provide matching of users and payload
  - VOs can produce enough audit logs for traceability
- Isolation has no inherent dependency on traceability
- Traceability does depend on isolation, but can be layered:
  - Batch systems isolate and track VO jobs
  - VO jobs can isolate and track VO jobs themselves



3. Current incident response workflow

Potential solution: Singularity

- Developed for HPC: http://singularity.lbl.gov/
- Unprivileged/non-daemonized tool creating containers:
  - Can provide complete isolation between the pilot and the payload
  - Can be used with CVMFS to provide a CernVM environment
  - Can mount over specific folders, e.g. job folders
- Completely unprivileged with upstream kernel:
  - No installation needed at sites, could be in CVMFS
- In RedHat/CentOS/SL 7.x requires root SUID:
  - Security assessment ongoing
  - Could offer a transition method - deployment more challenging



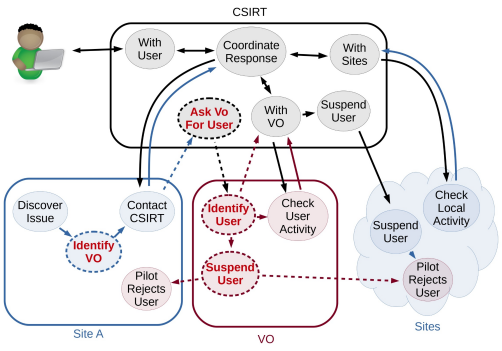
2. Proposed execution and traceability model

Simplifying site requirements

- Containers would not need special UID switch
- No need for global user to uid mapping
- Less dependencies on running environment:
  - Payload dependencies could be pulled from CVMFS
  - Payload operating system could be different from host (e.g SL6 on SL7)

VO role in Incident Response

- WLCG VOs already log job workflows to support debugging & workload management.
- Initial assessment shows that:
  - Enough data is stored to find the user/payload linked to an incident
  - Querying this data is possible but requires time and effort
- Better tools to aggregate & search workflow logs needed
- Traceability service challenges needed to test and certify VO capabilities
- Emergency credential suspension requires VOs
  - Automatic suspension feeds could be consumed by VOs
  - Sites can always suspend the entire VO as a last resort.



4. Proposed incident response workflow

Conclusions & Future work

- WLCG Traceability & Isolation Working Group will continue:
  - Investigating & testing possible isolation solutions
  - Working with VOs to develop incident response capability
- VOs may require new infrastructure to better support traceability
- New security challenges to be designed and tested

CERN

WLCG Worldwide LHC Computing Grid

GridPP UK Computing for Particle Physics