Security and VO management enhancements in Panda Workload Management System

Jose Caballero
Maxim Potekhin
Torre Wenaus

Presented by Maxim Potekhin at

HPDC08 – EGEE/OSG Workshop on VO management in Production Grids
Boston, June 24th 2008

Open Science Grid
Brookhaven National Laboratory

Outline

- Panda is a comprehensive Workload Management System that performs aggregation of job requests, allocation of Grid resources to requests according to pre-defined criteria, and tracking of job execution.

- Central to the concept of Panda is the use of pilot jobs which probe the environment on the remote worker node, before pulling down the payload job from the server and executing it. Such design allows for improved logging and monitoring capabilities, made possible by the pilot being a "smart wrapper" for the payload job.

- Recently, we enhanced the overall security of the Panda system by optionally allowing the pilot job to change UID via the use of gLExec wrapper. This is based on credentials of the end user (requestor) of the payload job, deposited on a caching service (MyProxy) and retrieved by the pilot prior to payload execution.

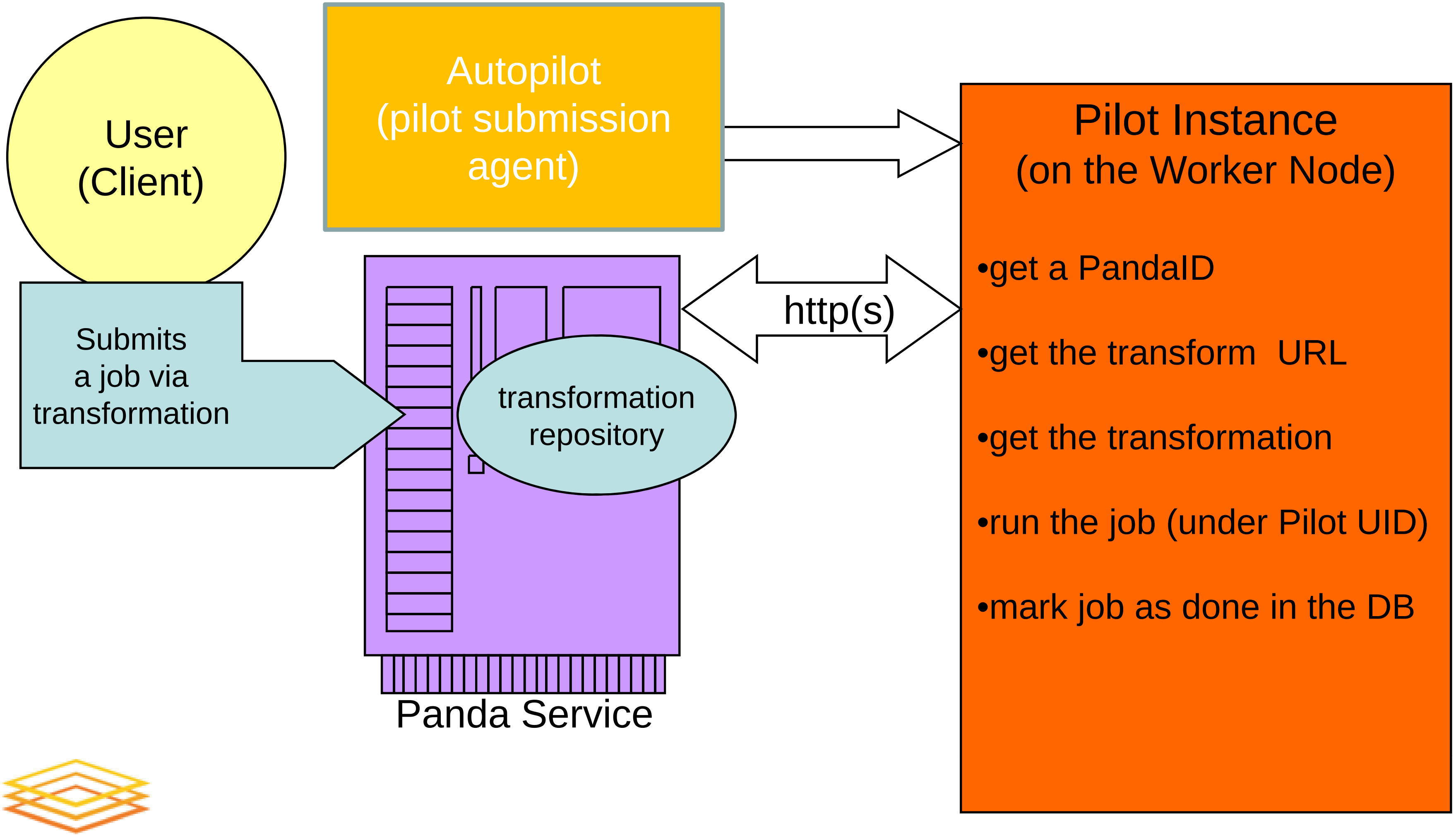
- In addition, work is under way to implement a resource allocation mechanism within Panda, based on user affiliation with Virtual Organizations (VOs) and resources allocated by individual sites to VOs.

The Panda job lifecycle

A brief outline of the job lifecycle within the Panda System **before** recent security enhancements:

- a user registers their job with the Panda Service (which may be comprised of more than one actual servers). This includes registering the payload definition (the “transformation” in Panda terminology), which is a URL of the script to be executed.
- separately, the “pilot” processes are created on the WNs of the target Grid sites.
- the pilots probe the environment and contact the Panda Service via https, requesting assignment of a job
- they receive the URL of the transformation script (job payload script)
- the transformation script is downloaded by the pilot from the Panda Service and executed, and records are correspondingly updated in the Panda Service database.

The Panda job lifecycle (pre-2008)



User (Client)

Autopilot (pilot submission agent)

Submits a job via transformation

transformation repository

Panda Service

http(s)

Pilot Instance (on the Worker Node)

- get a PandaID
- get the transform URL
- get the transformation
- run the job (under Pilot UID)
- mark job as done in the DB

The Panda job lifecycle: earlier security concerns

- In the Panda version which existed until recently, the payload was always executed on the remote site under the UID traceable to a “production account”, namely the one from which the Pilot jobs were launched, but not to the account of the user who submitted jobs for execution. While an audit of user activity is still possible as it is recorded in the Panda database, there is admittedly a lack of immediate authorization mechanism at the user-account level, which is a concern to *certain* sites.
  - solution: to employ the **gLExec** mechanism to *optionally* modify UID from a production account to a user-specific account. This approach includes GUMS or LCAS/LCMAPS as the backend repository of DN-to-UID mapping. A VOMS proxy caching service such as MyProxy will be necessary to properly handle credentials.

- In principle a user can make use of the pilot’s proxy to initiate actions with the pilot proxy’s authority
  - We are addressing this by instantiating pilots with limited proxies and by using a special “role=pilot” VOMS annotations to limit the privileges of the Pilot process as it runs on the Worker Node.

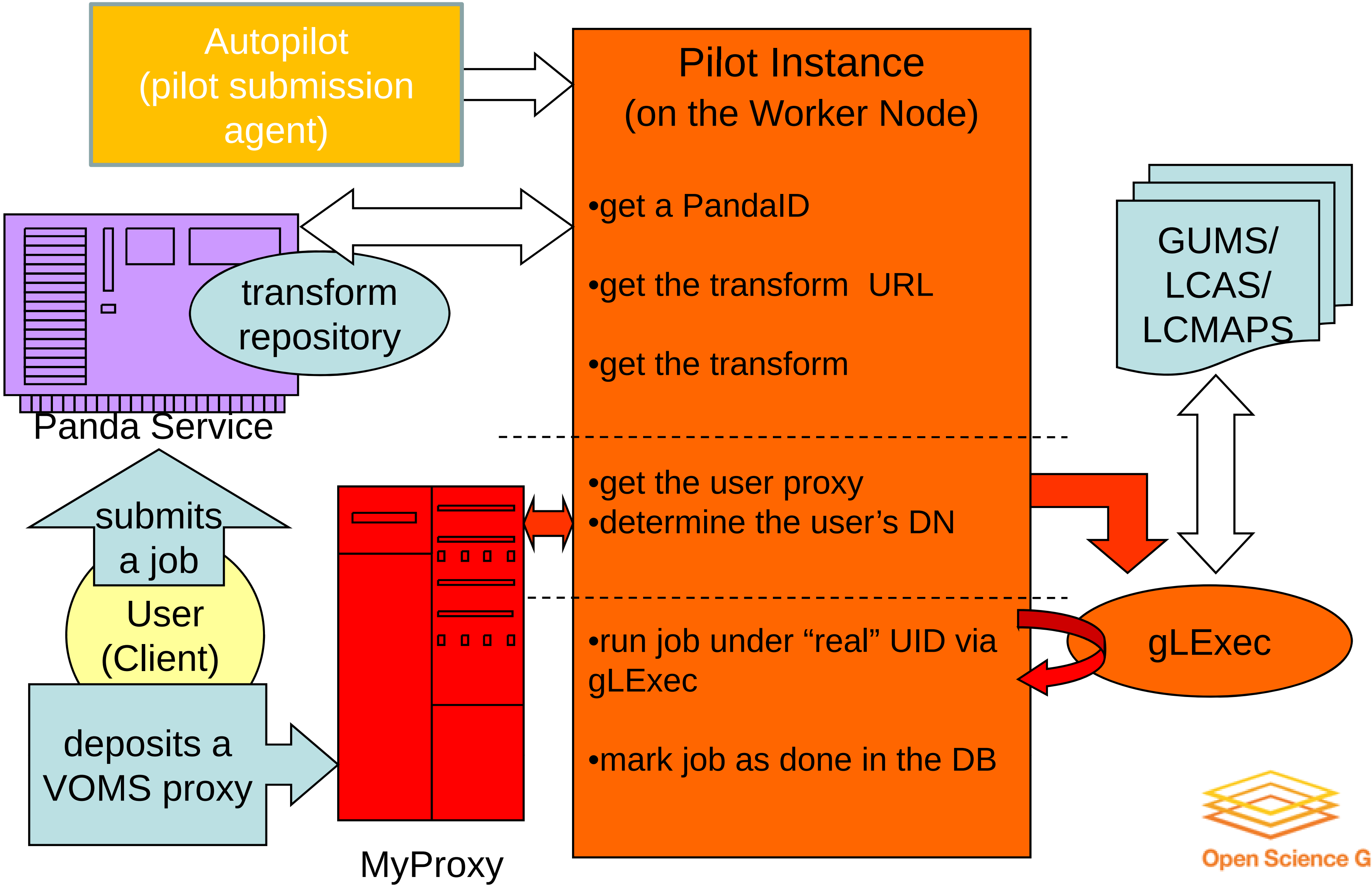
The Panda job lifecycle: introduction of gLExec

- **gLExec** functionality is enabled on specific sites only, the ones who
  - Have such requirement in their site policy
  - Are actually ready to support it, along with relevant middleware
- Only select US sites satisfy both requirements at the moment

- **gLExec** can operate in "log only" mode (which helps in auditing and incident investigation) and also in **setuid** mode, when the Pilot Job's uid is changed to that of the user who runs the job. In the latter case, such change is done based on the X509 proxy that must be provided prior to invocation of this utility. In our approach, the proxy is obtained from an instance of the MyProxy server (see the following diagram) and is not cached in any of the elements of the Panda system itself

- **gLExec**:
  - is a Grid version of suexec program
  - runs as setuid process on the CE
  - performs the switch based on results from GUMS/LCAS/LCMAPS mapping
  - can be configured to restrict which users allowed to invoke it

The Panda job lifecycle: introduction of gLExec



Autopilot (pilot submission agent)

Pilot Instance (on the Worker Node)

- get a PandaID
- get the transform URL
- get the transform
- get the user proxy
- determine the user's DN
- run job under "real" UID via gLExec
- mark job as done in the DB

transform repository

Panda Service

submits a job

User (Client)

deposits a VOMS proxy

MyProxy

GUMS/ LCAS/ LCMAPS

gLExec

gLExec integration: additional considerations

- gLExec is activated not in every pilot job, but only in those running on sites that are marked in the Panda database as “gLExec enabled”.
- Possible errors due to misconfiguration of gLExec, MyProxy client software etc on sites, are reported to the Panda service and logged for future debugging
- Because of change of identity while the job (Pilot) has already started on the remote batch system, the $HOME variable will be redefined. This may necessitate copying files to a new location.
- The environment specified by the Pilot when it starts is reset upon execution of gLExec. Variables such as $LD_LIBRARY_PATH and others need to be re-established.
- periodic renewal of the user proxies is done by a process forked from the Pilot and run with the Pilot identity
- security check:verification that the user DN and the “logname” mach. Otherwise, a malicious user could delegate a proxy using a different “logname” from his and runs jobs that he should not (logname is an identifier provided by the user to the MyProxy service when depositing the credentials there).

gLExec integration: additional considerations

- In the architecture being described, we are consciously avoiding having to implement an application-specific (Panda specific) cache of credentials, choosing instead to rely on established middleware elements such as MyProxy
- We are working on ways to minimize and mitigate impact of potential security incidents which theoretically may involve the Pilot proxy being compromised. Such measures include managing lifetimes of proxies and utilizing additional unique keys stored on the Panda service, which must be used to retrieve proxies from a MyProxy instance (note it’s not the same as caching proxies in Panda!).

gLExec integration: current status and plans

- BNL is running an instance of MyProxy specifically for Panda use, which ensures optimal configuration, control and security
- Panda jobs are been executed for testing purposes in gLExec mode on two US sites, BNL and FNAL
- BNL installation of gLExec is using the GUMS system as backend, as opposed to LCAS/LCMAPS which will be used by EGEE
- EGEE is currently working to resolve remaining issues in gLExec/LCAS/LCMAPS interface - Panda team is ready to expand testing and job submission to EGEE sites when this is done
- Possibility of GUMS installation on EGEE sites? Tested platform, expertise available.

Management of user access, VOs and resources in Panda

- Panda features several tiers of user access control:
  - A valid X509 certificate proxy is required to submit jobs to Panda
  - The Panda database table containing user info has a column to flag users who must be denied access for whatever reason (thus allowing for a quick ban during a security incident)
  - Before a job is dispatched for execution in the Panda service, the user’s DN is checked against a gridmap file that is periodically generated based on the VOMS service data, thus ensuring an existing affiliation of the user with an authorized VO
- Work is currently under way to use ReSS and similar tools to automatically populate “native” database tables in Panda with information mapping VOs to sites
  - More optimal distribution of workload according to resources made available, in near time
- Additionally, a resource allocation table is being added to the Panda database, which, in conjunction with addiotional job broker logic, will make it easier to follow usage level guidelines for each VO/site pair
  - Contains sub-VO (working group) level of detail

Conclusions

- Panda development team is proactive in finding ways to adhere to emerging site policies and improve the security of its Pilot-based execution framework
- We are open to all and any integration tests on EGEE sites, in that area
- Work has started on enhancing Panda with VO and resource management capabilities