Monitoring Virtual Machine Status with L&B

Šustr, Z., Sitera, J., Filipovič, J., Dvořák, F., Voců, M., Kouřil, D., Poul, M., Matyska, L.
CESNET, Czech Republic
The **Logging and Bookkeeping** service

- Long-standing part of the gLite middleware (EDG, EGEE, EMI)
  - Monitoring production compute jobs, mostly WMS
  - Back-end to the well known `glite-wms-job-status` command
- Collects event messages from grid elements
- **Processes** the information
- Publishes status information in several ways:
  - *query/response* interface – L&B querying API/HTML
  - *subscribe/publish* interface – L&B notifications
    - OpenWire/STOMP, RSS, L&B legacy
- Supports multiple types of grid processes
  - Compute jobs, collections, file transfers
  - Newest addition: **Virtual Machines**
Fig. 1: L&B receiving and producing messages over different channels
Virtual Machine (VM) Life Cycle

- Obvious similarities with a compute job life cycle
  - Well-defined life span
  - State transitions triggered by event messages
  - Shared infrastructure

- Even more similarity with pilot jobs
  - Start up with no specific payload defined
  - Attract attention when ready, get assigned payload

- Added benefits of using grid monitoring tools
  - Already in place
  - The same tool (L&B) can be used to monitor both layers
  - Easy to keep track of VM-Payload relationship

- VM seen as “just another job type”
• Inspired by Open Nebula 3.0 specification
• Driven by events collected from multiple sources:
  – Open Nebula or another VM manager
  – Hypervisor such as Xen
  – Actual VM instance
• Event logging instrumented through
  – a client library (C, C++, Java)
  – command line tools
• VM-specific states mapped to generic gLite states for universal queries

Fig. 2: VM state diagram
# Event Sources

<table>
<thead>
<tr>
<th>Component</th>
<th>Events</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CloudManager</td>
<td>Register, Create, Running,</td>
<td>Hostname, Physical host name, Owner, Requirements.</td>
</tr>
<tr>
<td></td>
<td>Host detail, Shutdown, Done</td>
<td></td>
</tr>
<tr>
<td>VM Image</td>
<td>Really Running, Shutdown</td>
<td>Runtime information, user tags</td>
</tr>
<tr>
<td>Hypervisor</td>
<td>Running, Shutdown</td>
<td></td>
</tr>
</tbody>
</table>

- Some types of events received from multiple components
  - Provide for redundancy and fine-grained monitoring
  - Fill in details known only at certain level
Job Relationships

- Extension of known L&B concept
- Previously: specific case implementations
  - subjobs ↔ collection/DAG parents
  - compute jobs ↔ sandbox transfers
- New: **generic bilateral relationship**
  - Registered/updated by events
  - Relationship status record maintained
    - Distinguishes *active* from *past* relationships
  - One of intended uses: linking compute jobs to VMs
    - Typically registered by batch system, known to both jobs

**Related Jobs**


**Fig. 3:** Jobs related to a VM – a screenshot
Benefits brought in by L&B

- Per-job ACL-based authorization
- Robust state interpretation
  - Resistant to events delivered out of order or not delivered at all
- Highly configurable output
  - Applies to synchronous queries as well as generated message streams
  - Complex sets of conditions supported
    - Messages sent over certain channels only if conditions are met
• Existing grid component has been extended to follow VMs as “another type of job”
• Keeping track of relationships with payload – regular compute jobs of different kinds
• Multiple channels available for receiving output, either by query or broadcast
• Potential for supporting event input over many channels
Thank You