Understanding Virtualized Infrastructure in Grid Job Monitoring

Zdeněk Šustr
Grid Department – MetaCentrum
CESNET z. s. p. o.
Zikova 4, Prague, 160 00, Czech Republic
Email: zdenek.sustr@cesnet.cz

Jiří Sitera
Grid Department – MetaCentrum
CESNET z. s. p. o.
Zikova 4, Prague, 160 00, Czech Republic
Email: jiri.sitera@cesnet.cz

***Abstract*—This paper is the first report on a new direction in the development of the Logging and Bookkeeping service, a gLite component tracking grid job life cycle. From the early days, Logging and Bookkeeping tracks not only jobs themselves but also the wider details of the job execution environment. Since a great portion of the infrastructure is now virtualized, the work at hand concerns tracking the virtualized nature of that runtime environment. With virtualization and cloud technologies being highly flexible and dynamic, we believe it is very important to gather and keep status information for machines used to run the workload. A newly created monitoring entity (a machine) will be integrated with job state information and provide an enhanced view of the current state and history of both the job and the infrastructure. This paper focuses on motivation, requirements coming from the Czech National Grid Initiative and possible consequences rather than the actual implementation. As a report on "work in progress" it describes an idea that is now being further elaborated and implemented to provide a solution for monitoring virtualized resources in the same context as the workload they are processing.**

***Keywords-grid*; *cloud*; *virtualization*; *job monitoring*.**

I. Introduction

Logging and Bookkeeping (LB), part of the gLite grid middleware, is a monitoring tool equipped for monitoring the states of all kinds of processes related to grid computing [1]. Besides traditional gLite Workload Management System (WMS) [2] jobs and logical groupings thereof such as oriented graphs (DAGs) or collections it also monitors input/output data transfers and the states of computing tasks submitted directly to a resource manager — the CREAM Computing Element (part of the gLite middleware stack) [3] or to TORQUE (Terascale Open-Source Resource and QUEue Manager) [4].

It collects event information from various grid elements and sums it up to determine the current status of any such process at the given moment. It is designed to accept additional state diagram implementations as required, relying on essential common features such as event delivery (based either on LB's own legacy messaging layer or standard STOMP/OpenWire messaging) or the querying interface. LB is highly security-oriented and has proven itself in WLCG (Worldwide LHC Computing Grid) operations. It is widely deployed across the European Grid Initiative's infrastructure.

In this article, Section II explains what the requirements are and why LB is deemed suitable for monitoring virtualized resources. Section III outlines the proposed solution to deliver essential functionality, and Section IV discusses additional issues to consider and focus on in the future.

II. Motivation to Include Machines in the LB Model

Using LB in monitoring virtualized resources is inspired by obvious similarities with the existing processes, backed by explicit requirements from infrastructure operators.

A. Virtual Machine as a Job

LB's main objective is to know everything about job scheduling and execution, making it possible to analyze the behavior of the infrastructure (failing components, misconfiguration) and possibly even provide job provenance capability (ensuring repeatability of jobs/experiments, storing computing environment characteristics and configuration). In contemporary grids and other computing infrastructures machines running grid jobs are themselves dynamic entities following a lifecycle similar to that of the job itself. It is not unreasonable to expect further blending of cloud and grid models where grid components run either in a cloud (StratusLab [5]), or in a mix with cloud services (MetaCentrum [6], WNoDeS [7]).

All things considered, tracking virtual machines (VMs) throughout their lifecycle in contemporary grids is as important as tracking jobs. Moreover there is an added value to tracking two kinds of entities in a common manner. Not only does it provide for a better understanding of mutual relationships and dependencies, but also for a unified view for users and administrators.

Figure 1 shows a simplified and illustrative example of the new higher-level view of the infrastructure state. It maps compute jobs to the underlying VM lifecycle and provides the user with an overview of its current state and possible problems. In the case of highly dynamic virtualized infrastructure it can be used to assess efficiency and induced tradeoffs. Data collected in this manner can also be used to produce higher-level statistics and monitoring (mapping

 ISBN: 978-1-61208-226-4

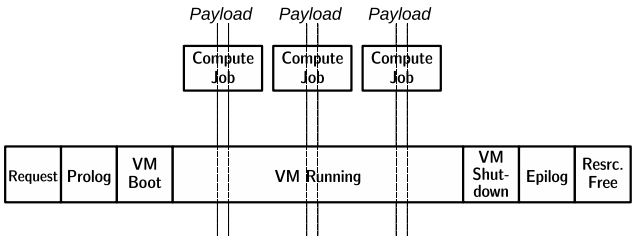

Figure 1. Viewing compute jobs as payload executing over a VM.

actual hardware resources to jobs), while the low-level information is still available for detailed inspection if required for debugging.

Key LB features (re)usable for machines:

- Recording primary events and using a state machine specific to the given type of process (job, VM) to combine all information contained therein and determine the current state of a process.
- Providing the ability to get processes grouped or annotated/tagged by the infrastructure, administrators or users.
- Architecture and implementation based on standards (messaging, authentication and authorization infrastructure, web services), allowing simple event gathering.
- Essential functions (logging events, querying for basic information) provided not only by library functions but also by command line tools.

B. Features Requested by the Czech NGI

MetaCentrum, the Czech National Grid Initiative (NGI), is designed as a mixed cloud/grid service, where resources from a single, consistently managed pool can be provided either as traditional batch system-managed resources or VMs, depending on current user needs [8]. The scheduler (Torque) can handle three types of requests:

1) Run a job
2) Run a job in a selected VM image
3) Run a VM

The desired functionality will provide a single, consistent view of the infrastructure, mapping all user requests to actual hardware. It should replace currently used data mining tools providing status feeds to the MetaCentrum portal and to the long-term usage statistics processor.

Since MetaCentrum is also involved in research of batch system scheduling strategies, gathering data relevant for this kind of assessment is another requirement.

Yet another requirement, albeit one that is already fulfilled by LB's design, calls for an ability to aggregate information from diverse sources (scheduler, virtualization hypervisor, accounting) and even manually triggered state transitions (for instance putting resources in and taking them out of maintenance).

C. Similar Works

Infrastructure monitoring tools such as *Nagios* or *Ganglia* focus primarily on the "running" state of the given process, and using them to monitor short-lived VM instances set up on demand is on the edge of practicality, anyway. Unlike them, this work is not intended to monitor infrastructure health and react to problems. There is just a minor overlap in that certain aspects of infrastructure health can be seen in job/VM status statistics provided by LB and we believe that understanding the relationship between the payload and VM layers will further improve the informative value of LB statistics.

Each infrastructure or cloud management tool has its own way (command line interface, portal) of providing users with the current VM status. But, we are not aware of any other work similar to LB – a service combining available information from different components into one higher-level view. It is one of the reasons for publishing this Work in Progress paper.

We expect that major virtualization stack implementations will be able to send raw status change events via the messaging infrastructure in the near future (indeed, some of them already do) and thus there will be interesting potential in processing them in the proposed way.

III. Proposed Solution

The proposed functionality is being implemented in progressive steps. Early phases are already in progress and can be discussed in detail, while the later phases consist mostly of open issues.

A. Implementation Phases

- Pilot implementation with a testbed instance of Open Nebula, running and keeping track of VMs and scheduled Torque jobs at the same time. This phase has already finished.
- Adjustment to MetaCentrum environment with Torque scheduling VMs as well as jobs. Making sure that the solution is adequately robust in all applicable use cases including those where some of the components (for instance some of the VMs) operate out of the scope of MetaCentrum and do not generate events. It is the current phase as of this publishing.
- Bringing in additional sources of information external to the batch system and virtualization stack: administrative operations, information system, accounting. Automated processing of information produced by LB: statistics, dashboards, etc.

B. Architecture for the First Phase

The primary goal of the first phase was to understand VM lifecycle and its relation to existing job lifecycle. The particular outcome from this phase consisted in finalizing

 ISBN: 978-1-61208-226-4

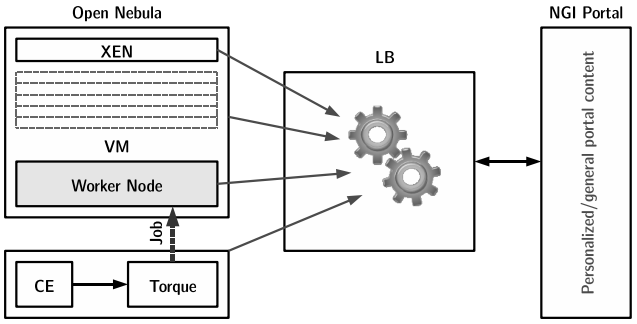

Figure 2. Architecture and components.

VM state machine design and the attribute set for the VM instance status record.

In this phase, the VM lifecycle was controlled solely by a single instance of the Open Nebula cloud computing toolkit, managed manually by administrators, while jobs were being assigned to VMs by a standard grid computing element through an instance of Torque. All job-related functionality was already in place (LB-aware Torque). The following sources of events were used to govern the VM lifecycle:

- *Open Nebula*—providing hooks for call-out scripts activated on any relevant state change
- *Hypervisor, (Specifically Xen)*—generating events showing the current VM state and parameters at hypervisor level
- *Hosted worker nodes*—Operating System running the Worker Node was instrumented (init scripts) to provide independent information from the running VM

The combination of events from all the above components into one higher-level view is a key role of LB in this concept. It makes the system more precise and robust, which has been well tested in the context of gLite job monitoring. Obviously on certain occasions, all three sources generate almost identical, i.e., redundant events. But there is still value in receiving almost identical events multiple times. It improves reliability, and the comparison between the three events provides for fine-grained job status tracking and simplifies troubleshooting. Besides that, different sources often provide values for different attributes unknown to the others.

System architecture for the initial implementation is shown in Figure 2. In that design, the only new feature that had to be implemented was VM instance support in LB (state machine, attributes, event types). Relationship to other relevant components of the system (virtual image identification, physical machine identification) is stored in the form of attributes in that instance. There are other attributes to cover the network status of the VM such as domain name, type of network connectivity (VLAN, private/public) and of course even more attributes identified as useful in the design/implementation process. The complete set of desirable attributes did not need to be pre-determined, though. LB allows any kind of additional attribute to be simply stored with the instance's status (functionality referred to as User Tags) with only slight limitations. One cannot, for instance, use relations such as "greater than" or "lower than" when querying for instances with a given value of such attribute. Since LB does not know the type of that attribute and cannot decide. The only comparison supported is string (in)equivalence.

Each instance is identified by a string constructed in the same manner as Job IDs currently used in LB, consisting of the LB server's identification, a short literal denoting the process type, and a random unique string. Domain names are not suitable for use as identifiers since they are often recycled (re-used by another instance) or even used by multiple VM instances at once.

Any event received by LB may or may not trigger a change in the state and/or attribute values of an instance. Thus the instance's current state and attributes constitute the most up-to date information set as collected from all the various sources mentioned above. LB is designed to overcome obstacles such as events delivered out of sequence, intermediate events not delivered at all, or events received from different sources with clocks skewed in different directions. This is achieved by relying on arbitrary hierarchical message sequence codes rather than time stamps in event sorting.

IV. Further Ideas and Open Issues

Given that this is still work in progress there are many concepts and ideas that deserve further investigation. Some of them, such as virtual cluster support, are necessities that must be addressed. Others fall into the "nice to have" category. They will receive attention at a later stage.

- *State Machine for Physical Machines?*—At the very least VM instance attributes will refer to a physical machine by name. But there is an obvious similarity between physical and virtual machines and a VM state diagram is easily applicable to physical machines. So the option is to register physical resources as "VM" instances as well, and reference the identifier instead. Then the same level of detail could be provided for virtual and physical machines alike, although some supported states will probably remain unused in the physical world.
- *Support of User Workflows*—Compared to traditional computing jobs, VMs are a little specific in that they always need to be assigned workload when running (i.e., having started for the first time, recovered from a downtime or finished migration), which makes them actually very similar to pilot jobs. Many user groups rely on their own workload management systems to distribute payload and it may be very convenient for them to receive notifications of relevant VM status changes.

 ISBN: 978-1-61208-226-4

That could be easily achieved with LB notifications generated on pre-determined conditions and sent out over LB's own legacy messaging chain or through a STOMP/OpenWire-enabled messaging broker. Users may choose, for instance, to be notified any time any of their machines reaches state *running*. More elaborate sets of conditions are also supported. The resulting notification contains the full VM status information and, if requested on registration, also the full history of events for that machine so far.

- *Virtual Cluster Implementation*—The Virtual Cluster service provided by MetaCentrum can create multiple VM instances per request [6]. All the resulting VMs have common attributes (type of network connection) and are closely related. It may be a good idea to reuse the "collections" functionality in LB, typically applied to grid jobs or sandbox transfers. From the user's point of view the state of the collection combines the states of all its members. Individual VM details are still accessible under the VM instance's own ID – the collection functionality simply adds another identifier (collection ID) to access aggregate information such as child status histograms.
- *Heterogeneous Environment (multiple hypervisors and cloud managers)*—LB should be able to provide a unified view of VMs running on different implementations of hypervisors or even cloud managers. The situation is similar to that of a unified state machine used for different job managers – CE implementations.
- *VLAN Status*—The Virtual Cluster service offered by MetaCentrum provides not only sets of machines but also networking connections in the form of virtual Ethernet (VLAN) [9]. The VLANs have their own lifecycle managed by a purpose-built VLAN manager (SBF). An ability to track the state of the network together with its attributes (private/public, additional service such as tunnel/NAT/FW) could be valuable in many scenarios.

V. Conclusion

Although this work is primarily driven by the Czech NGI's requirements, it will be found useful at a much wider scope. With instances of LB currently deployed at dozens of gLite-enabled grid sites across the European Grid Initiative's infrastructure, the VM monitoring feature – once released – will become available to a wide base of users, not only those already relying on LB for monitoring their own computing jobs, but also to those exploring the potential use of cloud services on grid-based platforms.

This paper's main goal was to show how the potential of job monitoring infrastructure can be reused in the virtualized world. Many cloud-oriented initiatives are currently looking for solutions enabling resource federation. LB, with its current presence resulting in easy adoption, will be a reasonable candidate for a monitoring and notification service.

Acknowledgment

This work is part of the National Grid Infrastructure MetaCentrum, provided under the programme "Projects of Large Infrastructure for Research, Development, and Innovations" (LM2010005).

Fundamental development and maintenance of the Logging and Bookkeeping service is co-funded by the European Commission as part of the EMI project under Grant Agreement INFSO-RI-261611.

References

[1] MetaCentrum Project, *Logging and Bookkeeping*, CESNET, 2008. [Online]. Available: http://egee.cesnet.cz/en/JRA1/LB/ [Accessed: August 28, 2012].

[2] M. Cecchi et al., The gLite Workload Management System, *J. Phys.: Conf. Ser.*, vol. 219, 2010.

[3] P. Andreetto et al., Status and Developments of the CREAM Computing Element Service, *J. Phys.: Conf. Ser.*, vol. 331, 2011.

[4] G. Staples, TORQUE resource manager, *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, ser. SC '06, 2006, ISBN 0-7695-2700-0.

[5] StratusLab Project, *StratusLab*, StratusLab, 2012, [Online]. Available: http://stratuslab.eu/ [Accessed: August 28, 2012].

[6] M. Ruda et al., *Virtual Clusters as a New Service of MetaCentrum, the Czech NGI*, CESNET, 2009. [Online]. Available: http://www.cesnet.cz/doc/techzpravy/2009/virtual-clusters-metacentrum/ [Accessed: August 28, 2012].

[7] D. Salomoni et al., "WNoDeS, a tool for integrated Grid and Cloud access and computing farm virtualization," *J. Phys.: Conf. Ser.* 331 052017, Dec 2011.

[8] J. Sitera, M. Ruda, P. Holub, D. Antoš, and L. Matyska, *MetaCentrum Virtualization – Use Cases*, CESNET, 2010. [Online]. Available: http://www.cesnet.cz/doc/techzpravy/2010/metacentrum-virtualization-use-cases/ [Accessed: August 28, 2012].

[9] D. Antoš, L. Matyska, P. Holub, J. Sitera, "VirtCloud: Virtualising Network for Grid Environments – First Experiences" in *The 23rd IEEE International Conference on Advanced Information Networking and Applications AINA*. Bradford, UK, 2009.

 ISBN: 978-1-61208-226-4