International Journal of Research in Engineering, Science and Management
Volume-3, Issue-2, February-2020
www.ijresm.com | ISSN (Online): 2581-5792

Adaptive Monitoring Platform as a Service for Cloud Computing Infrastructure and Services

Ashok S. Patil[1], G. Monisha[2], Y. Venkata Chaitanya Reddy[3]
[1]*Associate Professor, Department of Information Science and Engineering, Nagarjuna College of Engineering and Technology, Bangalore, India*
[2,3]*Student, Department of Information Science and Engineering, Nagarjuna College of Engineering and Technology, Bangalore, India*

***Abstract*: This paper presents a novel monitoring architecture addressed to the cloud provider and the cloud consumers. This architecture offers a monitoring platform-as-a-Service to each cloud consumer that allows to customize the monitoring metrics. The cloud provider sees a complete overview of the infrastructure whereas the cloud consumer sees automatically her cloud resources and can done other resources or services to be monitored. This is accomplished by means of an adaptive distributed monitoring architecture automatically deployed in the cloud infrastructure. This architecture has been implemented and released under GPL license to the community as "Monagas", open source software for integrating Navies and Open Stack. An intensive empirical evaluation of performance and scalability have been done using a real deployment of a cloud computing infrastructure in which more than 3,700 VMs have been executed.**

***Keywords*: Adaptive monitoring platform, Cloud computing infrastructure.**

1. Introduction

Cloud computing is changing radically the way in which hardware infrastructures are being utilized. The open source cloud computing stacks for managing private clouds infrastructures, i.e., infrastructure-as-a service (IaaS), like OpenStack, Apache Cloud Stack, Snooze, Open Nebula [1], and Eucalyptus [2] enables enterprises to update the management plane of their data centre in order to optimize the usage of the computational resources according to the constantly changing business requirements of their organizations. On the other hand, public cloud computing vendors like Amazon EC Microsoft Azure and Rackspace enable cloud consumers to extend their infrastructure beyond the physical boundaries of their hardware by renting third party computational resources. a pay-as-you-go model enabling the creation of an elastic infrastructure. Despite of the well-known advantages of cloud computing such as the important cost reduction in hardware acquisition and the optimization in the usage of hardware resources, cloud computing also demands important challenges which have to be seriously addressed in order to provide really attractive solutions for the business market. One of the main challenges associated to cloud computing monitoring is the lack of information and control with regards to the customization of the monitoring metrics that the cloud customers have over the rented cloud resources. In private clouds, the provider and the consumer of the cloud resources are the same person and thus she has a complete control over the physical hardware resources. She can easily install monitoring software to gather information over the physical and virtual infrastructures. However, the current monitoring solutions do not at well in the cloud computing scenario where the resources, typically virtual machines (VMs), storage disks and image disks, are usually virtualized. Virtualization entails a new life cycle for the virtual resources as they are constantly being created and destroyed which does not occur in physical resources. The public cloud scenario is even more challenging due to the additional fact that the cloud provider cannot install any software in the cloud consumer's VMs without her previous acknowledge. These properties cast aside traditional monitoring solutions which:

- Have not been design to at in the life cycle of virtual resources; and
- Are usually based on monitoring agents installed inside the machines to gather metrics. These facts make the traditional monitoring solutions not at for cloud infrastructures. As a rest step to address this issue, we developed Eason [3], a free and open-source monitoring solution.

Service for the cloud consumer. To explain the monitoring architecture that full's the above requirements, this paper has been organized as follows. In Section 2, the reader will see a complete state-of the-art of monitoring solutions for cloud computing infrastructures together with our differentiating points with respect to such solutions. After that, the monitoring architecture is carefully described in Section 3. Then, Section 4 describes some details about the open source implementation released to the community. Section 6 provides a detailed set of statistics about performance and scalability of the monitoring architecture proposed. Such statistics are gathered from a real cloud computing test bed infrastructure. Finally, Section 5 outlines some conclusions.

2. Related work

There are many research works related to the monitoring of cloud computing infrastructures. This revision of the state of-the-art is only focused on software used to monitor infrastructures and their services leaving aside other monitoring tools such as network traffic monitoring, Quos monitoring, SLA-monitoring, security monitoring or any other monitoring based on physical sensor devices.

Cloud computing infrastructures only suitable for small size deployments due to the lack of scalability associated to the centralized monitoring approach. Tovarnak and Pinter [7], Hangar et al. [8], Salve and Govindarajan [9], Kaisaris et al. [10] together with Shoo et al. [4], Huang and Wang [5] provide monitoring solutions focused on the installation of a software agent in the cloud customer's VMs. While this installation is acceptable for environments in which the cloud customer want explicitly to perform monitoring solutions, it does not cover the scenarios in which the cloud customer does not want to perform explicitly the monitoring of her VMs but the cloud provider needs to perform a nonintrusive monitoring of such VMs in order to have an updated overview of the complete infrastructure. We are particularly interested in offering a solution for both customers and providers of cloud infrastructures thus only architectures that consider monitoring agents as optional are suitable ulfilour requirements. Andreolini et al. [11] describe a distributed monitoring architecture designed specifically for cloud computing. However, they do not cover any particularity related to the monitoring of virtualized infrastructures like the mapping between virtual and physical resources and the management of destructions of virtual resources, etc. Moreover, they focus their design on the mandatory usage of software agents which does not at in our requirements. VMDriver [12] is focused on implementing transparent monitoring in which agents are not required and the information is gathered by means of the hypervisor. This feature enables the cloud provider to get basic information about the cloud consumer resources. This is a very significant step forward; however, it does not enable the customization and extension of different monitoring metrics. Sandoval et al. [13] analyze a number of already available monitoring architectures to determine the best choice in order to be adapted to the new requirements imposed by cloud computing infrastructures. As a result, they indicate Navies as the best alternative. There are a number of mature monitoring tools for traditional IT infrastructures already available. Nagios, Zennoss, Icinga, Zabbix and OpenNMS are good examples of free tools whereas LiveAction and Inmost Monitoring Solution examples of commercial ones. We have rapidly discarded all the software which imposes a monitoring agent like Pandora FMS, Ganglia, and Symons. Moreover, we also discarded commercial solutions and solutions which do not provide the source code. As a result, only the previously indicated eve distributed monitoring tools were identified as good candidates from a list of more than 40 monitoring solutions. Zambia, Zeros and Open NMS come with auto-discovery protocols which may not fit properly in virtualization scenarios in which machines are continuously being created development status. Besides, it has not been designed to scale in large deployments due to the fact that all the monitoring information is being sent to a central service which is a clear bottleneck. Koenig et al. [18] provide a radically new approach focused on the elasticity of the monitoring platform. This idea complements perfectly with our architecture and can be integrated. However, Koenig et al. do not integrate the monitoring architecture with the cloud computing stack so that the architecture is not able to automatically be adapted against changes in the infrastructure. We have done a complementary analysis of the traditional monitoring solutions. More than forty different monitoring tools have been analyzed, concluding with Navies as the best alternatives to be considered. Navies itself does not at well in cloud computing infrastructures due to the fact that it has been designed to monitor physical infrastructure rather than highly changing virtual infrastructures. So, we would like to remark that none of the above discussed monitoring architectures is able to full the requirements indicated in the introduction of this paper. This fact together with the clear lack of monitoring software which can be easily installed and integrated in a cloud computing infrastructure to provide monitoring capabilities to consumers and providers have been the main reasons which have motivated this research work.

3. Monitoring platform-as-a- service for cloud computing

The different components of the monitoring architecture proposed. In order to explain each component in detail, this section has been divided into different sections. Usually referred to as the cloud controller which runs the essential services of the cloud computing infrastructure. These services may be at least:

- Scheduler, to decide where to place new resources like VMs, storage disks, etc.;
- Volume manager, to manage storage volumes;
- Image manager, to manage OS images;
- Networking, to manage internetworking between resources;
- Authentication;
- Entry point API, to provide the entry point to the cloud services;

Billing, to manage the billing and usage of resources; These messages are used by our monitoring architecture to perform the self configuration of the physical resources to be monitored by the cloud provider. Changes in the virtual resources are also noticed by messages related to the creation and destruction of VMs and storage volumes. These messages allow Monagas to perform actions in the monitoring software to adapt this new status of the infrastructure quickly and to start/stop the monitoring of the new/old resources. The loss of a message about the physical resources does not entail serious challenges due to their periodic nature; however, a loss of any message related to the virtual resources may lead in a de-synchronization

between Monagas and the cloud computing infrastructure. To cope with this issue, we have designed a disaster recovery process for which, Monagas asks the cloud computing infrastructure about all the virtual resources available to synchronize the status between subsystems. This synchronization is done during the initialization of Monagas so that it is able to synchronize the current status of the system as part of its bootstrapping process. This feature enables to sync against any failure in Monagas and in the cloud infrastructure. This synchronization requires the access to the internal information of the cloud infrastructure implying admin privileges.

4. Implementation

Monagas has been prototypically implemented and released to the community as an open-source project under GPL license. Monagas enables the smooth integration between Navies and Open Stack. Monagas implements all the features indicated along this contribution. It has been implemented using Java and some required Java libraries: i) to connect with Rabbit message queue, the communication middleware used in Open Stack) and ii) to connect with None using a custom HTTP-based API. Monagas has been released with a complete automat zed installation and configuration process which enables the practitioner to perform the automatic installation and configuration of all the components presented in the architecture. The configuration includes also the preloading of the OS monitoring images in Open Stack, creating special customers, loading cryptographic information, etc. We encourage the reader to download and install MonPaaS in your Open stack cloud computing infrastructure since it is a 100 percent functional prototype. The only aspect of the implementation worthy to mention is the disaster recovery process. This feature has been implemented as optional via configuration parameters. The reason is that admin privileges are required by Monagas to perform this functionality. Notice that Monagas needs to ask the cloud provider about all the VMs belonging to ALL the cloud consumers and this is a privileged actions only available to the infrastructure administrator. This can be a source of security threats. Thus, we have decided to keep this functionality

5. Conclusion

A distributed and high scalable monitoring architecture for cloud computing infrastructures has been provided to the community. The architecture full successfully all the requirements indicated in Section 1 providing an architecture with multi-tenant support suitable for both cloud consumer and cloud providers. The architecture is able to provide disaster recovery capabilities, load balancing, adaptive monitoring and self-configuration among other advanced features. A significant step has been provided allowing the cloud consumer to customize her own metrics, services and resources, creating a real monitoring Platform-as-a-Service for cloud computing infrastructures. The architecture has been successfully validated against an intensive test bed in which more than 3,700 VMs have been executed to get all the empirical results. In fact, it has been empirically proved that our monitoring architecture only imposed a negligible overhead near to the 0.05 percent (15 seconds). Also, it has been proved that the monitoring architecture scales really well under different stressing workloads. As future work, we would like to analyze bigger scenarios in which the Navies monitoring services of the cloud consumer may be a potential bottleneck due to the creation of an incredibly number of VMs under the same cloud consumer. We would like to extend the monitoring architecture using workload balancing capabilities for the monitoring services of the cloud consumer as well. We also would like to design novel disaster recovery methods in which not only the Navies monitoring service of the cloud provider is recovered but also the Navies monitoring service of the cloud consumer can take advantage of this feature.

References

[1] D. Norma, R. Woolskin, C. Grzegorczyk, G. Obertelli, S. Soma, L. Yourself, and D. Zagorodnov, “The Eucalyptus Open-Source Cloud-Computing System,” Proc. Ninth IEEE/ACM Int’l Sump Cluster Computing and the Grid, 2009.
[2] J.M. Alcatraz Calera and J. Gutierrez, “Eason: Framework for Monitoring Cloud Computing Data Centers,” Source forge Project.
[3] J. Shoo, H. Wei, Q. Wang, and H. Mei, “A Runtime Model Based Monitoring Approach for Cloud,” Proc. IEEE Third Int’l Conf. Cloud Computing, 2010.
[4] H. Huang and L. Wang, “P&P: A Combined Push-Pull Model for Resource Monitoring in Cloud Computing Environment,” Proc. IEEE Third Int’l Conf. Cloud Computing, 2010.
[5] M. Rack, S. Venticinque, and T. Maher, “Cloud Application Monitoring: The mosaic Approach,” Proc. Third IEEE Int’l Conf. Cloud Computing Technology and Science, 2011.
[6] D. Tovarnak and T. Pinter, “Towards Multi-Tenant and Interoperable Monitoring of Virtual Machines in Cloud,” Proc. 14th Int’l Sump. Symbolic and Numeric Algorithms for Sciatic Computing, 2012.
[7] M. Shangri, J. Lakshmi, and S.K. Nandi, “Resource Usage Monitoring in Clouds,” Proc. ACM/IEEE 13th Int’l Conf. Grid Computing, 2012.