Cloud Watch and Monitoring of Servers as a Service

Dharmendra Kumar[1], Dinesh Chand[2]

[1]Department of Computer Science and Engineering Galgotias College of Engineering and Technology Greater Noida, India Dharam.kumar49@gmail.com

[2]Department of Computer Science and Engineering Galgotias College of Engineering and Technology Greater Noida, India dineshchaudhary0101@gmail.com

Abstract

Cloud computing and its pay-as-you-go economic model not only enable application developers and application service providers to perform on-demand utility computing, but also push the evolution of data enter technologies to become more open and more consumer driven. Typically, in addition to rent virtual server instances and pay for certain middleware services based on their usage, such as load balancing in EC2, Cloud consumers also need to monitor the performance of their applications in response to unexpected peaks of service requests or performance degradation in their multi-tier application frameworks. Similarly, Cloud providers need to monitor the large number of computing nodes in their data centrism response to virtual machine failures or performance degradation of virtual machines, ensuring the level of service quality agreement demanded by the Cloud consumers. Today's Cloud data centers are complex composition of large-scale servers, virtual machines, physical and virtual networks, middleware, applications, and services. Their growing scale and complexity challenge our ability to closely monitor the state of various entities, and to utilize voluminous monitoring data for better operation. Providing Monitoring-as-a-Service (MaaS) to Cloud administrators and users brings a number of benefits to both Cloud providers and consumers.

Key Words: *Cloud Computing, Data Security, Ec2 Server Instance, Amazon Web Services, Java.*

I. INTRODUCTION

State monitoring is a fundamental building block for Cloud services. The demand for providing state monitoring as services (MaaS) continues to grow and is evidenced by CloudWatch from Amazon EC2, which allows cloud consumers to pay for monitoring a selection of performance metrics with coarse-grained periodical sampling of runtime states. One of the key challenges for wide deployment of MaaS is to provide better balance among a set of critical quality and performance parameters, such as accuracy, cost, scalability and customizability. This dissertation research is dedicated to innovative research and development of an elastic framework for providing state monitoring as a service (MaaS). We analyse limitations of existing techniques, systematically identify the need and the challenges at different layers of a Cloud monitoring service platform, and develop a suite of distributed monitoring techniques to support for flexible monitoring infrastructure, cost-effective state monitoring and monitoring-enhanced Cloud management.

At the monitoring infrastructure layer, we develop techniques to support multi-tenancy of monitoring services by exploring cost sharing between monitoring tasks and safeguarding monitoring resource usage.

II. USE ABILITY AND SCALABILITY

The user interface is well designed that a normal user can easily use all of its features. The GUI application provides user with easy identification of their Amazon Web Services and all the servers are connected to the panel. Such application is to be developed for all major Web Browsers so that users of a large scale can use the system.

A. Android Application on Google Play Store

It will be very easy for android smartphone user to get the application on their android application market place named Google Play Store. This application will be absolutely free to download and install in their Android smartphones. The application will be developed for almost all the current running versions of android. The application will have simple user interface. All they need is to just login and Control their Amazon Web Services.

B. IOS Application on Apple App Store

For the iOS users, the application will be freely available on App Store. The interface will be almost same while the application will be available for all current running versions of iOS.

C. Windows Phone Application on Windows Store Windows have its own Store for applications, this app will easily be available for Windows Phone users for free. Windows Operating System is well known for its user interface and this app will be very user friendly.

D. Web Site which can be Accessed from Any Web Browser. The use of this system is not only limited to smartphones or tablets but whole application will also be available online which can be accessed from anywhere and from any device having a web browser and an internet connection. The user interface for web application will be simple to use and attractive.

E. Utilisation Control Capability

The Web application will have the option of Showing Utilization as an output, in addition to instruct the user with the Fixed Utilization so that user can also control the Bandwidth of Server by setting the alarm on a fixed utilization. For this feature to be available we will use Graphs to determine and showing the Utilization and convert it to required data [3].

III. LITERATURE SURVEY

In last 20 years, a large body of research has focused on developing tools and techniques for monitoring the QoS status of resources and applications over distributed systems (e.g., grids,clusters, and clouds). Some QoS monitoring techniques have been investigated and implementedin computational grids, such as Network Weather Service (NWS) [3], which monitors thenetwork and computing resource QoS and periodically forecast the QoS in a future arrival ofa application workload. The current version of NWS gathers the operating system level metricssuch as available CPU percentage, available non-paged memory, and TCP/IP Performance.Other monitoring tools [4, 5] that were popular in grid and cluster computing era included R-GMA,Hawkeye, Ganglia, MDS-I, and MDS-II. Aforementioned monitoring techniques and toolswere designed for managing static system configuration, where numbers of hardware and software resource types were assumed to remain constant over lifecycle of an application. In otherwords, these tools did not consider the issue of auto scaling and de-scaling primitivessupported by virtualized cloud resources. These tools were only concerned about monitoringthe QoS parameters for the hardware resources (CPU, storage, and network), while beingcompletely agnostic to application-specific QoS parameters and SLA requirements. The performance of these tools was optimized for monitoring the QoS of only one type of application(e.g., high performance computing application). On the other hand, in cloud computingdatacenters, multiple application instances can be multiplexed and co-allocated on single physicalresource. Clearly, the monitoring tools developed in grid and cluster computing era (while beinginnovative and useful) is not suitable to tackle the challenges on cloud computing environments and hosted application types [2].

II. IMPLEMENTATION

System implementation is the stage when the user has thoroughly tested the system and approves all the features provided by the system. The various tests are performed and the system is approved only after all the requirements are met and the user is satisfied.

The new system may be totally new, replacing an existing manual or automated system, or it may be a major modification to an existing system. In case of, proper implementation is essential to provide a reliable system to meet organizational requirements. Successful implementation may not guarantee improvement in the organization using the new system, but improper will prevent it. Implementation is the process of having systems personnel check out and put

new equipment into use, train users, install the new application and construct any files of data needed to use it. This phase is less creative than system design. Depending on the size of the organization that will be involved in using the application and the risk involved in its use, systems developers may choose to test the operation in only one area of the firm with only one or two persons. Sometimes, they will run both old and new system in parallel way to compare the results. In still other situations, system developers stop using the old system one day and start using the new one the next.

The user should be simple and easy to understand and use. Also be an interactive interface .The system should prompt for the user and administrator to login to the application and for proper input criteria

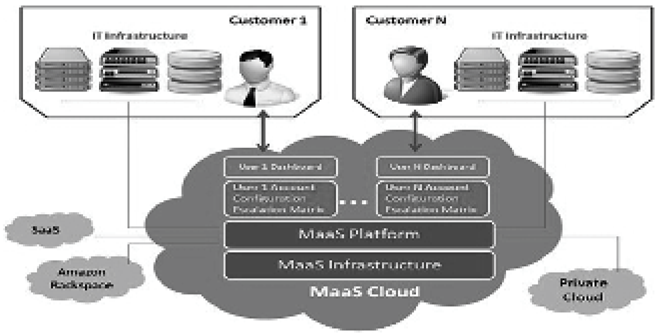

Figure 1. A general architecture of system.

The system can be divided in three sides to understand its implementation

A. Client Side

At the user side there will be a Graphical Interface which allows them to login their account and access the rented Servers in their account. These rented servers will be those which were rented by the user for personal use. The interface will be Smartphone application for the respective OS of the Smartphone. Web application can also be used by accessing it through any web browser at any computer by the user. All a user need is to login with their credentials and continue to control their appliances and doors.

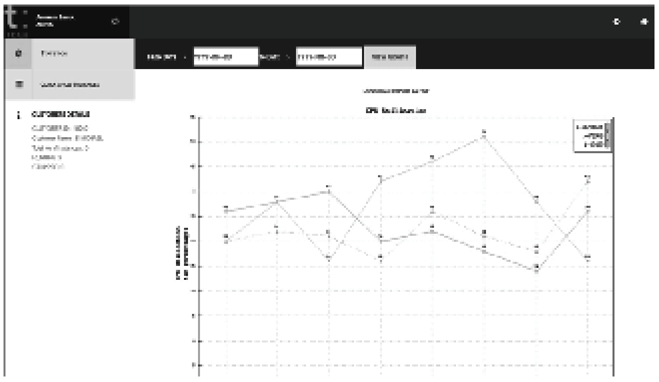

Figure 2. A general Interface of User Side.

A. Admin Side

The software provides good graphical interface for the user any administrator can operate on the system, performing the required task such as create, update, viewing the details of the Client.

Allows Admin to view quick reports like Server state and server utilization in between particular time.

Stock verification and search facility based on different criteria.

B. Medium Layer (the Virtual Side)

To make the above mentioned two sides there is a requirement of a medium. This medium is provided as a Server which will be used as a central point to transmit data from one side to another. This Server related mechanism is basically the Cloud Computing. We will use Amazon Web Services (AWS) as the Cloud service to serve the application and web site to access the database. For communicating data between both sides, it is needed to share same database among both the sides. As Amazon Web Services is an IaaS (Infrastructure as a Service) provider, it will take care of the infrastructure and several tools will be used for several apps as per the requirements. It is the responsibility of this layer to make the movement of data in between both the sides.

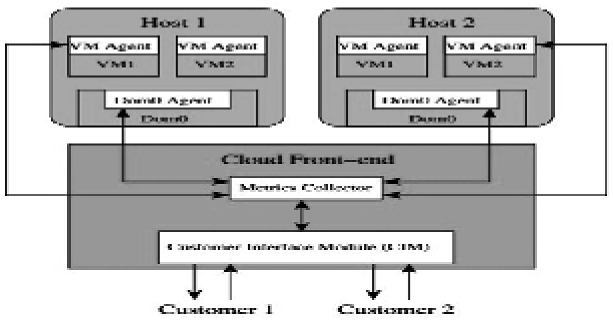

Figure 3. Block diagram showing multiple sides of system.

V. SIMULATION AND TEST

For simulation of the above discussed mechanism we have created a Web environment connecting few of the Amazon Web Services with the simulated Web Application and hosted a beta version of the web application on local host. The simulation went on in several phases [4]:

A. Modelling

All the Servers were shown in a Web Application with proper Utilization following the required precautions. As this was a generative test, we don't prescribe the exact behavior of our test performed, but rather what it could perform. The web application was connected through Internet in local host.

B. Test Generation

From the non- deterministic model, a well-defined set of planned actions is generated. The capability and the robustness of the system was tested and the actions such as the sessions creation and synchronization check were taken.

C. Simulation Invocation

Generated set of actions were executed. During each planned action a part of the system is interacted. We divided the system in parts (different sides- Client side, Admin side etc.) and observed the functioning of the simulation for each part.

D. Validation

Finally, at some point in time later, the action log of this simulation could be queried and aggregated to detect any failures or abnormalities. Being a separate process from running the test themselves, it also becomes possible to add new validations and “backfill” failures from previous points in time.

From the above simulation of the system we observed that the system was working fine and during the testing it was found that more Servers will be needed for Efficient testing. This clearly shows that number of Servers required in the panel (and hence the size of the panel) is directly proportional to the Efficient Testing to be done.

VI. CONCLUSION

Application QoS monitoring will continue to remain an important research area for cloud-basedsystems. More tangible efforts are needed for developing monitoring tools and techniques thatcan specify, reason, and monitor QoS related to a variety of application (enterprise, scientific,and streaming big data analytics) and cloud datacenter types (private and public). Further, researchshould also aim to correlate events with data from many different sources (e.g., holiday schedules,job schedules, and trends from social media about application usage sentiment) in order to predicthow external events can impact an application QoS. In this special issue, we have selected researchpapers that aim to address some of these challenges. We hope that the readers will find the articles of this special issue informative and useful.

This type of approaches collect all monitoring related data to a central repository where data processing and analysis are performed. Note that such centralized approaches still need to collect monitoring data from distributed nodes.

The second type of approaches selectively collect monitoring data from a distributed system or application based on the goal of monitoring and analysis. They still perform centralized processing and analysis on the collected data.

ACKNOWLEDGMENT

We gratefully acknowledge the contribution of each other in analyzing and writing this particular paper. We appreciate each other's knowledge and the understanding in between us.

Furthermore, we would like to thank Ms. Sonakshi Gupta, our lecturer and guide who let us get informed about the recent researches and technologies.We feel highly privileged to thank Prof. Dr. Bhawna Mallick, Head of Department, Department of Computer Science and Engineering, Galgotias College of Engineering and Technology, Greater Noida, India for her appreciation and motivation. It could never be possible to write this paper without her continuous support and motivation.

REFERENCES

[1] Schad J, Dittrich J, Quiané-Ruiz J. Runtime measurements in the cloud: observing, analyzing, and reducing variance. Proceedings of the VLDB Endowment 2010; 3(1-2): 460–471.

[2] Alhamazani K, Ranjan R, Mitra K, Jayaraman P, Rabhi F, Khan SU, Guabtni A, Bhatnagar V. An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art. Springer Journal of Computing, Elsevier, 2014.

[3] Wolski R, Spring N, Hayes J. The network weather service: a distributed resource performance forecasting service for metacomputing. Future Generation Computer Systems 1999; 15(5-6): 757–768.

[4] Zhang X, Freschl J, Schopf JM. A performance study of monitoring and information services for distributed systems. In Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing (HPDC '03). IEEE Computer Society: Washington, DC, USA,2003;270–281.