Scalable Monitoring Analytics Architecture in Software-Defined Infrastructure

Jieyu Lin
University of Toronto
Toronto, Ontario, Canada
jieyu.lin@utoronto.ca

Byungchul Park
University of Toronto
Toronto, Ontario, Canada
byungchul.park@utoronto.ca

Qi Zhang
University of Toronto
Toronto, Ontario, Canada
ql.zhang@utoronto.ca

Hadi Bannazadeh
University of Toronto
Toronto, Ontario, Canada
hadi.bannazadeh@utoronto.ca

Alberto Leon-Garcia
University of Toronto
Toronto, Ontario, Canada
alberto.leongarcia@utoronto.ca

ABSTRACT

Software-Defined Infrastructure provides a unified architecture for integrated management of virtualized heterogeneous resources in cloud infrastructures. Monitoring and measurement is an important component in the Software-Defined Infrastructure. In this paper, we describe the architecture of Software-Defined Infrastructure focusing on its monitoring and measurement manager. The monitoring and measurement manager has been implemented in a system called MonArch, which is capable of performing integrated monitoring of heterogeneous resources. MonArch has been deployed and operational in the SAVI Testbed. Flexible analytics capability and scalability of MonArch is demonstrated through two use cases: virtual machine communication analysis and anomaly detection.

1. INTRODUCTION

Cloud computing and virtualization technology have emerged and enable new types of service delivery over the Internet by partitioning physical resources (e.g., computing and network resources) into one or more virtual resources. With the rapid development of cloud computing and virtualization technologies, today's data centers provide virtualization of diverse physical resources including servers, switches, middleboxes, wireless access points, GPUs, and FPGAs.

As types of supported virtual resources are increasing, more sophisticated management systems are required for cloud service to provide efficiency, robustness, and security. However, current cloud management systems often use separate management tools for each resource type and having separated monitoring or management systems increases the capital and operational expenditures (CAPEX and OPEX).

Ideally, a cloud management system should able to monitor heterogeneous resources while providing unified interface for accessing and analyzing entire cloud system. Scalability and expandability are also required. Given large number of virtual resources from a variety of sources, the system should ensure that the incoming monitoring data can be stored and processed in an efficient and timely manner and can handle new type of resources that may be added to the infrastructure.

To this end, we have designed MonArch, a Monitoring Architecture for Software Defined Infrastructure (SDI) [2]. MonArch provides a flexible interface for collecting unstructured and structured monitoring data, as well as allowing the measurement data to be efficiently stored and processed in the system.

In our demonstration, we will use MonArch for two use cases: virtual machine communication analysis and anomaly detection. The virtual machine communication analysis requires processing historical monitoring data and correlating data that are from different resources and different layers. For the anomaly detection use case, we utilize MonArch's stream processing capability to identify anomalies with short delay.

2. SOFTWARE DEFINED INFRASTRUCTURE (SDI)

Current control and management systems mainly focuses on having separated controllers for difference resources. For example, in cloud environment, compute and network resources often have separate controllers. The cloud controller in OpenStack is responsible for managing virtual machines and storages, whereas the network is often managed by OpenFlow or other SDN controller. This approach is not ideal for best decision making and optimization of performance and cost. SDI [2, 1] presents an approach to integrate the management of different resources into a logically centralized point to provide more flexibility and intelligence.

Fig. 1 shows the high level architecture of the SDI Resource Management System (RMS). In a typical virtualized infrastructure, we have converged heterogeneous resources including both physical and virtual resources.

TRIDENTCOM 2015, June 24-25, Vancouver, Canada

DOI 10.4108/icst.tridentcom.2015.260151

An example of the virtual resources is a VM. Different types of resources are represented by different types of shapes in Fig. 1. Physical resources are denoted by shapes with solid line where as virtual resources are indicated by shapes with dotted line .

Each type of resource connects to its type-specific resource controller for direct control and management. A resource controller can be any traditional controller that is designed to manage a specific kind of resource. The use of these resource controllers allows us to leverage results from other research and projects.

Different types of resources may have connections or dependencies between each other, which gives rise to a need for a topology manager. The topology manager is responsible for communicating with different resource controllers for obtaining and maintaining the topology information of all the resources.

A monitoring and measurement manager is needed to monitor the converged heterogeneous resources, and to obtain real-time resource information and states. The monitoring and measurement manager keeps a historical record of the monitoring data for future reference and analysis. To meet monitoring and measurement needs from users, the monitoring and measurement manager has open API that allows user to access monitoring data based on access control policy. Moreover, due to the complex analytics requirement in today's cloud environment, the monitoring and measurement manager should also be capable of performing complex analysis on the monitoring data. This requires the system to support executing analytics job submitted by users. Other requirements of the monitoring and measurement manager include 1) scalability: the system should be able to scale up to support more monitoring resources; 2) Cross layer monitoring support: the system should be able to monitor resources in both the infrastructure layer as well as the application layer.

The monitoring and measurement manager and the topology manager together provide a multi-dimensional view of the infrastructure that includes spatial (topology), and temporal (resource states and information at different times) information. The monitoring and measurement manager can potentially construct a historical view of how the topology of the infrastructure has changed in the past.

The SDI manager is the decision-making point in this architecture. It is responsible for performing integrated resource management for converged heterogeneous resources. The SDI manager obtains the resource topology information from the topology manager, and the monitoring data and the analytics results from the monitoring and measurement manager. Based on the collected information, the SDI manager executes the integrated management algorithms and communicates with the resources controllers to realize various integrated resource management functions. Examples of integrated resource management functions include resource allocation and migration, real-time diagnosis, performance optimization, green networking, and fault tolerance.

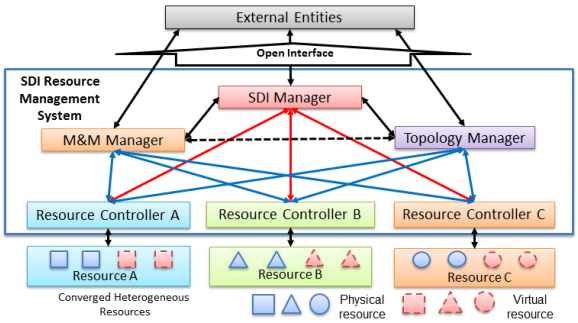

Figure 1: SDI Architecture

The SDI manager, topology manager, and monitoring and measurement manager have open APIs for external entities. The open APIs allow external entities to leverage the capabilities provided by SDI.

3. MONARCH SYSTEM

MonArch is the implementation of the monitoring and measurement manager in SDI. it is a system designed to tackle the challenges in monitoring and analytics in the cloud environment. MonArch has the following three highlights: 1) Integrated monitoring for heterogeneous resources: MonArch is capable of monitoring different types of resources in different layers, such as physical servers and network equipments, virtual machine, virtual network, web server, web application, and etc.; 2) Support for flexible queries: MonArch allows system administrators to execute and submit complex analytics task. Some example of the analytics tasks are: detection of security attack, performance diagnostics, network traffic classification, and resource allocation; 3) Scalable and Efficient Data Analytics: MonArch is capable of efficiently monitoring and processing large amount of data. It uses Apache Spark system for scalable stream and batch processing tasks.

The implementation of the MonArch system is shown in figure 2. Since we are building MonArch to work closely with OpenStack, and Ceilometer is the official telemetry component in OpenStack, we integrate Ceilometer as part of MonArch. The Agent and Super Agents are responsible for collect monitoring data from OpenStack, OpenFlow, NetFlow, and user applications. Collected monitoring data are sent to the Messaging System for queueing purpose. Historical monitoring data are stored in the HDFS distributed file system to support batch processing job. We use Apache Spark for both stream processing and batch processing. For batch processing, Spark jobs read and process monitoring data

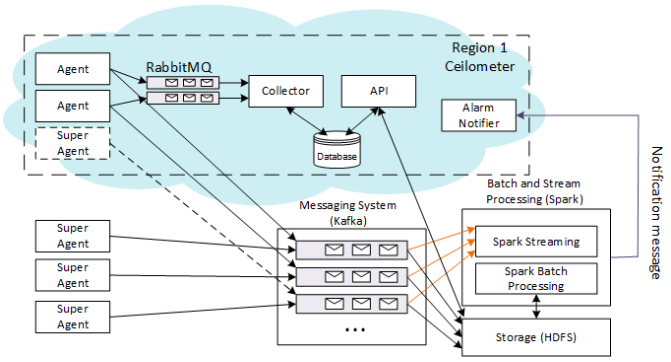

Figure 2: MonArch Detail

from HDFS to produce results. For stream processing, monitoring data are read directly from the Messaging System to reduce latency. In the current implementation, MonArch can collect, store and process monitoring data from heterogeneous resources and perform flexible data analysis to correlate data from different sources. Each component in this architecture is scalable.

In this demo, we will demonstrate the flexible analytics capability of the MonArch system. MonArch is currently deployed in the SAVI Testbed for monitoring purpose. Administrators can submit analytic jobs that process the production monitoring data. A dashboard is provided for visualizing the results. Since MonArch is capable of conducting both stream processing and batch processing, we will divide our demo into two parts: 1) batch processing; 2) stream processing.

3.1 Demo 1: Virtual machine communication analysis

For this demo, we will submit an analytics job to MonArch that create a VM communication graph by processing the historical monitoring data (including VM metrics, OpenFlow flow bandwidth usage, Application level data). This communication graph shows the basic interaction between VMs and the status of the applications that are running on the VMs. Figure 3 shows an example of visualizing the communication graph in the MonArch dashboard.

3.2 Demo 2: Anomaly detection.

Anomaly detection is another analytics capability of the MonArch system. MonArch is continuously performing stream processing on multi-layer monitoring data and identify anomalies. For this demo, we will launch an attack at an Apache web server running in a virtual machine. We will show how MonArch can detect an anomaly caused by the attack, and analyse monitoring data to show the communication graph of the abnormal VM cluster before and after the anomaly is detected.

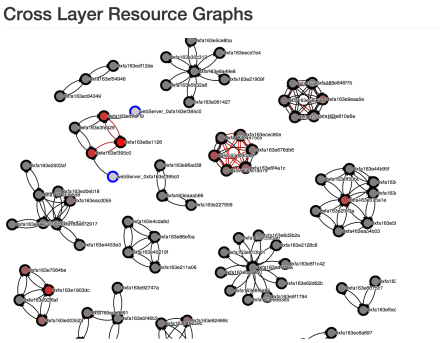

Figure 3: Virtual Machine Communication Graph

4. CONCLUSIONS

In this demo paper, we presented the monitoring and measurement manager in the SDI architecture and its implementation called MonArch. Two use cases are used to demonstrate MonArch flexible analytics capability and scalability. For our future work, we would like to improve the table placement of the historical monitoring data stored distributed file system to improve the processing efficiency.

5. REFERENCES

[1] J.-M. Kang, H. Bannazadeh, and A. Leon-Garcia. SAVI testbed: Control and management of converged virtual ICT resources. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 664–667. IEEE, 2013.

[2] J.-M. Kang, T. Lin, H. Bannazadeh, and A. Leon-Garcia. Software-Defined Infrastructure and the SAVI testbed. In *9th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TRIDENTCOM 2014)*, 2014.