FU-JIN : A Cloud Computing Environment Visualization System for Specifying Points of Failure

Tomonori Ikuse Shinya Kanda Masato Jingu Kyohei Moriyama
Masatoshi Enomoto Hiroaki Hazeyama Takeshi Okuda
Graduate School of Information Science, Nara Institute of Science and Technology
Takayama 8916-5, Ikoma, Nara, Japan
{tomonori-i, shinya-ka, masato-j, kyohei-m, masatosi-e, hiroa-ha, okuda}@is.naist.jp

Abstract

Distributed virtualization technologies in a cloud computing environment achieve high availability of computing resources and extend the scalability of application services. However, these technologies make it difficult to figure out data query paths and command paths in the cloud computing environment. To discover the points of failure in a cloud computing environment, tracing command paths and data query paths across all components of a cloud computing environment is required. We propose a log-visualization system, called "FU-JIN", that brings to light command paths and data query paths over a cloud computing environment to specify points of failure. In this paper and our poster, we present the concept and the design of FU-JIN for a private cloud computing environment.

1. Introduction

A cloud computing environment has stacked architecture[1], we call the XaaS stack (includes SaaS, PaaS and IaaS), which composed of visualization technologies and distributed technologies. In cloud computing environments, defective behaviors of a cloud application may be caused by errors on virtualized resources and/or defective behaviors of another cloud application. Because of the hierarchical architecture and the abstraction of components in a cloud computing environment, grasping lower layer behaviors from an upper layer is difficult, and vice versa. In addition, tracing job executions is hard work due to the aggregation, distribution and multiplexing of resources. Thus XaaS administrators have difficulties recognizing defective behaviors and specifying points of failure immediately and efficiently.

Tackling to the above mentioned obstacles, we propose a visualization system called "FU-JIN". FU-JIN dynamically visualizes the cloud computing environment command paths (Fig.1(a)) and data query paths (Fig.1(b)) according to XaaS administrators' viewpoints by correlating system log among all components. Correlations of data query paths and command paths are available to specify points of failure across XaaS layers, applications, resources.

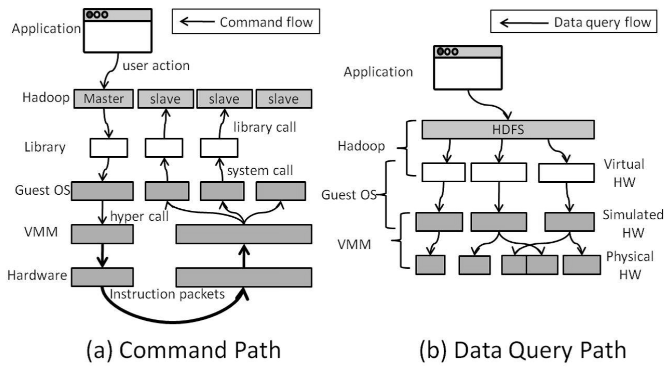

Figure 1. Visualized Paths

2. Requirements and Proposal

A cloud computing environment has a hierarchical architecture and made of an aggregation/distribution/multiplexing of components. Therefore, it is needed to monitor each component and correlate events across multiple layers and cloud applications. A visualization is an effective way to display the resulting correlation to administrators.

XaaS administrators' viewpoints also should be supported. Because each XaaS administrator needs information following different perspectives when the defective behavior is detected in their services. For example, SaaS administrator must check their application programs and the status of resources attached to their application to specify points of failure. In the case of IaaS, IaaS administrators must inspect the origin of the hardware failure and verify the running services in their own infrastructures.

When taken together, requirements are as follows: the monitoring system should 1) supports each XaaS administrators' viewpoints, 2) supports scalability and real time visualization mechanisms for specifying points of failure. Companies and researchers partly attempt to manage cloud computing environments[2, 3], but they are not able to support the administrators' perspective in XaaS layers. And it is difficult to specify points of failure immediately when the defective behavior of a cloud application is caused by another cloud application's error.

In order to meet these requirements, FU-JIN visualizes "command path" and "data query path" with each administrators' perspectives to detect defective behaviors and specify points of failure. FU-JIN also provides scalable monitoring by employing a distributed arrangement of monitoring agents (MAs).

3. Design of FU-JIN

FU-JIN monitors the whole cloud computing environment and visualizes command paths and data query paths. Fig.2 shows the design of FU-JIN's prototype. MAs are located within each Guest OSes and Host OSes. FU-JIN provides web-based visualization to administrators. FU-JIN is assumed to be deployed in a private cloud computing environment. At this time, we don't consider privacy concerns related to FU-JIN's monitoring approach, for example, privacy and authorization for accessing MAs.

FU-JIN visualizes query and command paths across the whole system from administrators' viewpoint on each XaaS layer. For example, in the SaaS layer, if a SaaS administrator clicks on his service instance, he can visualize command paths (Fig.1(a)) and data query paths (Fig.1(b)). These paths are expressed in the SaaS administrator's perspective. From fig.1(a) an administrator can find that no instructions reached the rightmost slave and all instruction paths stopped at the Guest OS layer. In addition, because all commands flow through network, he can recognize the network configuration of the Guest OS layer as a point of failure.

MAs monitor library calls, system calls, hyper calls and network packets so that FU-JIN visualizes such command paths in real time. And they also monitor page tables to visualize data query. These data are collected from various sources such as Hadoop, libvirt, the Guest OS, Xen and hardware by using syslog-ng, SNMP, IPMI and libvirtAPI. First, the MA in the Guest OS sends log data to the MA in the Host OS by syslog-ng function, then the MA in the Host OS collects a certain number of log data and tags them with time information. When the collected log data reaches

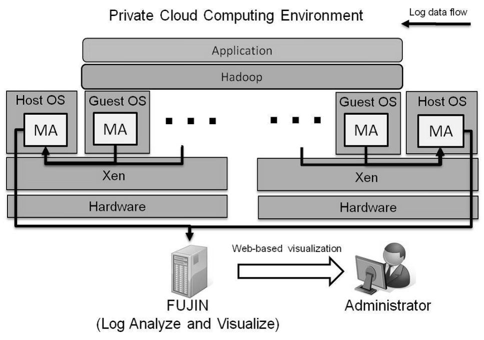

Figure 2. prototype of FU-JIN

a fixed threshold, the MA in the Host OS sends the log data to the analyzer.

Another consideration is how to analyze log data. Time based log data analysis is common and useful in tracing data queries and commands. Thus log data are analyzed by the analyzer based on the time information tagged by each MA and the relationships between the command call issuer and the call receiver.

4. Conclusion and Future Work

We introduced an approach based on cross-layer correlations of command paths and data query paths to find points of failure in a private cloud computing environment. We proposed and designed FU-JIN, a prototype to implement our proposal. We will try to adapt FU-JIN in real system operation. Through this phase, we will improve FU-JIN to comply with public cloud environment requirements.

5. Overall Structure of our Poster

We will first explain the problems and requirements for a cloud computing management system showed in section 1 and 2. Then, we will introduce the proposal and the design of FU-JIN prototype with figures (Fig.1 and 2). Finally, we give highlights on our future work.

References

[1] Cloud Security Alliance. Security guidance for critical areas of focus in cloud computing v2.1, December 2009.

[2] A. Konwinski and M. Zaharia. Anomaly detection in the data center. Technical report, RAD lab, 2008.

[3] M. Kutare, G. Eisenhauer, C. Wang, K. Schwan, V. Talwar, and M. Wolf. Monalytics: online monitoring and analytics for managing large scale data centers. In *ICAC '10: Proceeding of the 7th international conference on Autonomic computing*, pages 141–150. ACM, 2010.