
Hyun-wook Baek  
University of Utah  
baekhw@cs.utah.edu

Abhinav Srivastava  
AT&T Labs Research  
abhinav@research.att.com

Jacobus Van der Merwe  
University of Utah  
kobus@cs.utah.edu

Abstract
Infrastructure-as-a-Service (IaaS) cloud platforms simplifies the work of cloud tenants by providing clean virtual datacenter abstractions and the means to automate tenant system administrator tasks. When things go wrong, however, the inherent multi-player multi-layer environment of IaaS platforms complicates troubleshooting and root cause analysis. To address these concerns we present our work on CloudSight in which cloud providers allow tenants greater system-wide visibility through a transparency-as-a-service abstraction.

I. Overview
Because it is a multi-player multi-layer environment where tenant and provider in effect share administrative duties, troubleshooting in an infrastructure-as-a-Service (IaaS) cloud platform is an inherently difficult task. Specifically, problems might be due to a tenant’s own setup in the virtual datacenter abstraction layer, or it might be due to problems in the underlying cloud platform. Neither the cloud tenant, nor the cloud operator has a complete system wide view, often resulting in time consuming and expensive interactions between tenant and provider to identify and solve problems.

CloudSight is designed to address these concerns. With CloudSight, cloud providers will provide an API that allows cloud tenants to troubleshoot cloud issues in a system-wide manner. Conceptually, our goal is to provide cloud tenants with similar visibility into the cloud system as a whole, as they were accustomed to in a physical data center.

CloudSight collects event logs in a various IaaS components, learns the semantics of the logs using clustering and reconstructs the event logs into a tenant-understandable resource graph. Especially, for IaaS tenants, understanding the details of a system-wide view of an IaaS environment is significantly more involved than understanding a physical data center. Resource graph is reflecting historical changes of states of tenant’s resources so that tenants can trace back to the root cause of a problem by querying against their resource graphs.

Internally, CloudSight semi-automatically learns semantics of complex event logs collected from various places of IaaS platform using Key Clustering. Key clustering classifies relationship between keys into either synonym, hyponym, hypernym or none, and induces meaning of a key from its co-related keys. Thus, if an administrator offers a set of keys with their meanings, CloudSight can learn the semantics of other keys related with the keys.

To demonstrate the efficacy of our framework, we develop two tenant-oriented applications on top of CloudSight. 1) Time-traveling Cloud Debugger, which assists tenants to track their resources by setting timepoint similar to setting a breakpoint in a typical software debugger. 2) Resource State Verifier, which determines if a given resource is in an inconsistent state or not.
II. Innovation

- We model the CloudSight service as a resource graph which naturally fits tenants’ view of their cloud resources and allows us to effectively deal with the challenges involved with our approach.
- We apply a novel key clustering approach to stitch together event logs from different data sources to produce the resource graph and to reduce domain knowledge required to realize our approach.

III. HPC and Science Relevance

As the emergence of cloud computing, scientists and engineers could reduce costs by running high performance computing in the cloud. However, by the nature of HPC, running HPC in cloud is more sensitive to the state of the internal infrastructure. We believe CloudSight also allows people to diagnose and explain anomalous behavior or results of HPC in cloud.

IV. SCinet and R&E Requirements

- We need a booth for demonstration.
- We need a network connection to our experimental cloud infrastructure.

V. Network Topology

Our experimental cloud infrastructure will be deployed in Emulab. The experiment will consist of three compute nodes, a controller node, a network node, a database node and a gateway node. The network will be configured as the following diagram. On the nodes, we are going to deploy OpenStack with CloudSight.