Layering of the Provenance Data for Cloud Computing

Muhammad Imran and Helmut Hlavacs

Research Group Entertainment Computing, University of Vienna, Wien, Austria
imran.mm7@gmail.com,
helmut.hlavacs@univie.ac.at

Abstract. With the recent advancements in distributed systems, Cloud computing has emerged as a model for enabling convenient, on-demand network access to a shared resource pool of configurable elements such as (networks, servers, storage, applications, and services). Various applications are developed and deployed into the Cloud following the layered architecture. The layered approach includes infrastructure, virtualization, application, platform and client tiers. Provenance (the meta-data), is the information that helps cloud providers and users to determine the derivation history of a data product, starting from its origin. Each layer in the Cloud has its own provenance data and generally, provenance data for each layer address different audience. For example, Cloud providers are interested in the infrastructure provenance data to verify the high utilization of resources through audit trials. Cloud users on the other hand are interested in the performance of the deployed application and the verification of experiments. In this paper, we present various queries regarding the provenance data for different layers of Cloud. Hereby, we integrate the provenance data from individual layers and highlight the importance of integrated provenance. We also outline the relationship between various layers of the Cloud by using the integrated provenance.

1 Introduction

Cloud computing is generally defined by its distributed model of utility computing which offers virtualization of resources (storage, computation, networking) and provisioned to users "on demand" and "pay as you go" basis. This new paradigm attracted the research community and businesses to host and execute their complex scientific applications [1] [1, 2]. In this model, applications are deployed and executed by using the type of service offered in the Cloud. These services reside on various layers or tiers of the Cloud architecture. For instance, Cloud providers are interested in the IaaS (Infrastructure as a Service) [2] layer of the Cloud, which supports virtualization of resources to enable computation, storage and communication. These resources are utilized by Cloud applications e.g., getting email [3] or for sharing documents, often termed as SaaS (Software as a Service). To fill the gap between IaaS and SaaS, the PaaS (Platform as a

[1] http://aws.amazon.com/swf/ [2] www.eucalyptus.com [3] www.gmail.com

J.J. Park et al. (Eds.): GPC 2013, LNCS 7861, pp. 48–58, 2013.

Service) layer is used by developers to customize and easily develop, deploy and manage Cloud-aware applications e.g. salseforce.com [4], WSO2 [5] and/or providing Enterprise Service Bus (ESB) [6] as a service.

Provenance is the metadata which describes the derivation history of an object. This data includes the source and intermediate datasets and processes involved to create the object [3]. In computing science, provenance is an important ingredient for the verification and reproduction [4, 5] of scientific experiments. The architecture of Cloud computing is divided into various components and these components are placed on top of each other [6]. IaaS, PaaS and SaaS are the types or layers which are mostly used in the Cloud environment. The development and execution of Cloud-aware applications follow this layered architecture and each layer contributes specific metadata (provenance) for the overall application. Subsequently each layer in the Cloud has its own provenance and specific importance to that particular layer. For example, the provenance data at the IaaS layer is important to the Cloud provider for resources utilization and fault tracking [7]. Cloud users (research community) are more interested in the execution of their deployed applications; the datasets which are produced/consumed, and the processes used for the production of the result.

Moreover, the provenance collection at individual layers e.g., for IaaS it can ensures the appropriate allocation and usage of the resources. Similarly, in case of faults and errors appropriate actions can be taken to resolve them accordingly by using the provenance [8]. When provenance is integrated from individual layers, it provides the in-depth details of the relationships which exist among various layers of the Cloud while executing a particular application. The integrated provenance data provides multiple views and enables Cloud providers to keep track of their resource usage, application and service collaboration for users and deployment/testing usage for developers.

For understanding the Cloud layered approach and the overall provenance data at each layer, in this paper we have developed a Content Relationship Management (CRM) application. With this particular CRM application, we differentiate between various layers of Cloud and their corresponding provenance data. We provide detail of the CRM application and its various parts from the users perspective, the Cloud provider and application developer. Following are the key contributions of this paper:

- to provide an overview of the Cloud layered technology and the presentation of provenance data for each layer.
- to present various queries and their visualization for the individual layers of the Cloud and for the collective provenance data.
- to highlight the importance of integrated provenance using an example.
- to evaluate the overhead for provenance collection at individual layers.

The rest of the paper is organized as follows. Section 2 provides the related work of provenance in the field of e-Science. Section 3 discusses the requirements for building a CRM application in the Cloud and its individual components.

[4] `www.salesforce.com` [5] `www.wso2.com` [6] `http://www.mulesoft.org/`

Section 4 provides a brief overview of layers in Cloud computing and Section 5 present the various quires for the provenance data on individual layers, their visualization and discusses the importance of integrated provenance data. Section 6 evaluate the collection of provenance data from different layers and section 7 concludes this paper.

2 Related Work

Application level provenance has been the major attraction in grid, distributed and workflow computing [5]. The techniques used in these environments are to capture provenance in Service Oriented Architecture(SOA) e.g., PASOA [9]. Recently, the research community focused on the usage of provenance for Cloud computing while describing and addressing the various challenges offered by this new paradigm [10, 11].

Previously [12], we proposed a framework which addresses the various challenges offered by Cloud technology and present the mechanism to incorporate the collection and storage of provenance for the Cloud IaaS. On the development layer of the Cloud, e.g., in mule ESB and WSO2 carbon platforms, various parts of application are integrated together that communicate based on different protocols and languages. Integration of provenance into the development layer will clearly identify the current status of any application, changes made by different developers of a group or team and information about the old and current version of the services and applications. There are other work which consider provenance at the layers like a web browser [13] and virtual machine [14].

To establish the importance of integrating provenance data from different layers, Muniswamy Reddy et. al. [15] discussed the layering of provenance data for workflow execution. However their work focused on combining the provenance data from a workflow engine, web browser and the python wrapper by extending the Provenance Aware Storage System (PASS) [16]. For Cloud environment, a short survey about various techniques from Grid and distributed computing are discussed to track the data in Cloud by using a layered architecture [17]. In this paper, we extend our provenance framework [18] for the platform and software layers of the Cloud.

3 Scenario Description (Components of CRM Application)

In this scenario the objective is to automate the installation of a sample CRM application. The sample CRM application consists of three main components which are: 1) the web server 2) the database server and 3) the client application. **Component 1** is the web server where we have two different web services. Web service 1 takes the data from the user and submit or sync it to the database server. This sync can be performed either for one particular item e.g., contacts or for over all data e.g., contacts, appointments and tasks. Web service 2 takes

the data from the server and sync it to the client application. Again, the sync process is for one particular item or overall data.
Component 2 is the database server. It is mySQL database with various tables containing the information about an organization or a group e.g., contacts, appointments and tasks. The contacts table contains first name, last name, job title, group name etc. Appointments table contains information like place, time, appointment with, number of people attending, topic and location. Tasks table contains information like sender, receiver, title, subject, description etc.
Component 3 is the client application for a user to to see the tasks assigned to him/her and list the appointments. This also includes, who assigned the tasks, meetings and appointments time, members involved and location.
Summary: To link all these resources with each other, a script is required which deploys the application on Cloud and host the various components. Such a scrip will be passed to Cloud controller via user data. The end result would be a deployment of CRM system. Figure 1 presents the steps involved in deployment of such an application to the Cloud. We consider three resources to host web services, database server and client application. Each installed components requires some prerequisite and those need to be installed and configured. While deploying CRM, we observe and provide the details of the various components of Cloud and related provenance data.

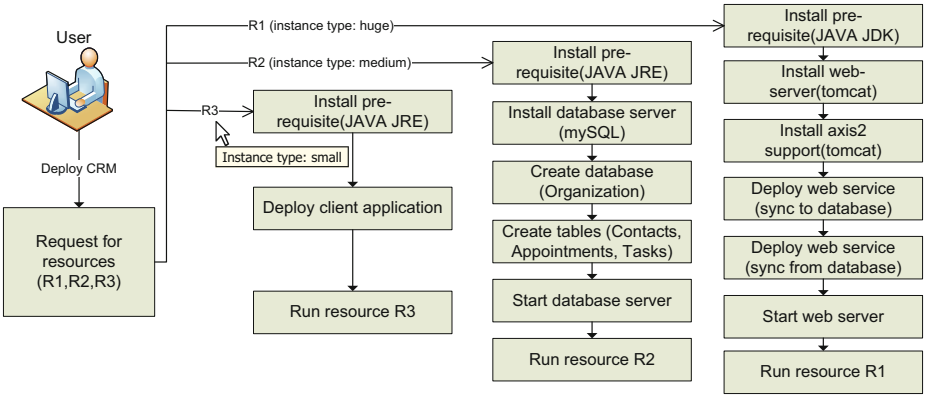

Fig. 1. Steps involved in deployment of CRM application to Cloud

4 Cloud Layered Architecture

The Cloud architecture is perceived differently by various research community and businesses [19, 20]. Mainly we consider the following layers/components.

- Infrastructure layer: provides physical and virtual resources for storage, communication and computation e.g., Eucalyptus.
- Platform layer: provides tools and libraries to ease the development cost and effort for building Cloud aware application e.g., WSO2.

– Software layer: The applications which are provided by various organizations to vast number or users e.g. web application (gmail).

Figure 2 presents the main components in Cloud computing from the view point of a user, developer and Cloud resources provider. To connect the layers, there is always a middle-ware in between. We collect the provenance data on that middleware level. Previously we explained the mechanism to collect IaaS provenance data in [12] by using the interceptor mechanism and why such data is important. For this work, we extend the same mechanism, guideline and architecture of provenance towards PaaS and SaaS layers of Cloud computing.

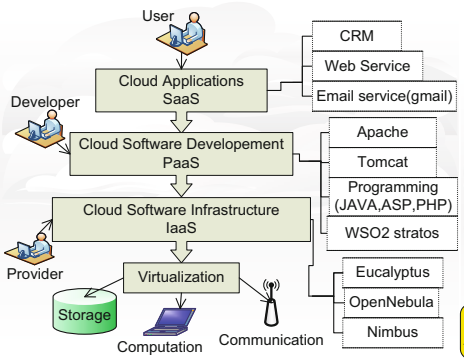

Fig. 2. Layered architecture of Cloud

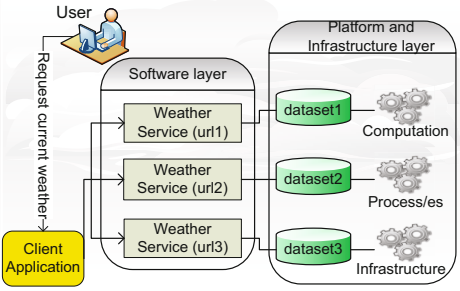

Fig. 3. Weather request from Cloud

4.1 Query and Visualization

The presentation of the collected provenance data is very important for users, administrators and application developers in Cloud and it depends on the submitted query. For a particular query, we may analyze the provenance data of one particular layer or integrated provenance. For example, when an administrator submit the following query:
Visualize the instance types from cluster1, requested by various users where the number of request are more than 100
This query requires the analysis of the infrastructure provenance. This provenance data is stored in a well defined xml file where the nodes represent the individual objects and the edges represent the relationship which exists in the provenance data. The numbers of relationship varies for the submitted query. For this particular query, the following relationships can be defined: i) request of instances types for cluster1 ii) relation of instances to various users iii) and relation of users to the cluster and instances. For analysis of this query and defining the relation, we apply pull based mechanism for the extraction of data from provenance. After the analysis of the submitted query and defining the relationships, we present the results in the graph form. These graph can be changed on run time and the results can be visualized in line, bar and pie forms.

5 Provenance and Cloud Layers

Following the layered architecture of Cloud, provenance is also divided into different layers. Each layer presents a different application domain for the usage of provenance data. The sections below investigate the provenance data and various queries which require individual and/or the integrated provenance.

5.1 IaaS Provenance and Queries

The infrastructure layer provides computational, storage and communication resources for the application deployment and services execution. Various parameters are considered when defining provenance data for IaaS Cloud e.g., 1) types of resources 2) types of instance 3) information about users 4) time taken by users for instances 5) data submitted by users before running a resource 6) information about cloud, clusters and node services. In the CRM application, these parameters maps to user (admin), resource types (R1, R2, R3), instance types (small, medium and huge), data (java jdk, tomcat, axis2 and mySQL versions).

Many applications can be defined depending on the granularity of the provenance data e.g., 1) to use the provenance data for auditing the usage of resources in Cloud. Provenance data can clearly mark the usage of resources from various clusters and nodes according to time, users, and resources types. 2) to find the similar requests in Cloud which are based on the instance types and user data. These similarities define patterns and are used for the efficient utilization of Cloud resources. The efficient utilization is achieved by reusing the existing running resources and predicting the upcoming requests. 3) the provenance data of various images is used for tracking malicious images uploaded in public Cloud and managing the access rights to various images [21]. Following are few example queries which can be generated for the Cloud infrastructure:

(i) visualize the instance types from last 24 hours (ii) visualize the standard instances from last 48 hours (iii) visualize the memory request from last week (iv) visualize the request for resources from most used to least used (v) list the prerequisite (user data) from R1 (vi) list the deployment time for resource R1, R2 and R3 (vii) validate the setup of CRM application.

The combination of the infrastructure provenance data with physical machine provenance e.g., memory, CPU utilization and the disk usage can further elaborate the provenance query:

- visualize the disk and memory usage for the most requested resources type.

These queries provides the administrator with an overview of the resources usage, instance types and data requested by users. Left side of the figure 4 present the memory requests for a particular cluster grouped for various users and right side of the figure 4 present the memory requests from various users in time by using the visualization module. Due to limited space, we will not present visualization of the provenance data for other layers.

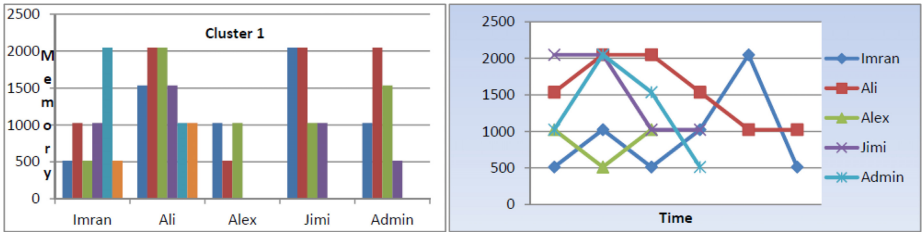

Fig. 4. Memory requests for a Cluster from various users

5.2 PaaS Provenance and Queries

Platform layer in Cloud provides the functionality for the development of new applications. This layer enables and manges the delivery of services that uses various communication protocol e.g. HTTP, XML and SOAP etc. The designer of these applications is responsible for assets availability and the management of application services pool. For example, Cloud is a favorable environment in stress testing, where a lot of resources are required just for a particular period of time. For this work, we consider WSO2 platform and it's various components.

WSO2 Carbon is the award winning PaaS and it provides many features to developers for building Cloud aware applications e.g., Enterprise Service Bus (ESB), Business Process management (BPM) and more. While developing applications using WSO2, the complex application is divided into various parts. Members of a team/s work on different components of a complex application. In the CRM application various parameters which are considered for the provenance data of platform layer are: 1) composition of the web services 2) the interaction mechanism between web services, database and web service engine, that is utilized by various protocols of communication 3) the interaction mechanism between web services and client application 4) composition of the database and corresponding tables and their structure. When one developer makes a change in a web service, other members will be able to find that particular change using the provenance data. Any change on platform layer e.g., uploading a new version of the web service will create a new node in the provenance data and hence the status will be updated. Some important queries on platform layer are following:

(i) visualize the components of the application where most bugs are found (ii) the identity of a person who made a change in CRM and the time when the change was made (iii) display the changes made to web services in last one month (CRM)

Consider a situation where the infrastructure is changed e.g, mySQL server is updated. The updated version does not support the existing communication mechanism with the deployed web services. This requires a change in the web services at the platform layer. This relation which exists between platform and infrastructure layer is exposed by integrating the provenance data from individual layers. The integrated provenance data and the corresponding relation will highlight the reason for any communication failure.

5.3 SaaS Provenance and Queries

SaaS is the application running on a Cloud platform. Various types of applications are deployed and executed on Cloud e.g., workflow, CRM and web applications. The provenance data of this layer depends on the type of application. In general, applications are deployed by using Service Oriented Architecture (SOA) in distributed computing. The important provenance parameters in SOA architecture and related queries are following:

(i) time taken by a particular application to generate the result (ii) time taken by individual services and components of the application (iii) tracking the dataset which are consumed and produced during the process (iv) information about the users who are invoking the services (v) the query about services or components taking part while executing the application e.g., services involved in executing a workflow (vi) input and output parameters passed to a particular service and/or method.

In the CRM application, the analysis of various events is an important aspect for organizations. The provenance data about users, time, and events is used for analysis and to get important informations like; the locations of the event, total time for the event, members who joined the event, the organizers of the event etc. There are other aspects of the provenance data for software layer e.g., trust, reliability and authenticity.

Considering a situation where changes are made to the web services on platform layer. This will require appropriate changes to the client application. If the client application is not updated, any sync process from database to the client application will result in failure. The provenance data from the client application will highlight the failure. User can use the provenance chart to find the failure, but it's reason is not clear until we layer the provenance data from platform to the software layer. Layering the provenance data will further explain the reason of failure and related data for changes made on platform layer.

5.4 Advantage of Integrated Provenance Data

In the above sections, we deployed the CRM application into the Cloud environment. We collected the provenance data on individual layers, provided various parameters, queries and the relations which exists between layers. Considering the fact that Clouds are abstract and the various layers are hidden from the user, we present the example in figure 3. Users request for the current weather information using a particular city and country. The client application randomly chooses one of the weather web services which are provided by different organizations. The selected service returns the current weather information. Since, these services use different datasets for the calculation of the weather, the result is not always the same. In scientific environment, it is important to know why the results differs from each other.

Without layering: Each layer of provenance gives valuable information. The software layers provides the provenance data for the selected web service and methods. The platform provenance provides the information regarding the

datasets which are consumed and the algorithm which is used for the calculation. The infrastructure layer provenance data gives information regarding the Cloud provider and the location of the computation, storage, and communication devices.

With layering: The integrated provenance data from software, platform and infrastructure layer identifies the datasets which are used, the web service which is consumed and information about the Cloud provider. These information provides the relation between various layers and hence highlight the reason that why different results are not always the same.

6 Overhead Evaluation

The integrated solution of provenance into Cloud infrastructures particularly for e-Science applications causes extra overhead of calculation and storage. The calculation overhead is the extra time needed for the collection, parsing and storage of the provenance data. In our experiments, the overhead is calculated for the individual layers of Cloud using the CRM application. The calculation is performed for the various components of the CRM application which correspond to Cloud layers. At the infrastructure layer, we tested the eucalyptus Cloud with *node controller* and *cluster controller* services. Platform layer is tested with WSO2 *application service* and *enterprise service bus*. The software layer is evaluated for the web services *snyntodatabase* and *syncfromdatabase* in CRM.

Table 1 presents the performance overhead of provenance from various components in Cloud and CRM application. The maximum times are the exceptional cases and therefore average time was calculated from multiple runs (50) of components and layers. The average time presents the overhead for collection, parsing and storing of the provenance. Formula 1 is used to calculate the overall overhead by summation of individual overhead from software, platform and infrastructure layer.

Depending on the granularity and storage mechanism, time required for provenance may slightly vary. The very low overhead explains the utility of our provenance collection technique which follows the interceptor based approach for collection and link based approach to store the data [12]. Given the overall advantages of provenance, this extra overhead is negligible.

$$Total\,Overhead = \sum_{i=0}^{n}(\mathbf{S})i + \sum_{i=0}^{n}(\mathbf{P})i + \sum_{i=0}^{n}(\mathbf{I})i \tag{1}$$

7 Conclusions

In this paper we emphasis on the provenance data at the various layers of Cloud infrastructures for the applications deployed there in. To achieve this, first, we identified individual layers in Cloud computing and presented the related provenance data for each layer. Then various queries were explored that could be answered using the hierarchical architecture of Cloud and deployed applications.

Table 1. Calculation time overhead for provenance in milliseconds

Cloud layers	Max time(ms)	Min time(ms)	Avg time(ms)
Software (CRM application)	18	2	7
Platform (WSO2 AS)	22	1	3
Platform (WSO2 ESB)	12	1	2.5
Infrastructure (Eucalyptus NC)	15	2	4
Infrastructure (Eucalyptus CC)	20	7	12
Combined			26 ms

These queries utilized the provenance of individual layer or the integrated provenance data. Further, the identification of relations is provided which exists for one particular layer or in the integrated provenance data. By exploiting the Cloud architecture, we divided the provenance into various layers and presented the mechanism to query and visualize different requests from the perspectives of various stakeholders including users, developers and Cloud providers themselves.

References

[1] Deelman, E., Singh, G., Livny, M., Berriman, B., Good, J.: The cost of doing science on the cloud: The montage example (2008)
[2] Vöckler, J.S., Juve, G., Deelman, E., Rynge, M., Berriman, B.: Experiences using cloud computing for a scientific workflow application, pp. 15–24. ACM, USA (2011)
[3] Barga, R.S., Simmhan, Y.L., Chinthaka, E., Sahoo, S.S.: Jackson: Provenance for scientific workflows towards reproducible research. IEEE Data Eng. Bull. (2010)
[4] Bose, R., Frew, J.: Lineage retrieval for scientific data processing: a survey. ACM Comput. Surv. 37(1), 1–28 (2005)
[5] Simmhan, Y.L., Plale, B., Gannon, D.: A Survey of Data Provenance Techniques. Technical report, Computer Science Department, Indiana University (2005)
[6] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee: Above the Clouds: A Berkeley View of Cloud Computing (2009)
[7] Imran, M., Hlavacs, H.: Applications of provenance data for cloud infrastructure. In: Eighth International Conference on Semantics, Knowledge and Grids (SKG), pp. 16–23 (2012)
[8] Crawl, D., Altintas, I.: A provenance-based fault tolerance mechanism for scientific workflows. In: Freire, J., Koop, D., Moreau, L. (eds.) IPAW 2008. LNCS, vol. 5272, pp. 152–159. Springer, Heidelberg (2008)
[9] Miles, S., Groth, P., Branco, M., Moreau, L.: The requirements of recording and using provenance in e-Science experiments. Technical report (2005)
[10] Muniswamy-Reddy, K.K., Seltzer, M.I.: Provenance as first class cloud data. Operating Systems Review 43(4), 11–16 (2009)
[11] Muniswamy-Reddy, K.K., Macko, P., Seltzer, M.: Provenance for the cloud. In: FAST 2010, pp. 197–210. USENIX Association (2010)
[12] Imran, M., Hlavacs, H.: Provenance in the cloud: Why and how? In: The Third International Conference on Cloud Computing, GRIDs, and Virtualization, pp. 106–112 (2012)

[13] Margo, D.W., Seltzer, M.I.: The case for browser provenance. In: Workshop on the Theory and Practice of Provenance (2009)
[14] Macko, P., Chiarini, M., Seltzer, M.: Collecting provenance via the xen hypervisor. In: Workshop on the Theory and Practice of Provenance (2011)
[15] Muniswamy-Reddy, K.K., Braun, U., Holland, D.A., Macko, P.: Maclean: Layering in provenance systems. In: USENIX, USA (2009)
[16] Muniswamy-Reddy, K.K., Holland, D.A., Braun, U., Seltzer, M.I.: Provenance-aware storage systems. In: USENIX, pp. 43–56 (2006)
[17] Zhang, O.Q., Kirchberg, M., Ko, R.K.L., Lee, B.S.: How to track your data: The case for cloud computing provenance. In: CloudCom 2011, pp. 446–453 (2011)
[18] Imran, M., Hlavacs, H.: Provenance framework for the cloud environment (iaas). In: The Third International Conference on Cloud Computing, GRIDs, and Virtualization (2012)
[19] Youseff, L., Butrico, M., Da Silva, D.: Toward a Unified Ontology of Cloud Computing. In: Grid Computing Environments Workshop, GCE 2008, pp. 1–10 (2008)
[20] Rochwerger, B., Breitgand, D., Levy, E., Galis, A., Nagin, K.: The reservoir model and architecture for open federated cloud computing (2009)
[21] Wei, J., Zhang, X., Ammons, G., Bala, V., Ning, P.: Managing security of virtual machine images in a cloud environment. In: CCSW, pp. 91–96. ACM (2009)