Nephele: Dynamic Allocation and Parallel Execution Security Model in Cloud Computing

M. Ranjith Kumar, S. Arunmozhi Varman and D.C Joy Winnie Wise
Department of CSE, Alpha College of Engg, Chennai, T.N, India.

ABSTRACT
In present year ad-hoc parallel data processing has emerged to be one of the killer applications for Infrastructure-as-a-Service (IAAS) clouds. Major Cloud computing companies have started to integrate frameworks for parallel data and making it easy for customers to access these. However, the processing frameworks which are currently used have been designed for static, homogeneous cluster setups and disregard the particular nature of a cloud. Consequently, the allocated compute resources may be inadequate for big parts of the submitted job and unnecessarily increase processing time and cost. In this project the data processing framework to explicitly exploit the dynamic resource allocation offered by today’s IAAS clouds for both, task scheduling and execution is constructed. Particular tasks of a processing job can be assigned to different types of virtual machines which are automatically instantiated and terminated during the job execution.

ARTICLE INFO
Article history:
Received: 12 September 2012;
Received in revised form: 1 February 2013;
Accepted: 19 February 2013;

Keywords
Nephele, CCP, Map reducing.

Introduction
Today’s processing frameworks typically assume the resources they manage consist of a static set of homogeneous compute nodes. Although designed to deal with individual node failures, they consider the number of available machines to be constant, especially when scheduling the processing job’s execution. While IaaS clouds can certainly be used to create such cluster-like setups, much of their flexibility remains unused.

One of an IaaS cloud’s key features is the provisioning of compute resources on demand. New VMs can be allocated at any time through a well-defined interface and become available in a matter of seconds. Machines which are no longer used can be terminated instantly and the cloud customer will be charged for them no more. Moreover, cloud operators like Amazon let their customers rent VMs of different types, i.e., with different computational power, different sizes of main memory, and storage. Hence, the compute resources available in a cloud are highly dynamic and possibly heterogeneous.

This model deals with dynamic environment hence the user data is insecure. In order to provide security an integration of sound signature with graphical password authentication system is designed.

The rest of the paper is organized as follows. Related work is discussed in section 2 and architecture and method of implementation is discussed in section 3. Design standards and benchmark metric are elaborated in section 4 and section 5 contains the conclusion of the paper.

Related Works
Amazon has integrated Hadoop as one of its core infrastructure services [4]. However, instead of embracing its dynamic resource allocation, current data processing frameworks rather expect the cloud to imitate the static nature of the cluster environments they were originally designed. Warneke and Odej Kao [1] proposed Dynamic allocation of resources for both task scheduling and execution using Nephele. The limitation with this is that it requires a reasonable amount of user annotations. Also it failed to discuss about the security constraints. The vast amount of data they have to deal with every day has made traditional database solutions prohibitively expensive [2]. A high-level scripting language (SCOPE) for writing data analysis jobs. Users are forced to map their applications to the map-reduce model in order to achieve parallelism.

MapReduce [7] is designed to run data analysis jobs on a large amount of data, which is expected to be stored across a large set of share-nothing commodity servers. MapReduce is highlighted by its simplicity: Once a user has fit his program into the required map and reduce pattern, the execution framework takes care of splitting the job into subtasks, distributing and executing them. A single MapReduce job always consists of a distinct map and reduce program. MapReduce has been clearly designed for large static clusters. Although it can deal with sporadic node failures, the available compute resources are essentially considered to be a fixed set of homogeneous machines.

The system our approach probably shares most similarities with is Dryad [8]. Dryad also runs DAG-based jobs and offers to connect the involved tasks through file, network, or in-memory channels. However, it assumes an execution environment which consists of a fixed set of homogeneous worker nodes. The Dryad scheduler is designed to distribute tasks across the available compute nodes in a way that optimizes the throughput of the overall cluster. It does not include the notion of processing cost for particular jobs.

Design
A. Outline and Architecture
We have designed Nephele, a new data processing framework for cloud environments. Nephele takes up many ideas of previous processing frameworks but refines them to better match the dynamic and opaque nature of a cloud. Nephele’s architecture follows a classic master-worker pattern as illustrated in Fig. 1.
Before submitting a Nephele compute job, a user must start a VM in the cloud which runs the so-called Job Manager (JM). The Job Manager receives the client’s jobs, is responsible for scheduling them, and coordinates their execution. It is capable of communicating with the interface. The cloud operator provides control over the instantiation of VMs. We call this interface the Cloud Controller. By means of the Cloud Controller the Job Manager can allocate or de-allocate VMs according to the current job execution phase. These comply with common Cloud computing terminology and refer to these VMs as instances. The term instance type will be used to differentiate between VMs with different hardware characteristics.

The actual execution of tasks which a Nephele job consists of is carried out by a set of instances. Each instance runs a so-called Task Manager (TM). A Task Manager receives one or more tasks from the Job Manager at a time, executes them, and after that informs the Job Manager about their completion or possible errors. Upon job reception the Job Manager then decides, depending on the job’s particular tasks, how many and more tasks from the Job Manager at a time, executes them, and after that informs the Job Manager about their completion or possible errors. Upon job reception the Job Manager then decides, depending on the job’s particular tasks, how many and what type of instances the job should be executed on, and when the respective instances must be allocated/deallocated to ensure a continuous but cost-efficient processing.

The newly allocated instances boot up with a previously compiled VM image. The image is configured to automatically start a Task Manager and register it with the Job Manager. Once all the necessary Task Managers have successfully contacted the Job Manager, it triggers the execution of the scheduled job. Nephele are expressed as a directed acyclic graph (DAG). Each vertex in the graph represents a task of the overall processing job, the graph’s edges define the communication flow between these tasks.

Due to the dynamic nature of cloud more than one user can access the resource at a time. Also more users can store their documents in a cloud resource hence the security is important problem. The hackers can hack the password if it is in textual format so an alternative approach is used here to overcome it. A graphical password methodology is introduced to protect the file from third person. The recovery can be done by sound system.

B. Integration of Sound Signature in Graphical Password Authentication System

Here a graphical password system with a supportive sound signature to increase the remembrance of the password is discussed. In proposed work a click-based graphical password scheme called Cued Click Points (CCP) is presented. In this system a password consists of sequence of some images in which user can select one click-point per image.

In addition user is asked to select a sound signature corresponding to each click point this sound signature will be used to help the user in recalling the click point on an image. System showed very good Performance in terms of speed, accuracy, and ease of use. Users preferred CCP to Pass Points, saying that selecting and remembering only one point per image was easier and sound signature helps considerably in recalling the click point.

C. Job Scheduling and Execution

After having received a valid Job Graph from the user, Nephele’s Job Manager transforms it into a so-called Execution Graph. An Execution Graph is Nephele’s primary data structure for scheduling and monitoring the execution of a Nephele job. Unlike the abstract Job Graph, the Execution Graph contains all the concrete information required to schedule and execute the received job on the cloud. It explicitly models task parallelization and the mapping of tasks to instances. Depending on the level of annotations the user has provided with his Job Graph, Nephele may have different degrees of freedom in constructing the Execution Graph.

D. Parallelization and Scheduling Strategies

As mentioned before, constructing an Execution Graph from a user’s submitted Job Graph may leave different degrees of freedom to Nephele. Using this freedom to construct the most efficient Execution Graph (in terms of processing time or monetary cost) is currently a major focus of our research. Unless the user provides any job annotation which contains more specific instructions we currently pursue a simple default strategy: Each vertex of the Job Graph is transformed into one Execution Vertex. The default channel types are network channels. Each Execution Vertex is by default assigned to its own Execution Instance unless the user’s annotations or other scheduling restrictions (e.g., the usage of in-memory channels) prohibit it. The default instance type to be used is the one with the lowest price per time unit available in the IaaS cloud.

Evaluation

In this section, we want to present first performance results of Nephele and compare them to the data processing framework Hadoop. We have chosen Hadoop as our competitor, because it is an open source software and currently enjoys high popularity in the data processing community. We are aware that Hadoop
has been designed to run on a very large number of nodes (i.e., several thousand nodes). However, according to our observations, the software is typically used with significantly fewer instances in current IaaS clouds. As a result, the data flow was controlled by the executed MapReduce programs while Nephele was able to govern the instance allocation/deallocation and the assignment of tasks to instances during the experiment. We devised this experiment to highlight the effects of the dynamic resource allocation/deallocation while still maintaining comparability to Hadoop as well as possible. Fig. 3 shows the performance result.

Fig. 3 Results: MapReduce and Nephele.

Conclusion

Nephele is the first data processing framework to explicitly exploit the dynamic resource allocation offered by today’s IaaS clouds for both, task scheduling and execution. Particular tasks of a processing job can be assigned to different types of virtual machines which are automatically instantiated and terminated during the job execution. Based on this new framework, we perform extended evaluations of MapReduce-inspired processing jobs on an IaaS cloud system. The dynamic allocation of resources provides high performance processing power and it allows the user to obtain the resources dynamically. The “Integration of Sound Signature in Graphical Password Authentication System” which is implemented in cloud provides security to the resources.

Nephele compute job, a user must start a VM in the cloud which runs the so called Job Manager (JM). After having received a valid Job Graph from the user, Nephele’s Job Manager transforms it into a so-called Execution Graph. An Execution Graph is Nephele’s primary data structure for scheduling and monitoring the execution of a Nephele job. In contrast to the Job Graph, an Execution Graph is no longer a pure DAG. The DAG constructed in present work is off without feedback loop, so in future work DAG can be constructed with feedback loop. Also in providing security, in future systems other patterns may be used for recalling purpose like touch of smells.

References