Cross Cloud MapReduce: an Uncheatable Map Reduce

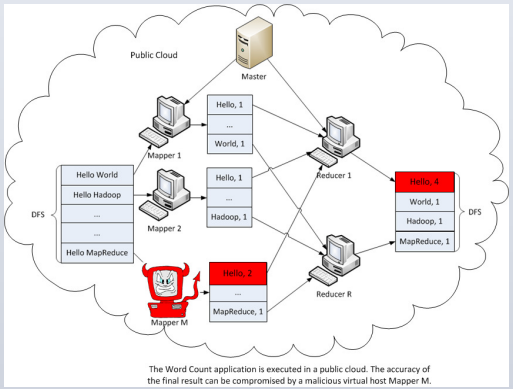
Yongzhi Wang, Jinpeng Wei
Florida International University

Mudhakar Srivatsa
IBM T.J. Watson Research Center

INTRODUCTION

MapReduce is becoming a popular data processing application on Cloud Environment. However:

- Security issues make many customers reluctant to move their critical computation tasks to cloud.[1]
- In MapReduce where jobs are carried out via the collaboration of a number of computing nodes, merely one malicious node may render the overall results useless.
- In a traditional MapReduce setting where each node is deployed on the cloud, the integrity of a computation can be easily compromised and difficult to detect.



We propose a new MapReduce Framework: CCMR (Cross Cloud MapReduce) :

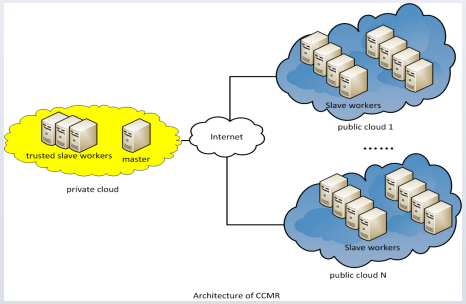
- It can be deployed among a single private cloud and multiple public clouds.
- It employed replication, hold-and-test, verification and credit-based trust management approaches
- It can eliminate malicious compute nodes and guarantee high computation accuracy while incurring acceptable overhead.

ATTACKER MODEL

- The attacker is a "powerful adversary" that controls malicious nodes in each public cloud environment.
- The adversary receives and shares information collected by the malicious nodes and instructs a select subset of malicious nodes to whether or not cheat the master in order to introduce as many errors as possible to the final result without detection.

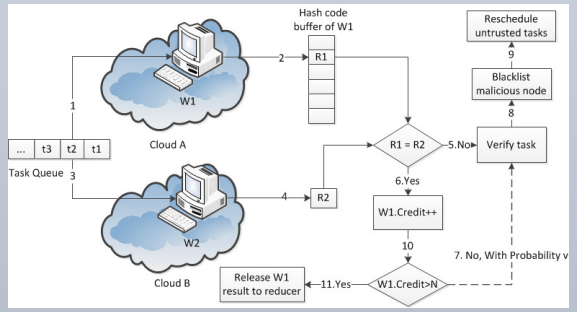
SYSTEM ARCHITECTURE

- The master and a small number of trusted slave workers [2] are deployed on the trusted private cloud within the customer's organization;
- Other slave workers are deployed on public clouds which are not trusted.



SYSTEM DESIGN

- Replicate each task to two workers from different clouds. (replication)
- Assign the replicated task to the second worker only after the first worker return the original task result. (hold-and-test)
- Verify the consistent result in a probabilistic manner. (verification)
- Buffer the task result and increment the credit of the worker who executed original task and passed the hold-and-test and verification. (credit-based trust)
- Accept the buffered result in a batch from the worker who achieve certain credit threshold.



Notes: 1. For instance, some members of EC2 can create and share malicious Amazon Machine Image (AMI) with the EC2 community; a malicious AMI, if widely used, could flood the community with hundreds of infected virtual instances.

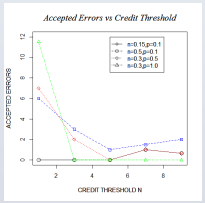
2. The trusted slave workers are used to verify the results returned by the untrusted workers: it arbitrates the inconsistent results if available and verifies the consistent results in a non-deterministic manner.

EXPERIMENT RESULT

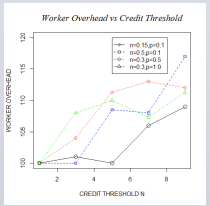
Experiment Environment:

- Private cloud with a Linux server (2.93 GHz, 8-core Intel Xeon CPU and 16 GB of RAM) as the master and the trusted worker.
- 6 Microsoft Azure extra small instances as 6 workers.
- 6 Amazon EC2 small instances as 6 workers.

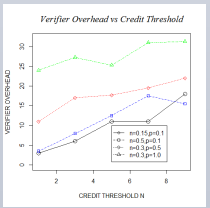
Accuracy, Overhead and Verifier Overhead experiment application: Hadoop Word Count with 100 map tasks and 1 reduce task.
Performance experiment application: Mahout 20 Newsgroup classification example.



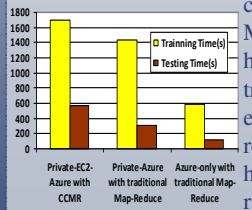
Accepted Errors in running a 100-map-task job under different environment configurations: Accepted Errors decreases with the increase of value N. Under the same condition, lower value of n or higher value of p means fewer Accepted Errors.



Worker Overhead in running a 100-map-task job under different environment configurations: Worker Overhead increases with the increase of value N.



Verifier Overhead in running a 100-map-task job under different environment configurations: Verifier Overhead increases with the increase of value N. Under the same condition, lower value of n means smaller Verifier Overhead.



Running time of Mahout 20 Newsgroup classification example. Homogenous MapReduce takes the shortest time; heterogeneous environment with traditional MapReduce increases the execution time by 145% and 177%, respectively; introducing CCMR to the heterogeneous cloud increases the running time by 18% and 82%, respectively.

www.PosterPresentations.com