A Novel MapReduce Framework for Improving Security of Cloud Computing

Junyi Deng, Yanheng Liu, and Jian Wang
College of Computer Science and Technology
Jilin University
Changchun, China
dengjunyi@vip.sina.com, {yhliu, wangjian591}@jlu.edu.cn

Lin Li
School of computer and information technology
Shanxi University
Taiyuan, China
lilynn1116@sxu.edu.cn

***Abstract*—Cloud computing is becoming a powerful parallel data processing method and it can be adopted by many network service providers to build a service framework. Though the cloud computing is able to efficiently process a huge amount of data, it is easy to be attacked due to the massive distributed cluster nodes. In this paper, we propose a novel Secured MapReduce Framework (SMRF), which establishes a close relation between the Speculative Execution (SE) and the security of the YARN. SMRF launches the speculative executions in a certain ratio, computes and compares their respective MD5 hashes of the intermediate and final results in the MapReduce process. Moreover, the proposed framework is able to discover the actual and potential malicious nodes in the Hadoop cluster. In addition, a prototype framework, called SecMR, is implemented based on Hadoop 2.3.0. The theoretical derivations and experiments show that the proposed SecMR not only guarantees the security of the MapReduce process, but also successfully locates two types of the malicious nodes in Hadoop while just increasing a little overhead.**

Keywords—Cloud computing, Hadoop, MapReduce, Speculative execution, Security

I. INTRODUCTION

With the rapid development of the hardware, software and high-speed network, many cloud service providers, e.g., Google and Amazon, are establishing more and more cloud computing (CC) realities [1] around the world [2] as shown in Fig. 1. However, many organizations and customers are still reluctant to accept CC due to the security issues [3]. Therefore, solving these problems is of great significance for the long-term development of CC [4].

Some safety precautions are already getting attention [5]. For instance, Gartner et. al identify seven security issues of CC that need to be solved [6]. Grobauer et. al discuss the security vulnerabilities of the cloud platform [7]. Jansen et. al propose the guidelines on privacy in public CC [8].

Hadoop is considered as the most widely used CC platform [9] and the MapReduce can be regarded as the most efficient framework for processing vast amounts of the distributed data. However, most of the current researches are still paying increased attention to the MapReduce performance rather than its security [10].

In this paper, we focus on improving the security of MapReduce 2.0 (MRv2/YARN: Yet another Resource Negotiator) [11], and a secured MapReduce 2.0 framework (SMRF) is proposed. Moreover, we provide a prototype called SecMR by extending Hadoop 2.3.0. Moreover, the theoretical derivation and the extensive experiments are performed to prove its validity, security and malicious node detection efficiency.

The rest of the paper is organized as follows. Section II introduces the SMRF design and the SecMR implementation in details. Section III provides the theoretical derivation of SecMR. Section IV shows the experiment results and comparisons on SecMR. Finally, the conclusions are drawn in Section V.

II. SMRF AND SECMR

A. SMRF

In MRv1, the programming model decomposes the problem processing procedure into the Map phase and the Reduce phase. Moreover, the runtime environment consists of JobTracker and TaskTracker. Although the programming model of MRv2 (YARN) is the same, NodeManager (NM) takes the place of TaskTracker, and JobTracker is separated into ResourceManager (RM) and MRAppMaster in YARN. Every MRAppMaster only manages one job and creates the new JobImpls and TaskImpls. TaskImpl contains Map Tasks and Reduce Tasks. Because SMRF will verify the validity of the intermediate and final results generated by Map and Reduce, the programming model and runtime environment are needed to be changed accordingly.

In the programming model of SMRF, there will be more TaskAttempts because it launches the speculative execution in a certain ratio. These additional TaskAttempts execute the same tasks and compute the MD5 hashes of results. The programming model of SMRF is shown in Fig. 1.

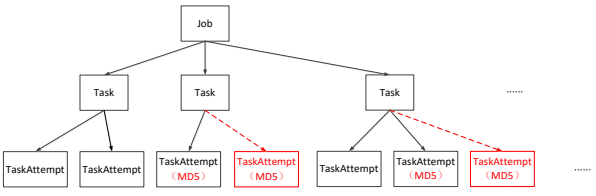

Fig. 1. The programming model of SMRF

B. SecMR

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template

measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

In YARN, RMApp is a data structure that preserves an application life cycle in RM. Its realization instance is RMAppImpl. This class maintains an application state machine that records several application states and state-driven events. The Finite State Machine (FSM) of RMApp is shown in Fig. 2.

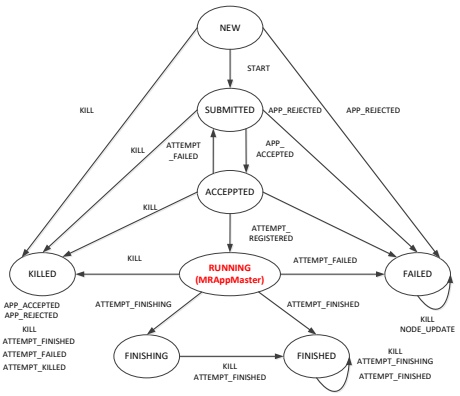

Fig. 2. The Finite State Machine of RMAppImpl

When MRAppMaster is launched, the application will enter into the core state "RUNNING" marked in red in Fig. 2. Every application may run several times. The transitions of states are determined by the return values of MRAppMaster. RMApp judges an application failed when all of RMAppAttempts failed. Therefore, MRAppMaster is the most important module in SMRF.

III. Theoretical Derivation

The Map speculative task and the Reduce speculative task are slightly different. However, the principles are basically the same. Thus, we take the Map task replication as the example to illustrate the theoretical arithmetic.

For easily comparing differences and similarities without losing generality, we set every MRv2 job disposing the same size of data. It means that the total blocks are fixed in every experiment, but the data of every block is absolutely different. Every Map task only processing one block implies that the number of the copied blocks is equal to the number of the replicated Map tasks. We assume the number of blocks (Map tasks) is *b*. And that a container is the abstraction conception of a resource set in YARN. It will be allocated by RM and supervised by NM. Every task is must executed in a container so that the number of containers is also *b*.

In addition, even for the security of MRv2, it is not practical to replicate all of MRv2 tasks by speculative execution because this must consume a lot of resources and time. So we introduce Execution Ratio *Er* to indicate *b*Er* blocks will be duplicated. It is equal to the number of the Map speculative tasks. If a MRv2 job involves one MRAppMaster and *n* containers, *m* containers might be malicious and $m<n$. A malicious node may execute the vicious actions in *P* probability. And the variable *t* represents the number of jobs executed by MRv2.

The aims of SMRF are to ensure the integrity of MRv2 results and find out the malicious nodes. Theoretical arithmetic will show the relationship between Detection Ratio and the above parameters as follows.

The probability of a malicious Map task in *b* Map tasks is 1/b, then 1-1/b is the probability that any Map task is not malicious. If there are *b*Er* duplicated blocks in a MRv2 job, $(1-1/b)^{b*E_r}$ is the probability that all of *b*Er* Map tasks are secure. And $\left[1-(1-1/b)^{b*E_r}\right]$ is the probability that at least one of *b*Er* Map tasks is not secure (i.e. a Map task is not executed in a secure container). Then the probability of the malicious containers(nodes) executing the vicious actions is $\left[1-(1-1/b)^{b*E_r}\right]*P$. And $1-\left[1-(1-1/b)^{b*E_r}\right]*P$ is the probability that the malicious nodes do not carry out the baleful behaviors. If the malicious nodes perform the tasks correctly in *t* MRv2 jobs, this probability is $\left\{1-\left[1-(1-1/b)^{b*E_r}\right]*P\right\}^t$. So $1-\left\{1-\left[1-(1-1/b)^{b*E_r}\right]*P\right\}^t$ is the probability that the malicious nodes expose themselves in *t* MRv2 jobs. In the meanwhile, it also is the probability of discovering the malicious nodes by SecMR.

Therefore, Detection Ratio can be defined as Equation (1).

$$D_{ratio}=1-\left\{1-\left[1-(1-1/b)^{b*E_r}\right]*P\right\}^t \quad (1)$$

We investigate the effects of three factors *b*, *P* and *t* on the detection ratio D_{ratio}.

Fig. 3 shows the change of D_{ratio} against the Execution ratio E_r and the number of the blocks *b*, where *t*=40 and *P*=0.2. One could observe that D_{ratio} increases along with the augment of E_r, but little decreases with increase of *b*.

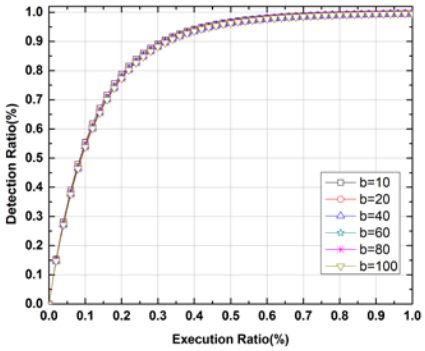

Fig. 3. The effect of the parameter *b* on the Detection Ratio

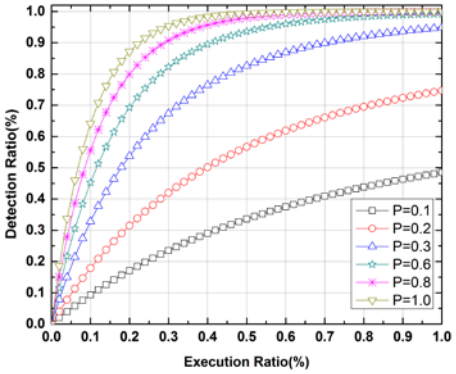

Fig. 4. The effect of the parameter *P* on the Detection Ratio

Fig. 4 shows the change of D_{ratio} against the Execution ratio E_r and the malicious behavior probability *P*, where *b*=20 and *t*=10. One could observe that D_{ratio} increases along with the augment of *P*. Given a certain E_r, the more malicious behaviors there are, the more effectively SecMR works.

Fig. 5 shows the change of D_{ratio} against the execution ratio E_r and the number of jobs *t*, where *b* = 20 and *P* = 0.2. One could observe that D_{ratio} increases along with the increase of E_r and *t*. If *t*=25 and E_r=30%, D_{ratio} is close to 90%.

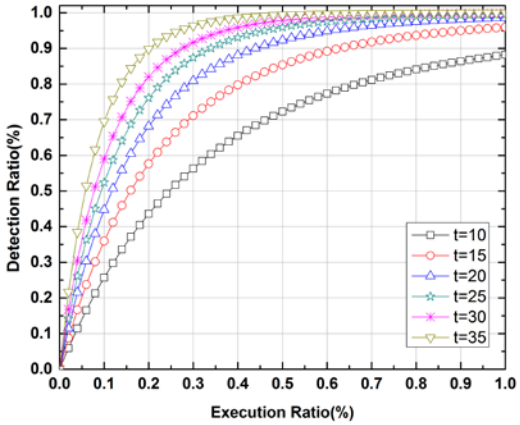

Fig. 5. The effect of the parameter *t* on the Detection Ratio

In a conclusion, the detection ratio D_{ratio} increases with the augment of the Execution ratio E_r, the number of jobs *t* and the malicious action Probability *P*. And the influence of the number of blocks *b* on D_{ratio} is weak. Theoretical derivation indicates that SecMR is quite effective to discover the malicious behaviors. As long as the parameters *b* and *t* are set appropriately, such as *b*=20 and *t* >= 25, we can set E_r at a low level (<=30%) to achieve a desired D_{ratio} (>=85%) when *P* >= 0.2. And the more *P* is, the better D_{ratio} is. Moreover, if combining Map speculative task and Reduce speculative task together, it is reasonable to believe D_{ratio} will be more than 90%.

IV. SecMR Experiments

We evaluate the validity and performance of SMRF by conducting three experiments on SecMR. Restricted by the experimental conditions, there are 1 RM node and 3 NM nodes in Hadoop cluster. RM machine is equipped with one 4-core 3.1GHz Intel Xeon(R) CPU E31225, 4GB memory, one 1TB SCSI disk and 1000M NIC. Three NM machines are respectively equipped with one 2-core 2.93GHz Intel CoreTM2 Duo CPU E7500, 4GB memory, one 160GB SCSI disk and 1000M NIC. All machines have the same software configurations, i.e., Ubuntu 14.04.1 LTS server (64 bit), Java SE Runtime 1.8.0 and Hadoop 2.3.0.

A. WordCount Benchmark

WordCount is a classic MapReduce program. It can count the occurrence numbers of each word in a specified data set. We choose various test files and compare the time cost in three different scenarios. The size of small file is larger than 64M to avoid affecting the scalability and performance of Hadoop. The results are the average values of 25 WordCount experiments according Section III. The corresponding histogram is shown in Fig. 6.

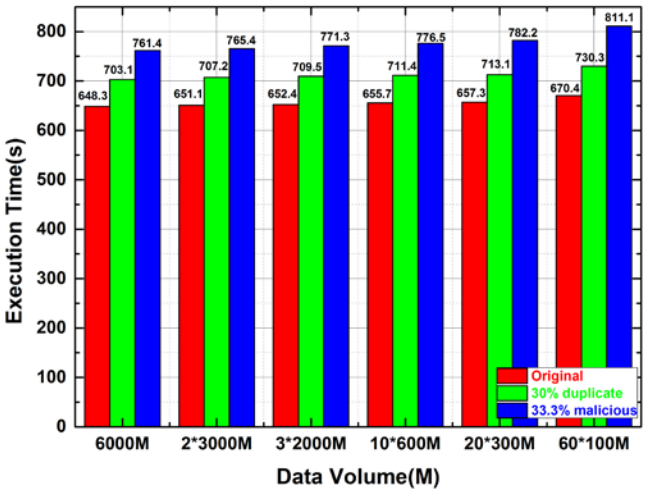

Fig. 6. The Execution Time of WordCount

In Fig. 6, one can get three conclusions. (1) In the original condition, the time cost is proportional to the number of blocks and the total input paths. In MapReduce Framework, “6000M”, “2*3000M” and “3*2000M” are divided into 96 blocks. “10*600M” and “20*300M” are equal to 100 blocks. “60*100M” is split into 120 blocks. And that the different numbers of the input paths make the execution time of “30*200M” different from that of “60*100M”. Because “60*100M” has 60 input paths that is twice as much as that of “30*200M”. (2) Without the malicious nodes, the time cost of WordCount just increases about 9% in the SecMR. A new speculative TaskAttempt is not equal to a new same task so the time of Job do not increase by 30%. The extra time cost mainly comes from the inconformity of TaskAttempt times and the low performance of small cluster. By contrast, MD5 hash computing and comparing bring little influence. (3) Owing to load balancing and a malicious node in the cluster, the tasks assigned to it are close to 33.3%. And the probability of malicious behavior is 20% so that the increasing time is mainly due to Task waiting for the returned values of extra speculative TaskAttempts. Time cost increases approximately between 16% and 21% compared to that of the original condition.

B. Mrbench Benchmark

Mrbench repeats a little job many times specified by the user, which is designed for checking whether the little job running on a Hadoop cluster is repeatable and efficient. It is used to test the performance to handle lots of little jobs and the security protection ability of SecMR. Therefore, the

times of Job repetition are set as 10, 15, 20, 25, 30 and 40. The experiment results in three scenarios are shown in Fig. 7.

One can get three conclusions. (1) In three conditions, every experiment is executed in the different repetition times but in the same configurations. The execution time is an average value of SecMR handling the same job in several times. The more times is repeated, the more accurate the execution time is. (2) Adding 30% speculative executions makes the execution time increase approximately 9%, mainly because the inconsistent Task Attempt completion, the MD5 hashes computation and comparison. (3) Execution time increases approximately 16% by reason of adding 33.3% malicious nodes will result in the inconformity of two comparative MD5 hashes, and the extra speculative TaskAttempts.

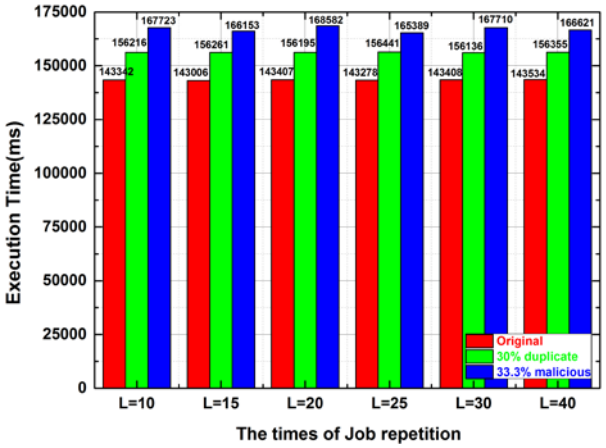

Fig. 7. The results of Mrbench Test

V. Conclusions

In this paper, a secured and high-integrated MapReduce 2.0 Framework (SMRF) is proposed. The relevant parameters of SMRF are accurately set according to the theoretical derivation. The prototype framework SecMR is implemented based on Hadoop 2.3.0, which takes advantage of Speculative Execution and MD5 hashes verification to ensure the integrity and validity of MapReduce results. Moreover, SMRF is able to locate two kinds of malicious nodes and the potential malicious nodes. Three experiments on SecMR adequately demonstrate its malicious nodes detection D_{ratio} and computing resources consumption can achieve the expected goals. Especially, D_{ratio} is more than 86% at least while just increasing a little overhead. Therefore, the proposed SMRF will use lower speculative execution ratio and less resource consumption to achieve a more desirable D_{ratio}, as long as it is built on the more excellent machines in the large Hadoop cluster and disposes the more Jobs.

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China (Grant No. 61373123), and in part by the Jilin Province Science and Technology Development Plan (Key Science and Technology Tackling) Project (Grant No. 20160204041GX).

References

[1] Armbrust M, Fox A, Griffith R, et al. A view of cloud computing[J]. Communications of the ACM, 2010, 53(4): 50-58.

[2] Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, *25*(6), 599-616.

[3] Takabi, H., Joshi, J.B.D., and Ahn, G.J.: 'Security and Privacy Challenges in Cloud Computing Environments', IEEE Secur. Priv., 2010, 8, (6), pp. 24-31

[4] Subashini, S., and Kavitha, V.: 'A survey on security issues in service delivery models of cloud computing', Journal of Network and Computer Applications, 2011, 34, (1), pp. 1-11.

[5] Jensen, M., Schwenk, J., Gruschka, N., Lo Iacono, L., and Ieee: 'On Technical Security Issues in Cloud Computing': 'Cloud: 2009 Ieee International Conference on Cloud Computing' (Ieee, 2009), pp. 109-116

[6] Brodkin, J.: 'Gartner: Seven cloud-computing security risks', Infoworld, 2008, pp. 1-3

[7] Grobauer, B., Walloschek, T., and Stocker, E.: 'Understanding cloud computing vulnerabilities', Security & privacy, IEEE, 2011, 9, (2), pp. 50-57.

[8] Jansen, W., and Grance, T.: 'Guidelines on security and privacy in public cloud computing', NIST special publication, 2011, 800, pp. 144.

[9] Srirama, S.N., Jakovits, P., and Vainikko, E.: 'Adapting scientific computing problems to clouds using MapReduce', Future Generation Computer Systems, 2012, 28, (1), pp. 184-192.

[10] Mackey, G., Sehrish, S., Bent, J., Lopez, J., Habib, S., and Wang, J.: 'Introducing map-reduce to high end computing', in Editor (Ed.)^(Eds.): 'Book Introducing map-reduce to high end computing' (IEEE, 2008, edn.), pp. 1-6.

[11] Vavilapalli, V.K., Murthy, A.C., Douglas, C., Agarwal, S., Konar, M., Evans, R., Graves, T., Lowe, J., Shah, H., Seth, S., Saha, B., Curino, C., O'Malley, O., Radia, S., Reed, B., and Baldeschwieler, E.: 'Apache Hadoop YARN: yet another resource negotiator'. Proc. Proceedings of the 4th annual Symposium on Cloud Computing, Santa Clara, California2013 pp. Pages