情報保護學會論文誌
第18卷 第5號, 2008. 10

e-Science 그리드를 위한 가볍고, 적응성있고, 신뢰성있는 처리 무결성 감사

정 임 영[1†‡], 정 은 진[2]
[1]서울대학교, [2]아이오와대학교

A Light-weight, Adaptive, Reliable Processing Integrity Audit for e-Science Grid

Im Young Jung[1†‡], Eunjin Jung[2]
[1]Seoul National University, [2]U. of Iowa

ABSTRACT

E-Science Grid is designed to cope with computation-intensive tasks and to manage a huge volume of science data efficiently. However, certain tasks may involve more than one grid can offer in computation capability or incur a long wait time on other tasks. Resource sharing among Grids can solve this problem with proper processing-integrity check via audit. Due to their computing-intensive nature, the processing time of e-Science tasks tends to be long. This potential long wait before an audit failure encourages earlier audit mechanism during execution in order both to prevent resource waste and to detect any problem fast. In this paper, we propose a Light-weight, Adaptive and Reliable Audit, *LARA*, of processing Integrity for e-Science applications. With the *LARA* scheme, researchers can verify their processing earlier and fast.

Keywords : HVEM Grid, *LARA*, Processing Integrity Audit

I. Introduction

HVEM Grid, one of the e-Science Grids, is designed for computation-intensive tasks of the reconstruction process of a set of 2 dimensional(2D) images to its 3 dimensional(3D) image. For smaller latency and better efficiency, the reconstruction process is executed parallel through many processors on the computation Grid in HVEM Grid[1,2,3].

Despite parallelism, certain tasks may involve more than one grid can offer in computation capability or incur a long wait time on other tasks. Resource sharing among Grids can solve this problem, only if the proper processing integrity check is in place. If there is dependency among intermediate results, then any intentional (for free incentives) or unintentional (due to system faults) errors in processing may harm the final result fatally.

E-Science Grid is a more controlled environment with its well organized policy than popular Grids such as a desktop Grid (e.g. SETI@Home), because of its expensive or specialized resources. The rigid management policy in e-Science Grid may give

접수일 : 2008년 8월 22일; 채택일 : 2008년 9월 28일
† 주저자, 교신저자, jyjung@dcslab.snu.ac.kr

researchers the false impression that even processing in other Grids would be secure and perfect. But, other Grids do not guarantee processing integrity, especially because e-Science applications tend to consume a large amount of resource and other Grids may find a way to cheat. To support Grid resource sharing, we need a fast and reliable integrity audit for the parallel processing, which imposes low overhead. In this paper, we propose a Light-weight, Adaptive, Reliable Audit, *LARA*, an audit mechanism to check processing integrity based on the underlying trust among Grids.

The rest of this paper is organized as follows. Section II describes the previous works on processing-integrity audit. Section III describes *LARA* with the image processing in HVEM Grid as an e-Science application. Section IV demonstrates the soundness of the proposed scheme by analyzing its characteristics and shows the performance. Conclusion is presented in Section V.

Ⅱ. Related Works

Integrity audit was studied in the aspect of data integrity and processing integrity.

The data integrity check is done by hash function in [4,5,6]. A new approach inserts verification records into outsourced database and checks data integrity by queries to this database [7].

Processing integrity has been studied in three aspects. First, the processing integrity is assumed to be the same with the program integrity [8,9,10]. Second, it should be ensured with a secure H/W platform. Third, it should be checked by the comparison of the processing results.

The methods for program integrity check have used hash function such as MD5 or CRC in general. But, these approaches focused on the analysis of the characteristic of the binary code produced from the program by compiler and linker, and the mechanical inspection for the binary without considering the program semantic. To audit the machine instructions directly requires as much amount of processing. For the e-Science applications which are computation-intensive and have a long processing time, this method is not a good choice due to its heavy overhead. And, because a user requests his task execution to other Grids and gets the results, there are security holes so that the results can be spoiled.

In the early 2000's, there were the papers to elevate the security level in the execution platform of programs [11,12] and the projects whose targets were to construct a secure execution platform(SEE). The representative examples are the projects of XOM [13,14,15] and AESIG [16], which were developed at the universities in US. As commercial technologies, there are TrustZone from ARM and Trust Execution Technology(TET) from Intel. But, these approaches require the development of new H/Ws and some modifications in the Operating System(OS). It is not realistic for the users to request that the Grids which provide their resources should equip SEE or TPM.

Another studies to audit the processing integrity propose to check the processing results. With several checkpoints, users can check the processing [17]. On the other hand, they can consider the techniques of parameter sweep [18]. In the former, users compare the results of the same task from several processors at each checkpoint. The latter is for the case that users can anticipate the processing results. The users consider the use of the resources shared because there are no available resources in their Grid. So, it is a heavy burden for them to utilize the resources if they should ensure the additional processors to audit the processing integrity for the former method [17]. An optimized solution for processing integrity audit should be provided. If the user knows the processing results or he searches better processing results with parameter sweep, he can verify the processing with the interim results [19]. The user intervention is necessary in the parameter sweep and it is difficult to generalize and automate the verification process.

Ⅲ. A processing integrity audit scheme on HVEM Grid

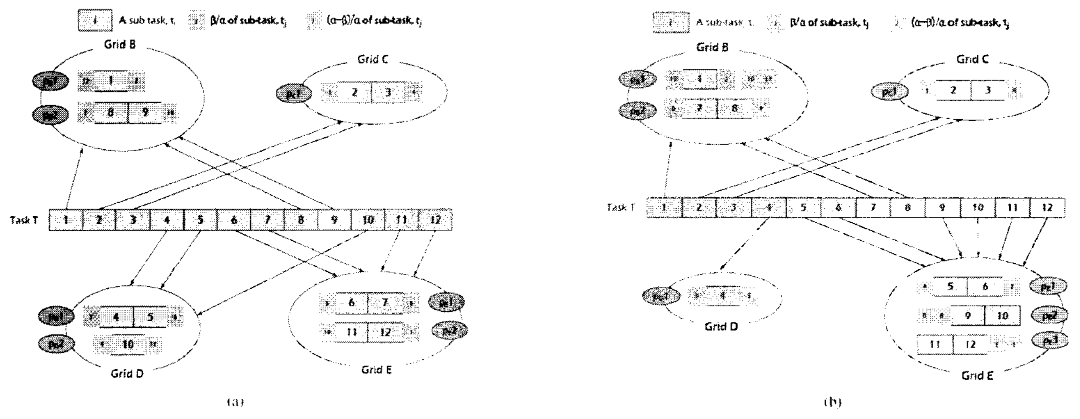
3.1 A image processing on HVEM Grid

In HVEM Grid, the 3D tomographic reconstruction is the image processing through its computation Grid. The target data for the reconstruction is tens of or hundreds of 2D EM images [2] which are taken continuously by HVEM [20] with its tilting degree of 1 or 0.5 to investigate the 3D structure of a new material. The 2D EM images are rendered to its 3D tomographic image which can visualize the volume structure of the material. This process is called a 3D tomographic reconstruction. For the 3D tomographic reconstruction, the 2D EM images are defined as a task T to be processed on the computation Grid. The task can be divided into *n* sub-tasks which are the sets of 2D EM images with continuous tilting angles. The sub-tasks are distributed to the Grids which share their resources and processed under the resource management policy of each Grid. The processing results for the sub-tasks are integrated to a 3D tomographic imag.

3.2 A Light-weight, Adaptive and Reliable integrity Audit(*LARA*) for task execution by [β:a] rate

It is important to check the processing integrity when the task is executed though the Grid resources shared. For a long processing, it would be helpful for users to know the problems during the processing as soon as possible. Thus, we need a fast audit method for processing integrity with low overhead. The processing integrity audit for the sub-tasks is done by the unit of processor.

As shown in [fig. 1(a)] and [fig. 1(b)], the parts of the sub-tasks are executed redundantly by β/a and $(a-\beta)/a$ for processing audit. That is, a sub-task is divided into β:$(a-\beta)$ and the two parts are executed redundantly. In [fig. 1(a)], a task T is divided into 12 sub-tasks whose sizes are the same. When a researcher requests the execution of the sub-tasks to 4 Grids, the two processors in Grid B take 2 and 3 sub-tasks each. When the sub-task t_1 is executed in Grid B, the latter $(a-\beta)/a$ part of t_{12} and the front β/a part of t_2 are allocated to be executed for processing integrity check. The processor p_B2 would process t_8 and t_9 with the latter $(a-\beta)/a$ part of t_7 and the front β/a part of t_{10}. The processing in Grid B can verify the sub-tasks t_{12} and t_7 which are executed in Grid E, t_2 in Grid C, and t_{10} in Grid D. The other Grids also provide the processing for integrity audit as Grid B. In [fig. 1(b)], Grid E takes the continuous images as its sub-tasks. In that case, p_E2 and p_E3 should execute the audit processing for each other within Grid E. But, *LARA* allocates the processing audit for the execution of t_{10} and t_{11} to Grid B; the latter $(a-\beta)/a$


A sub task, t_i
β/a of sub-task, t_j
$(a-\beta)/a$ of sub-task, t_j
Grid B
Grid C
Grid D
Grid E
Task T
(a)
(b)
[Figure 1] A task allocation with a sub-task audit by [β:a] rate

part of t_{10} and the front β/a part of t_{11} get to be one sub-task for processing integrity audit. When Grid B can not handle this extra processing in one round parallel execution, it needs another round of parallel processing for it.

When the researchers can know the available resources they can use and their specifications in advance, they can divide their tasks into its sub-tasks and schedule the processing of the sub-tasks. The scheduling can be for either one round of parallel execution or several rounds of it. In the former, the advantage of parallel processing would be maximized. In the latter, the users should do a new scheduling for the remaining sub-tasks considering the available resources after each round parallel processing. With the latter case, the researchers can utilize the Grid resources shared efficiently and can catch the problems in the processing early.

The processors or the Grids which take the sub-tasks, can not distinguish the sub-tasks allocated with those for integrity audit. In [fig. 1(a)], the amount of the task for audit is a half of the sub-tasks allocated to $p_{B}1$. $p_{B}2$ executes 1/3 of its sub-tasks for audit.

The continuous 2D EM images in HVEM Grid has a feature that the processing result of them should be continuous, too. And, they can be divided and processed parallel. For the audit of the processing integrity, the parts of the 2D images are processed redundantly. Roughly, we can verify the processing with much less than twice the number of the sub-tasks.

Ⅳ. Evaluation

4.1 Experimental setup and Assumptions

We used a computer with Intel Core2Duo 1.83GHz CPU and 3G RAM, running Windows XP for performance evaluation. We set that a task is divided into several sub-tasks with the same size to be processed parallel. And, the followings are assumed. First, each processor in Computation Grid processes three sub-tasks parallel at its maximum; this means that some processors can be more powerful than the others. Second, their processing time for sub-tasks is the same with that for one sub-task in the less powerful processor. Third, the failures or the problems in processing take place by the unit of processor not of sub-task. Fourth, there is no problem in the program which executes the sub-tasks.

4.2 Analysis for the reliability of *LARA*

Our proposal provides the reliability of the integrity audit by *LARA* in the following three aspects.

- Processing failures
  LARA checks the processing integrity by asking the processing for the parts of the sub-tasks to the other Grids. The Grids for integrity audit are chosen near at random, the failures can be detected and understood objectively and exactly.

- Unripe execution
  Unripe execution means that even though a processor executes the sub-tasks normally, it does not complete the execution requested. One example is that it only executes the sub-tasks less than the number of times requested. But, this can be checked by the partial redundant execution for integrity audit. Because it is difficult to distinguish the sub-task with the audit task and the Grids or the processors are not reserved for audit, users can check the unripe execution easily and objectively.

- Collusion
  In *LARA*, the collusion is possible only when the conspirators should know the whole mechanism for the sub-task allocation. With a partial knowledge for the allocation, it is difficult to collude because the processing for integrity audit is distributed near at random. In order to prevent a complete collusion, it is enough that the Grid which the user belong to takes charge of a part

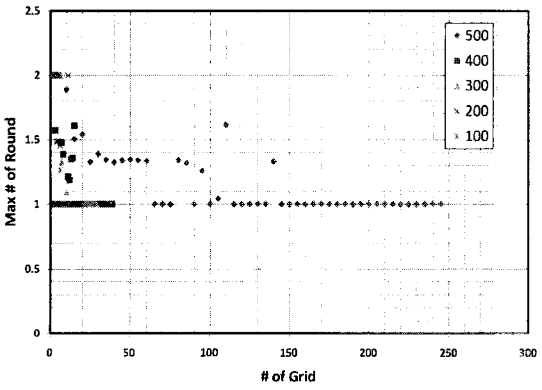
of the sub-task processing.

4.3.Performance Analysis

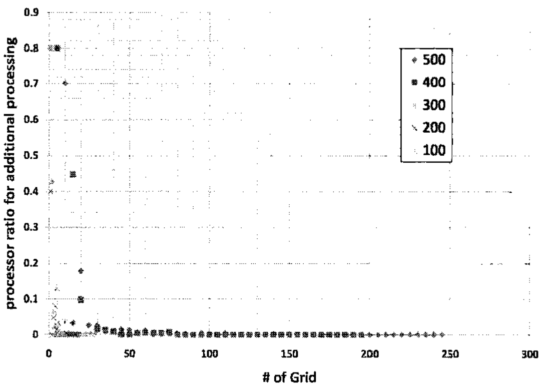
The experiment was designed so that all the sub-tasks complete in one round parallel processing on the Grid resources shared. Several rounds of parallel processing are possible when the sub-tasks are allocated to the available resources shared at the scheduling time and the rest are re-allocated according to the resource availability at the next scheduling time and so on. But, because every round of parallel processing iterates the same mechanism, we evaluated *LARA* only for one round of parallel processing. And, we let a be 2 and β be 1 in the evaluation. The experimental results are averaged with 1000 trials. We considered a random sub-task distribution to the shared resources in Grids.

[Fig. 2] shows the total execution time. With the assumption of a constant processing time in every processor, the execution time is compared with that without the processing for the integrity audit. In the figure, the values in x axis represent the number of Grids which providing their resources. We experiment with the number of sub-tasks from 100 to 500. We let the processing without audit be 1 and present the relative ratio of execution time in [fig. 2]. Without audit, the task processing is enough with one round parallel processing. But, with integrity audit, the extra round of parallel execution from 0 to 1 is needed. But, in the worst case, the total execution round is bounded to twice the parallel processing without integrity audit. But, if users try to allocate the sub-tasks evenly to the Grids which provide their resources to be shared, the execution overhead for integrity audit can be minimized because the even allocation may not need extra rounds of parallel processing for integrity audit. [Fig. 3] shows that the ratio of processors which should execute the extra processing for integrity audit in the parallel processing. Even though there needs the extra processing, the ratio of the processors for integrity audit rarely exceeds 50%. Even though more processors/grids are available, the processors for integrity audit would not be required. And, the extra rounds of parallel processing for the audit would be bounded to 1 in the worst case and to 0.5 in average.

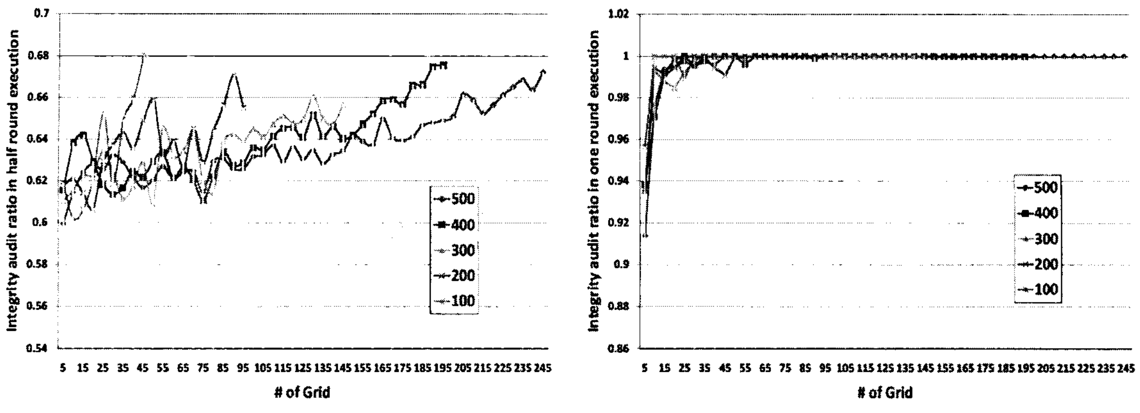
[Fig. 4(a)], and [Fig. 4(b)] show how early *LARA* can detect the problems in processing. These results are meaningful for the e-Science application which can deliver the interim results in its processing. When we let the amount of the sub-tasks whose integrity is audited be 1, after half round of the parallel processing, the amount of them is about 0.6 or 0.7. Without waiting for the completion of the parallel processing, we can verify 3/5 or 2/3 of the processing even with the half round of the execution. Because the sub-tasks are not distributed evenly in the Grids



[Figure 2] Max number of the round for parallel processing



[Figure 3] Processor Ratio for Additional Processing



(a) For half round (b) For one round

[Figure 4] Execution Ratio

in skew allocation, the tasks need more than one round parallel processing. Even this case, more than 0.9 of the tasks can be verified with one round parallel execution. But, there is no distinguishable difference at the ratio of task verification in between skew allocation and even allocation.

4.4 The characteristics of LARA

- Light-weightness
  LARA needs only small amount of the redundant processing for the sub-tasks without the processing replicas which process the whole task. And, the task allocation in *LARA* is not a burden because the users can get a help from the program dividing their tasks into sub-tasks and allocating them to the available resources in Grids after it is customized to each e-Science application. The comparison of results is also easy.

- Adaptivity
  In *LARA*, the task is allowed to be divided into its sub-tasks according to the available resources shared, to be executed completely in one round parallel execution. Or, it can be divided into the sub-tasks which can be executed in several rounds of parallel processing considering the dynamic availability of Grid resources. The adaptive scheduling maximizes the efficiency of Grid resource sharing and causes an earlier detection of the problems in the processing.

- Fastness
  Even with the processing of integrity audit, the total processing time of them will be the same with or below twice that without the audit. If the interim results can be delivered in the processing, users can verify 60%~70% of the tasks with a half round parallel processing.

- Randomness
  In *LARA*, users can adjust the amount and the unit of partial audit of sub-task with α and β. This endows the selection of the audit part of sub-task with randomness which is hard to be guessed.

V. Conclusion and Future work

We proposed the *LARA* mechanism as a processing integrity audit. Even in the worst case, our scheme ensures with less than or equal to extra one round parallel processing for the audit when to use the external resources which belong to the other Grids. We designed the light-weight audit scheme with the given Grid resources and engrafted adaptivity on it to ensure the best appropriate allocation for Grid resource sharing. But, it is difficult to trace which processor processes the integrity audit of a sub-task because each processor is allocated the integrity checking near at random. Moreover, one can not distinguish the sub-tasks with those for integrity audit. These devices provide the base for the reliability of *LARA*.

We consider the extension of *LARA* as a future work. *LARA* is somewhat specific to the parallel image processing on HVEM Grid in this paper. If *LARA* become applicable to general parallel processings, it would be more meaningful.

References

[1] Young-Heon Ahn et al. "Remote access and data acquisition system for high voltage electron microscopy". *Korean J. Electron Microscopy*, 36(1):7-16, 2006.

[2] Im Y. Jung, In S. Cho, Heon Y. Yeom, Hee S. Kweon, and J. Lee. "Hvem datagrid : Implementation of a biologic data management system for experiments with high voltage electron microscope". *GCCB 2006*, January 2007.

[3] Hyeong S. Kim and In Soon Cho and Heon Y. Yeom. "A task pipelining framework for e-science workflow management systems". *IEEE CCGrid*, 2008.

[4] Claudius Stern et al. "Reliable evidence of data integrity from an untrusted storage service". *ICNS*, 2008.

[5] M. Bellare, R. Canetti and H. Krawczyk. "Keying hash functions for message authentication". *CRYPTO'96*, 1996.

[6] G. Tsudik. "Message authentication with one-way hash functions". ACM SIGCOMM, 22(5): 29-38, 1992.

[7] X. Min, H. Wang, J. Yin, and X. Meng. "Integrity auditing of outsourced data". *VLDB2007*, pp. 782-793, September 2007.

[8] Akito MONDEN, Antoine MONSIFROT and Clark THOMBORSON. "Tamper-resistant software system based on a finite state machine". *IEICE transactions on fundamentals of electronics, communications and computer science*, 88(1):112-122, 2005.

[9] H. Chang and M. J. Atallah. "Protecting software code by guards". *ACM CCS-8 Workshop on Security and Privacy in DRM*, 2001.

[10] J. Lee, H. Kim, and H. Yoon. "Tamper resistant software by integrity-based encryption". *PDCAT*, 2004.

[11] S. Loureiro, L. Bussard, and Y. Roudier. "Extending tamper-proof. hardware security to untrusted execution environments". *USENIX CARDIS'02*, 2002.

[12] S. Ravi, A. Raghunathan, and S. Chakradhar. "Tamper resistance mechanisms for secure embedded systems". *17th International Conference on VLSI Design*, 2004.

[13] D. Lie, M. Mitchell, C. Tekkath, and M. Horowitz. "Specifying and verifying hardware for tamper resistant software". *IEEE Symposium on Security and Privacy*, 2003.

[14] D. Lie, C. Tekkath, and M. Horowitz. "Implementing an untrusted operating system on trusted. hardware". *ACM SOSP*, 2003.

[15] D. Lie, C. Tekkath, M. Mitchell, P. Lincoln, D. Boneh, J. Mitchell, and M. Horowitz. "Architectural support. for copy and tamper resistant software". *9th International Conference of Architectural Support for Programming Languages and Operating Systems*, 2000.

[16] G. E. Suh, D. Clarke, B. Gassend, M. van Dijk, and S. Devadas. "Aegis : Architecture for tamper-evident and tamper-resistant processing". *In Proceedings of the 17 Int'l Conference on Supercomputing*, 2003.

[17] P. Domingues, B. Sousa, and L. M. Silva. "Sabotage-tolerance and trust management in desktop grid computing". *Future Generation Computer Systems*, 23:904-912, 2007.

[18] M. Pan and A. Toga. "A grid enabled work-

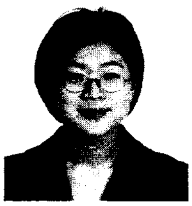
flow management system for managing parameter sweep applications in neuroimaging research". *Sixth IEEE International Symposium on Cluster Computing and the Grid Workshops*, 2006.

[19] Eduardo Huedo et al. "Experiences on adaptive grid scheduling of parameter sweep applications". *12th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, 2004.

[20] Arm1300s in Korea Basic Science Institute (KBSI).

〈著者紹介〉

정 임 영 (Im Young Jung) 정회원
1993년 2월 : 포항공과대학교 화학과 졸업
1999년 2월 : 서울대학교 전산과학과 졸업
2001년 2월 : 서울대학교 컴퓨터공학부 석사
2004년 3월~현재 : 서울대학교 컴퓨터공학부 박사과정
<관심분야> 분산시스템, 분산컴퓨팅, 그리드 컴퓨팅, e-Science 그리드, 시스템보안



정 은 진 (Eunjin Jung) 정회원
1999년 2월 : 서울대학교 전산과학과 졸업
2002년 8월 : Univ. Texas at Austin 석사
2006년 8월 : Univ. Texas at Austin 박사
2006년 8월~현재 : Univ. of Iowa 조교수
<관심분야> 보안, 분산시스템, 알고리즘, 네트워크