

DDC: Distributed Data Collection Framework for Failure Prediction in Tianhe Supercomputers

Wei Hu^{1,2}, Yanhuang Jiang^{1(✉)}, Guangming Liu^{1,2}, Wenrui Dong^{1,2},
and Guilin Cai¹

¹ College of Computer, National University of Defense Technology,
Changsha, China

w-hu@qq.com, yhjiang@nudt.edu.cn,

{liugm, dongwr}@nscc-tj.gov.cn, cc_cai@163.com

² National Supercomputer Centre of Tianjin, Tianjin, China

Abstract. Reliability has become an issue to the Tianhe supercomputer series with the scaling of the system. Proactive fault-tolerance based on failure prediction turns into an effective way to improve the system's fault tolerance ability. Data collection is the basis of the failure prediction which has a great impact on the prediction accuracy, while current data collection methods for failure prediction only got limited data with large overhead. This paper presents DDC data collection framework for failure prediction in Tianhe supercomputers. DDC adopts a distributed data collection architecture which can fully collect the data related to the compute nodes' health with high efficiency. Through the testing for DDC which ran on TH-1A, the results indicated that DDC had the advantage of low cost and good scalability.

Keywords: Supercomputer · Failure prediction · Data collection method

1 Introduction

Reliability wall has been one of the main obstacles to the roadmap of supercomputers toward Exascale [1]. Supercomputers typically have hundreds of thousands of components, for example, Tianhe-2 supercomputer has 16,000 compute nodes, meaning a total of 3.12 million compute cores. With the amount of components increasing quickly, the MTBF (Mean Time Between Failure) of the system decreases from days to hours [2]. Therefore, for long-running parallel applications it becomes difficult or impossible to complete without confronting failures on supercomputers. MPI (Message Passing Interface) which is the main parallel pattern of scientific applications uses message passing mechanism which could cause the entire application failure as long as one process failing, and this is becoming a major performance impediment for supercomputers due to the work loss.

Checkpoint/Restart(CPR), a typical passive fault-tolerance technology, is currently the most common fault-tolerance method which periodically stores all the compute nodes status and recovers from the failure through the rollback approach after the

failure's occurrence [3, 4]. However, owing to the shorter MTBF and mismatched I/O performance compared to the larger scale of supercomputer computing systems, the significant performance loss of CPR is non-trivial, and this may cause new performance problem with larger systems toward exscale computing.

Proactive fault-tolerance technology which can predict the system failure using prediction model and take protection measures with low overheads in advance is now becoming a new research hotspot. Prediction model is the key of the proactive fault-tolerance technology, meanwhile the prediction accuracy determines the availability of the whole proactive fault-tolerance system. The prediction model based on data driven is suitable for large-scale systems and has good accuracy. So the fundamental problem is to obtain the system status data related to failure which are used for the prediction model.

To deal with the problem which MTBF decreasing rapidly in Tianhe supercomputers, we are trying to establish a proactive fault-tolerance system in which data collection for failure prediction is an important part. This paper presents a Distributed Data Collection Framework (DDC) to solve the data collection problems in large-scale systems. The paper is structured as follows. Related work is provided in Sect. 2. Section 3 presents the multiple data sources combined prediction model and the DDC design in Tianhe-1A. Section 4 gives the evaluation of DDC and Sect. 5 draws the conclusions and outlines our future work.

2 Related Work

Currently the data which the failure prediction model uses fall into two types: one is the RAS-based (Reliability, Availability, and Serviceability) log data which the supercomputer monitor system provides; the other is the compute node hardware status data and running status data.

RAS log data collected by the monitor system are the records of the RAS related events that occur across the machine. These data include hard errors, soft errors, software problems and machine checks. The researchers of Rutgers University and IBM company [5–7], Lan research team [8–10], Oliner research team [11–13] and some other researchers [14–17] built failure prediction model based on RAS log data. RAS data which record the system hardware and software events have two flaws: firstly, RAS data are incomplete due to the event logging mechanism which cannot record the full-time status variation of the hardware and software, and this may lead to false negatives. Secondly, owing to the complexity of the system, the definitions of log events cannot be completely accurate which are easy to produce false positives. Based on the above reasons, the accuracy of the failure prediction model based on RAS data is limited due to the characteristics of the RAS data themselves.

The compute node hardware status data include hardware temperature, voltage, fan and power related status data. Scott [18], Nagarajan [19] and Rajachandrasekar [20] did the research on the failure prediction model using the hardware status data through IPMI (Intelligent Platform Management Interface). The compute node running status data typically refer to the CPU, memory, network and I/O-related status data, such as CPU load, memory usage, network statistics, I/O bandwidth and so on. Since most

compute nodes of supercomputer are isomorphic on which running are the similar scientific applications, so the data obtained from the compute node operating system may reflect the health status of the nodes. Sahoo [11] proposed failure prediction model using data sets consisted of log records and the compute node running status data. The research of failure prediction model using the compute node running status data is not so much due to the difficulties on data collection. Existing cluster monitor tools like PARMON [21], Ganglia [22] and Ovis-2 [23] have the function of data collection which cannot meet the actual needs owing to the small number of data attributes, nontrivial collection overheads.

From the present research it is easy to find that the data used for failure prediction for supercomputers have the following characteristics which are the motivations for the DDC development.

- One-sidedness: The existing data collection methods are mostly committed to collect the data of a particular aspect of the system which cannot accurately reflect the overall status of the target system.
- Discreteness: Data used now for failure prediction are discrete and bursty, which cannot fully record the status of the computing system and significantly affect the prediction accuracy.
- High Overhead: The overheads of data collection mainly consist of three parts, CPU overhead, network overhead and storage overhead. CPU overhead can be ignored due to the little CPU overhead itself versus to more powerful CPU performance. But with the increasing scale of the supercomputer, network overhead and storage overhead are the true blocks to the data collection.

3 The DDC Design

3.1 Tianhe Supercomputer Series

DDC was designed for Tianhe supercomputer series which were developed by National University of Defense Technology. This series of supercomputers including Tianhe-1, Tianhe-1A and Tianhe-2, are typical MPP (Massive Parallel Processing) systems which have the same features as follows.

- Using accelerator/co-processor technology. Tianhe-2 is using Intel Xeon Phi processors to speed up computation while Tianhe-1A, upgraded from Tianhe-1, is using NVIDIA GPUs to accelerate computation.
- Proprietary high-speed interconnection network. Tianhe supercomputer series adopt high-radix Network Routing Chips (NRC) and high-speed Network Interface Chips (NIC) to implement proprietary network protocol based on fat-tree topology.
- Parallel file system based on Lustre. Tianhe-1A uses the Lustre file system as the parallel file system while Tianhe-2 adopts H2IO(Hybrid and Hierarchy I/O stack) based on Lustre and I/O nodes.
- Monitor system based on dedicated Ethernet.

DDC applies to all Tianhe supercomputers which have the similar architecture. This paper describes DDC in Tianhe-1A supercomputer which can also run in Tianhe-2 by parameter configurations. Figure 1 shows the Tianhe-1A architecture in detail. There are 140 cabinets in Tianhe-1A, including 112 compute cabinets, 8 service cabinets, 6 communication cabinets, and 14 I/O cabinets. Each compute cabinet contains 4 compute frames and each frame contains 16 compute nodes.

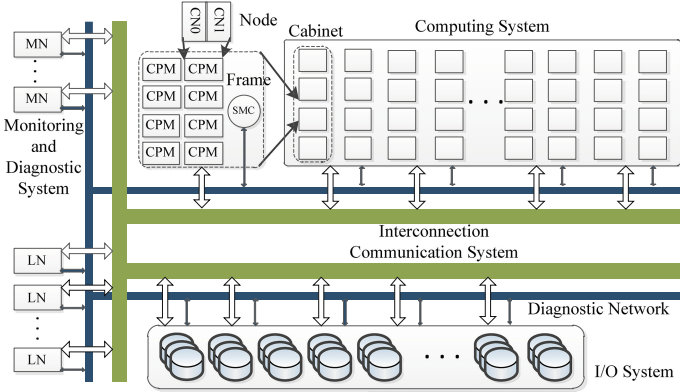


Fig. 1. Tianhe-1A architecture

3.2 Multiple Data Sources Combined Prediction Model

The failure prediction model which is the key of the whole proactive fault-tolerance system has obstacles in acquiring the valid data related to the health status of the compute nodes, and this can be solved by the DDC framework.

Based on the architecture of the Tianhe supercomputer series, this paper presents the multiple data sources combined prediction model as the Fig. 2 shows. This prediction model has two tiers. The first tier includes two different real-time prediction models which provide real-time failure prediction respectively. The second tier provides the intelligent prediction model based on the results from the first tier in order to give the optimal prediction results. All these models are based on the data collected from the system as the figure denotes. This paper focuses on the data collection method for failure prediction, the prediction models will be detailed in another paper.

3.3 An Overview of the DDC Design

Data collection which is the foundation of the entire procedure of failure prediction has two functions: The first is to provide data set for the training of prediction model, where the training data set includes not only the initial training set in the establishment phase of prediction model but also the incremental data set in the upgrade phase of prediction model. The second is to provide real-time data to prediction model for real-time failure prediction.

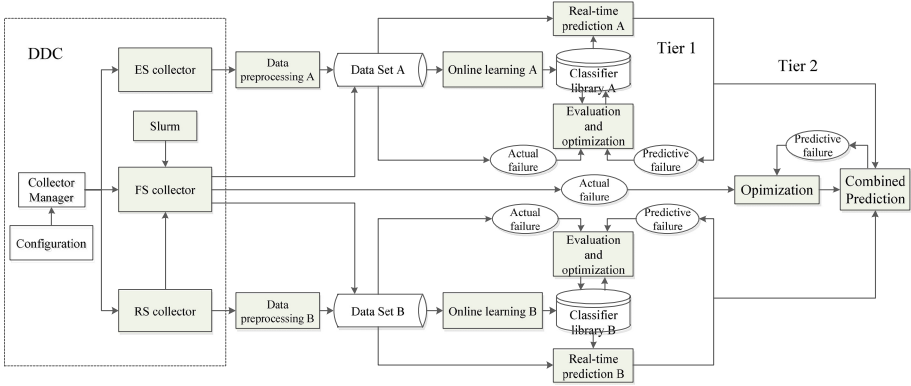


Fig. 2. Multiple data sources combined prediction model. DDC provides the data for the failure prediction model

Figure 3 presents an overview of the design of DDC framework. The DDC framework consists of collector manager, ES collector (collecting hardware status data for compute node), RS collector (collecting running status data for compute node) and FS collector (collecting failure state record for compute node).

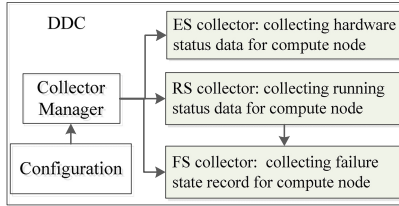


Fig. 3. An overview of DDC framework. Besides configuration files DDC incorporates four units, collector manager, ES collector, RS collector and FS collector

Collector Manager. Collector manager is used to control the operation of the entire data collection system whose main functions are as follows:

- Basic control and configuration. Collector manager is used to start or end the data collection, and configure operating parameters, such as data collection characteristics, collection time interval, data store path and so on.
- Monitoring the operating state of each data collection module. Collector manager periodically checks the logs of each data collection module to handle exception and keep the modules running normally.

ES Collector. ES collector is used to collect and record the hardware status data for compute node. These data mainly denote hardware status related to compute node

which reflect the real-time physical status of the hardware components such as temperature, voltage, current, fan speed and so on.

There is a dedicated Ethernet in the monitor system of Tianhe-1A which used for system control and maintenance, whose central hardware is SMC (System Management Controller). Each compute cabinet of Tianhe-1A is comprised of 4 frames, and each frame which includes 16 compute nodes is equipped with one SMC as shown in Fig. 1. SMC is not only responsible for internal monitoring of a frame but also provide external access interface by using a fixed IP address.

ES collector is designed to collect the compute node hardware status data through SMC in parallel using multiple threads based on dedicated Ethernet. As the Fig. 4 shows, ES collector has the following characteristics: firstly, it is efficient to collect all the 16 compute nodes data through once access to the corresponding SMC using client-server method. Secondly, the programming methods of TCP/IP socket and multi-thread optimize the access efficiency, reduce the access overhead and avoid making the manager node become a bottleneck. Thirdly, this method is fast and almost no overhead to the applications running on the compute nodes due to using dedicated Ethernet.

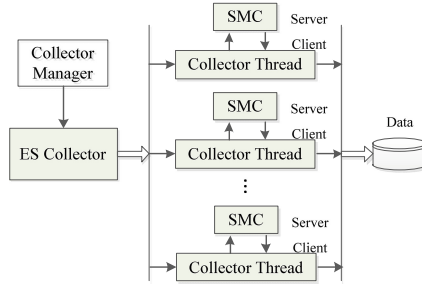


Fig. 4. ES collector collects the hardware status data for compute node using multiple threads based on dedicated Ethernet

RS Collector. RS collector is designed to collect and record the compute node running status data which is related to the system activity commonly referred to as SAR (System Activity Report) data. SAR data are the status data and statistical data of all parts of the system typically including CPU, memory, network, I/O and so on. Commonly compute nodes of supercomputer have the same configuration, therefore the SAR data can effectively reflect the system running status of compute nodes in real time.

With the increasing scale of the supercomputer, it means more overhead to collect much data from more compute nodes through critical path like interconnection and shared storage. Large scales of frequent data transmission and storage operations not only consume the system performance, but also affect the system stability.

To solve this problem, RS collector adopts a distributed data collection architecture which can greatly reduce the amount of data needed to be transmitted and stored. The main considerations are as follows:

- Every compute node has a data collection process which is responsible for data collection, transmission and store.
- All of the compute nodes are divided into groups wherein the nodes probe each other and collect data in the way like a one-way circular linked list. In any group each node not only collects and stores its own data but also sends the data to the next node in the linked list. In other words each node backups the t_b time length data of the previous node and the node itself in the linked list.
- The data set consists of two parts: When a node fails, the next node in the linked list will transmit the backup data to the shared storage as the failure node status data. The normal node status data are stored from the selected normal node according to the configuration which is only a small part of all the normal status data.
- The prediction model based on the compute node running status data is sent to each compute node to perform real-time failure prediction respectively.

Figure 5 shows the distributed architecture of the RS collector with the details related to data collection method, data transmission and storing method. The dashed lines represent the logical relationship of data transmission among the nodes, while the solid line arrows are the actual physical data transmission path.

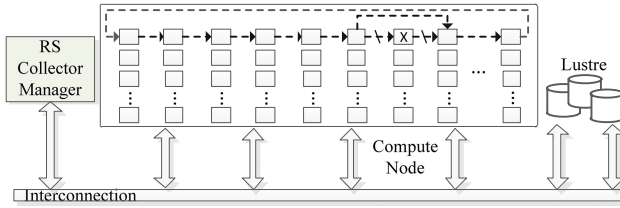


Fig. 5. RS collector collects the compute node running status data using distributed architecture

Data Collection. RS collector gets the running status data effectively through analyzing the `/proc` file system which is a virtual file system and part of the linux kernel. `/proc` provides a dynamic interface which can view the running information of the operating system, such as process information, CPU information and so on. In order to improve the efficiency of data collection and analysis, RS collector reads the `/proc` related files in parallel and then integrates the data to a complete record. The data collection interval of this way can be reduced to milliseconds with little overhead.

By analyzing the files or folders in the `/proc` of Tianhe-1A compute node such as `cpuinfo`, `slabinfo`, `uptime`, `net/`, `sys/`, `scsi/` and so on, the selected 136 data attributes which are closely related to the node running status are collected.

Data Storing and Transmission. That is still a small probability event for the compute node failure, so most of the data collected from the compute node are normal status data which don't need to transmit and store. For failure prediction model, a balanced training set consists of approximately equal numbers of normal and failure status records.

RS collector uses a loop-based data storing and transmission method like one-way circular linked list. This method can greatly reduce the network and I/O overhead since it only transmits and stores the running status data of the failed compute node and little data of normal compute node.

RS collector manager is used to configure and monitor the status of the running status data collection including node grouping, joining, leaving and so on.

The compute nodes are divided into several groups wherein nodes are from different cabinets to avoid the same failure event like power outage and so on. Nodes in the same group form a logical loop structure like one-way circular linked list through sorting the nodes by id numbers. Let us use I_{id} to denote the id number of each node, and $I_{initial}$ is the first node of the system. Suppose α is the grouping coefficient, so the N_{Group} (node grouping number) can be divided using the following formula:

$$N_{Group} = (I_{id} - I_{initial}) \% \alpha \quad (1)$$

There are a total of 7168 compute nodes in Tianhe-1A supercomputer with initial node id 0. Suppose that the grouping coefficient is 64, so the whole system is divided into 64 groups and each group has 112 nodes when all the nodes are on line. In Tianhe-1A each cabinet contains 4 frames with total 64 compute nodes, so the nodes in each group are not in the same cabinet when the grouping coefficient is 64.

As shown in Fig. 6, the node collects the running status data and stores it in duplicate for one copy locally in memory and another copy to the next node in the loop. When a node fails or shuts down, the FS collector records the event and notifies RS collector. RS collector performs deleting operation to notify the related previous and next node to complete the node deleting operation and failure status data saving operation. Figure 7 shows the procedure of a node receiving the data of the previous node. If receiving data from previous node within the timeout limit, the node stores the data in memory and waits for the next data transmission. If the data reception exceeds the timeout limit, the node will probe the previous node by sending a message. The abnormal signal will be sent to the FS collector or not according to the ACK reply. The state of the abnormal node will be judged by the FS collector. If the node failure is confirmed, the delete operation will be triggered. When a new node is added, RS collector will perform inserting operation to update the relevant group list and insert the node to the data collection loop.

FS Collector. FS collector is used to collect and record the failure state data of compute node through integrating three kinds of data including RS collector alarm data, SLURM (Simple Linux Utility for Resource Management) data and maintenance staffs records.

RS collector alarm data denote to the abnormal signals which have been mentioned in the last section. When one node probe the previous node without the ACK reply, the node will issue the abnormal signal related to the previous node to the RS collector manager. And this record will also be sent to FS collector.

SLURM is an open-source resource manager designed for Linux clusters which provides a framework for starting, executing and monitoring work on a set of allocated nodes. SLURM also records the node state variations including ALLOCATED,

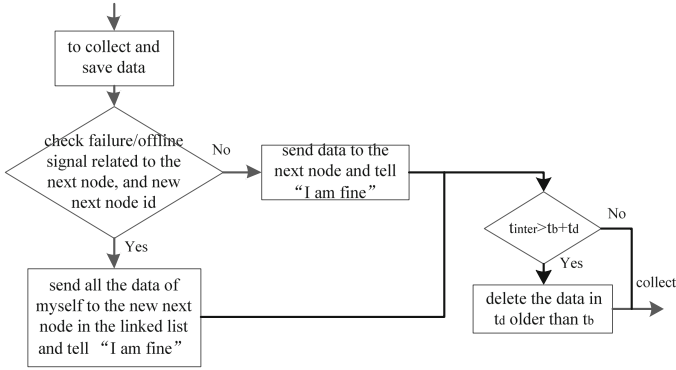


Fig. 6. The flow chart of collecting and transmitting data between the node and the next node in the loop

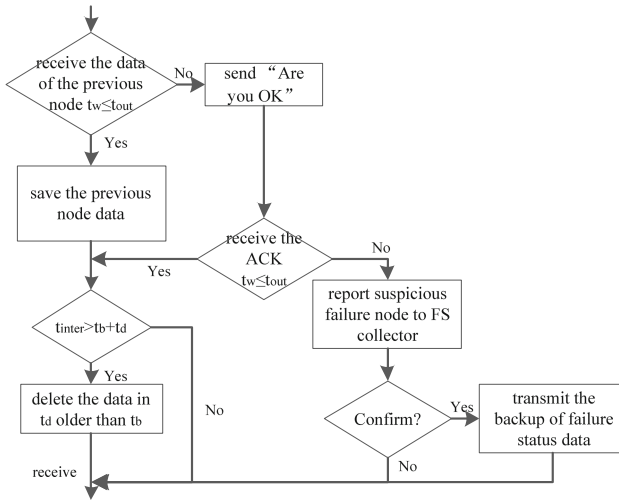


Fig. 7. The flow chart of receiving data and failure node judgment between the node and the previous node in the loop

DOWN, IDLE and so on. There is a problem that the node state which SLURM recorded is not timely or completely accurate.

The staffs who maintain the machine record the compute node state accurately but not timely. Based on the above, FS collector integrates the three methods to determine the node state as the Table 1 shows.

Table 1. Node state determination method of FS collector

RS collector	SLURM	Maintenance staffs	FS collector
Alarm	Normal	No	Normal, if SLURM state of the node turned to abnormal then change the node state to failure
Alarm	Abnormal	No	Failure
Normal	Abnormal	No	Normal, if the RS collector state of the node turned to abnormal then change the node state to failure
Any	Any	Abnormal	Failure
Remark	Abnormal states in SLURM includes DOWN(*) and ERROR(*), while others are normal states.		

4 Evaluation

This section focuses on the evaluations of DDC data collection framework. The Tianhe-1A supercomputer which the DDC runs on is detailed in Table 2.

Table 2. Specifications of Tianhe-1A

Item	Configuration
CPU	Xeon X5670 6C 2.93 GHz
Compute nodes amount	7168
Memory	229,376 GB
Interconnect	Proprietary(optic-electronic hybrid fat-tree structure, point to point bandwidth 160 Gbps)
Storage	Lustre(Lustre*4, total capacity 4 PB)
Operating system	Kylin Linux
SMC amount	448
Data collection interval	10 s

4.1 Hardware Status Data

Tables 3 and 4 show the data of ES collector collected from the sensors on the components of Tianhe-1A supercomputer including fans, communication board of frame (NRM), communication board of compute node (PDP), power supply, compute node mainboard and so on.

Table 3. Data collection from a single compute node

Attributes name	Type	Value	Unit
12 V	Voltage	11.98	V
Vbat	Voltage	3.28	V
ICH-1.5 V	Voltage	1.52	V
IOH-1.1 V	Voltage	1.13	V
5 V	Voltage	5.10	V
5Vsb	Voltage	5.08	V
3.3 V	Voltage	3.33	V
3.3Vsb	Voltage	3.30	V
CPU0 core	Voltage	0.90	V
CPU0 DR3-1.5 V	Voltage	1.54	V
CPU0 Temp	Temperature	29.00	C
CPU1 core	Voltage	0.95	V
CPU1 DDR3-1.5 V	Voltage	1.55	V
CPU1 Temp	Temperature	27.00	C
Thrm	Temperature	28.00	C
PDP-3.3 V	Voltage	3.3	V
PDP-2.5 V	Voltage	2.5	V
PDP-1.8 V	Voltage	1.8	V
PDP-1.5 V	Voltage	1.5	V
PDP-1.2 V	Voltage	1.2	V
PDP-Temperature	Temperature	40	C

4.2 Compute Node Running Status Data

The selected 136 data attributes collected by RS collector are divided into the following four parts: CPU related, Memory related, Network related and I/O related showed in Table 5.

4.3 ES Collector Overhead

ES collector collects and stores the hardware status data using dedicated Ethernet rather than Tianhe-1A high-speed interconnection network, so there is no overhead to the applications running on the compute nodes. Therefore, we tested the ES collector's performance overhead to the manager node by running the ps and vmstat command repeatedly and averaging the results. Table 6 shows the overhead of the manager node when ES collector collected data from 448 SMC servers (all of the Tianhe-1A SMCs). Figure 8 shows the overhead variation of ES collector with the different system scale for data collection. It is easy to find that the ES collector has little impact to the manager node and good scalability.

Table 4. Data collection from other components of the same frame

Component	Attributes name	Type	Value	Unit
Fan	Fan00 ~ Fan25	Fan	5763	RPM
Remark	In one frame 3 fan groups and 6 fans in each group			
NRM	12 V	Voltage	11.94	V
	3.3 V	Voltage	3.33	V
	STB3.3 V	Voltage	3.33	V
	Temp	Temperature	20.00	C
Remark	One NRM in each frame used for interconnection			
Power Supply	Temp	Temperature	48.00	C
	Fan1	Fan	1950	RPM
	Fan2	Fan	2100	RPM
	Input voltage	Voltage	222.00	V
	Input current	Current	2.59	A
	Output voltage	Voltage	11.97	V
	Output current	Current	40.00	A
Remark	Four power supplies in each frame			
SMC	1.8 V	Voltage	1.85	C
	STB3.3 V	Voltage	3.32	V
	5 V	Voltage	5.01	V
	12 V	Voltage	12.06	V
	Board Temp	Temperature	20.00	C
	Inlet Temp	Temperature	20.38	C
	Outlet1 Temp	Temperature	33.13	C
	Outlet2 Temp	Temperature	21.94	C
Remark	One SMC in each frame			

4.4 RS Collector Overhead

Table 7 presents the data collection overhead of RS collector on each node which is the average results by running ps command several times. When the data collection process collected or transmitted data, the average CPU utility rate was less than 0.6 %. The process did not take up CPU resources in the collection interval, so the average CPU utility rate was 0.06 % for each collection circulation. Data collection process of RS collector used 9.0 MB virtual memory, 1.1 MB physical memory and less than 1 ms to transmit the data to the next node. So the average bandwidth taken up by data transmission in 10 s is approximately 0.14 KB/s.

The data volume of each data collection for compute node running status data is about 700B. According to 10 s collection interval and the operation of deleting more than 3 h data every hour ($t_b = 3$ h, $t_d = 1$ h), the maximum amount of stored data is approximately 2 MB (1 MB for node itself, while 1 MB for backing up the previous node). All these overheads were at a low level.

Table 5. The compute node running status

Item	Attribute category	Amount
CPU related	CPU utilization	12
	Task creation and system switching activity	2
	Interrupt statistics	1
	Queue length and load averages	6
Memory related	Paging statistics	9
	Memory statistics	13
Network related	network statistics (devices, EDEV, SOCK, IP, EIP, ICMP, EICMP, TCP, ETCP, UDP)	78
I/O related	I/O and transfer rate statistics	5
	Status of inode	4
	Lustre statistics	6

Table 6. ES collector overhead of the manager node

CPU(%)	PhyMem(MB)	VirMem(MB)	I/O(MB/s)
0.4	5.8	67.5	3.8

Table 7. RS collector overhead of the compute node

CPU(%)	PhMem (MB)	VirMem (MB)	Data transmission time(s)	Bandwidth (KB/s)	Memory usage for data storage (MB)
Collecting < 0.6 Collection interval = 0	1.1	9.0	<0.001	0.14	2

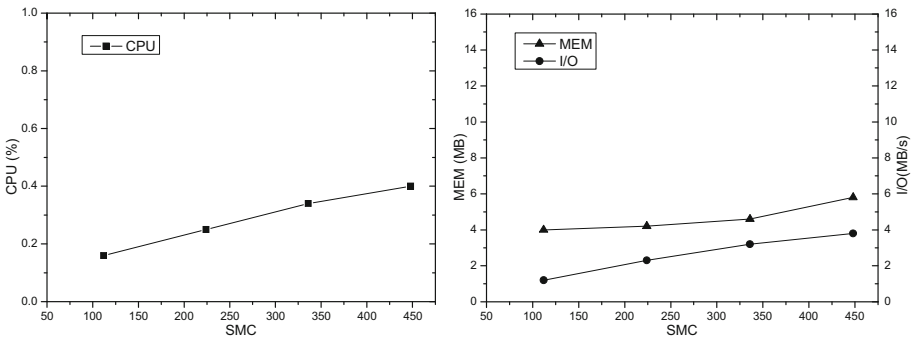


Fig. 8. Scalability test of ES collector

5 Conclusions and Future Work

This paper presents DDC data collection framework for failure prediction in Tianhe supercomputers. DDC adopts a distributed data collection architecture which can fully collect the data related to the compute nodes' health with little overhead. Through the testing for DDC which ran on TH-1A, the results indicated that DDC had the advantage of low cost and good scalability.

In the future, we will focus on the work related to data preprocessing and failure prediction model optimization. We will run the entire proactive fault-tolerance system firstly in Tianhe-1A.

Acknowledgments. This paper is supported by the National Natural Science Foundation of China (NSFC) No. 61272141, No. 61120106005 and the National High Technology Research and Development Program of China (863 Program) No. 2012AA01A301.

References

1. Yang, X., Wang, Z., Xue, J., Zhou, Y.: The reliability wall for exascale supercomputing. *IEEE Trans. Comput.* **61**(6), 767–779 (2012)
2. Philp, I.R.: Software failures and the road to a petaflop machine. In: *Proceedings of the 1st Workshop on High Performance Computing Reliability Issues*, San Francisco, CA, USA (2005)
3. Chen, Y., Plank, J.S., Li, K.: CLIP: a checkpointing tool for message-passing parallel programs. In: *SC 1997*, NY, USA (1997)
4. Hargrove, P.H., Duell, J.C.: Berkeley lab checkpoint/restart (BLCR) for Linux clusters. *J. Phys: Conf. Ser.* **46**(1), 494–499 (2006)
5. Liang, Y., Zhang, Y., Sivasubramaniam, A., Jette, M., Sahoo, R.: BlueGene/L failure analysis and prediction models. In: *DSN 2006*, Washington, DC, USA, pp. 425–434 (2006)
6. Liang, Y., Zhang, Y., Xiong, H., Sahoo, R.: Failure prediction in IBM BlueGene/L event logs. In: *The Seventh IEEE International Conference on Data Mining*, pp. 583–588 (2007)
7. Liang, Y., Zhang, Y., Xiong, H., Sahoo, R.: An adaptive semantic filter for Blue Gene/L failure log analysis. In: *IEEE International Parallel and Distributed Processing Symposium*, pp. 1–8 (2007)
8. Li, Y., Lan, Z.: Exploit failure prediction for adaptive fault-tolerance in cluster computing. In: *CCGRID 2006*, Washington, DC, USA, pp. 531–538 (2006)
9. Lan, Z., Gu, J., Zheng, Z., Thakur, R., Coghlan, S.: A study of dynamic meta-learning for failure prediction in large-scale systems. *J. Parallel Distrib. Comput.* **70**(6), 630–643 (2010)
10. Zheng, Z., Yu, L., Tang, W., Lan, Z., Gupta, R., Desai, N., Coghlan, S., Buettner, D.: Co-analysis of RAS log and job log on Blue Gene/P. In: *IPDPS 2011*, pp. 840–851 (2011)
11. Sahoo, R.K., Oliner, A.J., Rish, I., Gupta, M., Moreira, J.E., Ma, S., Vilalta, R., Sivasubramaniam, A.: Critical event prediction for proactive management in large-scale computer clusters. In: *KDD 2003*, NY, USA, pp. 426–435 (2003)
12. Oliner, A., Rudolph, L., Sahoo, R.: Cooperative checkpointing theory. In: *IPDPS 2006*, Washington, DC, USA, pp. 132–141 (2006)
13. Oliner, A., Ganapathi, A., Xu, W.: Advances and challenges in log analysis. *Commun. ACM* **55**(2), 55–61 (2012)

14. Yamanishi, K., Maruyama, Y.: Dynamic syslog mining for network failure monitoring. In: KDD 2005, New York, NY, USA, pp. 499–508 (2005)
15. Xu, W., Huang, L., Fox, A., Patterson, D., Jordan, M.I.: Detecting large-scale system problems by mining console logs. In: SOSP 2009, NY, USA, pp. 117–132 (2009)
16. Vaarandi, R.: A breadth-first algorithm for mining frequent patterns from event logs. In: Aagesen, F.A., Anutariya, C., Wuwongse, V. (eds.) INTELLCOMM 2004. LNCS, vol. 3283, pp. 293–308. Springer, Heidelberg (2004)
17. Gainaru, A., Cappello, F., Snir, M., Kramer, W.: Fault prediction under the microscope: a closer look into HPC systems. In: SC 2012, Los Alamitos, CA, USA (2012)
18. Scott, S.L., Engelmann, C., Vallee, G.R., Naughton, T., Tikotekar, A., Ostrouchov, G., et al.: A tunable holistic resiliency approach for high-performance computing systems. In: PPOPP 2009, NY, USA, pp. 305–306 (2009)
19. Nagarajan, A.B., Mueller, F., Engelmann, C., Scott, S.L.: Proactive fault tolerance for HPC with Xen virtualization. In: ICS 2007, NY, USA, pp. 23–32 (2007)
20. Rajachandrasekar, R., Besseron, X., Panda, D.K.: Monitoring and predicting hardware failures in HPC clusters with FTB-IPMI. In: IEEE 26th International Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), pp. 1136–1143 (2012)
21. Buyya, R.: PARMON: a portable and scalable monitoring system for clusters. *Softw. Pract. Exper.* **30**(7), 723–739 (2000)
22. Massie, M.L., Chun, B.N., Culler, D.E.: The ganglia distributed monitoring system: design, implementation, and experience. *Parallel Comput.* **30**(7), 817–840 (2004)
23. Brandt, J.M., Debusschere, B.J., Gentile, A.C., Mayo, J.R., Pebay, P.P., Thompson, D., et al.: Ovis-2: a robust distributed architecture for scalable RAS. In: IEEE International Symposium on Parallel and Distributed Processing, pp. 1–8 (2008)

Advanced Parallel Processing Technologies

11th International Symposium, APPT 2015, Jinan, China,

August 20-21, 2015, Proceedings

Chen, Y.; lenne, P.; Ji, Q. (Eds.)

2015, IX, 117 p. 55 illus., Softcover

ISBN: 978-3-319-23215-7