



An Operational Data Collecting and Monitoring Platform for Fugaku: System Overviews and Case Studies in the Prelaunch Service Period

Masaaki Terai^(✉), Keiji Yamamoto, Shin'ichi Miura, and Fumiyoshi Shoji

RIKEN Center for Computational Science, 7-1-26 Minatojima-minami-machi,
Chuo-ku, Kobe, Hyogo 650-0047, Japan

{[teraim](mailto:teraim@riken.jp),[keiji.yamamoto](mailto:keiji.yamamoto@riken.jp),[shinichi.miura](mailto:shinichi.miura@riken.jp),[shoji](mailto:shoji@riken.jp)}@riken.jp

Abstract. After a seven-year long-term development process, the super-computer Fugaku was officially launched as the successor to the K computer in March 2021. During this development process, we upgraded various system components and the data center infrastructure for official service in Fugaku. It was also necessary to upgrade the K computer operational data collection/monitoring platform for use in Fugaku. As a result, we are now in the process of developing and deploying an operational data collection/monitoring platform based on a three-tier pipeline architecture. In the first stage, the HPC system produces various types of log/metric data that are used to identify and monitor troubleshooting issues. Additionally, several thousand sensors operated by the building management system (BMS) generate metrics for power supply and cooling equipment. In the second stage, we aggregate the data into time-series databases and then visualize the results via a dashboard in the third stage. The dashboard provides an interactive interface for multiple data of the HPC system and data center infrastructure. During the course of this project, we resolved some issues found in the previous K computer platform. By using the redundant cores of the A64FX to allocate agents, it was determined that the new platform takes less than 20s to collect metrics from over 150k compute nodes and finally write them to persistent storage. This paper introduces the design of the system architecture and reports on the current state of the platform renewal project, and provides overviews of two use cases encountered during the prelaunch service period.

Keywords: Operational data · Monitoring · Log · Metrics · Dashboard · Open-source software · Prometheus · Elasticsearch · ODA · HPC

1 Introduction

A modern exascale supercomputer system depends on various information technology (IT) component types (e.g., servers, storage, and network systems) and

© Springer Nature Switzerland AG 2021

H. Jagode et al. (Eds.): ISC High Performance 2021 Workshops, LNCS 12761, pp. 365–377, 2021.

https://doi.org/10.1007/978-3-030-90539-2_24

facility equipment types (e.g., power supply and cooling systems), and each system component constantly produces a huge number of time-series log/metric data regardless of whether or not it has failure possibilities. While those log/metric data are most often used by operators/vendors to monitor the system behavior and identify issues during troubleshooting, such data have significant potential as sources for insights into supercomputing/data center operations. Accordingly, as far back as the startup of the K computer operations, we focused on the development of an operational data collection and monitoring platform [27] that would allow us to exploit as well as monitor the massive data produced during our operations. As applications of that platform, we launched various projects and have reported their results [15, 18, 19, 25].

Fugaku [8], the successor of the K computer, is an exascale supercomputer system. The basic system architecture concept, except for the microprocessor, is based on enhancements to the K computer design. The number of compute nodes with Arm-based microprocessors has reached 158,976 nodes, and over 500 PFLOPS have been achieved in the TOP500 benchmark [9]. On the other hand, we adopted a retrofit approach for the facility upgrades. While part of the facility equipment (including the electrical transformers, a heat exchanger, and a chiller with a cooling tower) for the power supply and cooling system was upgraded to handle 30 MW-scale power consumption, most of the K computer facility equipment was integrated into the Fugaku operation without major enhancements. Due to the total growth in power consumption, a rapid number of job submissions can cause megawatts level power consumption fluctuations that could result in temporary cooling capacity shortages. Since maintaining stable operations of both the HPC system and the data center infrastructure is imperative, it was necessary to implement a comprehensive monitoring system of the entire center in order to fully understand the energy usage issue.

The replacement process resulted in the need to develop a new operational data collection and monitoring platform that was created almost from scratch. In particular, our aim was to address the following issues that had been identified in the K computer collection and monitoring platform.

- In the K computer operation, we installed `fluentd` as an agent to collect log and metric data. However, `fluentd`, which is written in Ruby, consumes excessive amounts of memory.
- Due to the need to prevent performance interference, we could not install collector tools (agents) in the compute nodes of the K computer.
- `ZABBIX`, which was used in the K computer operation, is limited to 30k metrics per second due to a bottleneck in the relational database management system (RDBMS). There could have been an issue in terms of sampling frequency in the environment of Fugaku, which has more than 150k compute nodes.
- The building management system (BMS) for the facility is a subsystem of the collection system. (Due to the need to avoid expensive developments, we developed the platform separately.) However, the overhead of the loading metrics from the BMS subsystem will need to be considered if we are to achieve a near-real-time system.

The platform development is an ongoing project. Based on the issues above, we have attempted to improve the previous platform implementations and have resolved several technical issues in the process. In this paper, we report the current state of the project along with two use cases involving the platform.

- To reduce memory usage, we use `logstash` and `filebeat` instead of `fluentd` in the Fugaku environment.
- Since Fugaku’s microprocessor has redundant cores (assistant cores) in a node, we can assign collector agents to the cores without any interference to compute cores of the node and can monitor precise states than the K computer operations.
- We use `Prometheus` instead of `Zabbix`. `Prometheus` can collect up to 1M metrics per second. Finally, the platform can collect data from over 150k compute nodes and write them to persistent storage within 20 s.

The rest of the paper is organized as follows: Sect. 2 presents related work, while Sect. 3 provides the platform renewal project state, including an overview of data sources from the HPC system and data center infrastructure, and the architecture design of the operational data collection and monitoring platform. Section 4 describes use cases involving a dashboard. In Sect. 5, we present a summary and discuss our future work.

2 Related Works

For large-scale IT systems, a reliable monitoring and reporting system is essential for operating and identifying problems during troubleshooting processes. Historically, such monitoring is performed within the context of surveillance. Today, the monitoring processes cover numerous targets, including servers, network devices, storage, applications, and peripheral infrastructure. Meanwhile, as part of efforts to obtain operation/business insights, some data centers have begun considering analytics based on their collection and monitoring platforms [13, 16, 22]. Within the context of high-performance computing, some supercomputing centers now use Operational Data Analytics (ODA) [11, 12, 17] as a formalization process, and most ODA monitoring systems consider not only HPC systems but also peripheral data center infrastructures. Those consolidating platforms often focus on energy efficiency.

In terms of tools, those ODA platforms employ various types of open-source software [1–7]. Along with their pipeline mechanisms, those are roughly classified as collectors, message brokers, time-series databases, dashboards (visualization), and analysis tools. In ODA platforms, the time-series database plays a key role in enabling mass data collection with low overheads. One of the tools, `Prometheus`, is an open-source monitoring/reporting toolkit that is widely used in monitoring processes that can integrate a third-party database depending on the system. Some ODA projects outside the scope of this paper have used or attempted `Apache Cassandra`, `KairosDB`, `influxDB`, and `Victoria Metrics` [21]. Since dashboard systems also provide interactive visualization environments, some

supercomputing centers have attempted to adopt dashboards into their sites and have reported their results [10, 23, 24].

3 Operational Data Collection and Monitoring Platform in the Fugaku Environment

The development of the operational data collection and monitoring platform is a relatively new project among all the projects in Fugaku and is an ongoing project. In this section, we mention the overview of data sources and the architecture design in the platform.

3.1 Overview of Data Sources

In the Fugaku environment, there are two types of data sources available. One data group is derived from the HPC system, including compute nodes, boot I/O nodes, peripheral nodes (e.g., login nodes, pre/post nodes, and so on), storage (e.g., a Lustre-based parallel file system), and network devices (e.g., switches, routers, and firewalls). The other group is derived from the data center infrastructure, including the power supply and cooling equipment. The details of these data sources are explained below.

HPC System. Firstly, we begin with a brief overview of the Fugaku HPC system and peripheral IT equipment. Fugaku employs Arm-based A64FX processors [20] enhanced to fit into the current trends of various HPC applications. Each microprocessor has four core memory groups (CMGs), each of which integrates 13 cores, an L2-cache shared by those cores, a memory controller, and a ring bus network on a chip. Of the 13 cores, one works as a redundant (or assistant) core that is used for multi-purpose operating processes. A core memory group (CMG) directly connects a single second-generation High Bandwidth Memory (HBM2) unit. Each CMG can access external HBM2 units belonging to the other CMGs of the processor via the ring bus network. The A64FX processor has a hardware performance monitoring unit (PMU) to measure dynamic program behavior (e.g., number of CPU cycles, number of instructions, floating-point operations, and cache misses) [14]. These metrics are useful not only for performance profiling tools but also for our operations to survey workloads' behaviors.

Fugaku consists of 432 racks with 158,976 compute nodes. By connecting the compute nodes with the high-speed interconnect (Tofu interconnect D), Fugaku can be organized as a single huge system to provide exascale computing resources. In addition, there are various peripheral servers, including login nodes with x86/Arm-based microprocessors, directory service nodes with Lightweight Directory Access Protocol (LDAP), job scheduler nodes, pre/post nodes with a graphics processing unit (GPU)/large-capacity memory, various management servers, and network devices.

In addition, Fugaku uses a Lustre-based parallel file system as with the K computer. During the K computer operations, the global file storage system shared a portion of the compute racks installed in the computer room, which degraded system maintainability. In contrast, the Fugaku storage system has been installed in the data storage room and can be operated separately from the compute system. The storage capacity is up to 150 PB available in total. At the time this paper was written, we divided the storage into six file volumes. In terms of monitoring, the storage system consists of many servers, including management service (MGS), metadata service (MDS), object storage service (OSS) servers, and so on. These are monitoring targets in our operation.

To monitor the system's behavior and keep operators informed, the HPC system produces a huge number of log data and metrics. Most of the log data (e.g., syslog messages, job scheduler logs, parallel file system logs, and audit logs) are recorded by servers. Additionally, some of the network devices generate syslog messages. Generally, these are time-series event data in a text-based semi-/unstructured format. The servers also measure various metrics (e.g., load average, network transfer bytes, memory usage, microprocessors and board temperature, Self-Monitoring, Analysis and Reporting Technology (S.M.A.R.T.), and electric power consumption on power supply units). The PMU metrics are also classified in this metric type. Further, network devices generate Simple Network Management Protocol (SNMP) messages. Generally, these metrics are time-series data provided in a scalar/vector numerical structured format.

Data Center Infrastructure. Our supercomputing center has dedicated facilities for power supply and cooling systems. In the Fugaku development project, while the IT equipment was mostly built from scratch, the concept of power supply and cooling systems resembles the previous design of the K computer and reuse most of the installed equipment without additional major enhancements. In the section below, we introduce the facility overviews.

As well as the K computer, Fugaku uses two-type energy sources—electricity purchased from a public utility company and energy produced by the gas turbine power generators. The new HPC system is designed to consume electricity up to 37 MW, which is approximately three times more than the K computer. Prior to the Fugaku installation, part of the facility components (e.g., transformers, a heat exchanger, chillers and cooling towers, and a thermal storage tank) had been enhanced to address the increased power consumption. Accordingly, we used a retrofit approach and reused many facilities installed for the K computer. For instance, we continue to use the co-generation system (CGS) without additional enhancements in the Fugaku operation.

Like the HPC system, the data center infrastructure produces various types of metrics (e.g., air/water temperature, pump flow rate, generated electric power, and purchased electric power from a public utility company) and miscellaneous log data. Basically, the log data volume is smaller than the metrics. All the metrics in the facility are collected by the building management system (BMS), which works as a subsystem of the operational data collection and monitoring

system. When we enhanced the facility, we also added sensors to the facility equipment at the same time. At the time of this writing, the number of sensors had reached five thousand.

3.2 System Architecture

The operational data collection/monitoring platform consists of a three-tier data pipeline, as shown in Fig. 1. The first tier is data sources, including the HPC system and the data center infrastructure, as discussed in Subject. 3.1. The second tier is a data lake used to store the log/metric data in the databases (e.g., time-series databases, distributed full-text databases, traditional transactional databases, and a Portable Operating System Interface (POSIX) file system). The third tier, which is the end of the pipeline, provides visualization/business intelligence (BI) tools that can be used to exploit insights. The platform began operations during the Fugaku prelaunch service period. In the remainder of this section, the current state of the system architecture is explained.

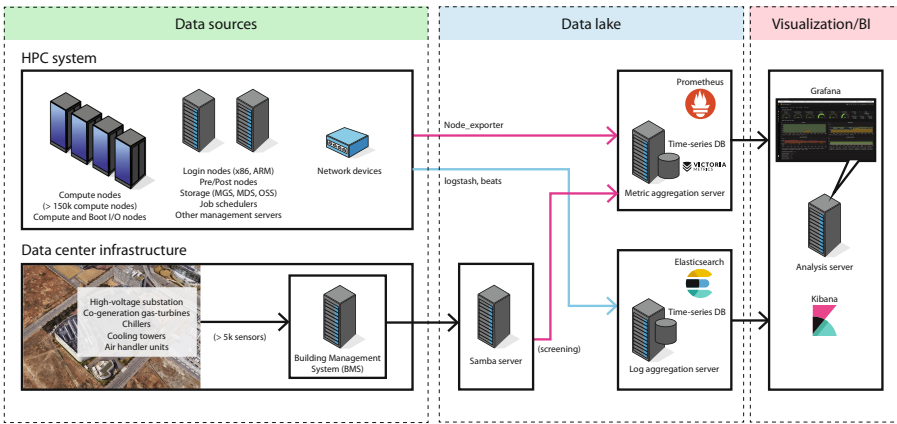


Fig. 1. Overview of the collection and monitoring platform.

Data Collection System. In the data lake tier, the data collection system uses two-type databases depending on the data type. For the metric data, we use a time-series database (TSDB) provided by Prometheus [6] as an open-source monitoring/alerting tool. For the log data, we use Elasticsearch [1] with logstash [4] as a data aggregation tool. To form the pipeline between the data sources and the data lake tier, we installed Prometheus exporter to serve as a data collection tool for each compute node and the boot I/O nodes. This tool is a shipper agent that captures metrics and sends them to the Prometheus server. Additionally, the exporter has been installed in other servers, including the login

nodes, pre/post nodes, and the MGS/MDS/OSS servers. On top of those, we installed `filebeat` in boot I/O nodes and the abovementioned servers. That is also a shipper agent in front of `logstash`.

During K computer operations, we could not deploy any data collection tools in the compute nodes because the microprocessor did not have any redundant cores that could be used to assign collectors. In the context of the HPC, performance is the primary concern. Therefore, we needed the ability to prevent a kind of performance interference (e.g., operating system jitter) that can be caused by the collectors running as a background daemon. In Fugaku, this issue was resolved by using the redundant cores.

Monitoring System. To monitor and visualize the time-series data stored in the databases, we use `Grafana` [2] and `Kibana` [3] as front-end tools for `Prometheus` and `Elasticsearch`, respectively. Both tools have fancy dashboards that allow users to customize their interfaces to meet specific requirements depending on the operators.

`Grafana` provides various graphical interfaces (e.g., line graphs, stacked graphs, heatmaps, tables, and so on). Also, `Kibana` has similar features that can be used to visualize log data. Both are managed via graphical user interfaces.

`Grafana` and `Kibana` focus on providing effective monitoring and reporting environments for troubleshooting. While those backend databases have derived query languages for use when searching data, the query languages impose some initial learning costs. In the K computer operations, we used `redash` [7] in addition to `Grafana` and `Kibana` to visualize metrics and data. `Kibana` is used to search data with the standard SQL language, which seems to be easier to learn than domain-specific languages. However, at the time of this writing, we have only deployed `Grafana` and `Kibana` for visualization, and our `Kibana` environment is not yet mature enough for use in practical operations because it still requires additional dashboard customization. Also, we believe that `Jupyter` [5] may be a more appropriate tool for more general users. Those general analysis tools (e.g. `redash`, `Jupyter`, and so on) in the third tier are outside the scope of this paper.

4 Results

In this section, we report on the implementation of monitoring dashboards with `Grafana` and introduce two use cases.

4.1 Case 1: The Integrated Monitoring Platform for the HPC System and the Data Center Infrastructure

Modern supercomputer systems require a complicated power supply and cooling facility management as with common plant operations. Hence, facility operators are required to monitor the power consumption of the entire center area 24 h a day, seven days a week, and efficiently manage CGS operations to ensure

backup electricity available for the IT equipment and to supply steam for the chillers as needed. From the facility operators' perspective, the amount of electricity supplied to the IT equipment is strongly dependent on workload behavior (e.g., memory throughput, flops, and communication frequency among compute nodes). While the job scheduler may play a key role in controlling workloads running simultaneously depending on the available electricity, it does not have a feature that can create supply-and-demand balances between the available electricity and the workloads. Accordingly, at the time of this writing, facility operators used the BMS that was installed at almost the same time the facility began operations to manage the power supply and cooling systems. That BMS was implemented on the intranet with access restrictions for security reasons. However, IT equipment operators also need online information regarding facility operations.

Based on the background, to integrate both the IT equipment information and the facility information, we implemented a combined dashboard. In Fig. 2, we show a portion of the facility metrics extracted from the dashboard, which were measured during the prelaunch service period that finished in early March 2021. The missing values in the middle of November were the result of planned maintenance. The dashboard can show the power consumption for the entire center as well as for Fugaku. Except for early October, the average power consumption was about 17.8 MW because the period of early October had a benchmarking campaign (e.g., HPL). Besides, the dashboard can visualize detailed power consumption (not shown in the figure) for each group of racks installed in the computer room. It can also show electricity supplied from the two CGSs and the public utility company. Although, the amount of purchased electricity was dominant in the total. Also, we confirmed that the switching operations with two CGSs within a certain period of time. That operations seem to extend the lifetime of the equipment and make it easier to maintain. Additionally, power usage effectiveness (PUE) is commonly used as a comparable metric to evaluate energy efficiency operations in data centers. Based on the results shown in the figure, the average PUE is about 1.2. Finally, we confirmed that all the results are consistent with the related study [26].

4.2 Case 2: Monitoring Overloading Jobs on the Lustre-Based Parallel File System

As with the K computer, Fugaku has a Lustre-based parallel distributed file system allocated to the home and working directories used by workloads. Although the Lustre-based file system is designed for use in large-scale parallel computing systems, it has encountered various issues when dealing with specific I/O patterns. For example, when a number of workloads write to a single file, those flooding I/Os often cause blocking responses or do not complete the processing due to the overloading because those transactions are serialized per file. Ideally, issues of this type would be resolved by modifying the user program itself. However, it is currently difficult to prevent the situation, and they are generally considered to be accidental issues. From the operators' perspective, it is



Fig. 2. A screenshot of electricity supply-and-demand and PUE during the prelaunch service period. The topmost graph (upper red box) shows power consumption for R-CCS as a whole and for the HPC system (only Fugaku). The middle graph (middle red box) shows the power generated by two CGSs and the purchased power from a public utility company. The bottommost graph (lower red box) shows the PUE and the exponential moving average of the PUE.

clear that the ability to detect the workload behavior and thus prevent accidental situations where file system overloading can result in system-wide failures is required. As shown in Figs. 3 and 4, we use the dashboard to monitor the Lustre-based file system state. Figure 3 describes OST write I/O performance using `dd` with parameters: `bs=1M`, `count=768`, `oflag=direct`. `dd` is issued at between 150 and 300-second intervals, avoiding simultaneous execution to the OSTs to be monitored.

The dashboard can output in CSV format. Tables 1 and 2 show sorted lists converted from a CSV file regarding the write I/O throughput for the OSTs of a volume (`vol100004`) shown in the Fig. 3. For instance, for a slow and fluctuated OST, it's not necessarily a malfunction, but it is likely to have some sort of problem. In this case, OST16 is appeared in both tables. While we showed that the case focuses on large fluctuations in Table 2, it is assumed that a case with small fluctuations can also be used for identifying issues. In addition, Fig. 4 describes MGS/MDS/OSS nodes performance. These dashboards are useful for finding and isolating the problem in operations.

5 Summary and Future Work

As we move toward bringing Fugaku into official service, we can report that most of the HPC system and a portion of the data center infrastructure have been upgraded to handle an unprecedented power consumption level of 30 MW.

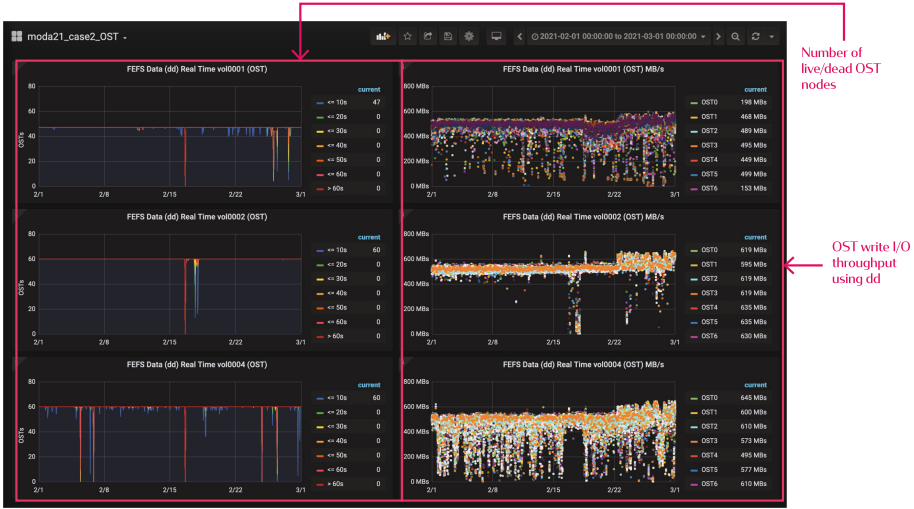


Fig. 3. An example of the Lustre-based file system state (OST write I/O performance). The left graphs describe the number of live/dead OST nodes using dd. If many OSTs have a long response time (60s is used as a criterion.), the possibility of failure is considered. The right graphs describe the OST write I/O throughput using dd.

Table 1. List of the ten slowest OSTs among the 60 OSTs of vo10004 shown in Fig. 3. The list is arranged in ascending order by average (mean).

OST name	mean [MB/s]	min [MB/s]	max [MB/s]	std [MB/s]
OST16	437.2	2.0	609.5	115.2
OST22	438.8	2.1	604.7	109.0
OST58	440.5	2.3	619.4	111.1
OST55	441.4	1.6	634.7	110.9
OST25	442.0	1.3	619.4	112.2
OST19	442.5	2.1	614.4	108.0
OST46	443.4	2.2	614.4	111.9
OST52	443.6	2.3	640.0	111.2
OST49	444.8	2.2	614.4	108.7
OST7	444.9	2.0	634.7	111.3

Table 2. List of the most fluctuated ten OSTs among the 60 OSTs of vo10004 shown in Fig. 3. The list is arranged in descending order by the standard deviation (std).

OST name	mean [MB/s]	min [MB/s]	max [MB/s]	std [MB/s]
OST47	459.2	2.2	629.5	117.0
OST16	437.2	2.0	609.5	115.2
OST53	462.1	1.4	640.0	114.2
OST32	460.2	2.1	629.5	114.1
OST0	469.7	3.5	645.4	114.0
OST17	459.2	2.0	619.4	113.6
OST20	457.3	2.1	629.5	113.1
OST23	456.5	2.1	614.4	112.8
OST26	455.6	2.1	609.5	112.7
OST2	466.3	3.5	629.5	112.7

Among the other necessary enhancements, we installed a new operational data collection and monitoring platform to operate in the Fugaku environment. In this paper, we reported the current state of this project to renew the platform.

The data collection and monitoring platform that was developed and deployed in this project is based on the three-tier data pipeline architecture. The first stage of the pipeline is data sources, including the HPC system and the data center

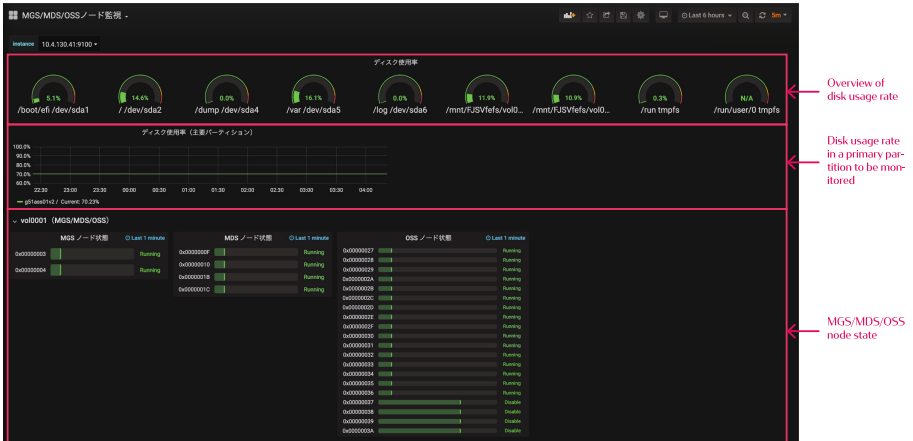


Fig. 4. An example of the Lustre-based file system state (MGS/MDS/OSS nodes performance). The topmost area (upper red box) describes an overview of disk usage rate in an MDS node of the Lustre-based filesystem. The middle graph (middle red box) describes disk usage rate in a primary partition to be monitored. The bottommost area (lower red box) describes updates on the part of the state of an MGS/MDS/OSS node. (Color figure online)

infrastructure. The second stage is a data lake used to store the log/metric data within time-series databases. The third stage, which is the end of the pipeline, provides visualization/BI features to exploit insights. To avoid vendor lock-in, various open-source software components have been adopted for use in this platform. At the time of this writing, the dashboard with **Grafana** was used to provides an interactive interface for multiple data provided by the HPC system and data center infrastructure. We also introduced two use cases describing the prelaunch service period. The first case showed comprehensive energy usage and the PUE metric of the center. The second case showed how the state for the Lustre-based filesystem was monitored to avoid overloading. In our future work, we will continue our efforts to resolve the ODA issue found in the K computer.

Acknowledgements. We would like to thank colleagues, especially Dr. Toshiyuki Tsukamoto (RIKEN R-CCS), who provided excellent advice and abundant materials regarding facility operations, and Dr. Yuichi Tsujita (RIKEN R-CCS) for valuable comments based on his expertise. We thank Mr. Kensuke Matsumoto (Fujitsu Limited), Mr. Naoki Ikeda (Fujitsu Limited), Mr. Yoshitaka Furutani (Fujitsu Limited), and Mr. Nobuo Ohgushi (Fujitsu Limited) for technical support and comments during the course of this project.

References

1. Elastic. github.com/elastic
2. Grafana. github.com/grafana/grafana

3. Kibana. github.com/elastic/kibana
4. logstash. github.com/elastic/logstash
5. Project Jupyter. github.com/jupyter
6. Prometheus. github.com/prometheus
7. redash. github.com/getredash/redash
8. Top500. www.top500.org/system/179807/
9. Wikipedia. [en.wikipedia.org/wiki/Fugaku\(supercomputer\)](https://en.wikipedia.org/wiki/Fugaku(supercomputer))
10. Bates, N., Hsu, C., Imam, N., Wilde, T., Sartor, D.: Re-examining HPC energy efficiency dashboard elements. In: 2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 1106–1109 (2016)
11. Bautista, E., Romanus, M., Davis, T., Whitney, C., Kubaska, T.: Collecting, monitoring, and analyzing facility and systems data at the national energy research scientific computing center. In: Proceedings of the 48th International Conference on Parallel Processing: Workshops. ICPP 2019, Association for Computing Machinery (2019)
12. Bourassa, N., et al.: Operational data analytics: optimizing the national energy research scientific computing center cooling systems. In: Proceedings of the 48th International Conference on Parallel Processing: Workshops. ICPP 2019, Association for Computing Machinery (2019)
13. Chen, J., Tan, R., Xing, G., Wang, X.: Ptec: a system for predictive thermal and energy control in data centers. In: 2014 IEEE Real-Time Systems Symposium, pp. 218–227 (2014)
14. Fujitsu: A64fx microarchitecture manual. github.com/fujitsu/A64FX
15. Matsuda, M., Matsuba, H., Nonaka, J., Yamamoto, K., Shibata, H., Tsukamoto, T.: Modeling the existing cooling system to learn its behavior for post-k supercomputer at riken r-ccs. In: Proceedings of the 48th International Conference on Parallel Processing: Workshops. ICPP 2019, Association for Computing Machinery (2019)
16. Minet, P., Renault, E., Khoufi, I., Boumerdassi, S.: Analyzing traces from a Google Data Center. In: 2018 14th International Wireless Communications Mobile Computing Conference (IWCMC), pp. 1167–1172 (2018). <https://doi.org/10.1109/IWCMC.2018.8450304>
17. Netti, A., et al.: Dcdb wintermute: enabling online and holistic operational data analytics on HPC systems. In: Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing (2020)
18. Nonaka, J., Hanawa, T., Shoji, F.: Analysis of cooling water temperature impact on computing performance and energy consumption. In: 2020 IEEE International Conference on Cluster Computing (CLUSTER), pp. 169–175 (2020)
19. Nonaka, J., Yamamoto, K., Kuroda, A., Tsukamoto, T., Koiso, K., Sakamoto, N.: A view from the facility operations side on the water/air cooling system of the k computer (2019). sc19.supercomputing.org/proceedings/tech/poster/tech/poster/pages/rpost246.html
20. Okazaki, R., et al.: Supercomputer fugaku CPU A64FX realizing high performance, high-density packaging, and low power consumption. Tech. Rep. Fujitsu Tech. Rev. (2020)
21. Ott, M., et al.: Global experiences with HPC operational data measurement, collection and analysis. In: 2020 IEEE International Conference on Cluster Computing (CLUSTER), pp. 499–508 (2020)
22. Santos, D., Mataloto, B., Ferreira, J.C.: Data center environment monitoring system. In: CCIOT 2019: Proceedings of the 2019 4th International Conference on Cloud Computing and Internet of Things, pp. 75–81. CCIOT 2019, Association for Computing Machinery (2019)

23. Sartor, D., Mahdavi, R., Radhakrishnan, B.D., Bates, N., Bailey, A.M., Wescott, R.: General recommendations for high performance computing data center energy management dashboard display. In: 2013 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum, pp. 892–898 (2013)
24. (SRCC), S.R.C.C.: HPC dashboards. github.com/stanford-rc/hpc-dashboards
25. Terai, M., Shoji, F., Tsukamoto, T., Yamochi, Y.: A study of operational impact on power usage effectiveness using facility metrics and server operation logs in the K computer. In: 2020 IEEE International Conference on Cluster Computing (CLUSTER), pp. 509–513 (2020)
26. Terai, M., Tsukamoto, T., Shoji, F.: Study on the facility enhancement by operational data analysis: a comparison of the operations in the K computer and fugaku. In: ISC 2021 Digital Research Poster (2021)
27. Yamamoto, K.: Operational data processing pipeline. In: SC19 BoF: Operational Data Analytics (2019). eehpcwg.llnl.gov/assets/sc19/bof/operational/data/processing/pipeline.pdf